STENOGRAPHY-BASED IMAGE MORSE CODE

USING PYTHON

A PROJECT REPORT

Submitted by

ASWIN K : 715522104008

DAVID VEDHA JEROME A : 715522104113

DHARINEESH DJ : 715522104116

DILIPAN AR : 715522104117

DHRUV R : 715522104118

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERINGPSG INSTITUTE OF TECHNOLOGY AND APPLIED RESEARCH, COIMBATORE 641 062

ANNA UNIVERSITY: CHENNAI - 600 025JUNE 2023

ABSTRACT

Image Steganography is the process of hiding information which can be text, image or video inside a cover image. The secret information is hidden in a way that it not visible to the human eyes. Deep learning technology, which has emerged as a powerful tool in various applications including image steganography, has received increased attention recently.

The main goal of this paper is to explore and discuss various deep learning methods available in image steganography field. Deep learning techniques used for image steganography can be broadly divided into three categories - traditional methods, Convolutional Neural Network-based and General Adversarial Network-based methods.

Along with the methodology, an elaborate summary on the datasets used, experimental set-ups considered and the evaluation metrics commonly used are described in this paper. A table summarizing all the details are also provided for easy reference. This paper aims to help the fellow researchers by compiling the current trends, challenges and some future direction in this field.

LITERATURE REVIEW

Steganography is the science that involves encrypting data in a suitable multimedia carrier, such as image, audio, and video files. The main purpose of image steganography is to hide the data in images.

This means that it encrypts the text in the form of an icon. Steganography is done when there is communication takes place between sender and receiver. In a day of data transfer over the network, security is paramount. Before the development of stenography, data security is a major research concern for researchers. Steganography is gaining importance due to the rapid development of users on the Internet and secret communication. In this paper, we discuss about various type of existing image steganography techniques and analyze the advantages and disadvantages of different types of image steganography techniques.

Keywords

- **Steganography**
- **Multimedia carrier**
- **Communication**
- **Data transfer**
- **Security**
- **Image steganography**

Image Stenography can be performed by encoding text or information in the image and decoding the information. The images are prompted using a python-based UI created using tkinter module.

