

**课 程 报 告**

**课程名称： 信息隐藏课程报告**

**专业班级： 信息安全201703**

**学 号： U201714513**

**姓 名： 徐立威**

**指导教师： 马晓静**

**报告日期： 2020年 5月 5 日**

**网络空间安全学院**

目录

[基于均匀嵌入策略的实时可逆图像认证方法 1](#_Toc22968)

[1. 选题 1](#_Toc23786)

[2. 背景 2](#_Toc17881)

[2.1. 时代背景 2](#_Toc11948)

[2.2. 技术背景 2](#_Toc22465)

[2.2.1. 主动验证技术 2](#_Toc1673)

[2.2.2. 被动验证技术 2](#_Toc15880)

[2.2.3. 现行研究方向——可逆图像认证 2](#_Toc17969)

[2.2.4. 论文研究方向 3](#_Toc29493)

[3. IPVO和RIA技术 4](#_Toc26460)

[3.1. Peng et al.的IPVO技术 4](#_Toc30272)

[3.1.1. 高位嵌入 4](#_Toc12476)

[3.1.2. 低位嵌入 5](#_Toc4165)

[3.1.3. 提取与恢复 5](#_Toc14178)

[3.2. Hong et al.的RIA技术 6](#_Toc22734)

[3.2.1. 图像分块 6](#_Toc14445)

[3.2.2. UB与EB块的分类 6](#_Toc30039)

[3.2.3. 身份验证码的嵌入 6](#_Toc29942)

[3.2.4. 提取认证 7](#_Toc13812)

[4. 改进的方案 8](#_Toc19094)

[4.1. 身份验证码的嵌入 9](#_Toc78)

[4.1.1. 步骤1：用适当的块模式划分图像 9](#_Toc21761)

[4.1.2. 步骤2：将AC码嵌入到UB块 11](#_Toc17223)

[4.1.3. 步骤3：将AC码和AI嵌入到EB块 11](#_Toc2437)

[4.1.4. 越值问题的解决 12](#_Toc16651)

[4.2. 图像身份验证和图像恢复 14](#_Toc877)

[4.2.1. UB块的提取 15](#_Toc10838)

[4.2.2. EB块的提取 15](#_Toc12445)

[5. 评价与思考 15](#_Toc7475)

# 基于均匀嵌入策略的实时可逆图像认证方法

# 选题

论文题：A real-time reversible image authentication method using uniform embedding strategy

出版刊物：IEEE Transactions on Computer-Aided Design of Integrated Circuits And System

作者：Heng Yao、Hongbin Wei、Chuan Qin、Zhenjun Tang

# 背景

## 时代背景

互联网与云计算的发展给我们的生活带来了诸多便利，但与此同时，在这样的大互联网环境下，隐私泄露变得愈加容易，所以如何保护个人隐私问题也成为当前最为关注的事情。

本篇研究的论文主要针对的就是多媒体应用上的隐私保护问题。如何保护图像不被人篡改和破坏以及如何验证该图像是否已经被人进行修改，是论文研究的探讨方向。

## 技术背景

目前，图像的认证主要有两种方法：主动验证和被动验证。

### 主动验证技术

主动验证是指提前的把身份认证码（authentication code）嵌入到图像当中，身份认证码（authentication code）也可简称为AC码，一旦有攻击者篡改了其中的内容，就可以通过提取图像当中的AC码来验证，这种方法也被称为脆弱水印技术。

### 被动验证技术

被动验证技术则不像主动验证那样提前的把AC码嵌入到信息当中，而是根据图像中的周期性重采样统计特征、噪声特征、计算机生成图像特征和JPEG块状伪影特征来进行数字统计从而识别。这些特征识别法可以在相关论文中找到验证原理，但并不是本文的重点，所以只是略微提及。

这种方法也称为数字图像鉴证技术，是根据图像的数字特征来鉴别图像是否被修改。

### 现行研究方向——可逆图像认证

虽然被动方法可以涵盖更实际的情况，但它们的检出率远低于主动验证方法。 因此，为了获得满意的验证结果，本文将从主动验证出发，在主动验证的基础上进一步的研究。

近年来，可逆图像认证（reversible image authentication，简称RIA技术）作为一种主动技术，已成为研究的热点。

RIA技术可以看作是可逆数据隐藏(reversible data hiding 简称RDH)和脆弱水印技术的有利组合。 然而，不像普通的脆弱水印技术，普通的脆弱水印技术不能恢复原始图像，具有着一定的局限性。RIA改进的地方是可以恢复图像，并且没有任何损失。

