A Digital Image Processing Project Report on Steganography using Python

A Project Work Submitted in
Partial Fulfilment of the requirements
for DEGREE In
ELECTRONICS AND COMMUNICATION ENGINEERING
BY

Name: KEERTHANA V BHAT
SRN: PES1201800549
Name: HEETH M THAKKAR
SRN: PES1201802022

Under the supervision of
Prof. Lavanya Krishna (Assistant Professor, Dept. of ECE)

INTRODUCTION

STEGANOGRAPHY

Steganography is the art and science of embedding secret messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message.
Steganography is a technique of hiding information in digital media. Therefore, it can be used to carry out hidden exchanges.

STEGANOGRAPHY V/S CRYPTOGRAPHY

- In contrast to Cryptography, steganography not only keeps others from knowing the hidden information but also keeps intruder from thinking that the information is present.
- Steganography's intent is to hide the existence of the message, while cryptography scrambles a message so that it cannot be understood.
- Steganography provides better security than cryptography because cryptography hides the contents of the message but not the existence of the message. So, no one apart from the authorized sender and receiver will be aware of the existence of the secret data.

TYPES OF STEGANOGRAPHY

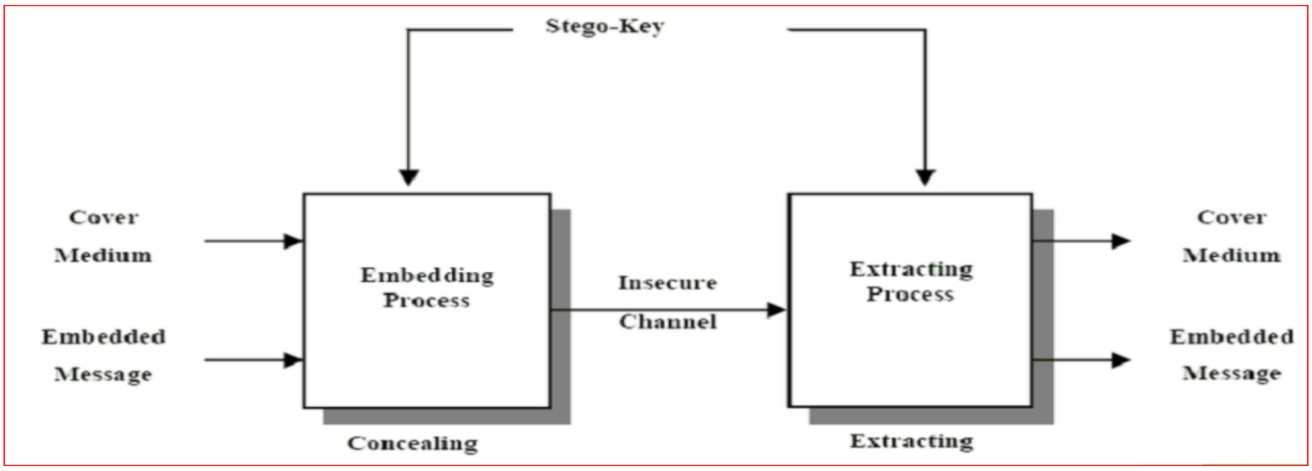
- Text Steganography
- Images Steganography
- Audio Steganography
- Video Steganography
- Network Steganography
- Email Steganography

CHARECTERISTICS OF STEGANOGRAPHY

- **Transparency**: Each cover media has certain data hiding capacity. If more information is hidden inside a cover image, then it will result in degradation of cover.

- **Robustness**: Ability of hidden message to remain undamaged even if the stego-image goes through some kind of transformation like cropping, blurring, linear or nonlinear filtering
- **Tamper Resistance**: If the hacker is successful in destroying the steganographic technique then this property makes it difficult for the attacker to alter or damage the original data

STEGANOGRAPHY MODEL



OVERVIEW OF THE PROJECT

IMAGE STEGANOGRAPHY

Image steganography is a technique of hiding data within the image in such a way that it prevents the unintended user from detection of hidden messages or data. It's one of the popular ways of hiding secret messages because an image consists of a lot of bits present in digital representation. Because digital images are insensitive to human visual system, therefore images could be good cover carriers The use of steganography can be combined with encryption as an extra step for hiding or protecting data.

OBJECTIVE

In this project we mainly concentrated on embedding data into an image.
The goal of steganography is to communicate securely in a completely undetectable manner and to avoid drawing suspicion to the existence of hidden messages. Many different carrier file formats can be used, but digital images are the most popular because of their frequency on the internet.

APPLICATION OF STEGANOGRAPHY

- Having confidential communication and storage of secret data.

