Computer Vision I: Python Exercise / project #1 (total 10 points)

Due October 8, 2024, 2 pm

[Ulyana Koida] [2400931382]

This is a small project designed to confirm certain characteristics of natural image statistics, as previously discussed in Chapter 2. Among the materials, you will find three distinct image sets:

1. Set A consists of four natural images, including two captured in urban settings and two in countryside scenes.

2. Set B comprises two man-made images, specifically paintings by Vincent van Gogh.

3. Set C encompasses a random noise image.

Python Library: To complete the project, please install the required dependencies provided in the *environment.yml* file (including cv2, PIL, numpy, scipy, matplotlib, tqdm). You are also welcome to utilize any libraries of your choice, **but please report them in your report (for autograder)! Again, report any customized library in the report (do not go too crazy as this will add a significant burden to TAs).**

What to hand in: Please submit both a formal report and the accompanying code. For the report, kindly provide a PDF version. You may opt to compose a new report or complete the designated sections within this document, as can be started by simply loading the tex file to Overleaf. Your score will be based on the quality of **your results**, **the analysis** (diagnostics of issues and comparisons) of these results in your report, and your **code implementation**.

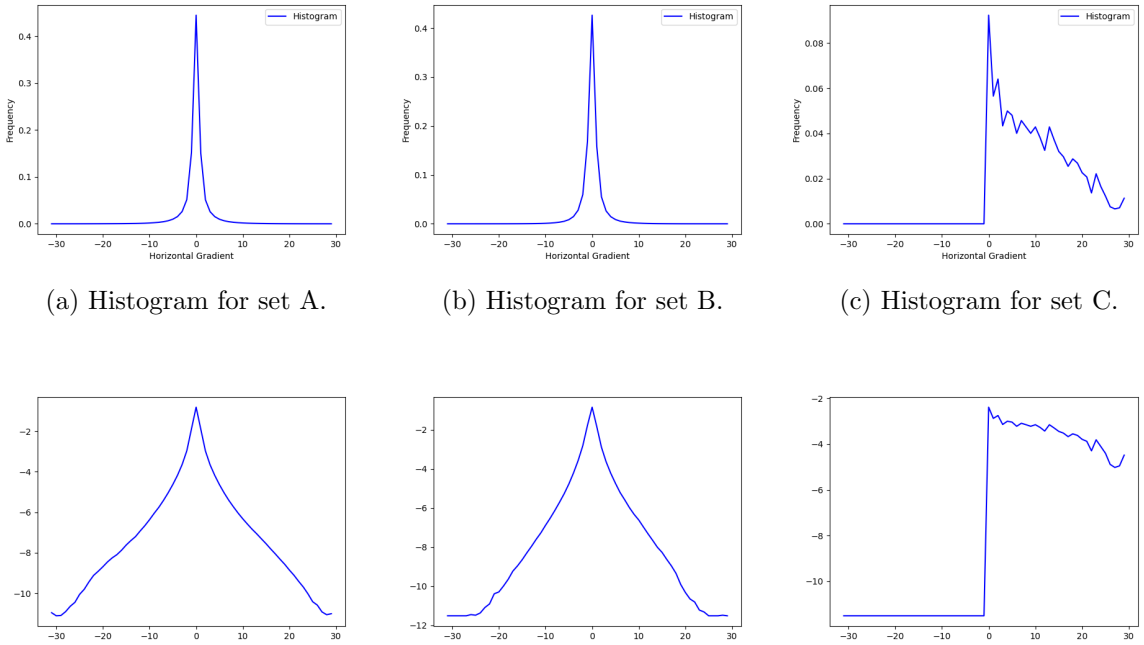
Notice: Hint code is provided, please **don't change** the name of the functions for automatic scoring, but feel free to add new functions.

Problem 1 (High kurtosis and scale invariance, 4 points). *Please read through this question description before you start. For this problem, please consider 3 sets: set A, set B, set C.*

For the sake of computational efficiency,, convert the images to grayscale and subsequently rescale the intensity levels to fall within the range [0, 31], i.e. 32 grey levels. Please note that this rescaling operation is not necessary for the random image selected from set C. Then convolve the images with a gradient filter denoted as $\nabla_x I$, which signifies the intensity difference between two adjacent pixels along the horizontal axis. Accumulate the histograms by computing the averages over all images within each respective set. To illustrate, the process of accumulating the histogram for "set A" entails the aggregation of histograms from all four natural images found within "set A". Subsequently, complete the following steps for all 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].

