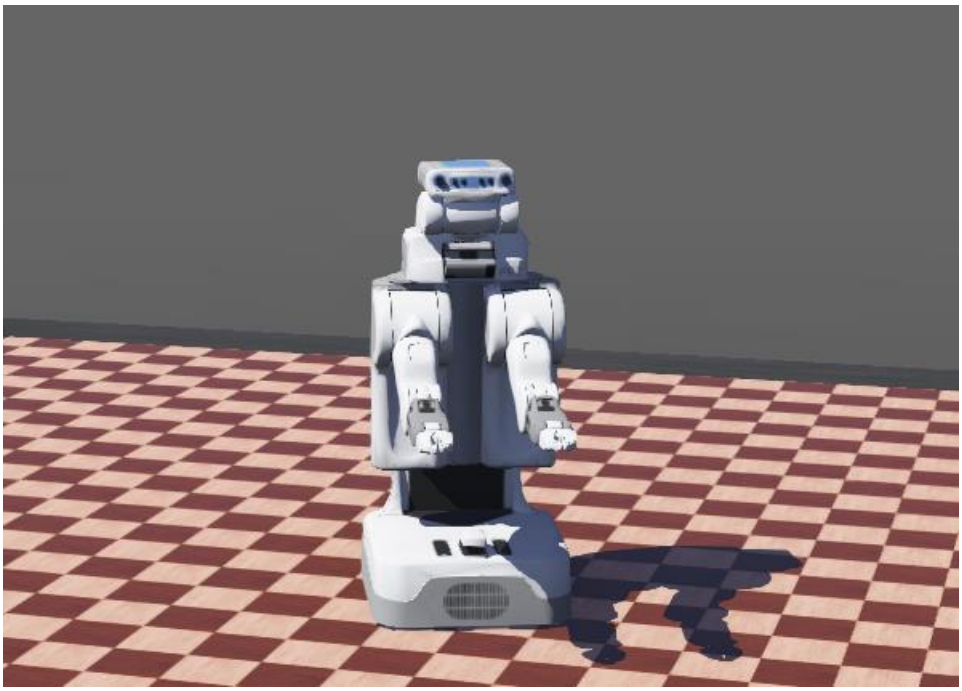
Nama :Juan Anemao Sokhi Zidomi

NIM :1103204007

Kelas :TK-44-01

Nama Robot:PR2



Robot ini adalah robot untuk mengangkat benda ketika di meja.Robot ini menggunakan sensor sebagai penggerak dalam menentukan tindakan .Robot Ini dirancang untuk melakukan kegiatan secara berurutan seperti maju ,mengambil benda,kemudian melakukan rotasi ,kemudian meletakkan benda.Biasanya robot PR2 digunakan dalam kegiatan industri

Source Code

Dibawah ini adalah untuk menginport library untuk robot pr2 dimana menginport sensor dan mendefenisikan kecepatan roda dan radius roda

```
#include <webots/camera.h>
#include <webots/device.h>
#include <webots/inertial_unit.h>
#include <webots/motor.h>
#include <webots/position_sensor.h>
#include <webots/robot.h>
#include <webots/touch_sensor.h>

#include <stdio.h>
#include <stdlib.h>

#define TIME_STEP 16

#define MAX_WHEEL_SPEED 3.0 // maximum velocity for the wheels [rad / s]
#define WHEELS_DISTANCE 0.4492 // distance between 2 caster wheels (the four wheels are located in square) [m]
#define SUB_WHEELS_DISTANCE 0.098 // distance between 2 sub wheels of a caster wheel [m]
#define WHEEL_RADIUS 0.08 // wheel radius
#define TOLERANCE 0.05 // arbitrary value
#define ALMOST_EQUAL(a, b) ((a < b + TOLERANCE) && (a > b - TOLERANCE))
```

