International Journal of Computer Applications (0975 – 8887)
Volume 97– No.17, July 2014

Hybrid Approach for Video Compression using Ant Colony Optimization and Modified Fast Haar Wavelet Transform

Abhay Suri
M.Tech Student
Deptt. Of ECE, RIEIT,
Railmajra, Ropar, Punjab,
India

Anudeep Goraya
Associate Professor
Deptt. Of ECE, RIEIT,
Railmajra, Ropar, Punjab,
India

ABSTRACT

In order to fulfill the requirement of growing demand of video on internet like its streaming, digital library and also due to limited channel bandwidth, video compression has become a necessity and in order to compress a video we have to reduce its temporal and spatial redundancy. Temporal redundancy deals with motion estimation and compensation. One of the popular methods for motion estimation is intensity based block matching which determines the movement of blocks between the adjacent frames, but a common drawback with this block motion estimation is the velocity of the blocks located at the boundary of the moving objects is not estimated accurately.

In this paper, a hybrid approach for video compression is presented in which the motion estimation using edge matching is presented. The Ant colony Edge Detector is used to create edges. The image is divided into non overlapping rectangular blocks. The best match to current block is search for in the previous frame to the search area and on the basis of mutual information match is found. In order to remove the spatial redundancy, the Modified Fast Haar Wavelet Transformation is used.

Keywords

Motion Estimation, Edge Detection, Video compression, Mutual Information.

1. INTRODUCTION

With rapid advances in multimedia communication and due to those applications for digital video like Video telephony, streaming media delivery on internet, CD, DVD Cellular media, educational purpose etc. requires large storage space. Video Compression is necessary to eliminate picture redundancy, allowing video information to be transmitted and stored in a compact and efficient manner [1].

Why can video be compressed? The reason is that video contains much spatial and temporal redundancy. In a single frame, nearby pixels are often correlated with each other. This is called spatial redundancy, or the intraframe correlation. Another one is temporal redundancy, which means adjacent frames are highly correlated, or called the interframe correlation. Therefore, our goal is to efficiently reduce spatial and temporal redundancy to achieve video compression. Temporal redundancy deals with motion estimation and compensation. One of the popular methods for motion estimation is intensity based block matching which determines the movement of blocks between the adjacent frames, but a common drawback with this block motion estimation is the velocity of the blocks located at the boundary of the moving objects is not estimated accurately.

So, the motion estimation using edge matching is presented. The Ant colony Edge Detector is used to create edges. The image is divided into non overlapping rectangular blocks. The best match to current block is search for in the previous frame to the search area and on the basis of mutual information match is found [4].

In order to reduce the high computational load of exhaustive search algorithm several fast search techniques such as Four-Step search, Three-Step search and New Three-Step search have been proposed as a matching criterion for motion estimation.

1.1 Motion Estimation

At first [13], we divide current frame into non-overlapping 16x16 macro blocks. For each macro block, find the best matching block in a reference frame. That is to say, motion estimation of a macro block involves finding 16 x 16 blocks in the search area in a reference frame that closely matches the current macro block. The reference frame is a previously-encoded frame from the sequence and may be before or after the current frame in display order. The search area in the reference frame is centred on the current macro block position.

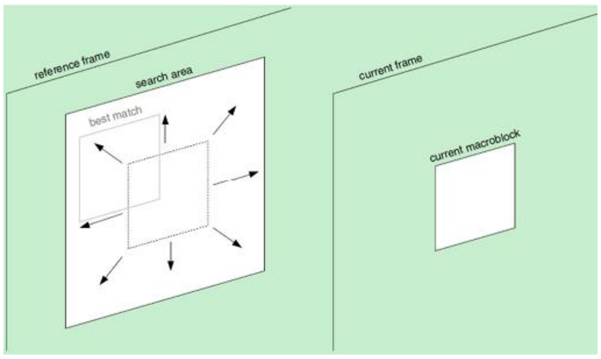

Figure 1: Motion estimation [13]

1.2 Motion Compensation

When the best matching block is found in the reference frame by motion estimation, we subtract the best matching block from the current macro block to produce a residual macro block .Within the encoder, the residual is encoded and decoded and added to the matching region to form a reconstructed macro block which is stored as a reference for further motion compensated prediction [13].

