**Compe 565,** **Semester 2021**

**HW 3: Motion Estimation for Video Compression**

**Prepared by**

Ethan Nagelvoort (821234668)

E-mail: enagelvoort@sdsu.edu

Electrical and Computer Engineering

San Diego State University

**Table of Contents**

* List of figures …………………………………………………………………………………….2
* Introduction ……………………………………………………………………………………....3
* Procedural Section ………………………………………………………………………………. 4

1. Exhausted Search (SAD method) ………………………………………………………4
2. Error Image……………………………………………………………………………..4
3. Reconstruction of P-frame from I-frame ……………………………………………...5

* Results……………………………………………………………………………………………..6
* Conclusion………………………………………………………………………………………..15
* References ……………………………………………………………………………………….15

**List of figures**

Figure 1: Motion Vectors for Frames 6 and 7

Figure 2 : Motion Vectors for Frame 7 and 8

Figure 3: Motion Vectors for Frames 8 and 9

Figure 4: Motion Vectors for Frames 9 and 10

Figure 5: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 7

Figure 6: Original vs. Reconstructed RGB Image for P Frame 7

Figure 7: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 8

Figure 8: Original vs. Reconstructed RGB Image for P Frame 8

Figure 9: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 9

Figure 10:Original vs. Reconstructed RGB Image for P Frame 9

Figure 11: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 9

Figure 12:Original vs. Reconstructed RGB Image for P Frame 10

Figure 13: Computations and additions/subtractions

**Introduction**

Video compression is the act of reducing storage and net cost of a video by manipulating its frames. This compression reduces temporal, spatial, psychovisual, and coding redundancies. Each frame is made up of a certain number of macroblocks and programmers manipulate these macroblocks to achieve video compression. Programmers also perform motion compensation and prediction at the macroblock level. This helps programmers remove temporal redundancies, one of the redundancies reduced in video compression.

In this homework, I utilized Matlab to manipulate a walking video in order to predict its motion vectors frame by frame. I did this by manipulating its frames at the macroblock level, which in turn reduced its temporal redundancies. I used the SAD method of prediction where you determine the minimum SAD in the frame to attain the perfect match to the macroblock you are currently manipulating. I also reconstructed the frames through decoding and obtained an error image.

**Procedural Section**

1. Exhausted Search (SAD method)

I first converted all the frames in the video to ycbcr and then extracted frames 6-10, which are the required frames. I then began a loop that would loop through all the frames and in the loop I then downsampled each frame to 4:2:0 format. I then have another set of for loops that allow me to traverse the frame one macroblock at a time. Each macroblock is 16x16 so each of these for loops increment by 16. I then have a series of if statements that check the sides of the frame to ensure that the current macroblock is within the frame. These if statements set side variables and their values depend on whether the macroblock being checked fits within the frame. These variables essentially set a search window. I then declare a reference block to be the top left corner of that search window and set a variable min to a very big number. I also record the position of the current block I am at . I then have another set of for loops that depend on the search window where I loop through every block there. Within these for loops, I determine the minimum value of SAD and once I find the block that has that value, I record the current motion vectors. This is because the macroblock with the minimum SAD is the best match. These motion vectors will then be stored into dx and dy. Right before the code iterates to proceed to the next frame, the function quiver is used, with the values stored in dx and dy to output the motion vectors.

1. Error Image

To do this I took the current frame and subtracted it by the best match in the reference frame. This gave me the residual frame and this calculation occurs after the nested for loop, where minimum SAD is determined. Then, to display it, I perform the absolute value operation since some of its values may be negative.

1. Reconstruction of P-frame from I-frame

The frames are in IPPPP format and so we were to predict and reconstruct the P frames from the I frame. I did this by attaining the y component of the I frame and setting to variable recon. Then in the code, somewhere after attaining the minimum SAD, I subtract recon from the residual frame previously calculated. This gives me the next P frame. I also set recon to this P frame so that the next P frame can be predicted from here. I then perform linear interpolation to upsample cb and cr to display both a reconstructed y-component of the P frames and rgb form of the p frames.

