UJIAN AKHIR SEMESTER
MATA KULIAH
PENGOLAHAN GAMBAR

Topik
Segmentasi Objek

PENYUSUN LAPORAN

Nama Mahasiswa	NIM	Kelas
KESYA AULIA RAHMADANI	062340833189	1MIN

PROGRAM STUDI MANAJEMEN INFORMATIKA
JURUSAN MANAJEMEN INFORMATIKA
POLITEKNIK NEGERI SRIWIJAYA

2023

UAS PENGOLAHAN GAMBAR-KESYA AULIA RAHMADANI

Semester	: 1	Tanggal	: 11 Januari 2023
Dosen Pengampu	: Sulistiyanto, MTI	Kelas	: 1MIN

CP : Mahasiswa dapat melakukan segmentasi objek

Nama Mahasiswa : KESYA AULIA RAHMADANI
NIM : 062340833189

TUJUAN

1. Dapat mengenal dan memahami tentang pengolahan citra digital berupa deteksi tepi, segmentasi pada objek.
2. Dapat menjelaskan cara kerja fungsi program dari deteksi tepi dan segmentasi objek.

ALAT DAN BAHAN (HW & SW)

Hardware : Monitor, Keyboard, Maouse

Software : Google Colab, Phyton

HASIL

1. Kode dan Hasil Program dari Citra Wajib **"pisang.jpg"**

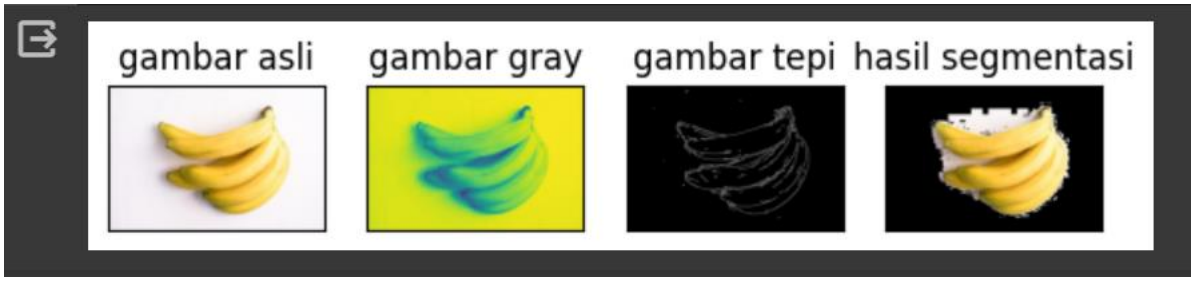
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread("pisang.jpg")
edge = cv2.Canny(img,50,100) #melakukan deteksi tepi

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #menentukan jenis warna
lower = np.array([0, 0, 0]) #menentukan ambang batas bawah warna
higher = np.array([100, 400, 255]) #menentukan ambang batas atas warna
mask = cv2.inRange(hsv, lower, higher) #membuat masking untuk mencari objek sesuai batas warna

result = cv2.bitwise_and(img, img, mask=mask)

#proses untuk menampilkan gambar
plt.subplot(141), plt.imshow(cv2.cvtColor (img, cv2.COLOR_BGR2RGB))
plt.title("gambar asli"),plt.xticks([]),plt.yticks([]) #menampilkan gambar asli
plt.subplot(142), plt.imshow (cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))
plt.title("gambar gray"),plt.xticks([]),plt.yticks([]) #menampilkan gambar abu
plt.subplot(143),plt.imshow(edge, cmap="gray")
plt.title("gambar tepi"),plt.xticks([]),plt.yticks([]) #menampilkan deteksi tepi
plt.subplot(144), plt.imshow(cv2.cvtColor (result, cv2.COLOR_BGR2RGB))
plt.title("hasil segmentasi"),plt.xticks([]),plt.yticks([]) #menampilkan segmentasi
plt.show()
```



2. Kode dan Hasil Program dari Citra Wajib **"lemon.jpg"**

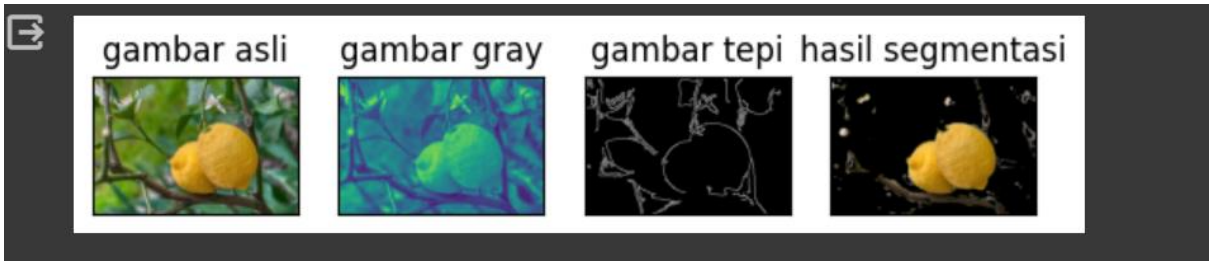
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread("lemon.jpg")
edge = cv2.Canny(img,100,500) #melakukan deteksi tepi

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #menentukan jenis warna
lower = np.array([0, 0, 0]) #menentukan ambang batas warna
higher = np.array([30, 255, 255]) #menentukan ambang
mask = cv2.inRange(hsv, lower, higher) #membuat masking untuk mencari objek sesuai batas warna

result = cv2.bitwise_and(img, img, mask=mask)

#proses untuk menampilkan gambar
plt.subplot(141), plt.imshow(cv2.cvtColor (img, cv2.COLOR_BGR2RGB))
plt.title("gambar asli"),plt.xticks([]),plt.yticks([]) #menampilkan gambar asli
plt.subplot(142), plt.imshow (cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))
plt.title("gambar gray"),plt.xticks([]),plt.yticks([]) #menampilkan gambar abu
plt.subplot(143),plt.imshow(edge, cmap="gray")
plt.title("gambar tepi"),plt.xticks([]),plt.yticks([]) #menampilkan deteksi tepi
plt.subplot(144), plt.imshow(cv2.cvtColor (result, cv2.COLOR_BGR2RGB))
plt.title("hasil segmentasi"),plt.xticks([]),plt.yticks([]) #menampilkan segmentasi
plt.show()
```



