第37卷 第14期 计 算 机 工 程 2011年7月
Vol.37 No.14 Computer Engineering July 2011

·安全技术· 文章编号：1000—3428(2011)14—0106—03 文献标识码：A 中图分类号：TP309

基于边缘匹配与最低有效位的图像隐写方法

彭 程，秦 拯，刘 鹏，刘建蓉

(湖南大学软件学院，长沙 410082)

摘 要：提出一种基于边缘匹配与最低有效位的图像隐写方法。根据人类视觉特性，利用像素与其相邻像素间的像素差值，将图像分为平滑区、过渡区、边缘区，在3个区域中分别采用不同比特数的最低有效位进行嵌入。实验结果表明，该方法能增加图像嵌入容量并可保持较高的载密图像质量。

关键词：边缘匹配；最低有效位；像素差值；人类视觉特性；隐写

Image Steganography Method Based on Side Match and Least Significant Bit

PENG Cheng, QIN Zheng, LIU Peng, LIU Jian-rong

(Software School, Hunan University, Changsha 410082, China)

【Abstract】This paper presents an image steganographic method based on SM and Least Significant Bit(LSB). According to human visual characteristic, it divides an image into smooth area, transition area, and edged area by the difference value of neighboring pixels. It uses different bits LSB substitution in three areas above. Experimental results show that the proposed method provides larger embedding capacity and maintains a high visual quality of stego-image.

【Key words】Side Match(SM); Least Significant Bit(LSB); pixel difference value; human visual characteristic; steganography

DOI: 10.3969/j.issn.1000-3428.2011.14.034

1 概述

随着互联网在人们日常交流与信息传输中的广泛应用，信息安全问题变得日益严峻。在这种背景下，通过隐蔽信息传输的过程来实现秘密信息的安全传输方法——隐写，成为研究的热点。隐写是一种隐蔽通信技术，它通过隐藏秘密数据的存在性获得通信的安全，其基本要求是有极高的安全性和足够的信息隐藏量[1]。

隐写的典型算法为最低有效位(Least Significant Bit, LSB)算法。LSB算法隐藏容量大、不可感知性好，且算法简单，嵌入与提取都有较好的实时性。

边缘匹配[2](Side Match, SM)算法是利用像素与其相邻像素间的像素差值来判断像素是位于平滑区还是边缘区，以此来决定嵌入量，在平滑区嵌入量少，而在边缘区则嵌入量大。其最大的优点是嵌入容量较大且视觉失真小。但是SM算法在平滑区的嵌入量低，从人类视觉特性来说，平滑区应该能容忍更大的嵌入量而不被察觉。同时SM算法还存在浪费峰值信噪比、秘密信息不能正确提取的缺陷。因此，本文提出一种基于边缘匹配与最低有效位的图像隐写方法。

2 边缘匹配隐写

SM隐写[2-3]首先计算一个像素与周围相邻像素的像素差值，以此确定可嵌入信息的多少。如果差值小，表示是平滑区域，嵌入的信息就少；反之，则认为是边缘区域，可嵌入较多的信息。可分别根据2个~4个相邻像素定义差值，随后通过对差值的修改完成对秘密信息的嵌入。

以基于2个相邻像素的数据嵌入为例。保持第1行和第1列像素像素值不变，从第2行第2列的像素开始进行逐行逐列的扫描，取位于当前像素上方和左方的像素计算差值。设被修改的当前像素 P_X 的像素值为 g_x，其上方和左方的像素 P_U，P_L 的像素值分别记为 g_u，g_l，按照下列规则依次进行差值计算和数据嵌入。像素差值 d 定义为：

