Государственное бюджетное образовательное учреждение г. Москвы

Школа №354 им. Д. М. Карбышева

Проект “Smart City Parking”

Авторы: Воля Максим

Девятов Леонид

Рынин Дмитрий

Руководитель: Богачёва Татьяна Петровна

Москва 2021

Цели:

Наша цель разработать умную парковку с зарядными станциями для электромобилей, чтобы сделать использование электромобилей более простым и удобным, тем самым увеличить спрос на них, уменьшить количество машин с бензиновыми двигателями и следовательно углеродные выбросы в атмосферу.

Задачи:

1) Узнать информацию о зарядных станциях для электромобилей и о самих автомобилях
2) Разработать вакуумную присоску
3) Разработать поворотный стол
4) Создать робота «паковщика»
5) Создать робота «перевозчика»
6) Отладить связь блоков EV3 и Raspberry по Wi-Fi (отправка сообщений)
7) Отладить связь Raspberry и Arduino по USB
8) Освоить вёрстку и программирование сайтов
9) Создать http сервер для обработки “get” запросов
10) Написать код для “парковщика”
11) Написать код для “перевозчика”
12) Написать код для поворотного стола
13) Написать код для Raspberry
14) Отладить совместную работу всех частей

Актуальность

Стало известно, что зарядные станции могут появиться возле всех новых торговых центров в столице. Московские власти и Министерство энергетики готовят поправки в градостроительный кодекс, согласно которым ни один новый торговый центр не может быть сдан в эксплуатацию без оборудования для зарядки аккумуляторов «зеленых» автомобилей.

Количество зарядных станцией зависит от региона и страны, например Китае активно строят инфраструктуру – сейчас в стране более 270 тыс. зарядных станций, а для сравнения, в России их чуть больше 350. Тем не менее, доля электромобилей от общего количества машинного парка в мире составляет ничтожные 0,5%. Количество электромобилей стремительно растёт, в то же время зарядные станции растут гораздо медленнее и на одну зарядную станцию в России приходится 30–40 электромобилей, в то же время в среднем по миру данный показатель - 10. На данный момент в России

функционирует около 200 коммерческих электрозаправочных станций. Эксперты, опрошенные ТАСС, одной из основных проблем развития рынка электромобилей назвали сложности с зарядной инфраструктурой в ряде регионов.

А сложности с зарядкой вынуждают владельцев электрокаров обратно пересаживаться на машины с бензиновыми двигателями, хотя эксплуатация неэкологичного авто выходит дороже и вредит экологии.

Мы решили это изменить и сделать использование электромобилей удобным и практичным.

Исследования:

Электрокары в настоящее время всё же распространены пока ещё не так, как транспортные средства с двигателями, работающими на бензине или дизельном топливе, поэтому производители продолжают работать над совершенствованием технологии зарядки.

Зарядка электромобиля возможна одним из четырех способов:
1. С помощью обыкновенной розетки с напряжением 220 В. Правда, данный вариант используется всё реже ввиду своей ненадежности.
2. От бытовой электросети, через которую проходит переменный ток. Этот способ более предпочтителен, чем предыдущий, поскольку кабель, покупаемый вместе с машиной, имеет внутри специальную защиту.
3. Трехфазная зарядка, являющаяся самой безопасной. Её основное преимущество – возможность полного контроля над процессом.
4. Быстрая зарядка электрокара.

Количество электрозаправок в мире:

Инфраструктура для электромобилей продолжает стремительно развиваться. Большая часть новой инфраструктуры построена в Китае и Европе. С немалым отставанием на третьем месте по количеству электрозаправок расположились США.

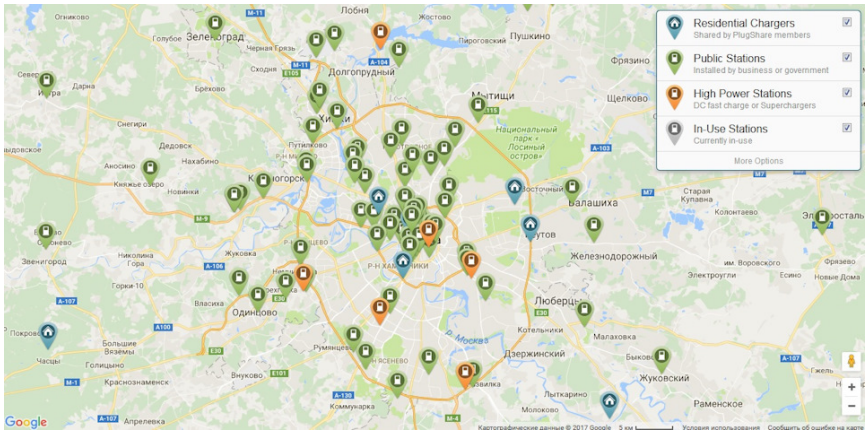
Наиболее агрессивную политику в отношении развития электромобилей проводит Китай. На него приходится свыше половины всех мировых зарядок. В Европе лидирует Франция.

Количество электрозаправок в России:

В Москве больше находится половины станций “заправки” – около 100.

Санкт-Петербург на втором месте по количеству электрозаправок. Здесь их около 30.

По 2 электрозаправки в Екатеринбурге, Самаре, Краснодаре, Перми и Уфе.



Слабые места электромобилей

В итоге время зарядки электромобиля — одно из главных слабых мест электромобилей по сравнению с бензиновыми автомобилями

Для каждой аккумуляторной батареи обозначена максимальная сила зарядного тока. Если превысить ее, то это может сказаться не только на потере емкости батареи, но и даже привести к полному выходу ее из строя.

