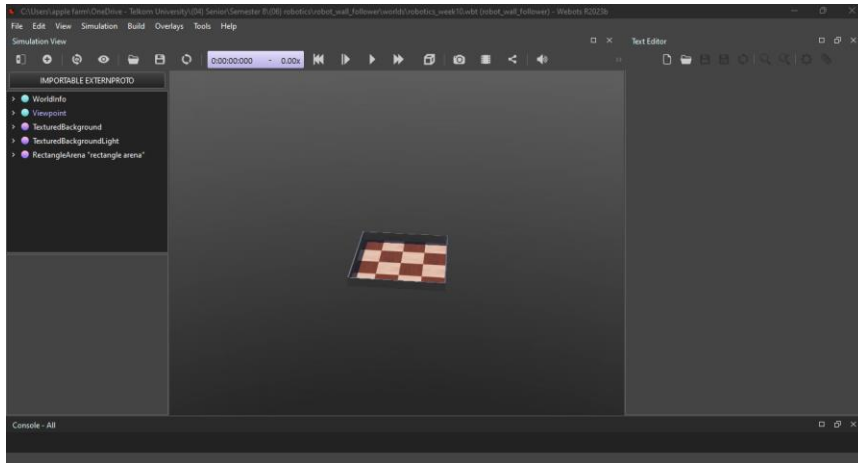


## Tugas Week 10 Robotika

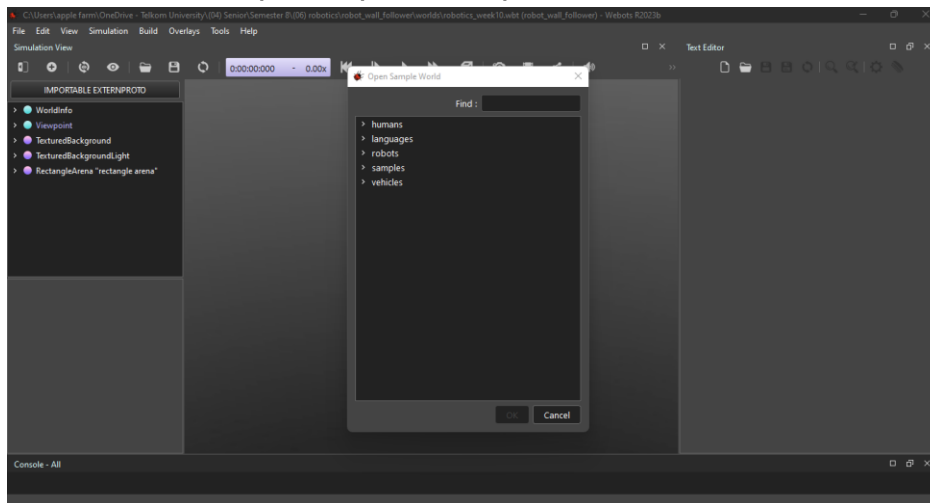
### Simulasi Robot dengan Kamera di *Webots*

#### Percobaan Pertama:

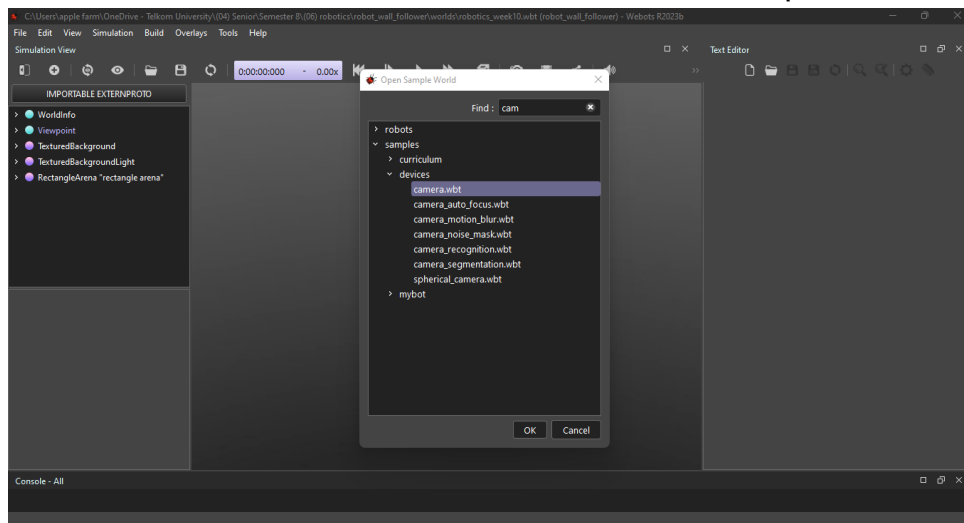
(1) Buka aplikasi Webots dan bikin file baru.