另外，和RDH不同的地方是，RIA还能识别具有中等改动的图像，也就是说识别功能更为强大。因为一些AC码必须以可逆的方式嵌入到图像当中，所以现在的RIA还主要应用RDH策略。

目前有许多的手段可以将信息可逆的嵌入到图片当中，例如自适应预测-误差扩展( adaptive prediction-error expansion 简称PEE)，成对PEE，多重直方图修饰 PEE，直方图移位，差异扩展，像素值排序( pixel value ordering 简称PVO)等技术。然而，为了验证图像的每个局部区域，AC嵌入应该以块实现，因此，大多数RIA是使用PVO方案独立地嵌入到每个块中。

### 论文研究方向

在论文中，研究人员提出了一种使用均匀嵌入策略的有效RIA方法，其中具有相同信息量的AC被嵌入到每个块中，这种方法有利于加快图像的认证速度。此外，还会通过对所有嵌入容量来寻求最佳块的大小，并且块的大小通常比其他的方法小。

# IPVO和RIA技术

研究人员先是参考了 Peng et al.和Hong et al.的研究，在他们的基础上进行改进，那么我也从两人的研究出发。

## Peng et al.的IPVO技术

2014年，Peng et al.提出了改进PVO技术，这个技术是通过修改最高位值和次高位的值实现的。 与传统的PVO方法相比，主要的改进是这些值的位置的考虑有所不同。

假设一个灰度图I的大小是M x N，最开始，我们先将灰度图分成相同大小的子快Bi，i=1,2,3,....,NB，这里的NB表示的就是分成块的个数。

另外设每个子块的总像素为P，{B1i，B2i，...，BPi}是子块Bi中的像素值。像素按升序排序，然后重新排列的序列表示为{Bσ（1）i，Bσ（2）i，...，Bσ(P)i}。其中上面的σ（x）是个关系映射，x表示升序排序中的顺序位，而映射的是原来的位置。

然后，嵌入过程可以沿着两个方向进行：高位嵌入和低位嵌入。两个方向都有着相似的原则，具体步骤如下：

### 高位嵌入

之前我们做好了像素的升序排序数组{Bσ（1）i，Bσ（2）i，...，Bσ(P)i}，取最高的两个像素位Bσ(P)i和Bσ(P-1)i，并将它们之间的差值定义为dmax=Bσ(P)i-Bσ(P-1)i。

接下来就可以进行嵌入操作，嵌入所使用的函数如图3.1所示。

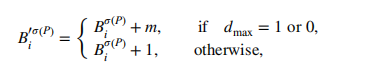


图3.1

其中m表示嵌入信息比特，也就是说如果如果高位两个差值大小小于等于1，则嵌入信息。

### 低位嵌入

同样的，低位嵌入与高位嵌入道理相同。取最低的两个像素位Bσ(1)i和Bσ(2)i，两个最低值之间的差异定义为dmin=Bσ(2)i-Bσ(1)i。根据dmin的数值，对新的Bσ(1)i进行图3.2不同的运算操作。

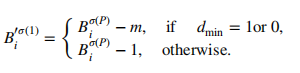


图3.2

如果dmin的值为0或1，则新的最低位等于最高位减去m，否则就是最高位减去1。至此，信息比特位已沿两个方向嵌入到像素当中。

### 提取与恢复

由于高位和低位两个方向的嵌入只是对像素值B进行修改，所以σ（x）的映射关系并没有被修改。

提取信息的过程：和嵌入信息的过程类似，我们先用Bi’表示嵌入后的{B1i，B2i，...，BPi}序列，按照升序排列形成{Bσ（1）i，Bσ（2）i，...，Bσ(P)i}，并求出dmax和dmin。公式如图3.3所示。

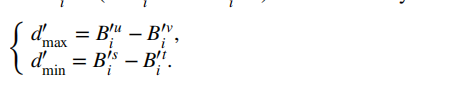


图3.3

其中u,v代表着最高两位，P和P-1，s，t代表最低两位2，1。如果dmax=1或0（或者dmin=1或0），就提取一个bit“0”出来；如果dmax=2或-1（或者dmin=2或-1），就提取出来一个bit“1”出来；其他的数值就表示之前并没有嵌入信息。

