Assignment 1: Connected Component Labeling and Analysis

name
UGA ID: 811xxxx

CSCI 8820: Computer Vision and Pattern Recognition
Department of Computer Science, University of Georgia
February 12, 2026

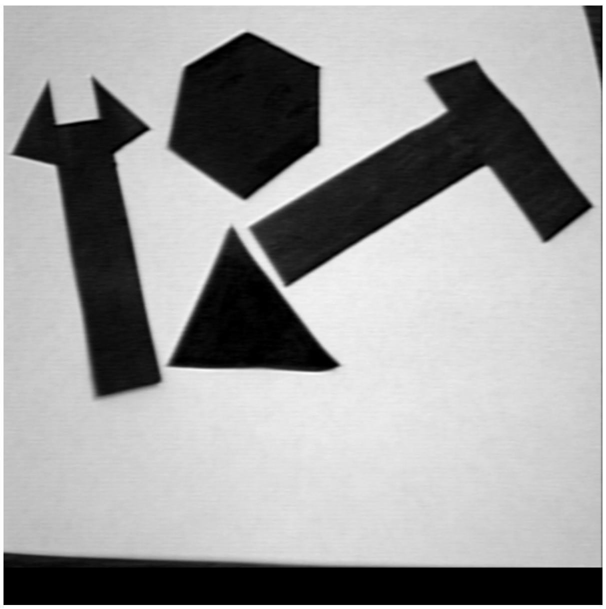
Contents

1 Introduction and Preprocessing

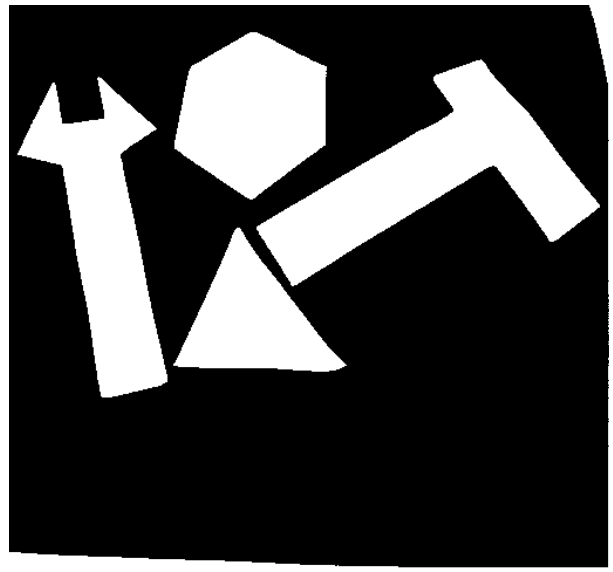
This report implements an iterative Connected Component Labeling (CCL) algorithm to segment and analyze objects in a grayscale image. The pipeline involves binary thresholding, labeling, and geometric feature extraction.

1.1 Thresholding and Binary Mask Generation

The input image B was converted to a binary image B_T using a fixed threshold $T = 128$. To ensure consistency, the mask was automatically inverted by $B_T = 1 - B_T$.



(a) Original Grayscale Image (B)



(b) Thresholded Binary Image (with inversion) (B_T)

Figure 1: Preprocessing: Conversion from 8-bit grayscale to binary mask.

2 Methodology

2.1 Connected Component Labeling (CCL)

An iterative 4-connected CCL algorithm was implemented using a two-pass approach:

1. **First Pass:** Raster-scanning the image and assigning temporary labels based on the 4 connectivity of the top and left neighbors. Collisions were managed via a union-find data structure.

2. **Second Pass:** Resolution of equivalences to merge components and sequential re-labeling of the final IDs.

2.2 Feature Extraction and Mathematical Models

Geometric properties were derived from the zeroth-, first-, and second-order central moments.

- **Area and Centroid:** The area and centroid coordinates of a binary object $B(i, j)$ are defined as

$$A = \sum_{i=1}^{n} \sum_{j=1}^{m} B(i,j), \tag{1}$$

$$X_c = \frac{1}{A} \sum_{i=1}^{n} \sum_{j=1}^{m} j B(i,j), \tag{2}$$

$$Y_c = \frac{1}{A} \sum_{i=1}^{n} \sum_{j=1}^{m} i B(i,j), \tag{3}$$

where A denotes the object area, and (X_c, Y_c) represents the centroid location.

- **Orientation (θ):** The principal axis orientation is obtained by minimizing the second central moment:

$$\theta = \frac{1}{2} \tan^{-1}\left(\frac{b}{a-c}\right), \tag{4}$$

where a, b, and c are second-order central moments.

