

# CS 576 Fall 2018– Assignment 1

## Instructor: Parag Havaladar

**Assigned on Mon Aug 27 2018**

**Solutions due on Mon Sept 17 2018 (before 12 noon)**

**Total Marks – 100 points**

### Written Part: (10 points)

Each question has marks displayed

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? (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)

### Programming – Spatial/Temporal Resampling and Aliasing

This programming assignment will help you gain a practical understanding of Resampling and Filtering in the spatial and temporal domain. It consists of *two parts*, the first one aimed to increase your understanding of sampling issues in the spatial domain and the second one deals with sampling issues in the temporal domain.

## Part 1 –Spatial Resampling and Aliasing (40 points)

In your program you will need to display two images side by side –

1. Your original image displayed on the left – This is an image of size 512x512 that you will create based on the criteria explained below.
2. Your processed output image displayed on the right – This image is the output of your algorithms on the original image to create a resampled image depending on parameters explained below.

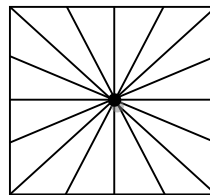
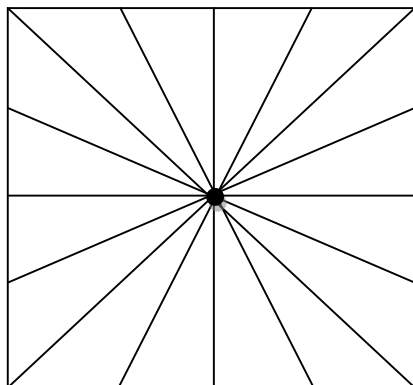
Input to your program will take three parameters where

- The first parameter  $n$  is the number of lines to create an image with radial pattern of  $n$  black lines starting from the center of the image towards the boundaries. The image has a white background. Each consecutive line is separated by  $360/n$  degrees. The idea here is by increasing  $n$ , you can increase the frequency content in an image.
- The second parameter  $s$  will be scaling value that scales the input image down by a factor. This is a floating point number eg  $s=2.0$  will scale the image down to 256x256. Note  $s$  need not be a complete integer.
- The third parameter will be a boolean value (0 or 1) suggesting whether or not you want to deal with aliasing. A 0 signifies do nothing (aliasing will remain in your output) – which means you need copy the direct mapped value from input to output. A value 1 signifies that anti-aliasing should be performed – which means that instead of the direct mapped value you need to copy a low pass filtered value to the output. See lecture for more explanation of this in class.

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

*My part1.exe 16 2.0 0*

This will create original 512x512 image to the left with 8 lines radially as shown, and the output is scaled down by 2.0 creating an image of size 256x256 as shown.



*Similarly, My part1.exe 360 2.0 0*

This will create an image with a more dense pattern with each line separated by one degree, ultimately scaled down to half its size.

### Analysis Questions for part 1 – submit document with written part

1. Let's try an experiment where  $s$  (scale factor) remains constant and  $n$  (number of lines) is allowed to vary. Comment on your results as to how aliasing is affected by using various constant values of  $s$  for changing  $n$  in each case. You may attach results, plot charts etc. to quantify your results.
2. Let's try another experiment, this time keep  $n$  (number of lines) constant and varying  $s$  (scale factor). Comment on your results as to how aliasing is affected by using various constant values of  $n$  for changing  $s$ . You may attach results, plot charts etc. to quantify your results.

## Part 2 –Temporal Aliasing (50 points)

In your program you will need to display two videos side by side –

1. Your original video displayed on the left – This is video of size 512x512 that you will create based on the criteria explained below. This is radial pattern just as in the previous case, but it is also rotating clockwise at a certain specified speed. When create and update your image at the respective times, it should simulate a rotating wheel.
2. Your processed output video displayed on the right – The output video is also of size 512x512 but in order to simulate temporal aliasing effects it will be given an fps rate of display, which means your output will be updated at specific times.

Input to your program will take four parameters where

- The first parameter  $n$  is the number of lines to create an image with radial pattern of  $n$  black lines starting from the center of the image towards the boundaries. The image has a white background. Each consecutive line is separated by  $360/n$  degrees. The idea here is by increasing  $n$ , you can increase the frequency content in an image.
- The second parameter  $s$  will be a speed of rotations in terms of rotations per second. This is a floating point number eg  $s=2.0$  indicates that the wheel is making two full rotations in a second,  $s=7.5$  indicates that the wheel is making seven and a half rotations in a second. Remember this is the original input video signal with a very high display rate.
- The third parameter will be an *fps* value suggesting that not all frames of the input video are displayed, but only a specific frames per second are displayed.

