A High Quality and Capacity Steganographic Method by Pixel-value Differencing

具有優良影像品質與高嵌入容量的像素值差異影像偽裝技術

魏瑋慶
逢甲大學通訊工程學系
Email: m9432005@fcu.edu.tw

陳志瀅
逢甲大學通訊工程學系
Email: chihchen@fcu.edu.tw

林秀峰
逢甲大學資訊工程學系
Email: hflin@fcu.edu.tw

谷秋月
南開科技大學機械工程系
逢甲大學通訊工程學系
Email: moon384@nkut.edu.tw

摘要 **— 影像偽裝術(Image Steganography)是一種有別於密碼學之加/解密作法的秘密通信技術。2003 年，Wu 與 Tsai 提出一個相當簡單而且有效的影像偽裝術，稱為像素值差異技術(pixel-value differencing technique; PVD)。它的觀念與作法主要是根據相鄰的兩個像素值的差異級別來決定嵌入此兩像素的機密訊息之位元數。如此得以在比較平滑的區域嵌入較少量的訊息而在像素值差異較大的邊緣區域嵌入較多的訊息，一方面可以降低偽裝影像的失真(distortion)度達成人類肉眼的不可察覺性(imperceptibility)，另一方面則可嘗試嵌入儘量多的訊息位元。此外，實驗結果指出，他們的方法還可以抵擋由 Fridrich 等人在 2001 年所發表的 RS 統計偵測攻擊。**

鑑於像素值差異技術的簡單性與有效性，在 Wu 與 Tsai 之後，許多學者競相提出改良的作法。其中，Wang、Wu、Tsai 與 Hwang 等人在 2008 年利用模運算，提出一個可以在嵌入訊息時降低像素值修改量的改良作法，大幅的提高了像素值差異法所產生的偽裝影像之品質。

本論文針對 Wang 等人的作法，提出〝修改權值〞的觀念來進一步改善像素值差異嵌入技術的偽裝影像品質以及嵌入容量，同時進行一系列的實驗。實驗結果證明，當嵌入的秘密訊息數量相同時，我們的作法所產生的偽裝影像比 Wang 等人的結果具有更好的影像品質，也即具有更高的 PSNR 值(Peak-Signal-to-Noise Ratio)。尤其當嵌入容量逐漸增大到令 Wang 等人的結果之 PSNR 值降低到無法滿足不可察覺性的要求時，我們的作法所產生的偽裝影像仍然具有相當安全的 PSNR 值。此外，實驗結果也指出，我們的方法與 Wang 等人的方法一樣的可以抵擋由 Fridrich 等人所發表的 RS 統計偵測的攻擊。因此，我們的方法比前此已發表的所有植基於像素值差異的嵌入技術更適合偽裝術的實際應用。

關鍵詞 **— 影像偽裝術、像素值差異嵌入技術、不可察覺性、不可偵測性、RS 偽裝分析術。**

Abstract **— Image steganography is a secret communication technique that is different from encryption/decryption methods from cryptology. In 2003, Wu and Tsai proposed a simple and effective image steganographic technique, called the pixel-value differencing (PVD) technique. Its main concept and technic is to decide the amount of information bits embedded in two neighboring pixels based on the difference of their pixel values. In this way, regions with higher smoothness will embed less information and those with lower smoothness will have the opposite, therefore reducing the degree of distortion of the image and achieving imperceptibility. In addition, experimental results show that their method is capable of resisting**

Fridrich et al.'s RS steganalysis.

Because of the simplicity and efficiency of the pixel-value differencing technique, Wang, Wu, Tsai, and Hwang proposed an improved method with lower pixel value modification using modulus computation in 2008, which further increased the image quality of the steganography utilizing the pixel-value differencing technique.

In this paper, we refer to Wang, Wu, Tsai, and Hwang's method and propose a new pixel-value differencing technique implementing the concept of using "weights", followed by carrying out a series of experiments. The results prove that our proposed method has higher stego-image quality than Wang, Wu, Tsai, and Hwang's method when embedding the same secret message, implying that we have higher PSNR value. Especially under the requirement of high embedding capacity, our method still maintains a safe PSNR (Peak Signal to Noise Ratio) value where the PSNR value of Wang, Wu, Tsai, and Hwang's method drops too low and fails to satisfy imperceptibility. Therefore, compared with previously proposed embedding techniques that base on pixel-value differencing, our method provides greater suitability for practical applications.

Keywords **— steganography, embedding by pixel-value differencing, imperceptibility, non-detectability, RS steganalysis.**

一、前言

影像偽裝術是指利用具有意義且非機密性的數位影像來掩護與傳送可以用位元串(bit stream)表示的機密訊息，使得傳送者與接收者之外的其它人難以察覺出或偵測出機密訊息的存在以及秘密通訊行為之進行的一種技術。其中，用來掩護機密訊息的數位影像稱為掩護影像(cover image)。將機密訊息藏入掩護影像的方法稱為訊息嵌入法(embedding method) ，通常是將秘密訊息的位元嵌入掩護影像在空間域的像素(稱為空間域嵌入法)或頻率域的係數(稱為頻率域嵌入法)。機密訊息被嵌入掩護影像後得到一個與原始掩護影像近似的影像稱為偽裝影像(stego-image)。而將機密訊息從偽裝影像中取出的方法則稱為訊息萃取法(extracting method) [9][10][11]。

影像偽裝術的安全性取決於不可察覺(imperceptibility)以及不可偵測(non-detectability)秘密訊息之存在的程度。所以，影像偽裝術的重點在於能否設計一套有效的訊息嵌入/萃取法使得攻擊者很難以肉眼察覺出或以統計方法偵測出偽裝影像中秘密訊息的存在。如果使用不當的嵌入作法而改變自然影像的某些統計特性，則利用適當的統計量加以檢驗與分析，常可偵測出肉眼無法察覺的異常現象，這種統計檢驗與分析的技術稱為偽裝分析學(Steganalysis)[6]。此外，在安全的前提下，一個實用有效的影像偽裝術，尚須要求具有儘可能高的嵌入容量(embedding capacity)以及儘可能低的計算複雜度(computation complexity)。

LSB(Least Significant Bit)嵌入法是最典型的空間域嵌入技術。它的作法乃是將掩護影像在空間域的某些連續或隨機選取的像素的 LSB(Least Significant Bit)位元由機密訊息位元來取代。目前許多常用的偽裝術軟體如 EZstego、Hide&seek、S-Tool4、Steganos 以及 Stego Dos 等都是採用 LSB 嵌入的作法。雖然 LSB 嵌入法具有觀念簡單、計算容易、以及嵌入容量較大的優點，但卻無法抵擋由 Fridrich 等人[6] 在 2001 年提出的 RS 統計偵測攻擊法。(我們將在第 2.3 節回顧 RS 統計偵測攻擊法的詳細過程與原理。)

2003 年，Wu 與 Tsai[4]提出一個相當簡單而且有效的影像偽裝術，稱為像素值差異技術(pixel-value differencing technique; PVD)。它的觀念與作法主要是根據相鄰的兩個像素值的差異級別來決定嵌入此兩像素的機密訊息之位元數。如此得以在比較平滑的區域嵌入較少量的訊息而在像素值差異較大的邊緣區域嵌入較多的訊息，一方面可以降低偽裝影像的失真(distortion)度達成人類肉眼的不可察覺性(imperceptibility)，另一方面則可嘗試嵌入儘量多

的訊息位元。此外，實驗結果指出，他們的方法還可以抵擋 Fridrich 等人[6]所發表的 RS 統計偵測攻擊。

鑑於像素值差異嵌入技術的簡單性與有效性，在 Wu 與 Tsai[4]之後，許多學者競相提出改良的作法。2005 年，Wu 等人[5]發表一個可以提高嵌入容量的改良的作法。首先利用兩個連續像素的差異值，再依據像素的差異值定位此兩像素是屬於光滑區域還是複雜區域。若是屬於光滑區域，則使用 LSB 取代法將機密訊息嵌藏到兩像素值中；若是屬於複雜區域，則使用 Wu 與 Tsai[4]的像素值差異法將機密訊息嵌藏到兩像素值中。實驗結果證明，他們的作法比 Wu 與 Tsai[4]的方法具有更大的嵌入容量，同時也能維持相當良好的偽裝影像品質。然而，Yang 等人[2]在 2006 年以實驗證明指出，Wu 等人[5]的嵌入技類似於 LSB 嵌入法，無法阻擋 RS 統計偵測攻擊。

接著，Yang 等人[1]、Chang 等人[7]、與 Chen 等人[12]分別在 2006 年與 2008 年提出利用較大區塊的多個相鄰像素差異級別來嵌入機密訊息的改良作法。實驗結果證明，他們的作法都比 Wu 與 Tsai[4]的方法具有更大的嵌入容量，同時也能維持相當良好的偽裝影像品質。此外，還都可以有效的抵擋 RS 統計偵測攻擊。

2008 年，Wang、Wu、Tsai 與 Hwang 等人[3]利用模運算，提出一個可以在嵌入訊息時大幅降低像素值修改量的改良作法，因而大幅的提高了像素值差異法所產生的偽裝影像之品質。此外，他們的作法還可以有效的解決嵌入溢位的問題以及 RS 統計偵測的攻擊。

2008 年，Kim 等人[8]針對 Wang 等人[3]的作法提出一個類似前述之 Wu 等人[5] 針對 Wu 與 Tsai[4]作法的改良方案。首先利用兩個連續像素的差異值，再依據像素的差異值定位此兩像素是屬於光滑區域還是複雜區域。若是屬於光滑區域，則使用 LSB 法將機密訊息嵌藏到兩像素值中；若是屬於複雜區域，則使用 Wang 等人[3]的作法將機密訊息嵌藏到兩像素值中。實驗結果證明，他們的作法比 Wang 等人[3]的方法具有較大的嵌入容量，同時也能維持相當良好的偽裝影像品質。然而，與 Wu 等人[5]的作法類似的，Kim 等人[8]的嵌入法也無法阻擋 RS 統計偵測攻擊。

