Aim: To study the object segmentation

Objective: To study the object segmentation using watershed algorithm. Example image segmentation with watershed algorithm.

Theory:

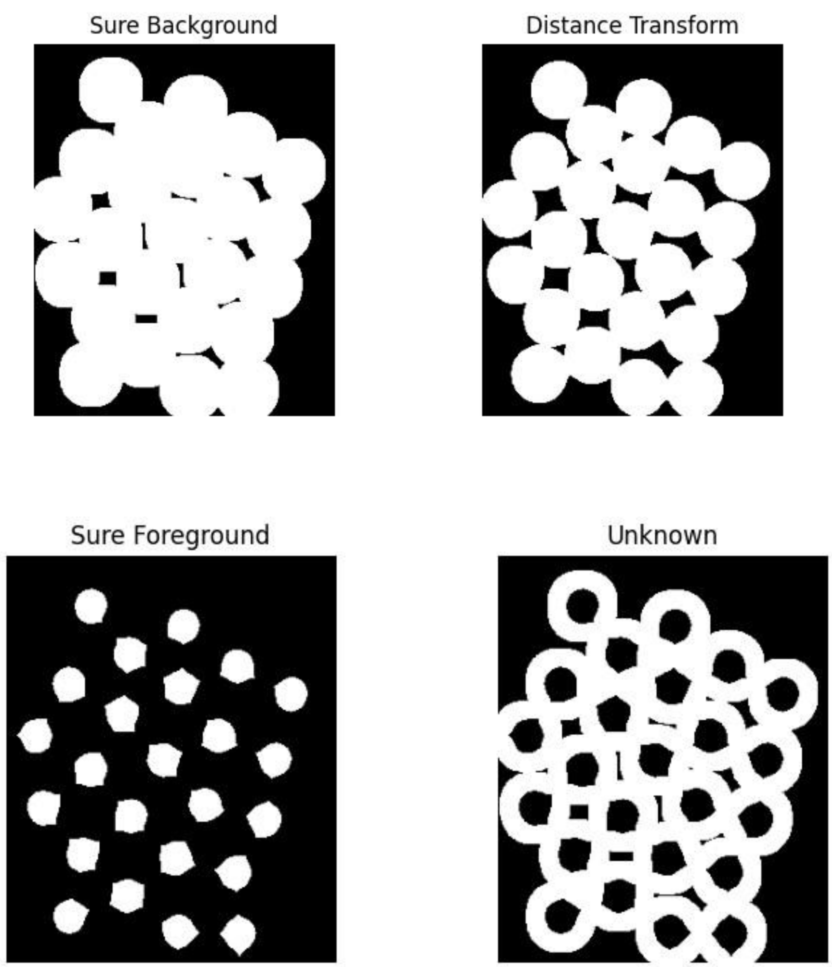
Image segmentation is the process of dividing an image into several disjoint small local areas or cluster sets according to certain rules and principles. The watershed algorithm is a computer vision technique used for image region segmentation **"**outline of an object**".**

Watershed algorithm:

The watershed algorithm uses topographic information to divide an image into multiple segments or regions. The algorithm views an image as a topographic surface, each pixel representing a different height. The watershed algorithm uses this information to identify catchment basins, similar to how water would collect in valleys in a real topographic map. The watershed algorithm identifies the local minima, or the lowest points, in the image. These points are then marked as markers. The algorithm then floods the image with different colors, starting from these marked markers. As the color spreads, it fills up the catchment basins until it reaches the boundaries of the objects or regions in the image.The catchment basin in the watershed algorithm refers to a region in the image that is filled by the spreading color starting from a marker. The catchment basin is defined by the boundaries of the object or region in the image and the local minima in the intensity values of the pixels.The algorithm uses the catchment basins to divide the image into separate regions and then identifies the boundaries between the basins to create a segmentation of the image for object recognition, image analysis, and feature extraction tasks.

The whole process of the watershed algorithm can be summarized in the following steps:

- **Marker placement:** The first step is to place markers on the local minima, or the lowest points, in the image. These markers serve as the starting points for the flooding process.
- **Flooding**: The algorithm then floods the image with different colors, starting from the markers. As the color spreads, it fills up the catchment basins until it reaches the boundaries of the objects or regions in the image.
- **Catchment basin formation**: As the color spreads, the catchment basins are gradually filled, creating a segmentation of the image. The resulting segments or regions are assigned unique colors, which can then be used to identify different objects or features in the image.
- **Boundary identification**: The watershed algorithm uses the boundaries between the different colored regions to identify the objects or regions in the image. The resulting segmentation can be used for object recognition, image analysis, and feature extraction tasks.



