TTK4255: Robotic Vision

Homework 2: Feature detection

Originally created by Simen Haugo, and modified by Mau Hing Yip.

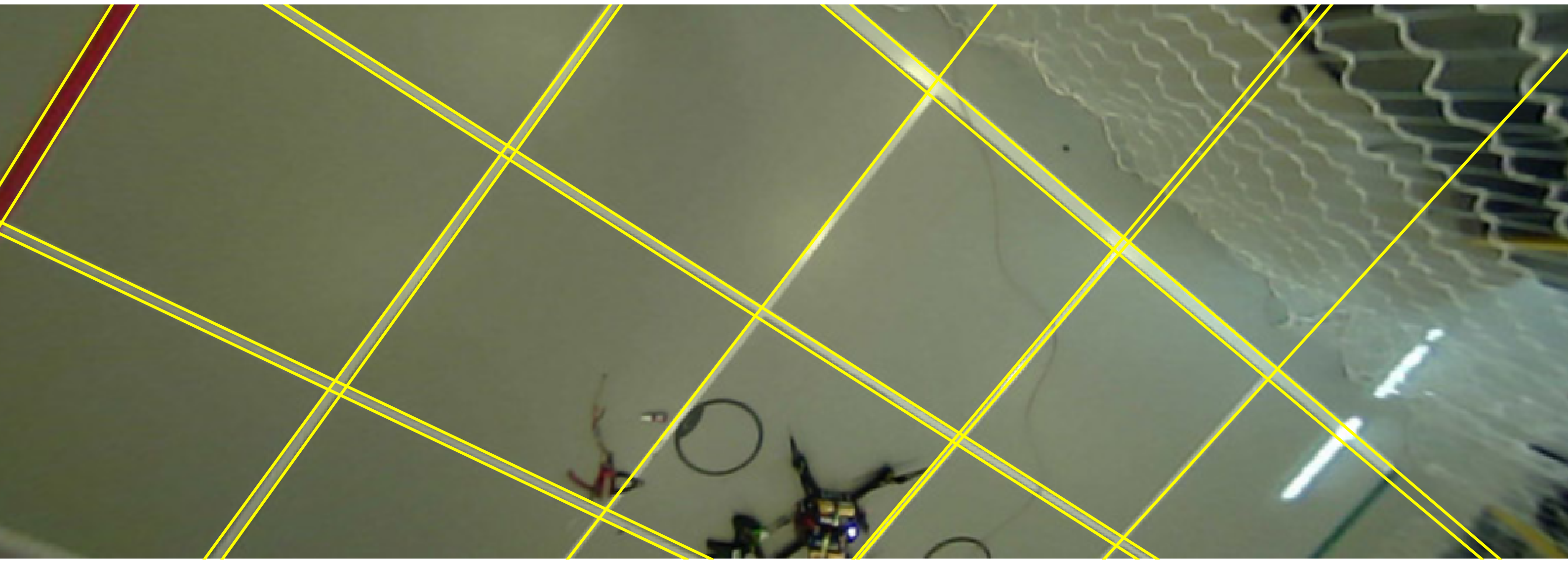

Figure 1: Sample output from a line detection algorithm you will implement.

Instructions

First make sure to read `about.pdf` in the "Course work" page on Blackboard. To get your assignment approved, you need to complete any **35%**. Upload the requested answers and figures as a single PDF. You may collaborate with other students and submit the same report, but you still need to upload individually on Blackboard. Please write your collaborators' names on your report's front page. If you want detailed feedback, please indicate so on the front page.

About the assignment

In this assignment you will learn how to detect some simple image features. Features play an important role in establishing point correspondences, which are used in state-of-the-art methods and systems for pose estimation and 3D reconstruction. Detecting simple features, such as edges or lines, is also often part of algorithms for detecting artificial markers.

Relevant reading

The Harris corner detector is presented in Szeliski §7.1.1. The Hough transform is described in §7.4.2. Note that the Hough transform described in §7.4.2 modifies the original algorithm to use edge directions, such that each involved point only votes for a single line.

Feature description, matching and tracking (§7.1.2–§7.1.4) are topics that we will return to later, and are not required reading for this assignment.