   Answer: The resulting histograms for set A, B and C are shown in Figure 1.



(a) Histogram for set A. (b) Histogram for set B. (c) Histogram for set C.

(d) Log histogram for set A. (e) Log histogram for set B. (f) Log histogram for set C.

Figure 1: Histogram and log-histogram $H(z)$ for the difference against the horizontal axis $z \in [-31, +31]$ in sets A, B and C.

As could be seen, the histograms for set A and set B are highly peaked, meaning that the histogram mainly accumulates one type of value (close to zero, speaking of a smaller gradient), and meaning that the horizontal gradient (measure of how different each pixel is) is rather small overall, and the set has similar structures within, with few outlying values. The case for C, however, is different, mainly due to the nature of the image: since it is a collection of black and white pixels, the difference between each will be rather big, leading the the result in the image. Considering that the rescaling to [-31, 31] was not implemented for set C images either, the result we get is expected.

A similar conclusion can be drawn for the log histogram of all three sets as for the non-log histogram.

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

   Answer: The data is in the following table, listed in the order of: mean, variance and kurtosis.

Set A	Set B	Set C
-0.000829025	0.000619147	123.052246
6.72702980	4.63183116	13838.091020
15.867897	11.62217179	-1.97809456

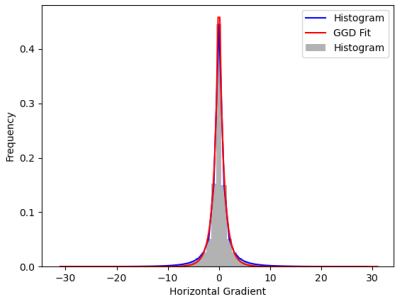
   The values of the mean accurately showcase the distribution of the histogram; the variance is consistent with the histogram results as well: smaller values for sets A and B since the histogram results show dense data around several values and little outliers, and a very large value for C, which shows how the data is not largely centered around any value, instead having many outliers.

   As for the value of kurtosis, the positive values for sets A and B show that both distributions are peaked; comparing to the normal distribution, which has kurtosis of 3, these sets have more outliers. The negative value of kurtosis for set C means the distribution is not peaked, rather a flatter one, which is consistent with the graphs and the patterns of data in C.

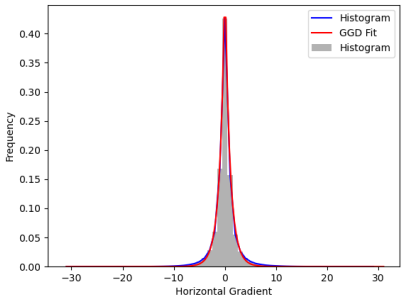
3. Fit this histogram to a Generalized Gaussian distribution $e^{|z/\sigma|^\gamma}$ and plot the fitted-curves super-imposed against the histogram. What is the value of γ in the fitted generalized Gaussian?

   Answer:

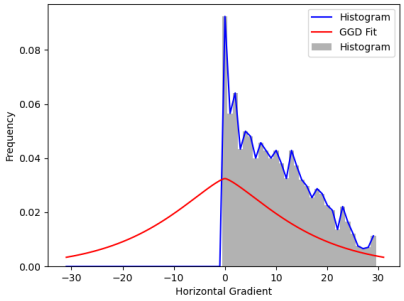
The histograms for sets A and B fit the curve well, while the histogram for set C does not. The results are expected given the histograms' nature and characteristics described above.



(a) GGD for set A. $\gamma = 0.82$



(b) GGD for set B. $\gamma = 0.90$



(c) GGD for set C. $\gamma = 1.31$

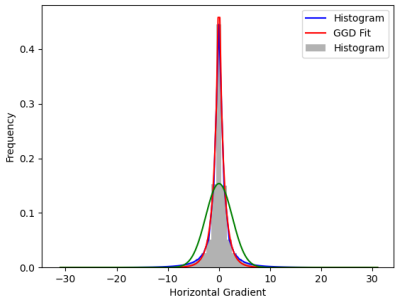
Figure 2: Fitted GGD plot for set A, B, C.