本論文參考 Wang 等人[3]的作法，提出一個“修改權值”的觀念來進一步降低在嵌入作業時對掩護影像之像素值的修改量的像素值差異嵌入技術，同時進行一系列的實驗。實驗結果證明，當嵌入的訊息數量相同時，我的作法所產生的偽裝影像比 Wang 等人[3]的結果具有更好的影像品質，也即具有更高的 PSNR 值 (Peak-Signal-to-Noise Ratio)。尤其當嵌入容量逐漸增大到令 Wang 等人[3]的結果之 PSNR 值降低到無法滿足不可察覺性的要求時，我們的結果仍然具有相當安全的 PSNR 值。換言之，在安全的前提下，我們的方法容許比 Wang 等人[3]的方法更大的嵌入容量。此外，實驗結果也指出， 我們的方法也可以有效的抵擋 RS 統計偵測的攻擊。因此，我們的方法將比前此已發表的所有植基於像素值差異的嵌入技術更適合偽裝術的實際應用。

本論文由五個小節所組成。第二節簡單的回顧 Wu 與 Tsai[4]的像素值差異的嵌入技術、Wang 等人[3]的改良嵌入技術、以及 Fridrich 等人[6]所提出的 RS 偽裝分析技術之過程與原理。第三節介紹我們的改良技術並說明訊息嵌入和萃取的觀念與作法。第四節呈現與討論我們的實驗結果。第五節則是我們的結論。

二、相關研究之回顧

2.1 Wu 與 Tsai[4]的像素值差異偽裝技術之回顧

2003 年，Wu 與 Tsai[4]提出一個相當簡單而且有效的灰階影像偽裝技術。他們的作法先將像素值域[0, 255]分段(或分級) 如下：

$$[0,255]=\bigcup_{k=1}^{n} R_k \qquad (2.1)$$

其中 $R_k=[l_k, u_k]$， $l_1=0, u_n=255, u_k=l_{k+1}-1$ ，$|R_k|=(u_k-l_k+1)=2^{t_k}$, $1\le k\le n$, 而且 $0\le t_1\le t_2\le\cdots\le t_n\le 8$, 。接著，將掩護影像分割為由兩個相鄰的像素所有組成的若干個區塊，再根據每一區塊的兩個像素值的差異級別來嵌入不同長度的訊息位元。其在掩護影像中嵌入訊息與從偽裝影像中萃取出嵌入訊息的作法可以簡略回

顧如下。

Wu 與 Tsai[4]的訊息嵌入演算法：

步驟：

1. 決定一組像素值域分段 $\{R_k=[l_k,\ u_k]\,|\,1\le k\le n\}$，其中 $|R_k|=2^{t_k}$, $0\le t_1\le t_2\le\cdots\le t_n\le 8,\ 1\le k\le n$。
2. 將掩護影像分割為由兩個相鄰的像素組成的若干個區塊。
3. 以一亂數種子與一擬亂數列產生器決定各個區塊的處理順序。
4. 依序對第 i 個區塊(假設其中兩個像素的灰階值為 g_i 和 g_{i+1})執行如下的訊息嵌入處理。
   (1) (i) 計算兩個像素值的差異
   $$d=g_{i+1}-g_i\ 。$$
   (ii) 決定 $|d|$ 所屬之區段
   $$R_k=[\ell_k,\ u_k)\ 。$$
   (2) 令
   $$f((g_i,g_{i+1}),x)=\begin{cases}(g_i-\lceil x/2\rceil,\ g_{i+1}+\lfloor x/2\rfloor), & \text{當}d\text{為奇數時；}\\ (g_i-\lfloor x/2\rfloor,\ g_{i+1}+\lceil x/2\rceil), & \text{當}d\text{為偶數時。}\end{cases}\tag{2.2}$$
   若 $d\ge 0$ 時
   $f((g_i,g_{i+1}),u_k-d)\notin[0,255]$
   或 $d<0$ 時
   $f((g_i,g_{i+1}),-u_k-d)\notin[0,255]$
   則不使用此區塊來嵌入任何訊息，回到步驟 3 繼續處理下一個區塊。(//註：因為此時若仍嵌入訊息，則步驟(4)之 g_i' 和 g_{i+1}' 的值極有可能 $\notin[0,\ 255]$，也即可能會產生溢位問題//)
   否則繼續執行步驟(3)。
   (3) 計算 $$d'=\begin{cases}l_k+b,\ d\ge 0;\\ -(l_k+b),\ d<0.\end{cases}\tag{2.3}$$
   其中，b 為依序自機密訊息中取出的 $t_k=\log(u_k-l_k+1)$ 個位元串之十進位值。
   (4) 修改 g_i 和 g_{i+1}，如下：
   $$(g_i',g_{i+1}')=f((g_i,g_{i+1}),m)=\begin{cases}(g_i-\lceil m/2\rceil,\ g_{i+1}+\lfloor m/2\rfloor), & \text{當}d\text{為奇數時；}\\ (g_i-\lfloor m/2\rfloor,\ g_{i+1}+\lceil m/2\rceil), & \text{當}d\text{為偶數時。}\end{cases}\tag{2.4}$$
   其中 $m=d'-d$。

Wu 與 Tsai[4]的訊息萃取演算法：

步驟：

1. 使用與嵌入處理相同的規則，將掩護影像分割為由兩個相鄰的像素組成的若干個區塊。
2. 使用與嵌入處理相同的亂數種子與擬亂數列產生器決定各個區塊的處理順序。
3. 依序對第 i 個區塊(假設其中兩個像素的灰階值為 g_i' 和 g_{i+1}')執行如下的訊息萃取處理。
   (1) (i) 計算兩個像素值的差異
   $$d'=g_{i+1}'-g_i'\ 。$$
   (ii) 決定 $|d'|$ 所屬之區段
   $$R_k=[\ell_k,\ u_k)\ 。$$
   (2) 令
   $$f((g_i',\ g_{i+1}'),\ x)=\begin{cases}(g_i'-\lceil x/2\rceil,\ g_{i+1}'+\lfloor x/2\rfloor), & \text{當}d'\text{為奇數時；}\\ (g_i'-\lfloor x/2\rfloor,\ g_{i+1}'+\lceil x/2\rceil), & \text{當}d'\text{為偶數時。}\end{cases}\tag{2.5}$$
   若 $d'\ge 0$ 時
   $f((g_i',g_{i+1}'),u_k-d')\notin[0,255]$
   或 $d'<0$ 時
   $f((g_i',g_{i+1}'),-u_k-d')\notin[0,255]$
   則回到步驟 2 繼續處理下一個區塊。(//註：因為此區塊並未被使用來嵌入訊息//)
   否則繼續執行步驟(3)。
   (3) 取出嵌入在第 i 個區塊的訊息之十進位值如下：
   $$b=\begin{cases}d'-l_k,\ d'\ge 0;\\ -(d'+l_k),\ d'<0.\end{cases}\tag{2.6}$$
   (4) 將 b 轉換成長度為 t_k 的位元串，即可得到原來嵌入此區塊的機密訊息位元串。

Wu 與 Tsai[4]的訊息嵌入及萃取演算法可以用以下的簡例進一步說明如下：假設像素值域分段為{R_1=[0, 7], R_2=[8, 15], R_3=[16, 31], R_4=[32, 63], R_5=[64, 127], R_6= [128, 255]}且第 i 個區塊的兩個相鄰像素值為 $g_i=98$ 和 $g_{i+1}=116$。則因兩像素差異值 d=18 屬於差異級別 R_3 =[16, 31]，得知可以嵌入訊息的位元個數為 $t_3=4$，所以自尚未嵌入之機密訊息位元串中依序取出長度為 4 的位元串(設為 1101)並轉換成十進位值 b=13。因為差異級別下界為 16，得知新像素差異值為 $d'=16+13=29$。由於兩像素的差異值從 18 擴增為 29，擴增量為 $m=d'-d=29-18=11$，所以根據(2.4)式將擴增量 11 平均分配到二個像素，最後得到的偽裝像素值為 $g_i'=g_i-\lfloor m/2\rfloor=98-\lfloor 11/2\rfloor=93$ 和 $g_{i+1}'=g_{i+1}+\lceil m/2\rceil=116+\lceil 11/2\rceil=122$。

欲從偽裝區塊 $(g_i', g_{i+1}')=(93, 122)$ 萃取機密訊息，則因 $d'=g_{i+1}'-g_i'$ =122-93=29 屬於差異級別 R_3 =[16, 31]，可知嵌入此區塊的機密訊息之十進位值為 $b=d'-l_3=29-16=13$，最後再將 b =13 轉換成長度為 $t_3=4$ 的位元串，即可得到原來嵌入此區塊的機密訊息位元串 1101。

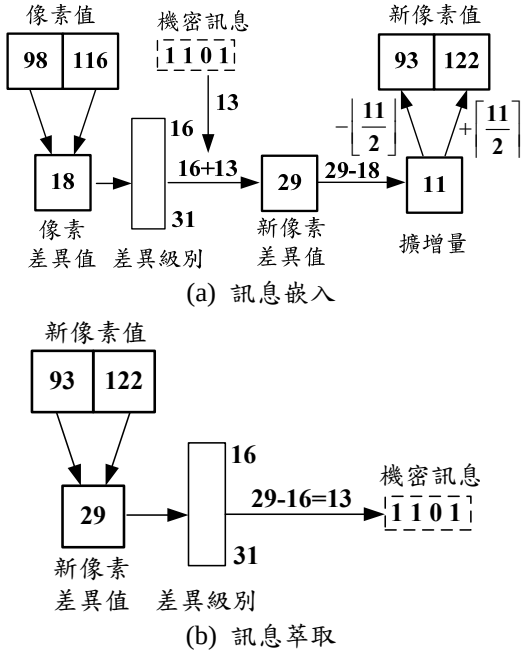


圖 2.1 Wu 與 Tsai[4]的方法簡例

在[4]中，Wu 與 Tsai 經由實驗證明他們的偽裝影像品質良好，而且可以抵擋 Fridrich 等人[6]的 RS 統計偵測攻擊。然而，因為對於一般的自然影像而言，大多數相鄰的兩個像素之值差異不大。所以，嵌入的訊息量並非令人滿意。