- Secure Private Files and Documents
- Hide password and encryption keys
- Giving protection to the data alteration
- Transmit message/data without revealing the existence of available message.
- The steganography can be used by the intelligence agencies for transmitting their secret information.
- Smart identity cards
- Medical Information

PROJECT WORKING

In image steganography, a message is embedded into an image by altering the values of some pixels, which are chosen by an encryption algorithm. The recipient of the image must be aware of the same algorithm in order to known which pixels he or she must select to extract the message. The image selected for this purpose is called the cover-image and the image obtained after steganography is called the stego-image.

The idea behind image-based Steganography is very simple. Images are composed of digital data (pixels), which describes what's inside the picture, usually the colors of all the pixels. Since we know every image is made up of pixels and every pixel contains 3-values (red, green, blue), steganography works by changing bits of useless data in regular computer files (such as graphics, sound, text, HTML) with bits of different, invisible information. This hidden information can be plain text, cipher text, or even images.

PRINCIPLE BEHING STEGANOGRAPHY

- ➢ Pixels are the building blocks of an image.
- ➢ Every pixel contains three values: (red, green, blue) also known as RGB values.
- ➢ Every RGB value ranges from 0 to 255 for 8-bit image.

ENCODE

STEP 1: Every byte of data is converted to its 8-bit binary code using ASCII values.

STEP 2: Now pixels are read from left to right in a group of 3 containing a total of 9 values (i.e. 3*3=9 RGB values). The first 8-values are used to store one character which was converted into an 8-bit binary in step 1.

STEP 3: Then the corresponding RGB value and binary data are compared, if the binary digit is 1 then the RGB value will be converted into odd else even.

STEP 4: The 9th value determines if more pixels should be read or not. If there is more data to be read, i.e. encoded or decoded, then the ninth pixel changes to even. Otherwise, if we want to stop reading pixels further, then make it odd.

Example: Suppose the message to be hidden is 'cat'. Since the message is of 3-bytes, therefore, pixels required to encode the data is 3 x 3 = 9. Consider a 4 x 3 image with a total 12-pixels, which are sufficient to encode the given data.

i. ASCII value of '**C**' is 67 whose binary equivalent is 01000011.
ii. Take first 3-pixels [(27, 64, 164), (248, 244, 194), (174, 246, 250)] of the image to encode.
iii. Now change the pixel to odd for 1 and even for 0.
iv. So, the modifies pixels are [(26, 63, 164), (248, 244, 194), (173, 245, 250)].
v. Now repeat the same for remaining characters.

DECODE

Decoding is getting secret information from image file. This process is called as Steganolysis

STEP 1: To decode, three pixels are read at a time, till the last value is odd, which means the message is over. The first 8 values of RGB give the secret data while the 9th value determines whether we need to move forward or not.

STEP 2: Every 3-pixels contain a binary data. The first 8 values can be extracted by the same encoding logic. If the value is odd the binary bit is 1 else 0.

STEP 3: By concatenating these bits to a string with every three pixels we will get a byte of secret data i.e. one character.

Example:

i. First read the three pixels of the corrupted image –
[(26, 63, 164), (248, 244, 194), (173, 245, 250)] .
ii. The first value: 26, which is even, therefore the binary bit is 0. Similarly, for 63, the binary bit is 1 and for 164 it is 0.
iii. This process continues until the 8th RGB value.
iv. We finally get the binary value: 01000011 after concatenating all individual binary values.
v. The final binary data corresponds to decimal value 72, and in ASCII, it represents the character H.

