CS 576 – Assignment 1
Instructor: Parag Havaldar

Assigned on 01/22/18,
Solutions due on 02/12/18 by midday 12 pm noon
Total Marks – 200 points

Late Policy: None, unless prior arrangement has been made

Written Part: (25 points)

Each question has marks displayed

Q.1 Suppose a camera has 450 lines per frame, 520 pixels per line with a color sub sampling scheme is 4:2:0. The camera has a frame rate of 25 Hz, and each sample of Y, Cr, Cb is quantized with 8 bits

- What is the bit-rate produced by the camera? *(2 points)*
- Suppose we want to store the video signal on a hard disk, and, in order to save space, re-quantize each chrominance (Cr, Cb) signals with only 6 bits per sample. What is the minimum size of the hard disk required to store 10 minutes of video *(3 points)*

Q.2 The following sequence of real numbers has been obtained sampling an audio signal: 1.8, 2.2, 2.2, 3.2, 3.3, 3.3, 2.5, 2.8, 2.8, 2.8, 1.5, 1.0, 1.2, 1.2, 1.8, 2.2, 2.2, 2.2, 1.9, 2.3, 1.2, 0.2, -1.2, -1.2, -1.7, -1.1, -2.2, -1.5, -1.5, -0.7, 0.1, 0.9 Quantize this sequence by dividing the interval [-4, 4] into 32 uniformly distributed levels (place the level 0 at -3.75, the level 1 at -3.5, and so on. This should simplify your calculations).

- Write down the quantized sequence. *(4 points)*
- How many bits do you need to transmit it? *(1 points)*

Q.3 Temporal aliasing can be observed when you attempt to record a rotating wheel with a video camera. In this problem, you will analyze such effects. Assume there is a car moving at 36 km/hr and you record the car using a film, which traditionally record at 24 frames per second. The tires have a diameter of 0.4244 meters. Each tire has a white mark to gauge the speed of rotation.

- If you are watching this projected movie in a theatre, what do you perceive the rate of tire rotation to be in rotations/sec? *(5 points)*
- If you use your camcorder to record the movie in the theater and your camcorder is recording at one third film rate (ie 8 fps), at what rate (rotations/sec) does the tire rotate in your video recording *(5 points)*
- If you use an NTSC camera with 30 fps, what is the maximum speed that the car can go at so that you see no aliasing in the recording *(5 points)*

Programming Part: (85 points)

This assignment will help you gain a practical understanding of Quantization and Subsampling to analyze how it affects visual media types like images and video.
We have provided you with a Microsoft Visual C++ project and a java class to display two images side by side (left – original and right – output of your program). Currently both left and right correspond to the same input image. You are free to use this display program as a start, or write your own in a language of your choice.

Input to your program will be five parameters where

- The first parameter is the name of the image, which will be provided in an 8 bit per channel RGB format (Total 24 bits per pixel). You may assume that all images will be of the same size for this assignment, more information on the image format will be placed on the class website
- The next three parameters control the subsampling of your Y U and V spaces respectively. For sake of simplicity, we will follow the convention that subsampling occurs only along the width dimension and not the height. Each of these parameters can take on values from *1* to *n* for some *n*, *1* suggesting no sub sampling and n suggesting a sub sampling by *n*
- The last parameter *Q* controls quantization of your R, G and B values. It is a number that specifies how many different values each channel can have

To invoke your program we will compile it and run it at the command line as

YourProgram.exe C:/myDir/myImage.rgb Y U V Q

where *Y U V Q* are the parameters as described above. Example inputs are shown below and this should give you a fair idea about what your input parameters do and how your program will be tested.

1. *YourProgram.exe image1.rgb 1 1 1 256*

There are 256 values (8 bits) per R G and B, and no subsampling in the Y, U or V -> which implies that the output is the same as the input