2.2 Wang 等人[3]的偽裝技術之回顧

2008 年，Wang 等人[3]利用模運算，提出一個可以在使用像素值差異法嵌入訊息時大幅降低像素值修改量的改良作法，因而大幅的提高了像素值差異法所產生的偽裝影像之品質。此外，他們的作法還可以有效的解決嵌入溢位的問題以及 RS 統計偵測的攻擊。

Wang 等人[3]的嵌入演算法：

步驟：

1. 決定一組像素值域分段 $\{R_k=[l_k, u_k]\,|\,1\le k\le n\}$，其中 $|R_k|=2^{t_k}$, $0\le t_1\le t_2\le\cdots\le t_n\le 8$, $1\le k\le n$。
2. 將掩護影像分割為由兩個相鄰的像素組成的若干個區塊。
3. 以一亂數種子與一擬亂數列產生器決定各個區塊的處理順序。
4. 依序對第 i 個區塊(假設其中兩個像素的灰階值為 g_i 和 g_{i+1})執行如下的訊息嵌入處理。
   (1) (i) 計算兩個像素值的差異
   $$d=g_{i+1}-g_i。$$
   (ii) 決定 $|d|$ 所屬之區段
   $$R_k=[\ell_k, u_k)。$$
   (2) 計算
   $$F_{rem(i)}=(g_i+g_{i+1})\bmod 2^{t_k} \qquad (2.7)$$
   (3) 依序自尚未嵌入的機密訊息中讀取出 t_k 個位元並計算其十進位值 b。
   (4) 依據下列規則修改 (g_i, g_{i+1}) 為 (g_i', g_{i+1}')。

Case 1:

若 $F_{rem(i)}>b$ 且 $m\le(2^{t_i})/2$ 且 $g_i\ge g_{i+1}$

則 $(g_i', g_{i+1}')=(g_i-\lceil m/2\rceil, g_{i+1}-\lfloor m/2\rfloor)$;

Case 2:

若 $F_{rem(i)} > b$ 且 $m \le (2^{t_i})/2$ 且 $g_i < g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i - \lfloor m/2 \rfloor, g_{i+1} - \lceil m/2 \rceil)$;

Case 3:

若 $F_{rem(i)} > b$ 且 $m > (2^{t_i})/2$ 且 $g_i \ge g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i + \lfloor m_1/2 \rfloor, g_{i+1} + \lceil m_1/2 \rceil)$;

Case 4:

若 $F_{rem(i)} > b$ 且 $m > (2^{t_i})/2$ 且 $g_i < g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i + \lceil m_1/2 \rceil, g_{i+1} + \lfloor m_1/2 \rfloor)$;

Case 5:

若 $F_{rem(i)} \le b$ 且 $m \le (2^{t_i})/2$ 且 $g_i \ge g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i + \lfloor m/2 \rfloor, g_{i+1} + \lceil m/2 \rceil)$;

Case 6:

若 $F_{rem(i)} \le b$ 且 $m \le (2^{t_i})/2$ 且 $g_i < g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i + \lceil m/2 \rceil, g_{i+1} + \lfloor m/2 \rfloor)$;

Case 7:

若 $F_{rem(i)} \le b$ 且 $m > (2^{t_i})/2$ 且 $g_i \ge g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i - \lceil m_1/2 \rceil, g_{i+1} - \lfloor m_1/2 \rfloor)$;

Case 8:

若 $F_{rem(i)} \le b$ 且 $m > (2^{t_i})/2$ 且 $g_i < g_{i+1}$

則 $(g_i', g_{i+1}') = (g_i - \lfloor m_1/2 \rfloor, g_{i+1} - \lceil m_1/2 \rceil)$.

其中，

$m = \left| F_{rem(i)} - b \right|, m_1 = 2^{t_i} - \left| F_{rem(i)} - b \right|$。

(2.8)

(5) 依據下列規則修改發生溢位的 (g_i', g_{i+1}') 為 (g_i'', g_{i+1}'')

Case 1: 若 $g_i \approx 0, g_{i+1} \approx 0$

而且 $g_i' < 0$ or $g_{i+1}' < 0$,

則將 (g_i', g_{i+1}') 修正為

$(g_i'', g_{i+1}'') = (g_i' + (2^{t_i})/2, g_{i+1}' + (2^{t_i})/2)$.

Case 2: 若 $g_i \approx 255, g_{i+1} \approx 255$

而且 $g_i' > 255$ or $g_{i+1}' > 255$,

則將 (g_i', g_{i+1}') 修正為

$(g_i'', g_{i+1}'') = (g_i' - (2^{t_i})/2, g_{i+1}' - (2^{t_i})/2)$.

Case 3: 若 g_i 與 g_{i+1} 反差大於128 (即 $|d_i| = |g_i - g_{i+1}| > 128$), 而且

Case 3-1: 若 $g_i' < 0$ 且 $g_{i+1}' \ge 0$,

則將 (g_i', g_{i+1}') 修正為

$(g_i'', g_{i+1}'') = (0, g_{i+1}' + g_i')$.

Case 3-2: 若 $g_i' \ge 0$ 且 $g_{i+1}' < 0$,

則將 (g_i', g_{i+1}') 修正為

$(g_i'', g_{i+1}'') = (g_i' + g_{i+1}', 0)$.

Case 3-3: 若 $g_i' > 255$ 且 $g_{i+1}' \ge 0$,

則將 (g_i', g_{i+1}') 修正為

$(g_i'', g_{i+1}'') = (255, g_{i+1}' + (g_i' - 255))$.

Case 3-4: 若 $g_i' \ge 0$ 且 $g_{i+1}' > 255$,

則將 (g_i', g_{i+1}') 修正為

$(g_i'', g_{i+1}'') = (g_i' + (g_{i+1}' - 255), 255)$.

(2.9)

Wang 等人[3]的萃取演算法：

步驟：

1. 將偽裝影像分割為由兩個相鄰像素所組成的若干個區塊。
2. 使用與嵌入作業相同的亂數種子與擬亂數列產生器決定各個區塊的處理順序。
3. 依序對第 i 個區塊(假設其中兩個像素的灰階值為 g_i' 和 g_{i+1}')執行如下的訊息萃取處理。

(1) (i) 計算兩像素的差異值

$d' = g_i' - g_{i+1}'$。

(ii) 決定 $|d'|$ 所屬之區段

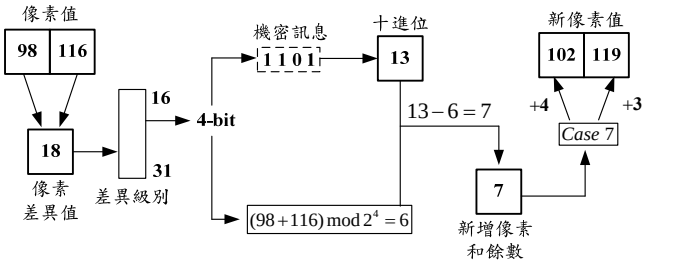
$R_k = [l_k, u_k]$ 以及可以嵌入此區塊的機密訊息位元數 t_k。

(2) 計算 $b = (g_i' + g_{i+1}') \bmod 2^{t_k}$，再將 b 轉換成長度為 t_k 的位元串，即可得到原來嵌入此區塊的機密訊息位元串。

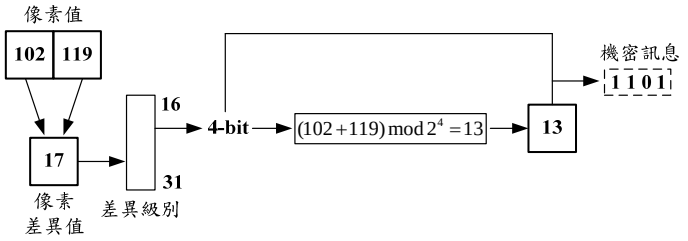
Wang 等人[3]的訊息嵌入及萃取演算法可以用一簡例進一步說明如下：假設像素值域分段為 {R_1=[0, 7], R_2=[8, 15], R_3=[16, 31], R_4=[32, 63],

R_5=[64, 127], R_6= [128, 255]}且第 i 個區塊的兩個相鄰像素值為 $g_i = 98$ 和 $g_{i+1} = 116$。則因兩像素差異值 d=18 屬於差異級別 R_3 =[16, 31]，得知可以嵌入訊息的位元個數為 $t_3 = 4$，所以自尚未嵌入之機密訊息位元串中依序取出長度為 4 的位元串(設為 1101)並轉換成十進位值 b=13。接著計算 $F_{rem(i)} = (g_i + g_{i+1}) \bmod 2^{t_k} = (98+116) \bmod 2^4 = 6$ 以及 $m = \left| F_{rem(i)} - b \right| = \left| 6 - 13 \right| = 7$。因為 $F_{rem(i)} \le b$ 且 $m \le (2^{t_i})/2$ 且 $g_i < g_{i+1}$，所以根據嵌入步驟 4-(4) Case 6，得到 $(g_i', g_{i+1}') = (g_i + \lceil m/2 \rceil, g_{i+1} + \lfloor m/2 \rfloor)$ =(98+4, 116+3)= (102, 119)。

