

Chapter 6

Experimental Results

This Chapter covers experimental results and observations of applying theory or algorithm developed in Chapter 5. For implementation of the algorithm data mining tool namely Weka has been used. Matlab provided a fast implementing platform for verifying the concept. The initial Section of this Chapter deals with the results obtained as the outcome of data mining tool (Weka) and the subsequent results have been obtained in the Matlab software. Weka produced the Decision Tree to generate decision rules. This Chapter describes the results of various experiments to show the robustness of the algorithms against image processing operations.

6.1 Results from Datamining Approach

- Input to Weka software is the Excel file containing the following attributes: *DC*, *AC1*, *AC2*, *AC3*, *AC4*, Target
- Records produced for necessary training i.e. number of Instances is: 20,480

Generating the decision tree is done using Weka 3.4, an open source software in which an input file is provided which contains all the records and the target block is specified as 'A'. A partial decision tree obtained is as shown in Figure 5.2, where the

leaf nodes corresponds to target as ‘A’ class. The output generated from the software module is as follows:

Test mode: 10-fold cross-validation

Time taken to build model: 0.14 seconds

Description	Count/Result	Result in %
Total Number of Instances	20480	-
Correctly Classified Instances	17967	87.7395%
Incorrectly Classified Instances	2419	11.8115%
Unclassified Instances	94	0.459 %
Kappa statistic	0.597	-
Mean absolute error	0.1523	-
Root mean squared error	0.2838	-
Relative absolute error	46.19%	-
Root relative squared error	69.99%	-

Table 6.1: Summary of results (Output from Weka)

Based on stratified cross-validation the following results are obtained as the outcome of the training. Table 6.1 provides the summary of result produced by Weka. Table 6.2 gives the detailed accuracy by class and confusion matrix is shown in Table 6.3. the outcome of the training.

TP Rate	FP Rate	Precision	Recall	F-Measure	Class
0.964	0.432	0.894	0.964	0.928	N
0.568	0.036	0.806	0.568	0.667	A

Table 6.2: Detailed accuracy by class

N	A	Classified as
15549	583	N
1836	2413	A

Table 6.3: Confusion matrix

6.2 General Results and Discussion

For testing the designed algorithm well known images e.g. Lena, Bridge, Sail Boat, Sail Ship, Cartoon, etc. are used. Image blocks obtained in gray scale images, where embedding is carried out using Decision Tree rules are shown in Figure 6.1(a) and Figure 6.1(b) for Lena and Barbara images respectively. The watermarked images after embedding are shown in Figure 6.2(a) and Figure 6.2(b).

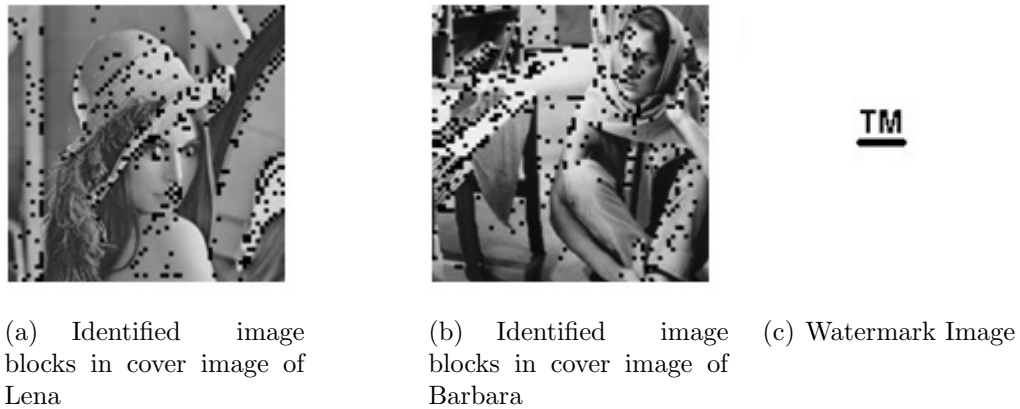


Figure 6.1: Gray scale Lena and Barbara images



Figure 6.2: Watermarked Lena and Barbara images

Quality factor is the standard JPEG parameter. It is a number between 0 and 100; higher numbers mean higher quality (less image degradation due to compression), but the resulting file size, however would be larger. Figure 6.3 and Figure 6.4 shows the extracted watermark from various images with varying quality factors.

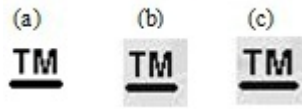


Figure 6.3: Extracted watermark



Figure 6.4: Extracted watermark with different JPEG quality factor

The correlation graph with variation in quality factor is as shown in Figure 6.5 It is observed that with the quality factor of 10, the watermark is almost destroyed.

Color logo image watermarking was also implemented and based on the training the

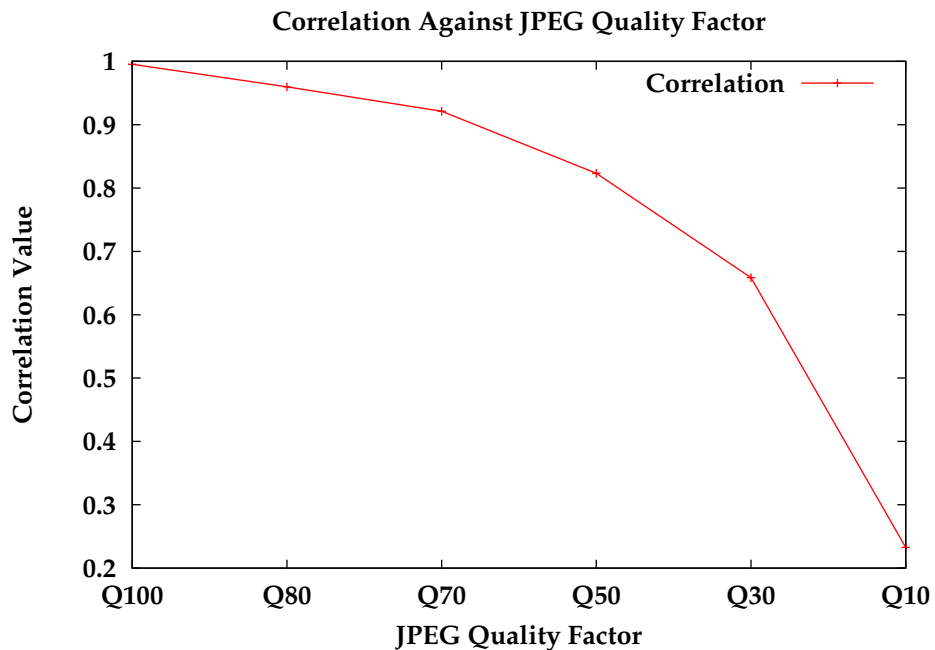


Figure 6.5: Correlation graph: Correlation Vs JPEG Quality factor

embedding was performed in to the R , G and B planes with different sets of images and the result obtained in PSNR is as shown below for images like Color Lena, Sail Boat, Cartoon, Bridge, Sea_Shore and Sail_Ship of size 512×512 as shown in Table

6.4 and Figure 6.6 shows the graphical representation for the same. Color planes

Images	Red Plane	Green Plane	Blue Plane
Color Lena	50.8609	47.9892	49.77
Sail Boat	47.32	47.31	47.43
Cartoon	47.43	49.75	47.55
Bridge	47.72	47.44	47.47
Sea_Shore	47.51	47.55	47.54
Sail_Ship	47.43	47.42	47.42

Table 6.4: PSNR values obtained in the RGB planes in color images after watermarking

in the Sail Boat, Color Lena, Cartoon, Bridge, Sea_Shore and Sail_Ship image where the embedding blocks have been identified is as shown in Table 6.5. As color image has R , G and B planes in each of these three planes, the intensity is different. So there is a need to choose proper threshold.

The threshold increases based on the intensity of the R , G and B components actually present to embed the watermark i.e. for reddish looking image such as Color Lena, we can keep higher threshold for getting suitable blocks in Red plane, but such higher threshold can not be kept for Green and Blue planes because this much higher threshold does not provide enough number of blocks for embedding.

In such cases threshold value can be lowered down to find sufficient number of blocks. This threshold according to our analysis is in the range of 500 to 1000.

Figure 6.8 shows the watermarked images of Figure 6.7 as the watermarked color logo inserted in the cartoon image in different image planes.

