UJIAN AKHIR SEMESTER
MATA KULIAH
PRAK. PENGOLAHAN GAMBAR & FOTOGRAFI

Topik
SEGMENTASI CITRA

PENYUSUN LAPORAN

Nama Mahasiswa	NIM	Kelas
ALIFAH	062340833230	1 MIO

PROGRAM STUDI D4 MANAJEMEN INFORMATIKA
JURUSAN MANAJEMEN INFORMATIKA
POLITEKNIK NEGERI SRIWIJAYA
2024

UAS PRAK. PENGOLAHAN GAMBAR & FOTOGRAFI

Semester	: 1	Tanggal	: 08 Januari 2024
Dosen Pengampu	: Sulistiyanto, MTI	Kelas	: 1MIO

Nama Mahasiswa **:** ALIFAH
NIM **:** 062340833230

ALAT DAN BAHAN (HW & SW)

1. Laptop
2. Aplikasi Visual Studio Code
3. Bahasa Pemrograman Python

PROJECT

Penjelasan Kode

Fotografi > lancar.py

```python
import cv2 # pernyataan Python yang digunakan untuk mengimpor modul OpenCV
import numpy as np # pernyataan Python yang digunakan untuk mengimpor modul NumPy

# Untuk Membaca citra
nama_file_citra = 'Fotografi\cat3.jpg'  # Untuk nama file citra
citra = cv2.imread(nama_file_citra) # Digunakan untuk membaca citra dari file

# Untuk Deteksi tepi menggunakan metode Canny
tepi = cv2.Canny(citra, 50, 200)

# Untuk Mengonversi citra ke dalam format HSV
citra_hsv = cv2.cvtColor(citra, cv2.COLOR_BGR2HSV)

# Untuk Menentukan rentang warna latar belakang yang ingin dihapus
lower_latar_belakang = np.array([0, 0, 100])
upper_latar_belakang = np.array([300, 60, 300])

# Untuk Membuat mask untuk warna latar belakang
mask_latar_belakang = cv2.inRange(citra_hsv, lower_latar_belakang, upper_latar_belakang)

# Untuk Mengganti piksel latar belakang dengan warna hitam
citra_tanpa_latar = cv2.bitwise_and(citra, citra, mask=~mask_latar_belakang)

# Untuk Menampilkan citra asli, citra deteksi tepi, dan citra tanpa latar belakang
cv2.imshow('Citra Asli', citra)
cv2.imshow('Deteksi Tepi', tepi)
cv2.imshow('Tanpa Latar Belakang', citra_tanpa_latar)
cv2.waitKey(0)
cv2.destroyAllWindows()
# cv2.waitKey dan cv2.destroyAllWindows Digunakan untuk menunggu penekanan tombol dan menutup jendela setelah selesai.

```

1. **import cv2** adalah pernyataan Python yang digunakan untuk mengimpor modul OpenCV (Open Source Computer Vision) ke dalam program Anda. OpenCV adalah pustaka sumber terbuka yang menyediakan algoritma dan fungsi untuk pengolahan citra dan penglihatan komputer. Dengan mengimpor modul OpenCV menggunakan pernyataan **import cv2** , Anda dapat menggunakan fungsi-fungsi dan alat-alat yang disediakan oleh OpenCV untuk berbagai tugas pengolahan citra dan visi komputer.

2. **cv2.canny** adalah metode deteksi tepi yang disediakan oleh pustaka OpenCV, yang umumnya digunakan untuk pengolahan citra dan visi komputer. Fungsi ini menerapkan algoritma deteksi tepi Canny untuk mengidentifikasi tepi dalam citra.

3. **cv2.imread** digunakan untuk membaca citra dari file, dan cv2.imshow digunakan untuk menampilkan citra di jendela. **cv2.waitKey** dan **cv2.destroyAllWindows** digunakan untuk menunggu penekanan tombol dan menutup jendela setelah selesai.

4. Untuk mengonversi citra ke dalam format HSV (Hue, Saturation, Value), dapat menggunakan fungsi cv2.cvtColor dari pustaka OpenCV. **cv2.COLOR_BGR2HSV**: Parameter yang digunakan untuk menentukan konversi warna dari BGR (Blue, Green, Red) ke HSV. Fungsi ini akan mengembalikan citra yang telah dikonversi ke format HSV.

5. Untuk menentukan rentang warna latar belakang yang ingin dihapus dengan NumPy, perlu menentukan nilai **lower_latar_belakang** dan **upper_latar_belakang** yang mencakup rentang warna latar belakang yang diinginkan. Ini biasanya dilakukan berdasarkan nilai-nilai dalam format warna tertentu, seperti BGR atau HSV, tergantung pada citra asli.

6. Membuat mask untuk warna latar belakang dengan **cv2.inRange** melibatkan menentukan rentang warna yang diinginkan dan kemudian menerapkannya pada citra asli.

