NEQR 量子图像下的差分扩展可逆数据隐藏算法*

项世军[1,2], 李 豪[1,2], 宋婷婷[1]

[1] 暨南大学 信息科学技术学院/网络空间安全学院 广州 中国 510632
[2] 中国科学院信息工程研究所信息安全国家重点实验室 北京 中国 100093

摘要 量子图像安全处理是一个新兴的研究领域,而量子图像数据隐藏是量子图像安全处理技术的一种, 在不损害载体的情况下可用于保护量子图像的版权和认证量子图像是否完整。目前尚缺乏对量子图像可逆数据隐藏的详细技术研究。结合差值扩展技术, 本文提出了一种量子图像可逆数据隐藏算法: 1)选用 NEQR 量子图像表示法来表示图像; 2)借鉴经典的差值扩展算法, 在 NEQR 量子图像上对量子比特进行处理, 可逆嵌入数据; 3)设计了信息嵌入、信息提取和载体无损恢复的量子线路图, 并进行了仿真。基于经典图像的实验结果表明, 本文算法是可逆的, 可用于将来对量子图像的认证和保护。

关键词 量子图像; NEQR; 差值扩展; 可逆; 数据隐藏
中图法分类号 TP37 **DOI 号** 10.19363/J.cnki.cn10-1380/tn.2018.11.07

Reversible data hiding algorithm in NEQR quantum images

XIANG Shijun[1,2], LI Hao[1,2], SONG Tingting[1]

[1]School of Information Science and Technology/School of Cyber Security, Jinan University, Guangzhou 510632, China
[2] State Key Laboratory of Information Security ,Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

Abstract Quantum image secure processing is a new research field and reversible data hiding in quantum image is an import issue in quantum image secure processing technologies, which can be used to protect copyright and integrity of quantum images. In literature, there is lack of reversible data hiding for quantum images. In this paper we propose a reversible data hiding algorithm by difference expansion. Firstly, NEQR is applied to store cover quantum image. Furthermore, the information is reversibly embedded into the cover quantum image by using classical deference expansion strategy. Quantum circuits of information embedding, information extraction and original quantum image's restoration are given in details. Simulation testing in classical way has shown that the proposed method is reversible, and can be used as a potential solution for protection of quantum images.

Key words quantum image; NEQR; difference expansion; reversibility; data hiding

1 引言

可逆数据隐藏(也称可逆水印或无损信息隐藏)技术是一种利用数字载体的冗余, 将秘密信息嵌入到数字载体当中, 并在接收端能够正确提取隐藏信息及百分之百恢复出原始载体的技术[1]。该技术多用于数字媒体的版权保护、完整性认证、篡改定位及恢复等, 并广泛应用于对保密性、安全性以及保真度要求较高的数字媒体, 如军事及医学图像、电子发票、法律文书图片等[2]。因此, 近年来, 可逆数据隐藏技术已引起了广泛关注, 研究者已提出了很多有效的可逆数据隐藏算法, 主要可以分为以下四大类: 无损压缩[3]、差值扩展[4]、直方图平移[5]和预测误差扩展[6]。其中, Tian 提出了基于差值扩展的图像可逆数据隐藏算法非常经典, 以相邻的两个像素为一组, 通过扩展它们之间的差值可以达到将近 0.5 bpp 的嵌入容量。

由 Moore 定律[7]可知, 电子计算机芯片的制造水平是有极限的, 当超过极限后 Moore 定律失效, 电子原器件之间的功能不再是经典物理学规律所能解释,

通讯作者: 项世军, 博士, 教授, Email: Shijun_Xiang@qq.com
本课题得到国家自然科学基金（No.61272414，No.61772234）和信息安全国家重点实验室开放课题基金(No.2016-MS-07) 资助。
收稿日期：2018-09-06；修改日期：2018-09-21；定稿日期：2018-09-28

而是由量子效应占据主导地位。另外, 理论上量子计算机在计算速度方面也要远超经典计算机[8]。尽管短期内实现量子计算机的可能性不大, 但随着量子计算机的研究不断深入, 关于量子计算机上的信息安全问题开始引起了人们的重视, 主要集中在量子密码[9]和量子信息隐藏[10-11]两个研究方向。

不同于量子密码注重于量子秘钥的安全发布, 量子信息隐藏的关注点在于如何将信息隐藏在量子载体[11]中来达到隐蔽通信和保护量子图像的目的, 前者称为量子隐写[12-13], 后者简称量子水印[14-20]。目前量子信息隐藏研究主要以量子图像作为载体, 在研究量子图像水印之前, 要先确定使用哪种量子图像表示方法来制备和表示量子图像。常见的量子图像表示法有 Entangled Image[21]、Real Ket[22]、FRQI[23]和 NEQR[24], 目前现有的量子图像水印算法大部分都是建立在 FRQI 或 NEQR 量子图像表示法上。在 FRQI 量子图像表示下, 文献[14]利用傅里叶变换提出了一种量子图像水印方案, 文献[15]提出的量子图像水印方案则是利用了小波变换。这两种算法都不能使载体图像得到恢复, 因此是不可逆的。在 NEQR 量子图像表示方法下, 文献[16]利用 LSB 最低有效位提出了量子图像水印方案, 文献[18]提出的量子图像水印方案不仅利用了 LSB 最低有效位, 还利用了 MSB 最高有效位。这两种算法在提取出水印信息后都不能恢复载体图像。文献[18]是利用摩尔条纹设计量子水印算法, 在提取水印时需要提供原载体图像, 因此该算法是不可逆的。文献[19-20]提出的水印方案则是先把水印信息置乱后嵌入进载体图像, 提取水印信息时需要用到原来的载体图像, 同样的这两种算法也是不可逆的。

针对这个现状, 本文提出了一种在 NEQR 量子图像表示方法下的量子灰度图像可逆数据隐藏算法。首先, 选择与经典灰度图像最为相似的 NEQR 图像表示法来表征量子图像, 达到方便数据隐藏的目的; 其次, 通过参考差值扩展可逆数据隐藏技术, 实现了在 NEQR 量子图像上的信息嵌入、提取和载体图像的无损恢复。从信息的嵌入, 到信息的提取和载体图像的恢复均是在量子图像上进行操作。文中给出了数据隐藏算法中每一步的量子线路图。最后, 通过经典计算机仿真了数据隐藏时的嵌入失真和嵌入容量等, 并对结果进行了分析。

本文算法的研究动机是在量子图像上实现可逆数据隐藏以达到保护量子图像安全的目的, 是经典可逆数据隐藏算法[4]在 NEQR 量子图像上的一个有益拓展, 与经典差值扩展算法有以下几个主要不同: 1)本文所提出的量子图像差分扩展算法的复杂度要小于经典图像差分扩展算法, 也就是说, 即便同样的算法在量子图像上运行的速度也要比在经典图像上运行的速度要快, 这得益于量子图像的表达和并行计算方式; 2)在经典的信息隐藏框架中, 含有信息的图像在从发送端到接收端的传输过程中存在被阻击的危险。而量子图像信息隐藏则要安全的多, 因为其传输过程采用的是量子通信的方式, 而量子通信相对于经典通信在相同的条件下要安全的多; 3)可逆数据隐藏算法在经典图像上相对容易实现, 在量子图像上使用时需要考虑量子图像的表达方式和量子的特性, 需要仔细设计每一步的量子线路图, 也就是说即便同样的算法在量子图像上实现与在经典图像上实现也是完全不一样的。因此, 本文所提出的算法可作为一个潜在的安全技术用于保护量子图像的安全。

本文接下来的安排如下, 第二部分简单介绍一下 NEQR 量子图像表示方法, 一些量子功能模块以及经典图像下的差值扩展算法, 第三部分是介绍在 NEQR 下的差值扩展算法, 第四部分是实验仿真, 然后是本文的总结。

2 基础知识

2.1 NEQR 量子图像表示法[24]

根据 NEQR 量子图像表示方法, 一幅 $2^n\times 2^n$ 的图像 I 可以表示为:

$$|I\rangle=\frac{1}{2^n}\sum_{i=0}^{2^{2n}-1}|c_i\rangle\otimes|i\rangle \tag{1}$$

$$\begin{aligned}&|i\rangle=|y\rangle|x\rangle=|y_{n-1}y_{n-2}\cdots y_0\rangle\\&|x_{n-1}x_{n-2}\cdots x_0\rangle,|y_i\rangle|x_i\rangle\in\{0,1\}\end{aligned} \tag{2}$$

$$\begin{aligned}&|c_i\rangle=\left|c_i^{q-1}\cdots c_i^1c_i^0\right\rangle,\\&c_i^k\in\{0,1\},k=q-1,\cdots,1,0\end{aligned} \tag{3}$$

序列 i 代表图像位置, 横坐标和纵坐标分别用 n 个量子比特来表示, x 序列代表横坐标, y 序列代表纵坐标。颜色用 q 个量子比特来表示(最多可表示 2^q 种颜色), 序列 c 代表图像的颜色信息。当使用灰度图像作为数据隐藏的载体时, 颜色信息需用到 8 个量子比特(2^8=256)。

