Relatório - Projeto CC7711

Ian Rocha
Lucas Kraus
Lucas Trevizoli
Felipe Del Amore

Esse projeto visa utilizar dos sensores do robô “E-Puck” para identificar quando uma caixa foi movida no cenário através de uma colisão.

Nosso código foi desenvolvido em Python.
Declaramos algumas variáveis universais de controle:

controllerPY.py 1

C: > Users > ian21 > Desktop > controllerPY.py > ...

```python
from controller import Robot, Motor, DistanceSensor, Supervisor, LED

passTime = 256
maxSpeed = 6.28
obstacleL = 100
LEDs = 10
movementBox = 1e-4
```

E as funcionalidades do robô que iremos utilizar, como motores, sensores de proximidade e LED:

```python
# Inicializando as funções do robô
def inicializar_robo():
    robo = Robot()
    left_motor = robo.getDevice("left wheel motor")
    right_motor = robo.getDevice("right wheel motor")

    leds = [robo.getDevice("led0")]
    leds[0].set(-1)

    left_motor.setPosition(float('inf'))
    right_motor.setPosition(float('inf'))

    left_motor.setVelocity(0.0)
    right_motor.setVelocity(0.0)

    return robo, left_motor, right_motor, leds

# Sensores de proximidade
def inicializar_proxSensors(robo):
    proxSensorVar = ["ps0", "ps1", "ps2", "ps3", "ps4", "ps5", "ps6", "ps7"]
    proxSensors = [robo.getDevice(nome) for nome in proxSensorVar]
    for sensor in proxSensors:
        sensor.enable(passTime)
    return proxSensors
```

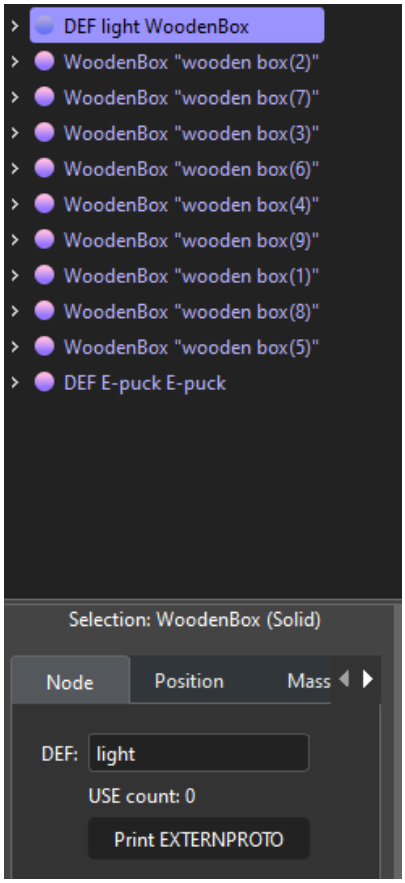
Temos uma função para desviar de obstáculos após impacto, para que possamos determinar se ele se moveu ou não:

```python
# Velocidade baseada em proximidade / desviar de obstáculos
def changeSpeed(sensorData):
    left_speed = maxSpeed
    right_speed = maxSpeed

    if sensorData[0] > obstacleL or sensorData[7] > obstacleL:
        left_speed = -min(left_speed, maxSpeed)
        right_speed = min(right_speed, maxSpeed / 2)

    elif sensorData[6] > obstacleL:
        left_speed = min(left_speed / 2, maxSpeed)
        right_speed = -min(right_speed, maxSpeed)

    elif sensorData[1] > obstacleL:
        left_speed = -min(left_speed, maxSpeed)
        right_speed *= min(right_speed / 2, maxSpeed)

    left_speed = max(min(left_speed, maxSpeed), -maxSpeed)
    right_speed = max(min(right_speed, maxSpeed), -maxSpeed)

    return left_speed, right_speed
```

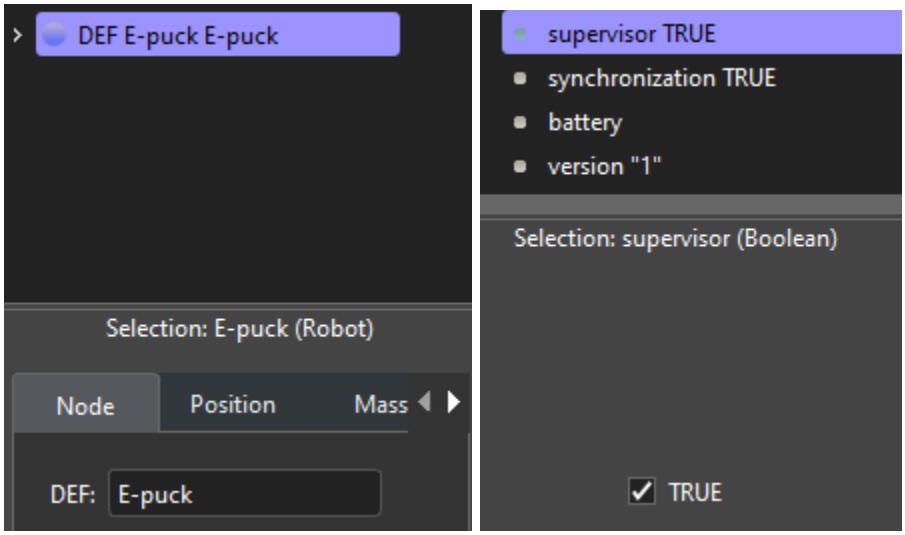
Por fim, temos a função main, que recebe a posição da caixa nomeada como “light” (leve) e mantém o robô andando até que a caixa se mova. Quando a caixa se mover, o robô para e os LEDs se acendem.

```python
def main():
    robo, left_motor, right_motor, leds = inicializar_robo()
    proxSensors = inicializar_proxSensors(robo)
    supervisor = Supervisor()
    objetoNode = supervisor.getFromDef("light")
    position = objetoNode.getPosition()
    xb, yb = position[0], position[1]
    x, y = xb, yb

    while robo.step(passTime) != -1:
        sensorData = [sensor.getValue() for sensor in proxSensors]
        caixa = objetoNode.getPosition()
        xb, yb = caixa[0], caixa[1]

        left_speed, right_speed = changeSpeed(sensorData)

        # Se a caixa "light" foi movida, acionar leds e parar robô
        if abs(xb - x) > movementBox or abs(yb - y) > movementBox:
            left_speed = 0
            right_speed = 0
            leds[0].set(leds[0].get() * -1)
            print("[+] Caixa movida!")

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

if __name__ == "__main__":
    main()
```

Para reproduzir o funcionamento do código, é necessário algumas alterações no WeBots. Primeiro, o DEF da caixa leve deve ser trocado para “light”:



E no robô, mudar o DEF para “E-puck” e ativar o supervisor:



Link para a pasta com o mundo e o controlador:
https://github.com/IanRRocha/EPuckController