Chapter 4

A Hybrid Steganography Technique Based on RR, AQVD and QVC

4.1 Introduction

PVD steganography techniques face FOBP, and APVD techniques face UBP. To eradicate these problems, in this chapter we describe a new technique. This technique is designed using the concepts RR, AQVD, and QVC. It performs embedding and extraction operation on 3×3 non-overlapping blocks. From a block, 2 new blocks are generated, remainder block (RB), and quotient block (QB). RB comprises remainders. A remainder is a decimal integer for 2 bits. That's why it shall be replaced by a pair of secret bits. QB comprises quotients. A quotient is a decimal integer for 6 bits. In 4 corner quotients of a QB, AQVD procedure is plied to camouflage the secret bits. In quotients of middle row of QB, QVC approach is plied to conceal the secret bits. Estimated PSNR is 39.74 decibels (dB) with capacity 3.21 BPB. Furthermore, RS and PDH attacks could not catch this technique.

4.2 The Proposed RR, AQVD, and QVC Technique [117]

The OI is scanned in raster order and 3×3 blocks are formed in dis-joint manner. From a 3×3 block, we develop again 2 blocks, namely RB, and QB. RB comprises of remainders and QB comprises of quotients. Each remainder is written in 2 bits. There are 9 quotients in QB. In 4 corner quotients AQVD is plied and in rest of the quotients QVC is plied. In the RB substitution is applied to hide data bits. The Subsections 4.2.1 and 4.2.2 describe the data concealing and retrieval procedures in detail.

4.2.1 The Data Embedding Procedure

Fig.4.1(a) is one 3×3 block. Here, the 9 pixels are: P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8 and P_9. The message is converted to an array of bits and camouflaged in these pixels using the steps mentioned below. Each channel R, G, and B of a color image shall be treated as a pixel. Thus, there are 8 bits in a pixel.

Step 1: From a block projected in Fig.4.1(a), 2 new blocks, (i) Fig.4.1(b), and (ii) Fig.4.1(c) are generated using Eq.4.1, and Eq.4.2 respectively. Fig.4.1(b) is the RB and Fig.4.1(c) is the QB.

$$R_i = P_i \bmod 4, \text{ for, } i = 1 \text{ through } 9 \qquad (4.1)$$

$$Q_i = P_i \operatorname{div} 4, \text{ for, } i = 1 \text{ through } 9 \qquad (4.2)$$

Step 2: The SV of R_1 is R_1'. It is obtained by taking 2 secret bits and writing it in decimal integer. In same way for i= 2 through 9, we can take 2 secret bits and write in a decimal integer to obtain the SV of R_i and denote it as R_i'. These newly obtained remainders make the stego-RB as projected in Fig.4.1(d).

Step 3: See Fig.4.1(c). AQVD procedure shall be plied at 4 corner quotients Q_1, Q_3, Q_7, and Q_9 by using Q_2, and Q_8 as references. The values of Q_2 and Q_8 are unaltered when data is camouflaged in Q_1, Q_3, Q_7, and Q_9.

For i =1, 3, 7, 9, we compute, $d_{ia} = (Q_i - Q_2)$ and $d_{ib} = (Q_i - Q_8)$.

The LB and UB of target quotient Q_i are l_i and u_i accordingly. These are enumerated in Eq.4.3. In this equation MAX is a function to select the larger one out of its 2 arguments and MIN is a function to select the smaller one out of its 2 arguments.

$$\{l_i, u_i\} = \begin{cases} \{\text{MAX}(Q_2 + 1, Q_8 + 1),\ 63\}, \text{if } (d_{ia} > 0) \text{ and } (d_{ib} > 0) \\ \{0, \text{MIN}(Q_2, Q_8)\}, \text{if } (d_{ia} \le 0) \text{ and } (d_{ib} \le 0) \\ \{(Q_2 + 1), Q_8\}, \text{if } (d_{ia} > 0) \text{ and } (d_{ib} \le 0) \\ \{(Q_8 + 1),\ Q_2\}, \text{if } (d_{ia} \le 0) \text{ and } (d_{ib} > 0) \end{cases} \qquad (4.3)$$

Hence 4 pairs of LBs and UBs are found. They are, l_1 and u_1 for Q_1, l_3 and u_3 for Q_3, l_7 and u_7 for Q_7, l_9 and u_9 for Q_9.

For i =1, 3, 7, 9, enumerate n_i plying Eq.4.4.

$$n_i = \text{MIN}\ (\lfloor \log_2 |u_i - l_i + 1| \rfloor, 2) \qquad (4.4)$$

Pick up next n_i secret bits and represent it as b_i, a decimal value. After embedding b_i in Q_i, its SV, Q_i' is found by operating Eq.4.5.

$$Q_i' = \underset{e}{\operatorname{argmin}} \{|e - Q_i| \mid |e - Q_2| \equiv b_i \pmod{2^{n_i}},\ e \in [l_i, u_i]\} \qquad (4.5)$$

Eq.4.5 is explained here. The SV, Q_i' for Q_i is chosen from range $[l_i, u_i]$ by satisfying 2 constraints, (i) $|e - Q_2| \bmod 2^{n_i} = b_i$, and (ii) $|e - Q_i|$ is lowest.

Now the QB is projected in Fig.4.1(e). Here, the 4 corner quotients are altered after hiding the bits.

Step 4: Refer Fig.4.1(e). On Q_4, Q_5 and Q_6, QVC is to be plied. For Q_4 the 2 reference pixels are Q_1' and Q_7'. In case of Q_5 the pixels Q_2 and Q_8 are the references. Similarly, for Q_6 the 2 reference pixels are Q_3' and Q_9'.

Enumerate 3 difference values, $d_4 = |Q_1' - Q_7'|$, $d_5 = |Q_2 - Q_8|$, and $d_6 = |Q_3' - Q_9'|$. For i = 4, 5, 6, enumerate n_i (the concealing capacity of Q_i) by operating Eq.4.6.

$$n_i = \begin{cases} \lfloor \log_2 d_i \rfloor, \text{if } d_i > 1 \\ 1, \text{otherwise} \end{cases} \qquad (4.6)$$

For i = 4 to 6, pick up n_i bits and write in decimal integer b_i. After concealing b_i in Q_i, the stego value Q_i' is found plying Eq.4.7.

$$Q_i' = Q_i - Q_i \bmod 2^{n_i} + b_i \qquad (4.7)$$

Enumerate the difference value, $df_i = Q_i' - Q_i$ and to minimize this difference, apply Eq.4.8. Now Fig.4.1(f), depicts the stego-QB.

$$Q_i' = \begin{cases} Q_i' + 2^{n_i}, \text{ if } (-2^{n_i} < df_i < -2^{n_i - 1}) \text{ and } ((Q_i' + 2^{n_i}) \le 63) \\ Q_i' - 2^{n_i}, \text{ if } (2^{n_i - 1} < df_i < 2^{n_i}) \text{ and } ((Q_i' - 2^{n_i}) \ge 0) \\ Q_i', \text{ otherwise} \end{cases} \qquad (4.8)$$

Step 5: Ply Eq.4.9, for i =1 through 9, to generate SVs for pixels.

$$P_i' = Q_i' \times 4 + R_i' \qquad (4.9)$$

Fig.4.1(g) is our stego-block after data hiding.

