生活·技术·探索

用Python实现图片“隐写术”

谢作如　浙江省温州中学

摘要: 计算机的发明和密码破译有着千丝万缕的关系，信息的编码则是加密和解密的基础。本文以图片隐写术为研究内容，用Python的PIL库实现了在图片中掩藏文字信息，并重新“解密”出来。案例涉及了字符编码、图片编码、进制转换、Python编程等方面的知识，将知识学习融入到有意义的探究过程之中。

关键词: 图像编码；进制；编程

中图分类号: G434　**文献标识码:** A　**论文编号:** 1674—2117（2022）19—0078—03

隐写术(steganography)一词来源于希腊词汇stegons和graphia，意即“隐藏”和“书写”，即把某些特殊信息隐藏于正常载体之中，从而实现掩盖特殊信息存在的事实，不易引起敌方的怀疑。信息隐藏的发展很大程度上得益于战争中隐蔽通信的需要。我国古代有文字可考的最早的信息隐藏见于《六韬》中对“阴符”的记载。其办法是先制造形制、花纹不同的兵符，每一种表示一种固定的含义。这种含义须事先约定好，只有当事人可以理解，即使被敌方截获，他们也不会知道其中的含义。

在教学《数据与计算》(高中信息技术必修模块1)中关于信息加密的时候，笔者给学生介绍了密码的加密解密技术，并在网页上演示了图片隐写术的做法，不想引起了学生极大的学习兴趣。于是笔者就产生了想法：用Python的PIL库来实现图片隐写术，从而让学生深入理解信息的编码原理和规则。

图片隐写术的原理分析

图片是由一个个像素组成的。在常见的彩色图片中，每一个像素都由三组数字表示，代表红色(r)、绿色(g)、蓝色(b)三种颜色。其中，每一种颜色的深浅一般由0～255之间的数字表示，即一个字节(8个Bit)，因而在Windows中查看这类图片的属性，会看到其位深度为“32”，如图1所示。

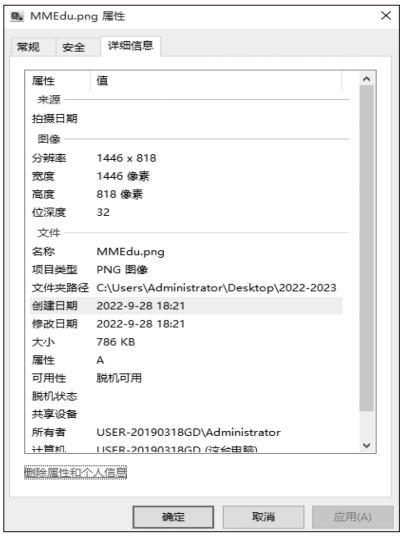

图1

既然每一个像素的单种颜色是采用8位来存储，那么如果改变其最低位的值，对整个图片来说，其视觉效果变化就很小，通过肉眼是看不出来的，如当一组颜色白色(255, 255, 255)，变为(254, 255, 255)时，几乎没有区别，因为差距仅仅是1/255。如下页图2所示的两张小狗，其中一张加入了隐藏的信息，但是几乎看不出任何区别。

那么，对于一张大小为800*600的图片来说，就有800*600*3个位的空间可用于隐藏信息。只要将待隐藏的文字信息转换为二进制，再将每个二进制码存储在图片的像素中，就达到了隐藏信息的目的。

编程实现：将加密信息写入图片

Python的PIL库可以读取图片的信息，包括所有的像素颜色。这就需要编写一段代码，逐步完成如下工作：


原始图片


带加密信息的图片

图2

```
# 将每一个像素颜色的数字最低位变为“0”
def makeImageEven(img):
    pixels = list(img.getdata())  # 得到的列表内容：[(r,g,b),(r,g,b)...]
    ePix = [(r>>1<<1,g>>1<<1,b>>1<<1) for [r,g,b] in pixels]
    return ePix
```

图3

```
# 去掉bin函数返回的二进制字符串中的'0b'，并在左边补足 '0'
def constLenBin(int):
    binary = bin(int)
    binary = "0" * (8-(len(binary)-2)) + binary[2:]
    return binary
```

