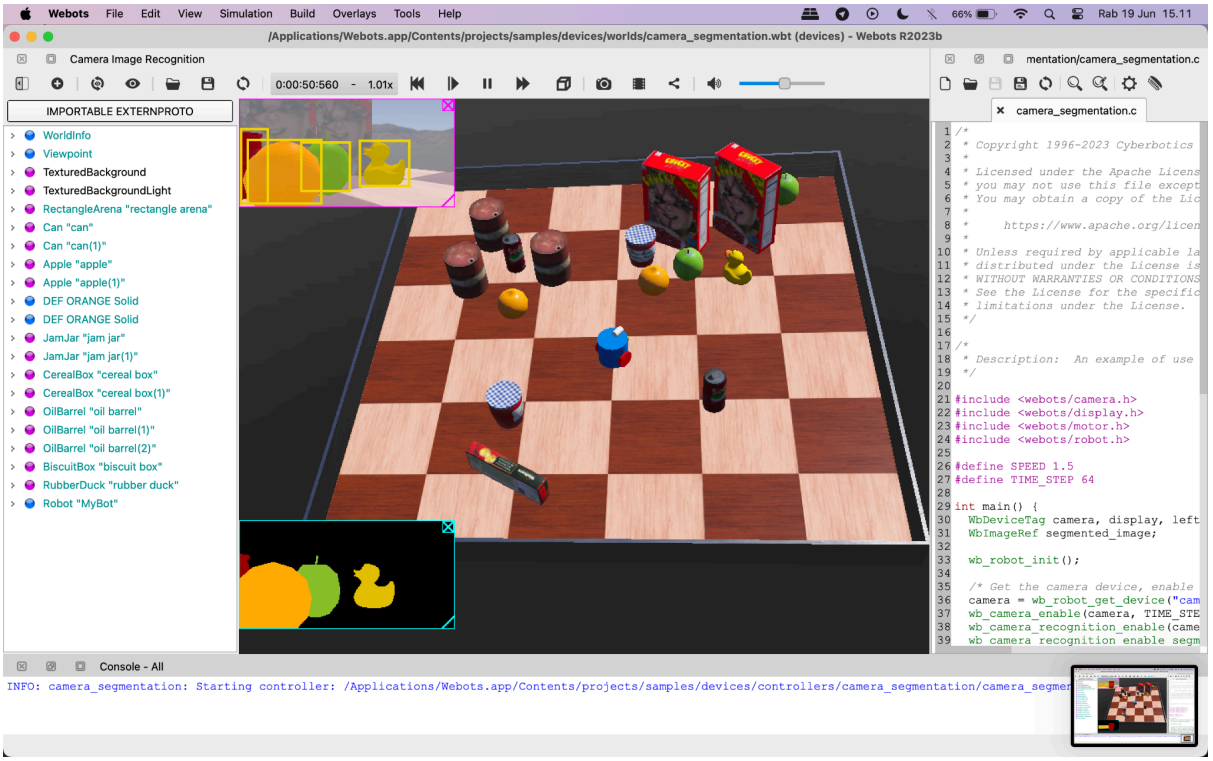
TUGAS ROBOTIKA WEEK 10
Alfian Mohamad Firdaus | 1103204002

Simulasi 1:



Kode di atas adalah contoh penggunaan perangkat kamera dengan kemampuan segmentasi pengenalan objek di Webots, sebuah platform simulasi robotika. Berikut adalah penjelasan rinci dari kode tersebut:

1. **Inklusi Library:**

   #include <webots/camera.h>

   #include <webots/display.h>

   #include <webots/motor.h>

   #include <webots/robot.h>

   Baris-baris ini mengimpor header file yang diperlukan untuk menggunakan API Webots untuk kamera, display, motor, dan robot.

2. **Mendefinisikan Konstanta:**

   #define SPEED 1.5
   #define TIME_STEP 64

   Konstanta SPEED mendefinisikan kecepatan motor, dan TIME_STEP mendefinisikan langkah waktu untuk simulasi dalam milidetik.

3. **Fungsi `main`**

   int main() {

   Fungsi utama program dimulai di sini.

4. **Inisialisasi Perangkat**

   WbDeviceTag camera, display, left_motor, right_motor;

   WbImageRef segmented_image;

   wb_robot_init();

   Menginisialisasi robot dan mendeklarasikan variabel untuk perangkat kamera, display, motor kiri dan kanan, serta referensi gambar segmentasi.