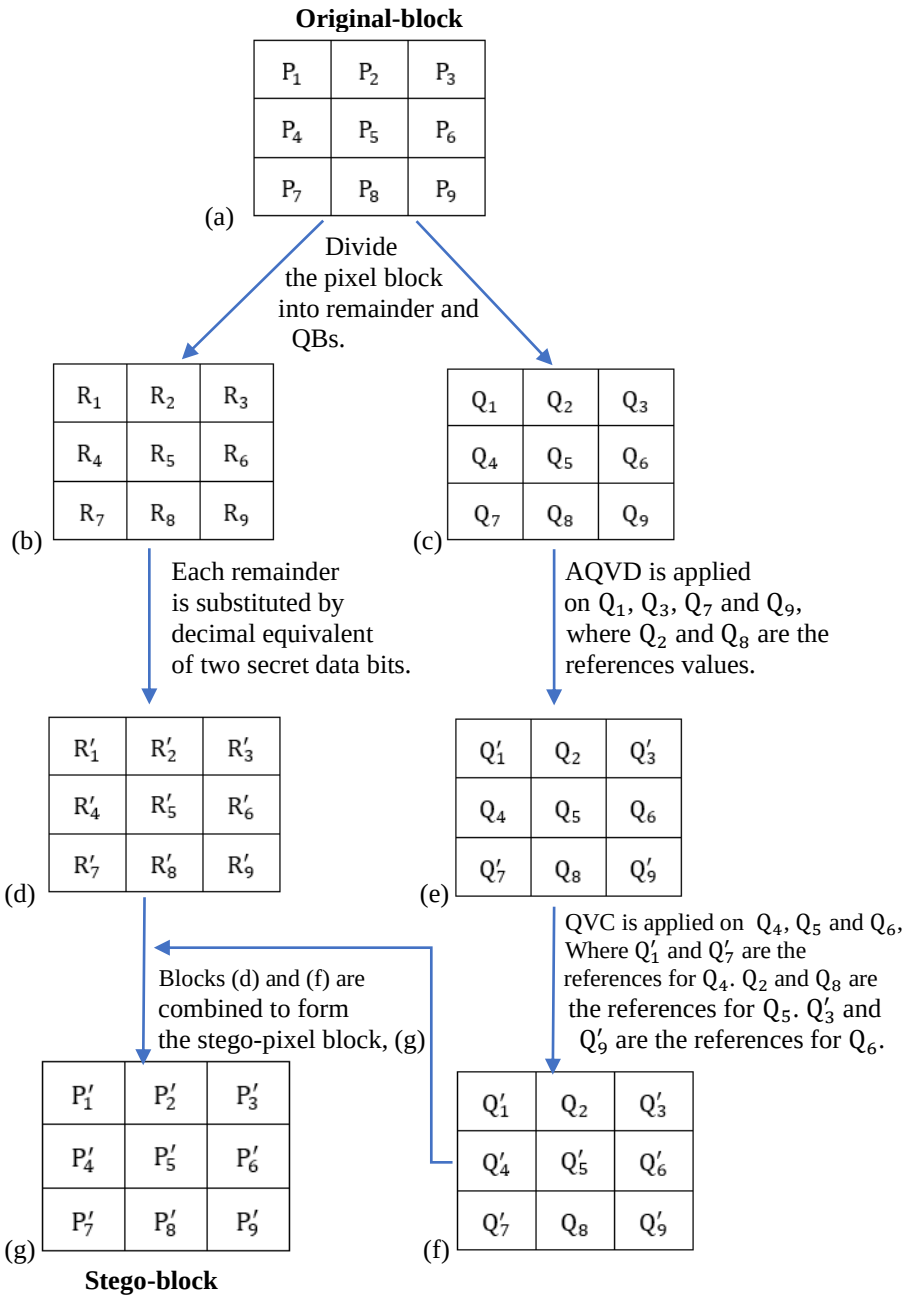


Fig.4.1 The data embedding procedure

4.2.2 The Data Extraction Procedure

The SI is cut into 3×3 blocks. Here, Fig.4.2(a) is a block. The 9 pixels in it are P'_i for i= 1 through 9. The retrieval of data is done by the below described procedure.

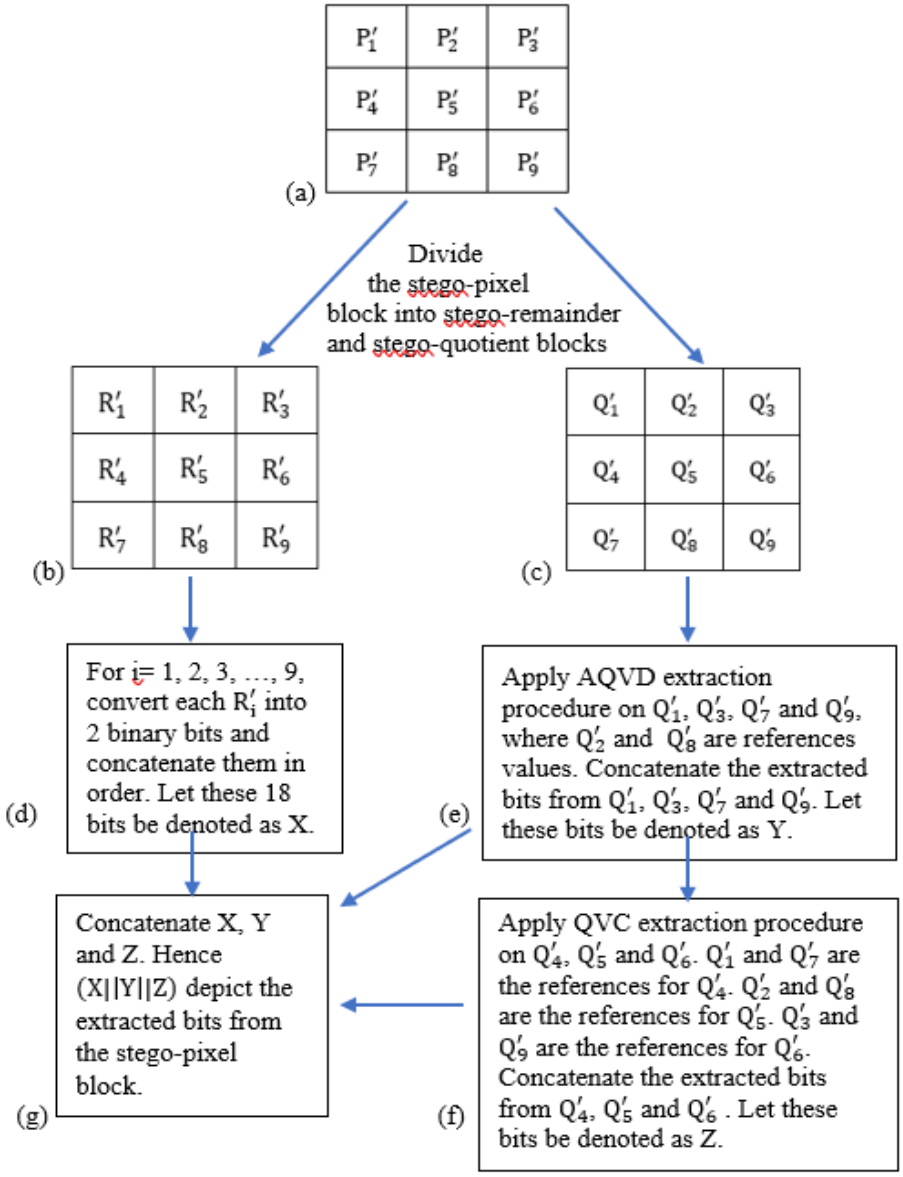


Fig.4.2 The data extraction procedure

Step 1: Ply Eq.4.10, and Eq.4.11 on stego-block (Fig.4.2(a)), to generate 2 new blocks, (i) Fig.4.2(b), and (ii) Fig.4.2(c). Fig.4.2(b) is stego-RB and Fig.4.2(c) is projected is stego-QB.

$$R'_i = P'_i \bmod 4, \text{ for, } i = 1 \text{ through } 9 \qquad (4.10)$$

$$Q'_i = P'_i \operatorname{div} 4, \text{ for, } i = 1 \text{ through } 9 \qquad (4.11)$$

Step 2: For, i = 1 through 9, write each R'_i in 2 binary bits and keep these 18 bits in a sequence and let it be represented by X.

Step 3: Next, we shall use retrieval part of AQVD procedure on Q'_1, Q'_3, Q'_7, and Q'_9, with Q'_2 and Q'_8 as guiding values.

For i value in {1, 3, 7, 9}, we generate $d_{ib} = (Q'_i - Q'_8)$ and $d_{ia} = (Q'_i - Q'_2)$. The LB and UB of Q'_i is l_i and u_i accordingly. It is enumerated in Eq.4.12.

$$\{l_i, u_i\} = \begin{cases} \{MAX(Q'_2 + 1, Q'_8 + 1),\ 63\}, \text{if } (d_{ia} > 0) \text{ and } (d_{ib} > 0) \\ \{0, MIN(Q'_2, Q'_8)\}, \text{if } (d_{ia} \leq 0) \text{ and } (d_{ib} \leq 0) \\ \{(Q'_2 + 1), Q'_8\}, \text{if } (d_{ia} > 0) \text{ and } (d_{ib} \leq 0) \\ \{(Q'_8 + 1), Q'_2\}, \text{if } (d_{ia} \leq 0) \text{ and } (d_{ib} > 0) \end{cases} \quad (4.12)$$

Use Eq.4.6 to compute n_i.

