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»

ОТЧЁТ

По лабораторной работе №3

по дисциплине «Аппаратное и программное обеспечение   
роботизированных систем»

«Программное обеспечение роботизированных систем»

ПРОВЕРИЛ:

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

ВЫПОЛНИЛИ СТУДЕНТЫ

ГРУППЫ 17-В-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Поляков А.А.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Кузьмин М.В.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Максимова О.В.

Нижний Новгород

2020

**Задача:** На сцене находится несколько роботов, в начальном положении они стоят друг за другом, первый робот начинает движение, два других анализируя его положение относительно себя также начинают движение ровно по его траектории. Роботы должны уметь обходить препятствия.

**Ход работы:**

**1. Создание виртуального пространства.**

Для выполнения данной лабораторное работы было создана сцена, ограниченная стенами размеров 5\*5 метров (Рисунок 1).

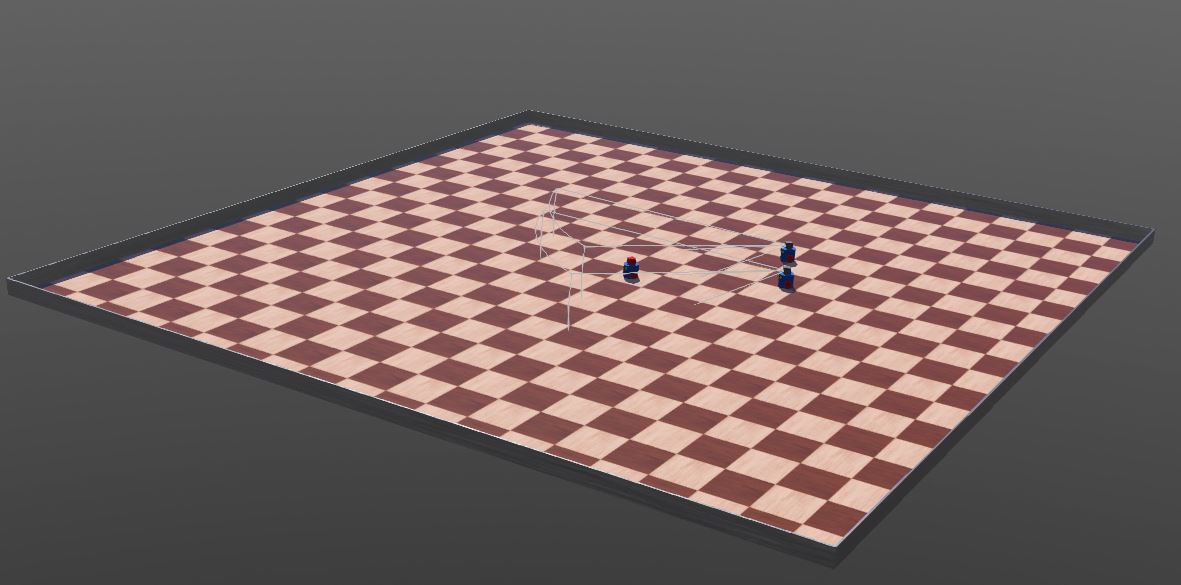


Рис. 1. Сцена 5\*5 метров.

Так же, на сцене были созданы препятствия (Рисунок 2).



Рис. 2. Созданные препятствия.

Для выполнения задания так же были подготовлены роботы, взятые из "мира" - **radar.wbt**, 1 из которых ведущий (Рисунок 3), который работает на 1 контролере, и 2 робота (Рисунок 4), работающих на отдельном контроллере.

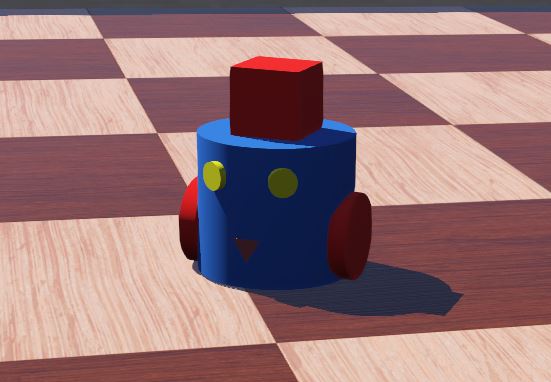


Рис. 3. Ведущий робот.

