IMAGE STEGANOGRAPHY

Emmanuel Maneswa

154409

Submitted to the
Graduation Projects Examination Jury
in partial fulfillment of the requirements for the degree of

Bachelor of Science

in

Software Engineering

European University of Lefke
June 2020

Approval of the Software Engineering Department

Assist. Prof. Dr. Ferhun Yorgancıoğlu
Head of Department

This is to certify that we have read this graduation project and that in our opinion it is fully adequate, in cope and quality, as an Undergraduate Project.

Assist. Prof. Dr. Cem B. Kalyoncu
Supervisor

Members of the examining committee

Name	Signature
1. Assoc. Prof. Dr. Hüseyin Ademgil	……..……
2. Assist. Prof. Dr. Cem B. Kalyoncu	……..……
3. Assist. Prof. Dr. Ferhun Yorgancıoğlu	……..……
4. Assist. Prof. Dr. Vesile Evrim	……..……
5. Dr. Ersin Çağlar	……..……

Date: ……..……

Abstract

IMAGE STEGANOGRAPHY

by

Emmanuel Maneswa

Software Engineering Department
European University of Lefke

Supervisor: Assist. Prof. Dr. Cem B. Kalyoncu

The project deals with the best approach in image steganography using Least Significant Bit (LSB) that further improves the quality of the existing LSB substitution methods to improve the security strength of the hidden information.

It is a new technique to substitute LSB of BGR true color image. The new security formation hides the secret information within the LSB of the image where the secret key encrypts the hidden information to protect it form unauthorized people. Generally, in LSB techniques hidden information is embedded into a fixed position of LSB of the image. For this purpose, knowing retrieval techniques, anyone can extract the hidden information. In my project, hidden information is embedded into different position of LSB of the image depending on the secret key.

As a result, it is difficult to extract the hidden information knowing the retrieval techniques. The implemented technique results in LSB based image steganography using secret key which gives good security issue than LSB based image steganography techniques.

Keywords: Steganography, Cryptography, Data Hiding, LSB, Digital Image, Hash, Security.

Acknowledgments

I would like to express my gratitude to my project supervisor, Assist. Prof. Dr. Cem B. Kalyoncu, who guided me throughout this project. I would also like to thank my advisor, Assist. Prof. Dr. Ferhun Yorgancıoğlu, for the continuous support of my BSc study and his motivation. I would also like to thank my family and friends who supported me.

Table of Contents

List of Figures

List of Tables

1. INTRODUCTION

1.1. Definitions

Usually nowadays if we want to transfer sensitive data, we encrypt the data before transferring it across the internet. Transferring messages like this, still can cause suspicion: there is clearly secret/sensitive data in your encrypted message that you are trying to hide. Attackers know precisely where to look to try to procure the information [1]. But privacy and anonymity is a concern for most people on the internet. Image Steganography allows for two parties to commune secretly and covertly. Steganography is the art of concealing information. In computer science, it refers to concealing data within a file or message. It serves a related motive to cryptography, but rather than encrypting data, steganography simply hides it from the user [2]. The word steganography comes from New Latin steganographia, which merges the Greek words steganos, meaning “concealed or covered”, and -graphia meaning “writing” [3].

In general, a steganographic system consists of a cover media into which the secret information is embedded. The embedding procedure fabricates a stego medium by replacing the information with data from the information to be hidden. Commonly secret information is embedded into the fixed position of Least Significant Bit (LSB) of a cover image which is the carrier to embed secret information. Since anyone can ensure that particular position of LSB contains secret information, so it is then easier for anyone to extract the secret information using the extraction method. The core purpose of image steganography is to guarantee security of the hidden information [4].

There are several researches available detailing properties of image steganography. Numerous steganographic techniques have been suggested. The most familiar of these is substituting the LSB of the pixels with secret information. A notable LSB established image steganography technique is given in [5] and it suggest an adaptive technique based on inter pixel relationship. The technique tremendously enhances the stego-image quality. Using this method it is feasible to extract the secret information for anyone by using the extraction method [6]. A different LSB established image steganography technique is given in [7] and it suggests three effective steganographic techniques that take advantage of the neighborhood

information to approximate the quantity of data to be embedded into the input pixel of the cover image and that embed exactly three bits of secret information in smooth sections and a varying amount of bits are embedded into the edged sections.This technique uses various pixels of the image store too many bits of the secret information but separate pixels stay unchanged. In result, some pixels are manipulated harshly whereas other pixels stay unused. Using this technique it is feasible to extract the hidden information for anyone, since they did not issue any security measures [6].

In order to ensure the security of the secret information, I have tackled it by using creative LSB based image steganography. In this technique I introduced a secret key which guarantee the security of secret information. The embedding of the secret information is determined by the secret key. The secret key determines the suitable location of secret information. It is extremely difficult to extract the secret information without the identical secret key. Here a bit of secret information is put in either LSB of Blue or Green or Red matrix of a particular pixel which is guaranteed by the secret key and the pixel column. I also use the secret key to encrypt the secret information to provide an extra layer of security. So by adding secret key, I increased the security intensity of the secret information in LSB based image steganography.

1.2. Goals

Project Goals (PG):

PG1: Hide secret information inside an image.
PG2: Invisibility/ Undetectability from humans.
PG3: Support multiple image formats.
PG4: Extract of all hidden information
PG5: Large embedding capacity.
PG6: Secure.
PG7: Support colour images.

Table 1.1. Project Goal 1

Goal Identifier	PG1
Project Goal	Hide secret information inside an image.
Goal Description	The primary goal of this project is to hide the secret information inside the cover image to produce a stego image containing the secret information.

Table 1.2. Project Goal 2

Goal Identifier	PG2
Project Goal	Invisibility/Undetectability from humans.
Goal Description	This goal states that the produced stego-image should be manipulated in away that the human eye will not spot the differences.

Table 1.3. Project Goal 3

Goal Identifier	PG3
Project Goal	Support multiple image formats.
Goal Description	This goal states that the system should support cover images of multiple image formats e.g. jpeg, png, tiff, webp, bmp, exr, hdr.

Table 1.4. Project Goal 4

Goal Identifier	PG4
Project Goal	Extract of all hidden information.
Goal Description	This goal states that the system should be able to extract the hidden information in its original state, meaning nothing should be missing.

Table 1.5. Project Goal 5

Goal Identifier	PG5
Project Goal	Large embedding capacity.
Goal Description	This goal states that the system should be able to hide a lot of information. Though this may depend on the size of image selected by the user. The bigger the image the increase in the amount of of information that can be hidden.

Table 1.6. Project Goal 6

Goal Identifier	PG6
Project Goal	Secure.
Goal Description	This goal states that the information hidden should be secure, meaning it should be encrypted before being hidden and then should be hidden according to a secret key provided by the user. And that secret key can only be used to know where the hidden information is when extracting it.

