RGB Image Steganography on Multiple Frame Video using LSB Technique

Saket Kumar[1], Ajay Kumar Yadav[2], Ashutosh Gupta[3], Pradeep Kumar[4]

[1]M.Tech Student, VLSI, Mewar University, Ghaziabad, India
er.saket.kum@amity.edu
[2]Department of ECE, Mewar University, Ghaziabad, India
ajay.ajaikr@gmail.com
[3,4]Department of ECE, Amity University Uttar Pradesh, Noida, India
[3]agupta5@amity.edu, [4]pkumar4}@amity.edu

Abstract: **This paper presents the steganography of an Image on a multiple frame video using Frame Decomposition Technique in which LSB Algorithm is used on multiple frame video for Steganography.The aim of the research is to get colored output image so image decomposition and block decomposition of image also required for this type of steganography. It declares that same image which steganography has been done will extract at output phase. The difference between the input image and output image is very less.Password encryption also providedto the video for security purpose. Message hiding technique using LSB algorithm has been used. After applying different attacks on frame, the data or watermarked image has no any effect and it can retrieve as it is. The color map of RGB image is individually watermark on each video frame with encryption of password. Without password it cannot be retrieve and without color map watermark image can be extract. Improved result in Gaussian Noise, Gamma Correction, Rotation and Resizing is gain.**

Keywords: FDT(Frames Decomposition Technique), LSB, Steganography of multiple frame video,

I. INTRODUCTION

Steganography process is the most ancient processspecially used for hiding an image in the second image.First time use of steganography can be traced in 440 BC when Herodotus mentions two examples in his history[1].Demaratussent a warning about a forthcoming attack to Greece by writing it directly on the wooden backing of a wax tablet before applying its beeswax surface[1].Steganography is wide used in today's world because in this technique only sender and recipient can find the message which is hidden in the second image and no one can find the message except them.

In this research the image is hiding on a multiple frame video using Component Division Technique. But before applying this technique we are extracting all the frames from the video and savedit in the current directories. An image has maximum 0-255 pixels. It means only 255 matrix will be generated from the image. The only condition of this Steganography is the video frame should be the minimum 255 frames. If the number of frames of the video is lesser than 255 frames then it cannot be hide in the video. The limitations are for security and exact image extraction. The video frames are divided into blocks and then several steganography processes is perform [2] [3] [4]. The color map of the image has been extracted then only two dimensional matrix of the image isremained and then Component Division technique is applied in the two dimensional image matrixes for matrix extraction.

We are also in-process of implementing this algorithm on reconfigurable system or FPGA as implemented in [7-8].

A. Selection of CD technique for MATRIX Decomposition

Component Division Technique of Matrix is the process in which we can extract different matrix of different values. In this technique all the remaining values will be zero except the values which choose. By the help of this process we will get the all binary matrix with appropriate value. After getting the appropriate matrixes of the image the hiding process of image will be started. In this stage we have mainly the different types of matrixes of the image which will further converted into binary images and saved in the directory. The process of converting the images in CDT matrixes is as follows in the flow chart in figure 1. The flow chart starts from input image and it decompose into two parts, one part is color-map of the image and the second part is 2-D RGB image matrix. The color-map has been saved in a variable named *im_color_map* and the second 2D RGB image matrix saved in a variable named *im_matrix.* All the operations will be done in the *im_matrix.* Before applying CDT, we are saving all the pixel values in a different variable named *im_pixel_values_unique* with unique pixel values from *im_matrix*. This pixel value matrix will help to fetch the CDT matrix in Binary form and again convert it into the CDT matrix. After extraction the *im_pixel_values_unique* matrix from the RGB image matrix, the second process of the CDT will be applied. Now applying a loop on the 2D RGB image matrix and extract all the matrix with single value and remaining values will be zero in the entire matrix. After getting the CDT matrix it will be converted in to Binary images and saved in the directory in the image format. These matrix images in completely binary images so we don't required converting our image in to gray scale and binary format.

