In this lab, you will implement both minimum eigenvalue (Kanade-Tomasi) 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 **17:00**.

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.

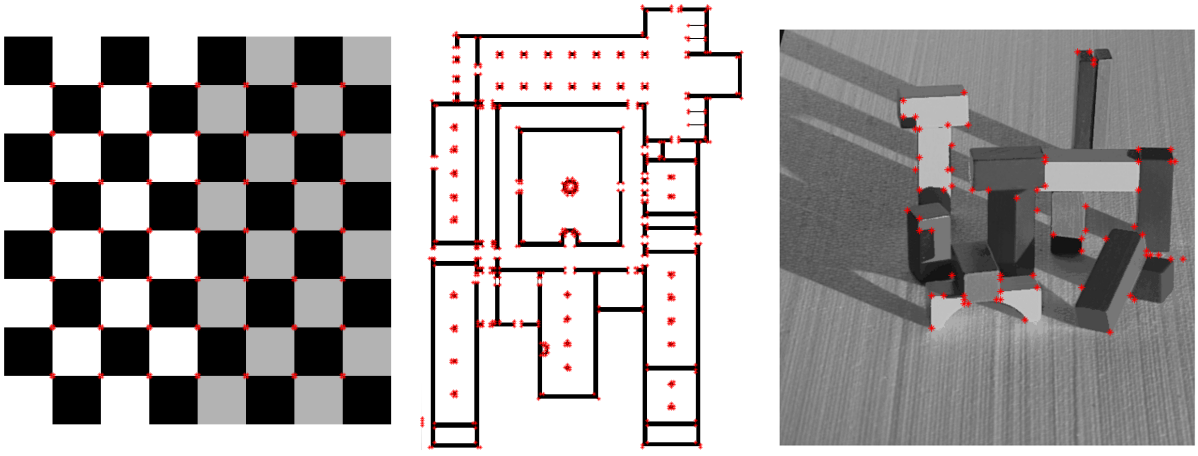
- **Minimum Eigenvalue Algorithm:** In order to detect the corners in a grayscale image by employing Kanade-Tomasi algorithm, you can apply the following steps:
  - Compute the gradients G_x and G_y (Hint: You can use `[Gx,Gy] = imgradientxy(I);`)
  - Create H matrix of each pixel in a window as follows

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

    where I_x and I_y are the image gradients of a window along x and y directions, respectively.
  - Compute the eigenvalues λ_1 and λ_2 of H.
  - If $\min(\lambda_1, \lambda_2)$ >Threshold, add the pixel coordinates to corner list.

Now write a function which takes an image and a threshold as inputs and utilize "**Kanade-Tomasi Corner Detection Algorithm**" to return the detected corner points. Your function name should be "lab4ktcorners.m".

Your results should look as follows:



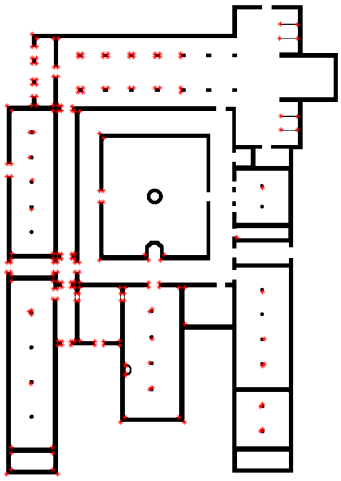
- **Harris Algorithm:** In order to detect the corners in a grayscale image by employing Harris operator, you can apply the following steps:
  - Compute the gradients G_x and G_y (Hint: You can use `[Gx,Gy] = imgradientxy(I);`)
  - Create H matrix of each pixel in a window as follows

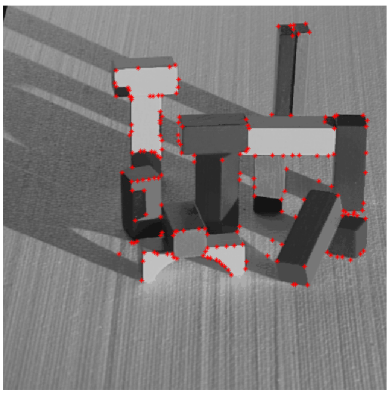
$$H = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix} \tag{2}$$

    where I_x and I_y are the image gradients of a window along x and y directions, respectively.
  - Compute $f = det(H)/trace(H)$ where the trace is the sum of the diagonals, i.e., $trace(H) = h11 + h22$.
  - 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. Your function name should be "lab4Harriscorners.m".

Your results should look as follows:





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 corner detector on different images. Comment on performance differences between Kanade-Tomasi and Harris corner operators. Moreover, redo Harris corner implementation using $R = det(H) - k(trace(H))^2$ measure of corner response and compare the performance of both f and R measures.

Deadline for post lab report submission to SUCourse: **09 November 2021, 23:55**.