Source Code ini menjelaskan tentang enumarasi roda ,sensor dan lengan robot pada robot pr2

```
enum { FLL_WHEEL, FLR_WHEEL, FRL_WHEEL, FRR_WHEEL, BLL_WHEEL, BLR_WHEEL, BRL_WHEEL, BRR_WHEEL };
enum { FL_ROTATION, FR_ROTATION, BL_ROTATION, BR_ROTATION };
enum { SHOULDER_ROLL, SHOULDER_LIFT, UPPER_ARM_ROLL, ELBOW_LIFT, WRIST_ROLL };
enum { LEFT_FINGER, RIGHT_FINGER, LEFT_TIP, RIGHT_TIP };

static WbDeviceTag wheel_motors[8];
static WbDeviceTag wheel_sensors[8];
static WbDeviceTag rotation_motors[4];
static WbDeviceTag rotation_sensors[4];
static WbDeviceTag left_arm_motors[5];
static WbDeviceTag left_arm_sensors[5];
static WbDeviceTag right_arm_motors[5];
static WbDeviceTag right_arm_sensors[5];
static WbDeviceTag right_finger_motors[4];
static WbDeviceTag right_finger_sensors[4];
static WbDeviceTag left_finger_motors[4];
static WbDeviceTag left_finger_sensors[4];
static WbDeviceTag head_tilt_motor;
static WbDeviceTag torso_motor;
static WbDeviceTag torso_sensor;
```

```
static WbDeviceTag left_finger_contact_sensors[2];
static WbDeviceTag right_finger_contact_sensors[2];
static WbDeviceTag imu_sensor;
static WbDeviceTag wide_stereo_l_stereo_camera_sensor;
static WbDeviceTag wide_stereo_r_stereo_camera_sensor;
static WbDeviceTag high_def_sensor;
static WbDeviceTag r_forearm_cam_sensor;
static WbDeviceTag l_forearm_cam_sensor;
static WbDeviceTag laser_tilt;
static WbDeviceTag base_laser;
```

Source code ini menjelaskan bahwa menggerakkan simulasi robot pr2 kelangkah selanjutnya

```
static void step() {
if (wb_robot_step(TIME_STEP) == -1) {
wb_robot_cleanup();
exit(EXIT_SUCCESS);
}
}
```