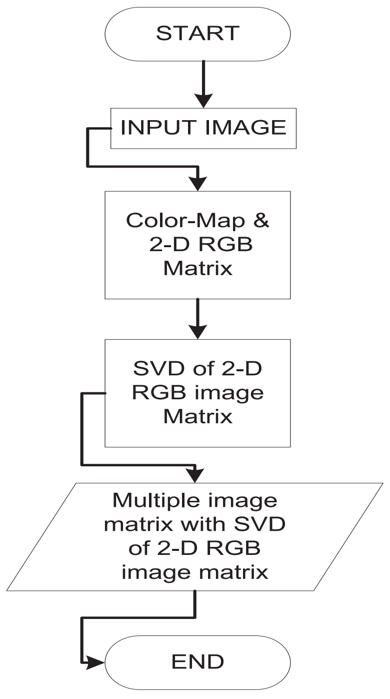
START
INPUT IMAGE
Color-Map & 2-D RGB Matrix
SVD of 2-D RGB image Matrix
Multiple image matrix with SVD of 2-D RGB image matrix
END

Figure. 1. Flow Chart of CDT Technique

B. Extract Frames from Input Video

After applying CDT technique on image we have all the matrixes from the image. Now the second work is to extract frames from the video. The extraction process is as discussed in flow chart figure 2. It is very important to split frames from video because each frames will hide a binary image of image matrix. The flow chart starts from the input video. After insertion of a video, total number of frames will be counted by calling a function *get()*.Get() function has a syntax by which it gives the detail of total number of frames available in the video. Now the challenge is if the total number of frames is less than or equal to the total number of image matrix which get by the image to be hide, then steganography process cannot be done. After the getting number of frames it will store in the variable named *numberOfFrames*. The next step is to extract all frames present in the video. By applying a *for* loop and extract the number of frames from video. The *for* loop will be start from 1 and end to *numberOfFrame.* Inside the *for* loop, extract *cdata* of the video by calling function *vidFrames ().*After getting the video frames *cdata* we are using *imwrite ()* function to write the frames in the image format in our current directory. The process is explained in detail in the algorithm part. In flow chart only steps are explained.

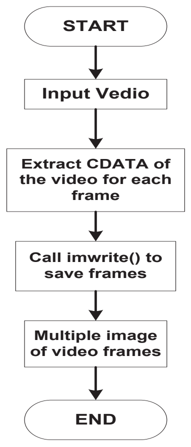
START
Input Vedio
Extract CDATA of the video for each frame
Call imwrite() to save frames
Multiple image of video frames
END

Figure. 2. Flow Chart of Splitting frames from Video.

C. Split Frames in to different Blocks of Image

The third step of this Steganography process is to divide the frames of the video into two blocks. Because in first block the color-map and the unique matrix of the image will be hide in the form of text. This process is essential because all the frame of the video will be dividing in to two parts and the two matrixes color-map and unique matrix will be hiding inside the blocks of the image. The binary image matrix extracted from the image will be hiding in the second block of the frame which got by dividing the frame into blocks. The process of the hiding all the data in image is shown in figure3. The color-map of the image and unique matrix of the image will hide in all the frame of the video because the measure challenge is if the frames of the video has been deleted by anyone and that frame has the information of color-map and unique matrix then the image cannot be recover from the video. Without these two data no one can recover the image from the video frames. For security of image and easy to recover the image, we have to hide these two matrix in all the frames of the video. And for this reason we are dividing all the frames into two blocks. The first block is smaller than the second block because the two matrixes will hide in the smaller block but the image frame needs bigger frame image for hiding. The second block is 2/3 part of the frame image. The flow chart of dividing the block in image is shown in figure 4. The algorithm is to be discussed in the algorithm stage.

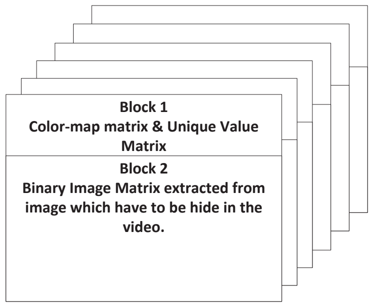
Block 1
Color-map matrix & Unique Value Matrix
Block 2
Binary Image Matrix extracted from image which have to be hide in the video.

