Analisis

1. Deteksi blob warna

Program ini adalah implementasi untuk deteksi blob warna menggunakan perangkat simulasi robot dari Webots. Secara garis besar, program ini memanfaatkan kamera yang dipasang pada robot untuk mendeteksi area warna dominan (blob) dalam gambar. Blob yang dapat dideteksi adalah merah, hijau, dan biru. Program ini kemudian mengatur pergerakan robot berdasarkan hasil deteksi tersebut.

Penjelasan Program

1. **Inisialisasi dan Pengaturan Awal**: Program dimulai dengan menginisialisasi perangkat robot, termasuk kamera untuk mengambil gambar dan motor untuk mengatur pergerakan roda. Kamera diaktifkan, dan dimensi gambar yang diambil (lebar dan tinggi) disimpan untuk analisis selanjutnya.
2. **Loop Utama**: Pada setiap langkah simulasi, gambar terbaru dari kamera diambil. Program menganalisis piksel di bagian tengah gambar untuk menghitung intensitas warna merah, hijau, dan biru. Hasil analisis ini digunakan untuk menentukan apakah ada blob warna dominan.
3. **Logika Deteksi Blob**:
    - Jika salah satu komponen warna (merah, hijau, atau biru) memiliki nilai tiga kali lebih besar dari komponen lainnya, program mengidentifikasi keberadaan blob dari warna tersebut.
    - Jika tidak ada warna yang dominan, robot terus berputar untuk mencari blob baru.
4. **Respon Terhadap Blob**:
    - Ketika blob terdeteksi, robot berhenti bergerak, mencetak pesan di konsol yang menunjukkan warna blob yang ditemukan, dan menyimpan gambar blob tersebut ke file di direktori pengguna.
    - Setelah itu, robot memasuki kondisi "jeda" untuk menghindari deteksi ulang blob yang sama.
5. **Pengaturan Kecepatan Motor**:
    - Robot bergerak dengan kecepatan tertentu untuk mencari blob jika tidak ditemukan blob.
    - Robot berhenti sepenuhnya ketika mendeteksi blob.
6. **Sistem Penyimpanan Gambar**: Program menggunakan fungsi sistem untuk menentukan direktori pengguna secara platform-independen (Windows atau Linux) dan menyimpan gambar dengan nama file yang sesuai dengan warna blob.

2. Auto focus

Program ini adalah implementasi sederhana dari fitur **autofokus kamera** dalam simulasi robot menggunakan Webots. Dengan memanfaatkan sensor jarak, program secara dinamis mengatur jarak fokus kamera berdasarkan objek yang terdeteksi, sehingga menghasilkan gambar yang lebih tajam pada objek tersebut.

Penjelasan Program

1. **Inisialisasi Robot dan Perangkat**:
    - Program dimulai dengan menginisialisasi perangkat yang digunakan, termasuk kamera, sensor jarak, dan motor untuk mengontrol roda.
    - Kamera diaktifkan untuk menangkap gambar, dan sensor jarak diaktifkan untuk mengukur jarak objek dari robot.
2. **Pengaturan Motor**:
    - Roda kiri dan kanan diatur untuk berputar secara berlawanan (kecepatan kiri negatif, kanan positif). Ini menyebabkan robot bergerak melingkar, sehingga dapat memindai area di sekitarnya.
3. **Loop Utama**:
    - Pada setiap siklus simulasi, program membaca nilai jarak dari sensor jarak.
    - Nilai ini kemudian digunakan untuk mengatur jarak fokus kamera melalui fungsi wb_camera_set_focal_distance(). Jarak fokus kamera diatur agar sesuai dengan objek yang terdeteksi, memungkinkan kamera menangkap gambar objek dengan lebih jelas.
4. **Autofokus Berdasarkan Jarak**:
    - Sensor jarak mengembalikan nilai dalam satuan milimeter, yang kemudian dikonversi ke meter (dengan membagi 1000).
    - Nilai jarak ini diatur sebagai jarak fokus kamera, sehingga fokus kamera selalu menyesuaikan dengan jarak objek di depan robot.
5. **Pembersihan dan Penghentian Program**:
    - Setelah loop utama selesai, program membersihkan sumber daya yang digunakan dengan memanggil wb_robot_cleanup().

