Nama : Mochamad Phillia Wibowo

NIM : 1103204191

Kelas : Robotika dan Sistem Cerdas (TK-44-G7)

Lecture 8: Playlist Path Planning 8-19

Lab 3 Task 1 : Webots motion estimation

Beberapa aspek penting dalam coding ini:

1. **Inisialisasi Robot dan Sensor:**
    - Robot diinisialisasi menggunakan modul **Robot()** dari controller.
    - Sensor-sensor seperti sensor jarak, sensor posisi, kamera, dan unit inersia diaktifkan dan diinisialisasi.
2. **Pengaturan Kontrol Motor:**
    - Motor kiri dan kanan diatur menggunakan handler motor.
    - Nilai-nilai seperti radius roda, kecepatan maksimum, dan jarak antar roda diatur.
3. **Fungsi Pembantu:**
    - Fungsi-fungsi pembantu seperti konversi satuan, perhitungan waktu, dan fungsi-fungsi untuk pergerakan dan rotasi robot diimplementasikan.
4. **Pemantauan Posisi dan Arah Robot:**
    - Posisi dan arah robot diupdate berdasarkan data dari sensor posisi, sensor inersia, dan perhitungan dari pergerakan roda.
5. **Navigasi ke Sel-sel pada Grid:**
    - Fungsi **move_to_cell** digunakan untuk membimbing robot ke sel-sel pada grid dengan mengatur pergerakan dan rotasi.
6. **Utama (Main):**
    - Fungsi utama **main** mengatur pergerakan robot untuk mengunjungi seluruh sel pada grid tanpa mengunjungi sel yang sudah dikunjungi sebelumnya.
7. **Kondisi Berhenti Robot:**
    - Sebuah fungsi **check_if_robot_should_stop** digunakan untuk memeriksa apakah robot sudah mengunjungi semua sel atau belum.

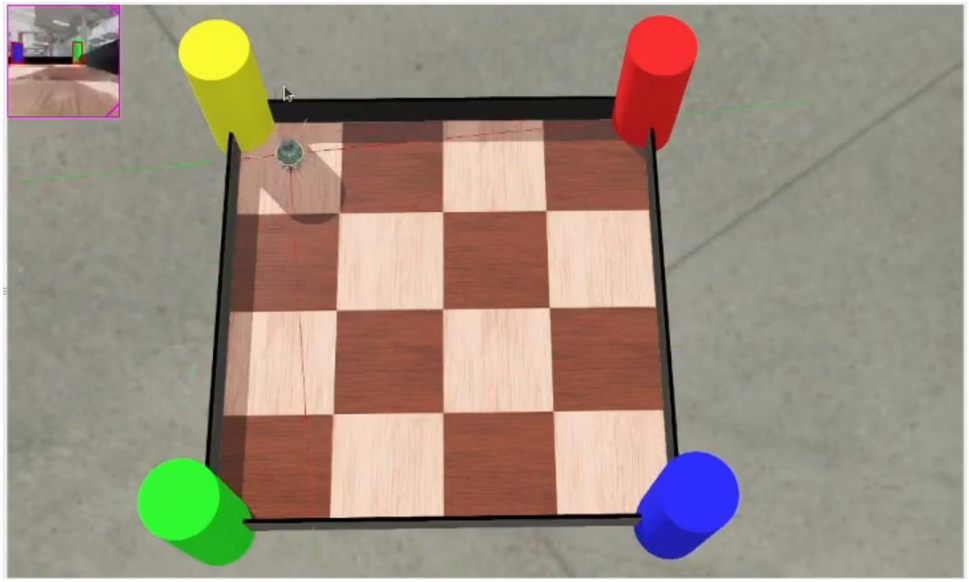
Lab 3 Task 2: Webots measurement estimation

Pada tugas ini, mengendalikan robot simulasi menggunakan Webots. Robot ini dilengkapi dengan sensor jarak, sensor posisi, kamera, dan unit inersia. Robot bergerak di atas grid sel berukuran 4x4 dan harus mengunjungi setiap sel satu per satu.

