Steganography scheme based on side match vector quantization

Lee Shu-Teng Chen
Ja-Chen Lin
National Chiao Tung University
Department of Computer Science and Information Engineering
1001 Ta Hsueh Road
Hsinchu, 30050 Taiwan
E-mail: stlee@cs.nctu.edu.tw

Abstract. We propose a steganographic method that is based on side match vector quantization (SMVQ). The secret data are encrypted and then embedded in an SMVQ-decompressed image. In the decoding process, both the hidden secret and the original SMVQ code can be reconstructed without any error. Experimental results demonstrate that the proposed method provides a high hiding capacity and acceptable stego-image quality, to an extent comparable to a well-known SMVQ-based reversible steganographic method. © *2010 Society of Photo-Optical Instrumentation Engineers.* [DOI: 10.1117/1.3366654]

Subject terms: data hiding; triangular inequality; reversible; χ-square analysis.

Paper 090739R received Sep. 22, 2009; revised manuscript received Jan. 8, 2010; accepted for publication Feb. 3, 2010; published online Mar. 30, 2010.

1 Introduction

The transmission of digital media, including text, image, audio, and video, via computer networks is becoming increasingly popular. However, because networks are open environments, transmitted digital signals may be intercepted or distributed by illegal users. Secret data can be protected by using steganographic methods[1–9] to embed the secret data in a cover image, such that attackers cannot identify a hidden secret in the stego-image. Of all steganographic approaches, the simplest is the least significant bit (LSB) substitution scheme, which hides certain bits in the LSBs of the pixels of a cover image. However, the quality of the stego-image becomes worse when the number of used LSBs increases to over three. To mitigate this problem, Wang et al.[1] designed a theoretically optimal LSB scheme that was based on a generic algorithm. Wu and Tsai[2] embedded secret data in the difference value of two consecutive gray values in a cover image. Thien and Lin[3] developed an efficient LSB substitution scheme that was based on a modulus function.

Because multimedia files are usually very large, some well-known image-compression techniques, such as vector quantization[10] (VQ) or Joint Photographic Experts Group[11] (JPEG) have been developed to save storage space. Although VQ is not as well known as JPEG, it has a simple structure, especially in the decoding phase. In the encoding phase of VQ, an image is divided into several nonoverlapping blocks, each of which is replaced by its nearest codeword that is selected from a main codebook that was created earlier using an algorithm such as the Linde–Buzo–Gray algorithm.[12] In VQ, the squared Euclidean distance (SED) between a codeword $M=(m_0,m_1,\ldots,m_{k-1})$ and a block $N=(n_0,n_1,\ldots,n_{k-1})$ is defined as

$$\mathrm{SED}(M,N)=\|M-N\|^2=\sum_{i=0}^{k-1}(m_i-n_i)^2, \quad (1)$$

where k is the block size (e.g., $k=16$ if each block is 4×4 pixels). The codeword M with the lowest $\mathrm{SED}(M,N)$ value is chosen to represent the block N. The VQ index of the chosen codeword is recorded for each block, and the whole image is represented by the VQ indices. Because each block is coded independently of every other, the artifact boundaries between neighboring blocks are clear. Therefore, Kim[13] proposed a side match VQ (SMVQ) approach to improve simultaneously visual quality and compression rate.

Some steganographic schemes that are based VQ or SMVQ have recently been investigated.[4–9] Hu[5] adopted VQ to compress secret images. The generated VQ indices and the parameters that are related to the secret images were all encrypted using the Data Encryption Standard (DES) cryptosystem.[14] The encrypted data were then embedded in a cover image by the LSB substitution scheme. Shie and Lin[6] also employed VQ to compress secret images. The created VQ indices and the parameters that are related to the secret images were embedded in a main codebook using an adaptive LSB substitution scheme. The modified main codebook was further encrypted using the Advanced Encryption Standard cryptosystem.[15] Wang and Tsai[7] utilized Huffman coding to compress the VQ indices of secret images, and the generated Huffman bit stream was hidden in a cover image. Although these hiding methods[5–7] could hide several secret images in a cover image or in a main codebook, the extracted secret images were not error free. To the contrary, Chang et al.[9] modified the codeword selection strategy of SMVQ and embedded secret data in an SMVQ-decompressed cover image. The hidden secret was extracted without any loss, and the SMVQ code of the cover image was also reconstructed.

To improve further both the hiding capacity and stego-image quality in Chang et al.'s method,[9] this work proposes a high-capacity steganographic method that is based on SMVQ. The rest of the paper is organized as follows. Sec-

0091-3286/2010/$25.00 © 2010 SPIE

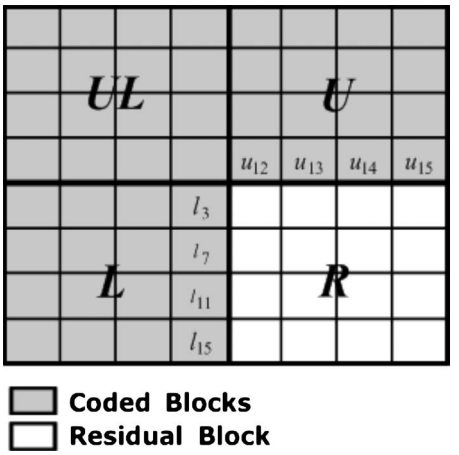

Fig. 1 Generation of local subcodebook for residual block *R* using its upper (*U*) and left (*L*) decoded blocks.

tion 2 reviews pertinent literature. Section 3 then describes the proposed method. Section 4 summarizes the experimental results and makes comparisons to other methods. Section 5 draws the conclusions.

