目录

一、 引言

1. 编写目的

随着信息技术的不断革新，传统印刷和记录媒介复制业中的版权保护问题遭受到了巨大的挑战。互联网因其匿名性，许多不法分子乘机在其上散播大量未经授权的版权图片视频作品，这样的现状极大地打击了媒体创作者的积极性。因而市场亟需一种软件，使得互联网上流传的图片视频能够溯源，从而追踪到非法散布者本人。Hollow 图片（视频）信息隐藏加解密软件（下文中一律简称为：本软件）正是迎合了这样的需求，希望能够帮助广大媒体创作者打击盗版，维护自己的合法权益。

2. 术语解释

整数小波变换(IWT): 是离散小波变换(DWT)的一种特殊实现方式。通常幅值为整数的数字信号经小波变换后仍能够得到整数变换结果，则称这种小波变换为 IWT。作为小波分析理论的拓展，IWT 不仅继承了 DWT 的许多优势，而且实现了很多 DWT 无法完成的功能。例如，具有真正意义的可逆性，能够实现图像的完全无损压缩；IWT 还具有计算复杂度很低，节省存储空间等优势。

3. 参考资料

本软件的图片加密解密算法参考了这篇论文：Hsiao CY., Tsai MF., Yang CY. (2017) High-capacity Robust Watermarking Approach for Protecting Ownership Right. In: Pan JS., Tsai PW., Huang HC. (eds) Advances in Intelligent Information Hiding and Multimedia Signal Processing. Smart Innovation, Systems and Technologies, vol 63. Springer, Cham.

二、 软件概述

1. 算法概述

本软件采用基于整数小波变换的图像信息隐藏加密算法，通过计算图像像素平方根值偏移量，实现了在宿主图像中嵌入多个数据位，从而能够隐藏图片信息。与同类算法相比，该算法提供了一个很大的隐藏存储空间，并且加密后的图片感知质量很好，与原图片画质相比几乎一致看不出区别。同时该算法还能够抵御一定程度的各种图片压缩以及剪切，添加噪声，旋转，改变亮度，边缘锐化，模糊等降低图片质量的攻击手段，具有高健壮性。

2. 功能

软件采用图形界面窗口交互式设计，主要分为三个功能模块：隐藏加密，隐藏解密，视频处理。并且由于本软件采用并发编程，本软件还支持用户基于自己电脑的性能设定同时运行的最大线程数，以获得最佳的使用体验。

本软件采用图 1 流程图所示的交互方式进行图片和视频的加密与解密：

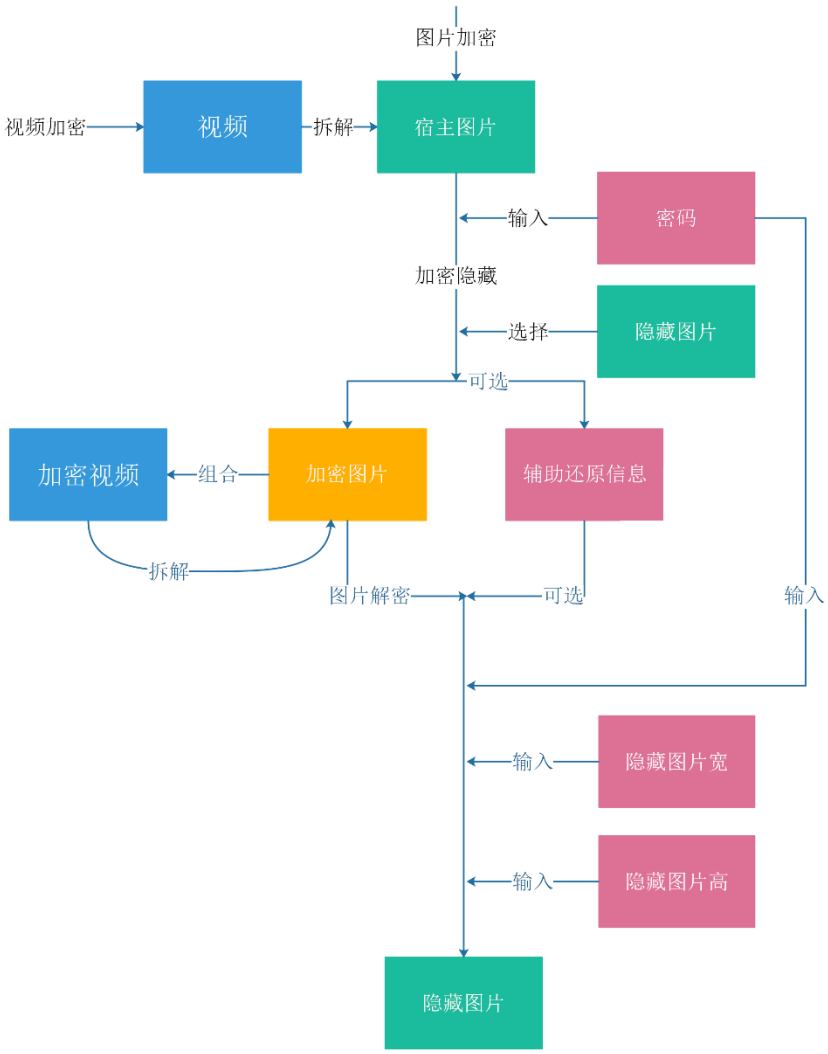

图 1 图片和视频加密与解密流程图

对于图片：

在加密时，用户需要选定宿主图片文件和隐藏图片文件，并设定一个加密密码。用户可以选择是否保存图片还原辅助信息文件到本地用于辅助解密。

如果需要批量加密隐藏图片，软件同时还支持选定存放批量宿主图片的文件夹，通过遍历文件夹下的所有文件，进行批量的宿主图片加密。加密后的图片文件会以 png 的编码格式，图像尺寸为宿主图像的尺寸保存在宿主图片存放的文件夹里，并且在宿主图片文件名（不包括扩展名）的基础上加上了‘ec’前缀以示区分。如果选择了保存图片还原辅助信息文件，则该信息文件将会以和加密图片同名（不包括扩展名），扩展名为.npy 的文件名进行保存。

