UJJIAN AKHIR SEMESTER
MATA KULIAH
PRAK. PENGOLAHAN GAMBAR & FOTOGRAFI

Topik
SEGMENTASI CITRA

PENYUSUN LAPORAN

Nama Mahasiswa	NIM	Kelas
Ritantri Eka Luna	062340833238	1 MIO

DOSEN PENGAMPUH

Sulistiyanto, MTI

PROGRAM STUDI MANAJEMEN INFORMATIKA
JURUSAN MANAJEMEN INFORMATIKA
POLITEKNIK NEGERI SRIWIJAYA
2023

DAFTAR ISI

BAB I

PENDAHULUAN

A. Latar Belakang

Deteksi tepi adalah cara-cara matematis untuk mengenali titik-titik dalam citra digital yang kecerahannya berubah drastis atau, secara formal, memiliki diskontinuitas. Titik-titik yang mengalami perubahan kecerahan secara drastis biasanya berada dalam suatu garis atau kurva yang disebut sebagai *tepi*. Deteksi tepi adalah alat dasar dalam pengolahan citra digital, penglihatan mesin, dan penglihatan komputer, khususnya dalam bidang deteksi fitur dan ekstraksi fitur.

Dalam pengelolahan citra digital, **segmentasi citra** adalah proses pembagian citra digital ke dalam beberapa bagian (objek). Segmentasi citra bertujuan untuk menyederhanakan penggambaran citra ke dalam bentuk yang lebih bermakna dan lebih mudah dianalisis. Segmentasi citra biasa dipakai untuk menentukan letak objek dan batasannya (garis, kurva, dan lain-lain) dalam citra.

Hasil segmentasi citra adalah (1) himpunan segmen yang secara kolektif menutupi seluruh citra atau (2) himpunan garis kontur yang dihasilkan dari citra (lihat pendeteksian tepi). Piksel-piksel dalam sebuah wilayah mirip sifatnya, seperti warna, intensitas, atau teksturnya

B. Rumusan Masalah

1. Bagaimana cara deteksi tepi pada sebuah objek di aplikasi python
2. Bagaimana cara segmentasi citra pada sebuah objek di aplikasi python
3. Bagaimana hasil output dari deteksi tepi dan segmentasi citra
4. Bagaimana cara kerja fungsi program/algoritma dan hasil dari deteksi tepi
5. Bagaimana cara kerja fungsi program/algoritma dan hasil dari segmemntasi citra

C. Tujuan

1. Mengetahui cara deteksi tepi pada sebuah objek di aplikasi python
2. Mengetahui cara segmentasi citra pada sebuah objek di aplikasi python
3. Mengetahui Bagaimana hasil output dari deteksi tepi dan segmentasi citra
4. Mengetahui cara kerja fungsi program/algoritma dan hasil dari deteksi tepi
5. Mengetahui cara kerja fungsi program/algoritma dan hasil dari segmemntasi citra

BAB II

PEMBAHASAN

A. Deteksi Tepi Pada Sebuah Objek Di Aplikasi Python

Untuk melakukan deteksi tepi pada objek dengan latar belakang hitam dan menghasilkan garis kontur putih yang merepresentasikan tepi dari objek, Kita dapat menggunakan deteksi tepi (seperti Canny) untuk mendapatkan kontur objek, dan kemudian menggambar kontur tersebut sebagai garis putih pada latar belakang hitam.

Berikut adalah contoh penggunaan deteksi tepi dan penggambaran kontur sebagai garis putih pada latar belakang hitam menggunakan OpenCV di Python:

```
as > uas deteksi tepi.py > ...
1  import cv2
2  import numpy as np
3  from matplotlib import pyplot as plt
4
5  # Baca gambar
6  img = cv2.imread("contoh_gambar.jpeg")# Ganti 'contoh_gambar.jpg' dengan nama file gambar Anda
7  img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
8  canny = cv2.Canny(img, 100, 200)
9
10 #pemberian judul gambar yang ditampilkan
11 titles = ['image', 'canny']
12 images = [img, canny]
13
14 for i in range(2):
15     #Plot hasil deteksi tepi
16     plt.subplot(1, 2, i+1), plt.imshow(images[i], 'gray')
17     plt.title(titles[i])
18     plt.xticks([]),plt.yticks([])
19
20 #menampilkan hasil
21 plt.show()
22
23
```

Pastikan untuk mengganti 'contoh_gambar.jpeg' dengan path menuju gambar yang ingin Anda proses. Anda juga dapat menyesuaikan nilai ambang rendah (low_threshold) dan tinggi (high_threshold) sesuai dengan kebutuhan Anda.

