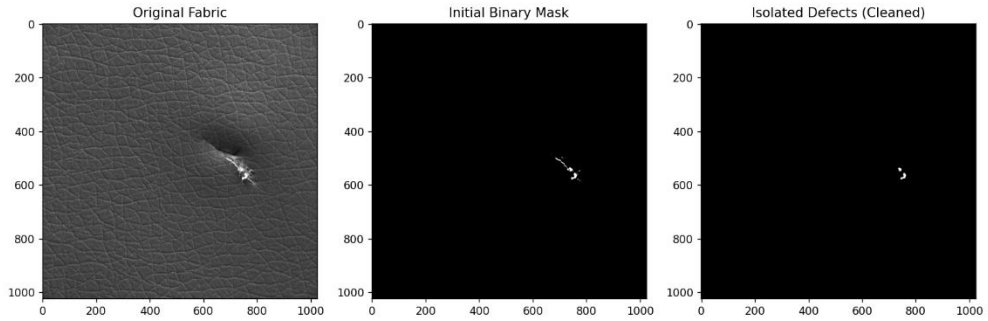
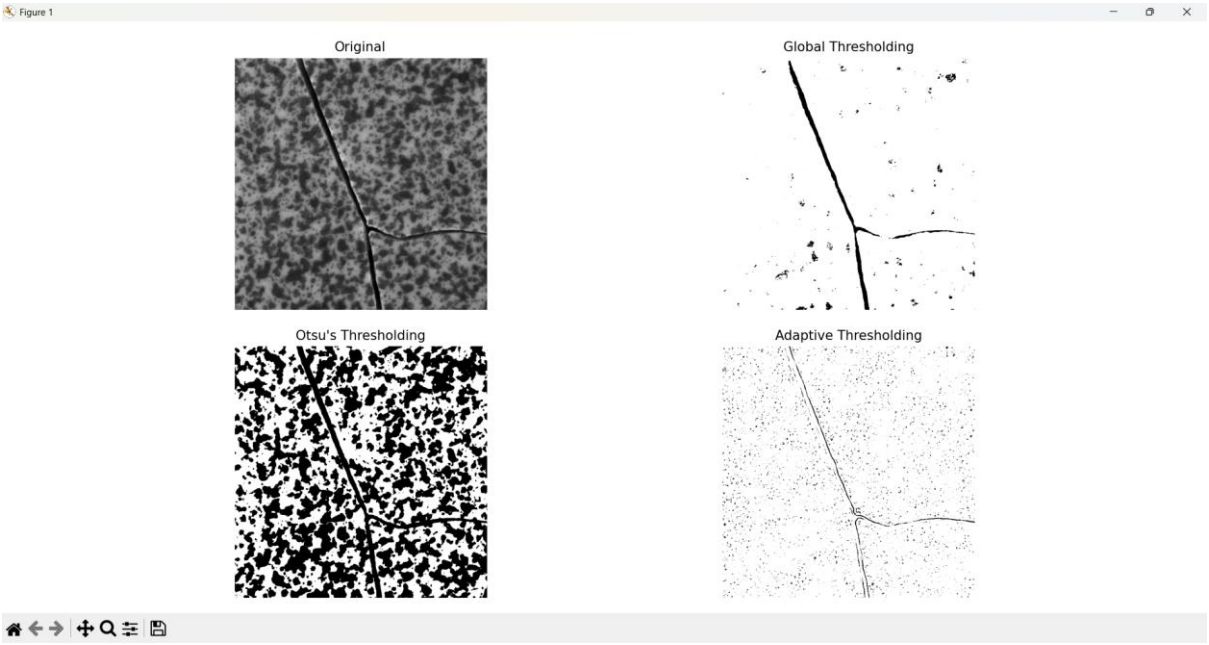
Output Screenshots of the .py files

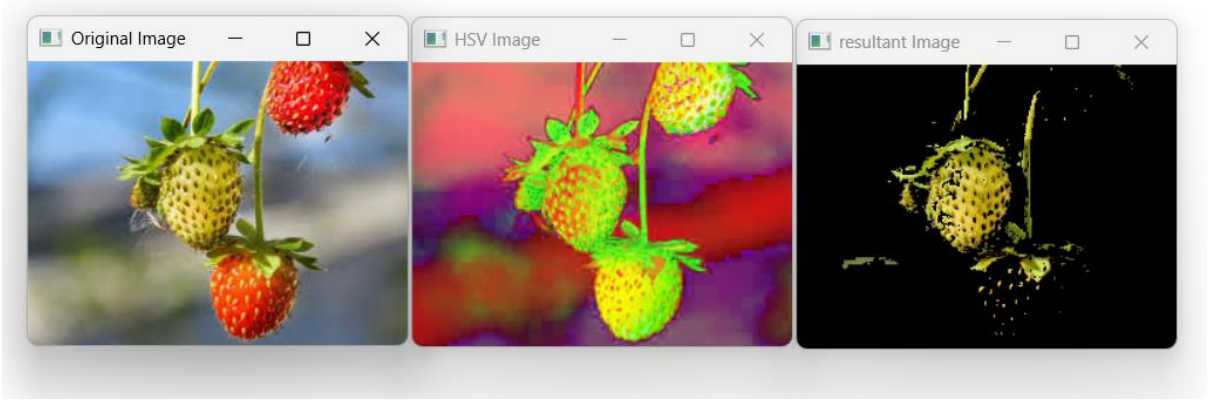
3. Use morphological operations (opening/closing) to isolate defective regions in a fabric image



