IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Steganography with higher security and Estimation of quality of images

Archana singh Parmar, Dr. Bijendra Singh

1.Phd scholar,BMU Rohtak, India

2. Professor,BMU Rohtak , India

Abstract : Today, the biggest issue when providing data to anyone is making sure that no one can read our message because hackers are becoming more skilled and we want to hide the fact that anything private is being transferred. In this work, we have combined encryption with steganography and then performed the image quality estimation using machine learning to know the difference between the cover image and processed image. Therefore, authentication, integrity, and confidentiality principles should be achieved to fully secure our data. In this paper different samples of data has been taken to check the effectiveness of work and comparision.

Keywords: ECC, Stegnography, image quality, machine learning

1. Introduction

The objective of this research project is to investigate various image cryptography techniques and apply Elliptic Curve Cryptography (ECC) to enhance the security of images. The research aims to explore the effectiveness of ECC in encrypting and decrypting images while ensuring confidentiality and integrity. Additionally, this proposed objective estimates the quality of the recovered images and assess the fidelity of the decryption process compared to the original images. By achieving these objectives, the research aims to provide insights into the robustness and reliability of Encased image security methods, contributing to the advancement of image encryption and quality evaluation techniques. Enhancement of image security through the Elliptic Curve Cryptography (ECC) method and estimation of image quality using machine learning is powerful combination to protect and assess the security of images

2. Working

This research has five major steps to achieve the objective as one is collection of images, second is apply the Elliptic Curve Cryptography (ECC) for Image Security, third is Estimation of Image Quality using Machine Learning as well as the standard parameters and the final forth is Integration of ECC and test images. Then perform the steganography process using Secure cover selection.

Step 1: Collection of Images

a) Gather a diverse dataset of original images representing different content types, formats, and resolutions.

b) Ensure the dataset includes both authentic images and images subjected to various encryption scenarios to simulate potential attacks and different encryption settings.

c) Collect ground truth data with image quality metrics (MSE, PSNR) for each original image.

Step 2: Apply Elliptic Curve Cryptography (ECC) for Image Security

a) Implement the ECC encryption algorithm using appropriate libraries or custom code.

b) Encrypt the original images using ECC with a public key to ensure confidentiality and integrity during image transmission or storage.

c) Generate corresponding private keys for each encrypted image to allow decryption.

Step 3: Estimation of Image Quality using Machine Learning

a) Extract relevant features from the original and decrypted images, such as MSE, PSNR, and SSIM.

b) Prepare the dataset for machine learning, including the original and decrypted images along with their respective quality metrics.

c) Train a supervised machine learning model using the dataset to estimate image quality based on the extracted features.

Step 4: Integration of ECC and Test Images

a) Select a set of 25 test images to assess the effectiveness of ECC encryption and decryption process.

b) Apply ECC encryption to the test images using the same public keys used in Step 2.

c) Decrypt the encrypted test images using the corresponding private keys.

d) Use the trained machine learning model from Step 3 to estimate the image quality of the decrypted test images and compare it with the ground truth data.

e) Analyze the results to assess the security and fidelity of the ECCbased image encryption.