Beberapa fungsi utama dalam program ini termasuk pemindahan robot ke sel tertentu, rotasi robot, pengukuran jarak menggunakan sensor, dan pembaruan status robot seperti pose dan sel yang sudah dikunjungi. Robot menggunakan algoritma trilaterasi dengan menggunakan informasi dari tiga silinder berwarna untuk memperkirakan posisinya.

Program juga memanfaatkan hardcoded nilai-nilai seperti posisi awal robot, posisi silinder, dan warna-warna silinder. Robot bergerak dengan kecepatan maksimum tertentu dan berhenti ketika mencapai tujuan atau ketika semua sel telah dikunjungi.

Program ini mencakup fungsi-fungsi seperti move_to_cell, turn_left, turn_right, dan trilateration untuk membantu robot mencapai setiap sel dengan mengandalkan informasi dari sensor dan kamera. Selain itu, terdapat kontrol untuk memastikan bahwa robot berhenti setelah mengunjungi semua sel yang ditentukan.



Lab 3 Task 3: Webots particle filters

Inisialisasi Robot dan Sensor:

```
robot = Robot()
fds = robot.getDevice('front_ds')
lds = robot.getDevice('left_ds')
rds = robot.getDevice('right_ds')
```

Robot dan sensor-sensornya diinisialisasi, termasuk sensor jarak di depan, kiri, dan kanan.

Inisialisasi Variabel dan Parameter Robot:

```
wheel_radius = 1.6 / 2.0
wheel_circ = 2 * 3.14 * wheel_radius
enc_unit = wheel_circ / 6.28
```

Parameter-parameter seperti radius roda, lingkar roda, dan unit encoder diatur.

Konfigurasi Labirin:

```
grid_maze = [[0, 1, 0, 1, 1], ...]
```

Labirin diatur dalam bentuk matriks dengan informasi tentang setiap sel termasuk dinding di arah utara, timur, selatan, dan barat.

Kelas Particle:

```
class Particle:
    def __init__(self, n, dir, importance):
        self.n = n
        self.dir = dir
        self.importance = importance
```

Kelas ini merepresentasikan partikel dalam *particle filter* dengan informasi seperti sel n, arah, dan tingkat kepentingan.

Fungsi-fungsi Gerak dan Rotasi Robot:

```
def move(inches, timestep, dest=None):
def rotate(degrees, seconds, timestep, direction):
def turn_left(ts, degrees=-90.5):
def turn_right(ts, degrees=90.5):
```

Fungsi-fungsi untuk menggerakkan dan memutar robot dengan memanfaatkan sensor posisi dan kontrol motor.

Fungsi-fungsi *Particle Filter* (Measurement, Motion Update, Resampling):

```
def measurement_estimation(state_walls):
def get_cell_weights():
def normalize_weights(weights):
def resample_particles(norm_weights):
def motion_update(control):
```

Fungsi-fungsi ini mengimplementasikan langkah-langkah *particle filter* seperti *measurement model*, *motion model*, dan *resampling*.

Loop Utama dan Eksekusi:

```
while robot.step(timestep) != -1:
    traverse_maze(timestep)
```

Program utama berupa loop yang memanggil fungsi *traverse_maze*, yang pada dasarnya menggabungkan langkah-langkah *particle filter* dengan pergerakan robot di dalam labirin.

```
Console - All
Pose: [15.032656000000008, -5.024848000000009, 12, -175.82870077930966]
Robot current grid cell: 12, Direction: North
From particles, robot thinks it is in cell: 8
Rotating right...
Motors stopped.
 _____________________________________
|  1        1         1         1     |
|1 V 0    0 V 0     0 V 0     0 V 1   |
|  1        1         0         0     |
|                                     |
|  1        1         0         0     |
|1 V 0    0 V 1     1 V 1     1 V 1   |
|  0        0         0         0     |
|                                     |
|  0        0         0         0     |
|1 V 1    1 V 0     0 V 1     1 V 1   |
|  0        1         1         0     |
|                                     |
|  0        1         1         0     |
|1 V 0    0 V 0     0 V 0     0 V 1   |
|  1        1         1         1     |
|_____________________________________|
particle counts:
[0, 0, 0, 0]
[0, 0, 0, 5]
[0, 0, 1, 74]
[0, 0, 0, 0]
INFO: 'particleFilterWalls' controller exited successfully.
```

