LSB图像隐写和峰值信噪比计算的python实现

原创

lioner299 于 2021-10-16 10:50:07 发布

分类专栏： LSB算法信息隐藏与提取 文章标签： python pygame matlab 图像处理

本文链接：https://blog.csdn.net/lioner299/article/details/120790922

LSB图像隐写的python实现

【 实验目的】

了解信息隐藏中最常用的LSB算法特点，掌握LSB算法原理，设计并实现一种基于图像的LSB隐藏算法；了解如何通过峰值信噪比来对图像质量进行客观评价，并计算峰值信噪比。

【实验环境】

(1) Windows 7 以上操作系统；
(2) Python3.8、Pycharm 2021.2.2集成开发环境；
(3) PNG 彩色无损压缩图像文件。

【原理简介】

任何多媒体信息，在数字化时，都会产生物理随机噪，而人的感观系统对这些随机噪声不敏感。信息隐藏技术就是利用这个原理，通过使用秘密信息比特替换随机噪声，从而完成信息隐藏目标。
BMP 灰度图像的每个像素值为 8bit 二进制值，表示该点亮度。不同位平面对视觉影响不同，图像高位平面对图像感官质量起主要作用，去除图像最低几个位平面并不会造成画面质量的明显下降。利用这个原理可用秘密信息(或称水印信息)替代载体图像低位平面以实现信息嵌入。
本实验中算法选用最低位平面来嵌入秘密信息。最低位平面对图像的视觉效果影响最轻微，但很容易受噪声影响和攻击，解决办法可采用冗余嵌入的方式来增强稳健性。即在一个区域(多个像素)中嵌入相同的信息，提取时根据该区域中的所有像素判断。

【主要代码】

- **隐藏加密部分**

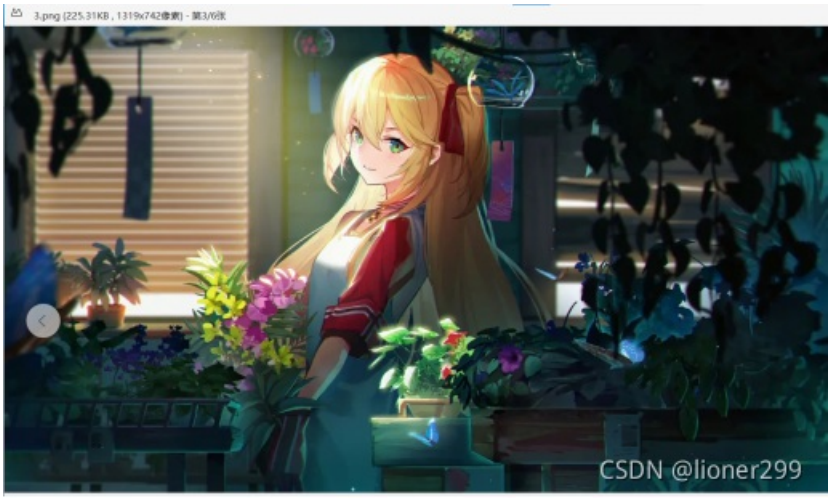
```
from PIL import Image
def plus(st):
    return st.zfill(8)
def get_key(strr):  # 实现对文字信息转成二进制
    tmp = strr
    with open(tmp,"r",encoding='utf-8') as f:
     str = ''
        s = f.read()
        for i in s:  # 这里i直接由文本转换成ASCII格式
            if isinstance(i,int):
                str = str+plus(bin(i).replace('0b',''))
```

```
            else:
                str = str+plus(bin(ord(i)).replace('0b',''))
    return str
def mod(x,y): # 模除运算
    return x%y
def encry(str1,str2,str3):
    im = Image.open(str1)
    width = im.size[0]  # 获取图片的长,宽类似
    print("length:"+str(width)+"\n")
    height = im.size[1]
    print("width:"+str(height)+"\n")
    count = 0
    key = get_key(str2)
#     print(len(key))
    keylen = len(key)
    test=''
    test2=''
    for w in range(0,width):
        if count == keylen:
                break
        for h in range(0,height):
            pixel = im.getpixel((h,w))  # 获取指定长宽坐标像素的RGB值
#             print(pixel)
            a = pixel[0]
            b = pixel[1]
            c = pixel[2]
            if count == keylen:
                break
            test=test+str(mod(a,2))
            a= a-mod(a,2)+int(key[count])
            test2=test2+str(mod(a,2))
            count += 1
            if count == keylen:
                im.putpixel((h,w),(a,b,c))  # 将指定坐标的像素值用（a，b，c）代替
                break

            test=test+str(mod(b,2))
            b = b - mod(b,2) + int(key[count])
            test2=test2+str(mod(b,2))
            count += 1
            if count == keylen:
                im.putpixel((h,w),(a,b,c))
                break

            test=test+str(mod(c,2))
            c = c - mod(c,2) + int(key[count])
            test2=test2+str(mod(c,2))
            count += 1
            if count == keylen:
                im.putpixel((h,w),(a,b,c))
                break
            if count % 3 == 0:
                im.putpixel((h,w),(a,b,c))
    im.save(str3)
    print('嵌入前：'+test)
    print('嵌入后：'+test2)
```