**Results**

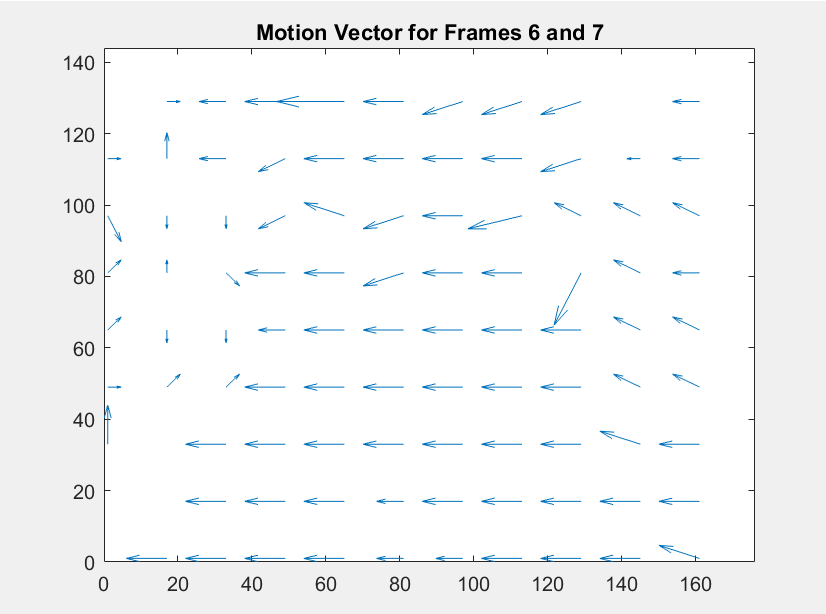


Figure 1: Motion Vectors for Frames 6 and 7

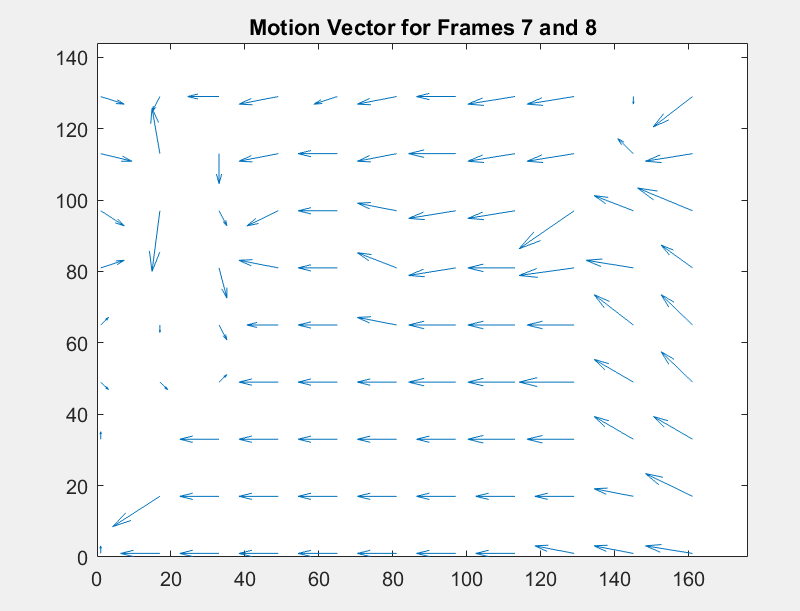


Figure 2 : Motion Vectors for Frame 7 and 8

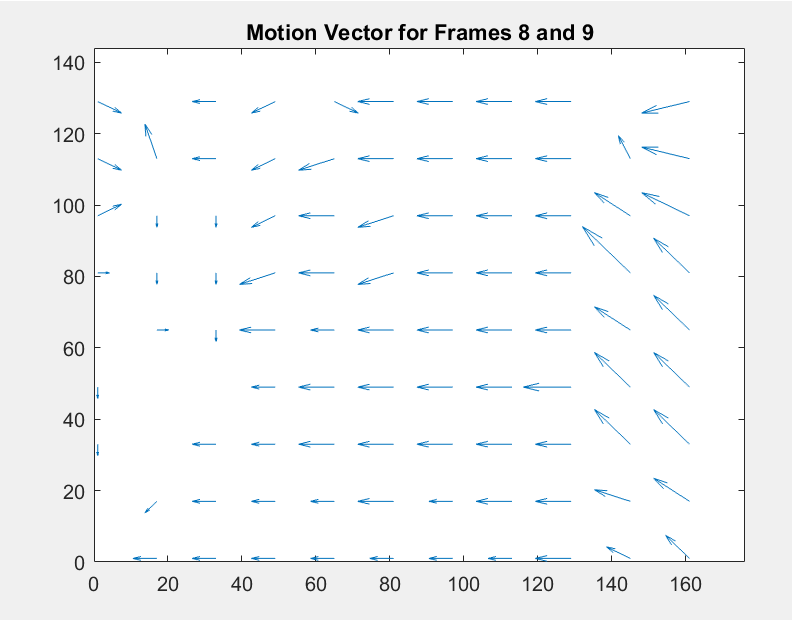


Figure 3: Motion Vectors for Frames 8 and 9

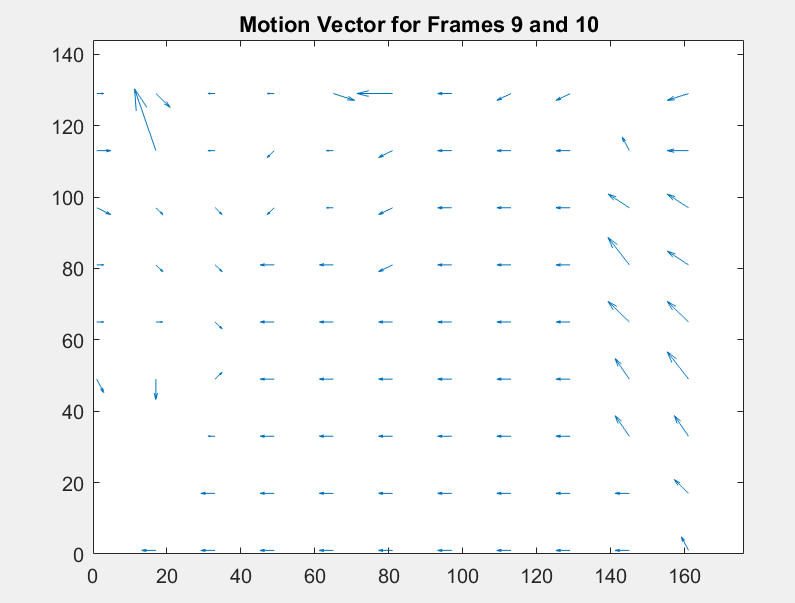


Figure 4: Motion Vectors for Frames 9 and 10

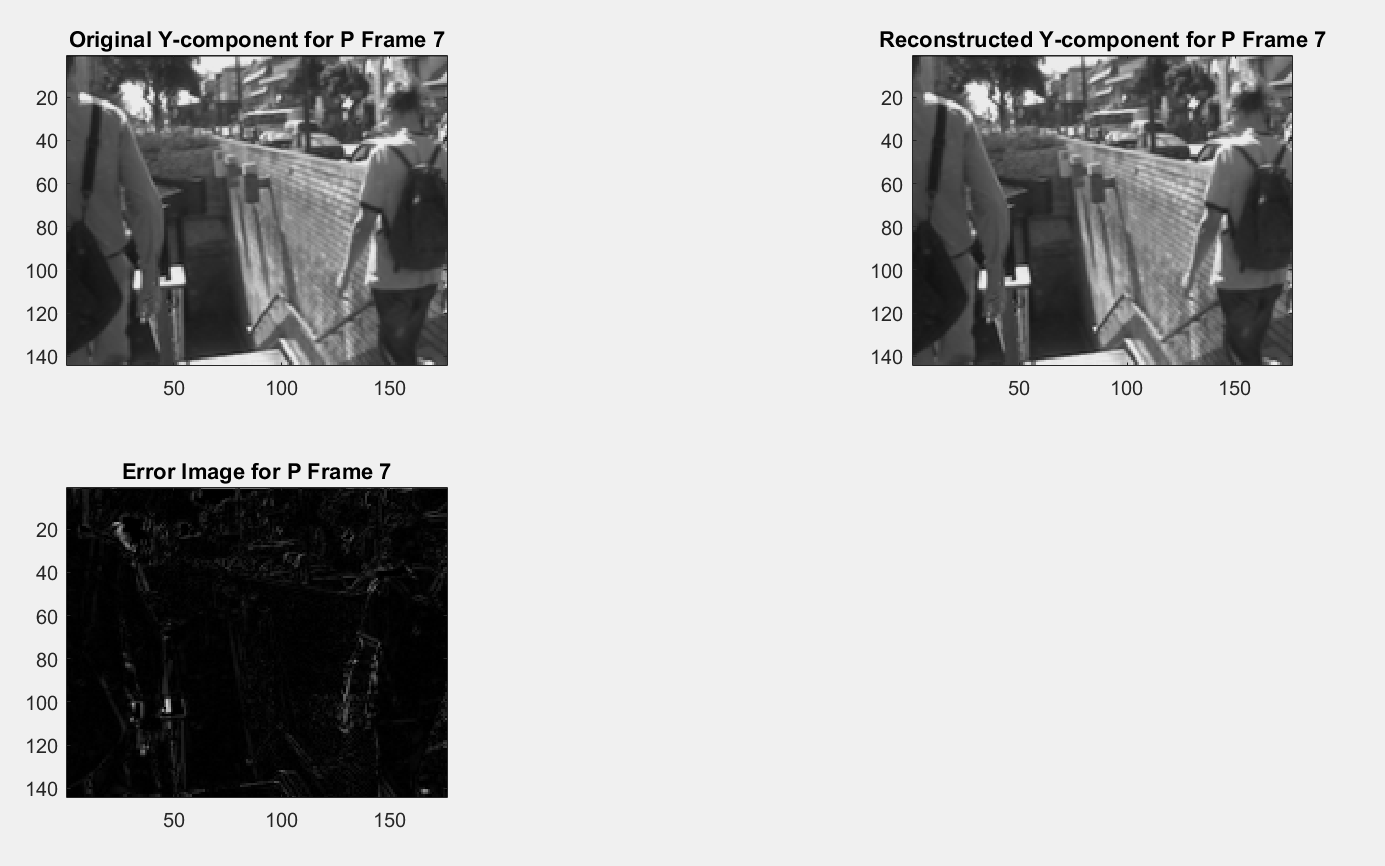


