Nama : Faraday Barr Fatahillah
NIM : 1103213028
Kelas : TK-45-02

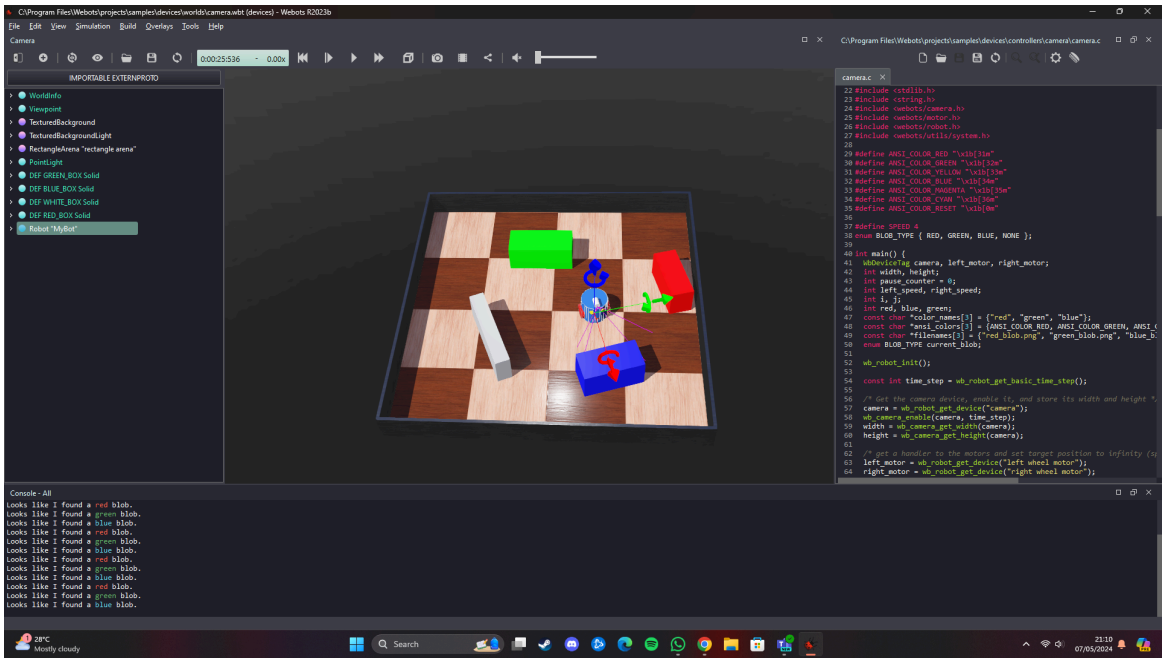
Tugas Week 10 Robotika

Topik yang dipilih

- Camera robot untuk mendeteksi blob warna (Merah, Hijau, dan Biru)
- Deteksi Objek dengan Kamera dan Pengenalan Objek pada Robot

Pembahasan Topik

1. **Camera Robot untuk Mendeteksi Blob Warna (Merah, Hijau, dan Biru)**



Robot di atas memiliki fungsi untuk mengidentifikasi warna yang telah ditangkap menggunakan kameranya dan memberikan output di log terminal. Berikut merupakan Source Code serta penjelasannya.

Source Code

```
// Inisialisasi Library
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
// Inisialisasi Kamera Webots (Robot sample yang telah dibuat oleh webots)
#include <webots/camera.h>
// Inisialisasi motor/penggerak dari robot
#include <webots/motor.h>
// Inisialisasi robot
#include <webots/robot.h>
```

