Decisive Image Characteristics to Perform Image Steganography in DCT

JUDITH BRAVO DE MEDINA ARRIBAS, University of Twente, The Netherlands

Steganography is the practice of hiding secret information inside something that is no secret, for instance, an image, audio message or even text. These practices have long been used in communications since, at first glance, if one does not know there is data hidden, it is imperceptible; thus, to extract the information, mathematical methods to obtain this data need to be followed. For instance, steganalysis is the study attempting to detect hidden messages in steganography. This research paper analyses a generated image steganography dataset, composed of 21 images, to obtain the typical characteristics of images used in image steganography, such as picture size, colour (RGB, Grayscale) or the capacity of hidden message. The study focuses on the transform domain, more precisely Discrete Cosine Transform (DCT), performing image steganography on the dataset elements through several available image steganography tools to achieve such a purpose. Among the tools used in the research, there are M4JPEG, Jstego and F5. Besides, Aletheia was used as a steganalysis tool throughout the research. Furthermore, different types of secret data were hidden inside the 21 dataset images; among these, 3 images and 4 text files of distinct characteristics.

Additional Key Words and Phrases: Image Steganography, Transform Domain, Discrete Cosine Transform, Secret Data, Cover Image, Stego Image.

1 INTRODUCTION

Different variations of steganography have been going around since the ancient Greek era; more than 2500 years later, new inventions have come to play, but its principles remain the exact [20]. Following the evolution in the digital era, messages hidden have been adapting to vary in form, from plain-text messages to audio or even video. The use of these secret message methods is becoming more and more popular nowadays. Governments use them as a type of hidden communication [5, 8, 12, 38], and even businesses have been using a form of steganography in everyday documentation with the use of watermarking [5, 20] or protection of intellectual property [20]. The key to image steganography resides in the fact that it is practically undetectable; unless it is known that there is a message hidden in the image, one would not perceive there is data hidden inside [14]. With the growth of digital steganography in recent years, the use of steganography in digital aspects has become a usual practice, precisely because it is not that obvious that data has been hidden [20]. However, as already established after using steganography for communication during the attacks on the Twin Towers in September 2001 [14], this methodology has been used by terrorists to communicate and adapted as a way of performing cyberattacks [17, 36]. This research is focused on analysing different images before and after the performance of image steganography. It was performed with a generated dataset of images with diverse characteristics (such as sizes or colours). The focus is on the transform domain, as for the manipulation of JPEG images, which is one of the popular image file formats nowadays [14, 27]. Regarding JPEG images, the most outstanding type in the transform domain for such purposes in image steganography is the Discrete Cosine Transform (DCT) [13, 19, 27], which is the centre of this research. DCT is believed to have less image resolution [14] once performed image steganography than other methods in the domain. Thus, the importance of this research is to analyse the characteristics of images and those existing tools searching which one is better for performing such image steganography. Among the existing DCT image steganography tools there are, for instance, M4JPEG [2], Jstego [35], Stegote [39], F5 [9] or Jsteg [7]. Lastly, a steganalysis tool, Aletheia [10], was used to process the stego images through steganalysis practices to prove if confidential data could be found among the image.

1.1 Research Questions

This research aims to analyse and review different DCT image steganography tools to obtain outstanding features by analysing the different image characteristics in the transform domain and comparing the differences of the images among the various DCT tools. Having stated the problem and the background of the research, the following research question can be inferred from it:

What are the specific outstanding features of an image such as colour (RGB or Gray-scale), image size and size of the hidden message space decisive to performing image steganography?

Furthermore, to answer the proposed research question, the following sub-research questions should be answered:

(1) If subjected to steganalysis, are these images easily identifiable as altered images, obtaining the hidden message?
(2) From the sample of DCT-steganography tools (M4JPEG, Jstego, F5 [2, 9, 35]) to perform DCT steganography, which is/are the most accurate (in terms of lowest Mean Square Error, MSE) as to be used?

The remainder of the paper is structured as follows: Section 2 discusses Related Work relevant to this research. In Section 3, the methodology followed during the study is explained. Section 4 presents the results and the following discussion of the research. Finally, Section 5 displays the conclusions and future work.

2 RELATED WORK

Distinct research has been performed in the field of image steganography, analysing the different methods of performing a secure communication hiding messages in images [6, 25], and distinct methodologies to extract the hidden information [24], as well as steganalysis [6]. Their focus is on the colour specifications of the images and the differences between a cover image and the altered one, based on the pixel colour and distribution in different pixel sections of the image [6, 8, 11, 36].

Research in image steganography at the frequency domain has been performed. They focused on methodologies used to perform image steganography in the different types (such as DCT) inside

TScIT 37, July 8, 2022, Enschede, The Netherlands

the transform domain [6, 8]. Moreover, there are already performed studies on the colour, gray-scale [23, 28, 31], and size of the images [23, 28, 32] in image steganography. These can be useful to have a broader view of image steganography in the transform domain and the further purpose of this research.

Having taken into account the existing literature, this research aims to bring to the field a new perspective on the outstanding features that images possess in DCT to be chosen to perform image steganography. A whole view of specific image features will be obtained, thus, providing more insight into steganalysis [6] with the possibility of mitigating future steganography cyberattacks.

Furthermore, research has been performed on the available mechanisms applicable to the transform domain [14, 25, 26, 31, 32] in image steganography. Nevertheless, focusing on DCT, there is a lack of analysing together the common highlights/ characteristics that make an image more suitable for performing image steganography. Naming a few of these common characteristics to be studied can be the image size, colour (RGB, Grayscale) or capacity of hidden messages. The study of DCT has been performed in several papers [13, 19], but there is a lack of studying both the commonly possessed image characteristics and the study of these by comparing the results of various DCT image steganography tools.

