IMAGE STEANOGRAPHY

BECE301L – DIGITAL SIGNAL PROCESSING

BY

Asmita Ghosh (21BEC1097)
Ashrit Saha(21BEC1257)
Spandan Gupta (21BEC1263)

SUBMITTED TO

Dr. RAMESH R

Associate Professor, VIT Chennai

VIT®
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

SCHOOL OF ELECTRONICS ENGINEERING

VELLORE INSTITUE OF TECHNOLOGY

CHENNAI – 600127

JULY 2023

CERTIFICATE

This is to certify that the Project work titled "" is being submitted by**, Asmita Ghosh(21BEC1097), Ashrit Saha(21BEC1257), Spandan Gupta (21BEC1263)** for the course **BECE301L – DIGITAL SIGNAL PROCESSING**, is a record of bonafide work done under my guidance. The contents of this project work, in full or in parts, have neither been taken from any other source nor have been submitted to any other Institute or University.

Prof. RAMESH R

Associate Professor

School of Electronics Engineering (SENSE)

VIT University, Chennai

Chennai – 600127

ABSTRACT

With the increasing need for secure communication and data transmission, the field of steganography has gained significant attention. Image steganography, a subset of steganography, involves concealing secret information within digital images. This paper explores the concept of image steganography and its practical implementation using the Least Significant Bit (LSB) technique.

The paper begins with an introduction to steganography, highlighting its purpose and importance in secure communication. It then focuses on image steganography, explaining the LSB method, where secret data is embedded by altering the least significant bit of each pixel in an image. The advantages and disadvantages of LSB-based image steganography are discussed, including its simplicity, capacity, and vulnerability to attacks.

The scope of this project encompasses the development of a MATLAB code that performs image steganography using LSB, along with a detailed explanation of the implementation process. The code takes an input image and encodes a predefined message within it, visualizing the encoding through the generated encoded and check images. Additionally, a decoding process is presented to retrieve the hidden message from the encoded image.

This project aims to provide a comprehensive understanding of image steganography, its applications, and the implications of using LSB-based techniques. The practical implementation demonstrates the potential uses of image steganography in secure communication and data protection.

ACKNOWLEDGEMENT

We wish to express our sincere thanks and deep sense of gratitude to our project guide, Dr.**Ramesh R,** Associate Professor, School of Electronics Engineering, for his consistent encouragement and valuable guidance offered to us in a pleasant manner throughout the course of the project work.

We are extremely grateful to **Dr. Susan Elias,** Dean of the School of Electronics Engineering, VIT Chennai, for extending the facilities of the School towards our project and for his unstinting support.

We express our thanks to our Head of the Department **Dr.Mohanaprasad. K for** his support throughout the course of this project.

We also take this opportunity to thank all the faculty of the School for their support and their wisdom imparted to us throughout the course.

We thank our parents, family, and friends for bearing with us throughout the course of our project and for the opportunity they provided us in undergoing this course in such a prestigious institution.

Asmita Ghosh (21BEC1097)

Ashrit Saha (21BEC1257)

Spandan Gupta (21BEC1263)

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 PURPOSE

- To develop an image steganography system for secure data transmission.
- Ensure confidentiality and concealment of sensitive information within images.
- Create a robust and reliable method to hide and extract hidden data from images.
- Enhance data security and prevent unauthorized access or detection of hidden information.

1.2 SCOPE

Figure 1.1 Military Use

Figure 1.2 Protection of Intellectual property(IP)

CHAPTER 2

DESIGN AND IMPLEMENTATION

2.1 INTRODUCTION

RISE IN DATA

Today, the world is seeing an unprecedented data explosion. The amount of data that we generate each day is simply staggering. According to the Forbes article "How Much Data Do We Create Every Day?" at our current rate, around 2.5 quintillion bytes of data every day, but the rate is only increasing as the Internet of Things (IoT) develops. 90% of the data in the world was generated in only the last two years.

What is Image Steganography?

Steganography is the process of concealing a secret message within a bigger one in a way that prevents anyone from knowing its existence or contents.

Steganography is used to keep two parties' communications between them secret. People that want to transmit a hidden message or code use steganography. Although steganography has many useful applications, it has also been shown that malware authors utilise it to hide the transmission of dangerous code.

Advantage of using Steganography over Cryptography

Techniques for steganography are increasingly utilised in conjunction with cryptography to provide security levels to hidden data. With this strategy, the secret message can be kept hidden and protected in nations where encryption is prohibited. Information is converted into ciphertext via cryptography, which is incomprehensible without a decryption key. Steganography masks the message's existence, while cryptography converts the data to ciphertext, making it challenging to intercept and decipher the encryption process. Steganography, on the other hand, masks the message's existence without changing the message's format.

Types of Steganography

On various transmission medium, including photos, video, text, and audio, steganography work has been done.

- Text Steganography
- Image Steganography
- Video Steganography
- Network Steganography (Protocol Steganography)

2.2 DESIGN APPROACH

Steganography technique Working

In contemporary digital steganography, information is first encrypted or otherwise obscured, and then it is inserted using a special algorithm into information that is part of a certain file format, such a JPEG image, audio file, or video file. There are numerous techniques to insert the hidden message into regular data files. One method is to conceal data in bits that correspond to consecutive rows of the same colour pixels in an image file. The output will be an image that looks just like the original but has "noise" patterns of unencrypted data. This is accomplished by applying the encrypted data to this redundant data in some covert manner.

