University of Zagreb

Faculty of Graphic Arts

Petar Branislav Jelušić

ROBUST IMAGE STEGANOGRAPHY METHOD SUITED FOR PRINTING

DOCTORAL THESIS

Mentor:

Associate Professor Ante Poljičak, PhD

Zagreb, 2023.

Sveučilište u Zagrebu

Grafički fakultet

Petar Branislav Jelušić

ROBUSTNA STEGANOGRAFSKA METODA PRILAGOĐENA PROCESU TISKA

DOKTORSKI RAD

Mentor:

izv. prof. dr. sc. Ante Poljičak

Zagreb, 2023.

UDK 655:655.3:621.798

Imenovano Povjerenstvo za ocjenu doktorskoga rada:

1. prof. dr. sc. Lidija Mandić, Sveučilište u Zagrebu Grafički fakultet, predsjednica
2. izv. prof. dr. sc. Davor Donevski, Sveučilište u Zagrebu Grafički fakultet, član
3. prof. dr. sc. Sonja Grgić, Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva, vanjska članica

Imenovano Povjerenstvo za obranu doktorskoga rada:

1. prof. dr. sc. Lidija Mandić, Sveučilište u Zagrebu Grafički fakultet, predsjednica
2. izv. prof. dr. sc. Davor Donevski, Sveučilište u Zagrebu Grafički fakultet, član
3. prof. dr. sc. Sonja Grgić, Sveučilište u Zagrebu Fakultet elektrotehnike i računarstva, vanjska članica
4. izv. prof. dr. sc. Tomislav Cigula, Sveučilište u Zagrebu Grafički fakultet, zamjenski član
5. izv. prof. dr. sc. Aleš Hladnik, Sveučilište u Ljubljani Prirodoslovno-tehnički fakultet, Slovenija, zamjenski vanjski član

Mentor:

izv. prof. dr. sc. Ante Poljičak, RIT Croatia, Hrvatska

Datum obrane doktorskoga rada: 30. kolovoza 2023.

Mjesto obrane doktorskoga rada: Sveučilište u Zagrebu Grafički fakultet

Povjerenstvo za obranu doktorskoga rada donijelo je sljedeću odluku:

„Obranio s ocjenom summa cum laude (*s najvećom pohvalom*) jednoglasnom odlukom Povjerenstva“

Information about the mentor:

Associate Professor Ante Poljičak is an instructor at RIT Croatia. He teaches multiple courses within *New Media Design* and *Web and Mobile Computing* programs. At RIT Croatia, he is also the area head of the *New Media Design* program. His expertise includes image processing and digital watermarking.

Hvala svima koji su bili uz mene tijekom protekle četiri godine.
Hvala vam što ste vjerovali u mene čak i kada ja nisam.

ABSTRACT

In this thesis, a robust image steganography method suited for printing is developed. The method aims to protect packaging products against counterfeiting without compromising the product's visual design. The motivation for the research lies in the fact that there exists a need for more print domain-based research in the relevant state-of-the-art literature.

The steganography method presented in this thesis does not require any additional processes or materials during reproduction. Thus, the method can seamlessly be integrated into a reproduction process. The presented method successfully combines high capacity, robustness to the *print-scan* reproduction channel, and high imperceptibility. As no such techniques are found in the relevant literature, the method developed in this thesis presents a significant scientific contribution.

The presented image steganography method is tested for robustness against image processing throughout the reproduction channel and for image quality. Experiments are conducted using 1000 images in CMYK color space. Additionally, a subjective survey is conducted to test the method for image quality. A total of 36 observers participated in the survey, and a dataset of 144 images is used. The results show that the method can sustain the reproduction channel image processing, proving that the method is indeed suited for the reproduction process. Even after embedding information using strong signals, image quality stays consistently high.

Keywords: steganography, data hiding, counterfeiting, packaging, *print-scan,* Discrete Fourier Transform, Grey Component Replacement

PROŠIRENI SAŽETAK

U ovoj doktorskoj dizertaciji prezentirana je robustna steganografska metoda razvijena i prilagođena za tisak. Osnovni cilj metode je pružanje zaštite od krivotvorenja ambalaže. Zaštita ambalaže postiže se umetanjem više bitova informacije u sliku pri enkoderu, a potom maskiranjem informacije kako bi ona bila nevidljiva ljudskom oku. Informacija se pri dekoderu detektira pomoću infracrvene kamere. Preliminarna istraživanja pokazala su da u relevantnoj literaturi nedostaje metoda razvijenih za domenu tiska. Razlog za takav nedostatak jest činjenica da razvijanje steganografskih metoda za tisak zahtjeva veću količinu resursa i materijala, u odnosu na razvijanje sličnih domena za digitalnu domenu. Također, metode za tisak često zahtijevaju višu razinu kompleksnosti, budući da se tijekom reprodukcije pojavljuju razni oblici procesiranja koji mogu kompromitirati informaciju u slici [1]. Da bi se sačuvala skrivena informacija, metoda mora biti otporna na procesiranje koje se događa tijekom reprodukcije.

Kako bi se postigla visoka razina otpornosti, informacija se može umetnuti unutar frekvencijske domene slike [2], [3]. Frekvencijskoj domeni slike možemo pristupiti pomoću matematičkih transformacija. Najčešće se koriste diskretna kosinusna transformacija (DCT), diskretna wavelet transformacija (DWT) i diskretna Fourierova transformacija (DFT) [2], [4]. Korištenje svake od navedenih transformacija ima određene prednosti i nedostatke, ovisno o kontekstu razvijanja metode [5]. Za metode prilagođene procesu tiska, diskretna Fourierova transformacija je optimalan odabir, budući da metode bazirane na DFT-u pružaju otpornost na geometrijske transformacije koje se događaju tijekom reprodukcije [5], [6].

U ovom istraživanju korištene su slike u $cmyk$ prostoru boja. Svaka slika najprije je podijeljena u blokove, a umetanje informacije vrši se za svaki blok pojedinačno. Pomoću DFT-a, k kanal slikovnog bloka se transformira u frekvencijsku domenu, gdje se vrši umetanje informacije. Akromatska zamjena koristi se za maskiranje vidljivih artefakata nastalih prilikom umetanja informacije. Primjeri uspješnog korištenja akromatske zamjene za maskiranje artefakata mogu se pronaći u [7] i [8]. Nakon umetanja informacije u svaki slikovni blok, blokovi se ponovno spajaju u jednu, jedinstvenu sliku. Akromatska zamjena tada mijenja vrijednosti c, m i y kanala slike, dok kanal k, u kojemu se nalazi umetnuta informacija, ostaje nepromijenjen. Time nakon maskiranja akromatskom zamjenom označena slika posjeduje ista vizualna svojstva kao i slika prije označavanja.

U eksperimentalnom dijelu rada koristi se 1000 slika u *cmyk* prostoru boja. U digitalnom okruženju provedeno je istraživanje otpornosti metode na slikovne napade specifične za reprodukcijski proces - skaliranje, *blur*, šum, rotaciju i kompresiju. Također, provedeno je istraživanje otpornosti metode na reprodukcijski proces, koristeći tiskane uzorke. Objektivna metrika *bit error rate* (BER) korištena je za evaluaciju. Mogućnost optimizacije metode testirala se procesiranjem slike *(unsharp filter)* i korištenjem *error correction kodova* (ECC). Provedeno je istraživanje kvalitete slike nakon umetanja informacije. Za evaluaciju su korištene objektivne metrike *peak signal to noise ratio* (PSNR) i *structural similarity index measure* (SSIM). PSNR i SSIM su tzv. *full-reference* metrike. Drugim riječima, potrebne su i neoznačena i označena slika istovremeno, kako bi se mogla utvrditi razina sličnosti između slika [9], [10]. Subjektivna analiza provedena je na 36 ispitanika, koristeći ukupno 144 uzorka slika. Ispitanici su ocijenjivali vidljivost artefakata na skali od nula (nevidljivo) do tri (vrlo vidljivo).

Rezultati pokazuju da metoda posjeduje visoku razinu otpornosti na reprodukcijski proces. Također, metoda se uistinu optimizirala korištenjem *unsharp* filtera i ECC-a. Kvaliteta slike ostaje visoka bez obzira na umetanje informacije, što su potvrdili rezultati eksperimenata s objektivnim metrikama i subjektivna analiza.

Ključne riječi: steganografija, skrivanje informacije, krivotvorenje, ambalaža, tisak, Fourierova transformacija, akromatska zamjena

TABLE OF CONTENTS

1 INTRODUCTION

1.1 Background

The technological advances during the last couple of decades changed the way we perceive, use, and share information. The amount of digital information continues to grow, as sharing media became an everyday occurrence. As the amount of this shared information is still growing, so is the need to regulate the sharing process itself. Therefore, data security, cybers-security, etc. are some of the most important aspects of the digital era.

Apart from sharing media, the technological advances also made possible the development of more advanced counterfeiting methods. Counterfeiting is a process that involves substitution, duplication, or direct copying of packaging and products [11]. It is a concern for both consumers and enterprises, as counterfeit goods can be found in various domains - food, beverages, textiles, medicines, pharmaceuticals, etc. [12], [13]. Counterfeiting presents numerous threats to consumers, such as tax losses, brand damages, and most importantly, health risks. To fight those threats, technologies such as RFID and 2D barcodes have been developed for protection against counterfeiting.

Fathi et al. examine RFID as a potential protection technology. They conclude with stating that high costs are the major setback when trying to implement RFID to commercial packaging. Furthermore, they also state that the technology is only able to be printed on certain materials, some of which are not the usual choice in standard packaging processes [14]. Zhou et. al. examine a low-cost integration of a RFID chip. They conclude by saying that a low-cost implementation is possible, but only within a system with specific packaging requirements [15]. Trenfield et. al. use a combination of a 3D printer and a 2D inkjet printer to fabricate a track-and-trace measure in a single step process. Both printing and detection yield positive results. However, the authors conclude by stating that the technology impacts the visual appearance of packaging, which could affect the acceptance of the final product [16].

From the examples above, it can be concluded that only some of the requirements were met by each proposed technology. A technique that meets all requirements - high-level security, low-cost implementation, while maintaining the same visual appearance, is digital data hiding. During the last decades, significant progress has been made in that field. Depending on the application, these techniques can be used for both the digital and the print domain.

Data hiding methods - primarily watermarking and steganography, are prominent topics in science, with a large number of published papers. However, this large number of published papers does not apply to the print domain, where relevant research is limited. This was the main motivation for developing a data hiding method suited for printing, which is presented in this thesis.

The lack of research in the print domain is not a result of lesser demand. Printed products need to be protected against counterfeiting just the same as digital products. Therefore, developing a method suited for printing would benefit the current *status quo*. Modern printing systems are, at least in certain parts of the process, digitalized, so it is possible to alter printed goods digitally. When developing such a method for securing printed goods against counterfeiting, the pre-condition is that the method is robust to the printing process. The print-scan communication channel is an aggressive compound image attack; the image is subject to changes in contrast, color space, sharpness and registration. Hence, the method should be robust to all image attacks within the communication channel - rotation, scaling, compression, blur and noise.

First mention of data hiding, in the context of the digital era, can be traced back to 1984, when Gustavus J. Simmons discussed information hiding in his work titled *"The Prisoner's Problem and the Subliminal Channel"* [17]. In 1988, when Komatsu and Tominaga first used the term digital watermarking in their work titled *"Authentication system using concealed image in telematics"* [18]. Since then, considerable progress has been made in the field of digital data hiding, focusing on sharing information and media in the digital domain. However, progress in the print domain has been less evident.

Since the beginning of the digital era of data hiding, watermarking has become a most-often used data hiding technique for security applications and ownership identification [19]. In 1992, Tirkel et. al. used the term *"electronic water mark"*, which marked the first time that the term watermarking was used in the digital era [20]. Another important data hiding technique is steganography. Both watermarking and steganography are techniques used to hide information in media, and both possess the same properties – capacity, robustness, and imperceptibility.

However, steganography techniques usually prioritize imperceptibility, while watermarking techniques emphasize robustness. More importantly, the main goals of steganography and

watermarking are different – steganography aims to transfer messages between peers, while watermarking has a goal to provide protection of the media [21].

Data hiding methods are often defined by the domain in which the data embedding occurs. Embedding can be performed either in the spatial domain or in the frequency domain [22]. Frequency domain methods mostly use Discrete Fourier Transform (DFT), Discrete Cosine Transform (DCT), or Discrete Wavelet Transform (DWT) for embedding [4], [5], [22]. There also exist methods that combine several embedding techniques. Ling et al. combine spatial domain embedding and DCT to minimize possible false-positive detection [23]. Singh et al. combine DCT and DWT. With using DCT for embedding, and DWT for detection, they conclude that the hybrid approach helps maximize advantages of both transforms [24]. Su et al. use spatial domain embedding in their method, but also use the DC coefficient of the Fourier transform to achieve the embedding. They conclude that this addition of using DC for embedding in the spatial domain improves performance [25].

Data hiding methods can also be defined by the data available to the detector (blind and informed detection). Finally, digital watermarking methods can be defined by the visibility of the embedded data (visible and invisible) [5].

The properties of data hiding methods can be divided into two categories - embedding properties (embedding efficiency, quality, capacity) and detection properties (robustness, probability of false detection) [5]. Each method aims to find the right balance between three interdependent components - invisibility, robustness, and capacity [26], [27]. Given that all three components mentioned above are equally crucial, tweaking the balance between them is one of the concerns of every data hiding method. This balance between data hiding properties will be further addressed in Section 2.6.

Digitally developed data hiding methods can be used outside of the digital domain. Some methods are especially suited for print-scan and print-cam processes [5], [27]–[29]. These processes can severely degrade the quality of a stego image, thus impacting the hidden data in an image, so the robustness of the data hiding method becomes a crucial factor. However, higher robustness results in higher visibility of the hidden data. In order to mask visible artifacts, different approaches are used. Cai-yin et al. use visual masking in regards to the human visual system (HVS). The authors state that the HVS is less sensitive to noise around high-activity image areas, and noise in extremely light or dark image areas. A spatial visual mask is applied to reshape the embedded watermark energy, thus improving image quality.

[30]. Similarly, Bender et al. modify luminance and chrominance of edge areas in images, which are also less likely to be detected by the observer [31]. Poljičak et al. use Gray Component Replacement (GCR) [28]. GCR allows the digital watermark to be detectable in the infra-red part of the spectrum, while it remains almost entirely imperceptible in the visible spectrum. Thanks to its circular shape, the watermark embedded in the frequency domain using the Fourier transform proves to be the optimum choice for geometric attacks (rotation, translation, scaling) and for both print-scan and print-cam processes [5], [28].

Cai-yin et al. use both DCT and DFT in their work. They embed the hidden data in both yellow and black channels of a CMYK image [30]. Upon embedding, they use two masking techniques to improve the quality of the marked image. The results lead to a conclusion that both masks positively affect the marked image quality. They state that their method has poor performance when it comes to geometric attacks, low-resolution scanning, and dark-tone areas of images.

Image quality is a property that is percieved by the observer. Hence, it can be quantified by subjective assessments. However, apart from subjective visual comparison, the quality of images can be measured using objective metrics. Objective metrics offer faster results, which can be easily compared to those achieved by other methods. Peak Signal to Noise Ratio (PSNR) is used in most state-of-the-art works [5], [26], [28], [32]–[38]. Despite its widespread use, PSNR has its shortcomings. Urvoy et al. note that PSNR values can lead to wrong conclusions [6]. Nevertheless, they concede that PSNR is still the most commonly used quality metric. Pramila et al. suggest that PSNR lacks Human Visual System (HSV) features. Instead of the PSNR, they use Structural Similarity Measure (SSIM), a metric also used in other researches [26], [28], [29].

The goal of this thesis is to develop a data hiding method for protecting packaging against counterfeiting. The protection is achieved by embedding data in images. The robustness of hidden data is tested against image attacks that occur within a standard packaging production workflow.

The thesis is organized as follows. An overview of data hiding is given in Section 2. The communication channel is analysed, and its effects in the context of this thesis are given in Section 3. The theoretical backgrounds of the Fourier transform and Gray component replacement are given in Section 4. and Section 5. Section 6. explains the assessment of image quality. The overview of the Grubbs' test for outliers is given in Section 7., and image

features and image activity are overviewed in Section 8. An overview of Error correcting codes is given in Section 9. The presented method and the experimental part of the thesis are explained in Section 10., and the results of all experiments are presented in Section 11. Finally, the conclusion is given in Section 12.

1.2 Hypotheses, methodology, and scientific contributions

The goal of this thesis is to develop a robust steganography method suited for printing. To maximize the method's performance, multiple parameters are examined in-depth. The method's robustness to the print-scan process is the highest priority. At the same time, the method's performance can be measured by the imperceptibility of embedding artifacts. The method's baseline performance can possibly be enhanced using image processing methods and error correction codes (ECC). With this in mind, the hypotheses are defined as follows:

- H1 - Hidden data remains detectable after print-scan
- H2 - Detection improves after image processing
- H3 - Error correction codes improve decoder performance
- H4 - Gray component replacement decreases artifact visibility
- H5 - Gray component replacement does not affect detection

The scientific contribution is evident in the following:

- Robust steganography method suited for the print-scan communication channel based on the Discrete Fourier Transform with Gray Component Replacement masking
- The determination of the frequency range used for data embedding for the balance between image quality and the detection rate
- The framework for evaluation of image attacks' effect on the detection of data hidden using a steganography method

A series of experiments is conducted, as shown in Figure 1 and Figure 2. Experiments are done in both digital and print domains, using objective metrics. A separate subjective experiment is conducted in the print domain. In conjunction, the series of experiments successfully proved all hypotheses, and achieved all scientific contributions as planned.

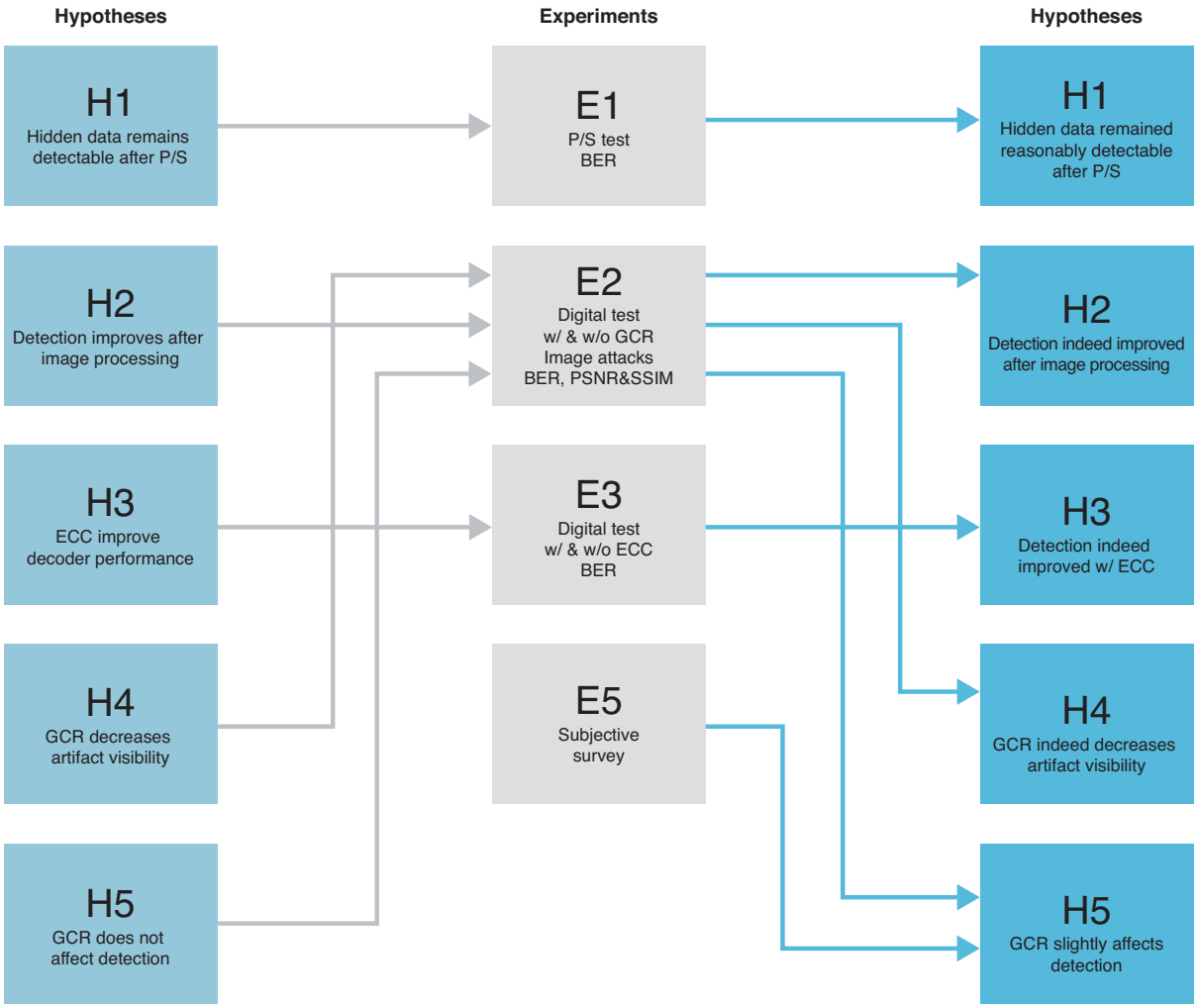

Figure 1. Preliminary hypotheses, experiments, and achieved hypotheses with their respective links.

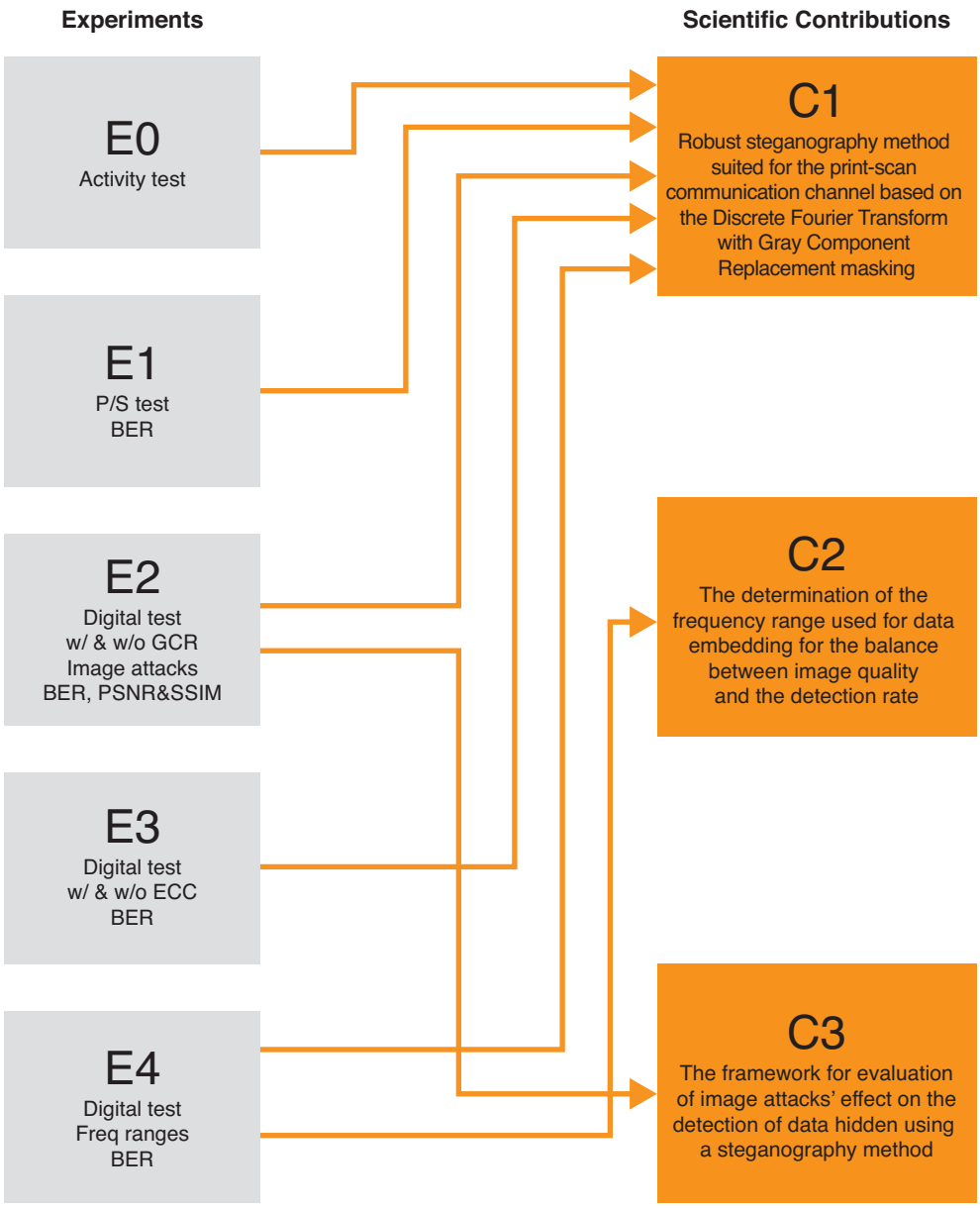

Figure 2. Experiments and scientific contributions with their respective links.

2 DATA HIDING

2.1 History

According to Sahu and Sahu, the term steganography was coined in 1499 by Johannes Trithemius [39]. Trithemius's book titled *"Steganographia"* was published over a hundread years later, in 1606 [40]. Until 1998, it was considered that the book was thematically closely related to magic and using spirits to communicate over vast distances. However, Jim Reeds "solved" the ciphers in Trithemius's *"Steganographia"* and revealed that the book was never about magic. The decryption key made it clear that the book was meant to be deciphered from the start [41]. Reeds's publication proved that *"Steganographia"* can be considered as the first book-length treatment of cryptography in Europe. First practices of steganography can be traced back to 7th century Greece, where Demeratus used wax-layered wooden tablets to hide scratched messages that could be safely transported and read by the receiver [21]. The word steganography itself is made of two Greek words: *Stegos* and *grafia*. Together, the words combined give a meaning of *"cover writing"*, or *"covered writing"* [21], [34], [39].

In the 16th century, an Italian mathematician and physicist, Jerome Cardano, used an ancient Chinese technique to send secret messages. He would use a paper with grid holes as a mask to uncover only a part of the original text, which would in turn make the secret message readable. The mask had to be shared between both the sender and the receiver, for the secret message to be understandable [21], [34]. Throughout history, similar techniques for sending secret messages were used during war times. German soldiers during World War I and World War II, Saudi kings, and Turkish soldiers are just some historic examples of using people to transfer secret messages. [21], [42].

The first mention of the digital era of steganography can be traced to 1984., when Gustavus J. Simmons discussed information hiding in his work titled *The Prisoner's Problem and the Subliminal Channel* [17]. Eight years later, in 1992., Kurak and McHugh raised concern about the possibility of embedding information into digital images in their work *A Cautionary Note On Image Downgrading* [43]. These and similar articles and research helped create an awareness of possible misuse of digital images, and media in general. The term that best describes that awareness and concern is cyber-security [44]. As computer power, the internet, and digital signal processing advanced rapidly by the end of the 20th century, the sharing of digital media became an everyday occurrence. Such a development resulted in a significant rise of cyber-crime, which benefited directly from the advances in technology [2].

A need for copyrighting and ownership protection of digital media was thus necessary. Above all else, this is where digital steganography finds its application today.

The main goal of steganography can be defined in several ways. Kadhim et. al. describe the basic idea of steganography as "*to conceal secret or private information inside a cover media in an undetectable manner"* [21]. In his lecture in 1996., D. Khan explains that *"steganography deals with the hiding of messages so that potential monitors do not even know that a message is being sent"* [42]. Apart from image steganography, which has the most widespread application, steganography is used to hide data in various other media, such as text, codes, audio, video, and DNA [21], [44].

Watermarking is a technique that has evolved from steganography [45]. Throughout history, watermarks were used to establish authenticity of paper manufacturers or certify the composition of paper [46], In 19. century, watermarking is used to protect paper banknotes [5]. Similar to steganography, watermarking also gained a widespread application with the development of technology since 1980s. Since then, watermarking has become a most-often used technique for security applications [19]. In 1992, Tirkel et al. used the term *"electronic water mark"* in their published work [20]. This usage marked the first time that the term *"watermarking"* was used in the digital era. Today, digital watermarking techniques have become more nuanced, with researchers investigating the possibility of application in specific fields. Giakoumaki et al. use watermarking to enhance data security in telemedicine [47]. Khamlichi et al. use image watermarking as an authentication system for medical imaging [48].

In its beginnings, cryptography was concerned with converting messages into an unreadable format to protect the content of the message. Cryptography's goal stays the same even today [49]. It is hard to track the first examples of cryptography. In his 1997 paper, Davies states that the first concrete historical evidence of cryptography usage date from the Venitian Republic during the Renaissance [50]. Being a powerful state, Venice needed secure communication, so even a cryptography school existed, even though the texbooks are lost. Davies concludes that the rapid development of information processing and computing power poses a difficult problem for cryptography [50]. According to Kumar and Pooja, confidentiality and authentication are two of the most important properties of cryptographic systems [51].

Gonzales and Woods, in their 2008. edition of the book *„Digital Image Processing“*, define image processing simply as *„processing digital images by means of a digital computer“* [52]. The first example of a digital image dates back to the 1920s, when the *Bartlane cable picture transmission system* was put in use. The *Bartlane method* was first set up between New York and London. The method reduced the time needed for an image to travel across the Atlantic from over a week, to about three hours [53]. Specialized equipment was used at the receiving end, and the final image was reproduced on a telegraph printer, simulating a halftone pattern. Significant improvements were made in this field in years to come, but a big breakthrough could not happen until the invention of a digital computer. The main reason for the need of a digital computer is the fact that digital image processing requires data storage, display, and transmission.

The advances in developing a digital computer started to transpire during the 1940s and 1950s. An outline of these advances is noted below:

1. The invention of the transistor, Bell Laboratories, 1948.
2. The development of programming languages COBOL (Common Business-Oriented Language) and FORTRAN (Formula Translator), 1950s and 1960s
3. The invention of the integrated circuit, Texas Instruments, 1958.
4. The development of operating systems, early 1960s
5. The development of the microprocessor, Intel, early 1970s
6. The introduction of a personal computer, IBM, 1981.
7. The miniaturization of components, since the late 1970s

These developments, along with advances in mass storage and display systems, were fundamental for the development of digital image processing [52].

There are multiple purposes of image processing [54]:

- To subjectively improve the quality of an image (image enhancement)
- To use as few bits as possible to represent the image, with minimum deterioration in its quality (image compression)
- To improve an image in an objective way (image restoration)
- To make explicit certain characteristics of the image (feature extraction)

Image processing is performed using image transformations, which are performed using operators. An operator takes the input image, and produces an output image, based on the transformation in hand.

2.2 Data hiding applications

Cryptography, steganography, and watermarking are the three main data hiding techniques [2], [21], [55]. Even though they share many common properties, each technique has its distinct priorities and application cases. The main goal of cryptography and steganography is the same – to transfer a secret message between peers. Their approaches to hiding that message, however, are different. Steganography does not change the original data and information, while cryptography converts the original data into a form not readable to the receiver [21], [56]. Another difference between the two techniques is in regard to "*breaking the system"*. The steganography system is broken when the third party gets information about the secret data, while the cryptography system is broken when the third party gets information about the original data. When it comes to data security, watermarking is the most-often used term. Both watermarking and steganography are techniques used to hide information in media, and both possess the same properties – capacity, robustness, and imperceptibility. However, steganography techniques usually value imperceptibility over robustness, while it is the other way around with watermarking. Also, the main goals of steganography and watermarking are different – steganography aims to transfer messages between peers, while watermarking has a goal to provide protection of the media [21]. The similarities and differences between the three techniques are provided in Table 1.

Table 1. Goals, priorities and system validations of main data hiding techniques. [21], [39], [57]

Characteristic	Steganography	Watermarking	Cryptography
Goal/ Objective	Preserve data from detection; Conceal the existence of communication	Copyright protection of cover media	Content protection
Priorities	High imperceptibility, robustness, security, capacity	Robustness	Robustness, complexity of encryption
System is invalid if	Detected	Removed/ replaced	De-ciphered

2.3 Data hiding model

Data hiding methods are often modeled as communication channels in which the information is being transferred from the encoder to the decoder [5]. Image steganography follows a similar principle. The first part of the steganographic process is the embedding technique (encoder), where a secret information is hidden inside the cover image, often using a stego-key [57]. At the other end of the communication channel, the extraction technique (decoder) is used to retrieve the hidden information from the stego image. If a stego-key is used at the encoder, the decoder needs to have a priori knowledge of that stego-key, in order to extract the hidden information [44], [57]. An illustration of a steganographic model can be seen in Figure 3.

