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»

ЛАБОРАТОРНАЯ РАБОТА №3

по дисциплине

Аппаратное и программное обеспечение роботизированных систем

РУКОВОДИТЕЛЬ:

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

(подпись) (фамилия, и.,о.)

СТУДЕНТ:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сипаков В.В.

(подпись) Марчус К.Р.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(шифр группы)

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

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

Нижний Новгород 2022

# Цель работы

Получение навыков работы с алгоритмами управления роботами

# Задание

3) обход лабиринта: робот должен найти выход из лабиринта;

Робот: e-puck (https://cyberbotics.com/doc/guide/epuck);

Структура сцены: размер лабиринта 3\*3 метра, расстояние между стенами лабиринта - 0.15 метра, в лабиринте на пути следования робота должны быть препятствия, которые робот должен успешно обходить; для обнаружения препятствий и ориентирования в лабиринте нужно использовать датчики расстояний

# Алгоритм

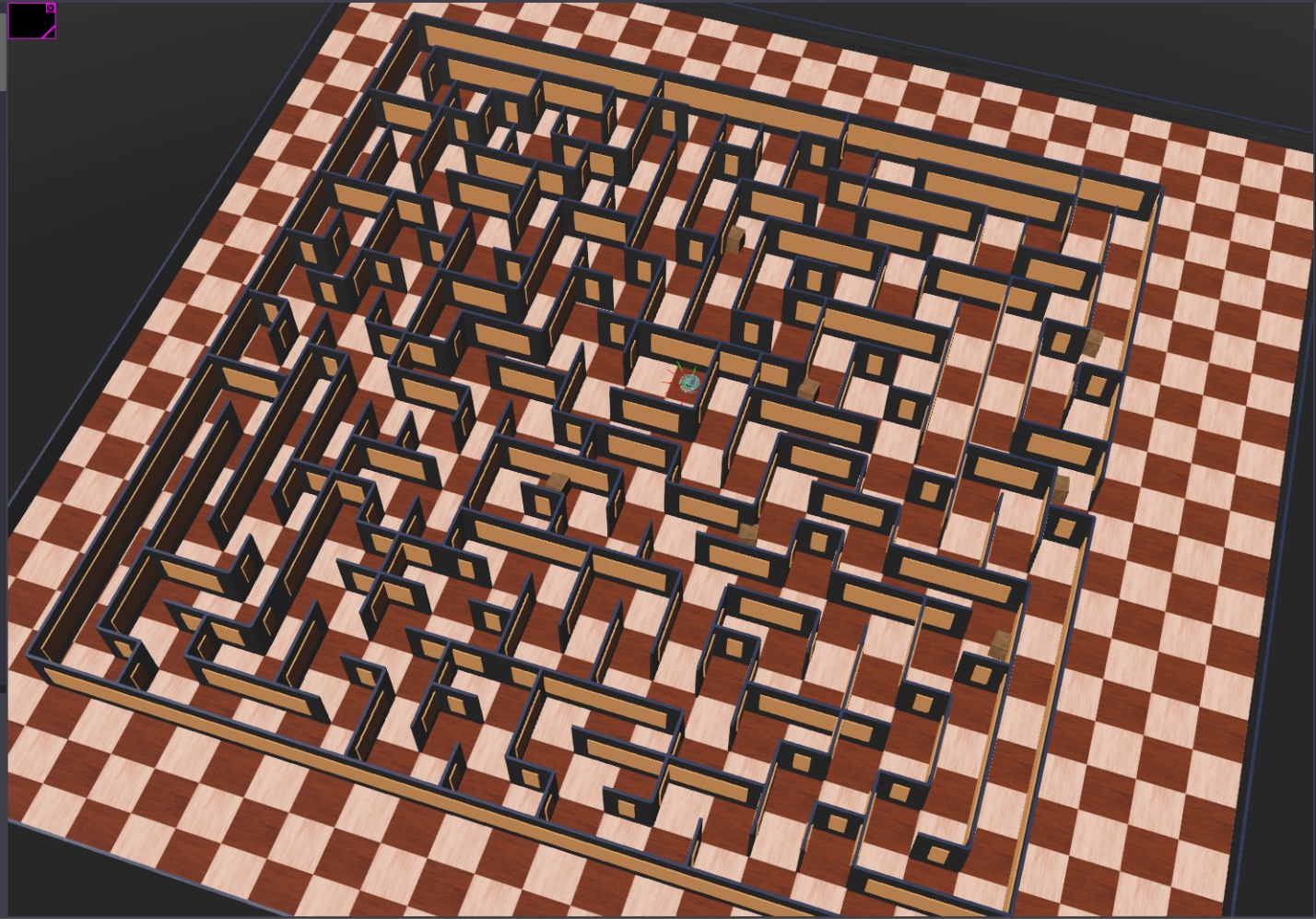
Используется самый простой алгоритм прохождения лабиринта по одной стене (в данном случае выбрана правая стена)