3 METHODOLOGIES

To achieve the aim of this research several steps were followed based on a study through different DCT-applied Image Steganography tools and the steganalysis procedure. Before conducting such a study, the images used in the research had to be gathered to compose an adequate dataset [18]. These 21 images shaping the dataset were obtained from distinct sources of images available to use for the public [3, 4, 22] and others taken with the use of a digital camera (Canon EOS 200D [1], contact the author for further information regarding these images). The images were chosen so that there were different characteristics represented among the pictures defining the dataset, for instance, different colour ranges (RGB, 24-bit images; or Grayscale, 8-bit images), pixel size and bytes size. Table 1 presents the characteristics of the RGB cover images in the dataset, 24-bit images, while Table 2 presents the features of the ones classified as Grayscale, 8-bit images. The corresponding images are displayed in Figures 1 (RGB) and 2 (Grayscale).

Additionally, 7 separate files to be used as the information to be hidden, 3 JPEG and 4 TXT files, were the chosen ones to play the role of the secret data. These were selected to obtain results from a variety of hidden files, thus, taking into consideration how the characteristics of a stego image may differ depending on the secret data. The images (labelled as sJpeg1, sJpeg2 and sJpeg3) were chosen to have different sizes and pixel dimensions. At the same time, the text files (labelled as Text 1, Text 2, Text 3 and Text 4) were generated with different file sizes, as can be observed in Tables 3 and 4. This was done so that cover images (Figures 1 and 2) could be studied with different ranges of hidden data and check upon the distinct characteristics after the stego image was obtained.

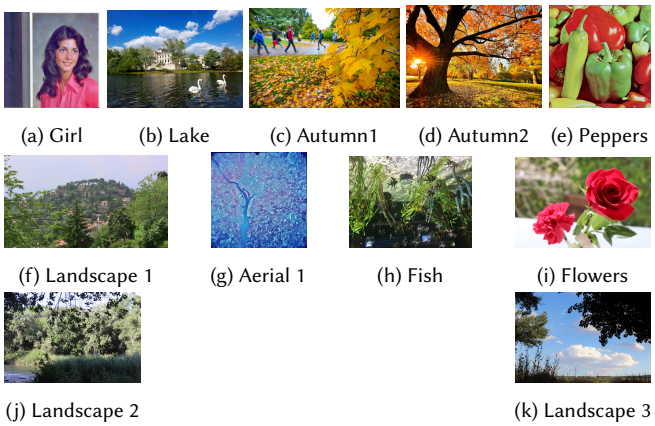
Once the dataset was completed, M4JPEG, Jstego and F5 were downloaded and installed. With this began the analysis of each tool to identify their different features, considering, for instance,

Table 1. RGB Image Cover Dataset Features

Label	Colour	Size	Dimensions
Girl	RGB (24-bit)	25,072 B	256x256
Peppers	RGB (24-bit)	131,965 B	512x512
Autumn1	RGB (24-bit)	364,997 B	600x400
Lake	RGB (24-bit)	327,447 B	600x427
Autumn2	RGB (24-bit)	83,555 B	600x450
Landscape1	RGB (24-bit)	1,161,515 B	1920x1080
Aerial1	RGB (24-bit)	4,968,345 B	2250x2250
Fish	RGB (24-bit)	3,606,220 B	4032x3024
Flowers	RGB (24-bit)	4,718,002 B	6000x4000
Landscape2	RGB (24-bit)	12,365,329 B	6000x4000
Landscape3	RGB (24-bit)	6,947,826 B	6000x4000

Table 2. Grayscale Cover Image Dataset Features

Label	Colour	Size	Dimensions
Boat	Grayscale (8-bit)	112,662 B	512x512
Bridge	Grayscale (8-bit)	112,662 B	512x512
Tank	Grayscale (8-bit)	98,221 B	512x512
Aerial2	Grayscale (8-bit)	387,991 B	1024x1024
Man	Grayscale (8-bit)	319,330 B	1024x1024
Plane	Grayscale (8-bit)	274,377 B	1024x1024
Landscape4	Grayscale (8-bit)	3,802,114 B	6000x4000
Landscape5	Grayscale (8-bit)	1,418,459 B	6000x4000
Landscape6	Grayscale (8-bit)	1,331,510 B	6000x4000
Landscape7	Grayscale (8-bit)	1,194,365 B	4032x3024



(a) Girl (b) Lake (c) Autumn1 (d) Autumn2 (e) Peppers

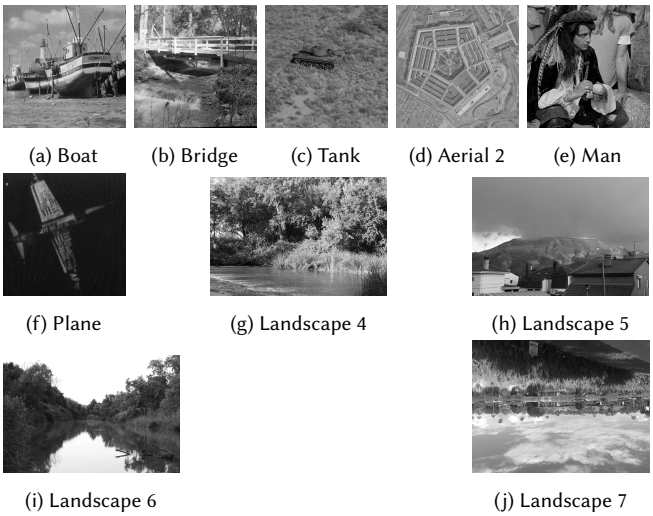
(f) Landscape 1 (g) Aerial 1 (h) Fish (i) Flowers

(j) Landscape 2 (k) Landscape 3

Fig. 1. RGB Cover Images

which cover images could be used per tool and the different data to be hidden that was supported. Furthermore, a steganalysis tool was downloaded, Aletheia, with a broad spectrum of methods to compare cover and stego images.

