Elective 2 – Robotics Technology   
SY 2024-2025, 2nd Semester

**LABORATORY ACTIVITY 1**   
Virtual Robotics Simulation

Submitted to**:**

**Engr. Rojay A. Flores   
Elective 2 Instructor**

Submitted by:

**Caples, Ma. Gellan A.   
BSECE – 4**

**KEY COMPONENTS OF THE ROBOT & HOW ARE COMPONENTS INTER-RELATED**

* **Motors and Actuators** – Receive commands from the controller to move the legs.
* **Body Frame** – The frame houses and supports all components.
* **Legs and Joints** – Carry out movement based on motor signals.
* **Sensors** and **Controller** – Help with perception and balance and runs the movement program.
* **Software (C Program in Webots**) – The brain, running a program to control the robot’s actions.

**IN YOUR OPINION, EXPLAIN WHERE COULD BE THIS KIND OF ROBOT CAN BE USED FOR?**

In my own perception, this robot can be a big help in **search and rescue particularly in terms of accidents or calamities** and even in **farming**. In rescue missions, it can go into disaster zones, find people who need help, and bring supplies. It can also reach dangerous places where it's too risky for humans to go. In farming, it can check crops, monitor soil, and spot pests or diseases early. It can also help with planting and spraying. Because it moves well on rough ground, it’s great for both saving lives and making farming easier.

**THE PROGRAM USED WITH COMMENTS ON THE INSTRUCTION THAT ARE EDITED OR ADDED.**

**/\***

**\* Copyright 1996-2024 Cyberbotics Ltd.**

**\***

**\* Licensed under the Apache License, Version 2.0 (the "License");**

**\* you may not use this file except in compliance with the License.**

**\* You may obtain a copy of the License at**

**\***

**\* https://www.apache.org/licenses/LICENSE-2.0**

**\***

**\* Unless required by applicable law or agreed to in writing, software**

**\* distributed under the License is distributed on an "AS IS" BASIS,**

**\* WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.**

**\* See the License for the specific language governing permissions and**

**\* limitations under the License.**

**\*/**

**/\***

**\* Description: Simple controller to present the Spot robot.**

**\*/**

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

**#include <webots/device.h>**

**#include <webots/led.h>**

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

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

**#include <math.h>**

**#include <stdio.h>**

**#include <stdlib.h>**

**#define NUMBER\_OF\_LEDS 8**

**#define NUMBER\_OF\_JOINTS 12**

**#define NUMBER\_OF\_CAMERAS 5**

**// Initialize the robot's information**

**static WbDeviceTag motors[NUMBER\_OF\_JOINTS];**

**static const char \*motor\_names[NUMBER\_OF\_JOINTS] = {**

**"front left shoulder abduction motor", "front left shoulder rotation motor", "front left elbow motor",**

**"front right shoulder abduction motor", "front right shoulder rotation motor", "front right elbow motor",**

**"rear left shoulder abduction motor", "rear left shoulder rotation motor", "rear left elbow motor",**

**"rear right shoulder abduction motor", "rear right shoulder rotation motor", "rear right elbow motor"};**

**static WbDeviceTag cameras[NUMBER\_OF\_CAMERAS];**

**static const char \*camera\_names[NUMBER\_OF\_CAMERAS] = {"left head camera", "right head camera", "left flank camera",**