在解密时，用户需要正确地输入被隐藏图片的宽与高，并且输入正确的密码，否则将会导致解密的失败，即解密出来的图片为一堆杂乱的像素点。如果有图片辅助还原信息，用户还可以选择文件的位置。如果是因为加密时未保存图片还原辅助信息或者是因为图片还原辅助信息丢失这类情况，用户可以选择不加载图片辅助还原信息文件，虽然这样做能识别出图片主体，但是解密出来的隐藏图片会有部分像素颜色显示异常。

为了响应用户批量解密需求，在加密图片所使用的被隐藏图片的宽与高和加密密码都相同的前提下，这里的加密图片选择和图片还原辅助信息文件选择也支持文件夹遍历。

当用户不存在图片还原辅助信息文件时，本软件的操作执行方式分为两种：

1. 选择单个加密图片进行解密。

2. 选择文件夹，通过遍历文件夹下的所有文件，进行批量的加密图片解密。

当用户拥有图片还原辅助信息文件时，本软件的操作执行方式分为四种：

1. 选择单个加密图片，并且选择了单个图片还原辅助信息文件，此时软件将对这两个文件进行对应解密。

2. 选择单个加密图片，并且选择了存放着图片还原辅助信息的文件夹，此时软件将会搜寻存放着图片还原辅助信息的文件夹下与加密图片同名的图片还原辅助信息文件（均不包括后缀名），进行一一对应解密。

3. 选择存放加密图片的文件夹，并且选择了存放着图片还原辅助信息的文件夹，此时软件将遍历存放加密图片的文件夹下的所有文件，并且搜寻存放着图片还原辅助信息的文件夹下与加密图片同名的图片还原辅助信息文件（均不包括后缀名），进行一一对应解密。

4. 选择存放加密图片的文件夹，并且选择了单个图片还原辅助信息文件，此时软件将遍历存放加密图片的文件夹下的所有文件，使用选择的图片还原辅助信息文件进行一一对应解密。（这种方式可能会造成图片解密的失败，应当谨慎使用）。

解密后得到的隐藏图片文件会以 png 的编码格式，图像尺寸为输入的尺寸保存在加密图片存放的同一文件夹里，并且在加密图片文件名（不包括扩展名）的基础上加上了‘de’前缀以示区分。

对于视频：

用户可以选择软件提供的视频处理功能。

在加密时，需要首先将视频按帧拆解为图片。用户需要选定视频文件，并且选定拆解出来的图片的存放文件夹。图片名将以纯数字，从 1 递增的顺序，png 的图片编码格式按顺序从视频中一帧一帧地被拆解出来。拆解完成后程序可以自动显示视频帧数，方便再次进行合成。随后，用户可以选定拆解出来的图片的存放文件夹进行批量图片加密。全部加密成功后，可以切换到视频合成选项，选定存放拆解出来的图片的文件夹和要存放视频的文件夹，进行拆解图片的视频合成。软件将遍历存放拆解出来的图片文件夹中以 ec 开头加上从 1 递增的数字为文件名，以 png 为扩展名的图片文件，然后使用 XviD 编码合成视频，视频尺寸按照图像尺寸大小生成的视频文件名为开始处理时的时间戳，扩展名为 avi。

在解密时，用户只需重复上述拆解步骤，将视频按帧拆解成图片，然后可以从中随机挑选一张或者更多的图片进行解密来进行视频的加密隐藏信息的获取。

3. 效果性能

因为本软件使用算法的优越性，所以使用本软件进行图片视频的加密，图片的隐藏效果极佳，即肉眼几乎无法区分哪个是加密图片，哪个是宿主图片，并且在极端干扰条件下，通过加密图片解密出来的隐藏图片也能识别出原隐藏图片的关键特征部分，效果良好，足以适应于市场上对于版权保护的需求。

同时，软件采用多线程并发编程以适应批量处理，具有较高的运行效率。

三、 运行环境

1. 硬件环境

本软件的编写与测试运行在作者本人的个人笔记本电脑上。由于条件限制，作者未测试

在其它硬件环境下的运行效果，理论上符合运行 Python 解释器硬件条件的，有 1G 以上的 RAM 和硬盘存储空间的所有 PC 机上都能运行本程序。

作者本人个人笔记本电脑配置：

CPU: Intel Core i7-8550U 1.80GHZ

RAM: 8GB

硬盘: SSD 128GB 机械 1TB

2. 软件环境

因为本程序为 Python 3 语言编写，因而凡是支持 Python 3 解释器的操作系统都可以编译本程序。作者的编译运行环境如下：

Windows 10 Version 1909（64 位）

Python 3.7.5（32 位）

同时，需要以下 Python 库：

numpy 1.17.3

opencv-python 4.1.1.26

pillow 6.2.1

pyQT5 5.13.1

在进行发布的时候，作者使用了 Pyinstaller 工具进行了打包，因为打包时使用的是 32 位的 Python，所以程序同时支持 32 位和 64 位的操作系统。因而本软件的打包版本运行环境如下：

Windows 7 以上

四、 技术细节

本程序一共使用了 10 个 Python 库，设计了 5 个独立函数和 5 个类，分别如下：

库：

1. PIL.Image

本软件用此库来进行图像的读取，尺寸大小的获取和图片的保存。

2. sys

用于访问和 Python 解释器联系紧密的变量和函数，是 UI 界面必需库。

3. os

本软件用此库来进行文件夹及文件是否存在的判断，和文件路径的处理获取。

4. time

本软件用此库来进行时间戳的获取，为视频合成时的视频时间戳文件名提供调用接口。

5. numpy

本软件用此库来进行图像数值化后数据的处理，因为该库底层采用 C 实现，而本软件图片处理部分涉及到大量的运算，因而用此库可以加快程序运行速度。

6. math

本软件用此库来实现图像处理过程中各种数值的运算。

7. random

本软件用此库来对经过处理的隐藏图片像素进行伪随机排序，因为设定随机数种子后每次伪随机排序结果相同，所以本软件将随机数种子值设定为密码，实现用密码解密还原图片的功能。