3.0.1 Metrics . Various performance metrics were taken into consideration for the development of this research. When dealing with images and hiding images inside of them, it was essential to carefully choose the metrics used to obtain conclusive and accurate



(a) Boat (b) Bridge (c) Tank (d) Aerial 2 (e) Man

(f) Plane (g) Landscape 4 (h) Landscape 5

(i) Landscape 6 (j) Landscape 7

Fig. 2. Grayscale Cover Images

Table 3. Image Secret Data Description

Label	File	Image	Size	Details
sJpeg 1	JPEG		27,058B 256x256	RGB, 24-bit
sJpeg 2	JPEG		20,679B 300x430	RGB, 24-bit
sJpeg 3	JPEG		273,090B 2048x3072	Grayscale, 8-bit

results that could be compared among the different images and DCT tools. First, calculating the embedding capacity (also known as embedding payload) of the image, in other words, the percentage representation of embedding bits in the overall pixels of the cover image, was fundamental. This way, we could understand the cover image's capacity to hide a message. Therefore, the embedding rate (ER) was used for this purpose. Equation 1 displays the formula to calculate the ER [40]. N stands for the number of embedded secret bits, while H and W stand for the height and width of the picture (pixel size), respectively.

$$ER = \frac{N}{HW} bbp \tag{1}$$

The Mean Square Error (MSE) was chosen to calculate the difference between the cover and stego images. MSE formula can be found in Equation 2, where m and n represent the dimensions of the image, c stands for cover image and s for stego image. MSE has not defined threshold; however, the lower the value, the better [21, 29].

Table 4. Text Secret Data Description

Label	File	Text	Size
Text 1	TXT	This is a hidden message	24 B
Text 2	TXT	This is the second hidden message as a text file and we are trying to prove if there is a big difference between the first text file and the second one *(x4)*	610 B
Text 3	TXT	The hidden message for this example aims to be of a bigger size than the other text file which is 26 bytes. There are different tools to be used in the development of this research project, more precisely the ones applied for DCT in the Transform Domain are the ones we focus on, and this text file is going to be used as a hidden message in the images. *(x3)*	1,075 B
Text 4	TXT	This is the last text file to be used as a trial of hidden message, this one is the one with the biggest number of characters and the results will differ from the ones obtained in the previous text files as it is big. *(x46)*	10,072 B

$$MSE = \frac{1}{mn} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (c(i,j) - s(i,j))^2 \tag{2}$$

Lastly, Equation 3 presents the formula to calculate the Peak-Signal-to-Noise-Ratio (PSNR). It was chosen to obtain the quality of the stego image, basically obtaining the degree of similarity between the cover and stego image. To calculate PSNR, we need the MSE, obtained from equation 2, and the MAX pixel value of the image. These maximum values will differ depending on the type of image; if we encounter an 8-bit image (Grayscale), the maximum value to be obtained will be $2^8 = 256$. On the other hand, for 24-bit (RGB) images, MAX will have a different value; in this case, $2^{24} = 16777216$. Considering the distinct value of MAX, so would be the range of PSNR, where the lowest the value, the more distortion is the stego image and the higher the value, the better the quality. For 8-bit images, if we obtain a MSE value of 1, $20log_{10}(255) = 48.13dB$, therefore if lowering the MSE value under 1, would give a value not bigger than 60dB. Following the same strategy, the minimum value obtained would be $20log_{10}(255/\sqrt{255}) = 20.06dB$. On the other hand, for 24-bit images, following the same reasoning as for 8-bit ones, for an MSE of 1, we have $20log_{10}(16777216) = 144.5dB$ [34], which lowers the MSE value under 1 will provide a PSNR no bigger than 170dB and no lower than $20log_{10}(16777216/\sqrt{16777216}) = 72.24dB$.

$$PSNR = 10\log_{10}\frac{(MAX-1)^2}{MSE} = 20\log_{10}\frac{(MAX-1)}{\sqrt[2]{MSE}}dB \quad (3)$$

3.1 Performing Steps for Sub-Question 1

(1) Perform image steganography in each image from the dataset (Figures 1 and 2), exploiting the selected DCT tools by using sJpeg1 (Table 3) as the hidden data.
(2) Repeat step 1 for each of the remaining 6 secret messages (Tables 3 and 4).
(3) Choose a sample dataset of the obtained stego images and create a table per DCT-tool with the different characteristics of the cover (Figures 1 and 2) and stego image.
(4) Perform steganalysis with the use of Aletheia [10].

3.2 Performing Steps for Sub-Question 2

(1) For each of the DCT tools, perform the mean MSE and PSNR for each of the RGB stego images of hidden data Text 1.
(2) Repeat step 1 with the other 6 secret messages.
(3) Repeat steps 1 and 2 for Grayscale stego images.

3.3 Performing Steps for the Main RQ

(1) Gather all the information obtained from sub-questions 1 and 2.
(2) Divide the 21 images from the dataset into 4 groups based on the image size.
(3) For the first group and secret data Text 1 (Table 4), get the average of the PSNR from the outputted M4JPEG stego dataset for both RGB and Grayscale images.
(4) Repeat step 3 with the remaining 6 secret data (Tables 3 and 4).
(5) Repeat steps 3 and 4 for the next 3 groups.
(6) Repeat steps 3 to 5 for Jstego and F5.
(7) Place the results on a table providing division by tool and colour range. (Refer to Table 16)

4 RESULTS AND DISCUSSIONS

Considering the steps and methodology discussed in the previous section, both the answer to the sub-questions and the main research question will be discussed in this section, providing the different results of the experiments and research. It is essential to mention that while at the beginning, it was mentioned that 5 tools were going to be used to perform image steganography, the sample has been reduced to 3 (M4JPEG, Jstego and F5). After further studying the functionality of Stegote and Jsteg, it was concluded that those tools could not be used since they are either outdated or have missing files. Thus, the code cannot be run in nowadays environments.

4.1 Sub-Question 1 Results

Hiding a message in each image and defining the dataset for each of the specified DCT image steganography tools was challenging. Each tool is developed uniquely, with specific characteristics that interfere with the ones established by any of the other selected tools. Therefore, from the available 21 images in the dataset, only a few could be used per tool. Which of these were used on each depends not only on the requirements of such application but also on the characteristic of the image, hence the broad spectrum of different images in the dataset.