- **Second-Order Central Moments:** These moments are computed as

$$a = \sum_{i=1}^{n} \sum_{j=1}^{m} [X'(i,j)]^2 B(i,j), \tag{5}$$

$$b = 2 \sum_{i=1}^{n} \sum_{j=1}^{m} X'(i,j) Y'(i,j) B(i,j), \tag{6}$$

$$c = \sum_{i=1}^{n} \sum_{j=1}^{m} [Y'(i,j)]^2 B(i,j), \tag{7}$$

where $X'(i,j) = j - X_c$ and $Y'(i,j) = i - Y_c$ denote centroid-shifted coordinates.

- **Eccentricity:** The eccentricity of the object is computed from the eigenvalues of the covariance matrix:

$$\text{Eccentricity} = \sqrt{1 - \frac{I_{\min}}{I_{\max}}}, \tag{8}$$

where $I_{\max}$ and $I_{\min}$ are the major and minor principal moments, respectively.

Note: Consistency of the moment calculations was verified by checking that the invariant relation $|(a+c) - (I_{\max} + I_{\min})|$ remained close to zero. With a maximum absolute error of 6.0×10^{-8} across all tests.

- **Compactness:** Compactness, a measure of shape circularity, is defined as

$$\text{Compactness} = \frac{P^2}{A}, \tag{9}$$

where P is the perimeter and A is the area.

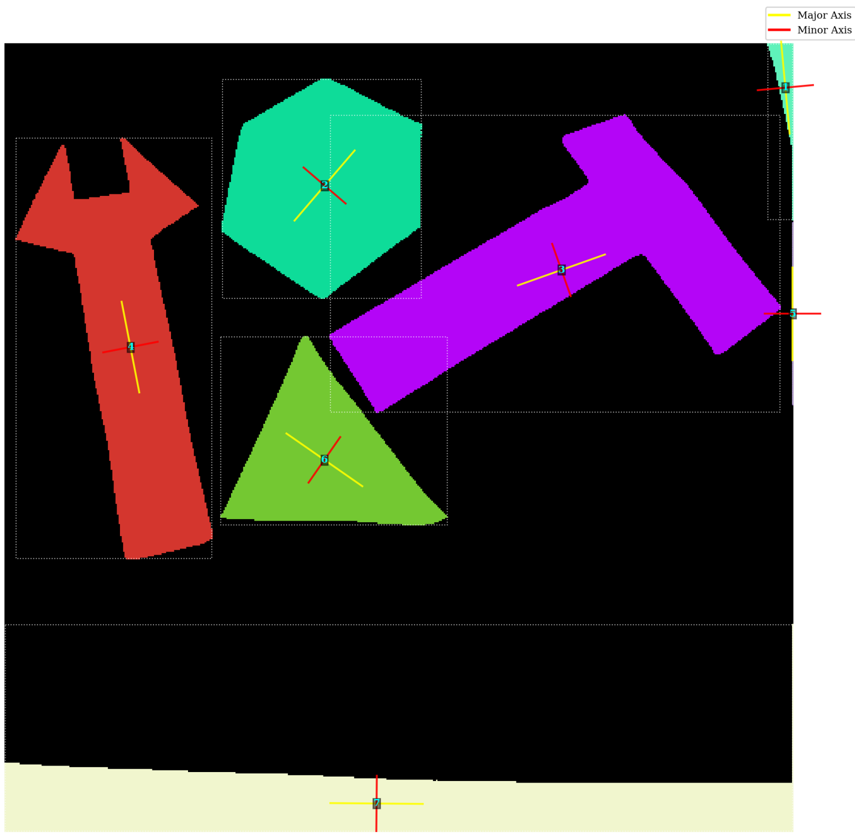
3 Experimental Results

The algorithm was evaluated across three minimum size thresholds: 100, 500, and 1000 pixels.
Note: Kindly correlate the Labeled Components in Figure 2, Figure 4 and Figure 6 with Component Description Table Figure 3, Figure 5 and Figure 7 respectively.

3.1 Case 1: Minimum Size Threshold = 100

At $T_{size} = 100$, the algorithm labels artifacts/noise in the image as components along with primary objects of interest as shown in *Figure* 2.