8. threading
为程序的多线程并发编程提供支持。
9. cv2
本软件用此库来进行视频的处理。
10. PyQt5
本软件的图形界面交互窗口框架。

函数：

1. output_file(picname, red, green, blue, width, height)

此函数对经过处理的图像的红绿蓝三通道合并, 并处理变换中的异常像素值，使其都在[0..255]范围内，然后使用 PIL.Image 库中的函数输出生成的图片文件。此函数参数: picname:输出照片存放文件地址, red,green,blue:分别为经过变换处理的红绿蓝三通道的 numpy 数组。width, height:输出图片的宽和高。

2. dwt(colour, width, height)

此函数进行离散小波变换处理。此函数参数: colour:要变换图片的一个通道的 numpy 数组。width, height:要变换图片的宽和高。此函数返回将 colour 经过离散小波变换后得到的 numpy 数组。

3. extract_watermark(TEMP2, width, height, mwidth, mheight, complex, passwd)

此函数对图片的隐藏信息进行提取。此函数参数: TEMP2:已经经过离散小波变换，需要进行提取隐藏信息的加密图片的一个通道的 numpy 数组。width, height:加密图片的宽和高。mwidth, mheigh:输入的隐藏图片宽和高。complex:该加密图片通道对应的图片还原辅助信息 numpy 数组。passwd:输入的密码。此函数返回经过最终经过解密生成的隐藏图片的对应通道的 numpy 数组。

4. hiding_data(TEMP2, filename, flag, writedata, passwd)

此函数向图片写入隐藏信息。此函数参数: TEMP2:已经经过离散小波变换，需要进行写入隐藏信息的宿主图片的一个通道的 numpy 数组。filename:要隐藏图片的文件地址。flag:图像的通道标识，红绿蓝分别为 0,1,2。writedata:用于标识是否向图片写入隐藏信息。passwd:输入的密码。此函数返回该加密图片对应通道的图片还原辅助信息 numpy 数组。

5. idwt(TEMP2, pixel, width, height)

此函数对图片进行离散小波反变换。此函数参数: TEMP2:已经写入隐藏信息的宿主图片一个通道的 numpy 数组。pixel:联系宿主图片各通道信息的 numpy 数组。width, height:宿主图片的宽和高。该函数返回最终生成的加密图片的对应通道 numpy 数组。

类：

1. MyThread(threading.Thread)

继承并重写线程类，使得线程运行能获取函数返回值。拥有 2 个方法，分别为构造方法和获取返回值的方法。

2. AuInfo(object)

作者信息窗口类，拥有 4 个方法，实现作者信息对话框交互显示的全部功能。

3. class helpw(object)

软件帮助使用说明窗口类,拥有 2 个方法，实现软件帮助使用说明对话框交互显示的全部功能。

4. ThrNum(object)

设定线程最大值窗口类，拥有 3 个方法，实现设定线程最大值对话框交互显示的全部功

能。

5. Ui_MainWindow(QtCore.QObject)

程序主界面窗口类，继承了 PyQT5.QtCore.Qobject 类，拥有 25 个方法，实现主应用程序的交互显示的全部功能。

下面是对程序实现细节的说明：

1. 算法步骤

为了使得算法具有高健壮性，本软件只将数据位嵌入整数小波变换第 1 级的高高（HH）子带域。即在位嵌入之前，输入图像就会按照 $d_{j,k}=s_{j-1,2k+1}-s_{j-1,2k}$ 和 $s_{j,k}=s_{j-1,2k}+[d_{j,k}/2]$ 这两个公式被分解。其中，$s_{j,k}$ 和 $d_{j,k}$ 分别代表在第 j 级的第 k 个低频和高频小波系数.“[ ]”符号代表向下取整运算。下面是该算法的详细描述：

1. 位嵌入（信息隐藏加密）

根据通常情况，令 $W_j=\{(w_{rj},w_{gj},w_{bj})\}_{j=0}^{ab-1}$ 为输入的 a*b 大小的经过扰乱的隐藏图片第 j 个像素值。同样的令 C=$\{(c_{rj},c_{gj},c_{bj})\}_{j=0}^{ab-1}$，其中 $c_{rj}=\sqrt{w_{rj}}-round\left(\sqrt{w_{rj}}\right)$，$c_{gj}=\sqrt{w_{gj}}-round\left(\sqrt{w_{gj}}\right)$，$c_{bj}=\sqrt{w_{bj}}-round\left(\sqrt{w_{bj}}\right)$，这三个参数是 W_j 的偏差值。具体算法如下所示：

输入：一个彩色三通道宿主图片 $S=\{f(r_i;g_i;b_i)|i=1,2,...MN\}$ 和一个使用加密密码算法经过扰乱的隐藏图片。

输出：一个加密图片 $\hat{S}$ 和一个图片还原辅助信息集合 C。

步骤 1：对宿主图片进行第 1 级整数小波变换来获得 IWT 系数高高子带域 H=$\{(h_{rj},h_{gj},h_{bj})\}_{j=0}^{(MN/4)-1}$。（**执行 dwt 函数**）

下述步骤 2-10 为 hiding_data 函数中所实现的算法步骤内容。

步骤 2：取 H 的三个通道分别对应的第 j 个像素值集合 H_j，下面步骤 2-9 所有操作都在 H_j 上执行。

步骤 3：如果参数 $h_{rj}>1$ 给变量 s_{rj} 赋值为 1，如果参数 $h_{gj}>1$ 给变量 s_{gj} 赋值为 1，如果参数 $h_{bj}>1$ 给变量 s_{bj} 赋值为 1.反之则分别把这三个变量值赋为-1，并且将三个参数值取绝对值。

