Watershed Segmentation

Authors: Dheeraj Kamath, Niharika Jayanthi

June 1, 2015

Mentor: Sanam Shakya

Goal

In this chapter,

- How to use marker-based water shed segmentation on images.
- We will see: **cv2.distanceTransform()**, **cv2.watershed()**

Theory

Watershed is an algorithm in image processing used for isolating objects in the image from the background. The algorithm accepts a grayscale image and a marker image. The markers is an image where you tell the watershed about the foreground objects and the background.

Working principle

Any grayscale image can be viewed as a topographic surface where high intensity denotes peaks and hill while low intensity denotes valleys. We start filling every isolated valleys(local minima) with different colored water. As water rises , depending on the peaks (gradients) nearby, water from different valleys, labeled with different colors will start to merge. To prevent this barriers are built in the locations where the water merges. We continue doing this till all peaks under water. The result will be a segmentation of the image. But this method gives a over segmented image due to noise or any other irregularities in the image. So a major enhancement in the water shed transformation consists in flooding the topographic surface from previously defined set of markers. It is an interactive image segmentation. What we do is to give different labels for our object we know. Label the region which we are sure of being the foreground or object with one color (or intensity), label the region which we are sure of being background or non-object with another color and finally the region which we are not sure of anything, label it with 0. That is our marker. Then apply watershed algorithm. Then our marker will be updated with the labels we gave, and the boundaries of objects will have a value of -1.

Applications

1. Watershed algorithm is used to monitor traffic. It automatically segments the lanes of a road to count the number of vehicles on different lanes.
2. It can be used to detect fractures in the surface of steel.
3. Counting of objects in images can be done using watershed algorithm. An example is counting of coffee beans.

Code

1) Since segmentation needs an image in grayscale, we use the following code to convert the image to grayscale.

```
'''
**********************************************************************
*                                                                    *
*          IMAGE PROCESSING ( WATERSHED SEGMENTATION )               *
*                                                                    *
*                                                                    *
*  TEAM MEMBERS : NIHARIKA JAYANTHI, DHEERAJ KAMATH                  *
*                                                                    *
*  MENTOR : SANAM SHAKYA                                             *
*                                                                    *
*  FILENAME : watershed.py                                          *
*                                                                    *
*  THEME : DEVELOP MODULES FOR IMAGE PROCESSING AND                  *
*          ROBOT LOCALISATION USING MARKERS                          *
*                                                                    *
*  FUNCTIONS : cv2.watershed(), cv2.distanceTransform()              *
*                                                                    *
*  GLOBAL VARIABLES : NONE                                           *
*                                                                    *
**********************************************************************
'''

#import opencvn numpy and matplotlib
import numpy as np
import cv2
from matplotlib import pyplot as plt

#Reading the image
img = cv2.imread('water_coins.jpg')
gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY) # Grayscale
ret, thresh = cv2.threshold(gray,0,255,cv2.THRESH_BINARY_INV
                            +cv2.THRESH_OTSU)   #thresholding

cv2.imshow("Otsu", thresh)

# noise removal
```

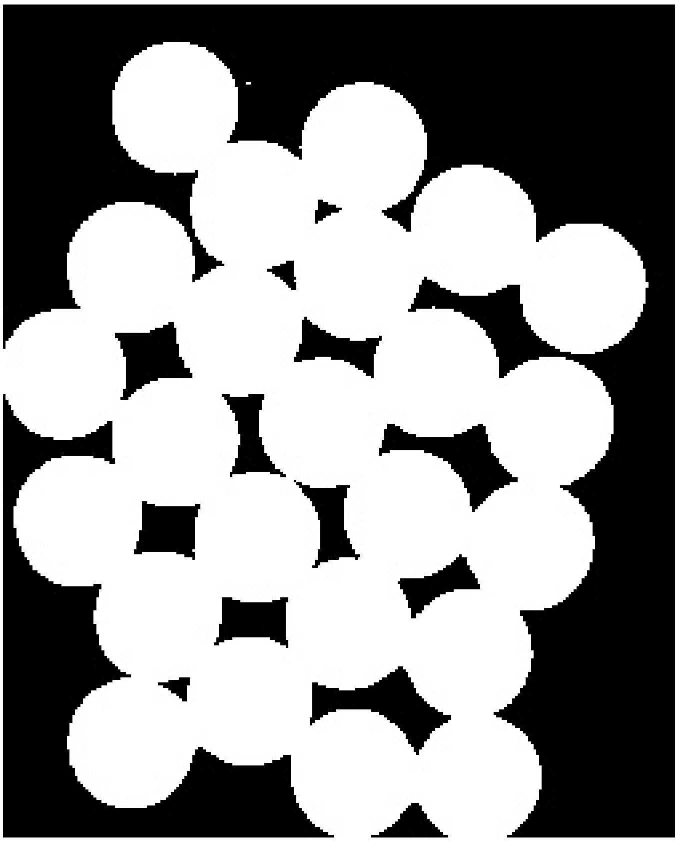
```
kernel = np.ones((3,3),np.uint8)
opening = cv2.morphologyEx(thresh,cv2.MORPH_OPEN,kernel,
                                        iterations = 2)

# sure background area
sure_bg = cv2.dilate(opening,kernel,iterations=3)


# Finding sure foreground area
dist_transform = cv2.distanceTransform(opening,
                                    cv2.cv.CV_DIST_L2,5)
ret, sure_fg = cv2.threshold(dist_transform,
                          0.7*dist_transform.max(),255,0)

# Finding unknown region
sure_fg = np.uint8(sure_fg)
unknown = cv2.subtract(sure_bg,sure_fg)

cv2.imshow("Sure bg", sure_bg)
cv2.imshow("Sure fg",sure_fg)
cv2.imshow("unknown",unknown)

#Marker Labelling (for those with python version below 3.0)
contours, hierarchy = cv2.findContours(sure_fg,cv2.RETR_TREE,
                                    cv2.CHAIN_APPROX_SIMPLE)

# Creating a numpy array for markers and converting the image
# to 32 bit using dtype paramter
marker = np.zeros((gray.shape[0], gray.shape[1]),
                                        dtype = np.int32)

marker = np.int32(sure_fg) + np.int32(sure_bg)

for id in range(len(contours)):
              cv2.drawContours(marker,contours,id
                                            ,id+2, -1)

marker = marker + 1
marker[unknown==255] = 0
```

```
"""
# Marker labelling (For those with python version 3.0 and above)
ret, markers = cv2.connectedComponents(sure_fg)

# Add one to all labels so that sure background is not 0, but 1
markers = markers+1

# Now, mark the region of unknown with zero
markers[unknown==255] = 0
"""

cv2.watershed(img, marker)
img[marker==-1]=(0,0,255)

cv2.imshow('watershed', img)

#To display using colormap
imgplt = plt.imshow(marker)
plt.colorbar()                  #Creates a bar of colors
plt.show()                      #Displays the windows

cv2.waitKey(0)
cv2.destroyAllWindows()          #Destroys all windows
```