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).

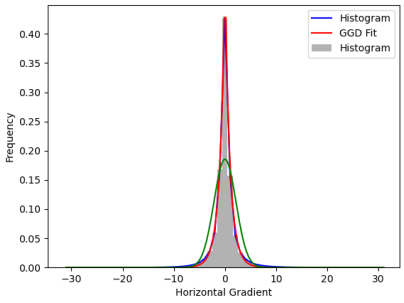
Answer:

The Gaussian distribution, plotted using the mean and the variance calculated earlier does not fit the histogram as well as the GGD curve in the previous step. The reason for that is the lack of adjustable parameters: while the GGD curve, while fitting, is optimizing for γ and σ, the Gaussian is being calculated with already pre-determined parameters. It provides an approximation, but due to this lack of optimization leads to results worse than that of the GGD.

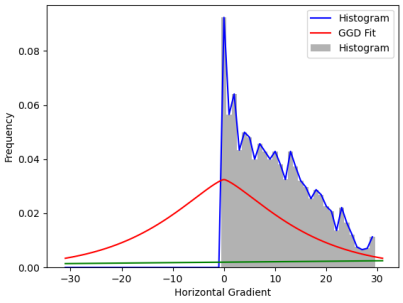
This case is even more obvious in the log-plots, where the curve only loosely fits the histogram.



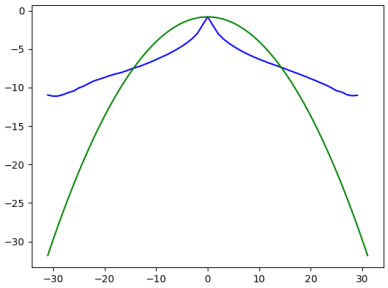
(a) Plot containing the histogram, GGD and Gaussian distribution for set A.



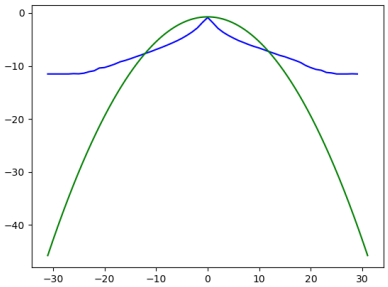
(b) Plot containing the histogram, GGD and Gaussian distribution for set B.



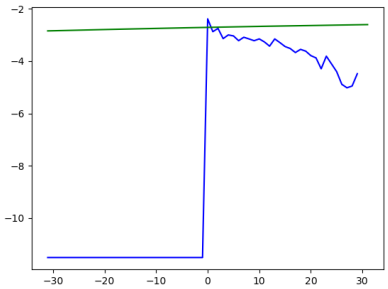
(c) Plot containing the histogram, GGD and Gaussian distribution for set B.



(d) Log-plot of the histogram and Gaussian distribution for set A.



(e) Log-plot of the histogram and Gaussian distribution for set A.



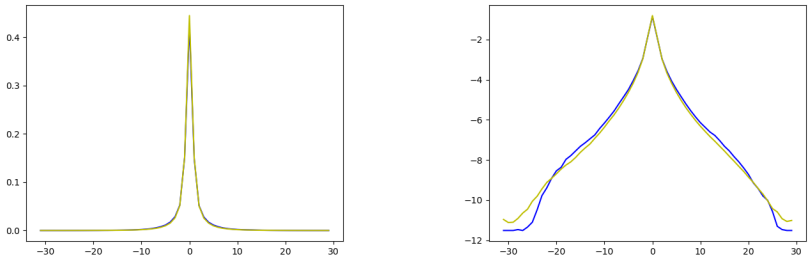
(f) Log-plot of the histogram and Gaussian distribution for set A.

Figure 3: Gaussian distribution plotted using mean and variance.