**"right flank camera", "rear camera"};**

**static WbDeviceTag leds[NUMBER\_OF\_LEDS];**

**static const char \*led\_names[NUMBER\_OF\_LEDS] = {"left top led", "left middle up led", "left middle down led",**

**"left bottom led", "right top led", "right middle up led",**

**"right middle down led", "right bottom led"};**

**static void step() {**

**const double time\_step = wb\_robot\_get\_basic\_time\_step();**

**if (wb\_robot\_step(time\_step) == -1) {**

**wb\_robot\_cleanup();**

**exit(0);**

**}**

**}**

**// Movement decomposition**

**static void movement\_decomposition(const double \*target, double duration) {**

**const double time\_step = wb\_robot\_get\_basic\_time\_step();**

**const int n\_steps\_to\_achieve\_target = duration \* 1000 / time\_step;**

**double step\_difference[NUMBER\_OF\_JOINTS];**

**double current\_position[NUMBER\_OF\_JOINTS];**

**for (int i = 0; i < NUMBER\_OF\_JOINTS; ++i) {**

**current\_position[i] = wb\_motor\_get\_target\_position(motors[i]);**

**step\_difference[i] = (target[i] - current\_position[i]) / n\_steps\_to\_achieve\_target;**

**}**

**for (int i = 0; i < n\_steps\_to\_achieve\_target; ++i) {**

**for (int j = 0; j < NUMBER\_OF\_JOINTS; ++j) {**

**current\_position[j] += step\_difference[j];**

**wb\_motor\_set\_position(motors[j], current\_position[j]);**

**}**

**step();**

**}**

**}**

**static void lie\_down(double duration) {**

**const double motors\_target\_pos[NUMBER\_OF\_JOINTS] = {-0.40, -0.99, 1.59, // Front left leg**

**0.40, -0.99, 1.59, // Front right leg**

**-0.40, -0.99, 1.59, // Rear left leg**

**0.40, -0.99, 1.59}; // Rear right leg**

**movement\_decomposition(motors\_target\_pos, duration);**

**}**

**static void stand\_up(double duration) {**

**const double motors\_target\_pos[NUMBER\_OF\_JOINTS] = {-0.1, 0.0, 0.0, // Front left leg**

**0.1, 0.0, 0.0, // Front right leg**

**-0.1, 0.0, 0.0, // Rear left leg**

**0.1, 0.0, 0.0}; // Rear right leg**

**movement\_decomposition(motors\_target\_pos, duration);**

**}**

**static void sit\_down(double duration) {**

**const double motors\_target\_pos[NUMBER\_OF\_JOINTS] = {-0.20, -0.40, -0.19, // Front left leg**

**0.20, -0.40, -0.19, // Front right leg**

**-0.40, -0.90, 1.18, // Rear left leg**

**0.40, -0.90, 1.18}; // Rear right leg**

**movement\_decomposition(motors\_target\_pos, duration);**

**}**

**static void give\_paw() {**

**// Stabilize posture**

**const double motors\_target\_pos\_1[NUMBER\_OF\_JOINTS] = {-0.20, -0.30, 0.05, // Front left leg**

**0.20, -0.40, -0.19, // Front right leg**

**-0.40, -0.90, 1.18, // Rear left leg**

**0.49, -0.90, 0.80}; // Rear right leg**

**movement\_decomposition(motors\_target\_pos\_1, 4);**

**const double initial\_time = wb\_robot\_get\_time();**

**while (wb\_robot\_get\_time() - initial\_time < 8) {**

**wb\_motor\_set\_position(motors[4], 0.2 \* sin(2 \* wb\_robot\_get\_time()) + 0.6); // Upperarm movement**

**wb\_motor\_set\_position(motors[5], 0.4 \* sin(2 \* wb\_robot\_get\_time())); // Forearm movement**

**step();**

**}**

**// Get back in sitting posture**

**const double motors\_target\_pos\_2[NUMBER\_OF\_JOINTS] = {-0.20, -0.40, -0.19, // Front left leg**

**0.20, -0.40, -0.19, // Front right leg**

**-0.40, -0.90, 1.18, // Rear left leg**

**0.40, -0.90, 1.18}; // Rear right leg**

**movement\_decomposition(motors\_target\_pos\_2, 4);**

**}**

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

**wb\_robot\_init();**

**const double time\_step = wb\_robot\_get\_basic\_time\_step();**

**// Get cameras**

**for (int i = 0; i < NUMBER\_OF\_CAMERAS; ++i)**

**cameras[i] = wb\_robot\_get\_device(camera\_names[i]);**

**// enable the two front cameras**

**wb\_camera\_enable(cameras[0], 2 \* time\_step);**

**wb\_camera\_enable(cameras[1], 2 \* time\_step);**

**// Get the LEDs and turn them on**

**for (int i = 0; i < NUMBER\_OF\_LEDS; ++i) {**

**leds[i] = wb\_robot\_get\_device(led\_names[i]);**

**wb\_led\_set(leds[i], 1);**

**}**

**// Get the motors (joints) and set initial target position to 0**

**for (int i = 0; i < NUMBER\_OF\_JOINTS; ++i)**

**motors[i] = wb\_robot\_get\_device(motor\_names[i]);**

**while (true) {**

**lie\_down(5.0);**

**stand\_up(5.0);**

**sit\_down(0.5);**

**give\_paw(0.5);**

**stand\_up(1.0);**

**sit\_down(1.0);**

**give\_paw(0.5);**

**stand\_up(2.0);**

**lie\_down(2.0);**

**stand\_up(2.0);**

**lie\_down(2.0);**

**stand\_up(2.0);**

**lie\_down(0.50);**

**stand\_up(0.50);**

**lie\_down(0.75);**

**stand\_up(0.75);**

**lie\_down(0.75);**

**stand\_up(0.75);**

**sit\_down(0.5);**

**}**

**wb\_robot\_cleanup();**

**return EXIT\_FAILURE;**

**}**

* **I just changed the default executions**