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

*My part2.exe 64 4.0 10.0*

In this case, the input video consists of images with 64 lines (as explained in part one), rotating clockwise at 4 revolutions per second (displayed on the left) and the right output is a temporally sampled version displayed at 10.0 frames per second. Here, for a rate of 4.0 rotations per second, the Nyquist factor is 8.0, so any fps above 8.0 should not result in temporal aliasing and the output should be the same as input.

*MyPart2.exe 64 4.0 7.5*

In this case, the input video consists of images with 64 lines (as explained in part one), rotating clockwise at 4 revolutions per second (displayed on the left) and the right output is a temporally sampled version displayed at 7.5 frames per second. Here, for a rate of 4.0 rotations per second, the Nyquist factor is 8.0, so any fps below 8.0 should result in temporal aliasing – manifested by the wheel not rotating the way it should

### **Analysis Questions for part 2 – submit document with written part**

Let's try an experiment where  $s$  (speed of rotation) remains constant and  $fps$  is allowed to vary. Study the value of the  $os$  (observed speed of rotation), especially when there is temporal aliasing.

1. Can you design a formula relating  $s$ ,  $fps$  and  $os$ .

Evaluate if your formula works for certain values of  $s$  and  $fps$ . If  $s = 10$  rotations per second,

2. What is the observed speed  $os$  for an  $fps$  of 25?
3. What is the observed speed  $os$  for an  $fps$  of 16?
4. What is the observed speed  $os$  for an  $fps$  of 10?
5. What is the observed speed  $os$  for an  $fps$  of 8?

### **Part 3 (Optional Extra Credit) (30 points)**

Change part2 of your assignment to take in two additional parameters – one to deal with scaling the temporal signal (just as you did with the static image in part 1) and the second to deal with anti aliasing in the resultant signal in both spatial and temporal domains.

Correspondingly,

- The fourth parameter  $s$  will be a scale factor that scales the input video down by a factor. This is a floating point number eg –  $s=2.0$  will scale the video down to 256x256. Note  $s$  need not be a complete integer. Also if the fourth parameter above is a 1, then you need to perform spatial antialiasing (like part1) along with temporal antialiasing.
- The fifth parameter will be a boolean value (0 or 1) suggesting whether or not you want to deal with aliasing. A 0 signifies do nothing (temporal and spatial aliasing will remain in your output). A value 1 signifies that temporal & spatial anti-aliasing should be performed – you need to design a method to decrease temporal aliasing that shows better output videos as well as incorporate the anti aliasing method for spatial aliasing that you implemented in part 1.

Together with these two parameters you should be able to create scaled videos of your input at different frame rates and simultaneously minimize any aliasing effects due to resampling temporally and spatially.

To invoke your extra credit, we will compile it and run it at the command line as

*MyExtraCredit.exe 64 4.0 7.0 1.0 1*

In this case, the input video consists of images with 64 lines (as explained in part one), rotating clockwise at 4 revolutions per second (displayed on the left) and the right output

is a temporally sampled version displayed at 7 frames per second. Here, for a rate of 4.0 rotations per second, the Nyquist factor is 8.0, will result in temporal aliasing which will have to be anti-aliased. The output size does not change, so there is nothing to do about spatial anti-aliasing.

*MyExtraCredit.exe 64 4.0 7.0 2.0 1*

In this case, the input video consists of images with 64 lines (as explained in part one), rotating clockwise at 4 revolutions per second (displayed on the left) and the right output is a temporally sampled version displayed at 7 frames per second. Here, for a rate of 4.0 rotations per second, the Nyquist factor is 8.0, will result in temporal aliasing which will have to be anti-aliased. The output size is also halved and it will induce spatial aliasing which will have to be anti-aliased.

### **What should you submit ?**

- Your source code ONLY (no data or binaries), your project file or makefile, if any and your analysis questions answered in a pdf or a word document. You should submit your work by making use of the D2L submit process. *Please do not submit any binaries or media files. You will be adversely penalized if you do.* We will compile your program and execute our tests accordingly.