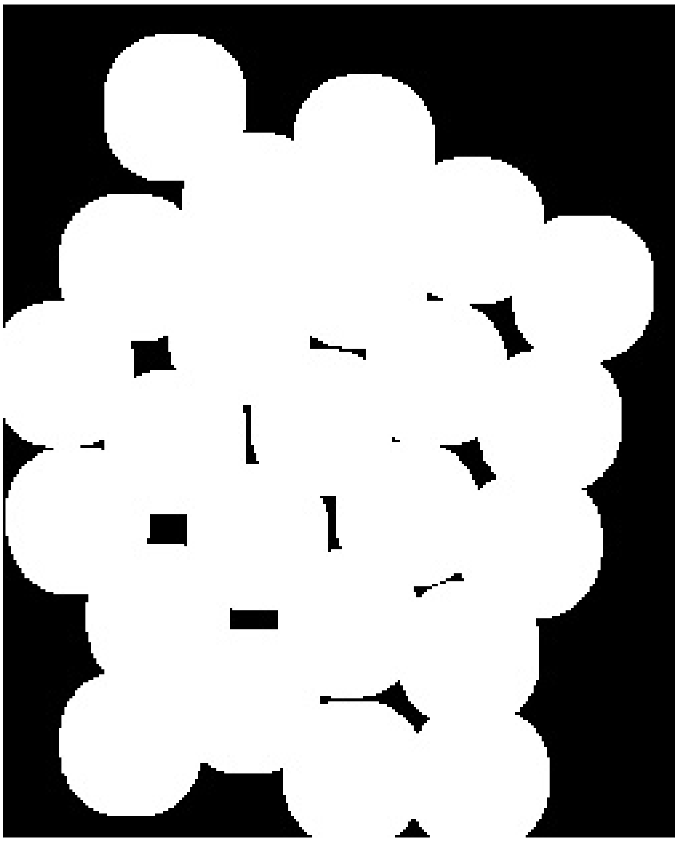
Output Images:

1) Input Image

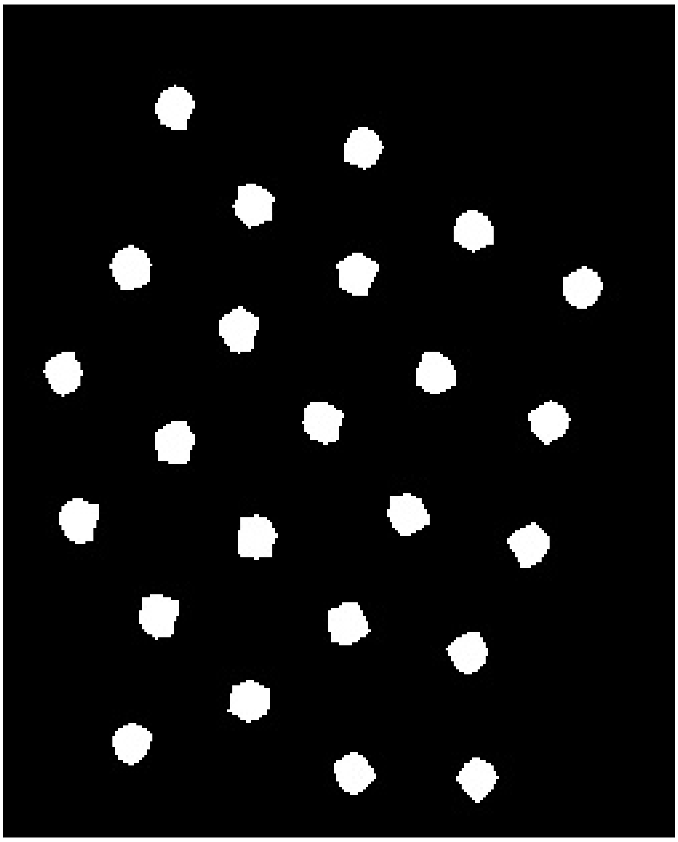
2) Otsu thresholding output image



3) Sure background output image



4) Sure foreground output image



5) Final output image

Additional Resources

1) http://cmm.ensmp.fr/~beucher/wtshed.html#examples
2) About watershed segmentation
3) Image processing lectures
4) Info on defining markers