SAN DIEGO STATE UNIVERSITY

Minds that move the world

Motion Estimation for Video Coding

TABLE OF CONTENTS

LIST OF FIGURES

1.0 Introduction

Motion Picture Experts Group (MPEG) is a group formed in 1988. They are part of the International Organization of Standardization (ISO) that is responsible for developing standards. MPEG Group is responsible for developing an international compression standard for audio and video compression and transmission. There have been many versions of the MPEG standards with the latest version of the standards being the H.264 Compression Standards. The inception of these standards marked a leap in the Telecommunications Industry as other protocols could be developed for efficient transmission of video and audio over wireless networks.

In the Signal Processing and Multimedia industry this is a very popular area of research. Much of the emphasis in this area is laid on finding Faster and more efficient ways to calculate Motion Vectors and also to find optimal Block Sizes. For example, there are various Motion Compensation Algorithms. Two of which are investigated in this project namely, Exhaustive Search Algorithm and Conjugate Directional Search Algorithm.

The report is organized as follows:

2.0 Procedure for Exhaustive Search Algorithm

3.0 Results for Exhaustive Search Algorithm

5.0 Procedure for Conjugate Directional Algorithm

6.0 Results for Conjugate Directional Algorithm

7.0 Comparisons of the Results for both Algorithms

8.0 Conclusions

9.0 References

2.0 PROCEDURE FOR EXHAUSTIVE SEARCH

2.1 AIM

The aim of this project is to perform Motion Compensation for the coast guard video sequence.

Two proposed algorithms are used. Firstly, the Exhaustive Search Algorithm is explored. The details of the procedure are outlined below.

2.2 STEPS UNDERTAKEN

2.2.1 Read the sequence.

The first step to any Image Processing related algorithm is to first load the images or video sequences into the work space so that it can be manipulated. It should be noted that since Matlab does not read YUV files. Instead these files should first be converted to .avi files. The video sequence used for this project is the Coastguard_qcif.avi. It should be noted that a qcif sequence has a frame size of 144x176. Once the file is read, the Reference and Target frames need to individually extracted and stored as the search algorithms work with 2 frames at a time.

2.2.2. Extract Block from the target Frame

Once the Reference and Target frames are set then, the next step would be to extract an individual Macroblock. A function was written to do this. This function takes in x and y coordinates in increments of 16 and returns a single block. The entire code is written for a single Macroblock and iterations step from one Macro Block to another.

2.2.3. Extract Search Window from Reference Frame

Once the Macroblock is extracted, the next step is to extract a search window for the Macroblock. However, the Search Window is extracted from the Reference Frame while the Macroblock comes from the Target Frame. Before, presenting a synopsis of Search Window extraction, one should keep in mind the nature of the Search Window. The basic concept of Search Window comes from the fact that the best match for a macroblock can be enclosed in a square window with the Macroblock being equidistant from all corners of the Search Window. However, this is not possible for all the blocks as some blocks are in the 4 corners of the image or are either in the first and last rows or first and last columns. Consequently, different window sizes were defined. For exhaustive search, every possible displacement within a rectangular search window is attempted. The displacement that produces the minimum distortion is chosen as the motion vector.

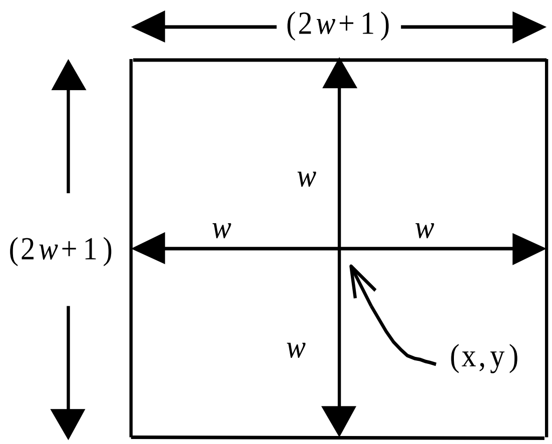

Figure 1.0: Shows the SW for ES

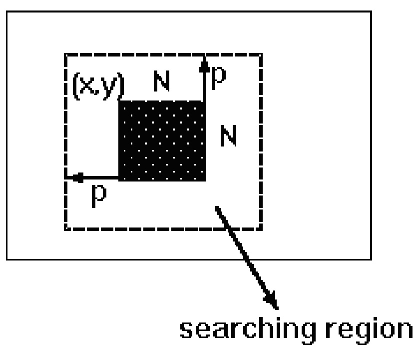

Figure 2.0: Shows a N x N SW

It should be noted that for a given block, if the Macroblock cannot be placed equidistant from the edges of the Search Window, then the Macroblock should be placed 8 pixels away in each allowable direction. Thus, the combinations of the Search Window Dimensions are as follows: **32 x 32; 24 x 24; 32 x 24 and 24 x 32**.