The visual results show that the data, could very well be inserted into the Blue Plane and some artifacts could be seen in the Red and Green plane in the background. Table 6.6 shows the result of the recovered watermark from the different planes. These

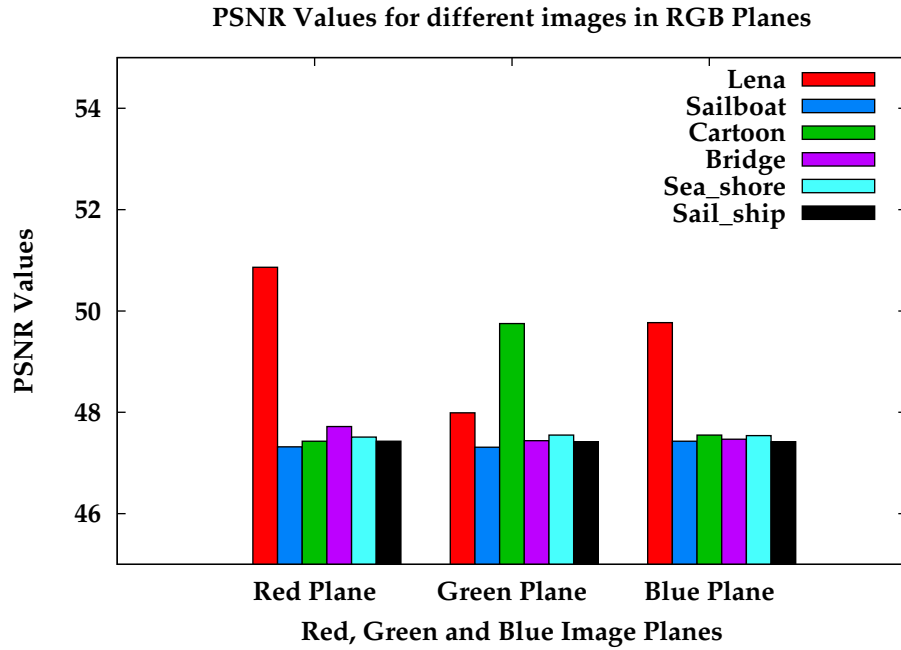


Figure 6.6: Graphical representation of the PSNR values obtained during watermarking in different image planes

C

Figure 6.7: Color logo watermark

watermarks are extracted after the storage of keys as shown in Section 6.7.

Table 6.7 shows a few cropped images of Color Lena, Sailboat and Cartoon along with the recovery of watermark. This attack is done intentionally to check for the recovery of watermark. The cropping is performed after the keys are stored within the image and the location of all these keys are preserved at distributed places as mentioned in Section 6.7.

A detailed analysis of how many blocks are available as ‘A’ category blocks for embedding in different images is as shown in Table 6.8. The watermark of size of 32×32 required such 256 blocks, however in the implementation it is observed that, this

	R Plane	G Plane	B Plane
Sail Boat			
Color Lena			
Cartoon			

Table 6.5: Watermarking block where embedding is performed in the RGB planes

Red Plane (a)	Green Plane (b)	Blue Plane (c)

Table 6.6: Recovered watermark from the cartoon image in RGB plane

approach is having enough embedding places and few images have almost double the required capacity to embed. Thus depending upon the size of the watermark to be embedded this approach could be used in a scalable manner.

Table 6.9 shows the ‘A’ category blocks identified within the bridge image, which one can obtain for any images for embedding.



Figure 6.8: Sample watermarked images of cartoon image in RGB plane

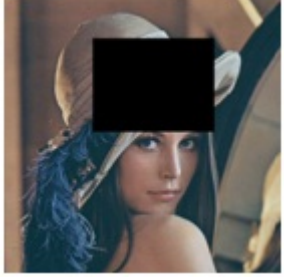



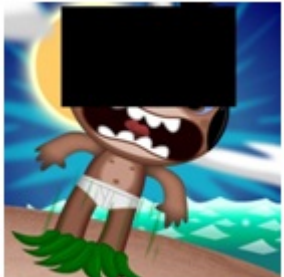

Image	Cropped Image	Recovered Watermark
Color Lena (Cropped 20 % approximately)		
Sail Boat (Cropped 60 % approximately)		
Cartoon (Cropped 25 % approximately)		

Table 6.7: Cropped and recovered watermarked images from Blue plane

DC Coefficient	Color Lena	Sea Shore	Sail Ship	Bridge	Sail Boat	Cartoon
300	501	366	714	729	425	820
400	493	361	683	708	425	764
500	469	356	646	692	425	725
600	454	353	625	675	419	704
700	434	350	611	649	395	673
800	413	344	529	622	353	641

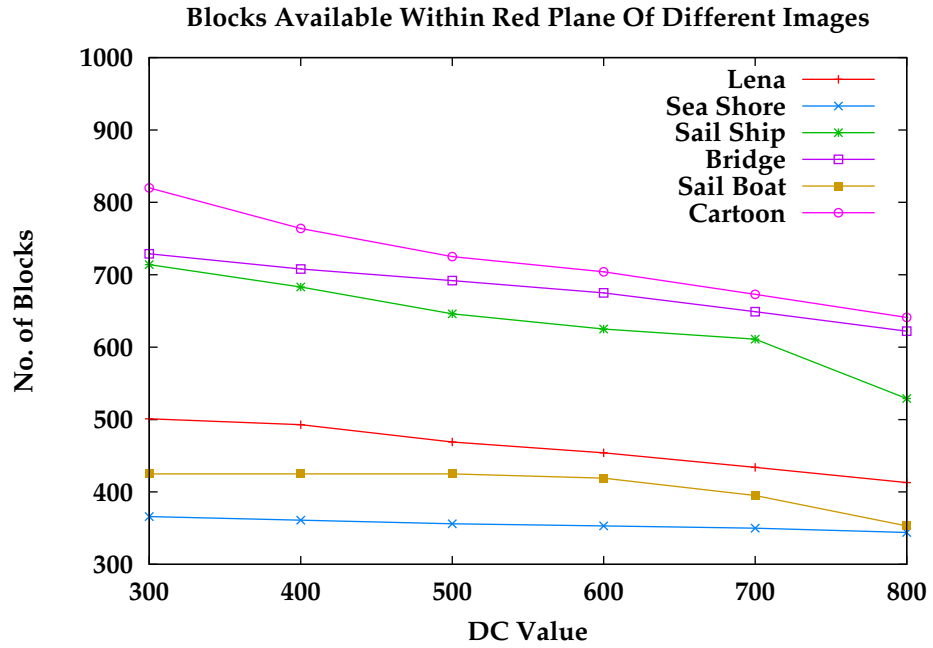
Table 6.8: Blocks identified as ‘A’ category blocks against different DC strength of test images in red plane.

DC Coefficient	Red	Green	Blue
300	729	742	693
400	708	729	683
500	692	712	661
600	675	689	637
700	649	666	608
800	622	640	574

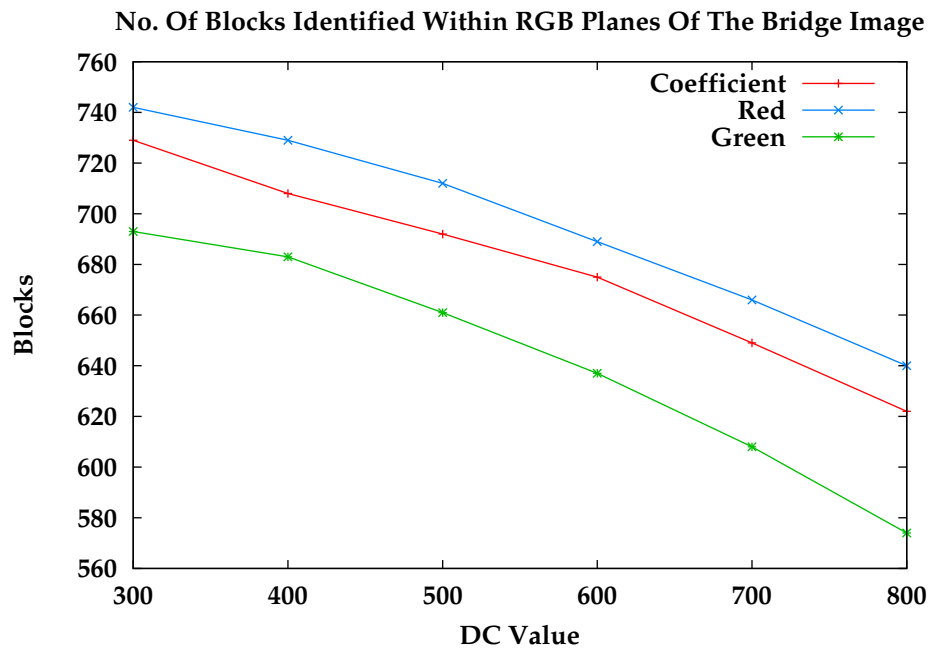
Table 6.9: Blocks identified against different DC strength in Bridge images in Red, Green and Blue image planes.

