**信息隐藏实验设计文档**

**1.1设计要求**

编写一个小程序，实现如下基本功能：

1. 将指定的文本信息隐藏到24位位图中，输出携密图片，比较原始图像和携密图像；
2. 从1产生的携密图像文件中恢复隐秘的文本信息；
3. 在完成1、2的基础上，先对待隐藏的文本信息加密，然后将密文隐藏到24位位图中；
4. 从3产生的携密图像文件中恢复出隐秘的文本信息。

**1.2系统功能描述**

本程序是在Visual Studio Code2023平台上使用c++编写生成的程序，可以读取24位图像文件的数据并拷贝生成新图像，将隐藏的文本经过简单的异或加密，加上结束标志后，用LSB方法隐藏入新位图中。同时，程序也能够从位图中提取可能隐藏的信息，该信息段以1个字节的0标志结尾。

**1.3系统设计**

为实现相应的功能，需要完成以下5个模块的设计：

1. 位图读取和数据提取模块

实现目标24位位图文件的提取，获取位图的宽度、高度、位图数据并存储。 该模块主要是通过readBmp()函数实现的，其输入参数是文件路径和要存储数据的vector容器，最终返回读取结果是否成功。

1. 位图文件生成和拷贝模块

实现创建一个新位图文件，并将原文件拷贝数据拷贝到该文件的功能。目的是生成信息隐藏后的新图片而不直接更改原图，便于比较。

该模块主要在主函数中实现，通过调用dst和src文件函数实现目标路径下文件的生成和拷贝。

1. 文本信息输入和处理模块

实现读取需要隐藏的文本信息，并将其通过异或加密函数进行加密的功能。

该模块主要通过加密函数encrypt()实现，输入参数是待处理的文本字符串。在函数中，逐个字符进行与0b11111111的异或操作，将有意思的字符串转化为无意思的乱码，达到加密目的。

1. 信息隐藏模块

将处理好的待隐藏文本逐位嵌入到位图数据的字节最低位中，并在末尾添加一字节的0标志信息结束，便于读取时识别信息末尾。

该模块主要是通过hideText()函数实现的，其输入参数是隐藏信息载体数据的vector容器和要隐藏的信息串，返回操作结果是否成功。该函数首先计算消息的长度，包括待添加的1字节标志，然后在消息末尾添加1个字节的0作为结束标志。接着判断嵌入空间是否足够，若有足够空间则依次取一个字节处理，将该字节逐位嵌入图像数据的每个字节的最低位中。

1. 信息提取模块

在携密图像文件的位图信息中提取字节最低位作为待恢复的序列，直到读取到一字节（8个0）为止，将读取到的二进制序列转换为字符并使用解密函数得到解码信息。

该模块主要是通过getText()函数实现的，其输入参数是待提取信息的vector容器和用于存储解密信息串的string对象，返回提取操作是否成功的结果。该函数首先将提取的二进制位转换为字节并解密，然后将解密后的信息输出。该函数的主要实现方式是循环遍历每个字节的最低位，并将连续的8个0作为结束标志。

