Analisis

Program Python

1. **Filter Moving Average**
   Program ini memanfaatkan widget interaktif FileUpload untuk mengunggah gambar, yang kemudian diproses menggunakan filter rata-rata 9x9 dengan OpenCV untuk menghasilkan efek penghalusan. Gambar asli dan hasil filter ditampilkan berdampingan menggunakan Matplotlib. Program ini menggabungkan pengolahan gambar dasar dan antarmuka interaktif untuk pengalaman visual yang intuitif.
2. **SIFT Feature Extraction**
   Program ini menggunakan widget interaktif FileUpload untuk mengunggah gambar dan mendeteksi fitur menggunakan algoritma **SIFT (Scale-Invariant Feature Transform)** dari OpenCV. Berikut analisis singkatnya:

   1. **Pembacaan Gambar:**
      Gambar yang diunggah diambil dari widget, dikonversi menjadi array numerik, lalu dibaca oleh OpenCV.
   2. **Konversi ke Grayscale:**
      Gambar diubah menjadi format grayscale untuk memudahkan deteksi fitur, karena SIFT bekerja pada intensitas piksel.
   3. **Deteksi Fitur dengan SIFT:**
      Algoritma SIFT mendeteksi **keypoints** (titik-titik penting pada gambar) dan menghasilkan deskriptor yang mewakili karakteristik lokal di sekitar tiap keypoint.
   4. **Visualisasi Hasil:**
      Keypoints yang terdeteksi digambarkan pada gambar asli, dan hasilnya ditampilkan menggunakan Matplotlib.

3. **Histogram Representation**
   Program ini memungkinkan pengguna mengunggah gambar, menampilkan gambar dalam format grayscale, dan menghasilkan histogram intensitas piksel. Berikut analisis singkatnya:

   1. **Upload Gambar:**
      Widget FileUpload digunakan untuk mengunggah file gambar, yang kemudian disimpan sementara dengan nama uploaded_image.jpg.
   2. **Baca dan Proses Gambar:**
      File yang diunggah dibaca oleh OpenCV sebagai gambar grayscale (mode 0). Jika gambar berhasil dibaca, program memprosesnya untuk ditampilkan.
   3. **Visualisasi Gambar dan Histogram:**
      - Gambar grayscale ditampilkan menggunakan Matplotlib.
      - Histogram intensitas piksel dibuat dengan menghitung distribusi nilai piksel (0-255). Grafik ini membantu memahami penyebaran intensitas pada gambar.

4. **Gaussian Blurring**

Program ini menggunakan widget interaktif untuk mengunggah gambar dan menerapkan filter Gaussian Blur menggunakan **Pillow (PIL)**. Berikut analisis singkatnya:

1. **Upload Gambar:**
   Widget FileUpload memungkinkan pengguna mengunggah file gambar, yang kemudian disimpan sementara dengan nama uploaded_image.jpg.
2. **Baca dan Proses Gambar:**
   Gambar yang diunggah dibuka dengan **Pillow**. Filter Gaussian Blur diterapkan pada gambar dengan radius blur yang tetap (dalam hal ini, 444).
3. **Visualisasi Gambar:**
   - Gambar asli ditampilkan di subplot pertama.
   - Gambar hasil filter Gaussian Blur ditampilkan di subplot kedua.
     Keduanya ditampilkan berdampingan menggunakan Matplotlib untuk perbandingan visual.
4. **Kesimpulan:**
   Program ini adalah alat sederhana untuk mengunggah gambar, menerapkan efek pengaburan (blur), dan menampilkan hasilnya. Gaussian Blur berguna untuk mengurangi noise atau membuat elemen gambar kurang jelas, sering digunakan dalam praproses citra.

5. Sobel Edge Detection

Program ini memungkinkan pengguna untuk mengunggah gambar dan melakukan deteksi tepi menggunakan metode **Sobel Edge Detection** dari OpenCV. Berikut analisis singkatnya:

1. **Upload dan Pemrosesan Gambar:**
   Gambar diunggah melalui widget FileUpload, disimpan sementara, lalu dibaca dengan OpenCV. Konversi ke format RGB dilakukan untuk memastikan kompatibilitas tampilan dengan Matplotlib.
2. **Praproses Gambar:**
   Sebelum deteksi tepi, gambar diberi filter Gaussian Blur untuk mengurangi noise, sehingga deteksi tepi lebih halus dan akurat.
3. **Deteksi Tepi Sobel:**
   - **Sobel X:** Mendeteksi perubahan intensitas piksel pada arah horizontal.
   - **Sobel Y:** Mendeteksi perubahan intensitas pada arah vertikal.
   - **Sobel XY:** Kombinasi dari deteksi Sobel X dan Y untuk mendeteksi tepi pada semua arah.
     Hasil deteksi dinormalisasi agar nilai piksel berada di rentang 0–2550-2550–255 untuk memudahkan visualisasi.
4. **Visualisasi:**
   Gambar asli, hasil Sobel X, Sobel Y, dan Sobel XY ditampilkan berdampingan dalam sebuah grid menggunakan Matplotlib, sehingga pengguna dapat membandingkan hasilnya dengan mudah.
5. **Kesimpulan:**
   Program ini memberikan alat sederhana untuk mendeteksi dan menganalisis tepi pada gambar menggunakan metode Sobel. Proses ini berguna dalam aplikasi seperti pengenalan objek, segmentasi, dan analisis fitur visual.

6. Histogram of Oriented Gradients

Program ini memungkinkan pengguna untuk mengunggah gambar dan menganalisis fitur **HOG (Histogram of Oriented Gradients)** menggunakan pustaka **scikit-image**. Berikut analisis singkatnya:

1. **Upload Gambar:**
   Widget FileUpload digunakan untuk mengunggah gambar, yang kemudian disimpan sementara dengan nama uploaded_image.jpg.
2. **Hitung Fitur HOG:**
   - Fitur HOG dihitung dari gambar yang diunggah. Parameter utama meliputi:
     - **Orientations:** Jumlah orientasi gradien (8).
     - **Pixels per Cell:** Ukuran sel piksel (16×16).
     - **Cells per Block:** Blok sel (1×1).
   - Opsi visualize=True digunakan untuk menghasilkan gambar representasi HOG.
3. **Normalisasi dan Visualisasi:**
   - Gambar hasil HOG dinormalisasi menggunakan rescale_intensity agar lebih mudah dilihat.
   - Gambar asli dan gambar HOG ditampilkan berdampingan menggunakan Matplotlib, memungkinkan pengguna untuk membandingkan hasil analisis fitur dengan input gambar.
4. **Kesimpulan:**
   Program ini berguna untuk mendeteksi pola dan fitur penting dalam gambar menggunakan teknik HOG, yang sering digunakan dalam pengenalan pola dan analisis objek, seperti deteksi wajah atau kendaraan. Antarmuka interaktif mempermudah pengguna dalam eksplorasi fitur gambar secara langsung.

Project Webots

1. Visual Tracking

Simulasi ini dirancang untuk menguji kemampuan robot dalam mendeteksi dan melacak objek bergerak, yaitu bola merah yang mengorbit di sekitarnya. Berikut analisis umumnya:

1. **Tujuan Utama:**
   Robot dilatih untuk selalu mengarahkan pandangannya ke bola yang bergerak dalam lintasan melingkar, menciptakan skenario interaktif antara robot dan target dinamis.
2. **Dua Komponen Utama:**
   - **Penggerak Bola:**
     Bola merah bergerak dengan lintasan melingkar tetap, menciptakan target bergerak yang dapat dideteksi dan diikuti oleh robot.
   - **Pengontrol Robot:**
     Robot memanfaatkan kamera untuk mendeteksi posisi bola berdasarkan warna, kemudian menggunakan pengendali proporsional sederhana untuk memutar tubuhnya agar selalu menghadap bola.

3. **Simulasi Dinamis:**
   Kombinasi bola bergerak dan kontrol robot menciptakan skenario dinamis yang menguji kemampuan robot untuk:
   - Mendeteksi target dengan akurat.
   - Merespons pergerakan target secara real-time.
   - Menjaga stabilitas dan akurasi rotasi.
4. **Manfaat Simulasi:**
   - Melatih algoritma penglihatan komputer untuk deteksi objek.
   - Menguji kinerja pengendalian robot terhadap target bergerak.
   - Menciptakan dasar untuk pengembangan robot otonom, seperti sistem pelacakan atau pengawasan.

2. Document Scanner

Dalam simulasi **document scanner**, di mana kamera terpasang di atas kardus yang bergerak dengan stiker di atasnya, terdapat beberapa masalah yang menghambat keberhasilan deteksi gambar dokumen, terutama pada bagian **tekstur kardus**.

