一种基于四值混沌阵列的数字图像加密算法

高山青 张士杰 刘 镔 罗向阳 刘粉林
(解放军信息工程大学 信息工程学院,郑州 450002)

摘 要 提出了一种基于四值混沌阵列的数字图像加密算法。利用混沌系统产生一个和图像大小相同的四值伪随机混沌阵列。加密时,将混沌阵列中不同的值分开处理,对不同的值对应的图像点像素进行不同的加密,然后根据给定的随机整数在混沌阵列中寻找加密结果的存放位置。解密是加密的逆过程。实验和分析结果表明,算法的时间复杂度和空间复杂度较低,加密效果较好,安全性较高。

关键词 图像加密 四值混沌阵列 时间复杂度 空间复杂度 安全性
中图法分类号: TP391.41 **文献标识码**: A **文章编号**: 1006-8961(2006)02-0244-07

An Image Encryption Algorithm Based on Four-value Chaotic Array

GAO Shan-qing, ZHANG Shi-jie, LIU Bin, LUO Xiang-yang, LIU Fen-lin
(*Information Engineering Institute, The PLA Information Engineering University, Zhengzhou* 450002)

Abstract This paper presents a digital image encryption algorithm based on four-value chaotic array. It uses chaotic maps to generate a four-value chaotic array whose size is as big as the image. In encryption processing, we deal with the different value in the chaotic array separately, and use different keys to encrypt pixels which correspond to different chaotic array value respectively and then put the result to the appropriate position in the chaotic array according to the given random integers. Decryption is the converse of encryption. The analysis and the results of experiments show that the scheme has low time complexity, low space complexity and good security.

Keywords image encryption, four-value chaotic array, time complexity, space complexity, security

1 引 言

随着 Internet 技术与多媒体技术的飞速发展,多媒体通信逐渐成为人们信息交流的一种重要手段。由于数字图像信息形象、生动,因而被人们广为使用,成为人们表达信息的重要手段之一。随之而来的是安全问题,有些数字图像可能会涉及到国家或企业机密;有些图像只希望被授权者使用;有些图像又可能会涉及个人隐私。如何保护这些图像数据的安全,加密无疑是一种较好的解决办法。因此,数字图像加密技术的研究在多媒体通信和保密通信中有着重要意义。

目前,较为常用的数字图像加密技术有基于矩阵变换/像素置换的图像加密技术、基于秘密分割和秘密共享的图像加密技术、基于现代密码学体制的图像加密技术以及基于混沌的图像加密技术等[1,2]。

基于混沌的图像加密技术主要分两类。一类是忽略图像数据的内容,把图像数据看成二进制数据流,利用混沌信号对图像数据流进行加密。这类算法解密的关键是混沌同步。但是 Dedieu 等人[3]指出:混沌同步对参数的敏感性不仅不意味着安全性,攻击者反而可以利用这一特点,用参数自适应同步

基金项目:国家自然科学基金项目(60374004);河南省杰出青年基金项目(0412000200);河南省高校杰出科研人才创新工程基金项目(2001CXKY008)
收稿日期:2004-12-18;**改回日期**:2005-05-23
第一作者简介:高山青(1980 ~)男。解放军信息工程大学信息工程学院硕士研究生。主要研究方向为数字图像加密、数字水印。E-mail: shanqing_gao@yahoo.com.cn

控制的方法对混沌的参数(即密钥)进行辨识,从而达到破译的目的。目前已有一些成功破译混沌掩盖加密方案的报道[1]。另一类是利用混沌来构造置换矩阵,采用像素置换的方式来加密[4,5]。这类方法的主要问题是没有改变图像数据值,加密的强度较低。

为此综合了两类混沌图像加密方法的思想,提出了一种基于四值混沌阵列的数字图像加密算法,它采用混沌系统生成一个和图像大小相同的四值伪随机混沌阵列(即将混沌迭代产生的轨迹四值化,混沌阵列中有 4 个不同的值,由于混沌系统的遍历性和均匀性,它们的数量是基本相等的),然后根据混沌阵列和给定的 4 个随机整数(本文称其为距离)对像素值进行加密,再利用混沌阵列和距离寻找加密后的像素值存放位置。实验和分析结果表明,该算法简单、加密效果较好、时间复杂度和空间复杂度较低、安全性较好,且易于实现。

