Nama :Juan Anemao S Zidomi

NIM :1103204007

Kelas :TK-44-01

Hack Webots : Salamender Robot

Robot ini bernama Salamender robot yang dirancang untuk melakukan eksplorasi pada Hutan atau alam liar yang mana untuk mengamati perilaku hewan atau objek yang sedang diamati .Adapun fitur yang di Salamender Robot yaitu kamera dan dapat menghindari rintangan ketika sedang bergerak .Robot salamander tersebut dapat dikendalikan secara otomatis maupun manual .Salamender Robot dapat berjalan dan berenang .

Adapun Source code yang digunakan pada Salamender Robot

```
#include <assert.h>
#include <math.h>
#include <stdio.h>
#include <webots/camera.h>
#include <webots/distance_sensor.h>
#include <webots/gps.h>
#include <webots/keyboard.h>
#include <webots/motor.h>
#include <webots/robot.h>
```

Penjelasan Source code :Disamping adalah source code untuk library pada salamander robot yang mana di library tersebut ada library untuk kamera

```
// Mendefenisikan 3 dimensi
#define X 0
#define Y 1
#define Z 2

/* 6 actuated body segments and 4 legs */
#define NUM_MOTORS 10

#define WATER_LEVEL 0.5

/* virtual time between two calls to the run() function */
#define CONTROL_STEP 32


static double spine_offset = 0.0;
static double ampl = 1.0;
static double phase = 0.0; /* current locomotion phase */

// Mendefenisikan control type
enum { AUTO, KEYBOARD, STOP };
static int control = AUTO;

/* locomotion types */
enum {
WALK,
SWIM,
};

static int locomotion = WALK;

/* Motor posisi
static double min_motor_position[NUM_MOTORS];
static double max_motor_position[NUM_MOTORS];

double clamp(double value, double min, double max) {
if (min > max) {
assert(0);
return value;
} else if (min == 0 && max == 0)
return value;

return value < min ? min : value > max ? max : value;
}
```

```
void read_keyboard_command() {
        static int prev_key = 0;
        int new_key = wb_keyboard_get_key();
        if (new_key != prev_key) {
        switch (new_key) {
        case WB_KEYBOARD_LEFT:
        control = KEYBOARD;
        if (spine_offset > -0.4)
        spine_offset -= 0.1;
        printf("Spine offset: %f\n", spine_offset);
        break;
        case WB_KEYBOARD_RIGHT:
        control = KEYBOARD;
        if (spine_offset < 0.4)
        spine_offset += 0.1;
        printf("Spine offset: %f\n", spine_offset);
        break;
        case WB_KEYBOARD_UP:
        if (ampl < 1.5)
        ampl += 0.2;
        printf("Motion amplitude: %f\n", ampl);
        break;
        case WB_KEYBOARD_DOWN:
        if (ampl > -1.5)
        ampl -= 0.2;
        printf("Motion amplitude: %f\n", ampl);
        break;
        case 'A':
        control = AUTO;
        printf("Auto control ...\n");
        break;
        case ' ':
        control = STOP;
        printf("Stopped.\n");
        break;
        }
        prev_key = new_key;
        }
}
```