图 1 是一幅 2×2 的灰度图像在 NEQR 表示方法下的一个例子, 有四个方块, 每个方块代表一个像素, 方块内第一行数字为灰度值, 第二行数字为坐标值, 它们均以二进制形式来表示。

该图像利用 NEQR 表示法可表示为:

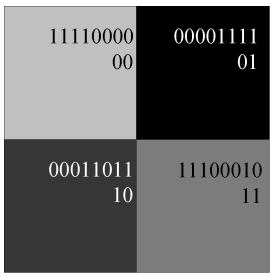


图 1　一幅 2×2 的灰度图像的 NEQR 表示[24]

Figure 1　A NEQR 2×2 gray image

$$|I\rangle=\frac{1}{2}[|11110000\rangle\otimes|00\rangle+|00001111\rangle\otimes|01\rangle+|00011011\rangle\otimes|10\rangle+|11100010\rangle\otimes|11\rangle] \tag{4}$$

2.2　差值扩展算法

2.2.1　信息嵌入过程

差值扩展可逆数据隐藏算法最先是在文献[4]中提出，假设现在有一对像素，它们的值分别为 x=205 和 y=198，要嵌入的信息 b 为 1。首先算得 x 和 y 之间的平均值 l(向下取整)为 201，差值的绝对值 h 为 7。2h=14，二进制为 1110。信息就嵌入到最后一位 0 上去，嵌入信息后的差值就变为 1111，即 15。用 $\hat{h}$ 来表示扩展后的差值，则上述过程可描述为：$\hat{h}=2\times h+b$

然后根据 $\hat{h}$ 和 l 计算新的像素值(x_1, y_1):

$$x_1=l+\left\lfloor\frac{\hat{h}+1}{2}\right\rfloor=201+\left\lfloor\frac{15+1}{2}\right\rfloor=209,$$
$$y_1=l-\left\lfloor\frac{\hat{h}}{2}\right\rfloor=201-\left\lfloor\frac{15}{2}\right\rfloor=194 \tag{5}$$

这样处理会带来一个问题，那就是新像素的值可能会不在 0~255 这个范围内，从而导致溢出失真。为此，在进行信息嵌入之前需要进行嵌入测试，并把那些可能产生溢出的像素对的位置进行标记，本文设计了溢出像素对的标记方法，第 3 部分会介绍。溢出的像素对不进行差值扩展，维持原来的值不变。在文献[4]里 Tian 给出了一个用于判定像素对能否进行差值扩展的判定标准，如下公式(6)所示。

$$|2\times h+b|\leqslant\min(2\times(255-l),2l+1) \tag{6}$$

2.2.2　信息提取和载体图像复原过程

信息提取和载体图像复原过程大致跟信息嵌入过程一样，这里同样以前面的像素对为例进行介绍。嵌入信息后的像素对为(x_1=209, y_1=194)。同样地，算得 x_1 和 y_1 之间的平均值 l(向下取整)为 201，差值的绝对值 $\hat{h}$ 为 15。把 $\hat{h}$ 除以 2，得到原先的差值 h(向下取整)为 7，余数为 1，该余数即为之前嵌入的信息 b。算出 l 和 h 后就能够得到原来的像素值 x y:

$$x=l+\left\lfloor\frac{h+1}{2}\right\rfloor=201+\left\lfloor\frac{7+1}{2}\right\rfloor=205$$
$$y=l-\left\lfloor\frac{h}{2}\right\rfloor=201-\left\lfloor\frac{7}{2}\right\rfloor=198 \tag{7}$$

2.3　量子加法器[25]、量子减法器[25]、量子计数器[26]和量子比较器[27]

量子加法器如图 2 所示[25]。输入 a 跟 b 都由 n 个量子比特组成。在进行加法时会用到 n+1 个辅助量子比特，其中的 n 个量子比特用来表示和，剩下的 1 个量子比特作为进位。它的功能为：$|a,b\rangle\rightarrow|a,a+b\rangle$。$a$+$b$ 由 n+1 个量子比特组成。

当量子加法器中逻辑门的顺序相反时，它就转变成了量子减法器(图 3 所示)[25]。类似地在进行减法会用到 n+1 个辅助量子比特，其中 n 个量子比特用来表示差，剩下的 1 个量子比特作为借位。它的功能为：$|a,b\rangle\rightarrow|a,b-a\rangle$。$b$–$a$ 也是由 n+1 个量子比特组成，多出来的量子比特表示借位，当借位为 0 时表示 b 大于等于 a，为 1 时表示 a 大于 b。而当 a 大于 b 时，实际功能为：$|a,b\rangle\rightarrow|a,2^n+(a-b)\rangle$

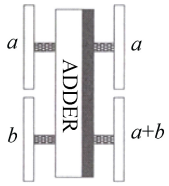


图 2　量子加法器

Figure 2　Quantum adder

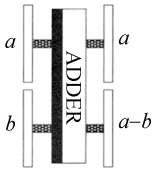


图 3　量子减法器

Figure 3　Quantum subtractor

量子计数器(图 4 所示)[26]是用来统计输入共有多少个 1 的，b 是输入的量子比特，a_0 到 a_{n-1} 是初始值为 00…0 的量子比特序列。每输入 1 个 1 量子比特，$a_{n-1}a_{n-2}...a_0$ 就加 1，最后得到的 a 序列的值即为 1 的总数。

量子比较器(图 5 所示)[27]通过两个输出 c_1c_0 来反映两个输入 ab 之间的大小关系，其中 ab 各由 n 个量子比特所组成。ab 之间的大小关系可以根据下面的准则来判断：1)如果 c_1c_0 =10，则 a>b; 2)如果 c_1c_0 =01，则 a<b; 3)如果 c_1c_0 =00，则 a=b。

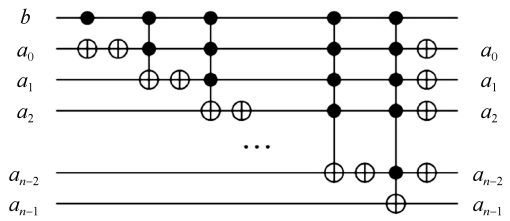


图 4　量子计数器

Figure 4　Quantum counter

为了避免介绍过程中过于繁琐, 后面的内容中用 COUNTING 模块来表示量子计数器, 用 COMPARATOR 模块来表示量子比较器。

3　NEQR 下的差值扩展可逆数据隐藏算法

本文的算法流程如图 6 所示。发送方通过 NEQR 量子图像表示法获得量子图像 I, 然后利用差值扩展算法把信息 M 嵌入到量子图像 I 中, 得到含有信息的量子载体图像 I'。接收方利用检测位判断信息在传输过程中是否安全, 在安全的情况下接收到含有信息的量子载体图像 I', 再从中提取出信息 M 和恢复量子载体图像 I。信息的嵌入、提取以及载体图像的恢复都是在量子图像中实现的。下面具体地把从信息嵌入到信息提取和载体图像恢复的过程进行介绍。

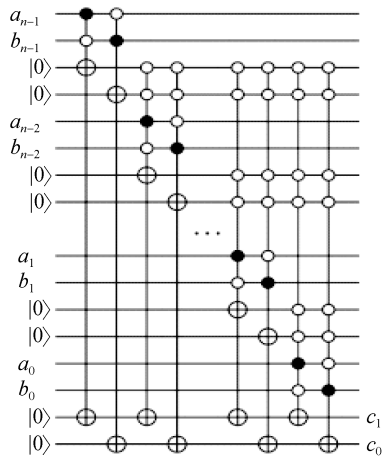


图 5　量子比较器

Figure 5　Quantum comparator

3.1　信息嵌入过程

信息嵌入过程的第一步是对载体图像 I 的所有的像素对进行检测, 得到溢出像素对的对数 L 和它们的位置。假设载体图像 I 的大小为 $2^n\times2^n$, 下面以图像 I 第一对像素为例来介绍检测过程。其余像素对的检测过程除了坐标不同以外都是类似的。

检测像素对的过程主要用到公式(6), 首先要把像素对的均值 l 和扩展后的差值 $\hat{h}$ 算出来。在 NEQR 表示下, 第一对像素的坐标为 0...0(共 n 个 0)×0...0(共 n 个 0)和 0...01(共 n–1 个 0)×0...0(共 n 个 0)。由于量子加法器第二个输出为和, 这样会导致原来的第二个像素值丢失, 故进行加法前先准备 8 个 0 量子比特组成量子比特序列, 通过量子逻辑门使其值与第二个像素的像素值 C 相同, 用来作为量子加法器的第二个输入。如图 7 所示。

图 7 中外黑里白的圆圈代表坐标的控制, 只有 x 或 y 的坐标值符合圆圈右上的要求时才进行相应的操作。以 C_0 为例, 在符合坐标要求下, 若 C_0 本身为 0, 则不进行取反; 若 C_0 为 1 则将 0 量子比特取反, 这就能使 0 量子比特变换后的值与 C_0 相同。通过对 8 个 0 量子比特进行同样的操作, 就能使它们的值和 C_7~C_0 相同。为了表示简洁, 图 7 可表示成图 8 的简洁表示形式, 图 8 中 C 代表 C_7~C_0。接下来的线路图也会用类似的简洁表示形式。

