Numerical Methods for Image Processing: Iterative Solutions for Edge Detection

1 Introduction

Numerical methods play a critical role in computer engineering applications, particularly in image processing. This project focuses on implementing **iterative numerical methods** such as **Jacobi and Gauss-Seidel** to solve the **discrete Laplace equation**, a key component in edge detection and image denoising. Students will use these methods to process images, analyze convergence behavior, and implement solutions using **Python or MATLAB**.

2 Mathematical Formulation

In image processing, edge detection can be formulated using the **Laplace equation**:

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0, \tag{1}$$

where $u(x, y)$ represents pixel intensity at location (x, y).

2.1 Finite Difference Approximation

Since images are represented as discrete grids of pixels, we approximate the second-order partial derivatives using **finite difference approximations**:

$$\frac{\partial^2 u}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}, \quad \frac{\partial^2 u}{\partial y^2} \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}. \tag{2}$$

Assuming uniform grid spacing h, the Laplace equation simplifies to:

$$u_{i,j} = \frac{1}{4}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}). \tag{3}$$

This equation can be solved iteratively using Jacobi, Gauss-Seidel, or SOR methods.

2.2 Example: 5x5 Grid Discretization

To help understand finite difference approximations, consider a small 5×5 grid representing pixel intensities:

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & u_{1,1} & u_{1,2} & u_{1,3} & 0 \\ 0 & u_{2,1} & u_{2,2} & u_{2,3} & 0 \\ 0 & u_{3,1} & u_{3,2} & u_{3,3} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

Applying the Laplace equation, each interior pixel satisfies:

$$u_{i,j} = \frac{1}{4}(u_{i+1,j} + u_{i-1,j} + u_{i,j+1} + u_{i,j-1}). \tag{5}$$

Boundary pixels (zeros) remain fixed due to Dirichlet boundary conditions.

3 Numerical Solution Using Iterative Methods

Students will implement the following **iterative methods**:

- **Jacobi Method**: Updates each pixel value using only previous iteration values.
- **Gauss-Seidel Method**: Uses updated values within each iteration to accelerate convergence.

Students will compare the efficiency and stability of these methods in processing images.

4 Programming Implementation

Students will implement the numerical methods in **Python or MATLAB** using the following steps:

1. Load and convert an input **grayscale image** (use the standard **Lena** test image) into a numerical matrix.
2. Discretize the image using a **grid size of 256x256 pixels**.
3. Construct the **Laplace equation system** for edge detection.
4. Implement **Jacobi, and Gauss-Seidel methods** as functions.
5. Apply the following **boundary conditions**:
    - The image borders are set to zero (Dirichlet boundary conditions).
    - Interior pixel values are iteratively updated.

6. Use a **stopping criterion** based on the relative error:

$$\frac{||u^{(k+1)} - u^{(k)}||}{||u^{(k)}||} < 10^{-6} \tag{6}$$

7. Run simulations and visualize results using **Matplotlib (Python) or MATLAB plotting functions**.

8. Compare the **convergence rates and computational efficiency** of each method.

9. Submit a progress report midway through the project.

5 Academic Integrity and Malpractices to Avoid

Students are expected to maintain **academic integrity** while working on this project. The following malpractices should be avoided:

- **Plagiarism:** All work, including code and written reports, must be original. Proper citations must be included if external sources are referenced.
- **Unauthorized Collaboration:** Groups should work independently. Sharing code or reports between groups is strictly prohibited.
- **Fabrication of Results:** All results must be generated from actual numerical simulations. Modifying or falsifying data is not acceptable.
- **Lack of Contribution:** Every group member should contribute meaningfully to the project. Free-riding will be discouraged through peer evaluations.
- **Copying from AI or Online Solutions:** While students may use online resources for learning, directly copying solutions without understanding or modification is not permitted.

6 Project Timeline

This project will be conducted over a total of **two to three weeks**, with an intensive **Project Week** for the majority of the work, followed by additional time for finalization and submission.

- **Project Week:** Groups work on model formulation, numerical methods, and initial results.
- **Week 2:** Refinement of numerical implementations, visualization, and analysis of results.
- **Week 3 (if needed):** Finalization of report, peer review, and submission of all deliverables.

7 Conclusion

This project applies **iterative numerical methods to image processing**, reinforcing concepts in **computational algorithms and engineering applications**. Students gain insights into how different numerical techniques can be used for **edge detection and denoising** in digital images.

Deliverables:

- Code implementation for **Jacobi, and Gauss-Seidel methods**.
- Processed images showing **edge detection and denoising results**.
- Short report discussing findings.
- Midway progress report for project tracking.
- Comparison of **iteration counts and computational time** for each method.

Rein Borkor (PhD)