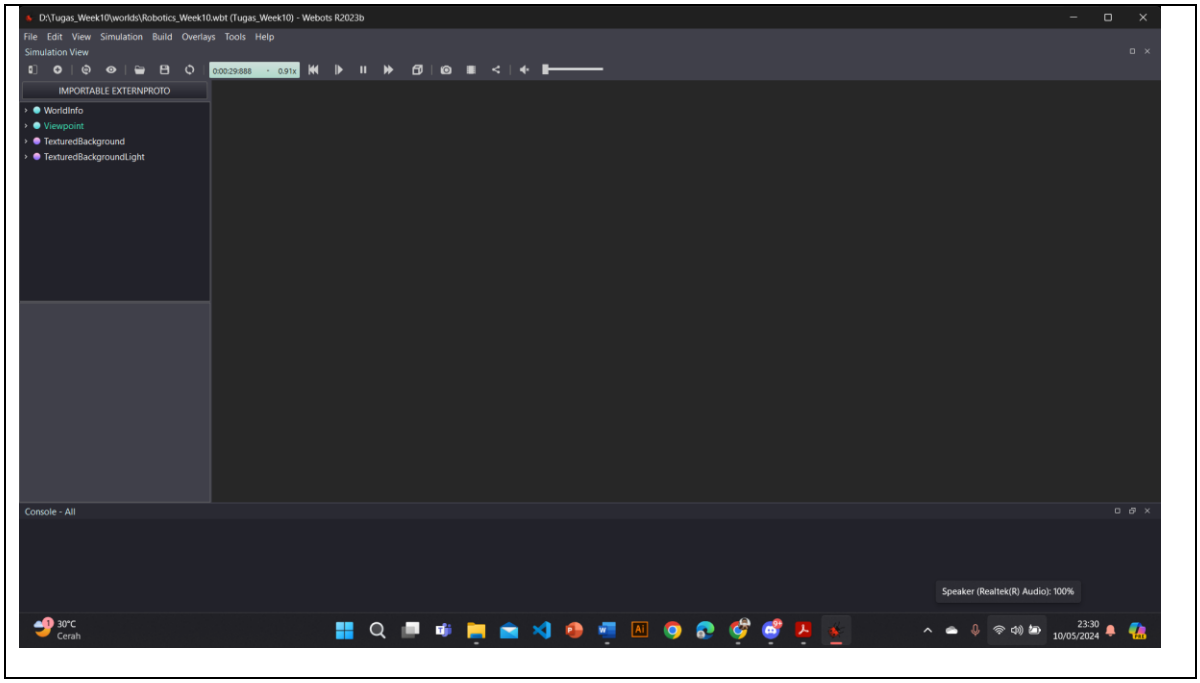
Nama	**Muhammad Abyan Ridhan Siregar**
Kelas	TK-45-01
NIM	1103210053