**1.3详细设计**

核心代码流程图如下：

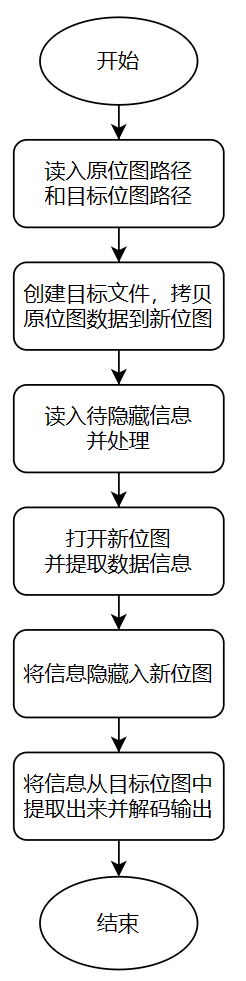


图1 核心代码流程图

**1.4使用说明**

根据终端打印的提示信息，依次输入原位图的路径、目标位图生成路径、待隐藏的文本信息，按下回车后程序自动运行实现对应功能。

**1.5结果分析**

测试前目录树如下图：

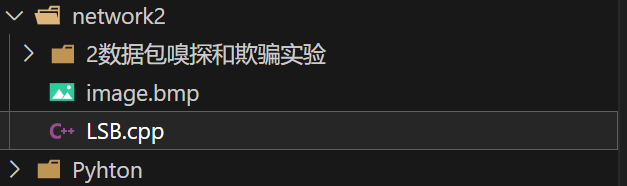


图2 测试前目录树截图

图中的image.bmp是待处理的位图，LSB.cpp为程序的源代码文件。

测试结果截图：

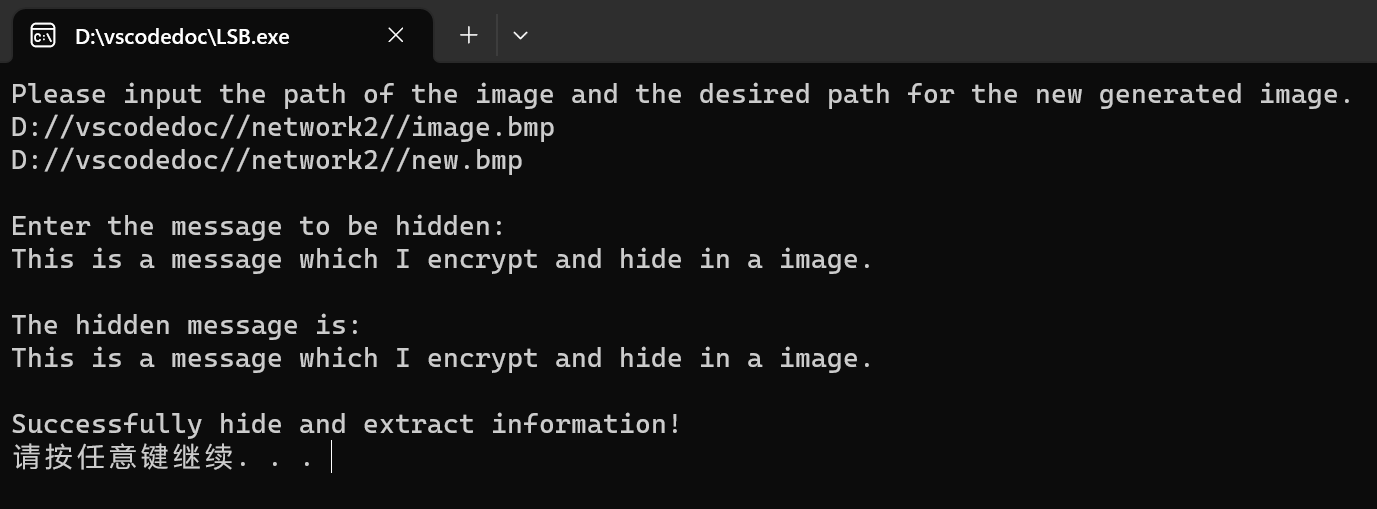


图3 测试结果截图1

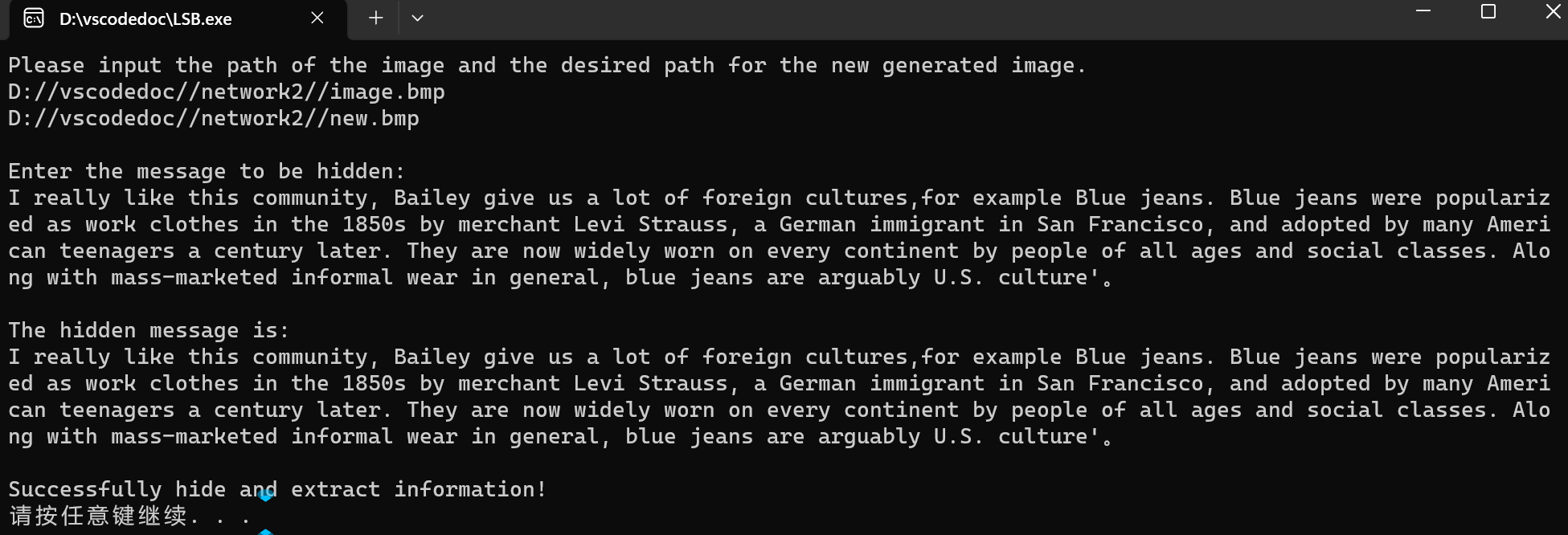


图4 测试结果截图2

可以看到程序正确输出了隐藏的文本信息，故可初步判断程序提取信息成功。

测试后目录树如下图：

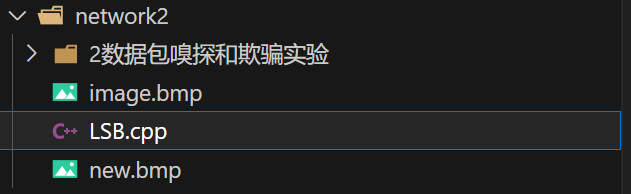