步骤 4：将 $\mathbf{w}_{rj}$, $\mathbf{w}_{gj}$, $\mathbf{w}_{bj}$,的平方根值分别进行四舍五入运算，赋给变量 d_{rj}, d_{gj}, d_{bj}。此外，计算 $\mathbf{w}_{rj}$, $\mathbf{w}_{gj}$, $\mathbf{w}_{bj}$,的平方根值分别减去 d_{rj}, d_{gj}, d_{bj} 所分别得到的值 c_{rj}, c_{gj}, c_{bj}，将 c_{rj}, c_{gj}, c_{bj} 集合保存为图片还原辅助信息 C_j。

步骤 5：计算下列公式并赋值：$h_{rj}=[\,h_{rj}-(h_{rj}\bmod 10)]/10$, $h_{gj}=[\,h_{gj}-(h_{gj}\bmod 10)]/10$, $h_{bj}=[\,h_{bj}-(h_{bj}\bmod 10)]/10$。

步骤 6：设置通道标志参数 $\alpha=1$。

步骤 7：如果 $\alpha=1$ 令 $h_k=h_{rj}, d_k=d_{rj}, s_k=s_{rj}$，如果 $\alpha=2$ 令 $h_k=h_{gj}, d_k=d_{gj}, s_k=s_{gj}$，如果 $\alpha=3$ 令 $h_k=h_{bj}, d_k=d_{bj}, s_k=s_{bj}$。如果 $\alpha>3$ 则标志对第 j 个像素值集合数据处理完成，直接跳转到步骤 10。

步骤 8：如果 $d_k>=10$ 时，执行以下语句：如果$(h_k \bmod 2)=1$，则计算并赋值 $h_k=[(h_k-1)*10+(d_k/10)+d_k \bmod 10]*s_k$ 否则计算并赋值 $h_k=[h_k*10+(d_k/10)+d_k \bmod 10]*s_k$，然后使 α 值递增 1，重复步骤 8。

步骤 9：如果 d_k< 10 时时，执行以下语句：如果($h_k$mod2)=1，则计算并赋值 h_k=[(h_k *10+d_k]* s_k 否则计算并赋值 h_k=[(h_k+1) *10+d_k]* s_k, 然后使 α 值递增 1，跳转到步骤 7。

步骤 10：跳转到步骤 2，继续对其它像素值集合进行处理，直到所有图片像素数据处理完成。

步骤 11：当对 H 执行反 IWT 运算，得到加密图片$\hat{S}$。（**执行 idwt 函数**）

2. 数据提取（信息隐藏解密）

输入：一个经过加密的彩色三通道图片 $S = \{f(r_i; g_i; b_i)|i=1,2,...MN\}$和一个图片还原辅助信息集合 C。

输出：一个提取出来的隐藏图片 W。

步骤 1：对宿主图片进行第 1 级整数小波变换来获得 IWT 系数高高子带域 H= $\{(h_{rj}, h_{gj}, h_{bj})\}_{j=0}^{(MN/4)-1}$。（**执行 dwt 函数**）

下列步骤 2-9 为 extract_watermark 函数中所实现的算法步骤内容。

步骤 2：取 H 的三个通道分别对应的第 j 个像素值集合 H_j，下面步骤 2-9 所有操作都在 H_j 上执行。

步骤 3：如果参数 h_{rj}>1 给变量 s_{rj} 赋值为 1，如果参数 h_{gj}>1 给变量 s_{gj} 赋值为 1，如果参数 h_{bj}>1 给变量 s_{bj} 赋值为 1.反之则分别把这三个变量值赋为-1，并且将三个参数值取绝对值。

步骤 4：计算下列公式并赋值：$w_{rj}= h_{rj}$ mod10, $w_{gj}= h_{gj}$ mod10, $w_{bj}= h_{bj}$ mod10。

步骤 5：设置通道标志参数 β =1。

步骤 6：如果 β = 1 令 $h_k= h_{rj}, w_k= w_{rj}, c_k= c_{rj}$, 如果 β= 2 令 $h_k= h_{gj}, w_k= w_{gj}, c_k= c_{gj}$, 如果 β= 3 令 $h_k= h_{bj}, w_k= w_{bj}, c_k= c_{bj}$。如果 β > 3 则标志对第 j 个像素值集合数据处理完成，直接跳转到步骤 2，继续对其它像素值集合进行处理。

步骤 7：计算 T= (h_k- w_k)/10，如果 Tmod2 不等于 1，则令 w_k=w_k+9。

步骤 8：分别计算 w_k=(w_k+ c_k)^2, h_k= h_k* s_k，跳转到步骤 6 直到所有图片像素数据处理完成。

步骤 9：最终得到提取出来的隐藏图片 W。

2. 异常过滤

为了防止异常数据的进入造成应用程序的崩溃，本软件使用了输入过滤和异常处理来提示用户输入正确的数据。这些异常过滤位于 4 个窗口类中，当发生了异常问题，就使用 pyQT5 中的 QMessageBox 组件进行弹窗提示。这些可能出现问题的地方如下：

ThrNum(object)：

1. 设定线程最大值函数中，用户可能会输入负数，这里本软件用一个判断进行弹窗警告。如果是输入的特殊符号，则进行异常捕捉提示。

Ui_MainWindow(QtCore.QObject)：

1. 用户可能会输入不存在的文件或者文件夹，或者在该输入文件的地方输入了文件夹，在该输入文件夹的地方输入了文件。这时可以采用 os.path.exists 函数判断文件或文件夹是否存在，用 os.path.isfile 函数判断是文件还是文件夹，分别根据逻辑进行弹窗提醒。
2. 对于要求用户输入图片宽和高，或者要求输入帧率的地方，同 ThrNum(object)中类似对设定线程最大值函数的异常过滤方法。
3. 尽管在用户点击“选择”按钮弹出的文件选择框里已经做了文件扩展名的过滤，但是