Table 1.7. Project Goal 7

Goal Identifier	PG7
Project Goal	Support colour images.
Goal Description	This goal states that the system should support colour images, since they have the RGB values for every pixel and it makes the system more secure (PG6) since there are multiple choices to choose from when hiding secret information.

1.3. Ethics

Association of Computing Machinery (ACM) and the IEEE Computer Society joined forced to create Software Engineering Code of Ethics and Professional Practices and cooperatively approved by the ACM and the IEEE-CS as the principle for practicing and teaching Software Engineering [8].

According to the ACM code of ethics principle 1.01. [8] and in accordance with the principle I take full responsibility in accepting this project as my own work. According to principle 1.03 [8] and in accordance with the principle I approve the software because I believe that it's safe, meets specifications, passes proper tests, and does not decrease quality of life, decrease privacy, or harm the environment. In general I believe the software is of public good [8].

According to the ACM code of ethics principle 3.02. [8] and in accordance with the principle I ensured proper and achievable objectives and goals for this project. According to

principle 3.10. [8] and in accordance to the principle I ensured adequate testing, debugging, and review of the software. According to principle 3.12. [8] and in accordance with the principle I worked to develop software that respect the privacy of those who will be affected by it. In general I believe I ensured software meets the highest professional standards possible [8].

According to the ACM code of ethics principle 8.01. [8] and in accordance with the principle I shall regularly try to further my knowledge of developments in the specification, analysis, design, development, maintenance, and testing of software and related documents, in conjunction with the management of the development process. According to principle 8.02. I shall regularly try to improve my ability to create reliable, safe, and useful quality software within reasonable time and at reasonable cost. According to principle 8.03. [8] I shall regularly try to improve my capacity to provide accurate, informative, and well-written documentation. In general, as a software engineer I shall take part in lifelong learning concerning the practice of my profession and shall encourage an ethical approach to the practice of the profession [8].

1.4. Required Software

1) Visual Studio Code

Visual Studio Code is a lightweight yet powerful free source code editor developed by Microsoft. It runs on desktop and is available for Linux, Windows and macOS. Its features include support for syntax highlighting, debugging, intelligent code completion, embedded Git, code refactoring, and snippets, and has a rich ecosystem of extensions for other features and programming languages [9]. This is the environment I used to write my code because it has a lot of advantages and it’s also a personal preference.

2) Git

Git is a free and open source distributed and version control system for tracking changes in source code throughout software development. It is developed for collaborating work between programmers, however it can be used track changes in any type of files [10]. I used it as as the version control system.

3) Bitbucket

Bitbucket is a web based version control repository repository hosting service owned by Atlassian. It is for hosting source code and development projects that use either Git or Mercurial version control systems [11]. I used it as the hosting service for the project.

4) OpenCV

OpenCV is a library of programming functions mostly focused on real time computer vision. The library is cross platform and free for use under the open source BSD licence [12]. I used it to load, alter, and save images and read image data.

5) Qt

Qt is a free and open source widget toolkit for developing Graphical User Interfaces (GUI) together with cross platform platform applications that run on numerous hardware and software platforms for instance Linux, Windows, macOS, Android or embedded systems with little to no change in the primary codebase and at the same time still being a native application with native capabilities and speed [13]. I used on of its modules Qt QML for developing the GUI.

6) Trello

Trello is a web based Kanban style list making application [14]. I used it to organize and prioritize project tasks in a scrum board style.

1.5. Programming Languages

1) C++

C++ is a general purpose programming language created by Bjarne Stroustrup as an extension of the C programming language, or "C with Classes". I used C++ as the application back-end, meaning the main algorithms for the desktop application are written in C++.

2) QML

QML is a user interface markup language. It is a declarative language for designing user interface centric applications. Inline JavaScript code handles imperative aspects. It is associated with Qt Quick. I used QML for creating the Graphical User Interface (GUI) for the desktop application.

3) JavaScript

JavaScript is a programming language that conforms to the ECMAScript specification. JavaScript is high-level, often just-in-time compiled, and multi-paradigm. I used JavaScript as to handle imperative aspects of the desktop application from QML, for instance calling C++ functions.

1.6 Third-Party Libraries

1) SHA-256

I used it to hash the secret key.

2) Easy Encryption

I used it to encrypt and decrypt the secret information.

2. LITERATURE REVIEW

2.1. History

The earliest steganographic technique dates back to the Greeks around 440 B.C. The Greek ruler Histiaeus used the first version of steganography which implied: shaving the head of his most trusted servant, tattooing the message onto the scalp, waiting for the hair to grow in order to conceal the secret message, and then sending the servant on his way to deliver the message with the instruction. The receiver would then have to shave the servant's head to disclose the secret message [15].

During the same time period, a different early kind of steganography was used. This technique involved Demeratus, who sent a warning message about an imminent invasion to Greece. This was done by writing the message directly on the wood of a wax tablet prior to applying the fresh layer of wax. This allegedly blank wax tablet was delivered successfully with its secret message [15].

Steganography kept going on over time to expand into different levels. In times of war it is used extensively. The American forces and the British used different styles of invisible ink throughout the American Revolutionary War. The invisible ink included fruit juice, vinegar, milk, and urine, for the hidden message. Heat or light was used to decode the secret message. The Germans began using microdots during World War II. The microdots were plans, documents, and images significantly decreased in size to the size of a period and then added to ordinary paperwork [15].

2.2. Publications

The easiest way for embedding secret information into an image is named Least Significant Bit (LSB) insertion. For a 24 bit true colour image, the number of modifications will be kept to a minimum and unnoticeable to the human eye. For instance, assume that we have thee pixels next to each other with the following RGB encoding:

Pixel 1:	10110010	11001010	00101110
Pixel 2:	10110110	11001110	00101100
Pixel 3:	10111011	11000111	00110101

Now assume we would want to embed the 9 bits of information **110100010**. If we hide these 9 bits in place the LSB of the above 9 bytes, we get the following pixels (bold bits have been modified):

Pixel 1:	1011001**1**	1100101**1**	00101110
Pixel 2:	1011011**1**	11001110	00101100
Pixel 3:	1011101**0**	11000111	0011010**0**

The below formula gives the most common description of the parts of this steganographic technique:

Secret Information + *Cover Image* = *Stego Image*

The above formula for this old technique is best described with the following figure, which contains useful additional details:

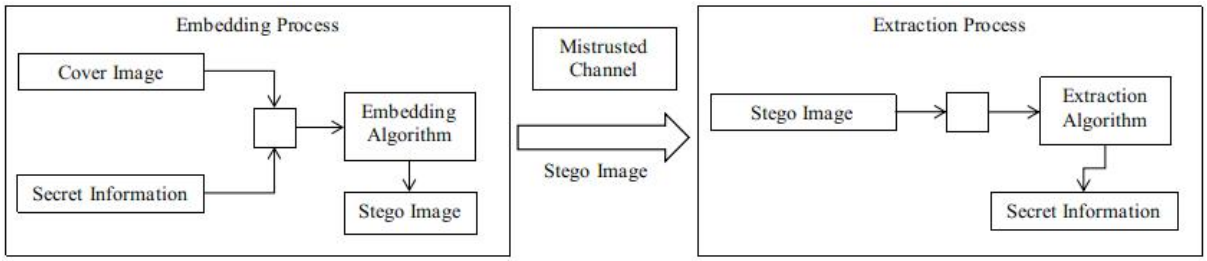

Figure 2.1. Steganographic Formulation (Publication)

In this technique the cover image is the image into which the secret information is embedded. The stego image is the output image resulting from the embedding, which will be the same type of image as the cover image. This technique has its weakness, the secret information is easy to retrieve for anyone knowing the retrieval method.