然后要计算两个像素值的均值, 过程如图 9 所示, C^1 和 C^2 分别代表第一个和第二个像素值。两个像素值的和的量子比特序列为 $Jl_7...l_0$, 其中 J 是进位。将和右移一位就能实现除以 2 的功能, 对 $Jl_7...l_0$ 进行量子比特两两互相交换就能实现右移一位的操作, 最终得到的 $l_7l_6...l_0$ 即为所要求的两个像素的平均值 l。把该功能模块称为 COUNT l。

图 10 为算两个像素值的扩展差值的线路图。h_8 为借位, h_7~h_0 为差的绝对值 h。由第二节可知 $\hat{h}=2h+b$, 把 h_8~h_0 向左移一位即可实现差值乘 2 的操作。经过移位之后 $\hat{h}_8$ 存放的是原来 h_7 的值, 若 $\hat{h}_8$ 值为 0, 那么左移一位后 $\hat{h}_7$ ~$\hat{h}_0$ 的值为 $2h$。若 $\hat{h}_8$ 值为 1, 左移一位后 $2h$ 需用 $\hat{h}_8$ ~$\hat{h}_0$ 来表示, 这时就可以直接判定这个像素对差值扩展后肯定会溢出, 因此只要 $\hat{h}_8$ 的值为 1, 则可直接认为像素对溢出。$\hat{h}_0$ 用于存放要嵌入的信息, 溢出检测时统一把 1 嵌入进 $\hat{h}_0$, 使得 $\hat{h}$ 达到最大值来进行检测。

l 和 $\hat{h}$ 都算出来后要进行像素对是否溢出的判定。由公式(6)可知, $\hat{h}$ 要和 $2\times(255–l)$、$2l$+1 中最小的进行比较, 因此要先确定 $2\times(255–l)$和 $2l$+1 中的最小值, 线路图如图 11 所示。

当 l_7 为 1 时, 说明均值 l 不小于 128, 最小值为 $2\times(255–l)$; 若为 0, 说明 l 小于 128, $2l$+1 为最小值。图 11 中, 一个 0 量子比特变换后的值与 l_7 相同, 用它来进行判定。减法器的第一个输入为 8 个 1 量子比特组成的序列, 即十进制中的 255。当 l_7 为 0 时,

$2l$+1 为最小值, 则把减法器输入端的 8 个 1 量子比特全部取反实现置零的操作, 使得减法器进行的运算为 0 和 l 相减, 这样输出的差的前 8 位量子比特仍是 l, 然后把 l 乘以 2 并加 1 得到 $2l$ +1。这里得到 $2l$+1 所用到的方法和前面算 $2h$+1($2h$+b 中 b 为 1 的情况)的方法是一样的。当 l_7 为 1 时, 2×(255–l)为最小值。这时照常进行减法, 得到差 255–l 后也用同样的方法乘以 2, 不同之处在于这里是把 0 和 l_0 进行互换, 就是说并没有加 1, 这样就能得到 2×(255–l)。用 l'表示 2×(255–l)和 $2l$+1 中的最小值。

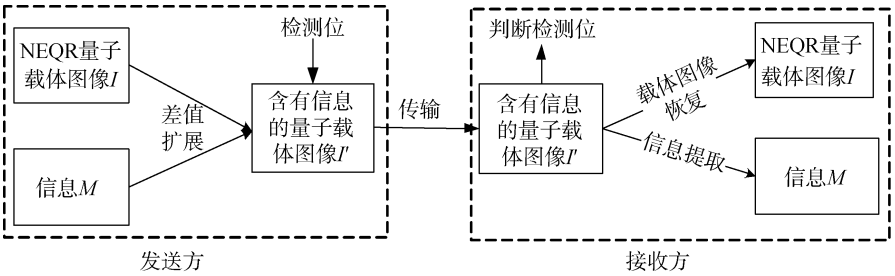


图 6　算法流程图

Figure 6　Procedure of proposed scheme

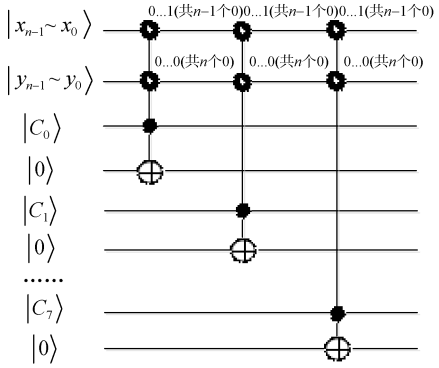


图 7　使 8 个 0 量子比特变换后的值与第二个像素值 C 相同

Figure 7　Modify eight 0 qubits according to the second input pixel value

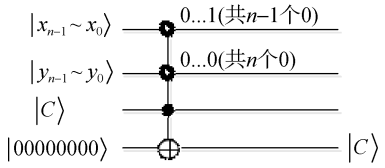


图 8　图 7 的简洁表示法

Figure 8　Simple representation of Fig.7

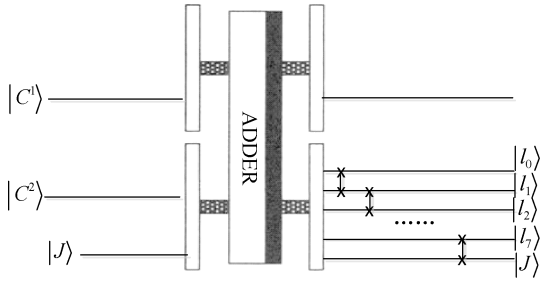


图 9　计算平均值(COUNT l 模块)

Figure 9　Calculate average(COUNT l module)

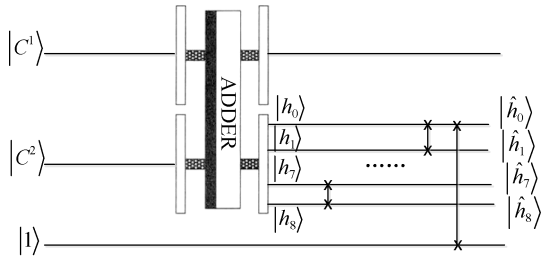


图 10　检测时计算扩展差值 $\hat{h}$

Figure 10　Calculate expanded difference $\hat{h}$ during the test

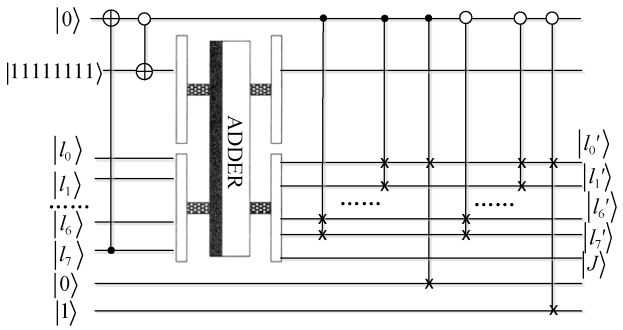


图 11　判定 2×(255–l)和 $2l$+1 中最小值

Figure 11　Judge the minimum value of 2×(255–l) and $2l$+1

然后要对 l' 和 $\hat{h}$ 进行大小比较, 对 $\hat{h}$ 大于 l' 的像素对要进行标记。下面介绍标记的方法, 先新建一幅 NEQR 下的与载体图像 I 大小相同的空图像, 两个初始值为 0 的量子比特 f_1f_2 作为像素值。把溢出像素对中第一个像素的坐标保存下来, 保存方法后面会介绍, 并在空图像上把相同坐标下的像素值 f_1 置 1, 如图 12 所示。

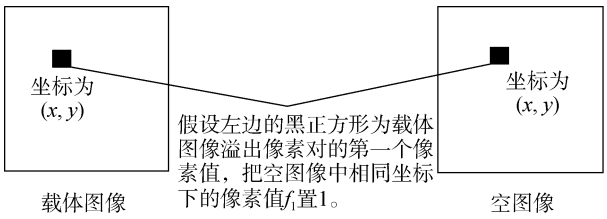


图 12　在空图像上进行溢出像素对位置的标记
Figure 12　Mark overflow pixel pairs on an empty image

所有的像素对都检测完之后, 会得到溢出像素对的对数 L, 以及完成了标记操作的空图像, 空图像上 f_1 为 1 的位置代表这个位置的像素对是不能进行差值扩展的, 图 13 介绍了 l'和 $\hat{h}$ 的比较以及标记空图像的过程。

