МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ

Основы стеганографии

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

«Встраивание информации в картинки»

Выполнил:
Студент группы N3351
Фам Хю Хоанг

Проверил: ассистент ФБИТ,
Университет ИТМО,
Давыдов Вадим Валерьевич

Санкт-Петербург

2020

Цель работы:

Целью данной лабораторной работы :

- Применение метода LSB для сокрытия
- Извлечение сообщения из стегоконтейнера
- Учет PSNR

Теоретическая часть:

LSB[1] -суть этого метода заключается в замене последних значащих битов в контейнере (изображения, аудио или видеозаписи) на биты скрываемого сообщения. Разница между пустым и заполненным контейнерами должна быть не ощутима для органов восприятия человека.

Суть метода заключается в следующем: Допустим, имеется 8-битное изображение в градациях серого. 00h обозначает чёрный цвет, FFh (11111111b) белый. Всего имеется 256 . Также предположим, что сообщение состоит из 1 байта например, 01101011b. При использовании 2 младших бит в описаниях пикселей, нам потребуется 4 пикселя. Допустим, они чёрного цвета. Тогда пиксели, содержащие скрытое сообщение, будут выглядеть следующим образом: 00000001 00000010 00000010 00000011. Тогда цвет пикселей изменится: первого — на 1/255, второго и третьего — на 2/255 и четвёртого — на 3/255. Такие градации, мало того, что незаметны для человека, могут вообще не отобразиться при использовании низкокачественных устройств вывода.

Методы LSB являются неустойчивыми ко всем видам атак и могут быть использованы только при отсутствии шума в канале передачи данных.

Обнаружение LSB-кодированного стего осуществляется по аномальным характеристикам распределения значений диапазона младших битов отсчётов цифрового сигнала.

Все методы LSB являются, как правило, аддитивными (A17 (Cox), L18D (Lange)).

BMP[2]-формат хранения растровых изображений, разработанный компанией Microsoft.В данном формате можно хранить только однослойные растры. На каждый пиксель в разных файлах может приходиться разное количество бит (глубина цвета). Microsoft предлагает битности 1, 2, 4, 8, 16, 24, 32, 48 и 64. В битностях 8 и ниже цвет указывается индексом из таблицы цветов (палитры), а при бóльших — непосредственным значением. Цвет же в любом случае можно задать только в цветовой модели RGB (как при непосредственном указании в пикселе, так и в таблице цветов), но в битностях 16 и 32 можно получить Grayscale с глубиной до 16 и 32 бит, соответственно. Частичная прозрачность реализована альфа-каналом различных битностей, но при этом прозрачность без градаций можно косвенно получить RLE-кодированием.

PSNR[3]-Пиковое отношение сигнала к шуму обозначается аббревиатурой PSNR и является инженерным термином, означающим соотношение между максимумом возможного значения сигнала и мощностью шума, искажающего значения сигнала. Поскольку многие сигналы имеют широкий динамический диапазон, PSNR обычно измеряется в логарифмической шкале в децибелах.

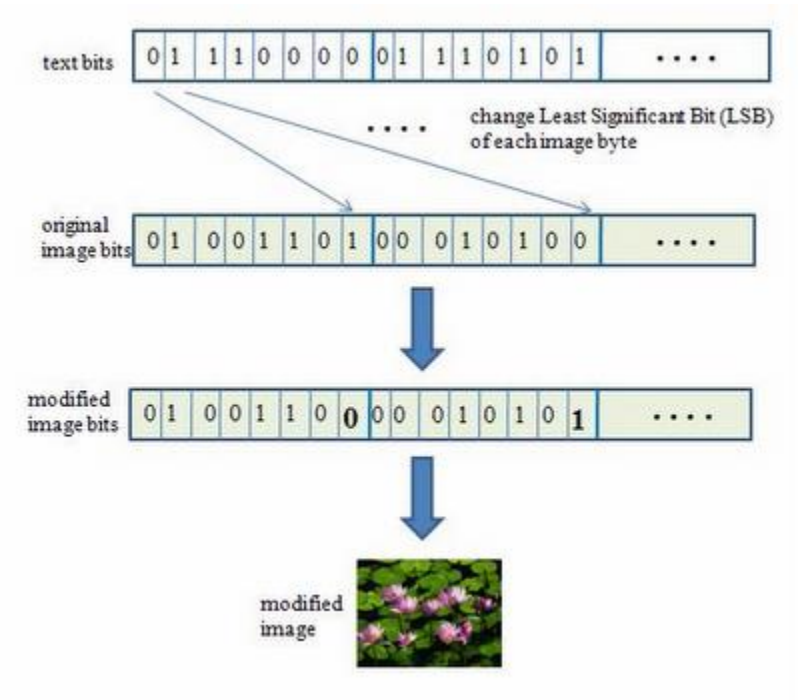
PSNR наиболее часто используется для измерения уровня искажений при сжатии изображений. Проще всего его определить через среднеквадратичную ошибку (СКО) или MSE

