Nama : Muhammad Husain Kaasyiful Ghitha

NIM : 1103220047

Kelas TK-45-G09

Robotika dan Sistem Cerdas

Analisis Tugas Robotika Week 10

Pada tugas ini, dilaksanakan beberapa penerapan pemrosesan citra (image processing), deteksi fitur (feature detection), dan deskripsi fitur (feature description) menggunakan bahasa Python. Simulasi-simulasi yang diprogram adalah sebagai berikut:

- Membuat dan Mengaplikasikan Filter Moving Average.
- Deteksi Fitur dengan SIFT.
- Representasi Histogram Gambar.
- Gaussian Smoothing.
- Deteksi Tepi dengan Sobel Filter.
- Representasi Fitur dengan Histogram of Oriented Gradients (HOG).

Selain memrogram kode Python untuk simulasi tersebut, terdapat juga simulasi yang berfokus pada implementasinya pada robot menggunakan simulator Webots. Simulasi-simulasi yang dilakukan pada Webots tersebut, di antaranya:

- Visual Tracking dengan OpenCV
- Document Scanner Simulation
- Fruit Detection Robot

Berikut adalah pembahasan mengenai kode Python dan simulasi-simulasi yang dilakukan.

1. Kode Python menggunakan OpenCV untuk simulasi Image Processing, Feature Detection, and Feature Description.

   Pada kode simulasi ini, penjelasan fungsi-fungsi yang dimuat di program adalalah sebagai berikut:

a. Filter Moving Average.

Filter Moving Average adalah teknik pemrosesan citra yang digunakan untuk menghaluskan citra dan mengurangi noise pada citra. Metode ini bekerja dengan menggantikan setiap piksel dalam citra dengan rata-rata dari nilai-nilai piksel yang ada di dalam sebuah kernel berisikan piksel tersebut sebagai pusatnya dan piksel sekelilingnya, dalam area yang ditentukan oleh kernel yang digunakan. Ketika menerapkan filter ini untuk pemrosesan citra, kernel yang digunakan biasanya berbentuk matriks kecil berdimensi ganjil (misalnya, matriks 3x3 atau 5x5) yang menggeser citra dan menghitung rata-rata nilai-nilai piksel di dalam jendela tersebut.

Di dalam program, hal yang pertama dilakukan adalah membaca gambar dengan menggunakan cv2.imread(image_path, 0), yang memuat gambar dari path yang diberikan dan mengonversinya menjadi citra dalam format grayscale (hitam-putih) yang ditandai dengan parameter kedua yang bernilai nol.Selanjutnya, filter moving average dibuat menggunakan NumPy dengan np.ones((5, 5), np.float32) / 25. Di sini, kernel yang digunakan adalah matriks berukuran 5x5 yang berisi nilai 1 di setiap elemennya. Kernel ini kemudian dibagi dengan 25, yang merupakan hasil dari 5x5 (jumlah elemen dalam matriks).

Setelah kernel dibuat, filter tersebut diterapkan pada gambar dengan menggunakan fungsi cv2.filter2D(image, -1, kernel). Fungsi ini melakukan konvolusi antara gambar dan kernel, yang berarti kernel akan digeser di atas citra, dan setiap piksel akan dihitung rata-ratanya berdasarkan nilai-nilai piksel di sekitarnya yang tertutup oleh kernel. Terakhir, gambar asli dan gambar yang telah difilter ditampilkan menggunakan matplotlib dengan ukuran gambar dan label yang disesuaikan.

b. Deteksi Fitur dengan SIFT.

SIFT (Scale-Invariant Feature Transform) adalah algoritma yang digunakan untuk mendeteksi dan menggambarkan fitur unik dalam citra, yang dapat bertahan terhadap perubahan skala, rotasi, dan pencahayaan. Fitur-fitur ini, yang disebut keypoints, merupakan titik-titik penting dalam citra yang memiliki karakteristik unik dan mudah dikenali meskipun citra tersebut mengalami transformasi.

Bagian program untuk deteksi fitur dimulai dengan membaca citra dalam format grayscaleSelanjutnya, objek detektor SIFT dibuat menggunakan cv2.SIFT_create().

Fungsi sift.detectAndCompute(image, None) kemudian digunakan untuk mendeteksi keypoints (titik-titik fitur) dan menghitung deskriptor dari setiap keypoint yang terdeteksi. Deskriptor adalah representasi numerik dari fitur di sekitar suatu keypoint.