3. Motion blur

Program ini adalah implementasi robot di Webots untuk mendeteksi warna blob (titik warna) seperti merah, hijau, atau biru dalam sebuah gambar yang diambil kamera. Program akan memproses gambar, mengidentifikasi warna dominan di bagian tengah gambar, dan menyimpan gambar tersebut jika blob ditemukan. Selain itu, robot memiliki perilaku "berputar" jika tidak menemukan warna yang diharapkan. Fitur ini secara tidak langsung memberikan efek **motion blur** dalam pendeteksian karena robot bergerak secara kontinu.

Penjelasan Program

1. Inisialisasi Robot dan Perangkat

- Fungsi wb_robot_init() digunakan untuk memulai simulasi.
- Kamera (camera) diaktifkan menggunakan wb_camera_enable().
- Dimensi kamera (width dan height) disimpan untuk membantu pengolahan gambar.
- Motor kiri dan kanan diatur untuk menggunakan mode kecepatan dengan wb_motor_set_position(left_motor, INFINITY).

2. Perilaku Robot Berdasarkan Keadaan

Program memiliki beberapa "state" tergantung pada kondisi pause_counter:

- **Case 1 (Blob ditemukan dan robot berhenti)**:
    - Robot berhenti selama pause_counter > 640 / time_step untuk fokus di depan blob.
- **Case 2 (Blob baru ditemukan, robot berputar sebentar)**:
    - Robot memutar arah tanpa menganalisis gambar untuk menghindari mendeteksi blob yang sama berulang kali.
- **Case 3 (Mencari Blob)**:
    - Jika pause_counter habis dan kamera tidak kosong, robot mulai menganalisis gambar.

3. Analisis Gambar

- Robot membaca citra dari kamera menggunakan wb_camera_get_image().
- Fokus analisis berada di area tengah gambar (sepertiga tengah gambar horizontal dan seperempat bawah secara vertikal).
- Setiap piksel dalam area tersebut dihitung nilai intensitasnya untuk **merah**, **hijau**, dan **biru**.

4. Logika Deteksi Blob

- Jika salah satu warna memiliki intensitas **tiga kali lebih besar** dari dua warna lainnya, robot menganggap blob ditemukan.
    - Blob merah jika red > 3 * green && red > 3 * blue.
    - Blob hijau jika green > 3 * red && green > 3 * blue.
    - Blob biru jika blue > 3 * red && blue > 3 * green.
- Jika tidak ada warna dominan, robot terus berputar mencari blob.

5. Tindakan Setelah Menemukan Blob

- Robot mencetak warna blob yang ditemukan.
- Gambar saat ini disimpan ke file:
    - Nama file ditentukan berdasarkan warna blob, seperti red_blob.png, green_blob.png, atau blue_blob.png.
    - Lokasi penyimpanan tergantung pada sistem operasi (Windows atau Unix-based).

6. Kecepatan Motor

- Robot mengatur kecepatan motor menggunakan wb_motor_set_velocity():
    - **Berputar**: Kecepatan kiri negatif dan kanan positif.
    - **Berhenti**: Kecepatan kedua motor diatur ke 0.

7. Pembersihan

- Program membersihkan sumber daya menggunakan wb_robot_cleanup() saat selesai.

Efek Motion Blur

Efek "motion blur" terjadi karena robot bergerak sambil menganalisis gambar. Ketika robot berputar, blob yang bergerak akan terlihat kabur, terutama jika kamera memiliki kecepatan refresh yang tinggi. Hal ini dapat dimanfaatkan untuk eksperimen, seperti simulasi visi robotik saat bergerak.

4. Noise mask

Program tersebut adalah sebuah implementasi dalam **Webots**, sebuah simulator robotika, untuk memanfaatkan perangkat kamera pada robot guna mendeteksi warna spesifik ("blob") dalam pandangan kamera. Program ini mengontrol robot agar berinteraksi dengan lingkungannya berdasarkan warna yang terdeteksi. Berikut adalah analisisnya secara general:

1. **Tujuan Program**
   Program ini bertujuan untuk mengidentifikasi warna merah, hijau, atau biru yang terlihat di tengah pandangan kamera robot. Berdasarkan hasil deteksi warna, robot melakukan tindakan spesifik seperti berhenti, menyimpan gambar, atau berputar untuk mencari warna lain.
2. **Pendefinisian Perangkat Robot**
   Program dimulai dengan inisialisasi perangkat robot, seperti kamera untuk menangkap gambar dan motor untuk menggerakkan roda kiri serta kanan. Kamera diaktifkan, dan dimensinya diambil untuk memproses piksel gambar.
3. **Struktur Logika Program**
   Program menggunakan sebuah **loop utama** yang terus berjalan selama simulasi aktif. Dalam loop ini, kamera mengambil gambar, yang kemudian diproses untuk menganalisis intensitas warna merah, hijau, dan biru di bagian tengah gambar. Berdasarkan hasil analisis ini, robot masuk ke dalam salah satu dari tiga kondisi:
   - **Tidak ada blob terdeteksi**: Robot terus berputar mencari warna baru.
   - **Blob terdeteksi**: Robot berhenti, menyimpan gambar warna tersebut ke direktori pengguna, dan menunggu beberapa saat.
   - **Proses transisi**: Robot memulai rotasi sebelum menganalisis gambar berikutnya untuk menghindari pendeteksian berulang.
4. **Analisis Gambar**
   Gambar dari kamera diproses dengan cara membaca intensitas piksel di area tertentu. Program membandingkan nilai intensitas warna untuk menentukan apakah ada warna dominan (blob) yang terlihat di tengah gambar.
5. **Interaksi dengan Pengguna**
   Jika sebuah warna berhasil terdeteksi, program mencetak pesan ke konsol dengan menggunakan warna ANSI untuk menyoroti warna yang ditemukan. Gambar warna tersebut juga disimpan secara otomatis di direktori pengguna untuk keperluan lebih lanjut.
6. **Kontrol Robot**
   Program menggunakan kecepatan roda kiri dan kanan untuk menentukan arah gerak robot. Jika warna terdeteksi, robot berhenti. Jika tidak ada warna, robot terus berputar dengan mengatur kecepatan berlawanan antara roda kiri dan kanan.
7. **Keunggulan dan Keterbatasan**

- **Keunggulan**: Program ini cukup modular dengan logika sederhana untuk mendeteksi warna dan mengontrol gerak robot. Penggunaan ANSI untuk warna pada konsol juga memberikan umpan balik visual yang menarik.
- **Keterbatasan**: Deteksi warna bergantung pada intensitas piksel, sehingga akurasi dapat terganggu oleh kondisi pencahayaan atau warna latar belakang. Area analisis juga terbatas pada bagian tengah gambar, sehingga objek yang berada di tepi kamera mungkin tidak terdeteksi.

8. **Potensi Pengembangan**
   Program dapat dikembangkan lebih lanjut dengan fitur seperti:
   - Menggunakan **algoritma pengolahan citra** lebih canggih untuk deteksi objek.
   - Menambahkan logika untuk menggerakkan robot menuju objek berwarna yang terdeteksi.
   - Integrasi sensor tambahan untuk memperkaya respons robot terhadap lingkungannya.

5. Object detection

Program ini merupakan contoh penerapan **object detection** menggunakan perangkat kamera di simulator **Webots**. Tidak hanya mendeteksi keberadaan objek, kamera juga mampu mengenali atribut objek seperti posisi, orientasi, ukuran, dan warna. Berikut adalah analisisnya:

1. Tujuan Program

Program ini dirancang untuk memanfaatkan fitur **recognition** pada kamera Webots untuk mendeteksi dan mengenali objek-objek di sekitarnya. Informasi objek yang terdeteksi, seperti model, ID, posisi relatif, orientasi, ukuran, dan warna, diambil dan ditampilkan melalui konsol.

2. Pendefinisian Perangkat Robot

- **Kamera**: Kamera diaktifkan dengan fitur *recognition* untuk mendeteksi objek.
- **Motor**: Motor kiri dan kanan dikendalikan dalam mode kecepatan agar robot terus bergerak secara berputar, memungkinkan kamera menangkap gambar lingkungan sekitarnya secara dinamis.

3. Struktur Logika Program

Program menggunakan **loop utama** untuk menjalankan langkah-langkah berikut secara berulang:

1. **Pengambilan Informasi Objek**
   Kamera mengambil data tentang jumlah objek yang terdeteksi. Program kemudian mengambil informasi detail untuk setiap objek yang terdeteksi.
2. **Output Atribut Objek**
   Untuk setiap objek, program mencetak atribut berikut ke konsol:
   - **Model dan ID**: Identitas objek, seperti jenis atau nama model.