7. **cv2.bitwise_and** dapat digunakan untuk mengganti piksel latar belakang dengan warna hitam (nol) pada citra. Ini melibatkan operasi bitwise AND antara citra asli dan mask invers.

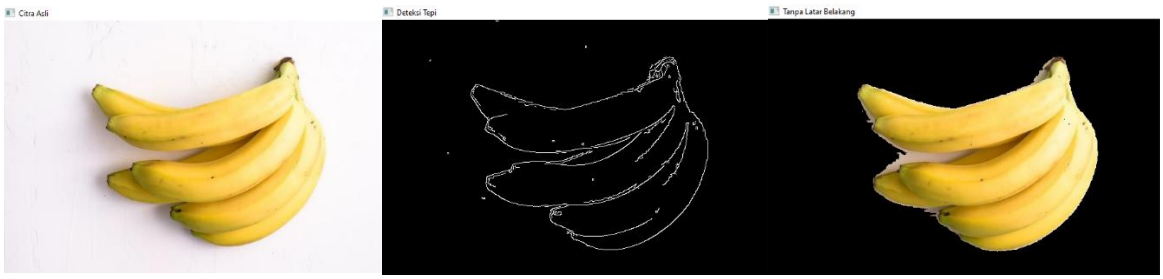
KODE & HASIL

1. Kode & Hasil Citra Input Wajib pisang.jpg

Fotografi > lancar.py

```python
import cv2
import numpy as np

# Membaca citra
nama_file_citra = 'Fotografi\pisang.jpg'  # Ganti dengan nama file citra Anda
citra = cv2.imread(nama_file_citra)

# Deteksi tepi menggunakan metode Canny
tepi = cv2.Canny(citra, 50, 150)

# Mengonversi citra ke dalam format HSV
citra_hsv = cv2.cvtColor(citra, cv2.COLOR_BGR2HSV)

# Menentukan rentang warna latar belakang yang ingin dihapus
lower_latar_belakang = np.array([0, 0, 0])
upper_latar_belakang = np.array([300, 48, 300])

# Membuat mask untuk warna latar belakang
mask_latar_belakang = cv2.inRange(citra_hsv, lower_latar_belakang, upper_latar_belakang)

# Mengganti piksel latar belakang dengan warna hitam
citra_tanpa_latar = cv2.bitwise_and(citra, citra, mask=~mask_latar_belakang)

# Menampilkan citra asli, citra deteksi tepi, dan citra tanpa latar belakang
cv2.imshow('Citra Asli', citra)
cv2.imshow('Deteksi Tepi', tepi)
cv2.imshow('Tanpa Latar Belakang', citra_tanpa_latar)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Gambar1.1 Kode Deteksi Citra



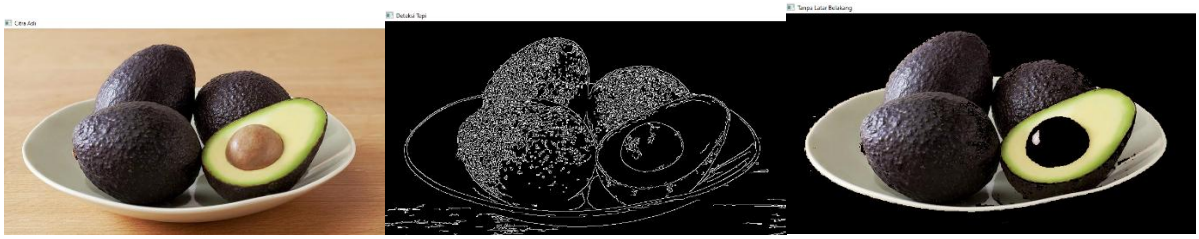
Gambar 1.2 Citra Asli & Hasil dari Citra Tepi dan Tanpa Latar Belakang

2. Kode & Hasil dari Citra Input wajib alpukat.jpg

Fotografi > lancar.py

```python
import cv2
import numpy as np

# Membaca citra
nama_file_citra = 'Fotografi\papukat.jpg'  # Ganti dengan nama file citra Anda
citra = cv2.imread(nama_file_citra)

# Deteksi tepi menggunakan metode Canny
tepi = cv2.Canny(citra, 50, 150)

# Mengonversi citra ke dalam format HSV
citra_hsv = cv2.cvtColor(citra, cv2.COLOR_BGR2HSV)

# Menentukan rentang warna latar belakang yang ingin dihapus
lower_latar_belakang = np.array([5, 55, 45])
upper_latar_belakang = np.array([20, 320, 250])

# Membuat mask untuk warna latar belakang
mask_latar_belakang = cv2.inRange(citra_hsv, lower_latar_belakang, upper_latar_belakang)