4. Apply global, adaptive, and Otsu thresholding to separate defective vs. non-defective regions. Provide comparative results



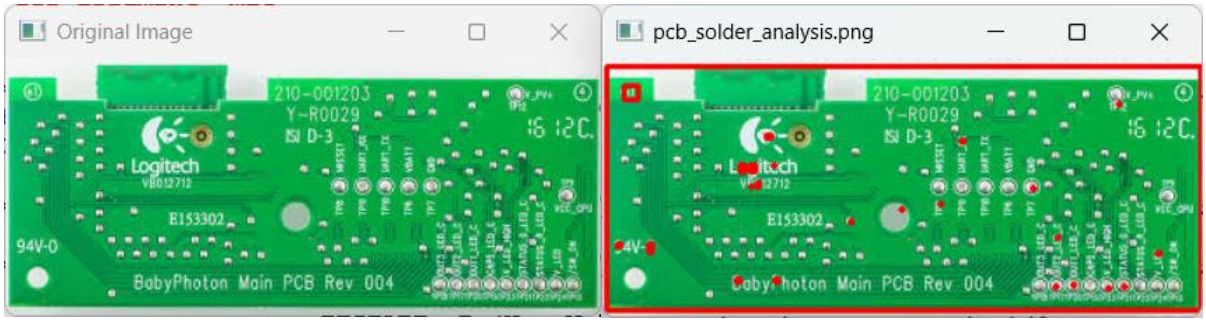
5. Develop a color-based thresholding method (HSV space) to classify fruits as ripe or unripe.



6. Use connected components analysis on a tablet image to detect missing, broken, or extra objects. Display bounding boxes for identified defects.

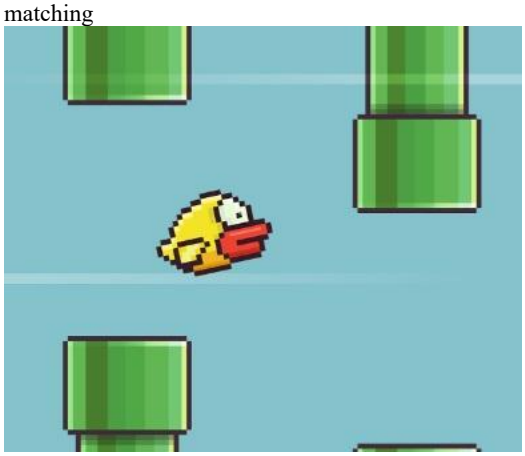
Output Screenshots of the .py files

7.Apply connected component labeling to count defective vs. good solder joints on PCB images. Provide statistics of the results.

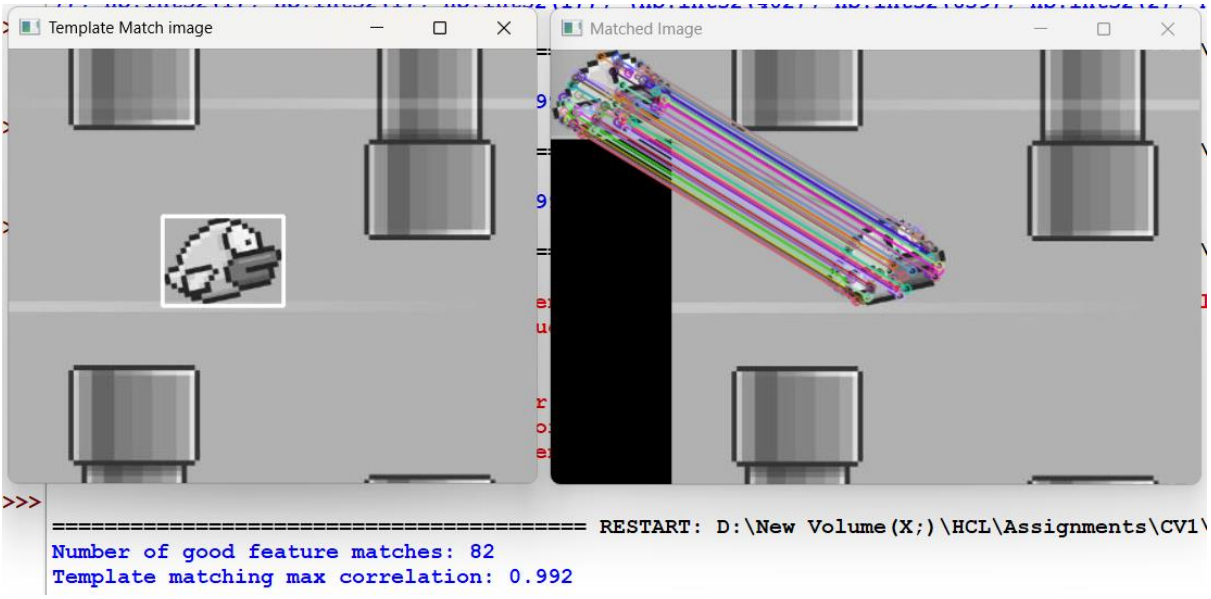



```
Solder joint statistics: {'total_joints': 25, 'good_joints': 0, 'defective_joints': 25, 'defective_percentage': 100.0, 'good_percentage': 0.0}
```