One typical application of steganography is the addition of a watermark, which is a logo or other identifying information concealed in multimedia or other content files. Online publishers frequently employ the method of watermarking to trace the origin of media assets that have been discovered being shared without authorization.

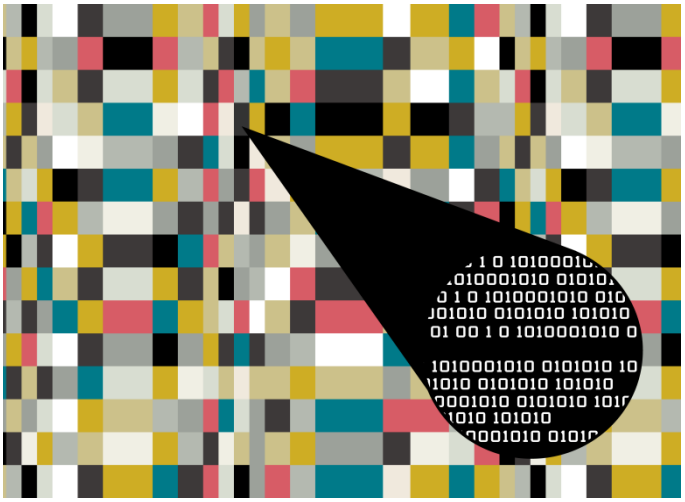

Figure 2.1 Image Steganography

Although there are many various applications for steganography, such as hiding sensitive data within certain file formats. By storing the message with smaller bites in the data file, it is possible to ensure that no one viewing the image file can tell the difference between the original image file and the encrypted file. You can carry out this procedure manually or with the aid of a steganography tool.

‘Least Significant Bit’ Steganography

A digital image can be thought of as a limited number of digital values, or pixels. The tiniest component of an image, a pixel stores values that indicate the brightness of a given colour at any given instant in time. Therefore, an image may be thought of as a matrix (or a two-dimensional array) of pixels with a set number of rows and columns.

With the Least Significant Bit (LSB) approach, the last bit of each pixel is changed to contain the data bit for the hidden message.

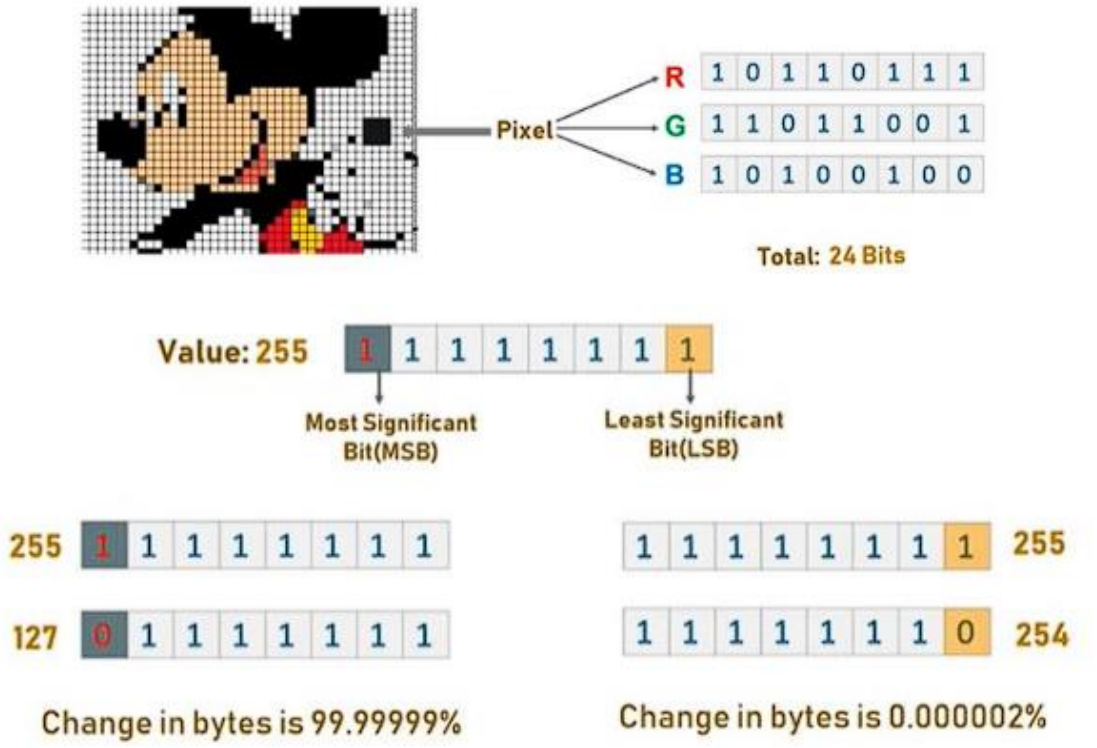

FIGURE 2.2 LSB Steganography

We employ least significant bit steganography because it is obvious from the preceding image that changing the MSB will have a greater influence than changing the LSB on the final result.