欲從偽裝區塊 (g_i', g_{i+1}') = (102, 119) 萃取機密訊息，則因 $d' = g_{i+1}' - g_i'$ =119-102=17 屬於差異級別 R_3 =[16, 31]，可知嵌入此區塊的機密訊息之十進位值為 $b = (g_i' + g_{i+1}') \bmod 2^{t_k} = (102+119) \bmod 2^4 = 13$。最後，再將 b =13 轉換成長度為 $t_3 = 4$ 的位元串，即可得到原來嵌入此區塊的機密訊息位元串 1101。



(a) 訊息嵌入



(b) 訊息萃取

圖 2.2　Wang 等人[3]的方法簡例

在[3]中，Wang 等人經由實驗證明他們的作法所產生的偽裝影像之品質比 Wu 與 Tsai[4]的結果平均約高 3dB。至於嵌入的容量則和 Wu 與 Tsai[4]作法相當。此外，Wang 等人[3]的作法也可抵擋 Fridrich 等人[6]的 RS 統計偵測攻擊。

2.3 RS 統計偵測法之回顧

本節我們將以一個灰階無失真壓縮影像格式(即 bmp 格式)來說明由 Fridrich 等人[6]所提出的 RS 偽裝偵測技術的原理及作法。

首先將影像中每 n 個相鄰的像素當作一個像素群組 (group)，$G = (x_1, x_2, \ldots, x_n)$，$x_i \in \{0,1,\ldots,255\}$。接著，採用兩種函數來將所有像素群組分類，一個是鑑別函數(discrimination function) f，另一個是翻轉函數(flipping function) F。利用這二個函數可將群組分類為規則(regular)群組、奇異(singular)群組與無用(unusable)群組。

鑑別函數 f 主要用以鑑別像素群組的平滑性(smoothness)。如果像素群組 G 中所含的雜訊愈多(例如：該像素群組所嵌入的訊息位元愈多的時候)，則像素群組的鑑別函數值便會增加。例如，若以像素群組中各像素的差異(variation)作為群組平滑性度量，則鑑別函數 f 可以定義如下：

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{n-1} \left| x_{i+1} - x_i \right|, \ 0 \le x_i \le 255. \tag{2.10}$$

翻轉函數 F 主要用以模擬將一像素的 LSB 位元進行翻轉(即 $0 \leftrightarrow 1$)的操作。它通常是像素值 {0, 1, 2,…, 255} 的一個 2-循環排列，所以通常具有 $F(F(x)) = x$，$x_i \in \{0,1,2,\ldots,255\}$ 或 $F^2 = Identity$ 之特性。RS 統計偵測技術使用以下三種翻轉函數：

(1) 翻轉函數 F_1：

$0 \leftrightarrow 1,\ 2 \leftrightarrow 3,\ \cdots,\ 252 \leftrightarrow 253,\ 254 \leftrightarrow 255.$

(2) 平移翻轉函數 F_{-1}：

$-1 \leftrightarrow 0,\ 1 \leftrightarrow 2,\ \cdots,\ 253 \leftrightarrow 254,\ 255 \leftrightarrow 256$

，或是表示為 $F_{-1}(x) = F_1(x+1) - 1$。

(3) 不翻轉函數：

$F_0(x) = x,\ \forall x \in \{0,\ 1,\ \cdots,\ 255\}$。

根據鑑別函數 f 與翻轉函數 F，RS 統計偵測法將所有的像素群組分類為以下三種群組：

(1) 規則群組 R (Regular Group)：

$G \in R \Leftrightarrow f(F(G)) > f(G)$ 。

(2) 奇異群組 S (Singular Group)：

$G \in S \Leftrightarrow f(F(G)) < f(G)$ 。

(3) 無用群組 U (Unusable Group)：

$G \in U \Leftrightarrow f(F(G)) = f(G)$ 。

其中 $F(G) = (F(x_1), F(x_2), \cdots, F(x_n))$ 表示利用翻轉函數 F 對群組 G 的每一像素進行翻轉處理，$f(F(G))$ 表示翻轉處理後的鑑別函數 f 值。另外，我們也可以對群組 G 的不同像素進行不同的翻轉操作。此時，可以令 $M = [M(1), M(2), \cdots, M(n)]$ 為一翻轉遮罩 (flipping mask)，$M(i) \in \{-1, 0, 1\}$，則 $F_M(G) = (F_{M(1)}(x_1), F_{M(2)}(x_2), \cdots, F_{M(n)}(x_n))$ 並且以 R_M 與 S_M 分別表示翻轉處理後，規則群組與奇異群組所占全部群組的比率。例如，假設 $G = (x_1, x_2, x_3, x_4)$ 且 $M = [0, 1, 1, 0]$，則 $F_M(G) = (F_0(x_1), F_1(x_2), F_1(x_3), F_0(x_4))$ 使用不翻轉函數 F_0 來處理 x_1 與 x_4，使用翻轉函數 F_1 來處理 x_2 與 x_3。

Fridrich 等人[6]指出，在一般沒有藏入機密訊息的掩護影像中，R_M 與 R_{-M} 以及 S_M 與 S_{-M} 的期望值近似相等:

$$R_M \cong R_{-M} \text{ 與 } S_M \cong S_{-M} \qquad (2.11)$$

但是，隨機的翻轉各像素的 LSB 位元後(例如在各像素的 LSB 位元嵌入雜訊)，便會破壞此一統計特性。RS 偵測法就是根據此一特性來偵測像素的 LSB 位元是否有嵌藏機密訊息。

RS 偵測法：偵測嵌入訊息的存在

輸入：

一個特定的嵌入作法 E，與一個掩護影像 I。

輸出：

0 或 1 (0 表示無法偵測到嵌入法 E 所產生的偽裝影像中藏有機密訊息；1 表示可以偵測)。

步驟：

1. 利用嵌入法 E，以 5%的增量依次嵌入不同百分長度的訊息到受測的掩護影像 I 中。(嵌入訊息的百分長度=(嵌入的位元數/掩護影像的像素數)×100%)
2. 使用翻轉遮罩 $M = [0, 1, 1, 0]$ 與 $-M = [0, -1, -1, 0]$，對每次嵌入訊息後所得的影像計算 R_M、R_{-M}、S_M、與 S_{-M} 之值，同時檢驗數學式(2.11)的統計假設是否成立。
3. 如果每次嵌入訊息後，步驟 2 的檢驗都成立，則輸出 0；否則輸出 1。

例如，令 E 為傳統的空間域 LSB 嵌入法且 I 為一 512×512 的灰階 Lena 影像。若依據上述步驟進行嵌入機密訊息與偵測的實驗同時繪製 R_M、R_{-M}、S_M、與 S_{-M} 之圖形(稱為 RS 曲線圖)，結果如圖 2.3 所示，一旦嵌入任何數量的雜訊，數學式(2.11)的統計假設 $R_M \cong R_{-M}$ 與 $S_M \cong S_{-M}$ 便不成立，RS 統計偵測法可以立即偵測出嵌入訊息的存在性。

RS 偵測法還可以進一步相當可靠的估計出 LSB 嵌入法所嵌入訊息的百分長度，其所根據的詳細原理與作法如下請參閱參考文獻[6]或[12]。

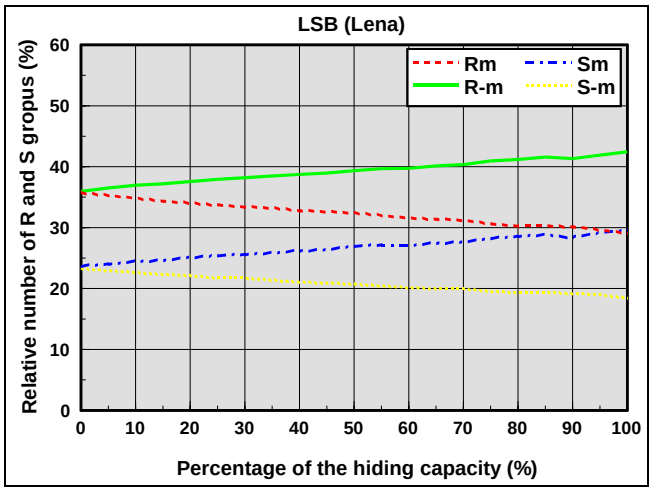


圖 2.3 傳統空間域 LSB 嵌入法的 RS 偵測曲線圖 (x-軸表嵌入訊息的百分長度，y-軸表規則群組與奇異群組的百分數)

三、我們的改良作法

在這一節，我們首先提出〝修改權值〞的觀念，然後據以提出一個改良的作法來進一步改善像素值差異嵌入技術的偽裝影像品質以及嵌入容量，並在第 4 節以實驗證明，當嵌入的秘密訊息數量相同時，我們的作法所產生的偽裝影像比

Wang 等人[3]的結果具有更好的影像品質，也即具有更高的 PSNR 值(Peak-Signal -to-Noise Ratio)。尤其當嵌入容量逐漸增大到令 Wang 等人的結果之PSNR值降低到無法滿足不可察覺性的要求時，我們的作法所產生的偽裝影像仍然具有相當安全的 PSNR 值。

3.1 我們的想法

我們的演算法與 Wu 與 Tsai[4]以及 Wang 等人[3]的作法一樣的，以灰階的影像當作掩護影像、以 0 與 1 位元串流表示嵌入的機密訊息、並將像素差異值的值域[0, 255]分級如下：

$$[0,255]=\bigcup_{k=1}^{n} R_k \ ,$$

其中 $R_k=[l_k,\ u_k]$，$l_1=0,\ u_n=255,\ u_k=l_{k+1}-1$，$|R_k|=u_k-l_k+1=2^{t_k},\ \ 1\le k\le n$，且 $0\le t_1\le t_2\le\cdots\le t_n\le 8$。