$$d=(g_u+g_l)/2-g_x \tag{1}$$

若 d 的值为−1、0和1，那么将1 bit的秘密信息用LSB隐写隐藏到像素 P_X 的最低位；否则先通过差值 d 计算可嵌入的比特数 n：

$$n=\mathrm{lb}|d|,\quad \text{if } |d|>1 \tag{2}$$

接着截取 n 位待嵌入二进制秘密信息，并将其转换为十进制数 b，然后把 b 嵌入到差值 d 中，获得一个新的差值 d'：

$$d'=\begin{cases}2^n+b & d>1\\ -(2^n+b) & d<1\end{cases} \tag{3}$$

由新的差值 d' 求得被嵌入像素 P_X 的新像素值 g'_x 为：

$$g'_x=(g_u+g_l)/2-d' \tag{4}$$

这样就完成了秘密信息在像素点 P_X 的嵌入。

提取秘密信息的过程很简单，按上述同样的规则对图像进行扫描，设待提取的当前像素 P_X^* 的像素值为 g_x^*，其上方和左方的像素 P_U^*，P_L^* 的像素值分别记为 g_u^*，g_l^*，按照下列规则依次进行差值计算和数据提取。新的像素差值 d^* 为：

$$d^*=(g_u^*+g_l^*)/2-g_x^* \tag{5}$$

基金项目：国家发改委信息安全专项基金资助项目(发改委办高技[2009]1886号文)；湖南省自然科学基金资助项目(09JJ3124)

作者简介：彭 程(1984－)，男，硕士研究生，主研方向：信息隐藏，信息安全；秦 拯，教授、博士；刘 鹏、刘建蓉，硕士

收稿日期：2011-01-06 **E-mail：**pengcheng@hnu.edu.cn

若 d^* 的值为−1、0 和 1，则按 LSB 法对像素 P_X^* 进行提取，否则提取 n bit 秘密信息。

$$n=\mathrm{lb}\left|d^*\right|,\quad \text{if}\quad \left|d^*\right|>1 \tag{6}$$

最后由式(7)提取出秘密信息的十进制数 b，再将 b 转换成二进制数，就完成了秘密信息的提取。

$$b=\begin{cases} d^*-2^n & d^*>1 \\ -d^*-2^n & d^*<1 \end{cases} \tag{7}$$

同时应避免由嵌入造成的溢出。如下 2 种情况像素 P_X 的像素值有可能超出[0, 255]这个范围：

(1) $d>1$ 并且 $(g_u+g_l)/2<2^{n+1}-1$；

(2) $d<1$ 并且 $(g_u+g_l)/2+2^{n+1}>256$。

因此，在嵌入时要对像素 P_X 进行检验，如果符合上述 2 种情况中的 1 种，则该像素不嵌入信息。提取时也要对像素 P_X^* 进行检验；如果符合上述 2 种情况中的 1 种，则该像素不进行信息提取。

3 基于边缘匹配与最低有效位的图像隐写方法

本文提出的方法是参照文献[4]，以 SM 隐写为基础，用 k-bit LSB 方法进行嵌入，从而提高嵌入容量，k 的值由差值 d 所属的区域决定。根据 SM 隐写中所计算出的差值 d，通过设定阈值 D 将图像划分为平滑区(R_1)、过渡区(R_2)、边缘区(R_3)。根据人类视觉特性，平滑区能够嵌入的信息最少，过渡区稍多，边缘区最多，因此，在这 3 个区域中分别采用不同比特数的 LSB 嵌入。

下面举例说明，像素差值 d 的绝对值的取值范围是 $[0,255]$，取阈值 $D_{12}=15$，$D_{23}=31$，将 $[0,255]$ 划分为平滑区 $R_1=[0,15]$，过渡区 $R_2=[16,31]$，边缘区 $R_3=[32,255]$，这 3 个区域的宽度定义为 $|R_1|$、$|R_2|$、$|R_3|$。在这 3 个区域分别采用 l bit, m bit, h bit 的 LSB 嵌入，根据前面提到的各个区域信息嵌入量关系，得出 $l\leqslant m\leqslant h$。为了保证嵌入后的差值 d^* 能够和嵌入前的差值 d 属于同一区间，还必须满足以下约束条件：

$$\begin{cases} l\leqslant \mathrm{lb}|R_1| \\ m\leqslant \mathrm{lb}|R_2| \\ h\leqslant \mathrm{lb}|R_3| \end{cases} \tag{8}$$