Step 5: Perform Steganography Process using Secure Cover Selection

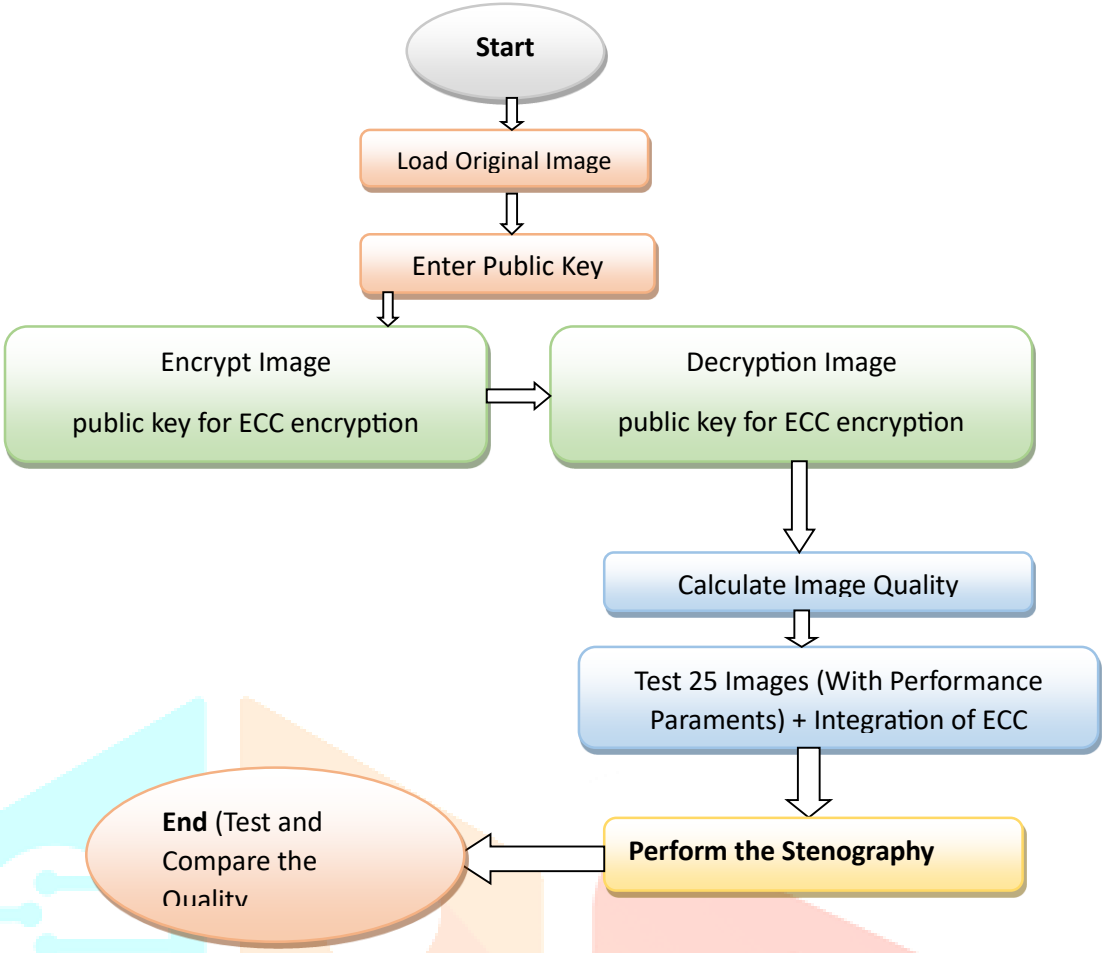
a) Apply steganography techniques to hide additional information within the ECC-encrypted images.

b) Choose secure cover images carefully to make detection of hidden data challenging.

c) Ensure that the steganography process does not compromise the integrity or security of the ECC encryption.

3. Working in depth :

The research entails five key steps, first, assembling a diverse image dataset; second, implementing Elliptic Curve Cryptography (ECC) for image security; third, utilizing machine learning to assess image quality; fourth, integrating ECC encryption and quality evaluation; and fifth, performing steganography to conceal data within ECC-secured images. The objective is to bolster image security while preserving quality, allowing for covert data embedding. This research has practical applications in safeguarding sensitive image data and transmitting hidden information within images while maintaining their integrity.



3.1 Data Collection

To conduct the research on enhancing image security through ECC and estimating image quality using machine learning, a diverse dataset of original images and their encrypted/decrypted counterparts will be collected. The dataset will include images of varying resolutions, formats, and content types to ensure representativeness. Additionally, it will encompass both authentic images and images subjected to various encryption and decryption scenarios, simulating potential attacks and diverse encryption settings. The dataset will also involve quality metrics calculated using PSNR and MSE. This comprehensive dataset will serve as the foundation for training a supervised machine learning model to accurately estimate image quality and validate the effectiveness of ECC in securing images.

3.2 Elliptic Curve Cryptography (ECC) for Image Security:

ECC is a widely used publickey cryptography method that relies on the difficulty of the elliptic curve discrete logarithm problem for its security. It offers stronger security compared to traditional methods like RSA for the same key length. To enhance image security, we apply ECC in two main ways:

a. Encryption: Use ECC to encrypt the image data. The image will be converted into an array of bytes, and ECC used to encrypt this data. The recipient, who possesses the corresponding private key, decrypt the image and view its content. This ensures that unauthorized individuals cannot access the image without the private key.

b. Digital Signature: ECC used to create digital signatures for images. A digital signature provides authentication, integrity, and nonrepudiation. The sender generates a digital signature using their private key, and the receiver verify the signature using the sender's public key to ensure the image has not been tampered with and is indeed from the claimed sender.