Code:

```
import cv2
import numpy as np
from IPython.display import Image, display
from matplotlib import pyplot as plt
# Plot the image
def imshow(img,ax=None):
 if ax is None:
  ret, encoded = cv2.imencode(".jpg", img)
  display(Image(encoded))
 else:
  ax.imshow(cv2.cvtColor(img,
cv2.COLOR_BGR2RGB))
  ax.axis('off')
#Image loading
img = cv2.imread("build.jpg")
#image grayscale conversion
gray=cv2.cvtColor(img,cv2.COLOR_BG2
GRAY)
# Show image
imshow(img)
#Threshold Processing
ret, bin_img = cv2.threshold(gray,0,
255,cv2.THRESH_BINARY_INV +
cv2.THRESH_OTSU)
imshow(bin_img)
```

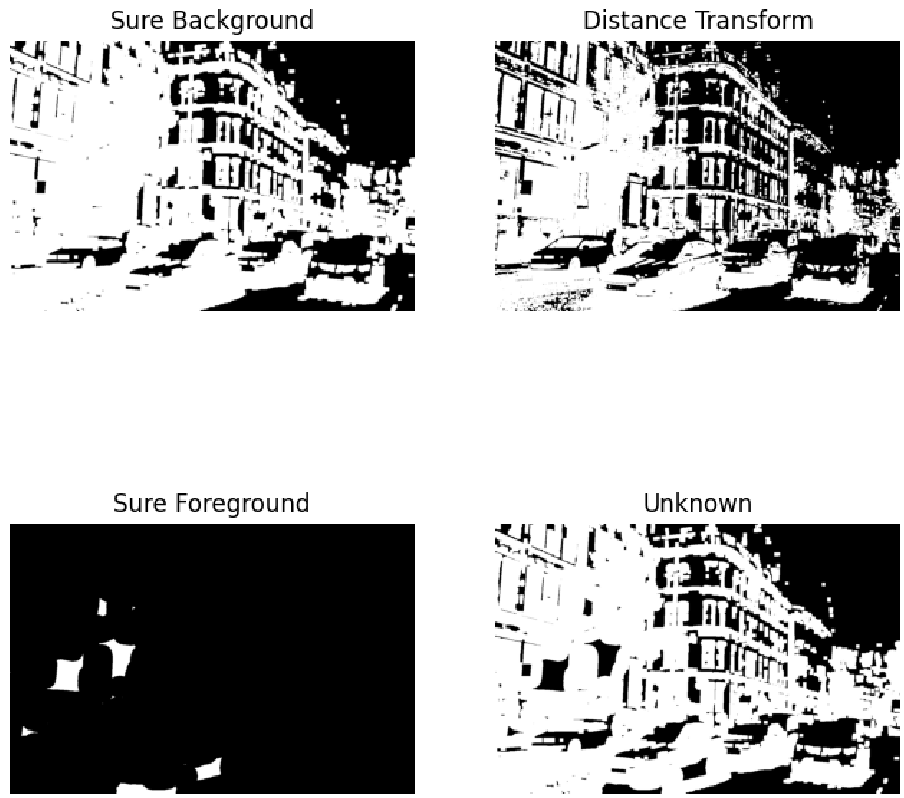
```
# noise removal
kernel =
cv2.getStructuringElement(cv2.MORPH_R
ECT, (3, 3))
bin_img = cv2.morphologyEx(bin_img,
cv2.MORPH_OPEN,
kernel,
iterations=2)
imshow(bin_img)
# Create subplots with 1 row and 2 columns
fig, axes = plt.subplots(nrows=2, ncols=2,
figsize=(8, 8))
# sure background area
sure_bg = cv2.dilate(bin_img, kernel,
iterations=3)
imshow(sure_bg, axes[0,0])
axes[0, 0].set_title('Sure Background')
# Distance transform
dist = cv2.distanceTransform(bin_img,
cv2.DIST_L2, 5)
imshow(dist, axes[0,1])
axes[0, 1].set_title('Distance Transform')
#foreground area
ret, sure_fg = cv2.threshold(dist, 0.5 *
dist.max(), 255, cv2.THRESH_BINARY)
sure_fg = sure_fg.astype(np.uint8)
```

```
imshow(sure_fg, axes[1,0])
axes[1, 0].set_title('Sure Foreground')
# unknown area
unknown = cv2.subtract(sure_bg, sure_fg)
imshow(unknown, axes[1,1])
axes[1, 1].set_title('Unknown')
plt.show()
```

Output:

Input Image:

Output Image:



Conclusion:

- This code illustrates how the watershed method is used for picture segmentation. An effective method for dividing a picture into discrete parts based on intensity and geographical clues is the watershed algorithm.
- It prepares the picture for segmentation by converting it to grayscale, applying thresholding, and carrying out morphological procedures. Following that, the watershed method determines regions of interest, such as "Sure Background" and "Sure Foreground," which are essential for object segmentation.
- The watershed approach may be used in this example to successfully separate items in an image, demonstrating how it can be utilized as a useful tool for a variety of computer vision and image processing applications.