考慮由相鄰的兩個像素所組成的一個區塊並假設其像素值為 $(g_i,\ g_{i+1})$、像素值差異為 $d=g_i-g_{i+1}$、像素值差異級別為 R_k(即 $|d|\in R_k$)、而且 R_k 的寬度 $|R_k|=2^{t_k}$，其中 $t_k=t$。則根據像素值差異嵌入技術的作法得知，可以嵌入該區塊的秘密訊息長度為 $t_k=t$ 位元。設此 t-位元嵌入訊息的十進位值為 b，則我們的改良想法與嵌入作法如下：首先決定一個修改權值 C，然後藉由修改 $(g_i,\ g_{i+1})$ 為 $(g_i',\ g_{i+1}')=(g_i+x,\ g_{i+1}+y)$ 使得 $b=(C\cdot g_i'+g_{i+1}')\bmod 2^t=[C\cdot(g_i+x)+(g_{i+1}+y)]\bmod 2^t$ 來嵌入秘密訊息。

所以，與 Wang 等人[3]的作法不同的是，我們擬採用修改權值 C 來降低對掩護區塊像素值 $(g_i,\ g_{i+1})$ 的修改量 $(x,\ y)$，以使由嵌入訊息所造成的失真度(distortion)為最小。

令 $F=(C\cdot g_i+g_{i+1})\bmod 2^t$ 且 $E=(b-F)\bmod 2^t$，則 $(x,\ y)$ 滿足 $E=(C\cdot x+y)\bmod 2^t$，而且為了使嵌入訊息後所造成的失真度為最小，$(x,\ y)$ 必須同時使 x^2+y^2 為最小。

因為，模 2^t 的運算，所以 C 與 E 的值均介於 0與 2^t-1 間，其中 $1\le t\le 8$。所以，對 C 的每一個可能值，我們可以利用窮盡法(exhaustive method)很容易的求出在不同 E 值時滿足 $E=(C\cdot x+y)\bmod 2^t$ 而且 x^2+y^2 為最小的最佳修改量 $(x,\ y)_E$ 之值。令 $T_C=\sum_{E=0}^{2^t-1}(x^2+y^2)_E$，則使 T_C 值最小的 C，即為最佳的修改權值。最佳修改權值的決定過程可以用底下的演算法詳細說明之。

演算法：最佳修改權值 C 的求法

輸入：

整數 t，$1\le t\le 8$，表根據像素值差異所決定的嵌入訊息的位元數。

輸出：

最佳修改權值 C。

步驟：

1. $T_C\leftarrow 0,\ S_C\leftarrow\phi$
2. For $C=0$ to 2^t-1 do
   For $E=0$ to 2^t-1 do
   (1) 利用窮盡法(exhaustive method)求出滿足 $E=C\cdot x+y \bmod 2^t$ 而且 x^2+y^2 為最小的 $(x,y)_E$之值。//註:因為$1\le t\le 8$，所以窮盡法可行。//
   (2) $T_C\leftarrow T_C+(x^2+y^2)_E$
   End for
   $S_C=S_C\cup\{T_C\}$
   End for
3. 求出S_C中最小的T_C, 然後輸出 C。

例如當 t=3 時，T_2=12(見表 3.1)而 T_5=10(見表 3.2)。事實上，C=5 為 t=3 時的一個最佳修改權值。

表 3.1 與表 3.2 (置於參考文獻之後)

將 t 的可能值 1, 2, 3, …, 8 分別輸入上述之演算法，便可求出針對不同的嵌入位元長度 t 之最佳修改權值 C 的值，如表 3.3 所示。因為對應於每一 t 值的最佳 C 值並非唯一，表 3.4 是我們

建議使用的最佳修改權值。

表 3.3 與表 3.4 (置於參考文獻之後)

為了能夠順利的萃取出嵌入的機密訊息，我們必須要求偽裝區塊之像素差異絕對值 $|d'|=|g'_i - g'_{i+1}|$ 與原始區塊之像素差異絕對值 $|d|=|g_i - g_{i+1}|$ 屬於同一差異級別 $R_k=[l_k, u_k]$。如此，則由 R_k 便可以得知在嵌入作業時所使用的最佳修改權值 C 以及嵌入的機密訊息位元數 t_k 之值。假設 $t_k = t$，於是可以先計算出嵌入訊息的十進位值 $b=(C\cdot g'_i + g'_{i+1}) \bmod 2^t$，再將 b 轉換成長度為 t 的位元串，即可得到原來嵌入此區塊的機密訊息位元串。

因為 $d'_i = g'_i - g'_{i+1} = (g_i + x) - (g_{i+1} + y) = g_i - g_{i+1} + x - y = d_i + x - y$，為了使 $|d'|$ 與 $|d|$ 屬於相同的級別 R_k，像素值修改量(x, y)必須進一步滿足：

$$\begin{cases} l_k \le d_i + x - y \le u_k & \text{當 } d_i \ge 0\text{時,} \\ -u_k \le d_i + x - y \le -l_k & \text{當 } d_i < 0\text{時。} \end{cases}$$

$$\Rightarrow \begin{cases} l_k - d_i \le x - y \le u_k - d_i & \text{當 } d_i \ge 0\text{時,} \\ -u_k - d_i \le x - y \le -l_k - d_i & \text{當 } d_i < 0\text{時。} \end{cases}$$

綜合以上的說明可知，我們的改良作法在嵌入機密訊息時對原始像素值 (g_i, g_{i+1}) 所作的修改量(x, y) 必須滿足在

$$\begin{cases} E = C\cdot x + y \bmod 2^{t_k}, \\ l_k - d_i \le x - y \le u_k - d_i & \text{當 } d_i \ge 0\text{時} \\ -u_k - d_i \le x - y \le -l_k - d_i & \text{當} d_i < 0\text{時} \end{cases}$$

的條件下，使 $x^2 + y^2$ 之值為最小。

3.2 我們的訊息的嵌入與萃取演算法

3.2.1 訊息的嵌入演算法

輸入：

CI 為一個灰階的掩護影像；$SM = b_1 b_2 \cdots b_n$, $b_i \in \{0, 1\}$, 為欲嵌入 CI 的機密訊息位元串；$\{R_k = [l_k, u_k] \mid 1 \le k \le n\}$ 為選用的像素值域[0, 255]之分段，其中 $|R_k| = u_k - l_k + 1 = 2^{t_k}, 1 \le k \le n$, 且 $0 \le t_1 \le t_2$ $0 \le t_1 \le t_2 \le \cdots \le t_n \le 8$；WT 為我們建議使用的最佳修改權值表(表 3.4)。

輸出：

SI 為嵌入機密訊息後的偽裝影像。

步驟：

1. (1) 將掩護影像分割為由兩個相鄰像素所組成的若干個區塊。
   (2) 以一亂數種子與一擬亂數列產生器決定各個區塊的處理順序。
2. 依序對第 i 個區塊(假設其中兩個像素的灰階值為 g_i 和 g_{i+1})執行如下的訊息嵌入處理。
   (1) (i) 計算兩像素的差異值 $d = g_i - g_{i+1}$。
   (ii) 決定 $|d|$ 所屬之區段 $R_k = [l_k, u_k]$ 以及可以嵌入此區塊的機密訊息位元數 t_k。
   (iii) 依序自尚未嵌入的機密訊息中讀取出 t_k 個位元並計算其十進位值 b。
   (iv) 根據 t_k 的值以及表 WT，決定最佳修改權值 C。
   (2) (i) 計算

$$F = (C \cdot g_i + g_{i+1}) \bmod 2^{t_k} \qquad (3.1)$$

與 $E = (b - F) \bmod 2^{t_k}$ (3.2)

之值。

(ii) 利用窮盡法(exhaustive method) 求解在

$$\begin{cases} E = C\cdot x + y \bmod 2^{t_k}, \\ l_k - d_i \le x - y \le u_k - d_i & \text{當 } d_i \ge 0\text{時,} \\ -u_k - d_i \le x - y \le -l_k - d_i & \text{當 } d_i < 0\text{時。} \end{cases} \qquad (3.3)$$

的條件下，使 $x^2 + y^2$ 之值為最小的(x, y)之值。//註:因為 $1 \le t_k \le 8$，所以窮盡法可行//

(3) 若 $0 \le g_i + x$與 $g_{i+1} + y \le 255$(意即沒有造成溢位)則將像素值 (g_i, g_{i+1}) 修改為 $(g'_i, g'_{i+1}) = (g_i + x, g_{i+1} + y)$ 完成嵌入作業。否則，執行步驟 3 來修正 (g'_i, g'_{i+1}) 之值，解決溢位問題完成嵌入作業。

3. Case 1:

若 $g_i \approx 0$ ， $g_{i+1} \approx 0$ 而且 $g'_i < 0$ or $g'_{i+1} < 0$, 則將(g'_i, g'_{i+1})修正為

$$(g''_i, g''_{i+1}) = (g'_i + \lfloor (2^{t_k-1})/C \rfloor,\ g'_{i+1} + 2^{t_k-1} + (2^{t_k-1} \bmod C))$$

Case 2:

若 $g_i \approx 255$ ， $g_{i+1} \approx 255$ 而且 $g'_i > 255$ or $g'_{i+1} > 255$, 則將(g'_i, g'_{i+1})修正為

$$(g''_i, g''_{i+1}) = (g'_i - \lfloor (2^{t_k-1})/C \rfloor,\ g'_{i+1} - 2^{t_k-1} - (2^{t_k-1} \bmod C))$$

Case 3:

若 g_i 與 g_{i+1} 反差大於 128(即 $|d_i| = |g_i - g_{i+1}| > 128$)，而且

Case 3-1:

若 $g'_i < 0$ 且 $g'_{i+1} \geq 0$，則將(g'_i, g'_{i+1})修正為

$$(g''_i, g''_{i+1}) = (0,\ (Cg'_i + g'_{i+1}) \bmod 2^{t_k})$$

Case 3-2:

若 $g'_i \geq 0$ 且 g'_{i+1}<0 ，則將(g'_i, g'_{i+1})修正為

$$(g''_i, g''_{i+1}) = (g'_i + \lfloor g'_{i+1}/C \rfloor,\ g'_{i+1} \bmod C)$$

Case 3-3:

若 g'_i>255 且 $g'_{i+1} \geq 0$ ，則將(g'_i, g'_{i+1})修正為

$$(g''_i, g''_{i+1}) = (255,\ g'_{i+1} + (g' - 255) \times C)$$

Case 3-4:

若 $g'_i \geq 0$ 且 g'_{i+1}>255，則將(g'_i, g'_{i+1})修正為

$$(g''_i, g''_{i+1}) = (g'_i + \lceil (2^{t_k-1})/C \rceil + 1,\ g'_{i+1} - 2^{t_k-1} - [C - (2^{t_k-1} \bmod C)])$$

//註:因為 $E = C \cdot x + y \bmod 2^{t_k}$ 而且 $x^2 + y^2$ 之值最小，所以 $|x| \leq \lfloor 2^{t_k-1}/C \rfloor$ 且 $|y| \leq 2^{t_k-1}$ ，也即 $|g'_i - g_i| \leq \lfloor 2^{t_k-1}/C \rfloor$ 且 $|g'_{i+1} - g_{i+1}| \leq 2^{t_k-1}$ 。於是，修正後的 g''_i 與 g''_{i+1} 不會產生溢位。//

3.2.2 訊息的萃取演算法

輸入：

SI 為一個根據我們的嵌入作法得到的偽裝影像；$\{R_k = [l_k, u_k] | 1 \leq k \leq n\}$ 為嵌入作業使用的像素值域之分段； WT 為我們建議使用的最佳修改權值表(表 3.4) 。

輸出：

$SM = b_1 b_2 \cdots b_n$, $b_i \in \{0, 1\}$, 為嵌藏在 ***SI*** 的機密訊息位元串。

步驟：

1. (1) 將偽裝影像分割為由兩個相鄰像素所組成的若干個區塊。
   (2) 使用與嵌入作業相同的亂數種子與擬亂數列產生器決定各個區塊的處理順序。
2. 依序對第 i 個區塊(假設其中兩個像素的灰階值為g'_i和g'_{i+1})執行如下的訊息萃取處理。
   (1) (i) 計算兩像素的差異值 $d' = g'_i - g'_{i+1}$ 。
   (ii) 決定 $|d'|$ 所屬之區段 $R_k = [l_k, u_k]$ 以及可以嵌入此區塊的機密訊息位元數 t_k。
   (iii) 根據 t_k 的值與表 WT，決定最佳修改權值 C 。
   (2) (i) 計算 $b = (C \cdot g'_i + g'_{i+1}) \bmod 2^{t_k}$ 。 (3.4)
   (ii) 將 b 轉換成長度為 t_k 的位元串，即可得到原來嵌入此區塊的機密訊息位元串。

3.3 簡例

為了進一步說明我們的演算法，我們以一個掩護區塊 (g_i, g_{i+1}) =(98, 116)以及如下的像素值域分段：R_1=[0, 7], R_2=[8, 15], R_3=[16, 31], R_4=[32, 63], R_5=[64, 127], R_6=[128, 255]為例來嵌入及取出秘密訊息 M=1101_2 如下：

訊息嵌入：

(1)(i) 計算二個像素的差異值為 $d = 98 - 116 = -18$ 。
(ii) 決定 $|d|$ =18 所屬之區段為 $R_3 = [16, 31]$。因為 $|R_3| = 31 - 16 + 1 = 16 = 2^4$ ，可以嵌入此區塊的機密訊息位元數為 t_3 =4。
(iii) 依序自尚未嵌入的機密訊息中讀取出 t_3 =4 個位元，設為1101_2 ，並計算其十

進位值 b=13。

(iv) 根據 t_3=4 的值以及表 WT，決定最佳修改權值 C =6。

(2) (i) 計算

$$F = (C \cdot g_i + g_{i+1}) \bmod 2^{t_3}$$
$$= (6\times 98 + 116) \bmod 2^4 = 0$$

與

$$E = (b - F) \bmod 2^{t_3}$$
$$= (13 - 0) \bmod 2^4 = 13$$

。

(ii) 因為 $d = -18 < 0$，利用窮盡法求解在

$$\begin{cases} E = C \cdot x + y \bmod 2^{t_k} \\ -u_k - d \le x - y \le -l_k - d \end{cases}$$

的條件下，也即在

$$\begin{cases} 7 = 6 \cdot x + y \bmod 2^4, \\ -5 \le x - y \le 10 \end{cases}$$

的條件下，使 $x^2 + y^2$ 之值為最小的 $(x, y)=(2, 1)$。

(3) 因為 $g_i + x = 98 + 2 = 100$ 與 $g_{i+1} + y = 116 + 1 = 117$ 均未產生溢位，所以將原像素值 (g_i, g_{i+1}) 修改為 $(g_i', g_{i+1}') = (g_i + x, g_{i+1} + y) = (100, 117)$ 完成嵌入程序。

訊息萃取：

(1) (i) 計算兩像素的差異值

$d' = g_i' - g_{i+1}'$ =100-117=-18。

(ii) 決定 $|d'|$ 所屬之區段為 R_3 =[16, 31]。因為 $|R_3| = 31 - 16 + 1 = 16 = 2^4$，嵌入此區塊的機密訊息位元數為 t_3 =4。

(iii) 根據 t_3 =4 與表 WT，決定最佳修改權值 C =6。

(2) (i) 計算

$$b = (C \cdot g_i' + g_{i+1}') \bmod 2^{t_3}$$
$$= (6\times 100 + 117) \bmod 2^4 = 13$$

。

(ii) 將 b=13 轉換成長度為 t_3 =4 的位元串，即可得到原來嵌入此區塊的機密訊息位元串為 M=1101_2 。

圖 3.1 我們的作法簡例
(置於參考文獻之後)

在 2.1 節與 2.2 節，我們曾分別使用 Wu 與 Tsai[4]的作法以及 Wang 等人[3]的作法對相同的掩護區塊 (g_i, g_{i+1}) =(98, 116)嵌入相同的機密訊息 M=1101_2 (見 2.1 節圖 2 與 2.2 節圖 3)。表 3.5 比較三種作法對此掩護區塊所作的修改量的平方和。

表 3.5 修改量的平方和之比較
(置於參考文獻之後)

由上述的演算法與例子可以知道我們新改良的像素值差異嵌入作法的觀念相當簡單，而且經由最佳修改權值的應用可以儘量降低對掩護區塊像素所作的修改量。由此不難想見，我們的作法所產生的偽裝影像會比 Wu 與 Tsai[4]以及 Wang 等人[3]的結果具有更好的影像品質，也即具有更高的 PSNR 值 (Peak-Signal-to-Noise Ratio)。在第 4 節，我們將進行一連串的實驗來支持我們的看法。

四、實驗與結果

在本節裡，我們將經由一系列的實驗來比較我們的作法、Wu 與 Tsai[4] 的作法、以及 Wang 等人[3]的作法之不可察覺性(imperceptibility)、最大嵌入容量(maximum embedding capacity)、與統計之不可偵測性(non-detectability)。

我們的實驗環境軟硬體、掩護影像、嵌入訊息、以及使用的修改權值表如下：

實驗環境硬體：

Lenovo T61。CPU：Intel Core2 Duo T7300 @ 2.00GHz，RAM：1.96GB。

實驗環境軟體：

作業系統：Windows XP Professional Service Pack 3。

應用程式：

Visual C# 2005。

掩護影像：

大小為 512×512 的 8 位元灰階影像：

“Lena”、“Jet”、“Baboon”、“Barbara”、“Peppers”、以及“Tiffany”(見圖 4.1)。

嵌入訊息：

隨機產生的 0 與 1 數列。

修改權值表：

採用第 3 節所建議的最佳修改權值表(表 3.4)如下。

圖 4.1 實驗用的 512×512 灰階掩護影像
(置於參考文獻之後)

表 4.1 實驗用的最佳修改權值表
(置於參考文獻之後)

4.1 不可察覺性(Imperceptibility)實驗

實驗內容：

分別利用 Wu 與 Tsai[4]的嵌入法、Wang 等人[3]的嵌入法，以及我們的嵌入法，計算偽裝影像的 PSNR 值來比較不同作法所產生的偽裝影像品質與不可察覺性(imperceptibility)。所採用的像素值域分段(或分級)如下：R_1=[0, 7], R_2=[8, 15], R_3=[16, 31], R_4=[32, 63], R_5=[64, 127], R_6=[128, 255]。

實驗結果：

表 4.2 與表 4.3 的實驗數據說明，在相同的嵌入容量下我們的嵌入法所產生的偽裝影像之 PSNR 值比 Wu 與 Tsai[4]的嵌入法還要高出約 7~11dB，而比 Wang 等人[3]的嵌入法還要高出約 4~8dB。很明顯地，我們的方法具有比 Wu 與 Tsai[4]以及 Wang 等人[3]的作法更好的不可察覺性。

表 4.2 不可察覺性實驗(我們的作法與 Wu 與 Tsai[4]的方法之比較)

表 4.3 不可察覺性實驗(我們的作法與 Wang 等人[3]的方法之比較)
(置於參考文獻之後)

4.1 最大嵌入容量(Maximum embedding capacity)實驗

實驗內容：

在本實驗中，為了表示的方便，我們將以分段的長度系列來表示像素值域分段(或分級)。例如若像素值域分段為 R_1=[0, 7], R_2=[8, 15], R_3=[16, 31], R_4=[32, 63], R_5=[64, 127], R_6=[128, 255]；則以遞增的分段長度系列 $\{|R_1|, |R_2|,\cdots, |R_6|\}$={8, 8, 16, 32, 64, 128}表示之。因為分段數愈少(即分段長度愈大)，則每次可嵌入的訊息位元數愈多，所以整體可以嵌入的機密訊息量就愈大。本實驗利用不同的像素值域分段來比較，當分段長度逐漸增大時，我們的嵌入法、Wu 與 Tsai[4]的作法、以及 Wang 等人[11]的作法的最大嵌入容量與影像品質之間的關係。