Figure 5: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 7

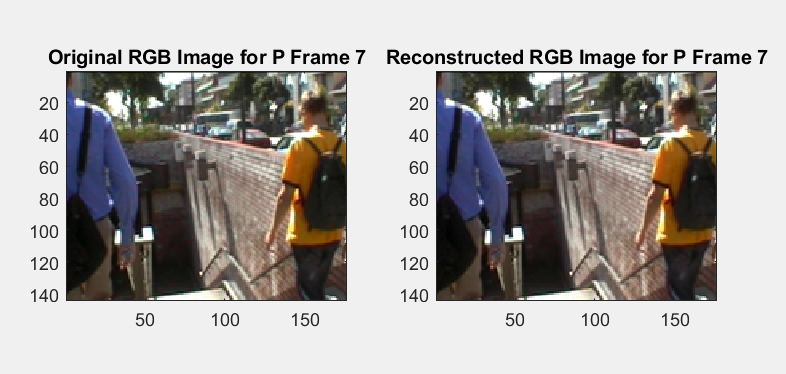


Figure 6: Original vs. Reconstructed RGB Image for P Frame 7



Figure 7: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 8

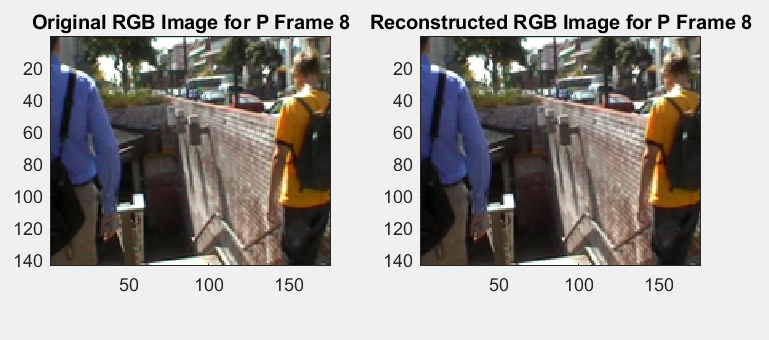


Figure 8: Original vs. Reconstructed RGB Image for P Frame 8

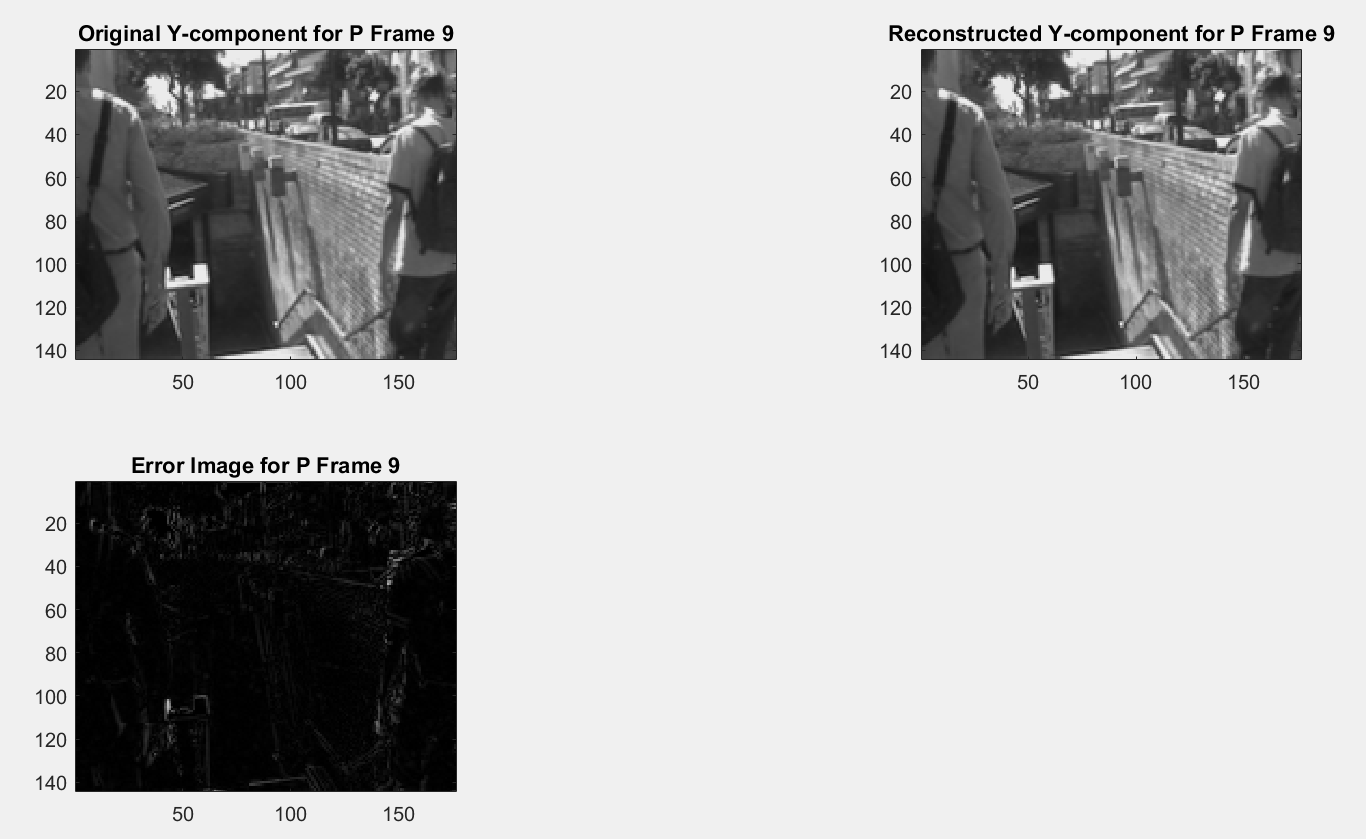