Figure. 3. Divide frame of the image into blocks.

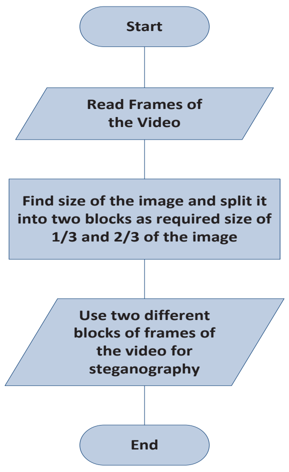
Start
Read Frames of the Video
Find size of the image and split it into two blocks as required size of 1/3 and 2/3 of the image
Use two different blocks of frames of the video for steganography
End

Figure. 4. Flow chart of splitting frames in two blocks.

D. Issues and challenges

In this type of Steganography the main limitation is as follows:

1. The number of frames of the video has been greater than 255 because in a color image matrix there are maximum 0-255 pixel values and if we are applying CDT on the image matrix then it will be split into maximum 255 frames of the image.

2. The second challenge is we have to save all the values of the matrix in a second matrix then only we can retrieve all the frame of the image with exact value. All the frames are to be converted into binary frames of the image.

II. ALGORITHM TO BE USED

A. CDT technique Algorithm

Here we will discuss briefly the above process in algorithmic form.

```
Step1: Start
Step2: Initialize the process to insert image.
Step3: Extract color-map and image matrix by [im_matrix, im_color_map] = rgb2ind (imread('12.jpg'), 128);
Step4: Save
im_pixel_values_unique = unique (im_pixel_value);
Step5: Call size() function to calculate the size of the im_matrix in [im_row, im_col] = size(im_matrix)
Step6: k =1:im_pixel_values_unique
    i = im_row
    j = im_col
    ifim_matrix(i, j) = k
    Then im_matrix(i, j) = k
    else
        im_matrix(i, j) = 0
Step7: Call logical () function to convert im_matrix to logical values.
Step8: Save all the matrix in the form of image by calling imwrite() function.
Step9: Jump to step 3.
```

Algorithm of Extration of Video frames from the image

Here we are discussing about the extraction process of frames of the image.

```
Step1: Start
Step2: Initialize the process to insert Video.
Step3: Call the VideoReader () function to read object of the Video and save it in variable named vidObj.
Step4: By calling get () function to get the information about total number of frames in the video.
Step5: Save the number of frames in a variable named numberOfFrames.
Step6: k =1:numberOfFrames
    Find cdata of the video.
    Save this cdata of the video with different name of image in *.png or *.jpg format.
```

Algorithm of dividing video frame images in two blocks with proper ratio

Here we are discussing the algorithm to divide image into two blocks with proper ratio.

```
Step1: Read the Video frame image by calling imread () function.
Step2: n = fix(size(im, 1)/3)
Step3: By calling matrix with this size, got the first block of image and save this block for steganography.
Step4: By calling matrix with this size, got the second block of image and save this block for steganography.
```

III. RESULTS & DISCUSSION

We choose different algorithm to complete this objective of Steganography using CDT Techniques. After utilizing different code we assemble all the parts of the algorithm and developed a flow chart for complete system then get the appropriate result of Steganography.The flow chart of complete system is shown in figure5.

This Flow chart shows complete process of encoding the image in the video by CDT technique. After hiding the image on the video the difference between the original video and final output video is zero. The difference calculates frame by frame. We are using LSB Technique for steganography. Below example shows the steganography process of the image on Video by CDT process.