Ультрабыстрые зарядные станции

Ультрабыстрые зарядные станции на сегодняшний день — самый быстрый способ зарядить электромобиль. Их можно встретить на автомагистралях или крупных публичных парковках. Такие станции обеспечивают постоянный или переменный ток большой мощности и могут зарядить автомобиль до 80% за 20–40 минут. В большинстве случаев ультрабыстрые станции отключаются, когда аккумулятор электромобиля заряжен примерно на 80%, чтобы защитить батарею и продлить срок её службы.

Ультрабыстрая зарядка может использоваться только на тех автомобилях, где возможность её применения предусмотрена изначально и присутствует специализированный тип зарядного разъёма.

Информация о разъёмах:

Во многих электромобилях под заправочным лючком можно обнаружить два разъема: один – для зарядки переменным током, второй – для подключения к постоянному току и быстрой зарядки.

Встречаются и машины на электротяге с одним зарядным портом. В этом случае порт либо совместим с переменным и постоянным током, либо используется для зарядки только переменным током, как правило медленной, хотя есть и исключения.

Единого разъема для зарядки электромобилей не существует до сих пор. Более того, виды разъемов зависят от страны или региона: Северная Америка, Европа, Китай и США — у всех разные.

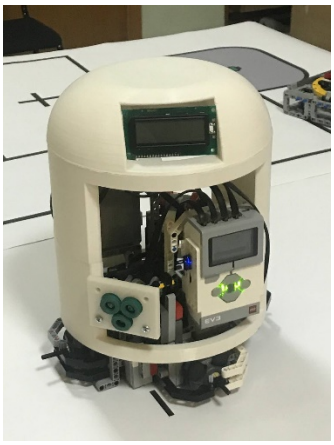
Type 1 J1772 – американский пятиконтактный разъем, разработанный еще в 2009 году и встречающийся практически на всех электромобилях, ввезенных в нашу страну из США. Рассчитан на напряжение 230 В и ток в 32	
Type 2 (Mennekes) Максимальная мощность для однофазной сети – 7,4 кВт, для трехфазной – 43 кВт. Однако чаще мощность при трехфазном подключении ограничена 22 кВт.	
CHAdeMO Рассчитан на постоянный ток в 125 А при напряжении 500 В и соответственно 62,5 кВт мощности.	
CCS Combo – еще один распространенный тип разъемов, используемый с 2012 года. Он может подключаться как к переменному, так и к постоянному току, что позволяет использовать его как с медленными, так и с быстрыми зарядками. При подключении к сети переменного тока происходит его выпрямление в постоянный.	

GB/T – Стандартом предусматриваются два типа разъемов: для медленной зарядки переменным током и для быстрой зарядки постоянным током.	
Tesla Supercharger – разъем, используемый в электромобилях марки Tesla. Максимальная зарядная мощность достигает 200-250 кВт при постоянном токе.	

Самые распространённые виды разъёмов:

Описание:

Робот «Заправщик»:



- 2 блокаev3,
- raspberry pi 3,
- Arduino uno,
- 3 мотора для откачки воздуха,
- ЖК дисплей,
- аккумулятор.

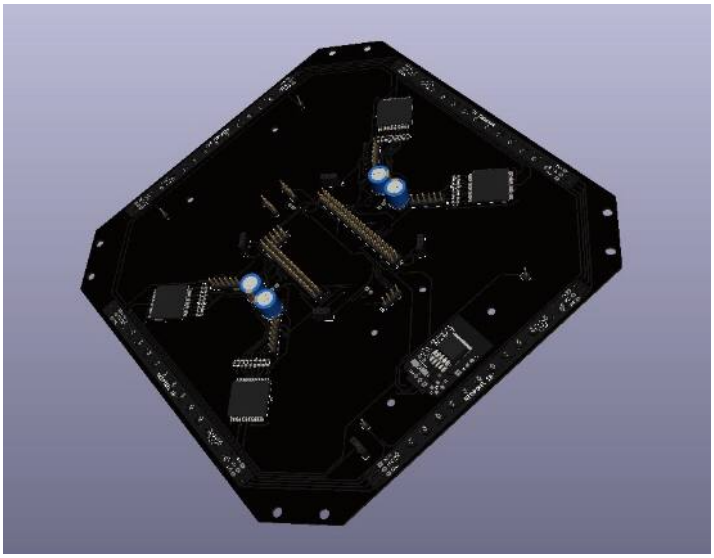
Робот “Заправщик” с помощью камеры определяется местоположение крышки заправочного люка конкретно данного автомобиля, после чего посылаются координаты позиционирования манипулятору, который с помощью вакуумной присоски бережно открывает крышку люка. Затем с помощью камеры определяется тип разъема и подбирается к нему соответствующий штекер.

Заправщик присоединяет нужный штекер к разъему автомобиля и начинается зарядка

Робот «Парковщик»:

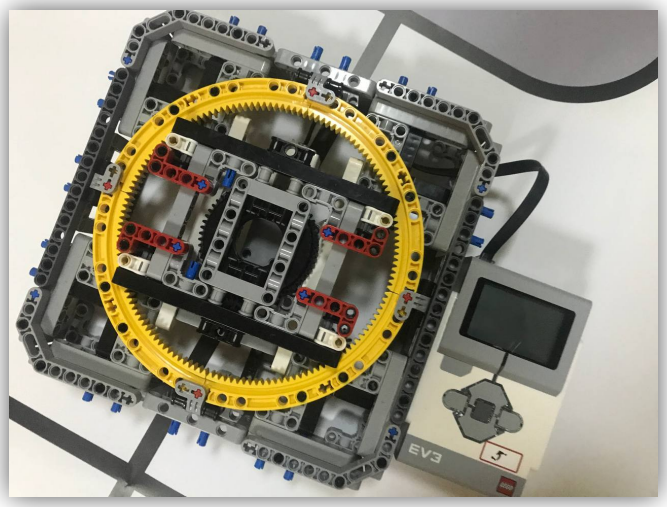
Робот парковщик, состоит из:

4 моторов на движение, что позволяет ему двигаться в разных направлениях за счет всенаправленных колес, моторы управляются Arduino Mega через специальные драйверы. Робот поднимет автомобиль с помощью 4 сервоприводов что позволяет ему поднять без перекосов. Для ориентации робот имеет гироскоп, энкодоры и датчики освещённости. Робот парковщик подает автомобиль для зарядки на специальную вращающуюся платформу.