2　相关理论及概念

混沌理论自 20 世纪 60 年代快速发展起来,并最终在 70 年代得到了基本确立。但时至今日仍没有给混沌下一个完全统一的定义。混沌最广为人知的特性就是所谓的"蝴蝶效应"(正规表述应该称为对初始条件和/或控制参数的敏感性,或者正的 Lyapunov 指数),这个特性使得由确定性方程产生的混沌轨道随着时间的流逝而变得越来越"不可预测"。

一些学者已经指出在混沌理论和密码学之间存在着紧密的联系。许多混沌系统的基本特性,如遍历性(ergodicity)、混合性(mixing)、确定性(exactness)和对初始条件的敏感性,都可以和密码学中的混淆(confusion)和散布(diffusion)概念联系起来。因此使用混沌系统去开发新的密码设计思路就变得很自然了[6]。

考虑到混沌系统在数字计算机上实现时会产生动力学特征退化的问题[7,8],文献[9]中提出了一种采用加扰和扩散的方法来改善数字化混沌系统的动力学特征退化。实验结果表明,该混沌系统具有较好的动力学特性,因此本文采用该混沌系统来生成混沌序列。当然,本算法也可以采用其他的混沌系统来生成混沌序列,只是在实验中发现该混沌系统性能较好,能满足我们的需求,故采用该混沌系统。

考虑如下一个具有$[0,a),[a,b),[b,1-a),[1-a,1)$4 个子区间的 1 维分段线性混沌映射:

$$g(x)=\begin{cases}4x & 0\leqslant x<a\\ 2-4x & a\leqslant x<b\\ g(1-x) & b\leqslant x\leqslant 1\end{cases}\tag{1}$$

选择对 $g(x)$ 定义区间的第 3 子区间 $c_3=[b,1-a)$ 进行扩散,将片断 c_3 按照 $e:(1-e)$ 的比例分成两段 $c_{31}=[b,b+\frac{e}{r}),c_{32}=[b+\frac{e}{r},1-a]$,$e$ 为扩散系数,r 为常量,x 为初值。取 $a=0.25,b=0.5,r=4$,按照选择性扩散的扰动算法形成新的分段线性混沌映射:

$$f(x)=\begin{cases}g(x) & x\notin c_3\\ g\left(\frac{4}{2e}(x-0.5)\right) & x\in c_{31}\\ g\left(\frac{\left(x-(0.5+\frac{e}{4})\right)}{1-e}+1-0.25\right) & x\in c_{32}\end{cases}\tag{2}$$

其中,$e\in(0,1)$ 为扩散系数,$x\in(0,1)$,x_0 为其初值,e,x 为实数。为简单起见,本文中 $e=0.001$ 为固定值,在构造混沌序列时只需要给定初值 x_0。关于该混沌系统的更多性质请参阅文献[9]。

3　加解密算法

本算法采用的四值混沌阵列,因为在实验时发现,对本算法来说,采用四值混沌阵列比采用常见的二值混沌阵列在二阶距、自然序[4,5]等性能有较明显的提高,但是考虑到对较小图像的加密,如 64 × 64 大小的图像,并且采用更多值的混沌阵列(如八值混沌阵列)对算法的性能没有明显的提高,因此采用四值混沌阵列。

给定混沌初值 x_0、4 个距离 d_0,d_1,d_2,d_3。首先使用 x_0 产生四值(0,1,2,3)混沌阵列,然后用距离 $d_i(i=0,1,2,3)$ 在混沌阵列中寻找置换位置。例如,对混沌阵列中的每个 0,首先加密其对应的图像点像素值,然后在混沌阵列中寻找和该点距离为 d_0 个 0 的位置作为加密后的像素值的存放位置,对混沌阵列中的其他值 1,2,3 同样处理,只是加密时密钥略有不同(具体见 3.2 和 3.3),距离分别对应为 d_1,d_2,d_3。在仿真实验和分析时发现:如果对每个位置为(i,j)的像素点都在混沌阵列中直接采用搜索距离 d_i 的方式寻找其像素值加密结果的存放位

