In this lab, you will implement both Kanade-Tomasi (minimum eigenvalue) and Harris algorithms to detect corner features in grayscale images.

Important Note: You should complete the lab until the end of the lab hours and submit all your codes to SUCourse as a single zip file. Deadline for in-lab code submission to SUCourse is **15:30**.

Things to do:

Your functions must be as generic as possible, i.e., don't make any assumptions about the size, the type and the colors of the images. Your functions must convert the image to grayscale if it is colored and you must employ the row and column numbers of the images as variables.

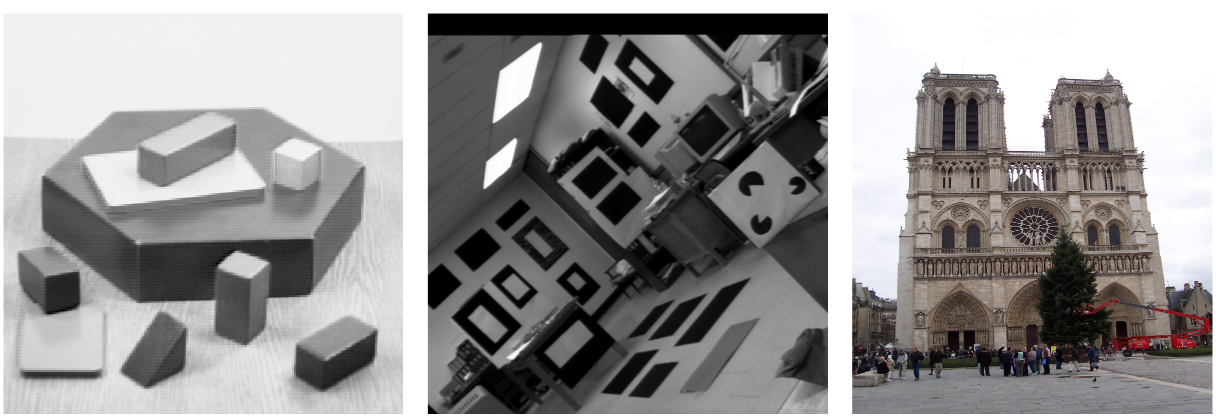
Kanade-Tomasi Algorithm: In order to detect the corners in a grayscale image I by employing Kanade-Tomasi algorithm, you can apply the following steps:

1. Apply Gaussian smoothing filter to an image using built-in `imgaussfilt` function.
2. Compute image gradients I_x and I_y using built-in `imgradientxy` function. Input to this function is a grayscale image I and outputs are image gradients I_x and I_y computed using Sobel gradient operator.
3. Compute the corner matrix H for each pixel in a window P of size $(2k+1) \times (2k+1)$ as follows

$$H = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{1}$$

4. Compute the eigenvalues, λ_1 and λ_2, of H.
5. Use threshold on eigenvalues to detect corners such that if $\min(\lambda_1, \lambda_2) > Threshold$ the pixel coordinates are added to the corner list.

Now write a function which takes an image I and a threshold as inputs and utilize "**Kanade-Tomasi Corner Detection Algorithm**" to return the detected corner points of the images given in the figure below. Your function name should be "lab3ktcorners.m".



(a) Blocks (b) Lab (c) Notre Dame

Figure 1: Input images for corner detection algorithms.

Harris Algorithm: In order to detect the corners in a grayscale image by employing Harris operator, you can apply the following steps:

1. Apply Gaussian smoothing filter to an image using built-in `imgaussfilt` function.
2. Compute image gradients I_x and I_y using built-in `imgradientxy` function.
3. Compute the corner matrix H for each pixel in a window P of size $(2k+1) \times (2k+1)$ as follows

$$H = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{2}$$

4. Compute $f = det(H)/trace(H)$ where the trace is the sum of the diagonals, i.e., $trace(H) = h11 + h22$.
5. If $f > Threshold$ add the pixel coordinates to corner list.

Now write a function which takes an image and a threshold as inputs and utilize "**Harris Corner Detection Algorithm**" to return the detected corner points of the sample images from the Figure 1. Your function name should be "lab3Harriscorners.m".

Harris corner detector can be implemented by using a measure of corner response R instead of f in the step 4 of the algorithm defined above. R is computed as follows:

$$R = det(H) - \kappa(trace(H))^2 \tag{3}$$

where $\kappa \in [0.04, 0.06]$ is an empirical constant.

Now modify "lab3Harriscorners.m" function by replacing f with R as a measure of corner response. Save modified function as "lab3Harriscorners2.m", and apply it to the images given in the Figure 1. Compare performance of both f and R measures.

Post Lab

Post lab reports must include brief explanations of each method that you used in this lab. Provide resulting images by utilizing all these methods and discuss your results. Discuss the performance of Kanade-Tomasi and Harris corner detectors on different images. Provide resulting images by utilizing all these functions with different parameters such as window size and threshold values. Comment on the performance differences of the implemented algorithms.

Deadline for post lab report submission to SUCourse: **07 November 2022, 23:55**.