In the year 2015, G. Prashanti and K. Sandhyarani did a survey [16] on the modern accomplishments of Least Significant Bits (LSBs) based image steganography. In the survey

the writers study about the advancements that increase the steganographic outcomes for instance high embedding magnitude, high robustness, and undetectability of the secret information. In the course of this survey two new methods are suggested. In the first method a secret grayscale image is hidden inside another grayscale image and the second method is used to embed secret information inside the cover image. These methods use a four state table to generate pseudo random numbers that will be used for embedding the secret data. These two techniques have magnificent security because the secret data is embedded according to the random chosen positions of the LSBs of the cover image with the help of the pseudo random numbers produced by the table [16].

In the year 2015, B. Feng, W. Lu, and W. Sun in their article [17] proposed a state of the art proposition of binary image steganography. This method is suggested to decrease the distortion on the quality of the image. In this technique of steganography the complement, rotation and mirroring invariant texture designs are removed from the binary image. They also suggested a quantification and based on this suggested quantification this procedure is pragmatically implemented. The pragmatic outcomes show that the suggested steganographic procedure has high statistical reliability with high embedding capacity and high stego-image quality [17].

In the year 2015, M. Nosrati, A. Hanani, and R. Karimi did a report [18] on heuristic genetic algorithm based steganographic technique for embedding secret information in the cover image. This technique optimally finds the suitable positions to hide the secret information in the cover image by focusing on the previous embedding methods. The technique attempts to make a minimum number of changes on the bits which leads to least alterations in the image histogram. To transform the secret information and LSBs to a set of blocks, division into separate parts is performed in this genetic algorithm. After this algorithm locate the suitable positions for embedding, the confidential blocks are embedded and it creates the key file which is used when extracting the secret information. The preliminary outcomes show that this genetic based technique is more effective than the basic LSB techniques with high stego-image quality [18].

In the year 2014, K. Qazanfari and R. Safabakhsh suggested [19] an improved edition of LSB++ technique. In this LSB++ technique they formulate the differences between sensitive pixels and permit protecting them shielding them from hiding of additional bits,

which results in minor tampering in the co-occurrence matrices. They also expand this technique to protect DCT coefficients of JPEG images. This improved technique results in hardly any trails in the co-occurrence matrices than previous LSB++ method. This technique is also safe against histogram based attacks because it does not cause any alterations in the histogram and therefore histograms of both stego-image and and cover image will be the same. The stego-image quality will be high because of the complete removal of additional bit embedding [19].

In the year 2014, N. Akhtar, S. Khan, and P. Johri in their article [20] introduce and implement the upgraded edition of traditional LSB techniques of image steganography. Their work increases the quality of the stego-image by utilizing bit inversion technique. They suggest two techniques of bit inversion. Both methods settle around bit inversion methods in which LSBs of pixels of the cover image are inverted if and only if they result with a particular pattern of pixel's bits. This results in decreased alterations in pixels if compared to conventional LSB techniques. For the proper extraction of the secret information, inverted bits have to be hidden someplace within the stego-image. The preliminary outcomes indicate that the Peak Signal-to-Noise Ratio (PSNR) value of the stego-image is enhanced, therefore the quality of the stego-image is improved [20].

In the year 2014, S. Islam, M. R. Modi, and P. Gupta suggested [21] an interestingly new steganography method to hide confidential information in the LSBs of the cover image. In their technique the least 2 significant bits of the edges are used to hide the secret information because the edge sections are extremely good parts to hide the confidential data than the other smooth sections of the cover image. In this technique the edge sections are discovered on the foundation of the amount of the confidential data, which indicate it does adaptive edge spotting. The preliminary outcomes reviews show that the suggested technique functions better than customary LSB image steganographic techniques and impressively secure against visual attacks [21].

In the year 2012, S. Gupta, G. Gujral, and N. Aggarwal suggested [22] a strengthened LSB algorithm for image steganography. In this suggested technique they only hide the secret information into the blue element of the RGB color space. In this method the RxC size cover image is chosen. After choosing the cover image only the blue element is utilized for hiding the secret information. They also benefit from the use of pixel filters to gain access to the

finest positions to to embed the secret information into the cover image in-order to obtain the best possible rate. The preliminary outcomes show that this method decreases the manipulation of the cover image and the stego-image has extremely good visible quality and modifications in the cover image are negligent to the human eye. This technique decreases the jump in the color scale since only blue elements of RGB are utilized to hide the secret information [22].

In the year 2010, C. Lee and W. Tsai suggested [23] a new technique of image steganography which utilizes PNG images to embed secret information. The technique for secret sharing is used to create limited shares from the specified data string with the help of some polynomial coefficients as information transporter for calculating the shares. These partial shares are then hidden into alpha channel and produce the stego-image which contains white noise. To minimize the white noise small prime numbers can be utilized. The suggested technique has an effective secret information hiding capacity with increased security level and high stego-image quality [23].

3. BACKGROUND INFORMATION

3.1. Proposed Method

For my project, I introduced a secret key so that I can protect the secret information. I take the binary representation of the encrypted secret information and then modify some of the LSB of one byte per pixel inside the cover image. In this method, I used the following formula:

Encrypted Secret Information + Secret Key + Cover Image = Stego Image

The above formula is best described by the following figure, which contains useful additional details:

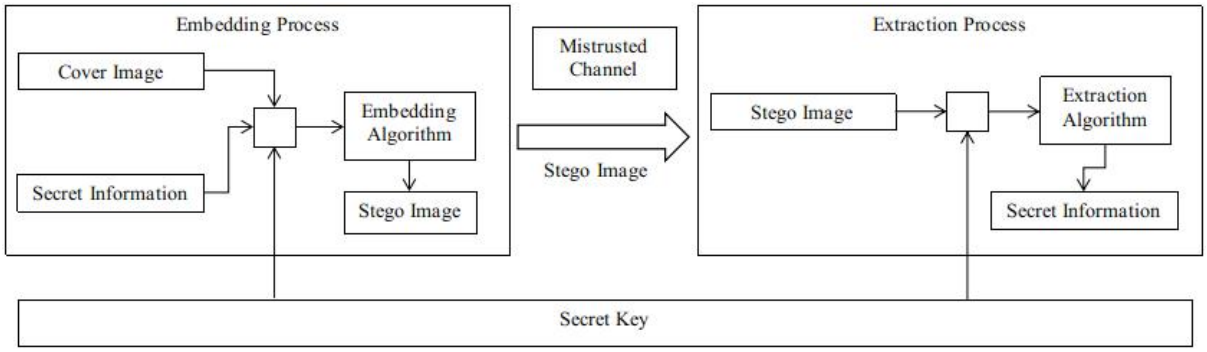

Figure 3.1. Steganographic Formulation (Proposed)

The secret key is then transformed into binary representation which will then be used as a circular string of bits.

