Python 图片隐写术

原创

Marpylili 已于 2022-01-20 17:14:16 修改 33 收藏

分类专栏： python项目练习 文章标签： python

于 2021-06-25 18:14:52 首次发布


```
from PIL import Image

"""
取得一个 PIL 图像并且更改所有值为偶数使（最低有效位为 0）
"""


def makeImageEven(image):
    pixels = list(image.getdata())  # 得到一个这样的列表： [(r,g,b,t),(r,g,b,t)...]
    evenPixels = [(r >> 1 << 1, g >> 1 << 1, b >> 1 << 1, t >> 1 << 1) for [r, g, b, t] in pixels]  # 最低有效位
为 0
    evenImage = Image.new(image.mode, image.size)  # 创建一个相同大小的图片副本
    evenImage.putdata(evenPixels)  # 把上面的像素放入到图片副本
    return evenImage


"""
内置函数 bin() 的替代，返回固定长度的二进制字符串
"""


def constLenBin(int):
    binary = "0" * (8 - (len(bin(int)) - 2)) + bin(int).replace('0b',
                                                                '')  # 去掉 bin() 返回的二进制字符串中的 '0b'，并在
左边补足 '0' 直到字符串长度为 8
    return binary


"""
将字符串编码到图片中
"""
```

```
def encodeDataInImage(image, data):
    evenImage = makeImageEven(image)  # 获得最低有效位为 0 的图片副本
    binary = ''.join(map(constLenBin, bytearray(data, 'utf-8')))  # 将需要被隐藏的字符串转换成二进制字符串
    if len(binary) > len(image.getdata()) * 4:  # 如果不可能编码全部数据， 抛出异常
        raise Exception("Error: Can't encode more than " + len(evenImage.getdata()) * 4 + " bits in this image.
")
    encodedPixels = [(r + int(binary[index * 4 + 0]), g + int(binary[index * 4 + 1]), b + int(binary[index * 4 +
 2]),
                      t + int(binary[index * 4 + 3])) if index * 4 < len(binary) else (r, g, b, t) for
                     index, (r, g, b, t) in enumerate(list(evenImage.getdata()))]  # 将 binary 中的二进制字符串信
息编码进像素里
    encodedImage = Image.new(evenImage.mode, evenImage.size)  # 创建新图片以存放编码后的像素
    encodedImage.putdata(encodedPixels)  # 添加编码后的数据
    return encodedImage


"""
从二进制字符串转为 UTF-8 字符串
"""


def binaryToString(binary):
    index = 0
    string = []
    rec = lambda x, i: x[2:8] + (rec(x[8:], i - 1) if i > 1 else '') if x else ''
    # rec = lambda x, i: x and (x[2:8] + (i > 1 and rec(x[8:], i-1) or '')) or ''
    fun = lambda x, i: x[i + 1:8] + rec(x[8:], i - 1)
    while index + 1 < len(binary):
        chartype = binary[index:].index('0')  # 存放字符所占字节数，一个字节的字符会存为 0
        length = chartype * 8 if chartype else 8
        string.append(chr(int(fun(binary[index:index + length], chartype), 2)))
        index += length
    return ''.join(string)


"""
解码隐藏数据
"""


def decodeImage(image):
    pixels = list(image.getdata())  # 获得像素列表
    binary = ''.join([str(int(r >> 1 << 1 != r)) + str(int(g >> 1 << 1 != g)) + str(int(b >> 1 << 1 != b)) + str
(
        int(t >> 1 << 1 != t)) for (r, g, b, t) in pixels])  # 提取图片中所有最低有效位中的数据
    # 找到数据截止处的索引
    locationDoubleNull = binary.find('0000000000000000')
    if locationDoubleNull % 8 != 0:
        endIndex = locationDoubleNull + (8 - (locationDoubleNull % 8))  # 将endIndex变成能整除8的数
    else:
        endIndex = locationDoubleNull
    data = binaryToString(binary[0:endIndex])
    return data


encodeDataInImage(Image.open("D:/55.jpg").convert('RGBA'), '666').save('encodeImage.png')
print(decodeImage(Image.open("encodeImage.png")))
```

```
D:\[illegible]
666

Process finished with exit code 0
```