3.3 Estimation of Image Quality using Machine Learning:

Assessing image quality is essential to ensure that the image content remains intact during encryption and decryption processes. Machine learning algorithms be trained to estimate image quality based on various image quality metrics, such as PSNR (Peak SignaltoNoise Ratio), SSIM (Structural Similarity Index), and perceptual quality metrics like MSE (Mean Squared Error) and LPIPS (Learned Perceptual Image Patch Similarity).

a. Dataset Preparation: Gather a dataset of original images and their encrypted/decrypted versions. we need pairs of images to train the machine learning model to predict the image quality.

b. Feature Extraction: Extract relevant features from the images and calculate the image quality metrics mentioned above.

c. Model Training: Use supervised machine learning techniques to train a regression model that takes the extracted features as inputs and predicts the image quality metric (e.g., PSNR, SSIM) as the output.

d. Model Evaluation: Evaluate the trained model on a separate test dataset to ensure its accuracy in estimating image quality.

3.4 Integration of ECC and test images

After encrypting an image using ECC, we use the trained machine learning model to estimate its quality. If the estimated quality falls below a predefined threshold, it might indicate that the image's quality has been significantly affected during the encryption process, potentially due to attacks or incorrect parameters. In such cases, we consider reencrypting the image or taking necessary corrective actions. By combining ECC for image security and machine learning for image quality assessment, we use ensure not only the confidentiality and integrity of the images but also evaluate their fidelity after encryption, providing a comprehensive approach to image security. The integration of ECC (Elliptic Curve Cryptography) in image security involves encrypting the original images using ECC with a public key and then decrypting them with the corresponding private key to restore the images. This process ensures confidentiality and integrity during image transmission or storage. Additionally, a set of 25 test images is used to evaluate the effectiveness of the ECC encryption and decryption process, with image quality metrics such as MSE, PSNR, and SSIM employed to assess the fidelity of the decrypted images compared to their originals.

3.5 Steganography Process

In the fifth step of our research plan, we delve into the intricacies of performing steganography with a focus on secure cover selection. Steganography involves concealing confidential information within non-secret data, and in this context, we are embedding additional data within ECC-encrypted images. To achieve this, we start by carefully choosing a secure cover image. This cover image should blend seamlessly with the encrypted image while providing ample capacity to hide the payload. It's essential to select a cover image that is contextually relevant and resistant to steganalysis techniques. Next, we embed the payload within the cover image, using various methods such as LSB substitution or frequency domain techniques. Throughout this process, we must ensure that the ECC-encrypted image's security and integrity remain intact. Testing and evaluation follow, where we examine the stego-images to confirm they closely resemble the original cover images and assess their resistance to detection. Finally, comprehensive documentation and reporting are crucial to detail the steganography process, cover image choices, embedding techniques, and results, including any challenges encountered and the level of security achieved. This step allows we to successfully hide information within ECC-encrypted images, making it difficult for unauthorized users to discern the presence of concealed data.

4. Mathematical model for Image analysis

Assume that we have an original image represented by a twodimensional matrix "M" of size H x W, where H is the height and W is the width of the image. Each element M(i, j) represents the pixel value at row "i" and column "j."

4.1) Image Encryption using ECC:

The encryption process represented as follows:

C(i, j) = E(M(i, j), PubK)

Above, C(i, j) is the encrypted pixel value, E() is the encryption function, M(i, j) is the original pixel value, and PubK is the public key used for encryption.

4.2) Image Decryption using ECC:

The decryption process represented as follows:

M_dec(i, j) = D(C(i, j), PrivK)

Above, M_dec(i, j) is the decrypted pixel value, D() is the decryption function, C(i, j) is the encrypted pixel value, and PrivK is the private key used for decryption.

4.3) Image Quality Estimation using Mathematical Model:

To estimate the image quality, we use a mathematical model that compares the original image "M" with the decrypted image "M_dec" and assigns a quality score "Q" based on the difference between the two images. One common method is Mean Squared Error (MSE), which measures the average squared difference between corresponding pixel values in the two images.

MSE = (1 / (H * W)) * ΣΣ((M(i, j) M_dec(i, j))^2)

Here, MSE represents the Mean Squared Error, H and W are the height and width of the image, and the summation is performed over all pixels in the image.