Поворотный стол:

Поворотный стол состоит из деталей Lego, в том числе среднего мотора, используется для поворота платформы



“**Raspberry**”

Raspberry с подключённой к ней USB камерой, используется для считывания QR-кода и определения разъёма, установленного в автомобиль, обработки запросов, пришедших на сервер, созданный на Python и находящийся так же на ней

Сайт

В ходе работы над проектом также был создан сайт, на котором пользователь регистрируется, вводя помимо стандартной информации и информацию об автомобиле. Оплата зарядки, указание номера станции и парковочное место автомобиля происходят на нём. ОН сделан на: html, CSS, java script

Алгоритм работы

1) Ожидание информации от сайта(статус зарядки, модель автомобиля и т.д.)
2) Отправка сообщения с raspberry на робота “парковщика”
3) Захват и перевозка автомобиля, установка на поворотный стол, отъезд на своё место “парковщика”
4) Сообщение от заправщика к поворотному столу => его поворот
5) Отправка сообщения от стола к raspberry(стол повернулся)
6) Поиск QR-кода с помощью камеры
7) В случае его отсутствия отправка сообщения столу(“повернись на 180”) и возвращение к 6)
8) Отправка блоку EV3 отвечающему за перемещение робота координат QR-кода
9) При подъезде отправка сообщения на другой блок робота(открытие заправ. люка) и на Arduino uno(открытие люка)
10) Отправка сообщения блоком EV3 об окончание открытия заправочного люка
11) Определение типа разъёма, отправка номера нужного штекера на блок, отвеч. за движение робота
12) Взятие штекера, подъезд к автомобилю и его зарядка
13) Отправка сообщения об окончание зарядки
14) Отправка данных на сайт (заправка окончена)
15) Захват автомобиля и установка его на парковочное место

