# Подробное описание конкурса «Роботы в городе» Школьный трек

# Задача конкурса

Исправить программу управления роботом (выдаётся вместе с заданием) и проехать маршрут из начальной точки (синяя) до конечной точки (красная), пример показан на рисунке 1. При этом необходимо пройти все точки, отмеченные белыми кружками в порядке их нумерации, траектория движения при этом должна соответствовать красной линии, движение должно происходить в направлении стрелок по правой полосе (правостороннее движение).

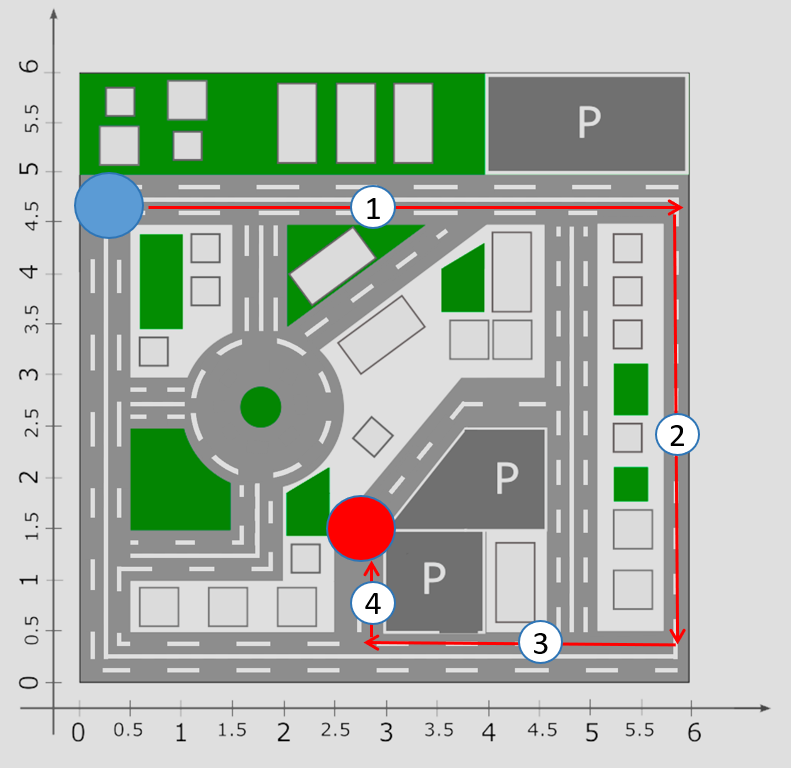
****

Рисунок 1 - пример маршрута

# Метрики оценивания прохождения маршрутов командами

### Начисление баллов:

### Баллы за проезд ключевых точек (включая старт и финиш): 1 балл за точку.

Штрафные баллы: -1 балл за выезд с участка дороги, -1 балл за столкновение с участником дорожного движения.

Оценивание знания программы проводит жюри сразу после проезда. Капитан команды объясняет все основные блоки его программы, показывает где были ошибки, которые команда исправила в ходе подготовки.

При равном количестве баллов за прохождение и знание программы, победителя выбирают по наименьшему времени выполнения попытки.

# Макет города

План полунатурного макета беспилотных транспортных систем (по осям отображены реальные физические размеры в метрах) изображён на рисунке 2:

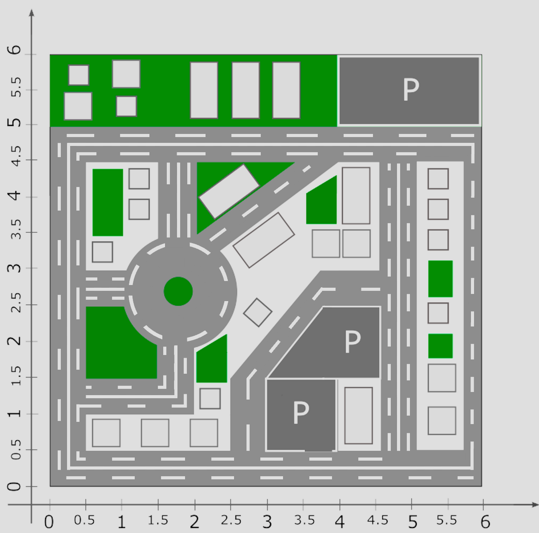


Рисунок 2 - план макета

Также вокруг макета (по углам) расставлена локальная радионавигационная система, имитирующая спутниковую группировку для обеспечения глобального позиционирования в условиях городской местности. Система представляет собой набор сверхширокополосных приемо-передающих радиомодулей MDEK 1001. На следующем плане можно наблюдать расставленные опорные радиомаяки с их именами:

Координаты опорных радиомаяков (x,y) соответствуют системе координат макета (дополнительно: высота маяков 2 м) изображены на рисунке 3:

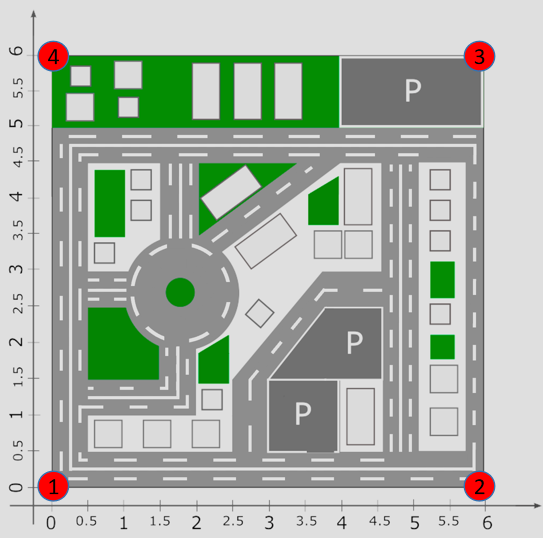


Рисунок 3 - расположение радиомаяков

# Колесные роботы

Колесный робот (рисунок 4), управляемый одноплатным компьютером Raspberry Pi 4B (ОЗУ 8 ГБ). Способ управления движением - дифференциальный привод (управление отдельно правой и отдельно левой стороной робота).

Язык программирования: Python (желательно версия 3.11).

Комплект датчиков:

- Ультразвуковой датчик HC-SR04 (на выходе дальности до препятствий);

- Инерциальный датчик GY-85 (на выходе ускорения по трем осям от датчика гироскопа и углы поворота по 3 осям от акселерометра).

- СШП приемо-передающий радиомодуль MDEK 1001 (на выходе координаты в системе координат макета).

- Камера (на выходе картинка с камеры разрешением 640 на 480 пикселей)

