PENJELASAN UTS PRAKTIKUM

1. `import cv2`: Mengimpor modul cv2 dari OpenCV, yang digunakan untuk pemrosesan gambar.

2. `import numpy as np`: Mengimpor modul numpy dengan alias np, yang sering digunakan untuk manipulasi array dan matriks, sangat berguna dalam pengolahan gambar.

3. `import matplotlib.pyplot as plt`: Mengimpor modul pyplot dari Matplotlib dengan alias plt, yang digunakan untuk menampilkan gambar.

4. `import matplotlib.image as img`: Mengimpor modul image dari Matplotlib dengan alias img, digunakan untuk membaca dan menulis gambar.

5. `%matplotlib inline`: Perintah khusus untuk Jupyter Notebook atau JupyterLab, yang memungkinkan tampilan gambar yang dihasilkan langsung di dalam notebook.

`color\_picture = img.imread('sarifa.pcd.jpg')`: Menggunakan fungsi `imread` dari modul `matplotlib.image` untuk membaca gambar dengan nama file "myname.jpg" dan menyimpannya dalam variabel `color\_picture`. Variabel ini akan berisi representasi array dari gambar tersebut.

`plt.imshow(color\_picture)`: Menggunakan fungsi `imshow` dari modul `matplotlib.pyplot` untuk menampilkan gambar yang telah dibaca. Fungsi `imshow` digunakan untuk menampilkan gambar dalam bentuk matriks atau array. Kita memberikan argumen `color\_picture` yang berisi array gambar yang telah dibaca sebelumnya.

Dengan melakukan dua langkah di atas, kita berhasil mengimpor dan menampilkan gambar "myname.jpg" menggunakan Matplotlib. Gambar tersebut akan ditampilkan di dalam notebook, karena saya telah menggunakan `%matplotlib inline` sebelumnya.

1. `red = color\_picture[:, :, 0]`, `green = color\_picture[:, :, 1]`, `blue = color\_picture[:, :, 2]`: Menggunakan slicing array untuk memisahkan saluran warna merah, hijau, dan biru dari gambar. Variabel `color\_picture` berisi array gambar yang telah dibaca sebelumnya, dan slicing dilakukan untuk memilih saluran warna masing-masing.

2. `f, (c1, c2, c3, c4) = plt.subplots(1, 4, figsize=(20, 10))`: Membuat subplot dengan ukuran 1 baris dan 4 kolom menggunakan `plt.subplots()`, dengan ukuran figur 20x10. Subplot akan digunakan untuk menampilkan gambar asli dan masing-masing saluran warna (merah, hijau, biru).

3. `c1.set\_title('Original Picture')`, `c2.set\_title('Red Detection')`, `c3.set\_title('Green Detection')`, `c4.set\_title('Blue Detection')`: Memberikan judul untuk masing-masing subplot.

4. `c1.imshow(color\_picture)`, `c2.imshow(red, cmap="gray")`, `c3.imshow(green, cmap="gray")`, `c4.imshow(blue, cmap="gray")`: Menampilkan gambar asli dan saluran warna merah, hijau, dan biru di masing-masing subplot. Pengaturan `cmap="gray"` digunakan untuk menampilkan gambar dalam skala abu-abu.

5. `c1.axis('off')`, `c2.axis('off')`, `c3.axis('off')`, `c4.axis('off')`: Mematikan sumbu (axis) pada masing-masing subplot agar tidak ditampilkan.

Dengan melakukan langkah-langkah tersebut, program berhasil memisahkan saluran warna (merah, hijau, biru) dari gambar asli dan menampilkannya secara terpisah dalam subplot. Subplot pertama menampilkan gambar asli, sedangkan subplot lainnya menampilkan deteksi warna merah, hijau, dan biru secara terpisah.

melakukan konversi format warna dari BGR (Blue-Green-Red) ke RGB (Red-Green-Blue), sehingga gambar dapat ditampilkan dengan benar menggunakan Matplotlib.

Berikut adalah penjelasan dari setiap baris kode:

1. `color\_picture = cv2.imread('myname.jpg')`: Menggunakan fungsi `imread` dari modul `cv2` untuk membaca gambar dengan nama file "myname.jpg" dan menyimpannya dalam variabel `color\_picture`. Fungsi `imread` membaca gambar dalam format BGR (Blue-Green-Red), yang merupakan format default untuk OpenCV.