置,那么算法的处理时间会随着 d_i 的增大而急剧地增加,因为假设每个像素的加密/解密时间为 t_1、图像大小为 $M\times N$、阵列中每搜索一步的时间为 t_2、搜索到存放位置后的处理时间为 t_3,那么算法的时间开销约为 $(t_1+t_3)\times(M\times N)\times\frac{1}{4}\sum_{i=0}^{3}d_i\times t_2$。由于 $0<\sum_{i=0}^{3}d_i<(M\times N)$,所以算法的平均时间开销约为 $\frac{1}{8}(t_1+t_3)\times(M\times N)^2\times t_2$。若问题的规模为 m,则时间复杂度为 $O(m^2)$,效率较低。因此,为降低算法的时间复杂度,在算法实现时不采用直接在混沌阵列中搜索位置的方法,而是在混沌阵列生成时构造 4 个 1 维数组,使用这 4 个数组分别记录混沌阵列中每个值(0,1,2,3)的位置,这样,对任意一个像素点,只需要计算一次数组的下标和访问一次 1 维数组就可以得到加/解密的像素值的放置位置。设计算数组下标并访问数组的时间 t_4,则算法的时间开销约为 $(t_1+t_3)\times(M\times N)\times t_4$,可见该时间与 x_0、d_i 无关。所以算法的时间复杂度降为 $O(m)$,效率较高,故采用这种方法实现本算法。

3.1 产生混沌阵列

设待加密图像大小为 $M\times N$。

(1)用户选定密钥混沌初值 x_0。

(2)设定迭代次数为 $M\times N$ 次。

(3)用式(2)进行迭代 $M\times N$ 次,得到混沌轨迹。

(4)给混沌轨迹设置阈值 $T_1=0.25$,$T_2=0.5$,$T_3=0.75$,将得到的轨迹四值化,得到长为 $M\times N$ 的 0、1、2、3 混沌序列。即如果 $0\leqslant f(x)\leqslant T_1$,置为 0;如果 $T_1<f(x)\leqslant T_2$,置为 1;如果 $T_2<f(x)\leqslant T_3$,置为 2;如果 $T_3<f(x)\leqslant 1$,置为 3,四值化为 0、1、2、3 序列。

(5)将上一步得到的序列按顺序排列成和图像大小相同的 $M\times N$ 的混沌阵列(混沌阵列不需要存储,其只是为了说明混沌序列中的每个数和图像像素点的对应关系才使用混沌阵列的概念)。

(6)记录混沌阵列中 0,1,2,3 的个数,设序列中 $i(i=0,1,2,3)$ 的个数为 n_i,构造 4 个 1 维数组,$a_0[n_0]$,$a_1[n_1]$,$a_2[n_2]$,$a_3[n_3]$(考虑到图像可能很大,如果在 VC 下实现,建议采用动态数组,因使用静态数组可能会导致缓冲区溢出),$a_i[n_i]$ 中记录混沌阵列中每个 i 在阵列列中的位置。例如 $a_0[100]=199$ 表示序列中第 100 个 0 是混沌阵列中第 199 个数。

3.2 加密过程

找到混沌阵列中每个 $i(i=0,1,2,3)$ 的位置,设为 (x_i,y_i),其对应的图像位置也为 (x_i,y_i),取出 (x_i,y_i) 点的像素值 p,对该像素值使用 $n_0,n_1,\cdots,n_{i-1},n_{i+1},\cdots,n_3$ 和 $d_0,d_1,\cdots,d_{i-1},d_{i+1},\cdots,d_3$ 作为密钥进行加密得到密文 p',即 $p'=p\wedge n_0\wedge n_1\wedge\cdots\wedge n_{i-1}\wedge n_{i+1}\wedge\cdots\wedge n_3\wedge d_0\wedge d_1\wedge\cdots\wedge d_{i-1}\wedge d_{i+1}\wedge\cdots\wedge d_3$(其中"∧"为异或运算),然后在混沌阵列中寻找与 (x_i,y_i) 相距 d_i 个 i 的位置 (x_i',y_i'),其对应图像位置也为 (x_i',y_i'),再将加密后的像素值 p' 作为 (x_i',y_i') 点的像素值。这样,每个像素值都被加密,加密处理由混沌阵列、4 个距离 $d_i(i=0,1,2,3)$ 决定,并且加密结果的存放位置也由混沌阵列、4 个距离 $d_i(i=0,1,2,3)$ 确定。