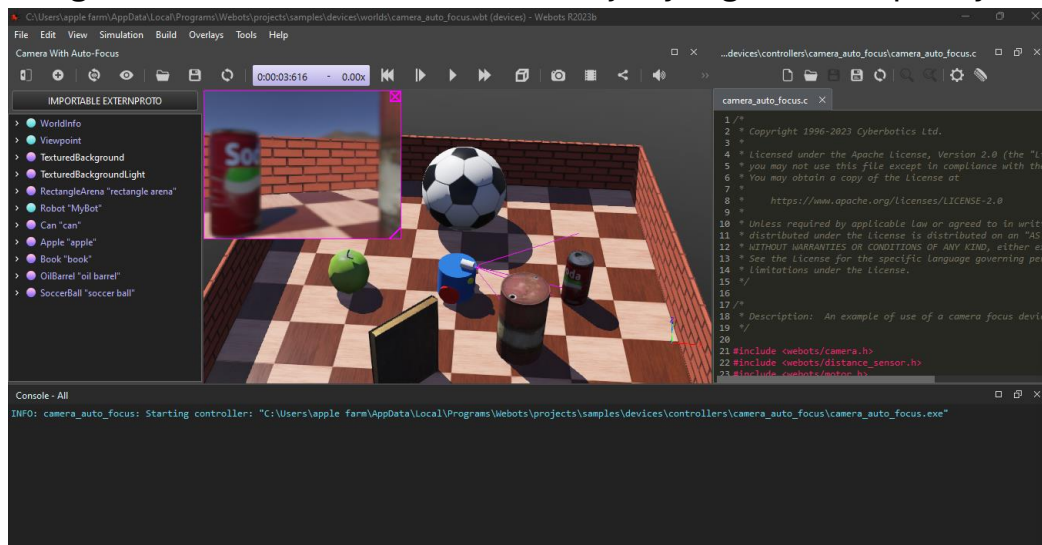
(2) Klik tab file lalu pilih *Open Sample World*,



(3) lalu ketik “Camera” di kolom Pencarian. 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, fitur **camera\_auto\_focus** memungkinkan kamera untuk secara otomatis menyesuaikan fokusnya guna mendapatkan gambar yang tajam dari objek dengan jarak yang bervariasi dari kamera. Fitur auto focus ini sangat bermanfaat dalam simulasi yang melibatkan kamera pada

robot yang bergerak atau dalam lingkungan di mana jarak objek dapat berubah secara dinamis. Dengan fitur ini, kamera dapat terus mempertahankan ketajaman gambar tanpa perlu penyesuaian manual, sehingga meningkatkan akurasi dan efisiensi pengamatan dalam simulasi robotik.

Source Code:

```
#include <webots/camera.h>
```

```
#include <webots/distance_sensor.h>
```

```
#include <webots/motor.h>
```

```
#include <webots/robot.h>
```

```
#define SPEED 1
```

```
#define TIME_STEP 32
```

```
int main() {
```

```
    WbDeviceTag camera, distance_sensor, left_motor, right_motor;
```

```
    wb_robot_init();
```

```
    /* Get the camera device, enable it */
```

```
    camera = wb_robot_get_device("camera");
```

```
    wb_camera_enable(camera, TIME_STEP);
```

```
/* Get the camera device, enable it */

distance_sensor = wb_robot_get_device("distance sensor");

wb_distance_sensor_enable(distance_sensor, TIME_STEP);


/* get a handler to the motors and set target position to infinity (speed
control). */

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);


/* Set the motors speed */

wb_motor_set_velocity(left_motor, -SPEED);

wb_motor_set_velocity(right_motor, SPEED);


/* Main loop */

while (wb_robot_step(TIME_STEP) != -1) {

    const double object_distance =
wb_distance_sensor_get_value(distance_sensor) / 1000;

    wb_camera_set_focal_distance(camera, object_distance);

}
```

```
wb_robot_cleanup();
```