2 Related Works

Section 2.1 briefly introduces the SMVQ technique.[13] Section 2.2 briefly reviews the hiding method of Chang et al.[9]

2.1 *SMVQ*

In SMVQ, an image is divided into some nonoverlapping blocks of 4×4 pixels each. All blocks in the top row and leftmost column of the image are called seed blocks and are encoded using VQ with the main codebook. The remaining blocks are called residual blocks, and each of them is encoded using SMVQ with a subcodebook that is created with reference to the adjacent coded-decoded blocks, to reduce the effects of the artificial block boundaries. According to Fig. 1, the neighboring decoded pixels of each residual block R are adopted to evaluate $r'_0=(u_{12}+l_3)/2$, $r'_1=u_{13}$, $r'_2=u_{14}$, and $r'_3=u_{15}$; $r'_4=l_7$, $r'_8=l_{11}$, and $r'_{12}=l_{15}$. The obtained values are then employed to compute the side match distortion (SMD) for each codeword $M=(m_0,m_1,\ldots,m_{15})$ in the main codebook as

$$\mathrm{SMD}(M)=\sqrt{\sum_{i=0,1,2,3,4,8,12}(r'_i-m_i)^2}. \tag{2}$$

Now, a subcodebook is generated by choosing from the main codebook $|C|$ codewords that have the lowest SMD values (where $|C|$ is the size of the subcodebook). The residual block R is then encoded using its nearest codeword in the generated subcodebook. Because the subcodebook is a proper subset of the main codebook, the index length in SMVQ encoding is reduced. Therefore, the compression rate of SMVQ surpasses that of VQ.

2.2 *Data Hiding Method of Chang et al.*[9]

First, the secret data are encrypted using available encryption techniques, such as DES[14] or Rivest–Shamir–Adleman.[16] Next, a cover image is divided into numerous nonoverlapping blocks of 4×4 pixels each. All seed blocks of the cover image are encoded using VQ, and the obtained VQ indices are immediately decoded without hiding any secret data. These decoded seed blocks then become the corresponding seed blocks in the stego-image. The residual blocks are processed in a raster scan. The upper and left decoded blocks of each residual block R are used to determine its subcodebook based on Eq. (2). Then the generated subcodebook is searched to find the codeword X that is nearest to R using Eq. (1). Finally, if the to-be-hidden encrypted bit equals zero, then the codeword X becomes the content of the corresponding residual block in the stego-image. Otherwise, another codeword Y that is nearest to X must be found in the subcodebook, and the weighted-mix $Z=\lfloor(2\times X+Y)/3\rfloor$ of the codewords X and Y becomes the content of the corresponding residual block in the stego-image. Notably, the hybrid Z is closer to X than to all other codewords in the subcodebook such that, after the secret data have been extracted, Z can be used to distinguish X from all other codewords in the subcodebook, enabling the original SMVQ code of the cover image to be reconstructed. This property is the so-called reversibility of the compression code.

3 Proposed Method

Section 3.1 introduces the principal idea. Section 3.2 elucidates the embedding process. Section 3.3 presents a detailed justification of Sec. 3.2. Section 3.4 describes the extraction of the secret and the reconstruction of the original SMVQ code.

3.1 *Use of Triangular Inequality*

In the method of Chang et al.,[9] the hybrid Z may be regarded as a perturbed version of X. In the proposed method, other possible perturbed versions of X are also considered to increase the hiding capacity. Hence, a sphere centered at X is defined such that all points on or in the sphere are regarded as candidates for Z. More specifically, let $k=\|X-Y\|$ be the Euclidean distance between X and Y and define a 16-dimensional sphere $S(X,t)$ with center X and radius t, where

$$t=\lceil k/2\rceil-1=\lceil\|X-Y\|/2\rceil-1. \tag{3}$$

Now, based on Lemma 1, any point Z on or in the sphere $S(X,t)$ is always closer to X than to all other codewords in the subcodebook. Because X is the unique codeword that is closest to Z, Z can be utilized to trace back X by inspecting all codewords in the subcodebook. Hence, each Z on or in the sphere $S(X,t)$ can always be used to find X. The reversibility of X is thus ensured.

Lemma 1. For each residual block R of a cover image, assume that the subcodebook of R is searched to find: (a) the codeword X that is closest to R and (b) another codeword Y that is closest to X. Let $k=\|X-Y\|$ be the Euclidean distance between X and Y, and let the sphere $S(X,t)$ be centered at X with the radius $t=\lceil k/2\rceil-1$. Then, any point Z on or in the sphere $S(X,t)$ will satisfy an inequality that identifies X: $\|Z-X\|<\|Z-Q\|$ holds for every other codeword Q in the subcodebook.