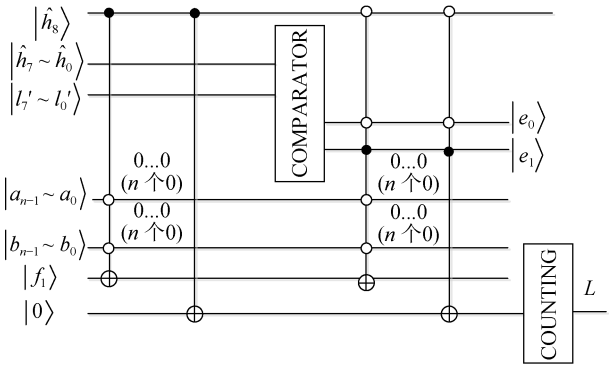


图 13　l'和 $\hat{h}$ 的比较以及在标记空图像
Figure 13　Compare l' and $\hat{h}$ and mark the empty image

用 ab 表示空图像, 大小为 $2^n \times 2^n$, 两个 0 量子比特 f_1 f_2 作为像素值。记这个空图像的横轴为 a, 纵轴为 b。图 13 为判定第一个像素对时的情况, 以它为例子进行说明。当 $\hat{h}_8$ 为 1, 像素对溢出, 这时在空图像 ab 中找出对应的像素对, 并把像素对中第一个像素下的f_1置为 1, 同时对 0 量子比特取反以进行计数。当 $\hat{h}_8$ 为 0 时进行后续判定, 比较器输出 e_1e_0 为 10 时说明 $\hat{h}$ 大于筛选后的 l', 这表明像素对溢出, 同样进行相应的 f_1 置 1 和计数操作。当所有的像素对都检测完后, 不仅会得到将溢出像素对位置标记好的空图像 ab, 还会得到溢出像素对的对数 L。

接下来, 利用载体图像最前面的像素的 LSB 位来保存溢出像素对中第一个像素的坐标信息。对于一幅 $2^n \times 2^n$ 的图像 I, 每个像素的坐标需要用 $2n$ 个量子比特来表示。也就是说, 每有一对溢出像素, 就要保存 $2n$ 个量子比特的坐标信息, 若共有 L 对溢出像素对, 则总共要保存 $L \times 2n$ 个量子比特。只要有第一个像素的坐标, 就能确定像素对的位置。

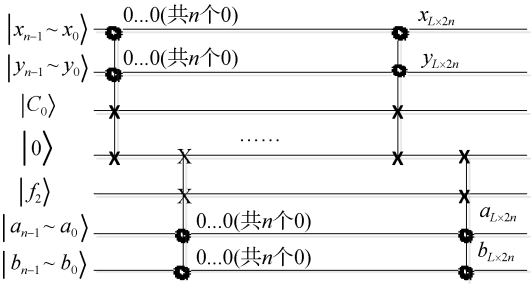


图 14　保存前 $L \times 2n$ 个像素的 LSB 位
Figure 14　Save the original first $L \times 2n$ LSBs

由于要用到前 $L \times 2n$ 个像素的原 LSB 位来保存坐标值, 因此要先把这 $L \times 2n$ 个原 LSB 位保存起来, 本文是把这些 LSB 位保存在空图像 ab 对应位置的 f_2 上。这 $L \times 2n$ 个像素值的原 LSB 会与待嵌入的信息 M 组合在一起成为最终要嵌入的信息 M'。像素值原 LSB 位的保存过程如图 14 所示。图 14 中, 像素值的 LSB 位为 C_0, 先把它和 0 量子比特进行互换, 然后再跟 f_2 互换, f_2 初始为 0, 经过两两互换后 f_2 的值变为 C_0 的值, C_0 的值变为 0, C_0 就可以用来存放溢出像素对的坐标值。图 14 中 $x_{L\times 2n}$, $y_{L\times 2n}$ 分别为载体图像 I 中第 $L \times 2n$ 个像素的横坐标和纵坐标的值, $a_{L\times 2n}$, $b_{L\times 2n}$ 分别为空图像 ab 中第 $L \times 2n$ 个像素的横坐标和纵坐标的值。不同载体图像的溢出坐标对数 L 不确定, 导致第 $L \times 2n$ 个像素的坐标没有准确的值。故用类似 $x_{L\times 2n}$ 的方式来表示不确定的坐标值。

然后是对溢出像素对的坐标信息进行保存, 坐标保存的方法是把溢出像素对中第一个像素的横坐标和纵坐标按照 $y_{n-1}y_{n-2}...y_0x_{n-1}x_{n-2}..x_0$ 的顺序保存在载体图像 I 的前 $L \times 2n$ 个像素的 LSB 位上。图 15 为这部分功能的线路图。

首先要在空图像 ab 中找出 f_1 为 1 的坐标, 从最开始的坐标 0...0(共 n 个 0) ×0...0(共 n 个 0)开始, 当 f_1 的值为 1 时需要进行坐标保存。f_1' 为 0 量子比特, 经过变换后 f_1'与 f_1 的值相同, 用 f_1' 来控制是否进行坐标保存。$k_{2n-1}k_{2n-2}...k_0$ 是初始值为全 0 的量子比特序列, 用于记录当前所检测的坐标, k_{2n-1} 到 k_n 这 n 个量子比特代表纵坐标的值, k_{n-1} 到 k_0 这 n 个量子比特代表横坐标的值。当一个坐标检测完后对量子比特序列 k 进行加 1 操作, 这样就能使得 k 的值跟要检测的坐标的值 $y_{n-1}y_{n-2}...y_0x_{n-1}x_{n-2}..x_0$ 是一直同步的。每次检测到 f_1 为 1, 计数器都会加 1, 根据计数器所记录的数值 N 来确定坐标保存的位置。也就是说, 只要知道了当前检测到的溢出像素对的对数, 那么坐标保存的位置也就得到确定。因为每个坐标值由 $2n$ 个量子比特表示, 那么每保存一个坐标值就要用到 $2n$ 个

像素值的 LSB 位, 因此计数器的值 N 和存放坐标的位置是捆绑在一起的。图 15 中用 N_1,N_2,…,N_L 代表计数器为 1,2,…,L 时的量子比特序列, N_1 对应的坐标存放位置为第 1 个像素到第 $2n$ 个像素的 LSB 位, N_2 对应的坐标存放位置为第 $2n$+1 个像素到第 $4n$ 个像素的 LSB 位……以此类推。不同载体图像的溢出坐标对数 L 不确定, 这就导致了要嵌入的坐标数和要嵌入的位置并不确定。图 15 中除了第一个像素的坐标值以外, 其余像素的坐标均用(x_t,y_t)来表示, x_t,y_t 分别表示第 t 个像素的横坐标和纵坐标。在找到匹配的保存位置后, 按照 $k_{2n-1}k_{2n-2}\cdots k_0$ 的顺序依次把 k 量子比特序列的值存放到 $2n$ 个像素的 LSB 位 C_0 上。C_0 在前一步操作后变为 0, 因此只需进行相应的取反的操作就能使得这 $2n$ 个 C_0 的值与 k 序列的值一样, 这样就达到了用 $2n$ 个像素保存坐标值的目的。当完成了 N_L 对应的坐标保存时, 代表所有溢出像素对的坐标已经保存完毕。

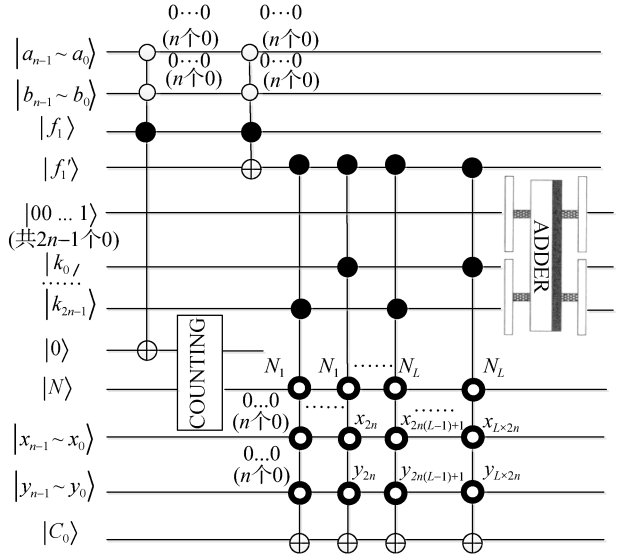


图 15 溢出像素的坐标存储

Figure 15 Save the coordinates of overflow pixel pairs

坐标信息存储完后就开始嵌入信息, 假设要嵌入的信息共有 M 个量子比特, 加上之前存储在 f_2 的 L×2n 个原 LSB 位, 共有 M+L×2n 个量子比特的信息要嵌入, 为了表示方便, 记最后要嵌入的信息 M+L×2n 为 M'。M' 中前 M 个量子比特是嵌入的信息, 后 L×2n 个量子比特为原 LSB 位。信息是在差值扩展的时候嵌进去的, 因此下一步就是计算新的像素值。

从第 L×2n+1 个像素开始进行差值扩展。图 16 的功能主要是计算均值 l 和扩展后的差值 $\hat{h}$, $(x_{L\times 2n+1}$, $y_{L\times 2n+1})$和$(x_{L\times 2n+2}$, $y_{L\times 2n+2})$为一对像素对。计算均值 l 用 COUNT l 模块即可。计算 $\hat{h}$ 时用到一个 count $\hat{h}$ and insert message 模块, 它的示意图如图 17 所示。图 16 中 C^1 代表像素对中第一个像素的值, C^2 代表像素对中第二个像素的值。