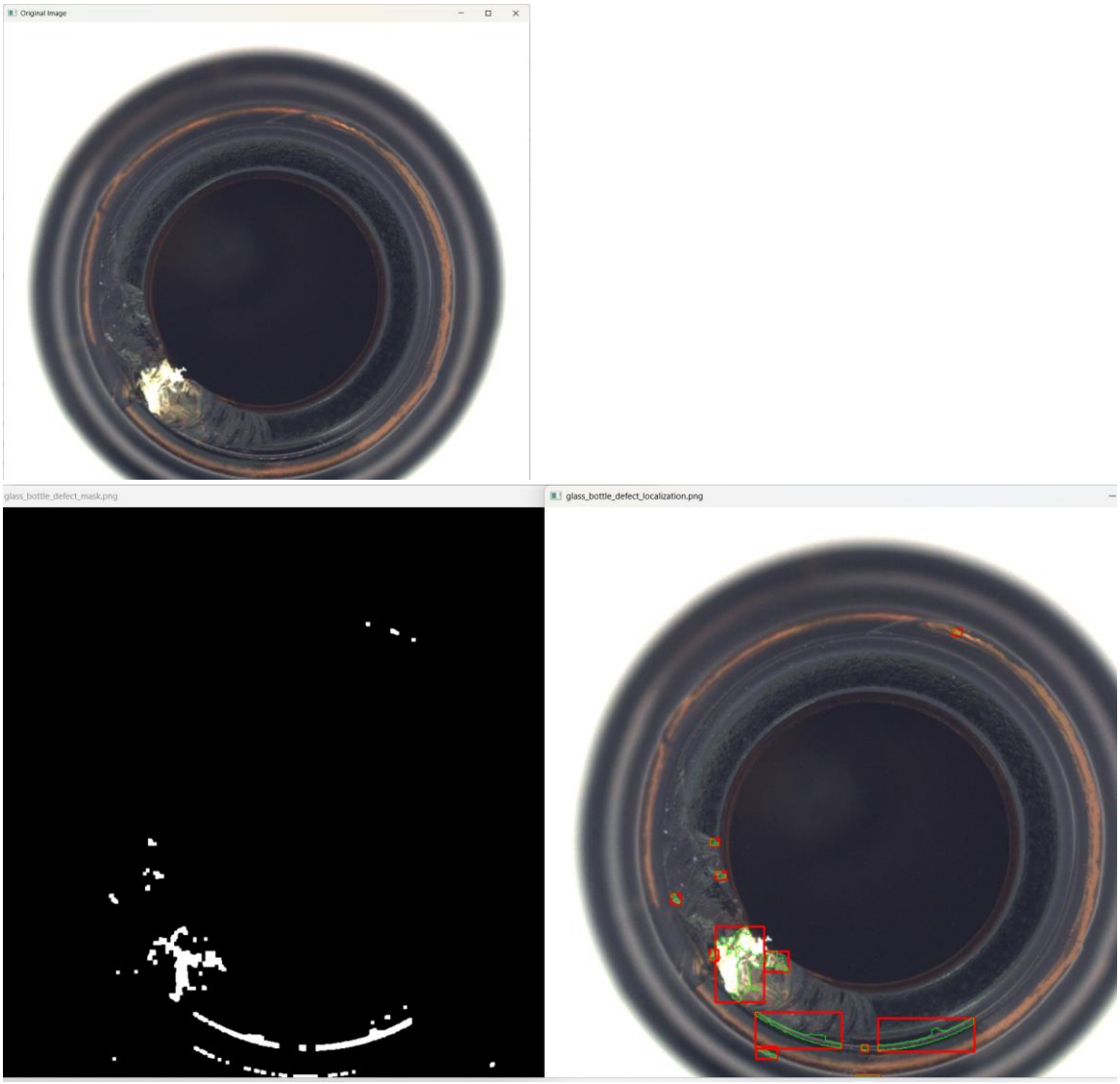
8.Implement SIFT or ORB feature matching to detect brand logos/serial numbers in product images. Compare with template matching



Output Screenshots of the .py files



9. Design a pipeline using edge detection + morphology to detect cracks or missing parts in glass bottle images. Show defect localization.



Output Screenshots of the .py files

10. Perform defect detection on PCB images using edge detection and morphology to identify broken tracks or missing solder points. Provide annotated outputs.