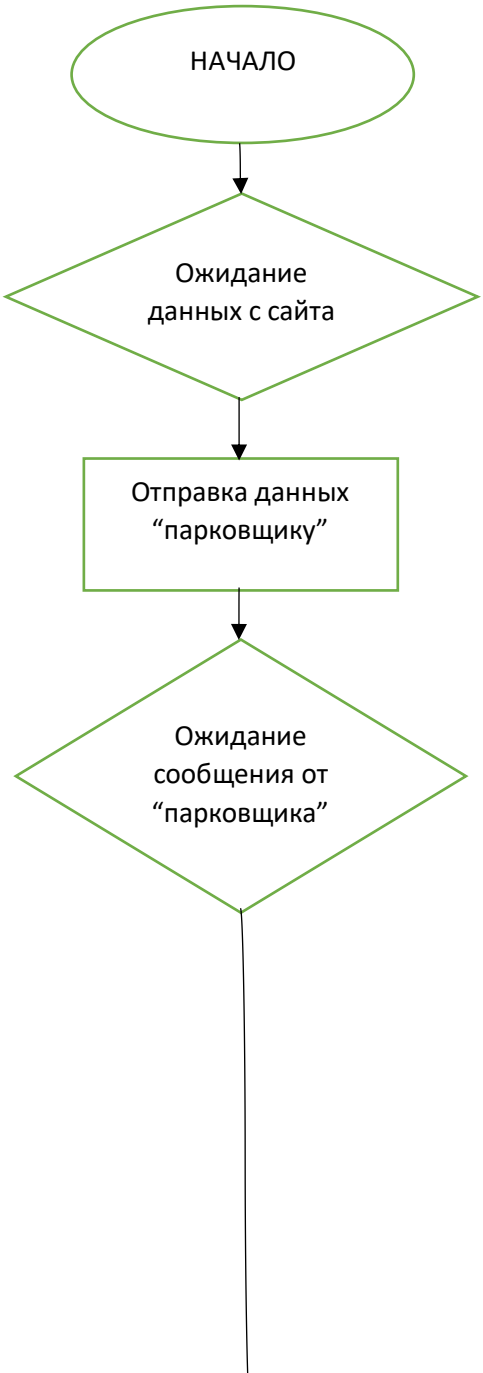
Итоги

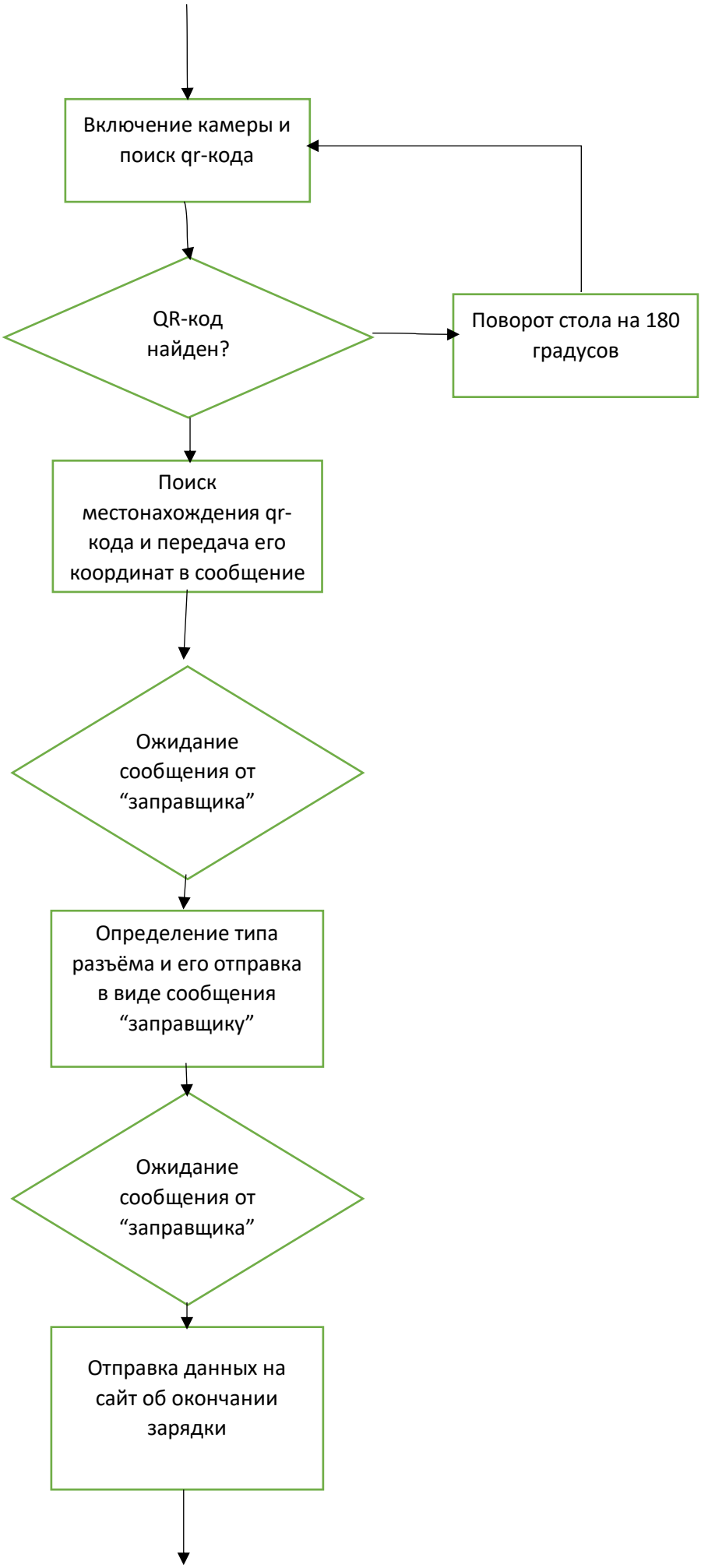
Мы узнали многое об электромобилях и зарядных станциях для них, разработали вакуумную присоску, сконструировали, собрали и запрограммировали оба манипулятора, сделали поворотный стол, написали код для него и для Raspberry(для этого пришлось освоить работу с ним и библиотекой open CV. Также смоделировали и распечатали разъёмы, штекеры и другие части проекта . После этого мы создали макет автомобиля и соединив все – получили макет заправочной станции, которую можно использовать на парковках в торговых центрах или в строящихся парковках в жилых районах.

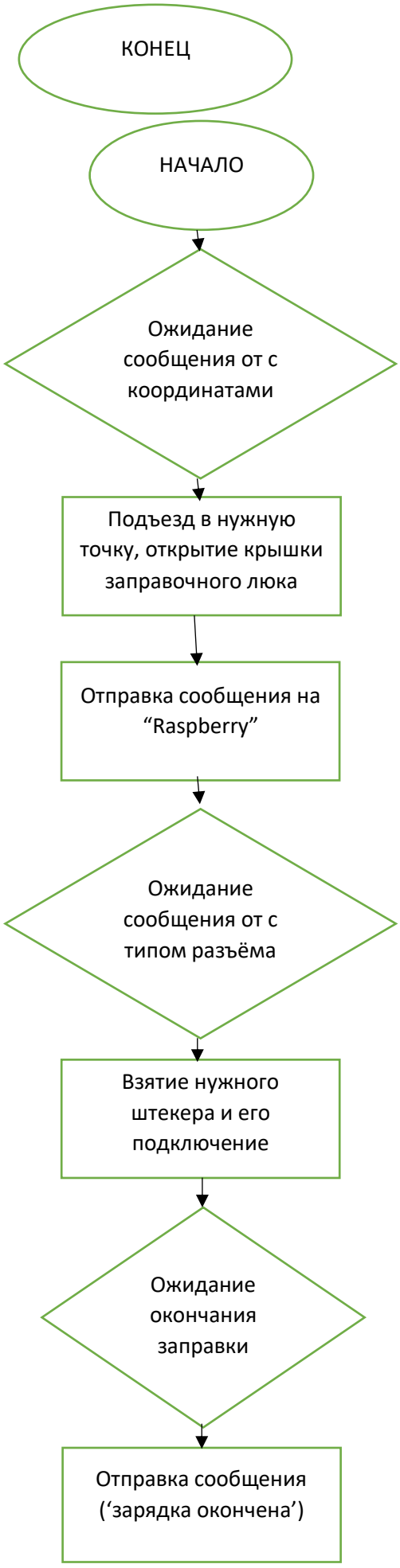
Приложение

Приложение 1 (блок-схемы программ)