實驗結果：

由表 4.4 與表 4.5 的實驗數據，可知我們的嵌入作法除了具有比 Wu 與 Tsai[4]以及 Wang 等人[3]的結果更好的偽裝影像品質外，當嵌入容量逐漸增大到令 Wu 與 Tsai[4]以及 Wang 等人[3]的結果之 PSNR 值降低到無法滿足不可察覺性的要求(PSNR 值<30)時，我們的作法所產生的偽裝影像仍然具有相當安全的 PSNR 值(見表 4.4 與表 4.5 的陰影部結果)。

表 4.4 最大嵌入容量實驗(我們的作法與 Wu 與 Tsai[4]的方法之比較)

表 4.5 最大嵌入容量實驗(我們的作法與 Wang 等人[3]的方法之比較)
(置於參考文獻之後)

4.1 不可偵測性(Non-detectability)實驗

實驗內容：

使用{8，8，16，32，64，128}的像素值域分段與 M =[0, 1, 1, 0]的像素群組遮罩，再利用我們的嵌入法以 5%的增量依次嵌入不同百分長度的訊息到 Lena 與 Baboon 掩護影像，對每次嵌入後的偽裝影像以 RS 偵測法計算並檢驗數學式(2.11)的統計假設 $R_M \cong R_{-M}$ 與 $S_M \cong S_{-M}$ 是否成立。如果成立，則表示 RS 偵測法無法偵測出我

們的偽裝影像中有秘密訊息的存在。

實驗結果：

RS 偵測法無法偵測出我們的偽裝影像中有秘密訊息的存在(見圖 4.2 與圖 4.3)。

圖 4.2 我們的嵌入法的 RS 圖(Lena)
圖 4.3 我們的嵌入法的 RS 圖(Baboon)
(置於參考文獻之後)

五、結論

像素值差異技術(pixel-value differencing technique; PVD) 是一個相當簡單而且有效的影像偽裝術。自從 2003 年，Wu 與 Tsai[4]提出該技術的觀念和作法後，有許多改良的作法陸續的被提出來。其中，Wang 等人[3]在 2008 年利用模運算提出的改良的作法，可以在嵌入訊息時降低像素值的修改量，大幅的提高了像素值差異法所產生的偽裝影像之品質，也即大幅的提高了像素值差異法的不可察覺性(imperceptibility)。

由於一個真正實用有效的影像偽裝術，除了要具備難以察覺與難以偵測的安全性外，尚須要求具有盡可能高的嵌入容量；本論文針對 Wang 等人的作法，提出〝修改權值〞的觀念來進一步改善像素值差異嵌入技術的偽裝影像品質以及嵌入容量，同時進行一系列的實驗來比較我們的作法、Wu 與 Tsai[4] 的作法、以及 Wang 等人[3]的作法之不可察覺性(imperceptibility)、不可偵測性(non-detectability) 以及最大的嵌入容量(maximum embedding capacity)。

實驗的結果證明，我們的方法有下列幾項優點：

1. 觀念簡單而且計算容易，。
2. 有效的解決嵌入溢位的問題，所有掩護區塊都可嵌入機密訊息。
3. 從偽裝影像萃取機密訊息時，不必參考原始的掩護影像。
4. 在相同的嵌入容量下，我們的方法所產生的偽裝影像比 Wu 與 Tsai[4]以及 Wang 等人的結果更好具有更高的 PSNR 值(Peak-Signal-to-Noise Ratio)；意即我們的方法具有比 Wu 與 Tsai[4]以及 Wang 等人[3]的作法更好的不可察覺性。
5. 當嵌入容量逐漸增大到令 Wu 與 Tsai[4]以及 Wang 等人[3]的結果之 PSNR 值降低到無法滿足不可察覺性的要求(PSNR 值<30)時，我們的作法所產生的偽裝影像仍然具有相當安全的 PSNR 值。換言之，在安全的前提下，我們的方法容許比 Wu 與 Tsai[4]以及 Wang 等人[3]的方法更大的嵌入容量。
6. 我們的方法與 Wu 與 Tsai[4]以及 Wang 等人[3]的方法一樣的，也可以抵擋由 Fridrich 等人[6]於 2001 年所提出的 RS 統計偵測攻擊。

六、參考文獻

[1] C. H. Yang, and C. Y. Weng, “A Steganographic Method for Digital Images by Multi-Pixel Differencing,” *International Computer Symposium*, ICS, pp. 611-615, 2006.

[2] C. H. Yang, S. J. Wang, and C. Y. Weng, “Analyses of Pixel-Value-Differencing Schemes with LSB Replacement in Stegonagraphy,” *Intelligent Information Hiding and Multimedia Signal Processing*, IIHMSP, Vol. 1, pp. 445-448, 2007.

[3] C. M. Wang, N. I. Wu, C. S. Tsai, and M. S. Hwang, “A high quality steganographic method with pixel-value differencing and modulus function,” *Journal of Systems and Software*, Vol. 81, Issue 1, pp. 150-158, 2008.

[4] D. C. Wu, and W. H. Tsai, “A steganographic method for images by pixel-value differencing,” *Pattern Recognition Letters*, Vol. 24, Issue 9-10, pp. 1613-1626, 2003.

[5] H. C. Wu, N. I. Wu, C. S. Tsai, and M. S. Hwang, “Image steganographic scheme based on pixel-value differencing and LSB replacement methods,” *IEE Proceedings.-Vision, Image and Signal Processing*, Vol. 152, Issue 5, pp. 611-615, 2005.

[6] J. Fridrich, M. Goljan, and R. Du, “Reliable Detection of LSB Steganography in Grayscale and Color Images,” *Proceedings of the ACM Workshop on Multimedia and Security*, pp. 27-30, 2001.

[7] K. C. Chang, C. P. Chang, P. S. Huang, and T. M. Tu, “A Novel Image Steganographic Method Using Tri-way Pixel-Value Differencing,” *Journal of Multimedia*, Vol. 3, No. 2, 2008.

[8] K. J. Kim, K. H. Jung, and K. Y. Yoo, “A High Capacity Data Hiding Method using PVD and LSB,” *International Conference on Computer Science and Software Engineering*, Vol. 3, pp. 876-879, 2008.

[9] M. M. Amin, M. Salleh, S. Ibrahim, and M. Z. I. Shamsuddin, “Information Hiding using Steganography,” *IEEE National Conference on Telecommunication Technology*, pp. 21-25, 2003.

[10] R. J. Anderson, and F. A. P. Petitcoals, “On The Limits of Steganography,” *IEEE Journal of Selected Areas in Communication*, Vol. 16, No. 4, pp. 474-481, 1998.

[11] W. Bender, D. Gruhl, N. Morimoto and A. Lu, “Techniques for data hiding,” *IBM System Journal*, Vol. 35, No.3&4, 1996.

[12] Y. S. Chen, C. Y. Gun, H. F. Lin, and C. Y. Chen, “An Improved Steganographic Method for Images by Pixel-value Differencing,” *Proceedings of 2008 International Conference on Advanced Information Technologies, AIT*, Apr. 2008.

==========

表 3.1 $t=3$ 時 $T_2=12$

E	$(x,\ y)_E$	$(x^2+y^2)_E$
0	(0, 0)	0
1	(0, +1)	1
2	(+1, 0)	1
3	(+1, +1)	2
4	(+2, 0)	4
5	(−1, −1)	2
6	(−1, 0)	1
7	(0, −1)	1
$T_2=\sum_{E=0}^{7}(x^2+y^2)_E=12$		

表 3.2 $t=3$ 時 $T_5=10$

E	$(x,\ y)_E$	$(x^2+y^2)_E$
0	(0, 0)	0
1	(0, +1)	1
2	(−1, −1)	2
3	(−1, 0)	1
4	(+1, −1)	2
5	(+1, 0)	1
6	(+1, +1)	2
7	(0, −1)	1
$T_5=\sum_{E=0}^{7}(x^2+y^2)_E=10$		

表 3.3 嵌入不同位元數的最佳修改權值表

嵌入位元數	1	2	3	4	5	6	7	8
最佳修改權值	1, 2	2	3,5	6,10	7,9,23,25	14,50	12,116	60,196

表 3.4 建議使用的最佳修改權值表(WT)

