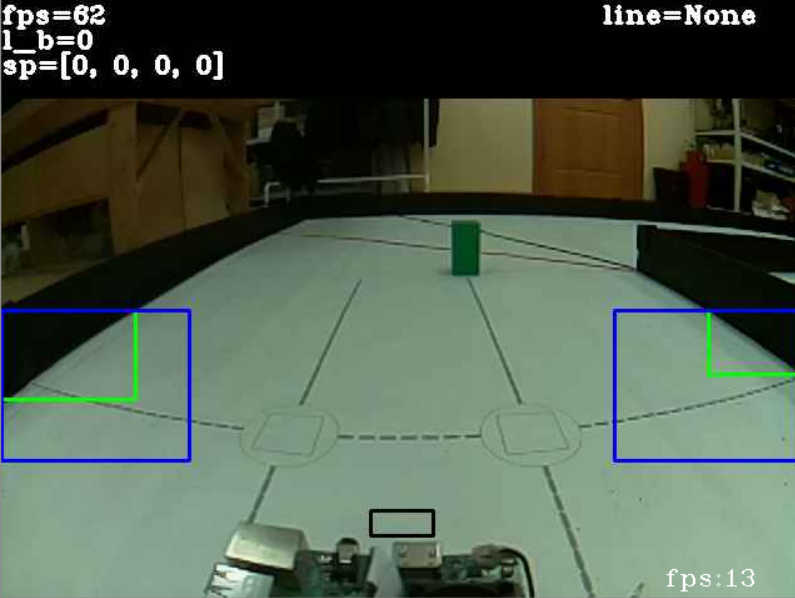
**Реализация системы распознавания препятствий**

Для того чтобы робот мог выполнять все поставленные задачи, мы написали программу на языке python. Разработку мы вели в среде Pycharm.

**Квалификация**

В этапе квалификации, робот должен проехать три круга и остановится в зоне старта. Для этого мы используем изображение с камеры, рис. 1. 



*рис. 1. Вид с камеры на этапе квалификации.*

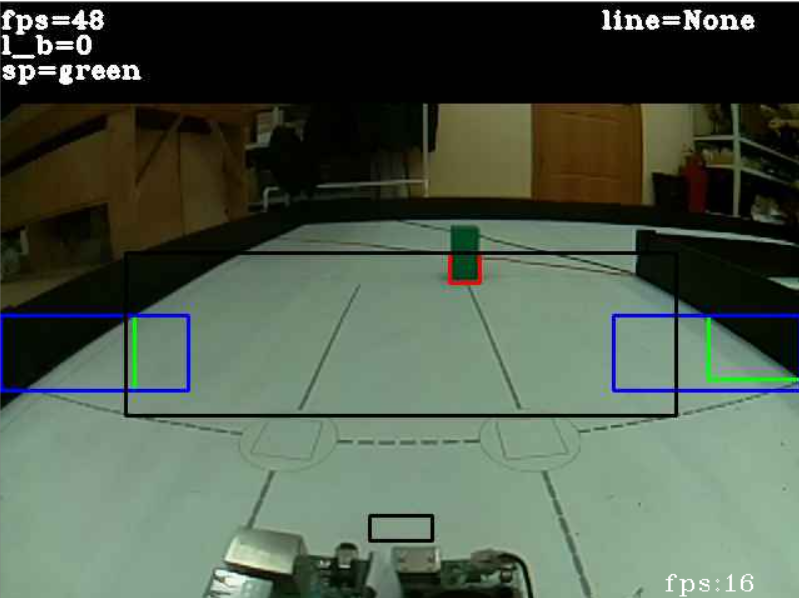
В этом этапе мы выбрали три зоны интереса, которые служат некими “датчиками”. Они выделены на изображении прямоугольниками, и подписаны на рисунке.

Датчик №1 – датчик синей и оранжевой линий. Когда датчик замечает первую линию он определяет её цвет и понимает в каком направлении он едет, и с этого момента считает только линии того цвета, который увидел первым (В коде для этого создана функция **dlin()**). Количество линий записывается в переменную **l\_b**, а цвет в **line** (Эти переменные выведены в телеметрию выше изображения камеры). Таким образом насчитав 12 линий одного цвета, он понимает что проехал 3 круга, и проезжая еще некоторое время останавливается в точке, с которой стартовал.

Датчики №2 и №3 – Датчики черных бортиков, именно на них построен наш алгоритм движения. Они отслеживают черный цвет, выделяют его контур зеленым цветом, и находят размеры этого контура, их ширину мы сохраняем в переменные **dl**, для левого датчика №2 с помощью функции **dlz(),** и **dr**, для правого датчика №3 с помощью функции **drz().** С помощью функции **p()** у нас реализован пропорционально-дифференциальный регулятор. В ней мы определяем угол поворота по формуле – **servo = k \* e + 0,1 \* (e - eold)**, в которой **servo** – значение угла поворота серво мотора, **k** – коэффициент равный 0,1, **e** – ошибка, которая равна разнице между показанием левого и правого датчика (**e = dl - dr**), а **eold** – это ошибка прошлых показаний. Также в функции **p()** у нас есть ограничение на максимальный угол поворота сервомотора от -50 до 50 и проверка на случай, при котором на датчиках мы не видим черный цвет. В таком случае мы в зависимости от направления движения поворачиваем на 20, при движении по оранжевым линиям, против часовой стрелки, и на -20, при движении по синим линям, по часовой стрелке.

**Финал**

В финале у робота появляется новая задача, а именно знаки красного и зеленого цвета, которые он должен объезжать с права и слева соответственно. На рис.2 представлено изображение с камеры в финале.





*рис. 2. Вид с камеры в финале.*

Здесь мы создали новую область интереса №4. В ней мы отслеживаем знаки впереди робота и их цвет, который записывается в переменную **sp**. Алгоритм обнаружения знаков реализован в функции **datz()**, в ней мы находим контур увиденного знака, и сохраняем его размеры. В зависимости от цвета увиденного знака, вызывается либо функция **pdzr()**, для красного знака, либо функция **pdzg()**, для зеленого знака. В этих функциях реализован алгоритм объезда знаков. Он работает по принципу пропорционально-дифференциального регулятора, только он заставляет робота ориентироваться не по бортикам, а по контуру знака. В случае красного знака ошибка показаний высчитывается по формуле **e = (xz + wz) – xo**, где **xz** это самая левая координата **x** знака, **wz** – ширина контура, а xo – переменная, в которой учитывается перспектива, относительно координаты y, и высчитывается она по формуле **xo = 240 – (yz + hz) \* 0,8** (**yz** – координата y нижней точки знака, **hz** – ширина контура знака, а **240** и **0,8** – подобранные коэффициенты). Для зеленого знака ошибка высчитывается как **e = xz – xo**, а учет перспективы идет по формуле **xo = 210 + (yz + hz) \* 1,1**. Таким образом, объезд зеленого знака происходит с левой стороны, а красного справа. Также в случае если робот не видит никакого знака, то он едет по обычному регулятору **p()**.