4.1.1 M4JPEG Tool . The first tool to be analysed in this research was M4JPEG. With it, steganography was performed for each image from the dataset by hiding the different secret data inside of them. Nevertheless, due to specific characteristics of this tool, only 7 images were successfully processed, obtaining a stego image. After checking the successful JPEG and the ones that were not, it could be inferred that Grayscale images were not accepted by this tool, as trying to use one of them as the cover image would throw an error. Furthermore, too large image files (more than 5KB byte size) were not accepted by M4JPEG, throwing an error specifying the file was too large. M4JPEG is focused on hiding secret data inside images having enough capacity to hide a message inside them, not distorting the original photo (in terms of augmenting the cover image dimensions, for instance). Not all 7 secret messages could be embedded inside the supported cover images. For example, when choosing *Autumn1* which is 364,997 bytes, M4JPEG displays that the total image capacity available to hide data is 6430 bytes. Looking at Table 4, the only data that could be hidden inside this cover image will be Text 1, Text 2 and Text 3 since the images (Table 3) have a larger byte size than the available capacity in *Autumn1*.

Table 5. M4JPEG Sample Metrics

Image	Data	ER	MSE	PSNR
Autumn 1	Text 2	0.018	0.10	154.70dB
Autumn 2	Text 1	0.0008	0.01	167.37dB
Lake	Text 3	0.034	0.14	152.96dB
Peppers	Text 1	0.0007	0.01	164.02dB
Landscape 1	sJpeg 1	0.104	0.28	150.0dB
Aerial 1	sJpeg 2	0.033	0.92	144.86dB

To process the obtained results, it was considered to use a sample of the obtained stego images since the total number obtained was up to 26 stego images. Table 5 presents the information of these sample pair cover-stego images obtained from one of the secret data hidden in the cover image. Moreover, to have a broader perspective of the differences between cover and stego images, different metrics were calculated for 3 cover-stego image pairs from the M4JPEG obtained dataset. Among the calculated metrics shown in Table 6 there is the embedding rate (ER), the mean square error (MSE) and the peak-to-noise-signal (PSNR), prior discussed in Section 3.

4.1.2 Jstego Tool . The following tool analysed in this research was Jstego, based on a Jsteg algorithm to perform DCT image steganography. Following the procedure with the previous tool, each image from the dataset was processed with Jstego trying to hide the different secret data (Tables 3 and 4) inside. With this tool, from the 21 images defining the database, only 6 were successful in the process

Table 6. M4JPEG Result Images

Cover	Stego	Colour	S. Data
		RGB 24-bit	Text 2
600x450 83,555B	600x450 510,535B		
		RGB 24-bit	Text 1
512x512 131,965B	512x512 131,972B		
		RGB 24-bit	sJpeg 1
1920x1080 1,161,515B	1920x1080 1,149,763B		

Table 7. Jstego Result Images

Cover	Stego	Colour	S. Data
		RGB 24-bit	Text 4
1920x1080 1,161,515B	1920x1080 484,703B		
		RGB 24-bit	sJpeg 2
6000x4000 4,718,002B	6000x4000 1,426,229B		
		Grayscale 8-bit	Text 1
6000x4000 1,331,510B	6000x4000 1,762,821B		

Table 8. Jstego Sample Metrics

Image	Data	ER	MSE	PSNR
Landscape 1	Text 4	0.039	11.24	133.99dB
Flowers	sJpeg 2	0.007	1.45	142.87dB
Landscape 3	Text 3	0.0004	3.13	139.54dB
Landscape 4	sJpeg 1	0.009	109.83	27.72dB
Landscape 5	Text 2	0.0002	111.11	27.67dB
Landscape 6	Text 1	$8e^{-6}$	92.48	28.47dB

of steganography. The remaining 15 were non-conclusive. Thus, no stego image was obtained from the cover. The main reason is that Jstego does not support images with smaller sizes (in Byte terms). Based on the study of the different images, it can be inferred that to potentially obtain a stego image, the cover should be at least 1KB. As with M4JPEG, this algorithm is also focused on hiding secret data inside images with enough capacity to hide a message (no distortion provoked). The total number of stego images obtained from the 6 accepted cover images was up to 34. If the 7 secret messages displayed in Tables 3 and 4 could have been hidden in all the covers, the sample would have been up to 42. Nevertheless, some of the secret data were too big to be hidden inside the covers due to capacity and size.

As with M4JPEG, the sample of obtained images for Jstego was reduced to 3. Table 7 presents the information of these sample dataset images and the obtained stego images from one of the secret data hidden. Following the procedure, the metrics for 6 different pairs of cover and stego images were also calculated and can be explored in Table 8.

4.1.3 F5 Tool . Finally, the last DCT image steganography tool to be evaluated was F5, which is based on the algorithm of the same name. With this tool, all the images were successful; in other words, all covers (Figures 1 and 2) for all secret data (Tables 3 and 4) returned a stego image. Differing from the previous results obtained using M4JPEG and Jstego, the 21 images shaping the dataset were used as cover images returning at least one stego image. Following the differences with the previous two tools, F5 does not possess any unique characteristics of why some covers are rejected. In fact, none of the images was rejected, no matter the size or colour, for instance. It is also curious that cover images would return a stego image even if the secret data would be higher than the capacity of the image to hide a message. Nevertheless, in this case, it would be easily noticed that the image has been altered since the size of the images is adequate based on the data hidden (hence, a smaller image would be augmented and distorted), which makes the pixels of the image a bit blurry and quick to wonder there is something wrong with the image.

Performing steganography with F5 in the proposed dataset returned up to 147 stego images. Cover images that were smaller than the message to be hidden did return a stego image, thus why the complete dataset could be used with F5 for the 7 proposed secret data files. Out of these 147 images, 3 pairs were chosen to be used as a sample; they can be observed in Table 9 shows the different characteristics of both the stego and cover image. Finally, the metrics calculated for the referred sample images in Table 9, and 3 additional pairs, can be found in Table 10.