Figure 6.9(a) shows the plot against different DC value and the ‘A’ category blocks in various images and Figure 6.9(b) shows the plot against different DC value and the identification of ‘A’ category blocks in the bridge image in all the R , G and B image planes.

The keys generated by the embedding function are the place where the part of the watermark is stored. For a Bridge image the keys produced for embedding the watermark of size 32×32 is as shown below, these numbers are nothing but the block



(a) Graph showing all 'A' category blocks against different values of DC in all images based on Table 6.8



(b) Graph showing all 'A' category blocks against different values of DC in Different planes of the Bridge image based on Table 6.9

Figure 6.9: Blocks identified in image plane(s)

numbers identified row by row in multiples of 8 as DCT is applied on the image of size 8×8 thus the numbers generated are $1, 2, \dots, n$, where n is required number of blocks.

While embedding the first watermark following key set is obtained with DC as 800, let us say this key set as set-1

40,41,42,43,46,51,54,64,73,90,104,105,107,108,112,116,119,120,125,152,154,156,172,173,174,176,184,186,189,194,200,228,233,239,251,257,288,296,301,302,312,315,317,320,333,343,344,371,394,408,434,435,438,439,440,441,442,445,452,457,471,505,510,512,535,571,572,574,590,595,596,604,607,620,627,630,632,637,680,684,690,693,694,704,714,730,741,746,748,752,756,758,762,766,768,777,781,783,785,788,790,805,809,810,819,823,827,853,867,868,869,881,882,884,886,889,890,891,893,894,896,908,937,939,943,946,950,951,952,981,988,995,1001,1013,1014,1015,1017,1018,1023,1029,1036,1042,1047,1066,1074,1075,1085,1087,1088,1096,1099,1111,1113,1118,1121,1124,1129,1132,1135,1142,1144,1145,1148,1150,1171,1177,1178,1181,1188,1196,1198,1203,1204,1205,1206,1207,1212,1213,1234,1235,1238,1240,1248,1252,1255,1258,1267,1270,1271,1273,1276,1280,1298,1304,1311,1317,1318,1325,1327,1330,1334,1338,1341,1343,1344,1363,1367,1371,1375,1382,1390,1397,1401,1404,1406,1407,1425,1429,1430,1431,1433,1435,1436,1438,1440,1442,1444,1446,1450,1453,1459,1461,1463,1466,1468,1469,1470,1488,1491,1496,1503,1505,1511,1515,1526,1534,1536,1553,1557,1561,1563,1565,1566,1574,1583,1588

A similar set of keys is be obtained to embed within the watermarked image for doing double watermarking; here are the keys obtained with DC as 400, while performing embedding in the earlier watermarked image to embed a different watermark. Let us say this set of keys as set-2

36,40,46,51,64,86,100,116,119,120,149,155,156,164,176,184,186,228,239,283,294,302,312,315,346,349,351,361,422,434,435,438,445,479,505,559,571,572,613,620,627,632,637,677,680,690,693,694,704,741,748,752,758,762,768,785,805,815,819,827,870,871,882,884,889,890,

894,934,935,**937,943**,944,**946,950**,999,**1001**,1007,1008,**1014,1015,1023,1042**,1062,1063,1064,
 1072,**1074,1075,1085,1087,1088**,1126,1128,**1135**,1136,**1142,1144,1145**,1150,1190,1196,1198,
 1200,**1203,1205,1212,1213,1255,1258**,1262,1265,**1267,1270,1271,1273,1280,1286,1298**,
1318,1325,1327,1329,1330,1334,**1341,1343,1344**,1351,**1382**,1389,**1390,1401,1404,1407**,
 1448,**1450**,1457,**1461,1466,1468,1469,1470,1511**,1512,**1515**,1516,1521,1534,1545,**1553**,
1574,1576,**1583**,1585,1589,1590,1592,1593,1598,1611,1619,1620,1621,1624,1632,1635,1639,
 1640,1643,1645,1647,1650,1651,1653,1658,1661,1664,1665,1677,1688,1689,1690,1691,1693,
 1699,1704,1708,1714,1716,1723,1725,1726,1730,1739,1748,1751,1758,1761,1768,1769,1773,
 1774,1781,1787,1788,1810,1811,1813,1817,1818,1820,1823,1825,1831,1833,1834,1839,1841,
 1842,1845,1854,1874,1885,1886,1895,1897,1900,1903,1905,1915,1933,1948,1953,1965,1966,
 1967,1968,1969,1973,1981,1982,1988,2000,2005,2006,2012,2015,2019,2021,2022,2029,2030,
 2033,2048,2054,2070

The image after performing the double watermarking in the red plane of the bridge image is as shown in Figure 6.10. Figure 6.11 and Figure 6.12 shows the recovery of watermark using different key sets. The keys shown in bold in set-1 and set-2 are those blocks which holds data for multiple watermarks.



Figure 6.10: Double watermarked Bridge image in red plane



Figure 6.11: Recovered second watermark from Figure 6.10 using key set-2

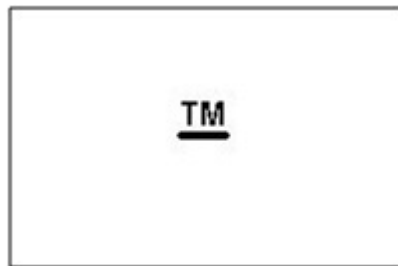


Figure 6.12: Recovered first watermark from Figure 6.10 using key set-1

Thus, we can say that the marked image itself could be used for embedding different watermark in the same image, resulting into multiple levels of watermarking, within the same image with different set of keys.

A similar set of keys for embedding a different watermark could be obtained for embedding in different planes of the image like green and blue, which further provides the flexibility for embedding multiple watermarks in different planes.

A very prominent feature observed in this implementation is that the previous watermark does not get damaged as new watermarks are embedded within the image. The overall effect is, that this mechanism allows multiple watermarks in multiple planes of the image.

However, a bigger size of watermark is to be embedded and if sufficient blocks are not available, the design will generate an error message in the encoder module and

will not proceed to embed. Normally the watermark to be inserted is always of less size, however for a special requirement of bigger size of watermark, the embedding algorithm can be modified by reducing the value of DC , so as to get more number of embedding blocks. By doing this, in the embedding side, one has to see that, by lowering the DC value to a very low level does not noticeably degrade the watermarked image. An attempt must always be to use higher DC value in the range of 700 to 1200. Thus, we can say that more the DC value the better the PSNR would be after embedding and lower will be the distortion observed by the Human Visual System. For performing multiple watermarking different values of DC in the encoder side will result into different identification of blocks and ultimately different set of keys. It is possible that the cover image itself is such that it is not giving good blocks for embedding even with the lower size of the watermark, in such cases embedding must be performed in different image planes.

6.3 Test for Robustness Against Various Image Processing Operations

Table 6.10 shows the test against histogram equalization and addition of Salt and Pepper noise of the order 2 % and 5 % having random distribution of pixels where noise is added. The extracted watermarks and their PSNR values are shown in Table 6.10. The watermarks are visually identifiable except one case of cartoon image. Similar thing is observed during addition of Gaussian noise, Poisson and Speckle noise, which are shown in Table 6.11. The presence of watermark is detected in all cases with different levels of noise and is visually recognizable.

Table 6.12 and Table 6.13 show the test for robustness against different values of gamma correction. The table also provide the recovered watermarks from the processed images as well as those from added noise. Watermarking is tested against gamma correction for values of γ ranging from 0.1 to 3.0. The results shows good

recovery of watermark for values of $\gamma > 0.4$ for all test images.

Figure 6.13 shows the plot of the PSNR values for the different images and different values of γ .



















