CS1115 Lab 10 (November 7, 2013)

Completing the lab is very important, but your work is not graded and it is not submitted. If you finish before the hour is over, then you can leave early or you can work on the "Finished Early" problems. If you do not finish the problems before the end of class, then be sure to ask enough questions so that you can complete the exercises in the next day or two on your own.

1 Image File Processing

Download `CornellIthaca.jpg` (or any other jpeg) from the website. Let $m \geq 2$ and $n \geq 2$ be integers. Imagine cutting the picture into m rows (of approximately equal height) and n columns (of approximately equal width) thereby producing mn "picture tiles". Randomly reassemble these tiles into another m-by-n array of picture tiles. Store the resulting image in a jpeg file called `Scrambled.jpg` and display it.

2 Sound File Processing

Download the file `noCry.wav` from the syllabus page.

(a) Create a file `noCry1.wav` which when played, plays the noCry soundbite, pauses for 1 second, and then plays the noCry soundbite again.

(b) Create a file `noCry2.wav` that plays the noCry soundbite over and over and over again for exactly 30 seconds.

(c) Suppose `[y,r] = wavread('noCry')`. It turns out that `y` has length $4m$ where $m = 19451$. Write a script that plays $y(1 : m)$ at rate r and then plays $y(m + 1 : 2 * m)$ at rate $(1.2)r$ and then plays $y(2 * m + 1 : 3 * m)$ at rate $(1.2)^2 r$ and then plays $y(3 * m + 1 : 4 * m)$ at rate $(1.2)^3 r$. There should be no pause between the four segments.

(d) Can you write a function `FasterAndFaster(F,n,factor)` that plays an "ever faster" version of the `.wav` file named by `F`? It should break the sound vector into n (approximately) equal segments and play the kth segment at rate $(factor)^{k-1} r$ where r is the original sampling rate. (In the preceding example, $n = 4$ and $factor = 1.2$.

3 Finished Early

Download a color jpeg file of your choice and produce three plots that nicely display vectors `nRed(1:256)`, `nGreen(1:256)`, and `nBlue(1:256)` where `nRed(i)`, is the number of pixels in the image whose red value is $i-1$, `nGreen(i)`, is the number of pixels in the image whose green value is $i - 1$, `nBlue(i)` is the number of pixels in the image whose blue value is $i - 1$. Use `subplot` so that the three graphics can be handily compared

Please delete your files from the computer before you leave the lab!