МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

НИЖЕГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ

УНИВЕРСИТЕТ им. Р.Е.АЛЕКСЕЕВА

Институт радиоэлектроники и информационных технологий

Кафедра «Вычислительные системы и технологии»

Отчет

по лабораторной работе №4

по дисциплине «Аппаратное и программное обеспечение роботизированных систем»

«Программирование алгоритмов управления роботов в Webots»

РУКОВОДИТЕЛЬ:

_____ Гай В.Е.

СТУДЕНТ:

_____ Ширшов А.А.

_____ Юрчук М.С.

19-В-1

Работа защищена «___» _____

С оценкой _____

Нижний Новгород 2022

Цель: получение навыков работы с алгоримами управления роботами.

Задание: выполнить вариант и загрузить программу на платформу для соревнований, записать видео работы, подготовить отчет с подробным описанием результатов. В отчет вставить результаты с соревнования.

Вариант 4. Держись за стену (Wall following)

Этот тест направлен на разработку программы управления роботом Pioneer, который должен следовать вдоль стены.

У робота есть ровно 1 минута, чтобы пройти как можно дальше, стараясь при этом держаться на расстоянии 50 см от стены. Программа сначала вычислит "идеальный" путь. Первый отрезок этого пути представляет собой прямую линию, идущую от начального положения робота до ближайшей точки, которая находится на расстоянии 50 см от стены. Затем путь продолжается параллельно стене на расстоянии ровно 50 см. Производительность робота будет рассчитываться на основе среднего расстояния до этого "идеального" пути.

Пояснение к коду:

Изначально программа работала лишь с 2 датчиками и поэтому робот не мог обходить препятствия. Решением может стать использование большего кол-ва датчиков.

В случае, если значения с датчиков по центру и слева маленькие, робот встретил препятствие — то робот поворачивает вправо.

В случае, если значения с датчиков по центру большие, а слева — маленькие, то робот следует по стене — он должен держать дистанцию 50 см, для этого используется ПИД-регулятор. Если значения с датчиков увеличились (front_left — 50 > 0) — робот повернет налево, иначе направо (front_left — 50 < 0).

В случае, если значения с центральных и левых датчиков большие — робот должен повернуть на право до тех пор, пока не встретит стену слева на расстоянии примерно 50 см.

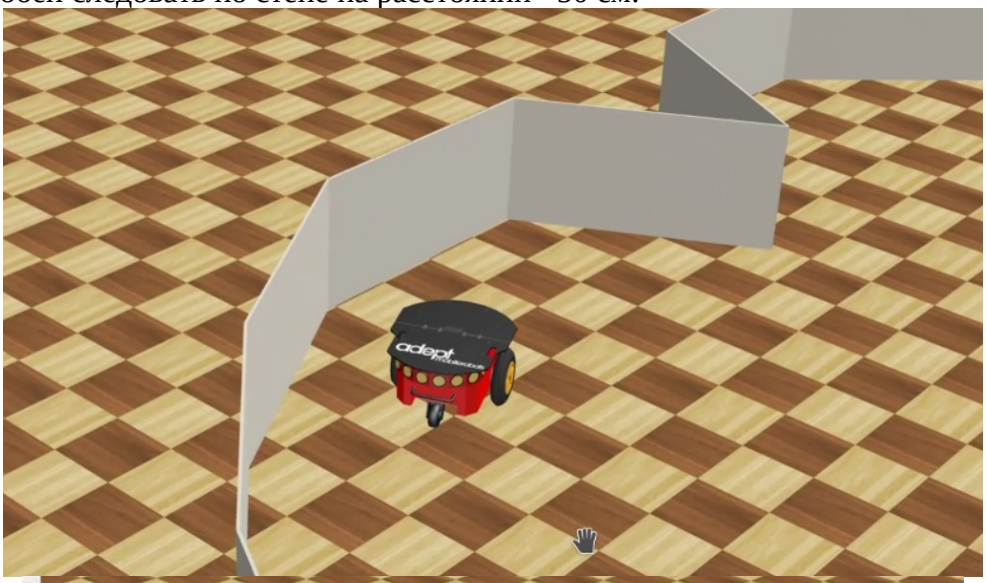
Исходный код:

