Jawaban LKP 7
Mata Kuliah Pengolahan Citra Digital

Nama : Muhammad Fakhri Alauddin Hidayat
NIM : G64170015
Hari : Kamis
Tanggal : 5 Maret 2020
Waktu : 08.00 - 10.00 WIB

Soal

Pelajari dan jalankan code Python mengenai image segmentation di bawah ini. Apa keluaran dari program tersebut? Jelaskan cara kerja image segmentation pada code tersebut.

```
"""
=================================================
Comparing edge-based and region-based segmentation
=================================================
In this example, we will see how to segment objects from a background. We use
the ``coins`` image from ``skimage.data``, which shows several coins outlined
against a darker background.
"""
import numpy as np
import matplotlib.pyplot as plt
from skimage import data
from skimage.exposure import histogram
coins = data.coins()
hist, hist_centers = histogram(coins)
fig, axes = plt.subplots(1, 2, figsize=(8, 3))
axes[0].imshow(coins, cmap=plt.cm.gray)
axes[0].axis('off')
axes[1].plot(hist_centers, hist, lw=2)
axes[1].set_title('histogram of gray values')
##########################################################
#########
#
# Thresholding
# ============
#
# A simple way to segment the coins is to choose a threshold based on the
# histogram of gray values. Unfortunately, thresholding this image gives a
# binary image that either misses significant parts of the coins or merges
```

```
# parts of the background with the coins:
fig, axes = plt.subplots(1, 2, figsize=(8, 3), sharey=True)
axes[0].imshow(coins > 100, cmap=plt.cm.gray)
axes[0].set_title('coins > 100')
axes[1].imshow(coins > 150, cmap=plt.cm.gray)
axes[1].set_title('coins > 150')
for a in axes:
a.axis('off')
plt.tight_layout()
##############################################################
#########
# Edge-based segmentation
# ======================
#
# Next, we try to delineate the contours of the coins using edge-based
# segmentation. To do this, we first get the edges of features using the
# Canny edge-detector.
from skimage.feature import canny
edges = canny(coins)
fig, ax = plt.subplots(figsize=(4, 3))
ax.imshow(edges, cmap=plt.cm.gray)
ax.set_title('Canny detector')
ax.axis('off')
##############################################################
#########
# These contours are then filled using mathematical morphology.
from scipy import ndimage as ndi
fill_coins = ndi.binary_fill_holes(edges)
fig, ax = plt.subplots(figsize=(4, 3))
ax.imshow(fill_coins, cmap=plt.cm.gray)
ax.set_title('filling the holes')
ax.axis('off')
##############################################################
#########
# Small spurious objects are easily removed by setting a minimum size for
# valid objects.
from skimage import morphology
coins_cleaned = morphology.remove_small_objects(fill_coins, 21)
fig, ax = plt.subplots(figsize=(4, 3))
ax.imshow(coins_cleaned, cmap=plt.cm.gray)
ax.set_title('removing small objects')
ax.axis('off')
##############################################################
#########
```

```
# However, this method is not very robust, since contours that are not
# perfectly closed are not filled correctly, as is the case for one unfilled
# coin above.
#
# Region-based segmentation
# ========================
#
# We therefore try a region-based method using the watershed transform.
# First, we find an elevation map using the Sobel gradient of the image.
from skimage.filters import sobel
elevation_map = sobel(coins)
fig, ax = plt.subplots(figsize=(4, 3))
ax.imshow(elevation_map, cmap=plt.cm.gray)
ax.set_title('elevation map')
ax.axis('off')
############################################################
#########
# Next we find markers of the background and the coins based on the extreme
# parts of the histogram of gray values.
markers = np.zeros_like(coins)
markers[coins < 30] = 1
markers[coins > 150] = 2
fig, ax = plt.subplots(figsize=(4, 3))
ax.imshow(markers, cmap=plt.cm.nipy_spectral)
ax.set_title('markers')
ax.axis('off')

############################################################
#########
# Finally, we use the watershed transform to fill regions of the elevation
# map starting from the markers determined above:
segmentation = morphology.watershed(elevation_map, markers)
fig, ax = plt.subplots(figsize=(4, 3))
ax.imshow(segmentation, cmap=plt.cm.gray)
ax.set_title('segmentation')
ax.axis('off')
############################################################
#########
# This last method works even better, and the coins can be segmented and
# labeled individually.
from skimage.color import label2rgb
segmentation = ndi.binary_fill_holes(segmentation - 1)
labeled_coins, _ = ndi.label(segmentation)
image_label_overlay = label2rgb(labeled_coins, image=coins)
```