3.1 嵌入过程

嵌入过程描述如下：

(1)设定阈值 D，将图像划分为平滑区、过渡区、边缘区，同时设定在 3 个区间中采用的 LSB 嵌入位数 l、m、h。

(2)保持第 1 行和第 1 列像素值不变，从第 2 行第 2 列的像素开始进行逐行逐列的扫描，取位于当前像素上方和左方的像素计算差值。由式(9)计算出待嵌入像素 P_X 与其相邻像素的差值 d_x：

$$d_x=\left|(g_u+g_l)/2-g_x\right| \tag{9}$$

(3)根据步骤(1)的设定，找出 d_x 所属的区间。通过式(10)来确定 k 的值：

$$k=\begin{cases} l & d_x\in R_1 \\ m & d_x\in R_2 \\ h & d_x\in R_3 \end{cases} \tag{10}$$

(4)从二进制秘密信息流中截取 k bit 的秘密信息，将像素 P_X 的像素值 g_x 转化为二进制数，然后将 k bit 的秘密信息通过 k-bit LSB 隐写方法替换掉 P_X 的 k-LSB 方法，则像素值 g_x 变为 g_x'。

(5)使用 OPAP[5]方法调整 g_x'，降低像素值的修改量，改善载密图像质量。令 δ_x 为新的像素值 g_x' 与嵌入前的像素值 g_x 的差：

$$\delta_x=g_x'-g_x,\quad -2^k<\delta_x<2^k \tag{11}$$

根据 δ_x 的值，由式(12)得到新的像素值 g_x''：

$$g_x''=\begin{cases} g_x'+2^k & -2^k<\delta_x<-2^{k-1}\ \text{and}\ g_x'<256-2^k \\ g_x' & -2^k<\delta_x<-2^{k-1}\ \text{and}\ g_x'\geqslant 256-2^k \\ g_x' & -2^{k-1}<\delta_x<2^{k-1} \\ g_x' & 2^{k-1}<\delta_x<2^k\ \text{and}\ g_x'<2^k \\ g_x'-2^k & 2^{k-1}<\delta_x<2^k\ \text{and}\ g_x'\geqslant 2^k \end{cases} \tag{12}$$

(6)通过式(13)计算新的差值 d_x''：

$$d_x''=\left|(g_u+g_l)/2-g_x''\right| \tag{13}$$

(7)如果 d_x 和 d_x'' 属于不同的区域，则按以下规则进行调整：

1)当 $d_x\in R_1$，$d_x''\notin R_1$ 时，如果 $(g_u+g_l)/2\geqslant g_x''$，则 g_x'' 调整为 $g_x''+2^k$；如果 $(g_u+g_l)/2<g_x''$，则 g_x'' 调整为 $g_x''-2^k$。

2)当 $d_x\in R_2$，$d_x''\in R_1$ 时，如果 $(g_u+g_l)/2\geqslant g_x''$，则 g_x'' 调整为 $g_x''-2^k$；如果 $(g_u+g_l)/2<g_x''$，则 g_x'' 调整为 $g_x''+2^k$。

3)当 $d_x\in R_2$，$d_x''\in R_3$ 时，如果 $(g_u+g_l)/2\geqslant g_x''$，则 g_x'' 调整为 $g_x''+2^k$；如果 $(g_u+g_l)/2<g_x''$，则 g_x'' 调整为 $g_x''-2^k$。

4)当 $d_x\in R_3$，$d_x''\notin R_3$ 时，如果 $(g_u+g_l)/2\geqslant g_x''$，则 g_x'' 调整为 $g_x''-2^k$；如果 $(g_u+g_l)/2<g_x''$，则 g_x'' 调整为 $g_x''+2^k$。

(8)经过步骤(7)的调整后，能够保证 d_x 和 d_x'' 属于同一区域，但是像素值 g_x'' 有可能超出 $[0,255]$ 这个范围。因此，在步骤(7)之后要对像素值 g_x'' 进行检验，如果 $g_x''\in[0,255]$，则用 g_x'' 替代 g_x，嵌入过程完成；如果 $g_x''\notin[0,255]$，则按以下步骤进行再次调整：

1)对像素值 g_x 进行修改。如 $0\leqslant g_x\leqslant 2^h$，则 $g_x=g_x+2^k$；如 $255-2^h\leqslant g_x\leqslant 255$，则 $g_x=g_x-2^k$。

2)用调整后的 g_x 重复步骤(2)~步骤(7)，完成嵌入。

3.2 提取过程

首先按上述同样的规则对图像进行扫描，设待提取的当前像素 P_X^* 的像素值为 g_x^*，其上方和左方的像素 P_U^*, P_L^* 的像素值分别记为 g_u^*, g_l^*，按以下步骤进行提取：