Lab 4 Task 1: Mapping a webots World

1. **Inisialisasi Robot dan Sensor:**
    - Membuat instance robot dan mengaktifkan sensor-sensor seperti sensor jarak, sensor posisi, kamera, dan sensor inersia.
2. **Konfigurasi Variabel dan Parameter Robot:**
    - Menentukan parameter dan variabel seperti radius roda, kecepatan maksimum, dan ukuran labirin.
3. **Konfigurasi Awal Labirin:**
    - Menetapkan konfigurasi awal labirin dengan mendefinisikan dinding-dinding luar dan sel-sel yang belum dikunjungi.
4. **Fungsi dan Metode Kontrol Robot:**
    - Berbagai fungsi dan metode diimplementasikan untuk mengendalikan pergerakan robot, rotasi, dan pemetaan labirin.
5. **Pemetaan Labirin:**
    - Program melakukan pemetaan labirin dengan menggerakkan robot, mendeteksi dinding menggunakan sensor jarak, dan memperbarui peta labirin berdasarkan informasi yang diperoleh.
6. **Pemantauan dan Pembaruan Posisi Robot:**
    - Robot memantau posisi dan orientasi saat bergerak, dan pembaruan posisi dan orientasi dilakukan secara berkala.
7. **Penanganan Rotasi dan Pergerakan:**
    - Terdapat fungsi untuk mengatur rotasi dan pergerakan robot berdasarkan sudut dan jarak tertentu.
8. **Penghentian Program:**
    - Program memeriksa kondisi untuk menentukan kapan harus berhenti, seperti ketika semua sel dalam labirin telah dikunjungi.
9. **Fungsi Utilitas:**
    - Beberapa fungsi bantu diimplementasikan, seperti konversi satuan, pemantauan sensor, dan pembaruan informasi orientasi robot.
10. **Loop Utama dan Eksekusi:**

- Program berjalan dalam loop utama hingga kondisi berhenti tercapai.

11. **Pencetakan dan Output:**
    - Menampilkan informasi penting seperti posisi robot, arah yang dihadapi, sel-sel yang telah dikunjungi, dan informasi jarak sensor.

Lab 4 Task 2: Webots SLAM

1. **Implementasi Algoritma SLAM:**
    - Dalam SLAM, kita biasanya menggunakan algoritma untuk melakukan estimasi posisi dan membangun peta. Contohnya, algoritma ekstend Kalman Filter (EKF) atau metode pemrograman dinamis Stochastic Gradient Descent (DP-SLAM). Implementasi ini akan memperhitungkan ketidakpastian dalam posisi dan pemetaan.
2. **Penggunaan Data Odometri dan Sensor:**
    - SLAM bergantung pada data odometri (pergerakan perkiraan robot) dan pengukuran sensor untuk memperkirakan posisi dan membangun peta. Oleh karena itu, informasi dari encoder roda, sensor jarak, dan sensor inersia perlu digunakan dalam algoritma SLAM.
3. **Fungsi Estimasi Posisi:**
    - Menambahkan fungsi atau modul yang bertanggung jawab untuk menghitung estimasi posisi robot berdasarkan data odometri dan sensor. Ini dapat melibatkan pemrosesan data menggunakan algoritma SLAM yang dipilih.
4. **Fungsi Pemetaan:**
    - Menambahkan fungsi atau modul yang bertanggung jawab untuk membangun dan memperbarui peta labirin berdasarkan pengukuran sensor. Informasi dari sensor jarak dan sensor kamera dapat digunakan untuk menentukan lokasi dinding dan fitur lingkungan lainnya.
5. **Integrasi Algoritma SLAM dengan Kontrol Robot:**
    - Mengintegrasikan algoritma SLAM dengan kontrol pergerakan robot. Hal ini dapat mencakup memperbarui estimasi posisi pada setiap langkah pergerakan, serta memperbarui peta saat robot mendeteksi atau memetakan lingkungan.
6. **Pengelolaan Ketidakpastian:**
    - Memastikan bahwa SLAM controller memperhitungkan dan mengelola ketidakpastian, baik dalam estimasi posisi maupun pemetaan. Ini sering melibatkan manajemen kovarian dari informasi yang diperoleh dari sensor dan odometri.
7. **Fungsi Kritis SLAM:**