A truecolor image uses 24 bits per pixel and every one byte represents the intensity of the primary colour blue, green or red (BGR), respectively. The number of the byte is stored as an 8 bit integer ranging from 0 to 255. Generally, the cover image is partitioned into three matrices as shown in the following figure 3.2:

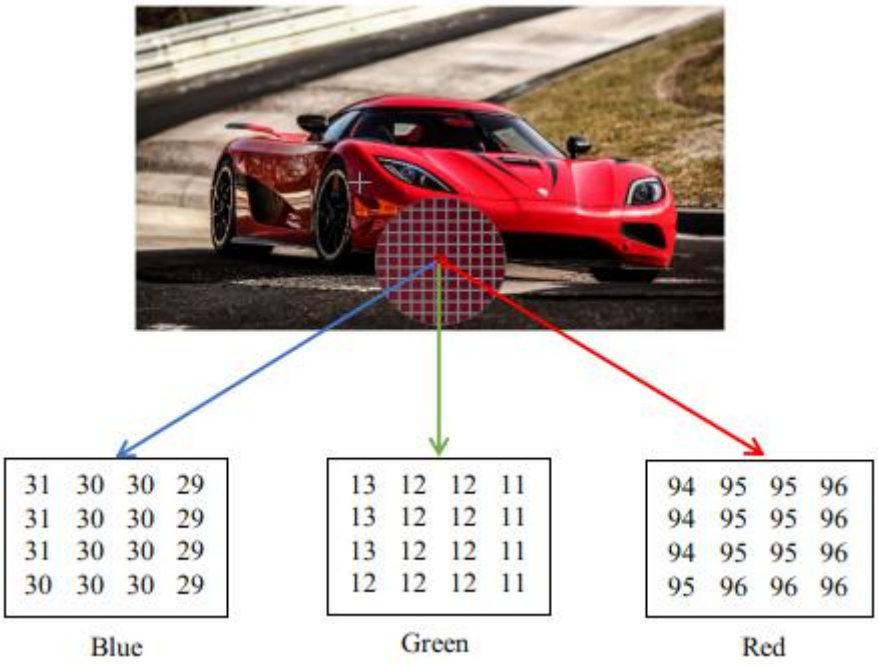

Figure 3.2. BGR Matrix Representation of the Image

3.2. Embedding Method

To embed the secret information, the cover image is taken. This cover image is then split into three matrices Blue, Green, and Red as shown in figure 3.2. The secret information is then encrypted using the secret key and then converted into binary representation string. This binary string of the secret information is called bit stream of secret information. The secret key is then converted into a binary representation string called bit stream of secret key, which will then be used as a circular bit stream.

For the pixel columns 0, 3, 6, 9, ... the Blue matrix and the Secret key are used for determining if the secret information is to be embedded into either the Green or the Red matrix. In this case, every one bit of the secret key is XOR with every one LSB of the Blue matrix. If the resulting value of the XOR operation is 0, the secret information bit is embedded into the LSB of Green matrix and if it's 1, the secret information bit is embedded into the LSB of Red matrix.

For the pixel columns 1, 4, 7, 10, ... the Green matrix and the Secret key are used for determining if the secret information is to be embedded into either the Blue or the Red matrix. In this case, every one bit of the secret key is XOR with every one LSB of the Green matrix.

If the resulting value of the XOR operation is 0, the secret information bit is embedded into the LSB of Red matrix and if it's 1, the secret information is embedded into the LSB of Blue matrix.

For the pixel columns 2, 5, 8, 11, ... the Red matrix and the Secret key are used for determining if the secret information is to be embedded into either the Blue or the Green matrix. In this case, every one bit of the secret key is XOR with every one LSB of the Red matrix. If the resulting value of the XOR operation is 0, the secret information bit is embedded into the LSB of Blue matrix and if it's 1, the secret information bit is embedded into the LSB of Green matrix.

The figure below shows how the secret information bit embedding operations are done:

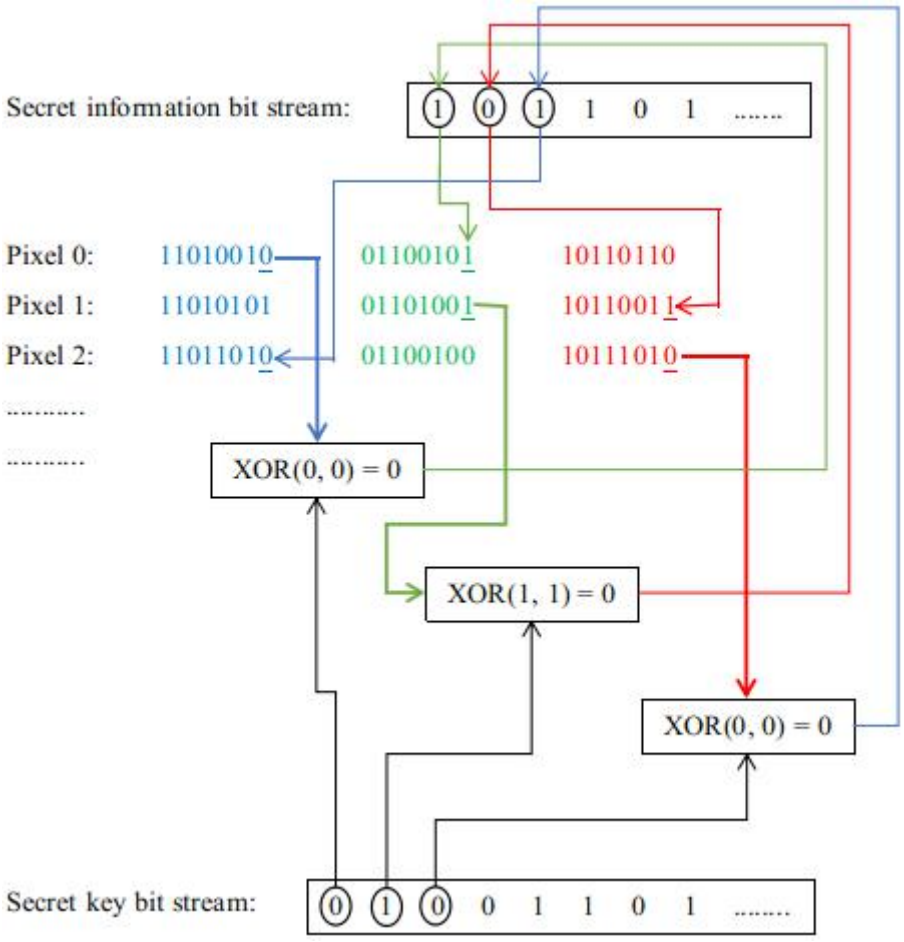

Figure 3.3. Bit Embedding Operations

The secret key bit stream is rotated in a circular manner to create a circular bit stream, the circulation continues until the embedding of the secret information bit stream is completed. The following figure is the flowchart for embedding the secret information into the cover image:

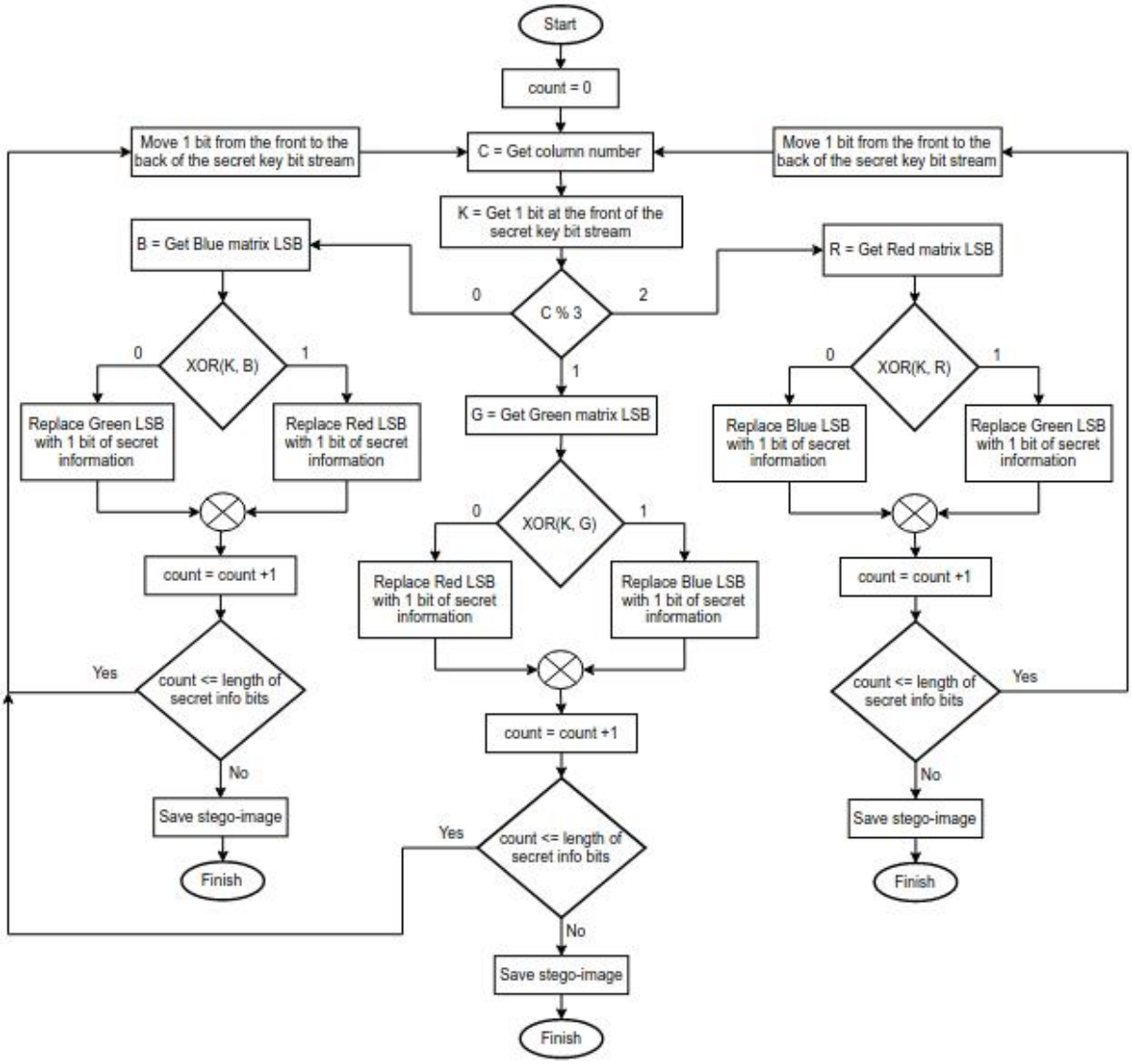

Figure 3.4. Secret Information Embedding Flowchart

The length of the secret information bit stream is embedded in the 1st rows of the cover image. The secret key is hashed and the hashed secret key bits are embedded between the 6th and the 10th rows of the cover image.

3.3. Extraction Method

To extract the secret hidden information, the stego-image is taken. This stego-image is then split into three matrices Blue, Green, and Red as shown in figure 3.2. The secret key is then needed in-order to extract the secret information. The secret key is then converted to a bit stream of secret key, which will then be used as a circular bit stream. During the extraction procedure the extracted secret information bits are stored in a string called bit stream of secret information.

For the pixel columns 0, 3, 6, 9, ... the Blue matrix and the Secret key are used for determining if the secret information is to be extracted from either the Green or the Red matrix. In this case, every one bit of the secret key is XOR with every one LSB of the Blue matrix. If the resulting value of the XOR operation is 0, the secret information bit is extracted from the LSB of Green matrix and if it's 1, the secret information bit is extracted from the LSB of Red matrix.

The figure below shows how the secret information bit extraction operations are done:

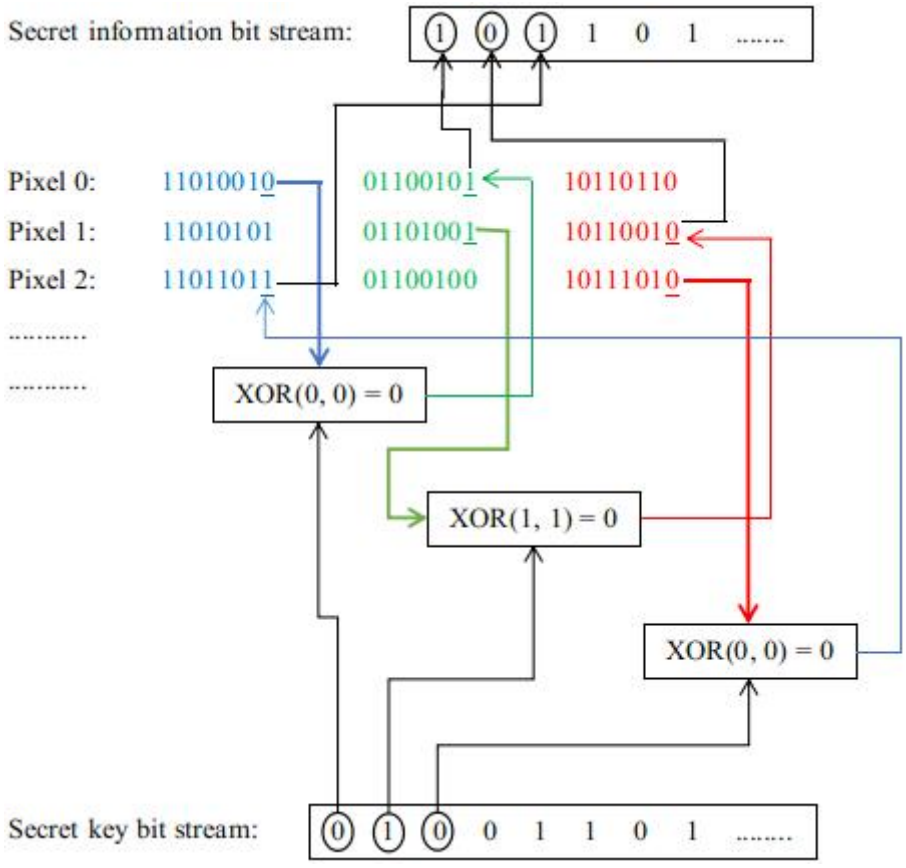

Figure 3.5. Bit Extraction Operations

For the pixel columns 1, 4, 7, 10, ... the Green matrix and the Secret key are used for determining if the secret information is to be extracted from either the Blue or the Red matrix. In this case, every one bit of the secret key is XOR with every one LSB of the Green matrix. If the resulting value of the XOR operation is 0, the secret information bit is extracted from the LSB of Red matrix and if it's 1, the secret information is extracted from the LSB of Blue matrix.

For the pixel columns 2, 5, 8, 11, ... the Red matrix and the Secret key are used for determining if the secret information is to be extracted from either the Blue or the Green matrix. In this case, every one bit of the secret key is XOR with every one LSB of the Red matrix. If the resulting value of the XOR operation is 0, the secret information bit is extracted from the LSB of Blue matrix and if it's 1, the secret information bit is extracted form the LSB of Green matrix.

The secret key bit stream is rotated in a circular manner to create a circular bit stream, the circulation continues until the extraction of the secret information bit stream is completed. The flowchart for extraction of the secret information from the stego-image is show in figure 3.6. Once the secret information bits extraction is complete, the secret information bit stream is then transformed from binary representation and then decrypted using the secret key to provide the exact secret information/message that was embedded.

Before extracting the secret information, the hidden secret information bits length is extracted from 1st five rows of the stego-image. The provided secret key is hashed and compared to the hidden extracted hashed key. If the hashes are not the same attempt to extract the secret information is terminated. If the hashes are the same then the extraction of the secret information is started. The embedded hashed secret key is extracted between the 6th and the 10th rows of the stego image.