2.2.4. Determine the Best Match Macroblock

The Exhaustive Search Algorithm Search is a block based search algorithm. The Motion Estimation over a Macroblock of pixels is a standard approach for estimating motion in a moving image sequence. In this algorithm, every possible displacement within a rectangular search window is attempted. The displacement that produces the minimum distortion is chosen as the motion vector. To explain this further,

- Consider a single Macroblock
- Consider the appropriate Search Window
- Align the Macroblock on the top left side of the Window.
- Consider the first 16 x 16 pixels in the Window
- Compute the pixel by pixel difference for that Macroblock
- Then calculate the absolute sum of the Difference Matrix and store it into an array
- Increment the column index of the Search Window by one pixel to the right and repeat step 4 – 6.
- Continue to increment the column index till the Macroblock does not exceed the column dimension of the matrix.
- Once, the column index has reached its dimension, then reset the column vector and increment to the next row
- Repeat steps 4-9 again till there are no more possible iterations
- This procedure is called a raster scan for the window.
- Then Choose the smallest Maximum Absolute Difference value as the point of

best match

- Then tabulate the coefficients for the new position of the Macroblock and store the Error Block.
- Pass new position coordinate values and store it for the Motion Compensation.

2.2.5. Mean Absolute Difference (MAD)

There are two popular ways to tabulate the distortion for a Macroblock. These are Mean Square Error (MSE) and MAD. For a Macroblock of 16 x 16, by using MSE there will be 256 Multiplications and 768 Additions, where as MAD uses 512 Additions. This is a 33.3% reduction in the number of additions performed. Furthermore, MAD performs just as well as MSE in terms of reducing the entropy. Hence, MAD is used. It is given by the formula below:

$$Distortion(B_k(n),d) \quad = \sum_{\forall (x,y)\in B_k(n)} (U_k(x,y) - U_r(x+d_x, y+d_y))^2 \qquad \textbf{(1)}$$

For Mean Absolute Difference (MAD), the distortion for a macroblock $B_k(n)$ is given by:

$$Distortion(B_k(n),d) \quad = \sum_{\forall (x,y)\in B_k(n)} \left| U_k(x,y) - U_r(x+d_x, y+d_y) \right| \qquad \textbf{(2)}$$

Where, for both cases, $U_k(x,y)$ represents pixels in the current frame, and $U_r(x,y)$ represents pixels in the reference frame.

2.2.6. Calculate and store MV coordinates

This function utilizes the positions of the Matched Macroblock and calculates the displacement of pixels. The results are then stored into a dynamic array that increments and stores as each block is dealt with. Once all the vectors are tabulated, the Motion

Vector Image is displayed. Each dot indicates a Macroblock and the arrows indicate the direction of motion of the Macroblock.

2.2.7. Reconstructed Image

Once the Motion Vectors have been generated, the Frame has to be reconstructed. Therefore, there is a decoder in every encoder. In order to decode, the image, the Error Image that was generated from the Motion Vectors is added pixel by pixel to the Reference Frame to retrieve the Target Frame. Subsequently, the Reconstructed Image is displayed.

