Computer Vision I: Python Exercise / project #1 (total 10 points)
Due September 23, 11:59pm

This is a small project aiming at verifying some properties of natural image statistics that we discussed in Chapter 2. Four natural images (two from urban and two from countryside scenes) are attached as examples, but you should take two sets of images on your own: **Set A**: Natural Images; **Set B**: Non-natural Images. For comparison, synthesize a uniform noise image, i.e. each pixel is drawn independently from a uniform number in $[0, 31]$, defined it as **Set C**.

What to hand in: Submit a report. Your score will be based on the quality of your results and the analysis (diagnostics of issues and comparisons) of these results in your report.

Problem 1 (High kurtosis and scale invariance, 4 points). For computational considerations, first convert an image to grey level and re-scale the intensity to [0,31] i.e. 32 grey levels. Convolve the images with a gradient filter $\nabla_x I$, i.e. the intensity difference between two adjacent (horizontally or vertically). You can either pick any single image for the following steps or accumulate the histograms (average) over the images in Sets A, B, and C and compare the differences between the three sets.

1. Plot the histogram $H(z)$ for the difference against the horizontal axis $z \in [-31, +31]$. Then do a log-plot $\log H(z)$. [Some bins will be zero, you can assign ϵ for such bins in the log-plot].

2. Compute the mean, variance, and kurtosis for this histogram [Report the numeric numbers in your report].

3. Fit this histogram to a Generalized Gaussian distribution $e^{|z/\sigma|^\gamma}$ and plot the fitted-curves super-imposed against the histogram. What is the value of γ in the fitted generalized Gaussian?

4. Plot the Gaussian distribution using the mean and the variance in step (2), and super-impose this plot with the plots in step (1) above (i.e. plot the Gaussian and its log plot, this is easy to do in python with matplotlib).

5. Down-sample your image by a 2×2 average (or simply sub-sample) the image. Plot the histogram and log histogram, and impose with the plots in step 1, to compare the difference. Repeat this down-sampling process 2-3 times.

Problem 2. (Verify the 1/f power law observation in natural images in Set A, 3 points).

Do an FFT (Fast Fourier Transform) on the grey image I which returns a Fourier image $\hat{I}(\xi, \eta)$ which is complex number matrix indexed by (ξ, η) for its horizontal and vertical frequencies. Compute the amplitude (modulus) of each complex number $A(\xi, \eta) = |\hat{I}(\xi, \eta)|$. Denote the frequency $f = \sqrt{\xi^2 + \eta^2}$, and transfer to a polar coordinate, and we calculate the total Fourier power $A^2(f)$ for each frequency (i.e. you need to discretize f, and calculate the $A^2(f)$ averaged over the ring for each f, stop f when the circle hits the boundary of the Fourier image).

1. Plot $\log A(f)$ against $\log f$. This should be close to a straight line for each image. Plot the curves for the 4 images in one figure for comparison. [Hint: First check the magnitude spectrum $\log |\hat{I}(\xi, \eta)|$, $|\hat{I}(0, 0)|$ typically is the largest component of the spectrum. The dc component is usually moved to the center of frequency rectangle.]

2. Compute the integration (summation in discrete case) of $S(f_0) = \int_\Omega A^2(\xi, \eta) d\xi d\eta$ over the domain
$$\Omega(f_0) = \{(\xi, \eta) : f_0 \leq \sqrt{\xi^2 + \eta^2} \leq 2f_0\}$$
Plot $S(f_0)$ over f_0, the plot should fit to a horizontal line (with fluctuation) as $S(f_0)$ is supposed to be a constant over f_0.

Problem 3. (A 2D scale invariant world, 3 points). Suppose we simulate a toy 2D world where the images consist of only 1D line segments. In an image, a line segment is represented by its center (x_i, y_i), orientation θ_i and length r_i. The line segments are independently distributed with uniform probability for their centers and orientations. The length follows a probability $p(r) \propto 1/r^3$, i.e. a cubic power law. You may control the density of line by a Poisson distribution. That is, in each unit area, the number of line segments has a certain constant mean. [Hint: How to sample r from $p(r)$? Calculate the Cumulative Distribution function of $p(r)$, then draw a random number in [0,1].]

1. Simulate 1 image I_1 of size 1024×1024 pixels with a total N lines. (You need to record all the N lines whose centers are within a range of the image, truncate long lines and hide (discard) lines shorter than a pixel.)

2. Simulate 2 new images I_2 and I_3 of size 512×512 and 256×256 pixels respectively. I_2 and I_3 are down-sampled version of I_1 and are generated by shortening the N line segments in I_1 by 50% and 25% respectively (discard lines shorter than 1).

3. Crop 2 image patches of size 128×128 pixels randomly from each of the three images I_1, I_2, I_3 respectively. Plot these six images [draw the line segments in black on white background].

If you did it right [Please try !], the 6 images must look the same (i.e. you should not be able to tell what scale the 6 images are cropped from). So this 2D world is scale-invariant.