    - **Posisi Relatif**: Posisi objek dalam koordinat relatif terhadap robot.
    - **Orientasi**: Orientasi objek menggunakan quaternions.
    - **Ukuran Objek**: Dimensi objek dalam unit simulasi.
    - **Posisi dan Ukuran pada Gambar Kamera**: Menentukan lokasi dan ukuran objek pada gambar yang dihasilkan kamera.
    - **Warna**: Menampilkan hingga beberapa warna dominan objek dalam format RGB.
3. **Pergerakan Robot**
   Robot terus bergerak dengan kecepatan tetap pada kedua rodanya, dengan konfigurasi roda kiri mundur dan roda kanan maju. Ini menyebabkan robot berputar, memberikan sudut pandang lebih luas untuk kamera dalam mendeteksi objek.

4. Kemampuan Deteksi dan Pengolahan Data

- **Fitur Deteksi**:
  Kamera Webots dengan *recognition* memungkinkan deteksi objek menggunakan atribut visual yang disediakan oleh simulator. Ini mencakup data posisi dan dimensi objek di dunia 3D simulasi.
- **Analisis Warna**:
  Program mengekstrak warna dominan dari objek. Data ini berguna dalam pengklasifikasian atau penandaan objek berdasarkan warna.
- **Lokasi pada Gambar**:
  Posisi objek dalam gambar membantu memahami lokasi objek dalam konteks pandangan robot.

5. Umpan Balik ke Pengguna

Program memberikan umpan balik ke pengguna melalui konsol. Informasi atribut objek yang rinci membantu pengguna menganalisis hasil deteksi objek secara visual tanpa integrasi ke logika lanjutan.

6. Keunggulan dan Keterbatasan

Keunggulan:

- **Detail Informasi**: Memberikan data yang sangat lengkap, mencakup posisi, orientasi, ukuran, dan warna.
- **Fleksibilitas**: Program dapat diperluas untuk menambahkan logika berbasis atribut objek (misalnya, mendekati objek tertentu atau menghindarinya).

Keterbatasan:

- **Lingkup Simulasi**: Deteksi berbasis *recognition* terbatas pada objek yang sudah terdefinisi dalam lingkungan simulasi Webots. Tidak ada pembelajaran mesin atau model pengenalan objek eksternal.
- **Kamera Statik**: Informasi posisi dan orientasi relatif hanya berguna dalam simulasi. Data tersebut mungkin kurang akurat untuk lingkungan nyata.

7. Potensi Pengembangan

- **Integrasi Respons Robot**:
  Menambahkan logika untuk merespons objek berdasarkan atributnya, seperti mendekati objek tertentu atau menghindari objek dengan warna atau ukuran tertentu.
- **Penggunaan AI/ML**:
  Mengintegrasikan kamera dengan algoritma pembelajaran mesin untuk meningkatkan pengenalan objek dengan data nyata.
- **Analisis Data yang Disimpan**:
  Informasi objek dapat disimpan dalam file untuk digunakan pada analisis lebih lanjut, seperti membangun peta atau analisis statistik.

6. Camera segmentation

Program ini menggunakan fitur **recognition segmentation** pada kamera di simulator **Webots** untuk menghasilkan dan menampilkan gambar hasil segmentasi objek yang terdeteksi. Berikut analisis lengkapnya:

1. Tujuan Program

Program ini bertujuan memanfaatkan fitur segmentasi pada kamera Webots untuk membedakan objek di lingkungan simulasi. Hasil segmentasi kemudian ditampilkan secara visual pada perangkat *display*.

2. Pendefinisian Perangkat Robot

- **Kamera**:
  Digunakan untuk menangkap gambar lingkungan simulasi serta menghasilkan gambar segmentasi berdasarkan objek yang dikenali.
- **Display**:
  Digunakan untuk menampilkan hasil segmentasi yang dihasilkan oleh kamera.
- **Motor**:
  Motor kiri dan kanan menggerakkan robot dalam pola rotasi untuk memungkinkan kamera menangkap berbagai sudut pandang lingkungan.

3. Struktur Logika Program

1. **Inisialisasi Perangkat**:
   Kamera, *display*, dan motor diinisialisasi. Kamera diaktifkan dengan fitur segmentasi dan pengenalan.
2. **Pergerakan Robot**:
   Robot diprogram untuk terus berputar dengan menggerakkan roda kiri mundur dan roda kanan maju. Ini memberikan sudut pandang dinamis untuk kamera.
3. **Pengambilan dan Penampilan Gambar Segmentasi**:
   - Mengecek apakah segmentasi kamera diaktifkan dan kamera siap (dengan memeriksa *sampling period*).
   - Jika data segmentasi tersedia, program mengambil gambar segmentasi dari kamera, membuat referensi gambar (*WbImageRef*), dan menampilkannya di perangkat *display*.

