Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

Aim: To study the Object segmentation

Objective: To study Object segmentation using the Watershed and GrabCut algorithms Example of foreground detection with GrabCut, Image segmentation with the Watershed algorithm

Theory: Object segmentation using the Watershed and GrabCut algorithms. Calculating a disparity map can be very useful to detect the foreground of an image, but StereoSGBM is not the only algorithm available to accomplish this, and in fact, StereoSGBM is more about gathering 3D information from 2D pictures, than anything else. GrabCut, however, is a perfect tool for this purpose. The GrabCut algorithm follows a precise sequence of steps:

1. A rectangle including the subject(s) of the picture is defined

2. The area lying outside the rectangle is automatically defined as a background. 3. The data contained in the background is used as a reference to distinguish background areas from foreground areas within the user-defined rectangle. 4. A Gaussians **Mixture Model (GMM)** models the foreground and back dabela undefined pixels as probable background and foregrounds. 5. Each pixel in the image is virtually connected to the surrounding pixels through virtu edges, and each edge gets a probability of being foreground or background, based on how similar it is in color to the pixels surrounding it. 6. Each pixel (or node as it is conceptualized in the algorithm) is connected to either a foreground or a background node.

7. after the nodes have been connected to either terminal (background or foreground, also called a source and sink), the edges between nodes belonging to different terminals are cut (the famous cut part of the algorithm), which enables the separation of the parts of the image

Image segmentation with the Watershed algorithm :-

Finally, we take a quick look at the Watershed algorithm. The algorithm is called Watershed, because its conceptualization involves water. Imagine areas with low density (little to no change) in an image as valleys, and areas with high density (lots of change) as peaks. Start filling the valleys with water to the point where water from two different valleys is about to merge. To prevent the merging of water from different valleys, you build a barrier to keep them separated. The resulting barrier is the image segmentation.

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.axis('off')

#Image loading
img = cv2.imread("image.png")

#image grayscale conversion
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
# 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_RECT, (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
```



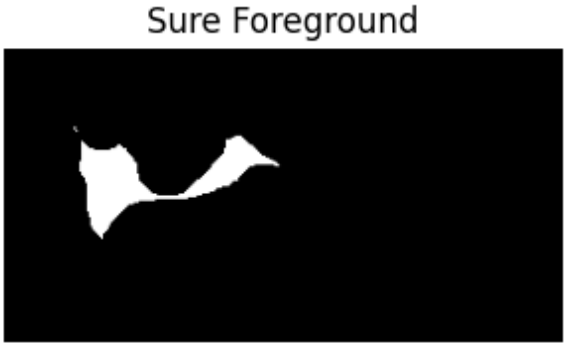
```
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(8, 8))
# sure background area
sure_bg = cv2.dilate(bin_img, kernel, iterations=3)
imshow(sure_bg, axes[0,0])
axes[0, 0].set_title('Sure Background')
# Distance transform
dist = cv2.distanceTransform(bin_img, cv2.DIST_L2, 5)
imshow(dist, axes[0,1])
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 = cv2.subtract(sure_bg, sure_fg)
imshow(unknown, axes[1,1])
axes[1, 1].set_title('Unknown')
plt.show()
```

Input Image:

Vidyavardhini's College of Engineering & Technology

Department of Computer Engineering

Output:



Conclusion:

In conclusion, this code demonstrates the application of the watershed algorithm for image segmentation. The watershed algorithm is a powerful tool for partitioning an image into distinct regions based on intensity and spatial cues. By converting the image to grayscale, applying thresholding, and performing morphological operations, it prepares the image for segmentation. The watershed algorithm then identifies areas of interest, such as "Sure Background" and "Sure Foreground," which are crucial for object segmentation. This example showcases how the watershed algorithm can be used effectively to segment objects within an image, providing a valuable technique for various computer vision and image processing applications.