Image Steganography: Beyond LSB

Enrique Casillas enriquec@mit.edu
Michael Kuoch mkuoch@mit.edu
Thanh Nguyen thanh_n@mit.edu
Steven Qu xsqu@mit.edu

6.857 Final Project Report
Spring 2022

Abstract

Image steganography is the technique of hiding secret information inside of an image. Besides the traditional least significant bit method, methods involving QR decomposition and domain transforms have become quite popular. This paper analyzes different combinations of this method, and shows that for the most part, a domain transform followed by QR factorization in single steganographic operation yields the highest stego image and recovered image quality. Inspired by these methods, a new symmetric-key steganography scheme is proposed, which offers a few advantages when compared to regular steganography. There are also some challenges related to image steganography that are discussed, including the difficulties in performing steganography under image compression.

Keywords: Image steganography, LSB, QR method, Discrete Wavelet Transform, Discrete Fourier Transform, SSIM, PSNR, Wasserstein, symmetric-key steganography

1 Introduction

Hiding information from adversaries is one of the key features of any security system, systems which are incredibly important in today's digital space. Different fields take different approaches towards keeping information secure. Image steganography is one of those methods, where the goal is to hide from users the fact that a secret even exists. This project evaluates several newer image steganography techniques by computing a set of evaluation metrics, and proposes a symmetric-key steganography scheme inspired by such methods.

First, we discuss image steganography and several techniques based on QR factorization and domain transforms outlined in Section 2. We present the evaluation of these methods in Section 3, then propose and evaluate a symmetric-key steganography scheme in Section 4. Lastly, we discuss in Section 5 the relationship between steganography and security by going through several security concepts such as robustness, detection, and game-theoretic definitions.

1.1 What is Steganography?

Steganography is the practice of concealing a message, image, or file within another message, image, or file. As compared to cryptography, which has the goal of converting secret content to prevent adversarial actors from uncovering those secrets, steganography focuses on hiding secret content to prevent adversarial actors from detecting that a secret even exist.

1.2 How does Steganography work?

The basic premise behind stegonography is the hiding of some **secret** media inside a **cover** medium. This paper specifically covers image steganography, so the secret and cover are both images. Figure 1 shows the high-level process of image steganography. Given a cover image C and a secret image S, a steganographic model f takes C and S as inputs and outputs Z, a **stego** image. The stego image looks like the cover image, but is also hiding the secret S, acting like a "carrier" whenever it is sent to a recipient through some channel. Because the stego image looks almost identical to the cover image, adversaries may not recognize that there is a secret being passed along. The process of converting a cover image and secret image into a stego image is called the **embedding** process, since we are embedding the secret information inside of the cover.

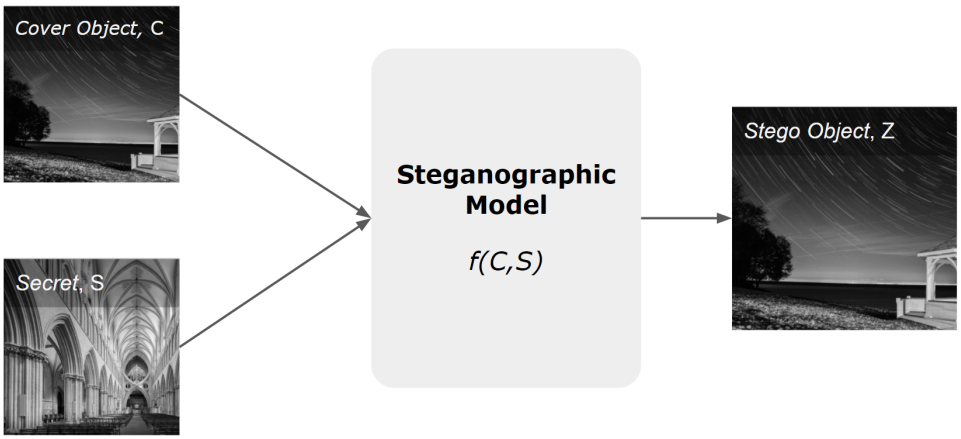

Figure 1: The basic premise behind the embedding process of image steganography. A stego-image is produced by hiding a secret image inside of a cover image.

The second half of steganography involves recovering the secret information from the stego object. In this case, the goal is to reconstruct the secret image with as little loss in quality as possible. This process is called the **extraction** process. Typically the stego image alone is not enough to be able to recover the secret image, so there may exist some sort of key or existing variable c that is used to recover the secret. Formally, the extraction process is defined through a function g that takes in as inputs the stego image Z and (optionally) that code c. Thus $g(Z, (c))$ outputs the **recovered** secret S , where the goal is to get $S \approx S$, that is, it looks like the original secret. We can refer to the functions/algorithms f and g by their respective processes, embedding and recovering.

