МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего 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

Программное обеспечение вычислительных сетей

РУКОВОДИТЕЛЬ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Гай В.Е.

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Аверьянова А.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сапожников В.О.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Конева К.C.

19-В-1

Работа защищена «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_

С оценкой \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород 2022

**Марафон гуманоидов (Humanoid marathon)**

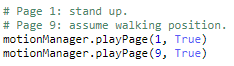
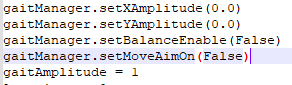
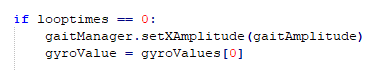
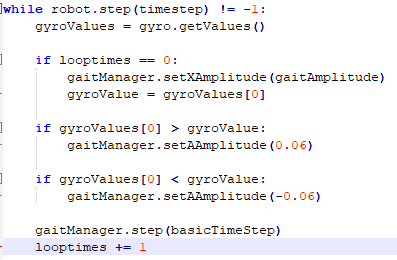
Этот тест направлен на разработку компьютерной программы, которая управляет гуманоидным роботом, чтобы пройти как можно дальше, учитывая ограниченное время автономной работы. Язык программирования - Python, модель гуманоидного робота - **ROBOTIS OP2**.

Окружающая среда представляет собой масштабную модель, основанную на размере робота.

Эталонная метрика — это расстояние, пройденное роботом.

Расстояние — это разница между начальным и конечным положением робота на мировой оси x, параллельной улице.

Если робот падает или пытается встать на четвереньки, тест немедленно завершается.

1. Для экономии заряда отключим светодиоды.   
   ****  
   Поскольку светодиоды не используются то и сам модуль LED нам не нужен  
   ****
2. Поскольку при падении робота испытание считается проваленным, то оставим одно предопределённое движение – встать в позицию для ходьбы, а движение подняться удалим.  
   ****
3. Отключим баланс и MoveAimOn. Так же увидим амплитуда по оси X (увеличим длину шага)  
   ****
4. Изменения в цикле движения:  
   Движение робот начинает не с 45 интервала времени, а с 0. При движении с 45 интервала робот после того, как встал в устойчивое положение делает несколько шагов на месте, когда при движении с 0 он сразу движется вперед.  
   ****
5. Так же мы читаем значения гороскопа и корректируем направления движения, чтобы робот двигался прямо.   
   Значение 0.06 было подобрано экспериментально, т.к. отличных знаениях робот меняет траекторию движения.

**Листинг**

import os

import sys

from controller import Robot

# Path operations to correctly locate the managers.

if sys.version\_info.major > 2:

sys.exit("RobotisOpManager library is available ony for Python 2.7")

libraryPath = os.path.join(os.environ.get("WEBOTS\_HOME"), 'projects', 'robots', 'robotis', 'darwin-op', 'libraries', 'python')

libraryPath = libraryPath.replace('/', os.sep)

sys.path.append(libraryPath)

from managers import RobotisOp2GaitManager, RobotisOp2MotionManager

# Names of position sensors needed to get the corresponding device and read the

# measurements.

positionSensorNames = ('ShoulderR', 'ShoulderL', 'ArmUpperR', 'ArmUpperL',

'ArmLowerR', 'ArmLowerL', 'PelvYR', 'PelvYL',

'PelvR', 'PelvL', 'LegUpperR', 'LegUpperL',

'LegLowerR', 'LegLowerL', 'AnkleR', 'AnkleL',

'FootR', 'FootL', 'Neck', 'Head')

# List of position sensor devices.

positionSensors = []

# Create the Robot instance.

robot = Robot()

basicTimeStep = int(robot.getBasicTimeStep())

# Initialize motion manager.

motionManager = RobotisOp2MotionManager(robot)

# Get the time step of the current world.

timestep = int(robot.getBasicTimeStep())

gyro = robot.getGyro('Gyro')

# Enable all the position sensors and populate the 'positionSensor' list.

for i in range(0, len(positionSensorNames)):

positionSensors.append(robot.getPositionSensor(positionSensorNames[i] + 'S'))

positionSensors[i].enable(basicTimeStep)

gyro.enable(basicTimeStep)

# Perform one simulation step to get sensors working properly.

robot.step(timestep)

# Page 9: assume walking position.

motionManager.playPage(9, True)

# Initialize OP2 gait manager.

gaitManager = None

gaitManager = RobotisOp2GaitManager(robot, "")

gaitManager.start()

gaitManager.setXAmplitude(0.0)

gaitManager.setYAmplitude(0.0)

gaitManager.setBalanceEnable(False)

gaitManager.setMoveAimOn(False)

gaitAmplitude = 1

looptimes = 0

gyroValue = 0.0

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

gyroValues = gyro.getValues()

if looptimes == 0:

gaitManager.setXAmplitude(gaitAmplitude)

gyroValue = gyroValues[0]

if gyroValues[0] > gyroValue:

gaitManager.setAAmplitude(0.06)

if gyroValues[0] < gyroValue:

gaitManager.setAAmplitude(-0.06)

gaitManager.step(basicTimeStep)

looptimes += 1