LAPORAN UJIAN AKHIR SEMESTER
MATA KULIAH
PENGOLAHAN CITRA DIGITAL

Topik

Deteksi Gambar & Segmentasi Gambar

PENYUSUN LAPORAN

Nama Mahasiswa	NIM	Kelas
Danny Agus Wijaya	062340833231	1 MIO

PROGRAM STUDI MANAJEMEN INFORMATIKA
JURUSAN MANAJEMEN INFORMATIKA
POLITEKNIK NEGERI SRIWIJAYA

2024

UAS Pengolahan Citra Digital 1MIO : Deteksi Gambar dan Segmentasi Gambar Lewat Python

Pertemuan	: UAS	Tanggal	: 10 Januari 2024
Semester	: 1	Kelas	: 1MIO
Dosen Pengampu	: Sulistiyanto, MTI	Tugas	: UAS

CP : Mahasiswa mampu mengerjakan tugas UAS dengan baik dan benar.

Nama Mahasiswa : Danny Agus Wijaya
NIM 062340833231

TUJUAN

1. Membuat program Python untuk menampilkan Gambar asli, Citra deteksi tepi, dan Citra hasil segmentasi.

ALAT DAN BAHAN (HW & SW)

1. Laptop
2. Aplikasi Visual Studio Code, Bahasa Pemrograman Python, PIP Numpy,PIP Matplotlib, dan PIP Open Cv.

TUGAS

1. Setiap capture gambar, diberi penjelasan

a). Gambar 1.1 :

- Fungsi ini melakukan deteksi tepi pada gambar menggunakan metode Canny.
- Gambar diubah ke skala abu-abu untuk mempermudah deteksi tepi.
- Metode Canny digunakan untuk menemukan tepi dalam gambar.
- Hasil deteksi tepi dikonversi kembali ke mode warna RGB sebelum dikembalikan.

b). Gambar 1.2 :

- Fungsi ini melakukan segmentasi gambar dengan menghapus latar belakang berdasarkan warna kuning (pisang).
- Gambar diubah ke ruang warna HSV untuk lebih baik dalam menangkap warna.
- Batas warna kuning dalam format HSV ditentukan.
- Mask dibuat untuk menentukan area yang akan dipertahankan (warna kuning).
- Operasi bitwise digunakan untuk menghapus latar belakang berdasarkan mask.

c). Gambar 1.3 :

- Membaca gambar asli dari file dengan nama Gambar yang digunakan.

d). Gambar 1.4 :

- Memanggil fungsi deteksi tepi untuk mendapatkan gambar dengan tepi yang terdeteksi.

e). Gambar 1.5 :

- Memanggil fungsi penghapusan background untuk mendapatkan gambar dengan latar belakang yang dihapus berdasarkan warna kuning.

f). Gambar 1.6 :

- Membuat jendela dengan tiga subplot menggunakan Matplotlib.

- Menampilkan gambar asli, gambar dengan deteksi tepi, dan gambar dengan segmentasi dalam satu jendela.

- Subplot pertama menampilkan gambar asli, subplot kedua menampilkan gambar dengan deteksi tepi, dan subplot ketiga menampilkan gambar dengan segmentasi dan latar belakang yang dihapus.

g). Gambar 2.1 :

Gambar Kode dan Hasil Program dari Citra Wajib “apel.jpg”.

h). Gambar 2.2 :

Gambar Kode dan Hasil Program dari Citra Wajib “apel2.jpg”.

i). Gambar 2.3 :

Gambar Kode dan Hasil Program dari Citra Wajib “Foto- Almamater.jpg”.

j). Gambar 2.4 :

Gambar Kode dan Hasil Program dari Citra Tambahan “lemon.jpg”.

k). Gambar 2.5 :

Gambar Kode dan Hasil Program dari Citra Tambahan “pisang.jpg”.

2. Tulislah dokumentasi percobaan syntax python dalam mendeteksi tepian dan segmentasi pada gambar.
   a. Syntax untuk Medeteksi Gambar

```
def edge_detection(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 150)
    return cv2.cvtColor(edges, cv2.COLOR_GRAY2RGB)
```

Gambar 1.1

   b. Syntax untuk Men-Segmentasi Gambar

```
def remove_background(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 0, 0])  # Nilai batas bawah untuk warna merah dalam format HSV
    upper_red = np.array([10, 400, 255])  # Nilai batas atas untuk warna merah dalam format HSV
    mask = cv2.inRange(hsv, lower_red, upper_red)
    result = cv2.bitwise_and(image, image, mask=mask)
    return result
```