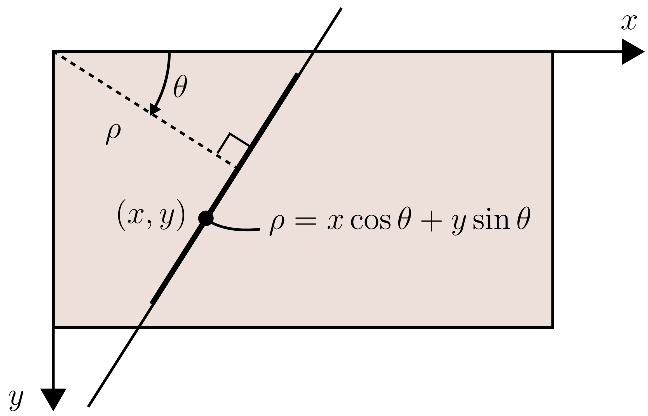

Figure 2: In normal form a line is represented by its angle θ and distance to the origin ρ. Any point (x, y) on the line satisfies the equation $\rho = x\cos\theta + y\sin\theta$. Note that the y-axis here is pointing downward to match the axes convention of images, so positive θ indicates clockwise rotation.

Part 1 Theory questions (30%)

Task 1.1: (10%) Consider a square image of size L, with $(x, y) \in [0, L] \times [0, L]$, and let lines be represented in normal form (ρ, θ). Keeping the angle fixed at each value below, what is the range of possible ρ values attainable by a *visible* line (a line that intersects the image) with that angle?

(a) $\theta = 0°$ (b) $\theta = 180°$ (c) $\theta = 45°$ (d) $\theta = -45°$

Task 1.2: (10%) Several corner detectors use the auto-correlation matrix of the image gradients in a local region around each point $\mathbf{x}$ of consideration. It can be computed efficiently using convolution:

$$\mathbf{A}(\mathbf{x}) = \begin{bmatrix} A_1(\mathbf{x}) & A_2(\mathbf{x}) \\ A_2(\mathbf{x}) & A_3(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} ((I_x^2) * w)(\mathbf{x}) & ((I_x I_y) * w)(\mathbf{x}) \\ ((I_x I_y) * w)(\mathbf{x}) & ((I_y^2) * w)(\mathbf{x}) \end{bmatrix}, \tag{1}$$

where w is a weighting function, and $(I_x(\mathbf{x}), I_y(\mathbf{x})) = \nabla I(\mathbf{x})$. Szeliski §7.1.1 describes "corner strength" measures based on this matrix. The measure proposed by Harris and Stephens is

$$\lambda_0\lambda_1 - \alpha(\lambda_0 + \lambda_1)^2 = \det(\mathbf{A}) - \alpha \operatorname{trace}(\mathbf{A})^2, \tag{2}$$

where λ_0, λ_1 are the eigenvalues of $\mathbf{A}$, and α is usually 0.06. The Harris-Stephens measure has the advantage that it can be computed without explicitly computing the eigenvalues.

(a) Is the Harris-Stephens measure invariant to additive bias ($I(\mathbf{x}) \to I(\mathbf{x}) + c$)?

(b) Ignoring discretization and instead assuming I is a continuous-domain function, argue that the Harris-Stephens measure is invariant to rotation if w is radially symmetric.

Task 1.3: (10%) The Harris-Stephens measure is not invariant to multiplicative gain ($I(\mathbf{x}) \to cI(\mathbf{x})$). However, does this affect which points are detected as corners if the acceptance threshold is specified *relatively* as a fraction of the largest occurring corner strength?

Part 2 Hough transform (40%)

Here you will implement the line detector in Szeliski §7.4.2 based on the Hough transform. The basic idea is to accumulate votes for the presence of specific lines (ρ, θ) in an accumulator array H. Specifically, an edge with location (x_i, y_i) and orientation θ_i votes for the line (ρ_i, θ_i), where $\rho_i = x_i \cos\theta_i + y_i \sin\theta_i$. The vote is added by incrementing the corresponding cell in the accumulator array. Because arrays can only store values at a finite set of indices, the $\rho\theta$-space must first be quantized into $N_\rho \times N_\theta$ bins, with a pair of ranges $[\rho_{\text{min}}, \rho_{\text{max}}] \times [\theta_{\text{min}}, \theta_{\text{max}}]$ mapping the continuous parameters to indices in H:

$$\text{row} = \text{floor}(N_\rho \cdot \frac{\rho - \rho_{\text{min}}}{\rho_{\text{max}} - \rho_{\text{min}}}) \in [0, N_\rho], \quad \text{column} = \text{floor}(N_\theta \cdot \frac{\theta - \theta_{\text{min}}}{\theta_{\text{max}} - \theta_{\text{min}}}) \in [0, N_\theta]. \tag{3}$$

After accumulating votes from all edges, those elements of H with more votes than their immediate neighbors, and a minimum number of votes, are extracted and interpreted as dominant lines.

The `task2` script provides code to generate the requested figures on a sample image. The basic edge detector from Homework 1 is also provided, which handles the first step of extracting edges.