1.3 Challenges of Steganography

In contrast to the challenges of cryptography, the challenges of steganography can be categorized as capacity, robustness, and transparency. Capacity is defined as amount of data that can be hidden in a cover-media while maintaining imperceptibility. Robustness is defined as the ability of the secret to be undamaged while extracting the stego-media. Transparency is defined as how perceivable the differences between the cover-media and stego-media are. For our analysis of different methods, we focus on evaluations that highlight the robustness and transparency of the final stego-image, with less emphasis on capacity.

2 Methods

We consider a variety of image steganographic methods. The most straightforward method to hide information in a way that is easily recoverable is the Least Significant Bit (LSB) method, which we use as a baseline control. Aside from LSB, we consider other steganographic operations which take greater advantage of the fact that images are matrices. We discuss how these operations can be composed to form more complex embeddings.

2.1 Least Significant Bit

Least Significant Bit (LSB) is an image steganographic method that hides information within the last n bits of a given cover image, for some chosen n. It takes advantage of the fact that these last n bits have the same information capacity as any n bits, but modifying them does not significantly change the visual appearance of the cover image. To fool the human eye, $n = 3$ is generally acceptable [5].

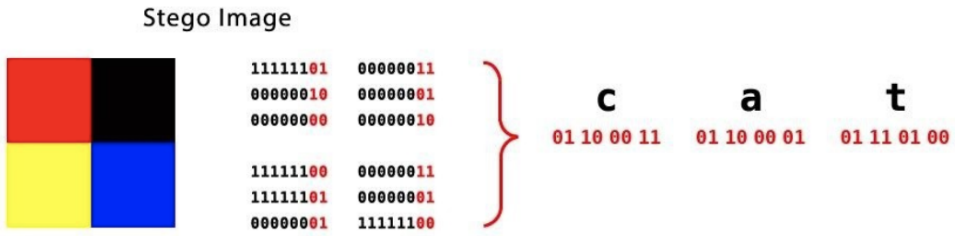

Figure 2: The word "cat" is hidden within a four-pixel RGB image, with $n = 2$ [6].

LSB is simple in terms of embedding and in terms of extraction, which confers some advantages. The secret does not need to be another image; thus, it can be decomposed in a way that is not immediately obvious to an onlooker. Also, the original cover image is not required in the extraction process, reducing the amount of communication needed to execute the scheme.

At the same time, LSB's simplicity entails several major weaknesses. For one, the total information that a cover image can plausibly hide will always be a fraction of its own size, so a cover image will not be able to perfectly hide a secret image of the same size using LSB; some form of compression must be taken, which reduces image reconstruction quality. Plain LSB is statistically a bad steganographic method: even if they can fool the human eye, LSB-produced stego images are easily detectable via steganalysis which looks at statistical anomalies in the last bits of images.

2.2 Basic steganographic operations

In this section, we consider three steganographic operations which use the fact that we have two images of identical size, the cover and the secret, each of which can be treated as matrices. Denote the cover image matrix as C and the secret image matrix as S.

The first operation is QR decomposition, which factors a given matrix into the product of two matrices, an orthogonal matrix Q and an upper triangular matrix R. This factorization is useful because the Q matrix can be effectively thought of as an arbitrary permutation, similar to a block operation in block ciphers. Knowing one of Q_{secret} or R_{secret} will not reveal anything about S [2].

To produce the stego image using QR decomposition, one factors both $C = Q_{\text{cover}} R_{\text{cover}}$ and $S = Q_{\text{secret}} R_{\text{secret}}$. Then, one sets $Q_{\text{stego}} = Q_{\text{cover}}$ and

$$R_{\text{stego}} = R_{\text{cover}} + \alpha \cdot R_{\text{secret}},$$

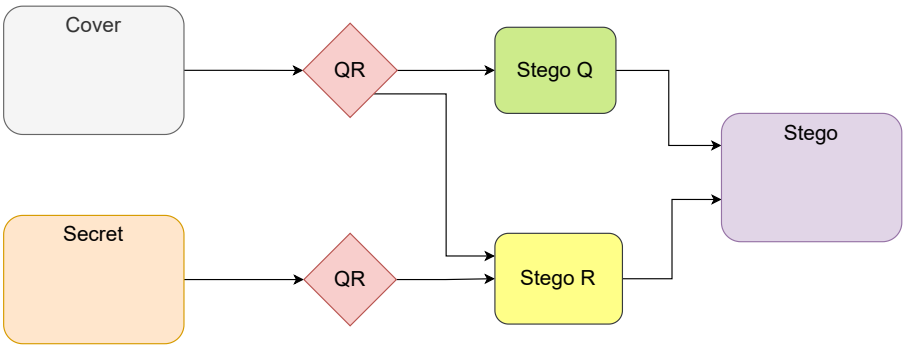

Figure 3: QR method only.