2. `color\_picture = cv2.cvtColor(color\_picture, cv2.COLOR\_BGR2RGB)`: Menggunakan fungsi `cvtColor` dari modul `cv2` untuk mengonversi format warna gambar dari BGR ke RGB. Argumen `cv2.COLOR\_BGR2RGB` menentukan jenis konversi yang dilakukan. Setelah konversi, gambar akan disimpan kembali dalam variabel `color\_picture`.

Dengan langkah-langkah tersebut, gambar "myname.jpg" berhasil dibaca menggunakan OpenCV dan dikonversi ke format RGB agar dapat ditampilkan dengan benar menggunakan Matplotlib.

Program menggunakan fungsi `cv2.calcHist()` dari OpenCV untuk menghitung histogram dari gambar. Histogram adalah representasi distribusi intensitas piksel dalam gambar.

Berikut adalah penjelasan dari setiap baris kode:

1. `histogram\_blue = cv2.calcHist([b], [0], None, [256], [0, 256])`: Menghitung histogram untuk saluran warna biru (b). Fungsi `calcHist()` menerima beberapa argumen:

- `images`: Daftar gambar input, dalam hal ini hanya gambar saluran warna biru (b).

- `channels`: Indeks dari saluran yang digunakan untuk perhitungan histogram, dalam hal ini, saluran warna biru (0).

- `mask`: Mask untuk membatasi area yang dihitung, dalam hal ini, tidak ada maska yang digunakan (None).

- `histSize`: Jumlah bin dalam histogram, dalam hal ini, 256 bin.

- `ranges`: Rentang intensitas piksel yang digunakan, dalam hal ini, rentang dari 0 hingga 255.

2. `histogram\_green = cv2.calcHist([g], [0], None, [256], [0, 256])`: Menghitung histogram untuk saluran warna hijau (g) dengan parameter yang serupa dengan perhitungan histogram untuk saluran warna biru.

3. `histogram\_red = cv2.calcHist([r], [0], None, [256], [0, 256])`: Menghitung histogram untuk saluran warna merah (r) dengan parameter yang serupa dengan perhitungan histogram untuk saluran warna biru.

4. `histogram\_colors = cv2.calcHist([color\_picture], [0, 1, 2], None, [256, 256, 256], [0, 256, 0, 256, 0, 256])`: Menghitung histogram untuk keseluruhan warna dalam gambar dengan menggunakan semua saluran warna (RGB). Histogram ini memiliki tiga dimensi untuk masing-masing saluran warna, dan setiap dimensi memiliki 256 bin.

Dengan langkah-langkah tersebut, program berhasil menghitung histogram untuk saluran warna biru, hijau, merah, serta untuk keseluruhan warna dalam gambar menggunakan fungsi `cv2.calcHist()`. Histogram ini dapat digunakan untuk menganalisis distribusi intensitas warna dalam gambar.

Program menampilkan analisi histogram menggunakan Matplotlib untuk menampilkan histogram dari saluran warna biru, hijau, merah, serta histogram untuk keseluruhan warna dalam gambar. Histogram digambarkan sebagai grafik yang menunjukkan distribusi intensitas piksel dalam rentang nilai tertentu.

Berikut adalah penjelasan dari setiap baris kode:

1. `plt.figure(figsize=(12, 8))`: Membuat figur baru dengan ukuran 12x8 inci untuk menampung subplot-subplot yang akan ditampilkan.

2. Subplot 1 (baris 1, kolom 1):

- `plt.subplot(2, 2, 1)`: Membuat subplot dengan ukuran 2 baris dan 2 kolom, dan menentukan bahwa subplot yang sedang dikerjakan adalah yang pertama.

- `plt.plot(histogram\_blue, color='b')`: Membuat plot dari histogram saluran warna biru dengan warna biru (blue).

- `plt.title('Blue Histogram')`: Memberikan judul "Blue Histogram" pada subplot ini.

- `plt.xlim([0, 256])`: Mengatur batas sumbu x (intensitas piksel) dari 0 hingga 255.

3. Subplot 2 (baris 1, kolom 2):

- `plt.subplot(2, 2, 2)`: Membuat subplot baru.

- `plt.plot(histogram\_green, color='g')`: Membuat plot dari histogram saluran warna hijau dengan warna hijau (green).

- `plt.title('Green Histogram')`: Memberikan judul "Green Histogram" pada subplot ini.