For i value in {1, 3, 7, 9}, we ply Eq.4.13 to generate b_i. Writet b_i in n_i bits. These bits are pulled out from Q'_i. Arrange all the obtained bits in a sequence and say it is Y.

$$b_i \equiv |Q'_i - Q'_2| \pmod{2^{n_i}}. \quad (4.13)$$

Eq.4.13 says that, there exists a set of candidate values of b_i, constrained by the constraint $(b_i \bmod 2^{n_i}) = |Q'_i - Q'_2|$. The smallest value is selected as the accepted value of b_i.

Step 4: Now ply retrieval part QVC procedure on Q'_4, Q'_5 and Q'_6. Q'_1 and Q'_7 are the guiding values for Q'_4. Q'_2 and Q'_8 are the guiding values for Q'_5. Q'_3 and Q'_9 are the guiding values for Q'_6.

Calculate three difference values, $d_4 = |Q'_1 - Q'_7|$, $d_5 = |Q'_2 - Q'_8|$, and $d_6 = |Q'_3 - Q'_9|$.
For i = 4, 5, 6, enumerate $n_i = \lfloor \log_2 d_i \rfloor$ and enumerate b_i in Eq.4.14. The b_i is the decimal equivalent of the n_i bits pulled from Q'_i.

$$b_i = Q'_i \bmod 2^{n_i} \quad (4.14)$$

Now, change each b_i to n_i bits. Arrange them as a bit sequence and call it as Z.

Step 5: Adjoin X, Y, and Z. Hence the retrieved data is $(X \,||\, Y \,||\, Z)$. The notation || is for adjoining.

4.3 Example of Embedding and Extraction

4.3.1 Example of Embedding Procedure

We have to hide the bit sequence "$(10\ 01\ 10\ 00\ 10\ 10\ 01\ 11\ 11\quad 01\ 1\ 11\ 10\ \ 1\ 0\ 11)_2$" in a block projected in Fig.4.3(a).

Step 1: plying Eq.4.1 and Eq.4.2, we generate remainder (Fig.4.3(b)) and QBs (Fig.4.3(c)) respectively.

Step 2: For each i =1 through 9, take 2 secret bits and write in decimal integer. We got $R'_1 = 2$, $R'_2 = 1$, $R'_3 = 2$, $R'_4 = 0$, $R'_5 = 2$, $R'_6 = 2$, $R'_7 = 1$, $R'_8 = 3$, and $R'_9 = 3$. These are SVs for the remainders. Fig.4.3(d) shows these as a block (stego-RB).

Step 3: From the QB enumerate 8 difference values, $d_{1a}= (Q_1 - Q_2) = -1$, $d_{1b}= (Q_1 - Q_8) = -3$, $d_{3a}= (Q_3 - Q_2) = 1$, $d_{3b}= (Q_3 - Q_8) = -1$, $d_{7a}= (Q_7 - Q_2) = 0$, $d_{7b}= (Q_7 - Q_8) = -2$, $d_{9a}= (Q_9 - Q_2) = 3$, and $d_{9b}= (Q_9 - Q_8) = 1$.

Using Eq.4.3, calculate $\{l_1, u_1\}$, $\{l_2, u_2\}$, $\{l_3, u_3\}$, and $\{l_4, u_4\}$.

$d_{1a} \le 0$ and $d_{1b} \le 0$, so $\{l_1, u_1\}$={0, MIN (38, 40)} = {0, 38}

$d_{3a} > 0$ and $d_{3b} \le 0$, so $\{l_3, u_3\}$={38+1, 40} = {39, 40}

$d_{7a} \le 0$ and $d_{7b} \le 0$, so $\{l_7, u_7\}$={0, MIN (38, 40)} = {0, 38}

$d_{9a} > 0$ and $d_{9b} > 0$, so $\{l_9, u_9\}$={MAX (38+1, 40+1), 63} = {41, 63}

Plying Eq.4.4, generate n_1, n_3, n_7, and n_9.

n_1= MIN ($\lfloor \log_2 |38 - 0 + 1| \rfloor$, 2) = 2, n_3= MIN ($\lfloor \log_2 |40 - 39 + 1| \rfloor$, 2) = 1, n_7= MIN ($\lfloor \log_2 |38 - 0 + 1| \rfloor$, 2) = 2, and n_9= MIN ($\lfloor \log_2 |63 - 41 + 1| \rfloor$, 2) = 2.

Pick up next n_1, n_3, n_7, and n_9 secret bits distinctly and represent them to decimal notation, we find b_1=1, b_3=1, b_7=3, and $b_9 = 2$.

Plying Eq.4.5, generate Q'_9, Q'_7, Q'_3, and Q'_1.

$$Q'_1 = \underset{e}{\operatorname{argmin}} \{|e - 37| \mid |e - 38| \equiv 1 \pmod{2^2},\ e \in [0, 38]\} = 37$$

$$Q'_3 = \underset{e}{\operatorname{argmin}} \{|e - 39| \mid |e - 38| \equiv 1 \pmod{2^1},\ e \in [39, 40]\} = 39$$

$$Q'_7 = \underset{e}{\operatorname{argmin}} \{|e - 38| \mid |e - 38| \equiv 3 \pmod{2^2},\ e \in [0, 38]\} = 35$$

$$Q'_9 = \underset{e}{\operatorname{argmin}} \{|e - 41| \mid |e - 38| \equiv 2 \pmod{2^2},\ e \in [41, 63]\} = 44$$

Fig.4.3(e) records 4 altered values of corner quotients after AQVD.

150	155	156
151	157	158
152	160	165

(a)

2	3	0
3	1	2
0	0	1

(b)

37	38	39
37	39	39
38	40	41

(c)

2	1	2
0	2	2
1	3	3

(d)

37	38	**39**
37	39	39
35	40	**44**

(e)

37	38	**39**
37	**38**	**39**
35	40	**44**

(f)

150	**153**	**158**
148	**154**	**158**
141	**163**	**179**

(g)

Fig.4.3 (a) original pixel block, (b) RB, (c) QB, (d) stego-RB, (e) QB after AQVD, (f) QB after QVC, and (g) stego-pixel block

Step 4: Generate 3 difference values, $d_4 = |Q'_1 - Q'_7| = |37 - 35| = 2$, $d_5 = |Q_2 - Q_8| = |38 - 40| = 2$, and $d_6 = |Q'_3 - Q'_9| = |39 - 44| = 5$.

Using Eq.4.6, calculate n_4, n_5, and n_6. $n_4 = \lfloor \log_2 2 \rfloor = 1$, $n_5 = \lfloor \log_2 2 \rfloor = 1$, and $n_6 = \lfloor \log_2 5 \rfloor = 2$.

Pick up next n_4, n_5, and n_6 secret bits distinctly and write in decimal integer. We got $b_4 = 1$, $b_5 = 0$, and $b_6 = 3$.

Using Eq.4.7, we obtain Q'_6, Q'_5, and Q'_4.

$Q'_4 = 37 - 37 \bmod 2^1 + 1 = 37$, $Q'_5 = 39 - 39 \bmod 2^1 + 0 = 38$, and $Q'_6 = 39 - 39 \bmod 2^2 + 3 = 39$.

Now enumerate 3 difference values, $df_4 = Q'_4 - Q_4 = 0$, $df_5 = Q'_5 - Q_5 = -1$, and $df_6 = Q'_6 - Q_6 = 0$. After plying Eq.4.8, we found $Q'_4 = 37$, $Q'_5 = 38$, and $Q'_6 = 39$. This is due to the satisfaction of case 3 in Eq.4.8. Fig.4.3(f) projects these altered values after QVC.

Step 5: Plying Eq.4.9, we get $P'_1 = 150$, $P'_2 = 153$, $P'_3 = 158$, $P'_4 = 148$, $P'_5 = 154$, $P'_6 = 158$, $P'_7 = 141$, $P'_8 = 163$, and $P'_9 = 179$. Fig.4.3(g) is resultant stego-block.

4.3.2 Example of Extraction Procedure

Let us see the data extraction from a block in Fig.4.4(a). Here the 9 SVs are, $P'_1 = 150$, $P'_2 = 153$, $P'_3 = 158$, $P'_4 = 148$, $P'_5 = 154$, $P'_6 = 158$, $P'_7 = 141$, $P'_8 = 163$, and $P'_9 = 179$.

