

Digital Image and Video Processing

Watershed Segmentation — Assignment

Name: Nandhu Krishnan A

Registration No: 80522014

Submitted to: Prof. Madhu S Nair

Date: November 6, 2025

Written Report

Print this page and insert your handwritten explanation after the title page.

Code

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

def imshow(title, img):
    """Display image with title using matplotlib (no OpenCV window)"""
    plt.figure(figsize=(6,6))
    if len(img.shape) == 3:
        plt.imshow(cv2.cvtColor(img, cv2.COLOR_BGR2RGB))
    else:
        plt.imshow(img, cmap='gray')
    plt.title(title)
    plt.axis('off')
    plt.show()

img = cv2.imread('images/coins.png')
imshow('01 Original', img)

gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
imshow('02 Grayscale', gray)

_, bin_img = cv2.threshold(gray, 0, 255,
                           cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)
imshow('03 Threshold (Otsu inverse)', bin_img)

kernel = cv2.getStructuringElement(cv2.MORPH_RECT, (3,3))
bin_img = cv2.morphologyEx(bin_img, cv2.MORPH_OPEN, kernel, iterations=2)
imshow('04 Morphological opening', bin_img)

sure_bg = cv2.dilate(bin_img, kernel, iterations=3)
dist = cv2.distanceTransform(bin_img, cv2.DIST_L2, 5)
_, sure_fg = cv2.threshold(dist, 0.5*dist.max(), 255, 0)
sure_fg = sure_fg.astype(np.uint8)
unknown = cv2.subtract(sure_bg, sure_fg)

fig, ax = plt.subplots(2,2,figsize=(10,10))
ax[0,0].imshow(sure_bg, cmap='gray'); ax[0,0].set_title('Sure Background')
ax[0,1].imshow(dist, cmap='gray'); ax[0,1].set_title('Distance Transform')
ax[1,0].imshow(sure_fg, cmap='gray'); ax[1,0].set_title('Sure Foreground')
ax[1,1].imshow(unknown, cmap='gray'); ax[1,1].set_title('Unknown')
for a in ax.flat: a.axis('off')
plt.tight_layout()
plt.show()
```

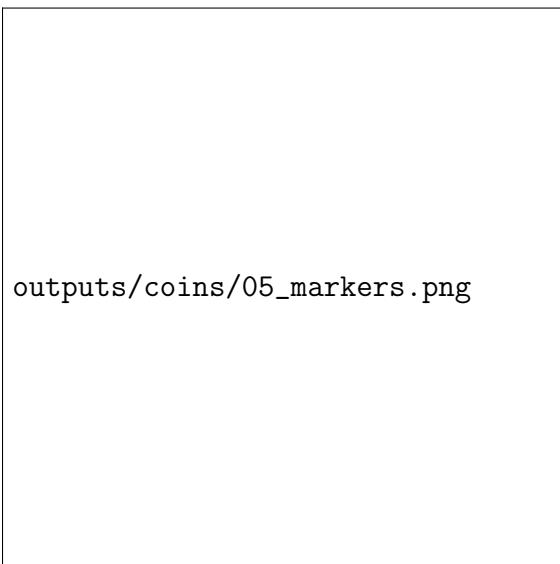
```

_, markers = cv2.connectedComponents(sure_fg)
markers = markers + 1
markers[unknown == 255] = 0
plt.figure(figsize=(6,6))
plt.imshow(markers, cmap='tab20b')
plt.title('06 Labeled markers')
plt.axis('off')
plt.show()

markers = cv2.watershed(img, markers)
img_out = img.copy()
contours = []
for label in np.unique(markers)[2:]:
    mask = np.where(markers == label, 255, 0).astype(np.uint8)
    cnts, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
    if cnts:
        contours.append(cnts[0])
img_out = cv2.drawContours(img_out, contours, -1, (0,23,223), 2)
imshow('07 Final result (watershed contours)', img_out)

```

Outputs



(Left to right, top to bottom): Original, Threshold, Markers, Final Result