**HW Assignment 1**

**PART I**

**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

a) What is the bit-rate produced by the camera? (1 points)

b) 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)

**Answer**:

No of bits for Y = 4\*8 = 32 bits , for U = 2\*8 = 16bits and V = 0 bits

Average bits per pixel = 48/4 = 12 bits/pixel

Bit rate = pixels per line \* lines per frame \* 12 \* frame rate = 70, 200, 000 = 70.2Mbps

Average bits per pixel = (4\*8+2\*6+0\*6)/4 = 11 [since we use only 6 bits per sample as given]

Therefore for 10 min we have 450\*520\*11\*25\*60\*10 bits = 3.861 \* e10  bits

**Q2** 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).

1. Write down the quantized sequence. (3 points)
2. How many bits do you need to transmit it? (1 points)

**Answer**

1. No of levels = 32 [the length of given list of real numbers]

Therefore, Quantized sequence is given by: Round ((i-(-3.75)/8)) \* 32

Output = [22, 24, 24, 28, 28, 28, 25, 26, 26, 26, 21, 19, 20, 20, 22, 24, 24, 24, 23, 24, 20, 16, 10, 10, 8, 11, 6, 9, 9, 12, 15, 19]

1. Since there are a total of 32 level , no of bits per level is given by [2n  = 32] therefore n = 5

Hence, total bits transmitted = levels \* bits/level = 32\*5 = 160 bits

**Q3** 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.

1. 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)
2. 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)

**Answer**

1. Diameter = .4244m Therefore, distance travelled by the wheel in one full rotation = 2\*Pi\*r = 3.13\*.4244 = 1.333 m/rotation

Speed = 36 Km/h = (36\*1000) / 60 = 10m/s

Nyquist sampling rate = 2\*7.5 fps = 15fps. Since, Nyquist sampling rate is less than recording fps, therefore there is no Aliasing

Speed of rotation of the wheel = 10/1.333 rotation/second = **7.5 rotation/sec**

1. At 8fps, the wheel cover 7.5 full rotations in 8 frames

Therefore no of degrees turned per frame = 7.5\*360/8 = 337.5 degrees/frame

Therefore the wheel is covering 22.5 degree/frame

22.5 degees/frame \* 8 fps/sec = 180 degrees/second = 180/360 rotations/second = **1/2 rotations/second** but in opposite direction

**PART 4**

**Q 1** a)

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.

**Answer**:

Initial Pixel aspect ratio = (4:3)/(176/144) = 1:1

While converting SD(4:3) to HD(16:9) there is a pixel stretching affect.

In a general case it can be dealt with a technique called **“Pillar Boxing”** of the image, to avoid stretching. The 4:3 to 16:9 conversions can be imagined as 12:9 to 16:9 conversions. Therefore, it’s evident that we lack in content along the width axis and therefore pillar box the image as an alternative to stretching and trying to fit the screen.

Pillar boxing can be achieved by simply **padding** the image sides [in this case left and right] with black pixels.

Padding has been implemented by simply taking the default image and adding **black-pixels** to the image so that when the image is scaled the Pixel aspect ratio(PAR) is maintained.



Figure 1: Resized from 176\*144 to 960\*540 with pillar boxing



Figure 2: Resized from 176\*144 to 960\*540 with distortion(streched)

**b)**

Do the same as above but for HD to SD.

Initial Pixel aspect ratio = (16:9)/(960/540) = 1:1

While converting HD(16:9) to SD(4:3) there is a pixel shrinking/squishing affect.

In a general case it can be dealt with a technique called **“Letterboxing”** of the image, to avoid squishing. Just opposite to the case of Pillarboxing here we add the black pixel above and below the image. Since we are filling in the entire content of the image onto a 4:3 display we are left with empty spaces at the top and bottom which are again filled by padding.



Figure 3: resized image to 176:144 using letterboxing



Figure 4: resized image to 176:144 with distortion (squished)

Moreover, other techniques such as **content aware scaling** may be used where a part of the image can be marked as “protected” from scaling. Then when the image is scaled the, its protects crucial parts from any stretching or squishing affect but scales the unprotected parts.

**Extra Credit Question**.

References:

<https://en.wikipedia.org/wiki/Bilinear_interpolation>

<https://docs.oracle.com/javase/7/docs/api/java/awt/RenderingHints.html>

Further smoothing of the image can be obtained by methods such as bilinear interpolation/ bi-cubic interpolation. Bi-linear interpolation refers to interpolating both in the x and the y direction. Given a 2 image the pixel at some point (x,y) is approximated using 4 nearby pixels. points *Q*11 = (*x*1, *y*1), *Q*12 = (*x*1, *y*2), *Q*21 = (*x*2, *y*1), and *Q*22 = (*x*2, *y*2), Qij indicated the know value of some function f at those 4 points.

The goal is to estimate the value of the function f at new position (x,y) by interpolating in both the directions

Therefore f(x,y) is a an element wise multiplication of bij and Q(ij)  [note both b and Q are vectors]

Below, are presented 3 images where image1 uses bilinear interpolation, image2 uses a simple 3X3 averaging filter and image3 is simply an up-scaled image with no pre-processing. The change in image quality is clearly visible.



Figure 5: up-scaling the image using Bilinear Interpolation



Figure 6: upscaling from SD to HD using a 3\*3 filter Anti-Aliasing is ON



Figure 7: up-scaling from SD to HD using no filter – Anti Aliasing is OFF