Figure 9: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 9

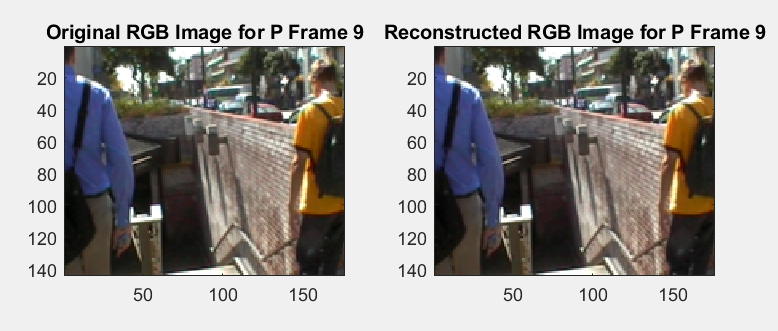


Figure 10:Original vs. Reconstructed RGB Image for P Frame 9

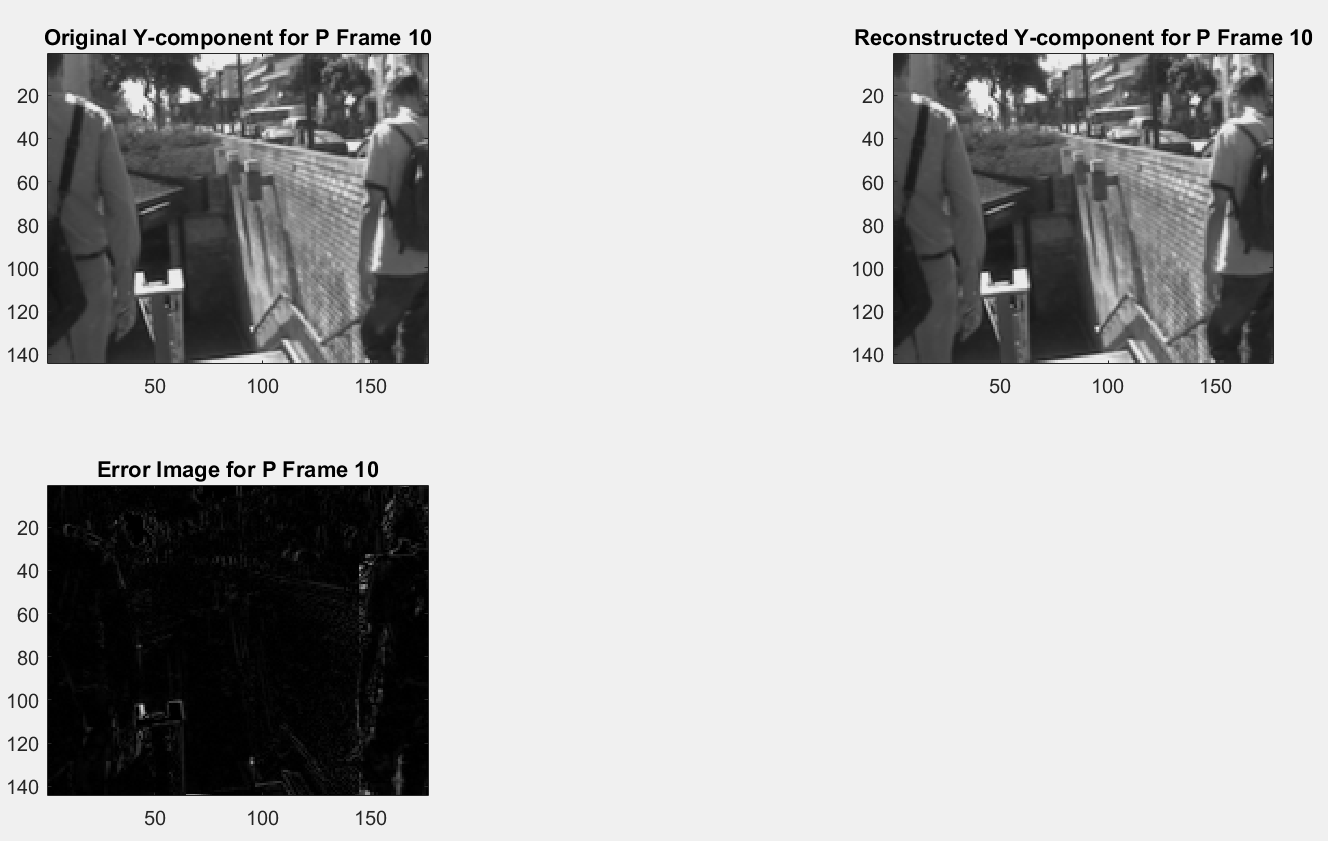


Figure 11: Original Y-comp, Reconstructed Y-comp, and Error Image for P Frame 9

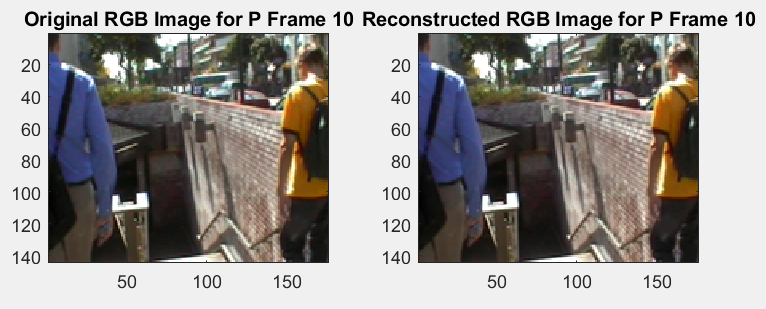


Figure 12:Original vs. Reconstructed RGB Image for P Frame 10