Proof. $\|X-Q\|\geq\|X-Y\|=k$ because Y is closest to X in the subcodebook. $\|Z-X\|\leq t<k/2$ because Z is on or in the

sphere $S(X,t)$. Applying the triangular inequality to ΔXQZ yields $\|Z-Q\| \geq (\|X-Q\|-\|Z-X\|) > k-k/2=k/2$. Therefore, $\|Z-X\| < k/2 < \|Z-Q\|$.

3.2 *Embedding Process*

Assume that secret data are encrypted using the DES approach and a cover image is divided into some nonoverlapping blocks of 4×4 pixels each. All blocks in the top row and leftmost column of the cover image are encoded by VQ, and the obtained VQ indices are then decoded without hiding the secret. The decoded blocks then become the corresponding block in the stego-image. The upper and left decoded blocks of each residual block R are used to compute the subcodebook of that residual block using Eq. (2). Then, the generated subcodebook is searched to find the codeword X that is closest to R. (Thus far, the process is similar to the encoding process of SMVQ.) The subcodebook is searched again to find the codeword Y that is closest to X. Now, based on the 16-dimensional codewords X and Y, and based on the discussion given between Eq. (3) and Lemma 1, the radius $t=\lceil\|X-Y\|/2\rceil-1$ of a sphere centered at X is determined. Then, according to an analysis to be explained below, Eqs. (4)–(6) are employed to determine the hiding capacity of each of the 16 components of X. Finally, the encrypted bits are sequentially grabbed and then embedded in each of the 16 components of X. The data-embedding process is as follows.

Input: (i) The encrypted data to be hidden and (ii) a cover image of several nonoverlapping blocks of 4×4 pixels each.

Output: The stego-image.

Step 1: Encode all seed blocks (blocks in the top row or leftmost column) of the cover image using VQ; the created VQ indices are then decoded without hiding any secret. Let these decoded seed blocks be the corresponding seed blocks in the stego-image.

Step 2: Process the residual blocks (blocks in neither the top row nor the leftmost column) of the cover image as follows.

Step 2a: Grab the next not-yet-processed residual block R of the cover image.

Step 2b: Generate the subcodebook that corresponds to the grabbed residual block R using Eq. (2). Find in the subcodebook the codeword X that is closest to R. Then, find in the subcodebook another codeword Y that is closest to X.

Step 2c: Let $k=\|X-Y\|$ be the Euclidean distance between the 16-dimensional codewords X and Y, and let the sphere $S(X,t)$ be centered at X with the radius $t=\lceil k/2\rceil-1$. Determine the hiding capacity $H(x_i)$ of the i'th component x_i of X using Eqs. (4)–(6) for each i in $0\leq i\leq15$.

Step 2d: For $0\leq i\leq15$, sequentially grab the next $H(x_i)$ not-yet-embedded bits from the encrypted data and let the single decimal value d_i be the decimal equivalent of the just-grabbed binary number with $H(x_i)$ bits. Then, use Eq. (8) to hide each d_i in x_i and determine the corresponding stego-value z_i. Let the generated stego-codeword $Z=(z_0,z_1,\ldots,z_{15})$ be the content of the corresponding residual block in the stego-image.

Step 2e: Repeat steps 2a–2d until all residual blocks have been examined. The whole stego-image is thus generated.

3.3 *Explanation of Proposed Design in Steps 2c and 2d in Section 3.2*

3.3.1 *Determine hiding capacity for each component of the codeword X*

The hiding capacity of each component of the codeword X is variable because it is determined by two codewords: $X=(x_0,x_1,\ldots,x_{15})$, which is closest to R, and $Y=(y_0,y_1,\ldots,y_{15})$, which is closest to X. The details are as follows. When secret data are embedded in X, the codeword X is modified to a stego-codeword $Z=(z_0,z_1,\ldots,z_{15})$. In the proposed design, Z is a perturbed version of X, and the chosen Z must stay on or in the sphere $S(X,t)$, according to Lemma 1. Therefore, the zx constraint $\Sigma_{i=0}^{15}(z_i-x_i)^2\leq t^2=(\lceil\|X-Y\|/2\rceil-1)^2$ must be satisfied when X is modified to Z to embed the secret.