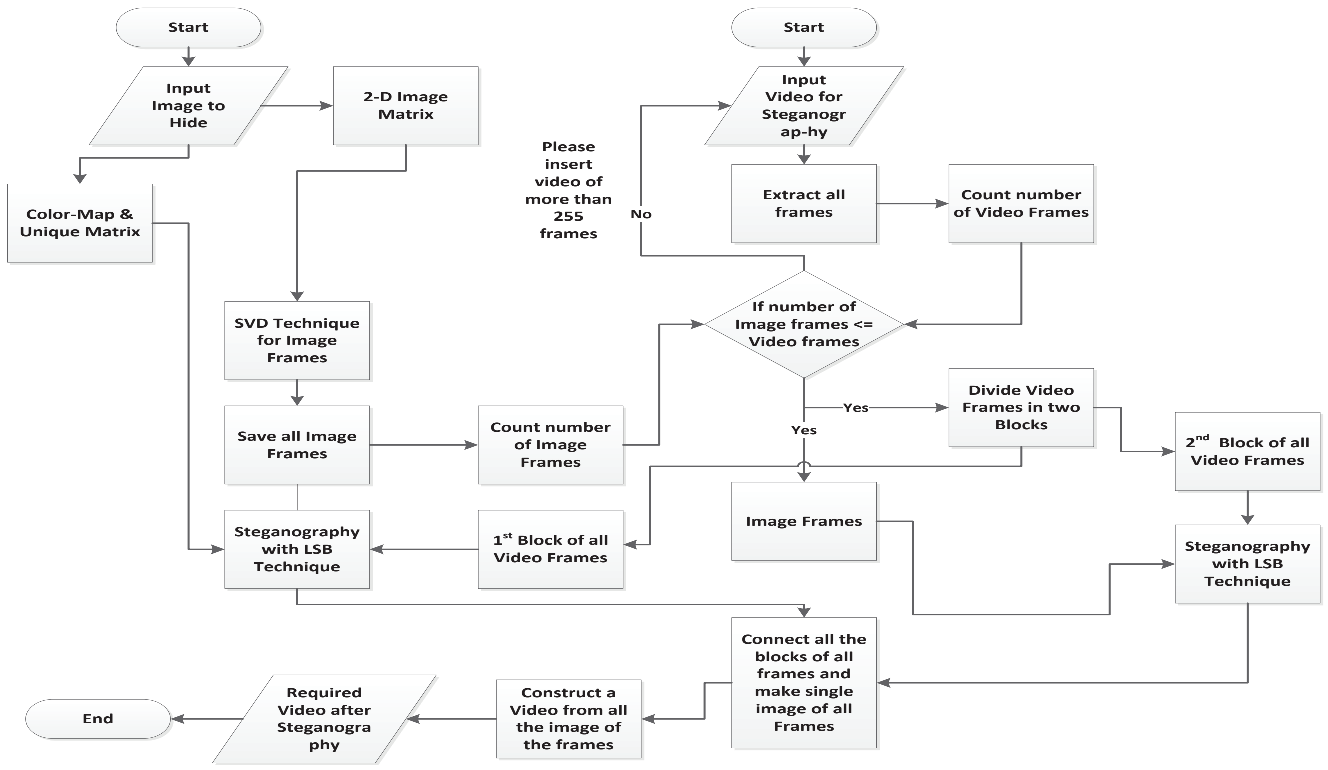
Start
Input Image to Hide
2-D Image Matrix
Color-Map & Unique Matrix
SVD Technique for Image Frames
Save all Image Frames
Count number of Image Frames
Steganography with LSB Technique
1st Block of all Video Frames
Start
Input Video for Steganogr ap-hy
Extract all frames
Count number of Video Frames
Please insert video of more than 255 frames
No
If number of Image frames <= Video frames
Yes
Yes
Divide Video Frames in two Blocks
2nd Block of all Video Frames
Image Frames
Steganography with LSB Technique
Connect all the blocks of all frames and make single image of all Frames
Construct a Video from all the image of the frames
Required Video after Steganogra phy
End

Figure. 5. Steganography Process of Image on Video by CDT Technique.