嵌入位元數	1	2	3	4	5	6	7	8
最佳修改權值	1	2	3	6	7	14	12	60

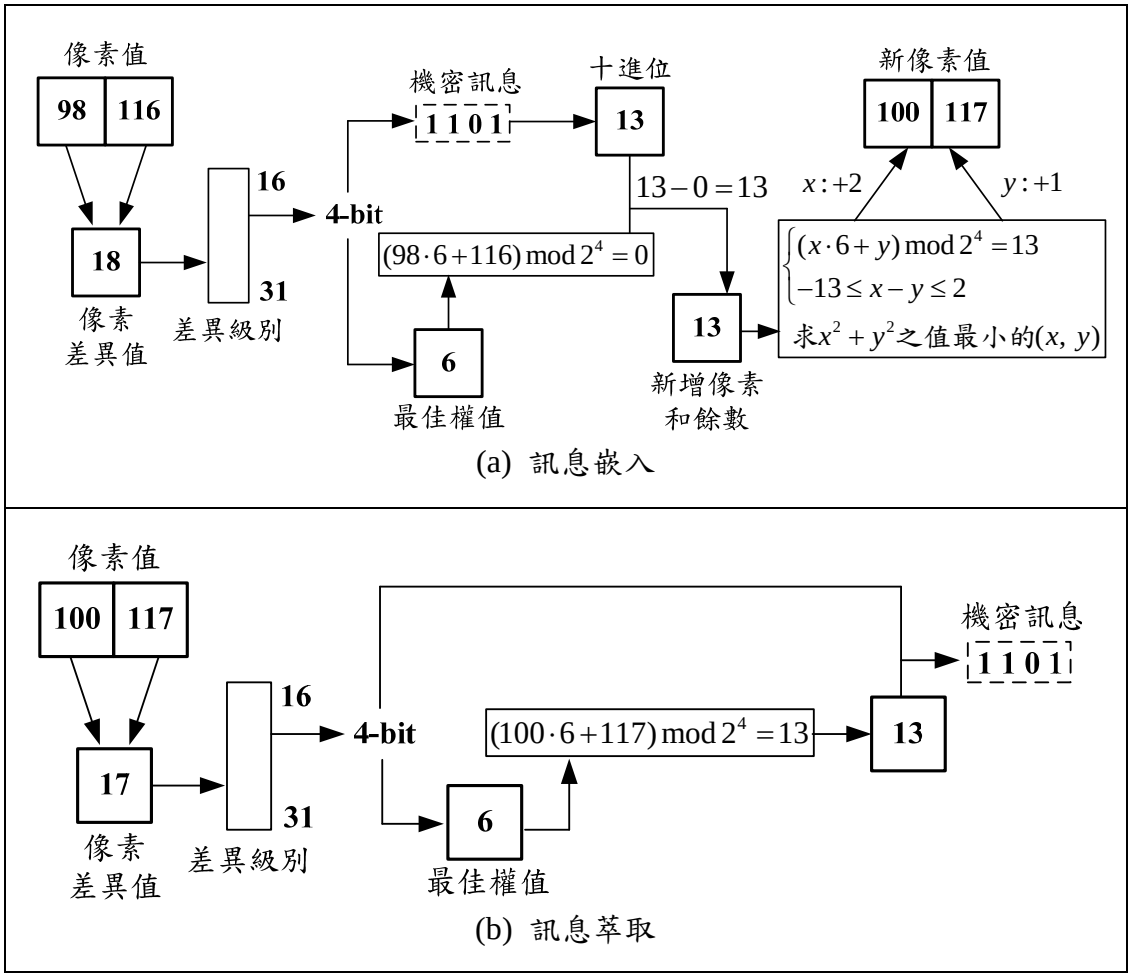


圖 3.1 我們的作用簡例

表 3.5 修改量的平方和之比較

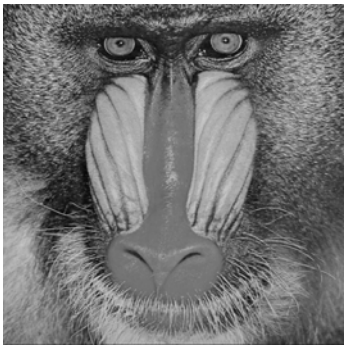
比較	Wu 與 Tsai[4]的作法	Wang 等人[3]的作法	我們的作法
掩護區塊像素值	(98, 116)	(98, 116)	(98, 116)
偽裝區塊像素值	(93, 122)	(102, 119)	(100, 117)
修改量的平方和	$5^2+6^2=61$	$4^2+3^2=25$	$2^2+1^2=5$

表 4.1(即表 3.4) 實驗用的最佳修改權值表

嵌入位元數	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
最佳修改權值	1	2	3	6	7	14	12	60

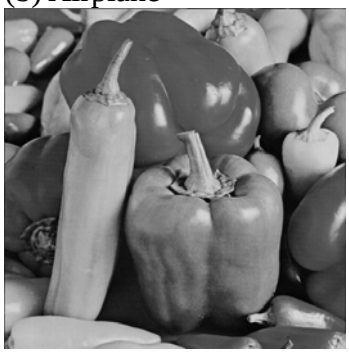
(a) Lena

(b) Airplane



(c) Baboon

(d) Barbara



(e) Peppers

(f) Goldhill

圖 4.1　實驗用的 512×512 灰階掩護影像

表 4.2 不可察覺性實驗(我們的作法與 Wu 與 Tsai[4]的方法之比較)

掩護影像 (512×512)	Wu 與 Tsai[4]的方法			我們的方法		
	Payload (bit)	Payload (BPP)	PSNR (dB)	Payload (bit)	Payload (BPP)	PSNR (dB)
Lena	407,410	1.55	41.50	407,442	1.55	49.21
Airplane	411,625	1.57	40.21	411,658	1.57	48.99
Baboon	459,501	1.75	36.66	459,841	1.75	47.15
Barbara	452,365	1.73	36.14	453,247	1.73	47.23
Peppers	405,502	1.55	41.56	407,260	1.55	49.08
Goldhill	411,877	1.57	41.02	411,896	1.57	48.95

表 4.3 不可察覺性實驗(我們的作法與 Wang 等人[3]的方法之比較)

掩護影像 (512×512)	Wang 等人[3]的方法			我們的方法		
	Payload (bit)	Payload (BPP)	PSNR (dB)	Payload (bit)	Payload (BPP)	PSNR (dB)
Lena	407,442	1.55	44.29	407,442	1.55	49.21
Airplane	411,658	1.57	43.06	411,658	1.57	48.99
Baboon	459,841	1.75	39.91	459,841	1.75	47.15
Barbara	453,247	1.73	39.10	453,247	1.73	47.23
Peppers	407,260	1.55	43.55	407,260	1.55	49.08
Goldhill	411,896	1.57	43.98	411,896	1.57	48.95

表 4.4 最大嵌入容量實驗(我們的作法與 Wu 與 Tsai[4]的方法之比較)

掩護影像 (Lena 512×512)	Wu 與 Tsai[4]的方法			我們的方法		
像素值域分段	Payload (bit)	Payload (BPP)	PSNR (dB)	Payload (bit)	Payload (BPP)	PSNR (dB)
{2, 2, 4, 4, 4, 8, 8, 16, 16, 32, 32, 64, 64}	205,964	0.79	48.32	205,964	0.79	52.27
{8,8,16,32,64,128}	407,410	1.55	41.50	407,442	1.55	49.21
{16,16,32,64,128}	527,694	2.01	37.00	527,726	2.01	46.64
{32,32,64,128}	655,762	2.50	30.54	655,799	2.50	43.89
{64,64,128}	774,234	2.95	23.93	786,434	3.00	40.96
{128,128}	674,086	2.57	18.73	917,504	3.50	38.01
{256}	8,848	0.03	31.75	1,048,576	4.00	35.00

表 4.5 最大嵌入容量實驗(我們的作法與 Wang 等人[3]的方法之比較)

掩護影像 (Lena 512×512)	Wang 等人[3]的方法			我們的方法		
像素值域分段	Payload (bit)	Payload (BPP)	PSNR (dB)	Payload (bit)	Payload (BPP)	PSNR (dB)
{2, 2, 4, 4, 4, 8, 8, 16, 16, 32, 32, 64, 64}	205,964	0.79	50.35	205,964	0.79	52.27
{8,8,16,32,64,128}	407,442	1.55	44.29	407,442	1.55	49.21
{16,16,32,64,128}	527,726	2.01	40.30	527,726	2.01	46.64
{32,32,64,128}	655,799	2.50	34.76	655,799	2.50	43.89
{64,64,128}	786,434	3.00	28.81	786,434	3.00	40.96
{128, 128}	917,504	3.50	22.78	917,504	3.50	38.01
{256}	1,048,576	4.00	16.33	1,048,576	4.00	35.00

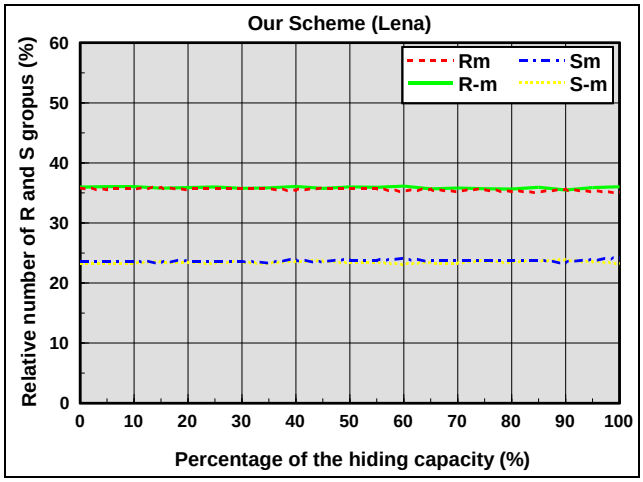


圖 4.2 我們嵌入法的 RS 圖(Lena)

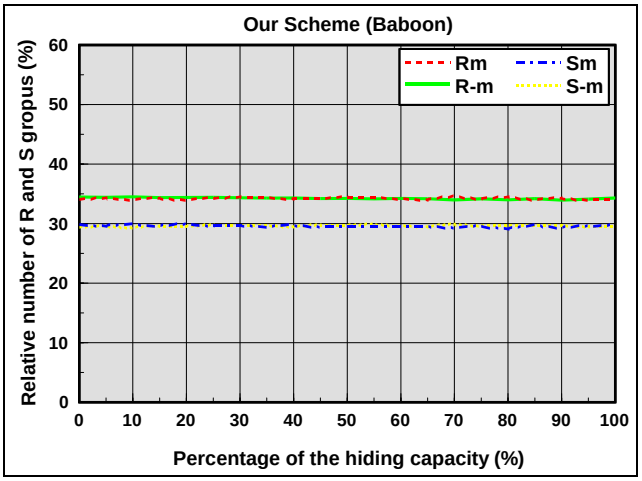


圖 4.3 我們嵌入法的 RS 圖(Baboon)