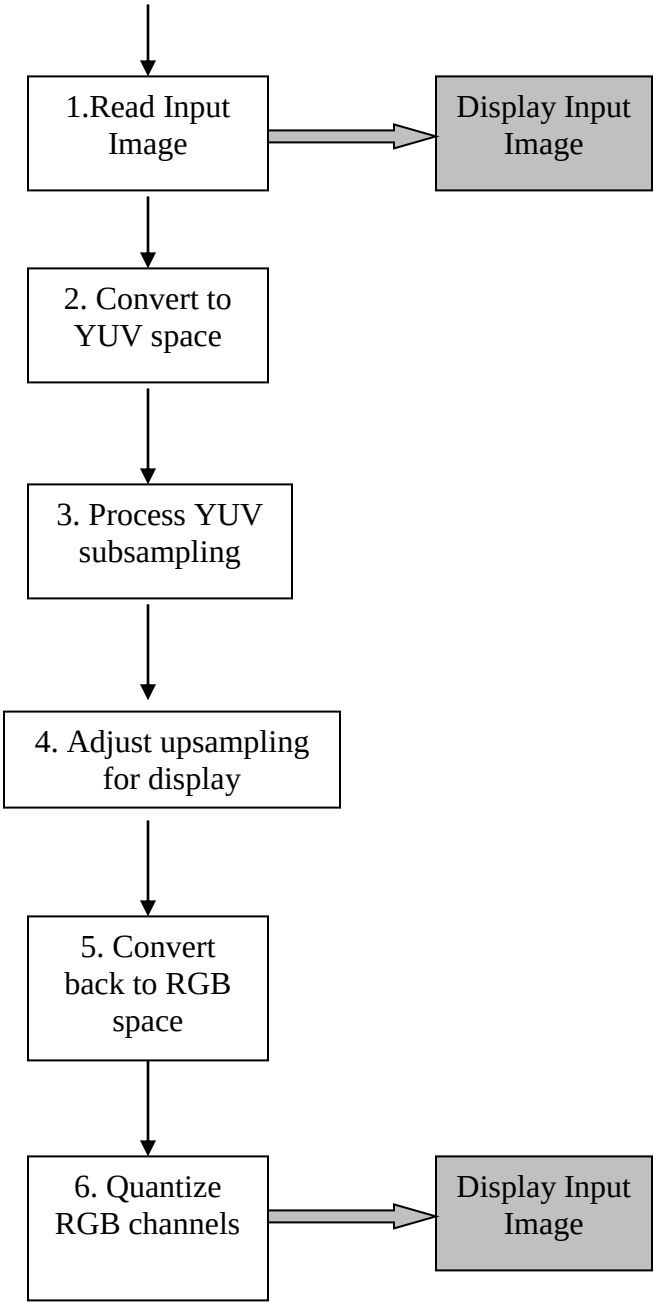
2. *YourProgram.exe image1.rgb 1 1 1 64*

There are 64 values (6 bits) per R G and B and no subsampling in Y, U or V.

3. *YourProgram.exe image1.rgb 1 2 2 256*

There are 256 values (8bits) per R, G and B (no additional color quantization), but the U and V channels are subsampled by 2. No subsampling in the Y channels.

Now for the details - In order the display an image on a display device, the normal choice is an RGB representation. This is what the format of the input image is. However, for YUV processing reasons, you will have to convert the image in YUV space, process your subsampling and reconvert it back to RGB space to show the output to display. Here is the dataflow pipeline that illustrates all the steps.



This code is already provided to you, if you choose to make use of it

The RGB to YUV with the conversion matrix is given below

Sub sample Y U and V separately according to the input parameters

Adjust sample values. Although samples are lost, prior to conversion to RGB all the channels have to of the same size

Apply the inverse matrix to get the RGB data

Quantize the color channels according to the input parameter and display

Conversion of RGB to YUV
Given R, G and B values the conversion from RGB to YUV is given by

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.274 & -0.322 \\ 0.211 & -0.523 & 0.312` \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

Remember that if RGB channels are represented by n bits each, then the YUV channels are also represented by the same number of bits.
RGB values are positive, but YUV can take negative values!

Conversion of YUV to RGB
Given R, G and B values the conversion from RGB to YUV is given by

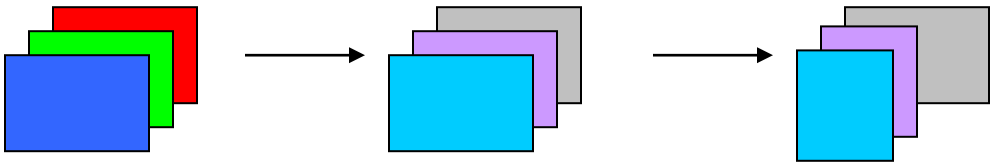
$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.000 & 0.956 & 0.621 \\ 1.000 & -0.272 & -0.647 \\ 1.000 & -1.106 & 1.703 \end{bmatrix} \begin{bmatrix} Y \\ U \\ V \end{bmatrix}$$

Remember that if YUV channels are represented by n bits each, then the RGB channels are also represented by the same number of bits.
YUV channel can have negative values, but RGB is always positive!

Sub sampling of YUV & processing
Sub sampling, as you know will reduce the number of samples for a channel.
Eg for the input parameters

YourProgram.exe image1.rgb 1 2 2 256

In this example, the YUV image is not subsampled in Y, but by 2 in U and by 2 in V resulting in



When converting back to the RGB space, all the YUV channels have to be of the same size. However the sampling throws away samples, which have to be filled in appropriately by average the neighborhood values. For example, for the above case a local image area would look like

$Y_{11}U_{11}V_{11}$	Y_{12}	$Y_{13}U_{13}V_{13}$	Y_{14}	. . . . .	line 1
$Y_{21}U_{21}V_{21}$	Y_{22}	$Y_{23}U_{23}V_{23}$	Y_{24}	. . . . .	line 2
$Y_{31}U_{31}V_{31}$	Y_{32}	$Y_{33}U_{33}V_{33}$	Y_{34}	. . . . .	line 3
$Y_{41}U_{41}V_{41}$	Y_{42}	$Y_{43}U_{43}V_{43}$	Y_{44}	. . . . .	line 4

The missing values may be filled in using filters. The example below shows simple averaging of neighborhood scanline values, but the quality of your output can change depending on your choice of filters.