SOURCE CODE

```
from tkinter import *
from tkinter import ttk
import tkinter.filedialog
from PIL import ImageTk
from PIL import Image
from tkinter import messagebox
from io import BytesIO
import  os


class Stegno:


    art ='''¯\_(ツ)_/¯'''
    art2 = '''
@(\/)
(\/)-{}-)@
@(={}=)/\)(\/)
(\/(/\)@| (-{}-)
(={}=)@(\/)@(/\)@
(/\)\(={}=)/(\/)
@(\/)\(/\)/(={}=)
(-{}-)"""""@/(/\)
|:   |
/::'   \\
/:::     \\
```

```
|::'      |
|::       |
\::.      /
':_____.'
`"""""""""`"'
```

```
    output_image_size = 0


    def main(self,root):
        root.title('MORSE CODE')
        root.geometry('500x600')
        root.resizable(width =False, height=False)
        f = Frame(root)


        title = Label(f,text='MORSE CODE')
        title.config(font=('courier',33))
        title.grid(pady=10)


        b_encode = Button(f,text="Encode",command= lambda
:self.frame1_encode(f), padx=14)
        b_encode.config(font=('courier',14))
        b_decode = Button(f,
text="Decode",padx=14,command=lambda :self.frame1_decode(f))
        b_decode.config(font=('courier',14))
        b_decode.grid(pady = 12)
```

```
    ascii_art = Label(f,text=self.art)
    # ascii_art.config(font=('MingLiU-ExtB',50))
    ascii_art.config(font=('courier',60),bg='#8fbc8f')

    ascii_art2 = Label(f,text=self.art2)
    # ascii_art.config(font=('MingLiU-ExtB',50))
    ascii_art2.config(font=('courier',12,'bold'),bg='#8fbc8f')

    root.grid_rowconfigure(1, weight=1)
    root.grid_columnconfigure(0, weight=1)

    f.grid()
    f.configure(bg='#8fbc8f')
    title.grid(row=1)
    b_encode.grid(row=2)
    b_decode.grid(row=3)
    ascii_art.grid(row=4,pady=10)
    ascii_art2.grid(row=5,pady=5)

def home(self,frame):
        frame.destroy()
        self.main(root)
```

```
    def frame1_decode(self,f):
        f.destroy()
        d_f2 = Frame(root)
        label_art = Label(d_f2, text='٩(^‿^)۶')
        label_art.config(font=('courier',90))
        label_art.grid(row =1,pady=50)
        l1 = Label(d_f2, text='Select Image with Hidden text:')
        l1.config(font=('courier',18))
        l1.grid()
        bws_button = Button(d_f2, text='Select', command=lambda
:self.frame2_decode(d_f2))
        bws_button.config(font=('courier',18))
        bws_button.grid()
        back_button = Button(d_f2, text='Cancel', command=lambda :
Stegno.home(self,d_f2))
        back_button.config(font=('courier',18))
        back_button.grid(pady=15)
        back_button.grid()
        d_f2.grid()


    def frame2_decode(self,d_f2):
        d_f3 = Frame(root)
        myfile = tkinter.filedialog.askopenfilename(filetypes = ([('png',
'*.png'),('jpeg', '*.jpeg'),('jpg', '*.jpg'),('All Files', '*.*')]))
        if not myfile:
```

```
            messagebox.showerror("Error","You have selected nothing
!")
        else:
            myimg = Image.open(myfile, 'r')
            myimage = myimg.resize((300, 200))
            img = ImageTk.PhotoImage(myimage)
            l4= Label(d_f3,text='Selected Image :')
            l4.config(font=('courier',18))
            l4.grid()
            panel = Label(d_f3, image=img)
            panel.image = img
            panel.grid()
            hidden_data = self.decode(myimg)
            l2 = Label(d_f3, text='Hidden data is :')
            l2.config(font=('courier',18))
            l2.grid(pady=10)
            text_area = Text(d_f3, width=50, height=10)
            text_area.insert(INSERT, hidden_data)
            text_area.configure(state='disabled')
            text_area.grid()
            back_button = Button(d_f3, text='Cancel', command= lambda
:self.page3(d_f3))
            back_button.config(font=('courier',11))
            back_button.grid(pady=15)
```

```
        back_button.grid()
        show_info = Button(d_f3,text='More
Info',command=self.info)
        show_info.config(font=('courier',11))
        show_info.grid()
        d_f3.grid(row=1)
        d_f2.destroy()

    def decode(self, image):
        data = ''
        imgdata = iter(image.getdata())

        while (True):
            pixels = [value for value in imgdata.__next__()[:3] +
                    imgdata.__next__()[:3] +
                    imgdata.__next__()[:3]]
            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


    def frame1_encode(self,f):
        f.destroy()
        f2 = Frame(root)
        label_art = Label(f2, text='\'\(°Ω°)/\'')
        label_art.config(font=('courier',70))
        label_art.grid(row =1,pady=50)
        l1= Label(f2,text='Select the Image in which \nyou want to hide
text :')
        l1.config(font=('courier',18))
        l1.grid()


        bws_button = Button(f2,text='Select',command=lambda :
self.frame2_encode(f2))