Following this zx constraint of the stego-codeword Z for the codeword X, the hiding capacity of each component x_i of X can be derived. Assume that $v(1\leq v\leq4)$ bits of secret data can be embedded when a value x_i is modified to a stego-value z_i. If the v-bit LSB substitution method is used, then the v secret bits are copied directly to fill the v LSBs of the stego-value z_i, and the leading $(8-v)$ bits of z_i is set to be identical to the leading $(8-v)$ bits of x_i. Later, the decimal value of the hidden v bits can be extracted easily and rapidly by using $z_i \bmod 2^v$. Using the v-bit LSB substitution method causes the difference between z_i and x_i to be at most 2^v-1. Hence, for each i in $0\leq i\leq15$, $(z_i-x_i)^2\leq(2^v-1)^2$, implying $\Sigma_{i=0}^{15}(z_i-x_i)^2\leq16(2^v-1)^2$. To satisfy the zx constraint $\Sigma_{i=0}^{15}(z_i-x_i)^2\leq t^2$, a sufficient condition is to let v satisfy $16(2^v-1)^2\leq t^2$, or equivalently $2^v\leq\sqrt{t^2/16}+1$. Accordingly, v is set to

$$v=\lfloor\log_2(\sqrt{t^2/16}+1)\rfloor=\lfloor\log_2(t/4+1)\rfloor. \tag{4}$$

After v is determined using Eq. (4), $16(2^v-1)^2$ is very likely still to be smaller than t^2, rather than equal to t^2. To increase the hiding capacity, some of the 16 components of X can hide one more bit $(v+1$ bits rather than v bits). Let $w(<16)$ be the number of components that can hide one more bit. Then, because the difference between x_i and z_i is at most 2^v-1 (or $2^{v+1}-1$) when v bits (or $v+1$ bits) are hidden in x_i to yield z_i using the LSB substitution method, the zx constraint $\Sigma_{i=0}^{15}(z_i-x_i)^2\leq t^2$ can still be satisfied if $w(2^{v+1}-1)^2+(16-w)(2^v-1)^2\leq t^2$. Hence, $w[(2^{v+1}-1)^2-(2^v-1)^2]\leq t^2-16(2^v-1)^2$. Accordingly, the formula for w is

$$w=\left\lfloor\frac{t^2-16(2^v-1)^2}{(2^{v+1}-1)^2-(2^v-1)^2}\right\rfloor. \tag{5}$$

The w components of X can be arbitrarily assigned among the 16 positions {0, 1, …, 15}. The simplest assignment is to allow the first w of the 16 components of X to hide one more bit

$$H(x_0) = H(x_1) = \cdots = H(x_{w-1}) = v + 1;$$

$$H(x_w) = H(x_{w+1}) = \cdots = H(x_{15}) = v. \quad (6)$$

To increase security, users can assign the w locations among the 16 positions {0, 1, 2, …, 15} at will.

3.3.2 *Hide each secret value d_i in the i'th component x_i of the codeword X and generate the stego-value z_i*

Probably the simplest means of hiding in the i'th component x_i of the codeword X the secret value d_i, which is the decimal equivalent of the $H(x_i)$ grabbed bits, is to replace the final $H(x_i)$ bits of the value x_i by the $H(x_i)$ bits of the secret value d_i. This method is called the $H(x_i)$-bit LSB substitution method. In symbols, the stego-value is $z_i^{(\mathrm{LSB})} = x_i - [x_i \bmod 2^{H(x_i)}] + d_i = x_i + e_i$, where

$$e_i = -(x_i \bmod 2^{H(x_i)}) + d_i \quad (7)$$

is the shift from x_i to $z_i^{(\mathrm{LSB})} = x_i + e_i$.

To improve further the quality of the stego-image, the distortion between the stego-value z_i and the i'th pixel value r_i of the current residual block R should be reduced further, if possible. The method thus refined is discussed below. Notably, to reduce $(z_i - r_i)^2$, two basic requirements must still be satisfied: (R1) $(z_i - x_i)^2 \leqslant [2^{H(x_i)} - 1]^2$ and (R2) the secret value d_i must be given by $d_i = z_i \bmod 2^{H(x_i)}$, which is also the secret extraction formula that is used in the $H(x_i)$-bit LSB substitution method. Requirement (R1) ensures that the zx constraint $\Sigma_{i=0}^{15}(z_i - x_i)^2 \leqslant t^2$ remains satisfied. [Indeed, if $(z_i - x_i)^2 \leqslant [2^{H(x_i)} - 1]^2$ for each i in $0 \leqslant i \leqslant 15$, then $\Sigma_{i=0}^{15}(z_i - x_i)^2 \leqslant w(2^{v+1} - 1)^2 + (16 - w)(2^v - 1)^2 \leqslant t^2$ will be satisfied, as described by the explanation given between Eqs. (4) and (5).] Requirement (R2) ensures the fast extraction of the secret data.

Combining these two basic requirements to reduce $(z_i - r_i)^2$ changes the hiding of the secret value d_i in the i'th component x_i of the codeword X as

$$z_i = \begin{cases} x_i + e_i \text{ or } x_i + e_i - 2^{H(x_i)} \text{(choose the one closer to } r_i) & \text{if } e_i > 0, \\ x_i + e_i \text{ or } x_i + e_i + 2^{H(x_i)} \text{(choose the one closer to } r_i) & \text{if } e_i < 0, \\ x_i + e_i = x_i, & \text{if } e_i = 0, \end{cases} \quad (8)$$

where $e_i = -[x_i \bmod 2^{H(x_i)}] + d_i$, as in Eq. (7). Clearly, the stego-value z_i that is given by Eq. (8) has greater potential to reduce $(z_i - r_i)^2$ than does the $z_i^{(\mathrm{LSB})} = x_i + e_i$ of the $H(x_i)$-bit LSB substitution method. Additionally, the requirement $d_i = z_i \bmod 2^{H(x_i)}$ is satisfied in all three cases of Eq. (8) because

$$x_i + e_i = x_i - (x_i \bmod 2^{H(x_i)}) + d_i, \quad (9)$$

yielding $(x_i + e_i) \bmod 2^{H(x_i)} = d_i$. Another requirement $(z_i - x_i)^2 \leqslant [2^{H(x_i)} - 1]^2$ is met by Lemma 2.