还是可能会发生图片，视频，图片辅助还原信息损坏或者恶意选择输入的情况，因而本软件在调用相关加密解密函数的时候做了异常处理，提示用户异常，避免软件崩溃。

3. 用户交互界面设计

本程序使用 PyQT5 框架进行 UI 的设计，使用了 TabWidget 进行分标签显示，menubar 构造菜单，LineEdit 进行行文本编辑，pushButton 按钮，CheckBox 来选择是否有和是否保存图片还原辅助信息，RadioButton 来实现视频拆解和合成功能的切换，progressbar 实现处理进度显示。

同时，为了实现线程对 UI 界面的控制，使用了 QtCore.pyqtSignal，用 PyQT 中的信号与槽功能，emit 方法进行消息的传递，避免 PyQT5 禁止的直接调用类中的方法，产生警告，最终导致崩溃。

另外，使用了 pyQT5 中的 QMessageBox 组件进行弹窗提示。同时使用 QtGui.QDesktopServices.openUrl 函数进行点击按钮后打开网页的跳转。同时使用 QFileDialog 来进行文件（夹）选取对话框的设计。

具体用户交互界面详情请见文档末尾“用户使用指南”。

4. 多线程编程设计

本程序采用并发编程方法，在 MyThread(threading.Thread)类中继承并重写线程类，使得线程运行能获取函数返回值。方便有返回值的 dwt 等函数并发运行。

当用户点击主窗口的开始按钮后，本软件会自动创建一个独立线程运行相应加密解密拆解合成视频进程，从而避免 UI 界面卡死无响应。

对于在 Ui_MainWindow 类中的加密解密图片方法 encrypt、decrypt，本软件在需要对红绿蓝三通道分别独立处理的 dwt 函数和 extract_watermark，hiding_data 函数的调用使用了多线程并发运行，从而加快了速度。同时在 encrypt、decrypt 中需要输入输出文件的部分都加了线程锁，防止发生冲突。

本软件还使用了 threading.BoundedSemaphore 方法设定线程运行最大值，并可以和用户进行交互，让用户进行设置，确保计算机发挥出最大的性能的同时不至于死机。

5. 视频拆解合成功能的实现

视频拆解合成功能分别位于 Ui_MainWindow 类中的 video2pic 和 pic2video 函数中，使用 OpenCV 实现。因为 Python-OpenCV 库不支持中文路径名称，所以为了实现在中文路径下视频的读取写入，本软件用 cv2.imencode 和 cv2.imencode().tofile 方法来替代原本的 cv2.videoreader 和 cv2.videowriter 方法。

五、 图片信息隐藏加密解密效果验证

1. 正常加密解密

选取如图 2 宿主图片，图 3 隐藏图片，如图所示，使用本软件进行隐藏加密。

其中图 2 宿主图片为三通道 RGB 彩色图片，使用 jpg 格式进行编码，图片宽 5418，高 3048，大小为 8.40MB。

图 3 隐藏图片为三通道 RGB 彩色图片，使用 jpg 格式进行编码，图片宽 1920，高 1080，大小为 203KB。

图 2 原宿主图片

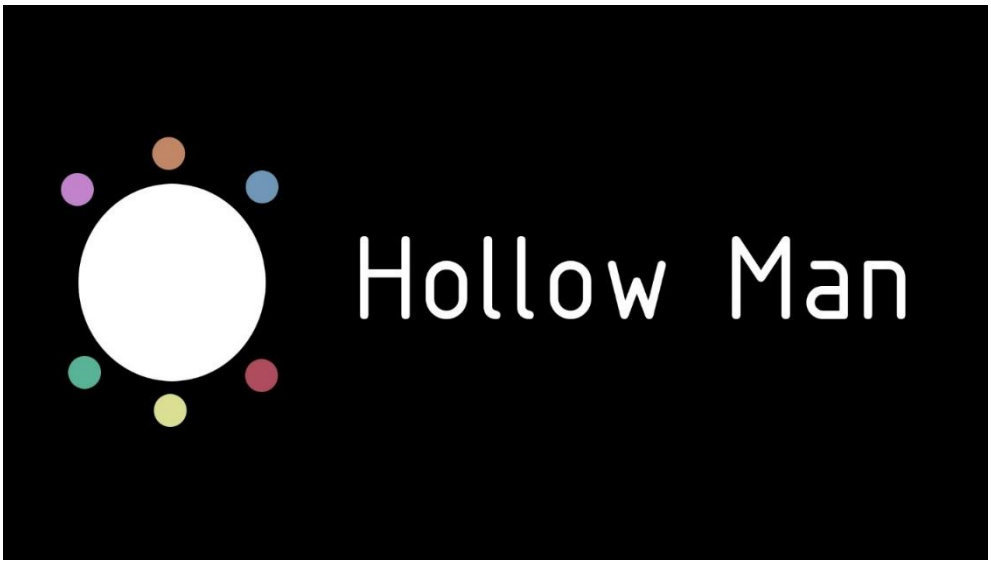

图 3 隐藏图片

得到如图 4 的加密图片。

图 4 加密图片

可以看到，加密后的图片和原图片几乎无区别。

对图片进行带图片还原辅助信息解密，得到图 5 所示解密图片。可以看到解密出来的图片也与原隐藏图片几乎无区别。

对图片进行无图片还原辅助信息解密，得到图 6 所示解密图片。可以看到图片一些像素点颜色出现了明显的不符合原图的情况，但是仍然可以识别，证明无图片还原辅助信息解密依然可行。

