Gabriel Ćwiek, 306387,   
<https://robotbenchmark.net/gabcwiwimip>  
Kody: <https://github.com/Astaree/isp>

Rozwiązania zadań na ocenę 4,5 – wszystkie z 1 i 2 \* w top 10, 3 \* w top 15:

# Spis treści:

* 1. [Pick & place](#_Pick_and_place:)
  2. [Pendulum](#_Pendulum)
  3. [Pit escape](#_Pit_escape)

# Pick and place:

"""Sample Webots controller for the pick and place benchmark."""

from controller import Robot

# Create the Robot instance.

robot = Robot()

# Get the time step of the current world.

timestep = int(robot.getBasicTimeStep())

# Inizialize base motors.

wheels = []

wheels.append(robot.getMotor("wheel1"))

wheels.append(robot.getMotor("wheel2"))

wheels.append(robot.getMotor("wheel3"))

wheels.append(robot.getMotor("wheel4"))

for wheel in wheels:

# Activate controlling the motors setting the velocity.

# Otherwise by default the motor expects to be controlled in force or position,

# and setVelocity will set the maximum motor velocity instead of the target velocity.

wheel.setPosition(float('+inf'))

# Initialize arm motors.

armMotors = []

armMotors.append(robot.getMotor("arm1"))

armMotors.append(robot.getMotor("arm2"))

armMotors.append(robot.getMotor("arm3"))

armMotors.append(robot.getMotor("arm4"))

armMotors.append(robot.getMotor("arm5"))

# Set the maximum motor velocity.

armMotors[0].setVelocity(0.2)

armMotors[1].setVelocity(0.5)

armMotors[2].setVelocity(0.5)

armMotors[3].setVelocity(0.3)

# Initialize arm position sensors.

# These sensors can be used to get the current joint position and monitor the joint movements.

armPositionSensors = []

armPositionSensors.append(robot.getPositionSensor("arm1sensor"))

armPositionSensors.append(robot.getPositionSensor("arm2sensor"))

armPositionSensors.append(robot.getPositionSensor("arm3sensor"))

armPositionSensors.append(robot.getPositionSensor("arm4sensor"))

armPositionSensors.append(robot.getPositionSensor("arm5sensor"))

for sensor in armPositionSensors:

sensor.enable(timestep)

# Initialize gripper motors.

finger1 = robot.getMotor("finger1")

finger2 = robot.getMotor("finger2")

# Set the maximum motor velocity.

finger1.setVelocity(0.03)

finger2.setVelocity(0.03)

# Read the miminum and maximum position of the gripper motors.

fingerMinPosition = finger1.getMinPosition()

fingerMaxPosition = finger1.getMaxPosition()

# Move forward.

for wheel in wheels:

wheel.setVelocity(14.0)

# Wait until the robot is in front of the box.

# Move arm and open gripper.

armMotors[1].setPosition(-0.53)

armMotors[2].setPosition(-0.9)

armMotors[3].setPosition(-1.1)

finger1.setPosition(fingerMaxPosition)

finger2.setPosition(fingerMaxPosition)

robot.step(260 \* timestep)

# Stop moving forward.

for wheel in wheels:

wheel.setVelocity(0.0)

# Move arm and open gripper.

armMotors[1].setPosition(-0.55)

armMotors[2].setPosition(-0.9)

armMotors[3].setPosition(-1.5)

finger1.setPosition(fingerMaxPosition)

finger2.setPosition(fingerMaxPosition)

# Monitor the arm joint position to detect when the motion is completed.

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

if abs(armPositionSensors[3].getValue() - (-1.2)) < 0.01:

# Motion completed.

break

# Close gripper.

finger1.setPosition(0.013)

finger2.setPosition(0.013)

# Wait until the gripper is closed.

robot.step(50 \* timestep)

# Lift arm.

armMotors[1].setPosition(-.3)

# Wait until the arm is lifted.

robot.step(5 \* timestep)

# Move forward.

wheels[1].setVelocity(-14)

wheels[2].setVelocity(-14)

armMotors[0].setPosition(-1.8)

armMotors[3].setPosition(-0)

armMotors[2].setPosition(-0.3)

armMotors[1].setPosition(-.9)

robot.step(450 \* timestep/2)