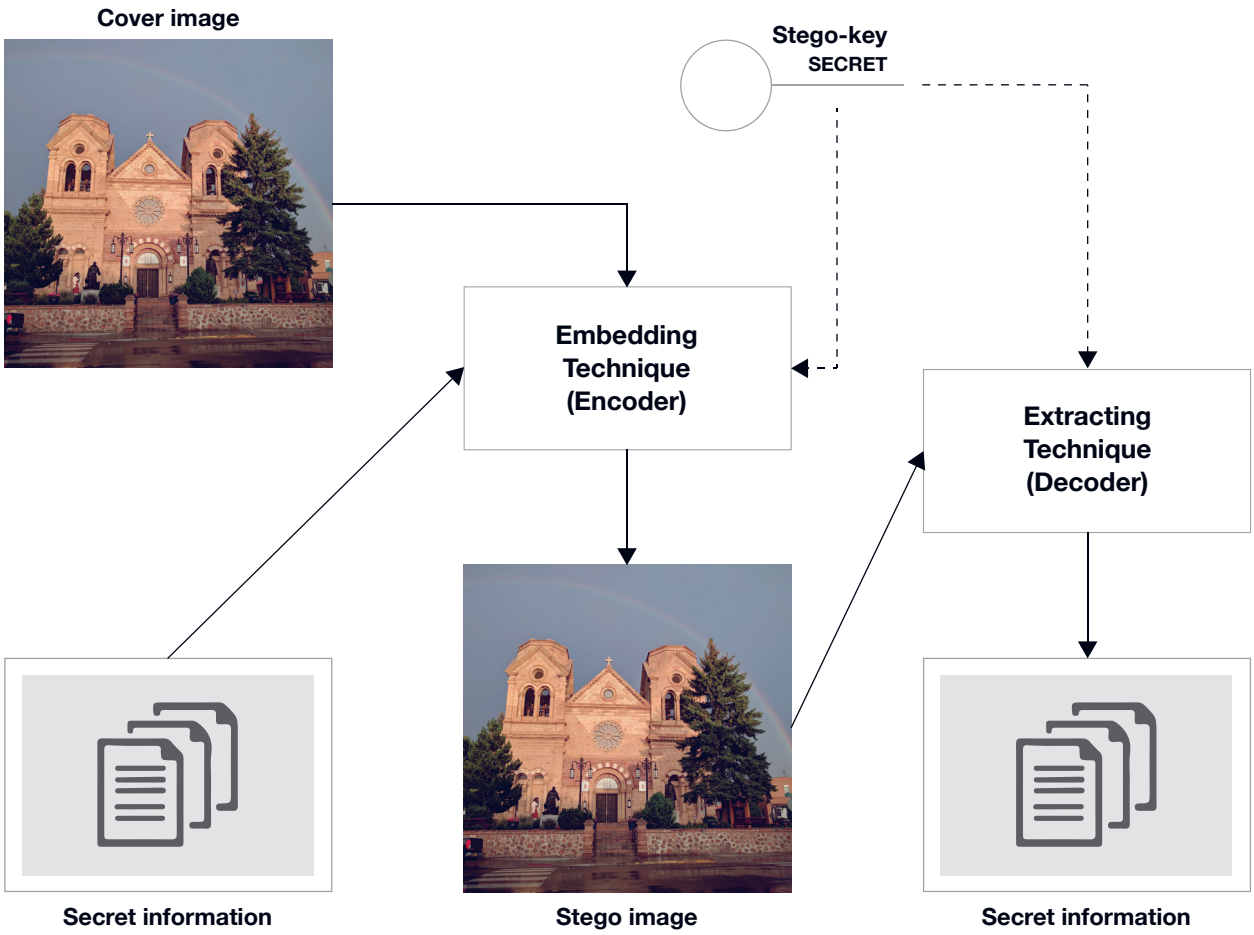

Figure 3. Schema of a data hiding model [57]

2.4 Types of data hiding

In all relevant literature, four different types of data hiding are recognized [19], [21], [44]:

- Text data hiding

- Image data hiding
- Audio data hiding
- Video data hiding

Some papers mention the fifth type - network or protocol data hiding [44].

2.5 Image data hiding techniques

When discussing data hiding, various techniques can be divided into different categories using different parameters [5]:

- information available at the decoder
- visibility
- robustness
- embedding domain

Each category is further explained below:

- **Information available at the decoder -** if a cover image is needed for detection, the data hiding method can be categorized as blind. If the detection can be performed without the cover image, the data hiding method can be categorized as non-blind, or informed [58]. Cedillo-Hernandez et al. use blind detection in their proposed method for medical imaging [59]. Muhammed et al. use the same approach, while noting that blind detection has a broader application, as the cover image may not always be available at the decoder. However, they conclude with stating that a blind data hiding technique is more complex to develop [60]. An example of non-blind detection can be found in [61].
- **Visibility -** depending on the application, a data hiding method can be visible or invisible. Most often, cryptographic methods are visible, while steganographic methods are invisible. Watermarking methods can be both invisible and visible [57].
- **Robustness -** if the main requirement of a data hiding method is copyright protection, the embedded data should be detectable even if an image has undergone image attacks. These types of methods are considered robust. However, data hiding methods can also be fragile, where the goal is to detect image tampering [62]. An example of fragile data hiding can be found in [63]. Examples of robust data hiding methods can be found in [59], [64]–[66].

- **Embedding domain -** spatial domain techniques are based on the simple mechanism of modifying certain bits of information in the cover image, without modifying the entire image [19]. One of the most common such techniques is the least-significant-bit (LSB). Techniques like LSB, as well as all spatial domain techniques are simple to implement, can achieve a high capacity, but are highly vulnerable to image attacks [67]. Pixel Value Difference (PVD) is another prominent spatial domain technique. PVD is a technique where two pixels of an image are chosen, and their value difference determines the method's capacity for those pixels - for higher differences, the capacity increases [57]. Pixel Pair Matching (PPM) is a technique that uses the coordinates of a chosen pixel pair to search another coordinate (and pixel pair). Data can be hidden in each defined pixel pair. PPM-based methods can achieve higher security levels, due to the relative positioning of hidden data [57].

  As opposed to spatial domain techniques, frequency domain techniques are robust to image attacks, and are generally considered to be a more secure way of embedding hidden information [5], [19], [28], [55], [68]. To represent an image to the frequency domain, a steganography method uses a pre-defined transform [69]. In most relevant literature, three transforms are commonly used: Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), and Discrete Fourier Transform (DFT) [5], [21], [55], [69]–[71]. DCT provides high robustness to JPEG compression attacks [28], but is susceptible to cropping and scaling [69]. DWT-based methods often correlate well with the human visual system, i.e. image quality is high due to less perceptible embedding artifacts [37]. DFT methods are most robust to geometric attacks, such as scaling, cropping, rotation, as well as to print-scan and print-cam processes [5], [28], [69]. DFT is analyzed in-depth in Section 4.

  There also exist methods that combine several embedding techniques. Ling et al. combine spatial domain embedding and DCT to minimize possible false-positive detection [23]. Singh et al. combine DCT and DWT. With using DCT for embedding, and DWT for detection, they conclude that the hybrid approach helps maximize advantages of both transforms [24]. Su et al. use spatial domain embedding in their method, but also use the DC coefficient of the Fourier transform to achieve the embedding. They conclude that this addition of using DC for embedding in the spatial domain improves performance [25].

2.6 Properties of data hiding methods

Each data hiding method possesses multiple properties. Having in mind the final application of the data hiding method is crucial when it comes to prioritizing certain properties over others. Though many more could be mentioned, three steganography properties can be regarded as most important, in the context of data hiding: imperceptibility, capacity, and robustness [5], [67], [69].

Imperceptibility

Imperceptibility, or in other words, invisibility or fidelity, is the extent to which an image appears the same as the original image [43]. In other words, if the imperceptibility of a data hiding method is high, the image quality must not be affected – apart from minor degradation in brightness or contrast. Quantitatively, imperceptibility is most often measured using an objective image quality metrics like the Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM) metrics [9]. However, subjective analyses are also commonly used [47].

Robustness

Robustness is the requirement that rates the ability of a data hiding method to stay detectable even after some common signal processing operations (image attacks) [47]. Some common such attacks include filtering, compression, print-scan, print-cam, rotation, translation, scaling, etc. [13]. Just as it is impossible to maximize all properties within a method, it is also impossible for a method to be robust to all image attacks. Depending on the final application, robustness to some image attacks will be of higher priority than others.

Capacity

Capacity, or payload, refers to the number of total bits concealed in the cover image [47]. It is most commonly expressed using Bits Per Pixel (BPP).

It is impossible to develop a data hiding method with all three properties maximized. Instead, it is a matter of accepting a trade-off between them [5], [31], [39], [69], [72]. Depending on the final application of the method, different properties can be prioritized. The trade-off is illustrated in Figure 4.

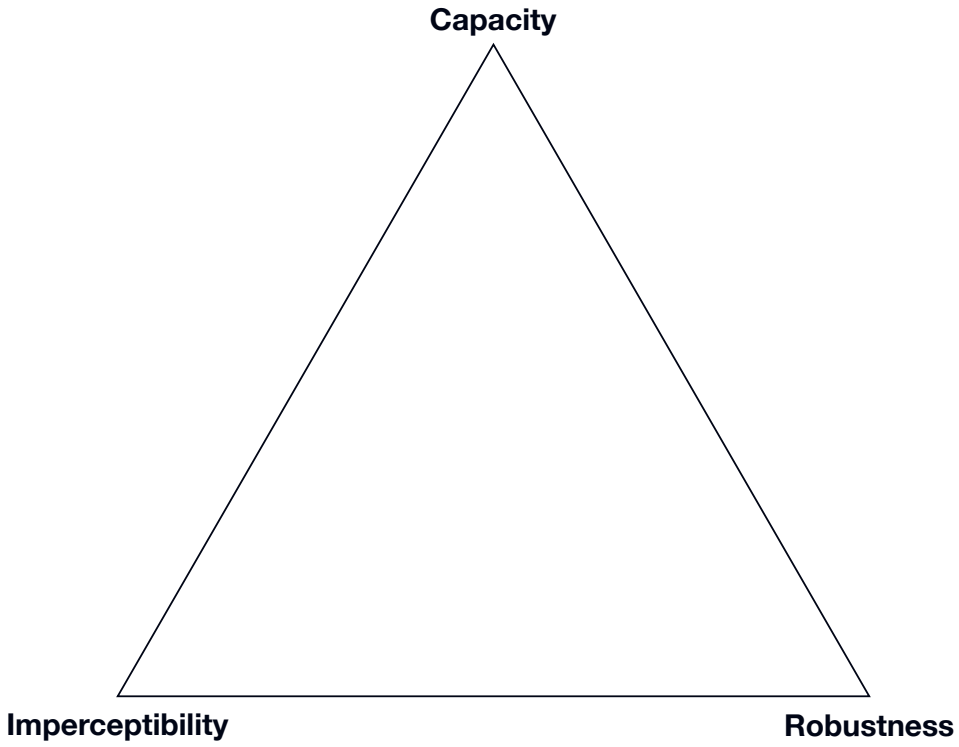

Figure 4. Illustrating the trade-off between three main data hiding properties

3 COMMUNICATION CHANNEL

Whenever an image goes through a communication channel, it is bound to sustain a level of degradation. The severity of the said degradation depends on the communication channel. This is important when developing a data hiding method, as the embedded data should survive the process once it arrives at the decoder. Song et al. suggest that *„the success of a data hiding methods relies heavily on its robustness to withstand attacks“* [73].

The degradation sustained through a communication channel can be a result of benevolent or malicious image attacks. Benevolent image attacks are a result of standard processing - there is no intent to degrade the image quality or to compromise the hidden data. Some examples of benevolent image attacks are compression, filtering, or geometrical transformations [74], [75]. Malicious image attacks, on the other hand, are intentional, with the goal to degrade the image quality and compromise the hidden data. The result of both benevolent and malicious image attacks is often the same, so it can be hard to distinguish the cause of image degradation [65]. Standard processing that usually occurs within the print-scan communication channel is discussed in-depth in Section 3.2.

3.1 Print-scan communication channel

The method presented in this thesis is suited for printing, meaning that images have to go through the print-scan communication channel before detection. In this section, the print-scan communication channel will be explained in depth. The block diagram of the print-scan communication channel is displayed in Figure 5., with each part of the communication channel explained below.

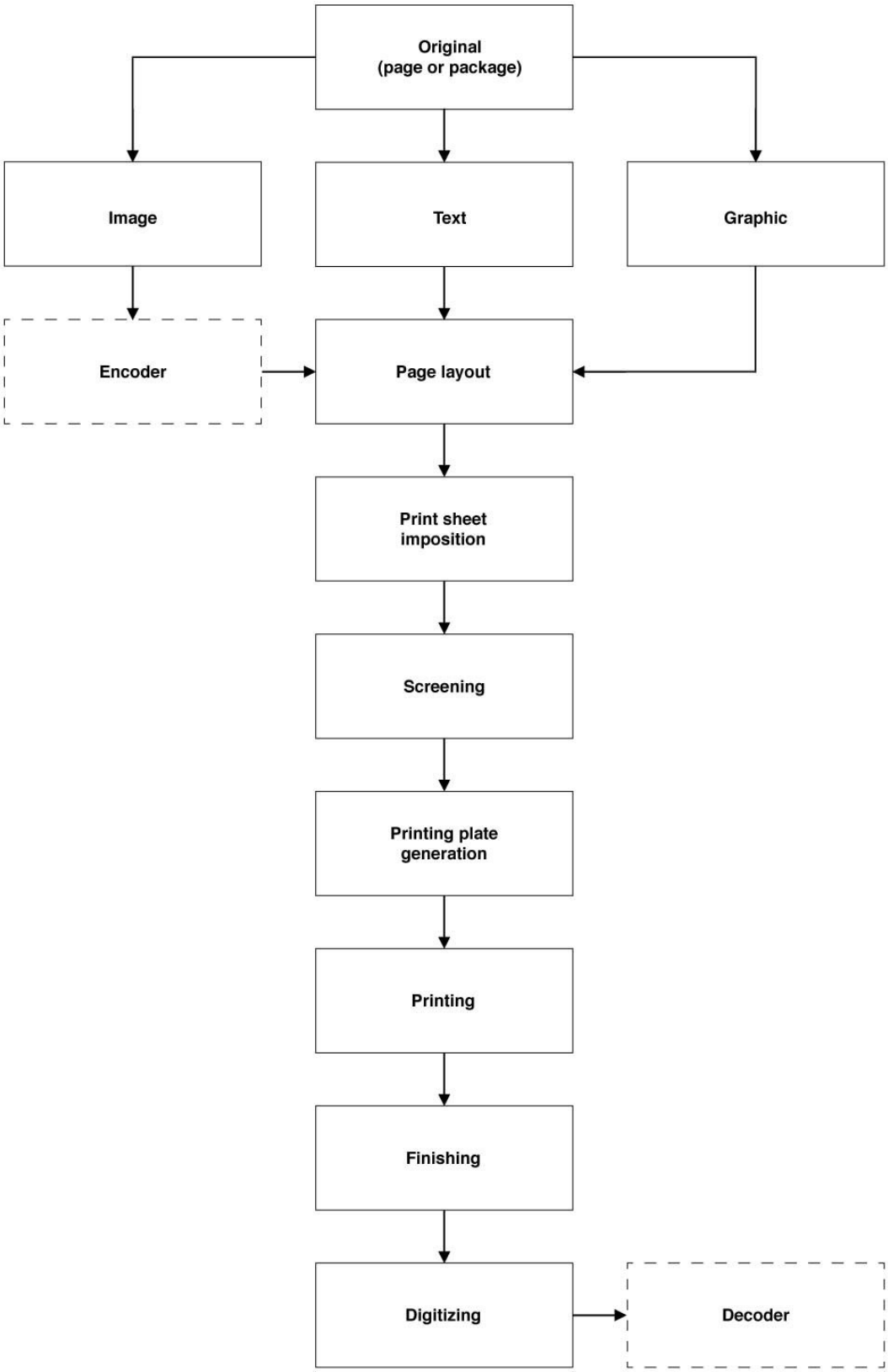

Figure 5. Block diagram of the print-scan communication channel.

Original

Each printing product starts with combining basic elements such as images, text, and graphics. This part of the printing process takes place before the encoder of the data hiding method.

Data embedding & GCR masking

The method presented in this thesis embeds data in images. Hence, images are the only basic element subject to data embedding. Gray Component replacement, which is used to mask embedding artifacts, takes place immediately after data embedding. The method's encoder is analysed in-depth in Section 10.1.1.

Page layout & print sheet imposition

Once all elements are ready for reproduction, they are positioned on the printing area, thus forming the page layout. Depending on the size of the printer, the page layout is finally positioned on the printing sheet.

Screening

During screening, a continuous-tone image is transformed into halftone image. This is needed because standard printers act as halftone devices [76], [77]. Color reproduction is achieved by using four color channels - cyan, magenta, yellow, and black. Thanks to limitations of the human eye, screening makes possible the modulation of continuous-tone images, while maintaining the image's visual appearance [5]. Screening is achieved by first calculating the mean value of a pixel neighborhood in an image and then, depending on the value, creating a halftone cell with a simulation of the respective grey value [5].

Most halftoning algorithms can be categorized as either amplitude modulated (AM) or frequency modulated (FM). AM halftoning algorithms achieve the simulation of grey levels by modifying the element's size, while FM algorithms achieve the same effect by modifying the concentration of elements [78]. The illustration of AM and FM halftoning is given in Figure 6.

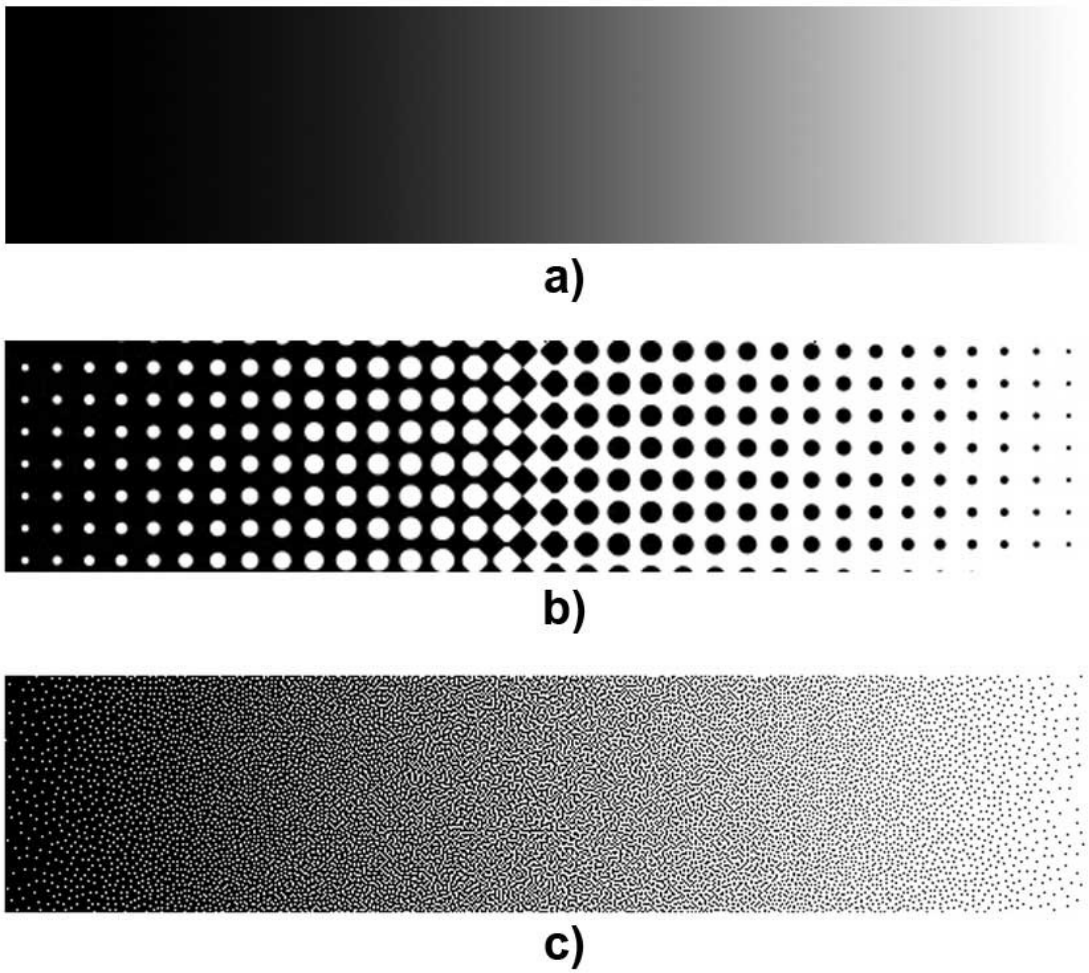

Figure 6. An illustration of a continuous-tone image (a), AM halftone result (b), and FM halftone result (c) [79].

Literature reveals a large number of halftoning algorithms that can mainly be categorized in amplitude modulated (AM) and frequency modulated (FM) algorithms [80], [81]. In AM halftoning the size of the dots varies to create the illusion of lighter or darker tones, whereas in FM halftoning, the concentration of dots is the fluctuating factor.

Printing plate generation & printing

After screening, the final halftone designs of all four color channels are ready for printing. The designs are first used to generate printing plates, which are then used in the printing process itself.

Finishing

Depending on the product's application, certain finishing operations such are done. Some of the most common finishing operations include [82]:

- Folding
- Stapling
- Stitching
- Binding
- Lamination
- Varnishing
- Digitizing

Once the product is printed, it has undergone all phases of the communication channel. In the case of the presented method, at this point it is possible to detect the embedded information at the decoder. The decoder uses an infra-red camera to digitize the image and make the detection possible. The method's decoder is analysed in-depth in Section 10.1.2.

The print-scan communication channel consists of multiple processes. Each of those processes can be controlled to a certain extent, thus minimizing the degradation done to an image. However, the totality of these small degradations amounts to a highly aggressive series of attacks that an image has to sustain throughout the channel. Most notable attacks occur during printing, where it is impossible to completely control the process, even with rigorous standardization. Industry-level processes predict up to $\pm 3\%$ changes in tone value increase (TVI). TVI, also known as dot-gain or the Yule-Nielsen effect, can be defined as *"a nonlinear relationship between the spectral reflectances of halftones and fulltones due to the internal propagation of light by scattering into the printing support"* [76].

In other words, a perfect reproduction can not be achieved due to concrete physical properties of the printing process. The predicted TVI changes are even higher for low-end printing devices such as office printers, where such high levels of standardization can not be achieved. In terms of color values, industry-level processes can successfully achieve precision within a $\Delta E = 5$ range. Once again, for low-end devices, expected.ΔE_{ab} values are considerably higher.

Numerous transforms occur during digitizing an image (in this case, capturing an image with an *infra-red* camera). An image can be subject to geometric transforms such as rotation, scaling, translation, and cropping. Also, a loss in contrast and sharpness is to be expected during digitizing. Finally, depending on the camera's limitations, compressions such as JPEG can further compromise the image.

3.2 Image processing in the print-scan communication channel

3.2.1 Geometrical transformations

Geometrical transformations influence the image's linear coordinates. In other words, these transforms change the spatial relations between image elements [52]. Geometrical transformations can be divided into two separate operations:

- The transform of coordinates
- Interpolation of transform coordinates' intensity values

The image elements' coordinate transform is expressed as follows:

$$(v, u) = T\{(x, y)\},$$

(1)

where (x, y) represent the cover image's coordinates, (u, v) represent the transformed image's coordinates, and T represents the function operator.

The coordinate transform can also be expressed as a matrix [83]:

$$b = aT,$$

(2)

i.e:

$$[u \quad v \quad 1] = [x \quad y \quad 1] \begin{bmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 1 \end{bmatrix},$$

(3)

Depending on the values of T, this transform can rotate and scale the image's coordinates. The benefit of expressing these transforms with a matrix is the operator's ability to perform multiple operations simultaniously - by simply multiplying matrices [5]. The coordinates can be transformed back to their original state by using the inverse transform T^{-1}. The matrices for rotation (4) and scaling (5) are given below:

$$\begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(4)

$$\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

(5)

After applying transforms, new element values have to be interpolated. Though there are many techniques of interpolation, three are used most often - *nearest neighbor*, *bilinear interpolation*, *bicubic interpolation* [84], [85].

3.2.2 Image filtering

The term *filter* was first used in frequency domain processing, where the filtering refers to passing or rejecting certain frequencies, or frequency components. For example, a filter that passes low frequencies is called *lowpass filter*. The result of using such a filter is a smoothed, blurred image. Similar results can be accomplished using spatial filters (*masks*, *kernels*). Spatial filtering offers much more versatility, as it can be used for nonlinear filtering, something that is not possible in the frequency domain [5], [52].

Each spatial filter consists of a *neighborhood*, and a *predefined operation* that is performed on a central pixel within its surrounding neighborhood. The values of new pixels are the result of the filtering operation. The filtered image is generated after the filter visits each pixel of the input image. The filtering operation can be linear or nonlinear. Linear filtering is done using a spatial filter - a mask the size of, most often 3×3 or 5×5 elements. The mask goes through each element of an image and computes the intensity of each central element using mask coefficients, and intensities of neighbouring elements. The mask is illustrated in Figure 7.

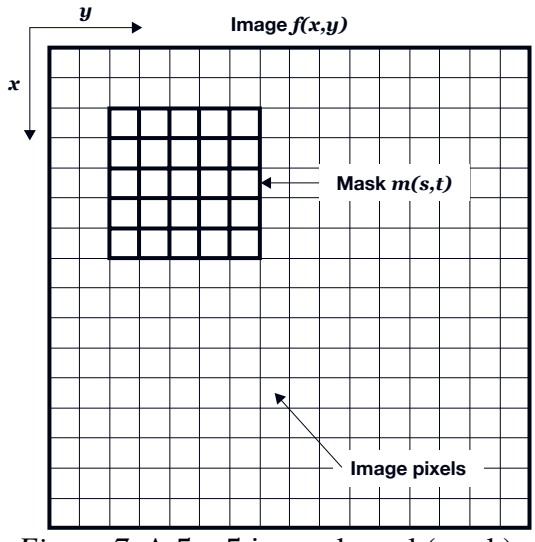

Figure 7. A 5 *x* 5 image kernel (mask)

The mathematical expression for spatial filtering is given below:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} m(s,t) f(x+s, y+t),$$

(6)

where $g(x,y)$ represents the filtered image, a, b represent image dimensions, and $m(s,t)$ represents the spatial filter (mask).

Generating a spatial filter requires the specifications of the filter mask coefficients. The coefficients are chosen based on what the filter needs to achieve, i.e., what the filtered image will look like. For example, one can replace the pixels in a 3×3 neighborhood around the central pixel, with the average pixel value in that neighborhood. The average value for any location (x,y) in the image is the sum of nine-pixel intensity values in the 3×3 neighborhood centered on (x,y), divided by nine. The mathematical expression is given below:

$$R = \frac{1}{9} \sum_{i=1}^{9} z_i,$$

(7)

where R represents the average pixel value, and $z_i, i = 1,2, \ldots, 9$ denotes individual pixel values in the neighborhood. The example given above results in *image smoothing* - one of the basic spatial filters. Some other most often used spatial filters in image processing are lowpass filter, mean filter, Gaussian filter, Wiener filter, weighted mean filter, anisotropic filter, bilateral filter for Gaussian noise, and median filter for Laplace noise [5], [86]. Equations and filters (masks) of lowpass, Gaussian and Laplace filters are given in Table 2.

Table 2. Commonly used spatial filter masks.

Filter name	**Equation**	**Filter (mask)**
Lowpass filter	$g(x,y) = \frac{1}{9} \sum_{s=-1}^{1} \sum_{t=-1}^{1} f(x+s, y+t)$; *(8)*	$\begin{matrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{matrix}$

Gaussian filter	$g(x,y) = \sum_{s=-1}^{1}\sum_{t=-1}^{1} e^{\frac{s^2+t^2}{2\sigma^2}} f(x+s,y+t);$ (9)	$\begin{matrix} 1/16 & 1/8 & 1/16 \\ 1/8 & 1/4 & 1/8 \\ 1/16 & 1/8 & 1/16 \end{matrix}$
Laplace filter	$\nabla^2 f(x,y) = \frac{\partial^2 f(x,y)}{\partial x^2} + \frac{\partial^2 f(x,y)}{\partial y^2};$ (10)	$\begin{matrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{matrix}$

Image smoothing

Smoothing filters are used for blurring and reducing noise in an image. The output of such filters is the average value of pixels inside the filter (mask) neighborhood. For this reason, smoothing filters can also be referred to as *averaging filters*. In some literature, the name *lowpass filters* also occurs [52]. By replacing each pixel value with the neighborhood's average, sharp transitions in an image are reduced. Those sharp transitions are what causes noise in an image. Therefore, smoothing filters can reduce noise in an image. The downside of smoothing filters is that some edges (also sharp transition areas) are blurred during the process. The mathematical expression of smoothing filters is given earlier, in (7).

Image enhancement

Image enhancement filters are based on the idea of enhancing local high frequencies (sharp areas) of an image. This can be achieved using an inverse approach - by suppressing the image's low frequencies. One such technique is *unsharp masking* [54]. The unsharp masking algorithm subtracts the blurred version of the image, from the image itself. By doing this, only the high frequencies are left, which, in turn, leads to an enhanced image. The blurred version of the image is most often achieved with a Gaussian mask. The mathematical expression of the unsharp masking for a local window $\sigma(x,y)$ is given below:

$$g(x,y) = A[f(x,y) - m(x,y)] + m(x,y),$$

(11)

where $f(x,y)$ represents the pixel value, $m(x,y)$ represents the *lowpass* pixel value, and A represents the amplification factor.

The amplification factor A is defined as:

$$A = \frac{kM}{\sigma(x,y)},$$

(12)

where k denotes a constant, and M denotes the average pixel value of the entire image. The pixel value does not change if $f(x,y) - m(x,y)$ is above a certain threshold.

Another image enhancement filter is the Wiener filter. It is also often used for image restoration. The algorithm of the Wiener filter tries to minimize the mean squared error between the original, and the filtered image [87].

3.2.3 Image compression - JPEG

The JPEG image compression standard was first introduced in 1992., when the group *„Joint Photographic Experts Group"* developed their standard *„JPEG 1"* [88]. Since then, the group developed numerous additions to the algorithm, including the *„JPEG2000"*, the most important one to date. The JPEG compression is based on the Discrete Cosine Transform (DCT). Below are the steps that JPEG compression is comprised of these steps [89], [90]:

1. **Color space transform**
   Converting the given image to YC_bC_r color space. This is done because the human visual system tends to focus on spatial content more than color content, for visual interpretation. Converting to the YC_bC_r color space separates luminance information from chrominance information.
2. **Downsampling**
   The term spatial resolution represents the total number of pixels in an image. If the number of pixels is higher, then the resolution of the image is higher. Downsampling reduces the total number of pixels in an image, depending on the sampling frequency [91]. After the downsampling is done, the resolution and size of the image both decrease. The spatial downsampling is done on the chromatic channels in the YC_bC_r space. The human eye is much more sensitive to changes in luminance, so downsampling the chromatic information does not affect the human perception of the image as much, making the changes less apparent [6].
3. **Dividing image into blocks**
   The image is divided into $8 \times 8\ px$ blocks.

4. **Discrete Cosine Transform**

   YC_bC_r spatial image data is transformed to a frequency domain representation using Discrete Cosine Transform (DCT). The DCT is being done for each image block separately. This step allows the JPEG algorithm to further compress the image data as outlined in the next steps by computing DCT coefficients.

5. **Quantization**

   The quantization is performed on each $8 \times 8\ px$ image block separately. Each pixel value, in the frequency domain, is divided by the quantization coefficient. This is where the JPEG algorithm achieves majority of the compression at the expense of image quality, making the compression *lossy*. If the quantization is successful, each image block will consist of mostly zeros, with only a few exceptions - in the low-frequency areas.

6. **Entropy encoding**

   The coefficients of the $8 \times 8\ px$ matrix are ordered using the *zig-zag sequence*. The *zig-zag sequence* first encodes the low-frequency coefficients (non-zero values first), and then the high-frequency coefficient (the encoding ends with zero values). Finally, *Run Length Encoding (RLE)* and *Huffman Encoding* are performed to achieve a shorter encoding bit-length [5], [89].

Even though the technique has its shortcomings - namely the introduction of visible blocking artifacts, JPEG is today the most commonly used image compression technique [88], [92], [93].

4 FOURIER TRANSFORM

When hiding data in images for the print domain, the embedding signal has to be considerably high, as the method has to sustain the process of reproduction. With that in mind, it is important to choose an optimal embedding technique. As stated in multiple sources [4], [5], [59], [94], embedding in the frequency domain is the optimal choice, as it offers more robustness than spatial domain embedding. Also, embedding artifacts are spread across the entire image, leading to less image deterioration.

Recent literature recognizes three mathematical transforms for transforming an image to the frequency domain:

- Discrete Cosine Transform (DCT)
- Discrete Wavelet Transform (DWT)
- Discrete Fourier Transform (DFT)

Each transform offers its upsides. The DCT is often applied in areas such as pattern recognition and data compression [68]. Methods using the DCT are robust against simple operations - low pass filtering, brightness and contrast, blurring, etc. Given that the JPEG compression is based on the DCT, methods using the DCT are also robust to JPEG image compression. However, DCT-based methods are not robust against geometric distortions [58]. The DWT decomposes the image into three spatial directions, giving it characteristics similar to the Human Visual System (HVS) [58]. It is also computationally efficient. Finally, the DFT provides high robustness against geometrical attacks, as it is inherent to rotation, translation, and scaling [5], [58]. Also, the DFT uses very detailed basis functions, so in general, it can approximate an image with greater precision than other transforms [54]. The very fact that the print-scan communication channel consists of multiple geometrical transforms, data hiding methods suited for the print domain should be DFT-based.

In 1822., a french mathematician Jean Baptiste Joseph Fourier published a book *"Théorie Analytique de la chaleur"*, or *"Analytical theory of Heat"* [95]. The book was a summation of the work that started some fifteen years ago, as Fourier started exploring periodic signals in 1807. However, due to objections on his work from contemporaries, the work was not public until Fourier decided to publish his own book [96], [97]. Fourier discovered that any periodic signal could be presented as a sum of a series of sine and cosine functions - as displayed in Figure 8.