Source code menjelaskan pergerekan lengan robot dengan menggunakan joint yang mana robot dapat melakukan rotasi dengan mengandalkan sensor sebagai feedback untuk melakukan pergerakan dan mengatur kecepatan robot ketika melakukan pergerakan rotasi

```
static void initialize_devices() {
int i;
wheel_motors[FLL_WHEEL] = wb_robot_get_device("fl_caster_l_wheel_joint");
wheel_motors[FLR_WHEEL] = wb_robot_get_device("fl_caster_r_wheel_joint");
wheel_motors[FRL_WHEEL] = wb_robot_get_device("fr_caster_l_wheel_joint");
wheel_motors[FRR_WHEEL] = wb_robot_get_device("fr_caster_r_wheel_joint");
wheel_motors[BLL_WHEEL] = wb_robot_get_device("bl_caster_l_wheel_joint");
wheel_motors[BLR_WHEEL] = wb_robot_get_device("bl_caster_r_wheel_joint");
wheel_motors[BRL_WHEEL] = wb_robot_get_device("br_caster_l_wheel_joint");
wheel_motors[BRR_WHEEL] = wb_robot_get_device("br_caster_r_wheel_joint");
for (i = FLL_WHEEL; i <= BRR_WHEEL; ++i)
wheel_sensors[i] = wb_motor_get_position_sensor(wheel_motors[i]);

rotation_motors[FL_ROTATION] = wb_robot_get_device("fl_caster_rotation_joint");
rotation_motors[FR_ROTATION] = wb_robot_get_device("fr_caster_rotation_joint");
rotation_motors[BL_ROTATION] = wb_robot_get_device("bl_caster_rotation_joint");
rotation_motors[BR_ROTATION] = wb_robot_get_device("br_caster_rotation_joint");
for (i = FL_ROTATION; i <= BR_ROTATION; ++i)
rotation_sensors[i] = wb_motor_get_position_sensor(rotation_motors[i]);

for (i = FL_ROTATION; i <= BR_ROTATION; ++i)
rotation_sensors[i] = wb_motor_get_position_sensor(rotation_motors[i]);
left_arm_motors[SHOULDER_ROLL] = wb_robot_get_device("l_shoulder_pan_joint");
left_arm_motors[SHOULDER_LIFT] = wb_robot_get_device("l_shoulder_lift_joint");
left_arm_motors[UPPER_ARM_ROLL] = wb_robot_get_device("l_upper_arm_roll_joint");
left_arm_motors[ELBOW_LIFT] = wb_robot_get_device("l_elbow_flex_joint");
left_arm_motors[WRIST_ROLL] = wb_robot_get_device("l_wrist_roll_joint");
```

```
for (i = SHOULDER_ROLL; i <= WRIST_ROLL; ++i)
left_arm_sensors[i] = wb_motor_get_position_sensor(left_arm_motors[i]);

right_arm_motors[SHOULDER_ROLL] = wb_robot_get_device("r_shoulder_pan_joint");
right_arm_motors[SHOULDER_LIFT] = wb_robot_get_device("r_shoulder_lift_joint");
right_arm_motors[UPPER_ARM_ROLL] = wb_robot_get_device("r_upper_arm_roll_joint");
right_arm_motors[ELBOW_LIFT] = wb_robot_get_device("r_elbow_flex_joint");
right_arm_motors[WRIST_ROLL] = wb_robot_get_device("r_wrist_roll_joint");
for (i = SHOULDER_ROLL; i <= WRIST_ROLL; ++i)
right_arm_sensors[i] = wb_motor_get_position_sensor(right_arm_motors[i]);

left_finger_motors[LEFT_FINGER] = wb_robot_get_device("l_gripper_l_finger_joint");
left_finger_motors[RIGHT_FINGER] = wb_robot_get_device("l_gripper_r_finger_joint");
left_finger_motors[LEFT_TIP] = wb_robot_get_device("l_gripper_l_finger_tip_joint");
left_finger_motors[RIGHT_TIP] = wb_robot_get_device("l_gripper_r_finger_tip_joint");
for (i = LEFT_FINGER; i <= RIGHT_TIP; ++i)
left_finger_sensors[i] = wb_motor_get_position_sensor(left_finger_motors[i]);

right_finger_motors[LEFT_FINGER] = wb_robot_get_device("r_gripper_l_finger_joint");
right_finger_motors[RIGHT_FINGER] = wb_robot_get_device("r_gripper_r_finger_joint");
right_finger_motors[LEFT_TIP] = wb_robot_get_device("r_gripper_l_finger_tip_joint");
right_finger_motors[RIGHT_TIP] = wb_robot_get_device("r_gripper_r_finger_tip_joint");
for (i = LEFT_FINGER; i <= RIGHT_TIP; ++i)
right_finger_sensors[i] = wb_motor_get_position_sensor(right_finger_motors[i]);

head_tilt_motor = wb_robot_get_device("head_tilt_joint");
torso_motor = wb_robot_get_device("torso_lift_joint");
torso_sensor = wb_robot_get_device("torso_lift_joint_sensor");

left_finger_contact_sensors[LEFT_FINGER] =
wb_robot_get_device("l_gripper_l_finger_tip_contact_sensor");
left_finger_contact_sensors[RIGHT_FINGER] =
wb_robot_get_device("l_gripper_r_finger_tip_contact_sensor");
right_finger_contact_sensors[LEFT_FINGER] =
wb_robot_get_device("r_gripper_l_finger_tip_contact_sensor");
right_finger_contact_sensors[RIGHT_FINGER] =
wb_robot_get_device("r_gripper_r_finger_tip_contact_sensor");

imu_sensor = wb_robot_get_device("imu_sensor");

wide_stereo_l_stereo_camera_sensor =
wb_robot_get_device("wide_stereo_l_stereo_camera_sensor");
wide_stereo_r_stereo_camera_sensor =
wb_robot_get_device("wide_stereo_r_stereo_camera_sensor");
high_def_sensor = wb_robot_get_device("high_def_sensor");
r_forearm_cam_sensor = wb_robot_get_device("r_forearm_cam_sensor");
l_forearm_cam_sensor = wb_robot_get_device("l_forearm_cam_sensor");
laser_tilt = wb_robot_get_device("laser_tilt");
base_laser = wb_robot_get_device("base_laser");
}
```

Source code menjelaskan tentang mengaktifkan perangkat-perangkat yang menggunakan sensor guna menggerakkan robot supaya dapat bertindak sesuai feedback dari sensor

```
static void enable_devices() {
int i = 0;
for (i = 0; i < 8; ++i) {
wb_position_sensor_enable(wheel_sensors[i], TIME_STEP);
// init the motors for speed control
wb_motor_set_position(wheel_motors[i], INFINITY);
wb_motor_set_velocity(wheel_motors[i], 0.0);
}

for (i = 0; i < 4; ++i)
wb_position_sensor_enable(rotation_sensors[i], TIME_STEP);

for (i = 0; i < 2; ++i) {
wb_touch_sensor_enable(left_finger_contact_sensors[i], TIME_STEP);
wb_touch_sensor_enable(right_finger_contact_sensors[i], TIME_STEP);
}

for (i = 0; i < 4; ++i) {
wb_position_sensor_enable(left_finger_sensors[i], TIME_STEP);
wb_position_sensor_enable(right_finger_sensors[i], TIME_STEP);
}

for (i = 0; i < 5; ++i) {
wb_position_sensor_enable(left_arm_sensors[i], TIME_STEP);
wb_position_sensor_enable(right_arm_sensors[i], TIME_STEP);
}

wb_position_sensor_enable(torso_sensor, TIME_STEP);
}
```