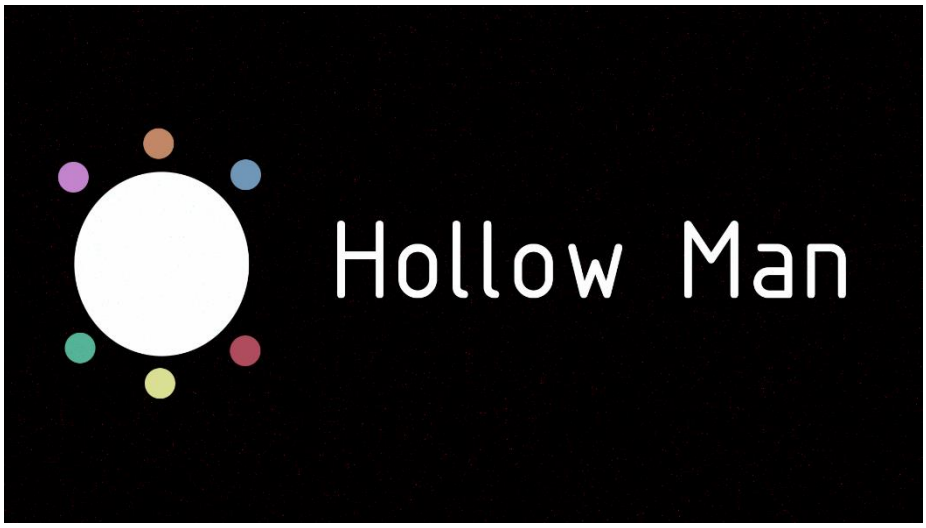

图 5 解密效果

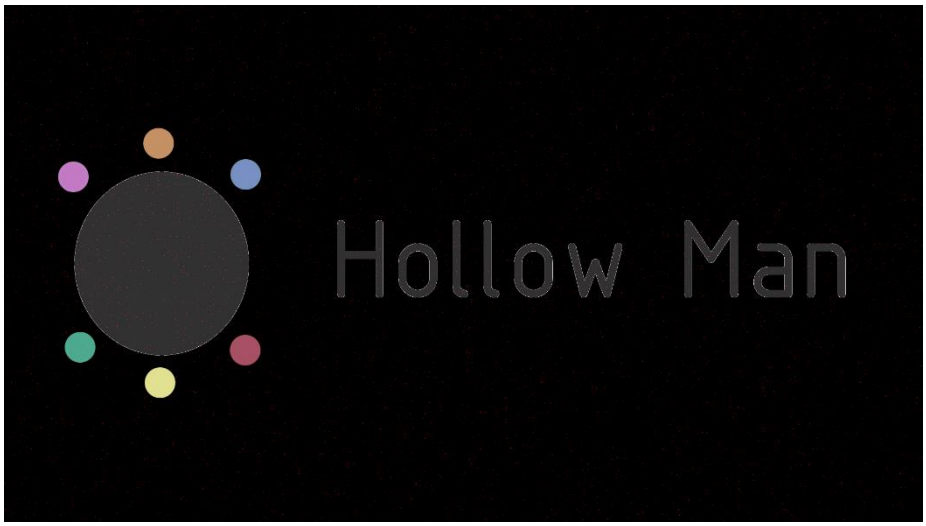

图 6 无图片还原辅助信息还原效果

2. 解密信息输入错误

当解密时输入错误的密码，如图 7，生成的图片为一堆杂乱的像素点，从中无法获取到任何有效信息。

类似的，当输错隐藏图片的宽和高时，也会出现这种情况。

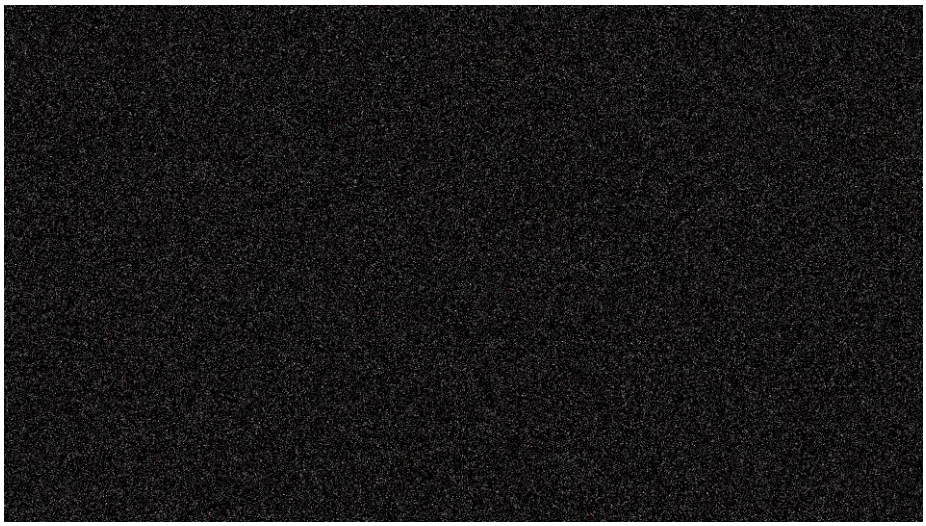

图 7 输错密码效果

3. 加密健壮性验证

将加密图片裁去一大半，如图 8 所示，可以看到如图 9 的解密效果，图片仍然可以识别，证明了本软件进行图片加密具有高健壮性的特点。

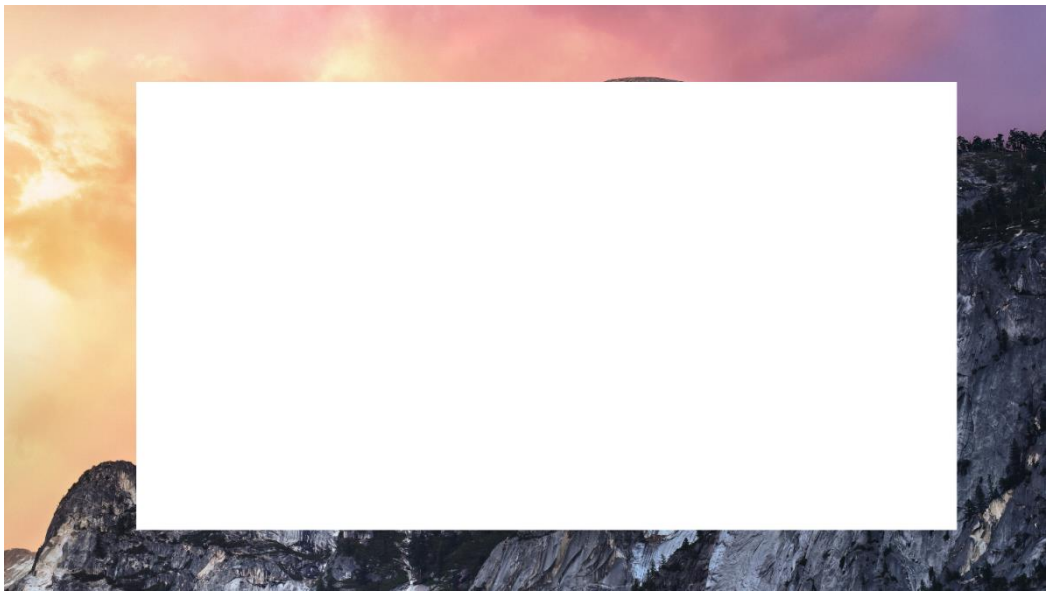