Notably, in Eq. (8), neither underflow ($z_i < 0$) nor overflow ($z_i > 255$) occurs whenever the stego-value z_i equals $x_i + e_i$ because $z_i = x_i + e_i = x_i - [x_i \bmod 2^{H(x_i)}] + d_i$ causes an overwriting of the final $H(x_i)$ bits of the 8-bit value x_i by the $H(x_i)$ bits of the secret value d_i. The resulting z_i is still an 8-bit value, and thus, there is no underflow or overflow. From this observation, in Eq. (8), only the candidate $x_i + e_i - 2^{H(x_i)}$ [or $x_i + e_i + 2^{H(x_i)}$] may suffer from underflow (or overflow). Therefore, when $2^{H(x_i)}$ is subtracted from (or added to) $x_i + e_i$ in Eq. (8) to yield one more candidate $x_i + e_i - 2^{H(x_i)}$ [or $x_i + e_i - 2^{H(x_i)}$] for the stego-value z_i, the candidate $x_i + e_i - 2^{H(x_i)}$ [or $x_i + e_i + 2^{H(x_i)}$] should be checked immediately and dropped if it is negative (or >255).

Lemma 2. The inequality $(z_i - x_i)^2 \leqslant [2^{H(x_i)} - 1]^2$ is satisfied for each z_i that is given by Eq. (8).

Proof. Equation (9) indicates that $x_i + e_i$ is the overwriting of the final $H(x_i)$ bits of x_i by the $H(x_i)$ bits of the secret value d_i. Because $x_i + e_i$ and x_i differ in their final $H(x_i)$ bits, their difference is at most $2^{H(x_i)} - 1$. Hence, $1 - 2^{H(x_i)} \leqslant e_i \leqslant 2^{H(x_i)} - 1$. Now, for the case of $e_i > 0$ (line 1) in Eq. (8), two assignments $[z_i = x_i + e_i$ and $z_i = x_i + e_i - 2^{H(x_i)}]$ for z_i are possible. If the assignment is $z_i = x_i + e_i$, then $z_i - x_i = e_i$, which is shown in the range $1 - 2^{H(x_i)} \leqslant e_i \leqslant 2^{H(x_i)} - 1$. Accordingly, $(z_i - x_i)^2 \leqslant [2^{H(x_i)} - 1]^2$. With respect to the alternative assignment $z_i = x_i + e_i - 2^{H(x_i)}$, because $e_i > 0$ and $e_i \leqslant 2^{H(x_i)} - 1$, $z_i - x_i = [x_i + e_i - 2^{H(x_i)}] - x_i = e_i - 2^{H(x_i)} \geqslant 1 - 2^{H(x_i)}$, and $z_i - x_i = e_i - 2^{H(x_i)} \leqslant 2^{H(x_i)} - e_i \leqslant 2^{H(x_i)} - 1$. Hence, $1 - 2^{H(x_i)} \leqslant z_i - x_i \leqslant 2^{H(x_i)} - 1$ still applies, indicating $(z_i - x_i)^2 \leqslant (2^{H(x_i)} - 1)^2$. This completes the proof in the case of $e_i > 0$ in Eq. (8). The proof for the case of $e_i < 0$ in Eq. (8) is similar and therefore omitted.

3.4 *Extraction of Hidden Data and Reconstruction of Original SMVQ Code*

When the stego-image is received, the secret data can be extracted, and the original SMVQ code can also be reconstructed. The details follow.

Input: The stego-image.

Output: The secret data and the original SMVQ code.

Step 1: Divide the stego-image into non-overlapping blocks of 4×4 pixels each. Encode each seed block of the stego-image using VQ. The obtained VQ indices are the SMVQ indices required for these seed blocks.

Step 2: Process all residual blocks of the stego-image as follows.

Step 2a: Grab the next not-yet-processed residual block Z of the stego-image.

Step 2b: Generate the subcodebook for the grabbed block Z using Eq. (2). Search in the generated subcodebook to find the codeword X that is closest to Z. Then, search in the subcodebook to find another codeword Y that is closest to X. Output the SMVQ index of the codeword X. The obtained SMVQ index is the desired SMVQ index for the current residual block.

Step 2c: On the basis of the 16-dimensional codewords X and Y, calculate the radius t using Eq. (3). For $0 \leqslant i \leqslant 15$, determine the hiding capacity $H(x_i)$ using Eqs. (4)–(6).

Step 2d: For $0 \leqslant i \leqslant 15$, extract the hidden data value from the i'th component z_i of Z using

$$d_i = z_i \bmod 2^{H(x_i)}. \quad (10)$$

Then, convert d_i into its binary equivalent.