Setelah menjalankan skrip ini, dua jendela akan muncul: satu menampilkan gambar asli dan yang lainnya menampilkan hasil deteksi tepi Canny. Jendela akan tetap terbuka sampai Anda menekan tombol apa pun di keyboard. Jika Anda ingin menyimpan hasil deteksi tepi ke file, Anda dapat menambahkan kode tambahan menggunakan cv2.imwrite().

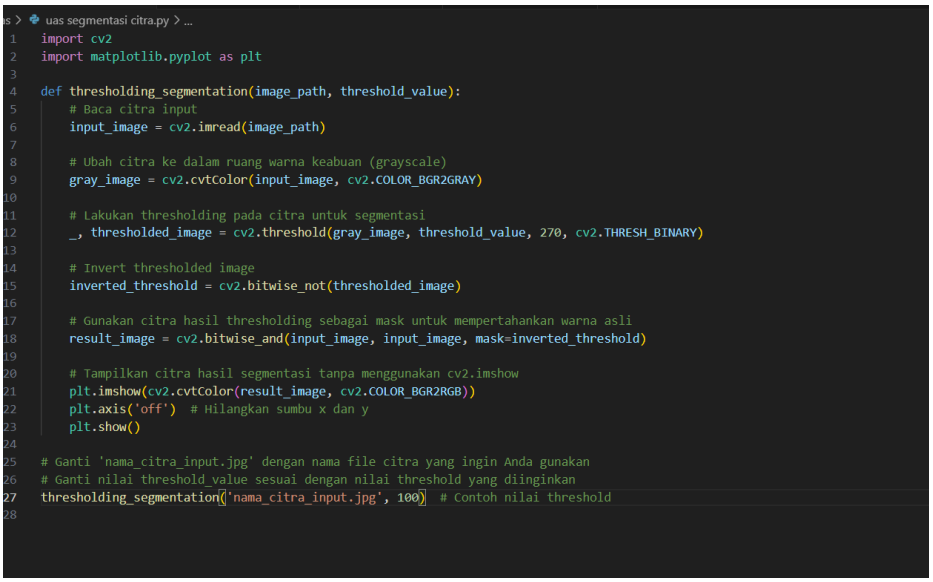
B. Segmentasi Citra Pada Sebuah Objek Di Aplikasi Python

Segmentasi citra adalah proses pemisahan atau pengelompokan piksel dalam citra ke dalam kelompok atau segmen yang memiliki karakteristik serupa. Tujuan dari

segmentasi citra adalah untuk mengidentifikasi dan memisahkan objek atau area yang menarik dalam sebuah gambar. Dalam konteks deteksi tepi atau pemisahan objek dari latar belakang, segmentasi membantu memisahkan area objek dari area latar belakang.

Proses segmentasi dapat dilakukan dengan berbagai metode, tergantung pada sifat dan karakteristik citra serta objek yang ingin diidentifikasi. Segmentasi citra thresholding adalah salah satu metode segmentasi yang paling sederhana dan umum digunakan. Pada dasarnya, thresholding melibatkan pemilihan nilai ambang (threshold) untuk memisahkan piksel menjadi dua kelas berdasarkan nilai intensitasnya. Piksel yang memiliki intensitas di atas ambang tersebut dianggap sebagai bagian objek, sedangkan piksel dengan intensitas di bawah ambang dianggap sebagai bagian latar belakang.

Berikut adalah contoh penggunaan segmentasi citra thresholding dalam Python menggunakan OpenCV :



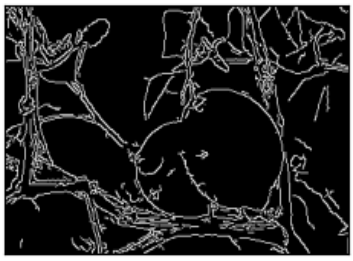
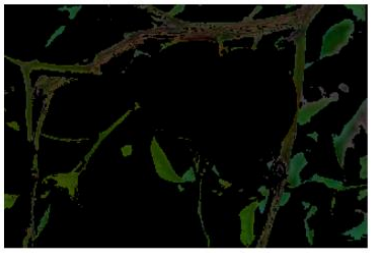
Pastikan untuk mengganti 'nama_citra_input.jpg' dengan nama file citra yang ingin Anda gunakan. Selain itu, sesuaikan nilai threshold_value sesuai dengan nilai threshold yang Anda inginkan untuk melakukan segmentasi citra dengan teknik thresholding.