        bws_button.config(font=('courier',18))
        bws_button.grid()
        back_button = Button(f2, text='Cancel', command=lambda :
Stegno.home(self,f2))
        back_button.config(font=('courier',18))
        back_button.grid(pady=15)
        back_button.grid()
        f2.grid()
```

```
    def frame2_encode(self,f2):
        ep= Frame(root)
        myfile = tkinter.filedialog.askopenfilename(filetypes = ([('png',
'*.png'),('jpeg', '*.jpeg'),('jpg', '*.jpg'),('All Files', '*.*')]))
        if not myfile:
            messagebox.showerror("Error","You have selected nothing
!")
        else:
            myimg = Image.open(myfile)
            myimage = myimg.resize((300,200))
            img = ImageTk.PhotoImage(myimage)
            l3= Label(ep,text='Selected Image')
            l3.config(font=('courier',18))
            l3.grid()
            panel = Label(ep, image=img)
            panel.image = img
            self.output_image_size = os.stat(myfile)
            self.o_image_w, self.o_image_h = myimg.size
            panel.grid()
            l2 = Label(ep, text='Enter the message')
            l2.config(font=('courier',18))
            l2.grid(pady=15)
            text_area = Text(ep, width=50, height=10)
```

```
        text_area.grid()
        encode_button = Button(ep, text='Cancel',
command=lambda : Stegno.home(self,ep))
        encode_button.config(font=('courier',11))
        data = text_area.get("1.0", "end-1c")
        back_button = Button(ep, text='Encode', command=lambda :
[self.enc_fun(text_area,myimg),Stegno.home(self,ep)])
        back_button.config(font=('courier',11))
        back_button.grid(pady=15)
        encode_button.grid()
        ep.grid(row=1)
        f2.destroy()


  def info(self):
     try:
        str = 'original image:-\nsize of original image:{}mb\nwidth:
{}\nheight: {}\n\n' \
              'decoded image:-\nsize of decoded image: {}mb\nwidth:
{}' \
             '\nheight:
{}'.format(self.output_image_size.st_size/1000000,
                        self.o_image_w,self.o_image_h,
                        self.d_image_size/1000000,
                        self.d_image_w,self.d_image_h)
```

```
            messagebox.showinfo('info',str)
        except:
            messagebox.showinfo('Info','Unable to get the information')
    def genData(self,data):
        newd = []

        for i in data:
            newd.append(format(ord(i), '08b'))
        return newd

    def modPix(self,pix, data):
        datalist = self.genData(data)
        lendata = len(datalist)
        imdata = iter(pix)
        for i in range(lendata):
            # Extracting 3 pixels at a time
            pix = [value for value in imdata.__next__()[:3] +
                   imdata.__next__()[:3] +
                   imdata.__next__()[:3]]
            # Pixel value should be made
            # odd for 1 and even for 0
            for j in range(0, 8):
                if (datalist[i][j] == '0') and (pix[j] % 2 != 0):
```

```
                if (pix[j] % 2 != 0):
                    pix[j] -= 1

            elif (datalist[i][j] == '1') and (pix[j] % 2 == 0):
                pix[j] -= 1
        # Eigh^th pixel of every set tells
        # whether to stop or read further.
        # 0 means keep reading; 1 means the
        # message is over.
        if (i == lendata - 1):
            if (pix[-1] % 2 == 0):
                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(self,newimg, data):
    w = newimg.size[0]
    (x, y) = (0, 0)
```

```
        for pixel in self.modPix(newimg.getdata(), data):

            # Putting modified pixels in the new image
            newimg.putpixel((x, y), pixel)
            if (x == w - 1):
                x = 0
                y += 1
            else:
                x += 1

    def enc_fun(self,text_area,myimg):
        data = text_area.get("1.0", "end-1c")
        if (len(data) == 0):
            messagebox.showinfo("Alert","Kindly enter text in TextBox")
        else:
            newimg = myimg.copy()
            self.encode_enc(newimg, data)
            my_file = BytesIO()

temp=os.path.splitext(os.path.basename(myimg.filename))[0]

newimg.save(tkinter.filedialog.asksaveasfilename(initialfile=temp,fi
letypes = ([('png', '*.png')]),defaultextension=".png"))
            self.d_image_size = my_file.tell()
```

```
            self.d_image_w,self.d_image_h = newimg.size
            messagebox.showinfo("Success","Encoding Successful\nFile
is saved as Image_with_hiddentext.png in the same directory")


    def page3(self,frame):
        frame.destroy()
        self.main(root)
root = Tk()
o = Stegno()
o.main(root)
root.mainloop()
```