Images	Histogram Equalization in Red plane	Recovered Water-mark	Salt and Pepper Noise 2 %	Recovered Water-mark	Salt and Pepper Noise 5 %	Recovered Water-mark
Color Lena	7.4975		8.1551		7.1716	
Sail Boat	6.6512		8.2391		7.3568	
Cartoon	7.3614		7.0195		6.3592	
Sea_Shore	7.216		8.2334		7.1893	
Bridge	7.1893		8.1385		7.4407	
Sail_Ship	7.2972		8.4654		7.343	

Table 6.10: PSNR values obtained after addition of noise in watermarked images



















Images	Gaussian Noise Mean 0 and Variance .01	Recovered Water- mark	Poisson Noise	Recovered Water- mark	Speckle Noise 3 %	Recovered Water- mark
Color Lena	7.9811		8.9569		6.96	
Sail Boat	7.9019		8.858		7.3753	
Cartoon	6.396		6.9433		6.0206	
Sea_Shore	7.638		8.6791		7.1938	
Bridge	7.7376		8.871		7.4501	
Sail_Ship	7.6776		8.9303		8.1385	

Table 6.11: PSNR values obtained after addition of noise in watermarked images











































Gamma Value	Color Lena PSNR	Recovered Water-mark	Sail Boat PSNR	Recovered Water-mark	Cartoon PSNR	Recovered Water-mark
0.1	5.4314		5.3087		6.3555	
0.2	5.8804		5.6845		6.6044	
0.3	6.2505		6.0037		6.6906	
0.4	6.5619		6.1585		6.7989	
0.5	6.7464		6.3445		6.9982	
0.6	6.8974		6.7224		7.0754	
0.7	7.0495		7.1057		7.279	
0.8	7.3614		7.3892		7.488	
0.9	7.5745		7.4313		7.5023	
1.0	9.0309		9.0309		8.2391	
1.5	7.5745		7.7025		7.6925	
2.0	7.4595		7.4595		7.488	
2.5	7.3753		7.3338		7.4548	
3.0	7.3109		7.2205		7.4079	

Table 6.12: PSNR values obtained against various parameter values for gamma correction











































Gamma Value	Sea_Shore PSNR	Recovered Water-mark	Bridge PSNR	Recovered Water-mark	Sail_Ship PSNR	Recovered Water-mark
0.1	5.473		5.4551		6.6122	
0.2	5.7318		5.7446		6.9307	
0.3	6.0003		6.141		7.0452	
0.4	6.2398		6.2937		7.0927	
0.5	6.4706		6.4108		7.1627	
0.6	6.7625		6.6277		7.1893	
0.7	6.9897		6.9939		7.27	
0.8	7.279		7.343		7.343	
0.9	7.4643		7.4785		7.4313	
1.0	9.0241		9.0309		9.0309	
1.5	7.5793		7.5696		7.6975	
2.0	7.2836		7.3476		7.6627	
2.5	7.1938		7.2474		7.5842	
3.0	7.1101		7.1188		7.5551	

Table 6.13: PSNR values obtained against various parameter values for gamma correction

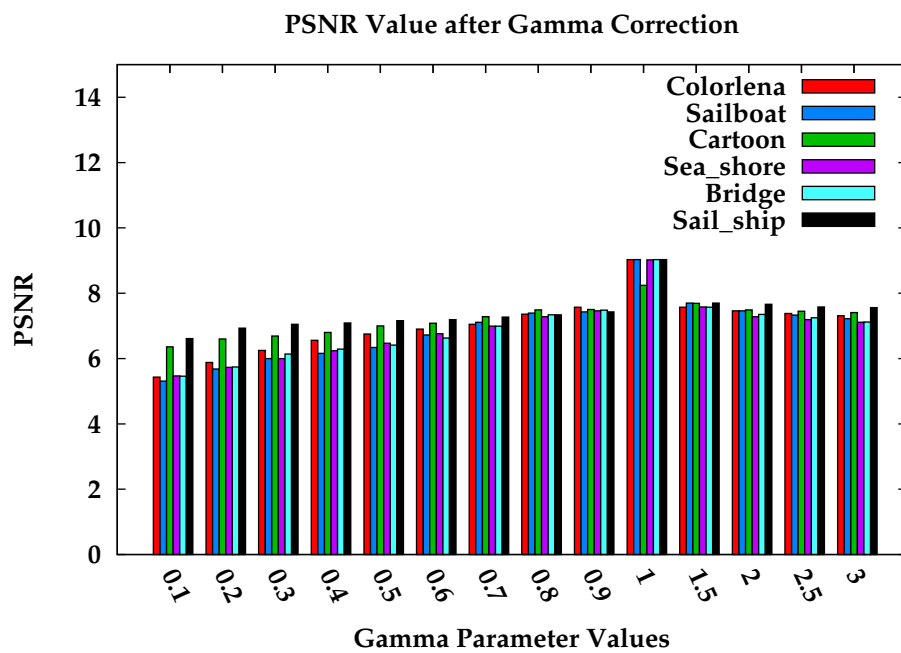


Figure 6.13: Plot of PSNR value after gamma correction

6.4 Geometric Attack

An attempt is made to check the algorithm against various geometric attacks like rotation, scaling, shearing and random local distortions. With all the geometric attacks applied, the algorithm is found to be sufficiently robust against such attacks. The outcome of these tests is shown in the following Subsections.

6.4.1 Attack against Rotation

The watermarked images are rotated by varying degrees in a counterclockwise direction around its center point. In the process of rotation, the output image becomes large enough to contain the entire rotated image. The pixels outside the rotated images are set to zero. We also cropped the rotated image to fit with the original size. The results with varying degree of rotation is shown in Figure 6.14. Result shows that the algorithm is able to withstand the attack against rotation. The watermark is clearly detectable in all cases and across test images.

6.4.2 Attack against Scaling

The watermarked images are scaled down to different size upto $1/2$, $1/4$ and even upto $1/8$. The watermarked image while scaling up, undergoes an interpolation using some methods. Our algorithm is tested with the bicubic and nearest neighbour interpolation methods. In the bicubic interpolation, the output pixel value is a weighted average of pixels in the nearest 4×4 neighborhood. Whereas in the nearest-neighbor interpolation, the output pixel is assigned the value of the pixel that the point falls within and no other pixels are considered. The results of scaling with different interpolation methods and varying sizes is as shown in Figure 6.15. The 512×512 image is down sampled by factor of 2, 4 and 8. The recovered watermarks are visually detectable for scaling by factor of 2. However the results may be acceptable for scaling by 4 and poor for scaling by 8.

6.4.3 Attack Against Shearing

A 2D affine transformation is applied on the watermarked image for performing horizontal and vertical shear of the image. The image size increased after applying shearing operation on the watermarked image. When horizontal and vertical both shearing operation is applied the image size gets reduced. We selected the actual image part and then converted it to the size of the actual image. The results of geometric shearing is as shown in Figure 6.16. Visual detection is possible for PSNR values above approximately 6 and it is also subject to type of image.

6.4.4 Random Local Distortion

An intentional attack is attempted to damage the watermarked images up to approximately 50 %, the recovered watermark shows noticeable recovery of watermark against such type of attack. Figure 6.17 shows the result obtained against random distortion attack applied across different images in different ways and the recovered watermark with the PSNR values obtained. The recovery of the watermark is visually recognizable considering the worst case damage of upto 50 % of different shapes and sizes distributed across the images.

6.5 Signal Processing Operations

The algorithm is checked for its robustness against various attacks like JPEG compression (Lossy and Lossless), various filtering attacks like averaging filter, disk averaging filter, Gaussian filter, motion blur filter, sharpened image filter, Laplacian filter, Prewitt and Sobel filter and Laplacian of Gaussian filter. Multiple watermarks are also attempted and found to be robust as shown in Section 6.2. Addition of noise is also tested and the design is found to be robust as shown in Section 6.3.

6.5.1 Attack against JPEG Lossy Compression

After the watermark is added within the cover image an attempt is made to check with various quality factor (Q) of JPEG compression algorithm in the range of 10 to 100. Figure 6.18 shows various values of Q and the PSNR obtained along with the recovered watermark. The presence of watermark is visually noticeable from the quality factor of 30 which increases proportionally with the increase in the value of quality factor. An attempt to check with lossless mode of JPEG is also performed on the watermarked images, the PSNR value of approximately 7.0927 is obtained with no loss in the recovered watermark with all the images. In the images like cartoon where there is a constant color, let us say blue as the dominating color and if we embed the watermark in the red plane of the image, which is not the dominating color, the watermark is visually perceived with the increase in the quality factor of 50 in the presence of noise.

6.5.2 Averaging Filter

An averaging filter of 3×3 has been used to perform averaging. The filter is shown in Table 6.14 and the results are shown in Figure 6.19. The results show very good recovery of watermark image in all cases.

0.1111	0.1111	0.1111
0.1111	0.1111	0.1111
0.1111	0.1111	0.1111

Table 6.14: Average filter mask

6.5.3 Disk Circular Averaging Filter

The algorithm was checked against a circular averaging filter (pillbox) within the square matrix of side $2*radius+1$. Our method retains the watermark with a radius of

3, 4 and 5 as shown in Figure 6.20. The recovered watermarks are visually noticeable for radius of 3 and degrades for radius greater than 4.