Source code ini menjeaskan tentang roda yang menggerakkan robot ,yang mana source code ini mengatur kecepatan robot untuk bergerak secara bersama-sama

```
static void set_wheels_speeds(double fll, double flr, double frl, double frr, double bll, double blr, double
brl, double brr) {
wb_motor_set_velocity(wheel_motors[FLL_WHEEL], fll);
wb_motor_set_velocity(wheel_motors[FLR_WHEEL], flr);
wb_motor_set_velocity(wheel_motors[FRL_WHEEL], frl);
wb_motor_set_velocity(wheel_motors[FRR_WHEEL], frr);
wb_motor_set_velocity(wheel_motors[BLL_WHEEL], bll);
wb_motor_set_velocity(wheel_motors[BLR_WHEEL], blr);
wb_motor_set_velocity(wheel_motors[BRL_WHEEL], brl);
wb_motor_set_velocity(wheel_motors[BRR_WHEEL], brr);
}
```

Source code ini menjelaskan tentang pengaturan torsi pada pada robot ,rotasi menggunakan sensor

```
static void set_wheels_speed(double speed) {
set_wheels_speeds(speed, speed, speed, speed, speed, speed, speed, speed);
}

static void stop_wheels() {
set_wheels_speeds(0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0);
}

static void enable_passive_wheels(bool enable) {
static double torques[8] = {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0};
int i;
if (enable) {
for (i = 0; i < 8; ++i) {
torques[i] = wb_motor_get_available_torque(wheel_motors[i]);
wb_motor_set_available_torque(wheel_motors[i], 0.0);
}
} else {
for (i = 0; i < 8; ++i)
wb_motor_set_available_torque(wheel_motors[i], torques[i]);
}
}

static void set_rotation_wheels_angles(double fl, double fr, double bl, double br, bool
wait_on_feedback) {
if (wait_on_feedback) {
stop_wheels();
enable_passive_wheels(true);
}

wb_motor_set_position(rotation_motors[FL_ROTATION], fl);
wb_motor_set_position(rotation_motors[FR_ROTATION], fr);
wb_motor_set_position(rotation_motors[BL_ROTATION], bl);
wb_motor_set_position(rotation_motors[BR_ROTATION], br);
```

Source Code ini menjelaskan tentang langkah selanjutnya dalam menentukan posisi rotasi ,dan Ketika robot bergerak ada suatu state bahwa robot tersebut akan diam ketika melakukan kegiatan yang diinginkan

```
if (wait_on_feedback) {
const double target[4] = {fl, fr, bl, br};

while (true) {
bool all_reached = true;
int i;
for (i = 0; i < 4; ++i) {
double current_position = wb_position_sensor_get_value(rotation_sensors[i]);
if (!ALMOST_EQUAL(current_position, target[i])) {
all_reached = false;
break;
}
}

if (all_reached)
break;
else
step();
}

enable_passive_wheels(false);
}
}

tatic void robot_rotate(double angle) {
stop_wheels();

set_rotation_wheels_angles(3.0 * M_PI_4, M_PI_4, -3.0 * M_PI_4, -M_PI_4, true);
const double max_wheel_speed = angle > 0 ? MAX_WHEEL_SPEED : -MAX_WHEEL_SPEED;
set_wheels_speed(max_wheel_speed);

double initial_wheel0_position = wb_position_sensor_get_value(wheel_sensors[FLL_WHEEL]);
double expected_travel_distance = fabs(angle * 0.5 * (WHEELS_DISTANCE +
SUB_WHEELS_DISTANCE));

while (true) {
double wheel0_position = wb_position_sensor_get_value(wheel_sensors[FLL_WHEEL]);
double wheel0_travel_distance = fabs(WHEEL_RADIUS * (wheel0_position -
initial_wheel0_position));

if (wheel0_travel_distance > expected_travel_distance)
break;

if (expected_travel_distance - wheel0_travel_distance < 0.025)
set_wheels_speed(0.1 * max_wheel_speed);

step();
}

set_rotation_wheels_angles(0.0, 0.0, 0.0, 0.0, true);
stop_wheels();
}
```