PROJECT CODE

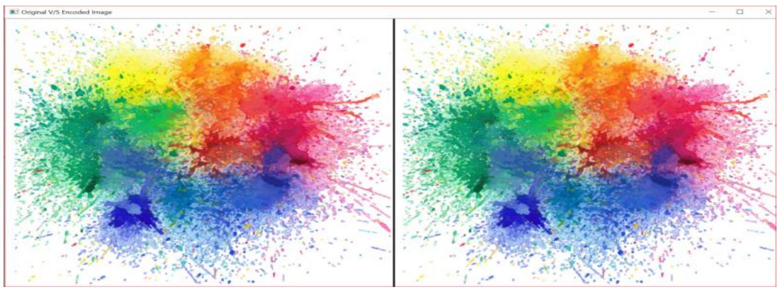
```
from PIL import Image
from os import system
import cv2
import numpy as np
import time
genData = lambda data : [(format(ord(i),'08b')) for i in data]
getPix = lambda imgdata : imgdata.__next__()[:3] +
                                    imgdata.__next__()[:3] +
                                     imgdata.__next__()[:3]

def modPix(pix, data):
            datalist = genData(data)
            lendata = len(datalist)
            imdata = iter(pix)

  for i in range(lendata):
            pix = [value for value in getPix(imdata)]
            for j in range(0, 8):
                    if (datalist[i][j] == '0' and pix[j] % 2 != 0):
                            pix[j] -= 1
                    elif (datalist[i][j] == '1' and pix[j] % 2 == 0):
                            if(pix[j] != 0):
                                    pix[j] -= 1
                            else:
                                    pix[j] += 1

            if (i == lendata - 1):
                    if (pix[-1] % 2 == 0):
                            if(pix[-1] != 0):
                                    pix[-1] -= 1
                            else:
                                    pix[-1] += 1
            else:
                    if (pix[-1] % 2 != 0):
                            pix[-1] -= 1

            pix = tuple(pix)
            yield pix[0:3]
            yield pix[3:6]
            yield pix[6:9]

  def encode_enc(newimg, data):
            w = newimg.size[0]
            (x, y) = (0, 0)
            for pixel in modPix(newimg.getdata(), data):
                    newimg.putpixel((x, y), pixel)
                    if (x == w - 1):
                            x = 0
                            y += 1
```

```
        else:
            x += 1


def open(enordec):
    while True:
        try:
            if enordec == "encode":
                choice = int(input("1. Select anImage\n
                                    2. Capture an Image\n"))
                if choice == 1:
                    name = input("Enter image name (with extension) : ")

                elif choice == 2:
                    name=input("Enter the name to save the image (with extension) : ")
                print("Press Space to Capture.")
                cam = cv2.VideoCapture(0)
                time.sleep(1)
                while True:
                    _, imag = cam.read()
                    cv2.imshow("Camera",imag)
                    key = cv2.waitKey(1)
                    if key == 32:
                        cv2.imwrite(name,imag)
                        break
                cam.release()
                cv2.destroyAllWindows()

            elif enordec == "decode":
                name = input("Enter image name (with extension) : ")

            image = Image.open(name, 'r')
            cv2.imshow("Image", cv2.imread(name))
            print("The size of the selected image is : ", image.size)
            cv2.waitKey(0)
            cv2.destroyAllWindows()

            inp = int(input("Is the displayed image correct?\n1. Yes\t2. No\n"))
            if inp == 1:
                break

        except:
            print("File Not Found. Please try again.")
    return [name, image]
def encode():
    [imgname, image] = open("encode")

    data = input("Enter data to be encoded : ")
```

```
		if (len(data) == 0):
			raise ValueError('Data is empty.')

		newimg = image.copy()
		encode_enc(newimg, data)

		new_img_name = input("Enter the name of new image(with extension) : ")
		newimg.save(new_img_name, str(new_img_name.split(".")[1].upper()))
		print("The new image with the specified name is saved.")

		img1 = cv2.imread(imgname)
		img2 = cv2.imread(new_img_name)
		imgjoin = np.concatenate((img1, img2), axis = 1)
		imgjoin = cv2.line(imgjoin, (image.size[0],0), (image.size[0],image.size[1]), (0,0,0), 2)
		cv2.imshow("Original V/S Encoded Image", imgjoin)
		cv2.waitKey(0)
		cv2.destroyAllWindows()


	def decode():
		[_, image] = open("decode")
		data = ''
		imgdata = iter(image.getdata())

		while (True):
			pixels = [value for value in getPix(imgdata)]
			binstr = ''
			for i in pixels[:8]:
				if (i % 2 == 0):
					binstr += '0'
				else:
					binstr += '1'

			data += chr(int(binstr, 2))
			if (pixels[-1] % 2 != 0):
				return data

	  while True:
		a = int(input("Image Steganography\n1. Encode\n2. Decode\n3. Exit\n"))

		if a == 1:
			encode()
		elif a == 2:
			data = decode()
			print("Decoded Word : ", data)

		elif a == 3:
			print('Exiting.')
```

```
        break

    else:
        print("Enter correct input")
    input("Press any key.")
    system('cls')
```

INPUT AND OUTPUT IMAGE



CONCLUSION AND FUTURE SCOPE

CONCLUSION

In the present world, data transfer using internet is rapidly growing because it is so easier as well as faster to transfer data to destination. So, many individuals and business people transfer business documents, important information using internet. Security is an important issue while transferring data using internet because any unauthorized individual can hack the data, reveal information to others, modify it to misrepresent an individual or organization, or use it to launch an attack.
Image Steganography essentially exploits human perception as human senses are not trained to look for files that have information inside of them. Therefore, we use Steganography technique for secret data exchange. The speed of embedding the data into an image is high in the proposed approach as it includes simple calculations.

FUTURE SCOPE

- The future work on this project is to improve the compression ratio of the image to the text.
- This program might not work as expected with JPEG images because JPEG uses lossy compression which means that the pixels are modified to compress the image and reduce the quality, therefore data loss happens.
- This project can be extended to a level such that it can be used for different types of image formats like .bmp, ,jpeg, ,tif etc. in the future.