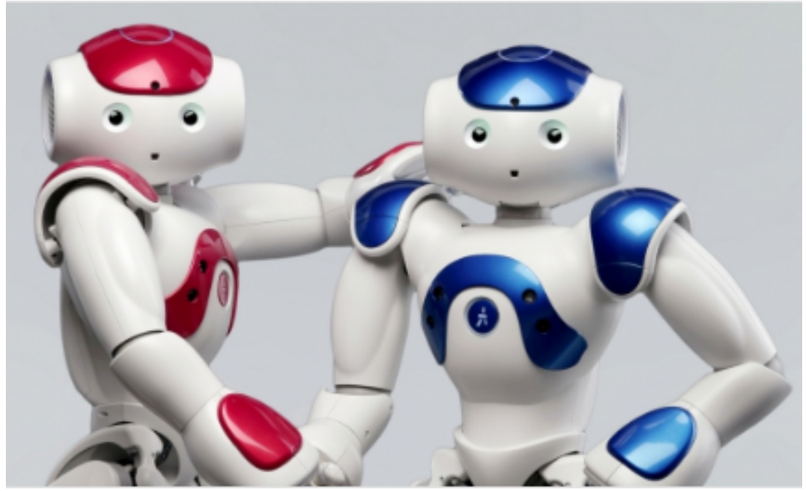
MOTION PLANNING AND TARGET SEARCHING IN COMPLEX ENVIRONMENTS

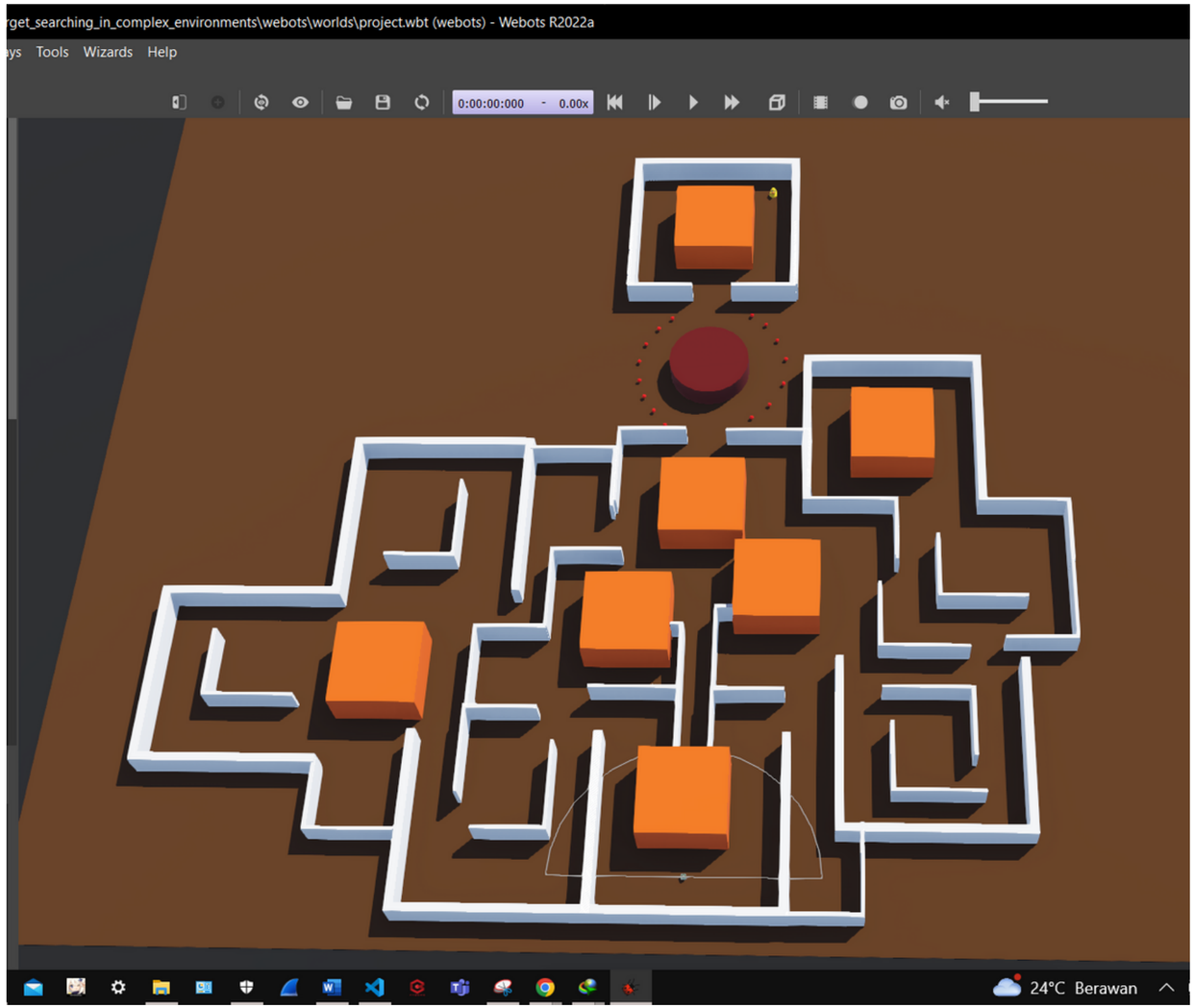
https://github.com/salvatorezam/motion_planning_and_target_searching_in_complex_environments