3.0 RESULTS FOR EXHAUSTIVE SEARCH

FRAME 1

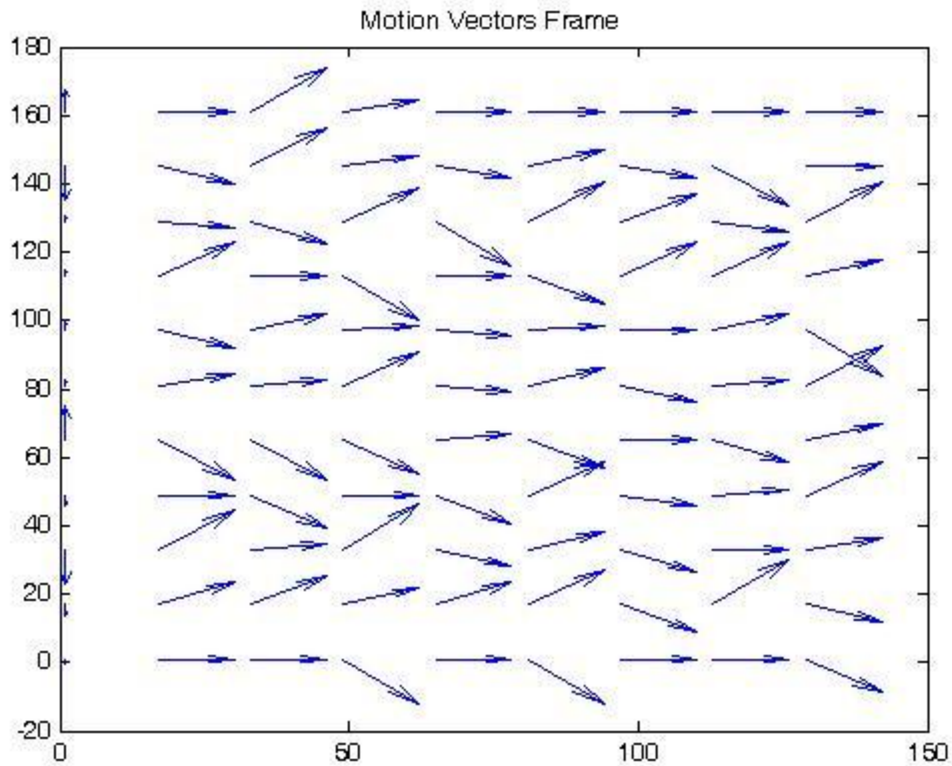

Figure 3.0: Motion Vector for Frame 1

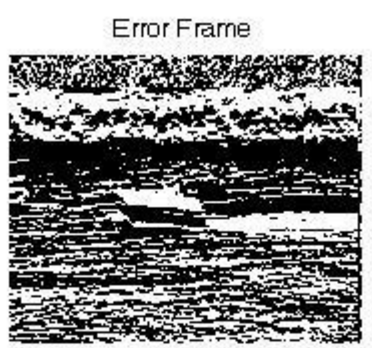

Figure 4.0 : Shows the Error Frame 1

Figure 5.0: Target Frame 1

Figure 6.0: Shows the Reconstructed Frame 1

FRAME_2

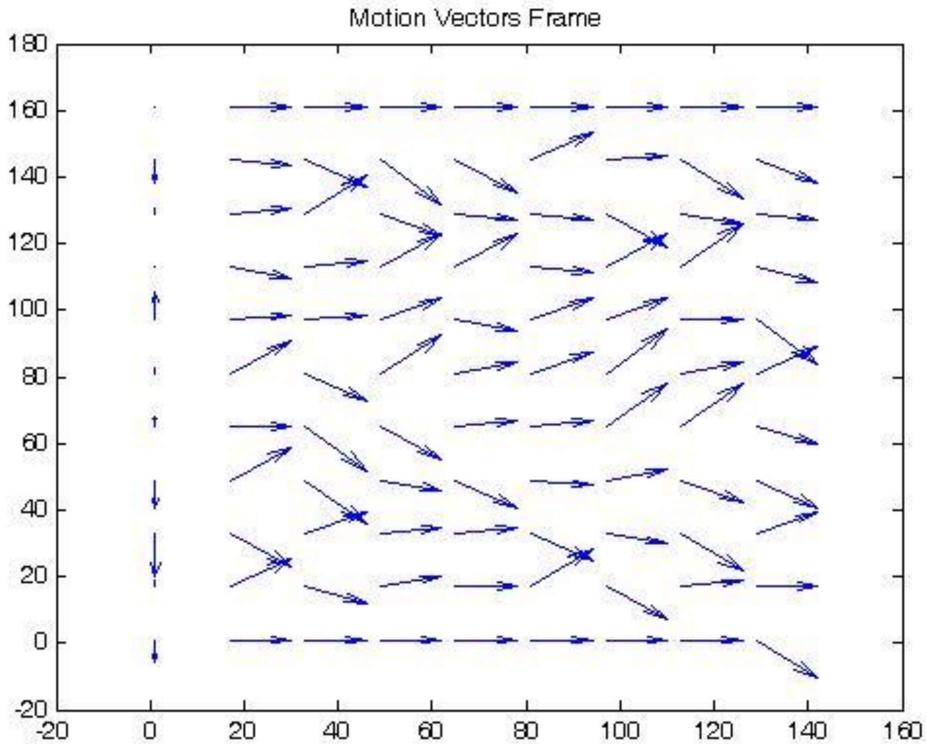

Figure 7.0: Shows the MV for Frame 2

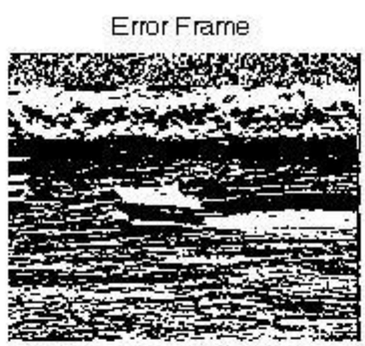

Figure 8.0: Shows Error Frame 2

Figure 9.0: Shows Target Frame 2

Figure 10.0: Shows Reconstructed Frame 2

FRAME 3

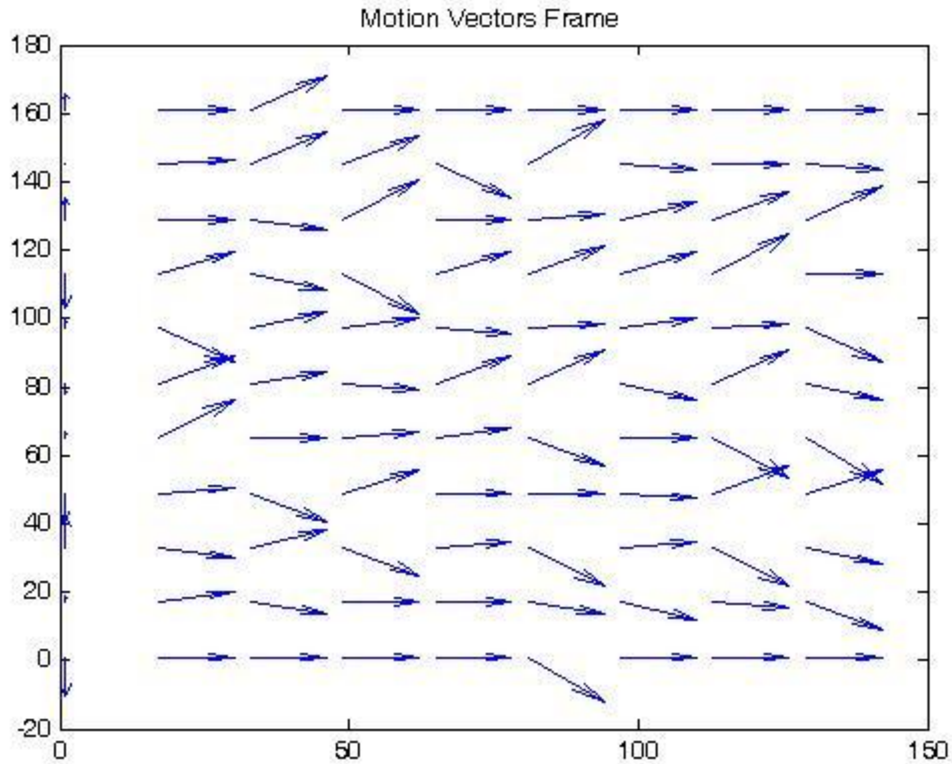

Figure 11.0: Motion Vectors for Frame 3

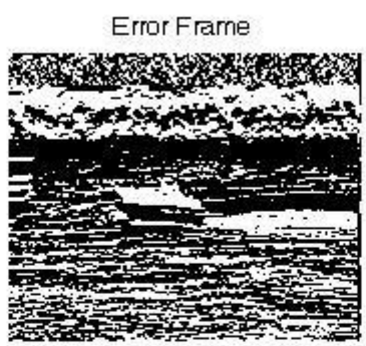

Figure 12.0: Error Frame 3

Figure 13.0: Target Frame 3

Figure 14.0: Reconstructed Frame 3

FRAME 4

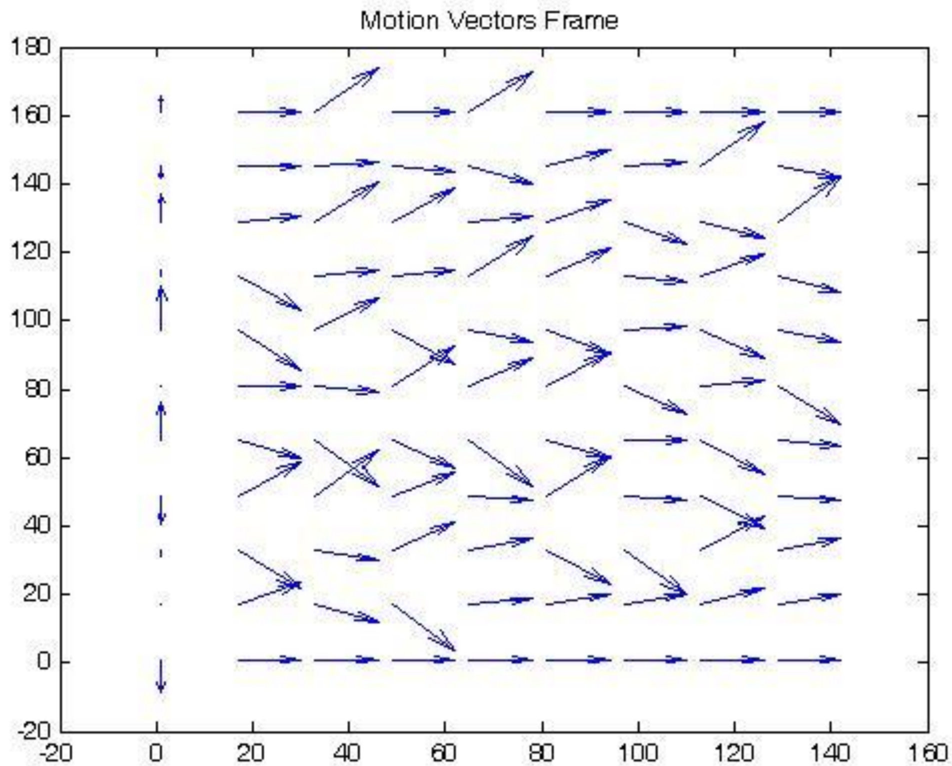

Figure 15.0: Shows Motion Vectors for Frame 4

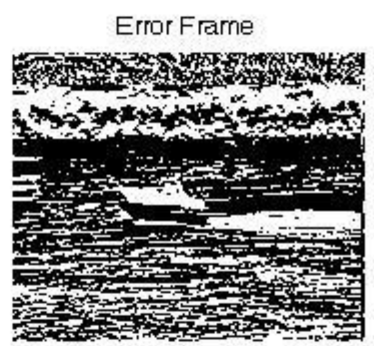

Figure 16.0: Error Frame 4

Figure 17.0: Shows the Target Frame 4

Figure 18.0: Shows the Reconstructed Frame 4

FRAME 5

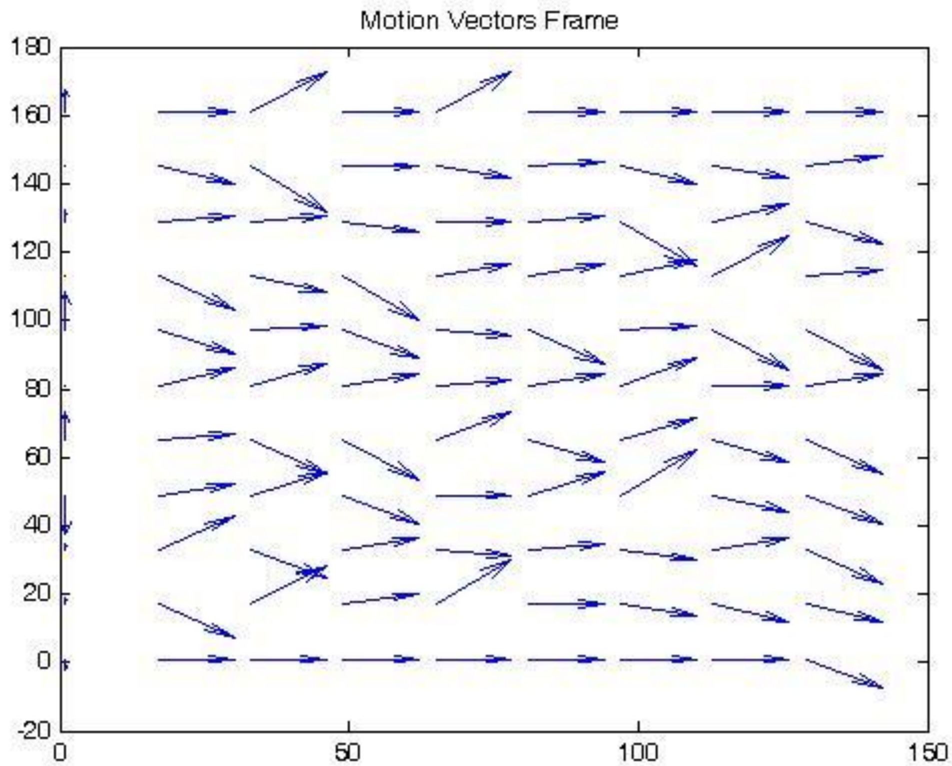

Figure 19.0: Shows the Motion Vectors for Frame 5

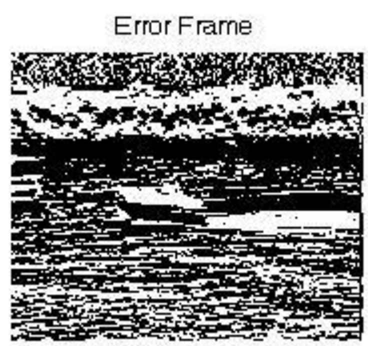

Figure 20.0: Shows the Error Frame 5

Figure 21.0: Shows the Target Frame 5

Figure 22.0: Shows the Reconstructed Frame 5

4.0 PROCEDURE FOR CONJUGATE SEARCH

4.1 General Procedure

Exhaustive search has a huge overhead because of lot of iterations that are involved. In order to overcome this, fast algorithms like logarithmic search and conjugate search were designed. In this project we aim at implementing conjugate search. The idea is to move the macroblock one pixel in all directions in the search window and find the direction of minimum error and proceed in that direction to find the best match. This algorithm drastically reduces the overhead but the accuracy will be lesser when compared to the exhaustive search.