"An arbitrary function, continuous or with discontinuities, defined in a finite interval by an arbitrarily capricious graph can always be expressed as a sum of sinusoids"

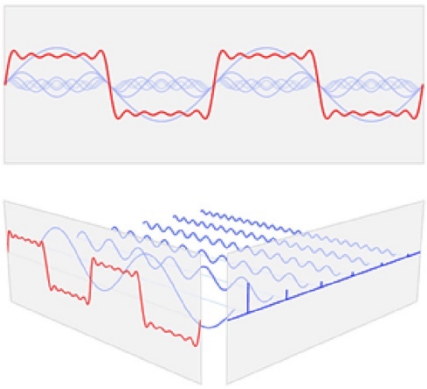

Figure 8. Displaying a signal as a sum of sine and cosine functions [98]

That finite interval defining an arbitrary function is known as the Fourier transform. One of the more important properties of the Fourier transform is the ability to transfer the signal back to its original form. This is done by the inverse Fourier transform. Today, the Fourier transform is considered to be a fundamental and most widely spread tool of signal processing, used in a variety of fields such as telecommunications, image processing, applied mechanics, acoustics, electromagnetics, and medical engineering [5], [96], [99], [100]. One of the main reasons for why this widespread development was made possible is the discovery of the fast Fourier transform. Until then, the use of the Fourier transform was limited due to its high computational costs. The fast Fourier transform was introduced by Cooley and Tukey. In their 1965. work *"An Algorithm for the Machine Calculation of Complex Fourier Series"* [101], the authors developed an algorithm based on the Fourier transform for breaking down large data samples into smaller samples by dividing the transform into two pieces at each step. In some instances, this could represent a time reduction of 96% [96]. This mechanism of using the Fourier transform made the computational costs much lower, leading to a widespread application of the algorithm.

4.1 1-dimensional Fourier transform

This section contains the overview of the Fourier transform - from the continuous Fourier transform using one variable, to the two-dimensional discrete Fourier transform (DFT).

The Fourier transform of the function $f(x)$ is:

$$F(u) = \int_{-\infty}^{\infty} f(x) e^{-2\pi i u x} \, dx,$$

(13)

where the $F(u)$ function represents the Fourier transform of the continuous function $f(x)$, while u and x represent continuous variables.

Its inverse function is defined as:

$$f(x) = \int_{-\infty}^{\infty} F(u) e^{2\pi i u x} \, du,$$

(14)

where the function $f(x)$ represents the inverse Fourier transform of the continuous function $F(u)$, while u and x represent continuous variables.

Equations (13) and (14) act as a Fourier transform pair and prove an important property of the Fourier transform - the possibility of reconstructing the initial function $f(x)$ from the transform function $F(u)$.

In practical terms, especially for image processing, continuous functions are not commonly used. Instead, a sampling of the function is done in order to get a discrete set of numbers. For the discrete set of numbers, (13) becomes:

$$F(u) = \sum_{x=0}^{n-1} f(x) e^{\frac{-2\pi i u x}{n}},$$

(15)

for $u = 0, 1, 2, \ldots, n-1$;

where $F(u)$ denotes the discrete Fourier transform of the function $f(x)$, while x is a variable of the discrete set of numbers, and n is the total amount of elements within the set.

The inverse function is defined as:

$$f(x) = \frac{1}{n} \sum_{u=0}^{n-1} F(u) e^{\frac{2\pi i u x}{n}},$$

(16)

for $x = 0, 1, 2, \dots, n-1$;

where $f(x)$ represents the inverse discrete Fourier transform of the function $F(u)$, u represents the discrete variable, and n represents the total amount of elements within the set.

4.2 2-dimensional Fourier transform

Since they are two-dimensional, digital grayscale images can be represented by a function with two variables - $f(x, y)$. Hence, the Fourier transform definition should be modified accordingly. The two-dimensional discrete Fourier transform of an image $f(x, y)$ is defined as:

$$F(u, v) = \sum_{x=0}^{m-1} \sum_{y=0}^{n-1} f(x, y) e^{-2i\pi\left[\frac{ux}{m} + \frac{vy}{n}\right]},$$

(17)

for $u = 0, 1, 2, \dots, n-1;\ v = 0, 1, 2, \dots, m-1$

where $F(u, v)$ represents a discrete two-dimensional Fourier transform of the image $f(x, y)$, with $m \times n$ dimensions.

The inverse discrete Fourier transform with two variables is defined as:

$$f(x, y) = \frac{1}{mn} \sum_{u=0}^{m-1} \sum_{v=0}^{n-1} F(u, v) e^{2i\pi\left[\frac{ux}{m} + \frac{vy}{n}\right]},$$

(18)

for $x = 0, 1, 2, \dots, n-1;\ y = 0, 1, 2, \dots, m-1$

where $f(x, y)$ represents the inverse discrete Fourier transform of the function $F(u, v)$.

Equations (17) and (18) act as a two-dimensional discrete Fourier transform pair.

4.3 Magnitude and phase of the Fourier transform

As a complex function, the two-dimensional discrete Fourier transform can be expressed in its polar shape - the magnitude and the phase. The polar shape of the function $F(u, v)$ is defined as:

$$F(u, v) = |F(u, v)| e^{-i\emptyset(u,v)},$$

(19)

for $u = 0, 1, 2, \dots, n-1;\ v = 0, 1, 2, \dots, m-1$

where $|F(u,v)|$ represents the magnitude (spectrum), while $\emptyset(u,v)$ represents the phase of the function $F(u,v)$. The magnitude and the phase of the function $F(u,v)$ are defined as:

$$|F(u,v)| = \sqrt{R^2(u,v) + I^2(u,v)},$$

(20)

$$\emptyset(u,v) = tan^{-1}\left[\frac{I(u,v)}{R(u,v)}\right],$$

(21)

for $u = 0, 1, 2, \dots, n-1;\ v = 0, 1, 2, \dots, m-1$

where R and I represent the real and the imaginary part of the function $F(u,v)$.

In Figure 9., an example of a function $f(x,y)$, with its magnitude $|F(u,v)|$ and phase $\emptyset(u,v)$ is depicted.

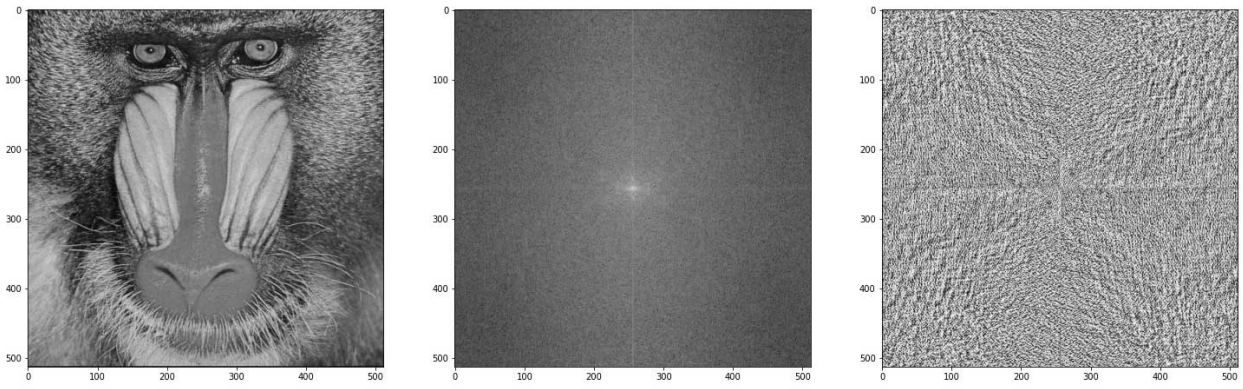

Figure 9. a) The image in the spatial domain; b) The image's magnitude (spectrum) in the frequency domain; c) The image's phase in the frequency domain

4.4 Fast Fourier transform

The widespread application of the Fourier transform for image processing would not be possible without the development of the fast Fourier transform [5]. Careful inspection of (17) and (18) reveals that, for an image of $M \times N$ dimensions, a total of $(M \times N)^2$ operations would be needed to complete the DFT. For an image of 1024×1024 elements, that total would amount to over 10^{12} operations. With the implementation of the fast Fourier

transform (FFT), the total time for completing the algorithm is equal to $T(n) = O(nlog_2 n)$. The total time for the same 1024×1024 image mentioned above would amount to only 2×10^6 operations - a significantly lower value.

The FFT algorithm became known to the public after Cooley and Tukey published their 1965. paper *"An Algorithm for the Machine Calculation of Complex Fourier Series"* [101]. However, the FFT algorithm is sometimes recognized as J. F. Gauss's discovery, as a similar algorithm can be found in Gauss's manuscripts *"Theoria interpolationis methodo nova tractata"* from 1805. - seventeen years before Fourier's first discoveries were published. The debate about who should claim the discovery of the FFT algorithm still exists [102]. Whatever the case may be, today's computer programs use Cooley and Tukey's version of FFT, so this section will cover the theory behind their recursive algorithm.

The algorithm works under the precondition that the vector for which the DFT is calculated consists of n elements, and that n has the value of a power of two. In mathematical terms:

$n = 2^a, a \in \mathbb{N}$.

Let:

$$\omega_n = e^{\frac{2\pi i}{n}},$$

(22)

With (22) in mind, it can be concluded that for any integer $n > 0, d > 0, and\ u \geq 0$, the following stands:

$$\omega_{dn}^{du} = \omega_n^u,$$

(23)

For each even integer $n > 0$ the following stands:

$$\omega_n^{n/2} = \omega_2 = -1,$$

(24)

Which yields:

$$\omega_n^{u+n/2} = -\omega_n^u,$$

(25)

If $n > 0$ is an even number, the squares of n solutions of the n^{th} complex root of 1 are equal to $n/2$ solutions of the $n/2^{th}$ complex root of 1. The analogy is shown in the equation below:

$$(\omega_n^u)^2 = (\omega_n^{u+n/2})^2 = \omega_{n/2}^u,$$

(26)

If we take n complex numbers ω_n^u, where n is an even number, and $u = 0, 1, \dots, n-1$, from (25) it can be concluded that, in the first half of the set ($u = 0, 1, \dots, n/2 - 1$), each ω_n^u will have its equal, only with the negative value, in the latter half of the set $\omega_n^{u+n/2}$. For purposes of calculating the DFT, an example is given using the n^{th} complex square root of 1, such that n is an even number, as for even n, a set of results consists of a pair of numbers with opposite values (as depicted in Figure 10.)

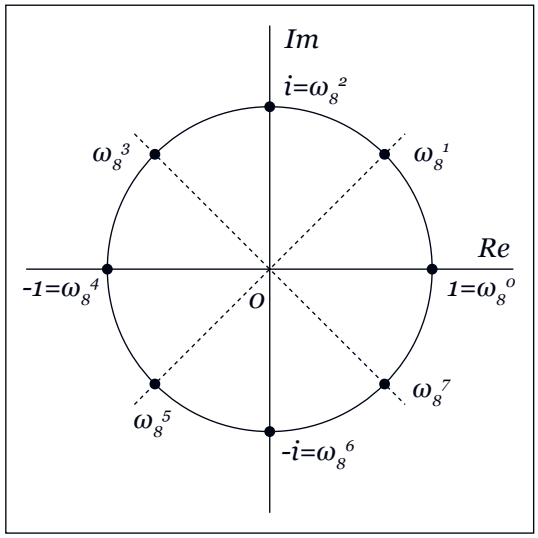

Figure 10. The results of complex square roots

The substitution from (22) and writing out the sum from (15), leads to:

$$F(u) = \sum_{x=0}^{n-1} f(x)(\omega_n^u)^x,$$

(27)

$$F(u) = f(n-1)(\omega_n^u)^{n-1} + f(n-2)(\omega_n^u)^{n-2} + \cdots + f(1)(\omega_n^u) + f(0),$$

(28)

for $u = 0, \dots, n-1$, under the assumption that n is an even number. This sum is then grouped in a way that the left sum of the equation consists of elements for even values of x, and the right sum consists of elements for odd values of x.

$$F(u) = [f(n-2)(\omega_n^u)^{n-2} + \cdots + f(0)] + [f(n-1)(\omega_n^u)^{n-1} + \cdots + f(1)(\omega_n^u)],$$

(29)

From the right bracket, we can extract ω_n^u:

$$F(u) = [f(n-2)(\omega_n^u)^{n-2} + \cdots + f(0)] + \omega_n^u[f(n-2)(\omega_n^u)^{n-1} + \cdots + f(1)],$$

(30)

where $u = 0, 1, \dots, n-1$. By using (28), elements from the left bracket can be translated in the following way:

$$\begin{aligned} &f(0) + f(2)(\omega_n^u)^2 + f(4)(\omega_n^u)^{2\cdot 2} + \cdots + f(n-2)(\omega_n^u)^{2(n/2-1)} = \\ &= f(0) + f(2)\left(\omega_{n/2}^n\right) + f(4)\left(\omega_{n/2}^u\right)^2 + \cdots + f(n-2)\left(\omega_{n/2}^u\right)^{n/2-1}, \end{aligned}$$

(31)

The same can be done with the elements from the right bracket. This yields a new shape for the general expression of $F(u)$:

$$\begin{aligned} F(u) = &\left[f(n-2)(\omega_{n/2}^u)^{n/2-1} + \cdots + f(2)\left(\omega_{n/2}^u\right) + f(0)\right] \\ &+ \omega_n^u\left[f(n-1)(\omega_{n/2}^u)^{n/2-1} + \cdots + f(3)\left(\omega_{n/2}^u\right) + f(1)\right], \end{aligned}$$

(32)

Expressions in the left and right brackets represent two new DFTs with half as many elements, leading to half as many complex square roots. Hence, the expression can be written as:

$$F(u) = DFT_{n/2}\big(f(e)\big) - \omega_n^u DFT_{n/2}\big(f(o)\big),$$

(33)

where $e = 0,2, \dots, n-2$; and $o = 1,3, \dots, n-1$.

Given that $\omega_{n/2}^{u+n/2} = \omega_{n/2}^u$, and $\omega_n^{u+n/2} = -\omega_n^u$, it can be seen in (33) that for elements $F(u)$ and $F(u + n/2)$, for $u = 0,1, \dots, n/2 - 1$, on a recursive level, the same DFT is

calculated, with the difference being that in $F(u + n/2)$, there is a negative prefix in front of the right bracket:

$$F(u + n/2) = DFT_{n/2}\big(f(e)\big) + \omega_n^u DFT_{n/2}\big(f(o)\big),$$

(34)

To conclude, this divide-and-conquer approach saves a half of all operations on this recursive level. The same can be done at the next level for $f(e)$ and $f(o)$, where DFT is calculated by complex solutions of $(n-2)^{th}$ root of 1. However, for the recursion to continue, n must be the power of two. In mathematical terms, $n = 2^a$. The FFT recursion for $n = 8$ is depicted in Figure 11:

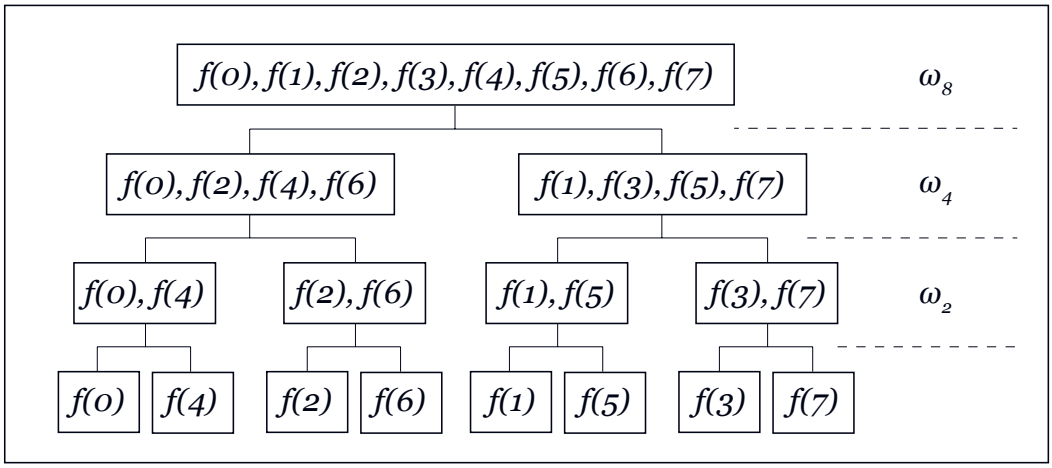

Figure 11. Recursion of FFT for $n = 8$

4.5 Fourier transform for image processing

In this thesis, FFT is used to manipulate images, i.e., to embed data inside images. Hence, it is important to understand how the Fourier transform can be used for image processing. In this chapter, such illustrative examples will be given.

When applying the Fourier transform, an image is transformed from the spatial domain into the frequency domain. Given that the Fourier transform is a complex function, that image is represented by both the magnitude and the phase (depicted in Figure 9). In the method presented in this thesis, only the magnitude of the frequency domain is manipulated. The center of the magnitude correlates to low-frequency areas of an image, while the edges of the magnitude correlate to high-frequency areas. With this in mind, it is possible to use different filters (masks) to achieve certain results. Image filtering is presented in-depth in 3.2; however, for a better understanding of how the Fourier transform can be used for image processing, some illustrative examples of image filtering are given below.

Lowpass filters are used to attenuate high frequencies of an image, while passing low frequencies. As the high-frequency areas of an image in the spatial domain are represented by sharp edges, the resulting image is blurred (Figure). Following the same logic, highpass filters are used to attenuate low frequencies, while passing high frequencies of an image. In the spatial domain, low-frequency areas of an image are represented by smooth gradients and subtle transitions, so the resulting image is consisted only of edges and sharp transitions (Figure 13).

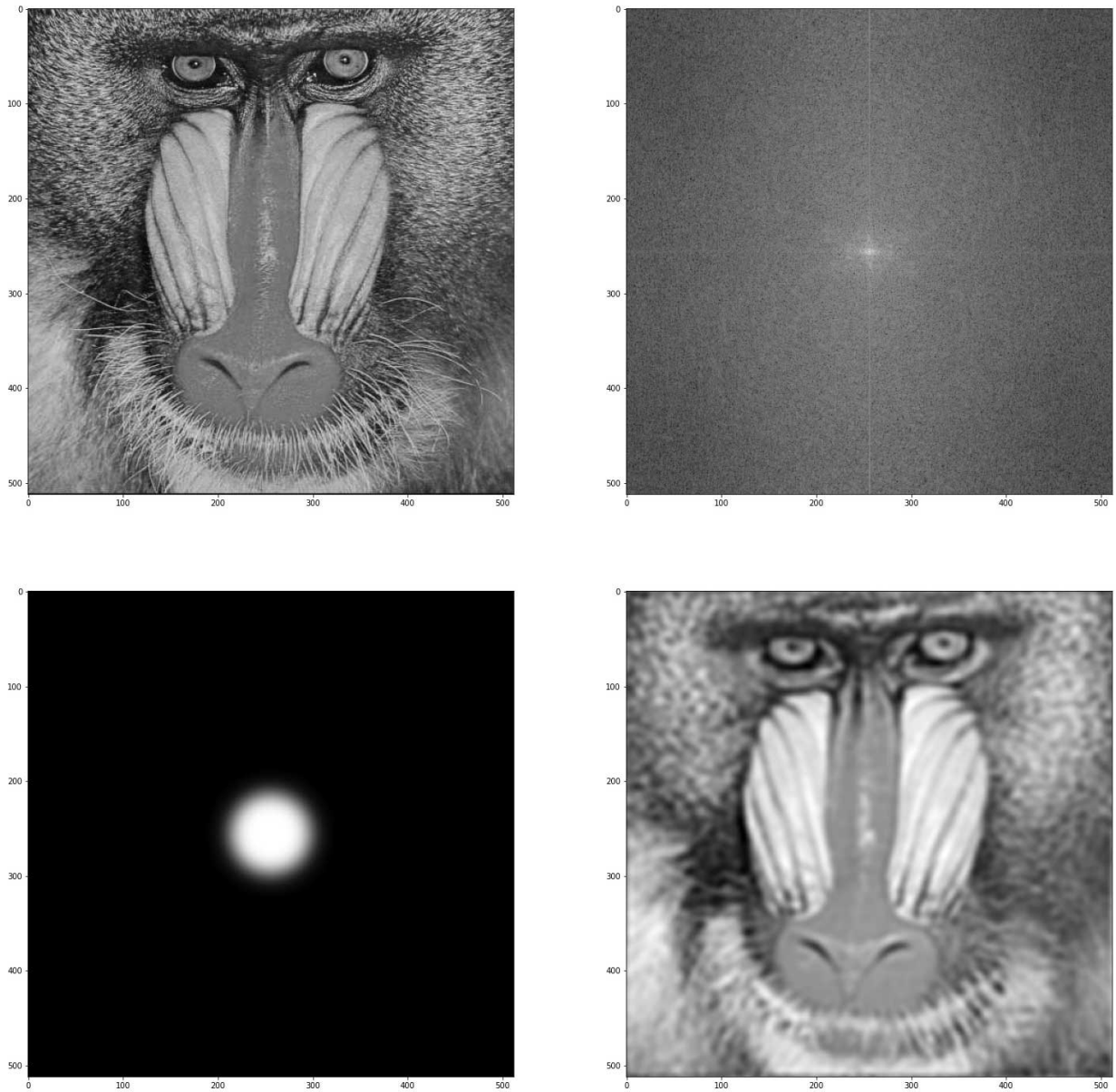

Figure 12. By masking all areas of the magnitude except the center, only low-frequency ranges are passed. The resulting image is blurred.

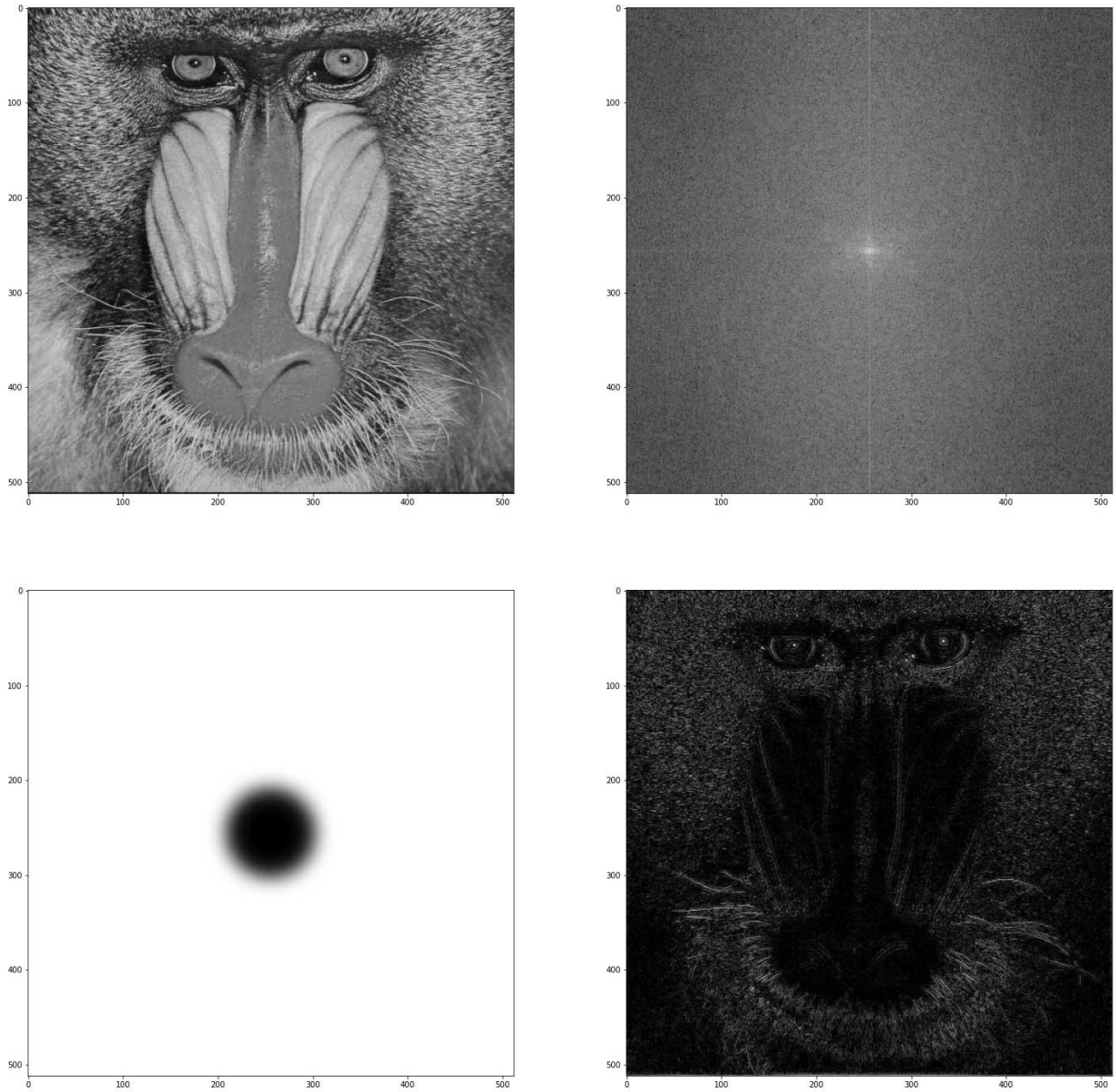

Figure 13. By only masking the center of the magnitude, low-frequency ranges are blocked. The resulting image consists only of sharp edges.

5 GRAY COMPONENT REPLACEMENT

5.1 Optimizing the printing reproduction process

There have been many attempts to colorimetrically optimize the reproduction process in printing. Often times, the fourth ink - black, in addition to three chromatic inks - yellow, magenta, and cyan, has been in the center of these attempts. The definitive theoretical principles of four-color printing are given by J. A. C. Yule in his 1940 work titled *„Theory of Subtractive Color Photography"* [103]. Yule argues that chromatic inks - $cyan$, $magenta$, and $yellow$, are not ideal block dyes. This is why, when mixed in equal amounts, the mixture results in dark brown instead of black. Hence, the $black$ color is added to achieve higher color densities. Furthermore, Yule states that the reproduction of a neutral gray color can be accurately represented either using only the three chromatic inks, only $black$, or any combination of these two. This redundancy of the four-color printing reproduction enables many color correction and ink optimization techniques used today [104]. Yule would later, in 1967, publish his seminal work *„Principles of color reproduction"*, based on the same principles discussed in his earlier works [105].

One of the first attempts of developing a color correction technique in four-color print was the 1948 paper *„Color Correction in Color Printing"* by A. Hardy and F. Wurzburg [106]. The authors argue that three-color printing results in loss of detail and examine the advantages of using four-color printing instead. Color corrections in four-color processes are suggested, using Neugebauer equations to solve cyan, magenta, and yellow values [107]. Many other methods and techniques were published and further developed in the second part of the century.

For purposes of this thesis, the main focus will be on the technique gray component replacement, a technique that first started to find its widespread application in the 1980s and 1990s, when multiple papers were published on the topic [108], [109], [110].

The definition of gray component replacement (GCR) can be found in multiple research papers [111]–[113]. Kang, in his work *„Computational Color Technology"*, defines GCR as *„a technique in color printing wherein the neutral or gray component of a three-color image is replaced during reproduction with a certain level of black ink"* [113]. Kang continues to describe the GCR algorithm as a complex process, consisting of several steps. These steps are explained below:

- Gray Balancing
- Color Correction
- Under Color Removal
    - Finding the minimum of three primary colors and determines the amount of each primary to be removed
- Black Generation
    - Determining the amount of the black ink to be added
- Under Color Addition
    - Adding color primaries to enhance the saturation and depth
- Mathematical analysis

Furthermore, Kang notes that the algorithm has a built-in mechanism for limiting maximum ink loading, which is GCR's key benefit. Apart from ink loading, other papers mention numerous other benefits of using GCR, such as [111], [112]:

- Extending the color gamut, especially the production of dense black colors
- Improving image definition, especially in shadow details
- Minimizing the use of expensive chromatic inks, by replacing them with a less expensive achromatic ink
- Reducing ink consumption, resulting in better drying
- Consistent color reproduction throughout a single pressrun
- Enabling better gray balance
- Enabling the use of light and low-grade papers

However, certain disadvantages of GCR can also be found in relevant literature, such as [104]:

- Appearance of visual artifacts (banding and contouring)
- Graininess (when used in conjunction with AM raster)

5.2 Application of GCR for watermark masking

Apart from its basic application in color correction, GCR can be used in data hiding as well. Queiroz et al. use GCR as an embedding tool in their works [114], [115]. The authors conclude that, even though the results show the potential of GCR use in watermarking, the working system is far from developed. As the main upside, the authors mention the method's robustness.

Poljičak et al., on the other hand, use GCR as a masking tool to hide visible embedding artifacts, thus keeping image quality consistently high, even after data embedding [7], [28], [64]. The results show that the GCR is able to improve image quality, and maintain the image quality consistent, regardless of implementation strength during data embedding. The GCR method is based on the simplex interpolation reverse model, presented in [116]. The method can be dissected into several steps:

1. GCR takes the CIELAB value for each pixel of the cover image
2. GCR takes the k value of the corresponding pixel in the embedded image
3. GCR computes c, m, y values to achieve the CIELAB values in such a way that the visual apearance of the embedded image matches the visual appearance of the cover image, while keeping the k channel intact

The visual representation of the GCR method can be seen in Figure 12.

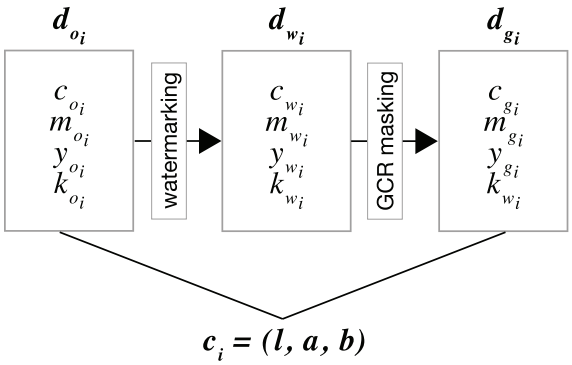

Figure 12. Schema of GCR masking

The same GCR algorithm used in [64] is used in the method presented in this thesis as well. Below, the method's setup is given. Further explanation is given in Section 10.1.1. The algorithm is setup as follows:

1. Generate a grid of n uniformly spaced points per axis in CMYK space:

$$D = \{(c_i, m_i, y_i, k_i) | i \in \{1,2, \dots, n^4\}\} \quad (35)$$

2. Convert points from CMYK color space to CIELAB color space using the ICC profile:

$$C = \{(l_i, a_i, b_i) | i \in \{1,2, \dots, n^4\}\} \quad (36)$$

3. Determine intervals in set C:

$$[l_{min}, l_{max}], [a_{min}, a_{max}], [b_{min}, b_{max}]$$

(37)

4. Divide intervals from (37) into $m-1$ sub-intervals for m per axis cube list grid-points.
5. Set up a lookup table (LUT) with $m-1$ sub-intervals from step 4. as inputs.
6. For each set of sub-cube vertices in CMYK color space $S_d \subset D$, take corresponding points in CIELAB space $S_c \subset D$:
   - in set S_c, determine intervals $[l_{min}, l_{max}], [a_{min}, a_{max}], [b_{min}, b_{max}]$.
   - place sub-cube as entries in LUT from step 5. where intervals intersect with LUT inputs.

6 IMAGE QUALITY ASSESSMENT

During the process of embedding data in digital images, it is to be expected that that image file will be modified in a certain way. This is no different than with any other digital image manipulation technique. The change in the image will result in a change of the image's visual appearance, or, in other words, image quality. To measure the level of change, one can take on two different approaches of image quality assessment (IQA):

1. Subjective perception - visual assessment done by human participants
2. Objective metrics - mathematical function used to quantify the difference between the cover (reference) image $f(x, y)$ and the processed image $g(x, y)$

Both approaches have their advantages and disadvantages. The subjective approach offers a real-life assessment, as the human perception of the image's visual quality should, in the end, be the definitive reference point. On the other hand, using an objective approach offers a fast and simple way to quantify large number of images, and a way to compare results with previous work.

However, no objective metric can provide a perfect simulation of the human visual system (HVS). With this in mind, this section will provide an overview of objective image quality metrics, as well as an in-depth explanation of two metrics used during this research.

6.1 Types of Image Quality Metrics

As digital images are subject to various distortions, there exist a large number of available IQA metrics. However, they can all be divided into two categories [117], [118]:

1. Full-reference approaches - these approaches focus on comparing a test image with the reference image. The reference image is considered to have perfect quality and presents the *ground truth*. Examples of full-reference approaches can be found when comparing images with embedded digital watermarks to original images [64], [119].
2. No-reference approaches - these metrics focus on the assessment of the test image exclusively. In other words, there is no reference image, or *ground truth*. No-reference approaches can also be referred to as *blind* [117].

As already implied above, full-reference approaches are more suited for evaluating data hiding methods. The presence of the reference image makes it possible to measure different levels of embedding, a key component of every such method. Cheon et al. recognize 33 different state-of-the-art IQA metrics - 28 of those are full-reference [120]. The authors

conclude their research by stating that no metric is universally superior, as the metric of choice should depend on the application at hand. Several full-reference IQA metrics are explained below.

Zhan et al. propose a metric using pixel classification based on structural variation [121]. The proposed metric evaluates the image quality by combining the distribution of structural variations and the degree of structural differences. The authors report results that are more in line with those from subjective evaluation, when compared to other state-of-the-art metrics.

Ding et al. consider two psychovisual properties for developing a novel metric in their work: anisotropy and local directionality [122]. They introduce a term called *directional anisotropic structure measurement (DASM)* to represent the structures of an image that are visually dominant. Using DASM, the authors are able to measure the degradations of those dominant image structures. The authors report that the metric achieves good performance, and correlates well with human perceptions.

Sun et al. offer a different approach for evaluating image quality [123]. Instead of using features from square patches of images, the authors use features from superpixels. Superpixel is a set of image pixels that share similar visual characteristics. By using superpixels instead of square image patches, the proposed *superpixel-based similarity index (SPSIM)* method is able to extract perceptually meaningful image information. Image quality value is calculated based on superpixel luminance similarity, superpixel chrominance similarity, and superpixel gradient similarity. The authors report that the proposed metric is competitive with other state-of-the-art metrics.