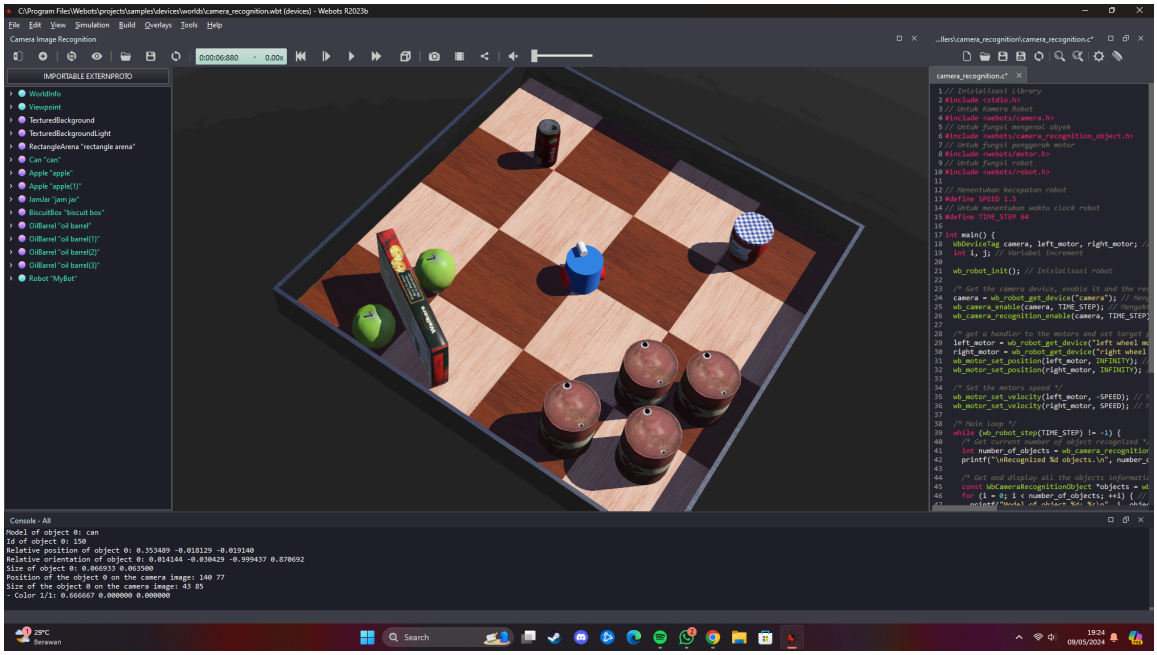
```
// Inisialisasi untuk manipulasi system (console)
#include <webots/utils/system.h>

// Inisialisasi Warna dengan bentuk ANSI
#define ANSI_COLOR_RED "\x1b[31m"
#define ANSI_COLOR_GREEN "\x1b[32m"
#define ANSI_COLOR_YELLOW "\x1b[33m"
#define ANSI_COLOR_BLUE "\x1b[34m"
#define ANSI_COLOR_MAGENTA "\x1b[35m"
#define ANSI_COLOR_CYAN "\x1b[36m"
#define ANSI_COLOR_RESET "\x1b[0m"

// Menentukan Variabel Global SPEED dengan nilai 4
#define SPEED 4
enum BLOB_TYPE { RED, GREEN, BLUE, NONE }; // Menentukan tipe-tipe warna
yang ada

int main() {
  WbDeviceTag camera, left_motor, right_motor; // Menentukan tag robot untuk motor
kanan dan kiri serta camera
  int width, height; // Variabel Penentuan tinggi dan lebar
  int pause_counter = 0; // Variabel untuk stop robot saat mendeteksi warna
  int left_speed, right_speed; // Variabel untuk kecepatan motor kanan dan kiri
  int i, j; // Variabel iterasi
  int red, blue, green; // Variabel untuk menentukan Red Blue atau Green
  const char *color_names[3] = {"red", "green", "blue"}; // Variabel Array untuk
menentukan output warna yang direkam di console
  const char *ansi_colors[3] = {ANSI_COLOR_RED, ANSI_COLOR_GREEN,
ANSI_COLOR_BLUE}; // Variabel Array untuk menentukan warna output warna yang
direkam di console
  const char *filenames[3] = {"red_blob.png", "green_blob.png", "blue_blob.png"}; //
Variabel penentuan nama file gambar setelah menangkap gambar oleh kamera
  enum BLOB_TYPE current_blob;

  wb_robot_init(); // Inisialisasi robot untuk diproses

  const int time_step = wb_robot_get_basic_time_step(); // Mengambil waktu clock dari
robot

  camera = wb_robot_get_device("camera"); // Inisialisasi kamera dari robot
  wb_camera_enable(camera, time_step); // Menyalakan kamera dengan waktu clock
sesuai dari robot
  width = wb_camera_get_width(camera); // Menentukan lebar dari kamera
  height = wb_camera_get_height(camera); // Menentukan tinggi dari kamera

  left_motor = wb_robot_get_device("left wheel motor"); // Inisialisasi motor roda kiri
  right_motor = wb_robot_get_device("right wheel motor"); // Inisialisasi motor roda
kanan
  wb_motor_set_position(left_motor, INFINITY); // Menentukan posisi motor kiri Infinity
agar bisa bergerak liniear atau gerakan rotasi
```

```
  wb_motor_set_position(right_motor, INFINITY); // Menentukan posisi motor kanan Infinity agar bisa bergerak liniear atau gerakan rotasi
  wb_motor_set_velocity(left_motor, 0.0); // Menentukan kecepatan awal motor kiri 0
  wb_motor_set_velocity(right_motor, 0.0); // Menentukan kecepatan awal motor kanan 0

  while (wb_robot_step(time_step) != -1) { // looping untuk menjalankan robot dan akan loop jika nilai step bukan -1
    const unsigned char *image = wb_camera_get_image(camera); // Variabel untuk menaruh data gambar yang telah diambil

    if (pause_counter > 0) // mengurangi pause counter kalau menangkap blob di kamera
      pause_counter--;