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}|I(i,j) - K(i,j)|^2$$

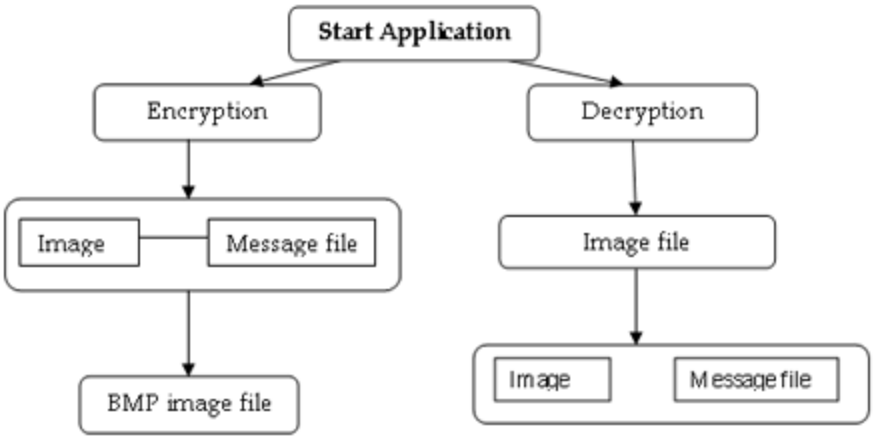
PSNR определяется так:

$$PSNR = 10\log_{10}\left(\frac{MAX_I^2}{MSE}\right) = 20\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right)$$

Блок-схема алгоритма:




```
Pixels: (00100111 11101001 11001000)
        (00100111 11001000 11101001)
        (11001000 00100111 11101001)
A: 01000001
Result: (00100110 11101001 11001000)
        (00100110 11001000 11101000)
        (11001000 00100111 11101001)
```



Практическая часть:

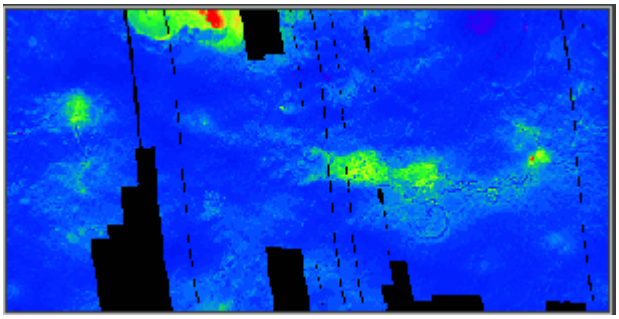
Программа была написана на языке Python. Программа делится на 3 модуля:

1. Стеганография
2. Пойск секретной информации
3. PSNR

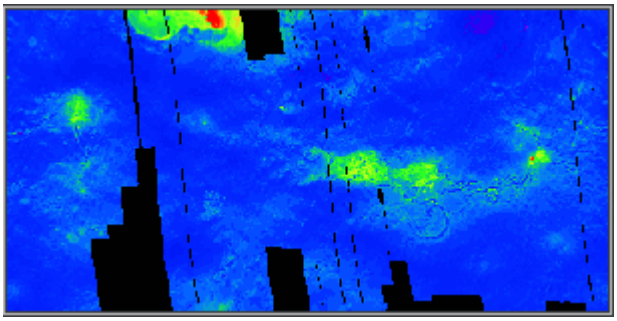
1.Steganography

```
E:\курс 3\ki2\Основы стеганофии\lab\lab2\code>lab2.py
-------------------------Steganography LSB-------------------------------------
1.steganography
2.Find secret text from image
3.PSNR

1
Enter file image!
input.bmp
Enter secret text!
university ITMO Saint-Petersburg Russia

E:\курс 3\ki2\Основы стеганофии\lab\lab2\code>
```