4.2 STEPS UNDERTAKEN

4.2.1 Read the sequence

The sequence is read as described in **2.2.1.**

4.2.2. Extract Block from the target Frame

Once the Reference and Target frames are set then, the next step would be to extract individual Macroblocks. This is achieved by using two for loops whose range runs across the size of the frame in increments of 16. Each time a macroblock is obtained and all the required computations are performed on it. In the subsequent iterations the other macroblocks are accessed and the procedure continued.

4.2.3. Extract Search Window from Reference Frame

Once the Macroblock is extracted, the next step is to extract a search window for the Macroblock. The search window is obtained by simply moving eight pixels in all directions from the obtained macroblock. It should be noted that the search window has to be defined in the reference frame.

This is achieved by using two indices which describe the boundary of the search window. But the search window for the macroblocks in the first and the last rows and columns cannot be obtained by moving in all four directions. Hence we will have different dimension search windows for these few macroblocks. Thus, the combinations of the Search Window Dimensions are as follows: **32 x 32; 24 x 24; 32 x 24 and 24 x 32.** This is implemented by using separate for and while loops for the first and the last rows and columns. The search window is as shown in Figure **2.0**. .

4.2.4. Determine the Best Match Macroblock

In order to reduce the computational complexity involved in exhaustive search, fast search algorithms were introduced. One such fast search algorithm is Conjugate Search. In this algorithm we attempt to move the macroblock one pixel in all four directions, find the Mean Absolute Differences (MAD) in these directions and at the initial position. MAD is found using the formula described in section 2.2.5. The algorithm is made explained below:

- Consider a single Macroblock
- Consider the appropriate Search Window
- Align the Macroblock on the top left side of the Window.
- Consider the first 16 x 16 pixels in the Window
- Compute the pixel by pixel difference for that Macroblock
- Then calculate the absolute sum of the Difference Matrix and store it into an array
- Increment the column index of the Search Window by one pixel to the right

and repeat step 4 – 6.

- Then from the original position decrement the column index of the Search Window by one pixel to the left and repeat steps 4-6.
- Similar to above, from the original position increment the row index of the Search Window by one pixel to the top and repeat steps 4-6.
- Again, from the original position decrement the column index of the Search Window by one pixel to the bottom and repeat steps 4-6.
- Find the minimum among the five absolute sums of the difference matrices that are obtained.
- Once the minimum is found, the macroblock is shifted in the direction of the minimum value.
- Repeat steps 5-11 again till either the error becomes zero or the macroblock hits one of the edges of the search window. The macroblock currently accessed becomes the best match.
- Then tabulate the coefficients for the new position of the Macroblock and store the Error Block.
- Pass new position coordinate values and store it for the Motion Compensation.