6.5.4 Gaussian Filter

Rotationally symmetric Gaussian low pass filter of size 3 x 3 with standard deviation σ (positive) is applied on the watermarked images. The default value for σ is 0.5. The design is tested against 0.1, 0.5 and 0.9 values of σ as shown in Figure 6.21. The recovered watermark images are very clear. Table 6.15, Table 6.16 and Table 6.17 provides the mask applied with varying values of σ as 0.1, 0.5 and 0.9 respectively.

0	0	0
0	1	0
0	0	0

Table 6.15: Gaussian filter mask with $\sigma = 0.1$

0.0113	0.0838	0.0113
0.0838	0.6193	0.0838
0.0113	0.0838	0.0113

Table 6.16: Gaussian filter mask with $\sigma = 0.5$

0.0673	0.1248	0.0673
0.1248	0.2314	0.1248
0.0673	0.1248	0.0673

Table 6.17: Gaussian filter mask with $\sigma = 0.9$

6.5.5 Motion Blur Filter

A motion blur filter is applied on the watermarked images which is a convolution with shifted version of the image. The linear motion of a camera represented by linear shift in pixels and with an rotation of angle θ in a counter clockwise direction is applied

over the images. Different values of length/shift and θ have been attempted on the watermarked images and the approach is found to be sufficiently robust for moderate shifts and rotation such as length of 10 and rotation up to 20 degrees, above which the watermarks is not recoverable as shown in Figure 6.22.

6.5.6 Sharpen Image Filter

A 3×3 unsharp contrast enhancement filter is applied on the watermarked images. The unsharp filter from the negative of the Laplacian filter with parameter α is applied on the watermarked images. The value of α controls the shape of the Laplacian and must be in the range 0.0 to 1.0. The default value for α is 0.2. An attempt is made to check with various values of α and is found to be highly robust as shown in Figure 6.23. Very good recovery of watermarks is observed in all the cases.

6.5.7 Laplacian Filter

A 3×3 filter approximating the shape of the two-dimensional Laplacian operator is applied on the watermarked images. The parameter α controls the shape of the Laplacian and must be in the range 0.0 to 1.0. The default value for alpha is 0.2. The presence of hidden message is noticeable in two cases, where as in other cases watermarks are not recovered. The results after applying Laplacian filter on the watermarked images is shown in Figure 6.24.

6.5.8 Prewitt and Sobel Filter

A 3×3 Prewitt and Sobel filter is applied on the watermarked images that emphasizes horizontal edges using the smoothing effect by approximating a vertical gradient. The presence of watermark as shown in Figure 6.25 is rarely noticeable in some of the images. This may be attributed to the fact that majority of the blocks identified by algorithm are near edges [9].

6.5.9 Laplacian of Gaussian Filter

A rotationally symmetric Laplacian of Gaussian filter of size 5×5 with standard deviation σ (positive) is applied on the watermarked images. Different values of σ has been attempted as shown in Figure 6.26. The design is observed to show most of the extracted images having some hidden message with σ in the range of 0.1 to 0.5.

6.5.10 Dithering of pixels

Dithering changes the colors of pixels in the neighbourhood, so that the average color in each neighbourhood, approximates the original RGB color. Here no dithering means quantization in colors only. The extracted watermarks do not give any meaningful information as shown in Figure 6.27.

6.5.11 Uniform Quantization

Uniform quantization and minimum variance quantization differ in the approach used to divide the RGB color cube. The tolerance determines the size of the cube shaped boxes into which the RGB color cube is divided. For example, specifying a tolerance of 0.1, the edges of the boxes are one-tenth the length of the RGB color cube. The approach is not found to be robust against uniform quantization as shown in Figure 6.28.

6.5.12 Minimum Variance Quantization

With minimum variance quantization, the color cube is cut into boxes (not necessarily cubes) of different sizes, the size of the boxes depend on how the colors are distributed in the image. The approach is not found to be robust against this type of quantization. Watermarks could not be recovered as shown in Figure 6.29.

6.6 Specialized Attack based on Knowledge of method

Changes to the color or intensity falls under this attack. The design is tested with contrast stretching and adding constant values to the watermarked images. The results obtained for contrast stretching are shown in Section 6.6.1 and the results for addition of constant value to the image plane are as shown in Section 6.6.2

6.6.1 Contrast Stretching

Contrast Stretching is applied on the watermarked images. The stretching is performed in the range of 0.01 to 0.99 and the results obtained are shown in Figure 6.30

6.6.2 Addition of Constant to an Image

Watermarked images were added with different constant value in the Red image plane. The design is found to be robust against this attack as shown in Figure 6.31. However in the case of cartoon image the watermark could not be detected as the red color is totally missing in the image.











































Images	Degree of rotation						
	5	15	30	45	60	75	90
Color Lena	7.0280	6.8396	6.9015	6.6986	6.7585	6.7786	7.0927
Recovered Watermark							
Sail Boat	7.0495	6.9224	6.9015	6.8602	6.8396	6.9643	7.0927
Recovered Watermark							
Cartoon	6.9643	6.7786	6.7184	6.6591	6.8192	6.9015	7.0927
Recovered Watermark							
Sea_Shore	6.9643	6.8602	6.8192	6.7989	6.7786	6.9015	7.0927
Recovered Watermark							
Bridge	6.9224	6.8602	6.7786	6.7585	6.7384	6.8192	7.0927
Recovered Watermark							
Sail_Ship	7.0495	6.8602	6.8396	6.8808	6.9433	7.0067	7.0927
Recovered Watermark							

Figure 6.14: PSNR and recovered watermark after geometric rotation of varying degree





































Images	512→256→512 Scaling by ½ Bicubic	512→256→512 Scaling by ½ Nearest	512→128→512 Scaling by ¼ Bicubic	512→128→512 Scaling by ¼ Nearest	512→64→512 Scaling by 1/8 Bicubic	512→64→512 Scaling by 1/8 Nearest
Color Lena	7.0927	6.6200	6.3923	5.8217	5.2600	4.7248
Recovered Watermark						
Sail Boat	7.0927	6.5812	6.3009	5.6159	5.4195	4.9575
Recovered Watermark						
Cartoon	7.0280	6.7184	6.4669	6.3555	5.7894	5.7543
Recovered Watermark						
Sea_Shore	7.0067	6.6986	6.3190	6.0376	5.2743	4.8655
Recovered Watermark						
Bridge	6.9643	6.4481	6.1936	5.7255	5.4492	4.8138
Recovered Watermark						
Sail_Ship	7.0927	6.8192	6.4669	6.3555	5.7414	5.5698
Recovered Watermark						

Figure 6.15: PSNR and recovered watermark after geometric scaling of different sizes



















Images	Horizontal Shear 0.5	Vertical Shear 0.5	Horizontal and Vertical Shear 0.5
Color Lena	6.7585	6.2469	5.5851
Recovered Watermark			
Sail Boat	5.9868	6.3190	5.8542
Recovered Watermark			
Cartoon	6.5237	6.7786	6.0037
Recovered Watermark			
Sea_Shore	6.5046	6.3739	5.6626
Recovered Watermark			
Bridge	6.0206	6.2648	5.9868
Recovered Watermark			
Sail_Ship	6.2828	6.6395	6.6591
Recovered Watermark			

Figure 6.16: PSNR and recovered watermark after geometrically shearing in horizontal, vertical and both sides













Images	PSNR	Intentional Attack (Approximately 50 % damage) Images scaled to ¼ size	Recovered Watermark
Color Lena	5.7733		
Sail Boat	5.8055		
Cartoon	6.6200		
Sea_Shore	6.1760		
Bridge	6.0547		
Sail_Ship	6.3555		

Figure 6.17: PSNR and recovered watermark after intentionally added random local distortions





























































Images	Q=10	Q=20	Q=30	Q=40	Q=50	Q=60	Q=70	Q=80	Q=90	Q=100
Color Lena	5.0244	5.6314	6.1063	6.3739	6.7184	6.7786	6.9433	6.9643	7.0927	7.0927
Recovered Watermark										
Sail Boat	5.1199	5.4940	5.8217	6.1236	6.2828	6.3009	6.4481	6.7184	6.9015	7.0280
Recovered Watermark										
Cartoon	5.9200	6.0547	6.0376	6.1236	6.0890	6.5427	6.5237	6.6986	6.9015	7.0067
Recovered Watermark										
Sea_Shore	5.0515	5.5698	6.1585	6.6986	6.8396	6.9224	7.0280	7.0710	7.0067	7.0927
Recovered Watermark										
Bridge	5.0110	5.4940	5.9533	6.5427	6.5427	6.7786	6.9224	7.0280	7.0927	7.0927
Recovered Watermark										
Sail_Ship	5.4940	5.5698	6.1936	6.3555	6.8192	6.9224	7.0067	7.0710	7.0927	7.0927
Recovered Watermark										