5. Down-sample your images 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.

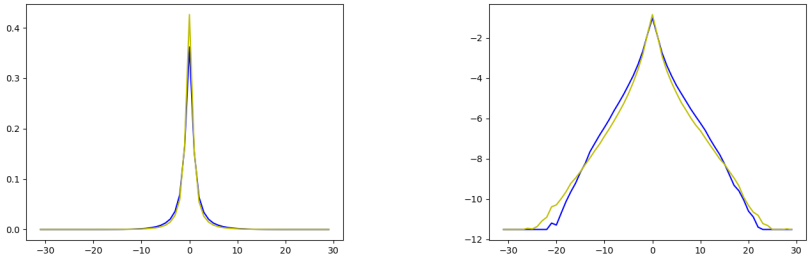
Answer:

The plots for the down-sampling results of each set and retry are shown below, the yellow being the initial histogram and the blue the down-sampled version:



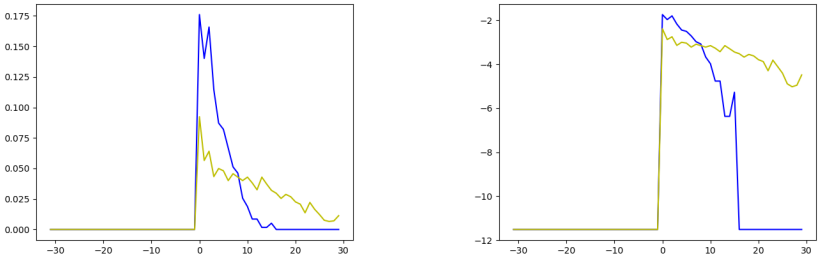
(a) Histogram. (b) Log histogram.

Figure 4: Downsampling Set A by 2x2.



(a) Histogram. (b) Log histogram.

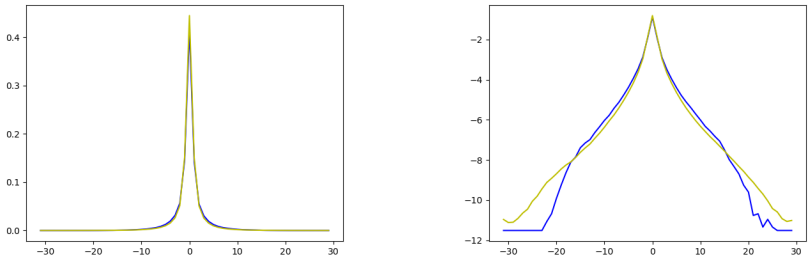
Figure 5: Downsampling Set B by 2x2.



(a) Histogram. (b) Log histogram.

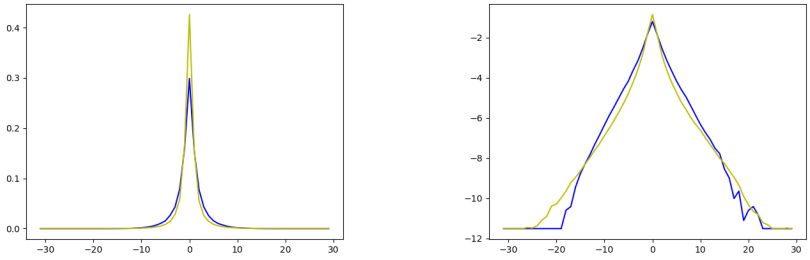
Figure 6: Downsampling Set C by 2x2.

As seen from these figures, the differences in histograms for sets A and B are not



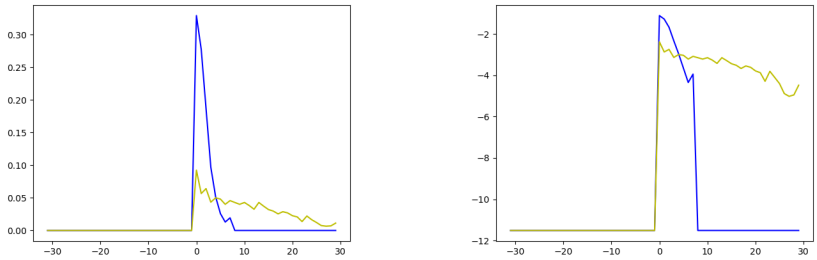
(a) Histogram. (b) Log histogram.

Figure 7: Downsampling Set A by 4x4.



(a) Histogram. (b) Log histogram.

Figure 8: Downsampling Set B by 4x4.



(a) Histogram. (b) Log histogram.

Figure 9: Downsampling Set C by 4x4.

that significant. After down-sampling, the histogram becomes less wide, slightly more squeezed towards the center, but does not change the distribution drastically. The reason for that is down-sampling the image only reduces it's high-frequency features, keeping the low frequency features and with them the general structure of the image and the resulting histogram.