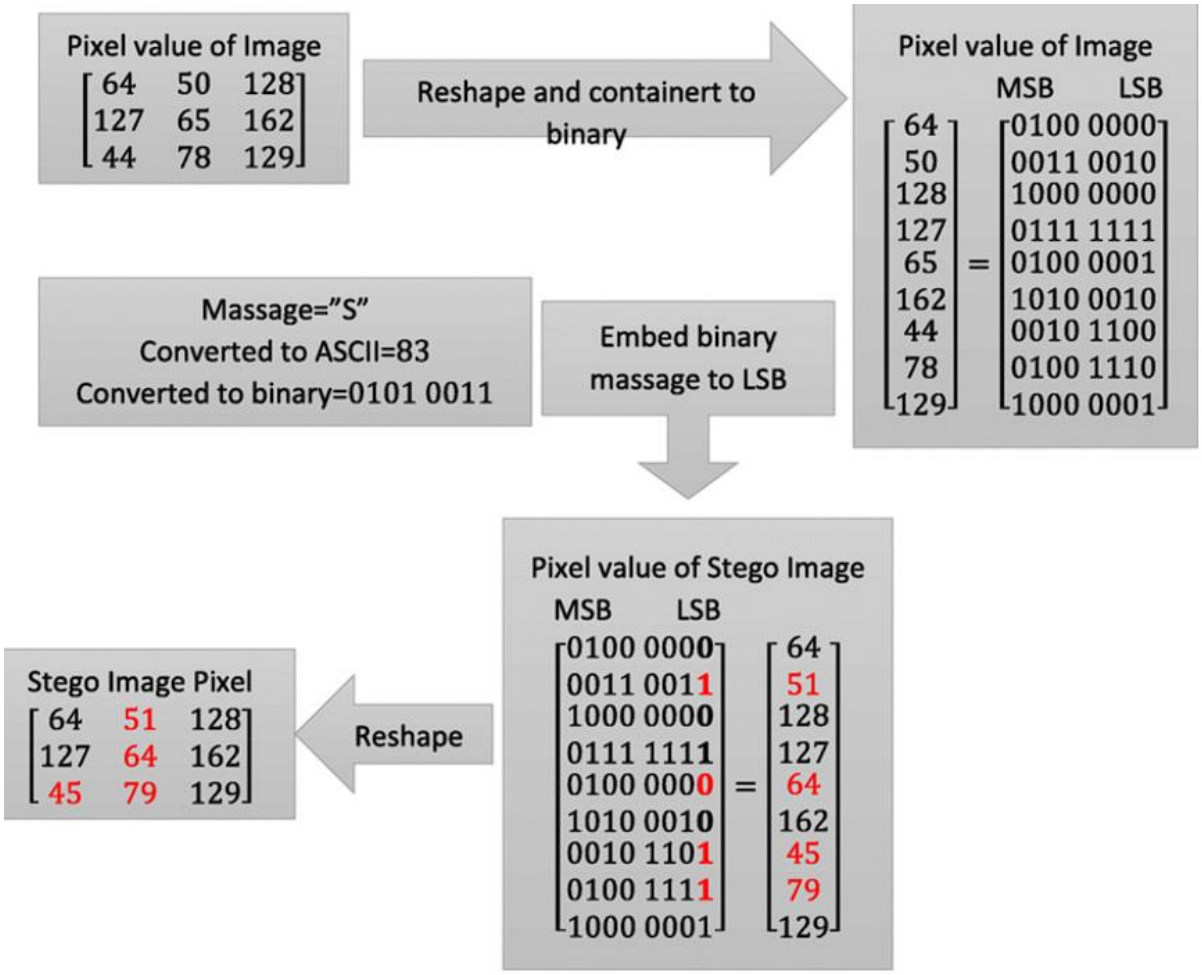

Figure 2.3 LSB Image Steganography Working

Advantages of this method:

- Compared to other image Steganography techniques this one is quick and simple to use.
- The output image differs from the input image by a very little amount.
- We can embed the message in the last two LSBs rather than just the LSB, which allows us to incorporate even big messages.
- This technique serves as the foundation for numerous sophisticated algorithms.
- We can embed the message in the last two LSBs rather than just the LSB, which allows us to incorporate even big messages.

Disadvantages of this method:

- Instead of only the LSB, we can embed the message in the last two LSBs, allowing us to include even large messages.

- This approach is outdated because it was employed in the past, before alternative encoding techniques were created.
- Depending on how many pixels are modified while embedding the message in more than one LSB, the image quality may decline.

2.3 OVERVEIW OF SOFTWARE

Data can be hidden using a variety of techniques, including encoding it to make it ready to be hidden inside another file, tracking which parts of the cover text file contain the hidden data, encrypting the data to be hidden, and allowing the intended recipient to decrypt the hidden data.