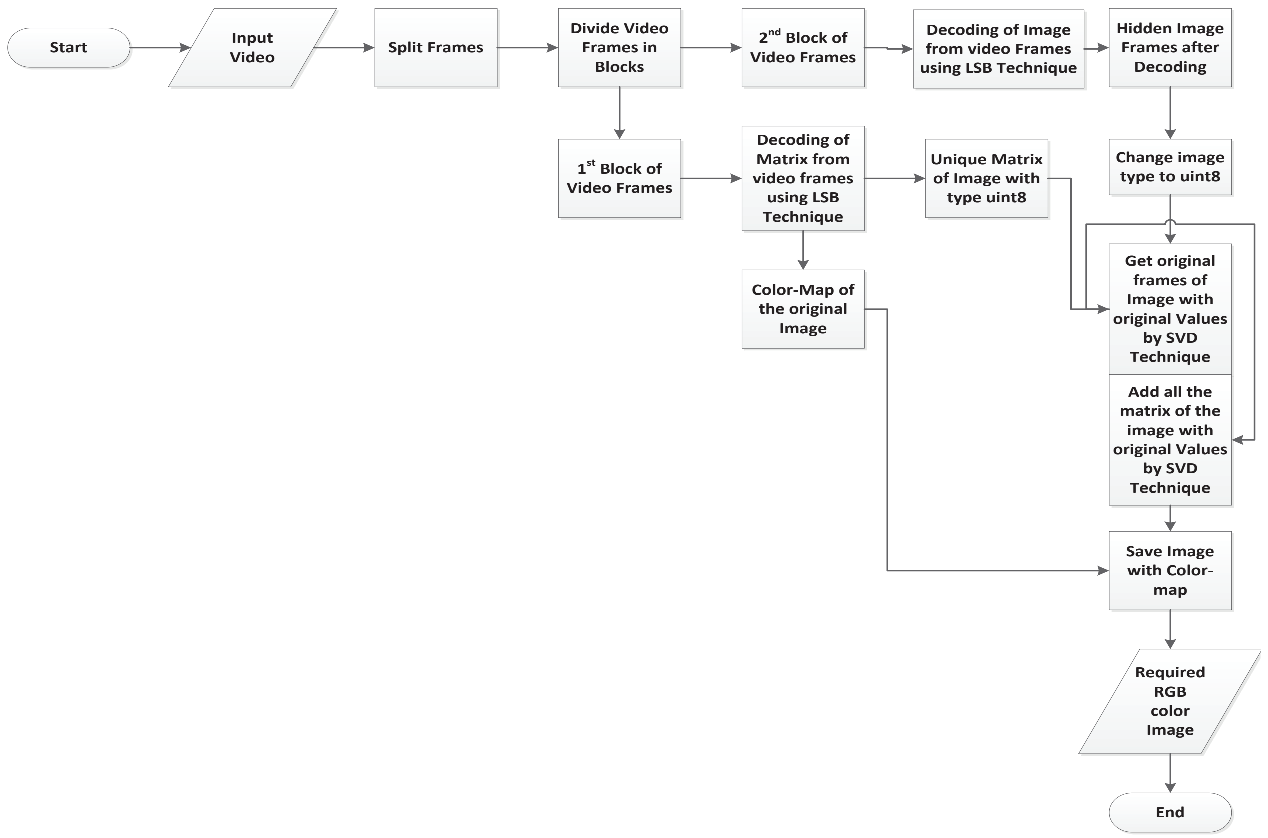
Start
Input Video
Split Frames
Divide Video Frames in Blocks
2nd Block of Video Frames
Decoding of Image from video Frames using LSB Technique
Hidden Image Frames after Decoding
1st Block of Video Frames
Decoding of Matrix from video frames using LSB Technique
Unique Matrix of Image with type uint8
Change image type to uint8
Color-Map of the original Image
Get original frames of Image with original Values by SVD Technique
Add all the matrix of the image with original Values by SVD Technique
Save Image with Color-map
Required RGB color Image
End

Figure. 6. Decoding process of Hidden Image Using CDT Technique

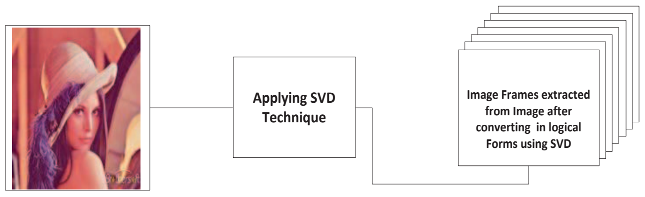
Applying SVD Technique
Image Frames extracted from Image after converting in logical Forms using SVD

Figure. 7. Image Frames extracted from the image after applying CDT Technique.

