

UJIAN TENGAH SEMESTER
MATAKULIAH: ROBOTIKA DAN SISTEM CERDAS

Nama: Jean Jeasen Timotius

Nim: 1103201257

Kelas: TK43G08

Hacking Webot : Moose demo



Gambar diatas ini merupakan Robot moose yang diprogram untuk membawa sebuah kotak mengeksplorasi lingkungan sekitar ,dan robot ini di program untuk menghindari obstacle , robot nantinya akan berputar haluan dan kembali melakukan eksplorasi, Robot Moose ini juga dapat dkendalikan secara manual dan otomatis.

Berikut adalah source code robot moose yang digunakan

```
#include <math.h>
#include <stdio.h>
#include <webots/compass.h>
#include <webots/gps.h>
#include <webots/keyboard.h>
#include <webots/motor.h>
#include <webots/robot.h>

#define TIME_STEP 16
#define TARGET_POINTS_SIZE 13
#define DISTANCE_TOLERANCE 1.5
#define MAX_SPEED 7.0
#define TURN_COEFFICIENT 4.0

enum XYZAComponents { X = 0, Y, Z, ALPHA };
enum Sides { LEFT, RIGHT };

typedef struct _Vector {
    double u;
    double v;
} Vector;
static WbDeviceTag motors[8];
static WbDeviceTag gps;
static WbDeviceTag compass
```

Pada bagian ini sedang mengambil library yang akan dipakai untuk memprogram robot moose, kemudian mendefine beberapa konstanta yang dibutuhkan robot moose seperti distance, max speed , dan lain-lain. Mendefenisikan enumerasi components untuk menyimpan indeks data dari GPS dan sides untuk menyimpan perintah instruksi side(Left,Right). Mendefinisikan variable yang digunakan robot mulai dari motors sebagai penggerak , gps untuk tracking position robot dan compass untuk menunjukkan arah pergerakan robot moose.

```
static Vector targets[TARGET_POINTS_SIZE] = {
    {-4.209318, 9.147717}, {0.946812, 9.404304}, {0.175989, -1.784311}, {-
2.805353, -8.829694}, {-3.846730, -15.602851},
```

```

    {-4.394915, -24.550777}, {-1.701877, -33.617226}, {-4.394915, -24.550777},
    {-3.846730, -15.602851}, {-2.805353, -8.829694},
    {0.175989, -1.784311}, {0.946812, 9.404304}, {-7.930821, 6.421292}

};
static int current_target_index = 0;
static bool autopilot = true;
static bool old_autopilot = true;
static int old_key = -1;

static double modulus_double(double a, double m) {
    const int div = (int)(a / m);
    double r = a - div * m;
    if (r < 0.0)
        r += m;
    return r;
}

static void robot_set_speed(double left, double right) {
    int i;
    for (i = 0; i < 4; i++) {
        wb_motor_set_velocity(motors[i + 0], left);
        wb_motor_set_velocity(motors[i + 4], right);
    }
}

```

Kode program ini mengatur pengendalian kecepatan motor dan mode autopilot atau manual pada robot simulasi dalam lingkungan Webots. Fungsi-fungsi tersebut akan digunakan dalam fungsi utama program untuk mengontrol pergerakan robot berdasarkan data dari sensor dan target yang ingin dicapai.

```

static void check_keyboard() {
    double speeds[2] = {0.0, 0.0};

    int key = wb_keyboard_get_key();
    if (key >= 0) {
        switch (key) {
            case WB_KEYBOARD_UP:
                speeds[LEFT] = MAX_SPEED;
                speeds[RIGHT] = MAX_SPEED;
                autopilot = false;
                break;

```

```

case WB_KEYBOARD_DOWN:
    speeds[LEFT] = -MAX_SPEED;
    speeds[RIGHT] = -MAX_SPEED;
    autopilot = false;
    break;
case WB_KEYBOARD_RIGHT:
    speeds[LEFT] = MAX_SPEED;
    speeds[RIGHT] = -MAX_SPEED;
    autopilot = false;
    break;
case WB_KEYBOARD_LEFT:
    speeds[LEFT] = -MAX_SPEED;
    speeds[RIGHT] = MAX_SPEED;
    autopilot = false;
    break;
case 'P':
    if (key != old_key) { // perform this action just once
        const double *position_3d = wb_gps_get_values(gps);
        printf("position: {%f, %f}\n", position_3d[X], position_3d[Y]);
    }
    break;
case 'A':
    if (key != old_key) // perform this action just once
        autopilot = !autopilot;
    break;
}
}

```

Kode Program diatas mendefenisikan input control yang diterima dari keyboard , Adapun input nya ialah dari arrow keyboard yaitu: UP,RIGHT,LEFT, dan DOWN. Setiap kali arrow keyboard di tekan maka robot moose akan bergerak. Setiap kali arrow button ditekan maka robot akan bergerak mengikuti konstatanta speed dan max speed yangtelah didefenisikan.

```

if (autopilot != old_autopilot) {
    old_autopilot = autopilot;
    if (autopilot)
        printf("auto control\n");
    else
        printf("manual control\n");
}