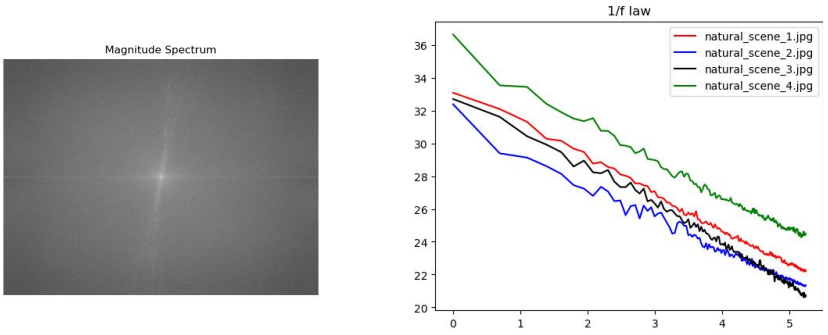
As for set C, since there initially was a lot more variance within the set, down-sampling might lead to a reduction of features more significant to the image's structure, leading to the result seen in figures 6 and 9, where the down-sampled histogram spikes more. As such, averaging or down-sampling a set like C, with a lot of variance and different frequencies, does not keep the structure of the histogram and the image as intact.

Problem 2. (Verify the 1/f power law observation in natural images, 3 points). *Please read through this question description before you start. For this problem, please only consider set A.*

Perform a Fast Fourier Transform (FFT) on the grayscale image denoted as I. This operation will yield a Fourier image denoted as $\hat{I}(\xi, \eta)$, which is a complex-number matrix indexed by horizontal and vertical frequencies (ξ, η). Subsequently, compute the amplitude (modulus) of each complex number, denoted as $A(\xi, \eta)$, and it can be calculated as follows: $A(\xi, \eta) = |\hat{I}(\xi, \eta)|$. Denote the frequency $f = \sqrt{\xi^2 + \eta^2}$, and to facilitate further analysis, convert the data to polar coordinates. In this coordinate system, calculate the total Fourier power, denoted as $A^2(f)$, for each frequency. To achieve this, discretize the values of f and compute $A^2(f)$ averaged over the respective ring corresponding to each value of f. It is important to terminate the process when the circle reaches 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 the frequency rectangle.]

   Answer: The magnitude spectrum image shows at the center the dc component, and the image spectral structure. The plot $\log A(f)$ against $\log f$, as expected for all images, forms a nearly straight line, verifying the 1/f power low observation in natural images. As such, we have confirmed that high frequency parts of the image correspond to low power and vice versa.



(a) Magnitude Spectrum plot. (b) $\log A(f)$ against $\log f$.

Figure 10: Plot of $\log A(f)$ against $\log f$ and of the magnitude spectrum $\log |\hat{I}(\xi, \eta)|$.

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.

Answer: For this part of the question, i did not manage to get completely straight lines for the images, most likely due to calculation or understanding errors. Using code, I iterated over the frequency domain of half the maximum radius to assure that the calculations would not be going out of bounds, and summed the squares of all values that fit the conditions. I suspect the error might lie in mapping the frequency domain onto the result of the sum, or on the initial parameters of the frequency domain. However, here I could not explore what was the final issue.

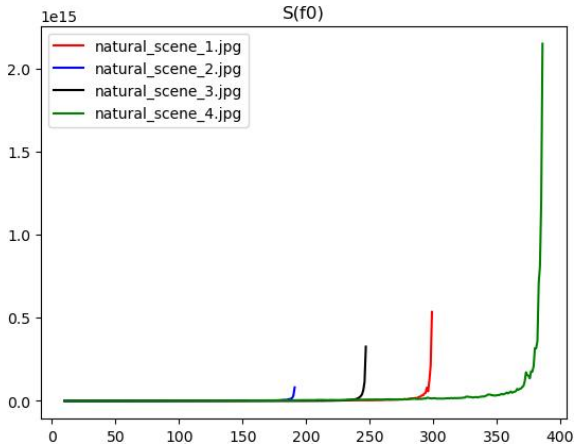


Figure 11: $S(f_0)$ over f_0 for four images.

