Université Mohammed V - Rabat
Faculté des Sciences
Département de Physique

Année univ. 2024-2025
Master IASE
S3
Module robotique
Pr. Younès RAOUI

Travaux pratiques 1

Exercice : Programmer un robot Turtlebot3 pour suivre un chemin dans Webots

Objectif

Dans cet exercice, vous allez apprendre à programmer un robot Turtlebot3 dans Webots pour qu'il suive un chemin spécifique. Le robot se déplacera à travers quatre points, en ajustant sa position et son orientation à chaque étape.

Étape 1 : Chemin et Orientations Donnés

Le robot doit se déplacer à travers les points suivants, avec les orientations spécifiées à chaque point .

- $(x_0, y_0, \theta_0) = (0, 0, 0)$
- $(x_1, y_1, \theta_1) = (0.4, 0, \pi/2)$
- $(x_2, y_2, \theta_2) = (0.4, 0.4, \pi/2)$
- $(x_3, y_3, \theta_3) = (0, 0.4, \pi/2)$

Ouvrez Webots et démarrez une nouvelle simulation avec le robot Pioneer.
Dans votre script de contrôle du robot, importez `numpy` et les classes `Robot` et `Motor` de Webots

Étape 2 : Définir les Points du Chemin et les Orientations

(A implémenter)

Étape 3 : Initialiser le Robot Webots et Définir le Pas de Temps

Initialisez une instance de `Robot` et définissez le pas de temps :

```
robot = Robot()
timestep = int(robot.getBasicTimeStep()) * 2
```

Étape 4 : Initialiser les Moteurs

Récupérez les moteurs des roues gauche et droite, définissez leurs positions sur l'infini pour le contrôle de vitesse, et réglez la vitesse initiale à zéro :

```
left_motor = robot.getDevice('left wheel motor')
right_motor = robot.getDevice('right wheel motor')
left_motor.setPosition(float('inf'))
right_motor.setPosition(float('inf'))
left_motor.setVelocity(0.0)
right_motor.setVelocity(0.0)
```

Étape 5 :
Définir la Position de Départ, l'Orientation et les Paramètres du Robot

Définissez la position de départ, l'orientation, le rayon des roues et la distance entre les roues :

```
X0 = np.array([0, 0])
ori = 0.0
wheel_radius = 0.02  # Ajustez selon votre robot
wheel_distance = 0.1  # Ajustez selon votre robot
```

Étape 6 : Boucle pour Chaque Point du Chemin

Pour chaque point dans `xr` et `yr`, calculez la distance et l'orientation nécessaires pour atteindre le point en utilisant la transformation homogène de la position. Utilisez les étapes suivantes dans la boucle : A faire

Calculer les Vitesses Désirées : Calculez la vitesse linéaire v et la vitesse angulaire ω.

$$v = \sqrt{v_x^2 + v_y^2}$$

$$\omega = \frac{\theta - \text{ori}}{\text{timestep}}$$

Calculer les Vitesses des Roues : Utilisez v et ω pour calculer les vitesses des roues gauche et droite :

$$\text{vitesse\_gauche} = \frac{v - \frac{\text{wheel\_distance}}{2} \cdot \omega}{\text{wheel\_radius}}$$

$$\text{vitesse\_droite} = \frac{v + \frac{\text{wheel\_distance}}{2} \cdot \omega}{\text{wheel\_radius}}$$

Régler les Vitesses des Roues : Réglez les vitesses calculées sur les moteurs gauche et droit :

```
left_motor.setVelocity(left_speed)
right_motor.setVelocity(right_speed)
```

Exécuter la Simulation : Faites avancer la simulation pendant quelques pas de temps pour permettre au robot de se diriger vers la cible :

```
for _ in range(int(timestep)):
    robot.step(timestep)
```

Mettre à Jour la Position et l'Orientation : Après avoir atteint chaque point, mettez à jour la position de départ et l'orientation pour l'itération suivante :

Affichez la position et l'orientation actuelles pour vérifier que le robot suit le chemin