CHAPTER 1

INTRODUCTION

1.1 PROJECT MOTIVATION

This report discusses the process of Image steganography.It is the practice of hiding secret information within an image in such a way that it is not easily detectable by anyone who does not know where to look. This technique involves modifying the pixels of an image to embed hidden messages, which can be encrypted to provide an extra layer of security. The process typically involves dividing the image into small blocks and then

modifying one or more bits of each block to encode the hidden message.

The modified image appears almost identical to the original, making it difficult for anyone who is not aware of the hidden information to detect its presence. Image steganography has many applications, including secure communication, digital watermarking, and copyright protection. As technology continues to evolve, so too will the techniques used in image steganography, ensuring that this valuable tool remains relevant and effective in today's digital world.

1.2 PROBLEM STATEMENT AND OBJECTIVES

PROBLEM STATEMENT:

To Encode A text in the image and decode it with the code?

OBJECTIVES:

1. Develop a Python code that load the image for Decoding.
2. Use ByteIO module for decoding and encoding purposes.
3. The save the Encoded Image in a new name.
4. Decoded it Separately and result will be obtained.

1.2 SCOPE AND APPLICATIONS

SCOPE OF THE PROJECT:

Image Steganography is the technique of hiding confidential information in an image file. It is used to protect sensitive data such as passwords, credit card numbers, and other confidential information. The scope of a project in Image Steganography includes understanding the concept of steganography, researching current methods and techniques in steganography, and developing new approaches and methods for hiding data in images. The project also involves creating a user interface for the steganography process, testing the developed methods, and evaluating the security of the steganography process. Furthermore, the project should include an analysis and comparison of various existing methods of steganography and the development of a theoretical framework for steganography. Finally, the project should also investigate the legal implications of using steganography and the potential for abuse .

1.3 LIMITATIONS

Susceptibility to Data Loss:

The hidden message may be lost or distorted during the transmission or processing of the image, resulting in a loss of data. Misuse: Steganography can be misused for illegal activities, including hiding malicious code or malware within an image, making it difficult to detect and prevent cybersecurity attacks.

The main limitation is the maximum size of the embedded data compared to the total data. If a piece of data is already very

compressed it might be wholly impossible to embed additional data in it. And even under ideal conditions you will rarely get more than 20% out of the carrier data.

I assume of course that the data hidden is encrypted first, making it appear completely random even to statistical analysis programs. This reduces those (ideal) 30% by half on average.

So assume that you use a bunch of image files of moderate compression level as carrier medium. Lets say you get on average 15% out of it, lets say the total batch of images has a size of 1GB.

This means that after encrypting and embedding your data, you will be able to transport 75MB of hidden data. This is not a lot.

Steganography is in general only used in situations where there is no other alternative because the very fact that A and B are communicating would lead to grave consequences. Trying to swap and share files for example is not such a situation.

CHAPTER 2

PROJECT ARCHITECTURE, DESIGN AND IMPLEMENTATION

The methodology of the project involves the following steps, algorithm and techniques for comparing the differences in the histogram of the two images generated by the Python code.

2.1 IMAGE LOADING AND PREPROCESSING:

The askopenfilename() function is being utilized in reading the images either as grayscale or RGB. In order to maintain the consistency in the image quality and dimensions, preprocessing is done such as cropping, resizing and normalization if needed.

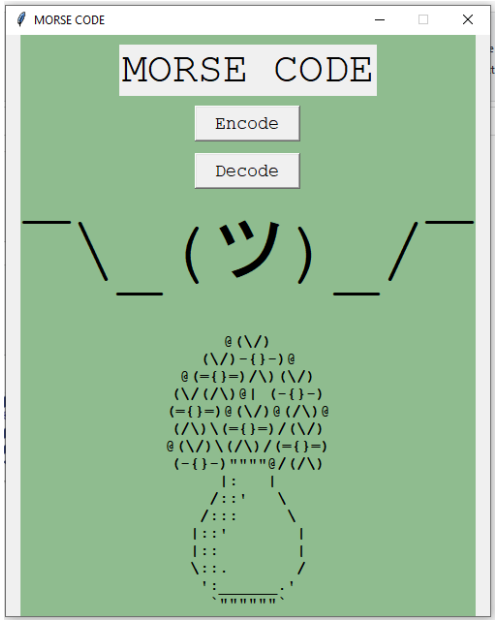
2.2 DECODING AND ENCODING:

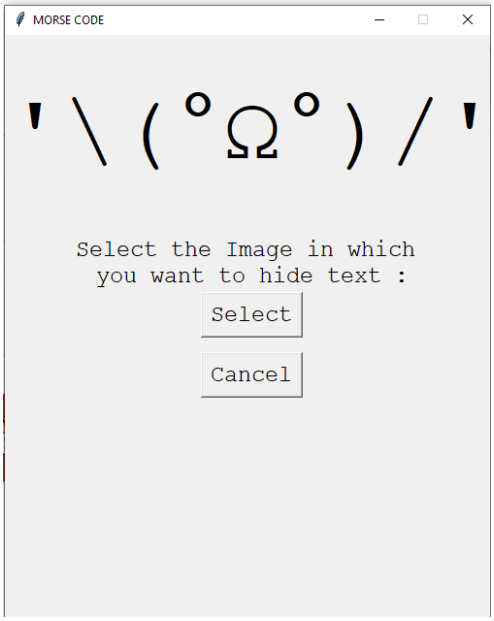
Encoding and decoding are used in many forms of communications, including computing, data communications, programming, digital electronics and human communications. These two processes involve changing the format of content for optimal transmission or storage.

In computers, encoding is the process of putting a sequence of characters (letters, numbers, punctuation, and certain symbols) into a specialized format for efficient transmission or storage. Decoding is the opposite process -- the conversion of an encoded format back into the original sequence of characters.

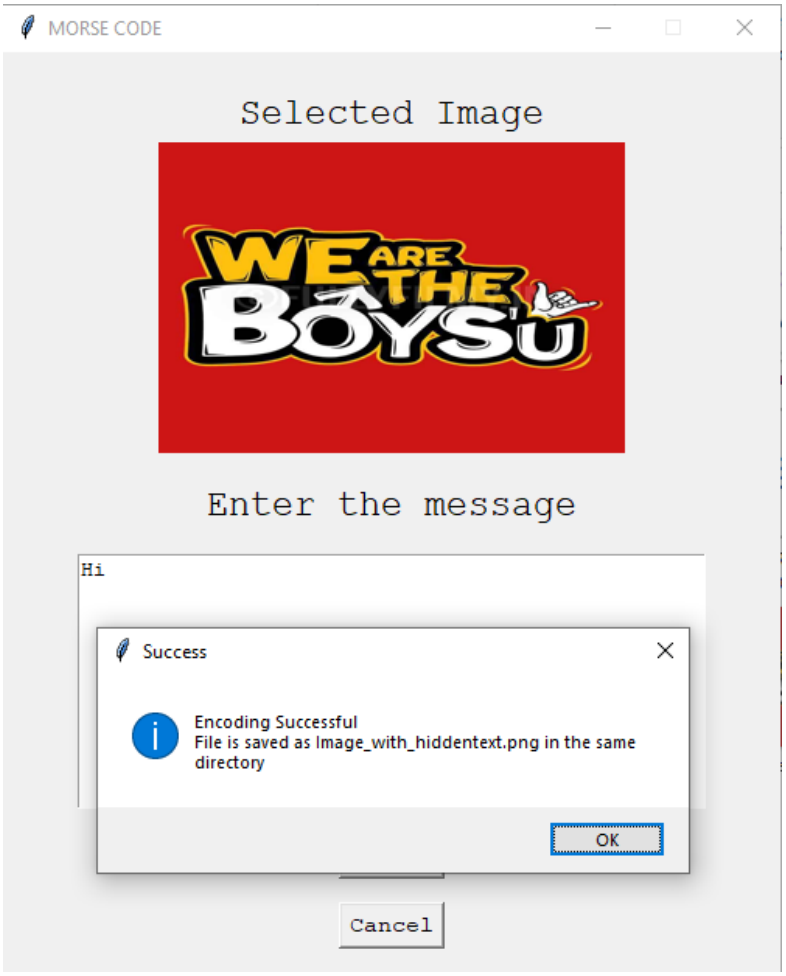
These terms should not be confused with *encryption* and *decryption*, which focus on hiding and securing data. (We can encrypt data without changing the code or encode data without deliberately concealing the content.)