- `plt.xlim([0, 256])`: Mengatur batas sumbu x.

4. Subplot 3 (baris 2, kolom 1):

- `plt.subplot(2, 2, 3)`: Membuat subplot baru.

- `plt.plot(histogram\_red, color='r')`: Membuat plot dari histogram saluran warna merah dengan warna merah (red).

- `plt.title('Red Histogram')`: Memberikan judul "Red Histogram" pada subplot ini.

- `plt.xlim([0, 256])`: Mengatur batas sumbu x.

5. Subplot 4 (baris 2, kolom 2):

- `plt.subplot(2, 2, 4)`: Membuat subplot baru.

- `plt.plot(histogram\_colors.flatten(), color='gray')`: Membuat plot dari histogram untuk seluruh warna (dengan histogram\_colors yang telah di-flatten) dengan warna abu-abu (gray).

- `plt.title('Gray Histogram')`: Memberikan judul "Gray Histogram" pada subplot ini.

- `plt.xlim([0, 256])`: Mengatur batas sumbu x.

Dengan langkah-langkah tersebut, program berhasil menampilkan keempat histogram dalam satu figur menggunakan Matplotlib. Setiap subplot menampilkan histogram untuk saluran warna biru, hijau, merah, dan histogram keseluruhan warna dalam gambar.

Program ini menunjukkan penggunaan beberapa fungsi untuk meningkatkan kecerahan gambar, serta mendeteksi dan menyorot warna-warna tertentu dalam gambar. Berikut penjelasan dari setiap bagian:

Fungsi `increase\_brightness(image, value=30)`

- Fungsi ini digunakan untuk meningkatkan kecerahan gambar.

- Input:

- `image`: Gambar yang akan ditingkatkan kecerahannya.

- `value`: Nilai kecerahan yang akan ditambahkan ke gambar (default adalah 30).

- Proses:

- Gambar diubah ke ruang warna HSV.

- Komponen kecerahan (Value) dari ruang warna HSV ditambah dengan nilai `value`.

- Komponen kecerahan yang telah diubah dipastikan berada dalam rentang 0 hingga 255.

- Gambar dikembalikan ke ruang warna BGR.

Fungsi `detect\_and\_highlight\_blue(image)`

- Fungsi ini digunakan untuk mendeteksi dan menyorot warna biru dalam gambar.

- Input:

- `image`: Gambar yang akan diproses.

- Proses:

- Gambar diubah ke ruang warna HSV.

- Dilakukan segmentasi warna untuk mendapatkan mask biru.

- Menggunakan operasi bitwise AND antara gambar asli dan mask biru untuk menyorot warna biru.

- Gambar hasil diterapkan dengan filter, sehingga piksel yang tidak berwarna biru diubah menjadi putih dan yang lainnya tetap hitam.

Fungsi `detect\_and\_highlight\_red\_blue(image)`

- Fungsi ini digunakan untuk mendeteksi dan menyorot kombinasi warna merah-biru dalam gambar.

- Input:

- `image`: Gambar yang akan diproses.

- Proses:

- Gambar diubah ke ruang warna HSV.

- Dilakukan segmentasi warna untuk mendapatkan mask merah dan biru.

- Menggunakan operasi bitwise OR antara mask merah dan mask biru untuk menyorot warna merah dan biru.

- Gambar hasil diterapkan dengan filter, sehingga piksel yang tidak berwarna merah atau biru diubah menjadi putih dan yang lainnya tetap hitam.

Fungsi `detect\_and\_highlight\_red\_green\_blue(image)`

- Fungsi ini digunakan untuk mendeteksi dan menyorot kombinasi warna merah-hijau-biru dalam gambar.

- Input:

- `image`: Gambar yang akan diproses.

- Proses:

- Gambar diubah ke ruang warna HSV.

- Dilakukan segmentasi warna untuk mendapatkan mask merah, hijau, dan biru.

- Menggunakan operasi bitwise OR antara ketiga mask untuk menyorot warna merah, hijau, dan biru.

- Gambar hasil diterapkan dengan filter, sehingga piksel yang tidak berwarna merah, hijau, atau biru diubah menjadi putih dan yang lainnya tetap hitam.

Main Program

- Gambar dibaca menggunakan OpenCV.

- Kecerahan gambar ditingkatkan menggunakan fungsi `increase\_brightness`.