Software Used – MATLAB R2023a

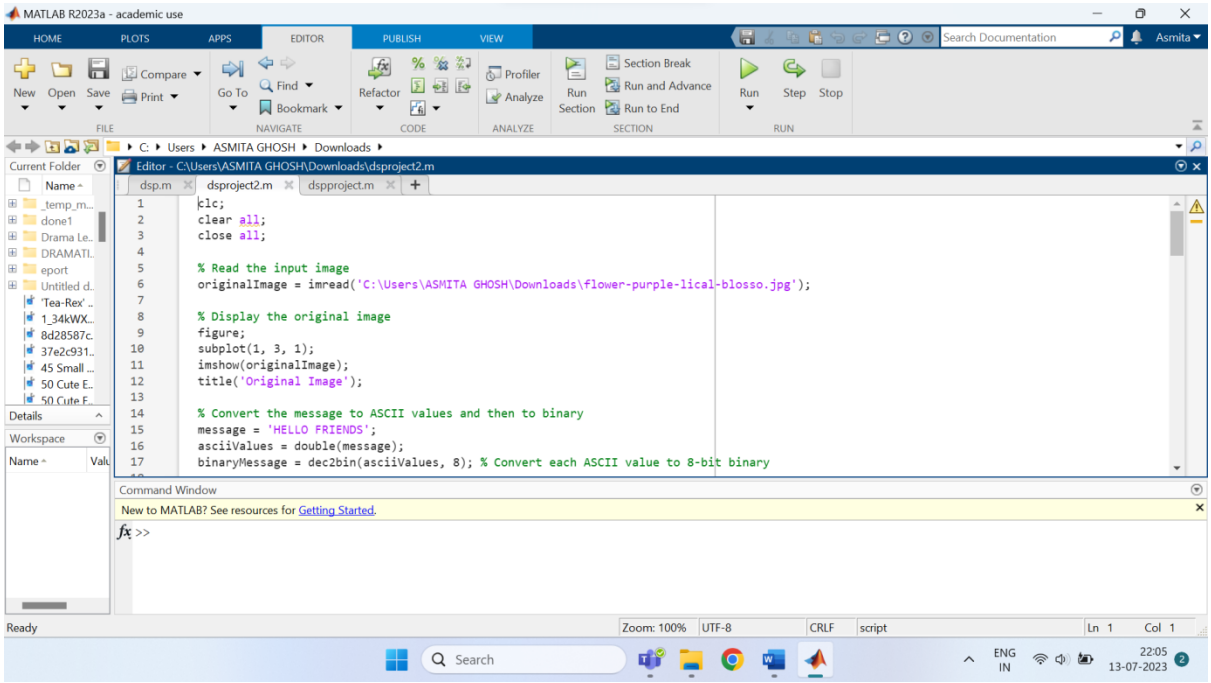

Figure 2.4 Matlab

CHAPTER 3

RESULTS AND ANALYSIS TESTING

3.1 WORK DONE

CODE:-

```
clc;
clear all;
close all;

% Read the input image
originalImage = imread('C:\Users\ASMITA GHOSH\Downloads\flower-purple-lical-
blosso.jpg');

% Display the original image
figure;
subplot(1, 4, 1);
imshow(originalImage);
title('Original Image');

% Convert the message to ASCII values and then to binary
message = 'Steganographically       ';
asciiValues = double(message);
binaryMessage = dec2bin(asciiValues, 8); % Convert each ASCII value to 8-bit
binary

% Prepare the encoded image as a copy of the original image
encodedImage = originalImage;

% Encode the binary message into the least significant bit of each pixel
binaryIndex = 1; % Index to track the position in the binary message
[M, N, ~] = size(originalImage);
pixelValues = zeros(M, N); % Matrix to store pixel (ASCII) values

for i = 1:M
    for j = 1:N
        % Check if there are more bits to encode
        if binaryIndex <= numel(binaryMessage)
            % Get the current pixel value
            pixelValue = double(originalImage(i, j));

            % Convert the pixel value to ASCII string
            pixelAscii = num2str(pixelValue);

            % Store the ASCII value in the matrix
            pixelValues(i, j) = str2double(pixelAscii);

            % Convert the pixel value to binary
            binaryPixel = dec2bin(pixelValue, 8);

            % Modify the least significant bit of the pixel value
            binaryPixel(8) = binaryMessage(binaryIndex);

            % Convert the modified binary pixel back to decimal
```

```
            newPixelValue = bin2dec(binaryPixel);

            % Update the pixel value in the encoded image
            encodedImage(i, j) = uint8(newPixelValue);

            % Increment the binary index
            binaryIndex = binaryIndex + 1;
        else
            % All bits have been encoded, exit the loop
            break;
        end
    end
    if binaryIndex > numel(binaryMessage)
        break;
    end
end

% Display the encoded image
subplot(1, 4, 2);
imshow(encodedImage);
title('Encoded Image');

% Create the check image to visualize the number of pixels consumed
checkImage = encodedImage;
pixelsConsumed = binaryIndex - 1; % Subtract 1 as the index was incremented after
the last encoding
checkImage(:, 1:pixelsConsumed) = 255; % Set the pixels to white

% Display the check image
subplot(1, 4, 3);
imshow(checkImage);
title('Check Image');

% Display the pixel (ASCII) values of the image
subplot(1, 4, 4);
imshow(uint8(pixelValues));
title('Pixel (ASCII) Values');

% Display the pixel (ASCII) values in the command window
disp('Pixel (ASCII) Values:');
disp(mat2str(pixelValues));

% Decoding process
decodedMessage = '';
binaryIndex = 1;
while true
    % Extract the least significant bit from the current pixel
    pixelValue = double(encodedImage(1, binaryIndex));
    binaryPixel = dec2bin(pixelValue, 8);
    extractedBit = binaryPixel(8);

    % Check if the extracted bit is a backslash (/)
    if extractedBit == '/'
        break; % Exit the loop if backslash is found
    else
        % Append the extracted bit to the binary message
        decodedMessage = strcat(decodedMessage, extractedBit);

        % Increment the binary index
```

```
            binaryIndex = binaryIndex + 1;
        end
    end

% Convert the binary message back to ASCII characters
decodedAscii = reshape(decodedMessage, 8, []).';
decodedMessage = char(bin2dec(decodedAscii))';

% Display the decoded message
fprintf('Decoded Message: %s\n', decodedMessage);
```