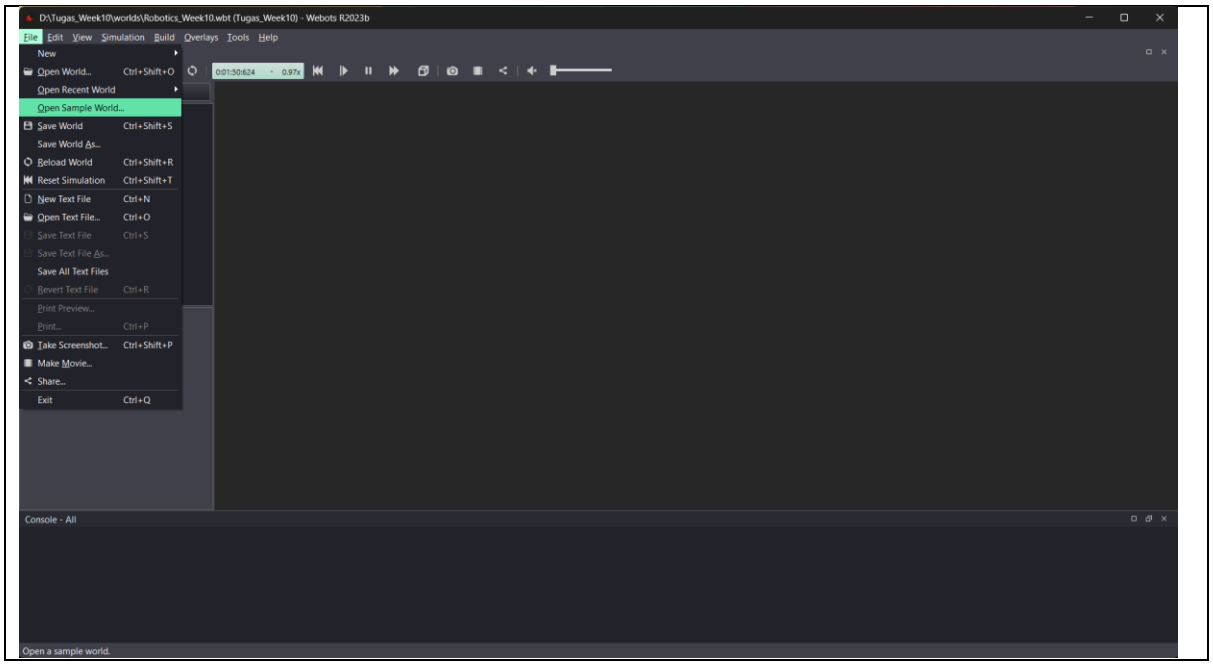
Tugas Robotika

“Simulasi Robot dengan Kamera di Webots”