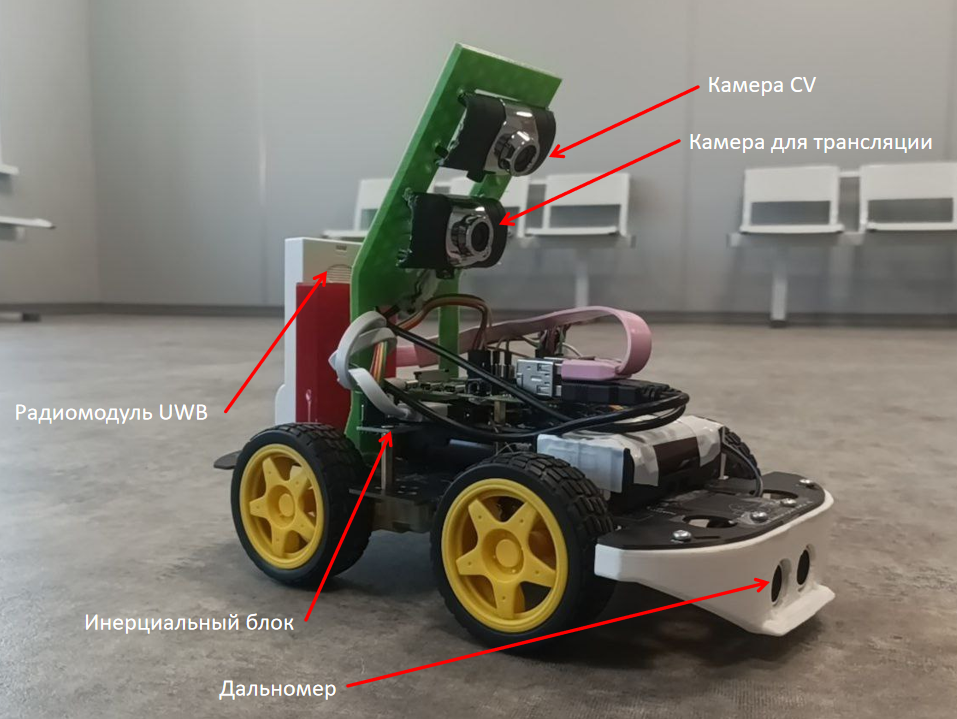


Рисунок 4 - колесный робот

Электрическая схема сборки робота представлена на рисунке 5:

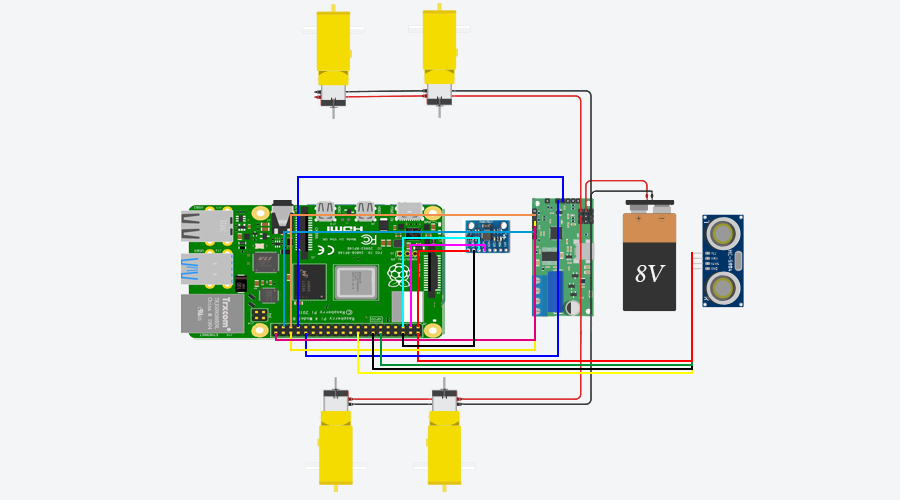


Рисунок 5 - электрическая схема робота

Порты Raspberry Pi, к которым подключаются все измерители и управляющие устройства:

Гироскоп:

3.3v - 3.3v

GND - GND

SCL - SCL (GPIO03)

SDA - SDA (GPIO02)

Драйвер моторов:

ENA - GPIO13

ENB - GPIO12

IN1 - GPIO19

IN2 - GPIO16

IN3 - GPIO21

IN4 - GPIO26

Ультразвуковой дальномер:

VCC - VCC

GND - GND  
 TRIG - GPIO18

ECHO - GPIO24

СШП (UWB) приемо - передатчик:

USB - '/dev/ttyACM0'

Веб камера:

USB - cv2.VideoCapture(2, cv2.CAP\_V4L)

# Web-интерфейс

Для получения доступа к макету необходимо зарегистрироваться с помощью формы регистрации, изображённой на рисунке 6:

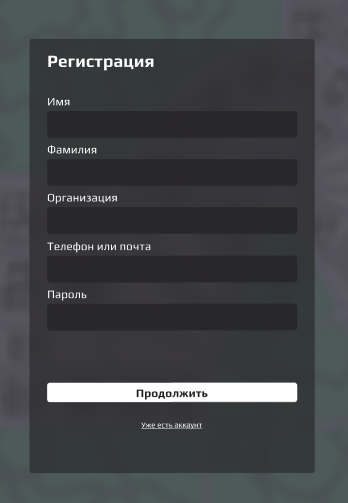


Рисунок 6 - регистрация

После регистрации нужно зайти по данным, указанным при регистрации:

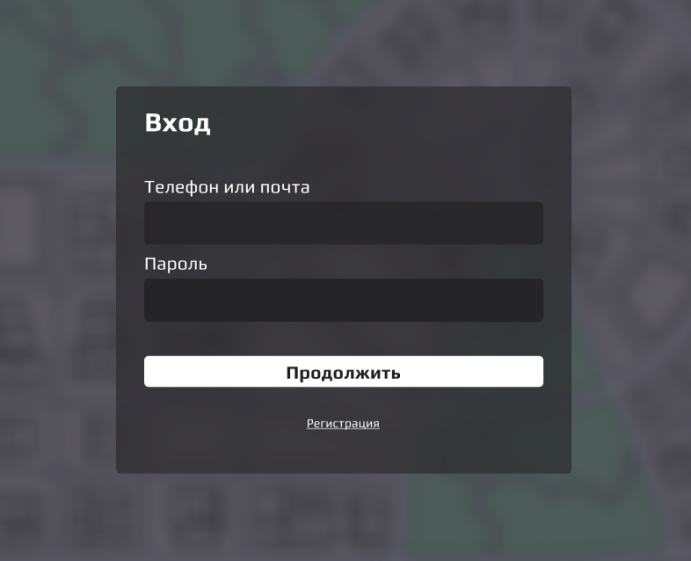


Рисунок 7 - авторизация

Для подключения выбираем свободного и доступного робота(рисунок 7):

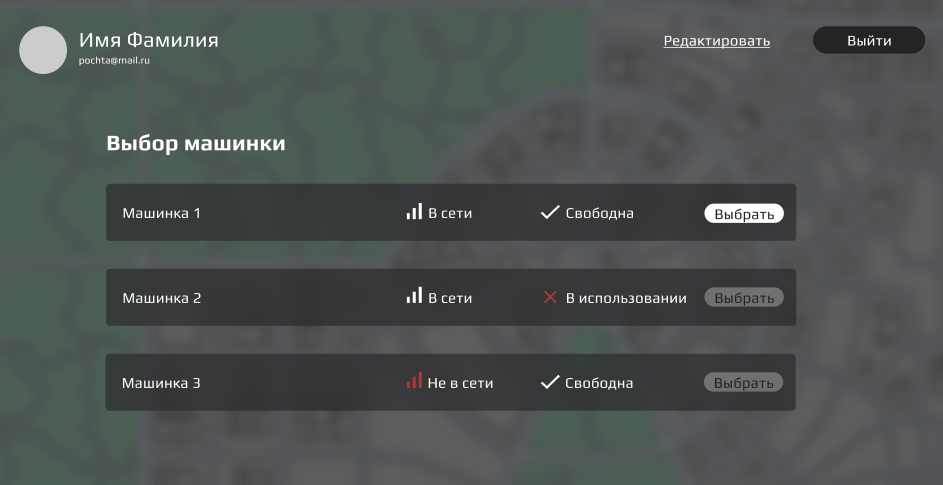


Рисунок 7 - выбор робота

На рисунке 8 изображено окно управления роботом. Для управления роботом можно использовать стрелки внизу экрана или на клавиатуре (управление с клавиш можно включить переключателем, который находится под изображением с камеры робота). В центре экрана расположены окна с потоком видео с робота и картой макета с местоположением робота. Справа расположено окно выдачи сообщений от выполняемой программы (логи с выводом в терминал).

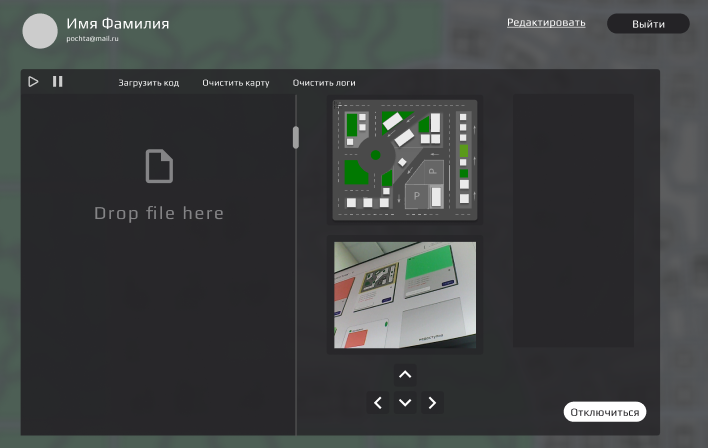


Рисунок 8 - окно управления роботом

# Инструкция по написанию программ для робота

Программа в виде файла c расширением **.py** загружается на робота посредством веб-интерфейса. Для этого в поле загрузки перетаскивается файл и нажимается кнопка загрузить код.

