Федеральное агентство по образованию

Нижегородский Государственный Технический Университет

Кафедра: «Вычислительные системы и технологии»

Аппаратное и программное обеспечение роботизированных систем

**Лабораторная работа №2**

Выполнил:

Студент группы 19-В-1

Бубнов Е.А

Проверил:

Гай В. Е.

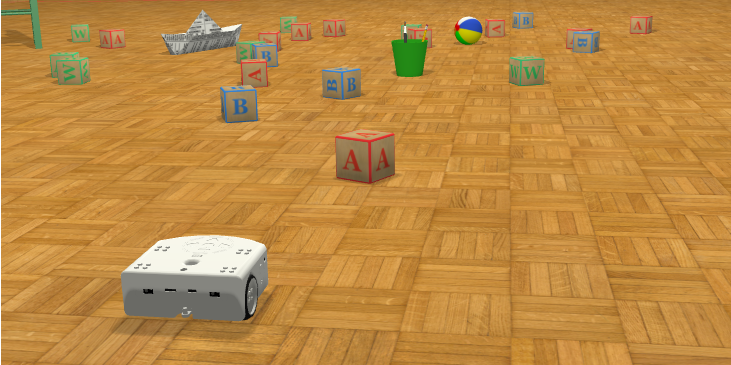
Нижний Новгород

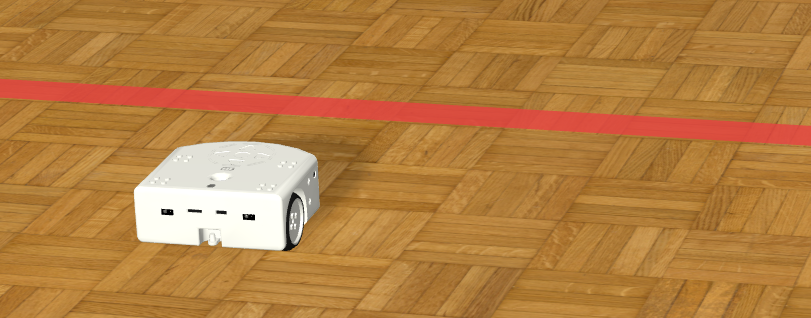
2022

Цель: Написать контроллер для робота в двух бенчмарках

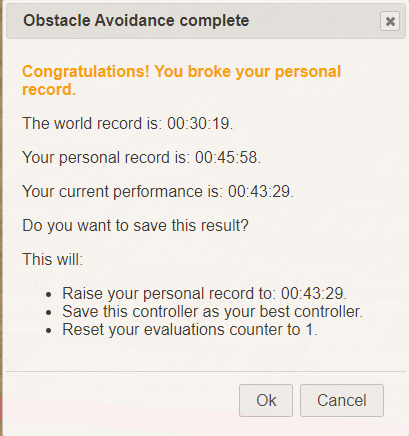
**Задание 1: Obstacle Avoidance**

Робот





Результат:



**Контроллер:**

"""Braitenberg-based obstacle-avoiding robot controller."""

from controller import Robot

# import Compass module

from controller import Compass

from math import \*

# Get reference to the robot.

robot = Robot()

# Get simulation step length.

timeStep = int(robot.getBasicTimeStep())

# Constants of the Thymio II motors and distance sensors.

maxMotorVelocity = 9.53

distanceSensorCalibrationConstant = 360

# get robot's Compass device

compass = robot.getDevice("compass")

# Get left and right wheel motors.

leftMotor = robot.getDevice("motor.left")

rightMotor = robot.getDevice("motor.right")

# Get frontal distance sensors.

outerLeftSensor = robot.getDevice("prox.horizontal.0")

centralLeftSensor = robot.getDevice("prox.horizontal.1")

centralSensor = robot.getDevice("prox.horizontal.2")

centralRightSensor = robot.getDevice("prox.horizontal.3")

outerRightSensor = robot.getDevice("prox.horizontal.4")

# Enable distance sensors.

outerLeftSensor.enable(timeStep)

centralLeftSensor.enable(timeStep)

centralSensor.enable(timeStep)

centralRightSensor.enable(timeStep)

outerRightSensor.enable(timeStep)

# enable the Compass

compass.enable(timeStep)

# Disable motor PID control mode.

leftMotor.setPosition(float('inf'))

rightMotor.setPosition(float('inf'))