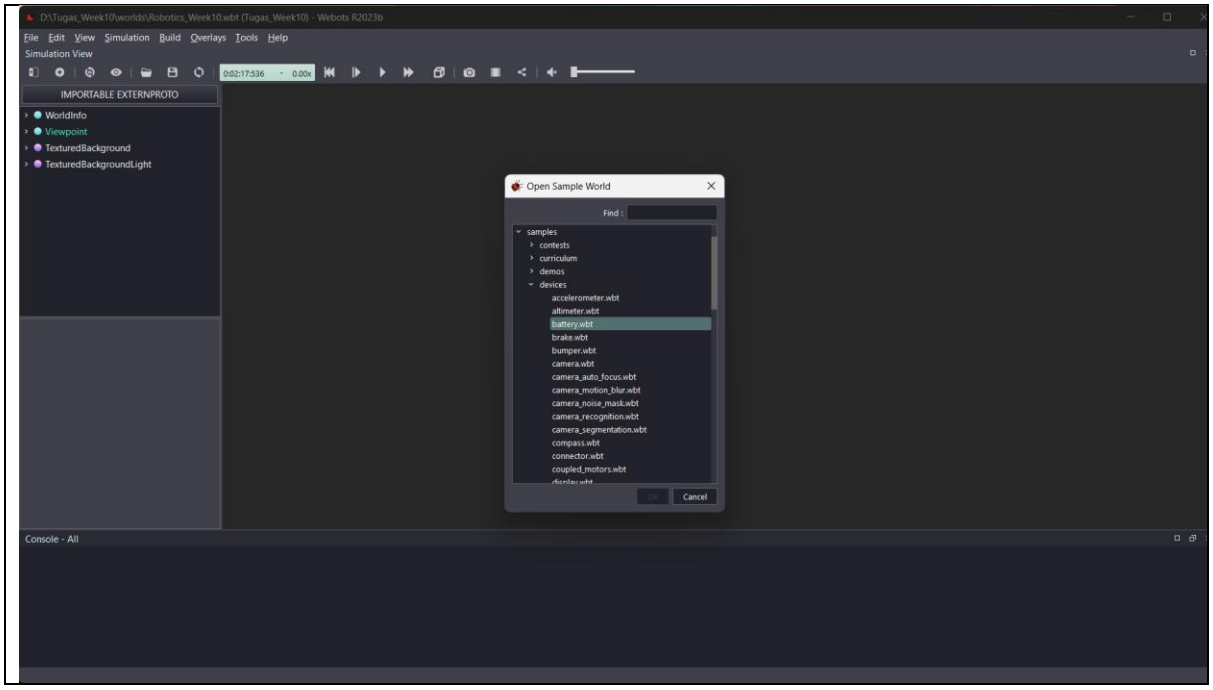
1. Buka Apps Webots



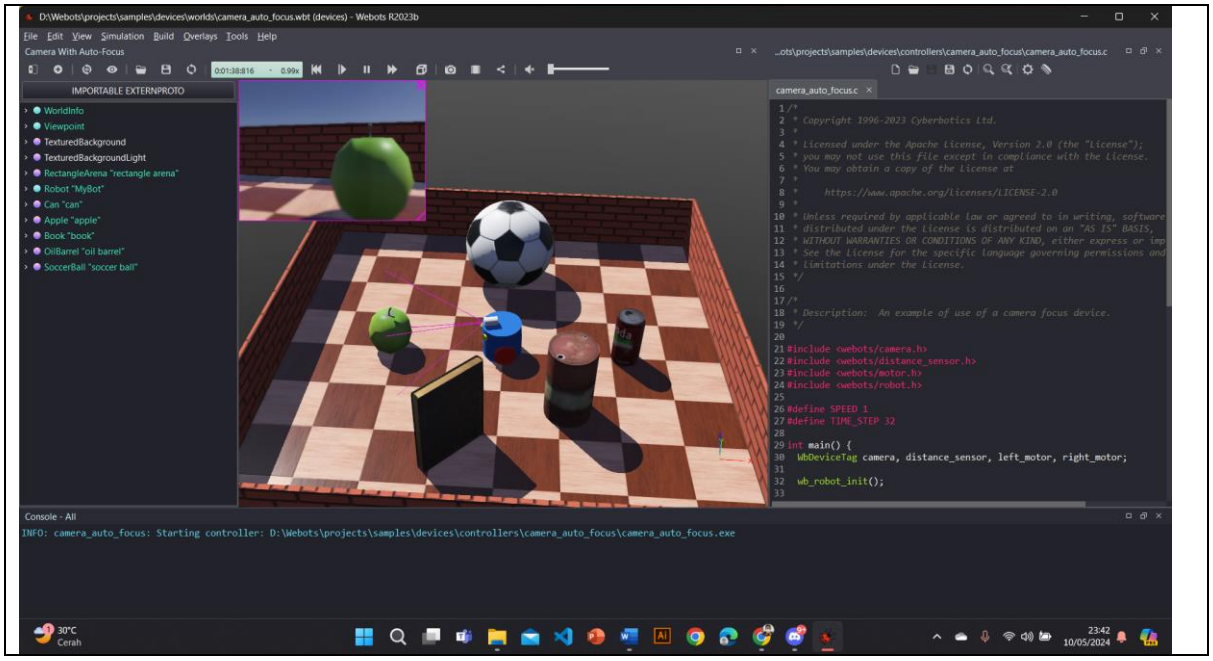
2. Klik tab file lalu pilih Open Sample World



3. Ketik Camera di kolom pencarian, lalu klik samples lalu devices



4. Untuk pertama saya memilih camera_auto_focus, yaitu Camera robot dengan fokus kamera berdasarkan objek yang ada di depannya