5. **Mengatur Kamera**

   camera = wb_robot_get_device("camera");

   wb_camera_enable(camera, TIME_STEP);

   wb_camera_recognition_enable(camera, TIME_STEP);

   wb_camera_recognition_enable_segmentation(camera);

   const int width = wb_camera_get_width(camera);

   const int height = wb_camera_get_height(camera);

   Mendapatkan perangkat kamera, mengaktifkan kamera, pengenalan, dan segmentasi dengan langkah waktu yang ditentukan. Kemudian mendapatkan lebar dan tinggi kamera.

6. **Mengatur Display**

   display = wb_robot_get_device("segmented image display");

   Mendapatkan perangkat display yang akan digunakan untuk menampilkan gambar yang disegmentasi.

7. Mengatur Motor

```
\left_motor = wb_robot_get_device("left wheel motor");
right_motor = wb_robot_get_device("right wheel motor");
wb_motor_set_position(left_motor, INFINITY);
wb_motor_set_position(right_motor, INFINITY);
wb_motor_set_velocity(left_motor, -SPEED);
wb_motor_set_velocity(right_motor, SPEED);
```

Mendapatkan perangkat motor untuk roda kiri dan kanan, mengatur posisi target motor ke tak hingga untuk kontrol kecepatan, dan menetapkan kecepatan motor.

8. Loop Utama

```
while (wb_robot_step(TIME_STEP) != -1) {
if (wb_camera_recognition_is_segmentation_enabled(camera) &&
wb_camera_recognition_get_sampling_period(camera) > 0) {const unsigned char
*data =wb_camera_recognition_get_segmentation_image(camera);
if (data) {
segmented_image = wb_display_image_new(display, width, height, data,
WB_IMAGE_BGRA);
wb_display_image_paste(display, segmented_image, 0, 0, false);
wb_display_image_delete(display, segmented_image);
                    }
          }
 }
```

Pada loop utama, program memeriksa apakah segmentasi pengenalan kamera diaktifkan dan memiliki periode pengambilan sampel yang lebih besar dari 0. Jika ya, program mengambil gambar segmentasi dari kamera dan menampilkan gambar tersebut di perangkat display. Gambar yang ditampilkan dihapus setelah dipaste.

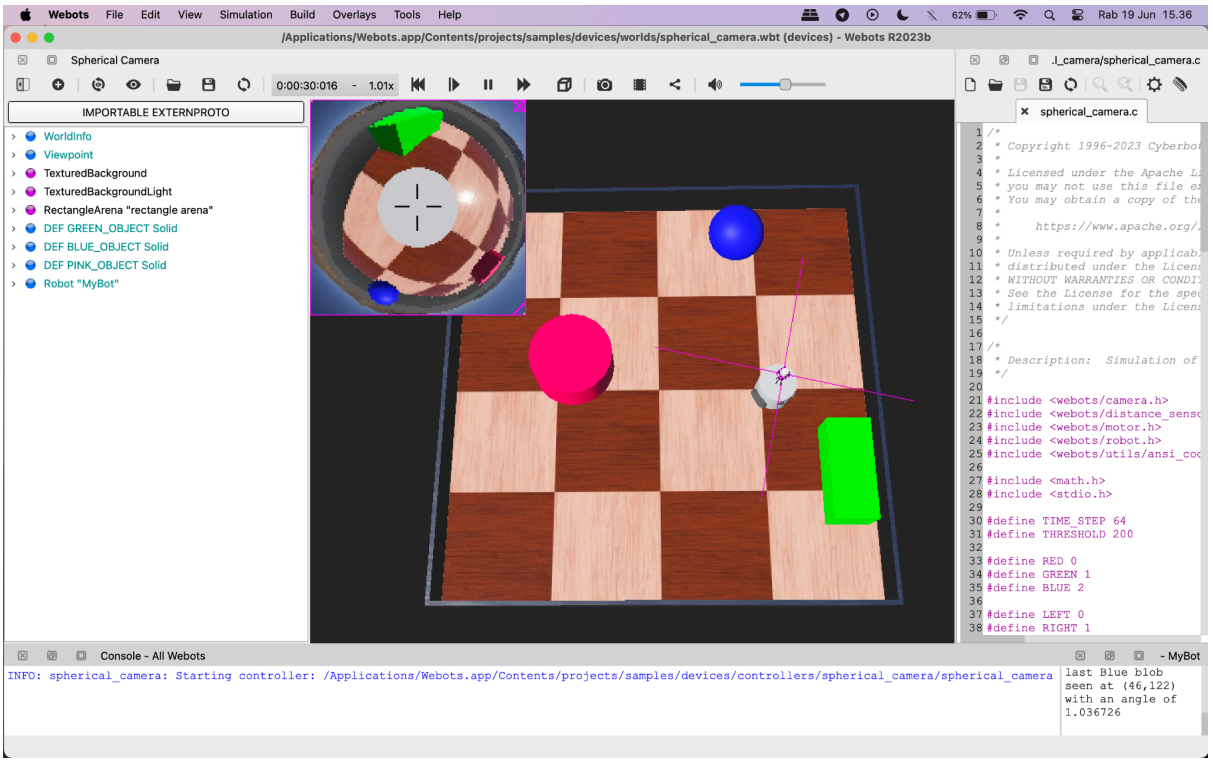
9. Bersihkan dan Akhiri Program

```
wb_robot_cleanup();
return 0;
}
```