1.3 The Block Matching Algorithms Used for Motion Estimation for Video Compression

1.3.1 Exhaustive Search (ES)

This algorithm, also known as Full Search, is the most computationally expensive block matching algorithm of all. This algorithm calculates the cost function at each possible location in the search window. As a result of which it finds the best possible match and gives the highest PSNR amongst any block matching algorithm. Fast block matching algorithms try to achieve the same PSNR doing as little computation as possible. The obvious disadvantage to ES is that the larger the search window gets the more computations it requires.[13]

1.3.2 Three Step Search (TSS)

This is one of the earliest attempts at fast block matching algorithms and dates back to mid 1980s. It starts with the search location at the centre and sets the 'step size' S =4, for a usual search parameter value of 7. It then searches at eight locations S pixels around location (0, 0). From these nine locations searched so far it picks the one giving least cost and makes it the new search origin. It then sets the new step size S = S/2, and repeats similar search for two more iterations until S = 1. At that point it finds the location with the least cost function and the macro block at that location is the best match. The calculated motion vector is then saved for transmission. It gives a flat reduction in computation by a factor of 9. So that for p = 7, ES will compute cost for 225 macro blocks whereas TSS computes cost for 25 macro blocks [10].

1.3.3 New Three Step Search (NTSS)

NTSS is a modified version of the three step search algorithm for searching small motion video sequences. For these video sequences, the motion vector distribution is highly centre biased and having provisions for half way stop to reduce computational cost. It was one of the first widely accepted fast algorithms and frequently used for implementing earlier standards like MPEG 1 and H.261. The TSS uses a uniformly allocated checking pattern for motion detection and is prone to missing small motions. The NTSS process is illustrated graphically. In the first step 16 points are checked in addition to the search origin for lowest weight using a cost function.

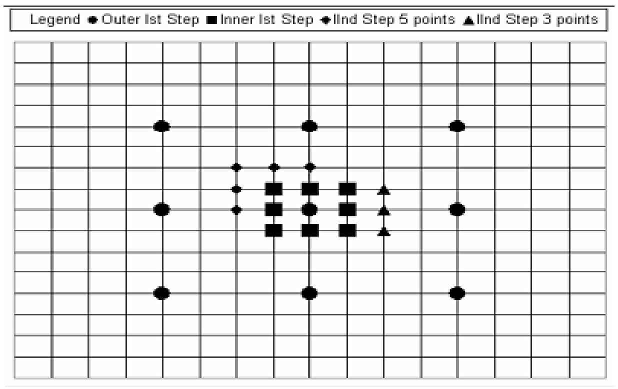
Legend ● Outer Ist Step ■ Inner Ist Step ◆IInd Step 5 points ▲IInd Step 3 points

Figure 2: New Three Step Search (NTSS)[12]

Of these additional search locations, 8 are a distance of S=4 away (similar to TSS) and the other 8 are at S=1 a way from the search origin. If the lowest cost is at the origin then the search is stopped right here and the motion vector is set as (0, 0). If the lowest weight is at any one of the 8 locations at S = 1, then we change the origin of the search to that point and check for weights adjacent to it. Depending on which point it is we might end up checking 5 points or 3 points. The location that gives the lowest weight is the closest match and motion vector is set to that location. On the other hand if the lowest weight after the first step was one of the 8 locations at S = 4, then we follow the normal TSS procedure. Hence although this process might need a minimum of 17 points to check every macro block, it also has the worst-case scenario of (17+8+8) = 33 locations to check [12].