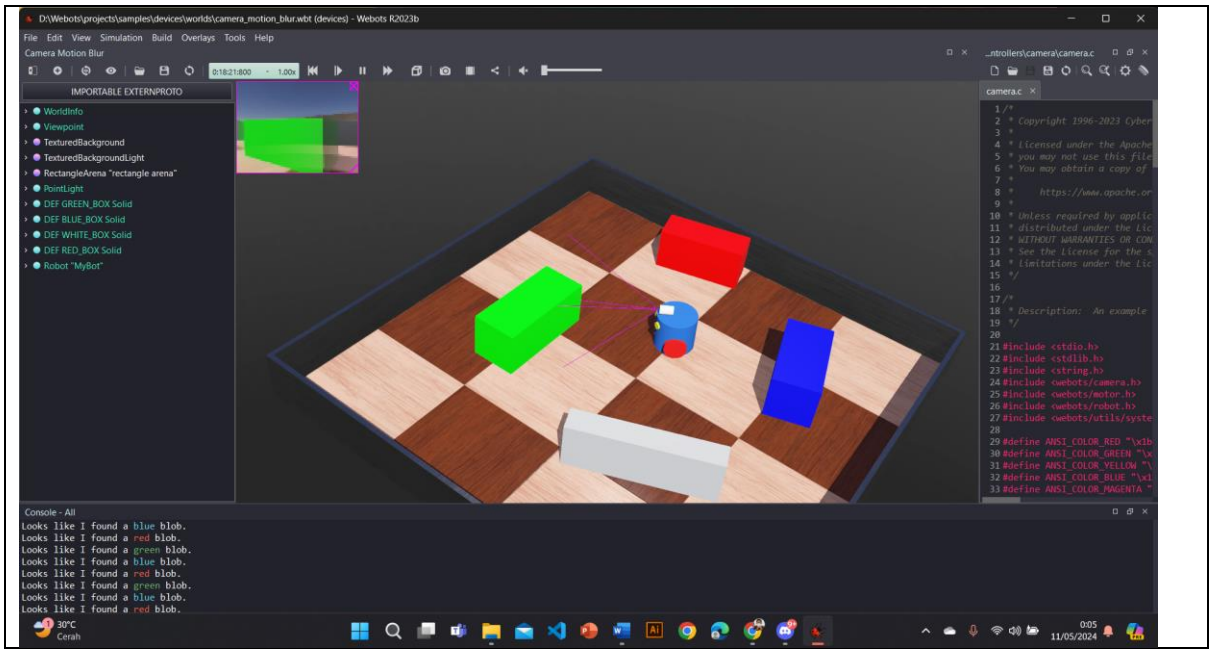


Penjelasan :

Dalam simulator Webots, camera_auto_focus adalah fitur yang memungkinkan kamera untuk secara otomatis menyesuaikan fokusnya untuk mendapatkan gambar yang tajam dari objek yang berbeda jaraknya dari kamera. Fitur auto focus sangat berguna dalam simulasi yang melibatkan kamera pada robot yang bergerak atau dalam lingkungan di mana jarak objek dapat berubah secara dinamis.

```
#include <webots/camera.h>
#include <webots/distance_sensor.h>
#include <webots/motor.h>
#include <webots/robot.h>

#define SPEED 1         // Kecepatan dasar robot
#define TIME_STEP 32     // Interval waktu pembaruan sensor dan aktuator

int main() {
  // Deklarasi variabel untuk perangkat kamera, sensor jarak, dan motor
  WbDeviceTag camera, distance_sensor, left_motor, right_motor;

  wb_robot_init();  // Inisialisasi sistem robot Webots

  /* Konfigurasi dan pengaktifan kamera */
  camera = wb_robot_get_device("camera");  // Mengambil handle perangkat kamera
  wb_camera_enable(camera, TIME_STEP);     // Mengaktifkan kamera dengan interval waktu
yang ditentukan

  /* Konfigurasi dan pengaktifan sensor jarak */
  distance_sensor = wb_robot_get_device("distance sensor");  // Mengambil handle perangkat
sensor jarak
  wb_distance_sensor_enable(distance_sensor, TIME_STEP);     // Mengaktifkan sensor jarak

  /* Pengaturan motor */
  left_motor = wb_robot_get_device("left wheel motor");      // Mengambil handle motor kiri
  right_motor = wb_robot_get_device("right wheel motor");    // Mengambil handle motor kanan
  wb_motor_set_position(left_motor, INFINITY);  // Mengatur posisi target motor menjadi tak
hingga (mode kecepatan)
  wb_motor_set_position(right_motor, INFINITY); // Mengatur posisi target motor menjadi tak
hingga (mode kecepatan)

  /* Set kecepatan awal motor */
  wb_motor_set_velocity(left_motor, -SPEED);  // Kecepatan motor kiri (mundur)
  wb_motor_set_velocity(right_motor, SPEED);  // Kecepatan motor kanan (maju)

  /* Loop utama */
  while (wb_robot_step(TIME_STEP) != -1) {
    // Mengambil nilai dari sensor jarak dan mengonversi dari mm ke meter
    const double object_distance = wb_distance_sensor_get_value(distance_sensor) / 1000;

    // Menyesuaikan fokus kamera berdasarkan jarak objek
    wb_camera_set_focal_distance(camera, object_distance);
  }

  wb_robot_cleanup();  // Membersihkan dan keluar dari simulasi

  return 0;
}
```

5. Untuk yang kedua, saya memilih camera_motion_blur. Kamera ini adalah kamera robot untuk mendeteksi blob warna (merah, hijau, dan biru) dan motion blur.