# Move forward.

wheels[0].setVelocity(-7)

wheels[1].setVelocity(-7)

wheels[2].setVelocity(-7)

wheels[3].setVelocity(-7)

armMotors[3].setPosition(-1.35)

robot.step(410 \* timestep/2)

# Rotate the robot.

wheels[0].setVelocity(-7)

wheels[1].setVelocity(7)

wheels[2].setVelocity(-7)

wheels[3].setVelocity(7)

robot.step(600 \* timestep/7)

# Move forward.

wheels[1].setVelocity(7)

wheels[3].setVelocity(7)

robot.step(520 \* timestep/7)

# Stop.

for wheel in wheels:

wheel.setVelocity(0.0)

# Move arm down

# Open gripper.

finger1.setPosition(fingerMaxPosition)

finger2.setPosition(fingerMaxPosition)

robot.step(50 \* timestep)

# Pendulum

"""Sample Webots controller for the inverted pendulum benchmark."""

from controller import Robot

import math

# Get pointer to the robot.

robot = Robot()

# Get the time step of the current world.

timestep = int(robot.getBasicTimeStep())

# Get pointers to the position sensor and enable it.

ps = robot.getPositionSensor('pendulum sensor')

ps.enable(timestep)

# Get pointers to the motors and set target position to infinity (speed control).

leftMotor = robot.getMotor("left wheel motor")

rightMotor = robot.getMotor("right wheel motor")

leftMotor.setPosition(float('+inf'))

rightMotor.setPosition(float('+inf'))

leftMotor.setVelocity(0.0)

rightMotor.setVelocity(0.0)

maxSpeed = min(rightMotor.getMaxVelocity(), leftMotor.getMaxVelocity())

# Define the PID control constants and variables.

KP = 32

KI = 77

KD = 5

integral = 0.0

previous\_position = 0.0

cnt = 0

# Initialize the robot speed (left wheel, right wheel).

leftMotor.setVelocity(0.0)

rightMotor.setVelocity(0.0)

# Main loop: perform a simulation step until the simulation is over.

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

# Read the sensor measurement.

position = ps.getValue()

# Stop the robot when the pendulum falls.

if math.fabs(position) > math.pi \* 0.5:

leftMotor.setVelocity(0.0)

rightMotor.setVelocity(0.0)

break

# PID control.

integral = integral + (position + previous\_position) \* 0.5 / timestep

derivative = (position - previous\_position) / timestep

speed = KP \* position + KI \* integral + KD \* derivative

# Clamp speed to the maximum speed.

if speed > maxSpeed:

speed = maxSpeed

elif speed < -maxSpeed:

speed = -maxSpeed

# Set the robot speed (left wheel, right wheel).

leftMotor.setVelocity(-speed)

rightMotor.setVelocity(-speed)

# Store previous position for the next controller step.

previous\_position = position

cnt+=1

KP = 32 + (cnt/1000)\*\*2

KI = 77 - (cnt/10000)\*\*2

if KI <0:

KI=0

KD = 5 + (cnt/650)\*\*2

# Pit escape

"""Sample Webots controller for the pit escape benchmark."""

from controller import Robot

robot = Robot()

timestep = int(robot.getBasicTimeStep())

# Max possible speed for the motor of the robot.

maxSpeed = 8.72

# Configuration of the main motor of the robot.

pitchMotor = robot.getMotor("body pitch motor")

pitchMotor.setPosition(float('inf'))

pitchMotor.setVelocity(0.0)

gyro = robot.getGyro("body gyro")

gyro.enable(timestep)

# This is the time interval between direction switches.

# The robot will start by going forward and will go backward after

# this time interval, and so on.

# At first we go forward.

pitchMotor.setVelocity(maxSpeed)

forward = True

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

now = robot.getTime()

# We check if enough time has elapsed.

if gyro.getValues()[0] > 0 and gyro.getValues()[1] > 0 or gyro.getValues()[0] > 0 and gyro.getValues()[2] > 0:

pitchMotor.setVelocity(maxSpeed)

else:

pitchMotor.setVelocity(-maxSpeed)