Dengan cara ini, citra hasil segmentasi akan ditampilkan secara langsung menggunakan Matplotlib tanpa harus menggunakan cv2.imshow. Seluruh tampilan GUI akan dihindari, dan citra hasil segmentasi akan ditampilkan dalam jendela pop-up yang dihasilkan oleh Matplotlib.

C. Hasil Output

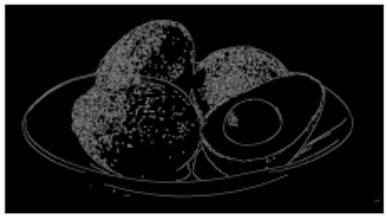
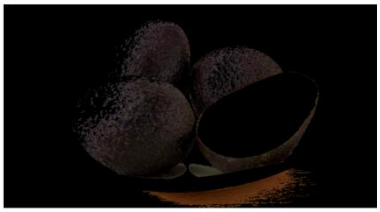
1. CITRA WAJIB

- Lemon

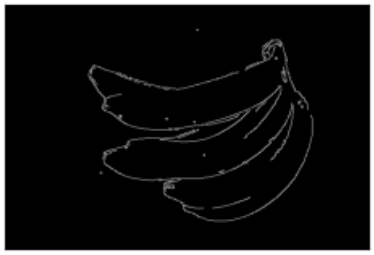
Gambar asli citra deteksi tepi citra hasil segmentasi

- Alpukat

Gambar asli citra deteksi tepi citra hasil segmentasi

- Pisang


Gambar asli citra deteksi tepi citra hasil segmentasi

2. CITRA BEBAS

- Motor



Gambar asli citra deteksi tepi citra hasil segmentasi

- Kucing



Gambar asli citra deteksi tepi citra hasil segmentasi

D. Cara Kerja Fungsi Program/Algoritma dan Hasil Dari Deteksi Tepi

1. **Cara Kerja Program/Algoritma Deteksi Tepi:**
   - Input Citra: Program dimulai dengan membaca citra yang akan diolah. Citra ini dapat berupa gambar dalam format yang didukung oleh OpenCV.
   - Konversi ke Keabuan (Grayscale): Agar lebih efektif, citra mungkin dikonversi ke citra keabuan (grayscale), karena deteksi tepi seringkali lebih baik dilakukan pada citra keabuan.

- Penerapan Algoritma Deteksi Tepi: Salah satu algoritma deteksi tepi yang umum digunakan adalah operator Sobel atau operator Canny. Operator Sobel, misalnya, digunakan untuk menghitung gradien citra yang kemudian dapat dijadikan acuan untuk menemukan tepi.
- Thresholding (Opsional): Hasil dari operator deteksi tepi mungkin diberlakukan thresholding untuk meningkatkan kontras dan membuat tepi lebih jelas.
- Tampilan Hasil: Citra hasil deteksi tepi ditampilkan menggunakan OpenCV atau alat visualisasi lainnya.

2. Hasil Deteksi Tepi:

- Visualisasi Tepi: Citra hasil deteksi tepi menyoroti lokasi tepi dalam citra asli. Nilai piksel tinggi menunjukkan keberadaan tepi yang signifikan.
- Penggunaan dalam Aplikasi Lain: Hasil deteksi tepi umumnya digunakan dalam berbagai aplikasi, seperti pengenalan objek, analisis citra medis, dan segmentasi objek.
- Penyederhanaan Struktur Citra: Deteksi tepi membantu menyederhanakan struktur citra dengan menyoroti perubahan intensitas yang signifikan, seperti tepi antara objek dan latar belakang.
- Berdasarkan Derivatif Intensitas: Deteksi tepi beroperasi dengan melibatkan perhitungan derivatif intensitas citra, yang menyoroti tempat-tempat di mana intensitas berubah tajam.
- Optimasi dan Peningkatan: Implementasi deteksi tepi dapat diopitimalkan dengan menyesuaikan parameter dan menggunakan metode-metode lanjutan, seperti algoritma Canny yang mengintegrasikan beberapa tahapan untuk hasil yang lebih baik.

E. Cara Kerja Fungsi Program/Algoritma dan Hasil Dari Segmentasi Citra

1. Cara Kerja Program/Algoritma:

- Input Citra: Program memulai dengan membaca citra yang diinginkan, yang dapat berupa berkas gambar dalam format yang didukung oleh OpenCV (seperti JPEG, PNG, dll.).
- Konversi Warna (Opsional): Beberapa algoritma segmentasi mungkin memerlukan citra dalam format warna tertentu. Sebagai contoh, HSV (Hue,

Saturation, Value) sering digunakan untuk segmentasi berdasarkan warna. Jika diperlukan, citra dapat dikonversi ke format warna yang sesuai.