Figure 6.18: PSNR values for different images with varying quality factor (Q) and the recovered watermarks for JPEG compression attack







Images	PSNR	Recovered Images
Color Lena	7.0927	
Sail Boat	6.9433	
Cartoon	6.9855	
Sea_Shore	6.8396	
Bridge	6.8192	
Sail_Ship	7.0710	

Figure 6.19: Averaging filter attack with PSNR and recovered watermark images































Images	Radius for circular averaging				
	3	4	5	6	10
Color Lena	6.6005	6.0376	5.5242	5.3029	5.0515
Recovered Watermark					
Sail Boat	6.2828	5.8217	5.5393	5.2886	5.1754
Recovered Watermark					
Cartoon	6.6986	6.3009	6.0890	5.8217	5.4940
Recovered Watermark					
Sea_Shore	6.2648	5.8705	5.6470	5.3754	4.9708
Recovered Watermark					
Bridge	6.2469	5.8542	5.7414	5.5091	5.1199
Recovered Watermark					
Sail_Ship	6.6788	6.4108	6.1760	5.9533	4.9047
Recovered Watermark					

Figure 6.20: Disk circular averaging filter attack with PSNR and recovered watermark images



















Images	Standard Deviation (σ)		
	0.1	0.5	0.9
Color Lena	7.0927	7.0927	7.0927
Recovered Watermark			
Sail Boat	7.0927	7.0927	7.0710
Recovered Watermark			
Cartoon	7.0280	7.0280	7.0280
Recovered Watermark			
Sea_Shore	7.0927	7.0927	6.9855
Recovered Watermark			
Bridge	7.0927	7.0927	6.9224
Recovered Watermark			
Sail_Ship	7.0927	7.0927	7.0927
Recovered Watermark			

Figure 6.21: Attack of Gaussian filter on watermarked images with PSNR and recovered watermarks

























Images	Length = 2 Theta = 5	Length = 10 Theta = 10	Length = 10 Theta = 20	Length = 20 Theta = 20
Color Lena	7.0927	6.3372	6.3923	5.6159
Recovered Watermark				
Sail Boat	7.0927	6.2828	6.1410	5.4195
Recovered Watermark				
Cartoon	7.0280	6.6200	6.6591	5.7573
Recovered Watermark				
Sea_Shore	7.0927	6.3739	6.2469	5.4641
Recovered Watermark				
Bridge	7.0927	6.2648	6.1760	5.6470
Recovered Watermark				
Sail_Ship	7.0927	6.384	6.6986	5.9366
Recovered Watermark				

Figure 6.22: Motion blur attack with varying values of length and θ along with PSNR and recovered watermarks





































Images	α for Laplacian filter					
	0.1	0.2	0.4	0.6	0.8	0.9
Color Lena	7.0067	7.0067	7.0067	7.0067	7.0495	7.0495
Recovered Watermark						
Sail Boat	6.7384	6.7184	6.6986	6.6986	6.6986	6.6986
Recovered Watermark						
Cartoon	7.0067	6.9855	6.9643	6.9643	6.9855	6.9855
Recovered Watermark						
Sea_Shore	6.8602	6.8396	6.8396	6.8602	6.8808	6.8602
Recovered Watermark						
Bridge	6.7989	6.7786	6.7786	6.7786	6.7989	6.7786
Recovered Watermark						
Sail_Ship	6.9855	6.9643	6.9643	6.9643	6.9643	6.9643
Recovered Watermark						

Figure 6.23: Contrast enhancement attack with varying α along with PSNR and recovered watermarks































Images	α for Laplacian filter				
	0.05	0.1	0.2	0.5	0.9
Color Lena	4.8526	4.8396	4.8526	4.8526	4.8655
Recovered Watermark					
Sail Boat	5.0651	5.0651	5.0651	5.0924	5.0924
Recovered Watermark					
Cartoon	4.5763	4.5763	4.5763	4.5763	4.5763
Recovered Watermark					
Sea_Shore	5.0379	5.0379	5.0651	5.0651	5.0651
Recovered Watermark					
Bridge	5.1476	5.1476	5.1476	5.1337	5.1337
Recovered Watermark					
Sail_Ship	4.6376	4.6376	4.6376	4.6499	4.6499
Recovered Watermark					

Figure 6.24: Laplacian filter with varying α along with PSNR and recovered watermarks













Images	Prewitt	Sobel
Color Lena	5.8542	5.8870
Recovered Watermark		
Sail Boat	5.6470	5.7097
Recovered Watermark		
Cartoon	5.0244	5.1061
Recovered Watermark		
Sea_Shore	5.7414	5.8055
Recovered Watermark		
Bridge	5.8870	5.9200
Recovered Watermark		
Sail_Ship	6.0890	6.1410
Recovered Watermark		

Figure 6.25: Prewitt and Sobel filter along with PSNR and recovered watermarks

















































Images	Standard Deviation (σ)							
	0.1	0.2	0.3	0.4	0.5	0.6	0.7	1.0
Color Lena	4.5885	4.6499	4.7501	4.7882	4.7882	4.8396	4.8786	4.9443
Recovered Watermark								
Sail Boat	4.6623	4.7248	4.8786	4.9443	4.9842	5.0924	5.1061	5.1476
Recovered Watermark								
Cartoon	4.4801	4.5521	4.5885	4.5885	4.5763	4.5763	4.5763	4.6007
Recovered Watermark								
Sea_Shore	4.7627	4.8786	4.9310	4.9047	4.9575	5.0651	5.1061	5.1337
Recovered Watermark								
Bridge	5.0651	5.0924	5.0787	5.0651	5.0924	5.1337	5.1476	5.1476
Recovered Watermark								
Sail_Ship	4.0607	4.4920	4.5885	4.6253	4.6376	4.6376	4.6376	4.6623
Recovered Watermark								

Figure 6.26: Laplacian of Gaussian filter with varying σ along with PSNR and recovered watermarks





































Images	Dither 8	No Dither 8	Dither 512	No Dither 512	Dither 2048	No Dither 2048
Color Lena	5.1476	5.1814	5.3458	5.1061	4.8010	4.8916
Recovered Watermark						
Sail Boat	5.1615	5.1754	4.7882	4.8396	4.6997	4.8786
Recovered Watermark						
Cartoon	5.9366	5.9700	6.0206	5.9868	5.6782	5.7097
Recovered Watermark						
Sea_Shore	5.2743	5.1894	5.6939	5.3463	5.1894	5.1476
Recovered Watermark						
Bridge	5.1615	5.1615	5.0515	5.1061	4.6747	4.5400
Recovered Watermark						
Sail_Ship	5.1476	5.7894	5.3901	5.0244	4.8138	4.6007
Recovered Watermark						

Figure 6.27: Dithering and Nodithering on watermarked images with different colors with its PSNR and recovered watermarks











































Images	Tolerance for uniform quantization						
	0.005	0.01	0.1	0.3	0.5	0.7	0.9
Color Lena	4.2484	4.2484	5.4492	5.2175	5.1337	5.1061	5.1061
Recovered Watermark							
Sail Boat	4.4327	4.4327	4.5642	5.2600	5.1892	5.1754	5.1754
Recovered Watermark							
Cartoon	5.4641	5.4641	5.8542	5.4048	5.2175	5.3463	5.3463
Recovered Watermark							
Sea_Shore	4.3976	4.3976	5.4641	5.2743	5.2175	5.1894	5.1894
Recovered Watermark							
Bridge	4.4210	4.4210	4.4445	5.2458	5.2316	5.2175	5.2175
Recovered Watermark							
Sail_Ship	4.9310	4.9310	5.0244	5.5393	5.3754	5.3608	5.3608
Recovered Watermark							

Figure 6.28: Uniform quantization with varying tolerance with its PSNR and recovered watermarks





































Images	RGB Color cube					
	4	8	16	32	128	256
Color Lena	5.0651	5.1476	5.1615	5.5242	5.6470	5.8055
Recovered Watermark						
Sail Boat	5.1615	5.1615	5.2175	5.2743	5.6005	5.2600
Recovered Watermark						
Cartoon	5.4048	5.9366	5.9700	6.0037	5.9034	5.7894
Recovered Watermark						
Sea_Shore	5.2034	5.2743	5.3754	5.4195	5.9034	5.9034
Recovered Watermark						
Bridge	5.1615	5.1615	5.2034	5.4790	5.7894	5.6159
Recovered Watermark						
Sail_Ship	5.6005	5.1476	5.6159	5.8379	6.3190	5.7414
Recovered Watermark						

Figure 6.29: Minimum variance quantization with different colors cubes with the PSNR and recovered watermarks







