CS 576 Spring 2009– Assignment 1
Instructor: Parag Havaldar

Assigned on 01/23/2017
Solutions due on 02/13/2017 (before noon)
Late Policy: None
There are four parts to this assignment

Part 1 - Written Part: (15 points)

Q.1 Suppose a camera has 450 lines per frame, 520 pixels per line, and 25 Hz frame rate. The color sub sampling scheme is 4:2:0, and the pixel aspect ratio is 16:9. The camera uses interlaced scanning, and each sample of Y, Cr, Cb is quantized with 8 bits

- What is the bit-rate produced by the camera? *(1 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 *(2 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. *(3 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? *(4 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 *(4 points)*

Part 2–Program to Display Video (25 points)

The first part of your assignment deals with reading a video file in a raw native format and display it at a defined frame rate. The parameters to your display program will be the filename, width, height, frame rate to displah. Ideally we would like to use standard video frame sizes (eg HD, CIF) but due to memory restrictions, processing power and display monitor sizes, we will use half the standard sizes in both dimensions for this assignment. For instance, HD video is **1920 x 1080**, but we'll work **with 960 x 540**. Similarly, CIF video is **352x288**, but we will work with **176 x 144**. Thereby example invocations to your display program will look like

YourDisplay.exe C:/myDir/input_vid.rgb 176 144 10
// Read the input file havig frame width=176, frame height=144, and play the video at 10
// frames per second. This frame size corresponds to the standard CIF video format
YourDisplay.exe C:/myDir/input_vid.rgb 960 540 20
// Read the input file havig frame width=960, frame height=540, and play the video at 10
// frames per second. This frame size corresponds to the standard CIF video format

For the purpose of this assignment, all videos given to you will have the same two sizes mentioned above but may varying number of frames. Additionally -

- The videos have been natively created at 10 frames/second, which means when played back at 10 fps they seem to play at the right speed. We may choose to play them at different speeds though to evaluate your synchronization.
- The videos are 3 channel RGB color with each pixel having 3 bytes and 1 byte per channel. These parameters will not change for your assignment, although we might view your video at different frame rates

Make sure your video plays in a loop when it reaches the end respecting the input frame rate parameter.

.

Data Format:
The format of your native input video will consist of a list of pixels frame-by-frame, channel-by-channel with every pixel represented as three bytes (or chars), one byte (or char) per channel. To elaborate, the data will be stored as *frames* - frame1, frame2, ….. frame n. Each *frame* is further stored as rows of bytes for each channel - *rrr…bbb…ggg….* And each *r, g* and *b* is a byte long. So, if for our standard definition video we will have an invocation of

YourDisplay.exe C:/myDir/input_vid.rgb 176 144 10

The input file will have 176x144 bytes of *red* of frame1, then 176x144 bytes of *blue* of frame1, then 176x144 bytes of *green* of frame1, followed by 176x144 bytes of *red* of frame2, then 176x144 bytes of *blue* of frame1, then 176x144 bytes of *green* of frame1, followed by frame 3 …. And so on.

We have provided a Microsoft Visual C++ project as well as a java program to read in an image in the format specified and display it. Instructions on how to compile/invoke this program are also given. This source has been provided as a reference for students who may not know how to read and display an image. You are free to use this as a start, or write your own in C/C++ or any other programming language such as Java. The only restriction is that we should be able to easily compile and run your program on the university computing platforms. For assignments we don't allow interpretative environments such as matlab.

Part 3 – Program to convert Video Formats (40 points)

This part of the assignment will help you gain a practical understanding of issues that relate to image sampling, aliasing effects, image aspect ratios and pixel aspect ratios. In this assignment you will be given as input a video in one format (standard definition or high definition) and you will be asked to convert it into the other respective format. These problems have very practical applications today – for example,

1. High definition 16:9 signals are now a broadcast standard, but there are devices that might still have a different viewable format eg 4:3 standard definition formats. The image size has to be reduced and the image(s) need to be down sampled, which will result in aliasing artifacts that need to be corrected with filters. Also, if the pixel aspect ratio is not maintained, it could result in additional distortion.
2. Consumers today buy high definition television sets, monitors and projectors that can view high definition signals properly. However, a lot of content, especially older content, which is broadcast by TV/cable networks, is still in standard definition NTSC formats. Here the image size has to be increased (up sampled) again causing artifacts and also pixel distortions due to aspect changes. Good filters and image synthesis techniques are typically required to production quality images here.