Source code ini menjelaskan tentang cara menggerakkan robot ke depan dalam jarak tertentu ,ketika robot tersebut mengambil benda

```
static void robot_go_forward(double distance) {
double max_wheel_speed = distance > 0 ? MAX_WHEEL_SPEED : -MAX_WHEEL_SPEED;
set_wheels_speed(max_wheel_speed);

double initial_wheel0_position = wb_position_sensor_get_value(wheel_sensors[FLL_WHEEL]);

while (true) {
double wheel0_position = wb_position_sensor_get_value(wheel_sensors[FLL_WHEEL]);
// travel distance done by the wheel
double wheel0_travel_distance = fabs(WHEEL_RADIUS * (wheel0_position -
initial_wheel0_position));

if (wheel0_travel_distance > fabs(distance))
break;

if (fabs(distance) - wheel0_travel_distance < 0.025)
set_wheels_speed(0.1 * max_wheel_speed);
step();
}
stop_wheels();
}
static void set_gripper(bool left, bool open, double torqueWhenGripping, bool wait_on_feedback) {
WbDeviceTag motors[4];
motors[LEFT_FINGER] = left ? left_finger_motors[LEFT_FINGER] :
right_finger_motors[LEFT_FINGER];
motors[RIGHT_FINGER] = left ? left_finger_motors[RIGHT_FINGER] :
right_finger_motors[RIGHT_FINGER];
motors[LEFT_TIP] = left ? left_finger_motors[LEFT_TIP] : right_finger_motors[LEFT_TIP];
motors[RIGHT_TIP] = left ? left_finger_motors[RIGHT_TIP] : right_finger_motors[RIGHT_TIP];

WbDeviceTag sensors[4];
sensors[LEFT_FINGER] = left ? left_finger_sensors[LEFT_FINGER] :
right_finger_sensors[LEFT_FINGER];
sensors[RIGHT_FINGER] = left ? left_finger_sensors[RIGHT_FINGER] :
right_finger_sensors[RIGHT_FINGER];
sensors[LEFT_TIP] = left ? left_finger_sensors[LEFT_TIP] : right_finger_sensors[LEFT_TIP];
sensors[RIGHT_TIP] = left ? left_finger_sensors[RIGHT_TIP] :
right_finger_sensors[RIGHT_TIP];

WbDeviceTag contacts[2];
contacts[LEFT_FINGER] = left ? left_finger_contact_sensors[LEFT_FINGER] :
right_finger_contact_sensors[LEFT_FINGER];
contacts[RIGHT_FINGER] = left ? left_finger_contact_sensors[RIGHT_FINGER] :
right_finger_contact_sensors[RIGHT_FINGER];
```

Source code ini menjelaskan tentang posisi robot ketika bergerak

```
static bool firstCall = true;
static double maxTorque = 0.0;
if (firstCall) {
maxTorque = wb_motor_get_available_torque(motors[LEFT_FINGER]);
firstCall = false;
}

int i;
for (i = 0; i < 4; ++i)
wb_motor_set_available_torque(motors[i], maxTorque);

if (open) {
static const double targetOpenValue = 0.5;
for (i = 0; i < 4; ++i)
wb_motor_set_position(motors[i], targetOpenValue);

if (wait_on_feedback) {
while (!ALMOST_EQUAL(wb_position_sensor_get_value(sensors[LEFT_FINGER]),
targetOpenValue))
step();
}
} else {
static const double targetCloseValue = 0.0;
for (i = 0; i < 4; ++i)
wb_motor_set_position(motors[i], targetCloseValue);

if (wait_on_feedback) {
// wait until the 2 touch sensors are fired or the target value is reached
while (
(wb_touch_sensor_get_value(contacts[LEFT_FINGER]) == 0.0 ||
wb_touch_sensor_get_value(contacts[RIGHT_FINGER]) == 0.0) &&
!ALMOST_EQUAL(wb_position_sensor_get_value(sensors[LEFT_FINGER]), targetCloseValue)) {
step();
}
double current_position = wb_position_sensor_get_value(sensors[LEFT_FINGER]);
for (i = 0; i < 4; ++i) {
wb_motor_set_available_torque(motors[i], torqueWhenGripping);
wb_motor_set_position(motors[i], fmax(0.0, 0.95 * current_position));
}
}
}
}
```

