

Создание робота для Roborace*

Dmitry Sklipus, Brest, Belarus

Author shares his development experience of Arduino-based robots targeted at Roborace competitions. Specifics of the competition is explained as far as some details of the hardware platform and the control algorithm.

Специфика Roborace

Roborace — это соревнования, в которых соревнуются роботы-автомобили на специальной кольцевой трассе. Можно провести некоторую аналогию между Roborace и гонками Формулы 1, за исключением двух моментов.

- Во-первых, вместо полномасштабных гоночных болидов участвуют уменьшенные модели авто и оригинальные конструкции с габаритными и весовыми ограничениями (максимальные ШхД=25х50 см и вес до 3 кг).
- Во-вторых, вместо пилотов автомобилем управляет бортовой компьютер, который анализирует показания различных датчиков и ориентировывает автомобиль на трассе, выбирает скорость движения, предотвращает столкновения с препятствиями и соперниками. Собственно “поведение” авто на трассе определяется управляющей программой бортового компьютера.

Roborace проводится в виде чемпионата, состоящего из этапов, которые организуются в различных городах Беларуси и за рубежом. Участие в чемпионате принимают как конструкции начального уровня (например, на базе конструктора типа LEGO), так и сложные робототехнические устройства. Регламенты соревнований формируются таким образом, чтобы охватить как можно более широкий спектр характеристик и возможностей робототехнических конструкций.

Рассмотрим трассу, изображенную на рисунке 31.1, по которой предстоит перемещаться роботу. Обязательными элементами являются черные линии и стенки. Исходя из этого можно строить стратегию движения робота по трассе: например, оборудовать робота

*sklipus@gmail.com, <http://lvee.org/ru/abstracts/163>

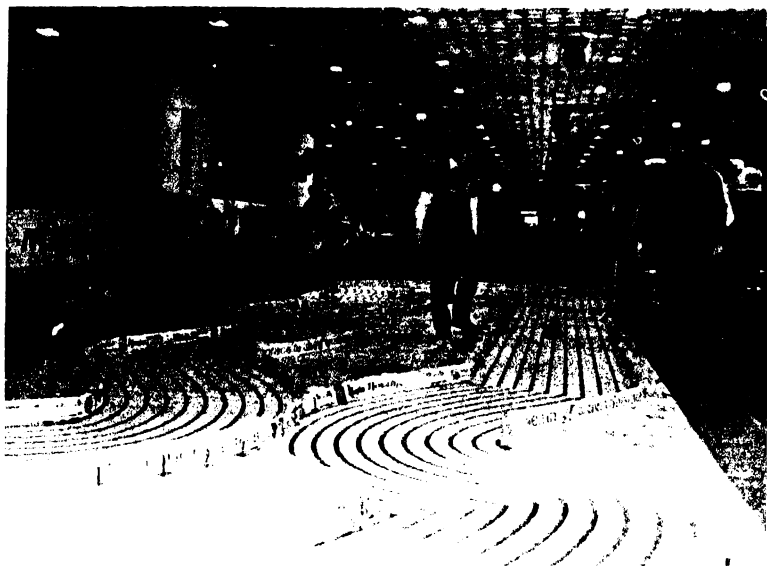


Рис. 31.1. Трасса для гонок роботов

датчиками черной линии и использовать линии трассы для навигации, или установить дальномеры для обнаружения препятствий и двигаться вдоль стенок.

В рамках данной статьи представлен один из роботов, разработанный мной для Robogame по второй стратегии (движение на основе показаний дальномеров).

Этапы создания робота

Создание робота для Robogame начинается с выбора шасси. Сейчас магазины предлагают большой выбор гусеничных и колесных платформ. Я рекомендую остановиться на классической схеме, когда задние колеса приводятся в движение электродвигателем, а передние управляются сервоприводом [1]. На рисунке 31.2 изображен робот для Robogame, построенный по подобной схеме.

Мне повезло за небольшие деньги приобрести модель в местном клубе радио-моделистов (читатели могут попытаться сделать то же самое в своем регионе: обычно у них много устаревших моделей). Так как в приобретенной модели не было тягового электродвигате-

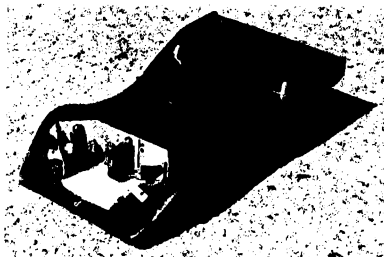


Рис. 31.2. Робот

ля, на нее был установлен купленный 12-вольтный мотор. Можно было также использовать обычную игрушку: они обычно довольно живучи, и требуется только модифицировать рулевое управление.

Так как в моем случае сервопривод уже был установлен, с ним проблем не возникло.

Следующий этап — выбор платы управления. Тут есть множество вариантов. Я выбрал Arduino как самый простой вариант. То же можно порекомендовать и читателю, особенно при недостатке опыта. Исходя из моего достаточно большого опыта, для таких роботов достаточно обычных 8-битных микроконтроллеров. Поэтому если не планируется использовать для отслеживания движений робота камеру, не стоит усложнять его более мощным процессором.

Сервопривод можно напрямую подключить к Arduino — например, через sensor shield, изображенный на рисунке 31.3. К нему также удобно подключать датчики.