(1)用与嵌入过程相同的阈值 D 将图像分为平滑区、过渡区、边缘区。同时确定与嵌入过程相同的 LSB 嵌入位数 l、m、h；在这里用来确定该提取的 bit 位数 k。

(2)计算新的像素差值 d_x^*：

$$d_x^*=\left|(g_u^*+g_l^*)/2-g_x^*\right| \tag{14}$$

(3)根据步骤(1)的划分，找出 d_x^* 所属的区间，通过式(15)来确定 k 的值。

$$k=\begin{cases} l & d^*\in R_1 \\ m & d^*\in R_2 \\ h & d^*\in R_3 \end{cases} \tag{15}$$

(4)对像素 P_X^* 的像素值 g_x^* 进行 k-bit LSB 提取，提取出来的二进制数就是嵌在像素 P_X^* 内的秘密信息。

4 实验结果与分析

4.1 实验结果

本文选用大小为 512×512 像素的标准灰度图像作为实验

载体图像。嵌入的秘密信息为随机二进制数据流。用峰值信噪比(Peak Signal to Noise Ratio, PSNR)来衡量载密图像质量。在实验中，本文提出的方法采用的阈值为 $D_{12}=15$, $D_{23}=31$ 。在 3 个区间的嵌入 bit 数分别为 $l=3$、$m=4$、$h=5$，即“3-4-5 嵌入”。

原始载体图像与嵌入秘密信息后的载密图像如图 1 所示。可以看出，2 幅图像在视觉上没有任何差别，信息隐藏是成功的。

(a)原始载体图像 (b)载密图像

图 1 原始载体图像与载密图像

SM 算法与本文方法的嵌入容量与 PSNR 比较见表 1。

表 1 2 种方法的嵌入容量和 PSNR 比较

载体图像	SM 算法		本文方法	
	容量/bit	PSNR/dB	容量/bit	PSNR/dB
Lena	426 964	40.148 3	799 998	38.585 7
Baboon	726 746	31.887 1	901 092	33.101 6
Airplane	401 272	39.218 1	805 504	37.974 2
Peppers	480 626	39.037 4	801 109	38.435 1
Sailboat	551 044	36.692 4	823 272	36.579 7
Tiffany	419 865	40.333 8	801 578	38.173 8
Boat	424 754	38.837 8	810 102	37.389 9
Couple	486 934	38.005 6	815 701	37.128 6
Man	482 367	37.534 2	818 005	36.832 6
Watch	414 285	38.736 3	811 610	37.341 5
平均值	481 486	38.043 1	818 797	37.389 9

SM 算法的平均嵌入容量为 481 486 bit，而本文方法的平均嵌入容量为 818 797 bit，比 SM 算法多了 337 312 bit，这是由于本文方法提高了平滑区的嵌入量；同时 PSNR 仅下降了 0.653 2 dB，且平均值大于 37 dB，可见本文提出的方法不仅大大提高了嵌入容量，还保持了较好的图像质量。

2 种方法的另外 2 个衡量图像质量的指标 MSE 和 IF 比较如表 2 所示。MSE 代表载密图像相对原始图像的改变量，IF 代表载密图像与原始图像的相似度。可以看出，2 种方法在这 2 个指标上几乎没有区别，进一步证明了本文方法有较好的图像质量。

表 2 2 种方法的 MSE 和 IF 比较

载体图像	SM 算法		本文方法	
	MSE	IF	MSE	IF
Lena	6.284 3	0.999 674	9.005 5	0.999 426
Baboon	42.109 2	0.997 735	31.836 1	0.998 294
Airplane	7.785 2	0.999 772	10.367 1	0.999 697
Peppers	8.115 7	0.999 532	9.323 2	0.999 463
Sailboat	13.926 1	0.999 235	14.292 3	0.999 216
Tiffany	6.021 3	0.999 862	9.901 4	0.999 773
Boat	8.497 5	0.999 463	11.860 1	0.999 247
Couple	10.292 4	0.999 387	12.595 4	0.999 251
Man	11.472 9	0.998 977	13.484 1	0.998 801
Watch	8.698 5	0.998 719	11.993 1	0.998 241
平均值	12.320 3	0.999 236	11.860 1	0.999 141