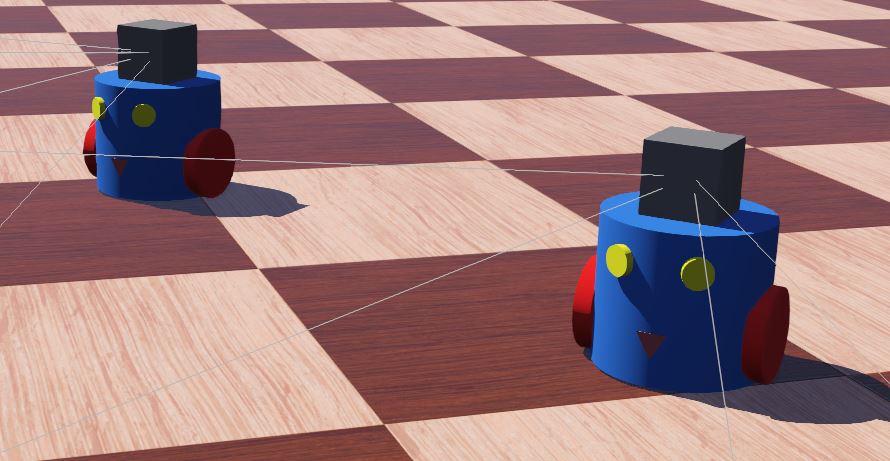


Рис. 4. Ведомые роботы.

**2. Описание алгоритма работы контроллеров.**

**2.1) Контроллер ведущего робота.**

Движение ведущего робота контролируется с помощью двух дистанционных сенсоров, и в зависимости от поступаемых с них данных, робот с помощью контроллера выбирает направление обхода препятствия.

Данные записываются в переменные ds\_l, ds\_r, соответственно, так же для контроля выхода из острых углов, была введена переменная k=0.

"Ядро" контроллера - это итерационный цикл **while**, в нем сначала проверяется переменная ***k*** на равенство 0. Переменная ***k*** принимает значение отличное от 0, только в случае если и с правого, и с левого дистанционного сенсора приходят данные о расстоянии меньше 30. В этом случае ***k*** приравнивается к 15. Это количество итераций достаточное для того чтобы робот повернул на угол при котором один из датчиков перестает видеть препятствие и робот начал движение по другому условию, оно было получено опытным путем. Стоит заметить, что k может быть больше 15, но не меньше, т.е. в случае когда число итераций меньше, роботу не будет хватать их, что бы выйти из острого угла. А если число превышает 15, то итерации прекращаются автоматически, так как один из сенсоров прекращает видеть препятствие. Робот начинает разворот в левую сторону, и теперь в начале цикла **while**, переменная ***k*** будет уменьшаться на 1, а цикл начинаться заново, до тех пор пока ***k***  не станет снова равна 0.  
Если же полученный данные, больше 50 только на правом или левом сенсоре, робот начинает поворот в сторону, противоположную от сенсора, на котором данные больше 50.

Поворот происходит с помощью изменения направления вращения одного из колес.

**2.2) Контроллер ведомых роботов.**

Движение же роботов следующих за ведущим, контролируется с помощью установленного на них радара, который определяет расстояние до, идущих перед ним, роботов, а также азимут между центром сектора обзора радара и наблюдаемыми роботами.

Так как данные роботы следуют за другим, в случае если ведущему потребуется поворот или разворот, они должны уметь уменьшать скорость в зависимости от ситуации. Для это была введена переменная ***Flag***. Она принимает значения False или True (изначально False).

Так же как и на предыдущем контроллере, в цикле **while** сначала проверяется переменная ***Flag***. Она имеет значение True только в том случае если с радара поступила информация о том, что расстояние до ближайшего робота меньше 0.3. И тогда коэффициент, на который умножается заданная скорость, будет 0.5, то есть робот замедлится.