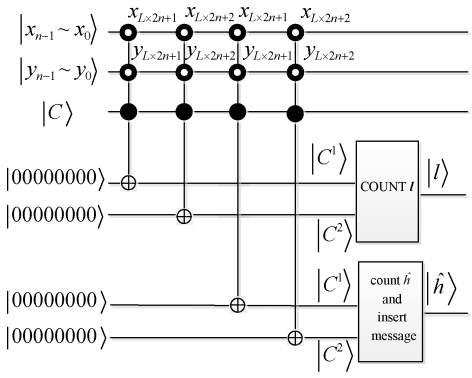


图 16 计算均值 l 和扩展后的差值 $\hat{h}$

Figure 16 Calculate average l and expanded difference $\hat{h}$

图 17 为 count $\hat{h}$ and insert message 模块。计算过程基本与前面检测算 $\hat{h}$ 相同, 不同之处在于检测过程 $\hat{h}_0$ 统统嵌入 1, 而这里要嵌入的是信息 M', M' 的嵌入受 f_1'控制, f_1'被设置成与 f_1 相同。只有 f_1'为 0, 即当前坐标对不是溢出坐标对时才把 M' 嵌进 $\hat{h}_0$ 中。

新像素值计算的线路图如图 18 所示, 主要用到式子(5)。$\hat{C}^1$ 表示新的大像素值, $\hat{C}^2$ 表示新的小像素值。图 18 用到了一个除以 2 的模块, 标记为/2, 如图 19 所示。通过整体向右移一位即可实现除以 2 的操作。(所有量子比特通用, 即可以把 $\hat{h}$ 换成其他量子比特)。

算出了新的像素值后进行像素值的替换, 图 20 给出了第 L×2n+1 个像素和第 L×2n+2 个像素的像素替换过程线路图。后面的像素对的像素值替换过程都是基本相同的。首先要对原来的两个像素值进行大小比较, 当第一个像素大于等于第二个像素时, 用较大的新像素 $\hat{C}^1$ 替换掉第一个像素, 较小的新像素 $\hat{C}^2$ 替换掉第二个像素。反之则用 $\hat{C}^1$ 替换掉第二个像素, $\hat{C}^2$ 替换掉第一个像素。这两个原像素的大小关系取决于比较器的输出, 当第一个像素比第二个像素大时, e_1e_0 为 10; 两个像素相等时 e_1e_0 为 00; 第一个像素比第二个像素小时, e_1e_0 为 01。因此 e_0 决定替换顺序, e_0 为 0 时 $\hat{C}^1$ 替换掉第一个像素, $\hat{C}^2$ 替换掉第二个像素; e_0 为 1 时 $\hat{C}^1$ 替换掉第二个像素, $\hat{C}^2$ 替换掉第一个像素。另外还要用 f_1 来控制是否进行替换操作。若当前像素对在空图像 ab 中所对应的溢出标志 f_1 为 1, 说明该像素对不符合嵌入条件, 不进行替换操作。类似地用 f_1' 代替 f_1 进行控制操作。

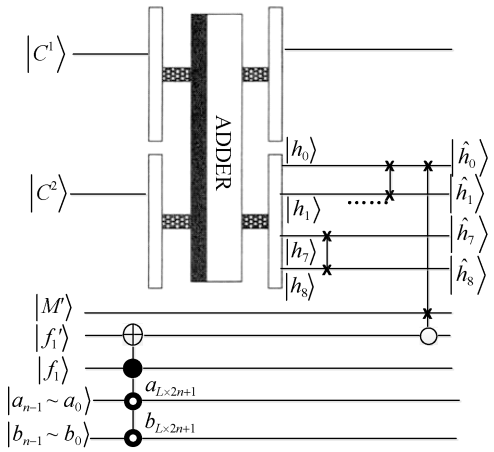


图 17 count $\hat{h}$ and insert message 模块

Figure 17 Calculate $\hat{h}$ (count $\hat{h}$ and insert message module)

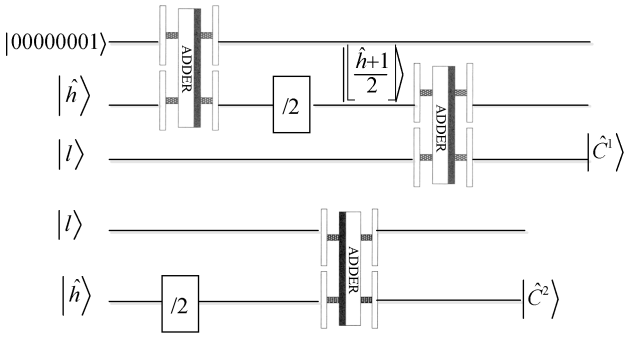


图 18 新像素值的计算

Figure 18 Calculate the new pixel pair

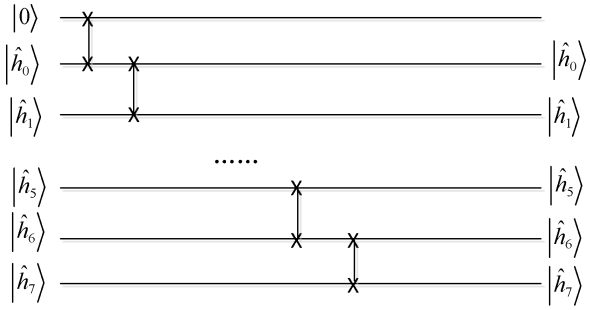


图 19 除以 2 模块(标记为/2)

Figure 19 Input divided by 2(/2 module)

信息 M' 都嵌入完后, 得到含有隐藏信息的载体图像 I', 整个嵌入过程到此结束。发送方在传输 I' 给接收方的过程中, 可能会受到外界的攻击或篡改。为了让接收方能够判定接收到的信息是否受到攻击或篡改, 可以利用诱骗光子作为检测位来实现这个功能[28]。发送方在发送 I' 给接收方之前, 可在 I' 的量子比特序列中的某些位置插入若干个检测位, 并告知接收方检测位的所在位置以及测量方法。在传输过程中, 若传输的信息受到了外界的攻击或篡改, 检测位势必会发生变化。接收方在接收到 I'后, 可根据检测位的值的正确率来判定所收到的信息是否受到了攻击或篡改。

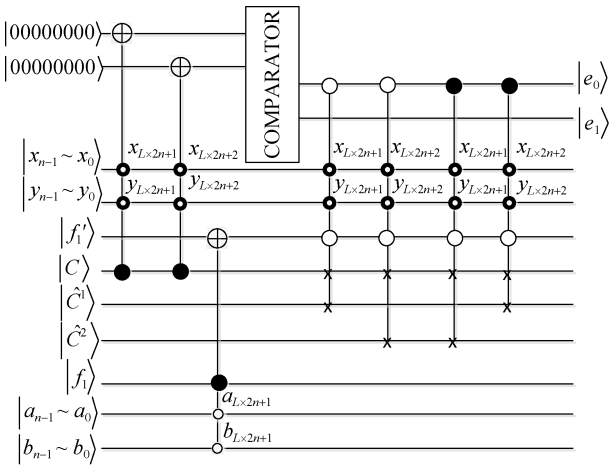


图 20 嵌入过程像素对的替换

Figure 20 Replacement of pixel pairs in embedded process

3.2 信息提取和载体图像恢复过程

接收方在收到 I'后, 先对检测位进行测量, 只有检测位的值的正确率在某个阈值范围内, 才能判定所接受到的信息没有受到攻击或篡改[28], 然后才开始信息提取和载体恢复过程。首先要做的是把溢出像素对的位置标记出来, 准备一幅 NEQR 下的与图像 I'大小相同的图像 LM, 它的像素值用一个 0 量子比特 f_1 来表示。图像 I'的前 $L\times 2n$ 个像素的 LSB 位(C_0)都存储着坐标值, 每个坐标信息按照 $y_{n-1}y_{n-2}...y_0x_{n-1}x_{n-2}.....x_0$ 的顺序存储在每 $2n$ 个像素值的 LSB 中, 根据这个坐标信息把图像 LM 上对应位置的 f_1 置 1。当 L 个溢出坐标都在图像 LM 上标记了之后, 这一步工作才算完成, 后续的还原工作就是靠图像 LM 上 f_1 的值来判定当前像素对是否有信息嵌入。图 21 以标记第一个溢出像素对为例介绍标记过程。

图 21 中, 量子比特序列 k 由 $2n$ 个 0 量子比特组成。从第一个坐标 0...0(共 n 个 0) ×0...0(共 n 个 0)开始, 每 $2n$ 个像素值的 LSB 位都按 k_{2n-1} 到 k_0 的顺序赋给量子序列 k, 其中 k_{n-1}~k_0 为溢出像素对的第一个像素的横坐标 x_{n-1}~x_0, k_{2n-1}~k_n 则为纵坐标 y_{n-1}~y_0。图像 LM 中, a 代表横坐标, b 代表纵坐标。只有当 a、b 的值分别为 k_{n-1}~k_0、k_{2n-1}~k_n 时, 才对坐标(a,b)下的 f_1 进行取反以实现置 1 操作。之后要把 k 序列用同样的方法恢复初始状态以便标记下一个溢出坐标对时继续使用。直到 L 个溢出坐标对都在图像 LM 上标记后, 该步骤才算完成。