1.3.4 Four Step Search (4SS)

Similar to NTSS, [11] 4SS this algorithm also exploits the centre biased characteristics of the real world video sequences by using a smaller initial step size compared with TSS and has a halfway stop provision. 4SS sets a fixed pattern size of S = 2 for the first step, no matter what the search parameter p value is. Thus it looks at 9 locations in a 5x5 window. If the least weight is found at the centre of search window the search jumps to fourth step. If the least weight is at one of the eight locations except the centre, then we make it the search origin and move to the second step. The search window is still maintained as 5x5 pixels wide. Depending on where the least weight location was, we might end up checking weights at 3 locations or 5 locations. The patterns are shown in. Once again if the least weight location is at the centre of the 5x5 search window we jump to fourth step or else we move on to third step. The third is exactly the same as the second step. IN the fourth step the window size is dropped to 3x3, i.e. S = 1. For the lower left path, it requires (9+5+3+8) = 25 checking points. For the upper right path, it requires (9+5+5+8)=27 checking points that is the worse case of the algorithm for d=7. Figure 6 shows two search paths of FSS for searching small motion. For the left path, it requires (9+8) =17 checking points. For the right path, it requires (9+3+8) =20 checking points. The location with the least weight is the best matching block and the motion vector is set to point that location.

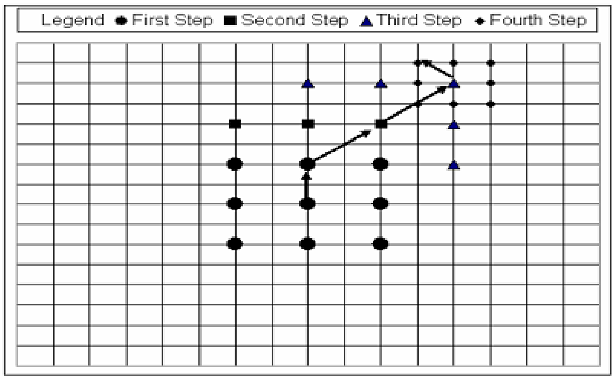
Legend ● First Step ■ Second Step ▲ Third Step ◆ Fourth Step

Figure 3: Four Step Search (4SS) [11]

1.4 Ant Colony Edge Detector

Ant colony optimization follows the criterion to improve the edge detection by the process whose aim is to identify points in an image where discontinuities or sharp changes in intensity occur. This process is crucial to understanding the content of an image and has its applications in image analysis and machine vision. It is usually applied in initial stages of computer vision applications. Edge detection aims to localize the boundaries of objects in an image and is a basis for many image analysis and machine vision applications [7]. Conventional approaches to edge detection are computationally expensive because each set of operations is conducted for each pixel. In conventional approaches, the computation time quickly increases with the size of the image. Thus, the Ant colony edge detector first smoothes the image to eliminate noise and then it finds image gradients by highlighting areas with spatial derivatives. The algorithm then tracks along these areas and suppresses pixels that are not at maximum. Hysteresis is then used to further reduce the gradient array [4].

1.5 Modified Fast Haar Wavelet Transformation

Since the Haar Transform is memory efficient, exactly without the edge effects, it is fast and simple. As such the Haar Transform technique is widely in wavelet analysis. Fast Haar Transform is one of the algorithms which can reduce the tedious the work of calculations. One of the earliest versions of FHT is included in HT .FHT involves addition, subtraction and division by 2. Its application in atmospheric turbulence analysis, image analysis, and image compression [8].

In MFHWT, first average sub signal at one level for a signal of length N is

$$a_m = \frac{f_{4m-3} + f_{4m-2} + f_{4m-1} + f_{4m}}{4}$$

For m = 1, 2, 3…………….N/4

And the first detail sub signal d_m at the same level is given as:

$$d_m = \left\{\begin{cases} \frac{(f_{4m-3}+f_{4m-2})-(f_{4m-1}+f_{4m})}{4}, m = 1,2 \ldots. N/4 \\ 0 \quad , \quad m = \frac{N}{2}, \ldots\ldots..N \end{cases}\right\}$$

Here four nodes are considered at a time instead of two nodes as in HT and FHT. The author has considered the values of *N/2* detail coefficients zero in each step than to find the *N/2* detail coefficients by FHT.