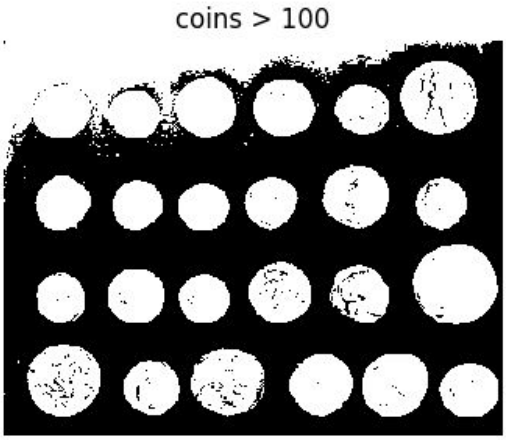
```
fig, axes = plt.subplots(1, 2, figsize=(8, 3), sharey=True)
axes[0].imshow(coins, cmap=plt.cm.gray)
axes[0].contour(segmentation, [0.5], linewidths=1.2, colors='y')
axes[1].imshow(image_label_overlay)
for a in axes:
a.axis('off')
plt.tight_layout()
plt.show()
```

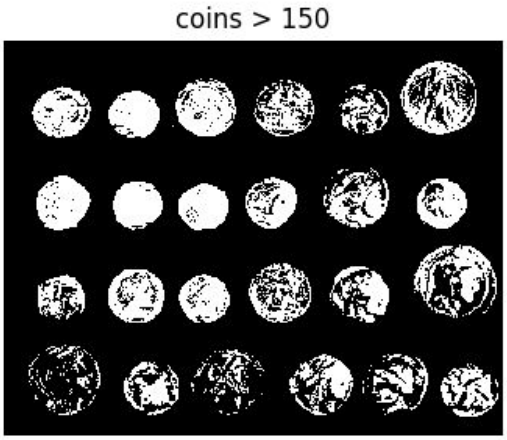
Jawaban

Kode diatas berusaha melakukan *image segmentation* untuk memisahkan objek dari latar belakangnya. Ada tiga cara yang dicoba pada kode diatas yaitu *thresholding*, *edge-based segmentation* dan *region-based segmentation*.

Thresholding

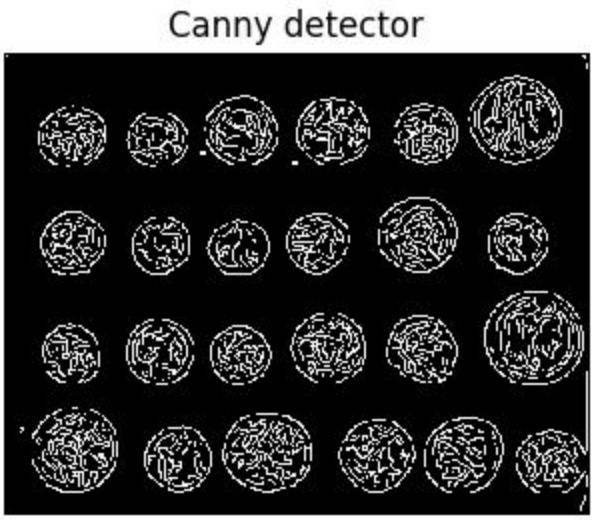
Pada *thresholding*, dipilih sebuah *threshold* (ambang batas) untuk memisahkan objek dan latar belakangnya pada histogram nilai abu-abu. Akan tetapi, hasil dari *thresholding* kurang bagus karena kemungkinan ada informasi yang hilang atau bahkan objek melebur dengan latar belakangnya.