Images	PSNR	Recovered Images
Color Lena	6.6591	
Sail Boat	6.2469	
Cartoon	7.0280	
Sea_Shore	6.9643	
Bridge	6.6005	
Sail_Ship	5.9868	

Figure 6.30: PSNR and recovered watermark after contrast stretching































Images	Constant addition				
	20	40	60	80	100
Color Lena	5.9700	5.7255	5.0651	4.7755	4.5642
Recovered Watermark					
Sail Boat	6.0037	5.9034	5.6782	5.5091	5.0924
Recovered Watermark					
Cartoon	6.0376	6.0206	5.9200	5.8379	5.6782
Recovered Watermark					
Sea_Shore	5.8217	5.7573	5.3901	5.2458	5.0651
Recovered Watermark					
Bridge	6.0037	5.9533	5.6005	5.4492	5.0515
Recovered Watermark					
Sail_Ship	6.0037	5.9868	5.9868	5.9366	5.9200
Recovered Watermark					

Figure 6.31: PSNR and recovered watermark after constant value addition

6.7 Storage of Keys

A separate software module is designed in Matlab to perform steganography in Region of Interest (ROI). Here the user have to select the object image which in our case is the same in which we have done the embedding of the watermark. The keys are required to be used on the receiver side. So the key files which are prepared by the embedding module is stored inside the text file. This text file is now to be inserted into this image. The decoder module will unhide this text file and use it further to extract the embedded watermark. Since the keys are not known to the end user, also the usage of Arnold transform makes the mechanism more secure.

A major feature of this module is that the user interface allows the end user to

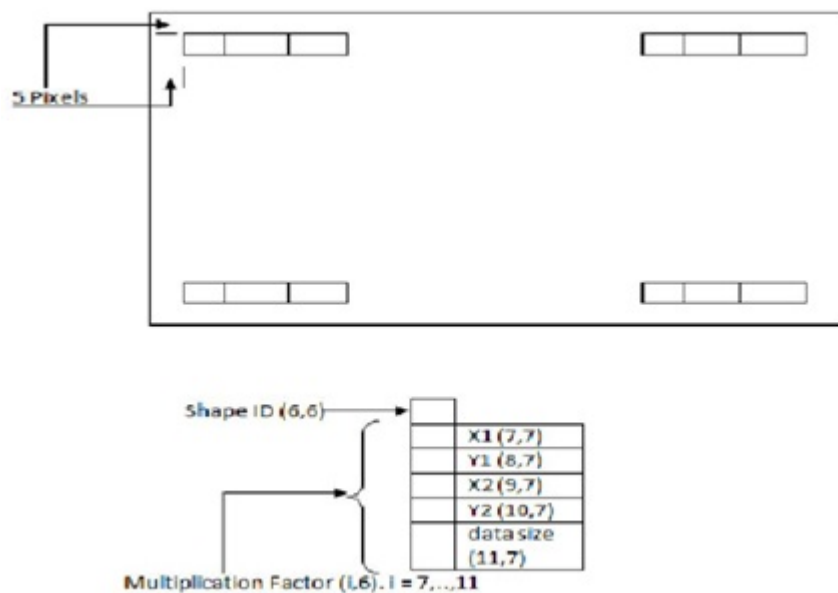


Figure 6.32: Embedding location

select any portion of the image to embed the keys. Also the information is managed in the image multiple times at different locations. The module under consideration identifies which portion of the image is attacked. If one of the corner information of the image is lost, then also the system locates the place where the keys are present.

This intelligence is provided into the system by embedding the location of storage at all the corners of the images in a specific way. Figure 6.32 provides a brief outline for the location where the storage of the keys is performed.

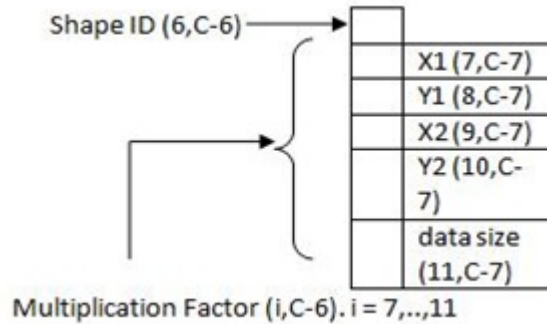


Figure 6.33: Upper right location for storage of keys within the image

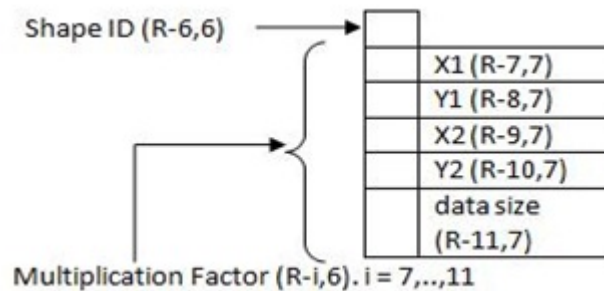


Figure 6.34: Bottom left location for storage of keys within the image

Figure 6.32 to Figure 6.35 show how the locations are stored into vector form at four different locations. Figure 6.36 shows the breakup of how 8 bit of information is stored within the R , G and B planes in the pixel of the image. Figure 6.37 and Figure 6.38 show interfaces provided to the user for storing the keys within the embedded image. So this particular image now contains both the watermark and also the keys. These keys are required on the decoder side to get back the watermark from the image.

The key storage module allows us to store multiple keys within the same image,

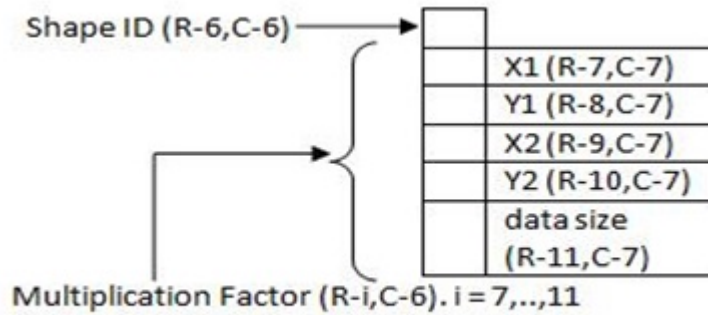


Figure 6.35: Bottom right location for storage of keys within the image

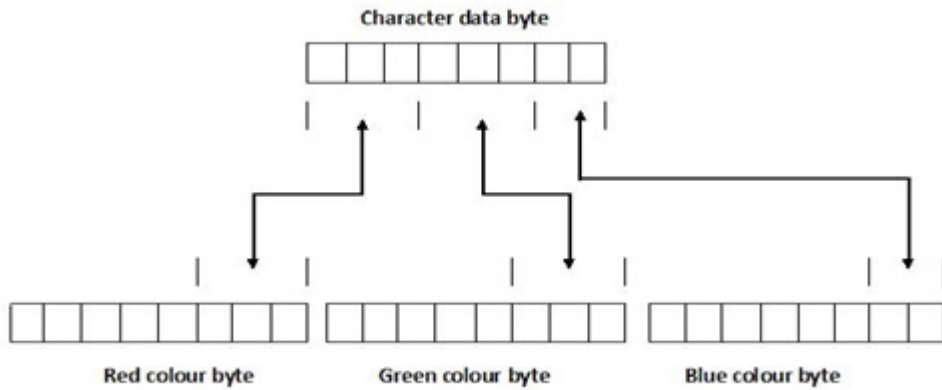


Figure 6.36: Simple LSB way to embed the keys into the pixel of R, G and B image plane.

and this is intimated to the user that the file in use is already having some information previously hidden. The design thus allows double steganography within the same image as shown in Figure 6.39.