The preliminary research for this thesis has shown that, no matter how many IQA metrics exist, the majority of state-of-the-art papers use two metrics for evaluating their methods: Peak Signal to Noise Ratio (PSNR), and Structural Similarity Index Measure (SSIM). No other proposed IQA metric has displayed a clear advantage over PSNR and SSIM. In conjunction, PSNR and SSIM provide a valuable contrast when evaluating image quality degradation. In the rest of this section, both metrics will be explained in-depth.

6.1.1 Peak Signal to Noise Ratio (PSNR)

PSNR is a metric that calculates the ratio between the maximum possible signal power and the power of the distortion which affects the image's quality [118]. This ratio (difference) between images is computed in decibels. Due to wide dynamic ranges of signals (such as

images), the PSNR is calculated as the logarithm term of the decibel scale. In image processing, PSNR values usually trend between 30 dB and 50 dB, while values from 40 dB and up are considered *good image quality* [5], [118]. The mathematical expression of PSNR is given in the two equations below:

$$PSNR = 10log_{10}(f_{max})/MSE,$$

(38)

where f_{max} represents the maximum possible value of an image element (for 8-bit images, this value is 255), and

$$MSE = \frac{1}{mn}\sum_{x=0}^{m-1}\sum_{y=0}^{n-1}[g(x,y) - f(x,y)]^2,$$

(39)

where $f(x,y)$ represents the cover image, and $g(x,y)$ represents the test image.

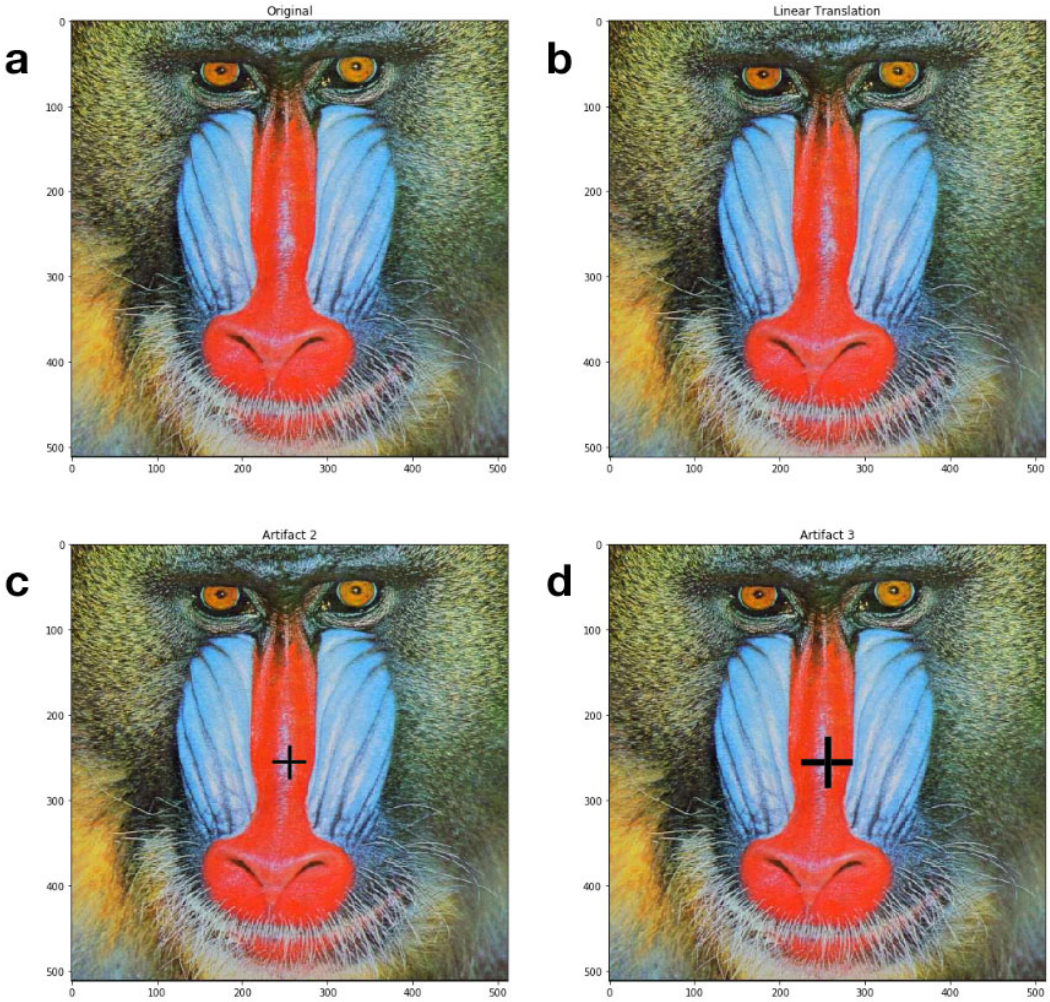

Figure 13. Illustration of PSNR's downside. a) Cover image; b) 5-pixel linear translation (PSNR = 15,56 dB); c) Artifact 1 (PSNR = 32,95 dB); d) Artifact 2 (PSNR = 29,17 dB);

During the last few decades, PSNR is the IQA metric of choice in the majority of relevant papers related to data hiding [5], [26], [28], [32]–[38]. Despite that fact, there are certain shortcomings of PSNR as a metric. Poljičak states that the main downside of PSNR is its low correlation with the subjective assessment of image quality [5]. To illustrate this, an example is given in Figure 13.

Even though the image in Figure 13. b) is visually much more similar to the original than the images c) and d) are, the PSNR value for image b) is the lowest. The reason behind these kind of results lies in the fact that PSNR takes each pixel from the original image and compares it to the modified image's pixel. In example b), the pixels are linearly translated, which means the original pixel value will not match the modified value in most cases. Regardless of the visual similarity, the resulting PSNR value will be low. On the other hand, in examples c) and d), most pixels stayed unchanged, hence the higher PSNR value. Numerous state-of-the-art papers also offer similar critiques, focusing on PSNR's lack of human visual system (HVS) features, and possibility of leading to wrong conclusions [6], [27]. When using PSNR as a metric, it is important to keep in mind these limitations.

6.1.2 Structural Similarity Index Measure (SSIM)

While PSNR is tightly connected to the mean squared error (MSE), and is considered to be strictly mathematical, SSIM is a perception-based model. Image degradation is considered as the change of perception in structural information [118]. The SSIM method is based on three major properties:

- Luminance - used to compare brightness between images
- Contrast - used to compare ranges between brightest and darkest image regions
- Structure - used to compare local luminance patterns between images

The mathematical expression of SSIM is given below [9], [118]:

$$SSIM(x,y) = [l(x,y)]^{\alpha} \cdot [c(x,y)]^{\beta} \cdot [s(x,y)]^{\gamma}, \quad (40)$$

where l represents the luminance, c represents the contrast, and s represents the structure; α, β, and γ are positive constants.

$$l(x,y) = \frac{2\mu_x\mu_y + C_1}{\mu_x^2 + \mu_y^2 + C_1},$$

(41)

$$c(x,y) = \frac{2\sigma_x\sigma_y + C_2}{\sigma_x^2 + \sigma_y^2 + C_2},$$

(42)

$$s(x,y) = \frac{\sigma_{xy} + C_3}{\sigma_x\sigma_y + C_3},$$

(43)

where μ_x and μ_y are the local means, σ_x and σ_y are the standard deviations, and σ_{xy} is the cross-variance for images x and y sequentially. C_1, C_2, and C_3 are positive constants used to avoid a null denominator.

Due to its perceptual properties, SSIM is often used in relevant literature - even though not as often as PSNR [26], [28], [29].

7 GRUBBS' TEST FOR OUTLIERS

7.1 Overview and history of Grubbs' test

Grubbs' test is a statistical test used to detect outliers in a normally distributed dataset. The test was introduced in 1950 by Frank E. Grubbs in his work "Sample Criteria For Testing Outlying Observations" [124]. Grubbs published two more papers and extended the proposed test in 1969 [125]. and 1972 [126]. Grubbs' test is sometimes also being referred to as "extreme studentized deviate" (ESD) [127] and "maximum normalized residual test" [128]. Typically, the Grubbs' test is used to detect the presence of one outlier within a population. If there are no outliers in the population, the hypothesis is rejected. If there is exactly one outlier in the population, the hypothesis is confirmed. The expression for the Grubbs' test is written as follows [129]:

$$G = \frac{\max_{i=1,\dots,N} |Y_i - \bar{Y}|}{s},$$

(44)

where G denotes the Grubbs' test, $\bar{Y}$ denotes the sample mean value, and s denotes the standard deviation.

Note that the expression above represents the two-sided version of the test, which is merely a combination of two one-sided versions expressed below:

$$G = \frac{\bar{Y} - Y_{min}}{s},$$

(45)

$$G = \frac{Y_{max} - \bar{Y}}{s},$$

(46)

where Y_{min} and Y_{max} denote the minimum and maximum sample values.

The Grubbs' test assumes an approximately normal (Gaussian) distribution of input data, and a normality test should be done beforehand [128].

To this day, Grubbs' test is being used in a wide variety of scientific fields. Ram B. Jain discussed its usage for environmental and chemical data [130]. Jain also proposed a recursive version of the test. The results are inconclusive, as the success rate of both tests depends on

the number of suspected outliers. Wu et. Al. Used the Grubbs' test to detect outliers in their work, predicting wind power using Radial Basis Functions [131].

There are other outlier tests that can be considered, apart from Grubbs' test – Dixon's test [132], and especially Tietjen-Moore's test [133], [134]. Tietjen and Moore presented their outlier test in the 1972 paper [133]. The test is not much different from Grubbs', as the title of the work suggests - *"Some Grubbs-Type Statistics for the Detection of Several Outliers"*. For purposes of this paper, preliminary experiments were done with all three outlier tests – Grubbs', Tietjen-Moore, and Dixon's. The results showed that, out of the three tests used in the experiments, the Grubbs' test is the optimal choice for watermark detection. Hence, the scope of this investigation will be limited to the Grubbs' test.

7.2 Grubbs' test for watermark detection

As these previous examples illustrate, the Grubbs' test is most often used to detect outliers and remove them from the population. However, in the context of digital watermarking, it can be used in a different way – the detected outlier can prove the presence of a watermark. Urvoy and Autrusseau used the Grubbs' test in this way in their 2014 work [135]. The authors used three previously proposed watermarking schemes and applied the Grubbs' test for detection purposes [136], [137], [138]. All three watermarking schemes were adapted to work on a similar principle – the area around the suspected position of the watermark is searched, and each position from the search area computes a cross-correlation value between the searched position and the actual watermark. After the search is complete, the decoder now has an array of cross-correlation values. If the suspected watermark was indeed within the searched positions, a single cross-correlation value should be significantly higher than the rest of the values. That central peak will be recognized as an outlier, and the presence of a watermark detected.

Apart from the cross-correlation data, the Grubbs' test takes another parameter as input – the significance level. In his 1969 work, Grubbs offers concrete "critical values" for each significance level, which represent the limit for a value being considered an outlier. Critical values will, along with the significance level, depend on the population size (the smaller the population size, the smaller the critical value). This means that, depending on the size of the dataset we are being given, different significance levels should be used. In their paper, Urvoy and Autrusseau tested significance levels from 10^{-2} to 10^{-8}, and the results showed that the best results are achieved when using $\alpha = 10^{-6}$ [6]. Poljičak et. al. found that, for a similar type

of watermark, the detection was most successful when using $\alpha = 10^{-4}$ [64]. In the context of watermark detection, the significance level is crucial for detection performance. If the chosen α is too high, the test will likely detect non-outlying values, resulting in False Positives. On the other hand, low significance levels will minimize those False Positives, but will, in turn, fail to detect some outliers, resulting in False Negatives.

As is with most statistical methods, this adjustable nature of the Grubbs' test via different significant levels makes the test suitable for watermark detection. By changing the α, the test is able to adapt to different watermarking schemes, frequency bands, and signal strengths. This is proved in [64], where the authors compared decoder performances when using a fixed threshold, against those when Grubbs' test was used for detection. The results showed that, for the proposed method, decoder performance was superior when using Grubbs' test, in comparison to using a fixed threshold. However, when using the same α with the method proposed in [6], the Grubbs' test negatively affected the decoder performance, in comparison to using a fixed threshold. This further proves that each watermarking method requires different approaches when implementing the Grubbs' test, depending on the input data, signal strengths, etc. This is explained in length in [135], where authors note "large differences between the experimented watermarking schemes and the resulting cross-correlation data." This makes it impossible to set a fixed threshold for watermark detection in advance, as the threshold would be apparent only in hindsight. With the implementation of the Grubbs' test, it is ensured that, with a proper significance level α, a single peak cross-correlation value will be recognized as an outlier, which in turn leads to a positive watermark detection.

8 IMAGE ACTIVITY

With the growing number of available digital images in online libraries, image feature extraction has become a popular topic. Understanding how individual features affect the final image can be of great important in a wide variety of fields, such as crime prevention, publishing, medicine, and architecture [139]. that In their work, Verma and Raman state that extracting features from images and categorizing these images is curently one of the main technical challenges [140]. Ojala et al. propose a local binary pattern (LBP) for extracting image features [141]. LBP is based on local intensity pixels, where the values of the central pixel of a chosen neighborhood are measured. In the context of data hiding, image feature extraction can help enhance a method's performance. More precisely, when embedding data in the frequency domain, the image's activity often plays a crucial part.

Previous tests have shown that methods similar to the one presented in this thesis are less likely to perform at a high level if the cover image is low in activity. To avoid such sub-optimal instances, in this thesis, the difference between pixels in a $5 \, x \, 5$ kernel is used as a measure of image local activity. More precisely, the local difference is calculated by substracting the mean pixel value of the chosen $5 \, x \, 5$ neighborhood from the central pixel value. This local difference calculation is done accross the entire image, and the final difference value is obtained by calculating the standard deviation of all local difference values.

Preliminary tests have shown that a successful data embedding is possible only for large enough local difference values. Hence, in the presented method, only those image blocks that possess a high enough local difference value are considered suitable candidates for data embedding. The implementation of activity testing within the presented method is further discussed in Section 10.1.1.

9 ERROR CORRECTION CODES

In a broad context, error correcting codes (ECC) have a purpose of ensuring error-free data transmission through communication channels [142]. Intuitively, it can be concluded that the same purpose can serve steganographic systems, as they need to guarantee integrity of information.

The history of ECC dates back to 1950, when R.W.Hamming published his work titled *„Error Detecting and Error Correcting Codes“* [143]. Hamming begins his work by stating that the motivation behind the study was led by *„a consideration of large-scale computing machines in which a large number of operations must be performed without a single error in the end result”.* He continues by saying that the problem of *„doing things right“* on a large scale is not particularly new, but digital systems have introduced the reality where a single error results in *„complete failure”.* The findings in this work are considered a backbone of future ECC systems, with several terms still being used today - Hamming distance, Hamming weight, Hamming bound, Hamming code, etc [142], [144], [145].

A full decade later, in 1960, I.S.Reed and G.Solomon published the paper *„Polynomial Codes Over Certain Finite Fields“* [145]. Due to its high complexity, in relation to Hamming codes, Reed-Solomon codes are the most popular ECC in practical use today [146]. Still, it is important to note that the Reed-Solomon codes require significantly more computational power and are harder to implement than Hamming codes. Since the publishing of the two major works mentioned above, there have been numerous papers published with proposed adjustments and improvements to existing ECC systems [146]–[148]. However, all those proposed systems rely on initial findings from Hamming, Reed and Solomon.

9.1 Hamming codes

A Hamming code can detect up to two errors in a message. Redundant bits are nested inside the original message at specific positions, making possible the error detection and correction. The encoder of a Hamming code is consisted of three steps [149]:

1. **Calculating the number of redundant bits**

   The expression below is used to calculate the number of redundant bits in a message:

$$2^r \geq m + r + 1,$$

(47)

where r represents the number of redundant bits, and m represents the number of bits in a message.

2. **Positioning the redundant bits**

   Redundant bits are positioned at bit positions of powers of 2. They are usually referred to as r_1 (at position 1), r_2 (at position 2), r_3 (at position 3), etc.

3. **Parity checking**

   At this step, the parity of each redundant bit is determined. *Even parity* implies that the total number of bits in a message is even, while *odd parity* implies an odd number of bits in a message.

The decoder of a Hamming code is consisted of four steps [149]. Note that the first three steps are the same as in the encoder.

1. **Calculating the number of redundant bits**
2. **Positioning the redundant bits**
3. **Parity checking**
4. **Error detection and correction**

   The binary values of parity bits are used here to calculate its decimal equivalent. If the decimal value is zero, there is no error in a message. Otherwise, the decimal value detects the exact bit position where the error exists. For example, if the binary value of $r_1 r_2 r_3 r_4 = 1010$, its decimal value, which is 10, implies that, at the position 10, there exists an error. To correct the error, that bit's value is flipped.

9.2 Reed-Solomon codes

Reed-Solomon ECCs have a wide application in fields such as digital memory storage, high-speed data transmission, QR codes, broadcast systems, and satellite communications [150]. A Reed-Solomon RS code can be specified as:

$$RS(n, k),$$

(48)

where n represents the block length, and k represents the number of bits in a message.

The parity check is of size $(n - k)$. The expression below is used to calculate the maximum number of errors that can be corrected using the algorithm:

$$2t = n - k,$$

(49)

where t represents the maximum number of corrected errors.

At the decoder, a message is represented as a polynomial $p(x)$, multiplied with the generator $g(x)$. The message vector $[x_1, x_2, x_3, \dots x_k]$ is then mapped to a polynomial of a degree less than k, such that:

$$p_x(\alpha_i) = x_1,$$

(50)

for all $i = 1, \dots k$, where p is a primitive polynomial that generates all elements of α [151].

Finally, the encoded message $s(x)$ is calculated as:

$$s(x) = p(x) \times g(x)$$

(51)

$s(x)$ is sent from the endocer along with the generator polynomial $g(x)$.

Upon receiving the message $r(x)$, the decoder divides $r(x)$ by the generator polynomial $g(x)$. Two results are possible. If the following expression stands, there are no errors:

$$\frac{r(x)}{g(x)} = 0$$

(52)

If the following is true:

$$\frac{r(x)}{g(x)} \neq 0,$$

(53)

then the error is evaluated using the following expression:

$$r(x) = p(x) \times g(x) + e(x),$$

(54)

where $e(x)$ denotes the error polynomial. The error polynomial $e(x)$ provides the information about exact error positions.

With the development of computational power, Reed-Solomon codes have seen a rise in popularity. Furthermore, Reed-Solomon codes can be generalized for simplicity, performance, or computational efficiency [152]. On of the most popular such generalizations of Reed-Solomon codes are Reed-Muller codes [153], [154].

10 EXPERIMENTAL

10.1 Proposed method

The application of the data hiding method presented in this thesis can be found in protecting printed goods (primarily packaging) against counterfeiting. For this method to be successful, the primary condition is for it to be robust to the print-scan process. In other words, the method should be able to function properly even after printing, with the message detectable via scanning. The data hiding method proposed in this thesis is suited for printing. The print-scan communication channel is an aggressive image attack, with several attacks happening simultaneously. Image attacks such as blur, noise, rotation, scaling, and compression offer a good simulation of the print-scan process. Hence, the method should be robust to these image attacks.

Preliminary research has shown that the Discrete Fourier Transform is the optimal choice for developing a robust data hiding method suited for printing. Frequency-domain embedding offers a dipersion of energy across the entire image i.e., making the degradation of image quality lower. The abovementioned benefits are the reason why the DFT is chosen for the development of the proposed method.

The data hiding method presented in this thesis is an additive method, and the embedding of data is being done in the frequency domain. The embedded data is created using three variables:

1. A vector created using a pseudo-random number generator
2. Magnitude coefficients of the image's frequency domain
3. Implementation strength

The above can be described with this mathematical expression:

$$M_{\mathrm{V}} = M(1 + \alpha \times V),$$

(55)

where M_{V}, M, and V denote the marked image, the original image, and the embedding data, respectively. Finally, α represents the implementation strength.

10.1.1 Encoder

The data is embedded in an image at the encoder. The embedding is done in the magnitude of the image's frequency domain by multiplying its coefficients. The encoder can be divided into three separate phases:

1. Pre-embedding
2. Embedding
3. Post-embedding

All operations and phases are displayed in the block diagram in Figure 16. The rest of this section will explain the encoder's operations in-depth.

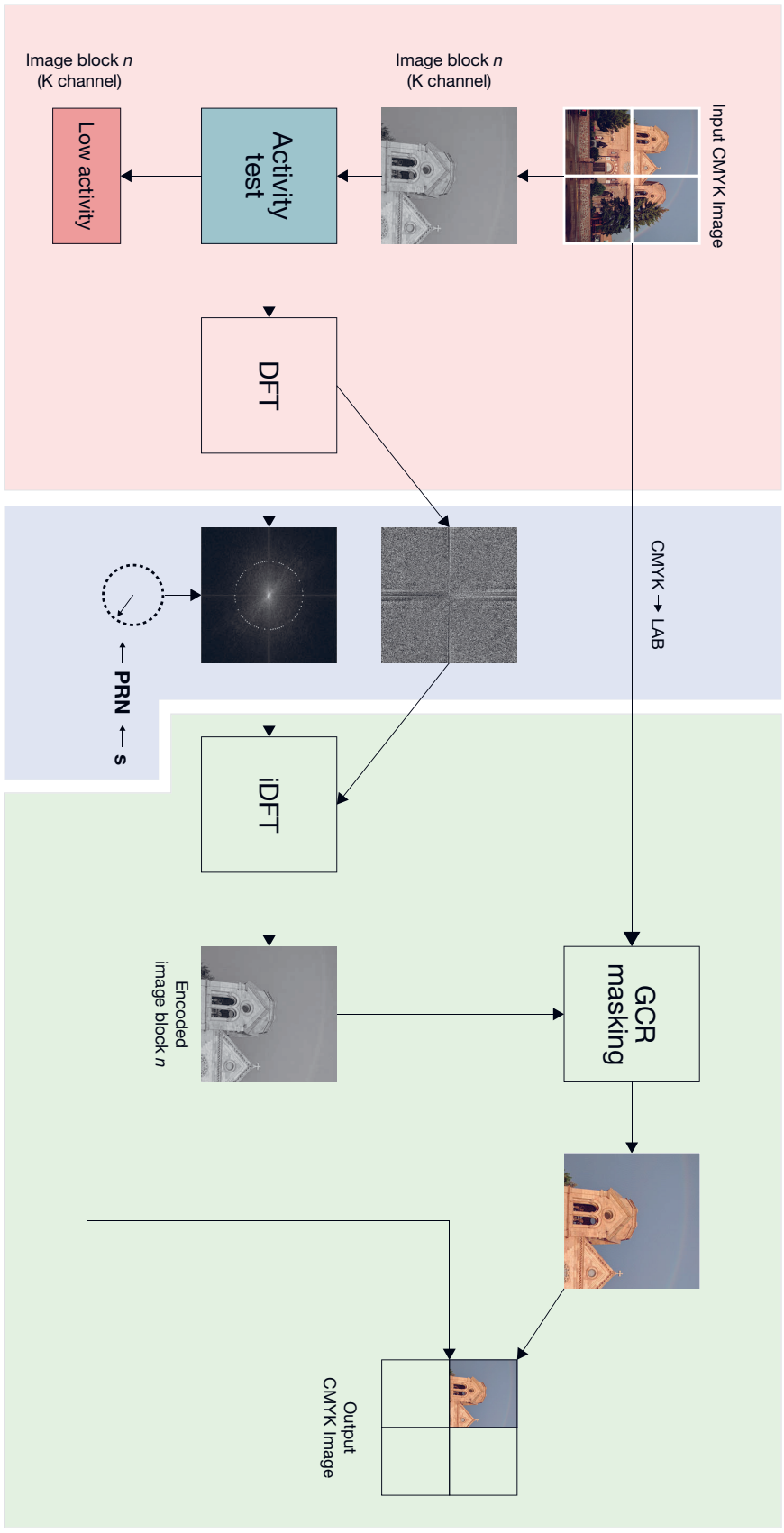

Figure 14. Schema of the encoder

10.1.1.1 Pre-embedding

The cover image $I(x, y, z)$ is in $cmyk$ color space. The dimensions of the image are $1024\ x\ 1024\ pixels$. First, the image is divided into a $4x4$ grid, thus creating a total of 16 image blocks $B_{cmyk}(x, y, z)$. Each image block is $256\ x\ 256\ pixels$ in dimensions, and a candidate for data embedding. The k channel of each image block $B_k(x, y)$ is then extracted, as only this channel will be used for data embedding.

Simultaneously, the $cmyk$ image block $B_{cmyk}(x, y, z)$ is transferred to LAB color space, to make possible the Gray Component Replacement (GCR), which is used to merge the k channel back to $cmyk$ color space after data embedding.

Each k channel image block $B_k(x, y)$ is tested for activity A. The activity A is defined as the mean standard deviation of the local difference d_l. The local difference d_l is calculated as a mean value difference between a central pixel, and each of its surrounding pixels in a $5\ x\ 5$ neighborhood. If the equation

$$A_{B_k(x,y)} > 10^{-4},$$

(56)

is true, image block $B_k(x, y)$ is considered suitable, and enters the embedding part of the encoder. Otherwise, image block $B_k(x, y)$ is considered not-suitable, and does not enter the embedding part of the encoder.

Preliminary tests have shown that this activity test is able to significantly improve detection, reducing bit error rate ϵ_r from an average of $\epsilon_r = 0.27$ to $\epsilon_r = 0.14$, while only slightly reducing the average capacity from 16 bits to 15.6 bits.

The Discrete Fourier Transform (DFT) is then used to transfer each image block to the frequency domain. As the DFT is a complex function, it is possible to calculate both the magnitude and the phase of the image. Data embedding is done in the magnitude of the frequency domain exclusively.

10.1.1.2 Embedding

The data embedding is completed with these two steps:

- The pseudo-random number generator (PRNG), using the secret key (seed) k and length l, generates a binary sequence (vector) of numbers v.

- Using the vector v, the radius r, and the image block $B_k(x, y)$ magnitude coefficients, the embedding data $D(x, y)$ elements are formed.

To understand the data generation within the presented method, an overview of PRNGs is provided below.

In 1890, the English statistician Francis Galton published an article titled *„Dice for Statistical Experiments"* [155]. A quote from the article reads:

"As an instrument for selecting at random, I have found nothing superior to dice. When they are shaken and tossed in a basket, they hurtle so variously against one another and against the ribs of the basket-work that they tumble wildly about, and their positions at the outset afford no perceptible clue to what they will be even after a single good shake and toss."

The quote illustrates how the scientific community perceived randomness at the time - as shear luck, chance. It was not until the 1950s that advantages were made in this field. In 1946, John von Neumann developed the first pseudo-random number generator (PRNG), and published it five years later, in 1951 [156], [157]. That same year, the *Linear Congruential Generator* was also developed [158]. Since then, many variations of PRNGs have been developed, such as *Lagged Fibonacci Generator, Blub Blum Shub, Marsenne Twister*, etc [5]. No matter the choice, each PRNG has a general form of:

$$X_i = f(X_{i-1}, \dots, X_{i-k}) \; mod \; p, i \geq 0, \tag{57}$$

where f is a function of the most recent k integers in the past and $X_{-k}, X_{-(k-1)}, \dots, X_{-1}$ are initial seeds taken from $\mathbb{Z}_p = \{0,1, \dots, p-1\}$. Different PRNGs have different functions f.

The *Mersenne Twister*, developed in 1998 by Matsumoto and Kurita, is still today the most used and widely spread PRNG [159], [160]. *Mersenne Twister's* development coincided with the technological advances and spread of internet, which made it even more popular throughout the years. This is also why *Mersenne Twister* is the default PRNG of most programming languages, including Python. Recently, there has been criticism of *Mersenne Twister*, but nevertheless, its popularity is still unparalleled [161].

The goal of PRNGs is to create a sequence of numbers that is hard to distinguish from a sequence of truly random numbers. With this in mind, a successful PRNG should satisfy these properties [160], [162]:

- high-dimensionality
- equi-distribution
- efficiency
- long period length
- portability

With the vector v, the radius r, and image block $B_k(x, y)$ magnitude coefficients, the embedding data elements $D(x, y)$ are formed using the expression below:

$$D(x_i, y_i) = v(j) \cdot \left(\frac{1}{9} \sum_{s=-1}^{1} \sum_{t=-1}^{1} M_k(x_i + s, y_i + t) \right),$$

(58)

where $j = (1,2,3, \dots, l)$, v represents the binary sequence (vector) of numbers, M_k represents the coefficients of image block $B_k(x, y)$ magnitude.

It can be seen from the expression that the elements of D depend on its neigborhood $M_k(x_i, y_i)$, where x_i and y_i are defined as:

$$x_i = \left(\frac{m}{2} + 1\right) + \left\lfloor r \cos\left(\frac{j \cdot \pi}{l}\right) \right\rfloor,$$

$$y_i = \left(\frac{n}{2} + 1\right) + \left\lfloor r \sin\left(\frac{j \cdot \pi}{l}\right) \right\rfloor,$$

(59)

where $j = (1,2,3, \dots, l)$, m and n represent the dimensions of image block $B_k(x, y)$, r represents the embedding radius, l represents the number of elements in the generated vector i.e., vector length, and the operator $\lfloor \cdot \rfloor$ represents the rounding to the nearest lower integer.

Once the embedding data is generated, the encoder modifies the magnitude coefficients using the expression below:

$$M_d(x, y) = M(x, y) + \alpha \times D(x, y),$$

(60)

where $M_d(x, y)$ represents the magnitude's modified coefficients, $M(x, y)$ represents the magnitude's original coefficients, and α represents the implementation strength.

Preliminary tests have shown that using different values for radius r can significantly affect the method's performance. By modifying r, different frequency ranges are affected. Four different r values are used: $LM = 0.15, M = 0.2, MH = 0.25, H = 0.30$. As seen in Figure 15., $r_{MH} = 0.25$ leads to lowest mean Bit error rate values. Thus, $r = 0.25$ is used as a default value in all experiments.

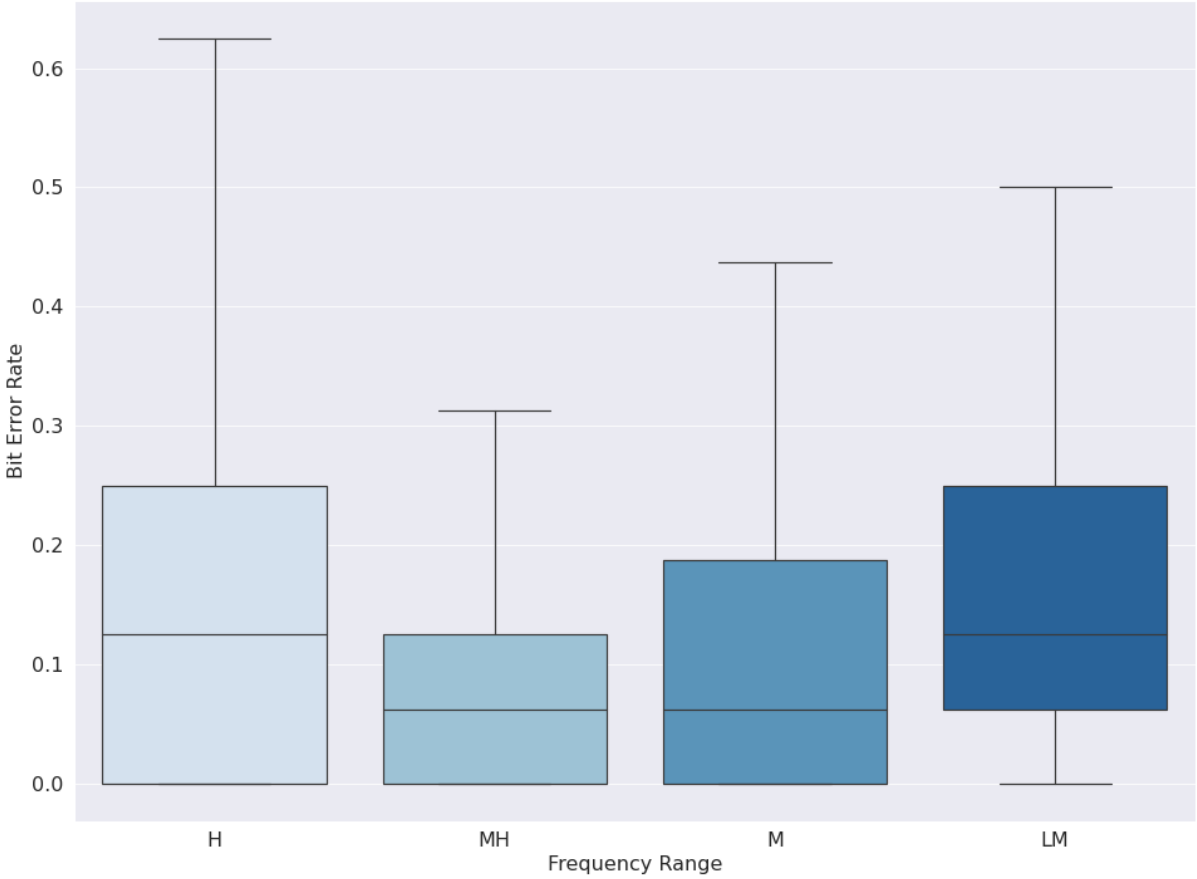

Figure 15. Bit error rate values for each of the four frequency ranges (r) used.