- Memastikan bahwa fungsi-fungsi kritis SLAM, seperti pembaruan posisi, pembaruan peta, dan manajemen ketidakpastian, diimplementasikan dengan benar.

8. **Penanganan Kasus Khusus dan Gangguan:**
    - Menangani kasus khusus dan gangguan, seperti ketika robot kehilangan penanda posisi atau menghadapi situasi yang tidak terduga. Ini melibatkan logika kontrol yang cermat untuk mengatasi kondisi ini.

Lab 4 Task 3: Path planning in a world with a known map

1. **Inisialisasi Robot dan Sensor:**
    - **Robot()**: Membuat instance robot.
    - Berbagai sensor seperti **front_ds**, **left_ds**, **right_ds** (distance sensors), **left wheel sensor**, **right wheel sensor**, **camera1** (camera), dan **inertial unit** diaktifkan dan diinisialisasi.
2. **Pengaturan Motor:**
    - Motor kiri dan kanan diatur menggunakan **left wheel motor** dan **right wheel motor**.
    - **setVelocity(0)** berfungsi untuk menghentikan motor.
    - **setPosition(float('inf'))** digunakan agar motor berputar secara terus-menerus.
3. **Parameter Robot dan Lingkungan:**
    - Beberapa parameter seperti **wheel_radius**, **wheel_circ**, **enc_unit**, **max_speed**, dan **d_mid** diatur untuk digunakan dalam perhitungan kinematika robot dan kontrol gerak.
4. **Maze dan Peta:**
    - **grid_maze**: Konfigurasi labirin dengan informasi tentang dinding di sekitar setiap sel, termasuk status kunjungan.
    - **visited_cells**: Menyimpan status kunjungan setiap sel pada grid.
5. **Fungsi Gerak dan Rotasi Robot:**
    - **move(inches, timestep)**: Bergerak maju sejauh sejumlah inches.
    - **rotate(degrees, seconds, timestep, direction)**: Rotasi robot sebesar degrees dalam waktu seconds ke arah tertentu.
6. **Fungsi Pemetaan dan Estimasi Posisi:**
    - **update_robot(rotating=False)**: Pembaruan posisi, orientasi, dan status grid pada setiap langkah simulasi.
    - **get_robot_x_y(vals)**: Perhitungan posisi robot berdasarkan sensor posisi roda.

- **get_current_grid_cell(x, y)**: Mendapatkan nomor sel grid saat ini berdasarkan koordinat x, y.

7. **Algoritma Path Planning (Wavefront):**
    - **wave_front(ts, goal_cell)**: Algoritma untuk merencanakan path menggunakan metode Wavefront.
    - **plan_path(ts, goal_cell)**: Membuat rencana path berdasarkan nilai wave count setiap sel.
    - **execute_plan(ts, plan)**: Mengeksekusi rencana path yang telah dibuat.
8. **Fungsi Utama (main()) dan Eksekusi:**
    - Pada fungsi utama, labirin diinisialisasi dengan metode Wavefront dan kemudian direncanakan path-nya.
    - Eksekusi path kemudian dilakukan, dan program berhenti ketika robot mencapai tujuan.