Introduction to Computer Vision

Lab 3: Edge Detection in Computer Vision

ESIN, UIR — Spring 2026

Ilias TOUGUI

1 Introduction

In the previous lab, we studied frequency-domain filtering and hybrid images. In this lab, we move to **edge detection** and ask a fundamental question:

How can a computer automatically find the edges of objects in an image?

The answer builds directly on everything you have done so far — Gaussian smoothing, convolution, and frequency analysis all play a role. Edge detection is not a single algorithm but a **family of methods**, each making different trade-offs between noise robustness, precision, and complexity.

By the end of this lab, you will be able to:

- Implement and compare three gradient-based detectors: Roberts, Prewitt, and Sobel.
- Implement the Laplacian of Gaussian (LoG) detector using zero-crossings.
- Build a complete Canny edge detector from scratch, step by step.

2 Lab Instructions

2.1 .ipynb Notebook

Open the provided notebook (better on Colab or Kaggle). It is organized into five parts:

1. **Part 1 — Gaussian Smoothing**
   Implement `gaussian_kernel()` and observe how kernel size and σ affect noise suppression before edge detection.

2. **Part 2 — Gradient-Based Edge Detectors**
   Implement `apply_gradient_operator()` and define the Roberts, Prewitt, and Sobel kernels. Compare their gradient magnitudes and noise sensitivity on clean and noisy images.

3. **Part 3 — Laplacian Edge Detector (LoG)**
   Implement `laplacian_edges()` using zero-crossing detection. Observe the trade-off between noise robustness and edge localisation as σ increases.

4. **Part 4 — Canny Edge Detector**
   Implement `non_maximum_suppression()` and `double_thresholding()`, then assemble the full `canny()` pipeline. The `edge_tracking_hysteresis()` function is provided.

5. **Part 5 — Comparison & Reflection**
   Run all five detectors under fair conditions (same Gaussian pre-smoothing), experiment with parameters, and answer the conceptual questions directly in the notebook markdown cells.

Deliverable & Submission Rules

- Submit a **single PDF** on Connect UIR — the completed notebook only, no separate report.
- Your **name and Student ID** must appear in the first cell of the notebook.
- All **written answers** must be filled in the `Your Answer:` fields directly in the notebook.
- Run all cells (`Runtime` → `Run all`) and make sure every output is visible before exporting.
- Convert your `.ipynb` to PDF via https://www.vertopal.com/en/convert/ipynb-to-pdf, then upload the PDF to Connect UIR.
- **Do not send your submission by email. (I won't correct it)**