Source code tersebut di gunakan untuk mendefenisikan bahwa salamander robot tersebut dapat dikontrol melalui secara manual dan otomatis .Yang mana dapat dikendalikan secara manual dengan menekan arrow kiri dan arrow kanan .

```
int main() {
const double FREQUENCY = 1.4; /* locomotion frequency [Hz] */
const double WALK_AMPL = 0.6; /* radians */
const double SWIM_AMPL = 1.0; /* radians */

/* body and leg motors */
WbDeviceTag motor[NUM_MOTORS];
double target_position[NUM_MOTORS] = {0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0};

/* distance sensors and gps devices */
WbDeviceTag ds_left, ds_right, gps;

/* Initialize Webots lib */
wb_robot_init();

wb_camera_enable(wb_robot_get_device("camera"), CONTROL_STEP);

/* for loops */
int i;

/* get the motors device tags (some motors are not used) */
const char *MOTOR_NAMES[NUM_MOTORS] = {"motor_1", "motor_2", "motor_3", "motor_4",
"motor_5",
"motor_6", "motor_leg_1", "motor_leg_2", "motor_leg_3", "motor_leg_4"};
for (i = 0; i < NUM_MOTORS; i++) {
motor[i] = wb_robot_get_device(MOTOR_NAMES[i]);
min_motor_position[i] = wb_motor_get_min_position(motor[i]);
max_motor_position[i] = wb_motor_get_max_position(motor[i]);
}

/* get and enable left and right distance sensors */
ds_left = wb_robot_get_device("ds_left");
wb_distance_sensor_enable(ds_left, CONTROL_STEP);
ds_right = wb_robot_get_device("ds_right");
wb_distance_sensor_enable(ds_right, CONTROL_STEP);

/* get and enable gps device */
gps = wb_robot_get_device("gps");
wb_gps_enable(gps, CONTROL_STEP);

/* enable keyboard */
wb_keyboard_enable(CONTROL_STEP);

/* print user instructions */
printf("----- Salamandra Robotica -----\n");
printf("You can steer this robot!\n");
printf("Select the 3D window and press:\n");
printf(" 'Left/Right' --> TURN\n");
printf(" 'Up/Down' --> INCREASE/DECREASE motion amplitude\n");
printf(" 'Spacebar' --> STOP the robot motors\n");
printf(" 'A' --> return to AUTO steering mode\n");
```

```
/* control loop: sense-compute-act */
while (wb_robot_step(CONTROL_STEP) != -1) {
read_keyboard_command();

if (control == AUTO) {
/* perform sensor measurment */
double left_val = wb_distance_sensor_get_value(ds_left);
double right_val = wb_distance_sensor_get_value(ds_right);

/* change direction according to sensor reading */
spine_offset = (right_val - left_val);
}

if (control == AUTO || control == KEYBOARD) {
/* increase phase according to elapsed time */
phase -= (double)CONTROL_STEP / 1000.0 * FREQUENCY * 2.0 * M_PI;

/* get current elevation from gps */
double elevation = wb_gps_get_values(gps)[Z];

if (locomotion == SWIM && elevation > WATER_LEVEL - 0.003) {
locomotion = WALK;
phase = target_position[6];
} else if (locomotion == WALK && elevation < WATER_LEVEL - 0.015) {
locomotion = SWIM;

/* put legs backwards for swimming */
double backwards_position = phase - fmod(phase, 2.0 * M_PI) - M_PI / 2;
for (i = 6; i < NUM_MOTORS; i++)
target_position[i] = backwards_position;
}

/* switch locomotion control according to current robot elevation and water level */
if (locomotion == WALK) {
/* above water level: walk (s-shape of robot body) */
const double A[6] = {-0.7, 1, 1, 0, -1, -1};
for (i = 0; i < 6; i++)
target_position[i] = WALK_AMPL * ampl * A[i] * sin(phase) + spine_offset;

/* rotate legs */
target_position[6] = phase;
target_position[7] = phase + M_PI;
target_position[8] = phase + M_PI;
target_position[9] = phase;
} else { /* SWIM */
/* below water level: swim (travelling wave of robot body) */
for (i = 0; i < 6; i++)
        target_position[i] = SWIM_AMPL * ampl * sin(phase + i * (2 * M_PI / 6)) * ((i + 5) / 10.0) +
        spine_offset;
        }
}
```

```
/* motors actuation */
for (i = 0; i < NUM_MOTORS; i++) {
        target_position[i] = clamp(target_position[i], min_motor_position[i], max_motor_position[i]);
        wb_motor_set_position(motor[i], target_position[i]);
        }
}

wb_robot_cleanup();
```

Program tersebut adalah main program yang menjalankan salamander robot ,yang mana dijalankan secara otomatis dan manual .Jika manual dapat digerakkan dengan cara menekan tombol arrow kiri dan kanan .Jika ingin otomatis dapat menombol tekan ‘A’ .