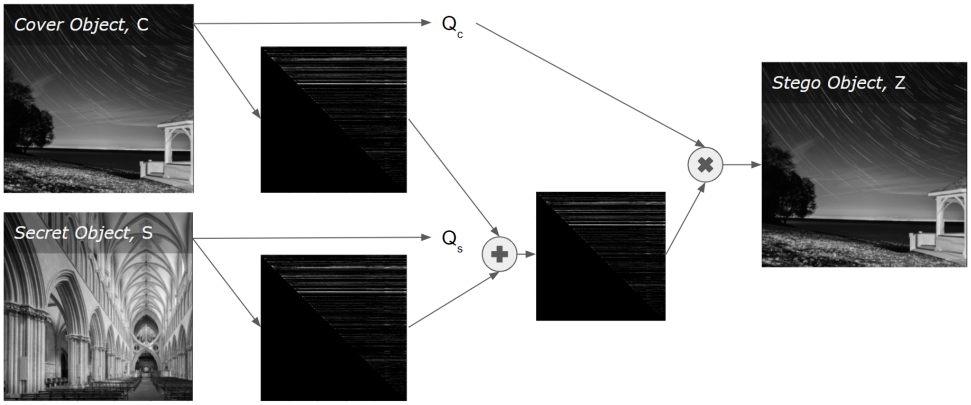

Figure 4: QR decomposition example process.

where $\alpha < 1$ is used to hide the presence of the secret image. The stego is then given by $Q_{\text{stego}} R_{\text{stego}}$. Figure 3 shows a flow diagram of the embedding process, and Figure 4 shows it applied to an actual cover image and secret image.

The extraction process involves taking a QR decomposition of the stego image and reverse-engineering the embedding. This reverse process will not be perfect due to rounding issues involving α, but it should produce an image similar to the original secret [2].

The second operation we consider is the general discrete transform, specifically the Discrete Fourier Transform (DFT) and the Discrete Wavelet Transform (DWT). Each essentially decomposes the image matrix based on spatial frequency, finding coefficients for each relevant frequency. In the DFT, each frequency corresponds to a sinusoidal function; in the DWT, they correspond to filters defined by a wavelet, a matrix of coefficients which one chooses ahead of time. For this paper, we use the Haar wavelet (also known as db1) as the wavelet of choice.

Given this decomposition, one can combine the lower frequency signals to produce a low-frequency component of the image matrix, and correspondingly a high-frequency component. This is nearly analogous to the Q and the R terms in QR decomposition, where the low-frequency component corresponds to R, where most of the information is present, while the high-frequency component corresponds to Q, which is nearly arbitrary and can be put on top of any sensible complement.

The third operation we consider is histogram shifting. This method applies to each pixel a function which only depends on the value of that pixel, and not its location. For example, one transformation could increment the brightness of each pixel by some value, taken mod 256. Essentially, this is equivalent to tracking the frequency of each pixel value in a histogram and then applying a transformation to that histogram. We do not implement or investigate this operation further in our work, mainly because it does not fully synergize with the nice matrix structure of the images.

2.3 Combining different operations

We combine different operations from the previous section to form our embeddings. Figure 3 already details the first method, which only uses QR decomposition. Our second method applies a discrete transform to both the cover image C and secret image S before applying QR decomposition to the respective low-frequency components; to construct the stego, the reverse transform must be applied, with an approximation of the respective high-frequency components added back in. Figure 5 shows a flow diagram of the embedding process for the second method.

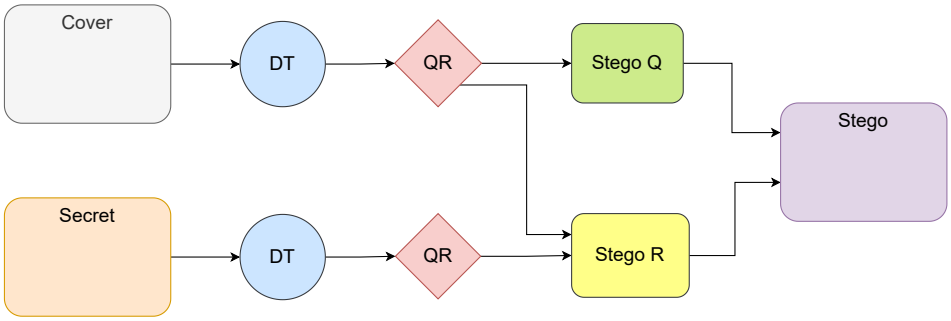

Figure 5: Discrete transform followed by QR method.

We also consider a third method where the order of the operations is reversed: we do a QR decomposition first, and then we apply the discrete transform. In this case, the secret image is hidden via combining its low-frequency approximation with a low-frequency approximation of the cover image; to put together the stego image, the high-frequency component of the cover image is used as the high-frequency component of the stego image. Additionally, the QR decomposition is optional; we consider two variants, the first which applies the QR decomposition to both images, and the second which applies the QR decomposition only to the secret image. Figure 6 shows a flow diagram of the embedding process for the third method.

In total, we consider eight distinct embeddings. The first method yields a single embedding. The second method can be used with DFT or DWT, producing two embeddings. The third method can be used with DFT or DWT for each of two variants, producing four embeddings. Finally, LSB is used as a control.

3 Evaluation

Evaluation is performed on two different image pairs. The first image pair involves comparing the cover image and stego-image to determine the transparency property of the steganographic methods. The second image pair involves comparing the secret image to the recovered secret image to determine robustness.