4.2 结果分析

在一般情况下，嵌入容量越高则 PSNR 越低，这是因为嵌入的信息越多，对原始载体图像的修改量越大。但是从上一节的实验结果可以看出，本文提出的方法将嵌入容量提高了 337 312 bit，PSNR 却仅仅下降了 0.653 2 dB，出现这种现象的原因是 SM 算法在嵌入过程中会出现对原始载体图像修改的数值大于嵌入信息的数值的情况。此外，SM 算法还存在不能正确提取秘密信息的缺陷，分析如下：

(1)令 SM 算法嵌入前后的像素值的修改量的绝对值为 α ，由式(1)、式(2)、式(4)可知：

$$\alpha=\left|g'_x-g_x\right|=\left|d'-d\right|$$

即 α 的值由 d' 和 d 决定，而嵌入的秘密信息的十进制数为 b 。令 $d'>1$ ， $d>1$ ， $d=2^n+f$ ，则由式(3)可得：

$$\alpha=\left|2^n+b-2^n-f\right|$$

当 $f>2b$ 时，有 $\alpha>b$ 。例如，假设 $g_u=58$ 、 $g_l=46$ 、 $g_x=38$ ，则有 $d=14$ 、 $n=3$ 、 $f=7$ 。取 3bit 秘密信息 001，则秘密信息的十进制数 $b=1$ 、 $d'=9$ 、 $\alpha=\left|9-14\right|=5$ 。可见，仅为了嵌入数值为 1 的秘密信息，像素值的修改量却达到了 5。这是对 PSNR 值的一种浪费，是导致低嵌入容量下低 PSNR 值的主要原因。由于 d' 和 d 都在[-255,255]对称，因此以上的分析不失一般性。

(2)SM 算法在 $d\in(-1,0,1)$ 时，会出现提取信息不正确的情况。例如，假设 $g_u=37$ 、 $g_l=37$ 、 $g_x=38$ ，则 $d=-1$ ，取嵌入的秘密信息为 $b=1$ 。用 LSB 方法嵌入得 $g'_x=39$ 。在提取过程中，由式(5)得 $d^*=-2$ ，再由式(6)得 $n=1$ ，最后由式(7)得 $b=0$ ，提取的信息与嵌入信息不符。由此可见 SM 算法会由于其本身的缺陷导致秘密信息传输不正确。

由上述分析可知，SM 算法存在浪费峰值信噪比，秘密信息不能正确提取的缺陷，本文方法则避免了上述情况出现。

5 结束语

本文提出了一种基于 SM 与 LSB 的隐写方法。实验证明，图像的平均嵌入容量提高了 337 312 bit，有效地控制了失真，具有较好的图像质量和视觉隐蔽性。本文提出的方法经过适当的修改后，也可用于基于 3 个、4 个相邻像素差值的隐写。在保证算法的综合性能不下降的前提下，下一步可以考虑提高算法的抗隐写分析能力，从而进一步提高算法的安全性。

参考文献

[1] 耿广志, 张汗灵, 熊彩琼. 自适应的无损像素差分隐写算法[J]. 计算机工程, 2009, 35(23): 136-137.

[2] 王威娜, 张新鹏, 王朔中. 针对边缘匹配嵌入法的密写分析以及嵌入率估计[C]//第十二届全国图像图形学学术会议论文集. 北京: [出版者不详], 2005.

[3] Chang Chin-Chen, Tseng Hsien-Wen. A Steganographic Method for Digital Images Using Side Match[J]. Pattern Recognition Letters, 2004, 25(12): 1431-1437.

[4] Yang Cheng-Hsing, Weng Chi-Yao, Wang Shiuh-Jeng. Adaptive Data Hiding in Areas of Images With Spatial LSB Domain Systems[J]. IEEE Transactions on Information Forensics and Security, 2008, 3(3): 488-497.

[5] Chan Chi-Kwong, Cheng Lee-Ming. Hiding Data in Images by Simple LSB Substitution[J]. Pattern Recognition, 2004, 37(3): 469-474.

编辑 顾姣健