file input.bmp



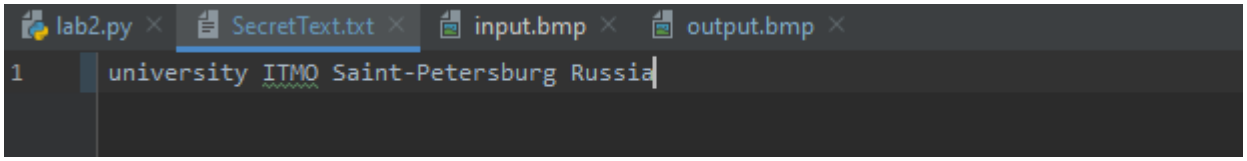
File output.bmp



2.Find secret text from image

```
E:\курс 3\ki2\Основы стеганофии\lab\lab2\code>lab2.py
------------------------Steganography LSB-------------------------------------
1.steganography
2.Find secret text from image
3.PSNR

2
Enter file image!
output.bmp
Please Open file SecretText!

E:\курс 3\ki2\Основы стеганофии\lab\lab2\code>
```

File Secret text

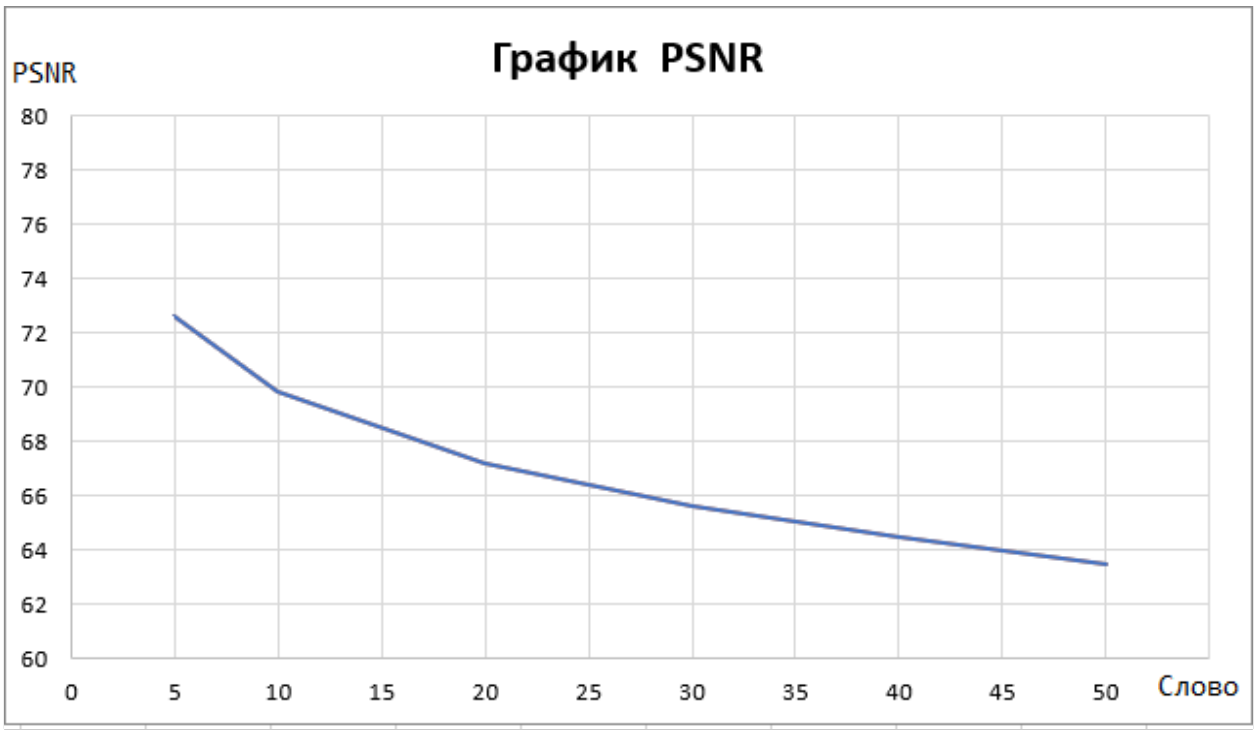


3.PSNR

```
E:\курс 3\ki2\Основы стеганофии\lab\lab2\code>lab2.py
------------------------Steganography LSB-------------------------------------
1.steganography
2.Find secret text from image
3.PSNR

3
PSNR:
72.56777860100624

E:\курс 3\ki2\Основы стеганофии\lab\lab2\code>
```