Implementation strength α is calculated for each image block $B_k^i(x, y), \{\mathrm{i} \mid 1 < \mathrm{i} < 16\}$. First, embedding is done using an implementation strength $\alpha = 100$. Then, the PSNR value between image blocks before and after embedding is calculated. If $PSNR = [35, 45]$, the embedding is concluded with the defined α. If $PSNR \neq [35, 45]$, then α is modified using the binary search algorithm. The same process is repeated until $PSNR = [35, 45]$ is true.

10.1.1.3 Post-embedding

The modified frequency domain's magnitude, depicted in Figure 16 is then, together with the phase, transformed back to the spatial domain using the inverse DFT (iDFT).

The embedded k channel of image block, $B'_k(x, y)$, is ready to be merged with c, m, and y channels into $B'_{cmyk}(x, y, z)$. However, before that merging is done, GCR masking is performed to hide embedding artifacts i.e., improve image quality.

GCR masking first takes the CIELAB color value for each pixel of the cover image block $B_{cmyk}(x, y, z)$, and the value of the corresponding pixel in embedded image block $B'_k(x, y)$. Then, the algorithm adjusts c, m, y values of $B_{cmyk}(x, y, z)$ in a way that, together with the k value from $B'_k(x, y)$, the resulting image block $B'_{cmyk_{gcr}}(x, y, z)$ reaches original CIELAB values of the cover image block $B'_{cmyk}(x, y, z)$. In other words, $B'_{cmyk_{gcr}}(x, y, z)$ possesses the same visual appearance as $B_{cmyk}(x, y, z)$, while also keeping the embedded data in its k channel intact.

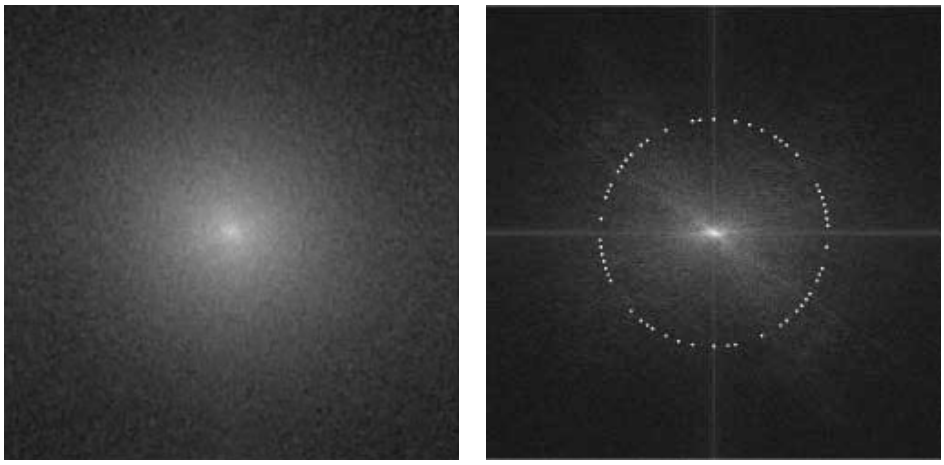

Figure 16. The magnitude spectrum before (left) and after (right) data embedding. (embedding strength is exaggerated for display purposes)

Finally, the output image $I'(x, y, z)$ is completed by merging all 16 image blocks $B_{cmyk_{gcr}}(x, y, z)$ together.

10.1.2 Decoder

The decoder of the proposed method is based on finding the correlation between these two vectors:

- The vector created using the PRNG
- The vector extracted from the image block

For the decoder to function properly, a priori knowledge of the following is needed:

- secret key (seed) k
- vector length l
- radius r

As the decoder does not require the cover image for detecting embedded data, the detection of this method is considered *blind detection*. The decision of whether the detection is positive or negative (if there is data present or not) is made based on the results from the Grubbs' test.

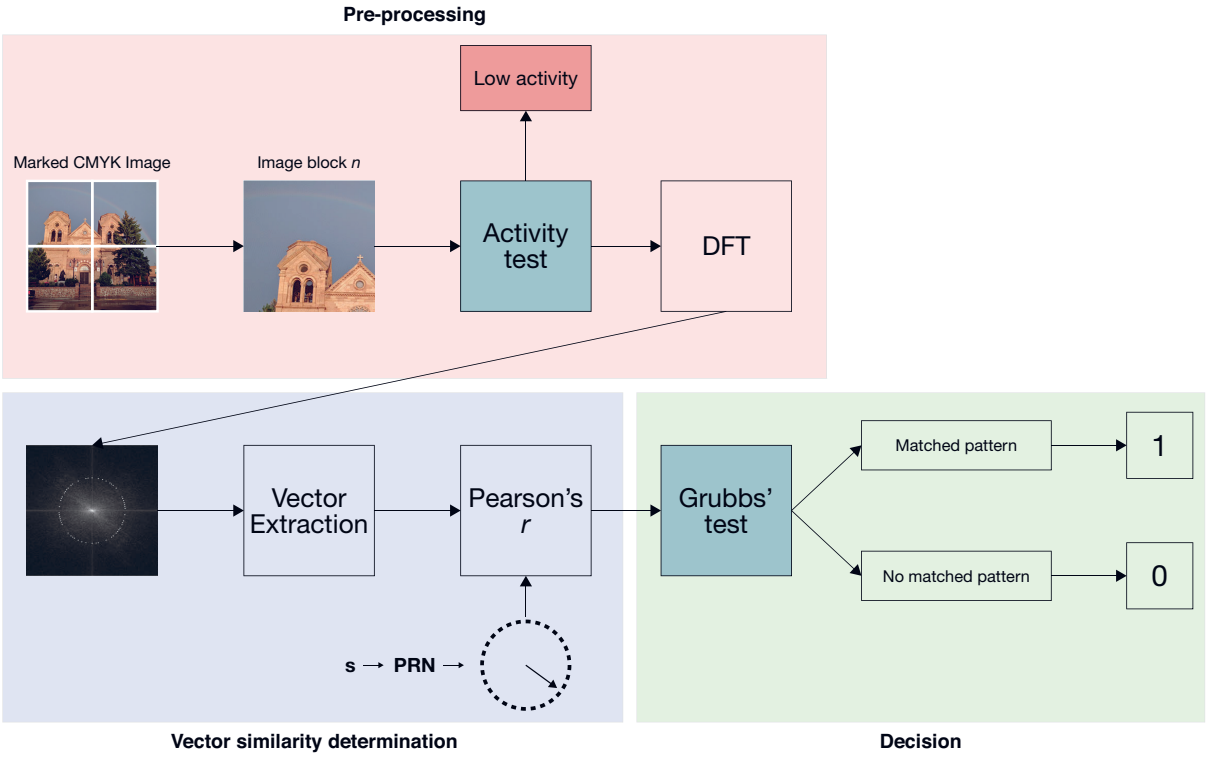

Figure 17. Schema of the decoder

The decoder can be divided into three separate phases:

1. Pre-processing
2. Vector similarity determination
3. Decision

All operations and phases are displayed in the block diagram in Figure 17. The rest of this section will explain the decoder's operations in-depth.

10.1.2.1 Pre-processing

The decoder receives a scanned and digitized version of the image $I'(x, y, z)$ as a grayscale image $I'_d(x, y)$. The received image is first resized to $1024 \times 1024\ px$ to match the image's dimensions at the encoder. This is crucial as the radius r is later used for vector extraction.

The image is then divided into a $4\ x\ 4$ grid, resulting in a total of 16 image blocks, each $256 \times 256\ px$ in dimensions.

Each k channel image block containing the embedded information, $B'_d(x, y)$, is tested for activity A using the same activity test as at the encoder. If image block $B'_d(x, y)$ is suitable, it enters the decoder. Otherwise, the block is regarded as not-suitable i.e., block $B'_d(x, y)$ has no data embedded.

Each suitable image block $B'_d(x, y)$ is transferred to the frequency domain using the DFT.

10.1.2.2 Vector similarity determination (detection)

With a priori knowledge of the secret key (seed) k, and the vector length l, the vector v is generated using the PRNG. Vector v is identical to the one generated at the encoder, and represents the reference point for comparison with the vector V_r to be extracted from the image block $B'_d(x, y)$. In other words, the correlation between these two vectors will directly lead to the decoder's decision.

As stated earlier, the decoder requires a priori knowledge of the radius r used for data embedding. However, various image attacks that occur through the communication channel can degrade the image, causing the location of the vector V_r to move from its original radius r. Thus, the decoder searches a certain area around the initial radius r, leading to higher probability of *„finding"* the vector V_r. Preliminary tests have shown that a radius search range of r_i, where $\{i \mid r - 32 < i < r + 32\}$, is the optimal choice.

For each radius r within the search range, the vector V_r is being extracted one element at a time. Each value from the element's 3×3 neighborhood is compared to the responding value of the vector v. The maximum value from the neighborhood is extracted. Mathematically, that value is expressed as:

$$V_{r_i}(k) = max\{Mv(x_i + s, y_i + t)\},$$

(61)

for $s, t \in [-1,0,1]$ and $k = (1,2,3, \dots, k_n)$, where $Mv(x_i, y_i)$ denotes the magnitude coefficient. The coordinates x_i and y_i are defined as:

$$x_i = \left(\frac{m}{2} + 1\right) + \left\lfloor r_i \, cos\left(\frac{k}{2sin^{-1}\left(\frac{1}{2r_i}\right)}\right)\right\rfloor,$$

(62)

$$y_i = \left(\frac{n}{2} + 1\right) + \left\lfloor r_i \, sin\left(\frac{k}{2sin^{-1}\left(\frac{1}{2r_i}\right)}\right)\right\rfloor,$$

(63)

for $= (1,2,3, \dots, k_n)$, where m, and n represent the image's dimensions, $r_i \in [r_{min}, r_{max}]$ represents the vector radius, and the operator $\lfloor \cdot \rfloor$ represents the rounding to the nearest lower integer.

The number of elements of the vector V_{r_i} depends on the vector radius r_i, and it is defined as:

$$k_n = \frac{\pi}{2sin^{-1}\left(\frac{1}{2r_i}\right)},$$

(64)

where k_n represents the number of elements in the vector V_{r_i}, and $r_i \in [r_{min}, r_{max}]$ represents the vector radius.

Prior to calculating the correlation between them, the two vectors are normalized to the interval $[0, 1]$. Also, the number of elements of both the generated vector v and the extracted vector V_{r_i} has to be equal to the number of elements of the longest extracted vector $V_{r_{max}}$. The maximum number of vector elements can be calculated using the expression below:

$$k_{n_{max}} = r_{max} \times \pi,$$

(65)

where $k_{n_{max}}$ represents the maximum number of vector elements, and r_{max} represents the maximum radius of extracted vectors.

The final $k_{n_{max}}$ is achieved by interpolating the extracted vectors using the *nearest neighbor* technique. When the vectors are normalized and have the same number of elements $k_{n_{max}}$, the correlation for each extracted vector v'_i and the generated vector r' is defined using

Pearson's product-moment correlation. Mathematically, the calculated correlation is expressed using the expression below:

$$C_{rv}(m) = \begin{cases} \sum_{n=0}^{k_n-|m|-1} \left(r'(n+m) - \frac{1}{k_n} \sum_{i=0}^{k_n-1} {r'}_i \right) \left({v'^*}_i - \frac{1}{k_n} \sum_{i=0}^{k_n-1} {v'^*}_i \right), m \geq 0 \\ C_{rv}^*(-m), m < 0 \end{cases}$$

(66)

where C_{rv} denotes the Pearson's product-moment correlation between vectors r and v, k_n denotes the vectors' length, and the operator $*$ denotes the complex conjugate.

Pearson's product-moment correlation coefficient for samples, otherwise known as Pearson's r or just r, is used to determine the strength and direction of a relationship between two variables [163]. Pearson's r is an estimate of the population paramether rho (ρ). If $\rho = 0$ for both variables, the variables are to be treated as independent. The correlation coefficient measures the tendency of two variables to change together in value - either to increase or decrease. To do this, the sums of products of two variables are divided by the degrees of freedom. The number of degrees of freedom for error in a sample is the number of chances for change and is usually defined as the sample size minus one. The general formula for the correlation coefficient r is given below:

$$r(x,y) = \frac{cov(x,y)}{\sigma_x \sigma_y}$$

(67)

where x and y represent the variables, σ_x and σ_y represent the standard deviation, and $cov(x,y)$ represent the covariance of variables x and y [164].

Pearson's r ranges from values of $r = -1$ to $r = 1$, with values close to zero representing weak relationships, and values close to 1 or -1 represent strong relationship - positive or negative. In theory, a value of $r = 1$ would assume a perfect positive correlation between two variables. A value of $r = 0$ assumes there is no relationship between variables.

10.1.2.3 Decision

After comparing all vectors V_{r_i} from the $r_i = +-32\ px$ search range to the original vector v, the decoder's decision is made using the Grubbs' test. As stated in Section 7, the Grubbs' test is used to detect outliers from a certain sample. In this case, the Grubbs' test is used to detect

a maximum-value outlier O_{max} from the array of calculated Pearson's r correlation values - each value representing the vector similarity within search range r_i. A significance level of $\alpha = 10^{-4}$ is used. If the Grubbs' test detects a single outlier O_{max}, the decoder's decision on the presence of embedded data is positive. If the Grubbs' test detects no single outlier O_{max}, the decoder's decision on the presence of embedded data is positive. This test is done for each suitable image block $B'_d(x, y)$, resulting in a maximum of 16 detected bits of information per image.

11 RESULTS AND DISCUSSION

11.1 Preliminaries

Table 3. List of image attacks and their respective parameters used in the experiments

	Image Attack	Parameters
1	GCR Masking	-
2	Scaling	Scaling factor α : 0.25 , 0.5 , 0.75 , 1.5 , 2
3	Rotation	Rotation angle: 0.25° , 1° , 3° ,5° , 10° , 15°
4	Blur	Kernel size: $3\ x\ 3$, $5\ x\ 5$, $7\ x\ 7$, $9\ x\ 9$, $11\ x\ 11$
5	Noise	Noise factor σ : 1 , 10 , 25 , 50 , 100
6	JPEG Compression	JPEG compression rate: 20 , 40 , 60 , 80 , 100
7	Print-scan	Image dimensions: $185\ x\ 185\ mm$
8	Unsharp masking	-
9	Error correction codes	*Hamming code, Reed-Muller code*

11.2 Tests in the digital domain

11.2.1 Images with and without GCR masking

Table 4. Bit Error Rate values - Images with and without GCR masking

	min	max	mean	median	std
No GCR	0	0.67	0.11	0.06	0.11
GCR	0	0.87	0.14	0.12	0.13

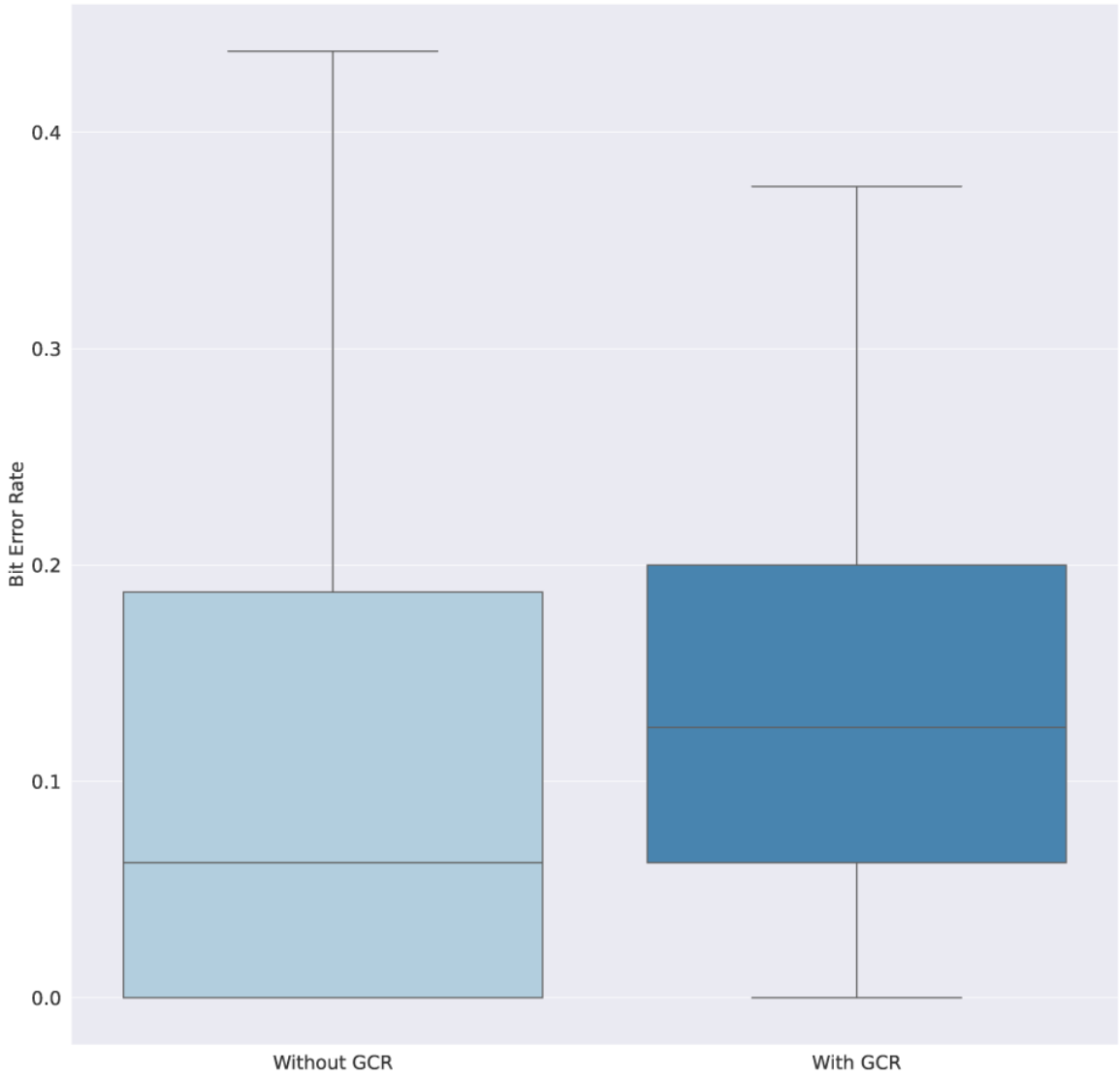

Figure 18. Boxplots displaying BER values for images with and without GCR masking applied.

Figure 19. An example of an image before (left) and after (right) the application of GCR masking.

In Table 4, Bit Error Rate (BER) values are displayed for images with and without GCR masking applied. A visual example of GCR masking is given in Figure 19. A mean value of 0.11 and median value of 0.06 for images without GCR masking are slightly lower than for images with GCR masking applied (0.14 and 0.12, respectively). Maximum value for images without GCR masking of 0.67 is also lower than a 0.87 value for images with GCR masking applied.

The boxplot in Figure 18. shows the same BER values as in Table 4. However, it is clear from the boxplot that, even though maximum BER values for images with GCR masking applies are higher, they can be considered outliers. With the exclusion of outliers from the data, the range of BER values is smaller for images with GCR masking applied.

While GCR masking, in theory, should not influence the detection in images, it is clear from the results that GCR masking, indeed, slightly influences detection. However, it is important to note that the BER values do not differ considerably when comparing images with and without GCR masking. Furthermore, the slight negative change in detection is more than reasonable when taking into account the significant positive influence of GCR masking in terms of image quality.

Table 5. PSNR values - Images with and without GCR masking

	min	max	mean	median	std
No GCR	18.57	18.57	47.93	37.46	6.59
GCR	52.35	66.3	60.11	60.18	1.26

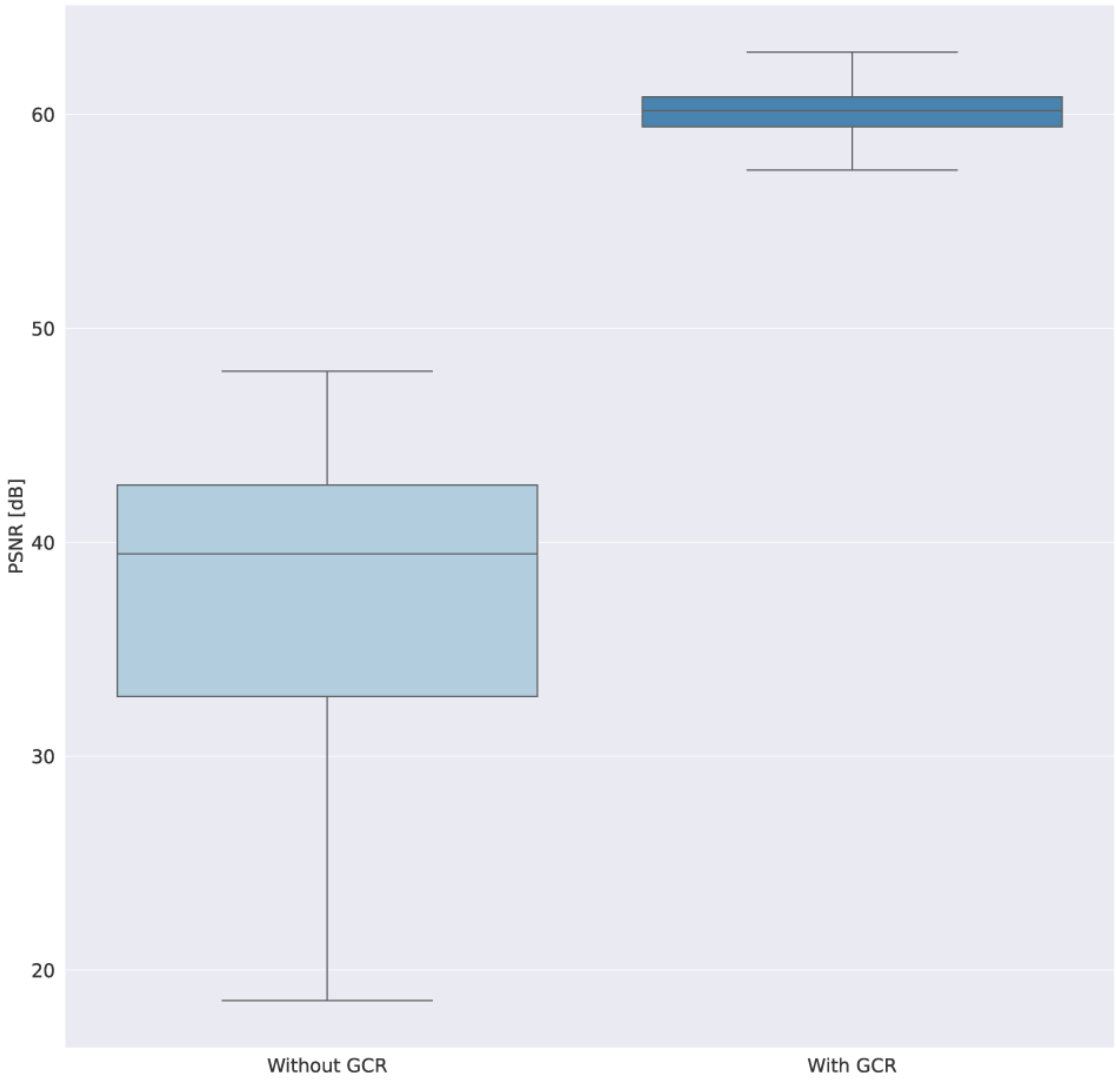

Figure 20. Boxplots displaying PSNR values for images with and without GCR masking applied.

In Table 5., Peak Signal to Noise Ratio (PSNR) values are displayed for images with and without GCR masking applied. All metrics clearly show a significant difference in image quality between images with and without GCR masking applied. Apart from a large difference in values for each metric, it is important to note that PSNR values for images with GCR masking applied are much more consistent, with a very high minimum value of 52.35.

The boxplot in Figure 20. shows the same PSNR values as in Table 5. From the boxplot, it is clear that GCR masking keeps the BER values consistent – around 60 dB. For images with no GCR masking applied, PSNR values are spread accross a much larger range.

Table 6. SSIM values - Images with and without GCR masking

	min	max	mean	median	std
No GCR	0.947	0.999	0.99	0.992	0.006
GCR	0.999	0.999	0.999	0.999	0

In Table 6., Structural Similarity Index Measure (SSIM) values are displayed for images with and without GCR masking applied. The results across the board follow the same trend as in Table 5., hence it can again be concluded that GCR masking significantly improves image quality, while keeping the values consistent.

The boxplot in Figure 21. shows the same SSIM values as in Table 6. The consistency of SSIM values for images with GCR masking applied is emphaszed even more when using SSIM as a metric, as values are virtually SSIM = 1.

The above results show that GCR masking indeed significantly improves image quality. Apart from the obvious fact that hiding (masking) artifacts itself is highly important for this data hiding method, the GCR masking's ability to maintain a consistent image quality also allows a stronger embedding signal, which in turn leads to higher detection rates.

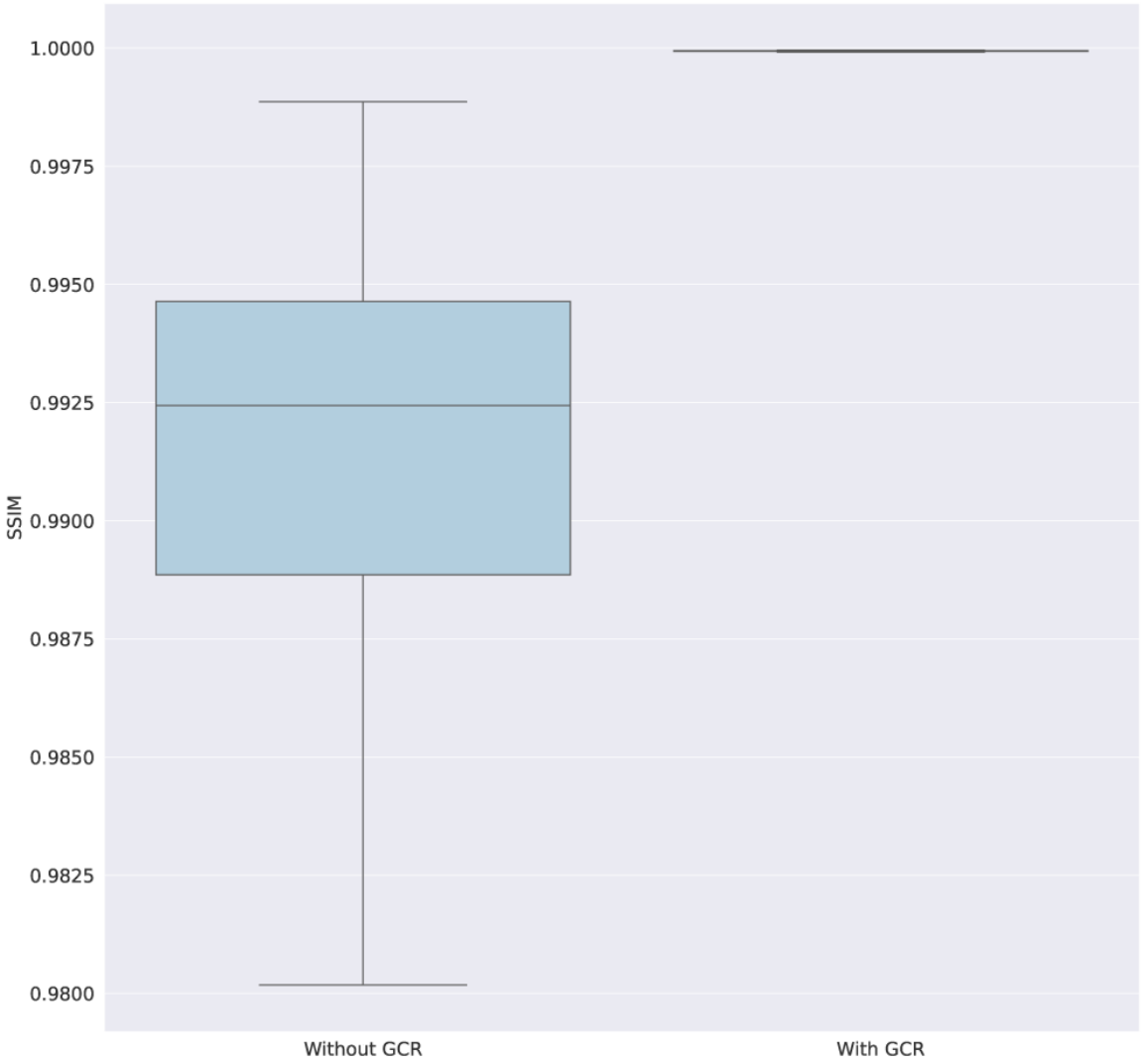

Figure 21. Boxplots displaying SSIM values for images with and without GCR masking applied.

11.2.2 Scaling

Table 7. Bit Error Rate values - Images scaled to certain dimensions

	min	max	mean	median	std
$\alpha = 0.25$	0.81	1	0.99	1	0.03
$\alpha = 0.5$	0	1	0.57	0.56	0.21
$\alpha = 0.75$	0	0.77	0.16	0.12	0.16
$\alpha = 1.5$	0	0.77	0.14	0.12	0.15
$\alpha = 2$	0	0.77	0.14	0.12	0.15

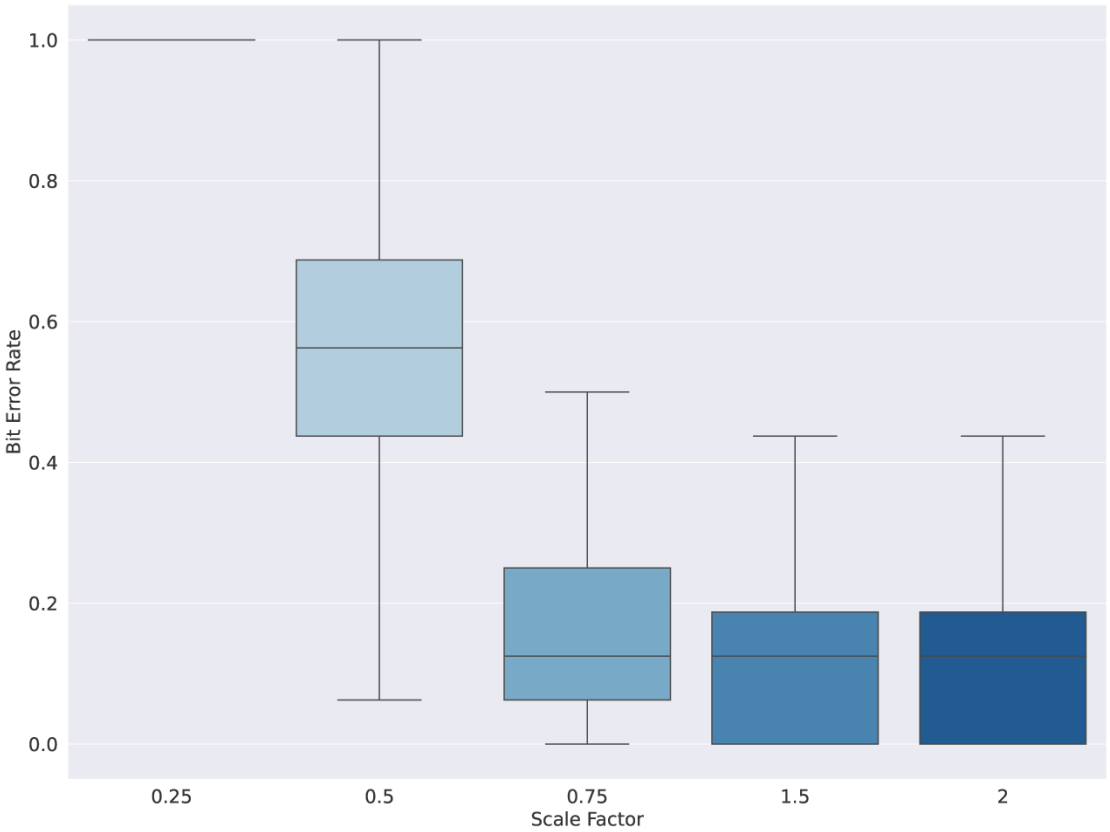

Figure 22. Boxplots displaying BER values for images after scaling.

Figure 23. An example of an image before (left) and after (right) scaling ($\sigma = 0.25$).

All images were originally $1024\ x\ 1024\ pixels$ *in dimensions. Scaling was done in a way so that each image was subject to either down-scaling or up-scaling (depending on the scale factor* α *), and then scaled back to its original dimensions. An example of an image before and after down-scaling is given in Figure 23*

Figure 23. The BER results displayed in Table 7. show that up-scaling images did not influence detection, as BER values stayed unchanged across the board. The same holds for down-scaling to $768\ x\ 768\ px$, where BER values also stayed unchanged accross the board. Down-scaling the image to $512\ x\ 512\ px$ resulted in significant loss of detection, with a mean BER value rising from 0.16 to 0.57, and a median BER value falling from 0.12 to 0.56.

However, detection was still possible to a certain degree. The most aggressive down-scaling, to $256\ x\ 256\ px$, resulted in high BER values. A mean BER value of 0.99 and a median BER value of 1 indicate that detection is practically impossible for such an aggressive down-scaling.

The boxplots in Figure 22. show the same BER values as in Table 7. Again, it is clear that the method is able to sustain up-scaling, and to a large degree, the down-scaling to $768\ x\ 768\ pixels$. More aggressive down-scaling factors result in severe loss in detection.