1. **Kardus 1 Tidak Tampil dengan Benar:**
   - Masalah utama yang terjadi pada kardus 1 adalah **tekstur yang tidak muncul**, yang dapat disebabkan oleh kesalahan dalam pemetaan tekstur pada objek dalam simulasi. Hal ini mungkin terkait dengan kesalahan dalam pengaturan file tekstur atau masalah dalam orientasi objek yang mengakibatkan tekstur tidak diterapkan dengan benar. Tanpa tekstur yang jelas, kamera kesulitan untuk membedakan permukaan kardus, yang mengarah pada ketidakmampuan scanner untuk mendeteksi dan memproses dokumen di atasnya.
2. **Resolusi Rendah pada Tekstur Kardus 2:**
   - Pada kardus 2, **resolusi tekstur yang terlalu rendah** mengakibatkan kualitas gambar yang buruk. Hal ini menyulitkan scanner untuk membedakan antara kardus dan dokumen yang terletak di atasnya. Ketika resolusi tekstur tidak cukup tinggi, detail penting seperti batas antara kardus dan dokumen menjadi kabur, mengurangi kemampuan kamera untuk mendeteksi dengan akurat posisi dan keberadaan dokumen yang perlu dipindai.
3. **Dampak pada Deteksi Dokumen:**
   - Karena kedua masalah di atas, **scanner tidak dapat membedakan dokumen dari kardus** dengan jelas. Gambar yang kabur atau tidak terdefinisi dengan baik menyebabkan kesulitan dalam mendeteksi elemen-elemen penting pada dokumen, seperti teks atau gambar yang perlu dipindai. Akibatnya, sistem scanner gagal untuk menangkap atau menampilkan gambar dokumen dengan akurat, sehingga fungsionalitas sistem terhambat.

Secara keseluruhan, masalah utama yang menyebabkan kegagalan dalam mendeteksi gambar dokumen adalah kesalahan pada **pemetaan tekstur kardus** dan **kualitas rendah tekstur kardus 2**, yang mengurangi akurasi deteksi objek dan memperburuk proses pemindaian dokumen.

3. Fruit Selector

Dalam simulasi **fruit selector** yang menggunakan arm hidrolik dengan kamera untuk membedakan buah apel hijau dan jeruk oranye, terdapat beberapa aspek penting yang perlu dianalisis terkait fungsionalitas dan potensi masalah dalam sistem ini:

1. **Fungsi Kamera dan Deteksi Warna:**
    - **Kamera** pada arm hidrolik berfungsi untuk memindai objek yang bergerak di atas **conveyor belt**, dan sistem deteksi warna digunakan untuk membedakan buah apel (hijau) dan jeruk (oranye). Dengan menggunakan pemrograman berbasis pengolahan citra (image processing), kamera dapat membedakan warna berdasarkan parameter tertentu, seperti ruang warna HSV (Hue, Saturation, Value), yang memungkinkan pengenalan buah berdasarkan warna.
2. **Penentuan Warna:**
    - **Pemilihan warna** sebagai parameter untuk membedakan buah adalah pendekatan yang efisien, tetapi dapat menjadi rentan terhadap **perubahan pencahayaan**, refleksi, atau gangguan warna lain di sekitar buah. Jika pencahayaan tidak optimal atau ada variasi warna yang tidak terduga pada buah (misalnya apel yang tidak sepenuhnya hijau atau jeruk yang memiliki warna kurang cerah), sistem dapat kesulitan dalam membedakan kedua buah tersebut dengan tepat.
3. **Proses Seleksi Buah:**
    - Setelah kamera berhasil mendeteksi buah berdasarkan warna, arm hidrolik harus **mengambil dan memindahkan buah ke keranjang yang sesuai** (apel ke keranjang hijau dan jeruk ke keranjang oranye). Gerakan arm hidrolik harus sangat presisi, terutama karena **konveyor belt terus bergerak** dan posisi buah dapat berubah dengan cepat.
    - Sistem pengontrol arm hidrolik dan pemrograman untuk **menghitung posisi dan pergerakan buah** menjadi sangat penting untuk memastikan bahwa arm hidrolik dapat dengan tepat memilih dan memindahkan buah ke keranjang yang sesuai tanpa salah tempat.