图 8 裁剪加密图片

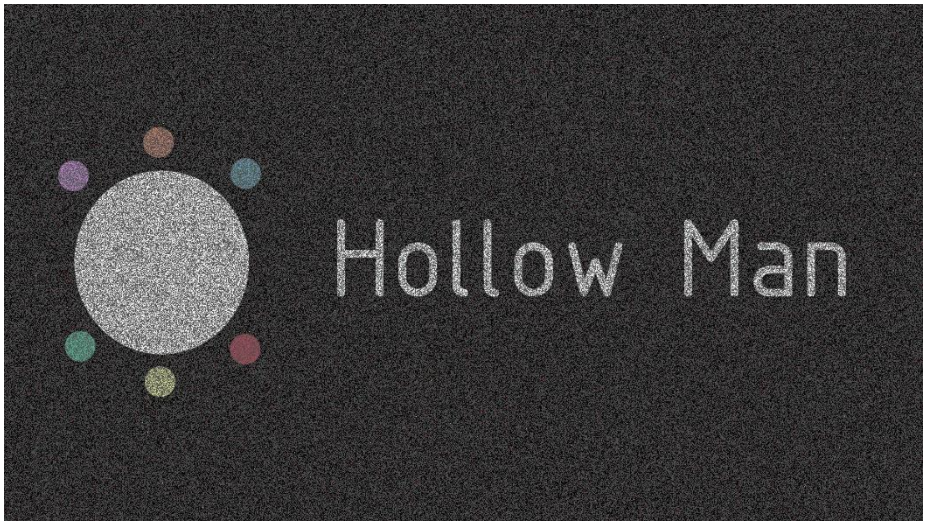

图 9 裁剪加密图片还原出的信息效果

注意：要隐藏图片尺寸越小，要附加隐藏信息的目标图片尺寸越大，效果越好。在给视频进行隐藏信息处理时，由于视频编码的压缩，效果可能并不是很明显。在加密时请不要使要隐藏图片的宽和高超过要附加隐藏信息的图片宽和高的二分之一。如果你这样做了，虽然不会有任何错误提示，但这会导致在隐藏解密时图片还原的必然失败！

六、 用户交互使用指南

1. 隐藏加密

运行本软件发布版本，初始默认界面如图 10：

图 10 隐藏加密初始界面

按照 UI 界面的提示，输入文件（夹）路径。你也可以点击按钮打开文件（夹）选择对话框，如果是选择文件，你可以看到图 11 中右下角已经自动做了相应文件过滤处理，在这种情况下你只能看到图片文件。