The MFHWT is faster in comparison to FHT and reduces the calculation work. In MFHWT, we get the values of approximation and detail coefficients one level ahead than the FHT and HT .In MFHWT we need to store only half of the original data used in FHT, due to which it becomes much more memory efficient. The number of non-zero coefficients is lesser in MFHWT than that in the other two transforms and it also preserves the energy of the original input image as in HT and FHT.

2. PROPOSED APPROACH

We have developed an approach using the Ant colony Edge detector which will create the edges.

The image is divided into non overlapping rectangular blocks. The best match to the current block is searched for in the previous of frame of the video within a search area for the location of the current block. Mutual information (MI) is used as the matching criterion. Once the motion Vectors are obtained, the MFHWT is used to obtain the compensation in the video sequence.

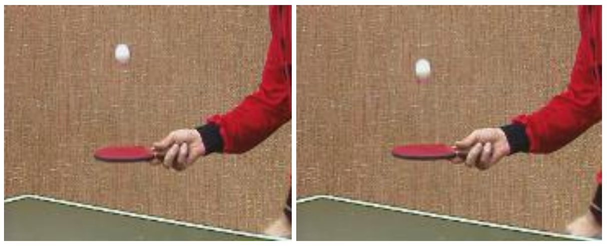

Figure 5: Frame number 1 and 2 of the Tennis Player Sequence[14,1]

Figure 6: Frame number 7 and 19 of the Suzie sequence [14, 1]

Hence the proposed edge-based motion estimation technique improves the motion estimation accuracy in terms of the peak signal-to-noise ratios of reconstructed frames.

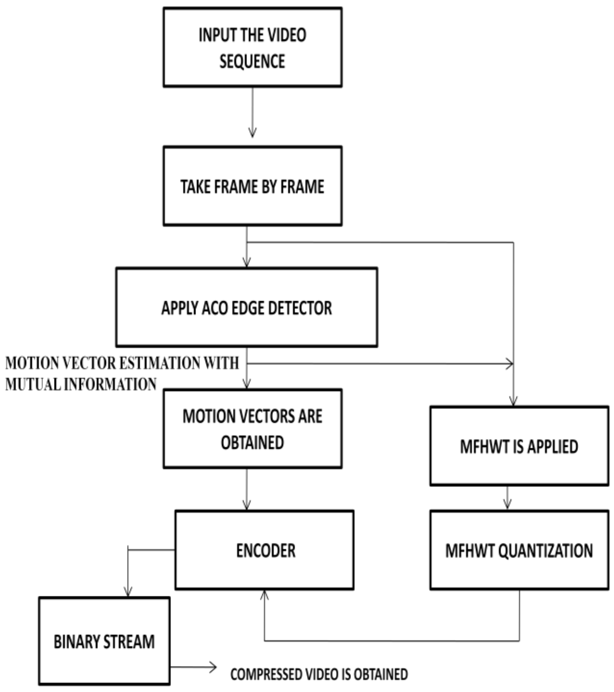

Figure 4: Flow Diagram of the proposed algorithm

3. SIMULATION RESULTS

The performance of the proposed approach has been checked on “Tennis Player” video sequence, and “Suzie” video sequence. Mean Squared Error (MSE) a criterion is used for finding out the best matching block.

Here the basic parameters of the proposed approach are presented in terms of PSNR and MSE. The approach is implemented with MATLAB.

$$\text{MSE} = \frac{1}{N^2}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\left(C_{ij} - R_{ij}\right)^2$$

Where N x N is the size of the macro block, C_{ij} and R_{ij} are the pixels being compared in current macro block and reference macro block, respectively.