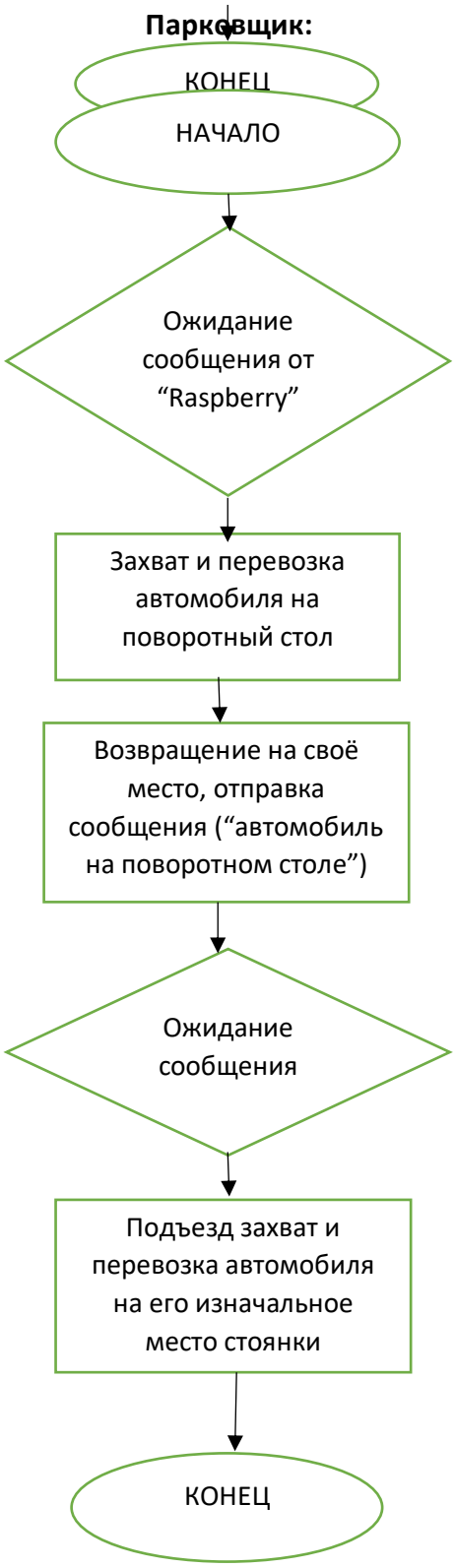
Raspberry:




Включение камеры и поиск qr-кода
QR-код найден?
Поворот стола на 180 градусов
Поиск местонахождения qr-кода и передача его координат в сообщение
Ожидание сообщения от “заправщика”
Определение типа разъёма и его отправка в виде сообщения “заправщику”
Ожидание сообщения от “заправщика”
Отправка данных на сайт об окончании зарядки


КОНЕЦ
НАЧАЛО
Ожидание сообщения от с координатами
Подъезд в нужную точку, открытие крышки заправочного люка
Отправка сообщения на “Raspberry”
Ожидание сообщения от с типом разъёма
Взятие нужного штекера и его подключение
Ожидание окончания заправки
Отправка сообщения (‘зарядка окончена’)

Парковщик:


КОНЕЦ
НАЧАЛО
Ожидание сообщения от “Raspberry”
Захват и перевозка автомобиля на поворотный стол
Возвращение на своё место, отправка сообщения (“автомобиль на поворотном столе”)
Ожидание сообщения
Подъезд захват и перевозка автомобиля на его изначальное место стоянки
КОНЕЦ

Приложение 2 (фрагменты кода)

Raspberry:

(Web-сервер)

```
import string,cgi,time
from os import curdir, sep
from http.server import BaseHTTPRequestHandler, HTTPServer
#import pri
import time
import sys

import gotoviy

class MyHandler(BaseHTTPRequestHandler):

    def do_GET(self):
        try:
            url = self.path

            #self.wfile.write(bytes("hey", "utf-8" ))

            if (url.find("set") > -1):
                self.send_response(200)
                self.send_header('Content-type',    'text/html')
                self.send_header('Access-Control-Allow-Origin', '*')
                self.send_header("Access-Control-Allow-
Methods" , "GET,POST,PUT,DELETE,OPTIONS")
                self.send_header("Access-Control-Allow-Headers", "Content-
Type, Access-Control-Allow-Headers, Authorization, X-Requested-With")
                self.end_headers()

                s = (url[url.find("set")+4::])
                if (s.find("&") > -1):
                    id = s[3:s.find("&")]
                    pl = s[s.find("&")+4::]
                    #self.wfile.write(bytes("done" + id + pl, "utf-8" ))
                    print("done\ncar:" + id + "\nplace:" + pl)
                    f = open('pay.txt','w')
                    f.write(str(id + "\n" + pl))
                    time.sleep(0.1)
                    f.close()
                    gotoviy.osnovn()

                return


            elif (url.find("get") > -1):
                self.send_response(200)
```

```
                self.send_header('Content-type',    'text/html')
                self.send_header('Access-Control-Allow-Origin', '*')
                self.send_header("Access-Control-Allow-
Methods" , "GET,POST,PUT,DELETE,OPTIONS")
                self.send_header("Access-Control-Allow-Headers", "Content-
Type, Access-Control-Allow-Headers, Authorization, X-Requested-With")
                self.end_headers()

                print("ok")
                f = open('status.txt','r')
                #print("01")
                status = f.read()
                #print("02")
                f.close
                #print("03")
                print("status:",status)
                self.wfile.write(bytes(status.encode('utf-8')))
                print("otpravil")
                if status == str(1):
                    f = open('status.txt','w')
                    f.write(str(0))
                    f.close
                    print("remove oll")
                return
            return
        except:
            #e = sys.exc_info()[1]
            #print(e.args[0])
            self.send_error(404,'File Not Found: %s' % self.path)

def main():
    try:
        server = HTTPServer(('', 8080), MyHandler)
        print ('started httpserver...')
        server.serve_forever()
    except KeyboardInterrupt:
        print ('^C received, shutting down server')
        server.socket.close()

if __name__ == '__main__':
    main()
```

