**《信息安全概论》实验报告**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **姓名** | | 魏永森 | | **年级** | | 19 |
| **学号** | | 20192242 | | **专业、班级** | | 计卓卓越1班 |
| **实验名称** | **实验二 信息隐藏实验** | | | | | |
| **实验时间** | **2022.4.19** | | **实验地点** | | **DS3 402** | |
| **实验成绩** |  | | **实验性质** | | **□验证性 □设计性 □综合性** | |
| 教师评价：  □算法/实验过程正确； □源程序/实验内容提交 □程序结构/实验步骤合理；  □实验结果正确； □语法、语义正确； □报告规范；  评语：  评价教师签名（电子签名）： | | | | | | |
| 一、实验目的  学习并掌握图像信息隐藏的基本原理和方法  实现基于LSB的信息隐藏和提取算法 | | | | | | |
| 二、实验项目内容  使用LSB算法在图片中隐藏如下信息：CQUWATERMASKEXP  从被隐藏数据的图片中解析出如上信息，建议使用Matlab  在实际应用中，隐藏信息量通常是不可预知的，同时，攻击者也很容易从最低位像素提取到隐藏信息。另一方面，如何确保信息来源于正确的发送者？针对这些问题，请设计完整的保密通信方案。 | | | | | | |
| 三、实验设计  任何多媒体信息在数字化时都会产生物理随机噪声，而人的感官系统对这些随机噪声并不敏感，通过使用秘密信息比特替换随机噪声，从而实现信息隐藏。在图像中，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。  LSB算法选用最低位平面来嵌入信息，最低位平面对图像的视觉效果影响最轻微，因此在视觉上很难察觉。作为大数据量的信息隐藏方法，LSB在保密通信中仍占据相当重要的地位。  LSB主要步骤：  加密：  1. 读入图片  2. 准备待隐藏的信息，将其转换为二进制  3. 遍历图像，对像素的最低1bit置0，同时在该比特位写入1位二进制表示隐藏的信息  解密：  1. 预知隐藏信息量（等同于key）  2. 提取出像素的最低1bit，组合成连续bit数据，转换为ASCII码对比是否与隐藏信息一致 | | | | | | |
| 四、实验过程或算法  **1.** 首先编写将隐藏信息转化为二进制字符串并还原的代码  # 编码  def str2bin(str):      return " ".join([bin(ord(c)).replace('0b','') for c in str])  # 解码  def bin2str(bin):      return "".join([ chr((int(s,2))) for s in bin])  调用过程    **2.** 之后读入图像矩阵，这里以灰度图像读入  # 初始化处理  mat = cv2.imread(r"E:\Python\IS\ex2\xiugai .jpg",cv2.IMREAD\_GRAYSCALE)  # 显示  cv2.imshow("image",mat)  # 关闭  cv2.waitKey()   #要加的两行代码  cv2.destroyAllWindows()  运行后可以显示出图像窗口，按下任意键关闭窗口。  **3.** 对图像矩阵的最低位进行修改  代码的实现原理在注释中体现  # 函数LSB,   图像矩阵,隐藏信息字符串  def LSB(mat,str):      code = str2bin(str)     # 转换为二进制串      # 记录图像矩阵      row = len(mat)      col = len(mat[0])      # 二进制串长度      size = len(code)      # 写入像素最低位      pos = 0      for i, j in itertools.product(range(row), range(col)):          # 检查像素最低位,奇数-1，偶数不变          if (mat[i,j] & 1) == 1:              mat[i,j] -= 1          # 开始写入数据, 判断跳出,          if(pos >= size):    # 二进制串读完              break          if(code[pos] == ' '):   # 空格跳过              pos += 1          if(code[pos] == '1'):   # 1 +1, 0 不变              mat[i,j] += 1          pos+=1      return mat  # 返回新的像素矩阵  **4.** 对修改的后的图像进行解密  代码的功能在注释中体现  # LSB提取隐藏信息    图像矩阵，隐藏信息字符长度(len(str))  def anti\_LSB(mat,num):          # 解密      res = ""    # 二进制字符串，8位一组，每组最后一位是空格      cnt = 0     # 计数， 7循环      sum = 0     # 记录res中存在 字符数(7bit  1字符)      row = len(mat)      col = len(mat[1])      for i, j in itertools.product(range(row), range(col)):          # 开始读数据          res += '1' if (mat[i,j] & 1) == 1 else '0'          cnt += 1          # 跳出判断需要在7循环之前          if cnt==7:              res += " "  # 空格分割7位，也就是一个字符              cnt = 0     # 重新计数              sum += 1    # 记录字符数          if sum == num:              break      return res.strip()  # 返回二进制字符串（按 空格 分割可以形成list）  **5.** 进行测试  读入图像矩阵，调用LSB函数进行加密并显示出图像观察有无明显变化，最后再调用anti\_LSB进行解密。  # 需要隐藏的信息  str = "CQUWATERMASKEXP"  # 加密后的矩阵  mat\_encode = LSB(mat,str)  # 显示当前图像  cv2.imshow("modify",mat\_encode)  cv2.waitKey()  cv2.destroyAllWindows()  # 解密  res = anti\_LSB(mat\_encode,len(str)).strip()  print(bin2str(res.split(" ")))  **设计新的保密通信方案：**  假设发送方A，接收方B。A使用LSB算法隐藏信息，然后将图片和key发送给B。  将LSB的像素由顺序插入改为乱序插入，鉴别发送方使用给消息加校验值，然后让发送方对提取到的信息进行检验对比的方法。  其中乱序插入的实现可以使用Logistic混沌序列，实现公式如下，X(k+1) = u \* X(k) \* [1 - X(k)],（k=0,1,…,n）。也就是只要发送方提供初始值u和X(0)还有序列长度k的值，那么接收方就根据计算可以得到一个长度为k的序列L。  消息鉴别使用了校验值。将消息转换为二进制明文M，之后发送方A先使用私钥key(As)对明文M进行处理，然后使用为鉴别码生成函数F得到一个校验值J，然后将M和J拼接成为一个新的消息M(link)。然后将M(link)隐藏到图片中发送给接收方B。  B在接收图片和得到u，X(0)，k之后首先可以从图片中根据混沌序列L提取到一串二进制数值M(link)’。接着使用函数F对M(link)’中的M部分进行校验，得到一个新的校验码J’。之后对比从图像中提取到的M(link)’的J部分和计算得到的J’是否相等，如果不相等说明图像在传输过程中被修改，相等则说明图像来自于A并且未被修改。 | | | | | | |
| 五、实验过程中遇到的问题及解决情况(主要问题及解决情况)  **1. 读入图像后无法显示出(imshow)**  使用搜索引擎后发现问题所在，需要在imshow后面添加两行代码，cv2.waitKey()   #要加的两行代码  cv2.destroyAllWindows()  使窗口保持显示需要使用waitKey, 然后按下任意键关闭。 | | | | | | |
| 六、实验结果及分析和（或）源程序调试过程  首先读入灰度图像    加密之后    从肉眼上，看不出两个图片的明显区别。  解密的时候，需要输入 加密后的图像矩阵 和 隐藏信息字符串长度"CQUWATERMASKEXP" ，也就是15 | | | | | | |