提取完信息之后，就进行图像的回复。也就是3.1.1和3.1.2的逆运算，如图3.4。

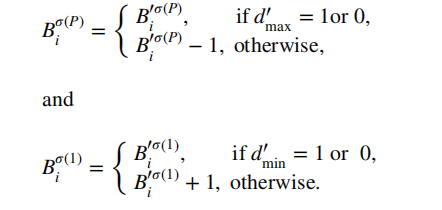


图3.4

## Hong et al.的RIA技术

以前的RIA方法主要将AC码嵌入到具有可逆嵌入数据能力（embeddable blocks简称EB）的块中，而对于其他块，没有嵌入额外的AC码。因此，那些没有嵌入信息的块(unembeddable blocks简称UB)缺乏内容保护。

为了克服这种缺陷，Hong et al.提出了一种改进的图像认证方法，可以将AC码嵌入到所有的块当中，包括EB和UB。具体方法如下所示。

### 图像分块

首先，图像I被划分为4 x 4个块，然后每个块被进一步划分为4个2 x 2个子块。对于每个子块，会根据IPVO方法寻找可嵌入的位置，IPVO算法已经在3.1进行了阐述。每个子块得到能嵌入比特位的最大值，并以此作为子快可嵌入数据的大小。

### UB与EB块的分类

前面已经确定好了每个4 x 4块的可嵌入信息的总容量，接下来所有的2 x 2块要会根据3.1计算好的容量进行分类，分成UB块和EB块。

具体的来说，如果该容量大小为0，就分为UB块或其余块；如果容量为1-7（bit），就定义为EB块。

### 身份验证码的嵌入

AC码可以从使用了哈希或者LSB替换的EB块中提取出来，但是使用LSB替换算法嵌入的EB块中的是不可逆的过程，所以必须还嵌入辅助信息（auxiliary information 简称AI）到EB当中。

假设一个EB块的容量为U，为了验证这个块，就需要从这个EB块中提取U−1位哈希值来生成验证位。另外，为了恢复每个UB块，利用LSB与U−1个认证位进行连接，以生成U位待嵌入的信息，这些空间将会利用IPVO算法进行嵌入。

简单的理解就是，为了把UB和EB块的空间都利用到，就综合利用LSB替换算法和IPVO算法。其中，LSB嵌入到UB块的第一像素；EB块则是利用IPVO嵌入U-1位信息。

### 提取认证

在图像认证阶段，待验证的图像I被分成4 x 4的大块，就像3.2.1分块一样，再分成4个2 x 2的小块，然后确定每个小块的总嵌入容量。

如果一个块容量为0，则提取第一像素的LSB作为AC码，如果该块是嵌入可逆位，则可以通过逆IPVO方法提取这些位。

最后，用生成的哈希码对所有AC码进行验证。如果验证结果都与哈希码匹配，则图像将被识别为真实的；如果有差异，伪造区域可以通过不同的分析来对伪造位置进行定位。

如果图像被证明是真实的，为了恢复图像，可以从EB的嵌入位中提取来自UB的原始LSB进行恢复。

# 改进的方案

在Hong et al.的方案当中，AC码嵌入到UB和EB块是不平衡的。具体来说，对于UB，只有一位可以用于身份验证，而对于EB使用了U−1位。也就是说，EB的验证能力明显大于UB的验证能力，利用率高的多。

然而，值得注意的是，图像每个块的局部纹理复杂度通常与其嵌入容量成反比。但是，在大多数情况下，认证比特率低的复杂区域更有可能支持恶意篡改攻击。因此，为了解决这个问题，研究者提出了一种改进的具有统一身份验证能力的RIA方法，将相同数量的AC码嵌入到每个块中用于一致的内容保护。