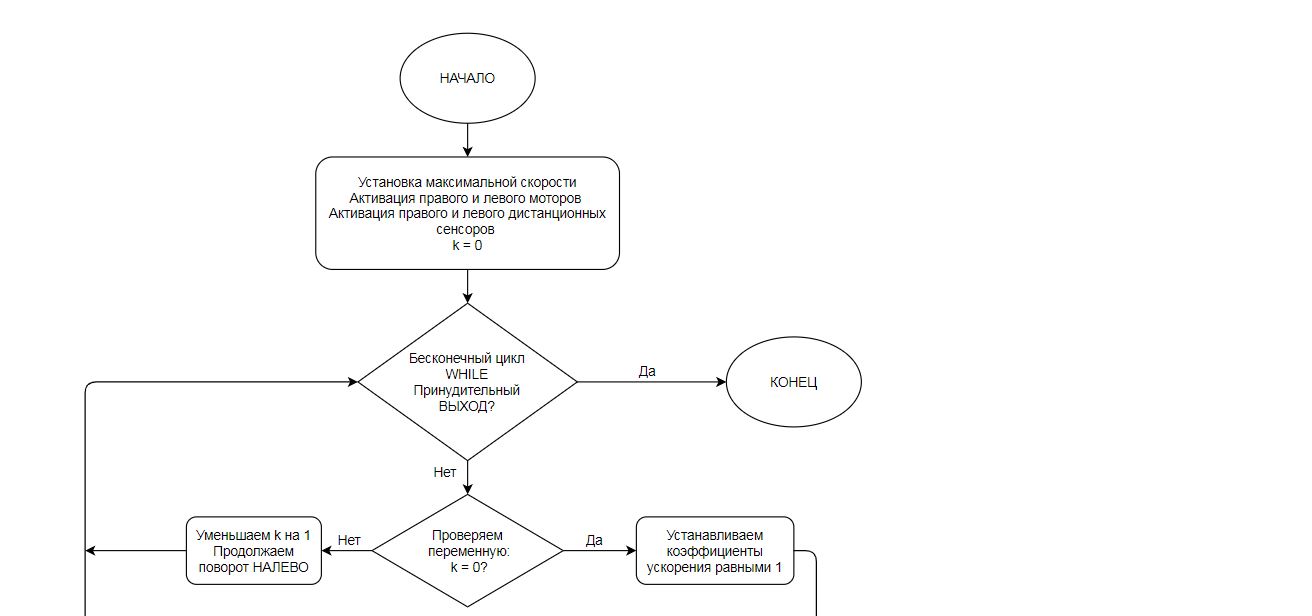
Предполагалось, что радар может видеть больше одного робота. И для того чтобы робот следовал за идущим перед ним и не путался в движениях, была введена переменная ***d\_min***. С помощью неё на каждой итерации цикла **while** определяется, какой из видимых радаром роботов ближайший.

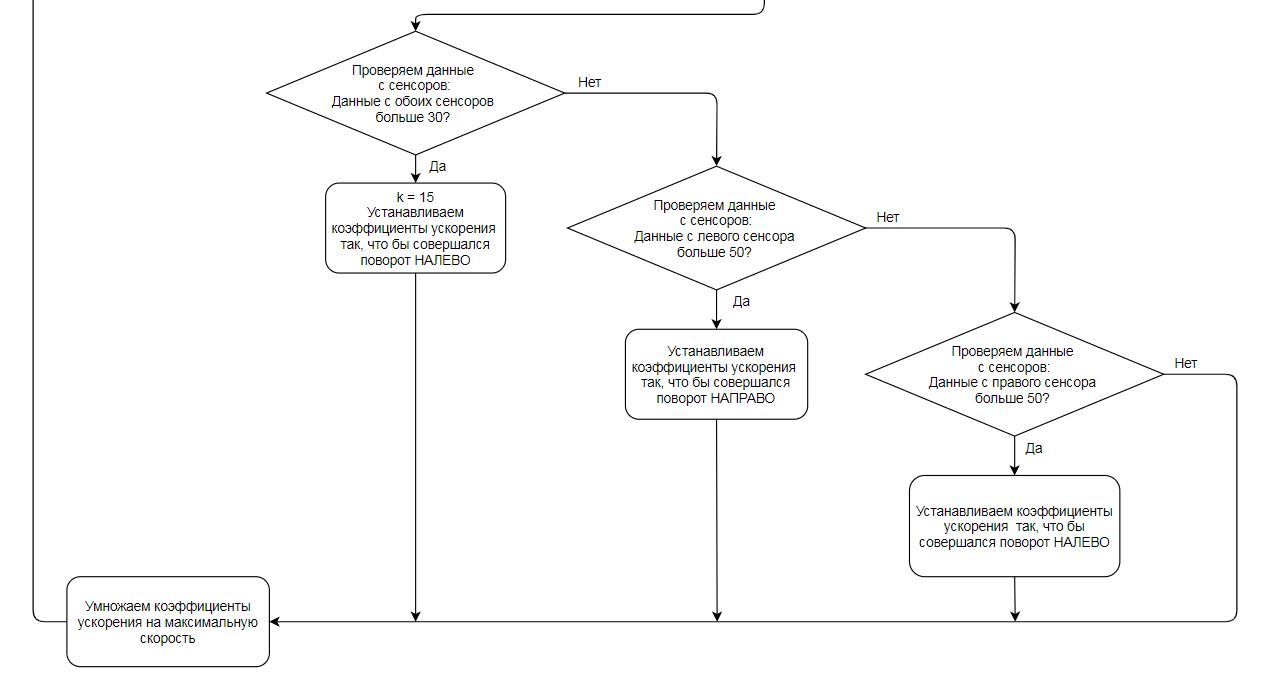
Так же, данные с радара, расстояние и азимут, записываются в соответствующие списки на каждой итерации.