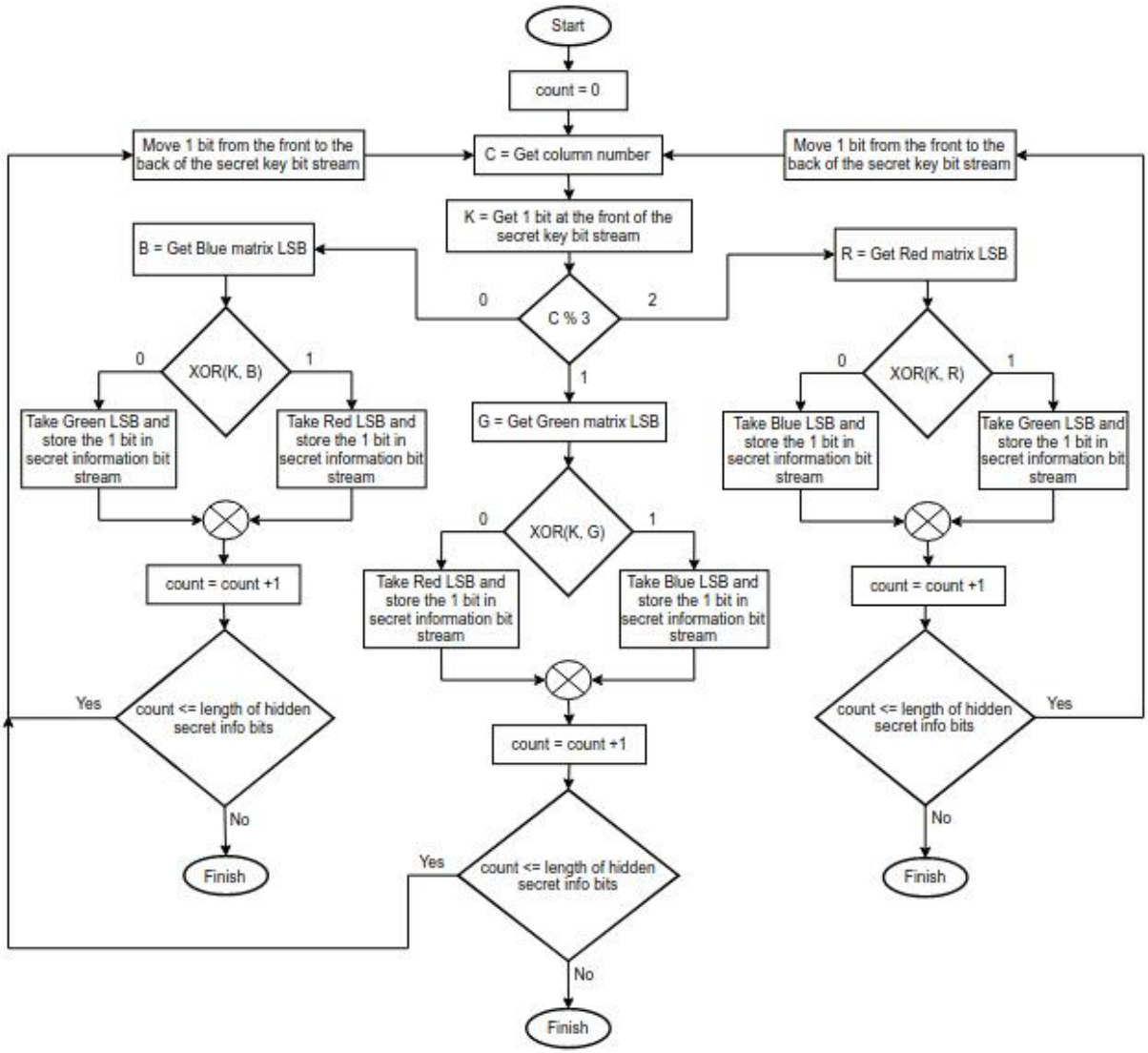

Figure 3.6. Secret Information Extraction Flowchart

4. IMPLEMEMTATION DETAILS

4.1. Implementation

The flowcharts provided in the previous chapter (figure 3.4. and figure 3.6.) give an insight of just how the embedding and the extraction methods of the method I proposed for my project. They give an Idea on how the decisions are made when choosing the primary color intensity of a pixel to embed or extract the secret information bit. In the following two figures I show how I implemented the Idea shown in the flowchart:

```
if(count < secret_info_.length()){
    cv::Vec3b pixel = img_.at<cv::Vec3b>(cv::Point(j, i));    // get pixel.
    switch(j % 3){
        /**
         * @brief Case 0: Use blue lsb for XOR. if 1 embed secret information in red lsb else embed in green lsb.
         *
         */
        case 0:
            if(XOR(key_[0], IsOdd((int)pixel[0])))
                img_.at<cv::Vec3b>(cv::Point(j, i)) = EmbedInRed(pixel, count);
            else
                img_.at<cv::Vec3b>(cv::Point(j, i)) = EmbedInGreen(pixel, count);
            break;

        /**
         * @brief Case 1: Use green lsb for XOR. if 1 embed secret information in blue lsb else embed in red lsb.
         *
         */
        case 1:
            if(XOR(key_[0], IsOdd((int)pixel[1])))
                img_.at<cv::Vec3b>(cv::Point(j, i)) = EmbedInBlue(pixel, count);
            else
                img_.at<cv::Vec3b>(cv::Point(j, i)) = EmbedInRed(pixel, count);
            break;

        /**
         * @brief Case 2: Use red lsb for XOR. if 1 embed secret information in green lsb else embed in blue lsb.
         *
         */
        case 2:
            if(XOR(key_[0], IsOdd((int)pixel[2])))
                img_.at<cv::Vec3b>(cv::Point(j, i)) = EmbedInGreen(pixel, count);
            else
                img_.at<cv::Vec3b>(cv::Point(j, i)) = EmbedInBlue(pixel, count);
            break;
    }
    CirculateKey();
}
else return;
```

Figure 4.1. Code Snippet for Embedding Decision Making

```
if(secret_info_.length() < secret_info_length){
    cv::Vec3b pixel = img_.at<cv::Vec3b>(cv::Point(j, i));
    switch(j % 3){
        /**      You, 2 months ago • Finished Extraction algorithm implementation
         * @brief Case 0: Use blue lsb for XOR. if 1 extract secret information from red lsb else extract from green lsb.
         *
         */
        case 0:
            if(XOR(key_[0], IsOdd((int)pixel[0]))) ExtractFromRed(pixel);
            else ExtractFromGreen(pixel);
            break;

        /**
         * @brief Case 1: Use green lsb for XOR. if 1 extract secret information from blue lsb else extract from red lsb.
         *
         */
        case 1:
            if(XOR(key_[0], IsOdd((int)pixel[1]))) ExtractFromBlue(pixel);
            else ExtractFromRed(pixel);
            break;

        /**
         * @brief Case 2: Use red lsb for XOR. if 1 extract secret information from green lsb else extract from blue lsb.
         *
         */
        case 2:
            if(XOR(key_[0], IsOdd((int)pixel[2]))) ExtractFromGreen(pixel);
            else ExtractFromBlue(pixel);
            break;
    }
    CirculateKey();
}
else return;
```

Figure 4.2. Code Snippet for Extraction Decision Making

In-order to determine the length of the hidden bits I store their length in bit notation between rows one to five of the cover image. So when doing the extraction process of the secret information I continue to extract the bits until the length of the extracted bits is equal to the length that will be extracted from the first five rows of the stego-image.

So to determine if the secret key is correct and avoid extracting garbage secret information, I store the secret key hide the secret in the cover image. This secret key is first hashed using SHA-256 hashing algorithm and then converted to binary notation and then embedded into the cover image. The hashed key bit stream is embedded between the sixth and the tenth rows of the cover image. So when doing the extraction process the provided key is hashed and and then the embedded hashed key bits are extracted and converted from binary notation. The two hashed keys are then compared and if they are equal then the extraction process proceeds and if they are not equal the extraction process is terminated.

To determine or change the LSB of a pixel primary color matric I firstly wanted to convert the unsigned char value to binary notation. But I then used a much simpler way which was to check if the value was odd or even. So if unsigned char value is odd I know that the LSB is 1 and if it's even I know the LSB is 0. When changing the LSB, if the unsigned

char value is odd I subtract 1 to change the LSB to 0 and if the unsigned char value is even I add 1 to change the LSB to 1. I do this because of the range of unsigned char which is 0 to 255 and so to avoid changing the values to values that are out of that range that is why I add when the value is even and I subtract when the value is odd. And I use this same technique when extracting the information to determine if the LSB of hidden information is 0 or 1.

For establishing a link between the C++ backend and the QML frontend, I use JavaScript to make function calls to C++ and handle the returned messages. In C++ I created a middlemen class that inherits from QObject which enables it to be called from the frontend. This middlemen class is used to handle calls from the frontend and and then pass the requests to the steganography class and get the response and the returns the requests to the frontend.

5. CONCLUSION AND FUTURE RECOMMENDATION

5.1. Benefits

The main benefit of this project is when you try to compare it to Cryptography. The Cryptography and Steganography are the two different aspect of the same situation. Cryptography encrypts the message make it complex whereas Steganography hides traces of the message. Hence Steganography is beneficial because it does not attract unwanted attention to the message and its communicating parties whereas for encryption, it does not matter how complex it is, it will raise suspicion and may be incriminating in the countries that don't allow encryption.

With the use of Steganography law enforcement agencies and corporation governments can have the ability to communicate secretly. For instance, if a government is planning to make major changes that will benefit their economy and citizens but a rival government could intercept these messages and know that there is something going on but if the government was to consider steganography in this case it would look like a normal day for the rival government. And it's even more beneficial for law enforcement, for instance a witness in a major crime is put in protective witness security and the officers themselves wanted to share the witness location with the headquarters or the government. Its safer to use steganography because they can't risk using encryption because if it's intercepted the unauthorized party might be able to decrypt it and know the witness location too.

In the proposed technique of this project I added a secret key into the basic concept of the LSB algorithm. This secret key is beneficial to make message hidden much more difficult to extract without the secret key. So even if an unauthorized user is informed that the message is hidden in the image, and they don't know the secret key the wouldn't be able to extract the information.

In the proposed method, since I use only the LSB of one color per pixel, The changes done to the image are kept to a minimum amount. This gives the stego-image a lower distortion.

5.2. Limitations

One of the major limitations to this technique is that since I use the LSB of one color per pixel to hide the secret information bit, that means that the less information is stored in the image compared to the basic LSB technique which hides in all colors of the pixel.