Step 1: Plying Eqs.4.10 and 4.11, we generate stego-RB (Fig.4(b)) and stego-QB (Fig.4(c)) values accordingly.

Step 2: Referring Fig.4.4(b), write the remainders in 2 bits and adjoin all of them. We get $X = 10\ 01\ 10\ 00\ 10\ 10\ 01\ 11\ 11_2$.

Step 3: Calculate 8 difference values, $d_{1a} = (Q'_1 - Q'_2) = -1$, $d_{3a} = (Q'_3 - Q'_2) = 1$, $d_{7a} = (Q'_7 - Q'_2) = -3$, $d_{9a} = (Q'_9 - Q'_2) = 6$, $d_{1b} = (Q'_1 - Q'_8) = -3$, $d_{3b} = (Q'_3 - Q'_8) = -1$, $d_{7b} = (Q'_7 - Q'_8) = -5$, and $d_{9b} = (Q'_9 - Q'_8) = 4$.

Plying Eq.4.12, we get $\{l_1, u_1\}$, $\{l_2, u_2\}$, $\{l_3, u_3\}$, and $\{l_4, u_4\}$.

$d_{1a} \leq 0$ and $d_{1b} \leq 0$, so $\{l_1, u_1\} = \{0, \text{MIN } (38, 40)\} = \{0, 38\}$

$d_{3a} > 0$ and $d_{3b} \leq 0$, so $\{l_3, u_3\} = \{38+1, 40\} = \{39, 40\}$

$d_{7a} \leq 0$ and $d_{7b} \leq 0$, so $\{l_7, u_7\} = \{0, \text{MIN } (38, 40)\} = \{0, 38\}$

$d_{9a} > 0$ and $d_{9b} > 0$, so $\{l_9, u_9\} = \{\text{MAX } (38+1, 40+1), 63\} = \{41, 63\}$.

Plying Eq.4.4, generate n_1, n_3, n_7, and n_9. $n_1 = \text{MIN } (\lfloor \log_2 |38 - 0 + 1| \rfloor, 2) = 2$, $n_3 = \text{MIN } (\lfloor \log_2 |40 - 39 + 1| \rfloor, 2) = 1$, $n_7 = \text{MIN } (\lfloor \log_2 |38 - 0 + 1| \rfloor, 2) = 2$, and $n_9 = \text{MIN } (\lfloor \log_2 |63 - 41 + 1| \rfloor, 2) = 2$.

Using Eq.4.13, $b_1 \equiv |37 - 38| \pmod{2^2} = 1$, $b_3 \equiv |39 - 38| \pmod{2^1} = 1$, $b_7 \equiv |35 - 38| \pmod{2^2} = 3$, and $b_9 \equiv |44 - 38| \pmod{2^2} = 2$.

Each b_i (For i = 1, 3, 7, 9) is written in n_i bits. All these bits are adjoined. We got $Y = (01\ 1\ 11\ 10)_2$.

Step 4: Enumerate 3 difference values, $d_4 = |Q'_1 - Q'_7| = |37 - 35| = 2$, $d_5 = |Q'_2 - Q'_8| = |38 - 40| = 2$, and $d_6 = |Q'_3 - Q'_9| = |39 - 44| = 5$.

Enumerate, $n_4=\lfloor \log_2 d_4 \rfloor=1$, $n_5=\lfloor \log_2 d_5 \rfloor=1$, and $n_6=\lfloor \log_2 d_6 \rfloor=2$.

Using Eq.4.14, calculate $b_4 = 37 \bmod 2^1 = 1$, $b_5 = 38 \bmod 2^1 = 0$, and $b_6 = 39 \bmod 2^2 = 3$.

Each b_i (For i = 4, 5, 6) is written in n_i bits. These bits are adjoined. We got $Z = (1\ 0\ 11)_2$.

Step 5: X, Y, and Z are the extracted bits from the remainders, AQVD and QVC accordingly. By adjoining all these bits, $(X \,||\, Y \,||\, Z) = 10\ 01\ 10\ 00\ 10\ 10\ 01\ 11\ 11\ \ 01\ 1\ 11\ 10\ \ 1\ 0\ 11_{\ 2}$.

150	153	158
148	154	158
141	163	179

(a)

2	1	2
0	2	2
1	3	3

(b)

37	38	39
37	38	39
35	40	44

(c)

Fig.4.4 (a) stego-pixel block, (b) stego-RB, and (c) stego-QB

4.4 Results and Discussion

This methodology is realized in a computer with i5 processor using MATLAB. For experimentation and testing, images are taken from SIPI data-base [115].

The OIs are color in type. One pixel is 3 bytes. As per Fig.4.5(a), there are 3 parts R, G, B in a pixel. In RGB image, if each channel size is m×n, then we have the image size m×n×3 bytes. The color image is changed to a 2D array by adjoining the 3 channels, as in Fig.4.5(b). Here, the size of 2D array is m×p, wherein p=3×n. We cut this 2D array into 3×3 disjoint blocks and ply the hiding procedure.

The embedding of data shall continue into blocks until all secret bits are exhausted. After embedding the size of 2D array does not change. This output 2D array is stego 2D array. Now

we shall form the color image back from the stego 2D array. First, we separate the R, G, and B channels, each of size m× n. From all the 3 channels, take 1 byte from the same coordinate and make all the 3 bytes as a pixel. This can be repeated for i = 1 through m and j = 1 through n, the color image can be obtained in this way.

To extract the secret bits from the stego-color image, first we have to represent it in 2D array of bytes as in Fig.4.5. Then we cut the 2D array of bytes into 3×3 disjoint blocks. On each block we apply the data extraction procedure.

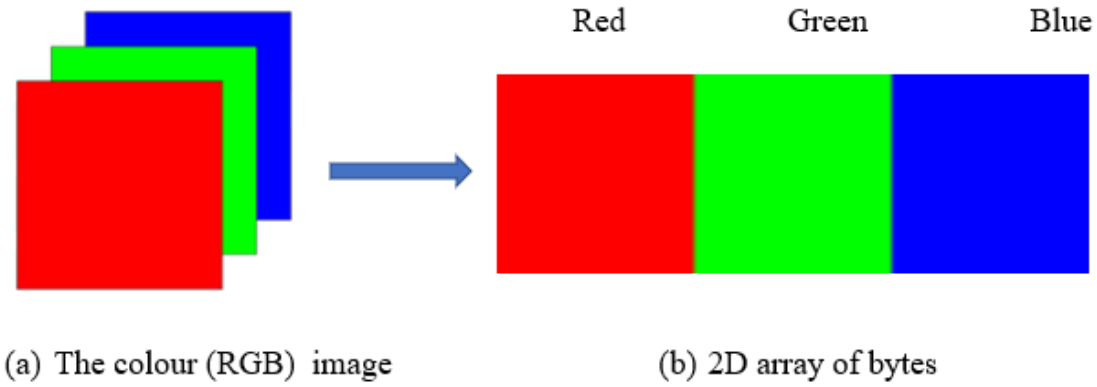


Fig.4.5 Transforming a colour (RGB) image into a 2D array of bytes

Fig.4.6 and Fig.4.7 depict a set of OI and SI respectively. Each SI of Fig.4.6 conceals 7 lakhs bits in itself. Here we cannot see any visual marks from the SIs. The SIs are visually same as their respective OIs.

Quality of proposed methodology is assessed by 6 metrics like, "(i) PSNR, (ii) HC, (iii) QI, (iv) BPB, (v) embedding time, and (vi) extraction time". With embedded data size of 1 lakh and 40 thousand bits, the assessed values are furnished in Table 4.1. Similarly, with embedded data size of 7 lakhs bits, the assessment values are furnished in Table 4.2. From these tables, we can see that the HC is 3.21 BPB. PSNR is 46.78 dB when data size 1 lakh 40 thousand bits and PSNR is 39.74 dB, with the data size is 7 lakh bits.