3. Kode dan Hasil Program dari Citra Wajib **"apel.jpg"**

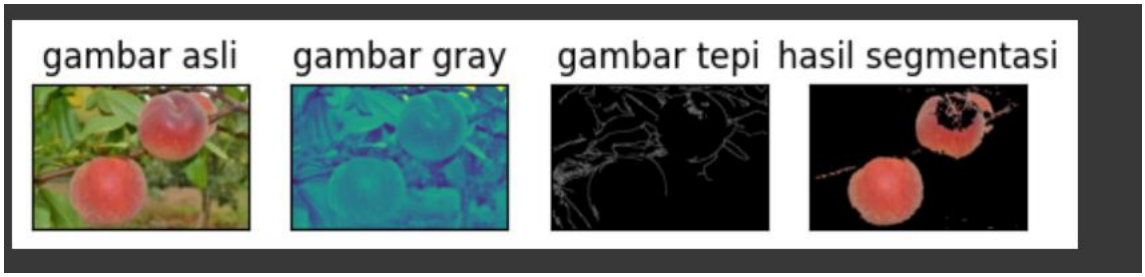
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread("apel.jpg")
edge = cv2.Canny(img,50,300) #melakukan deteksi tepi

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #menentukan jenis warna
lower = np.array([0, 50, 20]) #menentukan ambang batas bawah warna
higher = np.array([10, 200, 255]) #menentukan ambang batas atas
mask = cv2.inRange(hsv, lower, higher) #membuat masking untuk mencari objek sesuai batas warna

result = cv2.bitwise_and(img, img, mask=mask)

#proses untuk menampilkan gambar
plt.subplot(141), plt.imshow(cv2.cvtColor (img, cv2.COLOR_BGR2RGB))
plt.title("gambar asli"),plt.xticks([]),plt.yticks([]) #menampilkan gambar asli
plt.subplot(142), plt.imshow (cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))
plt.title("gambar gray"),plt.xticks([]),plt.yticks([]) #menampilkan gambar abu
plt.subplot(143),plt.imshow(edge, cmap="gray")
plt.title("gambar tepi"),plt.xticks([]),plt.yticks([]) #menampilkan deteksi tepi
plt.subplot(144), plt.imshow(cv2.cvtColor (result, cv2.COLOR_BGR2RGB))
plt.title("hasil segmentasi"),plt.xticks([]),plt.yticks([]) #menampilkan segmentasi
plt.show()
```



4. Kode dan Hasil Program dari Citra Tambahan **“buah.jpg”**

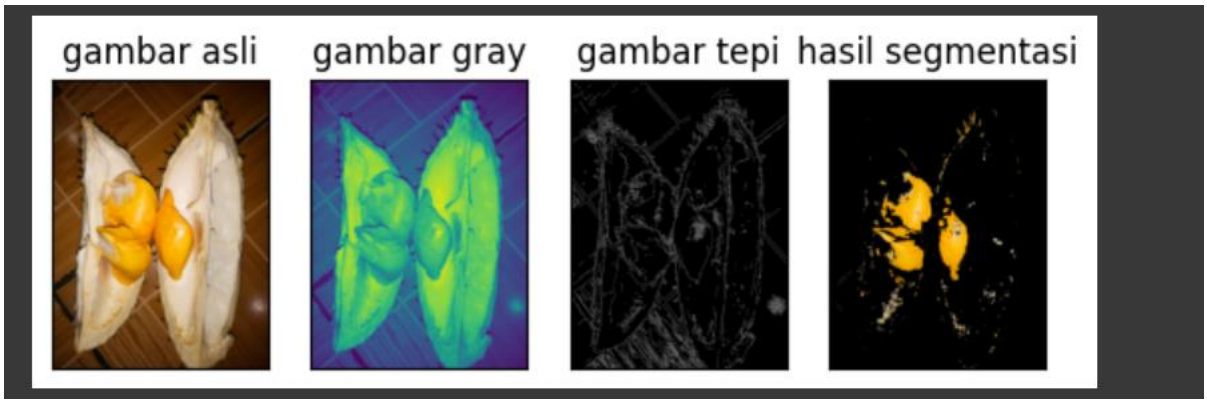
```
import cv2
import numpy as np
import matplotlib.pyplot as plt