Setelah keypoints dan deskriptor diperoleh, fungsi cv2.drawKeypoints() digunakan untuk menggambar keypoints pada citra, dengan pilihan cv2.DRAW_MATCHES_FLAGS_DRAW_RICH_KEYPOINTS yang menampilkan ukuran dan orientasi keypoints. Hasilnya kemudian ditampilkan menggunakan matplotlib dengan ukuran gambar dan pelabelan yang disesuaikan.

c. Representasi Histogram Gambar.

Representasi histogram warna pada gambar adalah sebuah representasi grafis yang menunjukkan distribusi frekuensi nilai pixel untuk masing-masing channel warna dalam gambar. Setiap gambar digital terdiri dari tiga channel warna utama, yaitu merah (Red), hijau (Green), dan biru (Blue) yang dikenal sebagai model warna RGB. Histogram warna menggambarkan seberapa sering nilai pixel tertentu (dari 0 hingga 255) muncul dalam setiap channel warna tersebut. Ini membantu untuk menganalisis kontras, kecerahan, dan distribusi warna dalam gambar.

Program representasi ini bekerja dengan cara membaca sebuah gambar dan menghitung histogram warna untuk masing-masing channel RGB (Biru, Hijau, Merah). Pertama, gambar dibaca menggunakan cv2.imread(). Kemudian, sebuah daftar colors didefinisikan untuk mewakili channel warna biru, hijau, dan merah. Looping digunakan untuk menghitung histogram untuk setiap channel warna dengan fungsi cv2.calcHist(), yang menghitung distribusi nilai pixel dalam rentang 0-255.

Setelah histogram untuk setiap channel dihitung, fungsi plt.plot() digunakan untuk menggambar histogram warna dengan masing-masing warna yang sesuai (biru, hijau, merah). Batas sumbu X diatur untuk menunjukkan rentang nilai pixel, dan grafik diberi label pada sumbu X dan sumbu Y. Judul histogram juga ditambahkan dengan plt.title(). Terakhir, plt.show() digunakan untuk menampilkan grafik histogram pada layar.

d. Gaussian Smoothing.

Gaussian Smoothing dalam pemrosesan citra adalah teknik filter yang digunakan untuk mengurangi noise dan meratakan citra dengan menggunakan fungsi Gaussian. Teknik ini menggunakan kernel berbentuk distribusi Gaussian (yang lebih sering dikenal sebagai distribusi normal) untuk menghitung nilai pixel baru berdasarkan pixel-pixel di sekitarnya. Fungsi Gaussian bergantung pada dua variabel, yaitu nilai simpangan (σ) dan posisi piksel. Semakin besar nilai simpangan, maka efek blur-nya akan semakin besar.

Pertama, citra dibaca dalam mode grayscale (hitam-putih). Selanjutnya, citra yang sudah dibaca diterapkan filter Gaussian menggunakan cv2.GaussianBlur. Parameter (5, 5) menunjukkan ukuran kernel (filter) 5x5, dan 2 adalah nilai standar deviasi (σ) yang mengontrol kekuatan blur. Setelah citra yang di-blur telah didapatkan, maka citra blur tersebut ditampilkan dengan gambar aslinya menggunakan matplotlib, dengan ukuran gambar dan pelabelan yang disesuaikan,

e. Deteksi Tepi dengan Sobel Filter.

Sobel Filter adalah alat yang digunakan dalam pemrosesan citra untuk mendeteksi tepi (edges) dalam sebuah gambar. Teknik ini sering diterapkan dalam berbagai aplikasi, seperti deteksi fitur dan deskripsi fitur. Sobel Filter bekerja sebagai operator diferensiasi, yang digunakan untuk menyoroti perubahan intensitas piksel dalam gambar. Filter ini bertujuan untuk mengidentifikasi fitur tepi objek atau garis dalam citra dengan cara menghitung gradien intensitas citra di setiap titik. Gradien ini menunjukkan arah dan kekuatan perubahan intensitas di sekitar piksel tertentu.

Program Python menerapkan Sobel Filter untuk mendeteksi tepi. Pertama, gambar dibaca dalam mode grayscale. Kemudian, dua Sobel Filter diterapkan: sobel_x mendeteksi tepi horizontal dengan menghitung gradien horizontal (menggunakan parameter (1, 0)), dan sobel_y mendeteksi tepi vertikal dengan menghitung gradien vertikal (menggunakan parameter (0, 1)). Kedua filter ini menggunakan kernel berukuran 3x3 (ksize=3) dan menghasilkan citra dengan tipe data CV_64F untuk mengakomodasi nilai gradien yang lebih besar. Setelah itu, kedua gradien tersebut digabungkan menggunakan fungsi cv2.magnitude(sobel_x, sobel_y) untuk menghasilkan citra sobel_combined, yang menunjukkan magnitude (kekuatan) gradien pada setiap titik, mencakup informasi tepi baik horizontal maupun vertikal. Seluruh citra representasi yang dibentuk dari Sobel Filter

kemudian ditampilkan menggunakan matplotlib dengan penyesuaian ukuran gambar dan pemuatan label-label yang diperlukan.

f. Representasi Fitur dengan Histogram of Oriented Gradients (HOG).

