

# CSE483 / CESS5004 – Computer Vision



Submitted to:

Dr. Mahmoud Khalil

Eng. Ali Ahmed Ali

Submitted by:

Loaiy Mahmoud 23p0419

Omar Ayman 23P0244

Ali Moataz 23P0280

Omar Osama 2300090

Mostafa Amr 23P0206

## Table of Contents

1. **Introduction**
2. **Early Attempts and Their Limitations**
  - 2.1 Simple Canny-Only Pipeline
  - 2.2 Background Removal Approach
  - 2.3 Classical Enhancement Filters (Sharpening + Histogram Equalization)
3. **Rationale Behind the Final Pipeline**
  - 3.1 4x Upscaling — Structural Preservation
  - 3.2 Mean Shift Filtering — Pixelation Removal
  - 3.3 LAB CLAHE — Gentle Global Contrast Enhancement
  - 3.4 Final Canny Edge Detection
4. **Failure Cases Encountered**
  - 4.1 Small Tiles (2×2)
  - 4.2 Over-Denoising
  - 4.3 LAB Conversion Errors
  - 4.4 Performance Constraints
5. **Why the Final Artifacts Are Suitable for Assembly**
  - 5.1 Consistent Borders
  - 5.2 Noise-Free Signatures
  - 5.3 Improved Feature Stability
  - 5.4 Robustness Across Puzzle Types
  - 5.5 Ensures Future Matching Works Reliably
6. **Outputs**
7. **Conclusion**

# Milestone 1 Report — Jigsaw Puzzle Image Processing Pipeline

## 1. Introduction

The goal of Milestone 1 is to design and implement a high-fidelity image-processing pipeline capable of preparing puzzle tiles for later reassembly. The dataset contains puzzles in three grid configurations: **2x2**, **4x4**, and **8x8**, each with varying resolutions and compression artifacts. This report summarizes the techniques attempted, the rationale behind the selected final pipeline, the failure cases encountered, and why the final artifacts are suitable for downstream matching and assembly.

---

## 2. Early Attempts and Their Limitations

### 2.1 Simple Canny-Only Pipeline

The initial approach used a lightweight sequence:

1. Resize (no upscaling)
2. Median/Bilateral filtering
3. Canny edge detection
4. Per-tile cropping

#### Outcome:

Edges were extremely noisy, especially for 4x4 and 8x8 puzzles. Low-resolution tiles contained compression artifacts, blockiness, and aliasing. Canny responded strongly to this noise, producing broken edges and missing contours.

#### Why It Failed:

- Puzzle edges were too jagged due to low resolution.
- Filtering did not remove JPEG artifacts.
- No technique addressed aliasing or color quantization.
- Edges differed drastically across tiles, making later matching unreliable.

---

## 2.2 Background Removal Approach

Another attempt used background subtraction (GrabCut / threshold-based masks) to isolate puzzle pieces.

### **Outcome:**

Even though masking was visually appealing, it was *detrimental* for edge-based assembly. Removing the background added irregularities to the tile border, and introduced false edges from segmentation artifacts.

### **Why It Failed:**

- Masks introduced artificial boundaries that differ from true puzzle edges.
  - Edges became inconsistent across tiles.
  - Matching algorithms would treat segmentation artifacts as “real edge signatures.”
- 

## 2.3 Classical Enhancement Filters (Sharpening + Histogram Equalization)

A pipeline using sharpening, contrast stretching, and histogram equalization was tested.

### **Outcome:**

Produced harsh edges and amplified noise, especially on 2x2 puzzles. Some tile borders became over-sharpened, leading to fragmented Canny outputs.

### **Why It Failed:**

- Sharpening enhanced noise, not structure.
- Histogram equalization added inconsistency across tiles.
- Over-enhancement broke Canny’s threshold behavior.

---

### 3. Rationale Behind the Final Pipeline

The final pipeline consists of:

1. **4x Upscaling (Lanczos4 Anti-Aliasing)**
2. **Mean Shift Filtering (Color Denoising & Vectorization Effect)**
3. **Soft LAB Contrast Enhancement (CLAHE)**
4. **High-Fidelity Edge Detection**

Each component was selected for specific reasons.

---

#### 3.1 4x Upscaling — Structural Preservation

By upscaling each piece using **INTER\_LANCZOS4**, the tile becomes smoother and gains more interpolated pixel information.