PSNR of the motion compensated images that are created by using motion vectors and macro blocks from the reference frame given by equation

$$\text{PSNR} = 10\log_{10} 255^2 / MSE$$

The average PSNR values for different search algorithms and different video sequences are stated as follows:

Table 1: Average PSNR Performance

S.No.	Video sequence	NTSS	TSS	4SS	Proposed
1	Tennis Player	28.262	27.386	28.402	30.4383
2	Suzie	30.435	26.759	29.144	34.5746

The average MSE values for different search algorithms and different video sequences are stated as follows:

Table 2: Average MSE Performance

S.No.	Video sequence	NTSS	TSS	4SS	Proposed
1.	Tennis Player	97.048	118.750	93.958	58.8181
2.	Suzie	58.827	137.181	79.192	22.6997

Since different block matching algorithms are used, their image qualities are not identical. Peak signal-to-noise ratio (PSNR) and Mean Square Error (MSE) is used as an indicator for quality comparison. The Different experimentations, for the cited above conventional algorithms, have been done and are presented in table 1 show PSNR performance for "Tennis Player" and "Suzie and table 2 gives the MSE for same sequences for 30 frames.

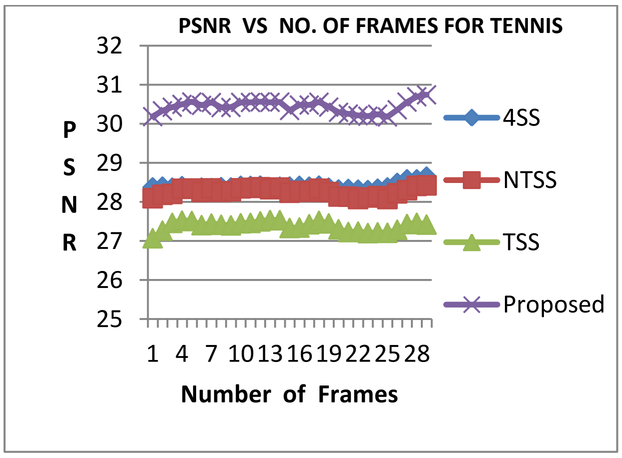

Figure 7: PSNR Performance for Tennis Sequence

Figure 7 shows the comparison of PSNR between the several search algorithms such as Four step search. Three step search, New Three step search and the proposed approach with respect to number of frames (30) for the Tennis player sequence.

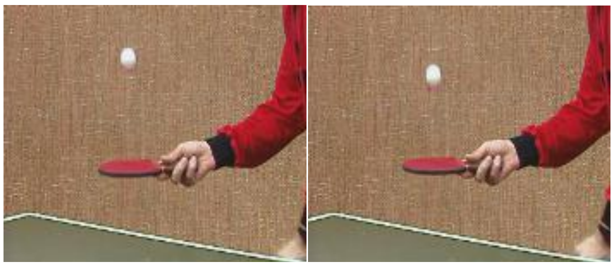

Figure 8: Uncompressed frames of Tennis sequence

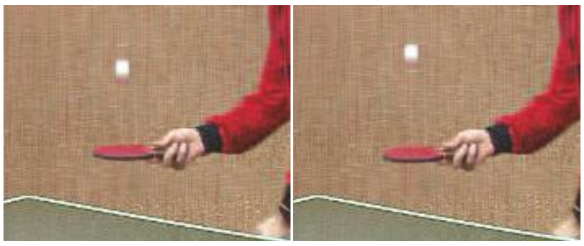

Figure 9: Compressed frames of Tennis sequence (4SS)