与所有现有的RIA方法类似，研究者改进的方法将分为两个过程：

1. 身份验证代码嵌入；
2. 图像身份验证和图像恢复。

由于后面会有许多新的符号，为了阐述方便就先把对应关系以图4.1展现给大家。

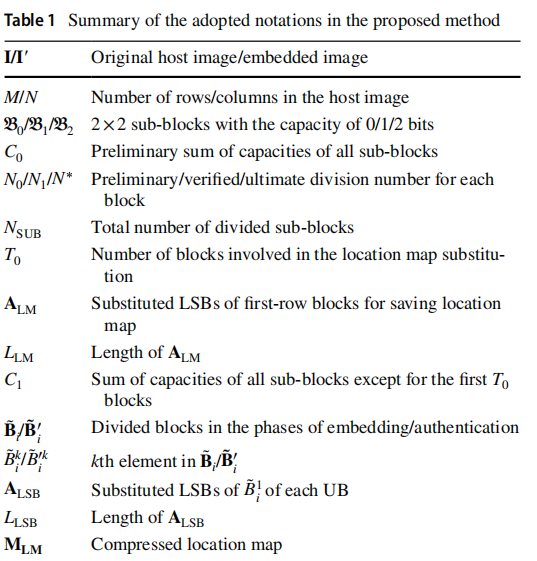


图4.1 符号表示

## 身份验证码的嵌入

为了将身份验证代码嵌入到可逆图像中，该过程包括三个步骤：图像分块、将AC码嵌入UB块、将AC码嵌入到EB块，这和3.2中的RIA步骤是相同的。

### 步骤1：用适当的块模式划分图像

与3.2.1的图像分块有所不同，在改进的方法当中，每个图像将使用动态分割策略被分割成块。

首先，将M×N大小的图像I分为非重叠的2 x 2大小的子块。

然后，根据IPVO方法评估每个子块的嵌入容量，每个子块的可能嵌入容量为0、1和2位，这里用B0,B1,B2表示。

接下来，我们使用容量数来确定整个图像的初步容量，表示为C0。基于C0，一个认证块中的子块数（用N0表示）可以通过图4.2中的公式来计算， NSUB是被分割的子块的总数。

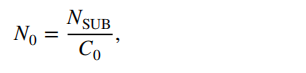


图4.2

N0值的分布从1-6，其概率分布图可以如图4.3所示。

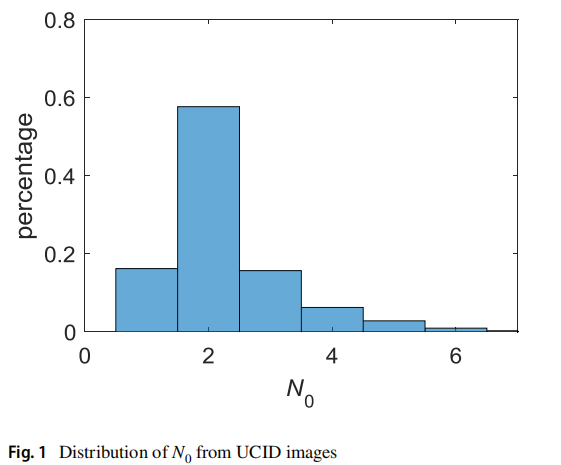


图4.3

基于N0的6种情况，设计了6种块的存储方式，如图4.4所示。

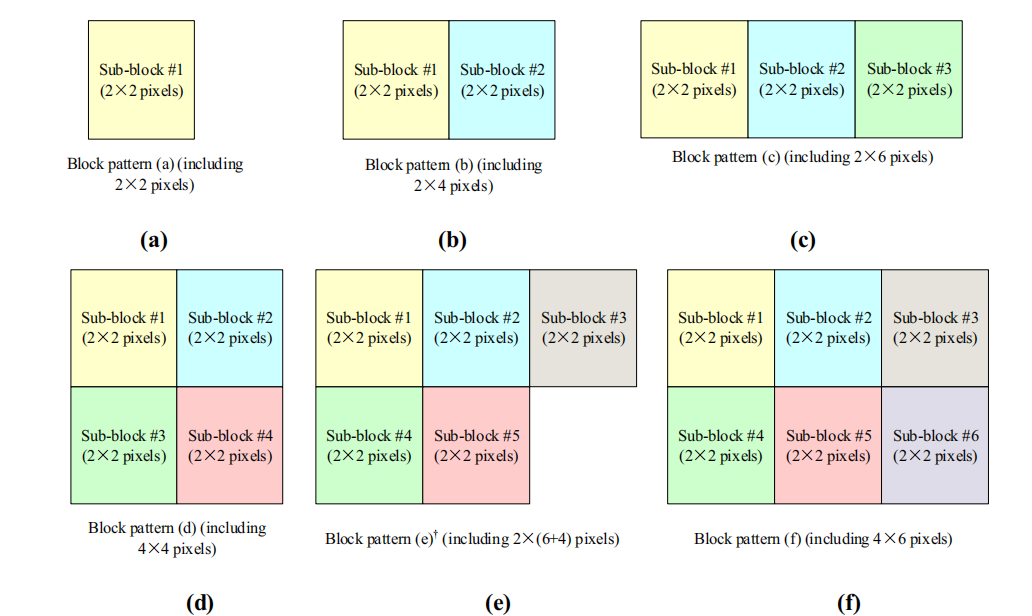


图4.4

由于块分配的不精确和AI替代等原因，子块数N0不是最终的分割数，这个数值不怎么的准确。那先暂时用N\*来表示最终精确的分割数目，这个数值将通过比较初步计算的N0及其修正数值N1共同确定。N1这个数值将会分析块过度计算和过少计算的原因来得到。