# Mengganti piksel latar belakang dengan warna hitam
citra_tanpa_latar = cv2.bitwise_and(citra, citra, mask=~mask_latar_belakang)

# Menampilkan citra asli, citra deteksi tepi, dan citra tanpa latar belakang
cv2.imshow('Citra Asli', citra)
cv2.imshow('Deteksi Tepi', tepi)
cv2.imshow('Tanpa Latar Belakang', citra_tanpa_latar)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Gambar1.3 Kode Deteksi Citra



Gambar1.4 Citra Asli & Hasil dari Citra Tepi dan Tanpa Latar Belakang

3. Kode & Hasil dari Citra Iput Wajib lemon.jpg

Fotografi > lancar.py

```python
import cv2
import numpy as np

# Membaca citra
nama_file_citra = 'Fotografi\lemon.jpg'  # Ganti dengan nama file citra Anda
citra = cv2.imread(nama_file_citra)

# Deteksi tepi menggunakan metode Canny
tepi = cv2.Canny(citra, 50, 250)

# Mengonversi citra ke dalam format HSV
citra_hsv = cv2.cvtColor(citra, cv2.COLOR_BGR2HSV)

# Menentukan rentang warna latar belakang yang ingin dihapus
lower_latar_belakang = np.array([30, 0, 0])
upper_latar_belakang = np.array([550, 350, 450])

# Membuat mask untuk warna latar belakang
mask_latar_belakang = cv2.inRange(citra_hsv, lower_latar_belakang, upper_latar_belakang)

# Mengganti piksel latar belakang dengan warna hitam
citra_tanpa_latar = cv2.bitwise_and(citra, citra, mask=~mask_latar_belakang)

# Menampilkan citra asli, citra deteksi tepi, dan citra tanpa latar belakang
cv2.imshow('Citra Asli', citra)
cv2.imshow('Deteksi Tepi', tepi)
cv2.imshow('Tanpa Latar Belakang', citra_tanpa_latar)
cv2.waitKey(0)
cv2.destroyAllWindows()

```

Gambar1.5 Kode Deteksi Citra

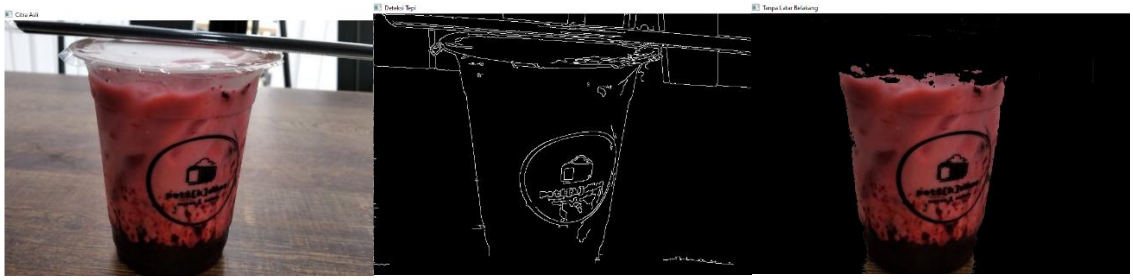
Gambar1.6 Citra Asli & Hasil dari Citra Tepi dan Tanpa Latar Belakang

4. Kode & Hasil Citra dari Input Tambahan es.jpg

Fotografi > lancar.py

```python
import cv2
import numpy as np

# Membaca citra
nama_file_citra = 'Fotografi\esss.jpg'  # Ganti dengan nama file citra Anda
citra = cv2.imread(nama_file_citra)

# Deteksi tepi menggunakan metode Canny
tepi = cv2.Canny(citra, 50, 250)

# Mengonversi citra ke dalam format HSV
citra_hsv = cv2.cvtColor(citra, cv2.COLOR_BGR2HSV)

# Menentukan rentang warna latar belakang yang ingin dihapus
lower_latar_belakang = np.array([0, 0, 0])
upper_latar_belakang = np.array([300, 100, 500])

# Membuat mask untuk warna latar belakang
mask_latar_belakang = cv2.inRange(citra_hsv, lower_latar_belakang, upper_latar_belakang)

# Mengganti piksel latar belakang dengan warna hitam
citra_tanpa_latar = cv2.bitwise_and(citra, citra, mask=~mask_latar_belakang)

# Menampilkan citra asli, citra deteksi tepi, dan citra tanpa latar belakang
cv2.imshow('Citra Asli', citra)
cv2.imshow('Deteksi Tepi', tepi)
cv2.imshow('Tanpa Latar Belakang', citra_tanpa_latar)
cv2.waitKey(0)
cv2.destroyAllWindows()