# Set ideal motor velocity.

initialVelocity = 0.7 \* maxMotorVelocity

# Set the initial velocity of the left and right wheel motors.

leftMotor.setVelocity(initialVelocity)

rightMotor.setVelocity(initialVelocity)

def get\_bearing\_in\_degrees():

north = compass.getValues()

rad = atan2(north[0], north[2])

bearing = (rad - 1.5708) / pi \* 180.0

if (bearing < 0.0):

bearing = bearing + 360.0

return bearing

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

# Read values from four distance sensors and calibrate.

outerLeftSensorValue = outerLeftSensor.getValue() / distanceSensorCalibrationConstant

centralLeftSensorValue = centralLeftSensor.getValue() / distanceSensorCalibrationConstant

centralSensorValue = centralSensor.getValue() / distanceSensorCalibrationConstant

centralRightSensorValue = centralRightSensor.getValue() / distanceSensorCalibrationConstant

outerRightSensorValue = outerRightSensor.getValue() / distanceSensorCalibrationConstant

changeLeft = initialVelocity - (centralRightSensorValue + outerRightSensorValue) / 2

changeRight = initialVelocity - (centralLeftSensorValue + outerLeftSensorValue) / 2 - centralSensorValue

#print((centralRightSensorValue + outerRightSensorValue) / 2)

#print((centralLeftSensorValue + outerLeftSensorValue) / 2 - centralSensorValue)

###########################################

if ( ceil(changeLeft) == 7. and ceil(changeRight)== 7. ):

#print(get\_bearing\_in\_degrees())

degree = get\_bearing\_in\_degrees()

#print(ceil(degree))

if (ceil(degree) < 90):

changeLeft = 7.

if (ceil(degree) > 90):

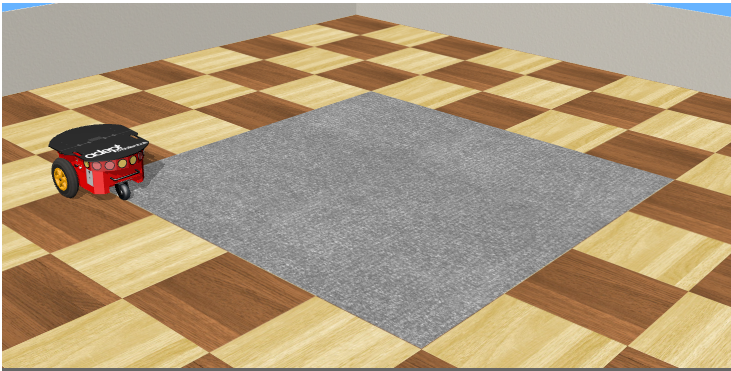
changeRight = 7.

# Set wheel velocities based on sensor values, prefer right turns if the central sensor is triggered.

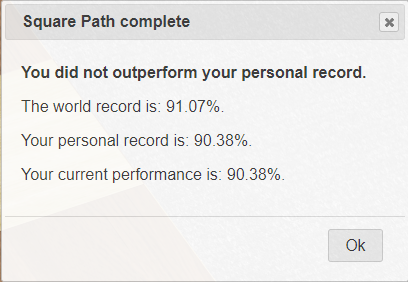
leftMotor.setVelocity(changeLeft)

rightMotor.setVelocity(changeRight)

**Задание 2: Square Path**



Результат:



**Контроллер:**

"""Sample Webots controller for the square path benchmark."""

from controller import Robot

# Get pointer to the robot.

robot = Robot()

# Get pointer to each wheel of our robot.

leftWheel = robot.getMotor('left wheel')

rightWheel = robot.getMotor('right wheel')

# Repeat the following 4 times (once for each side).

for i in range(0, 4):

    leftWheel.setPosition(float('inf'))

    rightWheel.setPosition(float('inf'))

    leftWheel.setVelocity(5.24)

    rightWheel.setVelocity(5.24)

    # Wait for the robot to reach a corner.

    if i == 0:

        robot.step(3900)

    if i == 1:

        robot.step(3950)

    if i == 2:

        robot.step(3960)

    if i == 3:

        robot.step(3960)

    leftWheel.setVelocity(5.24)

    rightWheel.setVelocity(-5.01)

    # Wait until the robot has turned 90 degrees clockwise.

    robot.step(480)

# Stop the robot when path is completed, as the robot performance

# is only computed when the robot has stopped.

leftWheel.setVelocity(0)

rightWheel.setVelocity(0)