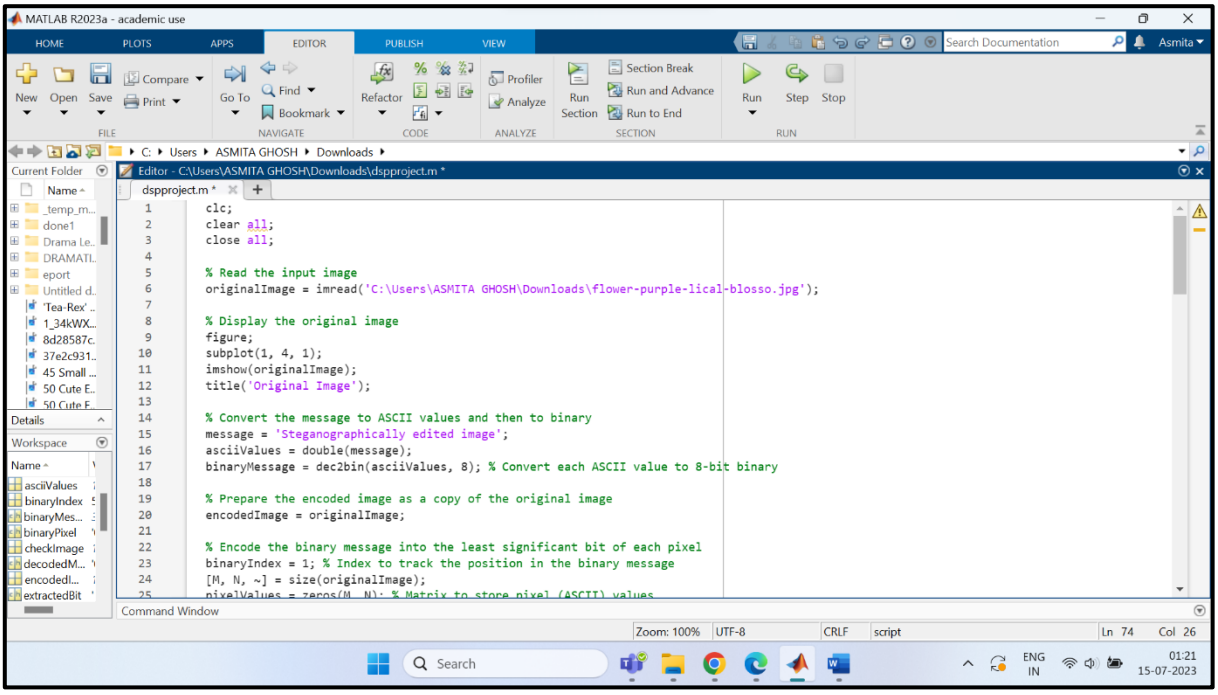

Figure 3.1 MATLAB CODE

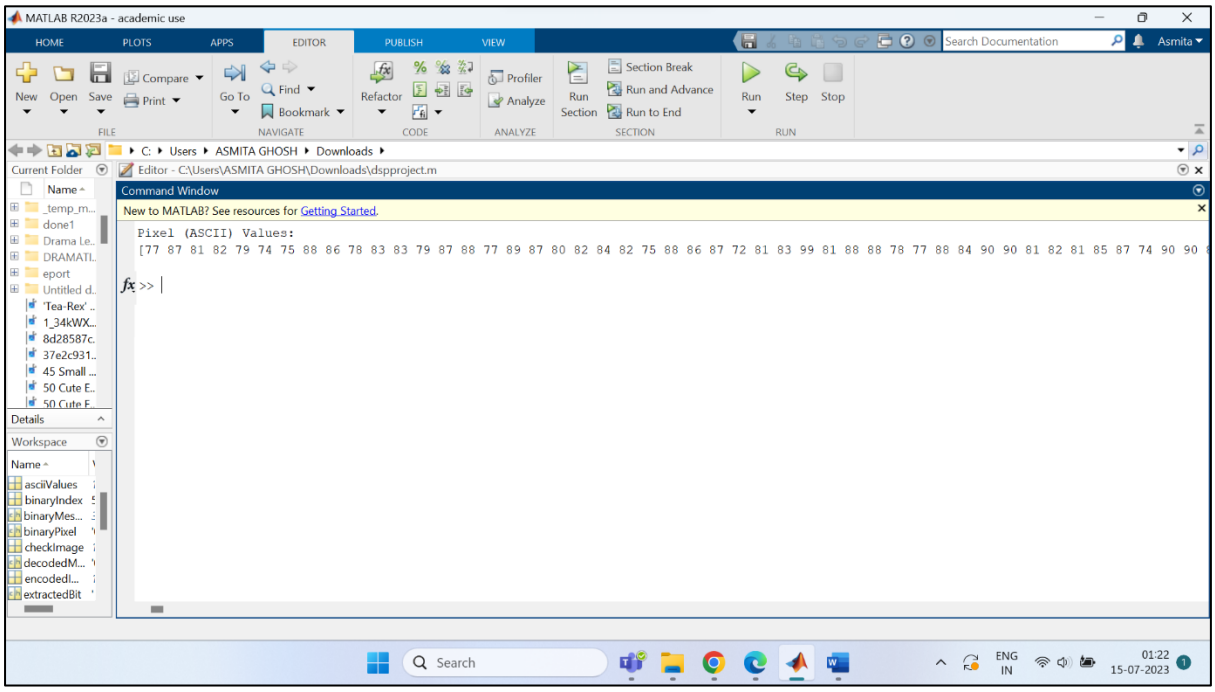

Figure 3.2 PIXELS OF IMAGE IN ARRAY MATRIX

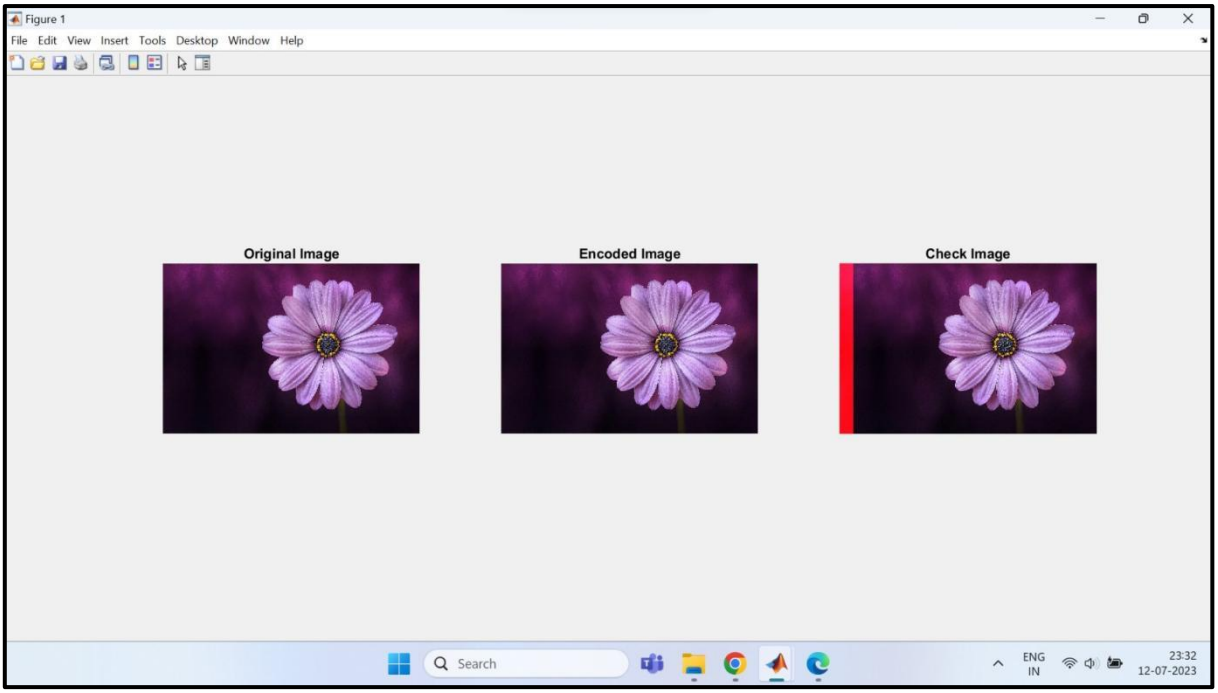

Figure 3.3 MATLAB OUTPUT

The input and output images appear identical to the human eye, as shown in the screenshot up top. The message is included in the output image. The red line shows the text that has been steganography and it's width increases and decreases according to the length of the text.

To check if image is steganographed:

```
clc;
clear all;
close all;
% Read the cover image
coverImage = imread('C:\Users\ASMITA GHOSH\Downloads\flower-purple-lical-blosso.jpg');
% Read the secret message or data
secretData = 'Digital Signal Processing';
% Convert secret data to binary
binaryData = dec2bin(uint8(secretData), 8); % Assuming 8-bit ASCII encoding
% Reshape the binary data into a single row vector
binaryData = binaryData(:)';
% Calculate the number of pixels required to embed the secret data
numPixels = numel(binaryData);
% Check if the cover image can hold the secret data
totalPixels = numel(coverImage);
```

```
ifnumPixels>totalPixels
error('Cover image does not have enough pixels to embed the secret data.');
end
% Embed the secret data into the LSBs of the cover image
stegoImage = coverImage;
stegoImage(:) = bitset(stegoImage(:), 1, str2num(binaryData));
% Display the cover image and stego image
figure;
subplot(1, 2, 1);
imshow(coverImage);
title('Cover Image');
subplot(1, 2, 2);
imshow(stegoImage);
title('Stego Image');
% Save the stego image
imwrite(stegoImage, 'stego_image.jpg');
```