Построить график PSNR :

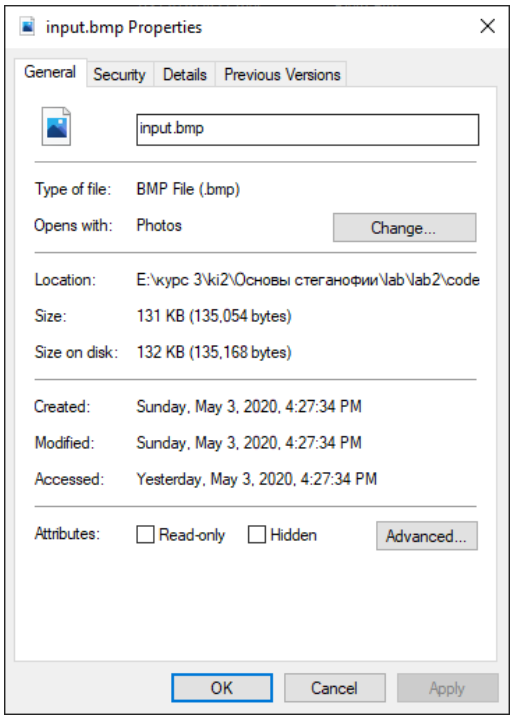
Слово	PSNR
5	72.64896
10	69.80571
20	67.18881
30	65.59419
40	64.43865
50	63.49685



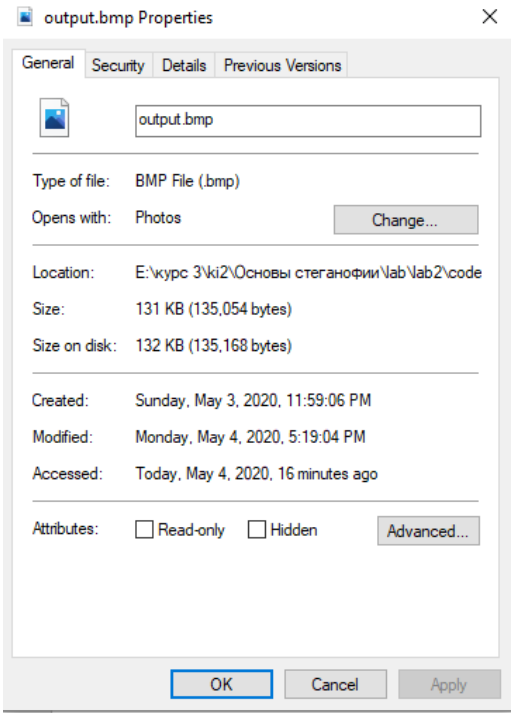
Можно сделать вывод ,что при увеличении количества слова PSNR уменьшится

Проведение экспертной оценки:

Картинка назад встраивание информации



После встраивания информации



Можно сделать вывод, что объем картинки не изменяется. Итак, все требования безопасности соблюдены, и две картинки, в одной из которых десятая часть информации замещена произвольными данными, практически неразличимы.

Выводы:

При выполнении данной лабораторной работы мною были изучены метод LSB. Я научился применять их и проводить последующую оценку их применению. По результатам работы были сделаны следующие выводы:

В результате мы получим новый оттенок, очень похожий на исходный. Эти цвета трудно различить даже на большой по площади заливке, хотя разница будет заметна по одной отдельной точке. Как показывает практика, замена двух младших битов не воспринимается человеческим глазом. В случае необходимости можно занять и три разряда, что весьма незначительно скажется на качестве картинки**.**

Список использованной литературы:

1. LSB[Электронный ресурс] – URL: https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B5%D0%B3%D0%B0%D0%BD%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F#%D0%9C%D0%B5%D1%82%D0%BE%D0%B4_LSB

