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

Программирование алгоритмов управления роботом в Webots

по дисциплине

Аппаратное и программное обеспечение роботизированных систем

РУКОВОДИТЕЛЬ:

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

(подпись)

СТУДЕНТ:

группы 19-В-1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Жеглов Г. В.

(подпись)

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

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

Нижний Новгород 2022

**Задание**

Движение по шоссе. Обгон автомобилей по шоссе.

Этот тест направлен на то, чтобы проехать как можно дальше за одну минуту, не сталкиваясь с другими транспортными средствами или статическими объектами.

Метрика, используемая для оценки контроллера транспортного средства, - это расстояние, пройденное по дороге за одну минуту.

Если транспортное средство касается каких-либо других транспортных средств или статических объектов или выезжает на аварийную полосу, моделирование останавливается (даже до одной минуты) и записывается уже пройденное расстояние.

Код контроллера

from vehicle import Driver

# name of the available distance sensors

sensorsNames = [

'front',

'front right 0',

'front right 1',

'front right 2',

'front left 0',

'front left 1',

'front left 2',

'rear',

'rear left',

'rear right',

'right',

'left']

sensors = {}

maxSpeed = 80

driver = Driver()

driver.setSteeringAngle(0.0) # go straight

# get and enable the distance sensors

for name in sensorsNames:

sensors[name] = driver.getDistanceSensor('distance sensor ' + name)

sensors[name].enable(10)

# get and enable the GPS

gps = driver.getGPS('gps')

gps.enable(10)

# get the camera

#camera = driver.getCamera('camera')

# uncomment those lines to enable the camera

# camera.enable(50)

# camera.recognitionEnable(50)

while driver.step() != -1:

# adjust the speed according to the value returned by the front distance sensor

frontDistance = sensors['front'].getValue()

frontRange = sensors['front'].getMaxValue()

rleftDistance = sensors['rear left'].getValue()

rleftRange = sensors['rear left'].getMaxValue()

leftDistance = sensors['left'].getValue()

leftRange = sensors['left'].getMaxValue()

rightDistance = sensors['right'].getValue()

rightRange = sensors['right'].getMaxValue()

fleft2Distance = sensors['front left 2'].getValue()

fleft2Range = sensors['front left 2'].getMaxValue()

fright0Distance = sensors['front right 0'].getValue()

fright0Range = sensors['front right 0'].getMaxValue()

fright1Distance = sensors['front right 1'].getValue()

fright1Range = sensors['front right 1'].getMaxValue()

fright2Distance = sensors['front right 2'].getValue()

fright2Range = sensors['front right 2'].getMaxValue()

fleft0Distance = sensors['front left 0'].getValue()

fleft0Range = sensors['front left 0'].getMaxValue()

fleft1Distance = sensors['front left 1'].getValue()

fleft1Range = sensors['front left 1'].getMaxValue()

r = rightDistance / rightRange

rl = rleftDistance / rleftRange

l = leftDistance / leftRange

fl2 = fleft2Distance / fleft2Range

fl0 = fleft0Distance / fleft0Range

fl1 = fleft1Distance / fleft1Range

fr0 = fright0Distance / fright0Range

fr1 = fright1Distance / fright1Range

fr2 = fright2Distance / fright2Range

fr = frontDistance / frontRange

speed = maxSpeed \* frontDistance / frontRange

driver.setCruisingSpeed(speed)

#print(rl,'|',l,'|',fl2)

# brake if we need to reduce the speed

speedDiff = driver.getCurrentSpeed() - speed

if speedDiff > 0:

driver.setBrakeIntensity(min(speedDiff / speed, 1))

else:

driver.setBrakeIntensity(0)

if((l>0.6)and(fl2>0.6)):

if(r>0.5):

angle1 = 0.02\*(-1+(1-fl2)+(1-fl0)+(1-fl1)-(1-fr)\*4-(1-fr1)-(1-fr0)-(1-fr2)-(1-r))

driver.setSteeringAngle(angle1)

else:

angle1 = 0.03\*(-1+(1-fl2)+(1-fl0)+(1-fl1)-(1-fr)\*4-(1-fr1)-(1-fr0)\*5-(1-fr2)-(1-r))

driver.setSteeringAngle(angle1)

else:

if((l>0.2)or(fl2>0.4)):

angle1 = 0.025\*((1-l)+(1-fl2)+(1-fl0)+(1-fl1)+(1-fr)\*4-(1-fr2)\*4-(1-fr1)\*4-(1-fr0)\*4-(1-r)\*2-rl\*0.25)

driver.setSteeringAngle(angle1)

elif((l<0.2)or(fl2<0.4)):

angle = 0.025\*((1-l)+(1-fl2)+(1-fl1)+(1-fl0)-(1-fr2)-(1-fr1)-(1-fr0)+(1-fr)-(1-r)+(1-rl))

if(angle<0.0):

driver.setSteeringAngle(0.01)

else:

driver.setSteeringAngle(angle)

Результат выполнения