The code for this part of the asssignment is as follows:

```
110     # TODO: Add your code here
111     basic_f = 1
112     max_r = min(centerx,centery)
113     f0 = np.arange(0, int(max_r/2), basic_f) ## since the domain
114     S_f0 = np.zeros_like(f0)
115     for each in f0:
116         lower_bound = each
117         upper_bound = 2*each
118         for f_val in range(max_r):
119             if (f_val >= lower_bound) & (f_val <= upper_bound):
120                 # pdb.set_trace()
121                 S_f0[each] = np.sum(A[f_val]**2)
122     return S_f0, f0
```

Figure 12: Python code for calculating $S(f_0)$.

Problem 3. (A 2D scale invariant world, 3 points). *Please read through this question description before you start.*

Let's consider the simulation of a 2D world in which the images consist of only 1D line segments. Each line segment in an image can be characterized by its staring point (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 of the line segments follows a probability distribution denoted as $p(r)$, which is proportional to $1/r^3$, representing a cubic power-law distribution. [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 truncate long lines and hide (discard) lines shorter than a pixel.)

   Answer:

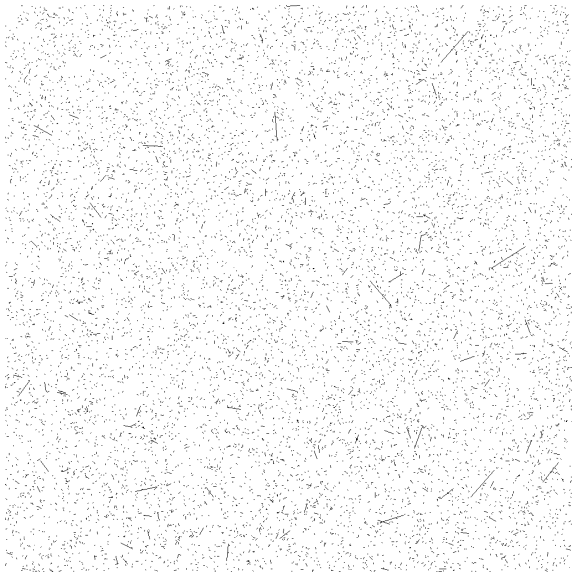
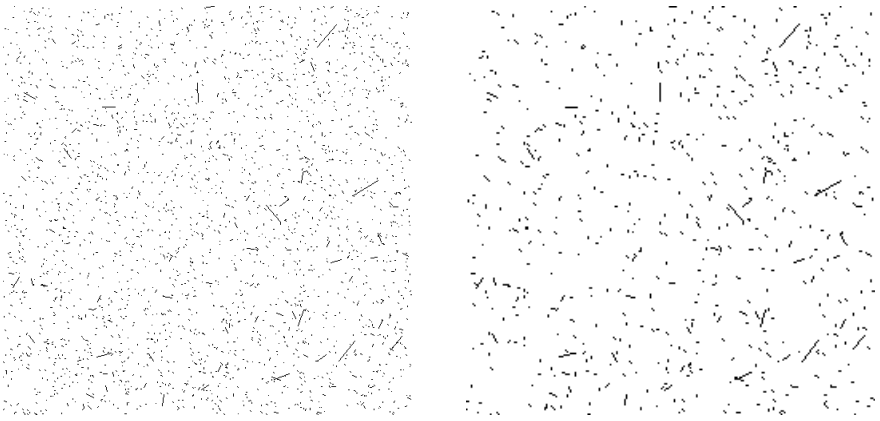


Figure 13: Simulated 1024x1024 image.

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).

   Answer: In order to simulate these images correctly, one of the important details is deleting the lines and points that have length less than a pixel; otherwise, the smaller-sized images will not be represented in the same way as the 1024x1024 one, and the scale invariance will not hold. The results for these two images are in Figure 14.



(a) Image generated by shortening the N line segments in I_1 by 50%.

(b) Image generated by shortening the N line segments in I_1 by 25%.

Figure 14: Simulated pictures of 512x512 and 256x256 sizes.

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].

   Answer: As could be seen from Figure 15, it is impossible to tell which scale the 6 images were cropped from, which proves that the constructed 2D world is indeed scale-invariant.

Figure 15: Cropped patches in comparison. From left to right: two images from 1024x1024, 512x512 and 256x256 images.

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). As a result, this 2D world is scale-invariant.