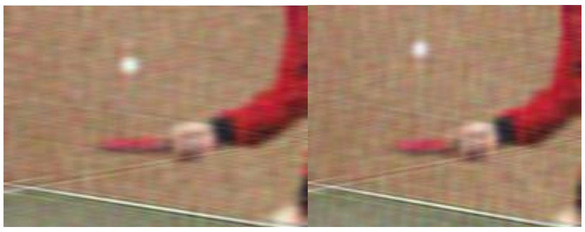

Figure 10: Compressed frames of Tennis sequence (NTSS)

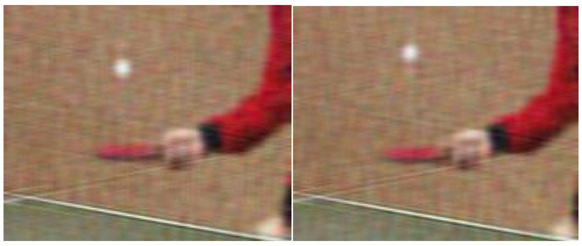

Figure 11: Compressed frames of Tennis sequence (Proposed)

Figure 8 shows the uncompressed frames of the sequence. Figure 9, 10 and11 shows the compressed frames using 4SS, NTSS and Proposed approach respectively.

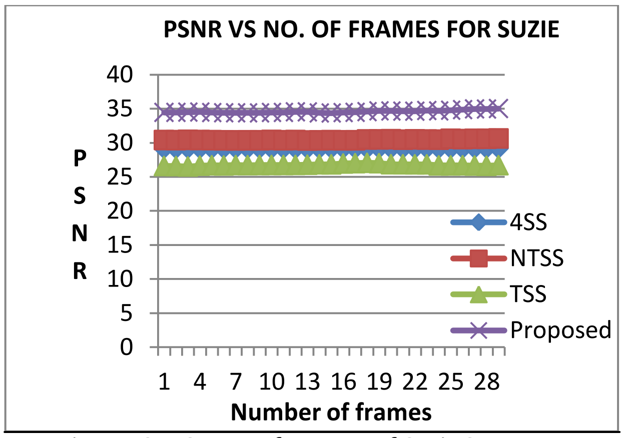

Figure 12: PSNR Performance of Suzie Sequence

Figure 12 shows the comparison of PSNR between the several search algorithms such as Four step search, Three step search, New Three step search and the proposed approach with respect to number of frames (30) for the Suzie sequence.

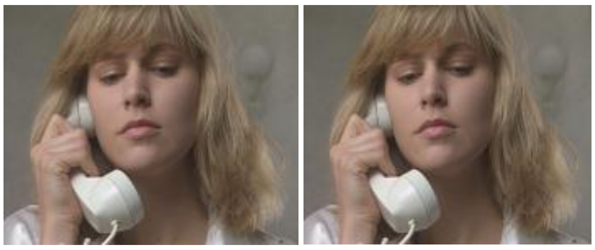

Figure 13: Uncompressed frames of Suzie sequence

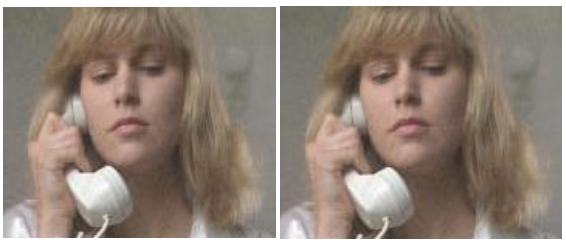
Figure 14: Compressed frames of Suzie sequence (4SS)

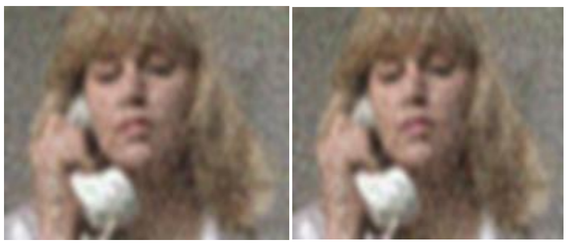
Figure 15: Compressed frames of Suzie sequence (NTSS)