图5 测试后目录树截图

可以观察到程序运行之后目录中出现了原来不存在的new.bmp文件，说明文件生成拷贝操作成功。

信息隐藏前后图片对比：

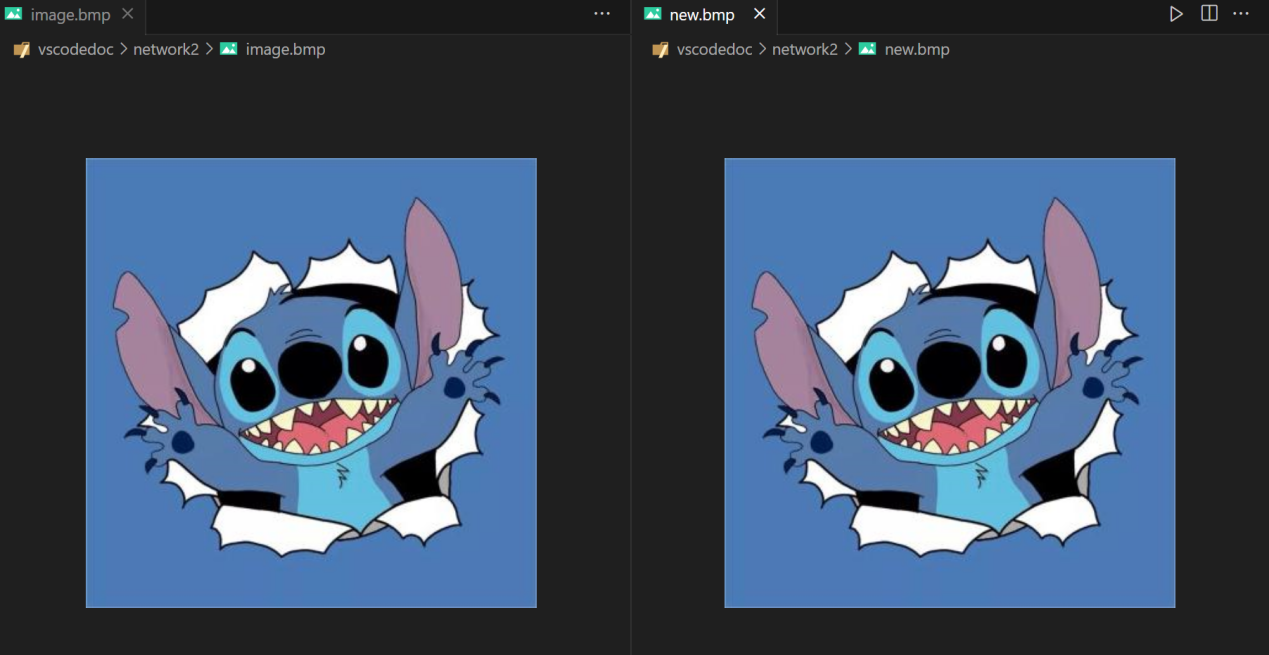


图6 初始位图和处理后的位图对比图

观察信息隐藏前后的不同图片，虽然两者的数据不一致，但是肉眼观察找不到差别，说明信息隐藏LSB方法的不可感知性。

同时，由于位图采用无压缩格式，且与设备无关，故在任何支持位图的设备上都能使用该程序进行信息隐藏和提取，且携密图像在不同设备传输后依然能通过正确的解密程序提取信息，彰显鲁棒性。

而对于本程序采用的异或加密方法，虽然较为简单，但可以很快速方便地将有意义的信息转换为无意义的乱码显示。且该程序可以很方便地更换加密方式，只需要修改加密解密函数。

而将文本信息隐藏到图片中的做法体现了技术交互性。

综上所述，该程序不仅能完成所有的功能要求，且有许多良好的特性。

**1.6程序源代码**

见附件LSB.cpp。