图4

```
# 将二进制字符串重新加入，形成新的图片
def encodeDataInImage(img, data):
    ePix = makeImageEven(img)
    # 将需要被隐藏的字符串转换成二进制字符串
    info = ''.join(map(constLenBin,bytearray(data, 'utf-8')))
    # 将binary的长度按照3的倍数进行补足
    info = info + '0'* (3-len(info) % 3)
    n = len(info) // 3
    print(f'需要覆盖{n}个像素')
    # 将 binary 中的二进制字符串信息写入像素
    newPixels = [(r+int(info[i*3+0]),g+int(info[i*3+1]),b+int(info[i*3+2])) for
i,(r,g,b) in enumerate(ePix[0:n])]
    print('被覆盖的像素信息：'+ str(newPixels))
    newPixels = newPixels + ePix[n:]
    encodedImage = Image.new(img.mode, img.size)  # 创建新图片
    encodedImage.putdata(newPixels)  # 添加编码后的数据
    return encodedImage
```

图5

①获取原图片的所有像素，将每一个像素颜色的数字最低位变为“0”；

②将待隐藏信息转换为二进制字符串，如果位数不够8个时，在最高位补0；

③将二进制的待隐藏信息依次与图片中每个像素的r、g、b颜色数字进行相加，形成新的图片。

为了方便调用，笔者编写了三个函数，分别实现这些功能，函数代码如图3、图4、图5所示。

准备工作已经完成，接下来要隐藏信息就调用“encodeDataInImage”函数。为了方便理解，笔者在代码中输出了一些特定的信息，如加密后的二进制代码和加密后的像素，具体效果如下页图6所示。

在运行代码后，目录中将多出一张名称为“加密.png”的图片。这就是加密过的图片。用肉眼你肯定看不出有任何变化，因为仅仅改动了17个像素，而且这17个像素的变化又非常小。需要注意的是，选择的“原始.png”图片不支持“透明”，即没有透明度的信息，支持“透明”的PNG文件的每一个像素是4组数字。

代码编写：提取图片中的加密信息

在得到加密后的图片后，下一步的工作就是解密，以验证这一做法是否可行。需要完成如下工作：获取图片的所有像素，将像素值的最低位取出并拼接为一个二进制字符串。需要注意的是，我们只要把“有效”的信息取出即可，如果二进制字符串出现连续的16个以上的0，说明后面就不存在有效信息了。

获取存储的信息的二进制值，按每8位为一组，将上述二进制转换为十进制形式，再转换为Bytes类型，用decode将十进制转为字符并存储到字符串中。

同样为了方便调用，笔者分别

```
In [2]: encodeDataInImage(Image.open("原始.png"), 'ai2022').save('加密.png')

        要写入的密文: 0110000101101001001100100011000000110010001100 10
        补全后的密文: 01100001011010010011001000110000001100100011001 0000
        需要覆盖17个像素
        被覆盖的像素信息: [(50, 51, 45), (50, 50, 44), (52, 53, 44), (55, 55, 46),
        (57, 56, 48), (59, 58, 50), (63, 61, 50), (64, 63, 52), (64, 62, 53), (6
        7, 62, 54), (68, 64, 52), (68, 65, 55), (70, 66, 55), (72, 66, 56), (73,
        67, 54), (72, 67, 54), (72, 70, 60)]
```

图6

```
# 读取像素中隐藏的密文,
def decodeImage(image):
      pixels = list(image.getdata())   # 获得像素列表
      # 提取图片中所有最低有效位中的数据
      binary = ''.join([str(int(r>>1<<1!=r))+str(int(g>>1
<<1!=g))+str(int(b>>1<<1!=b)) for (r,g,b) in pixels])
      # 找到数据截止处的索引
      mark = binary.find('0000000000000000')
      if mark!=-1:
            binary = binary[0:mark]
      return data
```