(connector_type)

```
import cv2 as cv
import zbarlight
from PIL import Image
from subprocess import call

pic = '/home/pi/Downloads/qr-code_11.gif'
```

```
q = 1000

def makePhoto(pic):
    call(['fswebcam',pic])

def flip(img):
    return cv.flip(img, -1)

def figura ():
    a = [0]
    b = [0]
    c = [0]
    d = [0]
    e =[0]
    figur = 1
    approx = 1000
    makePhoto(pic)
    img = cv.imread("qr-code_11.gif")
    img = flip(img)
    img = cv.cvtColor(img, cv.COLOR_BGR2RGB)
    gray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)

    _, binary = cv.threshold(gray, 75, 255,cv.THRESH_BINARY)

    #cv.imshow('thresh', binary)
    cv.imwrite('/home/pi/Downloads/demonstration_1.jpg', binary)

    contours, hierarchy = cv.findContours(binary, cv.RETR_TREE, cv.CHAIN_APPROX_S
IMPLE)

    for cnt in contours:
        area = cv.contourArea(cnt)
        if area > 100:
            cv.drawContours(img, contours, -1, (255, 0, 0), 2)

            #cv.imshow("h",img)
            cv.imwrite('/home/pi/Downloads/demonstration_2.jpg', img)

            peri = cv.arcLength(cnt, True)
            if len(a)==1:
                a = cv.approxPolyDP(cnt, 0.02 * peri, True)
            elif len(b)==1:
                b = cv.approxPolyDP(cnt, 0.02 * peri, True)
            elif len(c)==1:
                c = cv.approxPolyDP(cnt, 0.02 * peri, True)
            elif len(d)==1:
                d = cv.approxPolyDP(cnt, 0.02 * peri, True)
            elif len(e)==1:
                e = cv.approxPolyDP(cnt, 0.02 * peri, True)
            '''else:
```

```
                c = cv.approxPolyDP(cnt, 0.02 * peri, True)'''
            '''if (len(a)==4 and len(b)==4) or (len(a)==4 and len(b)==5):
                approx = 3'''
            if len(a)>1 and len(b)>1:
                approx =(len(c))
            elif len(a)>1 and len(b)==1:
                approx = (len(a))
            print(approx,len(a),len(b),len(c),len(d),len(e))
    if approx == 1000:
        figur  == 1
    elif approx == 3:
        figur = 1
    elif approx == 4:
        figur = 2
    elif approx >=  8:
        figur = 3
    return figur
def forma():
    q = figura()
    print (q)
```

(main_part)

```
import forma

import robotNet
import time

def osnovn():
    m = ''
    f = open('pay.txt','r')
    number = f.readline(1)
    f.close
    print("number:",number)
    mes = number
    mbox = robotNet.initMailBox()
    robotNet.sendMessage(number)
    while m != "3":
        m=robotNet.getMessage(mbox)
        time.sleep(0.1)
    print("m:",m)
    time.sleep(0.1)
    forma.forma()
    time.sleep(0.1)
    with open('status.txt', 'w') as file:
        file.write("1")
        file.close
    print('remove')
```

(robotNet)

```
from socket import *
def initMailBox():
     "This function creates mailbox and opens socket"
     mbx = socket(AF_INET, SOCK_DGRAM)
     mbx.bind(('',12345))
     return mbx
def sendMessage(message):
     s=socket(AF_INET, SOCK_DGRAM)
     s.setsockopt(SOL_SOCKET, SO_BROADCAST, 1)
     s.sendto(message.encode('utf-8'),('255.255.255.255',12345))
def getMessage(mailbox):
     return mailbox.recv(1024).decode('utf-8')
```

(qr-code coordinates)

```
import cv2
import numpy as np
from subprocess import call


pic = '/home/pi/Downloads/foto.jpg'
def makePhoto(pic):
    call(['fswebcam',pic])
makePhoto(pic)

img = cv2.imread('foto.jpg')
height, width = img.shape[:2]
print(height,"---", width )

detector = cv2.QRCodeDetector()
# detect and decode
data, bbox, _ = detector.detectAndDecode(img)
print(data,bbox)
if bbox  is not None:
    print(bbox,data)
    x1 = bbox[0][0][0]
    y1 = bbox[0][0][1]
    x2 = bbox[0][2][0]
    y2 = bbox[0][2][1]
    print("x1= ",x1,"y1= ",y1,"x2= ",x2,"y2= ",y2)
    x = ((x2-x1) / 2) + x1
    y = ((y2-y1) / 2) + y1
    print("x= ",x,"y= ",y)
    cv2.circle(img, (int(x), int(y)), 10, (0, 0, 255), -1)
    if x < ((width / 3)):
        location = "left"
    elif x > ((width / 3))*2:
        location = "right"
```

```
    else:
        location = "center"
    print(location)
else:
    print("none qr")

cv2.imshow("img", img)
cv2.waitKey()
cv2.destroyAllWindows()
```