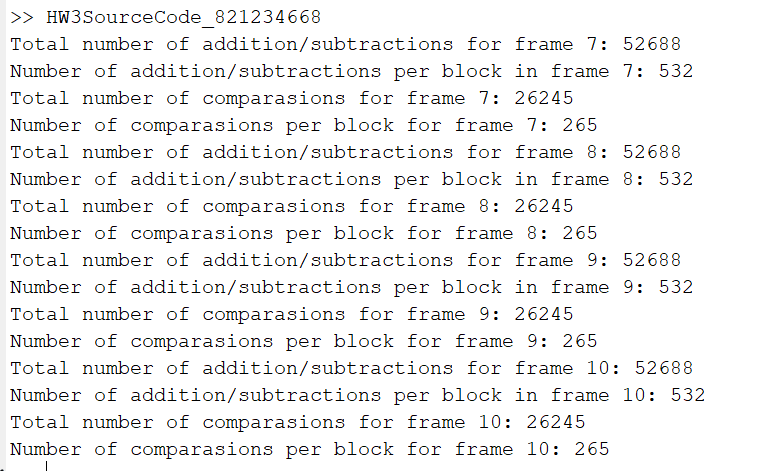


Figure 13: Computations and additions/subtractions

The number of additions/subtractions per a 16x16 block while utilizing SAD is 511 . This is according to the HW3 document provided to us. Since the number of additions/subtractions my method has is 532, which is very close to 511, then it can be concluded that my method of SAD and exhausted search is very accurate. This is also true because the number of comparisons I do is 265 which is very close to (2\*8+1)^2, which equals 289.

**Conclusion**

In this assignment, I learned how to utilize the exhaustive search method, combined with determining the minimum SAD, to determine motion vectors. This helps programmers predict P frames in a video in order to determine what will be the appropriate next frame. This process showed me how to manipulate macroblocks, leading to the reduction of temporal redundancies. I also learned that you can use the residual frame, once found, to reconstruct a P frame once it was subsampled.

**Reference**

[1] S. Kumar, “Video Compression Part1”, Compe 565, 2021

[2]S. Kumar, “Video Compression Part2”, Compe 565, 2021