Having gathered the information displayed in Tables 5 to 10, the images were subjected to steganalysis, using Aletheia, to analyse if the stego images were identified as altered, specifying that there was a hidden message. It should be noted that when subjected to steganalysis, some of the results were inconclusive, or Aletheia could not detect the hidden data; this will be evaluated later in this section. The total number of stego images obtained with M4JPEG, Jstego and F5 was up to 207 stego images; therefore, a sample of 6 images out of this 207 was chosen. The results of performing

Table 9. F5 Sample Metrics

Image	Data	ER	MSE	PSNR
Girl	Text 3	0.131	23.68	130.75dB
Boat	Text 2	0.019	16.62	35.92dB
Plane	Text 1	0.0002	16.16	36.05dB
Fish	Text 4	0.007	106.55	124.22dB
Landscape 4	sJpeg 3	0.091	5.37	40.83dB
Landscape 3	sJpeg 1	0.009	4.78	137.70dB

Table 10. F5 Result Images

Cover	Stego	Colour	S. Data
		RGB 24-bit	Text 3
256x256 25,072B	256x256 10,143B		
		Grayscale 8-bit	Text 2
512x512 112,662B	512x512 47,192B		
		Grayscale 8-bit	sJpeg 3
6000x4000 3,802,114B	6000x4000 3,987,491B		

steganalysis in this sample of images are presented in Table 11. Although Aletheia provides several services for steganalysis, for this research, the *Calibration attacks to JPEG steganography* [15] was performed on the stego images since we are using JPEG images in a DCT environment. Aletheia also provides other steganalysis attack methods. However, this is based on *Structural LSB detectors* [16], which was considered outside this research's scope. Furthermore, it has a wide range of methods to compare the differences between the stego and the cover images and differences between DCT coefficients or the probability of a cover image to be altered through specific image steganography algorithms, among others.

The *Calibration attacks to JPEG steganography* analyses the different channels in the image and retrieves whether there is hidden data in any of these channels (either 0, 1 or 2); whenever a ✓is placed, it means that hidden data was found in that channel. On the other hand, if there is a ✗, it means that there is no hidden data in the proposed channel; also, as a third option, in those rows where *Error* is placed, steganalysis was inconclusive; thus, no concrete result for any of the channels was obtained.

Table 11. Steganalysis attacks on obtained stego images

Tool	Cover	Secret	Channel 0	Channel 1	Channel 2
M4JPEG		Text 1	✓	✗	✓
		Text 3	✓	✗	✓
M4JPEG		Text 1	✓	✓	✓
		Text 2	✓	✓	✓
M4JPEG		sJpeg 1	✗	✗	✗
		sJpeg 2	✗	✗	✗
Jstego		Text 4	Error	Error	Error
		sJpeg 2	Error	Error	Error
Jstego		Text 2	Error	Error	Error
		Text 3	Error	Error	Error
Jstego		sJpeg 1	Error	Error	Error
		sJpeg 2	Error	Error	Error
F5		Text 3	Error	Error	Error
		sJpeg 3	Error	Error	Error
F5		sJpeg 1	Error	Error	Error
		sJpeg 3	Error	Error	Error
F5		sJpeg 2	Error	Error	Error
		sJpeg 3	Error	Error	Error

4.2 Sub-Question 2 Results

The total number of stego images obtained during the first phase of the research, discussed in the previous subsection, exceeds 200 images. Consequently, only a subset of these stego images was used to study the accuracy of the selected tools. For each of the tools and secret data files (Tables 3 and 4), 6 to 7 cover-stego image pairs in RGB and Grayscale were chosen to be studied, thus, reducing the sample dataset and the computational time to calculate the performance metrics. As described in the methodology section, the accuracy of the DCT image steganography tools was achieved through the Mean Square Error (MSE) and the Peak-Signal-to-Noise-Ratio (PSNR). The procedure to be followed was focused on obtaining the MSE and PSNR values for each of the images hiding specific secret data, for instance, for the first 6 samples (represented in the tables with an S, sample) RGB images with the hidden message being Text 1. Later, the mean of all the obtained results for both metrics was calculated, obtaining an average of the MSE and PSNR values for a specific secret data on a specific tool. These results can be observed in Tables 12, 13 and 14 for each of the DCT tools studied in this research. It is important to note that for some secret data messages in a specific tool, the number of stego images obtained was limited to 2 or 3; therefore, the sample of images could not be augmented to match the other sections with up to 6 sample images. Besides, as mentioned in the previous subsection, M4JPEG only accepts RGB images; hence, no Grayscale stego images were outputted to be used in this research.

Looking back at the results, Table 15 evaluates the different DCT tools, defining which are the accepted cover files extensions, as well as the type of secret messages that can be hidden. There is an evaluation of the MSE and PSNR values of each tool for both Grayscale and RGB images. This evaluation is distributed between H (high) and L (low). The range of what is considered high or low

Table 12. M4JPEG Metrics Table

Secret Data	RGB S	MSE	PSNR
Text1	6	0.01	169.29dB
Text2	6	0.07	157.08dB
Text3	6	0.13	154.78dB
Text4	3	0.54	148.50dB
sJpeg1	2	0.74	146.87dB
sJpeg2	2	0.58	147.85dB
Avg.		0.34	154.06dB

Table 13. Jstego Metrics Table

Secret Data	RGB S	MSE	PSNR	Grayscale S	MSE	PSNR
Text1	3	5.09	138.93dB	3	104.49	27.96dB
Text2	3	4.94	139.01dB	3	104.49	27.96dB
Text3	3	4.97	138.99dB	3	104.48	27.96dB
Text4	3	5.27	138.82dB	3	104.47	27.96dB
sJpeg1	2	8.09	136.36dB	3	104.40	27.96dB
sJpeg2	2	8.09	136.36dB	3	104.42	27.96dB
Avg.		6.08	138.08dB		104.46	27.96dB