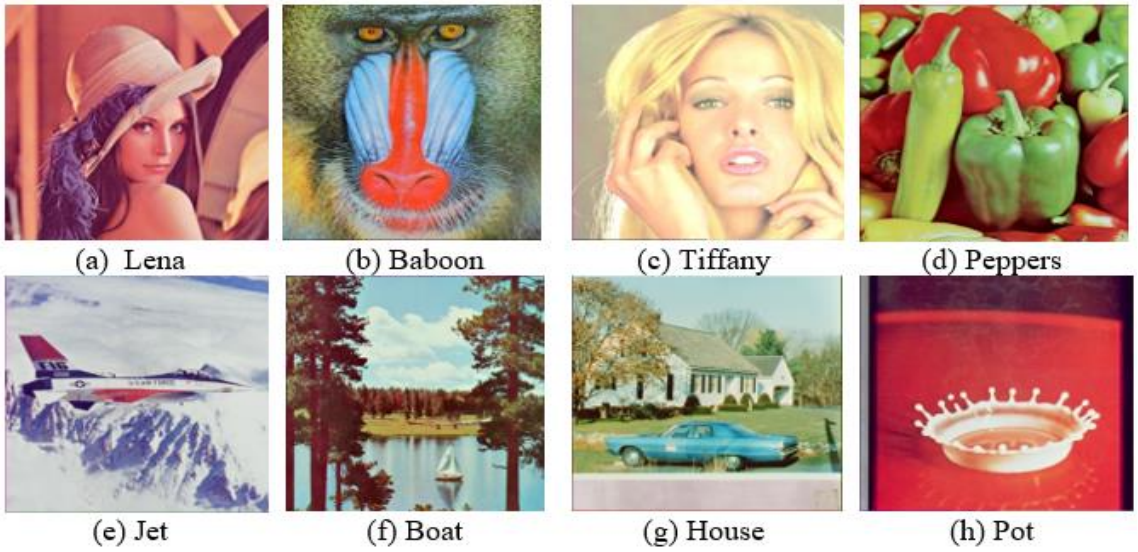


Fig.4.6 Test images

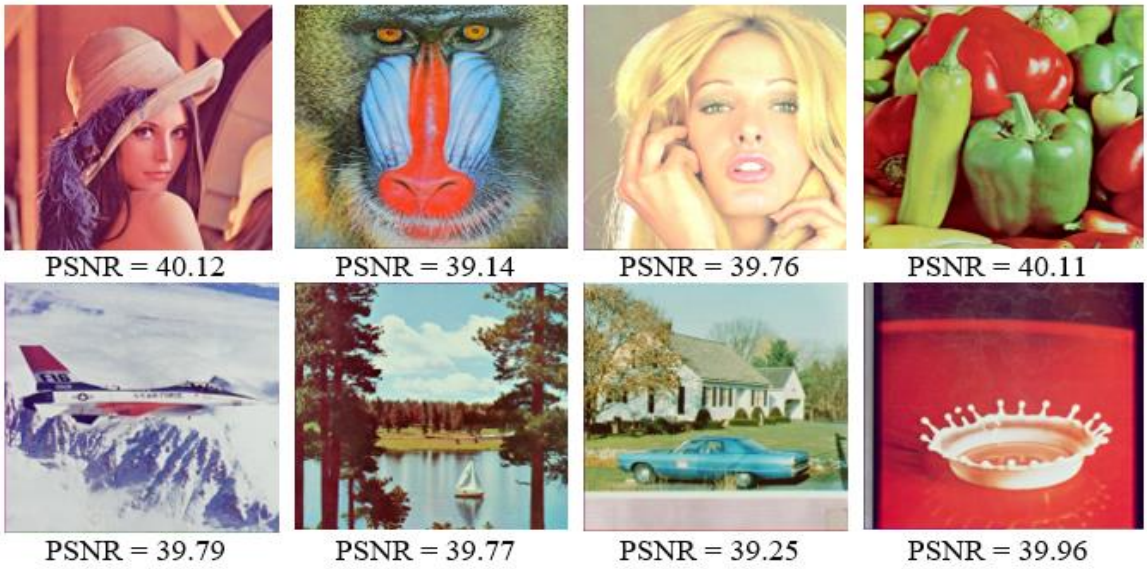


Fig.4.7 The SIs with 7,00,000 bits camouflaged in each of them

One may think that because of the integration of the 3 approaches RR, AQVD, and QVC, the processing time will be higher. It is not true. The truth is that all the 3 approaches are plied on a single block one after the other, so it takes only few second for embedding and extraction to do. Table 4.1 says that embedding time is 2.11 seconds and extraction time is 1.77 seconds, when the data size is 1 lakh 40 thousand bits Table 4. 2 says that embedding time is 9.89 seconds and extraction time is 8.4 seconds, when the data size is 7 lakhs bits.

Table 4.1 Estimated PSNR, HC, QI, and BPB values of proposed methodology when 1,40,000 (one lakh and forty thousand) bits are concealed

Image (512×512×3 bytes)	PSNR in dB	HC in bits	QI	BPB	Embedding time (seconds)	Extraction time (seconds)
Lena	47.05	2463031	0.9998	3.13	2.36	1.99
Baboon	45.99	2814096	0.9997	3.58	1.88	1.63
Tiffany	46.95	2461936	0.9996	3.13	2.11	1.87
Peppers	46.78	2497356	0.9998	3.17	2.09	1.73
Jet	46.82	2415669	0.9996	3.07	2.13	1.75
Boat	46.58	2575366	0.9998	3.27	2.10	1.73
House	46.96	2553991	0.9997	3.25	2.08	1.71
Pot	47.09	2425381	0.9999	3.08	2.15	1.78
Average	46.78	2525853	0.9997	3.21	2.11	1.77

Table 4.2 Estimated PSNR, HC, QI, and BPB of proposed methodology when 7,00,000 (seven lakhs) bits are concealed

Image (512×512×3 bytes)	PSNR in dB	HC in bits	QI	BPB	Embedding time (seconds)	Extraction time (seconds)
Lena	40.12	2463031	0.9991	3.13	9.97	8.47
Baboon	39.14	2814096	0.9987	3.58	9.19	8.00
Tiffany	39.76	2461936	0.9980	3.13	10.06	8.53
Peppers	40.11	2497356	0.9993	3.17	9.92	8.50
Jet	39.79	2415669	0.9982	3.07	10.22	8.56
Boat	39.77	2575366	0.9993	3.27	9.81	8.32
House	39.25	2553991	0.9987	3.25	9.82	8.33
Pot	39.96	2425381	0.9994	3.08	10.12	8.50
Average	39.74	2525853	0.9988	3.21	9.89	8.4

Table 4.3 shows a comparison of this scheme with existing APVD schemes. The readings in this Table are the average for 8 test images. This table clarifies that the BPB of the proposed scheme is 3.21, it is higher than the 3 existing schemes. Fig.4.8 says a diagrammatical comparison of BPB and PSNR of proposed scheme with 3 existing APVD schemes.

If we compare the results of the proposed RR+AQVD+QVC scheme (Chapter 4) with the proposed QVD+PVC scheme (Chapter 3), we can notice that the scheme in Chapter 3 possesses lower PSNR value, but higher BPB value as compared to the scheme described in this chapter.

Table 4.3 Comparison of Estimated PSNR, HC, QI, and BPB values with existing APVD methodologies when 1,40,000 (one lakh and forty thousand) bits are camouflaged

Technique	PSNR	HC	QI	BPB
Luo et al.'s (2011) 1-by-3 APVD [9]	48.45	294988	0.9998	0.37
Swain's (2016) 2-by-2 APVD [10]	45.37	1365075	0.9997	1.61
Pradhan et al.'s (2017) 2-by-3 APVD [53]	50.93	1445645	0.9998	1.83
Proposed 3-by-3 RR, AQVD and QVC	46.78	2537455	0.9997	3.21

