Project Proposal — LaTeX-OCR: A Rule-Based OCR System for Printed Mathematical Formulas

Members: B13201026 詹淯翔、B13201025 吳冠霆、B13201008 周暉祐

1. Introduction

For mathematics students, LaTeX is the standard tool for writing assignments, reports, and lecture notes. Yet when a formula appears only in a PDF or a scanned textbook, converting it into editable LaTeX often requires tedious manual transcription. Existing OCR tools perform poorly on mathematical expressions or depend on heavy deep-learning models that are unnecessary for simple printed formulas.

This project addresses that practical need by developing a **lightweight, rule-based OCR system** tailored for **single-line printed mathematical expressions**, using only classical computer-vision techniques such as contour analysis, connected components, and template matching. The goal is to reliably extract symbols and reconstruct basic mathematical structures—without relying on machine learning models or large datasets.

2. Objectives

The system assists students by converting printed mathematical expressions into editable LaTeX. It recognizes common symbols and reconstructs structures like superscripts, subscripts, fractions, and radicals using geometric rules. A compact template database ensures the system remains lightweight and easily extensible.

The project targets practical scenarios in coursework: single-line printed formulas with shallow structures found in lecture notes, textbooks, and PDF slides.

3. System Pipeline

1. **Preprocessing (OpenCV)**
    - Grayscale, denoising, adaptive thresholding
    - Edge/Canny detection and contour extraction
    - Purpose: produce clean symbol boundaries
2. **Symbol Segmentation**
    - Extract bounding boxes via contours/connected components
    - Crop symbol images
    - Baseline estimation to classify main-line, superscript, subscript
3. **Template-Based Symbol Recognition**
    - Compare each cropped symbol against the template database using normalized cross-correlation or Euclidean distance
    - If similarity falls below a confidence threshold, return to step 2 for re-segmentation
    - The template with the highest match score determines the final symbol recognition
4. **Structural Analysis (Rule-Based)**

- Geometric relation to baseline
- Superscript/subscript grouping
- Fraction bar detection (horizontal line via Hough/counter shape)
- Radical detection based on √-shape template

5. **LaTeX Code Generation**

- Convert recognized symbols + structures into LaTeX
- Output string directly compilable in standard LaTeX environments

4. Architecture

```
Input Image
  → Preprocessing (OpenCV)
  → Symbol Segmentation
  → Template-Based Recognition
  → Structural Interpretation
  → LaTeX Code Generation
  → Final Output
```

5. Tools

- **OpenCV** for thresholding, contour extraction, template matching
- **NumPy** for array manipulation
- **Python** as main implementation
- **LaTeX** to verify correctness of generated output

6. Milestones (6 Weeks)

Week 1–2: Preprocessing + contour segmentation

Week 2–3: Build template database + implement recognition

Week 4: Structural analysis (sup/sub, fraction, radical)

Week 5: Integration, test set creation, demo & report

7. Division of Work (3 Members)

Member	Responsibilities
A — Image Processing & Segmentation	Preprocessing, contour extraction, symbol cropping, baseline estimation
B — Template Matching & Structure Analysis	Build/maintain templates; implement recognition; detect superscripts/subscripts, fractions, radicals
C — LaTeX Generator & Integration	Implement LaTeX output module; pipeline integration; test case design; final documentation & presentation