4.3.1)Peak SignaltoNoise Ratio (PSNR):

PSNR is another metric used for image quality estimation. It measures the ratio between the maximum possible pixel value and the MSE. The higher the PSNR, the better the image quality.

PSNR = 10 * log10((max_pixel_value^2) / MSE)

Above, max_pixel_value represents the maximum possible pixel value in the image (e.g., 255 for an 8bit grayscale image).

4.3.2) Structural Similarity Index (SSIM):

SSIM is a more complex metric that considers structural information, luminance, and contrast of the images. It provides a more perceptually accurate quality assessment.

SSIM = (2 * μ_M * μ_M_dec + C1) * (2 * σ_M_M_dec + C2) / (μ_M^2 + μ_M_dec^2 + C1) * (σ_M^2 + σ_M_dec^2 + C2)

Above, μ_M and μ_M_dec are the means of the original and decrypted images, σ_M and σ_M_dec are the standard deviations, and C1 and C2 are constants to stabilize the division. The summations are calculated over all pixels.

By combining the encryption and decryption processes with ECC and using mathematical formulas to estimate image quality, we analysed the effectiveness of ECCbased encryption while quantifying the quality of decrypted images.

5. Procedure

5.1) ECC Encryption Implementation:

- ✓ Implement the ECC encryption algorithm using libraries like OpenSSL or a custom implementation.
- ✓ Create a MATLAB function for ECC encryption, e.g., `ecc_encrypt`, that takes the original image matrix and the public key as inputs and returns the encrypted image matrix.

5.2)Image Quality Estimation:

- ✓ Implement the image quality estimation algorithms (MSE, PSNR, SSIM) using mathematical formulas provided earlier.
- ✓ Create MATLAB functions for MSE and PSNR calculations, e.g., `calculate_mse`, and `calculate_psnr` respectively.

5.3)Create a GUI in MATLAB:

- ✓ Use MATLAB's builtin GUI development tools (e.g., GUIDE or App Designer) to create a graphical user interface (GUI).
- ✓ Design the GUI to have a button or file selector to load the original image, an input field for the public key, and buttons to perform encryption, decryption, and quality estimation.

5.4)Load and Display Images in the GUI:

- ✓ Add functionality to the GUI to load and display the original images.
- ✓ After loading an image, display it in a suitable GUI component (e.g., axes).

5.5)Image Encryption in the GUI:

- ✓ Implement the functionality to perform image encryption using ECC in the GUI.
- ✓ When the "Encrypt" button is pressed, call the `ecc_encrypt` function to encrypt the loaded image with the provided public key and display the encrypted image.

5.6) Image Decryption in the GUI:

- ✓ Implement the functionality to perform image decryption using ECC in the GUI.
- ✓ After encryption, add a "Decrypt" button in the GUI to decrypt the encrypted image using the corresponding private key and display the decrypted image.

5.7) Image Quality Estimation in the GUI:

- ✓ Implement the functionality to estimate image quality using MSE, PSNR, and SSIM in the GUI.
- ✓ After encryption and decryption, add buttons or fields to calculate and display the quality metrics (MSE, PSNR, SSIM) between the original and decrypted images.

5.8) Testing 25 Images:

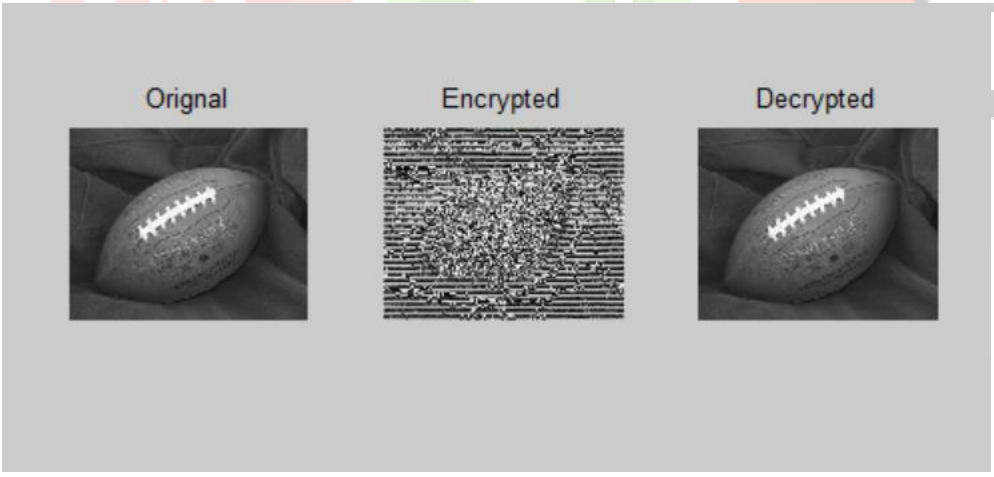
- ✓ Prepare a set of 25 test images (e.g., in a folder).
- ✓ In the GUI, add a feature to select and load any of the test images for encryption, decryption, and quality estimation.