img = cv2.imread("buah.jpg")
edge = cv2.Canny(img,50,100) #melakukan deteksi tepi

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #menentukan jenis warna
lower = np.array([20, 50, 20]) #menentukan ambang batas bawah warna
higher = np.array([100, 255, 255]) #menentukan ambang batas atas
mask = cv2.inRange(hsv, lower, higher) #membuat masking untuk mencari objek sesuai batas warna

result = cv2.bitwise_and(img, img, mask=mask)

#proses untuk menampilkan gambar
plt.subplot(141), plt.imshow(cv2.cvtColor (img, cv2.COLOR_BGR2RGB))
plt.title("gambar asli"),plt.xticks([]),plt.yticks([]) #menampilkan gambar asli
plt.subplot(142), plt.imshow (cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))
plt.title("gambar gray"),plt.xticks([]),plt.yticks([]) #menampilkan gambar abu
plt.subplot(143),plt.imshow(edge, cmap="gray")
plt.title("gambar tepi"),plt.xticks([]),plt.yticks([]) #menampilkan deteksi tepi
plt.subplot(144), plt.imshow(cv2.cvtColor (result, cv2.COLOR_BGR2RGB))
plt.title("hasil segmentasi"),plt.xticks([]),plt.yticks([]) #menampilkan segmentasi
plt.show()
```



5. Kode dan Hasil Program dari Citra Tambahan **“bunga.jpg”**

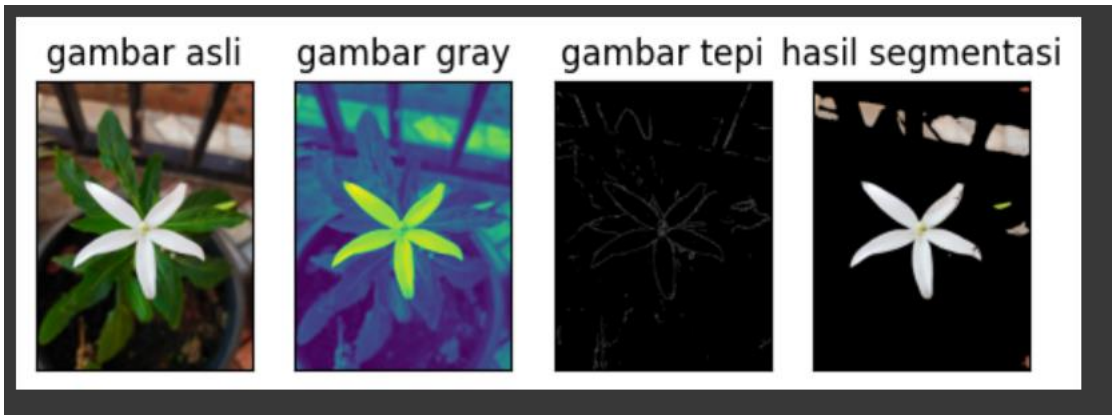
```
[25] import cv2
     import numpy as np
     import matplotlib.pyplot as plt

     img = cv2.imread("bunga.jpg")
     edge = cv2.Canny(img,30,70) #melakukan deteksi tepi

     hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #menentukan jenis warna
     lower = np.array([0, 0, 168]) #menentukan ambang batas bawah warna
     higher = np.array([172, 255, 255]) #menentukan ambang batas atas
     mask = cv2.inRange(hsv, lower, higher) #membuat masking untuk mencari objek sesuai batas warna

     result = cv2.bitwise_and(img, img, mask=mask)