The major disadvantage of this technique is if it were to fall in the wrong hands, for instance the like of terrorists, criminals, and hackers, etc then this would be used in a dangerous way. It could be to infiltrate governments or organizations or even worse, because it can be used to transmit computer viruses or trojan horses.

The other disadvantage to my technique is that it only allows saving the image in png format because of it's loss less algorithm where as it does not support saving the stego-image in image formats that use lossy compression algorithms. The need to support saving the stego-image in JPEG format is crucial because it's one of the most used image file format. But to support such image file format requires delving into the JPEG algorithm.

5.3. Future Recommendations

The digital world is in a constant state of advancements. Steganography is a technology with great competitive applications. In image steganography there has been noticeable progress, but there is still room for improvement as the available algorithms can be further enhanced. The enhancements can be accomplished using techniques such as Genetic algorithms, Fuzzy logic, and artificial neural networks based steganography. The future researches should take into considerations the Quantum Computing techniques which could extend the well-established steganography techniques for performance enhancements...

6. REFERENCES

[1] Zdziarski, Z. (2018, Oct 5). Image Steganography - An Introduction, Zbigatron. Viewed May 27, 2020, from <https://zbigatron.com/image-steganography-an-introduction>

[2] Christensson, P. (2020, Feb 24). Steganography Definition. Viewed May 27, 2020, from <https://techterms.com/definition/steganography>

[3] Merriam-Webster. (n.d.). Steganography. Merriam-Webster.com dictionary. Viewed May 27, 2020, from <https://www.merriam-webster.com/dictionary/steganography>

[4] Gangwar, A & Shrivastava, V. (2013). Improved RGB-LSB Steganography Using Secret Key. International Journal of Computer Trends and Technology. [Online]. 41(2). pp. 85-89. Viewed May 28, 2020, from

<https://pdfs..semanticscholar.org/7a7b/54d6557fd5f2945f8720d098fbf77d165457.pdf>

[5] Long, Min & Li, Fenfang. (2018, July). A Formula Adaptive Pixel Pair Matching Steganography Algorithm. Advances in Multimedia. 2018. 1-8. 1155/2018/7682098. Viewed May 28, 2020, from

<https://www.researchgate.net/publication/326229819_A_Formula_Adaptive_Pixel_Pair_Matching_Steganography_Algorithm>

[6] Karim, S. M. Masud & Rahman, Md & Hossain, MD. (2011, Dec). A new approach based image steganography using secret key. 286-291. 10. 1109/ICCITechn. 2011.6164800. Viewed May 28, 2020, from

<https://www.researchgate.net/publication/261421805_A_new_approach_for_LSB_based_image_steganography_using_secret_key>

[7] Haque, Sadia & Sharmin, Farhana. (2010, June). Variable Rate Steganography in Gray Scale Digital Images Using Neighborhood Pixel Information. Int. Arab J. Inf. Technol.. 7. 34-38. 10. 1109/ICCIT. 2009. 5407128. Viewed May 28, 2020, from

<https://www.researchgate.net/publication/220413577_Variable_Rate_Steganography_in_Gray_Scale_Digital_Images_Using_Neighborhood_Pixel_Information>

[8] Gotterbarn, Donald & Miller, Keith & Rogerson, Simon & Barber, Steven & Barnes, P & Burnstein, I & Davis, Michael & El-Kadi, Amr & Fairweather, NB & Fulghum, M & Jayaram, N & Jewett, T & Kanko, M & Kallman, E & Langford, D & Little, Joyce &

Mechler, E & Norman, MJ & Phillips, D & Werth, LH. (2001). Software Engineering Code of Ethics and Professional Practice. Science and Engineering Ethics. 7. 231-238. 10.1007/s11948-001-0044-4. Viewed June 2, 2020, from

<https://www.researchgate.net/publication/278417404_Software_Engineering_Code_of_Ethics_and_Professional_Practice>

[9] Getting Started. Microsoft. (n.d.). Viewed June 3, 2020, from

<https://code.visualstudio.com/docs>

[10] Git --distributed-is-the-new-centralized. (n.d.). Git. Viewed June 3, 2020, from

<https://git-scm.com>

[11] Bitbucket. (2020, May 22). Wikipedia. Viewed June 3, 2020, from

<https://en.wikipedia.org/wiki/Bitbucket>

[12] OpenCV. (2020, May 22). Wikipedia. Viewed June 3, 2020, from

<https://en.wikipedia.org/wiki/OpenCV>

[13] Qt (software). (2020, May 27). Wikipedia. Viewed June 3, 2020, from

<https://en.wikipedia.org/wiki/Qt_(software)>

[14] Trello. (2020, April 20). Wikipedia. Viewed June 3, 2020, from

<https://en.wikipedia.org/wiki/Trello>

[15] Siper, A & Farley, R & Lombardo, C. (2005, May 6). The Rise of Steganography. Viewed June 5, 2020, from <http://csis.pace.edu/~ctappert/srd2005/d1.pdf>

[16] Prashanti, G & Sandhyarani. (2015). A New Approach for Data Hiding with LSB Steganography. In: Satapathy S., Govardhan A., Raju K., Mandal J. (eds) Emerging ICT for Bridging the Future - Proceedings of the 49th Annual Convention of the Computer Society of India CSI Volume 2. Advances in Intelligent Systems and Computing, vol 338. Springer, Cham

[17] Feng, B & Lu, W & Sun, W. "Secure Binary Image Steganography Based on Minimizing the Distortion on the Texture," in IEEE Transactions on Information Forensics and Security, vol. 10, no. 2, pp. 243-255, Feb. 2015, doi: 10.1109/TIFS.2014.2368364.

[18] Nosrati, M & Hanani, A & Karimi, R. "Steganography in Image Segments Using Genetic Algorithm," 2015 Fifth International Conference on Advanced Computing & Communication Technologies, Haryana, 2015, pp. 102-107, doi: 10.1109/ACCT.2015.57.

[19] Qazanfari, K & Safabakhsh, R. (2014, Sept). A new steganography method which preserves histogram: Generalization of LSB++. Information Sciences. 277. 90-101. 10.1016/j.ins.2014.02.007.

[20] Akhtar, N & Khan, S & Johri, P. (2014, Feb). An improved inverted LSB image steganography. 10.1109/ICICICT.2014.6781374.

[21] Islam, S & Modi, M, R & Gupta, P. (2014, April). Edge-based image steganography. EURASIP Journal on Information Security. 2014. 8. 10.1186/1687-417X-2014-8.

[22] Gupta, S & Gujral, G & Aggarwal, N. (2012, July). Enhanced Least Significant Bit for Image Steganography. IJCEM International Journal of Computational Engineering & Management, Vol 15, Issue 4. ISSN (Online): 2230-7893

[23] Lee, C & Tsai, W. "A new steganographic method based on information sharing via PNG images," 2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE), Singapore, 2010, pp. 807-811, doi: 10.1109/ICCAE.2010.5451874.