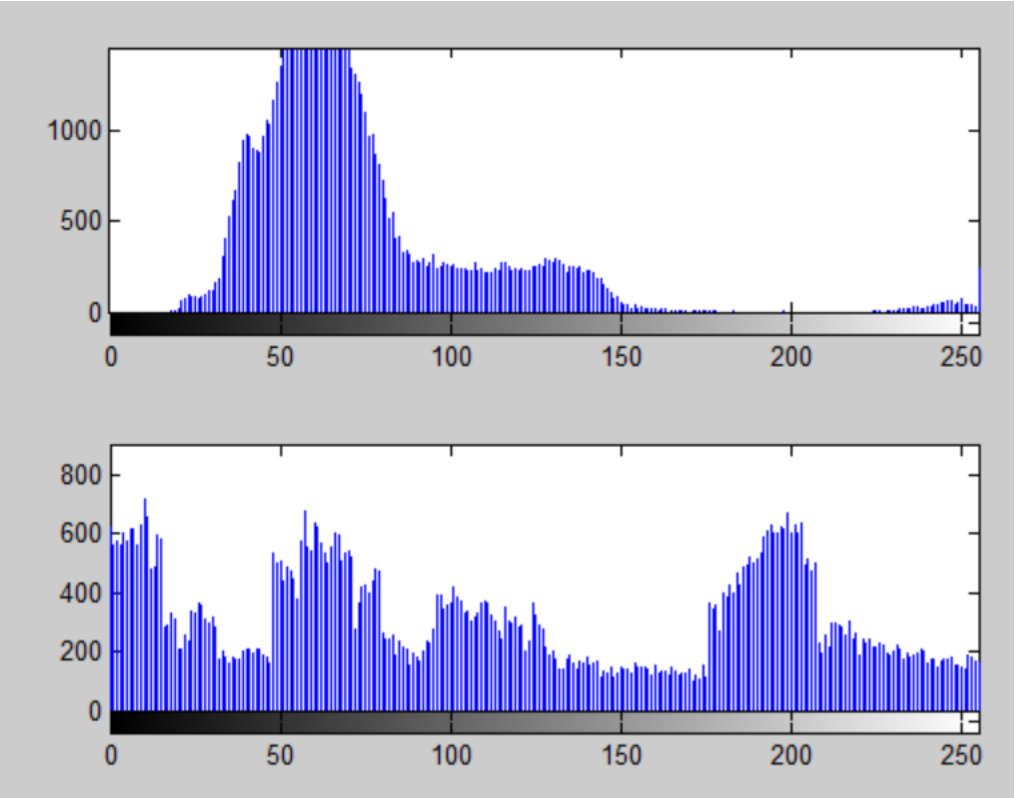
5.9) Run the GUI:

- ✓ Save the MATLAB GUI and run it.
- ✓ Load the original image and perform encryption, decryption, and quality estimation.
- ✓ Test the GUI on 25 images and observe the results.

6. Simulation and Result

Phase 1 – Result (pic 1)





Done

Elapsed time is 0.487376 seconds.