Luthfiana Zahra Firdausi
1103194081

Permasalahan

Terdapat robot yang diletakkan di dalam labirin. Robot tersebut diharuskan sampai pada lokasi tujuan (objek bebek) tanpa menabrak rintangan dan mencari jalur sederhana (tidak berputar ulang)



Metode Penting

1

LiDar (Light Distance and Rangin) merupakan metode pendeteksian objek yang menggunakan prinsip pantulan sinar untuk mengukur jarak objek (metode untuk menentukan jarak suatu objek).
Lidar digunakan untuk membuat robot keluar dari labirin tanpa menabrak penghalang labirin dengan cara memantulkan sinar yang nantinya akan ditangkap kembali oleh sensor pada robot sehingga robot tersebut dapat menghindari penghalang .

2

SLAM (Simultaneous localization and Mapping) merupakan teknik yang digunakan untuk membangun sebuah peta pada lingkungan yang belum diketahui melalui bantuan robot ataupun autonomous vechile, atau juga mengupdate peta yang sudah diketahui dan pada waktu yang sama mengenali lokasi dari robot tersebut.
SLAM bekerja dengan cara mendeteksi objek yang sudah ditentukan sebelumnya dan membuat robot mendekati objek hingga objek tersebut tidak terlihat pada sensor kamera robot

Material

1

Webots

perngkat lunak (software) yang digunakan sebagai model, program dan simulasi suatu robot bergerak (mobile robot).
webots merupakan aplikasi multiplatform dan open source yang mencakup semua alat yang diperlukan untuk pembuatan dan simulasi robot
Webots juga mendukung ROS (Robot Operating System) dan berbagai macam API untuk program dalam bahasa C, C++, Python, Java, dan Matlab.