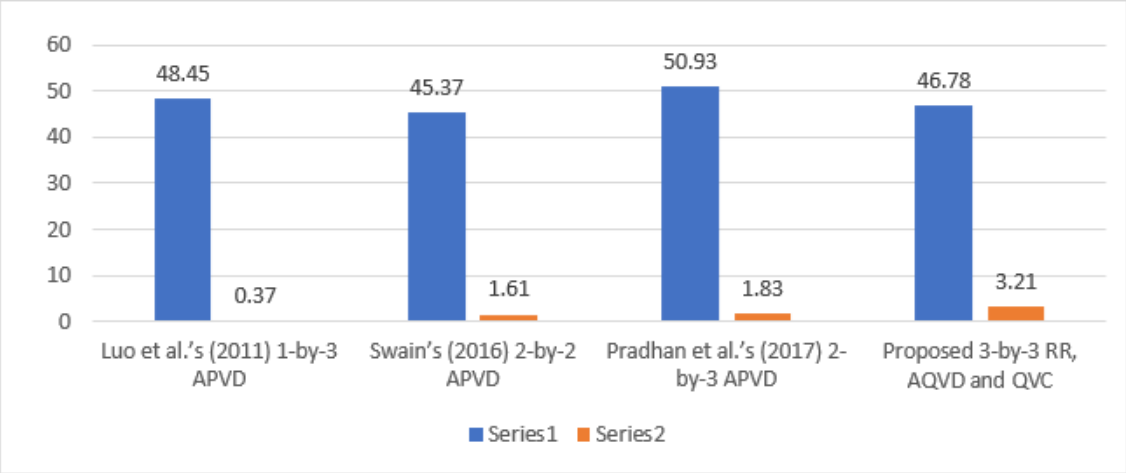


Fig.4.8 Performance comparison of proposed methodology with 3 existing APVD techniques, where series1 is PSNR and series2 is BPB

Table 4.4 makes a comparison of the proposed scheme with existing PVD/QVD schemes. The readings are averaged over 8 sample images. Table 4.4. says that that PSNR of proposed scheme is good. The PSNR of proposed scheme is 39.74 dB. Fig.4.9 is a bar graph for discriminating the BPB and PSNR of the proposed scheme with existing PVD/QVD schemes. This graph indicates that PSNR of proposed scheme is higher than 2 of the 3 techniques.

Table 4.4 Comparison of Estimated PSNR, HC, QI, and BPB values with the existing PVD/QVD techniques when 7,00,000 (seven lakhs) bits of data is hidden

Techniques	PSNR	HC	QI	BPB
Khodaei and Faez's (2012) 1-by-3 PVD [68]	38.57	2466275	0.9984	3.13
Pradhan et al.'s (2016) 3-by-3 PVD [79]	39.78	1885371	0.9985	2.39
Swain's (2019b) 3-by-3 QVD [80]	33.06	3581395	0.9947	4.55
Proposed 3-by-3 RR, AQVD and QVC	39.74	2525853	0.9988	3.21

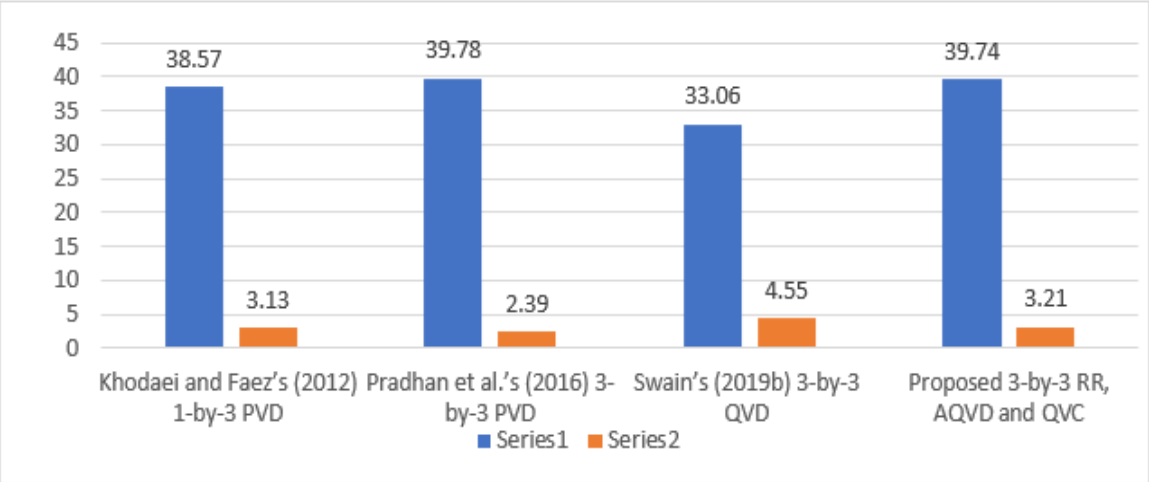


Fig.4.9 Performance comparison of proposed methodology with three existing PVD/QVD techniques, where series1 is PSNR and series2 is BPB

A comparison of proposed scheme with quantum-based techniques, Abd-El-Atty et al. [108], El-Latif et al. [107], and Peng et al. [109] is given in Table 4.5. This study records QI, PSNR, and HC. We can see from Table 4.5 that BPB wise quantum- based schemes are lesser than the proposed scheme. But, PSNR wise quantum-based schemes are good.

Table 4.5 Comparison with the quantum-based techniques

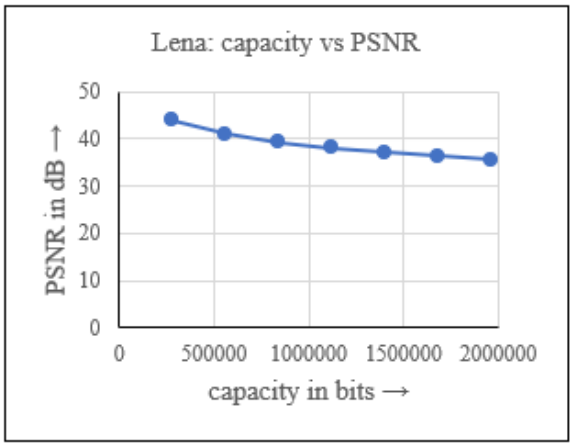
Algorithm	PSNR	HC	QI
El-Latif et al. (2019): A1 [107]	52.92	2-bits/24-bits	0.9887 (maximum)
El-Latif et al. (2019): A2 [107]	44.24	6-bits/24-bits	0.9437 (maximum)
El-Latif et al. (2019): A3 [107]	44.48	2-bits/8-bits	0.9459 (maximum)
Abd-El-Atty et al. (2020) [108]	44.23	2-bits/8-bits	0.9487
Peng et al. (2019) [109]	48.59	2-bits/24-bit	0.9727
Proposed 3-by-3 RR, AQVD and QVC	39.74	3.21 bits/8-bit	0.9988

Table 4.6 records the PSNR values of SIs with varied amounts of hidden data. The column 1 lists the test images. Column 2 lists the PSNR values when 2 lakhs 80 thousand bits of data are buried. Column 3 lists the PSNR values when five lakhs and sixty thousand (5,60,000) bits of data are buried. Column 4 lists the PSNR values when eight lakhs and forty thousand (8,40,000) bits of data are buried. Column 5 lists the PSNR values when eleven lakhs and twenty thousand (11,20,000) bits of data are buried. Column 6 lists the PSNR values when fourteen lakhs (14,00,000) bits of data are buried. Column 7 lists the PSNR values when sixteen lakhs and eighty thousand (16,80,000) bits of data are buried. Column 8 lists the PSNR values when nineteen lakhs and sixty thousand (19,60,000) bits of data are hidden. It is quite natural to see from Table 4.6 that the PSNR values declines when the data size increases. Column 8 indicates that the PSNR value does not decline below 35 dB.

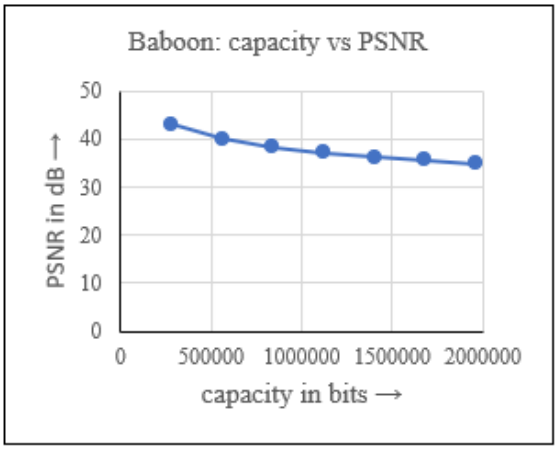
To further study on PSNR versus hidden data size, we plotted Fig.4.10 for the 8 test images. There are 8 plots. In each plot, the hidden data size (in bits) is represented along x-axis and PSNR is represented along y-axis. Fig.4.10(a) is for Lena image. Fig.4.10(b) is for Baboon image. Fig.4.10(c) is for Tiffany image. Fig.4.10(d) is for Peppers image. Fig.4.10(e) is for Jet image. Fig.4.10(f) is for Boat image. Fig.4.10(g) is for House image. Fig.4.10(h) is for Pot image. If we see the PSNR versus capacity plots for all the 8 images, we can notice that the PSNR value never declines below 35 dB.