为描述方便,下面以 $M\times N$ 的 8 位灰度 BMP 图像为例来具体说明加密过程,其他图像加密过程与此类似。

(1)用户给定混沌初值 x_0、距离 d_0,d_1,d_2,d_3,其中 $0<d_0<n_0$,$0<d_1<n_1$,$0<d_2<n_2$,$0<d_3<n_3$,用 3.1 节的方法构造 4 个 1 维数组 $a_0[n_0]$、$a_1[n_1]$、$a_2[n_2]$ 和 $a_3[n_3]$。

(2)将 $a_0[n_0]$、$a_1[n_1]$、$a_2[n_2]$ 和 $a_3[n_3]$ 分别处理。首先处理 $a_0[n_0]$。对数组中每个元素的值,计算出对应的图像像素点位置。例如对 $a_0[i]$,其对应的图像像素点位置为 $([(a_0[i]-1)/M],(a_0[i]-1)\%M)$,这里 $[a_0[i]/M]$ 表示 $a_0[i]/M$ 的整数部分,"%"表示取模运算,并且为了描述的方便,约定图像和混沌阵列都是从第 0 行、第 0 列开始。设该点的像素值为 p,加密后的像素值为 p',加密运算为

$$p'=p\wedge(n_1\%256)\wedge(n_2\%256)\wedge(n_3\%256)\wedge(d_1\%256)\wedge(d_2\%256)\wedge(d_3\%256)$$

寻找和位置 $([(a_0[i]-1)/M],(a_0[i]-1)\%M)$ 在混沌阵列中距离为 d_0 个 0 的位置,得到位置为 $([(a_0[(i+d_0)\%n_0]-1)/M],(a_0[(i+d_0)\%n_0]-1)\%M)$(对 $i+d_0$ 做模 n_0 的运算是保证找到的位置不越界,并保证加密的像素值放置位置是一一对应且不会产生位置碰撞)。将 p' 作为图像中该位置对应点的像素值,即数组中 $a_0[(i+d_0)\%n_0]$ 对应的图像像素点的像素值。

(3)同样处理 $a_1[n_1]$、$a_2[n_2]$ 和 $a_3[n_3]$。只是在处理 $a_1[n_1]$ 时距离为 d_1 个 1,加密运算为

$$p'=p\wedge(n_0\%256)\wedge(n_2\%256)\wedge(n_3\%256)\wedge$$

$(d_0\%256)\wedge(d_2\%256)\wedge(d_3\%256)$

处理 $a_2[n_2]$ 时距离为 d_2 个 2,加密运算为

$$p'=p\wedge(n_0\%256)\wedge(n_1\%256)\wedge(n_3\%256)\wedge(d_0\%256)\wedge(d_1\%256)\wedge(d_3\%256)$$

处理 $a_3[n_3]$ 时距离 d_3 个 3,加密运算为

$$p'=p\wedge(n_0\%256)\wedge(n_1\%256)\wedge(n_2\%256)\wedge(d_0\%256)\wedge(d_1\%256)\wedge(d_2\%256)$$

3.3 解密过程

解密算法是加密算法的逆过程。找到混沌阵列中每个 i 对应的图像像素点位置,设为(x_i,y_i)。寻找混沌阵列中与(x_i,y_i)相隔 d_i 个 i 的对应点的图像位置(x_i',y_i'),取出(x_i',y_i')点的像素值,使用 n_0,$n_1,\cdots,n_{i-1},n_{i+1},\cdots,n_3$ 和 $d_0,d_1,\cdots,d_{i-1},d_{i+1},\cdots,d_3$ 作为密钥进行异或解密运算,然后将解密结果作为图像(x_i,y_i)点的像素值。

下面仍以 $M\times N$ 的 8 位灰度 BMP 图像为例来具体说明解密过程。

(1)同 3.2 节第 1 步。

(2)同加密处理一样,将 $a_0[n_0]$、$a_1[n_1]$、$a_2[n_2]$和 $a_3[n_3]$分别处理。首先处理 $a_0[n_0]$。对数组中的每个元素的值,计算出对应的图像像素点位置。例如对 $a_0[i]$,其对应的图像像素点位置为$([(a_0[i]-1)/M],(a_0[i]-1)\%M)$。寻找和位置$([(a_0[i]-1)/M],(a_0[i]-1)\%M)$在混沌阵列中对应点在混沌阵列中距离 d_0 个 0 的位置,得到位置为$([(a_0[(i+d_0)\%n_0]-1)/M],(a_0[(i+d_0)\%n_0]-1)\%M)$,其对应的图像像素位置点也为$([(a_0[(i+d_0)\%n_0]-1)/M],(a_0[(i+d_0)\%n_0]-1)\%M)$。取出该点的像素值,设该值是 p',进行解密运算。设解密后的像素值为 p,解密运算为

$$p=p'\wedge(n_1\%256)\wedge(n_2\%256)\wedge(n_3\%256)\wedge(d_1\%256)\wedge(d_2\%256)\wedge(d_3\%256)$$