Histogram of Oriented Gradients (HOG) adalah sebuah teknik ekstraksi fitur yang digunakan untuk mendeskripsikan bentuk dan struktur objek dalam gambar, terutama dalam mendeskripsikan dan mengenali fitur dari suatu citra. HOG sangat efektif dalam mendeteksi objek yang memiliki bentuk yang jelas, sehingga HOG banyak digunakan dalam aplikasi pengenalan objek. Pada dasarnya, HOG mengekstraksi informasi tentang gradien arah dalam citra, yang dapat menggambarkan tepi-tepi yang ada dalam gambar, yang sering kali mencerminkan bentuk objek. Setelah menghitung gradien, citra dibagi menjadi blok-blok kecil yang disebut sel. Di setiap sel, arah gradien dihitung dan dihitung distribusi arah gradien dalam histogram. Selanjutnya, beberapa sel dikelompokkan untuk membentuk sebuah blok, yang kemudian dinormalisasikan dan dibentuk menjadi vektor fitur.

Program yang dibuat melakukan ekstraksi fitur HOG dari sebuah gambar dan memvisualisasikan hasilnya. Pertama, gambar dibaca dalam format grayscale. Kemudian, fitur HOG diekstraksi menggunakan fungsi hog() dari library skimage (scikit-image), dengan parameter yang disesuaikan: 9 orientasi gradien, ukuran sel 8x8 piksel, dan blok sel 2x2. Proses ekstraksi ini juga menghasilkan gambar HOG (dalam variabel hog_image) yang memvisualisasikan gradien dalam citra. Selanjutnya, apabila normalisasi tidak dilakukan secara otomatos, maka gambar HOG dinormalisasi secara manual agar rentang nilai pikselnya berada antara 0 dan 255 untuk memudahkan visualisasi. Gambar HOG yang telah dinormalisasi diubah menjadi tipe data uint8 untuk memastikan gambar dapat ditampilkan dengan benar menggunakan matplotlib.

2. Simulasi Webots yang berfokus pada Image Processing, Feature Detection, dan Feature Description menggunakan OpenCV.

a. Visual Tracking dengan OpenCV

Pada simulasi ini, sebuah robot TurtleBot3 diberi tugas untuk mengikuti (tracking) sebuah bola berwarna merah di dalam suatu arena. Dijelaskan bahwa robot

menggunakan library OpenCV untuk menerapkan HSV tresholding sehingga warna merah bola dapat dideteksi dan di-track.

Kontroler robot yang diberikan di repository bertujuan untuk mengarahkan robot agar mengikuti bola yang terdeteksi oleh kamera. Robot menggunakan kamera untuk menangkap gambar, yang kemudian diolah dengan OpenCV untuk mendeteksi bola berdasarkan warna tertentu dalam ruang warna HSV. Setelah gambar diproses, robot mencari kontur terbesar yang diasumsikan sebagai bola, lalu menghitung posisi tengah bola. Selisih antara posisi tengah bola dan tengah gambar (error) digunakan dalam kontroler proporsional (P controller) untuk menggerakkan motor roda kiri dan kanan. Jika bola berada di kiri, robot akan berputar ke kiri, dan sebaliknya jika bola berada di kanan. Nilai konstanta P mengontrol agresivitas gerakan robot, dengan nilai yang lebih tinggi membuat respons lebih cepat dan lebih agresif.

b. Document Scanner Simulation

Pada simulasi ini, sebuah robot pemindai (scanner) ditempatkan di suatu conveyor belt yang bertugas untuk mendeteksi dan memindai dokumen yang tertera di permukaan paket yang melalui conveyor belt tersebut. Robot memindai dokumen tersebut menjadi format lembar memanjang (portrait) yang biasanya merupakan format pencatatan paket (walau robot tidak menjamin bahwa posisi portrait tersebut tepat dan tidak terbalik). World simulasi tersebut menyediakan dua jenis kemasan paket, walau **salah satu dari paket tersebut memiliki tekstur yang usang dan tidak dapat dibaca oleh Webots**. Robot dapat membaca dokumen selama dokumen tersebut tidak tersembunyi untuk dapat dikenali dan dibaca.

