MACE ROBOTICS

Documentation du robot MR-Pico

Mise à jour : 02/02/24

Table des matières

1) Présentation du robot MR-Pico

MR-Pico est un petit robot mobile basé sur une carte Raspberry Pi Pico. Il est très facile à programmer et personnalisable. Sa petite taille de 100 mm de diamètre vous permet de facilement le programmer sur une table de bureau.

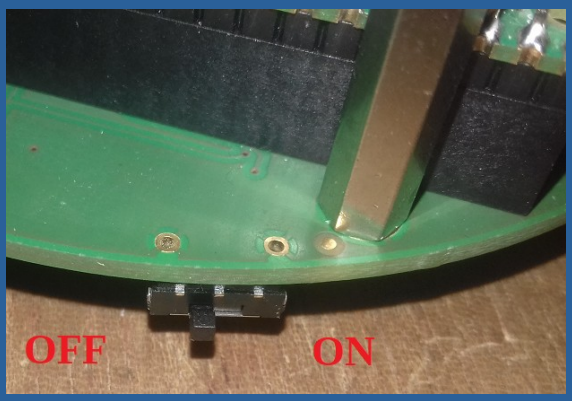
2) Logiciels & matériels nécessaire

Le langage MicroPython est préinstallé sur le robot MR-Pico.

Lien pour téléchager l'IDE : https://thonny.org/

3) Mise en marche du robot

Activez l'interrupteur ON/OFF sur la position ON.



- OFF : position à gauche
- ON : position à droite

4) Rechargement de la batterie

Pour recharger la batterie du robot MR-Pico vous avez besoin :

- un câble micro-USB,
- un PC ou un adaptateur secteur-USB

Ensuite pour recharger la batterie du robot :

1. Placez l'interrupteur sur la **position OFF** pour mettre le robot hors tension.
2. Branchez le câble micro-USB sur la carte Raspberry Pi PICO
3. La led bleu 'Charging' doit être allumé. Elle informe le rechargement de la batterie.
4. Lorsque la led est éteinte, ceci indique que la batterie est rechargée.