Table 4.6 capacity vs PSNR for various images in the proposed RR, AQVD and QVC technique

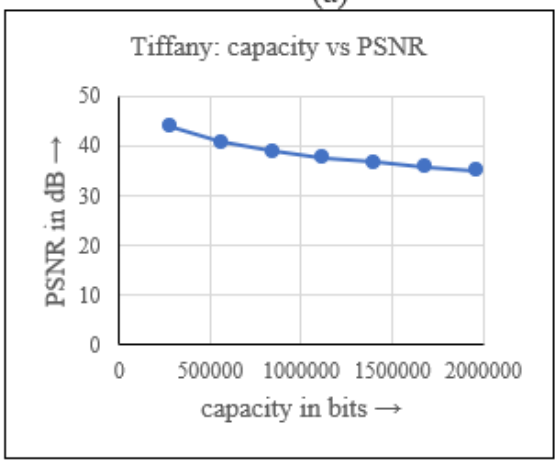
Image (512×512×3 bytes)	PSNR when 2,80,000 bits are hidden	PSNR when 5,60,000 bits are hidden	PSNR when 8,40,000 bits are hidden	PSNR when 11,20,000 bits are hidden	PSNR when 14,00,000 bits are hidden	PSNR when 16,80,000 bits are hidden	PSNR when 19,60,000 bits are hidden
Lena	44.06	41.09	39.32	38.05	37.06	36.28	35.61
Baboon	43.12	40.11	38.38	37.21	36.31	35.60	34.96
Tiffany	43.88	40.78	38.95	37.68	36.64	35.78	35.06
Peppers	43.95	41.06	39.94	38.12	37.17	36.41	35.78
Jet	43.83	40.78	38.97	37.60	36.58	35.79	35.13
Boat	43.74	40.75	38.95	37.71	36.78	36.03	35.39
House	42.96	40.11	38.54	37.35	36.47	35.68	35.03
Pot	44.05	40.95	39.13	37.84	36.81	35.97	35.27
Average	43.69	40.70	39.02	37.69	36.72	35.94	35.27



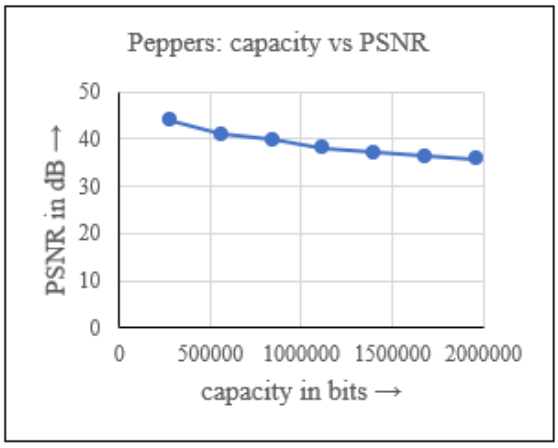
(a)



(b)



(c)



(d)

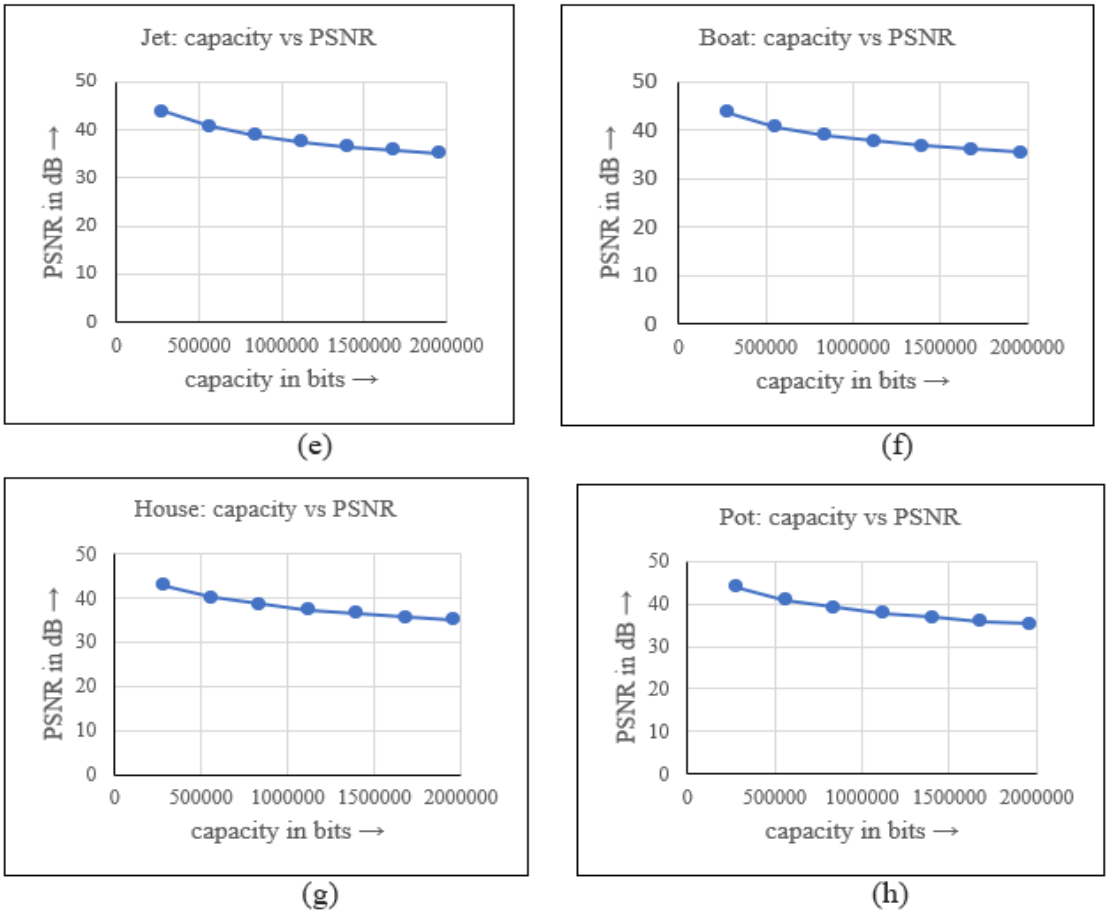


Fig.4.10 capacity vs PSNR graph for the various images of proposed RR, AQVD and QVC technique, (a) Lena: capacity vs PSNR, (b) Baboon: capacity vs PSNR, (c) Tiffany: capacity vs PSNR, (d) Peppers: capacity vs PSNR, (e) Jet: capacity vs PSNR, (f) Boat: capacity vs PSNR, (g) House: capacity vs PSNR, and (h) Pot: capacity vs PSNR.

4.5 Security Analysis

The RS test has been performed using four pointing values R_m, R_{-m}, S_m and S_{-m}. As per this test, steganography is detected if "$R_{-m} - S_{-m} > R_m - S_m$". Steganography is un-detected if "$R_m \approx R_{-m} > S_m \approx S_{-m}$". For this methodology, the RS test over 4 images is projected in Fig.4.11. It is viewable from these 4 sub-diagrams that "$R_m \approx R_{-m} > S_m \approx S_{-m}$". Amid these 4 images, Tiffany has smoothness, Baboon has more edges. The smoothness in Lena and Peppers is moderate. Thus, from these notes it is proven that our proposed scheme is undetected by RS test.