Task 2.1: (5%) Determine appropriate ranges for θ and ρ based on the input image resolution. Your ranges should ensure that all potentially visible and distinct lines can be represented in the array.

Note: As discussed in Szeliski, using edge directions in the voting gives a line two distinct orientations, depending on the edge direction (whether the image went from dark to bright or vice versa along the line normal). Therefore, the minimal range for θ here is of size 2π. Note also that the ranges provided in the book chapter assume the pixel coordinates have been normalized to the range $[-1, 1]$.

Task 2.2: (25%) Compute the accumulator array as described above and include a figure showing the array for the sample image. Let the horizontal axis of your array represent the angle θ and the vertical axis represent ρ; otherwise the provided code for the figure will be wrong. For now, use a small resolution ($N_\rho = N_\theta = 200$). You should be able to see several bright spots, corresponding to the parameters of the dominant lines in the sample image.

Task 2.3: (10%) Use the provided function `extract_local_maxima` to extract the dominant lines. The function takes a minimum acceptance threshold, specified as a fraction between 0 and 1 of the maximum array value. Set this to 0.2. Convert the row and column indices back to continuous (ρ, θ) quantities and include figures showing the location of local maxima in the accumulator array and the lines drawn back onto the input image. For better results, you can try to increase the accumulator array resolution and possibly adjust the acceptance threshold.

Note: The results in Fig. 1 were obtained by tuning N_ρ, N_θ, and using the LoG-based edge detector with sub-pixel localization described in Szeliski §7.2.1, and modifying `extract_local_maxima` to use a larger neighborhood.

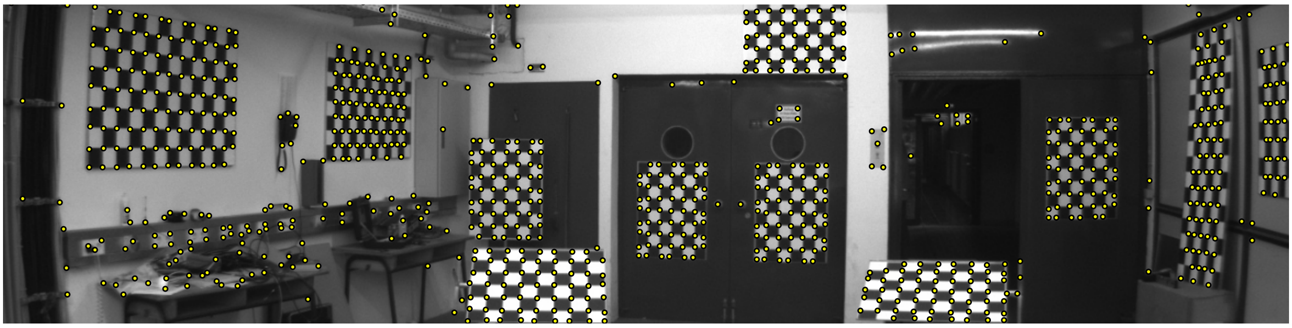

Figure 3: Detected Harris corners in a sample image from (Geiger *et al.*, 2012). The checkerboards are used for camera calibration. Some well-known calibration packages localize the interior vertices of the checkerboard patterns using the Harris corner detector. (Today there are more robust methods).

Part 3 Harris detector (30%)

The Harris corner detector is the corner detector obtained by applying the Harris-Stephens corner strength measure to the auto-correlation matrix in Eq. (1). The detector works by first estimating the image derivatives I_x and I_y. The quantities I_x^2, I_y^2, and $I_x I_y$ are formed at each pixel and are convolved with a weighting function w. The resulting corner strength image is then analyzed for local maxima.

The `task3` script provides code to generate the figures requested in Task 3.1 and 3.4.

Task 3.1: (15%) Compute the Harris-Stephens measure for the sample image `calibration.jpg`. Include a figure showing the resulting corner strength as a grayscale image.

Image derivatives should be computed by convolving with the partial derivatives of a 2-D Gaussian with standard deviation σ_D (the differentiation scale). The weighting function w should be a 2-D Gaussian with standard deviation σ_I (the integration scale). The hand-out code includes a function `derivative_of_gaussian` to compute the image derivatives. Use $\sigma_D = 1, \sigma_I = 3$ and $\alpha = 0.06$.

Task 3.2: (5%) What do negative corner strength values indicate?

Task 3.3: (5%) Why do some of the checkerboards have a weaker response than others?

Task 3.4: (5%) Use the provided function `extract_local_maxima` to extract strong corners. Use an acceptance threshold as 0.001 of the maximum corner strength (note that the function takes a relative threshold). Include a figure showing the extracted corners as a scatter-plot over the sample image.