具体地来讲，首先按照初步的模式将图像分成块，然后分析每个块相对于过度计算和过少计算的特征。对于每个块，我们评估每个子块的容量，并对它们进行求和，以获得块的嵌入容量。应该注意的是，对于每个块，如果一个子块有一个不足或过度，整个块都应该被视为一个问题块，容量设置为零。对于有比特流问题的块，我们记录它们的位置以生成一个下比特流位置映射，并使用无损熵方法压缩该映射，例如算术编码。在生成压缩位置映射后，我们将块替换为预先定义的第一行块的LSB。假设有T0个块参与了替换，并将LSB替换定义为长度为LLM的ALM。接下来，我们确定除第一排T0块外的所有块的容量，并将其表示为C1。然后可以按照图4.5的公式来计算出N1。

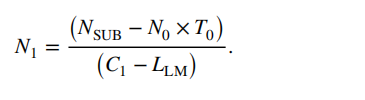


图4.5

在N1=N0的情况下，N0被视为最终验证除法数N\*；否则，N0将增加1，直到重新验证时N1等于N0，在这种情况下，增加后的N0被视为N\*。

### 步骤2：将AC码嵌入到UB块

在一般的AC嵌入过程中，图像被分割成具有适当参数N\*的块，其中0 bit空间的块嵌入信息都是无效的，这就造成了一定的空间浪费。

然而，本研究中，研究者的目标是每个块具有统一的身份验证能力，其中每个块都会嵌入相同数量的AC信息。为了实现目标，首先用不可逆的LSB替换将ACs嵌入到UB块中，然后将之前这些替换的LSB值以可逆地方式保存嵌入到UB块当中。

为了将AC码嵌入到UB块中，需要通过IPVO调节，将UB中每个子块的最大值和最小值进行扩展。将扩展块表示为Bi，其相应像素为{Bji，j=1，2，...，4×N\*}。接下来，我们使用MD5散列像素{B2i，B3i，...，B4×N\*i}(即块中除B1i外的所有像素)来生成128位哈希值，然后使用数字“1”在MD5中的奇偶性，指示AC的散列序列。 如果数字是奇数，则是1；如果是偶数，则是0。接下来，通过简单的LSB替换，将B1i的LSB替换为AC。

需要注意的是,要把LSB替换的值以嵌入的形式保存，方便图像的恢复。所以， 所有UB的B1i的保存LSB集表示为ALSB={A1LSB，A2LSB，...，ALLSB LSB}，其中LLSB是ALSB的长度。

### 步骤3：将AC码和AI嵌入到EB块

对于EB块，我们将一位AC嵌入到相应的块当中。但是，除了AC码外，还需要嵌入AI，包括生成的ALSB和ALM。因此，对于容量大于1的块，分配更多的AI位以使整个过程可逆，详细的嵌入过程在3.1已经说明。