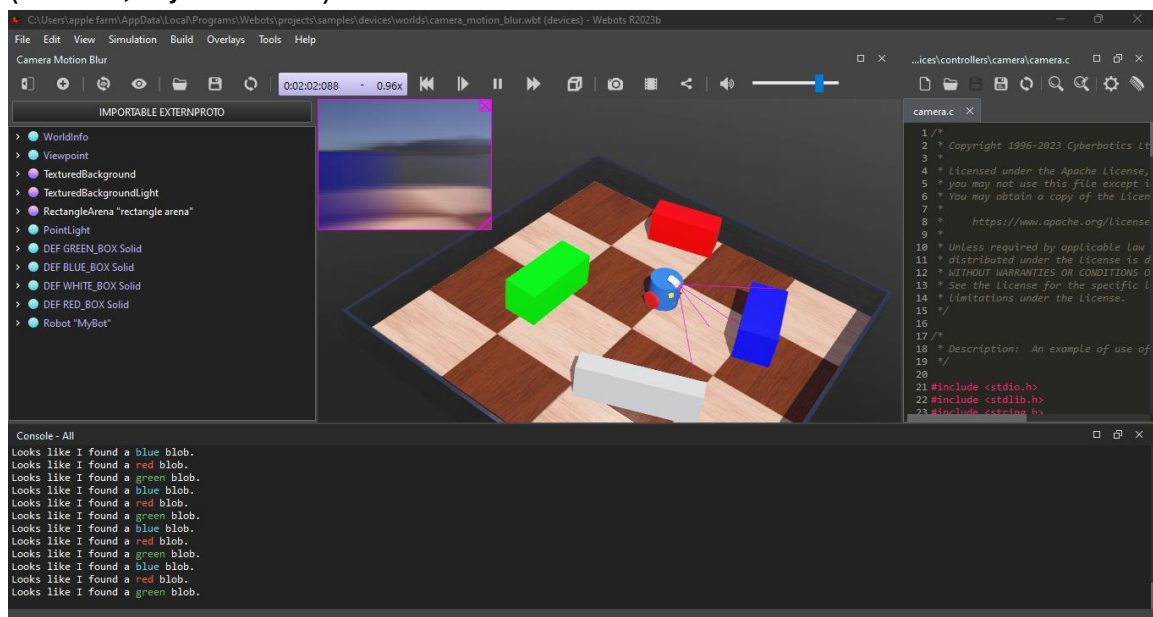
```
return 0;
```

```
}
```

## Percobaan Kedua:

(5) Untuk percobaan kedua saya, saya memilih “camera\_motion\_blur”.

Kamera ini memiliki fitur *motion blur* dan bisa *mendeteksi blob warna* (merah, hijau & biru).



## Penjelasan:

**-Camera\_motion\_blur** di Webots adalah fitur yang mensimulasikan efek blur yang terjadi saat kamera bergerak cepat atau saat menangkap gambar objek yang bergerak cepat. Fitur ini sangat berguna dalam

pengembangan dan pengujian algoritma pengolahan gambar dan visi komputer yang lebih realistis dalam lingkungan simulasi.

**-Camera\_motion\_blur** dalam simulator Webots juga dapat digunakan untuk mendeteksi blob warna, yaitu dengan mengidentifikasi area atau kelompok piksel yang memiliki warna serupa dan terpisah dari warna lainnya. Blob detection adalah metode pengolahan citra yang berguna untuk menangkap objek berdasarkan atribut warnanya, seperti mendeteksi blob warna merah, hijau, dan biru.

Source Code:

```
#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>


#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

enum BLOB_TYPE { RED, GREEN, BLUE, NONE };


int main() {

    WbDeviceTag camera, left_motor, right_motor;

    int width, height;

    int pause_counter = 0;

    int left_speed, right_speed;

    int i, j;

    int red, blue, green;

    const char *color_names[3] = {"red", "green", "blue"};

    const char *ansi_colors[3] = {ANSI_COLOR_RED,
ANSI_COLOR_GREEN, ANSI_COLOR_BLUE};

    const char *filenames[3] = {"red_blob.png", "green_blob.png",
"blue_blob.png"};

    enum BLOB_TYPE current_blob;


    wb_robot_init();
```

```
const int time_step = wb_robot_get_basic_time_step();

/* Get the camera device, enable it, and store its width and height */
camera = wb_robot_get_device("camera");
wb_camera_enable(camera, time_step);
width = wb_camera_get_width(camera);
height = wb_camera_get_height(camera);

/* get a handler to the motors and set target position to infinity (speed
control). */
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);

/* Main loop */
while (wb_robot_step(time_step) != -1) {
    /* Get the new camera values */
    const unsigned char *image = wb_camera_get_image(camera);
```



```
/* Decrement the pause_counter */  
  
if (pause_counter > 0)  
    pause_counter--;  
  
/*  
  
* Case 1  
  
* A blob was found recently  
  
* The robot waits in front of it until pause_counter  
* is decremented enough  
  
*/  
  
if (pause_counter > 640 / time_step) {  
    left_speed = 0;  
    right_speed = 0;  
}  
  
/*  
  
* Case 2  
  
* A blob was found quite recently  
  
* The robot begins to turn but don't analyse the image for a while,  
* otherwise the same blob would be found again  
  
*/
```

```
else if (pause_counter > 0) {  
    left_speed = -SPEED;  
    right_speed = SPEED;  
}  
/*  
* Case 3  
* The robot turns and analyse the camera image in order  
* to find a new blob  
*/  
  
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;  
  
    /*  
    * Here we analyse the image from the camera. The goal is to detect a
```

```
* blob (a spot of color) of a defined color in the middle of our
* screen.
* In order to achieve that we simply parse the image pixels of the
* center of the image, and sum the color components individually
*/
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);
    }
}

/*
* If a component is much more represented than the other ones,
* a blob is detected
*/
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;


/*

* Case 3a

* No blob is detected

* the robot continues to turn

*/

if (current_blob == NONE) {

    left_speed = -SPEED;

    right_speed = SPEED;

}

/*

* Case 3b

* A blob is detected

* the robot stops, stores the image, and changes its state

*/

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);

    pause_counter = 1280 / time_step;
```

```
    }  
}  
  
/* Set the motor speeds. */  
wb_motor_set_velocity(left_motor, left_speed);  
wb_motor_set_velocity(right_motor, right_speed);  
}  
  
wb_robot_cleanup();  
  
return 0;  
}
```