4.2.5. Calculate and store MV coordinates and Reconstructed Image

The motion vectors are calculated as described in **2.2.6.** The reconstructed image is obtained by adding the error image and motion vectors to the reference frame as described in **2.2.7**

5.0 Results for Conjugate Search

The Original image is as shown below. The subsequent images show various manipulations that were done on this image in the course of this project.

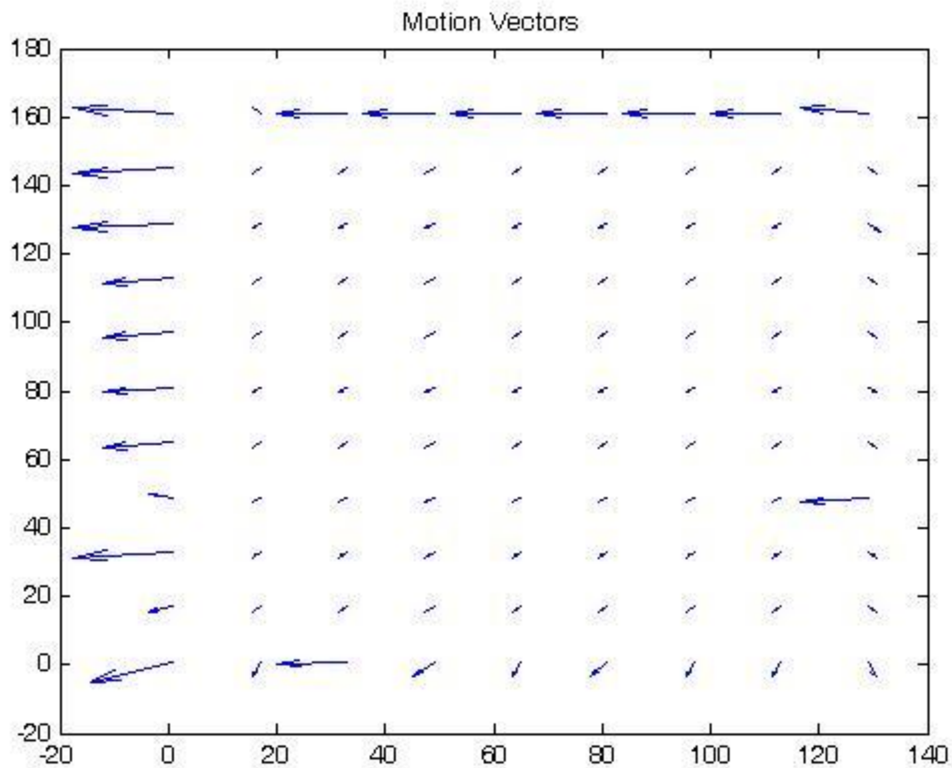

Figure 23.0: Shows the Motion Vector Frame 1

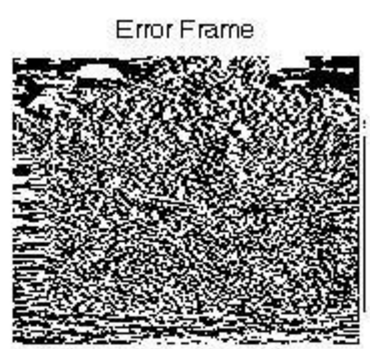

Figure 24.0: Shows the Error Frame 1

Figure 25.0: Shows the Target Frame 1