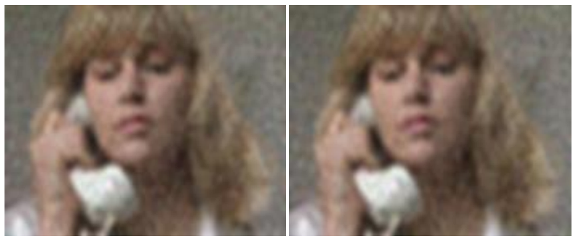
Figure 16: Compressed frames of Suzie sequence (Proposed)

Figure 13 shows the uncompressed frames of the sequence. Figure 14, 15 and16 shows the compressed frames using 4SS, NTSS and Proposed approach respectively.

Hence from the above discussion, we analyze that the proposed algorithm is better in terms of the PSNR and the most important thing is the quality of the video is not much degraded hence better compression is achieved in the environment of limited bandwidth.

4. CONCLUSION

The Motion estimation is a process which determines the motion between two or more frames of video sequence. Block matching algorithms are the most popular and efficient and fast of the various motion estimation techniques. Three block matching motion estimation algorithms, namely Three Step Search (TSS), New Three Step Search (NTSS), and Four Step Search (4SS) algorithms are compared and implemented. The experimental results show that the proposed edge-based motion estimation algorithm achieves a better motion estimation accuracy and the most important thing is the quality of the video is not much degraded hence better compression is achieved in the environment of limited bandwidth than the standard methods used.

5. REFERENCES

[1] M. Marzougui, A. Zoghlami, M. Atri and R. Tourki. Preliminary Study of Block Matching Algorithms for Wavelet-based t+2D Video Coding, IEEE 2013.

[2] Rohit Verma and Mohamed-Yahia Dabbagh, “Binary pattern based edge detection for motion estimation in h.264/avc”, IEEE 2013.

[3] Miok Kim, Nam Ling, John D. Ralston “A Mesh-based Method for Wavelet Video Coding using Edge-Detection in Low Frequency Subband”, IEEE 2013.

[4] Anthony Amankwah Chris Aldrich, “Motion estimation in flotation froth images based on edge detection and mutual information”, IEEE 2012.

[5] Josselin Gautier, Olivier Le Meur “Efficient Depth Map Compression based on Lossless Edge Coding and Diffusion”, IEEE 2012.

[6] Xie Liyin, Su Xiuqin, Zhang Shun, “A Review of Motion Estimation Algorithms for Video Compression”, International Conference on Computer Application and System Modeling ICCASM 2010, IEEE 2010.

[7] Anna Veronica Baterina and Carlos Oppus” Image Edge Detection Using Ant Colony Optimization, “International journal of circuits, systems and signal processing”, issue 2, vol. 4, 2010.

[8] Anuj Bhardwaj and Rashid Ali” Image Compression Using Modified Fast Haar Wavelet Transform”, World Applied Sciences Journal, 2009.

[9] Nada M. A. Al Salami, “Ant Colony Optimization Algorithm” UbiCC Journal, Volume 4, Number 3, August 2009.

[10] Aroh Barjatya, “Block Matching Algorithms For Motion Estimation”, DIP 6620 Spring, 2004.

[11] Lai-Man Po, and Wing-Chung Ma, “A Novel Four-Step Search Algorithm for Fast Block Motion Estimation”, IEEE Trans. Circuits and Systems for Video Technology, vol. 6, no. 3, pp: 313-317, June, 1996.

[12] Renxiang Li, Bing Zeng, and Ming L.Liou, “A New Three-Step Search Algorithm for Block Motion Estimation”, IEEE Trans. Circuits and Systems for Video Technology, vol. 4, no. 4, pp: 438-442, August 1994.

[13] A video compression tutorial by Hsin-Hui Chen Graduate Institute of Communication Engineering National Taiwan University, Taipei, Taiwan, ROC.

[14] www.stackoverflow.com/questions/../standard-test-videos-for-video-processing.