Webots

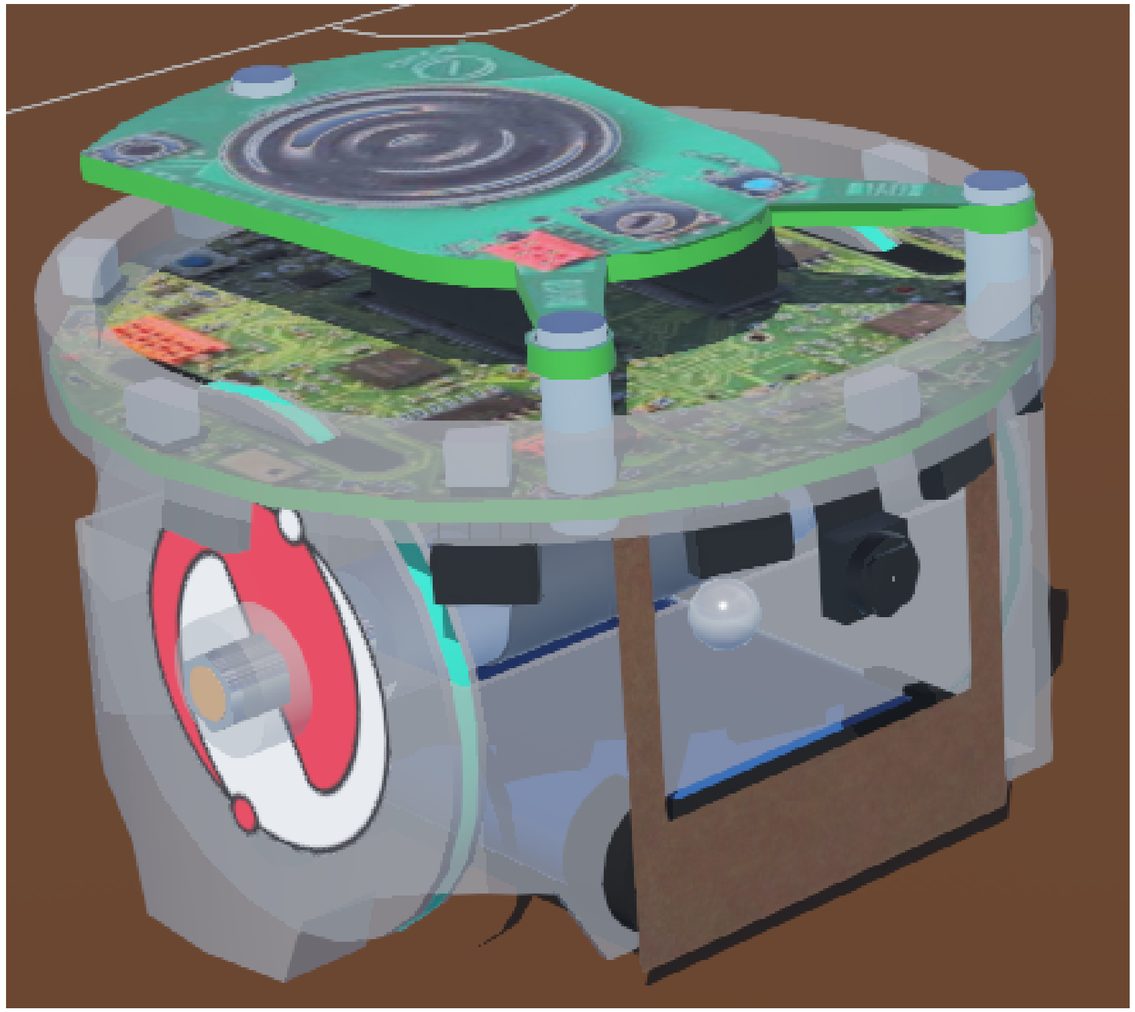
robot simulation

Material

2

E-puck robot

merupakan satu contoh dari robot yang telah di rakit pada aplikasi webots. E-puck memiliki beberapa sensor yang dapat digunakan untuk teknik lokalisasi. E-puck terdiri dari motor roda diferensial (encoder disimulasikan sebagai sensor posisi), sensor infrared merah yang berfungsi sebagai pengukur jarak dan cahaya, accelerometer, gyro, kamera, 8 LED disekelilingnya, bodi dan Led depan, Bluetooth dan ekstensi sensor ground



2

E-puck robot

yang digunakan pada percobaan kali ini lebih ditekankan pada

- motor: diperlukan untuk gerakan lurus dan rotasi robot
- sensor posisi: ada pada setiap roda, mereka mencatat kecepatan rotasinya untuk perkiraan jarak yang ditempuh
- LIDAR (Light Detection And Ranging): mendeteksi lingkungan sesuai jangkauannya melalui pemindaian laser
- camera: memberikan hasil vidio secara real-time dari perspektif robot untuk memperkirakan jarak dan jalur yang diambil