     #proses untuk menampilkan gambar
     plt.subplot(141), plt.imshow(cv2.cvtColor (img, cv2.COLOR_BGR2RGB))
     plt.title("gambar asli"),plt.xticks([]),plt.yticks([]) #menampilkan gambar asli
     plt.subplot(142), plt.imshow (cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))
     plt.title("gambar gray"),plt.xticks([]),plt.yticks([]) #menampilkan gambar abu
     plt.subplot(143),plt.imshow(edge, cmap="gray")
     plt.title("gambar tepi"),plt.xticks([]),plt.yticks([]) #menampilkan deteksi tepi
     plt.subplot(144), plt.imshow(cv2.cvtColor (result, cv2.COLOR_BGR2RGB))
     plt.title("hasil segmentasi"),plt.xticks([]),plt.yticks([]) #menampilkan segmentasi
     plt.show()
```



PEMBAHASAN

Deteksi Tepi, proses yang berfungsi untuk mendeteksi garis tepi yang membatasi dua wilayah citra. Deteksi tepi yang digunakan adalah metode canny dengan perintah sebagai berikut.

```
edge = cv2.Canny(img,50,100)
```

Untuk ukuran tepi dapat disesuaikan dengan masing-masing gambar yang dapat menentukan seberapa banyak garis tepi yang dihasilkan dalam gambar tersebut. Dalam deteksi tepi gambar telah diubah warna menjadi abu-abu.

Segmentasi Objek, setelah dilakukan deteksi tepi barulah proses segmentasi objek muncul. Segmentasi objek ini dilakukan untuk menghilangkan latar belakang atau bagain yang tidak diinginkan didalam gambar. Dalam melakukan segmentasi objek digunakan beberapa perintah diantarnya :

```
hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV) #menentukan jenis warna
```

Ruang warna HSV (Hue, Saturation, Value): Untuk segmentasi warna dan pengenalan objek berbasis warna berupa rona, saturasi, dan nilainya. Dapat dilakukan dengan menggunakan cv2.cvtColor() dan menggunakan flag cv2.COLOR_BGR2HSV untuk mengonversi ruang warna dari BGR ke HSV.
Tentukan rentang warna yang ingin deteksi.

```
lower = np.array([0, 0, 0]) #menentukan ambang batas warna
higher = np.array([30, 255, 255]) #menentukan ambang
```

Dengan perintah tersebut dapat menentukan batas bawah dan atas ruang warna. Kita dalam mengubah nilai-nilai ini untuk mendeteksi warna lain yang diinginkan. Nilai tersebut dapat digunakan untuk membuat mask. Mask digunakan untuk memisahkan gambar menjadi dua wilayah atau lebih.

```
mask = cv2.inRange(hsv, lower, higher)
```

Fungsi ini mengambil gambar HSV dan batas bawah dan atas rentang warna sebagai masukan. Untuk menerapkan fungsi mask dapat menggunakan cv2.bitwise_and(), yang menampilkan area yang berisikan nilai atau beririsan

```
result = cv2.bitwise_and(img, img, mask=mask)
```

Menampilkan gambar, untuk menampikan gambar kita gunakan perintah sebagai berikut :

```
plt.subplot(141), plt.imshow(cv2.cvtColor (img, cv2.COLOR_BGR2RGB))
plt.title("gambar asli"),plt.xticks([]),plt.yticks([]) #menampilkan
gambar asli
plt.subplot(142), plt.imshow (cv2.cvtColor(img, cv2.COLOR_BGR2GRAY))
plt.title("gambar gray"),plt.xticks([]),plt.yticks([]) #menampilkan
gambar abu
plt.subplot(143),plt.imshow(edge, cmap="gray")
plt.title("gambar tepi"),plt.xticks([]),plt.yticks([]) #menampilkan
deteksi tepi
plt.subplot(144), plt.imshow(cv2.cvtColor (result, cv2.COLOR_BGR2RGB))
plt.title("hasil segmentasi"),plt.xticks([]),plt.yticks([])
#menampilkan segmentasi
plt.show()
```

Hasil, menunjukan beberapa gambar berupa gambar asli, gambar yang telah diubah menjadi abu-abu, gambar yang telah mengalami deteksi tepi, dan gambar yang mengalami segmentasi objek dimana latar belakang dengan menentukan rentang warna sesuai dengan perintah yang telah dilakukan sebelumnya.

KESIMPULAN

Dalam laporan pengolahan gambar terdapat kode program yang berfungsi untuk mendeteksi tepi dan segmentasi objek. Deteksi tepi berfungsi untuk menandai bagian yang menjadi detail citra. Segmentasi objek berfungsi untuk memisahkan dua wilayah atau latar belakang dengan menggunakan rentang warna yang telah ditentukan nilai-nilainya. Selain itu, kode program menampikan gambar asli, gambar abu-abu, gambar garis tepi, dan gambar segmentasi secara bersamaan.