Gambar 1.2

   c. Membaca Gambar Asli

```
# Baca gambar asli
original_image = cv2.imread('opencv/pisang.jpg')
```

Gambar 1.3

d. Mendeteksi Gambar Asli

```
# Deteksi tepi pada gambar
edge_image = edge_detection(original_image)
```

Gambar 1.4

e. Men-Segmentasi Gambar Asli

```
# Hapus background pada gambar dengan segmentasi warna merah
segmented_image = remove_background(original_image)
```

Gambar 1.5

f. Syntax untuk Menampilkan 3 Gambar dalam Satu Jendela Output

```
# Tampilkan ketiga gambar dalam satu jendela
plt.figure(figsize=(10, 5))

# Gambar asli
plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
plt.title('Gambar Asli')
plt.axis('off')

# Gambar dengan deteksi tepi
plt.subplot(1, 3, 2)
plt.imshow(edge_image)
plt.title('Citra deteksi tepi')
plt.axis('off')

# Gambar dengan segmentasi warna merah (background dihapus)
plt.subplot(1, 3, 3)
plt.imshow(cv2.cvtColor(segmented_image, cv2.COLOR_BGR2RGB))
plt.title('Citra hasil segementasi')
plt.axis('off')

# Tampilkan jendela
plt.show()
```

Gambar 1.6

KODE & HASIL PROGRAM:

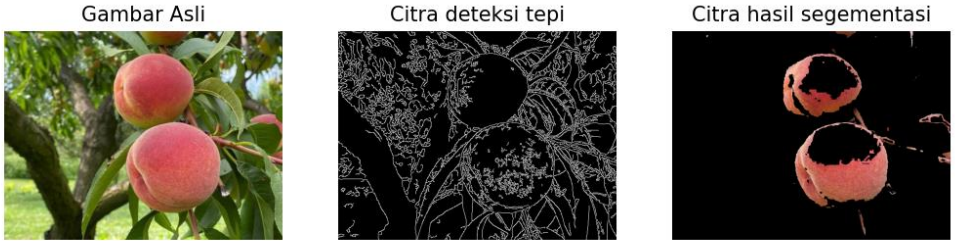
A. Kode dan Hasil Program dari Citra Wajib “apel.jpg” :

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

# Fungsi untuk melakukan deteksi tepi pada gambar
def edge_detection(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 150)
    return cv2.cvtColor(edges, cv2.COLOR_GRAY2RGB)

# Fungsi untuk melakukan segmentasi (hapus background) pada gambar
def remove_background(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 0, 0])  # Nilai batas bawah untuk warna merah dalam format HSV
    upper_red = np.array([10, 400, 255])  # Nilai batas atas untuk warna merah dalam format HSV
    mask = cv2.inRange(hsv, lower_red, upper_red)
    result = cv2.bitwise_and(image, image, mask=mask)
    return result

# Baca gambar asli
original_image = cv2.imread('opencv/apel.jpg')

# Deteksi tepi pada gambar
edge_image = edge_detection(original_image)

# Hapus background pada gambar dengan segmentasi warna merah
segmented_image = remove_background(original_image)

# Tampilkan ketiga gambar dalam satu jendela
plt.figure(figsize=(10, 5))

# Gambar asli
plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
plt.title('Gambar Asli')
plt.axis('off')

# Gambar dengan deteksi tepi
plt.subplot(1, 3, 2)
plt.imshow(edge_image)
plt.title('Citra deteksi tepi')
plt.axis('off')

# Gambar dengan segmentasi warna merah (background dihapus)
plt.subplot(1, 3, 3)
plt.imshow(cv2.cvtColor(segmented_image, cv2.COLOR_BGR2RGB))
plt.title('Citra hasil segementasi')
plt.axis('off')

