Assignment 2

Instructions

- Assume that the operations are to be done on single channel Grayscale images unless specified otherwise.
- You are advised not to alter the input `img` variable while applying the filters.
- Plagiarism is strictly not tolerated. Any act of copying will lead to a straight ZERO in the entire Assignment or even more.
- Code should be vectorized as usual —> 20% weightage is given for that
- Deadline- 8th October

Refer to the adjacent table to find the name of the image file.

IMG1	testPattern.jpg
IMG2	~~noise1.png~~ salt.png
IMG3	~~noise2.png~~ pepper.png
IMG4	~~noise3.png~~ salt_and_pepper.png
IMG5	blurry_spiral.png
IMG6	coins.png
IMG7	tiger.jfif

1 Convolution

1. Write a function `conv2D(img, kernel)` to convolve a 2D image with a 2D kernel of size `k x k` (Handle padding so that the resulting image is of the same size as input).
2. Apply a Mean Filter with `k=5` on `IMG1` by writing a new function `meanFilter()` that takes in, a 2D image, kernel size, generates the appropriate kernel and filters the image using `conv2D()`
3. Comment on how the time taken for the operation changes with kernel size `k`, and image dimensions `h` , `w`
4. Apply Gaussian Filter with `k=5` on `IMG1` using `conv2D()` and display the results by trying different values of σ (standard deviation) and corresponding kernels.

Comment on how these results compare with the mean filter's results as $\sigma \to \infty$?

2 Need for Speed

After setting very hard DP problems for AOS Midsem, Lokesh takes a programming tutorial about prefix sum array and Sankalp introduced Venn Diagrams. Tharun, your junior, who attended the session modified his meanFilter() such that the operating time is proportional to the kernel size k.

1. Write a new function `speedyMeanFilter()` that gives the same result as `meanFilter()` such that the operating time is independent of the kernel size (asymptotically) and challenge Tharun.
2. Apply the Mean Filter on `IMG1` using these two functions and plot the run time v/s k

3 Salt and Pepper

Salt and pepper, humble yet mighty, are the gatekeepers of flavor, transforming mere ingredients into culinary wonders, remove the salt and pepper noise from the image to reveal the unadulterated taste waiting to be severed.

1. Identify the type of noise added to `IMG2` and apply the appropriate filters to remove the noise.
2. Identify the type of noise added to `IMG3` and apply the appropriate filters to remove the noise.
3. Identify the type of noise added to `IMG4` and apply the appropriate filters to remove the noise.

4 Blurry? Don't Worry

During their batch trip, Revanth took many pictures. Among all the photos, he liked `IMG5` *the most, but it had a small problem, it was a bit blurry. His friend Kiran advised him to take the Digital Image Processing course earlier but Revanth ignored his* *advice; now he wants you to fix the photo soon so that he can set it as the profile picture.*

1. Implement a function `sharpen(img, A, k)` to perform Un-sharp Masking/ High-boost Filtering on an Image as per required kernel size and the amplifying factor A.
2. Display the results after using `sharpen()` on `IMG5` with different values of `A = [1, 2, 4, 9]`
3. Do the same by using a Gaussian Filter for Blurring in the process instead of Mean. Write your observations.

5 Counting Coins

In `IMG6`, detect edges using the following filters using a 3 x 3 kernel:

1. Prewitt 2. Sobel 3. Laplacian

- Plot the x-gradient, y-gradient for cases 1 & 2
- Plot the gradient magnitude for case 3
- Comment on the results

6 Hurry! Make it Blurry

In a dense, mysterious jungle, a talented wildlife photographer captures a stunning image of a majestic tiger hiding amidst the lush foliage. However, the image turns out to be slightly noisy due to challenging lighting conditions. Help the photographer to enhance the image by removing noise yet retaining the features.

1. Make a function `bilateralFilter( img , k ,` σ_s `,` σ_r `)` that does Bilateral Filtering on an image with the standard deviations of σ_s for the Spatial and σ_r for Intensity Range Gaussians.
2. Apply `bilaterFilter()` on `IMG7` with different values of k , σ_s and σ_r and show how they affect the outcome by changing one parameter at a time.
3. Comment on the difference between the results of bilateral and gaussian filters.