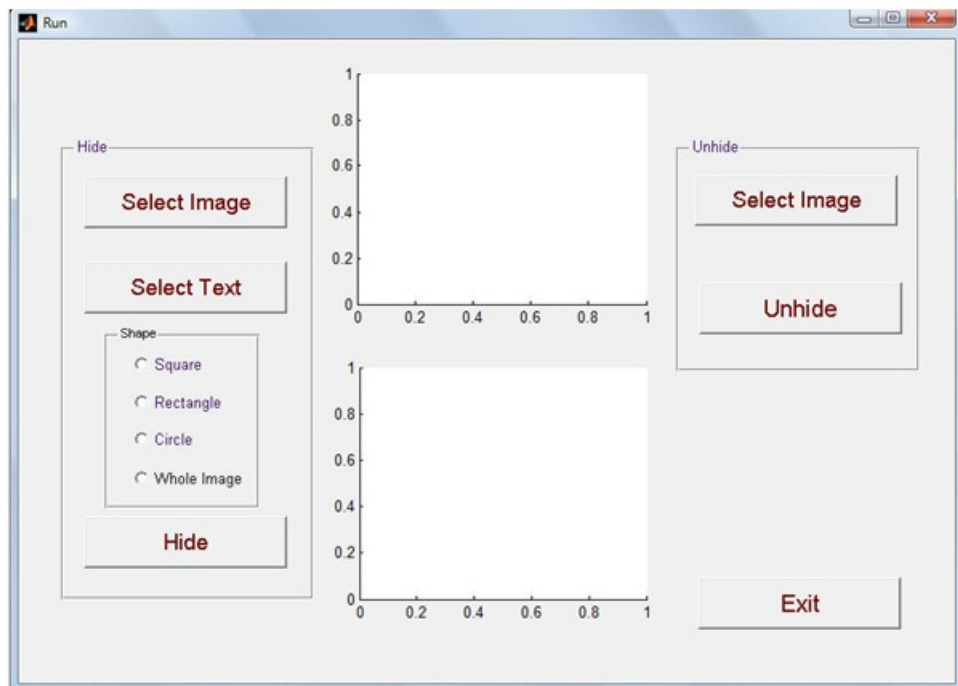


Figure 6.37: User screen to store keys within the image.

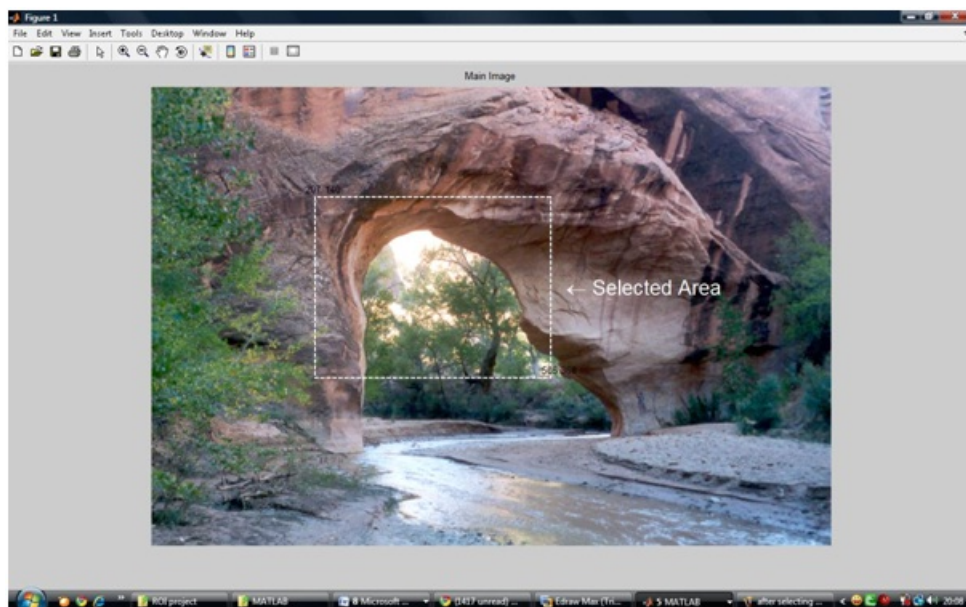


Figure 6.38: Screen after selection of points where storage will take place.

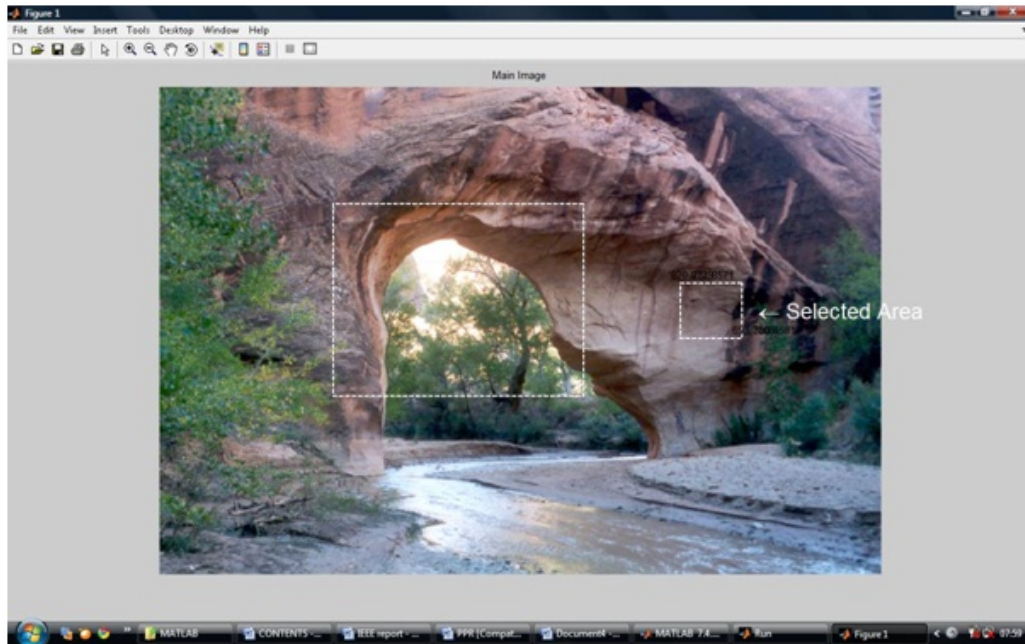


Figure 6.39: Double steganography within the same image.

6.8 StegAlyzer AS 3.2 Evidence Report

Steganalysis software like StegAlyzer AS 3.2 is a popular tool used for forensic investigations across the world. This tool is prepared by SARC and is available as a trial version from [11]. A case information to check that steganalysis software is able to detect the presence of hidden data or not, is performed and a report for the same is shown below. The report provided here, shows different scans and these scans, are the various signatures, which are available under the software to be tested individually. As the StegAlyzerAS 3.2 does not provide the option to test all signature, at just one go, individual signatures needs to be checked. All the signatures available under StegAlyzerAS were tested and a few test results, show that no signature were detected in the watermarked images.

StegAlyzerAS 3.2 Evidence Report: Case# 1

Case Information			
Case ID:	1	Organization Country:	INDIA
Investigator Name:	Samir B. Patel	Organization Name:	Insitute of Technology, Nirma University
Organization Address:	SG Highway, Charrodi	Organization State:	
Organization City:	Ahmedabad	Organization Zip:	382481
Organization Phone:	9427401616		

Case File: D:\Samir Everything\Samir\My Research papers\IJITST\blue\ReportStegAlyzer.ascf

Case Description: This is to check that steganalysis software is able to detect the presence of hidden data or not?

Case Notes:

Report Contents:

Click on a link to view the different sections of this report:

1. [Detected File Artifacts Table](#)
2. [Detected Applications Summary](#)
 - [1. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [2. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [3. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [4. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [5. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [6. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [7. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [8. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [9. D:\Samir Everything\Samir\My Research papers\IJITST](#)
 - [10. D:\Samir Everything\Samir\My Research papers\IJITST](#)
3. [Case Log](#)

File Artifact Table:

Artifact Name	False Positive	Application Name	Location Scanned	Verification Type	Verification Hash	Size	Found Under	Time Found
---------------	----------------	------------------	------------------	-------------------	-------------------	------	-------------	------------

This case does not contain any file artifacts.

[Return to Report Contents](#) [Return to Top of Table](#)

Detected Applications Summary:

- There were no steganography application artifacts detected during the course of this investigation.

[Return to Report Contents](#) [Return to Top of Listing](#)

Case Log [Return to Report Contents](#)

Time Stamp	Event Type	Event Item	Additional Information
13-05-2010 14:29:37	Created Case File	D:\Samir Everything\Samir\My Research papers\IJITST\blue\ReportStegAlyzer.ascf	Samir B. Patel
13-05-2010 14:29:40	Saved Case File	D:\Samir Everything\Samir\My Research papers\IJITST\blue\ReportStegAlyzer.ascf	Samir B. Patel
13-05-2010 14:29:40	Created Case File	D:\Samir Everything\Samir\My Research papers\IJITST\blue\ReportStegAlyzer.ascf	Samir B. Patel
13-05-2010 14:30:39	Scan Started	D:\Samir Everything\Samir\My Research papers\IJITST	Samir B. Patel
13-05-2010 14:30:39	Scan Completed	D:\Samir Everything\Samir\My Research papers\IJITST	Samir B. Patel
13-05-2010 14:30:57	Scan Started	D:\Samir Everything\Samir\My Research papers\IJITST	Samir B. Patel

The list is long which looks similar for all the remaining signatures; the display is truncated here for rest of the cases.

[Return to Report Contents](#) [Return to Top of Case Log](#)