将 p 作为图像中$([(a_0[i]-1)/M],(a_0[i]-1)\%M)$点的像素值,即数组中 $a_0[i]$对应的图像像素点的像素值。

(3)同理处理 $a_1[n_1]$、$a_2[n_2]$和 $a_3[n_3]$。只是在处理 $a_1[n_1]$时距离为 d_1 个 1,解密运算为

$$p=p'\wedge(n_0\%256)\wedge(n_2\%256)\wedge(n_3\%256)\wedge(d_0\%256)\wedge(d_2\%256)\wedge(d_3\%256)$$

处理 $a_2[n_2]$时距离为 d_2 个 2,解密运算为

$$p=p'\wedge(n_0\%256)\wedge(n_1\%256)\wedge(d_0\%256)\wedge(d_1\%256)\wedge(d_3\%256)$$

处理 $a_3[n_3]$时距离为 d_3 个 3,解密运算为

$$p=p'\wedge(n_0\%256)\wedge(n_1\%256)\wedge(n_2\%256)\wedge(d_0\%256)\wedge(d_1\%256)\wedge(d_2\%256)$$

可见本算法是对称加密算法,加密和解密使用相同的密钥,加/解密的运算也相同。也可以把解密处理当成加密、把加密处理当成解密来使用该算法。

4 仿真实验及结果

在 VC + +6.0 平台下编程仿真了本算法,并对不同大小的 8 位和 24 位 BMP 图像进行了大量的加解密实验。图 1 是以 Lena 图像为例的实验结果,其中图 1(a)为标准的 512 × 512 8 位 Lena 灰度图像,图 1(b)是取 $x_0=0.11$,$d_0=100$,$d_1=1\,000$,$d_2=10\,000$,$d_3=100\,000$ 的加密结果,图 1(c)是密钥正确的解密结果,本算法是无损对称加密算法,解密结果和原图没有任何变化。图 1(d)是取 $x_0=0.110\,000\,000\,000\,000\,1$,$d_0$、$d_1$、$d_2$ 和 d_3 不变,混沌初值有微小的 $1/10^{16}$变化时的解密结果。图 1(e)是取 $x_0=0.11$,$d_0=100\,000$,$d_1=10\,000$,$d_2=1\,000$,$d_3=100$,混沌初值不变,距离 d_i 变化时的解密结果。从图 1(d)可以看出,初值的微小变化,解密出的图像就是完全无意义的。从图 1(e)可以看出,距离的变化也会得出完全无意义的解密图像。

(a) Lena 原图　(b) 加密图像　(c) 解密后的 Lena 图像　(d) 混沌初值微小改变解密出的图像　(e) 混沌初值不变,d_0,d_1,d_2,d_3 解密出的图像

图 1　实验结果 1

Fig. 1　Experimental results 1

5 算法分析

5.1 算法时间复杂度分析

本算法中,生成混沌序列需要迭代 $M \times N$ 次,若问题的规模为 m,则生成混沌序列的时间复杂度为 $O(m)$。对所有像素的加解密并置乱的时间开销在第 3 节已经分析过,时间复杂度也为 $O(m)$。因此,整个算法的时间复杂度为 $O(m)$,为线性阶的时间复杂度,时间复杂度较低。

为检测算法的时间开销,对不同大小的 8 位和 24 位 BMP 位图进行了大量的加解密实验。实验所采用的硬件系统是 Pentium4 1.8G CPU,192M DDR 内存;软件系统为 Windows2000 操作系统,VC++6.0 编程平台。在实验中,对数据大小为 3.93M 的 1 344×1 024 的 24 位 BMP 图像加密所用的时间约为 0.531s,解密所用的时间约为 0.551s,加密速度约为 7.4M/s,解密速度约为 7.1M/s。可见算法的效率是较高的。

