Computer Vision CSE 381

Milestone 1

Team 32 Member :

Ahmed Mohamed Elmahdy Nagaty	**2100723**
Maria Ibraheem Nasseef Mikhaeil	**1901487**
Ahmed Elsayed Sayed Ahmed Mohmed	**20p7195**
Meena Maged Abdo Mekhaiel	**1900694**
Youssef Ashraf Mahmoud Othman Agha	**2301134**

Methodology and Techniques:

1. Overview

Our team adopted a modular pipeline approach, dividing the task into five distinct stages to ensure robustness against noise and varying orientations. This structured methodology transforms raw input into actionable mathematical features suitable for automated solving.

2. Pipeline Stages

2.1. Image Standardization & Basic Preprocessing

- **Techniques:** Input images were resized to a fixed reference width (800px) and converted to grayscale. A 5x5 Gaussian blur was applied immediately following conversion.
- **Rationale:** Standardization ensures that feature scale remains consistent across different input resolutions, preventing scale-variant errors. Gaussian smoothing was selected as the initial step to suppress high-frequency sensor noise before enhancement filters were applied.

2.2. Advanced Image Enhancement

- **Techniques:** We implemented a pipeline consisting of three specific filters:
    1. **CLAHE** (Contrast Limited Adaptive Histogram Equalization).
    2. **Bilateral Filtering** (d=9, sigma_color=75, sigma_space=75).
    3. **Laplacian Sharpening**.
- **Rationale:** Standard equalization often amplifies noise in uniform regions. CLAHE was chosen to improve local contrast specifically in dark or low-texture areas of the puzzle. Crucially, Bilateral Filtering was used instead of standard blurring because it smooths texture (smoothing the puzzle piece surface) while preserving sharp edges, which is vital for accurate contour detection later in the pipeline.

2.3. Segmentation (Tile Slicing)

- **Technique:** The preprocessed image was segmented into an NxN grid (e.g., 2x2) based on user selection.
- **Rationale:** This creates a "ground truth" dataset of known adjacent pieces. This structured slicing allows us to isolate the internal visual content of each piece to validate our feature extraction logic before attempting to solve irregular physical puzzle borders.

2.4. Contour Extraction

- **Techniques:**
    - **Auto-Canny Edge Detection:** We implemented a dynamic thresholding function where thresholds are calculated based on the median pixel intensity (v) of the tile: Lower = 0.66v, Upper = 1.33v
    - **Morphological Dilation:** Applied using a 3x3 rectangular kernel.

    - **Hierarchy Retrieval:** Utilized ***cv2.RETR_TREE***.
- **Rationale:** Fixed thresholds failed on images with varying brightness. The Auto-Canny method adapts to the specific lighting of each tile. Dilation was essential to bridge small gaps in artistic lines, ensuring ***findContours*** detected closed shapes rather than broken segments. We utilized ***RETR_TREE*** to capture internal details (the drawing inside the piece), which acts as the unique fingerprint for matching.

2.5. Feature Extraction (Hu Moments)

- **Technique:** 7-invariant Hu Moments were computed for each valid contour and log-transformed.
- **Rationale:** To facilitate matching in future milestones, we required descriptors that are invariant to rotation. Hu Moments compress the complex shape of the puzzle content into a lightweight vector, allowing for efficient Euclidean distance comparison regardless of the piece's orientation.

3. Failure Cases and Solutions

During development, we encountered and resolved three primary failure cases:

1. **The "Box" Detection**
    - *Issue:* Initially, the algorithm extracted the square border of the image slice as the primary contour.
    - *Solution:* We implemented a geometric filter (if area > 0.95 * tile_area) to discard any contour that matches the frame size, ensuring only internal content is captured.
2. **Sensitivity to Lighting**
    - *Issue:* Hardcoded Canny thresholds (50, 150) caused faint lines in darker tiles to be ignored.
    - *Solution:* The implementation of the ***auto_canny*** function based on median intensity statistics resolved this by dynamically adjusting sensitivity.
3. **Broken Contours**
    - *Issue:* Faint artistic lines were often detected as disconnected segments.
    - *Solution:* Applying morphological dilation connected these segments into valid, closed loops suitable for moment calculation.

4. Justification of Artifact Suitability

The processed artifacts (enhanced tiles and the ***puzzle_features.json*** file) are suitable for later assembly for the following reasons:

- **Decoupling:** The JSON file separates image processing from logic. The next milestone can focus purely on mathematical comparison of vectors without reprocessing images.
- **Invariance:** The extracted Hu Moments satisfy the requirement for rotation invariance, allowing the solver to match pieces even if they are rotated 90, 180, or 270 degrees.

- **Noise Robustness:** The combination of Bilateral filtering and Auto-Canny ensures that the features represent the actual visual content of the puzzle, not digital artifacts or film grain.

5. Conclusion

The implemented pipeline successfully converts raw input images into a structured, mathematical representation. By leveraging adaptive enhancement and dynamic edge detection, the system satisfies the requirements for robust preprocessing and feature extraction, providing a reliable foundation for the automated assembly engine.