需要注意的是文本信息对应的编码格式要写对（~~错了好几次，难过~~）



上图是嵌入前的图片，下图是嵌入后的图片（能看出区别吗，不能。因为~~我放了两张一样的上去~~ LSB算法就是利用的视觉冗余原理，能看出来你就是下一个最强大脑啦）

- **提取部分**

```
def toasc(strr):
    return int(strr,2)

def dest(le,str1,str2):  #le为所要提取的信息的长度，str1为加密载体图片的路径，str2为提取文件的保存路径
    a = ''
    b = ''
    im = Image.open(str1)
    length = le * 8
    width = im.size[0]
    height = im.size[1]
    print("length:"+str(width)+"\n")
    print("width:"+str(height)+"\n")
    count = 0  # 计数，判断信息是否提取完毕

    for h in range(0,height):
        if count == length:
                break
        for w in range(0,width):
            pixel = im.getpixel((w,h))
#             print('test:',int(str(pixel[0]),2))
            if count == length:
                break
            if count % 3 == 0:
                count += 1  # 已经处理过了一位R/G/B上的信息
                b = b + str(mod(int(pixel[0]),2))  # 由于嵌入时，时按照RGB的顺序对每一位上的三个值进行嵌入，所以第一个要
提取R上的信息
                if count == length:  # 当长度与目标信息的一致时，退出
                    break
            if count % 3 == 1:
                count +=1
                b = b + str(mod(int(pixel[1]),2))
                if count == length:
                    break
            if count % 3 == 2:
                count += 1
                b = b + str(mod(pixel[2],2))
            if count == length:
                break
    with open(str2,'w',encoding='utf-8') as f:
        print('提取: '+b)
        for i in range(0,len(b),8):
#             stra = ''
            stra = toasc(b[i:i+8])
#             print('test: ',isinstance(stra,int))
#             print(stra.decode())
            answera = chr(stra)
            print(answera)
            f.write(answera)
le = 30  # 由于未知长度，先设的大一些
```

结果是这样的：

经过解码得到隐藏信息如下

I am Admiral Hipper class cruiser

- 计算图像峰值信噪比

```
import cv2 as cv
import math
import numpy as np

def psnr1(img1,img2):
    #compute mse
    # mse = np.mean((img1-img2)**2)
    mse = np.mean((img1/1.0-img2/1.0)**2)
    #compute psnr
    if mse < 1e-10:
        return 100
    psnr1 = 20*math.log10(255/math.sqrt(mse))
    return psnr1

def psnr2(img1,img2):
    mse = np.mean((img1/255.0-img2/255.0)**2)
    if mse < 1e-10:
        return 100
    psnr2 = 20*math.log10(1/math.sqrt(mse))
    return psnr2

imag1 = cv.imread('D://Machine Learning//data set//1234//3.jpg')# 图片路径不能有中文，否则会打不开，报错“ Nonetype……
”
print (imag1.shape)
imag2 = cv.imread('D:/Machine Learning/data set/1234/answer.jpg')
#print(imag2.shape)
# imag2 = imag2.reshape(352,352,3)
#print(imag2.shape)
res1 = psnr1(imag1,imag2)
print("res1:",res1)
res2 = psnr2(imag1,imag2)
print("res2:",res2)
```

关于import cv2报错的问题，主要在库的安装上
opencv-python的安装可以参考这篇文章

> 如果觉得有用的话，可以点一波关注呀~