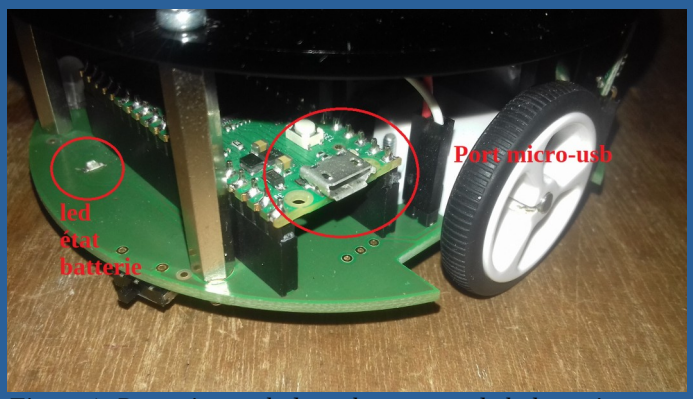


Figure 1: Port micro-usb de rechargement de la batterie

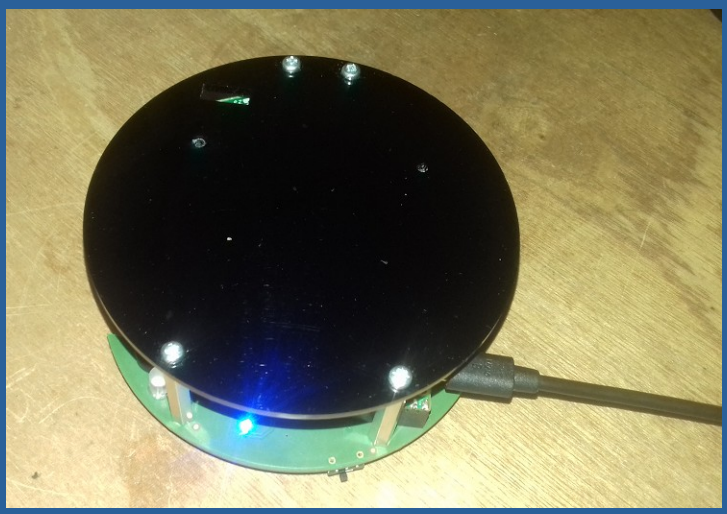


Figure 2: Robot en rechargement

5) État de la batterie, led interne verte

Si la led interne de la carte Raspberry Pi PICO ne clignote plus ceci indique que la tension de la batterie est trop basse. Lorsque la batterie est correctement rechargé la led interne de la Raspberry Pi PICO clignote de manière régulière.

A. Programmation du robot en python

Pour programmer le robot :

- ➔ Brancher un câble micro-usb sur le robot
- ➔ Ouvrir le logiciel Thonny
- ➔ Placer en position ON le robot et appuyer sur le bouton STOP de Thonny

Figure 3: Logiciel Thonny

1. Bibliothèques du robot

La librairie robot.py est disponible sur le github :

https://github.com/macerobotics/MR-Pico

Il est nécessaire d'installer sur le robot :

- robot.py
- encoder.py
- vl6180x.py

2. Avancer le robot

Un programme simple en langage Python pour faire avancer le robot MR-Pico :

```
import time
import robot

# avancer le robot avec une vitesse de 20 %
robot.forward(20)
```

3. Déplacements du robot

Un programme pour faire avancer, reculer et tourner le robot MR-Pico :

```
import time
import robot

# avancer le robot avec une vitesse de 20%
robot.forward(20)

time.sleep(2)

# reculer le robot avec une vitesse de 30%
robot.back(30)

time.sleep(3)

# tourner à droite avec une vitesse de 30%
robot.turnRight(30)

time.sleep(4)

# tourner à droite avec une vitesse de 30%
robot.turnLeft(30)

# avancer le robot avec une vitesse de 50%
robot.forward(50)
```

4. Lecture des encodeurs

Exemple de lecture des deux encodeurs du robot. Le robot est équipé de deux encodeurs magnétiques sur chaque moteur à courant continu :

```
import time
import robot

while True :
  coderRight = robot.encoderRight()
  coderLeft = robot.encoderLeft()
  print("coder Right =", coderRight)
  print("coder Left =", coderLeft)
  time.sleep(1)
```

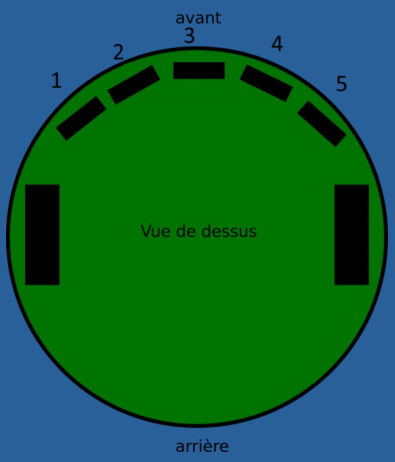
5. Lecture d’un capteur de proximité

Le robot MR-Pico est équipé de cinq connecteurs à sept pins afin de recevoir des capteurs de proximité de type :

- VL6180X de Pololu

MR-Pico peut avoir cinq capteurs maximum.

Voici le placement des cinq capteurs :



Lecture du capteur sur l’emplacement n°1 :

```
import time
import robot

while True :
  prox1 = robot.proxRead(1)
  print("Proxmimity sensor =", prox1)
  time.sleep(1)
```

Caractéristiques du capteur :

- Sortie en millimètres
- De 0 à 70 mm
- Si la distance est supérieure à 70mm, la sortie = 255

6. Utilisation du buzzer

Exemple d'utilisation du buzzer du robot :

```
import time
import robot

while True:
  robot.buzzer(600, 65_536/2)
  time.sleep(1)
  robot.buzzer(400, 65_536/3)
  time.sleep(5)
  robot.buzzer(1000, 65_536/3)
  time.sleep(5)
  robot.buzzerStop()
  time.sleep(5)
```