为了便于理解，嵌入过程的流程图如图4.6所示。

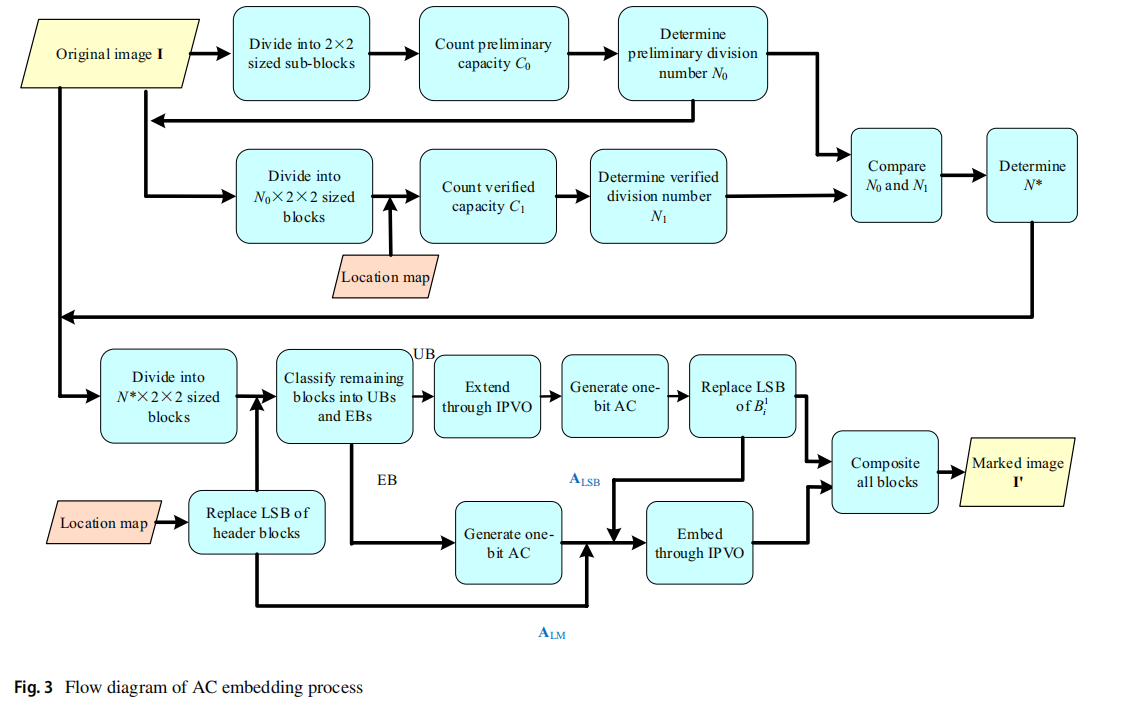


图4.6

### 越值问题的解决

对于基于IPVO的RDH方法，子块中的大多数最大/最小像素对被扩展为具有较高/较低趋势的一个。因此，一旦原始像素值为255或0，就会有值超过允许范围的可能。因为在IPVO中会进行+、-操作。

为了避免这个问题，255或0的像素值保持不变，同时记录它们所属块的位置，以避免与254和1的像素值混淆。

由于LSB替换操作是在UB的第一像素B1i的AC嵌入过程中进行的，一些原本是UB的块可能会被错误地判断为EB块。为了解决这一问题，将每个UB的第一子块的嵌入公式改为图4.7中的公式。

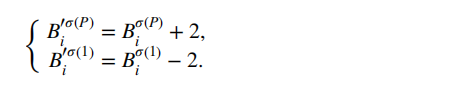


图4.7

如果是这样的话，254或1的像素值也可能导致越值现象。比如1-2=-1<0, 254+2>255。因此，对于UB的第一子块，0、1、254和255的像素值保持不变，并且它们的像素值不变，它们所属块的相应位置保存为位置映射。

然后将位置映射压缩并表示为MLM。 然后，我们通过LSB替换在第一排块中将MLM嵌入到图像I中。请注意，仍然存在这样的可能性：比如块的第一排的像素值已被篡改，然后一旦错误地提取MLM，剩余的AC码也将被影响。为了判断头块在传输过程中是否被篡改，每个标头的第一二个LSB 将被用于为身份验证位，这些位是从散列序列{B3i，B4i，...，B4×N\*i}派生的。如图4.8所示。