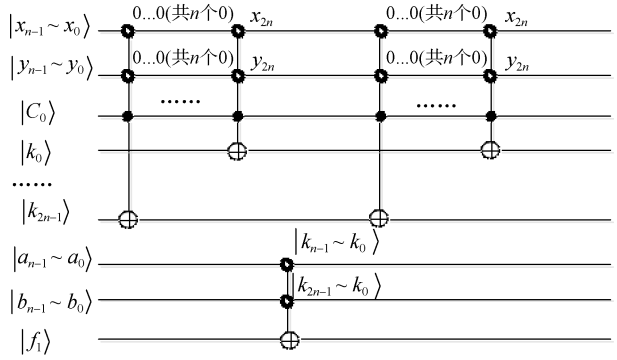


图 21 溢出坐标的提取和标记

Figure 21 Mark overflow pixel pairs on image *LM*

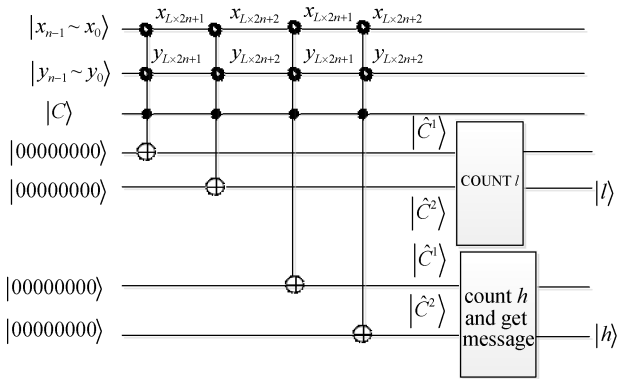


图 22 计算均值、差值以及提取信息

Figure 22 Calculate average *l*, original difference *h* and extract information

接下来是像素对和信息的还原。从第 $L\times2n+1$ 个像素开始, 共有 $M+L\times2n$ 对像素嵌入了信息。现以第 $L\times2n+1$ 个像素所在的像素对为例, 进行还原过程的介绍。首先要做的是计算像素对的均值, 以及原像素对的差值, 并提取出嵌入的信息, 过程如图 22 所示。计算均值 l 继续使用 COUNT l 模块即可, 而计算原像素对的差值 h 的时候用到了 count h and get message 模块, 这个模块不单只用来算 h, 还用来提取信息, $\hat{C}^1$ 和 $\hat{C}^2$ 分别表示第一个和第二个像素值。

count h and get message 模块的线路图如图 23 所示, 先对两个像素进行减法运算, 得出来的差值是 $\hat{h}$, 而 $\hat{h}=2h+b$。若该像素对有嵌入信息, 则嵌入的信息就在 $\hat{h}$ 的最后一位也就是 $\hat{h}_0$ 上。m 为 0 量子比特, 把 $\hat{h}_0$ 跟 m 进行交换操作, 嵌入的信息就在 m 中。后面还要通过 f_1 来判断这个 m 是不是发送方嵌入的信息。把 $\hat{h}_7$~ $\hat{h}_0$ 右移一位以实现除以 2 操作, 得到原像素对的差值序列 h_7~h_0, 即原像素对的差值 h。两个像素进行减法运算时还会得到一个表示借位的 $\hat{h}_8$, $\hat{h}_8$ 为 0, 说明 $\hat{C}^1$ 大于等于 $\hat{C}^2$; $\hat{h}_8$ 为 1, 说明 $\hat{C}^1$ 小于 $\hat{C}^2$。

l 和 h 计算出来后就可以计算原来的像素值, 计算过程基本与计算新像素值类似(如图 18 所示)。区别在于计算新像素用的是扩展后的差值 $\hat{h}$, 而计算原像素用的是原始差值 h。

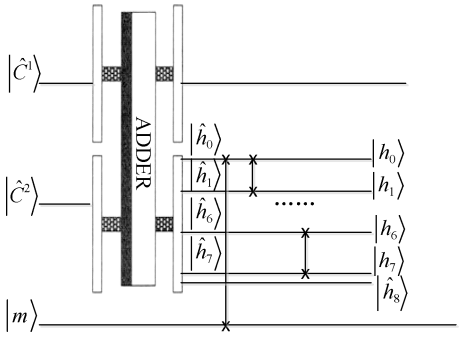


图 23 count *h* and get message 模块

Figure 23 Module: count h and get message

计算出原像素后进行像素对的替换, 同时还要判定提取出来的信息 m 是否为所嵌入的信息。图 24 以像素对($x_{L\times2n+1}$, $y_{L\times2n+1}$)和($x_{L\times2n+2}$, $y_{L\times2n+2}$)为例展示了这一过程。C^1 表示原像素对中较大的像素值, C^2 表示原像素对中较小的像素值。该像素对的溢出标志 f_1 在图像 LM 上的坐标为($a_{L\times2n+1}$,$b_{L\times2n+1}$), 当 f_1 为 0 时说明该像素对有信息嵌入。f_1' 被设置为与 f_1 相同, 由它来控制是否进行像素值的替换, 只有当 f_1'为 0 时才进行像素对的替换。$\hat{h}_8$ 为 0, 说明 $\hat{C}^1$ 大于等于 $\hat{C}^2$, C^1 应替换掉第一个像素 $\hat{C}^1$, C^2 应替换掉第二个像素 $\hat{C}^2$; 若 $\hat{h}_8$ 为 1, 则 C^1 替换掉 $\hat{C}^2$, C^2 替换掉 $\hat{C}^1$。以此实现像素对的复原。然后是嵌入信息的确认, U 为一组初始值为 0 的量子比特序列, 用于存储提取出来的嵌入信息, 当 f_1' 为 0 时, m 是发送方所嵌入的信息, 这时才把 m 和 U_i 进行互换, i=1,2,...,$M+L\times2n$。当完成了 $M+L\times2n$ 个像素对的还原后, U 里面也就包含了所有的嵌入信息, 其中前 M 个比特即为发送方想要发送给接收方的信息。

信息提取出来后, 原图像的还原工作尚未完成。U 里面还有 $L\times2n$ 个量子比特的信息为原载体图像 I 前 $L\times2n$ 个像素的原 LSB 位, 最后要做的就是把这 $L\times2n$ 个像素的原 LSB 位还原到原来的位置上。过程如图 25 所示。

从第一个坐标 0...0(共 n 个 0) ×0...0(共 n 个 0)开始还原像素值, 它的原 LSB 位存储在 U 的第 M+1 个比特中, 第二个坐标 0...1(共 n–1 个 0)×0...0(共 n 个 0)的原 LSB 位存储在 U 的第 M+2 个比特中……以此类推, 第 $L\times2n$ 个像素($x_{L\times2n}$, $y_{L\times2n}$)的原 LSB 位存储在 U 的第 $M+L\times2n$ 个比特中。当把前 $L\times2n$ 个像素的原 LSB 位都还原后, 就完成了整幅载体图像的还原。至此, 整个还原过程完成。

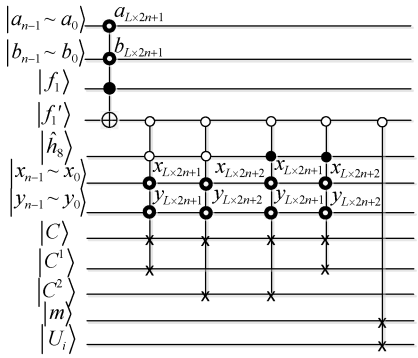


图 24 原像素对的替换和嵌入信息的确认

Figure 24 Replace the pixel pair and confirm the embedded information

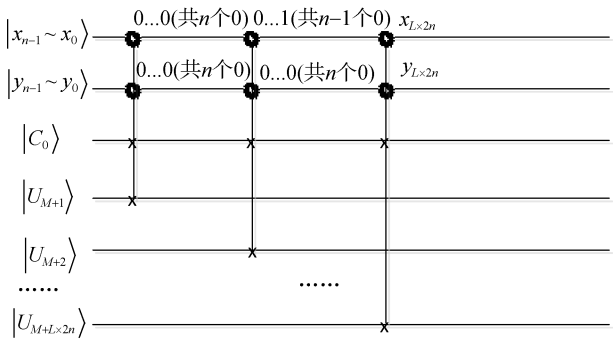


图 25 前 $L\times 2n$ 个像素的复原

Figure 25 Recovery of the first $L\times 2n$ pixels

4 仿真分析

目前量子计算机尚处于研发阶段, 因此仿真实验只能在经典计算机上进行。在经典计算机上用向量来描述量子图像, 用矩阵来描述各种量子计算算子。通过将图像向量与算子矩阵进行乘积运算可实现各种图像处理操作。目前在世界范围内经典计算机仍然是主流, 因此在显示和评价图像的时候需要经典计算机的辅助与支持, 故仿真的最后一步是要把量子图像转为经典图像。由量子计算原理可知, 每次测量都会使得量子态坍缩。当量子图像的数量足够多时, 就能够实现整幅图像的测量, 从而完成从量子图像到经典图像的转换, 测量的过程是以概率分布的形式进行仿真, 这也是目前量子图像水印算法文献中普遍采用的方法。仿真采用 matlab2012a 软件进行。仿真过程所使用的 5 个载体图像如图 26 所示, 载体图像均为 512×512 的灰度图像。嵌入的信息为二值图标的像素值组成的比特流, 图 27 为所用到需要隐藏的图标, 均为 100×100 的二值图像。