Table 1 PSNRs of SMVQ-decompressed images when test image is 512×512 gray-scale image Lena.

	Size of main codebook		
	128	256	512
Size of subcodebook	Quality of SMVQ-decompressed image (dB)		
16	28.80	28.32	26.78
32	30.20	30.11	29.11
64	31.15	31.60	31.33
128		32.39	32.72
256			33.44

Step 2e: Go to step 2a, unless all residual blocks have been examined. In that case, all of the encrypted secret bits have already been extracted, and all of the SMVQ indices have already been outputted.

After all the above steps have been performed, decrypt the extracted encryption bits to obtain the secret data. The indices obtained in steps 1 and 2b are always the same as in the original SMVQ code.

4 Experimental Results and Comparisons

Five 512×512 gray-scale images (Lena, Jet, Boat, Peppers, and Baboon) are used in the experiments. The secret data are a randomly generated bit stream. The peak-signal-to-noise ratio (PSNR), defined by

$$\mathrm{PSNR} = 10 \times \log_{10} \frac{255^2}{\mathrm{MSE}}, \tag{11}$$

is employed to measure the similarity between the original image and its modified image (such as a stego-image or an SMVQ-decompressed image) where MSE represents the mean square error between the pixel values of the original image and those of its modified image.

For the 512×512 gray-scale image Lena, Table 1 shows the quality of the SMVQ-decompressed images in which there is no hidden secret. Table 2 presents the quality of our stego-images and the number of hidden bits (hiding capacity). The main codebooks of sizes 128, 256, and 512 with the subcodebooks of sizes 16, 32, 64, 128, and 256 are adopted in SMVQ to construct Tables 1 and 2. The proposed method can hide a large amount of secret data, while maintaining acceptable quality of the stego-images. Additionally, the quality of our stego-images (with hidden secret) is even better in terms of PSNR than that of the SMVQ-decompressed images (without any hidden secret). For example, when a main codebook of size 256 is equipped with the subcodebooks of sizes 16, 32, 64, and 128, the PSNRs of our stego-images are 29.20, 31.18, 32.91, and 33.86 dB, respectively, which values are a little better than the 28.32, 30.11, 31.60, and 32.39 dB values of the plain SMVQ-decompressed images. Similar observations were made for other codebook sizes.

Figure 2(a) displays the SMVQ-decompressed image Lena without any hidden secret. Figure 2(b) depicts our stego-image Lena after a secret of size 413,667 bits was hidden. A main codebook of size 256 and a subcodebook of size 128 are adopted in SMVQ for Fig. 2. As shown in Figs. 2(c) and 2(d), the quality of our stego-image Lena slightly exceeds that of the plain SMVQ-decompressed image. Table 3 compares the method of Chang et al.[9] with the proposed method in terms of hiding capacity and stego-

Table 2 Our hiding capacities and PSNRs of stego-images when cover image is 512×512 gray-scale image Lena.

	Size of main codebook					
	128		256		512	
Size of subcodebook	Hiding capacity (bits)	Stego-image quality (dB)	Hiding capacity (bits)	Stego-image quality (dB)	Hiding capacity (bits)	Stego-image quality (dB)
16	504,608	29.88	416,942	29.20	360,683	27.83
32	504,993	31.52	414,019	31.18	358,901	29.90
64	505,624	32.70	413,786	32.91	360,100	32.40
128			413,667	33.86	360,609	34.04
256					360,360	34.90

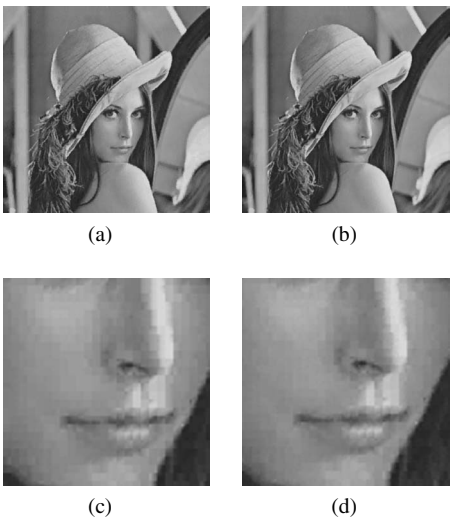

Fig. 2 SMVQ-decompressed image versus proposed stego-image: (a) 32.39 dB SMVQ-decompressed image Lena (no hiding), (b) proposed 33.86 dB stego-image Lena, (c) magnified nose and mouth of (a), and (d) magnified nose and mouth of (b).

image quality. It is observed that the proposed method has a higher hiding capacity and better stego-image quality than those in Chang et al.[9]

Chi-square analysis is used by attackers to determine whether a stego-image contains secret data. As noted by Chang et al.,[9] their method more effectively resists Chi-square attack than does the LSB substitution scheme. To demonstrate that the proposed method can also resist Chi-square attack, Guillermito's[17] Chi-square steganography test program is run to perform the statistical analysis. Figure 3(a) presents a 512×512 gray-scale image, Baboon, without any hidden secret. Figure 3(b) displays the stego-image Baboon that is generated by embedding the secret in Fig. 3(a) using the LSB substitution scheme. Figure 3(c) depicts our stego-image Baboon after a secret of size 675,226 bits was hidden. Figures 3(d)–3(f) plot the results obtained when the Chi-square analysis tool was used to examine the pixels in Figs. 3(a)–3(c), respectively. The cross curve indicates the probability that the pairs of values

Table 3 Comparisons of hiding capacity and stego-image quality between method of Chang et al.[9] and proposed method.