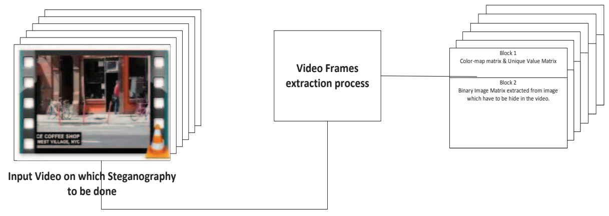
Video Frames extraction process
Block 1
Color-map matrix & Unique Value Matrix
Block 2
Binary Image Matrix extracted from image which have to be hide in the video.
Input Video on which Steganography to be done

Figure. 8. Frames extraction from video

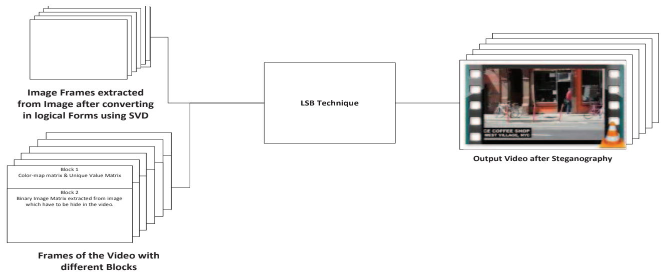
Image Frames extracted from Image after converting in logical Forms using SVD
LSB Technique
Output Video after Steganography
Block 1
Color-map matrix & Unique Value Matrix
Block 2
Binary Image Matrix extracted from image which have to be hide in the video.
Frames of the Video with different Blocks

Figure. 9. Steganography of Image on Video.

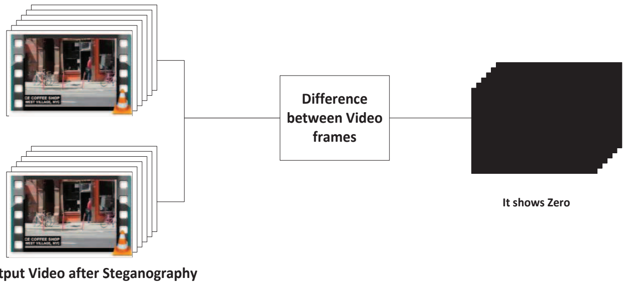
Difference between Video frames
It shows Zero
Output Video after Steganography

Fig. 10. Difference between Video Frames after Steganography and Before Steganography.

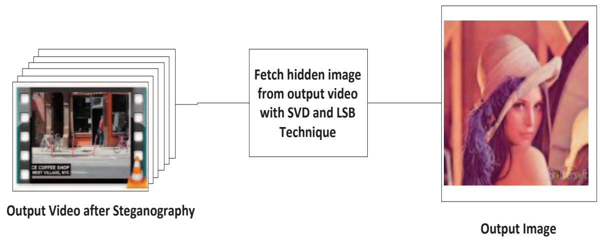
Fetch hidden image from output video with SVD and LSB Technique
Output Video after Steganography
Output Image

Figure.10. Fetch hidden image from video using SVD & LSB Technique

PSNR: The Peak-Signal-To-Noise ratio (PSNR) is using for deviation of the watermarked and attacked frames from the original video frames and it is defined as:

$$PSNR = 10Log_{10}\left(\frac{255^2}{MSE}\right)$$

Where, MSE (Mean Squared Error) is between the original and distorted frames and is defined as:

$$MSE = \left(\frac{1}{mn}\right)\sum_{i=1}^{m}\sum_{j=1}^{n}[I(i,j) - I^{'}(i,j)]$$

In this equation, I and I' represent the pixel values of the original frame and distorted frame, respectively. If it gives high values for PSNR then it indicates more perceptibility of watermarking, which is expressed in decibels (dB).