为有效评估算法的不可感知性, 本文用峰值信噪比(PSNR)来评估信息的隐藏效果即图像的视觉效果。峰值信噪比由均方误差来定义, 可用公式(8)来表示。

$$MSE=\frac{1}{\mathrm{mn}}\sum_{i=0}^{2^{m-1}}\sum_{j=0}^{2^{n-1}}[(I(i,j)-K(i,j))^2] \tag{8}$$

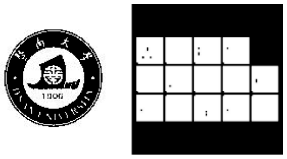
$$PSNR=20\times\log_{10}\left(\frac{MAX_I}{\sqrt{MSE}}\right) \tag{9}$$

公式(8)中 I 为原始载体图像, J 为含信息的图像。$I(i,j)$,$K(i,j)$代表了各自所对应的像素值。公式(9)为 PSNR 的计算公式, MAX_I 为图像的最大像素值。

(a) lena (b) goldhill (c) Elaine (d) airplane (e) cameraman

图 26 载体图像

Figure 26 The carrier images



(a) (b)

图 27 二值图标

Figure 27 The watermark icons

4.1 信息嵌入失真

在嵌入信息之前, 要先对载体图像检测并保存溢出像素对的坐标信息。故在嵌入信息之前就已经有失真, 这是保证可逆性所必须的。各载体图像保存坐标信息后的 PSNR 值如表 1 所示。

从表 1 可以看到, 不同的载体图像它们保存了坐标信息后的 PSNR 值有明显差异, 这是因为不同载体图像它们各自的溢出像素对数 L 都不相同, 这就使得在嵌入信息前需要保存的信息数也不同。PSNR 值越大说明载体图像的 L 值越小。L 值同时也影响到了最终要嵌入的信息大小, 因为存放坐标用

的原 LSB 位需要作为隐藏信息的一部分嵌入到载体图像中去。嵌入二值图标 a)、b)的像素值比特流后的图像分别如图 28、图 29 所示。它们的 PSNR 值如表 2 所示。

表 1 各载体图像保存了坐标信息后的 PSNR 值

Table 1 PSNR after saving coordinate information

载体图像	保存了坐标信息后的 PSNR 值
lena	72.9965
goldhill	74.2476
elaine	81.1768
airplane	67.0541
cameraman	61.2292

表 2 嵌入了二值图标后的 PSNR 值

Table 2 PSNR after embedding the watermark Icon

载体图像	嵌入二值图标 a)的 PSNR	嵌入二值图标 b)的 PSNR
lena	50.3438	50.4432
goldhill	53.1281	53.2797
elaine	49.2904	49.3882
airplane	47.6785	47.2780
cameraman	41.0593	41.0670

结合表 1、表 2 分析, 可以得出来三点结论: 1)载体图像的溢出像素对数 L 对图像的嵌入失真有明显的影响。表 1 中 PSNR 值较大的载体图像, 在表 2 中它的 PSNR 值表现也比较好。如载体图像 lena 它在表 1 中的 PSNR 值要比载体图像 airplane 要大, 在表 2 中 lena 的 PSNR 表现也比 airplane 要好。2)溢出像素对数 L 并不是影响嵌入失真唯一的因素, 表 1 中载体图像 elaine 的 PSNR 值是五幅图像中最好的, 然而从表 2 中可以看出来, 嵌入了二值图标后它的 PSNR 值表现并不是最好的。lena、goldhill 两幅图像在表 1 中的 PSNR 值均低于 elaine, 但嵌入了二值图标后这两幅图像的 PSNR 表现却要优于 elaine。出现这种情况是因为差值扩展算法的原理, 在产生新像素对的时候, 两像素值之间的差值越小, 所生成的新像素对跟原像素对之间的差值也越小, 进而会使得图像失真越小。因此若载体图像像素对之间差值不大, 则失真会比较小, 反之则大。因此会出现像载体图像 elaine 这种情况, 虽然图像的 L 小但是最后的 PSNR 值表现反而不如一些 L 更大的图像。3)从表 2 可看出, 嵌入不同的信息, PSNR 值基本稳定, 说明该算法性能主要决定因素还是在载体图像。

(a) lena (b) goldhill (c) Elaine (d) airplane (e) cameraman

图 28 嵌入了二值图标 a)的载体图像

Figure 28 Carrier images embedded with watermark icon a)

(a) lena (b) goldhill (c) Elaine (d) airplane (e) cameraman

图 29 嵌入了二值图标 b)的载体图像

Figure 29 Carrier images embedded with watermark icon b)

4.2 复杂度

量子计算机比经典计算机拥有更强更快的计算能力, 这主要体现在复杂度上, 复杂度分为空间复杂度和时间复杂度。本文提出的量子图像差分扩展算法与经典图像差分扩展算法相比, 降低了空间复杂度。经典图像的尺寸为 $2^n\times2^n$ 的灰度图像需要用 $2^n\times2^n\times8$ 个比特来表示, 本文所使用的 NEQR 量子图像表示只需 $2n+8$ 个量子比特即可表示一幅尺寸为 $2^n\times2^n$ 的灰度图像, 与经典图像需要 $2^n\times2^n\times8$ 个比特来表示相比, 极大地降低了空间复杂度。这主要得益于量子态的叠加特性[8]以及 NEQR 的特点。

时间复杂度是指执行一个算法所需的时间, 对

于经典计算机来说, 时间复杂度是用执行算法所需步数来表达, 而量子计算机中算法的时间复杂度是以逻辑门的数量来描述[29]。现对算法分成 3 个阶段分别分析其时间复杂度。首先是统计溢出像素对数阶段。经典差分扩展算法下统计一个溢出像素对时需要 16 次操作, 一个 $2^n\times2^n$ 的灰度图像共有 2^{n-1} 个像素对, 时间复杂度为 $O(2^{2n+3})$; 本文设计的量子算法大量地用到了 n-CNOT($n\geqslant3$)量子门, 故复杂度的大小主要取决于 n-CNOT 门的个数。在统计溢出像素对时用到 8 个(2n+1)-CNOT 量子门, 12Ln 个 2n-CNOT 量子门和 Ln 个(5n+1)-CNOT 量子门, 可算得复杂度为 $O(348Ln^2)$。通过比较两个复杂度可以看出随着图像尺寸的增大, 经典差分扩展算法在统计溢出像素对数阶段的时间复杂度呈指数增长, 远大于本文算法的时间复杂度。然后是信息嵌入阶段。假设待嵌入的信息共有 m 个比特, 经典差分扩展算法下嵌入一个比特需要 8 次操作, 时间复杂度为 8m; 本文算法在信息嵌入阶段用了 32 个(2n+1)-CNOT 量子门和 64 个(2n+2)-CNOT 量子门, 复杂度为 O(2304n)。比较两个复杂度可以发现, 经典差分扩展算法嵌入信息时的时间复杂度会随着嵌入信息量的增大而增大, 而本文算法嵌入信息阶段的时间复杂度只与图像尺寸有关。最后是恢复阶段。经典差分扩展算法恢复过程复杂度为 O (8m+4nL), 本文算法在恢复过程用了 32 个(2n+1)-CNOT 量子门, 64 个(2n+2)-CNOT 量子门和 6Ln 个 2n-CNOT 量子门, 复杂度为 $O(144Ln^2)$。与信息嵌入阶段类似, 经典差分扩展算法在恢复阶段的时间复杂度会随着嵌入的信息量和图像尺寸增大而增大, 而本文算法恢复阶段的时间复杂度只与图像尺寸有关。

综合以上 3 个阶段来考虑, 随着图像尺寸以及嵌入信息量的增加, 本文所提出的量子图像差分扩展算法相较于经典图像差分扩展算法在时间复杂度上的优势会越来越明显。本文算法与[19-20]等现有的量子水印算法相比, 为了保证数据隐藏算法的可逆性, 相应的量子线路规模有所增加, 时间复杂度也有所增加。