Membersihkan sumber daya yang digunakan oleh Webots dan mengakhiri program.

TUGAS ROBOTIKA WEEK 10
Alfian Mohamad Firdaus | 1103204002

Simulasi 2:



Kode di atas adalah contoh simulasi penggunaan kamera spherical di Webots, sebuah platform simulasi robotika. Kode ini menggunakan kamera untuk mendeteksi blob berwarna dan menggunakan sensor jarak untuk mengendalikan pergerakan robot. Berikut adalah penjelasan rinci dari kode tersebut:

1. **Inklusi Library**

   #include <webots/camera.h>

   #include <webots/distance_sensor.h>

   #include <webots/motor.h>

   #include <webots/robot.h>

   #include <webots/utils/ansi_codes.h>

   #include <math.h>

   #include <stdio.h>

   Baris-baris ini mengimpor header file yang diperlukan untuk menggunakan API Webots untuk kamera, sensor jarak, motor, robot, dan utilitas ANSI untuk manipulasi output terminal. Selain itu, library `math.h` dan `stdio.h` diimpor untuk fungsi matematika dan input/output.

2. **Mendefinisikan Konstanta**

```
#define TIME_STEP 64
#define THRESHOLD 200

#define RED 0
#define GREEN 1
#define BLUE 2

#define LEFT 0
#define RIGHT 1

#define X 0
#define Y 1
```

Konstanta `TIME_STEP` mendefinisikan langkah waktu untuk simulasi dalam milidetik. `THRESHOLD` digunakan untuk mendefinisikan nilai ambang batas warna. `RED`, `GREEN`, dan `BLUE` adalah indeks untuk warna, sementara `LEFT` dan `RIGHT` adalah indeks untuk sensor jarak dan motor. `X` dan `Y` adalah indeks untuk koordinat.

3. **Fungsi `coord2D_to_angle`**

```
double coord2D_to_angle(double x, double y) {
  if (x > 0.0 && y >= 0.0)
    return atan(y / x);
  else if (x > 0.0 && y < 0.0)
   return atan(y / x) + 2.0 * M_PI;
  else if (x < 0.0)
   return atan(y / x) + M_PI;
  else if (x == 0.0 && y > 0.0)
   return M_PI_2;
  else if (x == 0.0 && y < 0.0)
    return 3.0 * M_PI_2;
  else /* (x == 0.0 && y == 0.0) */
    return 0.0;
```

```
}
```

Fungsi ini mengkonversi koordinat 2D menjadi sudut dalam radian. Fungsi ini menggunakan fungsi `atan` untuk menghitung sudut dari koordinat `(x, y)`.

4. **Fungsi `main`**

```
int main(int argc, char **argv) {
```

Fungsi utama program dimulai di sini.

5. **Inisialisasi**

```
 // iterator used to parse loops
 int i, k;

 // init Webots stuff
 wb_robot_init();

 // init camera
 WbDeviceTag camera = wb_robot_get_device("camera");
 wb_camera_enable(camera, 2 * TIME_STEP);
 int width = wb_camera_get_width(camera);
 int height = wb_camera_get_height(camera);
 int color_index[3][2] = {{0, 0}, {0, 0}, {0, 0}};
 int x, y, r, g, b;
```

Menginisialisasi robot dan perangkat kamera, serta mendapatkan lebar dan tinggi gambar kamera. Array `color_index` digunakan untuk menyimpan koordinat terakhir blob warna yang terdeteksi.