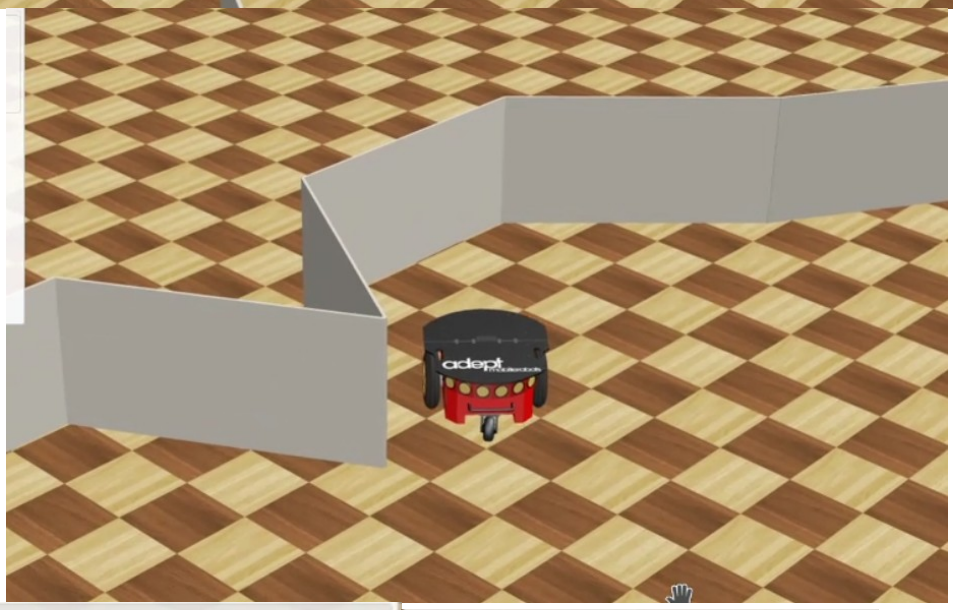
```
"""Sample Webots controller for the wall following benchmark."""

from controller import Robot
from math import sqrt

def clamp(num, min_value, max_value):
   return max(min(num, max_value), min_value)

def getDistance(sensor):
    return clamp(((1000 - abs(sensor.getValue())) / 100), 0.0, 5.0)


def PIDctl(prev):
    up = kp*prev[4]
    ud = (prev[4] - prev[0]) * kd
    ui = sum(prev)

    if (ui > max_ui):
        ui = max_ui
    elif (ui < min_ui):
        ui = min_ui
    ui *= ki

    # Debug

    print(">> ", prev[4])
    print(">> %3.3f | %3.3f | %3.3f = %3.3f" % (up, ud, ui, up + ud + ui))

    return up + ud + ui
```