	Chang et al.'s method		Proposed method	
Cover image	Hiding capacity (bits)	Stego-image quality (dB)	Hiding capacity (bits)	Stego-image quality (dB)
Lena	16,129	32.45	413,667	33.86
Jet	16,129	31.09	433,881	32.53
Boat	16,129	29.93	467,367	31.39
Peppers	16,129	29.19	413,322	33.94
Baboon	16,129	23.66	675,226	26.52

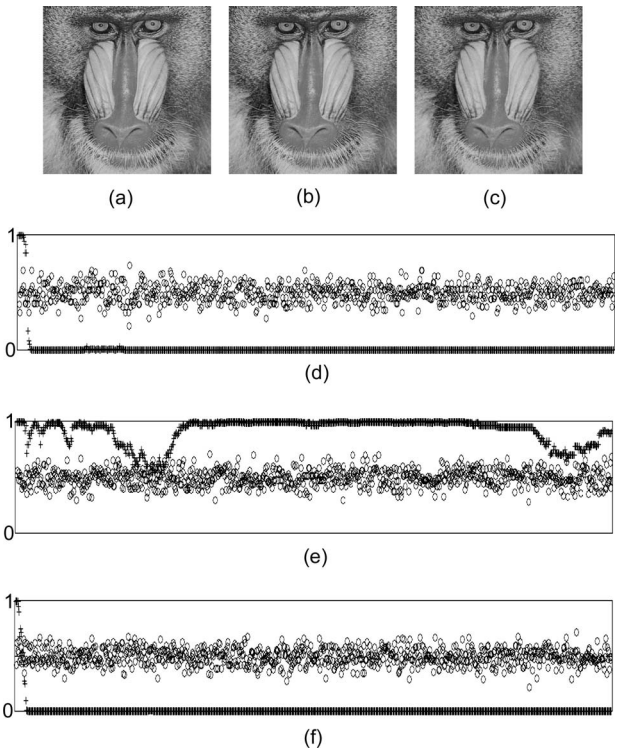

Fig. 3 Results of Chi-square analysis: (a) gray-scale image Baboon without hidden secret, (b) stego-image Baboon that is generated using LSB substitution scheme, (c) proposed stego-image Baboon; (d–f) results of Chi-square analysis for the pixels in (a), (b), and (c), respectively.

are randomly distributed, and the circular curve represents the average value of all LSBs in one block of pixels. On the basis of the example in Section 3.5 of Guillermito's work,[17] the circular curves in Figs. 3(d)–3(f) suggest to the attackers nothing because these circular curves are all around $(0+1)/2=0.5$. However, the cross curve in Fig. 3(e) is close to 1, indicating that some secret data are very probably embedded in Fig. 3(b). In contrast, the cross curves in Figs. 3(d) and 3(f) are both flat along the zero axis. Therefore, the attackers may ignore the hidden secret in Fig. 3(c).

Figure 4 plots the Chi-square analysis results for the gray-scale image Lena. All the circular curves in Figs. 4(d)–4(f) are still around $(0+1)/2=0.5$. However, the cross curve in Fig. 4(e) differs from that in Fig. 4(d), suggesting to the attackers that some secret data are embedded in Fig. 4(b). In contrast, the cross curves in Figs. 4(d) and 4(f) are still flat along the zero axis, and thus, the secret in Fig. 4(c) is hidden more safely than that in Fig. 4(b).

5 Summary

This work proposes a steganographic method that is based on SMVQ. In decoding, after lossless extraction of the secret data, the original SMVQ code can be reconstructed. The reconstructed SMVQ code can then be transmitted economically and reused many times to hide a new secret. Experimental results show that the proposed method provides a high hiding capacity and acceptable stego-image quality. From Fig. 2, the quality of our stego-image is even better than that of the SMVQ-decompressed image without