Заправщик

```
from ev3dev.ev3 import *

from time import sleep
from subprocess import call
import robotNet
mA = MediumMotor('outA')
mB = MediumMotor('outB')
mC = MediumMotor('outC')
mD = MediumMotor('outD')
btn = Button()
#cl1 = LightSensor ('in1')
#cl2 = LightSensor ('in2')
#cl3 = LightSensor ('in3')
#cl4 = LightSensor ('in4')

print('go')

#mbox = robotNet.initMailBox()
#msj = robotNet.getMessage(mbox)
#msk = robotNet.getMessage(mbox)
#msg = robotNet.getMessage(mbox)
msg = int(input())
print(msg)

def povorot(a,b):
    mA.run_timed(time_sp= a, speed_sp=b)
    mB.run_timed(time_sp= a, speed_sp=b)
    mC.run_timed(time_sp= a, speed_sp=b)
    mD.run_timed(time_sp= a, speed_sp=b)
    return 0

def vpered (a,b):
    mA.run_timed(time_sp=a, speed_sp=-b)
    mC.run_timed(time_sp=a, speed_sp=b)
    return 0

def nazad (a,b):
    mA.run_timed(time_sp=a, speed_sp=b)
    mC.run_timed(time_sp=a, speed_sp=-b)
```

```
    return 0

def vlevo (a,b):
    mB.run_timed(time_sp=a, speed_sp=b)
    mD.run_timed(time_sp=a, speed_sp=-b)
    return 0

def vpravo (a,b):
    mB.run_timed(time_sp=a, speed_sp=-b)
    mD.run_timed(time_sp=a, speed_sp=b)
    return 0

def Vpravo (a,b):
    mC.run_timed(time_sp=a, speed_sp=b)
    return 0

def Vlevo (a,b):
    mA.run_timed(time_sp=a, speed_sp=-b)
    return 0

def MotorStop():
    mA.stop(stop_action='hold')
    mB.stop(stop_action='hold')
    mC.stop(stop_action='hold')
    mD.stop(stop_action='hold')
    return 0

def R_diagonal (a,b):
    mA.run_timed(time_sp=a, speed_sp=-b)
    mB.run_timed(time_sp=a, speed_sp=-b)
    mC.run_timed(time_sp=a, speed_sp=b)
    mD.run_timed(time_sp=a, speed_sp=b)
    return 0

def BL_diagonal (a,b):
    mA.run_timed(time_sp=a, speed_sp=b)
    mB.run_timed(time_sp=a, speed_sp=b)
    mC.run_timed(time_sp=a, speed_sp=-b)
    mD.run_timed(time_sp=a, speed_sp=-b)
    return 0

def L_diagonal (a,b):
    mA.run_timed(time_sp=a, speed_sp=-b)
    mB.run_timed(time_sp=a, speed_sp=b)
    mC.run_timed(time_sp=a, speed_sp=b)
    mD.run_timed(time_sp=a, speed_sp=-b)
    return 0

def BR_diagonal (a,b):
    mA.run_timed(time_sp=a, speed_sp=b)
```

```
    mB.run_timed(time_sp=a, speed_sp=-b)
    mC.run_timed(time_sp=a, speed_sp=-b)
    mD.run_timed(time_sp=a, speed_sp=b)
    return 0
```