Controller yang diberikan di repository bekerja dengan menggunakan kamera untuk mengenali gambar dokumen dan menampilkan dokumen tersebut di layar robot. Proses dimulai dengan inisialisasi perangkat keras, yaitu robot itu sendiri, kamera, dan display. Kamera diaktifkan untuk mengambil gambar pada interval waktu 100. Gambar yang diambil kemudian diubah dari format yang digunakan oleh Webots (buffer citra) ke format yang dapat diproses lebih lanjut oleh OpenCV dan numpy. Gambar ini akan digunakan untuk mendeteksi area dengan warna tertentu melalui pemfilteran HSV, yang didasarkan pada rentang warna tertentu yang telah ditentukan sebelumnya (dalam variabel HSV_LOW_RANGE dan HSV_UP_RANGE).

Setelah gambar diproses untuk segmentasi warna, script mencoba untuk menemukan dan memanipulasi dokumen yang terdeteksi dalam gambar. Fungsi get_warped_document digunakan untuk menemukan dan merubah perspektif dokumen yang terdeteksi agar terlihat lurus dan dapat diproses lebih lanjut. Jika proses ini gagal (misalnya karena tidak ada dokumen yang terdeteksi), maka warna biru kosong ditampilkan di layar robot. Gambar dokumen yang telah diproses kemudian diubah ukurannya agar sesuai dengan ukuran layar robot dan ditampilkan menggunakan fungsi display_numpy_image. Proses ini berjalan dalam loop selama simulasi robot masih aktif. Jika flag SAVE_TO_DISK diaktifkan, gambar yang diproses akan disimpan dalam format JPEG.

c. Fruit Detection Robot

Repositori GitHub yang diberikan berisi simulasi robot pendeteksi buah menggunakan OpenCV untuk mengembangkan sistem robot lengan UR5e yang dapat mendeteksi dan berinteraksi langsung dengan buah-buahan (misalnya apel dan jeruk) di Webots. Simulasi ini membuat sistem yang dapat mendeteksi dan mengenali objek (buah) dalam gambar sensor dan sekaligus melakukan tindakan fisik (memetik dan menjatuhkan buah) berdasarkan pengenalan objek tersebut secara real-time.

Controller robot yang diberikan mengintegrasikan visi komputer menggunakan OpenCV dengan kontrol robot untuk mendeteksi dan memanipulasi buah dalam simulasi. Robot ini dilengkapi dengan berbagai sensor, termasuk kamera untuk pengenalan objek, sensor jarak untuk mendeteksi kedekatan dengan objek, dan sensor posisi untuk melacak gerakan lengan robot. Kamera robot mengambil gambar, yang kemudian diproses menggunakan OpenCV untuk mendeteksi objek, seperti apel dan jeruk, dengan memanfaatkan teknik pemrosesan citra seperti konversi ke grayscale dan deteksi tepi menggunakan algoritma Canny edge detection (cv2.Canny()).

Setelah gambar diproses, sistem melakukan deteksi objek menggunakan camera recognition dari Webots. Kamera mengenali objek di sekitar robot dan mengidentifikasi apakah objek yang terdeteksi adalah apel atau jeruk berdasarkan nama modelnya. Berdasarkan hasil deteksi ini, status robot beralih antara beberapa tahap, mulai dari WAITING (menunggu objek) hingga PICKING (memetik objek), ROTATING (memutar lengan robot), dan akhirnya DROPPING (menjatuhkan

objek). Pengendalian status robot ini dilakukan dengan menggunakan sebuah state machine yang mengatur setiap tahap interaksi robot dengan objek.

Pengendalian lengan robot dilakukan dengan menggunakan motor yang disimulasikan di Webots. Lengan robot terdiri dari beberapa motor yang mengatur gerakan bagian-bagian tubuh robot UR5e, seperti shoulder, elbow, dan wrist. Berdasarkan status yang aktif, motor-motor ini diprogram untuk bergerak ke posisi tertentu, misalnya untuk mengambil buah dengan gripper, memutar lengan untuk menjatuhkan buah, atau kembali ke posisi semula setelah proses selesai. Kecepatan motor juga disesuaikan agar pergerakan robot terasa halus dan akurat. Selain itu, controller ini juga mencatat jumlah apel dan jeruk yang telah berhasil dipetik dan ditampilkan pada layar antarmuka Webots. Informasi ini diperbarui secara real-time, memberikan feedback visual tentang jumlah buah yang berhasil dideteksi dan diproses oleh robot.