5.2 算法空间复杂度分析

算法中主要的空间开销是保存混沌序列所使用的 4 个 1 维数组。它们总的大小等于图像像素数。如果待加密图像大小为 $M \times N$,采用 int 型的数组,则它们总的大小为 $M \times N$ 个 int 型的空间。可见,空间开销只与图像大小有关,与图像位数无关。例如,对一幅 1 024×768 大小的图像,其空间开销约为 $1024 \times 768 \times 4/(1024 \times 1024) \approx 3$M。若问题规模为 m,则算法的空间复杂度为 $O(m)$,为线性阶的空间复杂度。

5.3 算法安全性分析

算法采用混沌系统来生成混沌序列,由于混沌系统生成的序列有较好的随机性,对初值极其敏感,所以算法的安全性较好。在仿真实验中可以看到,即便初值只有 $1/10^{16}$ 的变化,得出的混沌序列就完全不同,导致解密的结果也完全无意义。又对像素值加解密运算是通过将像素值和混沌阵列中每个值的数量以及 $d_i(i=0,1,2,3)$ 进行异或操作,并且像素值的置换位置是由随机数 d_i 和混沌阵列共同确定,即便是知道 d_i,如果不知道混沌阵列,也不能确定置换位置;反过来,知道混沌序列,但是不知道 d_i,也不能确定置换位置,解密不出原始图像,这在仿真实验中得到了证实。因此要正确解密出图像,必须同时得到正确的混沌初值和距离 d_i。

下面分析密钥空间。

为了得到较好的加密效果和较大的置乱二阶距[8,9],推荐取较大的距离。对不同大小的图像,设定阈值 T_d,取 $d_i \geq T_d$。不同大小的图像,T_d 值不同。经过大量的实验发现,对于 $M \times N$ 的图像,通常取 $T_d = (M+N)/10$ 就能得到较好的效果。考虑到混沌序列的均匀性,混沌阵列中每个值的个数大致相等,处理 $a_i[n_i]$ 时,如果 d_i 大于 n_i,则相当于实际的距离为 $d_i \bmod n_i$。所以 d_i 的密钥空间约为 $(\min\{d_0,d_1,d_2,d_3\}-T_d)$。从理论上说,混沌对初值是极度敏感的,但是由于在计算机上实现时受计算机精度的影响,所以混沌系统对初值的敏感是有一定限度的。本文所采用的混沌系统对初值的敏感程度可达到小数点后 16 位,实验也证实当初值有 $1/10^{16}$ 的微小变化时得出的解密图像就完全不同,但是实验结果又表明,当初值变化小于 $1/10^{16}$ 时,解密出的图像就是原始图像了,所以该混沌初值的密钥空间约为 10^{16}。因此算法的密钥空间约为 $[(\min\{d_0,d_1,d_2,d_3\}-T_d)]^4 \times 10^{16}$。例如,对于 512×512 的图像来说,密钥空间约为 $[(512\times512)/4-(512+512)/10]^4 \times 10^{16} \approx 1.8 \times 10^{29}$,密钥空间是较大的,足以满足实际的需求。并且随着计算机精度的提高,算法的密钥空间还将大大扩展。

从上面的分析和实验结果可见,该算法的安全性是较高的。

5.4 算法加密效果分析

从仿真实验部分的加密及密钥不正确的解密图像可以看出该算法的加密效果较好。为了对加密效果有一个较直观的评价,下面给出 Lena 原图、加密图及其密钥不正确解密图的灰度直方图如 2 所示。

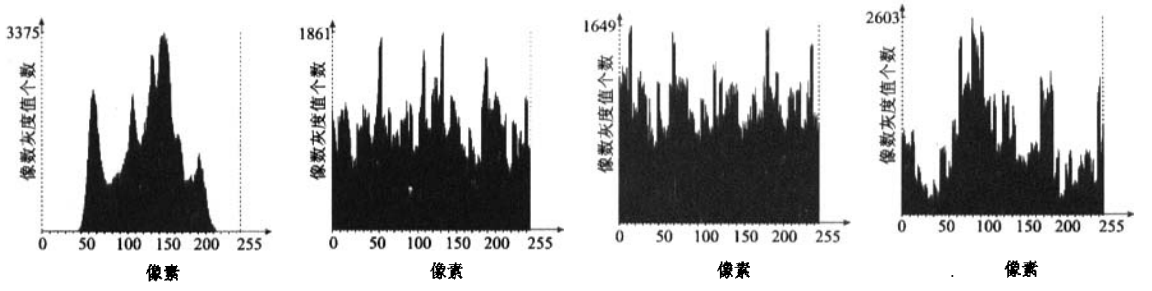
从图 2 可以看出,加密后的图像与原图的灰度直方图有较大变化,密钥不正确解密出的图像灰度直方图也与原图的灰度直方图有较大的变化。可见,该算法的加密效果较好。从密钥错误解密图像的灰度直方图还可以一定程度上看出算法的安全性较好。