Алгоритм заключается в определении трех флагов: стена спереди, стена справа и стена по диагонали.

* Если обнаружена стена спереди, то нужно повернуть налево.
* Если стены спереди нет, но есть стена справа: продолжать движение.
* Если стены ни спереди, ни справа нет, нужно повернуть направо и найти стену.
* При этом если обнаружена стена по диагонали (т.е. показания углового сенсора превысили определенный порог), значит робот находится слишком близко к стене, и нужно повернуть налево, чтобы от нее отъехать.

# Результат работы

Построенный односвязный лабиринт с расставленными внутри препятствиями:



**maze\_solver.py:**

*"""maze\_solver controller."""*sensors\_dict = {  
 0: 'front right',  
 1: 'mid right',  
 2: 'outer right',  
 3: 'rear right',  
 4: 'rear left',  
 5: 'outer left',  
 6: 'mid left',  
 7: 'front left',  
}  
  
from controller import Robot  
  
robot = Robot()  
  
timestep = int(robot.getBasicTimeStep())  
  
maxMotorVelocity = 2.5  
distanceSensorCalibrationConstant = 100  
  
leftMotor = robot.getDevice('left wheel motor')  
rightMotor = robot.getDevice('right wheel motor')  
  
sensors = [robot.getDevice('ps0'), robot.getDevice('ps1'),  
 robot.getDevice('ps2'), robot.getDevice('ps3'),  
 robot.getDevice('ps4'), robot.getDevice('ps5'),  
 robot.getDevice('ps6'), robot.getDevice('ps7')]  
  
initialVelocity = maxMotorVelocity  
  
for sensor in sensors:  
 sensor.enable(timestep)  
  
leftMotor.setPosition(float('inf'))  
rightMotor.setPosition(float('inf'))  
  
obstacleFlag = False  
i = 0  
  
stickFlag = False  
  
while robot.step(timestep) != -1:  
  
 sensorData = []  
  
 for sensor in sensors:  
 sensorData.append(round(sensor.getValue(), 4) / distanceSensorCalibrationConstant)  
  
 print(sensorData)  
  
 frontRightSD = sensorData[0]  
 midRightSD = sensorData[1]  
 outerRightSD = sensorData[2]  
 outerLeftSD = sensorData[5]  
 midLeftSD = sensorData[6]  
 frontLeftSD = sensorData[7]  
  
 rightWall = outerRightSD > 2.5  
 frontWall = frontRightSD > 2.5  
 midRightWall = midRightSD > 3  
 print(frontWall, rightWall)  
  
 if stickFlag:  
 if frontWall:  
 rightMotor.setVelocity(initialVelocity)  
 leftMotor.setVelocity(-initialVelocity)  
 print('turning left (wall)')  
 else:  
 if rightWall:  
 rightMotor.setVelocity(initialVelocity)  
 leftMotor.setVelocity(initialVelocity)  
 print('going forward')  
 else:  
 rightMotor.setVelocity(initialVelocity / 20)  
 leftMotor.setVelocity(initialVelocity)  
 print('turning right (stick to the wall)')  
  
 if midRightWall:  
 rightMotor.setVelocity(initialVelocity)  
 leftMotor.setVelocity(initialVelocity / 20)  
 print('turning left (too close to the wall)')  
  
 else:  
 rightMotor.setVelocity(initialVelocity)  
 leftMotor.setVelocity(initialVelocity)  
 if frontWall or rightWall:  
 stickFlag = True  
  
 pass