После проверки на расстояние, нужно определить необходимость в повороте. Для этого из списка берется модуль верхнего азимута, и сравнивается с допустимым значением (0.15). Если значение больше то далее выбирается направление. Переменная ***p***  приравнивается к "1" или "-1" в зависимости от того, меньше или больше азимут. И далее ***p*** перемножается с коэффициентом ***speed\_l*** и ***speed\_r*** , для правого и левого колеса соответственно, которые в свою очередь умножаются на заданную в каждом из контроллеров скорость **MAX\_SPEED.**

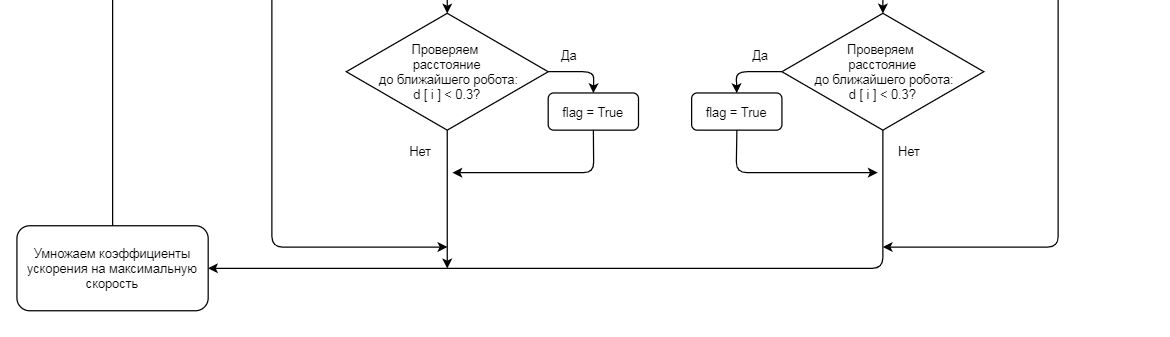
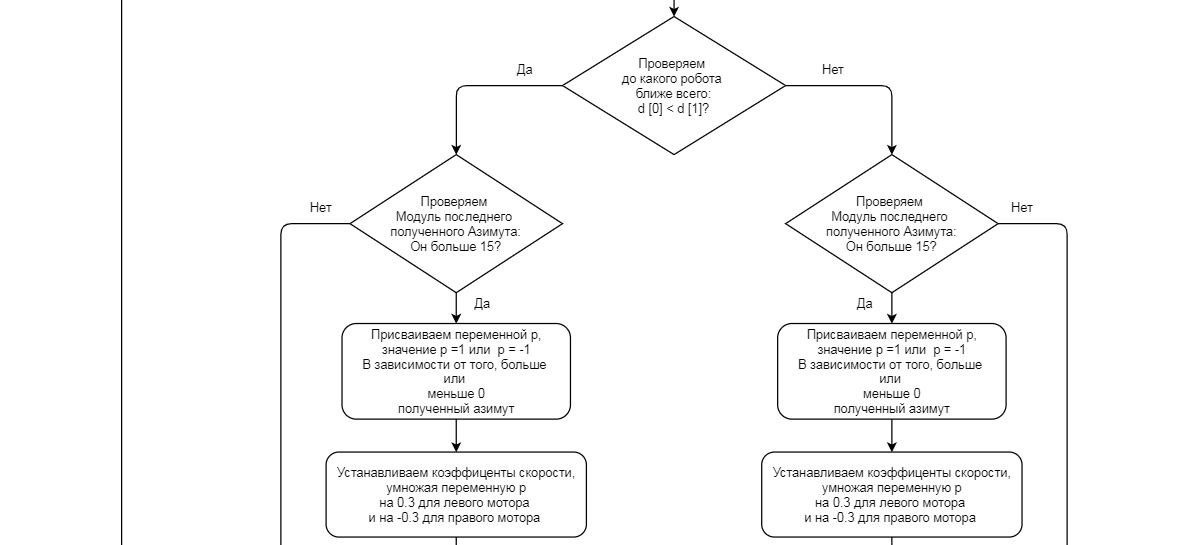
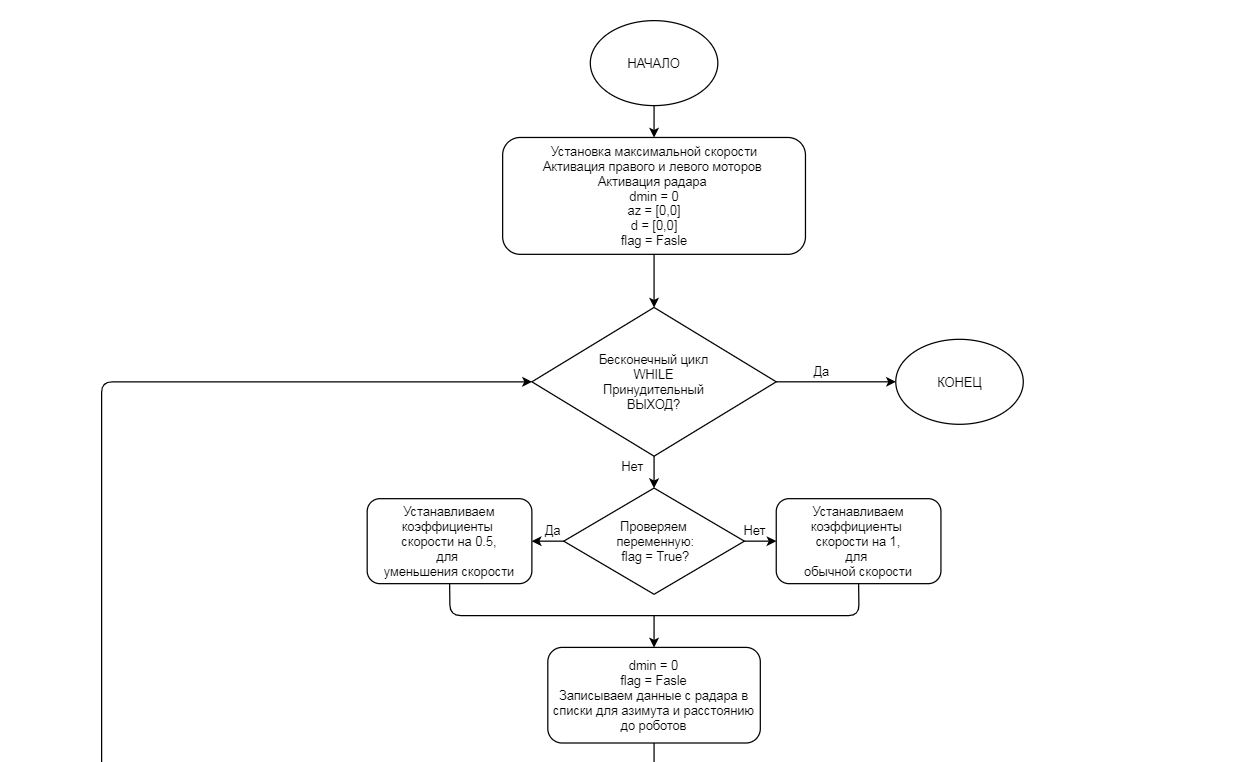
**3. Блок-схема контроллеров.**

Блок-схема ведущего робота:



****

Блок-схема ведомых роботов:

****

**4. Код контроллеров.**

Контроллер ведущего робота:

"""my\_controller1 controller."""