    - Setelah gambar segmentasi ditampilkan, referensi gambar dihapus untuk membebaskan memori.
4. **Loop Utama**:
   Program terus berjalan di loop utama hingga simulasi dihentikan, memungkinkan pengambilan gambar segmentasi dan tampilan *real-time*.

4. Penjelasan Fitur dan Alur Kerja

Recognition Segmentation:

- Fitur segmentasi pada kamera Webots memisahkan objek yang terdeteksi dari latar belakang, menghasilkan gambar segmentasi.
- Setiap objek diberi *label warna unik* dalam gambar segmentasi, sehingga memudahkan pengenalan objek.

Penerapan Tampilan Gambar Segmentasi:

- Gambar segmentasi yang dihasilkan menggunakan format **BGRA** (Blue, Green, Red, Alpha).
- Program memanfaatkan fungsi wb_display_image_paste untuk menampilkan gambar ini di perangkat *display*.

5. Keunggulan dan Keterbatasan

Keunggulan:

1. **Visualisasi Segmen yang Jelas**:
   Program memvisualisasikan hasil segmentasi secara langsung, membantu pengguna memahami deteksi objek secara intuitif.
2. **Kemudahan Implementasi**:
   Webots menyediakan API yang sederhana untuk segmentasi, mempermudah pengembangan.
3. **Dinamisitas Lingkungan**:
   Pergerakan robot memungkinkan pengambilan data dari berbagai sudut, menciptakan sudut pandang yang lebih luas.

Keterbatasan:

1. **Terbatas pada Simulasi**:
   Fitur segmentasi ini hanya berlaku dalam lingkungan simulasi Webots, dengan objek yang telah didefinisikan sebelumnya.
2. **Ketergantungan pada Performa Display**:
   Visualisasi segmentasi tergantung pada *display*. Jika resolusi *display* rendah, detail segmentasi mungkin kurang jelas.
3. **Tidak Ada Logika Tindak Lanjut**:
   Program hanya menampilkan hasil segmentasi tanpa menggunakan data tersebut untuk tindakan, seperti navigasi berbasis segmentasi.

6. Potensi Pengembangan

1. **Integrasi dengan Logika Navigasi**:
   Menggunakan data segmentasi untuk mendukung navigasi berbasis objek, seperti menghindari rintangan atau mengikuti objek tertentu.
2. **Penyimpanan Data Segmentasi**:
   Menyimpan gambar segmentasi ke file untuk analisis lebih lanjut, misalnya dengan algoritma *computer vision* eksternal.
3. **Optimasi Visualisasi**:
   Menambahkan fitur interaktif pada *display*, seperti penyorotan objek tertentu atau menampilkan label objek.
4. **Implementasi AI/ML**:
   Menggunakan data segmentasi sebagai input untuk model pembelajaran mesin guna meningkatkan pengenalan dan interaksi dengan objek.

7. Spherical camera

Program ini adalah simulasi yang menggunakan **kamera spherical** untuk mendeteksi warna blob (*red*, *green*, *blue*) dan mengatur kecepatan motor berdasarkan nilai sensor jarak (*distance sensor*). Berikut adalah analisis mendetail:

1. Tujuan Program

Simulasi ini dirancang untuk menggabungkan penggunaan kamera dengan sensor jarak untuk mendeteksi warna spesifik (*red*, *green*, *blue*), menghitung posisi sudut blob warna, dan mengontrol gerakan robot berdasarkan pembacaan sensor jarak.

2. Perangkat yang Digunakan

1. **Kamera**:
   - Kamera spherical mendeteksi warna *red*, *green*, dan *blue* dalam gambar.
   - Informasi warna digunakan untuk menghitung koordinat blob dan sudutnya.
2. **Sensor Jarak** (*ultrasonic sensors*):
   - Dua sensor jarak, *us0* dan *us1*, mendeteksi jarak ke objek di sekitar robot.
   - Nilai sensor digunakan untuk menentukan kecepatan motor berdasarkan matriks koefisien.
3. **Motor**:
   - Motor kiri dan kanan mengontrol kecepatan robot, memungkinkan gerakan berbasis pembacaan dari kamera dan sensor jarak.

3. Struktur Logika Program

Inisialisasi:

1. **Kamera**:
   - Kamera spherical diaktifkan dengan interval waktu dua kali nilai TIME_STEP untuk memberikan akurasi pengambilan gambar.
   - Ukuran gambar (width dan height) digunakan untuk mengakses setiap piksel.