Image C: Filtered Components (Size ≥ 100)



Component Description Table (Size ≥ 100)

ID	Area	Centroid	Bounding Box	Orient(deg)	Elongation	Eccentricity	Perimeter	Compactness
1	620	(506.4, 28.2)	[495,0,511,114]	84.7	57.10	0.991	196	61.96
2	13148	(207.3, 91.7)	[141,23,270,165]	-49.4	1.18	0.388	380	10.98
3	19762	(361.1, 146.6)	[211,46,503,239]	-19.6	5.54	0.905	693	24.30
4	15327	(81.3, 196.6)	[7,61,134,334]	79.0	13.80	0.963	732	34.96
5	119	(511.0, 175.0)	[511,116,511,234]	90.0	0.00	1.000	119	119.00
6	8913	(207.2, 269.8)	[140,190,287,312]	34.8	1.49	0.573	383	16.46
7	18472	(241.1, 493.0)	[0,377,511,511]	0.5	158.65	0.997	1200	77.96

Figure 2: Labeled Components and Corresponding Feature Table for Case 1

3.2 Case 2: Minimum Size Threshold = 500

Increasing the threshold effectively filters out noise while correctly labelling the primary objects.

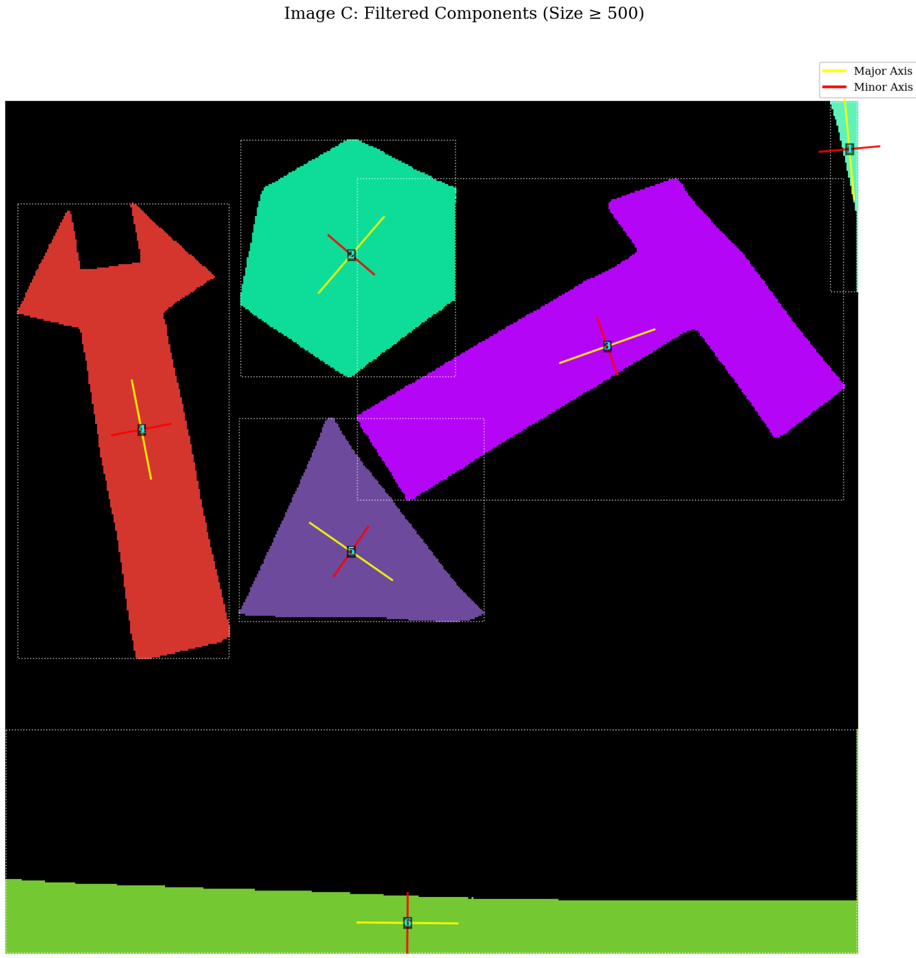


Figure 3: Labeled Components ($Size \geq 500$).

Component Description Table (Size ≥ 500)

ID	Area	Centroid	Bounding Box	Orient(deg)	Elongation	Eccentricity	Perimeter	Compactness
1	620	(506.4, 28.2)	[495,0,511,114]	84.7	57.10	0.991	196	61.96
2	13148	(207.3, 91.7)	[141,23,270,165]	-49.4	1.18	0.388	380	10.98
3	19762	(361.1, 146.6)	[211,46,503,239]	-19.6	5.54	0.905	693	24.30
4	15327	(81.3, 196.6)	[7,61,134,334]	79.0	13.80	0.963	732	34.96
5	8913	(207.2, 269.8)	[140,190,287,312]	34.8	1.49	0.573	383	16.46
6	18472	(241.1, 493.0)	[0,377,511,511]	0.5	158.65	0.997	1200	77.96

Figure 4: Extracted Features for Case 2 .

3.3 Case 3: Minimum Size Threshold = 1000

At $T_{size} = 1000$, leaves only the most significant geometric structures.