Re =

6.6861 7.8210

Can take more pics

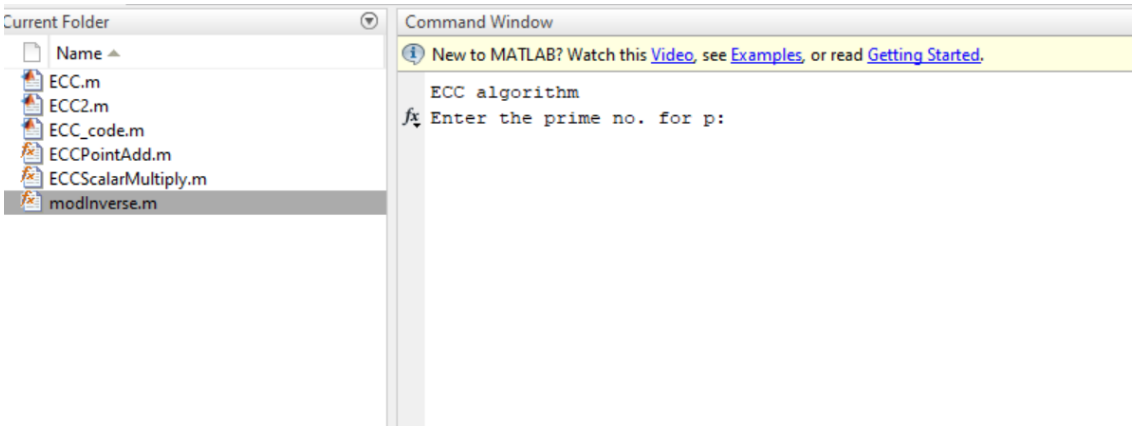
phase -2

```
clc;
clear;

% Rest of the main code

% Function calls
resultPoint = ECCScalarMultiply(point, scalar, a, p);
resultPoint = ECCPointAdd(point1, point2, p);
inverse = modInverse(a, p);

% Rest of the main code
```

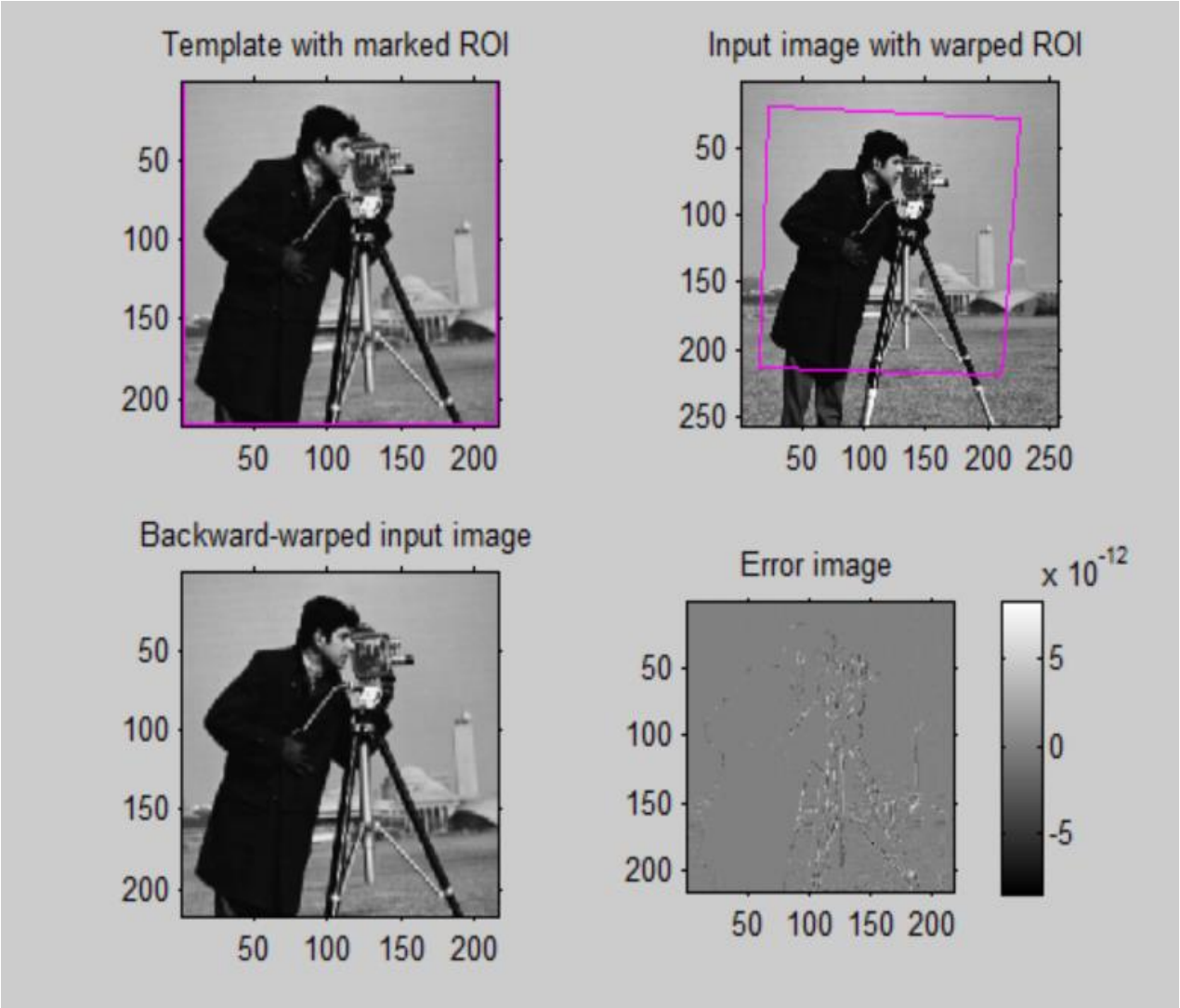



```
d=145
Public key is (17,667)
Private key is (145,667)
Enter the message: Hellow
ASCII equivalent of message
    72   101   108   108   111   119


The encrypted message is
    591 417 280 280 136 2
The decrypted mes in ASCII is
    72  101 108 108 111 119
The decrypted message is: Hellow
```