Source code ini menjelaskan peregrakan lengan robot dengan menggunakan sensor sebagai feedback

```
static void set_right_arm_position(double shoulder_roll, double shoulder_lift, double
upper_arm_roll, double elbow_lift,
double wrist_roll, bool wait_on_feedback) {
wb_motor_set_position(right_arm_motors[SHOULDER_ROLL], shoulder_roll);
wb_motor_set_position(right_arm_motors[SHOULDER_LIFT], shoulder_lift);
wb_motor_set_position(right_arm_motors[UPPER_ARM_ROLL], upper_arm_roll);
wb_motor_set_position(right_arm_motors[ELBOW_LIFT], elbow_lift);
wb_motor_set_position(right_arm_motors[WRIST_ROLL], wrist_roll);
if (wait_on_feedback) {
while
(!ALMOST_EQUAL(wb_position_sensor_get_value(right_arm_sensors[SHOULDER_ROL
L]), shoulder_roll) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(right_arm_sensors[SHOULDER_LIFT])
, shoulder_lift) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(right_arm_sensors[UPPER_ARM_ROL
L]), upper_arm_roll) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(right_arm_sensors[ELBOW_LIFT]),
elbow_lift) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(right_arm_sensors[WRIST_ROLL]),
wrist_roll)) {
step();
}
}
}
static void set_left_arm_position(double shoulder_roll, double shoulder_lift, double
upper_arm_roll, double elbow_lift,
double wrist_roll, bool wait_on_feedback) {
wb_motor_set_position(left_arm_motors[SHOULDER_ROLL], shoulder_roll);
wb_motor_set_position(left_arm_motors[SHOULDER_LIFT], shoulder_lift);
wb_motor_set_position(left_arm_motors[UPPER_ARM_ROLL], upper_arm_roll);
wb_motor_set_position(left_arm_motors[ELBOW_LIFT], elbow_lift);
wb_motor_set_position(left_arm_motors[WRIST_ROLL], wrist_roll);

if (wait_on_feedback) {
while
(!ALMOST_EQUAL(wb_position_sensor_get_value(left_arm_sensors[SHOULDER_ROLL]
), shoulder_roll) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(left_arm_sensors[SHOULDER_LIFT]),
shoulder_lift) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(left_arm_sensors[UPPER_ARM_ROLL
]), upper_arm_roll) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(left_arm_sensors[ELBOW_LIFT]),
elbow_lift) ||
!ALMOST_EQUAL(wb_position_sensor_get_value(left_arm_sensors[WRIST_ROLL]),
wrist_roll)) {
step();
}
}
```

Source code ini menjelaskan posisi ketika menggerakkan lengan

```
static void set_initial_position() {
set_left_arm_position(0.0, 1.35, 0.0, -2.2, 0.0, false);
set_right_arm_position(0.0, 1.35, 0.0, -2.2, 0.0, false);

set_gripper(false, true, 0.0, false);
set_gripper(true, true, 0.0, false);

set_torso_height(0.2, true);
}

int main(int argc, char **argv) {
wb_robot_init();

initialize_devices();
enable_devices();
set_initial_position();

// go to the initial position
set_left_arm_position(0.0, 0.5, 0.0, -0.5, 0.0, true);
set_right_arm_position(0.0, 0.5, 0.0, -0.5, 0.0, true);
robot_go_forward(0.35);

while (true) {
set_gripper(true, false, 20.0, true);
set_gripper(false, false, 20.0, true);
// lift the arms
set_left_arm_position(0.0, 0.5, 0.0, -1.0, 0.0, true);
set_right_arm_position(0.0, 0.5, 0.0, -1.0, 0.0, true);
// go to the other table
robot_go_forward(-0.35);
robot_rotate(M_PI);
robot_go_forward(0.35);
// move the arms down
set_left_arm_position(0.0, 0.5, 0.0, -0.5, 0.0, true);
set_right_arm_position(0.0, 0.5, 0.0, -0.5, 0.0, true);
// open the grippers
set_gripper(true, true, 0.0, true);
set_gripper(false, true, 0.0, true);
}

wb_robot_cleanup();

return EXIT_SUCCESS;
}
```