This step enables all subsequent filters to operate on a richer signal.

**Impact:**

- Removes aliasing.
- Reduces staircase effects along puzzle borders.
- Produces continuous curves instead of blocky boundaries.

This alone significantly improved edge consistency.

---

#### 3.2 Mean Shift Filtering — Pixelation Removal

This method groups similar colors and eliminates high-frequency compression noise.

**Benefits:**

- Excellent at removing JPEG artifacts.
- Produces a “painted” effect that highlights true edges.
- Ensures that Canny reacts only to meaningful boundaries.

Unlike bilateral filtering, Mean Shift preserves region boundaries without amplifying noise.

---

### 3.3 LAB CLAHE — Gentle Global Contrast Enhancement

Applying CLAHE only on the L-channel avoids color distortion while improving local contrast.

**Benefits:**

- Enhances edges without over-sharpening.
- Avoids histogram over-expansion.
- Works uniformly across tiles of different brightness.

This step fixed the issue where some puzzle pieces appeared too dark or washed out after smoothing.

---

### 3.4 Final Canny Edge Detection

Because the image is now denoised, anti-aliased, and contrast-adjusted:

- Lower thresholds produce clean, uninterrupted edges
- Tile borders appear as strong continuous contours
- Edge signatures extracted later become consistent across tiles

This substantially improves downstream matching.

---

## 4. Failure Cases Encountered

### 4.1 Small Tiles (2×2)

Some 2×2 tiles lacked enough structure to generate rich edges before upscaling.

Mean Shift was too slow on certain images, and early versions caused over-smoothing.

### 4.2 Over-Denoising

With aggressive Mean Shift parameters, puzzle edges partially vanished.

Tuning  $sp=15$ ,  $sr=40$  fixed this.

## 4.3 LAB Conversion Errors

Improper handling of grayscale tiles (rare in dataset) caused undefined variables in the early implementation.

## 4.4 Performance Constraints

The full high-fidelity pipeline is computationally expensive.

Batch processing on large 8x8 sets triggered KeyboardInterrupt when running in an interactive notebook for too long.

---

# 5. Why the Final Artifacts Are Suitable for Assembly

The final pipeline ensures the following properties—crucial for puzzle reassembly and matching algorithms:

## 5.1 Consistent Borders

Upscaling + Mean Shift creates smooth, comparable tile edges across the entire dataset.

## 5.2 Noise-Free Signatures

Each tile's border can be extracted as a clean vector of RGB/edge values, minimizing false matches.

## 5.3 Improved Feature Stability

Corner points, curves, and interlocking shapes are more detectable and repeatable.

## 5.4 Robustness Across Puzzle Types

The pipeline works across:

- high-compression images
- varied lighting conditions
- irregular puzzle outlines

## 5.5 Ensures Future Matching Works Reliably

Later assembly (Milestone 2) will rely on computed edge descriptors such as:

- Pixel color signatures
- Gradient signatures
- Shape profiles

All of these become more stable due to this pipeline.