Figure 26.0: Shows the Reconstructed Frame

FRAME 2

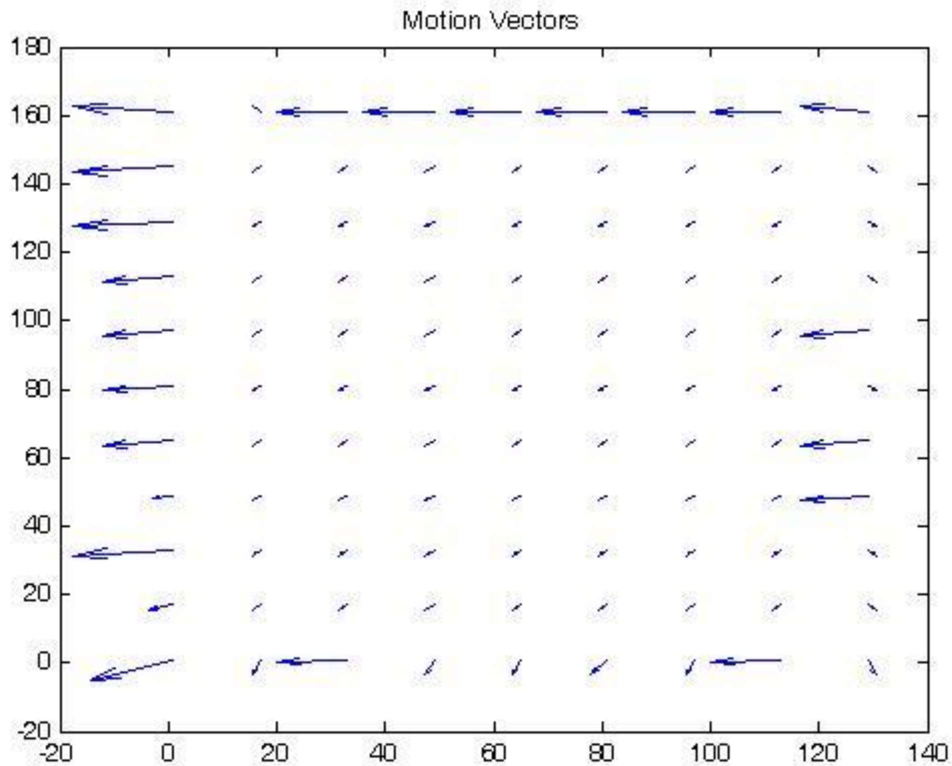

Figure 27.0: Shows the MV for Frame 2

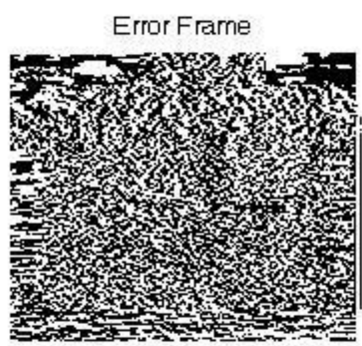

Figure 28.0: Shows the Error Frame 2

Figure 29.0: Shows Target Frame 2

Figure 30.0: Shows the Reconstructed Frame 2

FRAME 3

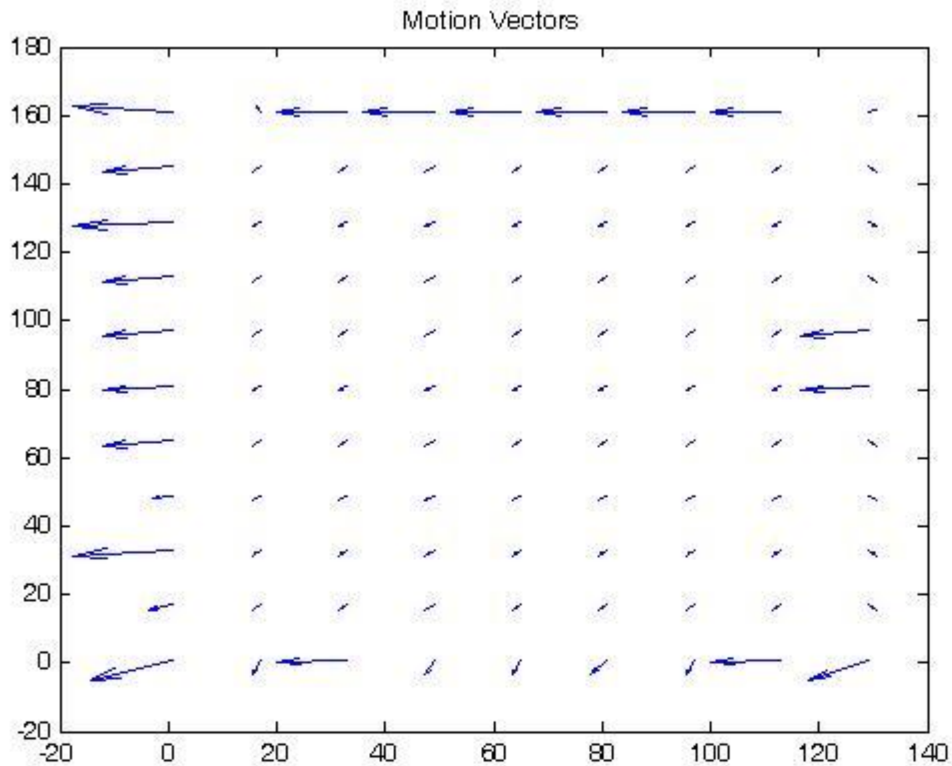

Figure 31.0: Shows the Motion Vector for Frame 3

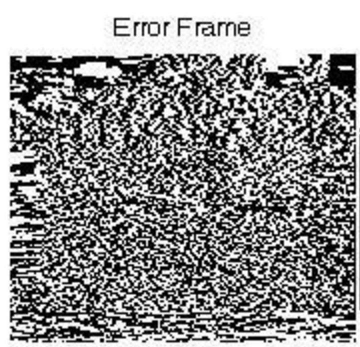

Figure 32.0: Error Frame 3

Figure 33.0: Shows Target Frame 3

Figure 34.0: Shows the Reconstructed Frame 3

FRAME 4

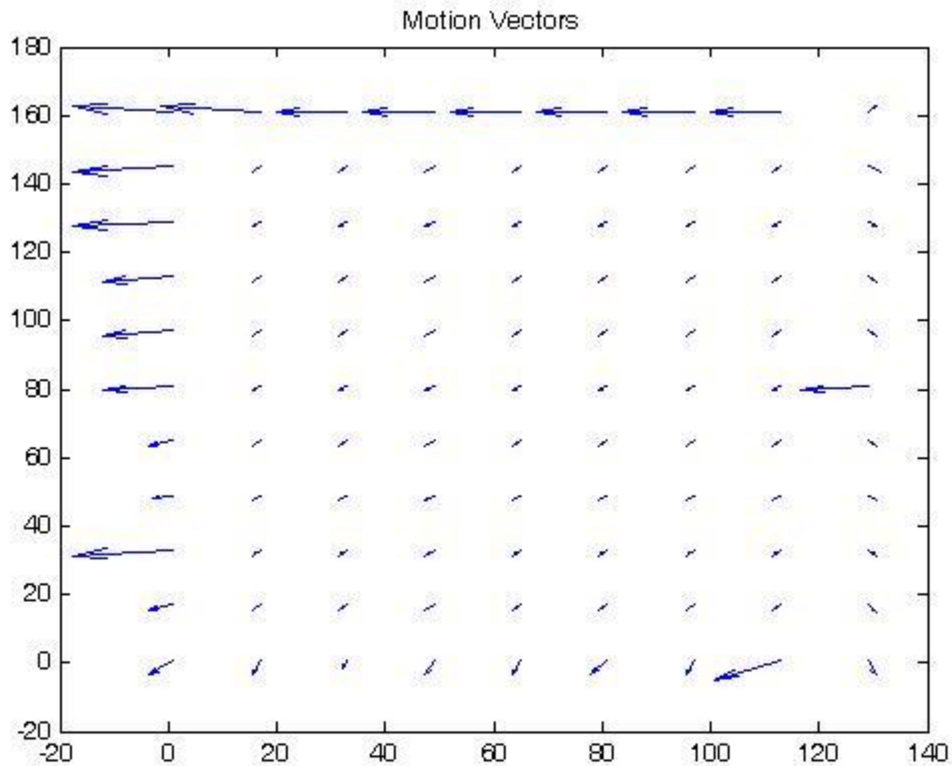