7. Lecture de la tension batterie

Exemple de la lecture de la tension de la batterie du robot :

```
import time
import robot

while True :
  tension = robot.battery()
  print("Battery tension =", tension)
  time.sleep(1)
```

8. Utilisation d’un servomoteur

Le robot MR-Pico a deux connecteurs pour le contrôle de servomoteurs :

Contrôle d’un servomoteur sur les connecteurs à 3 pins.

```
import time
import robot

while True:
  robot.servo_Angle(0,0)# angle de 0°
  time.sleep(2)

  robot.servo_Angle(0,90)# angle de 90°
  time.sleep(2)

  robot.servo_Angle(0,180)# angle de 180°
  time.sleep(2)
```

9. Utilisation de la led RGB

Utilisation de led RGB du robot :

```
import time
import robot

while True :
  robot.ledRgb(1,0,0)
  time.sleep(1)
  robot.ledRgb(0,1,0)
  time.sleep(1)
  robot.ledRgb(0,0,1)
  time.sleep(1)
```

10. Gestion des obstacles avec 3 capteurs

Un exemple de gestion des obstacles avec l'utilisation de 3 capteurs d'obstacles :

```
import time
import robot


SEUIL_OBS = 60 # 60 mm


while True:
    p2 = robot.proxRead(2) # Lecture des capteurs
    p3 = robot.proxRead(3)
    p4 = robot.proxRead(4)

    if ((p2 < SEUIL_OBS)and(p3 < SEUIL_OBS)and(p4 < SEUIL_OBS)):
        robot.stop()
    elif ((p2 < SEUIL_OBS)and(p3 > SEUIL_OBS)and(p4 > SEUIL_OBS)):
        robot.turnRight(25)
    elif ((p2 > SEUIL_OBS)and(p3 > SEUIL_OBS)and(p4 < SEUIL_OBS)):
        robot.turnLeft(25)
    else:
        robot.forward(25)
```

11. Déplacements précis avec contrôle

Dans cette partie vous allez apprendre à gérer les déplacements du robot MR-Pico avec la prise en compte des encodeurs en quadrature. Les encodeurs des moteurs vont permettre de connaître la vitesse et le déplacement du robot.

- Faire avancer ou reculer le robot d'une distance *n* en millimètre
- Faire tourner d'un angle *b* en degré

Exemple, pour faire avancer le robot de 150 mm puis de 100 mm en ligne droite :

```
import time
import robot


robot.forwardmm(150,20)

time.sleep(2)
```

```
robot.forwardmm(100,25)
```

Exemple, pour faire tourner le robot de 90° vers la droite :

```
import time
import robot


robot.forwardmm(150,20)

time.sleep(2)

robot.turnAngle(90,25)
```

Exemple, pour faire reculer le robot de 300 mm :

```
import time
import robot


robot.forwardmm(-300,20)
```

Exemple, pour faire tourner le robot de 90° vers la gauche:

```
import time
import robot


robot.forwardmm(150,20)

time.sleep(2)

robot.turnAngle(-90,25)
```

12. Contrôle des moteurs

Exemple, pour faire tourner le robot vers la droite avec la gestion des moteurs à deux vitesses différentes :

- motorRight(Direction, vitesse)
  - Direction : 0 ou 1
  - Vitesse : 0 à 100

```
import time
import robot

robot.motorRight(1,45)
robot.motorLeft(0,25)
time.sleep(2)
robot.stop()
```

6) API

- battery() : lecture de la tension batterie
- proxRead() : lecture capteur de proximité
- encoderRight() : lecture encodeur droit
- encoderLeft() : lecture encodeur gauche
- encoderReset() : reset encodeurs
- ledRgb() : gestion de la led RGB
- buzzer() : gestion du buzzer
- motorRight() : contrôle moteur droit
- motorLeft() : contrôle moteur gauche
- forward() : avancer
- back() : reculer
- turnRight() : tourner à droite
- turnLeft():tourner à gauche
- stop() : stop robot