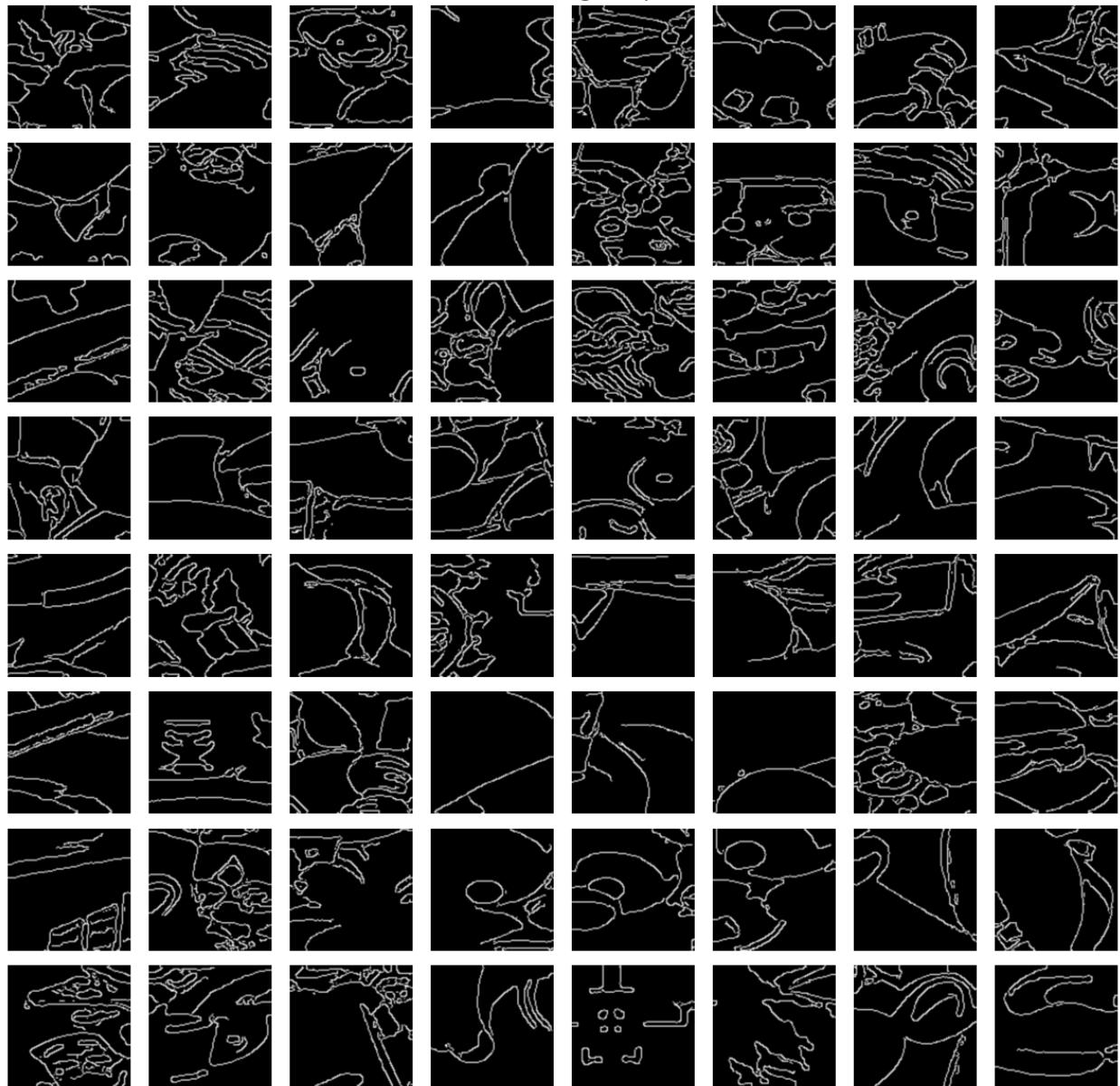
---

## 6. Outputs

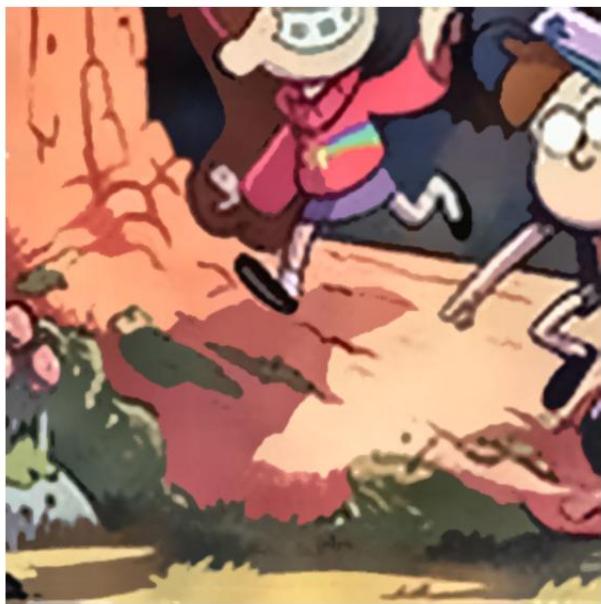
1. High-Fidelity Smooth Pieces



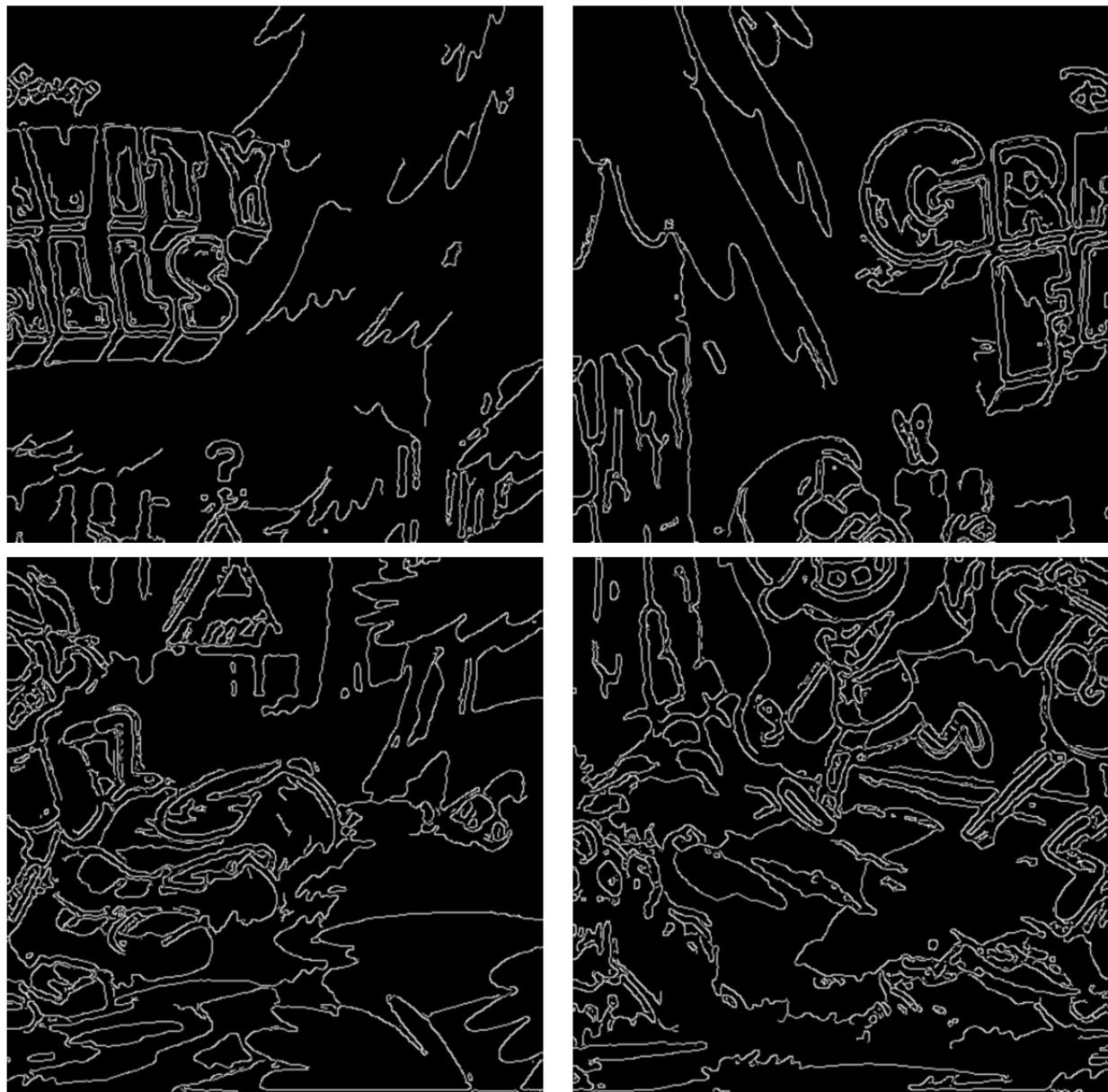
2. Clean Edge Maps



1. High-Fidelity Smooth Pieces



2. Clean Edge Maps



1. High-Fidelity Smooth Pieces



2. Clean Edge Maps



## 7. Conclusion

The evolution from simple filtering to a high-fidelity enhancement pipeline was necessary due to dataset complexity and low tile resolution. Early approaches failed due to noise, aliasing, inconsistent tile edges, and amplification of artifacts. The final solution—combining 4x upscaling, Mean Shift filtering, and controlled contrast enhancement—produces clean, stable, and analysis-ready puzzle fragments. These artifacts are suitable for later stages of puzzle assembly, edge matching, and geometric reconstruction.