2. BMP[Электронный ресурс] – URL: https://ru.wikipedia.org/wiki/BMP

3. PSNR[Электронный ресурс] – URL: https://ru.wikipedia.org/wiki/%D0%9F%D0%B8%D0%BA%D0%BE%D0%B2%D0%BE%D0%B5_%D0%BE%D1%82%D0%BD%D0%BE%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D1%81%D0%B8%D0%B3%D0%BD%D0%B0%D0%BB%D0%B0_%D0%BA_%D1%88%D1%83%D0%BC%D1%83

Приложение

```
from PIL import Image
import math

def ConvertStringToBinary(string):
    binary= ''
    for i in string:
        binary+=(format(ord(i), '08b'))
    return binary

def ConvertBinarryToString(binary):
    string = ""
    for i in range(0,len(binary),8):
        integer = int(binary[i:i+8], 2)
        character = chr(integer)
        string += character
    return string


def ConvertDecimalToBinary(number):
    binary=[]
    binray= "{0:016b}".format(number)
    return binray

def ConvertBinaryToDecimal(binary):
    dec=int(binary, 2)
    return dec


def ConvertDataToList(data,lenght):
    dataList=[]
    for i in range(0,lenght):
        dataList+=list(data[i])
    return dataList

def ModifyData(data,message):
    lenghtMessageBin=ConvertDecimalToBinary(len(message))
    for i in range(0,16):
        if(lenghtMessageBin[i]=='0'):
          if(data[i]%2==1):
            data[i]=data[i]-1
        else:
            if(data[i]%2==0):
```

```
                data[i]=data[i]+1

    messageBinary=ConvertStringToBinary(message)
    for i in range(0,len(messageBinary)):
        if(messageBinary[i]=='0'):
            if(data[i+16]%2==1):
                data[i+16]=data[i+16]-1
        else:
            if(data[i+16]%2==0):
                data[i+16]=data[i+16]+1
    return data


def ReadData(data):
    lengtMessageBinary=''
    for i in range(0,16):
        if(data[i]%2==0):
            lengtMessageBinary+='0'
        else:
            lengtMessageBinary+='1'
    lenghtMessage=ConvertBinaryToDecimal(lengtMessageBinary)
    message=''
    for i in range(16,16+lenghtMessage*8):
        if(data[i]%2==0):
            message+='0'
        else:
            message+='1'

    return message


def Encryto(fileImage,message):
    image=Image.open(fileImage,'r')
    newImage=image.copy()
    width,height=newImage.size
    data=ConvertDataToList(list(newImage.getdata()),height*width)

    data=ModifyData(data,message)


    dataImage=[]
    for j in range(0,height*width*3,3):
        dataImage.append(tuple(data[j:j+3]))
    newImage.putdata(dataImage)
    newImage.save('output.bmp')
```

```
    image.close()
    newImage.close()

def Decrypto(fileImage):
    image=Image.open(fileImage,'r')
    width,height =image.size
    data=ConvertDataToList(list(image.getdata()),height*width)
    message=ConvertBinarryToString(ReadData(data))
    file_secret = open('SecretText.txt',mode = 'w',encoding='UTF-8')
    file_secret.write(message)
    file_secret.close()
    image.close()

def PSNR(image1,image2):
    imageAfer=Image.open(image1,'r')
    imageBefor=Image.open(image2,'r')
    width,height=imageAfer.size
    dataAfter=ConvertDataToList(list(imageAfer.getdata()),height*width)
    dataBefor=ConvertDataToList(list(imageBefor.getdata()),height*width)
    sum=0
    for i in range(0,len(dataAfter)):
        sum+=math.pow((dataAfter[i]-dataBefor[i]),2)
    MSE=sum/(width*height)
    PSNR=10*math.log((255*255/MSE),10)
    return PSNR



 ##---------------------------------------main programme-------------------------
----------------------------------------
print('-------------------------Steganography LSB--------------------------------
------')
print ('1.steganography'+'\n'+'2.Find secret text from image \n'+'3.PSNR\n')
option=int(input())
if(option==1):
     image=input('Enter file image!\n')
     message=input('Enter secret text!\n')
     Encryto(image,message)
elif( option==2):
     Decrypto(input('Enter file image!\n'))
     print ('Please Open file SecretText!')
elif(option==3):
    print ('PSNR:')
    print (PSNR('input.bmp','output.bmp'))
```