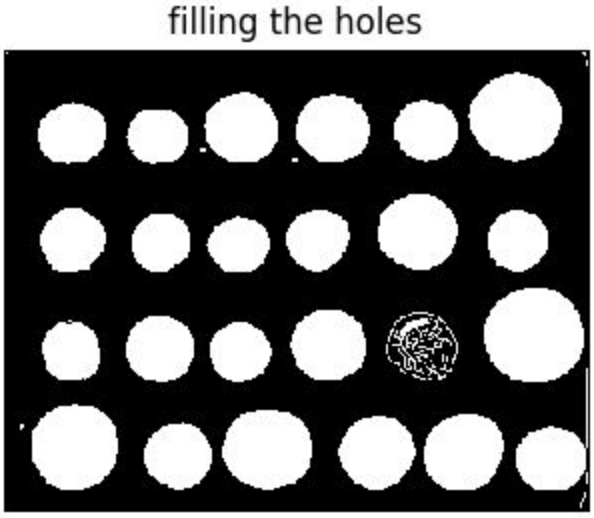


Edge-based Segmentation

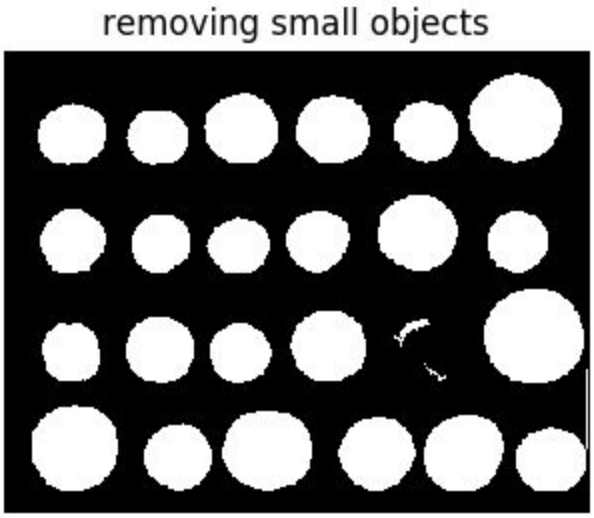
Selanjutnya adalah *edge-based segmentation*, teknik ini menggunakan *edge*(tepi) sebagai patokan untuk menentukan dimana objek berada. Pada kode diatas, digunakan *Canny Edge-detector* untuk mendeteksi edge dari setiap objek pada gambar.



Setelah itu, setiap daerah yang berada di dalam tepi akan diisi dengan warna putih melalui teknik morfologi matematika.



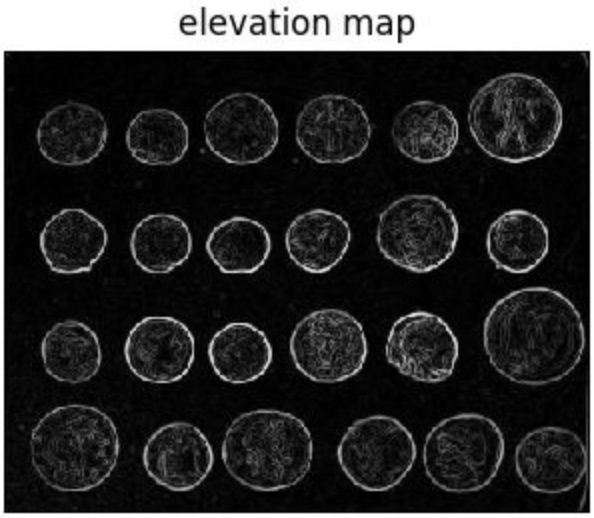
Jika dilihat diatas, masih terdapat beberapa objek kecil yang bukan merupakan objek yang seharusnya. Objek-objek palsu tersebut kemudian dihilangkan dengan cara menentukan ukuran minimal untuk objek yang benar.