Accordingly, this part your assignment should take as input either an image or a video stream in one format and produce an output in the other format. Input to your program will be 4 parameters where:

- The first parameter is the name of the input media file.
- The next parameter is the name of the output file
- The third parameter controls the operation of the program. SD2HD should run the code for task 1 above to convert an input video of size 176 x 144 to 960 x 540, HD2SD should run the code for task 2 to convert an input video of size 960 x 540 to 176 x 144.
- The last parameter controls the whether or not filtering for anti aliasing should be turned on. A value of 0 indicates this should be off, and a value of 1 indicates it should be switched on.

Here are example command line invocations of your program.

For task 1

```
YourProgram.exe C:/myDir/input_vid.rgb  C:/myDir/output_vid.rgb  SD2HD 0
YourProgram.exe C:/myDir/input_vid.rgb  C:/myDir/output_vid.rgb  SD2HD 1
```

For task 2

```
YourProgram.exe C:/myDir/input_vid.rgb  C:/myDir/output_vid.rgb  HD2SD 0
YourProgram.exe C:/myDir/input_vid.rgb  C:/myDir/output_vid.rgb  HD2SD 1
```

You can view your output videos by using your display program you completed in part1.

Some thoughts:

1. While converting an HD video stream (960x540) to a SD stream (176x144), the conversion process reduces the number of samples (effectively subsampling), which may cause aliasing. Your conversion should incorporate anti aliasing based on a simple averaging filter. The option to turn on (or off) anti aliasing should be controlled by the last input argument (see above). The averaging filter works by replacing each pixels value in the source image by the average of a local neighborhood area (3x3 or 5x5) and then subsampling to create the target image.
2. While converting from a SD video stream (176x144) to a HD video stream (960x540), the conversion process here increases the number of samples, and as a result pixels may get duplicated in the target image. This will cause multiple blocky areas with perhaps visible discontinuitie. To minimize the effect of this, perform a similar averaging operation depending on the input anti aliasing parameter.

Part 4 –Analysis Questions on video conversion (20 points)

*For this part, you may need to write a new program, but you don't need to submit any code, just a report (word, pdf etc) that explains your answer with accompanying image results. Use the image frames from the content we have supplied to illustrate your answers***.**

1. When you convert SD to HD, in addition to aliasing, there is a pixel scaling effect that changes your pixel aspect ratio, as illustrated in class as well. This is disturbing, and not desirable, but unavoidable in these cases. Can you think of any smart operations that you can do to minimize this stretching effect? Write code to create such an output and submit your result showing example before and after output images.
2. Do the same as above but for HD to SD.

Extra credit

3. Converting an SD to HD images is much harder than down sampling to SD from HD. This is because SD to HD effectively involves increasing resolution when it is not there. It is hard to create pixels or just algorithmically imagine them. Now filters do provide some relief to minimize certain problems, but can you think of any methods or processes to create good HD content from SD. Provide sample frames from the video given showing input frame, output frame produced by your

program in part2 of this assignment and any enhanced outputs that you have generated beyond the scope of what was asked in part 2. Be sure to include any references or pointers that you read or used in your analysis.

What should you submit ?

- For the programming parts 2 and 3 please submit your source code ONLY, and your project file or makefile, if any. Please use instructions from the TA to submit using the den course page. *Please do not submit any binaries or media files, this causes problems*. We will compile your program and execute our tests accordingly.
- For written parts 1 and 4, submit an electronic document (word, pdf, pagemaker etc format) clearly showing detailed working.