INPUT

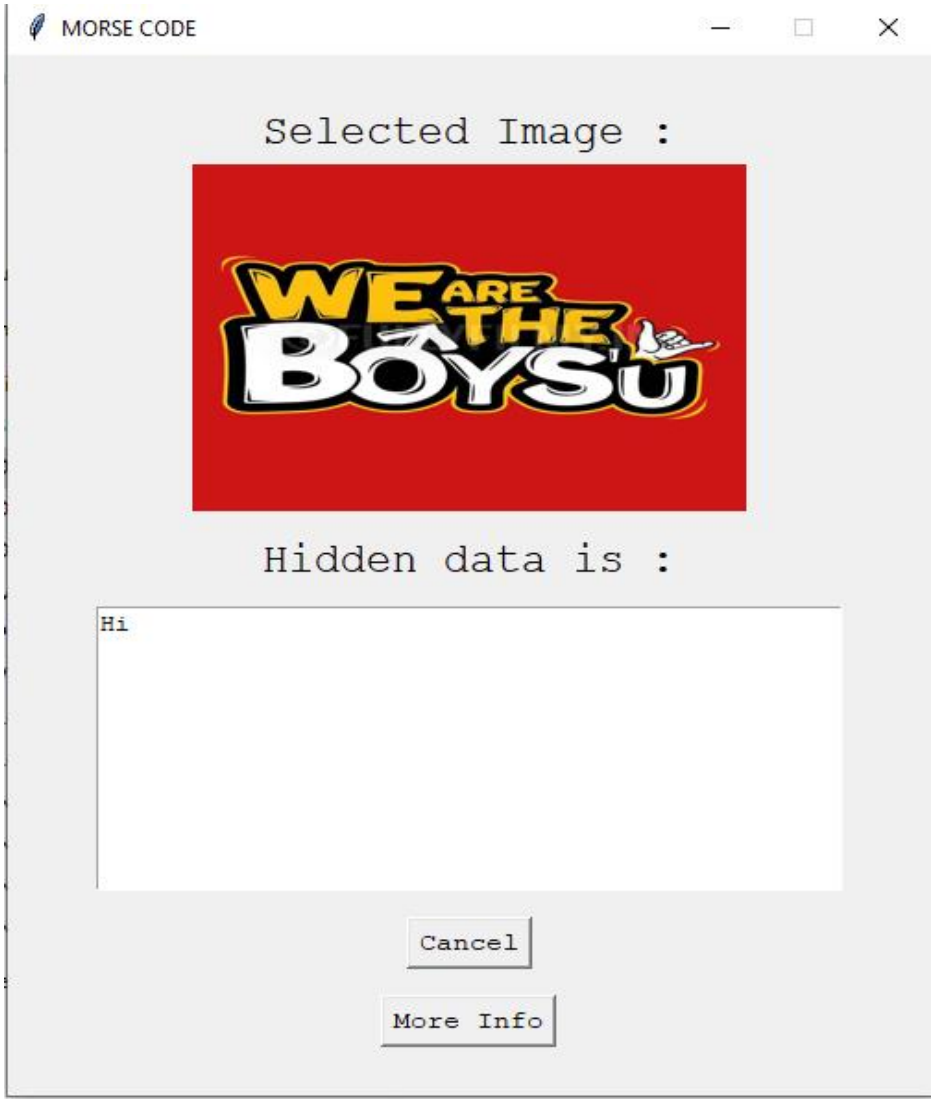

MORSE CODE
MORSE CODE
Encode
Decode
¯_(ツ)_/¯

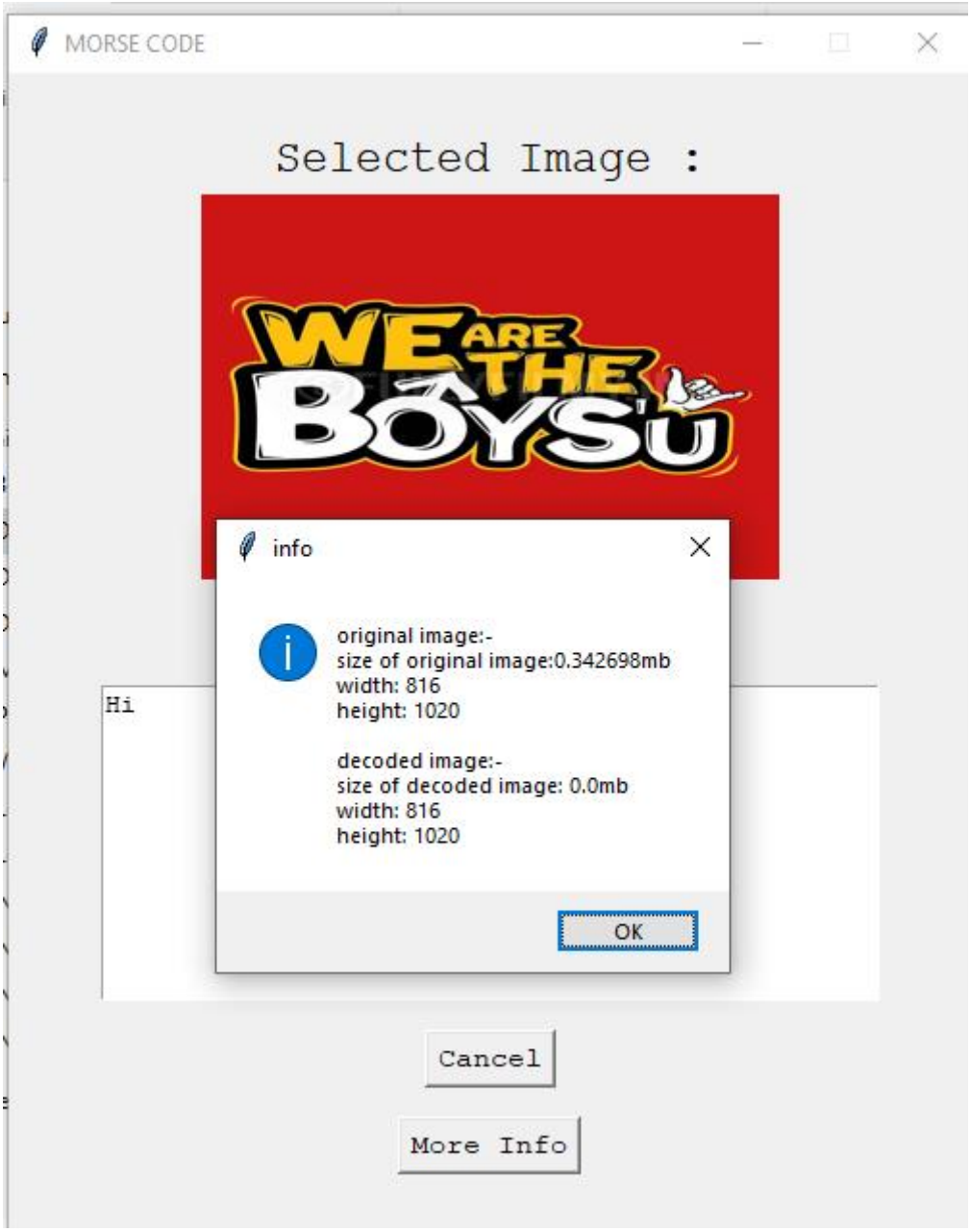

MORSE CODE
' \ (° Ω °) / '
Select the Image in which
you want to hide text :
Select
Cancel


MORSE CODE
Selected Image
WE ARE THE BOYSU
Enter the message
sdcjck
Encode
Cancel


MORSE CODE
Selected Image
WE ARE THE BOYS'U
Enter the message
Hi
Success
Encoding Successful
File is saved as Image_with_hiddentext.png in the same directory
OK
Cancel

OUTPUT


MORSE CODE
Selected Image :
WE ARE THE BOYS
Hidden data is :
Hi
Cancel
More Info



REFERENCE

https://r.search.yahoo.com/_ylt=AwrKDrRyeXxkR8A8_6G7HAx.;_ylu=Y29sbwNzZzMEcG9zAzEEdnRpZAMEc2VjA3Ny/RV=2/RE=1685907954/RO=10/RU=https%3a%2f%2fieeexplore.ieee.org%2fdocument%2f9335027/RK=2/RS=MHXFckHYGw3NA14T3cd4_f.852c-

https://r.search.yahoo.com/_ylt=AwrKDrRyeXxkR8A8B6K7HAx.;_ylu=Y29sbwNzZzMEcG9zAzMEdnRpZAMEc2VjA3Ny/RV=2/RE=1685907954/RO=10/RU=https%3a%2f%2fieeexplore.ieee.org%2fdocument%2f9711628%2f/RK=2/RS=CuDgNnY1t4XZETLQDS8.2aHl8EQ-

A Systematic Review of Computational Image Steganography ..