5.5 算法的置乱网络性能分析

单纯考虑置乱网络的性能,采用文献[4]、[5]中所使用的图像置乱网络性能指标来评价算法中所提出的置乱网络的性能。

(1)时间特性

对于一个 2 维 $M \times N$ 像素的图像,L 为生成加



(a) Lena 原图的灰度直方图　(b) 加密后的灰度直方图　(c) 混沌初值变化 $1/10^{16}$ 解密图像的灰度直方图　(d) 距离 d_0, d_1, d_2, d_3 变化解密图像的灰度直方图

图 2　实验结果 2

Fig. 2　Experimental results 2

密图像全部像素点坐标混沌映射的迭代总次数,显然迭代次数 L 越小,算法的运算速度越快。本算法中,对 $M \times N$ 的图像,混沌迭代的次数为 $M \times N$ 次,远小于文献[4]和[5]中的迭代次数。

(2)不动点

如果原图像像素点经过置乱网络置乱后,像素点的行列坐标值没有发生变化,则称此像素点为不动点。若不动点的数目 k 越少,则说明置乱的效果越好,保密性也就越高。文献[4]的不动点比例约为 0.1%,文献[5]中的不动点比例为 0.1 ~ 0.4%。而本文的置乱网络中,不动点的比例为 0%。

(3)自然序

如果相邻的像素点,其置乱后的行列地址虽然都发生了变化,但仍然相邻,称之为自然序。自然序包括行自然序、列自然序、斜自然序、圆自然序。若置乱后图像的自然序越少,则置乱的效果越好,保密性也就越高。

对 32 × 32 的图像取随机的混沌初值和距离进行加密,然后统计自然序。例如,使用 $x_0 = 0.11$、$d_0 = d_1 = d_2 = d_3 = 100$进行加密,得到自然序的数量为 75,比例为 75/1 024 ≈ 7.3%。文献[4]的自然序比例约为 6%,文献[5]的比例约为 9%。可见本文的自然序的性能和文献[4]和[5]是相当的。

(4) 二阶距

$$\tau^{(2)} = \sum_{I=1}^{M}\sum_{J=1}^{N}((I-i)^2 + (J-j)^2) \qquad (3)$$

式中,I, J 为原图像像素点的坐标,i, j 为经置乱后加密图像像素点的坐标,若二阶距 $\tau^{(2)}$ 越大,则说明经置乱后像素点总体的位移越大,即与原图像越不相关,置乱效果越好。

根据式(3),本算法的二阶距计算公式为

$$\begin{aligned}\tau^{(2)} = &\sum_{i=1}^{n_0}(([[a_0[i]-1)/M] - [(a_0[(i+d_0)\%n_0]-1)/M])^2 + \\ &((a_0[i]-1)\%M - (a_0[(i+d_0)\%n_0]-1)\%M)^2) + \\ &\sum_{i=1}^{n_1}(([[a_1[i]-1)/M] - [(a_1[(i+d_1)\%n_1]-1)/M])^2 + \\ &((a_1[i]-1)\%M - (a_1[(i+d_1)\%n_1]-1)\%M)^2) + \\ &\sum_{i=1}^{n_2}(([[a_2[i]-1)/M] - [(a_2[(i+d_2)\%n_2]-1)/M])^2 + \\ &((a_2[i]-1)\%M - (a_2[(i+d_2)\%n_2]-1)\%M)^2) + \\ &\sum_{i=1}^{n_3}(([[a_3[i]-1)/M] - [(a_3[(i+d_3)\%n_3]-1)/M])^2 + \\ &((a_3[i]-1)\%M - (a_3[(i+d_3)\%n_3]-1)\%M)^2) \qquad (4)\end{aligned}$$