# Tampilkan jendela
plt.show()
```

Gambar 2.1

B. Kode dan Hasil Program dari Citra Wajib “apel2.jpg” :

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

# Fungsi untuk melakukan deteksi tepi pada gambar
def edge_detection(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 150)
    return cv2.cvtColor(edges, cv2.COLOR_GRAY2RGB)

# Fungsi untuk melakukan segmentasi (hapus background) pada gambar
def remove_background(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 0, 0])  # Nilai batas bawah untuk warna merah dalam format HSV
    upper_red = np.array([10, 400, 255])  # Nilai batas atas untuk warna merah dalam format HSV
    mask = cv2.inRange(hsv, lower_red, upper_red)
    result = cv2.bitwise_and(image, image, mask=mask)
    return result

# Baca gambar asli
original_image = cv2.imread('opencv/apel2.jpg')

# Deteksi tepi pada gambar
edge_image = edge_detection(original_image)

# Hapus background pada gambar dengan segmentasi warna merah
segmented_image = remove_background(original_image)

# Tampilkan ketiga gambar dalam satu jendela
plt.figure(figsize=(10, 5))

# Gambar asli
plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
plt.title('Gambar Asli')
plt.axis('off')

# Gambar dengan deteksi tepi
plt.subplot(1, 3, 2)
plt.imshow(edge_image)
plt.title('Citra deteksi tepi')
plt.axis('off')

# Gambar dengan segmentasi warna merah (background dihapus)
plt.subplot(1, 3, 3)
plt.imshow(cv2.cvtColor(segmented_image, cv2.COLOR_BGR2RGB))
plt.title('Citra hasil segementasi')
plt.axis('off')

# Tampilkan jendela
plt.show()
```



Gambar 2.2

C. Kode dan Hasil Program dari Citra Wajib “Foto-Almamater.jpg”:

```python
def edge_detection(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 10, 20)
    return cv2.cvtColor(edges, cv2.COLOR_GRAY2RGB)

# Fungsi untuk melakukan segmentasi (hapus background) pada gambar
def remove_background(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_red = np.array([0, 0, 0])  # Nilai batas bawah untuk warna merah dalam format HSV
    upper_red = np.array([180, 145, 300])  # Nilai batas atas untuk warna merah dalam format HSV
    mask = cv2.inRange(hsv, lower_red, upper_red)
    result = cv2.bitwise_and(image, image, mask=mask)
    return result

# Baca gambar asli
original_image = cv2.imread('opencv/Foto-Almamater.jpg')

# Deteksi tepi pada gambar
edge_image = edge_detection(original_image)

# Hapus background pada gambar dengan segmentasi warna merah
segmented_image = remove_background(original_image)

# Tampilkan ketiga gambar dalam satu jendela
plt.figure(figsize=(10, 5))

# Gambar asli
plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
plt.title('Gambar Asli')
plt.axis('off')

# Gambar dengan deteksi tepi
plt.subplot(1, 3, 2)
plt.imshow(edge_image)
plt.title('Citra deteksi tepi')
plt.axis('off')

# Gambar dengan segmentasi warna merah (background dihapus)
plt.subplot(1, 3, 3)
plt.imshow(cv2.cvtColor(segmented_image, cv2.COLOR_BGR2RGB))
plt.title('Citra hasil segementasi')
plt.axis('off')

# Tampilkan jendela
plt.show()
```

Gambar 2.3

D. Kode dan Hasil Program dari Citra Tambahan “lemon.jpg”:

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

# Fungsi untuk melakukan deteksi tepi pada gambar
def edge_detection(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 150)
    return cv2.cvtColor(edges, cv2.COLOR_GRAY2RGB)

# Fungsi untuk melakukan segmentasi (hapus background) pada gambar
def remove_background(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_red = np.array([20, 100, 100])  # Nilai batas bawah untuk warna merah dalam format HSV
    upper_red = np.array([30, 255, 255])  # Nilai batas atas untuk warna merah dalam format HSV
    mask = cv2.inRange(hsv, lower_red, upper_red)
    result = cv2.bitwise_and(image, image, mask=mask)
    return result

# Baca gambar asli
original_image = cv2.imread('opencv/lemon.jpg')

# Deteksi tepi pada gambar
edge_image = edge_detection(original_image)

# Hapus background pada gambar dengan segmentasi warna merah
segmented_image = remove_background(original_image)

# Tampilkan ketiga gambar dalam satu jendela
plt.figure(figsize=(10, 5))

# Gambar asli
plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
plt.title('Gambar Asli')
plt.axis('off')

# Gambar dengan deteksi tepi
plt.subplot(1, 3, 2)
plt.imshow(edge_image)
plt.title('Citra deteksi tepi')
plt.axis('off')

# Gambar dengan segmentasi warna merah (background dihapus)
plt.subplot(1, 3, 3)
plt.imshow(cv2.cvtColor(segmented_image, cv2.COLOR_BGR2RGB))
plt.title('Citra hasil segementasi')
plt.axis('off')