6. **Inisialisasi Sensor Jarak**

```
 // init distance sensors
 WbDeviceTag us[2];
 double us_values[2];
 double coefficients[2][2] = {{6.0, -3.0}, {-5.0, 4.0}};
 us[LEFT] = wb_robot_get_device("us0");
 us[RIGHT] = wb_robot_get_device("us1");
```

```
for (i = 0; i < 2; i++)
  wb_distance_sensor_enable(us[i], TIME_STEP);
```

Menginisialisasi dua sensor jarak (ultrasonik) dan koefisien yang digunakan untuk menghitung kecepatan motor berdasarkan nilai sensor.

7. Inisialisasi Motor

```
// get a handler to the motors and set target position to infinity (speed control)
  WbDeviceTag left_motor = wb_robot_get_device("left wheel motor");
  WbDeviceTag right_motor = wb_robot_get_device("right wheel motor");
  wb_motor_set_position(left_motor, INFINITY);
  wb_motor_set_position(right_motor, INFINITY);
  wb_motor_set_velocity(left_motor, 0.0);
  wb_motor_set_velocity(right_motor, 0.0);
```

Mendapatkan perangkat motor untuk roda kiri dan kanan, mengatur posisi target motor ke tak hingga untuk kontrol kecepatan, dan menetapkan kecepatan motor awal ke 0.

8. Inisialisasi Kecepatan

```
 // init speed values
double speed[2];
```

Mendeklarasikan array `speed` untuk menyimpan nilai kecepatan motor.

9. Loop Utama

```
  while (wb_robot_step(TIME_STEP) != -1) {
    // read sensors
    const unsigned char *image = wb_camera_get_image(camera);
    for (i = 0; i < 2; i++)
      us_values[i] = wb_distance_sensor_get_value(us[i]);

    // compute speed
    for (i = 0; i < 2; i++) {
      speed[i] = 0.0;
      for (k = 0; k < 2; k++)
        speed[i] += us_values[k] * coefficients[i][k];
```

```
}

// compute blob direction
for (y = 0; y < height; y++) {
  for (x = 0; x < width; x++) {
    r = wb_camera_image_get_red(image, width, x, y);
    g = wb_camera_image_get_green(image, width, x, y);
    b = wb_camera_image_get_blue(image, width, x, y);
    if (r > THRESHOLD && g < THRESHOLD && b < THRESHOLD) {
      color_index[RED][X] = x;
      color_index[RED][Y] = y;
    } else if (r < THRESHOLD && g > THRESHOLD && b < THRESHOLD) {
      color_index[GREEN][X] = x;
      color_index[GREEN][Y] = y;
    } else if (r < THRESHOLD && g < THRESHOLD && b > THRESHOLD) {
      color_index[BLUE][X] = x;
      color_index[BLUE][Y] = y;
    }
  }
}
```

Pada loop utama, program membaca gambar dari kamera dan nilai dari sensor jarak. Berdasarkan nilai sensor jarak dan koefisien yang ditentukan, program menghitung kecepatan motor. Kemudian, program menghitung arah blob berdasarkan warna (merah, hijau, biru) yang terdeteksi dalam gambar.

10. Menampilkan Hasil

```
// print results
ANSI_CLEAR_CONSOLE();
for (i = 0; i < 3; i++)
  // clang-format off
    // clang-format 11.0.0 is not compatible with previous versions with respect to nested conditional operators
    printf("last %s blob seen at (%d,%d) with an angle of %f\n",
           (i == GREEN) ? "Green" :
```

```
            (i == RED)   ? "Red" :
                        "Blue",
             color_index[i][X], color_index[i][Y],
                                coord2D_to_angle((double)(color_index[i][X] + width / 2),
(double)(color_index[i][Y] + height / 2)));
   // clang-format on
```

Program membersihkan konsol dan mencetak hasil koordinat dan sudut blob warna terakhir yang terdeteksi.

11. Mengatur Kecepatan Motor

```
   // set actuators
   wb_motor_set_velocity(left_motor, 3.0 + speed[LEFT]);
   wb_motor_set_velocity(right_motor, 3.0 + speed[RIGHT]);
  }

  wb_robot_cleanup();

  return 0;
}
```

Mengatur kecepatan motor berdasarkan hasil perhitungan sebelumnya dan membersihkan sumber daya yang digunakan oleh Webots setelah loop utama selesai.