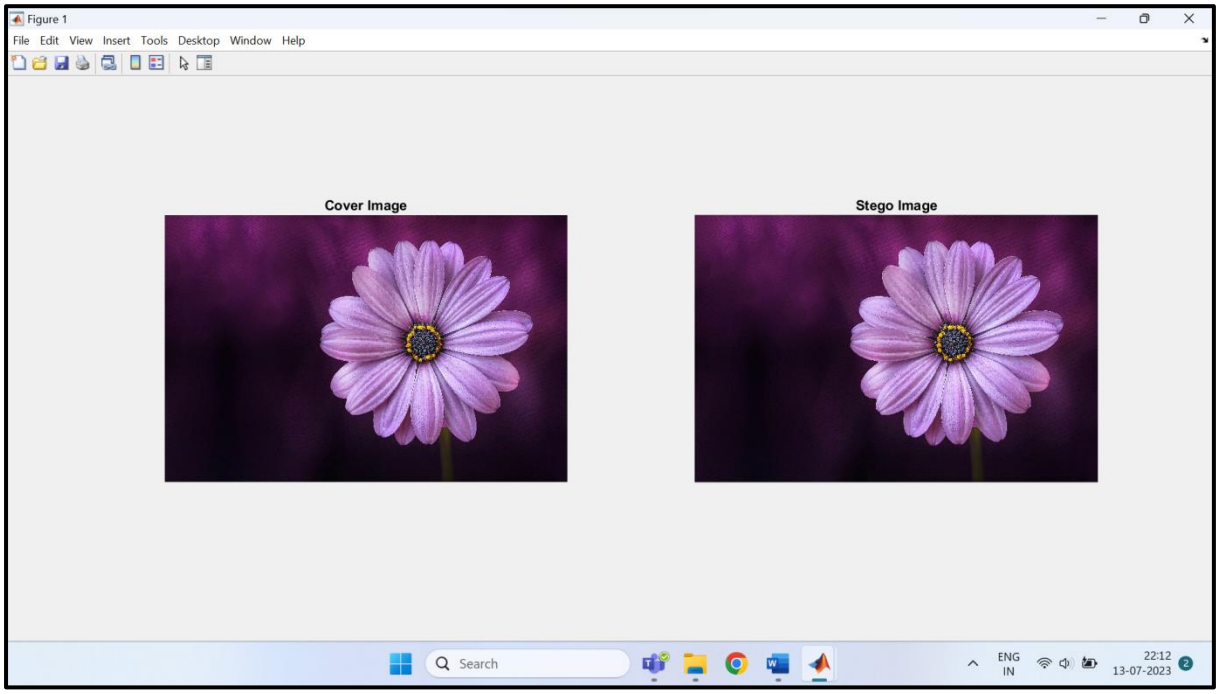

Figure 3.4 MATLAB OUTPUT FOR CHECKING

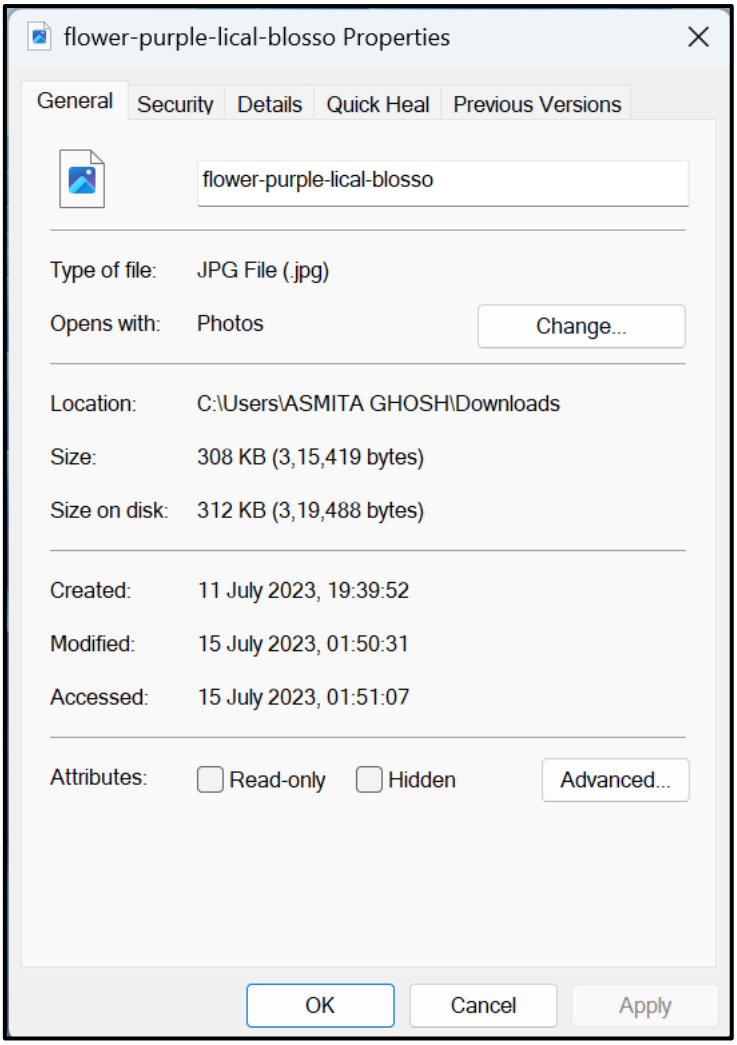

Figure 3.5 INPUT IMAGE'S SIZE

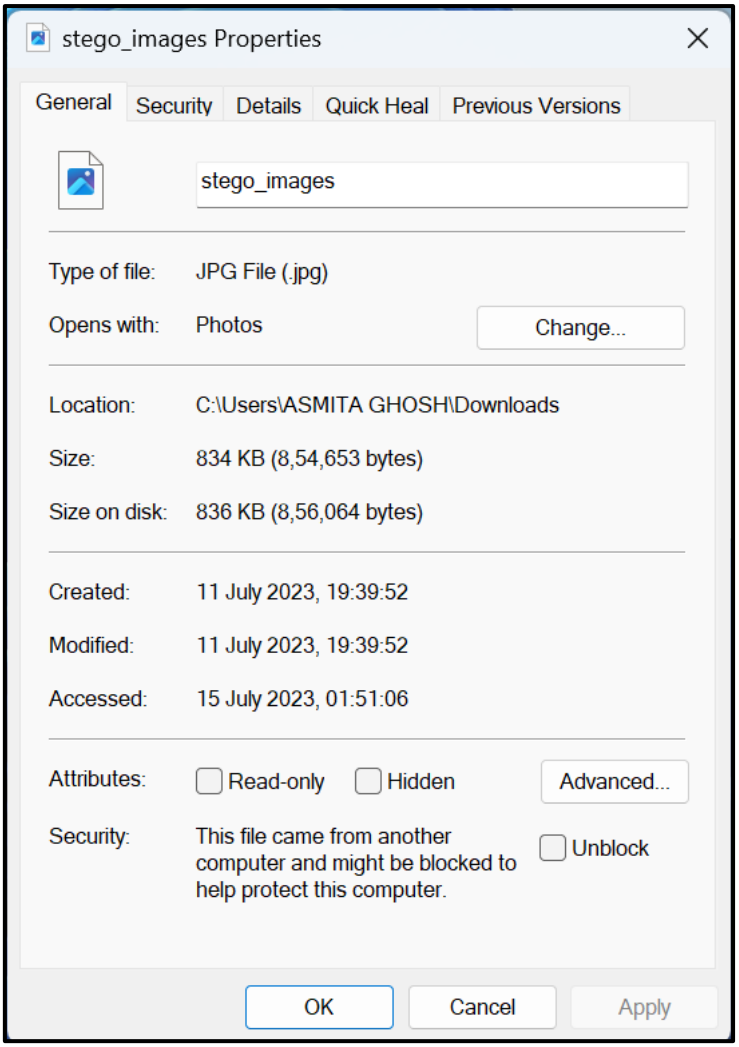

Figure 3.6 OUTPUT IMAGE'S SIZE

This part of the code and the result save the output steganography image in the file and we can see it's memory size has increased. This effectively proves the successful implementation. Our input image has thus been watermarked that isn't visible to the naked eye, but can be used for secure message transmission.

CHAPTER 4

CONCLUSION AND FUTURE ENHANCEMENT

4.1 CONCLUSION

• We can draw the conclusion that steganography is a very effective method of watermarking photos, and it also requires little processing effort.

• The Matlab/Octave codes are extremely memory-efficient and flawless. The user may easily understand the encoding and decoding process..

4.2 FUTURE ENHANCEMENT

But the concept of steganography is still relatively new. The world of computers is constantly evolving, thus it stands to reason that steganography will also improve. There will probably soon be more effective and sophisticated Steganalysis methods. The increased sensitivity to subtle cues is a positive development.

It is hoped that Steganalysis will progress in the future, making it much simpler to identify even minute information contained within a picture.

APPENDIX

The project report's appendix section contains more details and supporting documentation on the subject of image steganography. It contains an encoded image, some sample MATLAB code, and the outcomes of the encoding and decoding operations. The appendix tries to improve comprehension and give a thorough rundown of the completed research.

REFERENCES

- Image Steganography project report authorised by Parth V, Deep V and Divakar T
- Geeks for geeks

BIODATA

NAME: Ashrit Saha

MOBILE NUMBER: 7439933150

E-MAIL: ashrit.saha2021@vitstudent.ac.in

PERMANENT ADDRESS: 10/1/2R Atal Sur Road, Kolkata - 700015

NAME: Asmita Ghosh

MOBILE NUMBER: 9339982402

E-MAIL: asmita.ghosh2021@vitstudent.ac.in

PERMANENT ADDRESS: R-9/2, Saratpally, Midnapore, West Bengal, 721101

NAME: Spandan Gupta

MOBILE NUMBER: 9173715137

E-MAIL: spandan.gupta2021@vitstudent.ac.in

PERMANENT ADDRESS: C-504 Nandanvan-3 vesu Surat