11.2.3 Rotation

Table 8. Bit Error Rate values - Rotated images

	min	max	mean	median	std
Rotation 0.25°	0	0.56	0.15	0.13	0.11
Rotation 1°	0	1	0.44	0.44	0.21
Rotation 3°	0.06	1	0.52	0.5	0.21
Rotation 5°	0.06	1	0.57	0.56	0.21
Rotation 10°	0.19	1	0.73	0.75	0.16
Rotation 15°	0.31	1	0.8	0.81	0.13

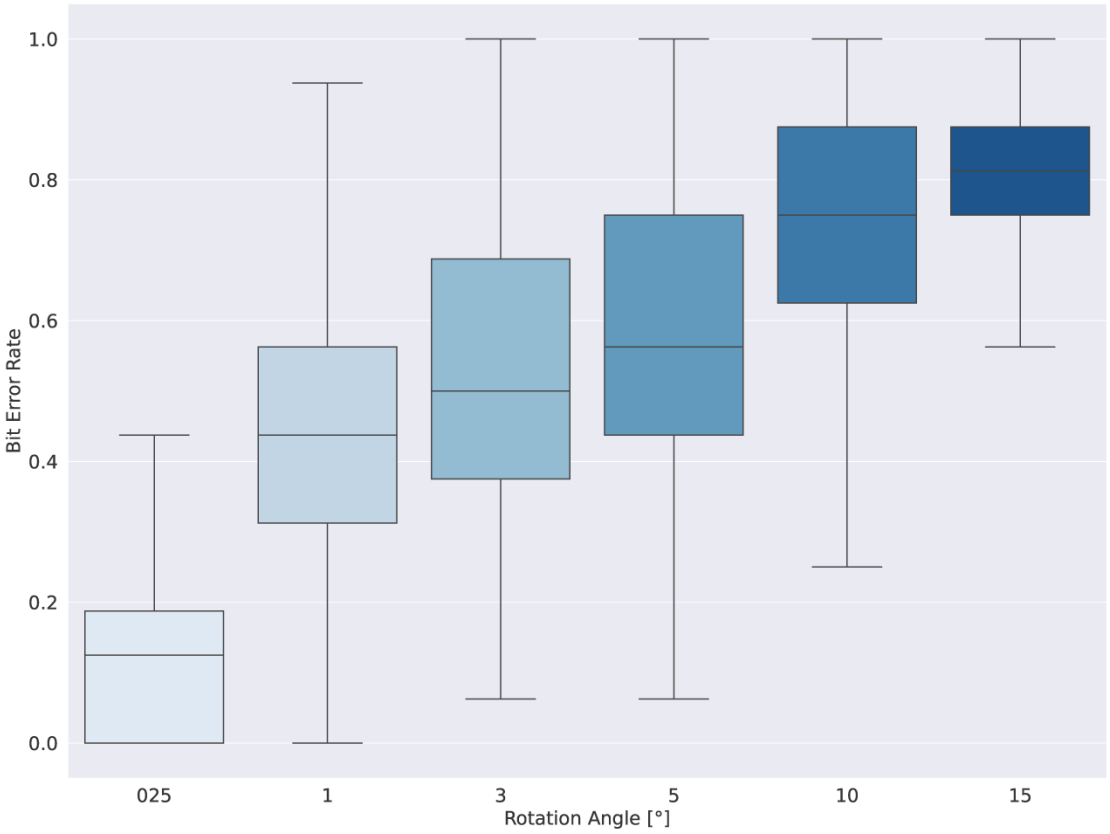

Figure 24. Boxplots displaying BER values for images after rotation.

Figure 25. An example of an image before (left) and after (right) rotation of $5°$.

The method's robustness to rotation was tested using angles ranging from $0.25°$ up to $15°$. These angles of rotation were chosen based on preliminary results, which showed that the method can not absorb rotation for larger angle values. An example of an image before and after rotation is given in Figure 25.

The BER values displayed in Table 8. show that the method is only able to successfully absorb the least aggressive rotation of $0.25°$. Mean BER value of 0.15 and a median BER value of 0.13 are in line with baseline values of 0.14 and 0.12, respectively. However, for each of the next rotation angles, BER values are considerably higher. Both the mean and median BER values for an angle of $1°$ are 0.44, while these values gradually rise to 0.8 for an angle of $15°$.

The boxplots in Figure 24. display the same BER values as in Table 8. It is clear from the results that the method is only robust to the slightest rotation angle of $0.25°$, while the BER rise significantly after more aggressive rotation angles.

These results show that the method is robust only to the slightest rotation angles. Even though the Discrete Fourier Transform is inherently robust to rotation, these results are to be expected due to the introduction of the block-based approach, where each block contains information. If the image is not aligned properly, the division into image blocks will result in individual blocks not being extracted properly. Finally, this will lead to an unsuccessful detection.

11.2.4 Blur

Table 9. Bit Error Rate values - Blurred images

	min	max	mean	median	std
$3\ x\ 3\ pixels$	0	0.69	0.14	0.12	0.13
$5\ x\ 5\ pixels$	0	1	0.44	0.44	0.23
$7\ x\ 7\ pixels$	0	1	0.76	0.81	0.2
$9\ x\ 9\ pixels$	0.25	1	0.92	0.94	0.1
$11\ x\ 11\ pixels$	0.69	1	0.97	1	0.05

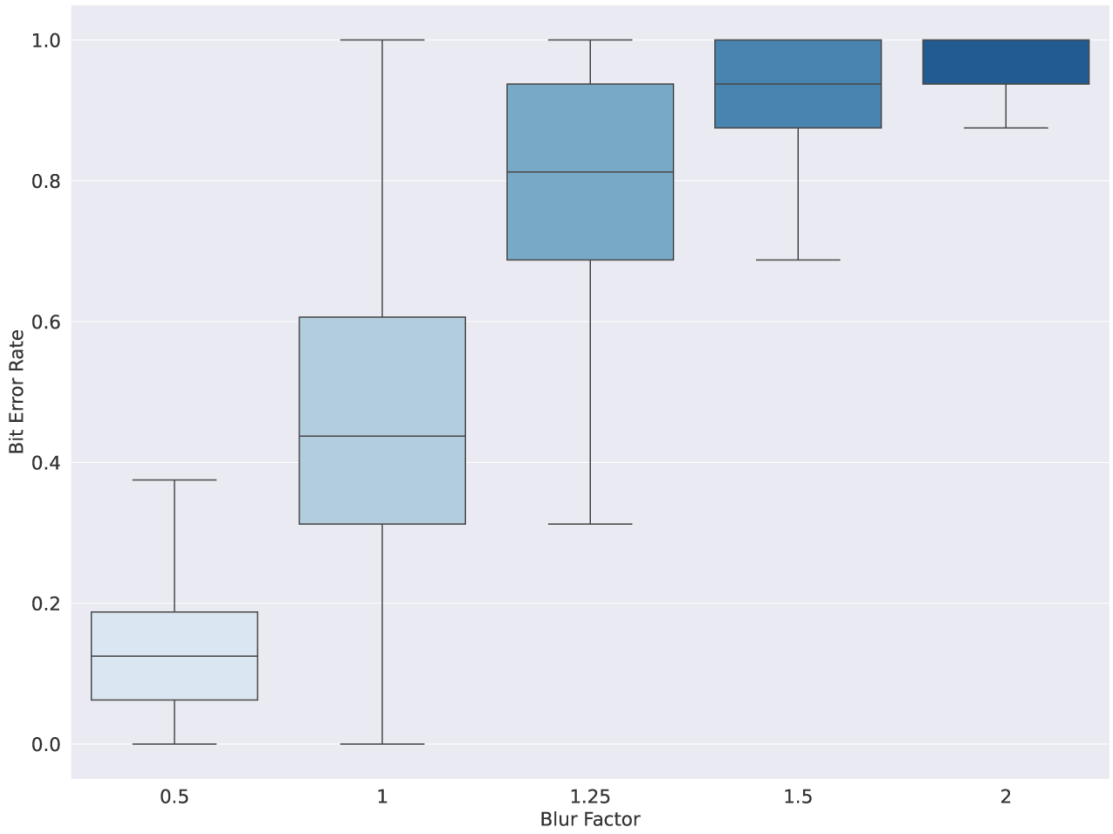

Figure 26. Boxplots displaying BER values for images after applying blur.

Figure 27. An example of an image before (left) and after (right) applying a 7×7 *blur kernel.*

The method's robustness to blur was tested using five different kernel sizes. The smallest kernel size used was 3×3 , and the largest kernel size was 11×11 . An example of an image before and after blurring is given in Figure 27. The BER values displayed in Table 9. show that the method is able to maintain baseline BER values for the 3×3 blur kernel. However, for larger blur kernels, BER values are higher. For a 5×5 blur kernel, mean and median BER values are 0.44. Those values rise to 0.76 and 0.81, respectively, for a 7×7 blur kernel. Mean and median BER values for 9×9 and 11×11 blur kernels are over 0.9, indicating that detection is almost non-existant.

The boxplots in Figure 26. display the same results as in Table 9. It can be concluded that the method is completely robust to the least aggressive blur kernel 3×3. For blur kernel 5×5, BER values are already significantly higher, while for all other blur factors, BER values are close to 1, indicating that the detection is almost entirely not possible.

These results indicate that the method is able to sustain the smallest blur kernel of 3×3 , and partially sustain the blur kernel of 5×5 . However, for larger kernel sizes, the BER values are close to 1, indicating that a successful detection is not possible.

11.2.5 Noise

Table 10. Bit Error Rate values - Images with added noise

	min	max	mean	median	std
$\sigma = 1$	0	0.77	0.14	0.12	0.15
$\sigma = 10$	0	0.87	0.22	0.19	0.2
$\sigma = 25$	0	0.94	0.37	0.37	0.25
$\sigma = 50$	0	1	0.58	0.62	0.26
$\sigma = 100$	0.38	1	0.9	0.94	0.12

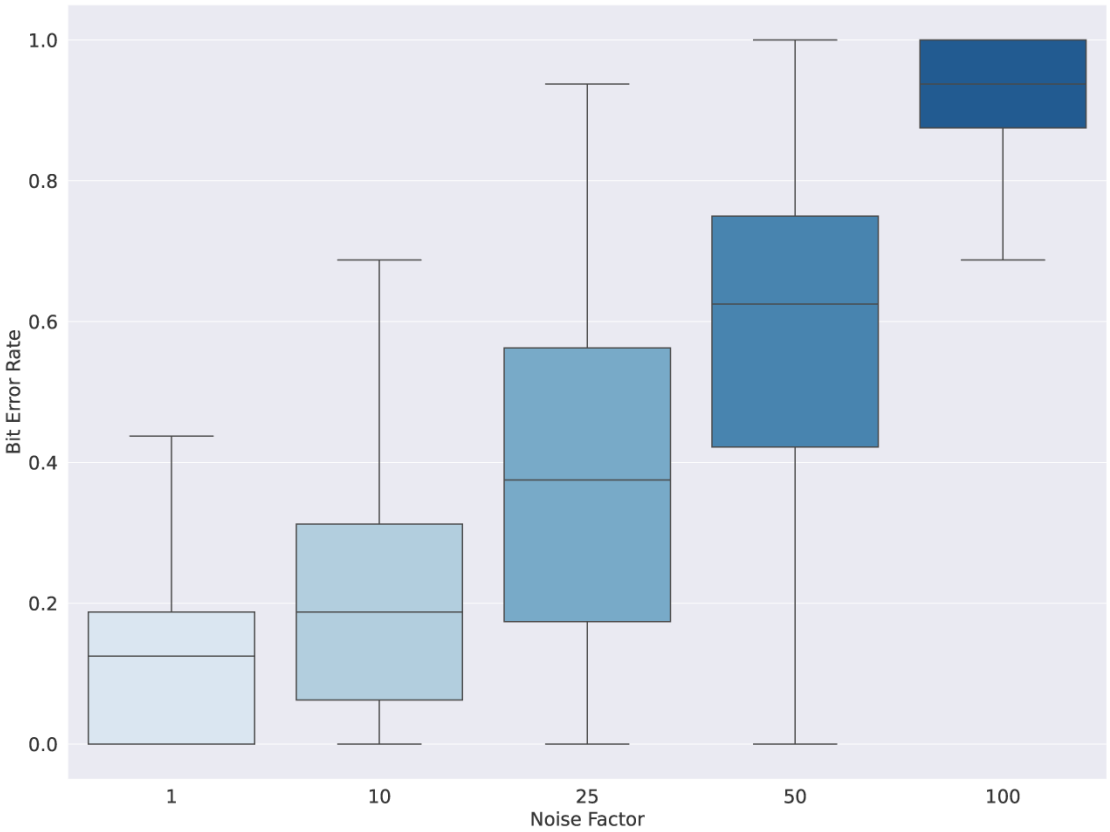

Figure 28. Boxplots displaying BER values for images after adding noise.

Figure 29. An example of an image before (left) and after (right) adding noise of $\sigma = 25$.

The method's robustness to noise was tested using five different noise factors σ . Noise was added to images using a Gaussian distribution, while the amount of noise added was controlled with the noise factor σ . σ denotes the standard deviation of the Gaussian distribution. An example of an image before and after applying noise is given in Figure 29. The BER values displayed in Table 10. show that the method is able to maintain BER close to baseline values for up to $\sigma = 10$. Mean and median BER values for $\sigma = 25$ are 0.37, showing a drop in successful detection. With larger noise factors, BER values rise even more, with mean and median BER values reaching over 0.9 for $\sigma = 100$ – indicating an almost non-existant detection.

The boxplots in Figure 28. display the same BER values as in Table 10. It is clear from the results that the method is able to sustain noise factors up to $\sigma = 10$, with BER staying close to baseline values. For more aggressive σ , the BER significantly rises.

Adding noise directly modifies image's frequencies, thus compromising the decoder's performance. Hence, for this method to function properly, only low levels of noise can be added.

11.2.6 JPEG compression

Table 11. Bit Error Rate values - images compressed with JPEG

	min	max	mean	median	std
JPEG 20	0	1	0.55	0.56	0.27
JPEG 40	0	1	0.34	0.31	0.24
JPEG 60	0	0.87	0.23	0.19	0.19
JPEG 80	0	0.75	0.16	0.12	0.14
JPEG 100	0	0.5	0.14	0.12	0.11

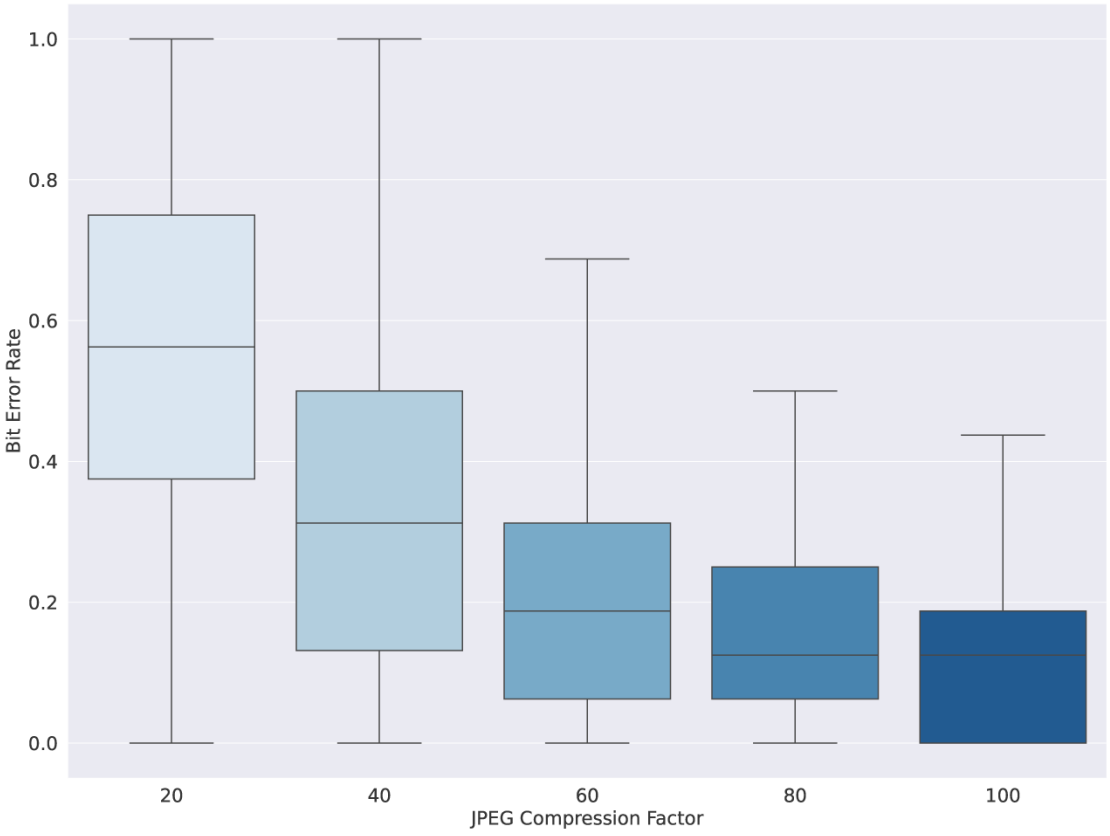

Figure 30. Boxplots displaying BER values for images after JPEG compression.

Figure 31. An example of an image before (left) and after (right) JPEG compression with a rate of 20.

The method's robustness to JPEG compression was tested using five different compression rates – 20, 40, 60, 80, and 100. An example of an image before and after JPEG compression is given in Figure 31.

The BER values for images after JPEG compression are displayed in Table 11. Mean BER values for compression rates 20, 40, 60, 80, and 100 are 0.14, 0.16, 0.23, 0.34, 0.55, while median BER values are 0.12, 0.12, 0.19, 0.31, and 0.56, respectively. The boxplots in Figure 30. display the same BER values as in Table 11.

The results show that the method is able to sustain JPEG compression rates 100, 80, and 60 with baseline or close to baseline values. For JPEG compression rates 40 and 20, BER values are higher, but a partially successful detection can still be achieved.

11.3 Print-scan

11.3.1 Unsharp masking

Table 12. Bit Error Rate values - images with and without the unsharp masking applied

	min	max	mean	median	std
Regular	0	0.87	0.25	0.25	0.19
Unsharp Mask	0	0.87	0.23	0.19	0.18

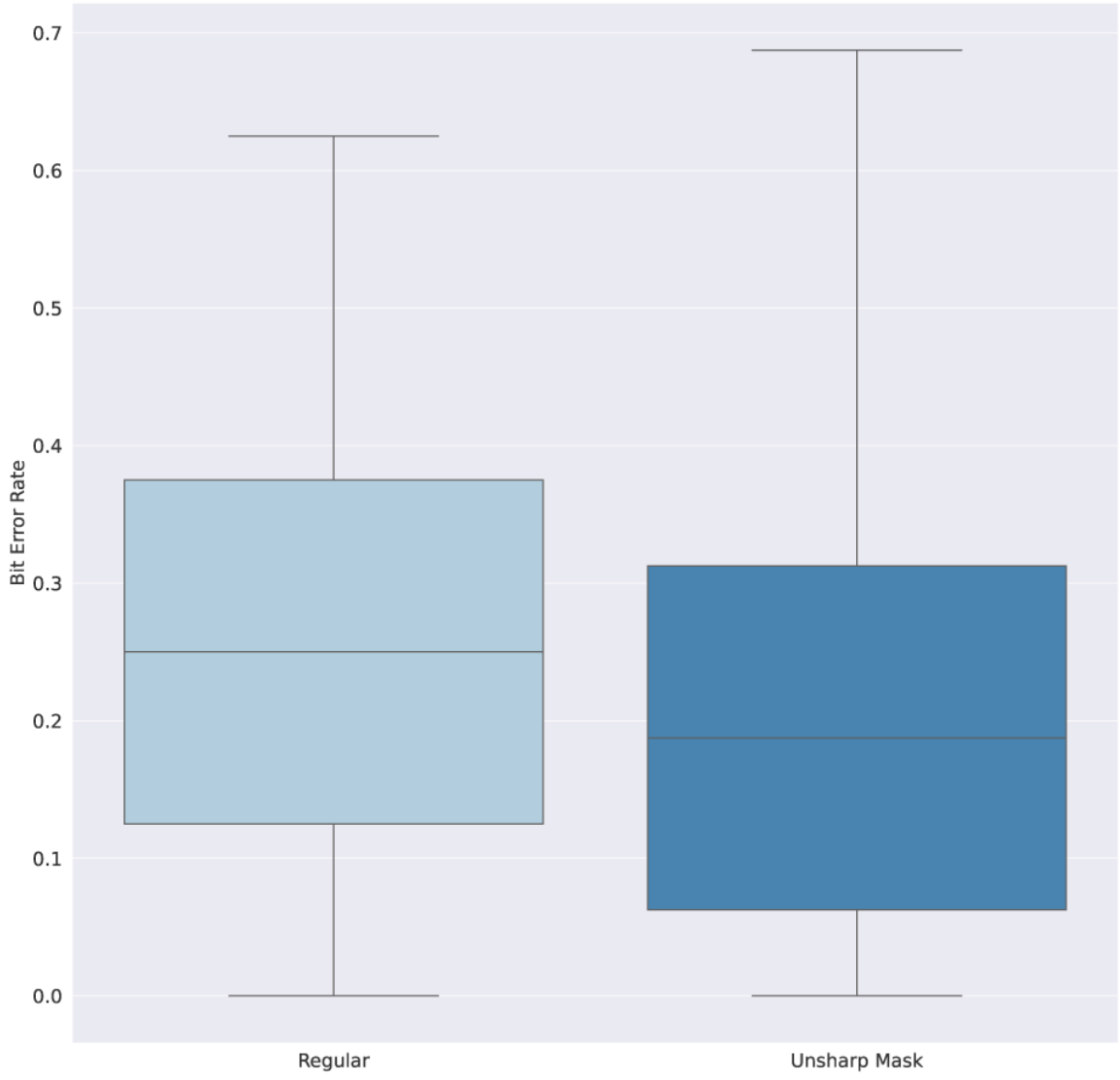

Figure 32. Boxplots displaying BER values with and without unsharp masking.

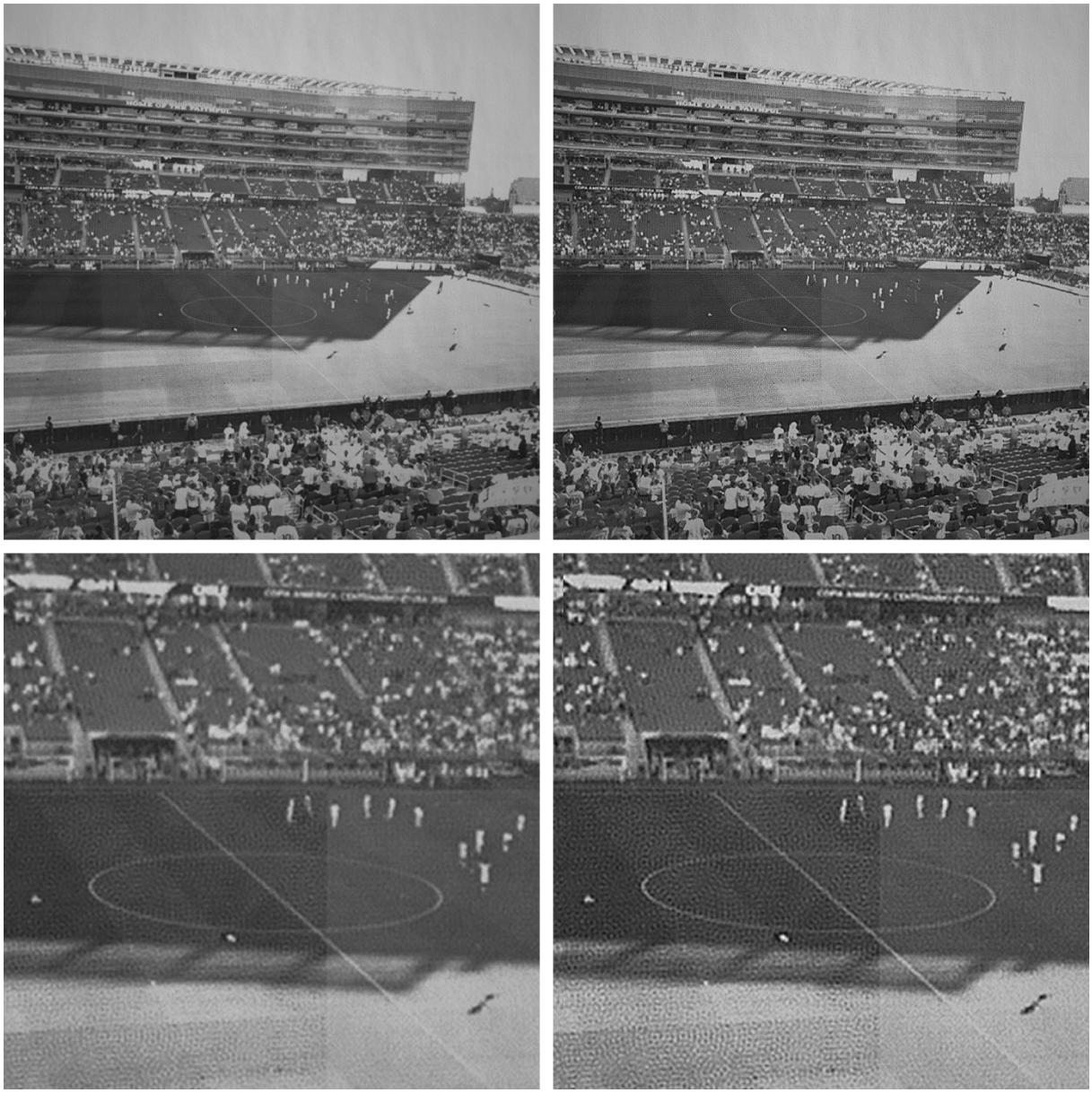

Figure 33. An example of an image before (left) and after (right) unsharp masking. The top row displays entire images, while the bottom row displays zoomed parts of images.

The method's robustness against the *print-scan* process was tested by printing image samples using a laser color printer Xerox Phaser 6510. Each image was $185\ x\ 185\ mm$ in dimensions. A custom color profile was made specifically for the printer. After printing, each image sample was scanned using the *Basler acA2440-35um* infra-red camera. The scanned image was then cropped to its final shape.

To prove the possibility of image processing tools improving detection, each image was also subject to *unsharp masking*. The results were then compared to those without *unsharp*

masking applied. An example of a scanned image with and without *unsharp masking* applied is given in Figure 33.

The BER values for images with and without the *unsharp masking* applied are displayed in Table 12. Mean and median BER values for images without *unsharp masking* applied are both 0.25, while those values are 0.23 and 0.19, respectively, for images with Unsharp Masking applied. The boxplots in Figure 32. display the same BER values as in Table 12.

From these results, it can be concluded that image procesing can indeed improve detection. In this case, *unsharp masking* was able to offset some limitations of the IR camera used during the experiment. The sharpness that is lost during the *print-scan* process compromising the detection, was partially successfully restored, thus improving the decoder's performance.

11.3.2 Error Correction Codes

Table 13. Bit Error Rate values - images with and without error correction codes applied

	min	max	mean	median	std
Regular	0	0.87	0.25	0.25	0.19
Hamming	0	0.64	0.17	0.18	0.16
Reed-Muller	0	1	0.2	0	0.39

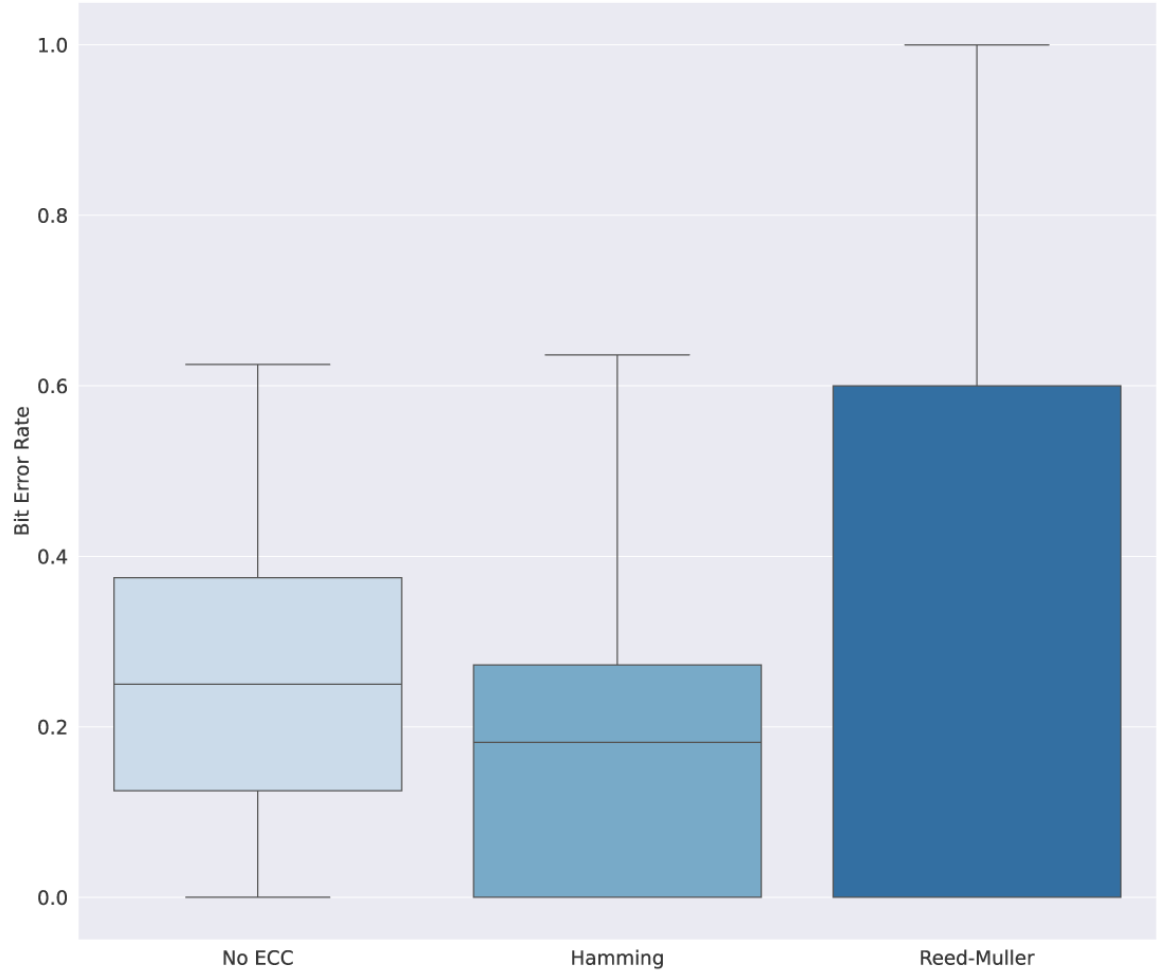

Figure 34. Boxplots displaying BER values with and without error correction codes.

To prove the posibility of *error correction codes* (ECC) improving detection, tests were conducted using two different ECC algorithms – *Hamming code* and *Reed-Muller code*. Each ECC algorithm required certain number of image blocks to be predefined as parity bit blocks, meaning that those blocks would not be taken into account when embedding the stego message at the encoder. Hence, using ECCs decreases the maximum possible capacity of the

method. The method’s baseline capacity is 16 bits, while that capacity fell to 11 bits when using the *Hamming code*, and to 5 bits when using the *Reed-Muller code*.

The BER values for images with and without ECCs are displayed in Table 13. It is clear that *Hamming code* indeed improve detection, with mean and median BER values dropping from 0.25 to 0.17, and from 0.25 to 0.18, respectively. When using *Reed-Muller code*, the mean BER value is 0.2, and the median BER value is 0. This indicates that for more than a half of tested images, the method was able to detect the stego message without errors.

The boxplots in Figure 34. display the same BER values as in Table 13. It should be noted that for *Reed-Muller code*, the boxplot includes a higher BER value range. This is due to its limited capacity, causing BER values to rise with even a small number of wrongly decoded bits.

It can be concluded that both ECC algorithms improved detection, with *Reed-Muller code* providing more significant positive influence. However, while the method’s decoder improved, it is important to note that ECCs limit the method's capacity. As with any other data hiding method, it is crucial to define the method’s priority. If the method’s capacity is the highest priority, using ECCs may not be the optimal choice. However, if the emphasis should be more towards a successful detection rate, using ECCs can indeed improve the method’s overall performance.

11.4 Subjective quality analysis

11.4.1 Setup and preliminaries

As mentioned earlier in Section 6, a subjective analysis of image quality can not be completely replaced with any IQA metric. For this reason, a subjective survey was conducted. A total of 144 image samples with embedded data were used in the experiment. Due to a high number of samples, images were divided into six groups, resulting in 24 images per group. Each observer was shown an example of a cover image before the start of the experiment, and then used a 4-step scale to assess the visibility of embedding artifats in each of the 24 image samples:

- Grade 0 - Imperceptible
- Grade 1 - Low
- Grade 2 - Moderate
- Grade 3 - High

The experiment was conducted in controlled lighting conditions, using the *GretagMacbeth (X-Rite) Judge II-S* viewing booth. Sixty-two observers participated in the experiment, resulting in a total of 1488 evaluated image samples.

The images were embedded with three different signal strengths, with a goal to further examine the effect of data embedding on perceived image quality. Six different image motives were used (displayed in Figure 37). Motives mostly featured skin tones and food, as these are some of the most prominent motives in packaging. Dark tones and light, smooth gradients were featured on two image motives as well, to include the most extreme possible cases in terms of image activity and presence of black color (k channel).