    if (pause_counter > 640 / time_step) { // Kondisi saat robot menemukan blob dan menunggu di depannya
      left_speed = 0;
      right_speed = 0;
    }

    else if (pause_counter > 0) { // Kondisi saat robot menemukan blob dan mulai belok dan tidak menangkap gambar lagi karena takut blobnya itu sama
      left_speed = -SPEED;
      right_speed = SPEED;
    }

    // Kondisi saat robot menemukan blob baru dan di analisis (Robot stop)
    else if (!image) {  // image may be NULL if Robot.synchronization is FALSE
      left_speed = 0;
      right_speed = 0;
    } else {  // pause_counter == 0
      /* Reset the sums */
      red = 0;
      green = 0;
      blue = 0;

      // Saat stop robot cek warna di pixel tengah dari kamera dan mengecek warna apakah warna merah biru atau hijau
      for (i = width / 3; i < 2 * width / 3; i++) {
        for (j = height / 2; j < 3 * height / 4; j++) {
          red += wb_camera_image_get_red(image, width, i, j);
          blue += wb_camera_image_get_blue(image, width, i, j);
          green += wb_camera_image_get_green(image, width, i, j);
        }
      }

      // Jika tangkapan kamera itu lebih banyak di salah satu warna maka warna tersebut adalah warna yang benar
      if ((red > 3 * green) && (red > 3 * blue))
```

```
      current_blob = RED;
    else if ((green > 3 * red) && (green > 3 * blue))
      current_blob = GREEN;
    else if ((blue > 3 * red) && (blue > 3 * green))
      current_blob = BLUE;
    else
      current_blob = NONE;

    // Jika tidak terdeteksi blob, maka akan lanjut berputar
    if (current_blob == NONE) {
      left_speed = -SPEED;
      right_speed = SPEED;
    }

    // Blob terdeteksi, robot berhenti, menyimpan gambar, dan mengubah state nya,
dan memberikan perintah di console
    else {
      left_speed = 0;
      right_speed = 0;
      printf("Looks like I found a %s%s%s blob.\n", ansi_colors[current_blob],
color_names[current_blob], ANSI_COLOR_RESET);
      // compute the file path in the user directory
      char *filepath;
#ifdef _WIN32
      const char *user_directory =
wbu_system_short_path(wbu_system_getenv("USERPROFILE"));
      filepath = (char *)malloc(strlen(user_directory) + 16);
      strcpy(filepath, user_directory);
      strcat(filepath, "\\");
#else
      const char *user_directory = wbu_system_getenv("HOME");
      filepath = (char *)malloc(strlen(user_directory) + 16);
      strcpy(filepath, user_directory);
      strcat(filepath, "/");
#endif
      strcat(filepath, filenames[current_blob]);
      wb_camera_save_image(camera, filepath, 100);
      free(filepath); // Menghapus data filepath
      pause_counter = 1280 / time_step; // Mengubah pause counter
    }
   }

   // Menentukan kecepatan dari motor
   wb_motor_set_velocity(left_motor, left_speed);
   wb_motor_set_velocity(right_motor, right_speed);
 }

 wb_robot_cleanup(); // Menghapus data-data pada robot dan mengulangnya dari
awal
```

```
  return 0;
}
```