“Парковщик”

```
const int inaPin1 = 7;
const int inbPin1 = 9;
const int inaPin2 = 13;
const int inbPin2 = 11;
const int inaPin3 = 40;
const int inbPin3 = 42;
const int inaPin4 = 36;
const int inbPin4 = 38;

const int pwmPin1 = 3;
const int pwmPin2 = 5;
const int pwmPin3 = 44;
const int pwmPin4 = 46;

const int diagaPin = 10;
const int diagbPin = 12;
const int buttonPin = 2;
const int trimPin = A0;
int _speed = 50;
#define SIG_A_PIN 3
#define INT_A_PIN 2

long encA = 0;
long encB = 0;
long i = 0;
const int message = 2;
int line = 470;

void vpered() {
  digitalWrite(inaPin3, HIGH);
  digitalWrite(inbPin3, LOW);
  analogWrite(pwmPin3, _speed);

  digitalWrite(inaPin1, LOW);
  digitalWrite(inbPin1, HIGH);
  analogWrite(pwmPin1, _speed);

  digitalWrite(inaPin2, LOW);
  digitalWrite(inbPin2, HIGH);
  analogWrite(pwmPin2, _speed);

  digitalWrite(inaPin4, HIGH);
  digitalWrite(inbPin4, LOW);
```

```
  analogWrite(pwmPin4, _speed);
}

void vpravo() {
  digitalWrite(inaPin3, HIGH);
  digitalWrite(inbPin3, LOW);
  analogWrite(pwmPin3, _speed);

  digitalWrite(inaPin1, HIGH);
  digitalWrite(inbPin1, LOW);
  analogWrite(pwmPin1, _speed);

  digitalWrite(inaPin2, LOW);
  digitalWrite(inbPin2, HIGH);
  analogWrite(pwmPin2, _speed);

  digitalWrite(inaPin4, LOW);
  digitalWrite(inbPin4, HIGH);
  analogWrite(pwmPin4, _speed);
}

void vlevo() {
  digitalWrite(inaPin3, LOW);
  digitalWrite(inbPin3, HIGH);
  analogWrite(pwmPin3, _speed);

  digitalWrite(inaPin1, LOW);
  digitalWrite(inbPin1, HIGH);
  analogWrite(pwmPin1, _speed);

  digitalWrite(inaPin2, HIGH);
  digitalWrite(inbPin2, LOW);
  analogWrite(pwmPin2, _speed);

  digitalWrite(inaPin4, HIGH);
  digitalWrite(inbPin4, LOW);
  analogWrite(pwmPin4, _speed);
}

void nazad() {
  digitalWrite(inaPin3, LOW);
  digitalWrite(inbPin3, HIGH);
  analogWrite(pwmPin3, _speed);

  digitalWrite(inaPin1, HIGH);
  digitalWrite(inbPin1, LOW);
  analogWrite(pwmPin1, _speed);

  digitalWrite(inaPin2, HIGH);
  digitalWrite(inbPin2, LOW);
```

```
  analogWrite(pwmPin2, _speed);

  digitalWrite(inaPin4, LOW);
  digitalWrite(inbPin4, HIGH);
  analogWrite(pwmPin4, _speed);
}

void _stop() {
  analogWrite(pwmPin3, 0);
  analogWrite(pwmPin1, 0);
  analogWrite(pwmPin2, 0);
  analogWrite(pwmPin4, 0);
}

void povorotLeft() {
  digitalWrite(inaPin3, LOW);
  digitalWrite(inbPin3, HIGH);
  analogWrite(pwmPin3, _speed);

  digitalWrite(inaPin1, LOW);
  digitalWrite(inbPin1, HIGH);
  analogWrite(pwmPin1, _speed);

  digitalWrite(inaPin2, LOW);
  digitalWrite(inbPin2, HIGH);
  analogWrite(pwmPin2, _speed);

  digitalWrite(inaPin4, LOW);
  digitalWrite(inbPin4, HIGH);
  analogWrite(pwmPin4, _speed);
}

void povorotPravo() {
  digitalWrite(inaPin3, HIGH);
  digitalWrite(inbPin3, LOW);
  analogWrite(pwmPin3, _speed);

  digitalWrite(inaPin1, HIGH);
  digitalWrite(inbPin1, LOW);
  analogWrite(pwmPin1, _speed);

  digitalWrite(inaPin2, HIGH);
  digitalWrite(inbPin2, LOW);
  analogWrite(pwmPin2, _speed);

  digitalWrite(inaPin4, HIGH);
  digitalWrite(inbPin4, LOW);
  analogWrite(pwmPin4, _speed);
}
```

```
void doKresta() {
  while (analogRead(2) <= line and analogRead(1) <= line and analogRead(3) <= lin
e and analogRead(0) <= line) {
    vpered();
  }
  _stop();
}

void doLinePered() {
  while (analogRead(2) <= line and analogRead(1) <= line) {
    vpered();
  }
  _stop();
}

void doLineZad() {
  while (analogRead(2) <= line and analogRead(1) <= line) {
    vpered();
  }
  _stop();
}

void doLineLevo() {
  while (analogRead(0) <= line and analogRead(3) <= line) {
    vlevo();
  }
  _stop();
}

void doLinePravo() {
  while (analogRead(0) <= line and analogRead(3) <= line) {
    vpravo();
  }
  _stop();
}
void podCar() {
  vpered();
  delay(2000);
  _stop();
  if (message == 1) {
    doLinePered();
    vpered();
    delay(200);
    doLinePered();
    nazad();
    delay(100);
    _stop();
  }
  if (message == 2) {
    vpered();
```

```
    delay(2000);
    doLinePered();
    vpered();
    delay(200);
    doLinePered();
    nazad();
    delay(100);
    _stop();
  }
  if (message == 3) {
    vpered();
    delay(4000);
    doLinePered();
    vpered();
    delay(200);
    doLinePered();
    nazad();
    delay(100);
    _stop();
  }
}

void onLine() {
  vlevo();
  delay(500);
  doLineLevo();
  vlevo();
  delay(500);
  doLineLevo();
  _stop();
  vpravo();
  delay(80);
  povorotLeft();
  delay(600);
  while (analogRead(2) <= line or analogRead(1) <= line) {
    povorotLeft();
  }
  povorotLeft();
  delay(100);
  _stop();
}

void setup() {
  pinMode(buttonPin, INPUT);
  pinMode(inaPin1, OUTPUT);
  pinMode(inbPin1, OUTPUT);
  pinMode(inaPin2, OUTPUT);
  pinMode(inbPin2, OUTPUT);
  pinMode(inaPin3, OUTPUT);
  pinMode(inbPin3, OUTPUT);
```

```
  pinMode(inaPin4, OUTPUT);
  pinMode(inbPin4, OUTPUT);
  pinMode(pwmPin1, OUTPUT);
  pinMode(pwmPin2, OUTPUT);
  pinMode(pwmPin3, OUTPUT);
  pinMode(pwmPin4, OUTPUT);
  pinMode(diagaPin, INPUT);
  pinMode(diagbPin, INPUT);
  pinMode(trimPin, INPUT);
  pinMode(INT_A_PIN, INPUT);
  pinMode(SIG_A_PIN, INPUT);
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(A2, INPUT);
  pinMode(A3, INPUT);
  Serial.begin(9600);
}


void loop() {
  onLine();
  vlevo();
  delay(400);
  _stop();
  delay(100);
  while (analogRead(0) <= line and analogRead(3) <= line) {
    if (analogRead(2) <= line - 10) {
      digitalWrite(inaPin1, LOW);
      digitalWrite(inbPin1, HIGH);
      analogWrite(pwmPin1, _speed + 20);

      digitalWrite(inaPin4, HIGH);
      digitalWrite(inbPin4, LOW);
      analogWrite(pwmPin4, _speed + 20);
    }
    if (analogRead(2) >= line - 10){
      digitalWrite(inaPin3, LOW);
      digitalWrite(inbPin3, HIGH);
      analogWrite(pwmPin3, _speed + 20);

      digitalWrite(inaPin2, HIGH);
      digitalWrite(inbPin2, LOW);
      analogWrite(pwmPin2, _speed + 20);
    }
  }
  _stop();
  while (0 != 1) {}
}
```