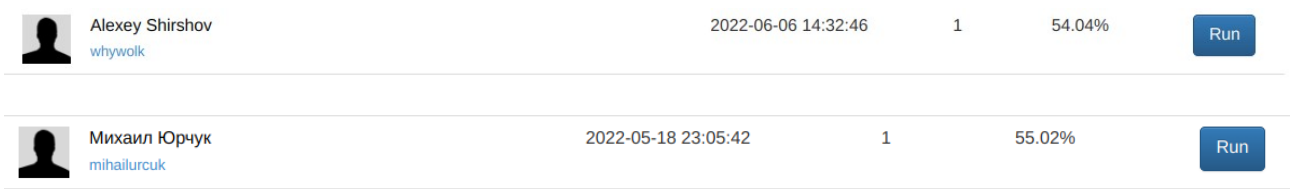
```python
# Maximum speed for the velocity value of the wheels.
# Don't change this value.
MAX_SPEED = 5.24

prev_left = [0] * 5
prev_left_center = [0] * 5
prev_center = [0] * 5

kp = 1
kd = 0.5
ki = 0.2
min_ui = -0.2
max_ui = 0.2


robot = Robot()

# Get the time step of the current world.
timestep = int(robot.getBasicTimeStep())

# Get pointer to the robot wheels motors.
leftWheel = robot.getMotor('left wheel')
rightWheel = robot.getMotor('right wheel')

# We will use the velocity parameter of the wheels, so we need to
# set the target position to infinity.
leftWheel.setPosition(float('inf'))
rightWheel.setPosition(float('inf'))

# Get and enable the distance sensors.
front_left_sensor = robot.getDistanceSensor("so0")
back_left_sensor = robot.getDistanceSensor("so15")
front_sensor_1 = robot.getDistanceSensor("so1")
front_sensor_2 = robot.getDistanceSensor("so2")
front_center_sensor = robot.getDistanceSensor("so3")
front_sensor_4 = robot.getDistanceSensor("so4")
front_sensor_5 = robot.getDistanceSensor("so5")

front_left_sensor.enable(timestep)
back_left_sensor.enable(timestep)
front_sensor_1.enable(timestep)
front_sensor_2.enable(timestep)
front_center_sensor.enable(timestep)
front_sensor_4.enable(timestep)
front_sensor_5.enable(timestep)

# Move forward until we are 50 cm away from the wall.
leftWheel.setVelocity(MAX_SPEED)
rightWheel.setVelocity(MAX_SPEED)
while robot.step(timestep) != -1:
    if getDistance(front_center_sensor) < 0.25:
        break

# Rotate clockwise until the wall is to our left.
leftWheel.setVelocity(MAX_SPEED)
rightWheel.setVelocity(-MAX_SPEED)
while robot.step(timestep) != -1:
    # Rotate until there is a wall to our left, and nothing in front of us.
    if getDistance(back_left_sensor) < 0.26:
        break

leftWheel.setVelocity(MAX_SPEED)
rightWheel.setVelocity(MAX_SPEED)
```

```
# Main loop.
while robot.step(timestep) != -1:
        # Read values from sensors
        front_left_sensor_value = getDistance(front_left_sensor) # 0
        back_left_sensor_value = getDistance(back_left_sensor)
        front_sensor_1_value = getDistance(front_sensor_1)
        front_sensor_2_value = getDistance(front_sensor_2)
        front_center_sensor_value = getDistance(front_center_sensor) # 3
        front_sensor_4_value = getDistance(front_sensor_4)
        front_sensor_5_value = getDistance(front_sensor_5)

        print("front: ", front_left_sensor_value, ", back: ", back_left_sensor_value, ",
center: ", front_center_sensor_value)
        print("sensor 1: ", front_sensor_1_value, ", sensor 2: ", front_sensor_2_value)

        left_wheel_speed = 0.8
        right_wheel_speed = 1


        if min(front_center_sensor_value, front_sensor_2_value, front_sensor_1_value) <
0.6:
                left_wheel_speed = 1
                right_wheel_speed = 0.35
                print("going idk")


        elif front_left_sensor_value < 0.6 and back_left_sensor_value >
front_left_sensor_value:
                left_wheel_speed = 1
                if front_left_sensor_value <= 0.01:
                        right_wheel_speed = 0.1
                else:
                        right_wheel_speed = 1 - front_left_sensor_value / 0.6
                print("need to go right")


        elif back_left_sensor_value < 0.6 and front_left_sensor_value > 1:
                if back_left_sensor_value <= 0.01:
                        left_wheel_speed = 0.1
                else:
                        left_wheel_speed = 1 - back_left_sensor_value / 0.6
                right_wheel_speed = 1
                print("need to go left")


        elif min(front_left_sensor_value, back_left_sensor_value, front_sensor_1_value) >
0.6:
                left_wheel_speed =      1 - (min(front_left_sensor_value,
back_left_sensor_value, front_sensor_1_value) / 5)
                right_wheel_speed = 1
                print("going right")

        print("left speed: ", left_wheel_speed, "right speed: ", right_wheel_speed)
        leftWheel.setVelocity(clamp(left_wheel_speed * MAX_SPEED, 0.0, 4.192)) #
MAX_SPEED * 0.8 = 4.192
        rightWheel.setVelocity(clamp(right_wheel_speed * MAX_SPEED, 0.0, MAX_SPEED))



# Stop the robot when we are done.
leftWheel.setVelocity(0)
rightWheel.setVelocity(0)
```

Результат:
Робот способен следовать по стене на расстоянии ~50 см:







Вывод: в результате выполнения данной лабораторной работы были получены алгоритмы для управления роботов с помощью обратной связи (датчиков). Был разработан алгоритм, по которому робот может следовать по стене на расстоянии 50 см.