Figure 35.0: Shows Motion Vector Frame 4

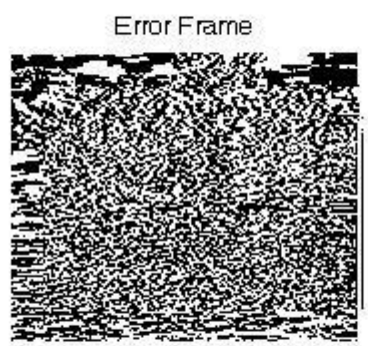

Figure 36.0: Error Frame 4

Figure 37.0: Target Frame 4

Figure 38.0: Shows reconstructed Frame 4

FRAME 5

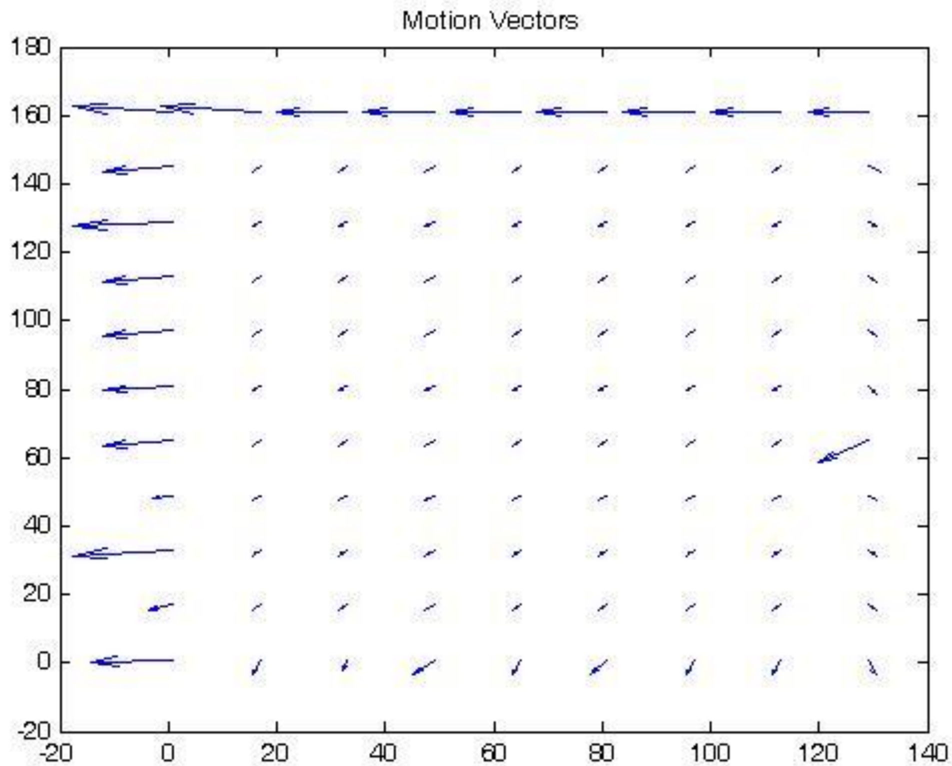

Figure 39.0: Shows Motion Vectors for Frame 5

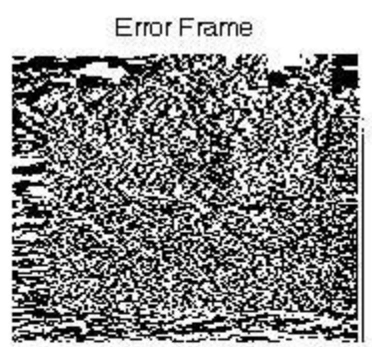

Figure 40.0: Shows error Frame 5

Figure 41.0: Shows the Target Frame 5

Figure 42.0: Shows the Reconstructed Frame

6.0 Discussion

7.0 Conclusion

8.0 References

1] Ze-Nian Li and Mark.S.Drew, "Fundamentals of Multimedia," Princeton Hall, 2004.

2] Website: www.**wikipedia**.org

3] Multimedia Communication Systems