6- Code

https://colab.research.google.com/drive/1q5BN6B3SX8HZQERcWieHf_JjBTKHNd6o?usp=sharing

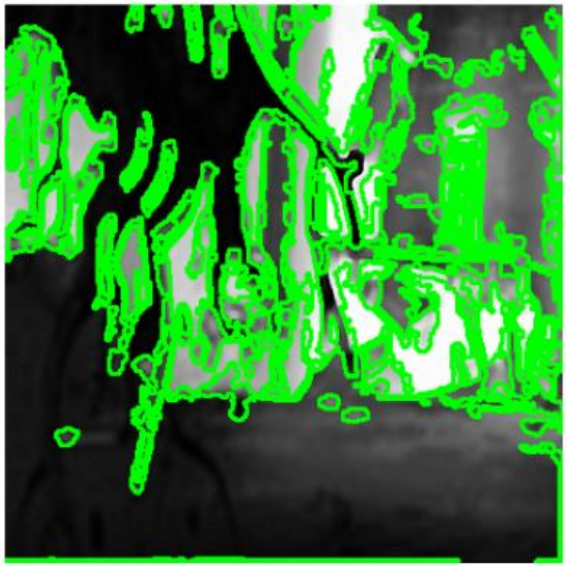
(0, 0)

(0, 1)

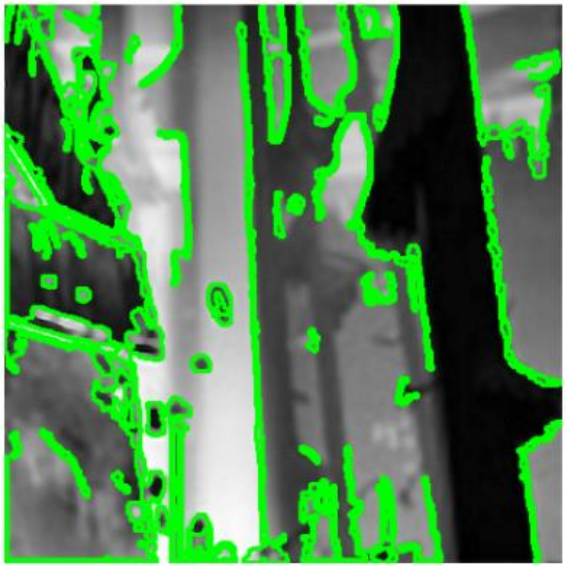
(1, 0)

(1, 1)

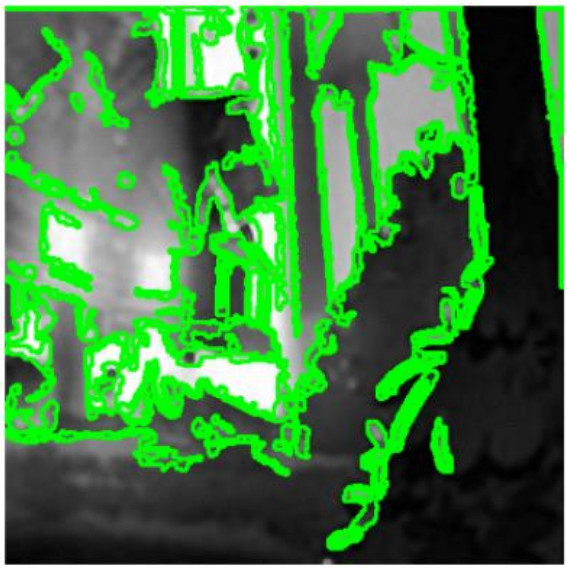
Tile 0 Contours: 96



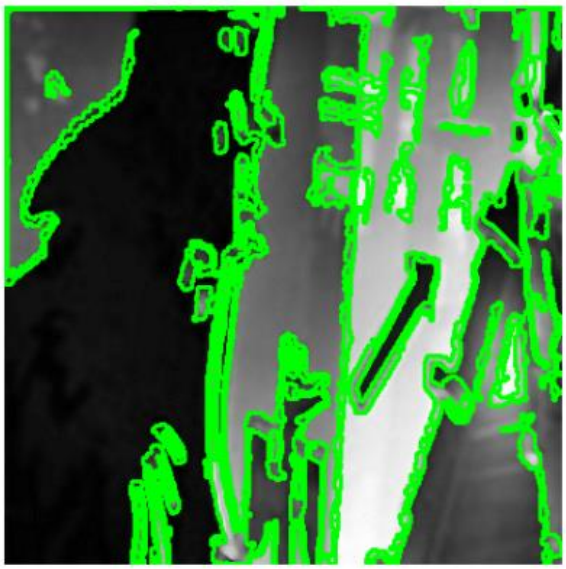
Tile 1 Contours: 54



Tile 2 Contours: 64

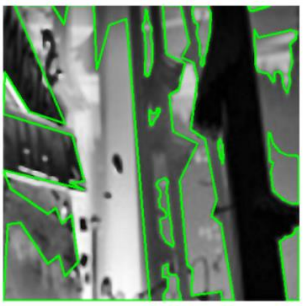


Tile 3 Contours: 52

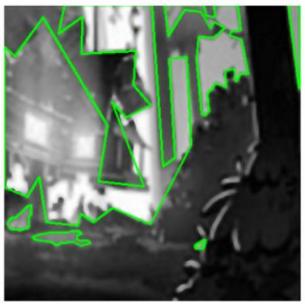


Tile 0

Tile 1



Tile 2



Tile 3