2. **Sensor Jarak**:
    - Dua sensor ultrasonik diaktifkan untuk membaca nilai jarak dengan interval waktu TIME_STEP.
    - Matriks coefficients menentukan kontribusi masing-masing sensor ke perhitungan kecepatan motor.
3. **Motor**:
    - Motor kiri dan kanan disiapkan dalam mode kontrol kecepatan (*speed control*) dengan posisi target diatur ke **INFINITY**.

Loop Utama:

1. **Pembacaan Data**:
    - **Sensor Jarak**: Nilai dari *us0* dan *us1* dibaca.
    - **Kamera**: Gambar diambil, dan setiap piksel dianalisis untuk mendeteksi keberadaan warna *red*, *green*, dan *blue*.
2. **Deteksi Blob Warna**:
    - Setiap piksel gambar dianalisis menggunakan nilai intensitas merah, hijau, dan biru:
        - Jika intensitas merah melebihi THRESHOLD, piksel dianggap bagian dari blob merah.
        - Logika serupa diterapkan untuk warna hijau dan biru.
    - Posisi blob terakhir untuk setiap warna dicatat dalam array color_index.
3. **Penghitungan Sudut Blob**:
    - Fungsi coord2D_to_angle menghitung sudut blob warna dalam koordinat polar berdasarkan posisi x dan y.
4. **Penyesuaian Kecepatan Motor**:
    - Kecepatan motor dihitung menggunakan nilai sensor jarak dan matriks coefficients.
    - Robot bergerak dengan menambahkan kecepatan dasar (3.0) dengan hasil perhitungan berbasis sensor.
5. **Output Data**:
    - Informasi posisi terakhir dari masing-masing warna blob dan sudutnya dicetak ke terminal menggunakan ANSI_CLEAR_CONSOLE untuk tampilan yang diperbarui secara *real-time*.

4. Analisis Fitur

Deteksi Warna:

- Program memanfaatkan intensitas RGB dari gambar untuk mendeteksi warna spesifik.
- Algoritma cukup sederhana, hanya bergantung pada batas nilai THRESHOLD.

Penghitungan Sudut Blob:

- Fungsi coord2D_to_angle menentukan sudut blob warna dalam radian berdasarkan koordinat *2D*. Ini berguna untuk navigasi berbasis sudut atau tindakan berbasis lokasi.

Penggunaan Sensor Jarak:

- Matriks coefficients memberikan kontribusi sensor jarak terhadap kecepatan motor. Konfigurasi ini memungkinkan robot menghindari objek atau menyesuaikan arah berdasarkan jarak ke hambatan.

5. Keunggulan dan Keterbatasan

Keunggulan:

1. **Integrasi Sensor**:
    - Menggabungkan data dari kamera dan sensor jarak untuk menghasilkan perilaku robot yang dinamis.
2. **Deteksi Warna yang Efisien**:
    - Program dapat melacak tiga warna utama (*red*, *green*, *blue*) secara simultan.
3. **Logika Navigasi Sederhana**:
    - Kecepatan motor disesuaikan berdasarkan jarak objek, memungkinkan navigasi berbasis sensor.

Keterbatasan:

1. **Ketergantungan pada Threshold**:
    - Deteksi warna bergantung pada batas nilai THRESHOLD, yang mungkin tidak akurat dalam kondisi pencahayaan berbeda.
2. **Tidak Ada Tindakan Berdasarkan Blob**:
    - Meskipun warna blob terdeteksi, data ini tidak digunakan untuk tindakan seperti mengikuti warna tertentu.
3. **Kinerja pada Resolusi Tinggi**:
    - Pemrosesan setiap piksel dapat menjadi lambat jika resolusi kamera tinggi, mengurangi performa *real-time*.

6. Potensi Pengembangan

1. **Navigasi Berbasis Blob**:
    - Menggunakan data posisi blob untuk menentukan arah gerak robot, misalnya mengikuti blob merah atau menghindari blob biru.
2. **Optimasi Deteksi Warna**:
    - Mengganti logika sederhana berbasis threshold dengan algoritma *machine learning* untuk meningkatkan akurasi deteksi warna.
3. **Penyimpanan Data**:
    - Menyimpan data blob seperti posisi dan sudut untuk analisis lebih lanjut atau visualisasi.
4. **Peningkatan Perilaku Robot**:
    - Menggabungkan logika berbasis blob dengan data sensor jarak untuk perilaku robot yang lebih cerdas.