# Tampilkan jendela
plt.show()
```

Gambar 2.4

E. Kode dan Hasil Program dari Citra Tambahan “pisang.jpg”:

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

# Fungsi untuk melakukan deteksi tepi pada gambar
def edge_detection(image):
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    edges = cv2.Canny(gray, 50, 150)
    return cv2.cvtColor(edges, cv2.COLOR_GRAY2RGB)

# Fungsi untuk melakukan segmentasi (hapus background) pada gambar
def remove_background(image):
    hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    lower_red = np.array([20, 100, 100])  # Nilai batas bawah untuk warna merah dalam format HSV
    upper_red = np.array([30, 255, 255])  # Nilai batas atas untuk warna merah dalam format HSV
    mask = cv2.inRange(hsv, lower_red, upper_red)
    result = cv2.bitwise_and(image, image, mask=mask)
    return result

# Baca gambar asli
original_image = cv2.imread('opencv/pisang.jpg')

# Deteksi tepi pada gambar
edge_image = edge_detection(original_image)

# Hapus background pada gambar dengan segmentasi warna merah
segmented_image = remove_background(original_image)

# Tampilkan ketiga gambar dalam satu jendela
plt.figure(figsize=(10, 5))

# Gambar asli
plt.subplot(1, 3, 1)
plt.imshow(cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB))
plt.title('Gambar Asli')
plt.axis('off')

# Gambar dengan deteksi tepi
plt.subplot(1, 3, 2)
plt.imshow(edge_image)
plt.title('Citra deteksi tepi')
plt.axis('off')

# Gambar dengan segmentasi warna merah (background dihapus)
plt.subplot(1, 3, 3)
plt.imshow(cv2.cvtColor(segmented_image, cv2.COLOR_BGR2RGB))
plt.title('Citra hasil segementasi')
plt.axis('off')

# Tampilkan jendela
plt.show()
```

Gambar 2.5

Analisis Cara Kerja Fungsi Program/Algoritma:

Deteksi Tepi (Fungsi `edge_detection`):

- Fungsi menerima input gambar (`image`).
- Gambar diubah ke skala abu-abu menggunakan `cv2.cvtColor`.
- Deteksi tepi dilakukan menggunakan metode Canny dengan nilai ambang batas 50 dan 150.
- Hasil deteksi tepi dalam skala abu-abu dikonversi kembali ke mode warna RGB menggunakan `cv2.cvtColor`.
- Gambar dengan tepi yang terdeteksi dikembalikan sebagai output.

Segmentasi Penghapusan Background (Fungsi `remove_background`):

- Fungsi menerima input gambar (`image`).
- Gambar diubah ke ruang warna HSV menggunakan `cv2.cvtColor`.
- Nilai batas bawah dan atas untuk warna dari latar belakang yang di hapus dalam format HSV ditentukan.
- Mask dibuat menggunakan `cv2.inRange` untuk menentukan area yang akan dipertahankan.
- Operasi bitwise (`cv2.bitwise_and`) digunakan untuk menghapus latar belakang berdasarkan mask.
- Gambar hasil segmentasi dikembalikan sebagai output.

Baca Gambar Asli:

- Gambar asli dibaca menggunakan `cv2.imread` dengan nama file(harus disesuaikan dengan nama file/gambar yang yang digunakan).

Deteksi Tepi pada Gambar Asli:

- Fungsi deteksi tepi (`edge_detection`) dipanggil untuk mendapatkan gambar dengan tepi yang terdeteksi.

Penghapusan Background pada Gambar Asli:

- Fungsi penghapusan background (`remove_background`) dipanggil untuk mendapatkan gambar dengan latar belakang yang dihapus.

Tampilkan Ketiga Gambar dalam Satu Jendela:

- Jendela dengan tiga subplot dibuat menggunakan `plt.figure`.
- Subplot pertama menampilkan gambar asli, subplot kedua menampilkan gambar dengan deteksi tepi, dan subplot ketiga menampilkan gambar dengan segmentasi dan latar belakang yang dihapus.
- Hasilnya ditampilkan menggunakan `plt.show`.

Hasil dari Kode Program:

- Subplot pertama menampilkan gambar asli.
- Subplot kedua menampilkan gambar dengan deteksi tepi menggunakan metode Canny.
- Subplot ketiga menampilkan gambar dengan segmentasi, di mana latar belakang yang telah dihapus.

KESIMPULAN

Program ini menggunakan deteksi tepi dan segmentasi berbasis warna untuk menghasilkan tiga gambar dalam satu jendela: gambar asli, gambar dengan deteksi tepi, dan gambar segmentasi dengan latar belakang yang dihapus. Fungsi-fungsi tersebut memanfaatkan pustaka OpenCV untuk pemrosesan gambar dan Matplotlib untuk tampilan visual. Program ini dapat berguna dalam pemrosesan gambar untuk menyoroti objek tertentu atau menghapus latar belakang berdasarkan warna.