NC: The normalized coefficient (NC) gives the measurement of the robustness of watermarking and its peak value is 1. [6]

$$NC = \frac{\sum_i \sum_j W(i,j).W'(i,j)}{\sqrt{\sum_i \sum_j W(i,j)}\sqrt{\sum_i \sum_j W'(i,j)}}$$

In this equation W and W' are representing the original message image and watermarked image respectively. After watermarking the video, the following images are taken with various attacks performed on them.Following table shows the improved PSNR and NC values collected from the watermarked video after performing various attacks on the frame as compared to the results obtained in [5].

TABLE I: PSNR & NC for Recovered Image

Attacks	PSNR	NC
GAUSSIAN NOISE	35.2876	0.8251
SALT & PEPPER NOISE	28.4891	0.7314
ROTATION	36.8564	0.8136
RESIZING	42.3546	0.7121
GAMMA CORRECTION	32.4563	0.6635
CONTRAST ADJUSTMENT	39.2135	0.61235
SHARPENING FILTER	43.2535	0.6935

IV. Conclusion

After Steganography of image on Video we got the exact video which we input and there is no any difference between the input video and output video. The process of CDT and LSB are really useful but there are some limitations in the process. The major limitation is the number of frames should be greater than 255 because in an image there are maximum 255 pixels and after applying CDT we will get maximum of 255 matrix of image. It means it will give maximum 255 frames of a single image. For steganography process of an image on video by CDT Technique, it requires the condition number of video frames>=number of Image frames. After Decoding the image we can get the exact RGB image due to help of color-map and unique matrix. The difference of the output RGB image and Input RGB image will be zero.

REFERENCES

[1] Petitcolas, FAP, Anderson RJ; Kuhn MG (1999)."Information Hiding: A survey" Proceedings of the IEEE (special issue)87 (7): 1062–78. doi:10.1109/5.771065. Retrieved 2008-09-02.

[2] Bamatraf, A; Ibrahim, R.; Salleh, M.N.B.M., "Digital watermarking algorithm using LSB, " Computer Applications and Industrial Electronics (ICCAIE), 2010 International Conference on, vol., no., pp.155, 159, 5-8 Dec. 2010, doi: 10.1109/ICCAIE.2010.5735066

[3] Tao Chen; Hongtao Lu, "Robust spatial LSB watermarking of color images against JPEG compression, " Advanced Computational Intelligence (ICACI), 2012 IEEE Fifth International Conference on, vol., no., pp.872, 875, 18-20 Oct. 2012, doi: 10.1109/ICACI.2012.6463294

[4] Dehkordi, AB.; Esfahani, S.N.; Avanaki, AN., "Robust LSB watermarking optimized for local structural similarity, " Electrical Engineering (ICEE), 2011 19th Iranian Conference on, vol., no., pp.1, 1, 17-19 May 2011

[5] Kumar, Saket; Gupta, Ashutosh; Chandwani, Ankur; Yadav, Gaurav; Swarnkar, Rashmi, "RGB image watermarking on video frames using DWT, " Confluence The Next Generation Information Technology Summit (Confluence), 2014 5th International Conference -, vol., no., pp.675, 680, 25-26 Sept. 2014

[6] Ankur Choudhary, SPS Chauhan, M Afshar Alam, Safdar Tanveer, “Schur decomposition and dither modulation: an efficient and robust audio watermarking technique”, Conference Proceedings, Proceedings of the CUBE International Information Technology Conference, 744-748, ISSN:1450311857, ACM2012.

[7] Solomon Raju Kota, Ashutosh Gupta, Shashikant Nayak, and Sreekanth Varma.” Module Based Implementation of Partial Reconfiguration Using VHDL on Xilinx FPGA” International Journal of Recent Trends in Engineering, vol 2, No.7, November 2009.

[8] Ashutosh Gupta, Kota Solomon Raju “Design and implementation of 32-bit controller for interactive interfacing with reconfigurable computing systems” International Journal of Computer science & Information Technology (IJCSIT), Vol 1, No 2, November 2009.