Table 14. F5 Metrics Table

Secret Data	RGB S	MSE	PSNR	Grayscale S	MSE	PSNR
Text1	6	34.59	131.49dB	6	13.72	40.17dB
Text2	6	37.17	130.45dB	6	17.0	37.72dB
Text3	6	42.99	130.59dB	6	15.12	39.08dB
Text4	6	40.21	129.47dB	6	18.33	39.24dB
sJpeg1	6	35.14	131.67dB	6	19.02	37.96dB
sJpeg2	6	43.51	130.40dB	6	21.16	37.35dB
sJpeg3	6	37.72	130.16dB	6	18.74	37.27dB
Avg.		52.11	130.60dB		20.23	38.40dB

differs from MSE to PSNR, based on the final average presented in each table (Tables 12 to 14). Moreover, it is essential to mention that while a higher result in PSNR indicates a good quality of the stego image (less distortion), MSE is the other way around. The greater the value, the higher the difference between the cover and the stego image. Therefore, for MSE, we considered any value lower than 10 to be established as High for both RGB and Grayscale. Besides, considering that the range of PSNR for RGB images falls up to 170dB, values higher than 140dB were considered High PSNR [30]. On the other hand, the range flows up to 60dB for Grayscale images; thus, PSNR was considered High for values greater than 35dB [33, 37].

Based on the results obtained from the previous subsection and performance metric results presented in Tables 12 to 14, as well as the evaluation in Table 15, it can be inferred which tool(s) has better accuracy for image steganography. No concrete tool would be perfect for every aspect of image steganography applying DCT.

Table 15. DCT Image Steganography tools details

Tool	Cover type	Secret Data	RGB MSE	RGB PSNR	Grayscale MSE	Grayscale PSNR
M4JPEG	JPEG	TXT, JPEG	L	H	N/A	N/A
Jstego	JPEG	TXT, JPEG	L	H	H	L
F5	JPEG	TXT, JPEG	H	L	H	H

The accurate decision should be considered depending on the image chosen as the cover. As can be observed in Table 12, the performance of executing image steganography using M4JPEG with RGB images is accurate; the levels of MSE are low, especially for small Text files, as can be seen in Text 1 or Text 2. Texts 3 and 4 have a bit higher results but overall, keep it under 1, which offers an almost nonexistent distortion in the stego image. As for the sJpeg secret data, levels go up, almost reaching 1 in the case of sJpeg1; however, again, the result is good enough to establish a low MSE; almost no error between the cover and the stego image. On the same line, PSNR has, on average high values, referring to a low level of distortion. Nevertheless, in both cases, either Jstego or F5, RGB values for MSE are higher than for M4JPEG, especially for F5, where the error rate goes up on average. PSNR is quite distant from the high levels obtained in M4JPEG, too. Thus, the distortion is more significant for both F5 and Jstego. From this discussion, it can be inferred that with RGB images, no matter the hidden data size, M4JPEG is a more accurate decision to perform image steganography.

On the other hand, for Grayscale images, we can only consider Jstego and F5. As mentioned in the previous subsection, M4JPEG only accepts RGB cover images. Comparing both MSE results for F5 and Jstego, the average is more excellent for Jstego; the error rate between cover and stego image is more significant in this case, again, no matter the size of the hidden message. In terms of PSNR, the same reasoning applies; Jstego has lower PSNR values; thus, the level of distortion is more remarkable for the stego images obtained from this DCT tool. These can be observed in Table 8 as the Grayscale stego image is lighter than the cover one; therefore, the error rate is expected to be higher and the PSNR to be lower. The conclusion inferred from such statements is that for Grayscale images, F5 offers better and more accurate results than Jstego.

4.3 Main Research Question Results

The previous subsections have focused on the process of stego images through steganalysis and the accuracy of the different DCT tools when performing image steganography. With these results taken into consideration and gathering all the previous information, this subsection aims to answer the main research question by evaluating the most common characteristics of successful cover images (those that returned a stego image). To proceed, the cover images were divided into 4 groups, depending on their sizes; they can be observed in Tables 16 and 17.

To reduce the number of images and work with a smaller sample, for each group, 2 to 3 randomly selected cover-stego pairs of images were chosen to calculate the average PSNR for each secret data (Tables 3 and 4) and exploited DCT tools in this research. The study per group and tool was performed twice, once with RGB images (24-bit) and another with Grayscale images (8-bit). Tables 16 and 17 present the evaluation per secret data, classifying the information upon the different DCT tools, either with a check mark(✓), a cross(✗) or a check mark-star (✓★). Only averages of high PSNR are considered to place ✓, being the images slightly distortion, best quality. For pairs where the PSNR is low, a ✓★ is placed, indicating that a stego image was obtained, but the level of distortion is noticeable or higher. Finally, a ✗ represents that there were no images to be studied for the chosen secret data for a group in a specific DCT tool, as no stego images were obtained, for instance, the situation with Grayscale images in M4JPEG.

Table 16. Key Image Characteristics per DCT Tools hiding TXT

S.Data size	Cover size (up to)	M4JPEG		Jstego		F5	
		RGB	GS	RGB	GS	RGB	GS
Text1	512x512	✓	✗	✗	✗	✓★	✓
	1024x1024	✓	✗	✗	✗	✓★	✓
24 B	4032x3024	✓	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✓	✓★	✓	✓
Text2	512x512	✓	✗	✗	✗	✓★	✓
	1024x1024	✓	✗	✗	✗	✓★	✓
610B	4032x3024	✓	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✓	✓★	✓	✓
Text3	512x512	✓	✗	✗	✗	✓★	✓★
	1024x1024	✓	✗	✗	✗	✓★	✓
1,075 B	4032x3024	✓	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✓	✓★	✓	✓
Text4	512x512	✗	✗	✗	✗	✓★	✓★
	1024x1024	✗	✗	✗	✗	✓★	✓
10,072 B	4032x3024	✓	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✓	✓★	✓	✓