Чтобы запустить выполнение файла нужно нажать

остановить

Важно не использовать русский язык в комментариях в файле программы, программы с русскими символами не запускаются через веб-интерфейс.

Далее представлены основные функции, которые **необходимо** использовать для движения и получения данных с измерителей. Создание собственных функция для чтения данных с измерителей или движения могут не работать. Обращаем внимание на то, что ограничение связано только с обращением к измерителям и к портам моторов, остальные функции и алгоритмы и работа с модулями доступны без ограничений.

В файле **user.py** (будет выслан вместе с данным руководством) реализованы основные функции для получения данных от измерителей и управления моторами на роботе. Так же в этом файле есть базовая структура программы, которую рекомендуется использовать при написании своей программы.

Все необходимые библиотеки нужно заранее добавить в список и отправить запрос Организаторам (https://t.me/RazorvinAndrey), с указанием версии в формате:

numpy==1.25.1

Без отправки запроса, использование библиотек не из списка

предустановленных будет невозможным, так как библиотеки

устанавливаются заранее.

Список предустановленных библиотек:

numpy==1.25.1

opencv-python==4.8.0.74

Для вывода сообщений на удаленный интерфейс достаточно просто отправить сообщение в **print(“some message”, flush=True)**,с обязательным параметром **flush=True**, без него сообщение не передаться на веб-интерфейс.

Базовые функции для получения данных и управления роботом:

●**GetPosition() –** получить координаты на карте. Возвращает строку,

содержащую информацию о текущем местоположении машинки

относительно 4 источников сигнала (стойки) - две координаты

(X, Y).

Для проверки на получение координат нужно уточнить

существует ли **data = GetPosition()**

●**GetGyro()** - Получение данных с гироскопа, ускорение по 3 осям.

●**GetAxiler()** - Получение данных об углах по 3 осям от акселерометра.

●**GetImage()** – Возвращает из камеры текущий фрейм или, если нет подключения к камере или нет фрейма выдает **False**.

●**GetDistance()** – определить расстояние до ближайшего

препятствия в сантиметрах. Возвращает: расстояние и ошибку.

Если все в порядке: дистанция, "ОК"

Если превышено время ожидания ответа: 0, "Тайм-аут"

Если расстояние превышает 3 метра или равно 0: расстояние "Вне

досягаемости";

●**MotorForward()** – движение машины вперед, на вход подается скорость от 0 до 100. Движение будет

продолжаться до тех пор, пока не будет вызвана другая функция

движения или функция MotorStop().

●**MotorBackward()** – движение машины назад, на вход подается скорость от 0 до 100. Движение будет

продолжаться до тех пор, пока не будет вызвана другая функция

движения или функция MotorStop().

●**MotorTurnRight()** – поворот машины направо, на вход подается скорость от 0 до 100. Поворот будет

продолжаться до тех пор, пока не будет вызвана другая функция

движения или функция MotorStop().

●**MotorTurnLeft()** – поворот машины направо, на вход подается скорость от 0 до 100. Поворот будет

продолжаться до тех пор, пока не будет вызвана другая функция

движения или функция MotorStop().

●**MotorStop()** – остановка машины. Машина будет стоять до

тех пор, пока не будет объявлено другое движение.

Код базовой программы **user.py** будет приложен к данной инструкции.

**Материалы для подготовки**

**Туториалы и примеры работы с Raspberry PI**:

1. <https://raspberry3.ru/sozdanie-avtonomnogo-robota-na-baze-raspberry-pi-poshagovoe-rukovodstvo/>
2. <https://raspberrypi.ru/574-bubot-ochen-legkiy-freymvork-na-python-3-dlya-programmirovaniya-robotov-i-domashney-avtomatizatsii>