2. Deteksi Objek dengan Kamera dan Pengenalan Objek pada Robot



Robot di atas memiliki fungsi untuk mendeteksi objek yang ada di depannya pada saat kamera menyorot. Data-data dari objek sendiri terdapat pada library yang sudah dipanggil pada saat inisialisasi library. Terdapat berbagai macam objek yang sudah dimasukkan dan tinggal kita rekam. Berikut merupakan penjelasan dari Source Code robot tersebut

Source Code

```
// Inisialisasi Library
#include <stdio.h>
// Untuk Kamera Robot
#include <webots/camera.h>
// Untuk fungsi mengenal obyek
#include <webots/camera_recognition_object.h>
// Untuk fungsi penggerak motor
#include <webots/motor.h>
// Untuk fungsi robot
#include <webots/robot.h>

// Menentukan kecepatan robot
#define SPEED 1.5
// Untuk menentukan waktu clock robot
```

```
#define TIME_STEP 64

int main() {
  WbDeviceTag camera, left_motor, right_motor; // Inisialisasi variabel perangkat yang digunakan
  int i, j; // Variabel increment

  wb_robot_init(); // Inisialisasi robot

  /* Get the camera device, enable it and the recognition */
  camera = wb_robot_get_device("camera"); // Mengambil perangkat Kamera
  wb_camera_enable(camera, TIME_STEP); // Mengaktifkan kamera dengan kecepatan clock
  wb_camera_recognition_enable(camera, TIME_STEP); // Mengaktifkan recognition function untuk kamera dengan kecepatan clock

  /* get a handler to the motors and set target position to infinity (speed control). */
  left_motor = wb_robot_get_device("left wheel motor"); // Menentukan motor kiri
  right_motor = wb_robot_get_device("right wheel motor"); // Menentukan motor kanan
  wb_motor_set_position(left_motor, INFINITY); // Menentukan posisi motor kiri Infinity agar bisa bergerak liniear atau gerakan rotasi
  wb_motor_set_position(right_motor, INFINITY); // Menentukan posisi motor kanan Infinity agar bisa bergerak liniear atau gerakan rotasi

  /* Set the motors speed */
  wb_motor_set_velocity(left_motor, -SPEED); // Menentukan speed motor kiri
  wb_motor_set_velocity(right_motor, SPEED); // Menentukan Speed Motor kanan

  /* Main loop */
  while (wb_robot_step(TIME_STEP) != -1) {
    /* Get current number of object recognized */
    int number_of_objects = wb_camera_recognition_get_number_of_objects(camera); // Memanggil fungsi object recognition dan mengembalikan id
    printf("\nRecognized %d objects.\n", number_of_objects); // banyak hasil banyak object yang direkam

    /* Get and display all the objects information */
    const WbCameraRecognitionObject *objects = wb_camera_recognition_get_objects(camera); // Membuat object untuk memanggil komponen objek nanti
    for (i = 0; i < number_of_objects; ++i) { // Looping pirnt banyak objek
      printf("Model of object %d: %s\n", i, objects[i].model); // Print model object
      printf("Id of object %d: %d\n", i, objects[i].id); // Print id object
      printf("Relative position of object %d: %lf %lf %lf\n", i, objects[i].position[0], objects[i].position[1],
             objects[i].position[2]); // print posisi object
      printf("Relative orientation of object %d: %lf %lf %lf %lf\n", i, objects[i].orientation[0], objects[i].orientation[1],
             objects[i].orientation[2], objects[i].orientation[3]); // print orientasi object
      printf("Size of object %d: %lf %lf\n", i, objects[i].size[0], objects[i].size[1]); // print
```

```
ukuran object
      printf("Position of the object %d on the camera image: %d %d\n", i,
objects[i].position_on_image[0],
             objects[i].position_on_image[1]); // print posisi object pada kamera
      printf("Size of the object %d on the camera image: %d %d\n", i,
objects[i].size_on_image[0], objects[i].size_on_image[1]); // print ukuran object pada
kamera
      for (j = 0; j < objects[i].number_of_colors; ++j) //looping untuk print warna object
yang ada
        printf("- Color %d/%d: %lf %lf %lf\n", j + 1, objects[i].number_of_colors,
objects[i].colors[3 * j],
               objects[i].colors[3 * j + 1], objects[i].colors[3 * j + 2]);
    }
  }

  wb_robot_cleanup(); // reset robot

  return 0;
}
```