Considering a successful stego image implies that the distortion between the original image and the outputted one should be as imperceptible as possible, in other words, having a higher PSNR and a lower MSE. Besides, it is crucial to consider an ER where the BBP is high enough to enclose all the bits from the secret message into the cover image, enabling higher performance and being able to carry more secret bits. Tthe perfect match would have been an image with a high PSNR, low MSE and a medium to high ER; nonetheless, this is just a best-case scenario. The findings in this research conclude that a good combination of characteristics to obtain a successful stego image starts with an 8-bit, Grayscale cover image. It can be observed in Tables 16 and 17 that these images perform better than the RGB ones, specifically using F5. The levels of PSNR are higher in most groups with barely noticeable distortion. The performance of RGB (24-bit) images is lower than the 8-bit ones. However, it can be observed that using M4JPEG would provide lower distortion stego images for pictures of sizes up to 4032x3024. RGB images with larger

Table 17. Key Image Characteristics per DCT Tools hiding JPEG

S.Data size	Cover size (up to)	M4JPEG		Jstego		F5	
		RGB	GS	RGB	GS	RGB	GS
sJpeg1	512x512	✗	✗	✗	✗	✓★	✓★
	1024x1024	✗	✗	✗	✗	✓★	✓
27,058 B	4032x3024	✓	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✓	✓★	✓	✓
sJpeg2	512x512	✗	✗	✗	✗	✓★	✓★
	1024x1024	✗	✗	✗	✗	✓★	✓★
20,679 B	4032x3024	✓	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✓	✓★	✓	✓
sJpeg3	512x512	✗	✗	✗	✗	✓★	✓★
	1024x1024	✗	✗	✗	✗	✓★	✓★
273,090 B	4032x3024	✗	✗	✗	✗	✓★	✓
	6000x4000	✗	✗	✗	✗	✓★	✓

sizes, for example, 6000x4000, should be treated with Jstego or F5, both having a high PSNR when calculated for this type of image. Nonetheless, if choosing a big file as secret data, as can be sJpeg3, F5 would be the only acceptable possibility, although the PSNR lowers down and provokes a distortion of the image. With the support of the previous arguments, the combination of characteristics that would offer a great image to perform image steganography using a DCT tool would start with a Grayscale, 8-bit cover image. Besides, the size should be more significant than 1024x1024 pixels and an ER value high enough to embed large files as sJpeg3, which is 273KB.

5 CONCLUSIONS AND FUTURE WORK

This study aimed to analyse the different characteristics that make an image potentially attractive to be used in image steganography. Selected DCT image steganography tools, namely M4JPEG, Jstego and F5, were used to perform image steganography by hiding secret data of distinct characteristics inside a dataset of 21 images. Overall, the results discussed in Section 4 show that specific characteristics make an image attractive to be used as a cover image, obtaining a successful stego image. Additionally, using performance metrics such as PSNR or MSE, it was proven that images with specific characteristics, such as Grayscale (8-bit), perform better as they have a lower distortion rate.

Even though the results of this research have shaped a new light into those images that can be used successfully to perform image steganography using DCT tools/algorithms, this could be further expanded. First, a broader spectrum of secret messages could be used, increasing the byte size of new text or image files and using other types of files to be hidden, such as video or audio, for instance. Moreover, the range of DCT tools used could be increased with new algorithms that could enlarge the sample of results, especially in Grayscale images, as M4JPEG could not recognise them. Future work on this study should focus on the broader range of secret files to be used, as well as the size, plus adding more DCT tools that could amplify the sample of results for each image.

REFERENCES

[1] Canon 2017. *Canon EOS 200D*. Canon. https://www.canon.nl/cameras/eos-200d/

[2] GitHub 2018. *M4JPEG Project*. GitHub. https://digitnet.github.io/m4jpeg/m4jpeg-tool/m4jpeg-download.htm

[3] Iowa State University 2022. *Campus Stock Images*. Iowa State University. https://iowastate.photoshelter.com/galleries/C0000rWLI9E_oKgc/G0000J2URC8AfrrQ/Campus-Stock-Images

[4] USC Viterbi School of Engineering n.d.. *The USC-SIPI Image Database*. USC Viterbi School of Engineering. https://sipi.usc.edu/database/

[5] D. Artz. 2001. Digital steganography: hiding data within data. *IEEE Internet Computing* 5, 3 (2001), 75–80. https://doi.org/10.1109/4236.935180

[6] Mayra Bachrach and Frank Y. Shih. 2011. Image steganography and steganalysis. *Wiley Interdisciplinary Reviews: Computational Statistics* 3, 3 (May 2011), 251–259. https://doi.org/10.1002/wics.152

[7] Lucke Champine. 2019. *jsteg*. GitHub. https://github.com/lukechampine/jsteg

[8] Abbas Cheddad, Joan Condell, Kevin Curran, and Paul Mc Kevitt. 2010. Digital image steganography: Survey and analysis of current methods. *Signal Processing* 90, 3 (2010), 727–752. https://doi.org/10.1016/j.sigpro.2009.08.010

[9] daniellerch. 2019. *F5*. GitHub. https://github.com/daniellerch/stego-collection/tree/master/F5

[10] daniellerch. 2021-11-18. *Aletheia*. GitHub. https://github.com/daniellerch/aletheia

[11] A.M. Fard, M.-R. Akbarzadeh-T, and F. Varasteh-A. 2006. A New Genetic Algorithm Approach for Secure JPEG Steganography. *IEEE International Conference on Engineering of Intelligent Systems* (2006). https://doi.org/10.1109/ICEIS.2006.1703168

[12] Yashika Garg and Amneet Kaur. 2017. A Case study on Steganography and its Attacks. *International Journal of Engineering Trends and Technology* 47 (05 2017), 453–457. https://doi.org/10.14445/22315381/IJETT-V47P274

[13] Stuti Goel, Arun Kumar, and Manpreet Kaur. 2013. A DCT-Based Robust Methodology for Image Steganography. *www.ijcst.com* 4 (07 2013).

[14] Nagham Hamid, Abid Yahya, R.Badlishah Ahmad, and Osamah Al-qershi. 2012. Image Steganography Techniques: An Overview. *International Journal of Computer Science and Security* 6 (06 2012), 168–187.