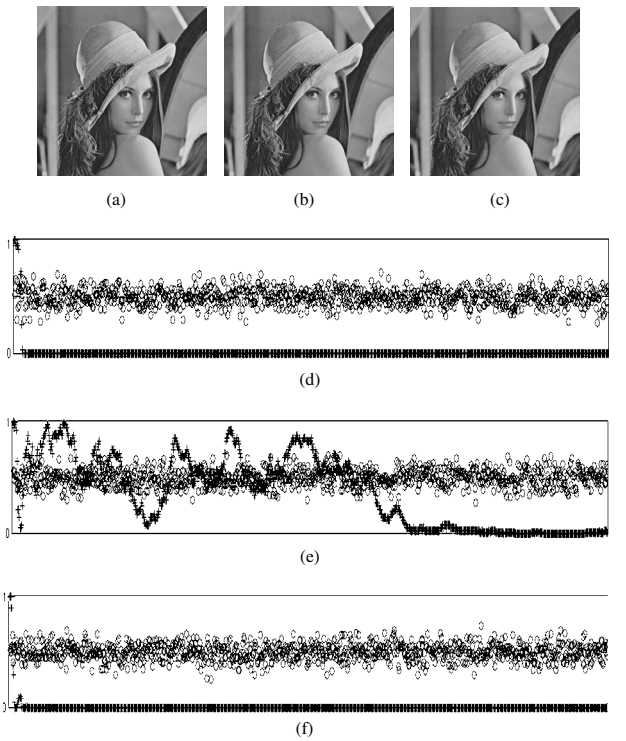

Fig. 4 Another result of Chi-square analysis. Everything is as in Fig. 3, except image Baboon is replaced by image Lena.

any hidden secret. As presented in Table 3, the proposed method outperforms that of Chang et al.[9] in terms of hiding capacity and stego-image quality. Therefore, as well as reconstructing the original SMVQ code, the proposed method also offers the advantages of high hiding capacity and low image distortion.

Acknowledgments

The work is supported by National Science Council of Republic of China under Grant No. NSC97-2221-E-009-120-MY3. The authors thank the reviewers for valuable suggestions to improve the paper.

References

1. R. Z. Wang, C. F. Lin, and J. C. Lin, "Image hiding by optimum LSB substitution and generic algorithm," *Pattern Recogn.* **3**(3), 671–683 (2001).
2. D. C. Wu and W. H. Tsai, "A steganographic scheme for images by pixel-value differencing," *Pattern Recogn. Lett.* **24**(9–10), 1613–1626 (2003).
3. C. C. Thien and J. C. Lin, "A simple and high-hiding capacity scheme for hiding digit-by-digit data in images based on modulus function," *Pattern Recogn.* **36**(12), 2875–2881 (2003).
4. K. L. Chung, C. H. Shen, and L. C. Chang, "A novel SVD-and VQ-based image hiding scheme," *Pattern Recogn. Lett.* **22**(9), 1051–1058 (2001).
5. Y. C. Hu, "Grayscale image hiding scheme based on vector quantization," *IEE Electron. Lett.* **39**(2), 202–203 (2003).
6. S. C. Shie and S. D. Lin, "Secret image transmission based on VQ and data embedding," *Int. J. Imaging Syst. Technol.* **17**(1), 1–9 (2007).
7. R. Z. Wang and Y. D. Tsai, "An image-hiding scheme with high hiding capacity based on best-block matching and k-means clustering," *Pattern Recogn.* **40**(2), 398–409 (2007).
8. Y. J. Chang, R. Z. Wang, and J. C. Lin, "Hiding images using modified search order coding and modulus function," *Int. J. Pattern Recognit. Artif. Intell.* **22**(6), 1215–1240 (2009).
9. C. C. Chang, W. L. Tai, and C. C. Lin, "A reversible data hiding scheme based on side match vector quantization," *IEEE Trans. Circuits Syst. Video Technol.* **16**(10), 1301–1308 (2006).
10. R. M. Gray, "Vector quantization," *IEEE ASSP Mag.* **1**(2), 4–29 (1984).
11. W. B. Pennebaker and J. L. Mitchell, *JPEG Still Image Data Compression Standard*, Van Nostrand Reinhold, New York (1993).
12. Y. Linde, A. Buzo, and R. M. Gray, "An algorithm for vector quantizer design," *IEEE Trans. Commun.* **28**(1), 84–95 (1980).
13. T. Kim, "Side match and overlap match vector quantizers for images," *IEEE Trans. Image Process.* **1**(4), 170–185 (1992).
14. National Bureau of Standards, "Data Encryption Standard," Federal Information Processing Standard Publ. No. 46 (1977).
15. National Bureau of Standards, "Advanced Encryption Standard," Federal Information Processing Standards Publ. No. 197 (2001).
16. R. L. Rivest, A. Shamir, and L. Adleman, "A scheme for obtaining digital signatures and public-key cryptosystems," *Commun. ACM* **21**(2), 120–126 (1978).
17. S.Guillermito, "Chi-square steganography test program," ⟨http://www.guillermito2.net/stegano/tools/index.html (2004)⟩.

Lee Shu-Teng Chen received his BS in computer and information science from National Chiao Tung University (NCTU), Taiwan, in 1999, and MS in computer science and information engineering from National Taiwan University, Taiwan, in 2001. He is a PhD candidate in the Department of Computer Science and Information Engineering at NCTU. His current research interests include data hiding and image sharing.

Ja-Chen Lin received his BS and MS from NCTU, Taiwan. In 1988, he received his PhD in mathematics from Purdue University, West Lafayette, Indiana. He joined the Department of Computer and Information Science at NCTU, where he became a professor. His research interests include pattern recognition and image processing. Dr. Lin is a member of the Phi-Tau-Phi Scholastic Honor Society.