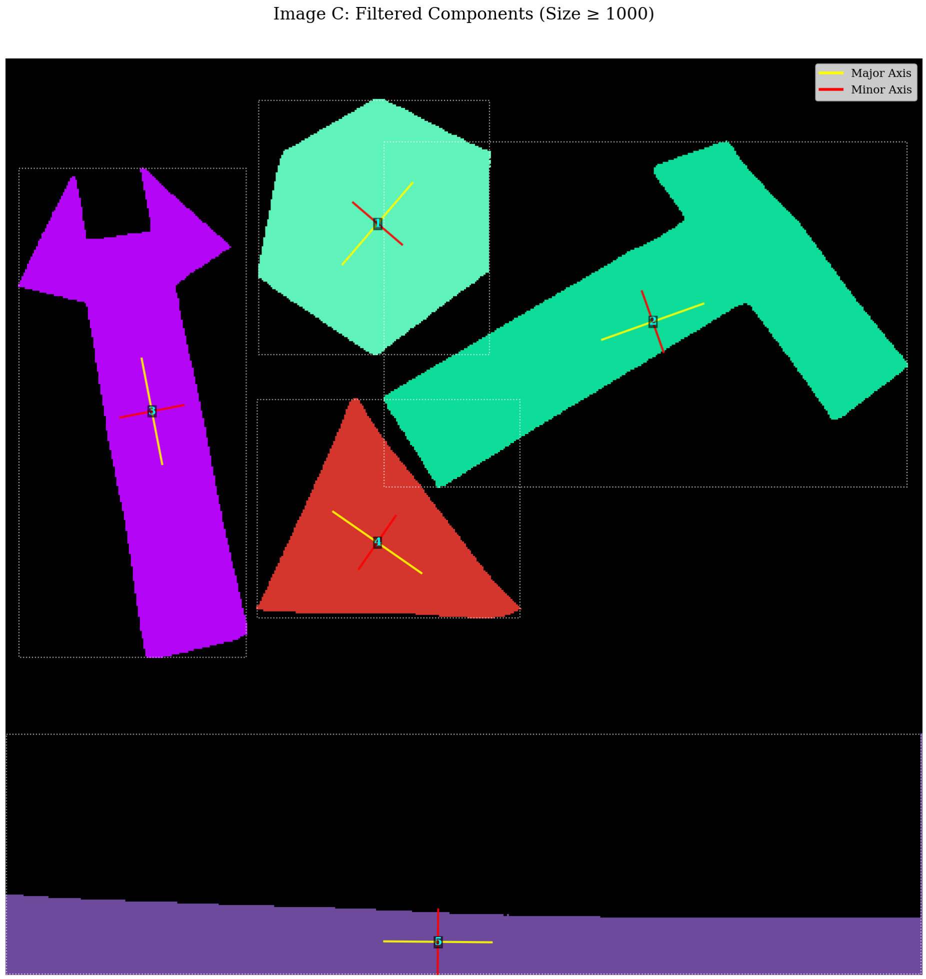


Figure 5: Labeled Components ($Size \geq 1000$).

Component Description Table (Size ≥ 1000)

ID	Area	Centroid	Bounding Box	Orient(deg)	Elongation	Eccentricity	Perimeter	Compactness
1	13148	(207.3, 91.7)	[141,23,270,165]	-49.4	1.18	0.388	380	10.98
2	19762	(361.1, 146.6)	[211,46,503,239]	-19.6	5.54	0.905	693	24.30
3	15327	(81.3, 196.6)	[7,61,134,334]	79.0	13.80	0.963	732	34.96
4	8913	(207.2, 269.8)	[140,190,287,312]	34.8	1.49	0.573	383	16.46
5	18472	(241.1, 493.0)	[0,377,511,511]	0.5	158.65	0.997	1200	77.96

Figure 6: Extracted Features for Case 3 .

4 Discussion and Analysis

4.1 Noise Analysis and Size Filter Trade-off

The experimental results highlight a fundamental trade-off:

- **Lower Threshold (100 & 500):** Small noise components are detected as objects.
- **High Threshold (1000):** Fine structural details may be lost if they do not meet the pixel count (T_{size}) requirement.

4.2 Geometric Validation

To ensure the accuracy of the moment-based orientation, the invariant property $I_{max} + I_{min} = a + c$ was calculated. The error was consistently negligible (6.0×10^{-8}), validating the orientation calculation and axis visualization.

5 Conclusion

The iterative CCL algorithm effectively labelled the components in the test image.

Note: For source code kindly refer to appendix at the end.

Appendix: Source Code

The following Python script was used to perform all operations described in this report.

```python
# Paste your python code here.
# It will be formatted automatically with syntax highlighting.
```