Feature	Description
Size	7.4 cm in diameter, 4.5 cm high
Weight	150 g
Battery	about 3 hours with the provided 5Wh LiION rechargeable battery
Processor	Microchip dsPIC 30F6014A @ 60MHz (about 15 MIPS)
Motors	2 stepper motors with 20 steps per revolution and a 50:1 reduction gear
IR sensors	8 infra-red sensors measuring ambient light and proximity of obstacles in a 4 cm range
Camera	color camera with a maximum resolution of 640x480 (typical use: 52x39 or 640x1)
Microphones	3 omni-directional microphones for sound localization
Accelerometer	3D accelerometer along the X, Y and Z axes
Gyroscope	3D gyroscope along the X, Y and Z axes
LEDs	8 red LEDs on the ring and one green LED on the body
Speaker	on-board speaker capable of playing WAV or tone sounds
Switch	16 position rotating switch
Bluetooth	Bluetooth for robot-computer and robot-robot wireless communication
Remote Control	infra-red LED for receiving standard remote control commands
Expansion bus	expansion bus to add new possibilities to your robot
Programming	C programming with the GNU GCC compiler system
Simulation	Webots facilitates the programming of e-puck with a powerful simulation, remote control and cross-compilation system

Algoritma

Setelah mengetahui tujuan dari robot tersebut, terdapat 4 operasi utama (mode), berupa:

- DEFAULT: robot bergerak dengan lurus menuju tujuan, dapat bergerak lurus lagi setelah berbelok
- AVOID_OBSTACLE: jika robot mendeteksi halangan, maka robot menghindari halangan tersebut (dapat terdeteksi melalui LiDar)
- FLANK_OBSTACLE: jika robot mendeteksi penghalang sudah dilalui dan jarak antara penghalang dan robot sudah jauh, maka robot akan kembali ke mode DEFAULT
- SLAM: mode ini diaktifkan jika robot mendeteksi objek yang sudah ditentukan, robot akan mengatur poin objek tersebut dan mendekatinya.

Terdapat pengontrol yang berisi informasi untuk mengatur sistem operasi dan juga memicu mode sesuai dengan keadaan.

```
# this variable determines the operational mode of the robot in each of the simulation iteration:
#   - DEFAULT: the robot turns to the goal direction and proceeds forward
#   - AVOID_OBSTACLE: an obstacle is detected between the robot and the goal -> avoid the obstacle
#   - FLANK_OBSTACLE: the robot flanks the obstacle that it just surpassed
#   - SLAM: the detection of the reference_point_obj enables the SLAM mode
```

Contoh Penggunaan Algoritma

perintah awal dijadikan DEFAULT agar robot dapat bergerak lurus kearah tujuan

```
403     if ACTION_TYPE == 'DEFAULT':
```

perintah AVOID_OBSTACLE diberikan agar disaat robot mendeteksi sebuah penghalang robot akan menghindarinya

```
427         elif ACTION_TYPE == 'AVOID_OBSTACLE':
428             print(f'Operational mode {ACTION_TYPE} - avoiding obstacle.')
```

perintah FLANK_OBSTACLE menandakan bahwa halangan sudah dilalui, jika jarak antara robot dan penghalang yang dilalui sudah jauh, robot akan kembali ke mode DEFAULT

```
470     elif ACTION_TYPE == 'FLANK_OBSTACLE':
471         print(f'Operational mode {ACTION_TYPE} - flanking obstacle.')
```

perintah SLAM akan membuat robot bergerak mendekati objek penanda sesuai urutan terdekat

```
494     elif ACTION_TYPE == 'SLAM':
495         print(f'Operational mode {ACTION_TYPE} - SLAM.')
```