4.3 可逆性和嵌入容量

仿真实验中, 我们对所提出的数据隐藏算法的可逆性进行了测试, 结果显示: 1)在没有攻击的情况下, 从含有信息的图像中提取得到的比特流同嵌入时的完全相同, 即嵌入的信息能够完全无误正确提取; 2)在信息提取后能对载体图像进行完美的无损恢复。这很好地验证了所提出的数据隐藏算法是可逆的。在实际运行中, 如果有若干个量子门出现错误, 就会导致整个系统出现错误。本文主要设计基于 NEQR 图像、具有可逆性的可逆数据隐藏算法, 主要考虑在没有攻击下如何提取信息和恢复原始载体, 一旦信息不能正确提取, 则认为载体图像遭受攻击。从可逆数据隐藏的角度, 量子门发生错误时导致的信息提取错误可以看成是攻击的一种。在将来的研究中, 如何区别系统量子门错误和实际的攻击是一个重要的考虑。在实际的应用中, 针对量子门发生的错误概率可以在信息嵌入时结合量子纠错码[30], 在牺牲部分嵌入容量的情况下提高信息提取的正确率。

为了确保所提出数据隐藏算法的可逆性, 不可避免地要牺牲部分嵌入容量来存放额外的信息, 主要为不能用于嵌入信息的像素对的位置信息。由于不同的载体图像在进行差值扩展时溢出的像素对的数目 L 不一样。根据 L 的不同, 需要额外嵌入的信息的比特数也会发生变化。因此对于一张载体图像, 它的理论嵌入容量为 0.5 bpp(2 个像素嵌入 1 个比特), 实际的嵌入容量则与 L 有关。对于 512×512 的灰度图像, L 个位置需要前 18L 个像素的 LSB 位来记录位置信息, 这些 LSB 的值作为水印的一部分进行嵌入(需要 36L 个像素来进行扩展嵌入), 此时实际的水印嵌入容量比 0.5 bpp 要小一些, 可计算为: $\dfrac{(512\times512-18\mathrm{L}-36\mathrm{L})\div2}{512\times512}$。

5 总结

本文在 NEQR 量子图像表示方法的基础上, 通过借鉴并改进经典图像处理算法中的差值扩展策略, 提出了一种量子灰度图像可逆数据隐藏算法。我们给出了可逆数据隐藏算法的所有量子线路图, 包括信息的嵌入、信息的提取和载体图像的恢复。借鉴经典的计算机仿真, 我们对所提出的数据隐藏算法的可逆性和嵌入失真进行了测试, 仿真结果显示所提出的数据隐藏算法是可逆的, 显示了本文所提出的可逆数据隐藏算法可作为一种潜在的安全技术用于量子图像的安全和保护。

量子信息隐藏是一个兴起的研究方向, 在此基础上, 我们下一步的工作有如下考虑: 1)根据量子图像的特点设计失真更小、嵌入容量更大的数据隐藏算法; 2)针对量子门可能的错误如何结合纠错码进行设计更加可靠数据隐藏算法也是我们的一个主要考虑; 3)研究利用量子的纠缠特性, 设计更为有效安全的量子信息隐藏算法。

参考文献

[1] Honsinger CW, Jones P, Rabbani M, Stoffel JC. Lossless recovery of an original image containing embedded data. *US Patent application,* Docket No: 77102/E-D, 1999.

[2] Feng JB, Lin IC, Tsai CS, Chu YP. Reversible watermarking: current status and key issues. *International Journal of Network Security,* 2006, 2(3): pp.161-171.

[3] Celik MU, Sharma G, Tekalp AM, Saber E. Lossless generalized-LSB data embedding. *IEEE Transactions on Image Processing,* 2005, 14(2): pp. 253-266

[4] Tian J. Reversible data embedding using a difference expansion. *IEEE Transactions on Circuits and Systems for Video Technology,* 2003, 13(8): pp. 890-896.

[5] Ni ZC, Shi YQ, Ansari N, Su W. Reversible data hiding. *IEEE Transactions on Circuits and Systems for Video Technology,* 2006, 16(3), pp. 354-362.

[6] Li XL, Yang B, Zeng TY. Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection. *IEEE Transactions on Image Processing,* 2011, 20(12): pp.3524-3533.

[7] Moore GE. Cramming more components onto integrated circuits. *Proceedings of the IEEE,* 1998, 86(1): pp.82-85.

[8] Feynman RP. Simulating physics with computers. *International Journal of Theoretical Physics,*1982,21(6-7): pp.467-488.

[9] Bennett CH, Brassard G. Quantum cryptography: public key distribution and coin tossing. *IEEE International Conference on Computers Systems and Signal Processing.*1984: pp.175-179

[10] Divincenzo DP, Leung DW, Terhal BM. Quantum data hiding. *IEEE Transactions on Information Theory* , 2001, 48 (3) : pp.580-598.

[11] Julio Gea-Banacloche. Hiding messages in quantum data. *Journal of Mathematical Physics,* 2002, 43 (9) : pp.4531-4536

[12] Qu ZG, Chen XB, Zhou XJ, et al. Novel quantum steganography with large payload. *Optics Communications,* 2010, 283(23): pp.4782-4786.

[13] Liao X, Wen Q, Song T, et al. Quantum steganography with high efficiency with noisy depolarizing channels. *IEICE Transactions on Fundamentals of Electronics Communications & Computer Sciences,* 2013, 96(10): pp.2039-2044.

[14] Zhang WW, Gao F, Liu B, et al. A watermark strategy for quantum images based on quantum Fourier transform. *Quantum Information Processing* , 2013, 12(4), pp.793-803.

[15] Song XH, Wang S, Liu S, et al. A dynamic watermarking scheme for quantum images using quantum wavelet transform. *Quantum Information Processing,* 2013, 12(12), pp.3689-3706

[16] Jiang N, Zhao N, Wang L. LSB based quantum image steganography algorithm. *International Journal of Theoretical Physics,*2016,55(1): pp.107-123.

[17] Jiang, N, Wang, L.A novel strategy for quantum image steganography based on Moir pattern. *International Journal of Theoretical Physics,* 2015, 54(3), pp.1021-1032.

[18] Naseri M, Heidari S, Baghfalaki M, et al. A new secure quantum watermarking scheme. *Optik - International Journal for Light and Electron Optics,* 2017, 139: pp.77-86.

[19] Miyake S, Nakamae K. A quantum watermarking scheme using simple and small-scale quantum circuits. *Kluwer Academic Publishers,* 2016, 15 (5): pp.1-16.

[20] Li P, Zhao Y, Xiao H, et al. An improved quantum watermarking scheme using small-scale quantum circuits and color scrambling. *Quantum Information Processing,* 2017, 16(5):127.

[21] Venegas-Andraca SE, Ball JL. Processing images in entangled quantum systems. *Kluwer Academic Publishers,* 2010, 9 (1) : pp.1-11.

[22] Latorre JI. Image compression and entanglement. *Computer Science,* 2005.

[23] Le PQ, Dong F, Hirota K. A flexible representation of quantum images for polynomial preparation, image compression, and processing operations. *Kluwer Academic Publishers,* 2011, 10 (1) : pp.63-84.

[24] Zhang Y, Lu K, Gao Y, et al. NEQR: a novel enhanced quantum representation of digital images. *Quantum Information Processing,* 2013, 12(8): pp.2833-2860.

[25] Vedral VV, Barenco A, Ekert A. Quantum networks for elementary arithmetic operations. *Physical Review A Atomic Molecular & Optical Physics,* 1996, 54(1):147.

[26] Ma L, Lu J. Construction of controlled quantum counter. *Chinese Journal of Quantum Electronics,* 2003, 20(1): pp.47-50.

[27] Wang D, University H, Kaifeng. Design of quantum comparator based on extended general Toffoli gates with multiple targets. *Computer Science,* 2012, 39(9): pp.302-306

[28] Fatahi N, Naseri M. Quantum Watermarking Using Entanglement Swapping. *International Journal of Theoretical Physics,* 2012, 51(7): pp.2094-2100.

[29] Jiang N, Wu W, Wang L, et al. Quantum image pseudocolor coding based on the density-stratified method. *Quantum Information Processing,* 2015, 14(5): pp.1735-1755.

Pastawski F, Yoshida B, Harlow D, et al. Holographic quantum error-correcting codes: toy models for the bulk/boundary correspondence. *Journal of High Energy Physics,* 2015, 2015(6): pp.1-55.

项世军 于 2006 年于中山大学获得计算机软件与理论专业博士学位。现任暨南大学信息科学技术学院教授。研究领域为多媒体信息安全。研究兴趣包括加密域信息隐藏、可逆水印和量子信息隐藏等。Email: Shijun_Xiang@qq.com

李豪 于 2016 年在暨南大学电子信息工程获得学士学位。现在暨南大学通信与信息系统专业攻读硕士学位。研究领域为图像水印，量子图像处理。Email: haoli777@ qq.com

宋婷婷 于 2014 年在北京邮电大学密码学专业获得军事学博士学位。现任暨南大学信息科学技术学院副研究员。研究领域为量子通信、量子密码。研究兴趣包括量子密钥分发协议的实际安全性、量子随机数生成器的实际成码率等等。
Email: tingtingsong@jnu.edu.cn