对 32 × 32 的图像取随机的混沌初值和距离进行加密,然后利用式(4)计算其二阶距。例如,使用 $x_0 = 0.11$、$d_0 = d_1 = d_2 = d_3 = 100$,得到 $\tau^{(2)} = 430\,696$。文献[4]的平均二阶距为 348 665,文献[5]的平均二阶距为 348 635。可见本文置乱网络二阶距的性能比文献[4]和[5]稍好。

综合考虑算法的时间复杂度、空间复杂度、安全性、加密效果及其置乱网络的性能,可以看出本文提出的算法综合性能是较好的。

6　结　论

本文提出了一种基于混沌阵列的数字图像加密算法。它采用具有较好动力学特性的混沌系统来产生和图像像素值数量相同的 4 值混沌序列,用 4 个 1 维数组分别记录 4 值化后的值:0、1、2、3 的位置,然后对这 4 个数组对应的像素分别进行加密和置换操作。对像素值的加密与混沌序列中 0、1、2、3 的数

量相关;对位置的置换由混沌序列中 0、1、2、3 的分布情况及其用户设定的置换距离 d_0, d_1, d_2, d_3 来决定。分析和实验结果表明,算法的时间和空间复杂度较低、加解密速度较快、密钥空间较大。即便是攻击者知道加密所采用的混沌系统和具体的加密算法,但只要不知道完整的密钥,即混沌系统的初值和置乱距离 d_0, d_1, d_2, d_3,想要解密出原始图像也是比较困难的。

参考文献(References)

1 Li C G, Han Z Z, Zhang H R. Image encryption techniques: a survey[J]. Journal of Computer Research and Development, 2002, **39**(10):1317 ~ 1324. [李昌刚,韩正之,张浩然. 图像加密技术综述[J]. 计算机研究与发展, 2002,**39**(10): 1317 ~ 1324.]

2 Li C G, Han Z Z, The new evolution of image encryption techniques [J]. Information and Control, 2003,**32**(4): 339 ~ 343. [李昌刚, 韩正之. 图像加密技术新进展[J]. 信息与控制, 2003,**32**(4): 339 ~ 343.]

3 Dedieu H, Ogorzalek M J. Indentifiability and identification of chaotic systems based on adaptive synchronization [J]. IEEE Transactions on Cirruits and Systems- I :1997,**44**(10):948 ~ 962.

4 Qin H L, Hao Y L, Sun F. Design of picture permutation network on chaos[J]. Computer Engineering and Applications, 2002,**38**(7): 104 ~ 106. [秦红磊,郝燕玲,孙枫. 一种基于混沌的图像置乱网络的设计[J]. 计算机工程与应用, 2002,**38**(7):104 ~ 106.]

5 Liu Y J, Liu X D, Wang G X. The design of a class of modified chaotic picture scrambling networks [J]. Journal of Image and Graphics, 2004,**9**(3): 360 ~ 364. [刘云江,刘向东,王光兴. 一类改进型基于混沌的图像置乱网络设计[J]. 中国图象图形学报, 2004,**9**(3): 360 ~ 364.]

6 Li Shu-jun. Analyses and New Design of Digital Chaotic Ciphers [D]. Xi'an: Xi'an Jiaotong University, 2003. [李树均. 数字化混沌密码的分析与设计[D]. 西安:西安交通大学,2003.]

7 Li S J, Mou X, Cai Y. Pseudo-random bit generator based on couple chaotic systems and its application in stream-ciphers cryptography [A]. In: Progress in Cryptology-INDOCRYPT 2001: Second International Conference on Cryptology[C], Chennai, India, 2001, LNCS **2247**: 316 ~ 329.

8 Li S J, Mou X, Cai Y, *et al*. On the security of a chaotic encryption scheme: Problems with computerized chaos in finite computing precision[J]. Computer Physics Communications, 2003, **153**(1): 52 ~ 58.

9 Liu Bin, Zhang Yong-qiang, Liu Fen-lin. A new scheme on perturbing digital chaotic systems[J]. Computer Science, 2005,**32**(4):71 ~ 74. [刘镔,张永强,刘粉林. 一种新的数字化混沌扰动方案[J]. 计算机科学, 2005,**32**(4): 71 ~ 74.]