$U_{12} = (U_{11} + U_{13})/2$ $\quad\quad$ $V_{12} = (V_{11} + V_{13})/2$
$U_{14} = (U_{13} + U_{15})/2$ $\quad\quad$ $V_{14} = (V_{13} + V_{15})/2$
…. And so on, to get

$Y_{11}U_{11}V_{11}$	$Y_{12}U_{12}V_{12}$	$Y_{13}U_{13}V_{13}$	$Y_{14}U_{14}V_{14}$	line 1
$Y_{21}U_{21}V_{21}$	$Y_{22}U_{22}V_{22}$	$Y_{23}U_{23}V_{23}$	$Y_{24}U_{24}V_{24}$	line 2
$Y_{31}U_{31}V_{31}$	$Y_{32}U_{32}V_{32}$	$Y_{33}U_{33}V_{33}$	$Y_{34}U_{34}V_{34}$	line 3
$Y_{41}U_{41}V_{41}$	$Y_{42}U_{42}V_{42}$	$Y_{43}U_{43}V_{43}$	$Y_{44}U_{44}V_{44}$	line 4

Note the samples that you use to fill in values will change depending on the subsampling parameters. The YUV components can now be converted to RGB space.

RGB Color Quantization.
Assume that the quantization levels are uniformly distributed. Initially we have 8 bits per pixel per channel to start with. So the Q input value to your program can take on values from the range 255 – 0. For instance

- Q=256, implies 8 bits per channel or 256 possible values for each channel, so the output number of bits is same as input.
- Q=8, implies 3 bits per channel or 8 possible values which may be 0, 31, 63, 95, 127, 159, 191, 223,
- Q=64, implies 6 bits per channel or 64 possible values which may be 0, 3, 7, 11, … 239, 243, 247, 251
- Remember Q may not necessarily be a power of 2.

So design your quantization function accordingly.

What should you submit ?

- Your source code, and your project file or makefile, if any, using the submit program. ***Please do not submit any binaries***. We will compile your program and execute our tests accordingly. The tests will include varying values of Y, U, V and Q.
- Along with the program, also submit an electronic document (word, pdf, pagemaker etc format) using the submit program that answers the fore-mentioned analysis questions. You may use any (or all) input images for this analysis.

Analysis Question 1: (20 points)

Subsampling obviously degrades the image quality. Here you are going to analyze this degradation. Can you formulate a way to quantify this degradation as an error number or error percentage? Explain how you are measuring your error.

Compute this error value for output images where each input Y, U, V is individually varying while the other two remain constant. (Assume Q is constant at 256). Plot your error metric values for each.

- Keep U and V constant at 1, and vary Y to compute different image outputs. For each compute your error metric and finally plot all these values to get a *distortion* curve.
- Repeat the above process to compute your *distortion* curve keeping Y and V constant at 1 and vary U.
- Repeat the above process to compute your *distortion* curve keeping Y and U constant at 1 and vary V.

Do you see any patterns in the curves?

What conclusion(s) can you draw from your analysis?

Analysis Question 2: (30 points)

Keeping Y constant at 1 (no subsampling), as you increase the value of U and V, the outputs generated by filtering in values for the missing Us and Vs start to produce visible artifacts.

- What artifacts do you see and explain what causes them?
- Suggest a method to rectify or minimize these artifacts explaining your algorithm.
- Write a program to implement your idea and show results. You don't have to submit your program code, but attach outputs for the given image data sets showing the images before your change and with the artifacts alongside images after your change which minimized the artifacts.

Analysis Question 3: (40 points)

For an input of Y=1, U=1,V=1 and Q=2, you have no sampling in Y U and V but have 2 values per color channel resulting in 8 distinct colors. However, the method suggested in the programming section produces colors which might not be the best causing the output image colors to look very different compared to the original colors. Research into and implement an algorithm to better the final quantized colors.

- Explain your algorithm or method of choice. Remember your algorithm needs to work for all possible values of Q, not just Q=2.
- Modify your code using your idea to quantize your output to the best colors choice of colors. Again, you don't have to submit your program code, but attach outputs for the given image examples showing the images before your change and after your implementation showing better colors. Do this for different values of Q, Q = 2, 3, 4