图7

```
# 将密文转化为正常的文字
def binaryToString(binary):
      print('剥离的密文: ' + binary)
      # 将最后一个元素补足8位,即后面补零
      binary = binary + '0' * (len(binary) % 8)
      # 将二进制字符串按照8个一组形成一个列表
      t = []
      for i in range(0,len(binary),8):
            t.append(binary[i:i+8])
      # 将列表的元素从字符串转换为数字
      new_list = [int(x,2) for x in t]
      # 转换为bytes类型
      mycode = bytes(new_list)
      # 转换为utf-8编码的文字
      s = mycode.decode('utf-8')
      return s
```

图8

```
In [4]: encodeDataInImage(Image.open('原始.png'), '人工智能MMEdu').save('加密.png')
        data = decodeImage(Image.open("加密.png"))
        info = binaryToString(data)
        print(f'图片中隐藏的信息是:{info}')

        要写入的密文: 111001001011101010111010111001011011011110100101111001101001100110111010111010001
        000001110111101010011010100110101000101011001000111 0101
        补全后的密文: 111001001011101010111010111001011011011110100101111001101001100110111010111010001
        000001110111101010011010100110101000101011001000111010100
        需要覆盖46个像素
        被覆盖的像素信息: [(51, 51, 45), (50, 50, 45), (52, 52, 45), (54, 55, 47), (57, 56, 49), (58, 5
        9, 50), (63, 61, 51), (64, 63, 52), (65, 63, 53), (66, 62, 55), (68, 65, 53), (68, 65, 55),
        (70, 67, 55), (73, 67, 56), (73, 66, 54), (73, 66, 55), (73, 71, 61), (74, 70, 63), (75, 72,
        63), (74, 72, 63), (71, 70, 60), (71, 71, 60), (71, 73, 63), (70, 75, 62), (75, 77, 67), (74,
        81, 68), (74, 80, 69), (74, 82, 70), (76, 84, 71), (77, 87, 72), (81, 93, 79), (85, 94, 81),
        (86, 101, 82), (88, 105, 85), (90, 109, 86), (95, 114, 92), (101, 119, 96), (105, 126, 103),
        (110, 134, 108), (115, 138, 113), (122, 147, 121), (126, 150, 125), (132, 156, 128), (141, 16
        1, 133), (144, 165, 136), (149, 166, 140)]
        剥离的密文: 111001001011101010111010111001011011011110100101111001101001100110111010111010001 00
        0001110111101010011010100110101000101011001000111 0101
        图片中隐藏的信息是:人工智能MMEdu
```

图9

编写了两个函数来实现以上功能（如图7、图8）。

分别执行decodeImage和binaryToString两个函数，即可输出解密后的信息，如图9所示。

项目总结

实现图片隐写术的方法很多。本案例仅仅使用了一个像素单色数据的最低位，如果要隐藏更多的信息，甚至可以利用这个字节的后四位。当然，如果使用了后四位来加密，那么用于加密的原始图片的背景颜色最好比较复杂，这样加密后就不容易被发现，不能选择存在大片相同颜色的图片，如蓝天白云的风景图。此外，用于加密的原始图片可以是任意格式，但是保存后的图片需要用PNG或者BMP格式，即未压缩格式，不然加密后的图片一旦被压缩，信息就被破坏了。

计算机的发明和密码破译有着千丝万缕的关系，信息的编码又是加密和解密的基础。本案例涉及了字符编码、图片编码、进制转换、Python编程等方面的知识，将知识学习融入到有意义的探究过程之中，值得作为一个信息技术实验让学生在课堂上研究。本案例代码还有很多细节可以让学生探究，如为什么出现16个0就说明后面的信息是无效的、为什么加密时二进制字符串（binary）的长度要按照3的倍数进行补足等。

此外，信息的加密解密是一个很不错的信息技术学习主题，完全可以作为一个项目让学生深入研究。有趣的是，笔者的学生在研究完这个案例后，就开始研究如何在声音和视频中隐藏信息。虽然他们可能会遇到很多困难，但这一探究过程对他们学习信息技术有着非常重要的意义。

注：为了帮助更多的教师能体验这一项目，笔者借用上海人工智能实验室推出的OpenInnoLab平台，新建了一个名为“图片隐写术的研究”的项目。大家访问这一平台（https://www.openinnolab.org.cn/），即可看到所有的运行效果，如果觉得不错，还可以“克隆”出一个新的项目进行研究。*e*