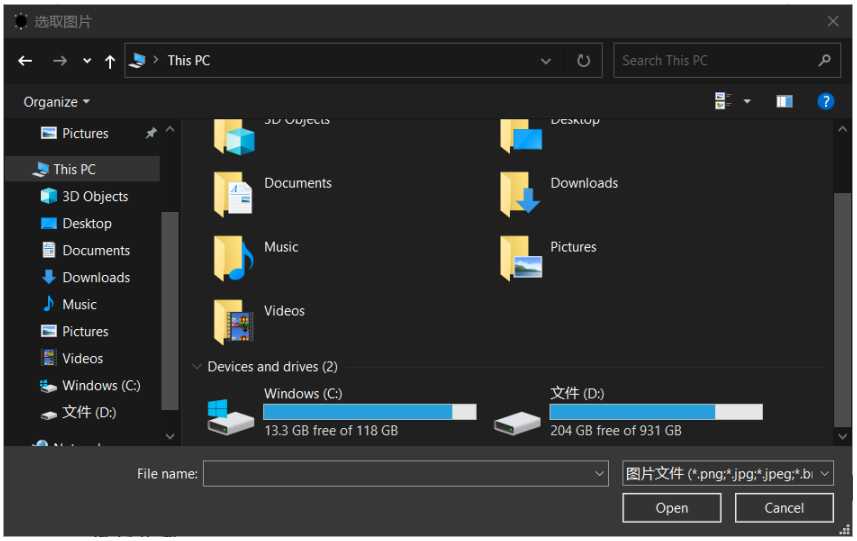

图 11 文件选择对话框

如果你不小心输入了不存在的文件，或者忘记输入某项，或者是其它错误等，都会有类似图 12 的提示框提醒你：

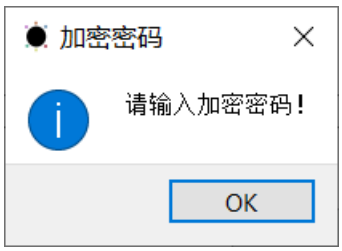

图 12 问题提示框

你可以通过点击“保存图片还原辅助信息”前面的框来做出是否保存图片还原辅助信息选择。

当你选择好了并且输入了一切必要的信息，没有任何错误输入，点击“开始加密”，软件

便会开始进行图片的加密处理，如图 13 所示：

图 13 隐藏加密运行界面

此时，程序是禁止你进行其它操作比如隐藏解密，视频处理等操作的，如果你切换到其它标签会发现类似于图 14 这样所有文本框和按钮都变成了灰色：

图 14 隐藏加密运行时其它标签界面

如果你选择批量处理图片，并且在处理过程中感觉到运行速度太慢或者电脑反应性能下降到无法容忍的程度时，可以此时在菜单栏中选择“选项”-“设定线程数”。如图 15，16 所示。

图 15 菜单选项

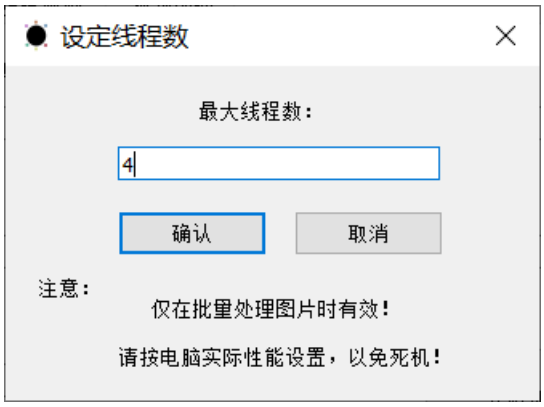

图 16 设定线程数

每次设定线程数对话框打开都会显示当前所设定的线程数，你可以根据需要进行调整，按确定应用设置。

加密完成时，进度条变为 100%，“开始加密！”按钮恢复可用状态，你可以继续执行其他操作了！

2. 隐藏解密

隐藏解密类似于隐藏加密的操作方法，如图 17，18 所示。图 19 所示的是无图片还原辅助信息时隐藏解密完成界面，可以看到，当你没有图片还原辅助信息时，选择图片还原辅助信息的相关按钮和文本框都会变为灰色不可用状态。

图 17 隐藏解密初始界面

图 18 隐藏解密运行界面

图 19 隐藏解密完成界面（无图片还原辅助信息时）

这里要注意，在解密时，你需要选择被附加隐藏信息的图片，如果存在图片还原辅助信息，你可以选择图片还原辅助信息文件或者文件夹，如果你选择的是存放图片还原辅助信息文件的文件夹，程序将会自动搜寻该文件夹下和你选择的被附加隐藏信息图片同名的文件，如果不存在则不能（在批量状态下）加载图片还原辅助信息。同时，你需要正确地输入被隐藏图片的宽与高，并且输入正确的解密密码。如果你输入的信息与加密时的不符，将会导致解密的失败，即解密出来的图片为一堆杂乱的像素点。如果需要批量解密图片，请在“被附加隐藏信息的图片（文件夹）”处选择要处理图片所存放的文件夹。

3. 视频处理

如果你想要进行视频拆解，首先按照 UI 程序的提示输入所有相关信息，此时的视频帧率是不用填的。程序界面如图 20 所示，程序会将视频按帧拆解成图片，视频拆解出来的图片默认保存为 png 格式,同时自动显示视频帧率 fps，方便再次合成。

图 20 视频拆解界面

完成后，你可以对保存的文件夹下的图片进行信息隐藏加密（注意不要改动拆解的原始图片名称，并且在后续加密和合成视频过程中再次使用此文件夹，方便合成视频）。

所有图片加密完成后，点击“视频合成”前面的圆圈，你可以此时需要输入原视频的视频帧率（如果软件在拆解视频时已经帮你填好则可以不变），选择图片存放的文件夹（注意不要改动图片加密后的名称）重新合成原视频。默认按照图片的尺寸大小确定视频尺寸大小，保存为 avi 格式，XviD 编码。

图 21 视频合成界面

对于视频的隐藏还原，同理，将视频按帧拆解成图片，然后可以从中随机挑选一张或者更多的图片解密来进行视频的加密隐藏信息的获取。

4. 其它功能

1. 将鼠标焦点放在程序上，你可以看到已经经过处理的文件数。

图 22 已处理文件提示

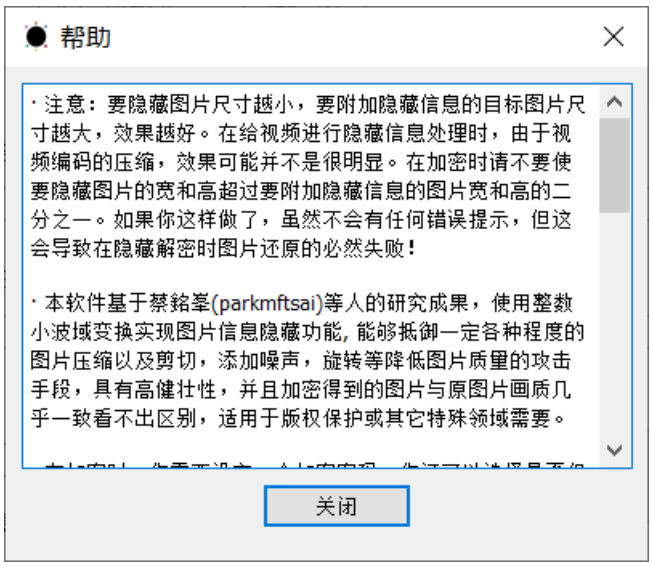

图 23 帮助

2. 点击菜单栏中“选项”-“关于”-“帮助”，你可以查看程序使用帮助。

3. 点击菜单栏中“选项”-“关于”-“作者”，你可以查看程序作者的相关信息。