**CompE565, Spring 2016**

**Project 4**

**Prepared by**

*Benjamin Hunt Yusuf Shaikh*

E-mail: [*yusufsh23@gmail.com*](mailto:yusufsh23@gmail.com)*, btjammin@sbcglobal.net*

Electrical and Computer Engineering

San Diego State University

**List of figures:**

List of all the figures in the document

**Figure (1) =** Motion Vectors between Frames 1 and 2.

**Figure (2) =** Motion Vectors between Frames 2 and 3.

**Figure (3) =** Motion Vectors between Frames 3 and 4.

**Figure (4) =** Motion Vectors between Frames 4 and 5.

**Figure (5) =** Frame 1: Original, Reconstructed and Error

**Figure (6) =** Frame 2: Original, Reconstructed and Error

**Figure (7) =** Frame 3: Original, Reconstructed and Error

**Figure (8) =** Frame 4: Original, Reconstructed and Error

**Figure (9) =** Frame 5: Original, Reconstructed and Error

# Introduction:

# Procedural Section:

This project consisted of multiple functions that were called within each other. We Decided to create one main Execute file that simply Read the Video saved it in a variable and transferred it to an Encoder Function.

This Encoder Function then has a goal to obtain all the Compressed Y Cb and Cr portions of each frame of the video, send these portions to Decoder for the next step.

Before starting of with the main task, we started off by creating two empty arrays where the vectors would be stored. One for Y and the other for CbCr This process was done in project 3 and we simply repeated it here.

Now for the main task of compressing the Y Cb and Cr components and sending it to the Decoder we had to create three 5-Dimensional Arrays (Because we divided the video into 5 frames). The reason that there were three of these arrays is due to Y, Cb and Cr. We created a loop, which would count the current frame we are at. The first time count went through the loop, the…

*First dimension of the Y component array, first dimension of the Cb component array* and *first dimension of the Cr component array*

…Would receive the pixel values of Frames compressed Y, Cb and Cr component respectively. This would then be repeated 5 times until all 5 dimensions have been filled also all 5 frames have been lapsed.

The process of compression was done through Subsample and Interpol Up sampling done in project 1. After each of Y Cb and Cr components were Up sampled they were concatenated together into one file of YCbCr.

We then had to find the error of from the original image and this concatenated image. A simple subtraction did such. We saved this file as the variable Error.

This Error file was then sent to Encode, which completed the process of Quantizing and DCT using the Quantizing and DCT matrix. The encoder file would send back the Quantized Y, Cb and Cr functions.

These are the Y Cb and Cr functions that were stored in the 5D arrays as said before. Once the Array was all filled up we would send them to Decoder Functions.

At this point we would transfer the 5-Dimension Y , Cb , and Cr Array also with the Y MV array and CbCr MV array to the Decoder function. This Decoder function would do simply do the Inverse DCT and DeQuantization of each dimension of each of Portion ( Y, Cb and Cr).

To complete this task we sent each individual dimension of Y Cb and Cr to the Decode function where it was Dequantized and IDCTed and returned. How each dimension was called was through a ‘for’ loop counting the dimension that it is at. After doing such each new Y Cb and Cr was saved in their corresponding spots ready for the next step.

The next step was to use MotionComp used in project 3 to compress each component further in comparison to the original, which we saved. We used the Motion vectors here.

Once again like in Encoder we Concatenated the three components together and found the error.

We would then display the original image, Reconstructed image (Concatenated) and the error between the two.

The last part was actually completed in encoder however was codded after. This was to display the Vectors. As done in project 3 we created a figure of size X(-20 to 180) and Y(0 to 160). Each vector started at a multiple of 20 and was evenly distributed.

Functions Used: Execute, Encoder, Encode, Decoder, Decode, Mincost, MotionComp, MotionEstEs, ZigZag, and MadComputations.

# Results:

# 

# 

# 

# Conclusion:

# In all, this homework combined everything that we incorporated this semester in the lectures and the past homework. We Took the File, which was a video, broke it into 5 separate frames, pictures, and started the compression process. In project 1 we learned to separate an image into 3 different components the Y, Cb, and Cr. We completed the something here for each individual frame. After the frames were separated into the three components we had to do the DCT, Quantization, ZigZag, DeQuant and Inverse DCT on each function.

YCbCr -> DCT -> Quantization -> ZigZag -> DeQuantization -> IDCT ->YCbCr

# This is the simple process of compression, which we learned in our classroom lectures. We also completed this in our second project. Our third project then incorporated all the techniques that we used for video compression: MAD Computations, MinCost, MotionComp, MostionEstEs. In conclusion we learned that to compress a video to the fullest extent we have to have all of the previous methods intact together.

# References:

Place all references using the format given below.

[1] S. Kumar, "Lecture 1 & 2" Journal or Conference Name, Vol. 18, No. 1, 1988.

[2] "Documentation." Distinct Block Processing. MathWorks, n.d. Web. 09 Mar. 2016.

[3] Barjatya, Aroh. "Block Matching Algorithms for Motion Estimation - File Exchange - MATLAB Central." *Block Matching Algorithms*. MathWorks, n.d. Web. 31 Mar. 2016.