from controller import Robot

MAX\_SPEED = 4.9 # установка максимальной скорости робота

robot = Robot()

timestep = int(robot.getBasicTimeStep())

left\_motor = robot.getMotor('left wheel motor') # активация правого и левого моторов

right\_motor = robot.getMotor('right wheel motor')

left\_motor.setPosition(float('inf'))

right\_motor.setPosition(float('inf'))

left\_motor.setVelocity(0.0)

right\_motor.setVelocity(0.0)

ds\_l = robot.getDistanceSensor('ds0') # данные с левого сенсора в переменную ds\_l

ds\_l.enable(timestep)

ds\_r = robot.getDistanceSensor('ds1') # данные с правого сенсора в переменную ds\_r

ds\_r.enable(timestep)

k = 0

while robot.step(timestep) != -1: # бесконечный цикл, выход только принудительный

    l = ds\_l.getValue() # данные полученные с сенсоров заносим в переменный, для выбора # направления поворота

    r = ds\_r.getValue()

    if k != 0: # проверка переменной, с помощью которой в несколько итераций # осуществляется выход их острого угла

        k -= 1

        continue # прерывание цикла для осуществления выхода из острого угла

    speed\_l = 1 # установка начальных коэффициентов скорости

    speed\_r = 1

    if l > 30 and r > 30: # случай попадания робота в острый угол

        k = 15

        speed\_l = -0.5 # установка соответствующих коэффициентов скорости

        speed\_r = 0.5

    elif l > 50: # случай препятствия СЛЕВА

        speed\_l = 0.4 # установка соответствующих коэффициентов скорости

        speed\_r = -0.4

    elif r > 50: # случай препятствия СПРАВА

        speed\_l = -0.4 # установка соответствующих коэффициентов скорости

        speed\_r = 0.4

    left\_motor.setVelocity(speed\_l\*MAX\_SPEED) # установка скорости, путем умножения коэффициента на # максимальную скорость

    right\_motor.setVelocity(speed\_r\*MAX\_SPEED)

Контроллер ведомых роботов:

"""my\_controller2 controller."""

from controller import Robot

from controller import Radar

MAX\_SPEED = 5 # установка максимальной скорости робота

robot = Robot()

timestep = int(robot.getBasicTimeStep())

left\_motor = robot.getMotor('left wheel motor') # активация правого и левого моторов

right\_motor = robot.getMotor('right wheel motor')

left\_motor.setPosition(float('inf'))

right\_motor.setPosition(float('inf'))

left\_motor.setVelocity(0.0)

right\_motor.setVelocity(0.0)

radar = robot.getRadar('radar') # активация радара

radar.enable(64)

dmin = 0 # установка начальных значений переменных

az = [0,0]

d = [0,0]

k = 0

flag = False

while robot.step(timestep) != -1: # бесконечный цикл, выход только принудительный

    if flag: # проверка переменной flag, для определения # необходимости торможения

        speed\_l = 0.5

        speed\_r = 0.5

    else:

        speed\_l = 1

        speed\_r = 1

    dmin = 0

    for i in range(radar.getNumberOfTargets()): # получение данных с радара и занесение их в списки

        az[i] = radar.getTargets()[i].azimuth

        d[i] = radar.getTargets()[i].distance

    flag = False

    if radar.getNumberOfTargets() > 1: # определение какому из видимых роботов следовать, # путем определения ближайшего, через расстояние # (полученного с радара)

        if d[0] < d[1]:

            dmin = 0

        else:

            dmin = 1

    if dmin == 0:

        if abs(az[0]) > 0.15: # определение необходимости поворота, через сравнение # Модуля азимута (полученного с радара) с # максимальным допустимы отклонением (0.15)

            p = 1 if az[0] >= 0 else -1 # определение направление поворота, путем сравнения # азимута с 0, и установка переменной p в # соответствующее значение

            speed\_l = 0.3\*p # установка соответствующих коэффициентов скорости

            speed\_r = -0.3\*p

        if d[0] < 0.3: # проверка на необходимость торможения, путем # сравнения расстояния с минимально допустимым # приближением к ближайшему роботу, если расстояние #слишком мало то устанавливаем переменную flag, в # значение True

            flag = True

    if dmin == 1:

        if abs(az[1]) > 0.15:

            p = 1 if az[1] >= 0 else -1

            speed\_l = 0.3\*p

            speed\_r = -0.3\*p

        if d[1] < 0.3:

            flag = True

    left\_motor.setVelocity(speed\_l\*MAX\_SPEED) # установка скорости, путем умножения коэффициента на # максимальную скорость

    right\_motor.setVelocity(speed\_r\*MAX\_SPEED)

**Вывод:** в ходе выполнения данной лабораторной работы были освоены навыки работы с радаром. А также методы контроля движения робота в зависимости от появления на его пути препятствий, и в случае, когда ему необходимо следовать и повторять движения другого объекта.