```

```

robot_set_speed(speeds[LEFT], speeds[RIGHT]);
old_key = key;
}
static double norm(const Vector *v) {
    return sqrt(v->u * v->u + v->v * v->v);
}
static void normalize(Vector *v) {
    double n = norm(v);
    v->u /= n;
    v->v /= n;
}
static void minus(Vector *v, const Vector *const v1, const Vector *const v2) {
    v->u = v1->u - v2->u;
    v->v = v1->v - v2->v;
}

```

Kode program diatas ini menentukan kapan robot dalam mode autopilot dan mode manual , kita bisa emngatur mode nya dengan menekan A pada keyboard.

```

static void run_autopilot() {
    // prepare the speed array
    double speeds[2] = {0.0, 0.0};

    // read gps position and compass values
    const double *position_3d = wb_gps_get_values(gps);
    const double *north_3d = wb_compass_get_values(compass);

    // compute the 2D position of the robot and its orientation
    const Vector position = {position_3d[X], position_3d[Y]};

    // compute the direction and the distance to the target
    Vector direction;
    minus(&direction, &(targets[current_target_index]), &position);
    const double distance = norm(&direction);
    normalize(&direction);

    // compute the error angle
    const double robot_angle = atan2(north_3d[0], north_3d[1]);
    const double target_angle = atan2(direction.v, direction.u);
    double beta = modulus_double(target_angle - robot_angle, 2.0 * M_PI) - M_PI;
}

```

```

// move singularity
if (beta > 0)
    beta = M_PI - beta;
else
    beta = -beta - M_PI;

// a target position has been reached
if (distance < DISTANCE_TOLERANCE) {
    char index_char[3] = "th";
    if (current_target_index == 0)
        sprintf(index_char, "st");
    else if (current_target_index == 1)
        sprintf(index_char, "nd");
    else if (current_target_index == 2)
        sprintf(index_char, "rd");
    printf("%d%s target reached\n", current_target_index + 1, index_char);
    current_target_index++;
    current_target_index %= TARGET_POINTS_SIZE;
}
// move the robot to the next target
else {
    speeds[LEFT] = MAX_SPEED - M_PI + TURN_COEFFICIENT * beta;
    speeds[RIGHT] = MAX_SPEED - M_PI - TURN_COEFFICIENT * beta;
}

// set the motor speeds
robot_set_speed(speeds[LEFT], speeds[RIGHT]);
}

```

Kode program diatas ini mekukan perhitungan-perhitungan mulai dari membaca nilai GPS,menghitung posisi robot secara 2D , menghitung arah dan jarak robot terhadap target,Menghitung sudut beta antara robot dengan target , mengecek apakah robot sudah mendekati target, dan mengatur kecepatan motor kiri dan kanan dari robot.

```

int main(int argc, char *argv[]) {
    // initialize webots communication
    wb_robot_init();
}

```

```

// print user instructions
printf("You can drive this robot:\n");
printf("Select the 3D window and use cursor keys:\n");
printf("Press 'A' to return to the autopilot mode\n");
printf("Press 'P' to get the robot position\n");
printf("\n");

wb_robot_step(1000);

const char *names[8] = {"left motor 1", "left motor 2", "left motor 3", "left
motor 4",
                        "right motor 1", "right motor 2", "right motor 3", "right motor 4"};

// get motor tags
int i;
for (i = 0; i < 8; i++) {
    motors[i] = wb_robot_get_device(names[i]);
    wb_motor_set_position(motors[i], INFINITY);
}

// get gps tag and enable
gps = wb_robot_get_device("gps");
wb_gps_enable(gps, TIME_STEP);

// get compass tag and enable
compass = wb_robot_get_device("compass");
wb_compass_enable(compass, TIME_STEP);

// enable keyboard
wb_keyboard_enable(TIME_STEP);

// start forward motion
robot_set_speed(MAX_SPEED, MAX_SPEED);

// main loop
while (wb_robot_step(TIME_STEP) != -1) {
    check_keyboard();
    if (autopilot)
        run_autopilot();
}

```

```
}  
  
wb_robot_cleanup();  
  
return 0;  
}
```

Pada tahap ini program menjelaskan Menampilkan instruksi kepada pengguna melalui fungsi printf(), Mendefinisikan nama-nama motor yang akan digunakan dalam simulasi, Mengambil tag (handle) dari GPS dan mengaktifkannya menggunakan fungsi wb_gps_enable(), Mengaktifkan keyboard menggunakan fungsi wb_keyboard_enable(), Melakukan loop utama dengan menggunakan while (wb_robot_step(TIME_STEP) != -1). Setelah loop selesai, dilakukan pembersihan menggunakan fungsi wb_robot_cleanup() sebelum program diakhiri.