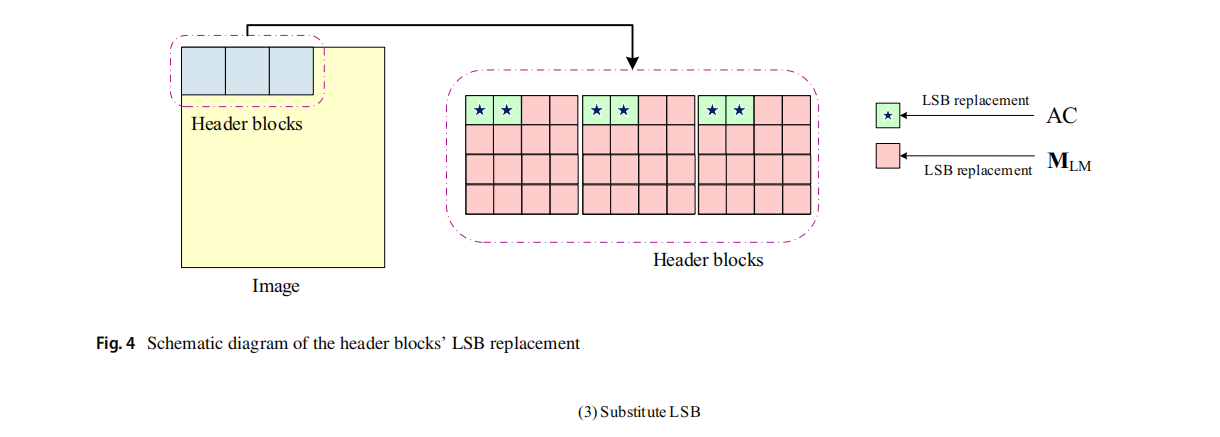


图4.8

下面是对一个图像的实际转换过程，如图4.9。

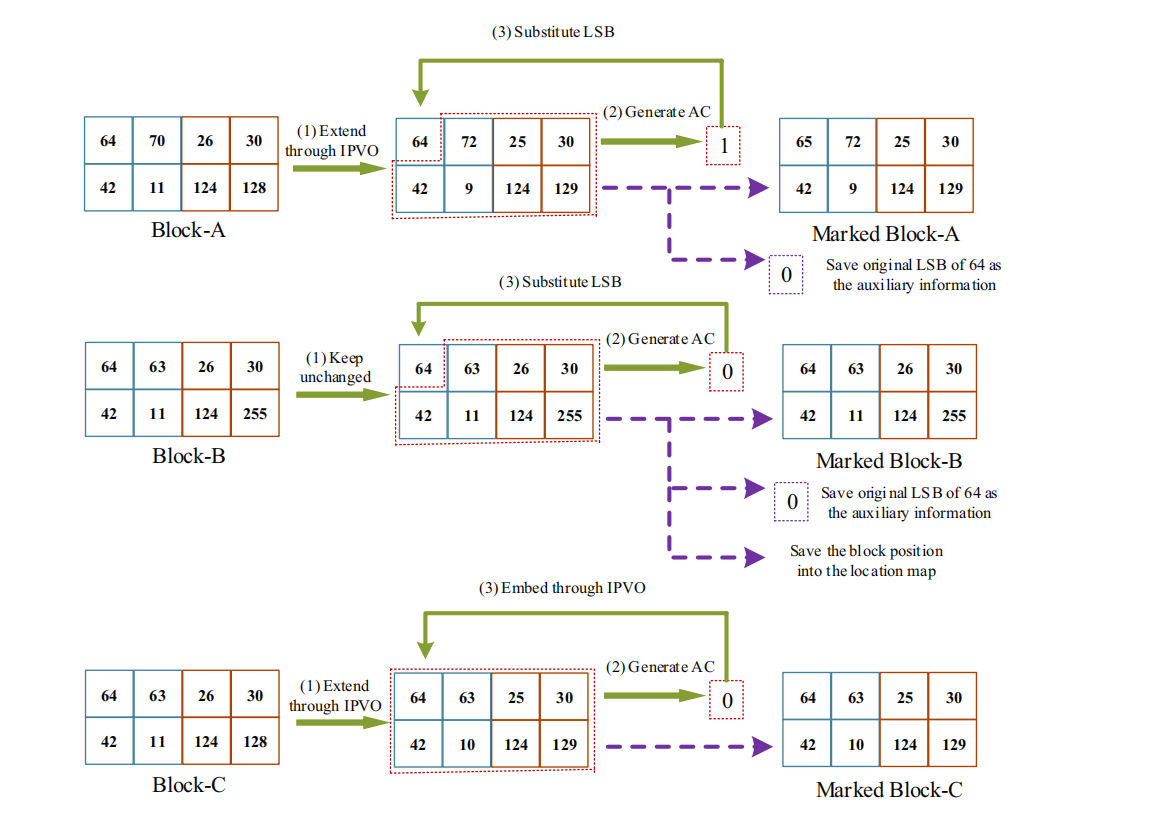


图4.9

## 图像身份验证和图像恢复

在图像身份验证和图像恢复过程中，我们假设要身份验证的图像是I。我们首先提取N∗位信息，然后用相应的模式将图像分割成块，如图4.10所示。

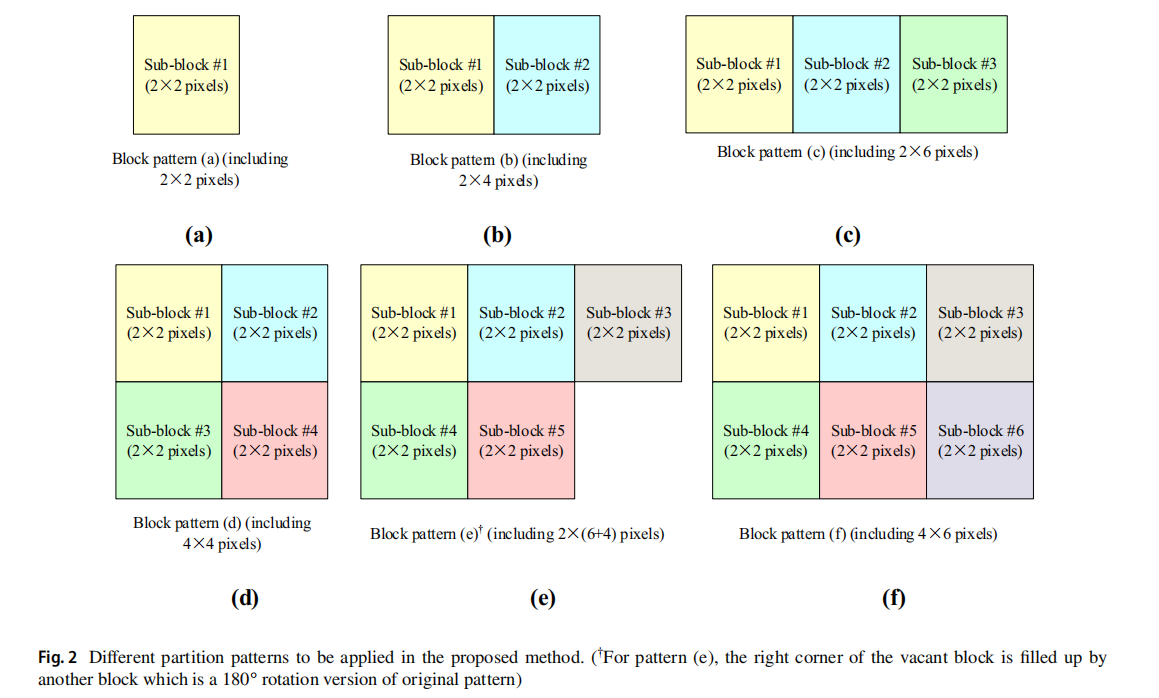


图4.10

接下来，我们从第一行块中提取MLM。具体而言，我们首先提取位置映射的长度，然后提取相应的长度位进行位置映射恢复。在位置映射MLM恢复后，具有越值问题的块被指示并标记为不变的块。接下来，根据每个块的嵌入容量，剩余的块被指定为UB或EB。 为了验证整个图像的身份验证，在所提出的方法中，我们分析了UB，其次是EB。

### UB块的提取

AC数据流是通过MD5将像素哈希化生成的，所以我们只需计数哈希序列中数字“1”的奇偶性，它在数据隐藏过程中遵循了相同的AC生成方法，这样就可以获得当前的AC码。 同时，通过提取B1i的LSB，可以得到原始AC。通过比较当前和原始AC码，就可以进行UB块中的的身份验证。

### EB块的提取

在验证UB之后，使用EB块进一步验证图像的身份验证。对于每个EBi，当前AC码是通过应用MD5方法来对像素hash化{B1i，B2i，...，B4×N∗i}，计数哈希序列中数字“1”的奇偶性来生成当前AC码。然后使用IPVO技术获得原始的AC码。

每个EB块可以进一步分为N\*个2×2个子块，并且可以从所有子块中提取至少一个位。在这里，我们认为第一提取的位是块的原始AC码。类似于UB，通过比较当前和原始AC码可以进行EB块的身份验证。