Phase 3

ECC- Process



In testing we takes five cover images and messages to be hiding then we notice size of

CI,MI and EI the result will be shown-:

CI size	MI size	EI size	Decode	Result
1. 16.5mb	14.1kb	16.5mb	No change	100%
2. 2.92 mb	10kb	2.92mb	No change	100%
3. 436kb	4.84kb	436kb	No change	100%
4. 271kb	11.4kb	271kb	No change	100%
5. 1.37 mb	11.4 kb	1.37mb	No change	100%

7. **Conclusion** : In trail we different cover image and message image are taken then the size has been noted and writen in table and after process encoding and decoding.Observation show that there is no change in any one of image so imperceptibility is high and even we hid the message of different size in cover image still the EI size is same as CI size and result is 100 Percent every time. So this technique is quite effective

8. REFERENCES

1. Hemanta Kumar and Mohanta M, Secure Data Hiding Using Elliptical Curve Cryptography and Steganography, International Journal of Computer Applications (0975 – 8887) , December 2014 , 108, pp.3.

2. Yamunesh Goswami, Anuj Bhargava and Prashant Badal ,Improved Method For A Secure Image Cryptography Based On RSA and DES Algorithm and LSB Steganography Technique, International Journal of Advance Engineering and Research Development, 2017, 4 (11), pp. 731-737.

3. Md. Khalid Imam Rahmani, Kamiya Arora and Naina Pal, A Crypto-Steganography: A Survey, (IJACSA) International Journal of Advanced Computer Science and Applications, 2014, 5 (7), pp.149.

4. Hyder Yahya Atown, Hide and Encryption Fingerprint Image by Using LSB and Transposition Pixel by Spiral Method, International Journal of Computer Science and Mobile Computing, 2014, 3 (12), pp. 624-632.

5. Rdharani and M.L. Valarmathi,A Study on Watermarking Schemes for Image Authentication, International Journal of Computer Applications, Vol.2, 2010, pp.24-32.

6. M. Mohamed Sathik and S.S. Sujatha,An Improved Invisible Watermarking Technique for Image Authentication, International Journal of Advanced Science and Technology, 2010, 24, pp. 61-73.

7. Chandhari and B.L. Gunjal, Image Watermarking Algorithm in DWT Domain, nternational Journal of Modern Engineering Research, 2012, 2, pp.1940-1943.

8. Ashadeep Kaur, Rakesh Kumar and Kamaljeet Kainth, Review Paper on Image Steganography. International Journal of Advanced Research in Computer Science and Software Engineering, 2016, 6 (6), pp. 499-502.

9. Anil Kumar,A Secure Image Steganography Based on RSA Algorithm and Hash- LSB Technique, IJARCSSE, 2013, 3 (7), pp. 363-372.

10. Ankita Gangwar and Vishal Srivastava. Improved RGB -LSB Steganography Using Secret Key, International Journal of Computer Trends and Technology (IJCTT), 2013, 4, pp. 85-89

11. Abikoye Oluwakemi, AdewoleKayode S and OladipupoAyotunde J, Efficient Data Hiding System using Cryptography and Steganography, International Journal of 78 SSAHE-Journal of Interdisciplinary Research, Vol. 1, Issue 1, 2020