11.4.2 Results

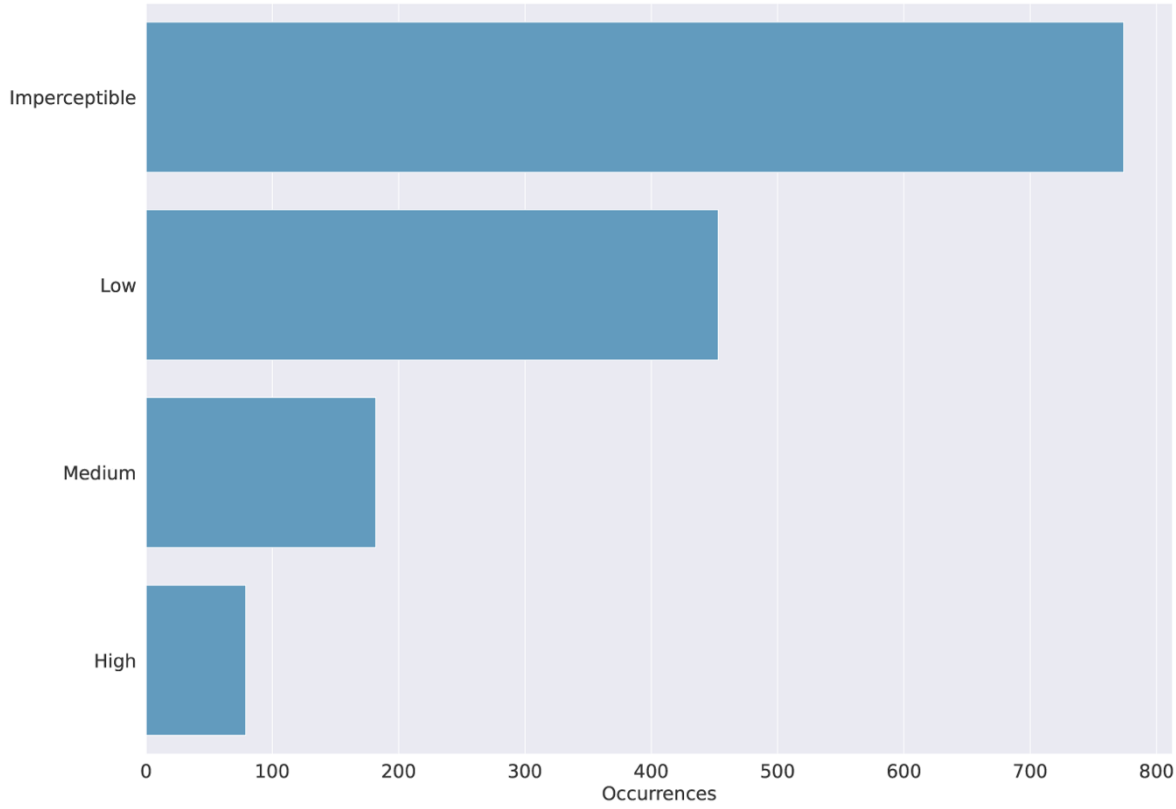

Figure 35. Total occurrences of each grade for images in the subjective experiment.

The total number of occurrences for each grade value is displayed in Figure 35. Out of 1488 total evaluations, 774 were valued as *imperceptible*. This means that in 49.7% of all image evaluations, observers did not see visible embedding artifacts. Another 453 evaluations were valued as *low*, meaning that the embedding artifacts were barely visible. By summing *imperceptible* and *low* grades, a total of 84.7% images had imperceptible or barely visible embedding artifacts. 182 images were valued as *medium*, and only 79 images were valued as *high*.

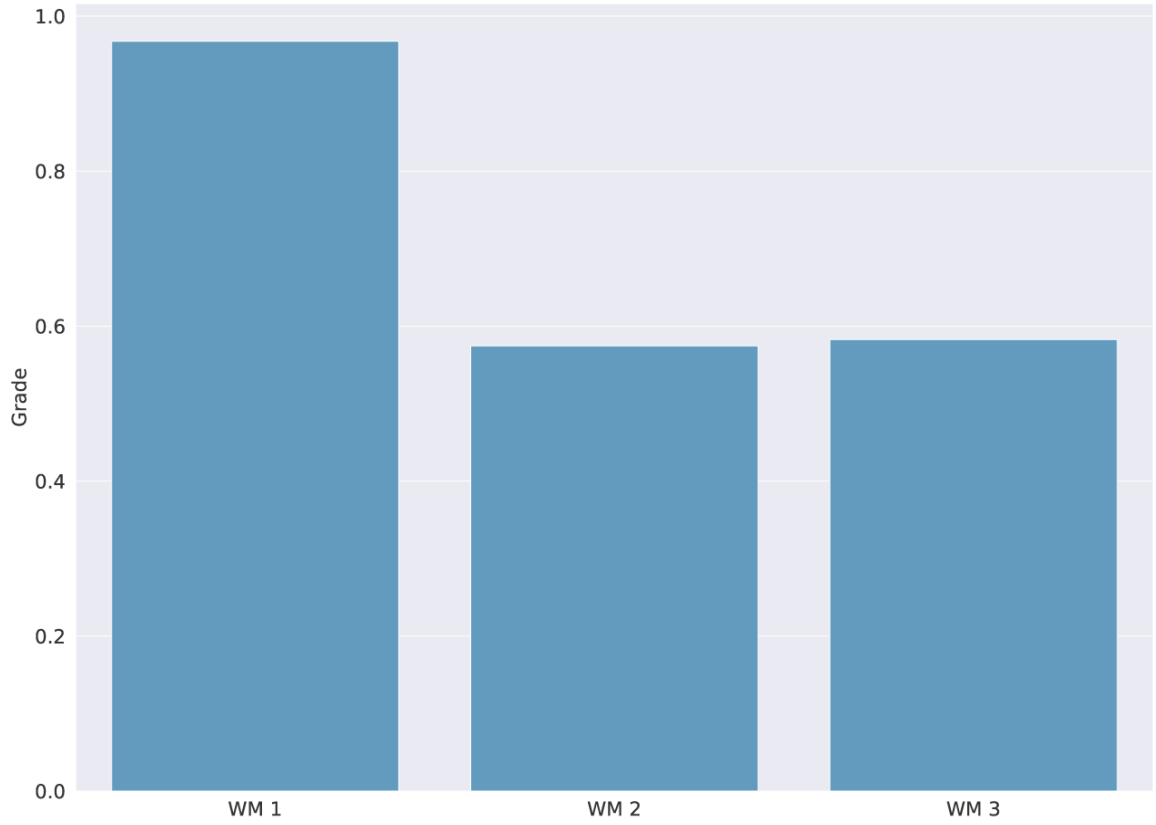

Figure 36. Mean grade value for each of the signal strength for data embedding. WM 1 – strongest signal; WM 3 – weakest signal.

Three different embedding signals were used in the experiment to examine how embedding strength affects perceived image quality. These results are displayed in Figure 36. It is clear that the strongest signal, WM1, achieved a significantly higher mean grade value than the two weaker signals, WM2 and WM3. The mean grade value for WM1 is 0.97, while the value is 0.57 and 0.58 for WM2 and WM3, respectively. It can be concluded that the strongest embedding signal should be avoided, as it results in considerably more visible artifacts. The other embedding signals used can be used freely, as they result in virtually the same level of imperceptibility.

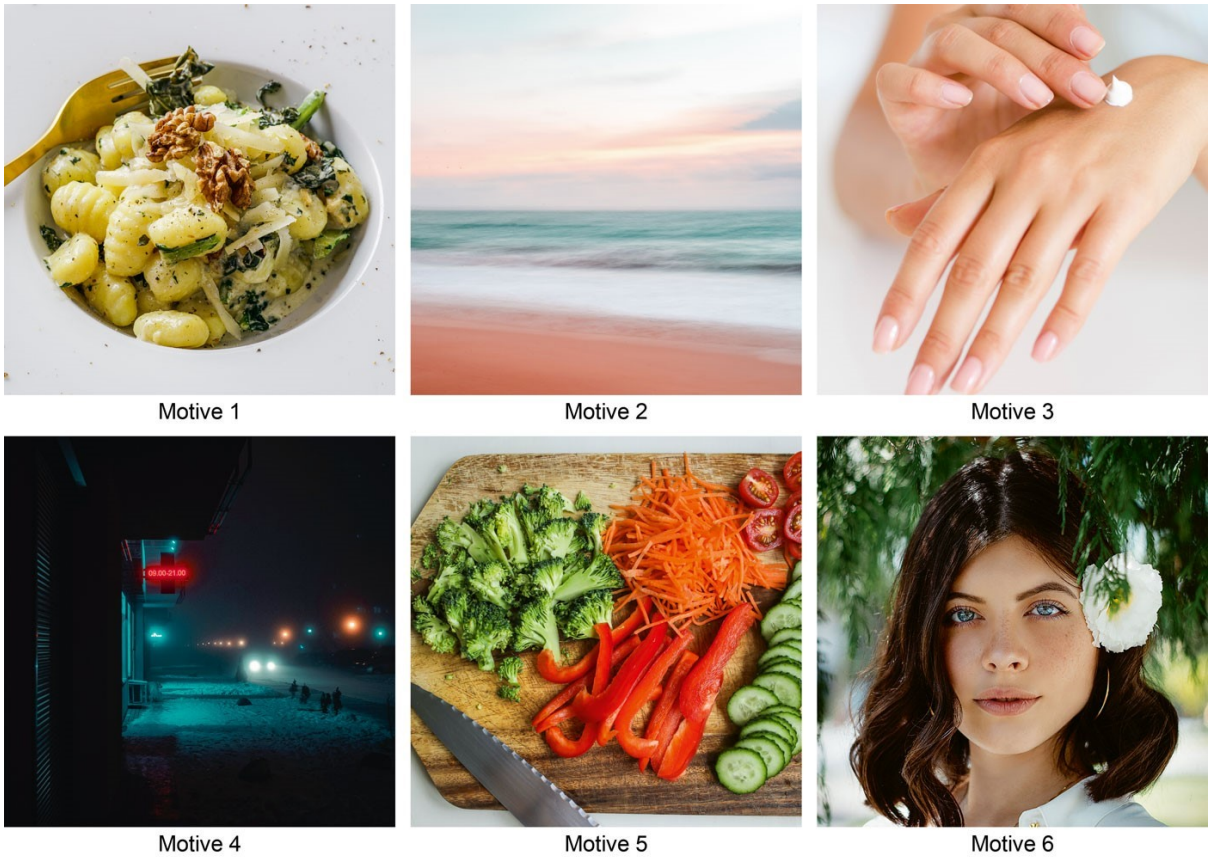

Figure 37. Image motives used in the subjective survey.

To examine how image-dependent the method is, six different motives were used in the experiment, as displayed in Figure 37. Results displayed in Figure 38. show that the mean grade value indeed depends on the evaluated motive. When evaluated, *Motive 2* achieved the highest mean grade value, followed by *Motive 3*. Upon closer inspection, it is clear that both motives are mostly light, with smooth gradients. The lack of black color makes the k channel of the image low in values, while the smooth gradients result in low activity in the frequency domain. Given that the embedding is done in the frequency domain of the image's *k channel*, these two image properties, in conjunction, make the required embedding signal significantly strong. Finally, using such a strong embedding signal results in higher image degradation and more visible artifacts. *Motive 5* achieved the lowest mean grade value. Similar conclusions can be drawn, as the image is clearly rich in high-activity areas. Hence, the embedding signal does not need to be as strong, resulting in less visible artifacts.

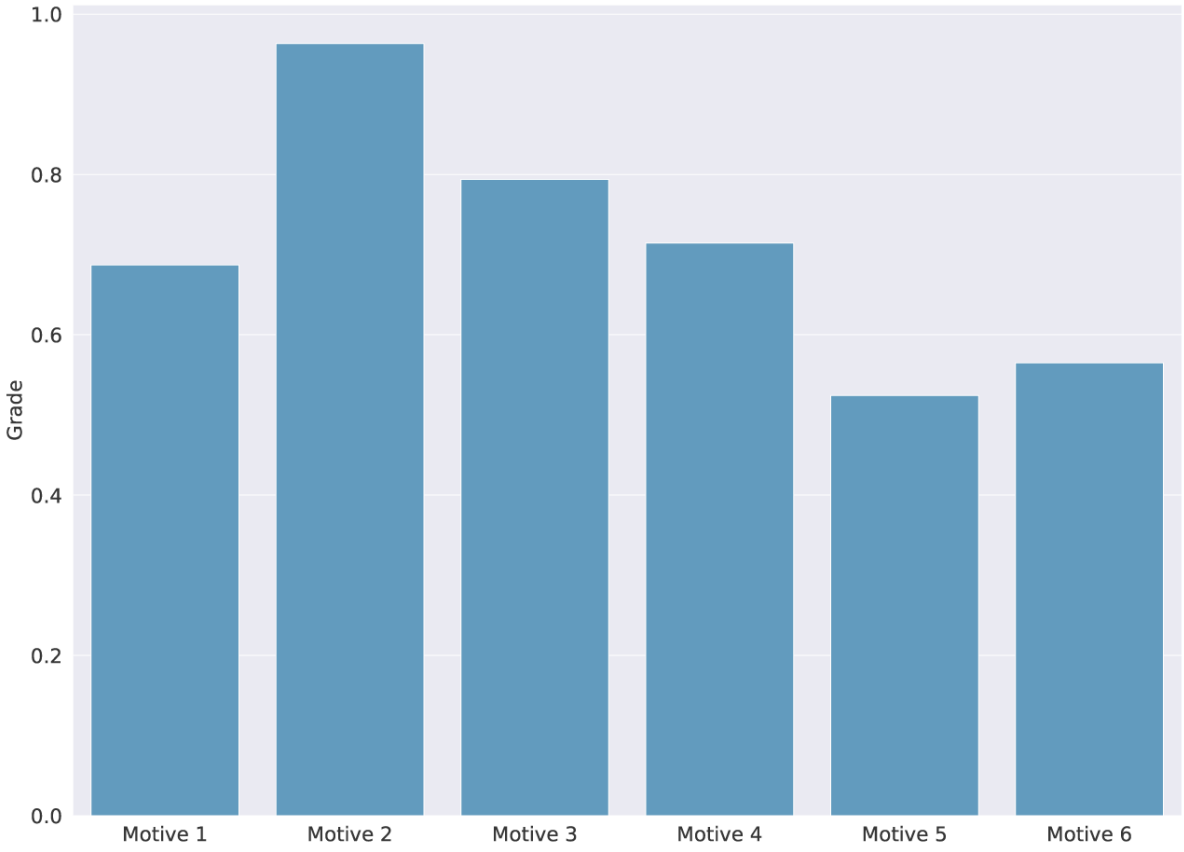

Figure 38. Mean grade values for each motive used in the experiment.

In terms of materials, two different paper types and four different paper varnishes were used in the experiment:

- Paper types
  - Standard offset paper *(kundsdruck)*
  - Ning
- Varnishes
  - No varnish
  - Water-based
  - Offset
  - Ultra-violet

The results in Figure 39. and Figure 40. show that, regardless of the varnish and paper used, mean grade values stay consistent. This means that the method can properly function with various industry-level materials.

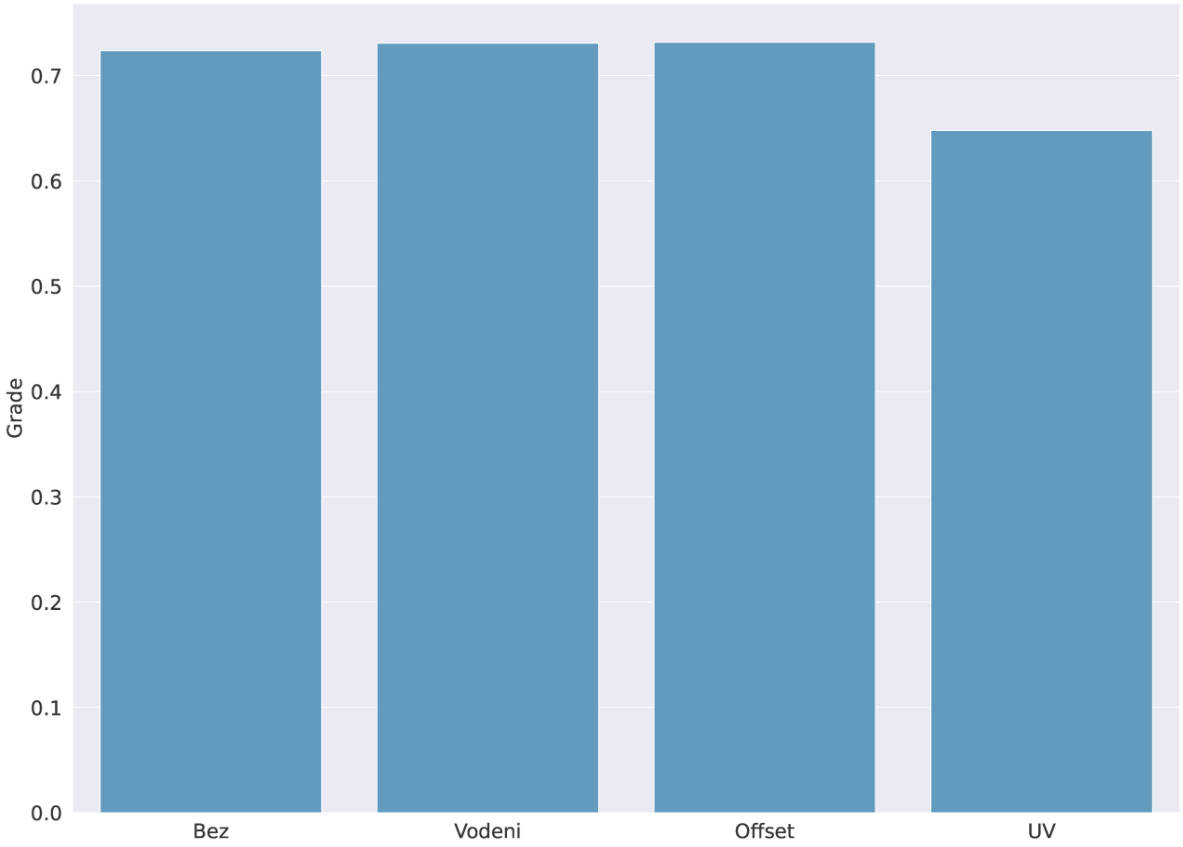

Figure 39. Mean grade values for all four paper varnishes used in the experiment.

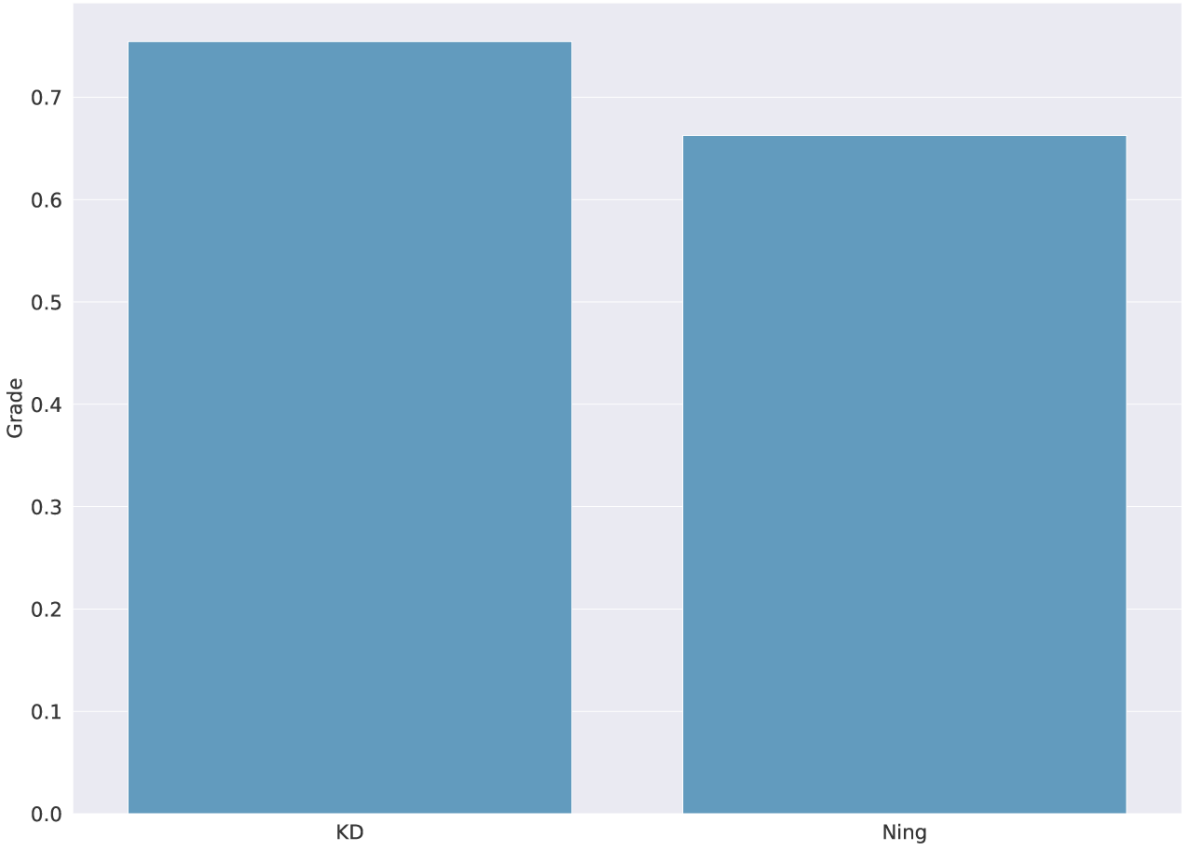

Figure 40. Mean grade values for both paper types used in the experiment.

11.5 Comparison with state-of-the-art

State-of-the-art research offers numerous methods fit for comparison with the method presented in this thesis. However, none of those methods feature all the same properties as the presented method, so the comparisons in this section will be given by separate, individual properties.

Solanki et al. use DFT to develop a data hiding method for document security and image copyright protection [1]. Authors use DFT for data embedding multiple bits of information. The final achieved capacity varies between 125 and 275, for images $512\ x\ 512\ pixels$ in dimension. The method achieved BER values from 0.07 up to 0.16 after the *print-scan* attack, depending on the image. When compared to the method presented in this thesis, this method achieved a much higher *bit per pixel* value, while the BER values are similar in both cases. However, Solanki et al. did not test image quality in their research. Also, the authors failed to mention the size of the image dataset used in experiments. It is also important to note that the method is suited to work with grayscale images. Finally, they conclude by stating that for some low-activity images, much fewer bits could be hidden. To conclude, while the method presented by the authors achieved some high-quality results, the lack of image quality metrics along with a limited dataset, make the results incomplete in comparison to the method presented in this thesis.

Liao et al. also use DFT to embed data into grayscale images. In their method, they embed data into high-frequency areas of the frequency domain [2]. The method is highly successful in terms of data recovery, with BER values ranging from 0.02 to 0.1. These results indicate an almost perfect detection. The method is only able to embed a single mark in an image, resulting in a *zero-bit* capacity. Reported PSNR values range from 40 to 45 dB. In conclusion, while the method is able to achieve extremely high detection rates, its *zero-bit* capacity, limitation to grayscale images only, and PSNR values between 40 and 45 dB are inferior to the method presented in this thesis.

Wang and Lee present their DFT-based steganography method and start by stating that robustness is the highest priority when it comes to real-world applications of data hiding for the print domain [3]. They use a block-based approach and template matching to achieve better detection of information. For their experiment, they use grayscale images $1024\ x\ 1024\ pixels$ in dimensions, separated into $128\ x\ 128\ pixel$ blocks. Only blocks with a high enough complexity values are used for data embedding. Reported BER values for the

print-scan attack range from 0.03 to 0.08, while maintaining similar values for other attacks such as cropping and rotation. PSNR values range between 36 and 38 dB. The authors' method is able to achieve high detection. However, it is difficult to assess the overall quality of the method with a dataset limited to a total of two samples. Finally, the reported PSNR values are considerably low when compared to other similar methods.

12 CONCLUSION

The goal of this thesis is to develop a data hiding method suited for printing. More precisely, the data embedded in an image has to be able to sustain all image attacks that are occuring during the print-scan communication channel. To achieve this, an image is first transformed to the frequency domain using the Discrete Fourier Transform (DFT). Data is embedded in the magnitude of the frequency domain. Gray Component Replacement (GCR) is used to mask the embedding artifacts, thus ensuring imperceptibility of data to the Human Visual System (HVS).

With the continuous development of technology, there is a need for new security systems against counterfeiting. A data hiding method such as the one proposed in this thesis presents a high-level security solution, with no additional processes needed in the context of print production. Furthermore, in the context of securing packaging against counterfeiting, the proposed method can be used without compromising the visual design, as the information is hidden in an image, i.e. no additional visual elements are needed.

Tests are conducted to test the method's robustness to image attacks. The objective metric tests are conducted using 1000 images in CMYK color space. Each image was divided into 16 image blocks, and data is embedded individually into each one. The method is tested for robustness against the *print-scan* process, as well as other image attacks such as rotation, scaling, blur, noise, and JPEG compression. A subjective experiment is also conducted, using 144 images. Sixty-two participants are included in the survey, adding up to a total of 1488 samples tested. Participants are asked to assess the presence of embedding artifacts, thus testing the method's imperceptibility.

The results show that the method is able to extract data from images after the print-scan process with a mean bit-error-rate (BER) of 0.25. In other words, on average, 75% of embedded data is successfully detected at the decoder. This number increases to 77% when applying image processing. When using error correction codes, BER values increase to 83%. For other image attacks tested, the method achieves reasonable robustness. The results of the subjective experiment show that 49,7% of samples were graded as *imperceptible*, and another 30% as *low*. In other words, the participants assessed 84,7% of samples as having no visible embedding artifacts whatsoever, or having barely visible embedding artifacts.

The central goal of this thesis is developing a data hiding method that is suited for printing. The results from experiments clearly show that the said goal is achieved. When comparing

the presented method to other similar state-of-the-art methods, the biggest added value of the proposed method is two-fold:

1. The method does not require additional print processes and/ or additional visual elements for detection
2. Thanks to GCR masking, the method is able to maintain considerably higher image quality than other state-of-the-art methods

All five hypotheses are supported by evidence displayed in Section 11, and all three expected scientific contributions are achieved, as discussed in Section 1.2. Below, each hypothesis and scientific contribution are given, with the respective experiments that are conducted.

Scientific contributions

C1 (Robust steganography method suited for the print-scan communication channel based on the Discrete Fourier Transform with Gray Component Replacement masking)

- Experiments conducted:
  - E0 (Activity test)
  - E1 (Print-scan test)
  - E2 (Digital test with and without GCR)
  - E3 (Digital test with and without ECC)
  - E4 (Digital test with various frequency ranges)

C2 (The determination of the frequency range used for data embedding for the balance between image quality and the detection rate)

- Experiments conducted:
  - E4 (Digital test with various frequency ranges)

C3 (The framework for evaluation of image attacks' effect on the detection of data hidden using a steganography method)

- Experiments conducted:
  - E2 (Digital test with and without GCR)

Hypotheses

H1 (Hidden data remains detectable after print-scan)

- Experiments conducted:
  - E1 (Print-scan test)

H2 (Detection improves after image processing)

- Experiments conducted:
    - E2 (Digital test with and without GCR)

H3 (Error correction codes improve decoder performance)

- Experiments conducted:
    - E3 (Digital test with and without ECC)

H4 (Gray component replacement decreases artifact visibility)

- Experiments conducted
    - E2 (Digital test with and without GCR)

H5 (Gray component replacement does not affect detection)

- Experiments conducted
    - E2 (Digital test with and without GCR)
    - E5 (Subjective survey)

13 REFERENCES

[1] K. Solanki, U. Madhow, B. S. Manjunath, S. Chandrasekaran, and I. El-Khalil, "'Print and scan' resilient data hiding in images," *IEEE Trans. Inf. Forensics Secur.*, vol. 1, no. 4, pp. 464–478, 2006, doi: 10.1109/TIFS.2006.885032.

[2] A. Cheddad, J. Condell, K. Curran, and P. Mc Kevitt, "Digital image steganography: Survey and analysis of current methods," *Signal Processing*, vol. 90, no. 3, pp. 727–752, 2010, doi: 10.1016/j.sigpro.2009.08.010.

[3] A. Poljicak, "Discrete Fourier transform–based watermarking method with an optimal implementation radius," *J. Electron. Imaging*, vol. 20, no. 3, p. 033008, 2011, doi: 10.1117/1.3609010.

[4] A. Dixit and R. Dixit, "A Review on Digital Image Watermarking Techniques," *Int. J. Image, Graph. Signal Process.*, vol. 9, no. 4, pp. 56–66, 2017, doi: 10.5815/ijigsp.2017.04.07.

[5] A. Poljičak, "Zaštita vlasništva reproducirane slike umetanjem digitalnog vodenog žiga," p. 133, 2011.

[6] M. Urvoy, D. Goudia, and F. Autrusseau, "Perceptual DFT Watermarking With Improved Detection and Robustness to Geometrical Distortions," *IEEE Trans. Inf. Forensics Secur.*, vol. 9, no. 7, pp. 1108–1119, Jul. 2014, doi: 10.1109/TIFS.2014.2322497.

[7] A. Poljicak, D. Donevski, and L. Mandic, "Applicability of the GCR masking in an image watermarking method," *Proc. Elmar - Int. Symp. Electron. Mar.*, vol. 2017-Septe, no. September, pp. 215–218, 2017, doi: 10.23919/ELMAR.2017.8124471.

[8] D. Donevski, A. Poljicak, M. Prsa, and T. Hudika, "Adjustable color transformation model," *Proc. Elmar - Int. Symp. Electron. Mar.*, vol. 2019-Septe, no. September, pp. 69–72, 2019, doi: 10.1109/ELMAR.2019.8918906.

[9] A. Horé and D. Ziou, "Image quality metrics: PSNR vs. SSIM," *Proc. - Int. Conf. Pattern Recognit.*, no. August, pp. 2366–2369, 2010, doi: 10.1109/ICPR.2010.579.

[10] K. H. Thung and P. Raveendran, "A survey of image quality measures," *Int. Conf. Tech. Postgraduates 2009, TECHPOS 2009*, 2009, doi: 10.1109/TECHPOS.2009.5412098.

[11] J. M. Soon and L. Manning, “Developing anti-counterfeiting measures: The role of smart packaging,” *Food Res. Int.*, vol. 123, no. April, pp. 135–143, 2019, doi: 10.1016/j.foodres.2019.04.049.

[12] H. P. Nguyen, F. Retraint, F. Morain-Nicolier, and A. Delahaies, “A Watermarking Technique to Secure Printed Matrix Barcode - Application for Anti-Counterfeit Packaging,” *IEEE Access*, vol. 7, pp. 131839–131850, 2019, doi: 10.1109/ACCESS.2019.2937465.

[13] K. Dégardin, A. Guillemain, P. Klespe, F. Hindelang, R. Zurbach, and Y. Roggo, “Packaging analysis of counterfeit medicines,” *Forensic Sci. Int.*, vol. 291, pp. 144–157, 2018, doi: 10.1016/j.forsciint.2018.08.023.

[14] P. Fathi, N. C. Karmakar, M. Bhattacharya, and S. Bhattacharya, “Potential Chipless RFID Sensors for Food Packaging Applications: A Review,” *IEEE Sens. J.*, vol. 20, no. 17, pp. 9618–9636, 2020, doi: 10.1109/JSEN.2020.2991751.

[15] H. Zhou, S. Li, S. Chen, Q. Zhang, W. Liu, and X. Guo, “Enabling Low Cost Flexible Smart Packaging System with Internet-of-Things Connectivity via Flexible Hybrid Integration of Silicon RFID Chip and Printed Polymer Sensors,” *IEEE Sens. J.*, vol. 20, no. 9, pp. 5004–5011, 2020, doi: 10.1109/JSEN.2020.2966011.

[16] S. J. Trenfield *et al.*, “Track-and-trace: Novel anti-counterfeit measures for 3D printed personalized drug products using smart material inks,” *Int. J. Pharm.*, vol. 567, no. June, 2019, doi: 10.1016/j.ijpharm.2019.06.034.

[17] G. J. Simmons, “The Prisoners’ Problem and the Subliminal Channel,” *Chaum D. Adv. Cryptology.*, pp. 51–67, 1984, doi: 10.1007/978-1-4684-4730-9_5.

[18] N. Komatsu and H. Tominaga, “Authentication system using concealed images in telematics,” *Mem. Sch. Sci. Eng. Waseda Univ.*, vol. 52, no. 52, pp. 45–60, 1988.

[19] S. Bhattacharyya, I. Banerjee, and G. Sanyal, “A Survey of Steganography and Steganalysis Technique in Image, Text, Audio and Video as Cover Carrier,” *J. Glob. Res. Comput. Sci.*, vol. 2, no. 4, 2011.

[20] A. Z. Tirkel, G. A. Rankin, R. M. Van Schyndel, W. J. Ho, N. R. A. Mee, and C. F. Osborne, “Electronic Water Mark,” *Digit. Image Comput. Technol. Appl.*, pp. 666–673, 1993.

[21] I. J. Kadhim, P. Premaratne, P. J. Vial, and B. Halloran, “Comprehensive survey of

image steganography: Techniques, Evaluations, and trends in future research," *Neurocomputing*, vol. 335, pp. 299–326, 2019, doi: 10.1016/j.neucom.2018.06.075.

[22] N. Agarwal, A. K. Singh, and P. K. Singh, "Survey of robust and imperceptible watermarking," *Multimed. Tools Appl.*, vol. 78, no. 7, pp. 8603–8633, 2019, doi: 10.1007/s11042-018-7128-5.

[23] H. C. Ling, R. C. W. Phan, and S. H. Heng, "On the security of a hybrid watermarking algorithm based on singular value decomposition and Radon transform," *AEU - Int. J. Electron. Commun.*, vol. 65, no. 11, pp. 958–960, 2011, doi: 10.1016/j.aeue.2011.06.008.

[24] A. K. Singh, M. Dave, and A. Mohan, "Hybrid technique for robust and imperceptible multiple watermarking using medical images," *Multimed. Tools Appl.*, vol. 75, no. 14, pp. 8381–8401, 2016, doi: 10.1007/s11042-015-2754-7.

[25] Q. Su *et al.*, "New Rapid and Robust Color Image Watermarking Technique in Spatial Domain," *IEEE Access*, vol. 7, pp. 30398–30409, 2019, doi: 10.1109/ACCESS.2019.2895062.