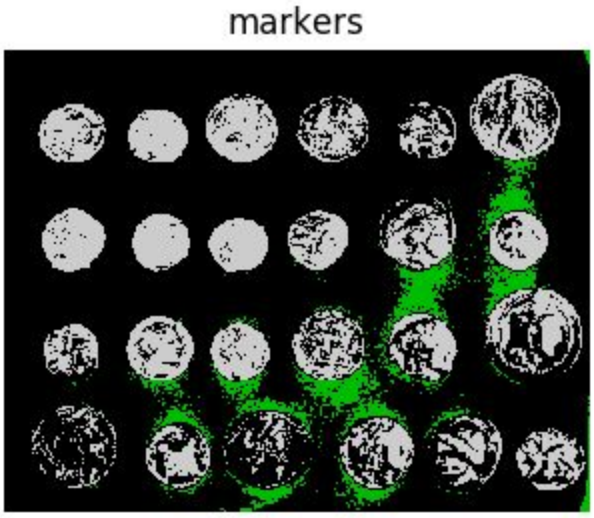
Diatas adalah hasil akhir pemisahan objek dari latar belakangnya dengan menggunakan metode *edge-based segmentation*. Jika dilihat lagi, ternyata ada objek yang tidak terdeteksi dengan benar. Oleh karena itu, selanjutnya akan dicoba metode *region-based segmentation*

Region-based Segmentation

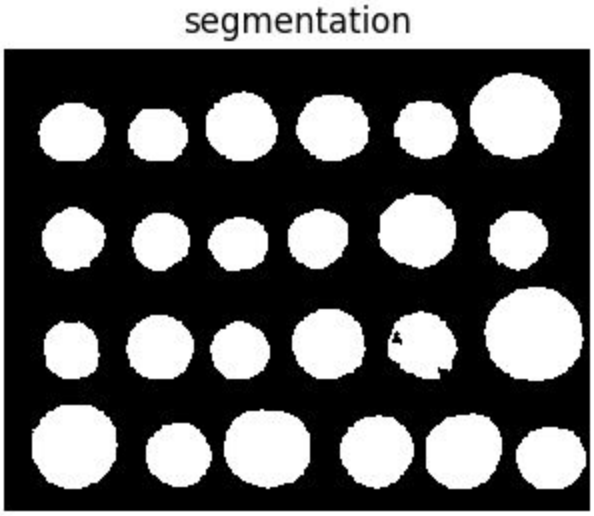
Teknik yang akan digunakan untuk melakukan *region-based segmentation* pada kode ini adalah watershed transform. Pertama, dicari *elevation map* dari gambar menggunakan *Sobel Gradient*.



Kemudian, dicari penanda untuk bagian latar belakang dan objek berdasarkan bagian ekstrim dari histogram nilai keabuan.



Selanjutnya, digunakan *Watershed Transform* untuk mengisi daerah dari *elevation map* dengan bantuan penanda yang sebelumnya sudah dicari.



Diatas adalah hasil akhir pemisahan objek dari latar belakangnya dengan menggunakan metode *region-based segmentation*. Selanjutnya dilakukan pelabelan hasil pemisahan pada gambar untuk melihat posisinya di gambar original.

Berdasarkan hasil akhir pemisahan objek dari latar belakangnya dengan menggunakan metode *edge-based segmentation* dan *region-based segmentation*, terlihat bahwa *region-based segmentation* menghasilkan pemisahan yang lebih baik dibandingkan dengan *edge-based segmentation*.