# 评价与思考

在本学期的学习当中，我们在图像方面主要学习的是图像嵌入，学习LSB替换、JPEG、F3、F4、F5算法，但这些算法都有个共同的特点就是，我只管嵌入信息不求怎么恢复这个图像。而论文的方向是在嵌入信息的同时，还能恢复过来，并在前人的基础上分析改进。

其中IPVO的借助差值在高低位嵌入的思路和我之前所学大有不同，个人感觉这种方法也算有一定的误导作用，如果别人单纯的以为是LSB替换，是无法获得正确的信息的。因为灰度像素值的范围是0-255，如果按照原来的公式计算，就有可能产生-1，256这种非法数据。在这个层面上，采取对1，254不嵌入的方法，有点与课堂上的思路一致。并为了进一步区分UB块和EB块，促使IPVO技术和RIA技术综合，将公式稍微做了下调整，同时也对非法数据流的产生进一步思考，完善了理论。

RIA技术当中，存在着对图像空间位置利用效率的问题，因为原始的UB块是不会进行嵌入的，属于无效部分，但是在论文当中，UB块充当着恢复以及进一步认证的作用，这充分提高了嵌入效率以及功能性。信息嵌入不再是明文直接输入，还运用了哈希算法，来增加破解难度，可以说是信息隐藏和密码学的综合。 在研究的过程中，理解英文方面略有问题，特别是对文中的flow琢磨不透，有时感觉理解成数据流、比特流说的通，有时觉得解释成流程图也可以，所以觉得个人的英语水平仍需提高。

当然，这篇论文感觉还没完全吃透，内心也有一丝疑问。因为根据我的理解，UB块是利用LSB算法来存储EB块中被替换的数值，这样子就可以在读取完EB块的数据后，对EB块进行恢复，实现可逆。但是UB块毕竟也是进行了修改，那么所谓的“可逆”是不用考虑UB块的恢复吗？