[15] Daniel Lerch Hostalot. n.d.. *Introduction to steganalysis with Aletheia*. https://daniellerch.me/stego/aletheia/intro-en/#calibration-attacks

[16] Daniel Lerch Hostalot. n.d.. *Introduction to steganalysis with Aletheia*. https://daniellerch.me/stego/aletheia/intro-en/#structural-attacks

[17] T Hunter. 2021. *Steganography: The Undetectable Cybersecurity Threat*. Built-In. https://builtin.com/cybersecurity/steganography

[18] J-br01. 2022. *image-steganography-dataset*. https://github.com/J-br01/image-steganography-dataset

[19] Mao Jia-Fa, Niu Xin-Xin, Xiao Gang, Sheng Wei-Guo, and Zhang Na-Na. 2015. A steganalysis method in the DCT domain. *Multimedia Tools and Applications* 75 (06 2015). https://doi.org/10.1007/s11042-015-2708-0

[20] J Judge. 2001. Steganography: Past, Present, Future. *SANS Institute* (12 2001). https://doi.org/10.2172/15006450

[21] Narjes Khalifa and Mohamed Benrejeb. 2019. Chapter 16 - On Nonidentical Discrete-Time Hyperchaotic Systems Synchronization: Towards Secure Medical Image Transmission. In *Recent Advances in Chaotic Systems and Synchronization*, Olfa Boubaker and Sajad Jafari (Eds.). Academic Press, 329–349. https://doi.org/10.1016/B978-0-12-815838-8.00016-9

[22] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. n.d.. *Stanford Dogs Dataset*. Standford University. http://vision.stanford.edu/aditya86/ImageNetDogs/

[23] Prof. V. Krishna and Dr. C K Kumbharana. 2021. An Analysis of Different Image Formats for Steganography. *International Journal of Emerging technologies and innovate Research* 8, 10 (2021), 299–307. http://www.jetir.org/papers/JETIR2110040.pdf

[24] Xuelong Li. 2003. Watermarking in secure image retrieval. *Pattern Recognition Letters* 24, 14 (2003), 2431–2434. https://doi.org/10.1016/S0167-8655(03)00072-2

[25] Der-Chyuan Lou and Chia-Hung Sung. 2004. A steganographic scheme for secure communications based on the chaos and euler Theorem. *IEEE Transactions on Multimedia* 6, 3 (2004), 501–509. https://doi.org/10.1109/TMM.2004.827493

[26] Sai Ma, Qingxiao Guan, Xianfeng Zhao, and Yaqi Liu. 2018. Adaptive Spatial Steganography Based on Probability-Controlled Adversarial Examples. *Multimedia Tools and Applications* 78, 22 (04 2018), 32503–32522.

[27] Tayana Morkel, Jan Eloff, and Martin Olivier. 2005. An overview of image steganography. . Proceedings of the Fifth Annual Information Security South Africa Conference (ISSA2005)), Sandton, South Africa, 1–11.

[28] V. Nagaraj, V. Vijayalakshmi, and G. Zayaraz. 2013. Color Image Steganography based on Pixel Value Modification Method Using Modulus Function. *IERI Procedia* 4 (2013), 17–24. https://doi.org/10.1016/j.ieri.2013.11.004 2013 International Conference on Electronic Engineering and Computer Science (EECS 2013).

[29] Walker Rowe. 2018. *Mean Square Error & R2 Score Clearly Explained*. https://www.bmc.com/blogs/mean-squared-error-r2-and-variance-in-regression-analysis/

[30] U. Sara, M. Akter, and M. Uddin. 2019. Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study. *Journal of Computer and Communications* 7 (2019), 8–18. https://doi.org/10.4236/jcc.2019.73002

[31] Dipti Kapoor Sarmah and Anand J. Kulkarni. 2018. JPEG based steganography methods using Cohort Intelligence with Cognitive Computing and modified Multi Random Start Local Search optimization algorithms. *Information Sciences* 430-431 (2018), 378–396. https://doi.org/10.1016/j.ins.2017.11.027

[32] Dipti Kapoor Sarmah and Anand J Kulkarni. 2019. Improved Cohort Intelligence—A high capacity, swift and secure approach on JPEG image steganography. *Journal of Information Security and Applications* 45 (2019), 90–106. https://doi.org/10.1016/j.jisa.2019.01.002

[33] D.I.M. Setiadi. 2021. PSNR vs SSIM: imperceptibility quality assessment for image steganography. *Multimed Tools Appl* 80 (2021), 8423–8444. https://doi.org/10.1007/s11042-020-10035-z

[34] Shahinur. 2019. *what is PSNR*. https://shahinur.com/en/what-is-psnr/

[35] signallock. 2013. *jstego*. SourceForge. https://sourceforge.net/projects/jstego/

[36] Jayson E. Street, Brian Baskin, and Kristin Sims. 2015. 3 - Security Threats Are Real (STAR). In *Dissecting the Hack*, Jayson E. Street, Brian Baskin, and Kristin Sims (Eds.). Syngress, Boston, 111–211. https://doi.org/10.1016/B978-0-12-804278-6.00003-3

[37] M.S. Subhedar and V.H Mankar. 1865–1886. Secure image steganography using framelet transform and bidiagonal SVD. *Multimed Tools Appl* 79 (1865–1886). https://doi.org/10.1007/s11042-019-08221-9

[38] Nandhini Subramanian, Omar Elharrouss, Somaya Al-Maadeed, and Ahmed Bouridane. 2021. Image Steganography: A Review of the Recent Advances. *IEEE Access* 9 (2021), 23409–23423. https://doi.org/10.1109/ACCESS.2021.3053998

[39] triinuerik. 2019. *Stegote: STEGANOGRAPHY TOOL FOR HIDING INFORMATION IN JPEG AND PNG IMAGES*. GitHub. https://github.com/triinuerik/stegote

[40] Yanping Zhang, Juan Jiang, Yongliang Zha, Heng Zhang, and Shu Zhao. 2013. Research on Embedding Capacity and Efficiency of Information Hiding Based on Digital Images. *International Journal of Intelligence Science* 03 (01 2013), 77–85. https://doi.org/10.4236/ijis.2013.32009