- Berbagai fungsi deteksi warna dipanggil untuk menyorot warna-warna tertentu dalam gambar.

- Gambar-gambar hasil dianalisis ditampilkan menggunakan Matplotlib.

Dengan langkah-langkah tersebut, program berhasil meningkatkan kecerahan gambar, serta mendeteksi dan menyorot warna-warna tertentu dalam gambar dengan cara yang ditentukan.

Program ini untuk menampilkan gambar-gambar dalam jendela pop-up dan menyimpannya sebagai file gambar. Berikut penjelasan untuk setiap baris kode:

1. `cv2.imshow('Original Picture', color\_picture)`: Menampilkan gambar asli dalam jendela pop-up dengan judul "Original Picture".

2. `cv2.waitKey(0)`: Menunggu sampai tombol apa pun ditekan (0 menandakan waktu tak terbatas) sebelum melanjutkan eksekusi.

3. `cv2.destroyAllWindows()`: Menutup semua jendela yang ditampilkan.

4. `cv2.imwrite('hasil-red-detection.jpg',red)`: Menyimpan gambar hasil deteksi warna merah sebagai file JPEG dengan nama "hasil-red-detection.jpg".

5. `cv2.imwrite('hasil-blue-detection1.jpg',blue)`: Menyimpan gambar hasil deteksi warna biru sebagai file JPEG dengan nama "hasil-blue-detection1.jpg".

6. `cv2.imwrite('hasil-green-detection.jpg',green)`: Menyimpan gambar hasil deteksi warna hijau sebagai file JPEG dengan nama "hasil-green-detection.jpg".

7. `cv2.imshow('hasil-brightened-image(contrastimage))', brightened\_image)`: Menampilkan gambar hasil peningkatan kecerahan dalam jendela pop-up dengan judul "hasil-brightened-image(contrastimage)".

8. `cv2.waitKey(0)`: Menunggu input dari pengguna sebelum melanjutkan.

9. `cv2.destroyAllWindows()`: Menutup semua jendela yang ditampilkan.

10. Langkah-langkah yang sama dilakukan untuk menampilkan dan menyimpan gambar-gambar hasil deteksi warna biru (`blue\_detection`), hasil deteksi kombinasi warna merah-biru (`red\_blue\_detection`), dan hasil deteksi kombinasi warna merah-hijau-biru (`red\_green\_blue\_detection`).

Program tersebut menggunakan OpenCV untuk menampilkan gambar-gambar dalam jendela pop-up dan menyimpannya sebagai file gambar dengan format JPEG. Ini berguna untuk mengeksplorasi dan memeriksa hasil-hasil dari proses-proses yang dilakukan pada gambar.

Konsep Teori

Program-program di atas mengimplementasikan beberapa konsep dasar dalam pemrosesan gambar, antara lain:

1. Pemrosesan Gambar dengan OpenCV: Program menggunakan OpenCV untuk membaca, memanipulasi, dan menganalisis gambar. OpenCV adalah salah satu library yang paling populer untuk pemrosesan gambar dan komputer vision.

2. Manipulasi Ruang Warna: Program menggunakan konversi ruang warna (seperti BGR ke RGB dan HSV) untuk memanipulasi warna dalam gambar. Ini memungkinkan deteksi warna yang lebih tepat dan peningkatan kecerahan gambar dengan lebih mudah.

3. Histogram: Program menghitung dan menampilkan histogram gambar, yang merupakan representasi visual dari distribusi intensitas piksel dalam gambar. Histogram membantu dalam analisis warna dan kontras gambar.

4. Segmentasi Warna: Program menggunakan segmentasi warna untuk mendeteksi dan menyorot piksel-piksel yang memiliki warna tertentu dalam gambar. Segmentasi warna adalah teknik yang umum digunakan dalam deteksi objek berwarna dalam pemrosesan gambar.

5. Visualisasi dengan Matplotlib: Program menggunakan Matplotlib untuk menampilkan gambar dan histogram secara visual. Matplotlib adalah library yang sering digunakan untuk visualisasi data dalam Python.

6. Penyimpanan dan Tampilan Gambar: Program menyimpan gambar-gambar hasil proses dalam format file yang berbeda dan menampilkannya dalam jendela pop-up. Ini memungkinkan pengguna untuk menyimpan dan memeriksa hasil-hasil dari proses pemrosesan gambar.