- Segmentasi: Program kemudian menerapkan teknik segmentasi tertentu. Dalam contoh di atas, kita menggunakan Segmentasi Berwarna dengan mengidentifikasi objek berdasarkan rentang warna tertentu.
- Pemrosesan Mask: Hasil segmentasi disajikan dalam bentuk "mask", yang adalah citra biner dengan nilai piksel 1 untuk bagian yang di-segmentasi dan 0 untuk bagian lainnya.
- Penerapan Mask ke Citra Asli: Mask tersebut diterapkan ke citra asli menggunakan operasi bitwise, sehingga hanya bagian yang di-segmentasi yang tetap terlihat.
- Tampilan Hasil: Citra asli dan citra hasil segmentasi ditampilkan menggunakan OpenCV.

2. Hasil Segmentasi Citra:

- Visualisasi Segmentasi: Citra hasil segmentasi menyoroti bagian-bagian tertentu dalam citra asli yang sesuai dengan kriteria segmentasi yang telah ditentukan.
- Penggunaan dalam Aplikasi Lain: Hasil segmentasi dapat digunakan dalam berbagai aplikasi, seperti pengenalan objek, ekstraksi fitur, analisis pola, atau aplikasi lainnya yang memerlukan pemisahan atau pengenalan bagian-bagian tertentu dalam citra.
- Berdasarkan Kriteria Tertentu: Hasil segmentasi mencerminkan bagaimana citra tersebut dipecah atau diidentifikasi berdasarkan kriteria tertentu, seperti warna, tekstur, atau karakteristik lainnya.
- Optimasi dan Peningkatan: Pada implementasi praktis, Anda mungkin perlu mengoptimalkan dan menyesuaikan parameter atau teknik segmentasi sesuai dengan jenis citra dan tujuan aplikasi Anda.

BAB III

PENUTUP

Kesimpulan

Dilihat dari Relevansinya Deteksi tepi dan segmentasi citra memiliki peran krusial dalam berbagai aplikasi, termasuk visi komputer, pengolahan citra medis, dan analisis gambar. Informasi yang diperoleh dari deteksi tepi dan segmentasi dapat digunakan untuk ekstraksi fitur, pengenalan objek, serta meningkatkan akurasi algoritma lainnya, Pilihan algoritma untuk deteksi tepi dan segmentasi dapat disesuaikan dengan karakteristik citra dan kebutuhan analisis yang spesifik.Tantangan utama mencakup penyesuaian parameter agar sesuai dengan jenis citra yang berbeda dan memastikan hasil yang akurat.Pengembangan teknik dan algoritma yang lebih canggih dapat meningkatkan performa deteksi tepi dan segmentasi citra.

Deteksi Tepi:

TABEL PERBEDAAN DETEKSI TEPI DAN SEGMENTASI CITRA

	Tujuan	Algoritma	Implementasi di Python	Hasil
Deteksi Tepi	Deteksi tepi merupakan langkah kritis dalam pengolahan citra untuk mengidentifikasi perubahan signifikan dalam intensitas piksel, yang dapat mencerminkan batas antara objek dan latar belakang.	Ada berbagai algoritma deteksi tepi, salah satunya adalah menggunakan operator Sobel. Algoritma ini menghitung gradien citra untuk menyoroti tepi.	Implementasi deteksi tepi di Python dapat dilakukan menggunakan library OpenCV. Operator Sobel digunakan untuk menghitung gradien, dan hasilnya dapat diterapkan dengan menggunakan metode thresholding untuk menyoroti tepi yang signifikan.	Hasil deteksi tepi membantu menyederhanakan struktur citra dan memberikan informasi kritis tentang batas objek.

	Tujuan	Algoritma	Implementasi di Python	Hasil
Segmentasi Citra	Segmentasi citra bertujuan untuk mempartisi citra ke dalam bagian-bagian yang homogen berdasarkan ciri-ciri tertentu seperti warna, tekstur, atau intensitas.	Ada banyak metode segmentasi, salah satunya adalah thresholding. Dalam contoh ini, metode thresholding digunakan untuk memisahkan bagian objek dari latar belakang berdasarkan nilai intensitas piksel.	Implementasi segmentasi citra di Python dapat dilakukan menggunakan OpenCV. Metode thresholding sederhana digunakan untuk membuat citra biner yang menyoroti bagian objek.	Hasil segmentasi citra memberikan gambaran yang jelas tentang bagian-bagian objek yang diinginkan dalam citra.