Penjelasan :

- Camera_motion_blur pada Webots adalah fitur yang mensimulasikan efek buram yang terjadi saat kamera bergerak cepat atau saat mengambil gambar objek yang bergerak cepat. Fitur ini membantu dalam pengembangan dan pengujian algoritma pengolahan gambar dan visi komputer yang lebih realistis dalam lingkungan simulasi.

- Camera_motion_blur dalam simulator Webots dapat digunakan juga untuk mendeteksi blob warna dengan mengidentifikasi daerah atau kelompok piksel yang memiliki warna yang serupa dan terpisah dari warna lainnya. Blob detection adalah metode pengolahan citra yang berguna untuk menangkap objek berdasarkan atribut warnanya. Dalam konteks mendeteksi blob warna merah, hijau, dan biru

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <webots/camera.h>
#include <webots/motor.h>
#include <webots/robot.h>
#include <webots/utils/system.h>

// Definisi warna ANSI untuk output terminal yang berwarna
#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"
```

```
#define SPEED 4  // Konstanta kecepatan dasar robot
enum BLOB_TYPE { RED, GREEN, BLUE, NONE };  // Enum untuk jenis blob yang dikenali

int main() {
 // Deklarasi variabel untuk perangkat, dimensi, dan status
 WbDeviceTag camera, left_motor, right_motor;
 int width, height;
 int pause_counter = 0;  // Counter untuk cooldown setelah deteksi blob
 int left_speed, right_speed;  // Kecepatan motor kiri dan kanan
 int i, j;  // Iterator untuk loop
 int red, blue, green;  // Penyimpanan jumlah warna dalam gambar
 const char *color_names[3] = {"red", "green", "blue"};  // Nama-nama warna
 const char *ansi_colors[3] = {ANSI_COLOR_RED, ANSI_COLOR_GREEN,
ANSI_COLOR_BLUE};  // Kode warna ANSI untuk output
 const char *filenames[3] = {"red_blob.png", "green_blob.png", "blue_blob.png"};  // Nama file
untuk menyimpan gambar
 enum BLOB_TYPE current_blob;  // Jenis blob yang terdeteksi

 wb_robot_init();  // Inisialisasi robot

 const int time_step = wb_robot_get_basic_time_step();  // Mendapatkan timestep dasar dari
simulasi

 /* Setup kamera dan motor */
 camera = wb_robot_get_device("camera");
 wb_camera_enable(camera, time_step);
 width = wb_camera_get_width(camera);
 height = wb_camera_get_height(camera);

 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, 0.0);
 wb_motor_set_velocity(right_motor, 0.0);

 /* Loop utama */
 while (wb_robot_step(time_step) != -1) {
  const unsigned char *image = wb_camera_get_image(camera);  // Mengambil gambar dari
kamera

  if (pause_counter > 0)  // Mengurangi pause_counter jika > 0
   pause_counter--;

  /* Logika kontrol berdasarkan status pause_counter */
  if (pause_counter > 640 / time_step) {
   left_speed = 0;  // Berhenti jika baru menemukan blob
   right_speed = 0;
  } else if (pause_counter > 0) {
   left_speed = -SPEED;  // Berputar jika dalam masa cooldown
```

```
    right_speed = SPEED;
   } else if (!image) {  // Jika tidak ada gambar, berhenti
    left_speed = 0;
    right_speed = 0;
   } else {  // Jika tidak dalam cooldown dan ada gambar
    red = 0;
    green = 0;
    blue = 0;

    /* Analisis gambar untuk deteksi blob */
    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);
     }
    }

    /* Logika deteksi blob berdasarkan dominasi warna */
    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;

    if (current_blob == NONE) {
     left_speed = -SPEED;  // Jika tidak ada blob, robot terus berputar
     right_speed = SPEED;
    } else {
     left_speed = 0;  // Jika blob ditemukan, robot berhenti
     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);
     // Menyimpan gambar dan setting filepath
     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);
            pause_counter = 1280 / time_step;  // Reset cooldown
          }
        }

   /* Set kecepatan motor */
   wb_motor_set_velocity(left_motor, left_speed);
   wb_motor_set_velocity(right_motor, right_speed);
  }

  wb_robot_cleanup();  // Membersihkan dan keluar dari simulasi

  return 0;
}
```