```

Gambar1.7 Kode Deteksi Citra



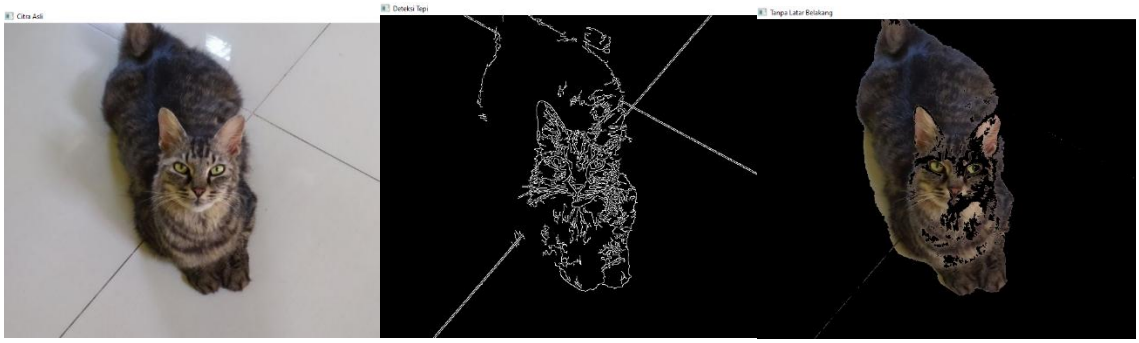
Gambar1.8 Citra Asli & Hasil dari Citra Tepi dan Tanpa Latar Belakang

5. Kode & Hasil dari Citra Input Tambahan kucing.jpg

Fotografi > lancar.py

```python
import cv2
import numpy as np

# Membaca citra
nama_file_citra = 'Fotografi\cat3.jpg'  # Ganti dengan nama file citra Anda
citra = cv2.imread(nama_file_citra)

# Deteksi tepi menggunakan metode Canny
tepi = cv2.Canny(citra, 50, 200)

# Mengonversi citra ke dalam format HSV
citra_hsv = cv2.cvtColor(citra, cv2.COLOR_BGR2HSV)

# Menentukan rentang warna latar belakang yang ingin dihapus
lower_latar_belakang = np.array([0, 0, 100])
upper_latar_belakang = np.array([300, 60, 300])

# Membuat mask untuk warna latar belakang
mask_latar_belakang = cv2.inRange(citra_hsv, lower_latar_belakang, upper_latar_belakang)

# Mengganti piksel latar belakang dengan warna hitam
citra_tanpa_latar = cv2.bitwise_and(citra, citra, mask=~mask_latar_belakang)

# Menampilkan citra asli, citra deteksi tepi, dan citra tanpa latar belakang
cv2.imshow('Citra Asli', citra)
cv2.imshow('Deteksi Tepi', tepi)
cv2.imshow('Tanpa Latar Belakang', citra_tanpa_latar)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Gambar1.9 Kode Deteksi Citra



Gambar1.10 Citra Asli & Hasil dari Citra Tepi dan Tanpa Latar Belakang

Kesimpulan

Setelah melakukan deteksi citra dengan Python menggunakan OpenCV dan NumPy, deteksi tepi menggunakan Canny, serta mengonversi citra ke dalam format HSV, beberapa kesimpulan dapat diambil:

A. Deteksi Citra dengan OpenCV:

- OpenCV memberikan alat yang kuat untuk membaca, memanipulasi, dan menganalisis citra.
- Fungsi cv2.imread digunakan untuk membaca citra dari file.

B. Menggunakan NumPy untuk Manipulasi Data:

- NumPy mempermudah manipulasi array dan operasi numerik pada citra.
- import numpy as np memberikan akses ke berbagai fungsi dan alat NumPy.

C. Deteksi Tepi menggunakan Canny:

- Metode Canny efektif dalam mendeteksi tepi pada citra.
- Diperlukan penyesuaian nilai ambang untuk mengoptimalkan deteksi tepi.

D. Konversi ke Format HSV:

- cv2.cvtColor memungkinkan konversi mudah antara format warna, seperti dari BGR ke HSV.
- Format HSV memungkinkan representasi warna yang lebih intuitif dengan komponen Hue, Saturation, dan Value.

E. Tahap Pengolahan Citra Berurutan:

- Langkah-langkah deteksi citra, deteksi tepi, dan konversi format dapat diintegrasikan secara berurutan untuk mendapatkan hasil yang diinginkan.
- Kesesuaian parameter dalam setiap langkah sangat penting.

F. Eksperimen dan Penyesuaian:

- Deteksi citra seringkali melibatkan eksperimen dengan nilai-nilai ambang dan parameter.
- Diperlukan pemahaman mendalam terhadap karakteristik citra dan tujuan deteksi.

Dengan menggabungkan OpenCV dan NumPy, Anda dapat mengembangkan alur kerja pengolahan citra yang kuat dan fleksibel. Eksperimen dan penyesuaian parameter menjadi kunci untuk mencapai hasil yang optimal dalam konteks deteksi tepi dan konversi format citra.