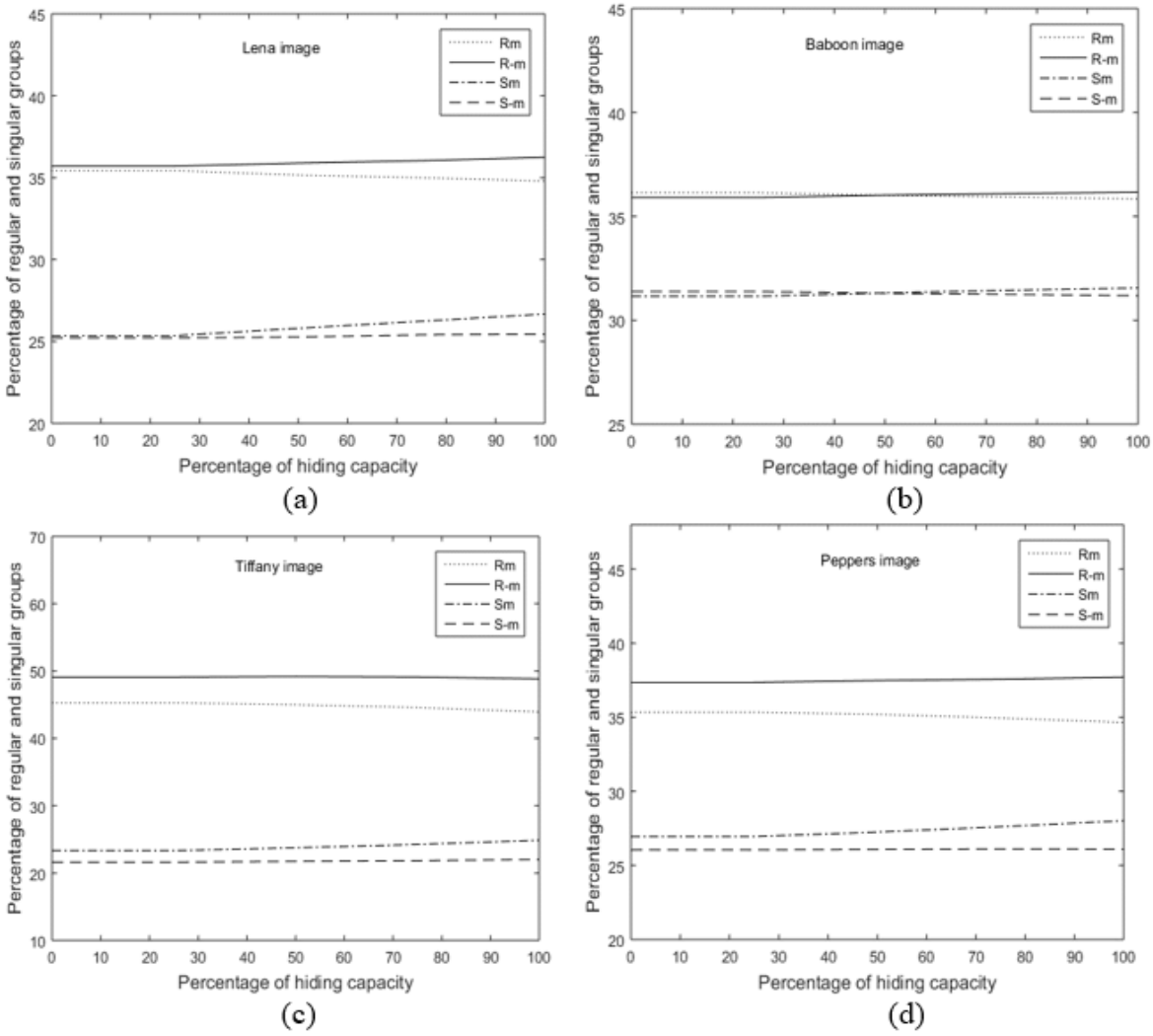


Fig.4.11 RS attack over the proposed technique

PDH test is done by drawing PDH graphs. It is a 2-dimensional graph with PD and its frequency on x and y-axes accordingly. It is said that PDH for the OIs will be smooth in appearance, but PDH for SIs generated from PVD methodologies will be curly in appearance. Fig.4.12 speaks the PDH for images Lena, Tiffany, Pepper, and Baboon. In all the 4 sub-figures, the solid lines and dotted lines project the OIs and SIs accordingly. It is observable from the 4 sub-figures that there exist no curly appearances in the PDH curves of the four SIs. So, PDH test is unable to detect this proposed scheme.

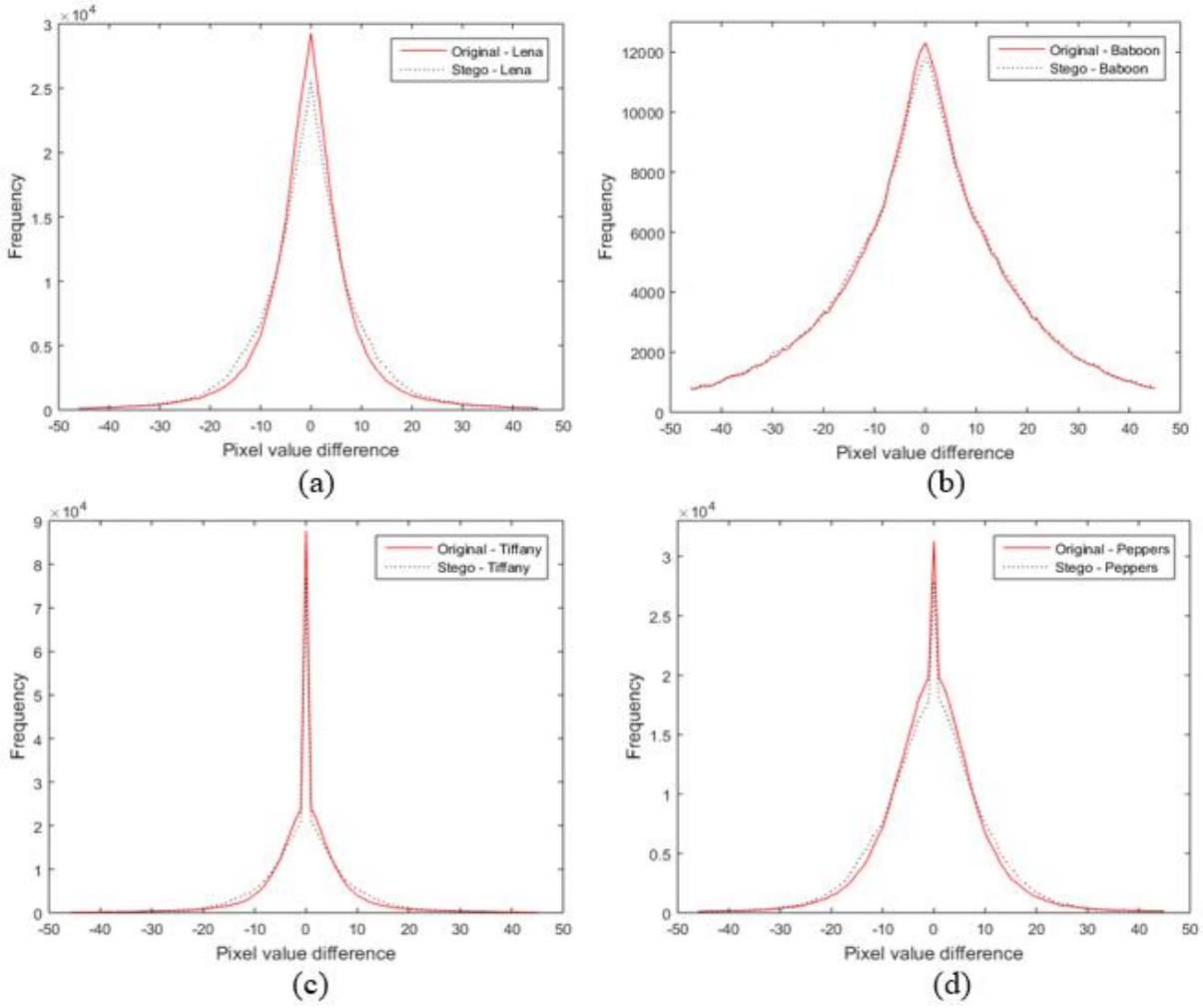


Fig.4.12 PDH attack over the proposed technique

4.6 Conclusion

This Chapter discussed a hybrid scheme for data hiding and extraction. It does in 3 stages, (i) RR, (ii) adaptive QVD, and (iii) QVC. Data hiding is done in 3×3 blocks. From a 3×3 block, we spawn 2 new blocks, namely RB, and QB. RB comprises of remainders and QB comprises of quotients. Each remainder is written in 2 bits. There are 9 quotients in QB. In 4 corner quotients AQVD is plied and in rest of the quotients QVC is plied. Experimental data says that, HC of this scheme is obviously greater than the existing APVD schemes. It is also seen that the PSNR of this scheme is greater related PVD/QVD schemes.

We can also judge that; the proposed scheme brought a trade-off in between HC and PSNR. From the discussion over RS and PDH tests, it is true to mention that the proposed scheme is secured. This is the initial research contributed with integration of the 3 ideas like RR, AQVD and QVC. It can be worth to mention that quotients Q_2, and Q_8 have been used as references, they are not used for hiding. In future we can modify the proposed procedures to hide data in these 2 quotients too.