[26] B. Lei, E. L. Tan, S. Chen, D. Ni, T. Wang, and H. Lei, "Reversible watermarking scheme for medical image based on differential evolution," *Expert Syst. Appl.*, vol. 41, no. 7, pp. 3178–3188, 2014, doi: 10.1016/j.eswa.2013.11.019.

[27] A. Pramila, A. Keskinarkaus, V. Takala, and T. Seppänen, "Extracting watermarks from printouts captured with wide angles using computational photography," *Multimed. Tools Appl.*, vol. 76, no. 15, pp. 16063–16084, 2017, doi: 10.1007/s11042-016-3895-z.

[28] A. Poljicak, D. Donevski, P. B. Jelusic, T. Tomasegovic, and T. Cigula, "Analysis of the GCR communication channel for image steganography," *Proc. Elmar - Int. Symp. Electron. Mar.*, vol. 2019-Septe, no. September, pp. 65–68, 2019, doi: 10.1109/ELMAR.2019.8918869.

[29] A. Pramila, A. Keskinarkaus, and T. Seppänen, "Increasing the capturing angle in print-cam robust watermarking," *J. Syst. Softw.*, vol. 135, pp. 205–215, 2018, doi: 10.1016/j.jss.2017.10.029.

[30] W. Cai-yin, K. Xiang-wei, and L. Chao, "Process color watermarking: the use of visual masking and dot gain correction," *Multimed. Tools Appl.*, vol. 76, no. 15, pp.

16291–16314, 2017, doi: 10.1007/s11042-016-3909-x.

[31] W. Bender, D. Gruhl, N. Morimoto, and A. Lu, “Techniques for data hiding,” *IBM Syst. J.*, vol. 35, no. 3–4, pp. 313–335, 1996, doi: 10.1147/sj.353.0313.

[32] J. Advith, K. R. Varun, and K. Manikantan, “Novel digital image watermarking using DWT-DFT-SVD in YCbCr color space,” *1st Int. Conf. Emerg. Trends Eng. Technol. Sci. ICETETS 2016 - Proc.*, pp. 4–9, 2016, doi: 10.1109/ICETETS.2016.7603032.

[33] T. P. Duy, D. Tran, and W. Ma, “An intelligent learning-based watermarking scheme for outsourced biomedical time series data,” *Proc. Int. Jt. Conf. Neural Networks*, vol. 2017-May, pp. 4408–4415, 2017, doi: 10.1109/IJCNN.2017.7966414.

[34] M. Andalibi and D. M. Chandler, “Digital Image Watermarking via Adaptive Logo Texturization,” *IEEE Trans. Image Process.*, vol. 24, no. 12, pp. 5060–5073, 2015, doi: 10.1109/TIP.2015.2476961.

[35] K. Gourrame *et al.*, “A zero-bit Fourier image watermarking for print-cam process,” *Multimed. Tools Appl.*, vol. 78, no. 2, pp. 2621–2638, 2019, doi: 10.1007/s11042-018-6302-0.

[36] N. Jimson and K. Hemachandran, “DFT Based Coefficient Exchange Digital Image Watermarking,” *Proc. 2nd Int. Conf. Intell. Comput. Control Syst. ICICCS 2018*, no. Iciccs, pp. 567–571, 2019, doi: 10.1109/ICCONS.2018.8663122.

[37] A. K. Singh, “Improved hybrid algorithm for robust and imperceptible multiple watermarking using digital images,” *Multimed. Tools Appl.*, vol. 76, no. 6, pp. 8881–8900, 2017, doi: 10.1007/s11042-016-3514-z.

[38] R. K. Sheth and V. V. Nath, “Secured digital image watermarking with discrete cosine transform and discrete wavelet transform method,” *Proc. - 2016 Int. Conf. Adv. Comput. Commun. Autom. ICACCA 2016*, pp. 1–5, 2016, doi: 10.1109/ICACCA.2016.7578861.

[39] A. K. Sahu and M. Sahu, “Digital image steganography and steganalysis: A journey of the past three decades,” *Open Comput. Sci.*, vol. 10, no. 1, pp. 296–342, 2020, doi: 10.1515/comp-2020-0136.

[40] J. Trithemius, *Steganographia*. 1499.

[41] J. Reeds, “Solved: The ciphers in book iii of trithemius’s steganographia,”

Cryptologia, vol. 22, no. 4, pp. 291–317, 1998, doi: 10.1080/0161-119891886948.

[42] D. Kahn, "The history of steganography," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 1174, pp. 1–5, 1996, doi: 10.1007/3-540-61996-8_27.

[43] J. Kurak, C; McHugh, "A cautionary note on image downgrading," *Proc. Eighth Annu. Comput. Secur. Appl. Conf.*, pp. 153–159, 1992, doi: 10.1109/CSAC.1992.228224.

[44] R. Jain and J. Boaddh, "Advances in digital image steganography," *2016 1st Int. Conf. Innov. Challenges Cyber Secur. ICICCS 2016*, no. Iciccs, pp. 163–171, 2016, doi: 10.1109/ICICCS.2016.7542298.

[45] S. P. Mohanty, "Digital Watermarking : A Tutorial Review," *Channels*, 1999.

[46] H. Berghel, "Watermarking cyberspace," *Commun. ACM*, vol. 40, no. 11, pp. 19–24, 1997, doi: 10.1145/265684.265687.

[47] A. Giakoumaki, K. Perakis, A. Tagaris, and D. Koutsouris, "Digital watermarking in telemedicine applications - Towards enhanced data security and accessibility," *Annu. Int. Conf. IEEE Eng. Med. Biol. - Proc.*, pp. 6328–6331, 2006, doi: 10.1109/IEMBS.2006.260283.

[48] Y. I. Khamlichi, Y. Zaz, and K. Afdel, "Authentication system for medical watermarked content based image," *Wseas Trans Signal Process*, vol. 5, pp. 826–830, 2006.

[49] T. M. Damico, "A Brief History of Cryptography," *Inq. Journal/Student Pulse*, vol. 1, no. 11, [Online]. Available: http://www.inquiriesjournal.com/articles/1698/a-brief-history-of-cryptography.

[50] D. Davies, "A brief history of cryptography," *Inf. Secur. Tech. Rep.*, vol. 2, no. 2, pp. 14–17, 1997, doi: 10.1016/s1363-4127(97)81323-4.

[51] A. Kumar and K. Pooja, "Steganography- A Data Hiding Technique," *Int. J. Comput. Appl.*, vol. 9, no. 7, pp. 19–23, 2010, doi: 10.5120/1398-1887.

[52] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. 2008.

[53] M. D. McFarlane, "Digital Pictures Fifty Years Ago," *Proc. IEEE*, vol. 60, no. 7, pp. 768–770, 1972, doi: 10.1109/PROC.1972.8775.

[54] M. Petrou and C. Petrou, *Image Processing: The Fundamentals*. 2010.

[55] M. S. Subhedar and V. H. Mankar, “Current status and key issues in image steganography: A survey,” *Comput. Sci. Rev.*, vol. 13–14, no. C, pp. 95–113, 2014, doi: 10.1016/j.cosrev.2014.09.001.

[56] M. S. Taha, M. S. Mohd Rahim, S. A. Lafta, M. M. Hashim, and H. M. Alzuabidi, “Combination of Steganography and Cryptography: A short Survey,” *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 518, no. 5, 2019, doi: 10.1088/1757-899X/518/5/052003.

[57] M. Hussain, A. W. A. Wahab, Y. I. Bin Idris, A. T. S. Ho, and K. H. Jung, “Image steganography in spatial domain: A survey,” *Signal Process. Image Commun.*, vol. 65, pp. 46–66, 2018, doi: 10.1016/j.image.2018.03.012.

[58] V. M. Potdar, H. Song, and C. Elizabeth, “A survey of digital image watermarking techniques,” *2005 3rd IEEE Int. Conf. Ind. Informatics, INDIN*, vol. 2005, pp. 709–716, 2005, doi: 10.1109/INDIN.2005.1560462.

[59] M. Cedillo-Hernandez, F. Garcia-Ugalde, M. Nakano-Miyatake, and H. Perez-Meana, “Robust watermarking method in DFT domain for effective management of medical imaging,” *Signal, Image Video Process.*, vol. 9, no. 5, pp. 1163–1178, 2015, doi: 10.1007/s11760-013-0555-x.

[60] N. Muhammad, N. Bibi, Z. Mahmood, and D. G. Kim, “Blind data hiding technique using the Fresnelet transform,” *Springerplus*, vol. 4, no. 1, pp. 1–15, 2015, doi: 10.1186/s40064-015-1534-1.

[61] T. Manasrah and A. Al-Haj, “Management of medical images using wavelets-based multi-watermarking algorithm,” *2008 Int. Conf. Innov. Inf. Technol. IIT 2008*, pp. 697–701, 2008, doi: 10.1109/INNOVATIONS.2008.4781697.

[62] S. Bhalerao, I. A. Ansari, and A. Kumar, “A secure image watermarking for tamper detection and localization,” *J. Ambient Intell. Humaniz. Comput.*, vol. 12, no. 1, pp. 1057–1068, 2021, doi: 10.1007/s12652-020-02135-3.

[63] F. Di Martino and S. Sessa, “Fragile watermarking tamper detection via bilinear fuzzy relation equations,” *J. Ambient Intell. Humaniz. Comput.*, vol. 10, no. 5, pp. 2041–2061, 2019, doi: 10.1007/s12652-018-0806-3.

[64] A. Poljicak, D. Donevski, P. B. Jelusic, and T. Cigula, “Robust DFT watermarking method with gray component replacement masking,” *Multimed. Tools Appl.*, 2022, doi: 10.1007/s11042-022-12756-9.

[65] C. Lin and S. Chang, “A Robust Image Authentication Method Distinguishing JPEG Compression from Malicious Manipulation,” *IEEE Trans. CIRCUITS Syst. VIDEO Technol.*, vol. 11, no. 2, pp. 153–168, 2001.

[66] K. Fares, K. Amine, and E. Salah, “A robust blind color image watermarking based on Fourier transform domain,” *Optik (Stuttg).*, vol. 208, no. February, p. 164562, 2020, doi: 10.1016/j.ijleo.2020.164562.

[67] N. Subramanian, O. Elharrouss, S. Al-Maadeed, and A. Bouridane, “Image Steganography: A Review of the Recent Advances,” *IEEE Access*, vol. 9, pp. 23409–23423, 2021, doi: 10.1109/ACCESS.2021.3053998.

[68] S. Tyagi, H. V. Singh, R. Agarwal, and S. K. Gangwar, “Digital watermarking techniques for security applications,” *Int. Conf. Emerg. Trends Electr. Electron. Sustain. Energy Syst. ICETEESES 2016*, pp. 379–382, 2016, doi: 10.1109/ICETEESES.2016.7581413.

[69] M. Begum and M. S. Uddin, “Digital image watermarking techniques: A review,” *Inf.*, vol. 11, no. 2, 2020, doi: 10.3390/info11020110.

[70] S. Dhawan and R. Gupta, “Analysis of various data security techniques of steganography: A survey,” *Inf. Secur. J.*, vol. 30, no. 2, pp. 63–87, 2021, doi: 10.1080/19393555.2020.1801911.

[71] M. Douglas, K. Bailey, M. Leeney, and K. Curran, “An overview of steganography techniques applied to the protection of biometric data,” *Multimed. Tools Appl.*, vol. 77, no. 13, pp. 17333–17373, 2018, doi: 10.1007/s11042-017-5308-3.

[72] A. F. Qasim, F. Meziane, and R. Aspin, “Digital watermarking: Applicability for developing trust in medical imaging workflows state of the art review,” *Comput. Sci. Rev.*, vol. 27, pp. 45–60, 2018, doi: 10.1016/j.cosrev.2017.11.003.

[73] C. Song, S. Sudirman, M. Merabti, and D. Llewellyn-Jones, “Analysis of digital image watermark attacks,” *2010 7th IEEE Consum. Commun. Netw. Conf. CCNC 2010*, 2010, doi: 10.1109/CCNC.2010.5421631.

[74] B. Nassiri, R. Latif, A. Toumanari, and F. M. R. Maoulainine, “Secure transmission of medical images by watermarking technique,” *Proc. 2012 Int. Conf. Complex Syst. ICCS 2012*, 2012, doi: 10.1109/ICoCS.2012.6458577.

[75] T. H. Chen, G. Horng, and W. Bin Lee, “A publicly verifiable copyright-proving

scheme resistant to malicious attacks," *IEEE Trans. Ind. Electron.*, vol. 52, no. 1, pp. 327–334, 2005, doi: 10.1109/TIE.2004.841083.

[76] M. Hébert, "Yule-Nielsen effect in halftone prints: graphical analysis method and improvement of the Yule-Nielsen transform," *Color Imaging XIX Displaying, Process. Hardcopy, Appl.*, vol. 9015, p. 90150R, 2014, doi: 10.1117/12.2037370.

[77] F. R. Ruckdeschel and O. G. Hauser, "Yule-Nielsen effect in printing: a physical analysis," *Appl. Opt.*, vol. 17, no. 21, p. 3376, 1978, doi: 10.1364/ao.17.003376.

[78] P. Žitinski Elías, "Halftoning for Multi-Channel Printing: Algorithm Development, Implementation and Verification," Linköping University, Linköping, 2014.

[79] "AM & FM Halftoning." https://www.kothariinfotech.com/software.html.

[80] D. R. Wyble and R. S. Berns, "A Critical Review of Spectral Models Applied to Binary Color Printing," *Color Res. Appl.*, vol. 25, no. 1, pp. 4–19, 2000, doi: 10.1002/(SICI)1520-6378(200002)25:1<4::AID-COL3>3.0.CO;2-X.

[81] M. Shaw, G. Sharma, R. Bala, and E. N. Dalal, "Minimal-effort characterization of color printers for additional substrates," *Final Progr. Proc. - IS T/SID Color Imaging Conf.*, vol. 1, pp. 202–207, 2002.

[82] J. Wang, C. C. Hou, and P. C. Lin, "Two-phase assessment for the environmental impacts from offset lithographic printing on color-box packaging," *J. Clean. Prod.*, vol. 53, pp. 129–137, 2013, doi: 10.1016/j.jclepro.2013.03.048.

[83] G. Wolberg, *Digital image warping*. IEEE computer society press, 1990.

[84] P. P. S. Parsania and D. P. V.Virparia, "A Review: Image Interpolation Techniques for Image Scaling," *Int. J. Innov. Res. Comput. Commun. Eng.*, vol. 02, no. 12, pp. 7409–7414, 2015, doi: 10.15680/ijircce.2014.0212024.

[85] T. M. Lehmann, C. Gönner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Trans. Med. Imaging*, vol. 18, no. 11, pp. 1049–1075, 1999, doi: 10.1109/42.816070.

[86] G. Chen, F. Zhu, P. A. Heng, and L. Xu, *Image denoising, local factor analysis, bayesian ying-yang harmony learning*. Elsevier Inc., 2015.

[87] M. A. King, P. W. Doherty, R. B. Schwinger, and B. C. Penney, "A Wiener filter for nuclear medicine images," *Med. Phys.*, vol. 10, no. 6, pp. 876–880, 1983, doi:

10.1118/1.595352.

[88] G. Hudson, A. Léger, B. Niss, I. Sebestyén, and J. Vaaben, “JPEG-1 standard 25 years: past, present, and future reasons for a success,” *J. Electron. Imaging*, vol. 27, no. 04, p. 1, 2018, doi: 10.1117/1.jei.27.4.040901.

[89] A. . Raid, W. . Khedr, M. A. El-dosuky, and A. Wesam, “Jpeg Image Compression Using Discrete Cosine Transform - A Survey,” *Int. J. Comput. Sci. Eng. Surv.*, vol. 5, no. 2, pp. 39–47, 2014, doi: 10.5121/ijcses.2014.5204.

[90] N. Das *et al.*, “Keeping the Bad Guys Out: Protecting and Vaccinating Deep Learning with JPEG Compression,” pp. 1–15, 2017, [Online]. Available: http://arxiv.org/abs/1705.02900.

[91] Y. Zhang, D. Zhao, J. Zhang, R. Xiong, and W. Gao, “Interpolation-dependent image downsampling,” *IEEE Trans. Image Process.*, vol. 20, no. 11, pp. 3291–3296, 2011, doi: 10.1109/TIP.2011.2158226.

[92] S. Dhawan, “A Review of Image Compression and Comparison of its Algorithms,” *Int. J. Electron. Commun. Technol. J. Electron. Commun. Technol.*, vol. 7109, no. 1, pp. 22–26, 2011.

[93] J. Uthayakumar, T. Vengattaraman, and P. Dhavachelvan, “A survey on data compression techniques: From the perspective of data quality, coding schemes, data type and applications,” *J. King Saud Univ. - Comput. Inf. Sci.*, 2018, doi: 10.1016/j.jksuci.2018.05.006.

[94] S. Sharma, “Image Watermarking in Frequency Domain using Hu ’ s Invariant Moments and Firefly Algorithm,” no. April, pp. 1–15, 2022, doi: 10.5815/ijigsp.2022.02.01.

[95] J.-B.-J. Fourier, “Théorie analytique de la chaleur.” Firmin Didot, 1822.

[96] R. X. Gao and R. Yan, *From Fourier Transform to Wavelet Transform: A Historical Perspective*. 2011.

[97] J. Herivel and P. Williams, “Joseph Fourier: The Man and the Physicist,” *Phys. Today*, vol. 28, no. 11, pp. 65–66, 1975, doi: 10.1063/1.3069206.

[98] “Fourier series visualization.” https://upload.wikimedia.org/wikipedia/commons/2/2b/Fourier_series_and_transform.

gif.

[99] F. Chapeau-Blondeau and E. Belin, “Fourier-transform quantum phase estimation with quantum phase noise,” *Signal Processing*, vol. 170, 2020, doi: 10.1016/j.sigpro.2019.107441.

[100] E. O. Brigham, *The Fast Fourier Transform and Its Applications by E. Oran Brigham*, vol. 12. 1988.

[101] J. W. Cooley and J. W. Tukey, “An Algorithm for the Machine Calculation of Complex Fourier Series,” *Math. Comput.*, vol. 19, no. 90, pp. 297–301, 1965.

[102] M. T. Heideman, D. H. Johnson, and C. S. Burrus, “Gauss and the History of the Fast Fourier Transform,” *IEEE ASSP Mag.*, vol. 1, no. 4, pp. 14–21, 1984, [Online]. Available: https://www.researchgate.net/publication/269107473_What_is_governance/link/548173090cf22525dcb61443/download%0Ahttp://www.econ.upf.edu/~reynal/Civil wars_12December2010.pdf%0Ahttps://think-asia.org/handle/11540/8282%0Ahttps://www.jstor.org/stable/41857625.

[103] J. A. C. Yule, “Theory of Subtractive Color Photography,” *J. Opt. Soc. Am.*, vol. 30, no. 8, pp. 322–331, 1940, [Online]. Available: https://www.osapublishing.org/josa/abstract.cfm?uri=josa-30-8-322.

[104] D. Donevski, A. Poljicak, and M. S. Kurecic, “Colorimetrically accurate gray component replacement using the additive model,” *J. Vis. Commun. Image Represent.*, vol. 44, pp. 40–49, 2017, doi: 10.1016/j.jvcir.2017.01.018.

[105] J. A. C. Yule, *Principles of color reproduction, applied to photomechanical reproduction, color photography, and the ink, paper, and other related Industries*. 1967.

[106] A. C. Hardy and F. L. Wurzburg, “Color correction in color printing,” *J. Opt. Soc. Am.*, vol. 38, no. 4, pp. 300–307, 1948, doi: 10.1364/JOSA.38.000300.

[107] H. E. Neugebauer, “Die theoretischen grundlagen des mehrfarbenbuchdrucks,” *Zeitschrift für Wissenschaftliche Photogr.*, 1937.

[108] C. Nakamura and K. Sayanagi, “Gray Component Replacement by the Neugebauer Equations,” *Proc. SPIE*, vol. 1184, pp. 50–63, 1989, doi: 10.1117/12.963893.

[109] B. A. Frost, “The Gray Component Replacement - A New Trend in Color Reproduction,” *Proc. Tech. Assoc. die Graph. Arts*, pp. 394–401, 1986.

[110] H. R. Kang, “Gray Component Replacement Using Color Mixing Models,” *Proc. SPIE*, vol. 2171, pp. 287–296, 1994, doi: 10.1117/12.175317.

[111] S. Bandyopadhyay and S. Mandal, “Effect of Gray Component Replacement on Color Reproduction,” *Soc. Imaging Sci. Technol. Image Process. Image Qual. Image Capture, Syst. Conf.*, pp. 188–191, 2000.

[112] B. H. Kang, M. K. Cho, H. K. Choh, and C. Y. Kim, “Black color replacement using gamut extension method,” *Int. Conf. Digit. Print. Technol.*, pp. 384–386, 2005.

[113] H. R. Kang, *Computational Color Technology*. 2006.

[114] R. L. De Queiroz, K. M. Braun, and R. Loce, “Spatially varying gray component replacement for image watermarking,” *Proc. - Int. Conf. Image Process. ICIP*, vol. 1, no. iv, pp. 385–388, 2005, doi: 10.1109/ICIP.2005.1529768.

[115] R. L. de Queiroz, K. M. Braun, and R. P. Loce, “Detecting spatially varying gray component replacement with application in watermarking printed images,” *J. Electron. Imaging*, vol. 14, no. 3, p. 033016, 2005, doi: 10.1117/1.2001671.

[116] L. K. Mestha and S. A. Dianat, “Interpolation of multidimensional functions,” in *Control of color imaging systems*, CRC Press, 2009, pp. 249–301.

[117] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, “Image quality assessment: From error visibility to structural similarity,” *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004, doi: 10.1109/TIP.2003.819861.

[118] U. Sara, M. Akter, and M. S. Uddin, “Image Quality Assessment through FSIM, SSIM, MSE and PSNR—A Comparative Study,” *J. Comput. Commun.*, vol. 07, no. 03, pp. 8–18, 2019, doi: 10.4236/jcc.2019.73002.

[119] P. B. Jelušić, A. Poljičak, D. Donevski, and T. Cigula, “Analysis of the DFT-based Watermarking Method for Image Steganography,” *Int. Conf. Syst. Signals Image Process.*, 2021.

[120] M. Cheon, T. Vigier, L. Krasula, J. Lee, P. Le Callet, and J. S. Lee, “Ambiguity of objective image quality metrics: A new methodology for performance evaluation,” *Signal Process. Image Commun.*, vol. 93, no. January, p. 116150, 2021, doi:

10.1016/j.image.2021.116150.

[121] Y. Zhan, R. Zhang, and Q. Wu, “A Structural Variation Classification Model for Image Quality Assessment,” *IEEE Trans. Multimed.*, vol. 19, no. 8, pp. 1837–1847, 2017, doi: 10.1109/TMM.2017.2689923.

[122] L. Ding, H. Huang, and Y. Zang, “Image Quality Assessment Using Directional Anisotropy Structure Measurement,” *IEEE Trans. Image Process.*, vol. 26, no. 4, pp. 1799–1809, 2017, doi: 10.1109/TIP.2017.2665972.

[123] W. Sun, Q. Liao, J. Xue, and F. Zhou, “SPSIM : A Superpixel-Based Similarity Index for Full-Reference Image Quality Assessment,” vol. 27, no. 9, pp. 4232–4244, 2018.

[124] F. E. Grubbs, “Sample Criteria for Testing Outlying Observations,” *Ann. Math. Stat.*, vol. 21, no. 1, pp. 27–58, 1950, doi: 10.1214/aoms/1177729885.

[125] F. E. Grubbs, “Procedures for Detecting Outlying Observations in Samples,” *Technometrics*, vol. 11, no. 1, pp. 1–21, 1969, doi: 10.1080/00401706.1969.10490657.

[126] F. E. Grubbs and G. Beck, “Extension of Sample Sizes and Percentage Points for Significance Tests of Outlying Observations,” *Technometrics*, vol. 14, no. 4, pp. 847–854, 1972, doi: 10.1080/00401706.1972.10488981.

[127] K. K. L. B. Adikaram, M. A. Hussein, M. Effenberger, and T. Becker, “Data transformation technique to improve the outlier detection power of grubbs’ test for data expected to follow linear relation,” *J. Appl. Math.*, vol. 2015, 2015, doi: 10.1155/2015/708948.

[128] N. Heckert *et al.*, *Handbook 151: NIST/SEMATECH e-Handbook of Statistical Methods*. Gaithersburg: NIST Interagency/Internal Report (NISTIR), 2002.

[129] “Engineering Statistics Handbook - Grubbs’ Test for Outliers.” https://www.itl.nist.gov/div898/handbook/eda/section3/eda35h1.htm.

[130] R. B. Jain, “A recursive version of Grubbs’ test for detecting multiple outliers in environmental and chemical data,” *Clin. Biochem.*, vol. 43, no. 12, pp. 1030–1033, 2010, doi: 10.1016/j.clinbiochem.2010.04.071.

[131] X. Wu, B. Hong, X. Peng, F. Wen, and J. Huang, “Radial basis function neural network based short-term wind power forecasting with Grubbs test,” *DRPT 2011 - 2011 4th Int. Conf. Electr. Util. Deregul. Restruct. Power Technol.*, pp. 1879–1882,

2011, doi: 10.1109/DRPT.2011.5994206.

[132] W. J. Dixon, “Processing Data for Outliers,” *Biometrics*, vol. 9, no. 1, pp. 74–89, 1953.

[133] G. L. Tietjen and R. H. Moore, “Some Grubbs-Type Statistics for the Detection of Several Outliers,” *Technometrics*, vol. 14, no. 3, pp. 583–597, 1972, doi: 10.1080/00401706.1972.10488948.

[134] G. L. Tietjen, R. H. Moore, and R. J. Beckman, “Testing for a single outlier in simple linear regression,” *Technometrics*, vol. 15, no. 4, pp. 717–721, 1973, doi: 10.1080/00401706.1973.10489106.

[135] M. Urvoy and F. Autrusseau, “Application of grubbs’ test for outliers to the detection of watermarks,” *IH MMSec 2014 - Proc. 2014 ACM Inf. Hiding Multimed. Secur. Work.*, no. 1000, pp. 49–60, 2014, doi: 10.1145/2600918.2600931.

[136] W. Lin, S.-J. Horng, P. Fan, C.-L. Lee, and Y. Pan, “An Efficient Watermarking Method Based on Significant Difference of Wavelet Coefficient Quantization,” *IEEE Trans. Multimed.*, vol. 11, no. 5, pp. 1037–1041, 2009, doi: 10.1109/TMM.2008.922795.

[137] R. Kwitt, P. Meerwald, and A. Uhl, “BLIND DT-CWT DOMAIN ADDITIVE SPREAD-SPECTRUM WATERMARK DETECTION Roland Kwitt and Peter Meerwald and Andreas Uhl Department of Computer Sciences , University of Salzburg , Austria { rkwitt , pmeerw , uhl }@ cosy . sbg . ac . at,” *Science (80-.).*, pp. 0–7, 2009.

[138] V. Solachidis and I. Pitas, “Circularly symmetric watermark embedding in 2-D DFT domain,” *IEEE Trans. Image Process.*, vol. 10, no. 11, pp. 1741–1753, 2001, doi: 10.1109/83.967401.

[139] Y. Liu, D. Zhang, G. Lu, and W. Y. Ma, “A survey of content-based image retrieval with high-level semantics,” *Pattern Recognit.*, vol. 40, no. 1, pp. 262–282, 2007, doi: 10.1016/j.patcog.2006.04.045.

[140] M. Verma and B. Raman, “Local neighborhood difference pattern: A new feature descriptor for natural and texture image retrieval,” *Multimed. Tools Appl.*, vol. 77, no. 10, pp. 11843–11866, 2018, doi: 10.1007/s11042-017-4834-3.

[141] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Multiresolution gray-scale and rotation

invariant texture classification with local binary patterns," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 971–987, 2002, doi: 10.1109/TPAMI.2002.1017623.

[142] F. R. Ishengoma, "The Art of Data Hiding with Reed-Solomon Error Correcting Codes," *Int. J. Comput. Appl.*, vol. 106, no. 14, pp. 975–8887, 2014.

[143] R. W. Hamming, "Error detecting and error correcting codes," *Bell Syst. Tech. J.*, vol. 29, no. 2, pp. 147–160, 1950.

[144] C. Munuera, "Steganography and error-correcting codes," *Signal Processing*, vol. 87, no. 6, pp. 1528–1533, 2007, doi: 10.1016/j.sigpro.2006.12.008.

[145] I. S. Reed and G. Solomon, "Polynomial Codes Over Certain Finite Fields," *J. Soc. Ind. Appl. Math.*, vol. 8, no. 2, pp. 300–304, 1960.

[146] S. Gao, "A new algorithm for decoding Reed-Solomon codes," *Commun. Inf. Netw. Secur.*, pp. 55–68, 2003, doi: 10.1007/978-1-4615-1509-8_7.

[147] R. J. McEliece and D. W. Sarwate, "On Sharing Secrets and Reed-Solomon Codes," *Commun. ACM*, vol. 24, no. 9, pp. 583–584, 1981.

[148] V. Guruswami, S. Member, and M. Wootters, "Repairing Reed-Solomon Codes," *IEEE Trans. Inf. Theory*, vol. 63, no. 9, pp. 5684–5698, 2017.

[149] N. Raut, "Error Correcting Codes - Hamming codes." https://www.tutorialspoint.com/error-correcting-codes-hamming-codes.

[150] Rishi Rathor, "Error Correcting Codes - Reed-Solomon codes." https://www.tutorialspoint.com/error-correcting-codes-reed-solomon-codes.

[151] "Wolfram - Primitive Polynomial." https://mathworld.wolfram.com/PrimitivePolynomial.html.

[152] G. G. La Guardia, "Asymmetric quantum Reed-Solomon and generalized Reed-Solomon codes," *Quantum Inf. Process.*, vol. 11, no. 2, pp. 591–604, 2012, doi: 10.1007/s11128-011-0269-3.

[153] R. Mori and T. Tanaka, "Non-binary polar codes using Reed-Solomon codes and algebraic geometry codes," *2010 IEEE Inf. Theory Work. ITW 2010 - Proc.*, pp. 1–5, 2010, doi: 10.1109/CIG.2010.5592755.

[154] L. Yu, S. Lin, H. Hou, and Z. Li, "Reed-Solomon Coding Algorithms Based on Reed-Muller Transform for Any Number of Parities," pp. 1–12, 2023, doi:

10.1109/TC.2023.3262922.

[155] F. Galton, “Dice for Statistical Experiments,” *Nature*, vol. 42, no. 1070, pp. 13–14, 1890.

[156] C. Tashian, “A Brief History of Random Numbers.” https://tashian.com/articles/a-brief-history-of-random-numbers/.

[157] J. von Neumann, “Various techniques used in connection with random digits,” *National Bureau of Standards Applied Mathematics Series*, vol. 12. pp. 36–38, 1951.

[158] D. H. Lehmer, “Mathematical methods in large-scale computing units,” *Annu. Comput. Lab. Harvard Univ.*, vol. 26, pp. 141–146, 1951.

[159] M. Matsumoto and T. Nishimura, “Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator,” *ACM Trans. Model. Comput. Simul.*, vol. 8, no. 1, pp. 3–30, 1998, doi: 10.1145/272991.272995.

[160] L. Y. Deng and D. Bowman, “Developments in pseudo-random number generators,” *Wiley Interdiscip. Rev. Comput. Stat.*, vol. 9, no. 5, 2017, doi: 10.1002/wics.1404.

[161] S. Vigna, “It is high time we let go of the Mersenne Twister,” pp. 1–14, 2019, [Online]. Available: http://arxiv.org/abs/1910.06437.

[162] L. Y. Deng, “Efficient and portable multiple recursive generators of large order,” *ACM Trans. Model. Comput. Simul.*, vol. 15, no. 1, pp. 1–13, 2005, doi: 10.1145/1044322.1044323.

[163] S. Boslaugh, *Statistics in a nutshell: A desktop quick reference*. 2012.

[164] E. W. Weisstein, “Statistical Correlation.” https://mathworld.wolfram.com/StatisticalCorrelation.html.

14 BIOGRAPHY

Petar Branislav Jelušić was born on August 8, 1990 in Rijeka. He graduated from Prva riječka hrvatska gimnazija in 2009 in Rijeka. The same year, he enrolled in the Faculty of Graphic Arts at the University of Zagreb. In 2016 he obtained his Master's degree under the mentorship of Associate Professor Ante Poljičak with the thesis *"Development of the automatic system for printing calculations using neural networks."* In 2017, Petar started his small business for graphic and web design. In 2019, he started his Ph.D. studies at the Faculty of Graphic Arts, University of Zagreb, where he also started working as a research assistant. His research includes image processing, computer vision, and data hiding. His interests also include neural networks and predictive models. The results of his research were presented at multiple international scientific conferences and published in multiple academic papers. In 2022, he was a visiting scholar at Utrecht Data School, Utrecht University, Netherlands. The same year, he visited the Artevelde University of Applied Sciences in Gent, Belgium, as a part of the Erasmus international program.

Below is the list of the author's scientific papers published during his Ph.D. studies:

Poljičak, Ante; Donevski, Davor; **Jelušić, Petar Branislav**; Cigula, Tomislav; *„Robust DFT watermarking method with gray component replacement masking."* // Multimedia tools and applications, 82 (2022), 1-22 doi:10.1007/s11042-022-12756-9 (international review, paper, scientific)

Jelušić, Petar Branislav; Poljičak, Ante; Donevski, Davor; Cigula, Tomislav; *„Low-Frequency Data Embedding for DFT-Based Image Steganography."* // International Journal of Software Science and Computational Intelligence, 14 (2022), 1; 1-11 doi:10.4018/ijssci.312558 (international review, paper, scientific)