Мотор подключить к Arduino напрямую не получится. Нужно использовать специальные Motor Driver. Сейчас их достаточно много в продаже, и есть инструкции по подключению. Я использовал Motor Driver, разработанный в нашей лаборатории (рис. 31.4).

В соревнованиях роботов приходится много внимания уделять батареям. Я использую литий-полимерные аккумуляторы. Они очень хорошо себя зарекомендовали. Один из хаков, которые я применяю в своем роботе, касается преобразователя напряжения. Штатный преобразователь в Arduino не очень хорош, поэтому для экономии энергии аккумулятора неплохо использовать Step-Down регулятор [2]. Конечно, можно использовать и обычный линейный преобразователь.

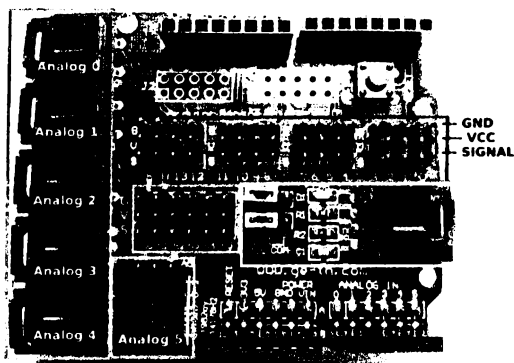


Рис. 31.3. Sensor shield v4

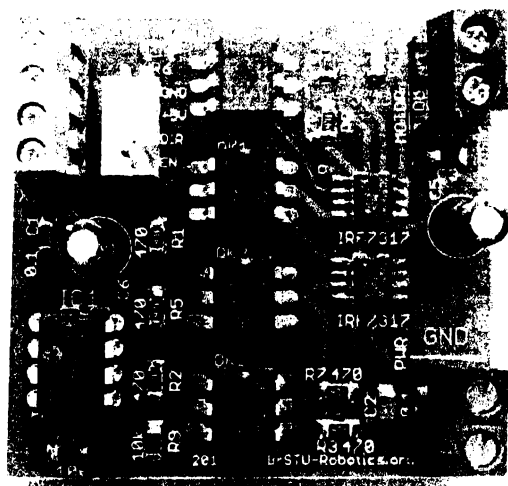


Рис. 31.4. Motor driver

Самая главная часть робота это датчики — то, что обеспечивает его информацией об окружающем мире, о препятствиях и о других роботах. В средней ценовой категории мы можем выбирать из *ультразвуковых* и *инфракрасных* датчиков. В своем роботе я использую инфракрасные датчики GP2Y0A02YK0F. Мне не нравятся

ультразвуковые датчики из-за того, что может происходить зашумление одного датчика другим. Например, у меня возникали такие ситуации: правый датчик посылал сигнал, а левый его принимал. Я все ещё работаю над правильным размещением ультразвуковых датчиков и над управлением ими. Надежду их запустить постоянно подпитывает их цена.

На представленной здесь модели робота установлено три инфракрасных датчика. Датчики можно увидеть на рисунке 31.2. Они установлены в глубь корпуса по двум причинам:

1. для уменьшения мертвой зоны датчика, которая у данной модели составляет 20 см;
2. корпус робота защищает датчики от механических повреждений во время столкновений с другими роботами.

Боковые датчики установлены под углом 45 градусов. Хорошо, если в конструкции робота предусмотрена регулировка угла их установки. Общую схему робота можно посмотреть на рисунке 31.5.

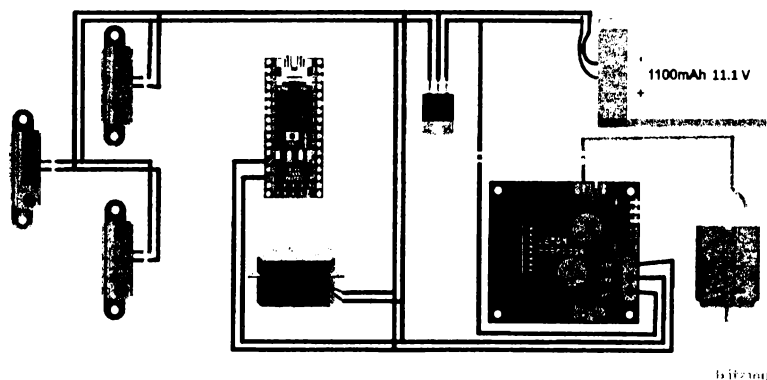


Рис. 31.5. Общая схема робота

Программирование робота

Так как на роботе используется Arduino, то программирование выполняется с использованием Arduino IDE. Программа робота представляет собой замкнутый цикл, который состоит из следующих блоков:

1. Фильтрация показаний датчиков;
2. Вычисление угла и скорости движения робота;
3. Передача управляющих сигналов на механизмы.

В данной структуре отсутствует блок получения информации от датчиков. Так как датчики возвращают аналоговый сигнал, в Arduino IDE есть функция `analogWrite()`. Данная функция замечательно работает, если не важна скорость измерения. Но так как робот разрабатывался для соревнований, было принято решение вынести обработку датчиков в прерывание.

Все платы Arduino, построенные на микроконтроллере ATmega, имеют возможность проводить измерения АЦП в автоматическом режиме. Нужно один раз настроить этот режим, а потом пользоваться полученными значениями. В результате контроллер постоянно проверяет датчики, не тратя на это процессорное время. Фильтрация показаний датчиков осуществляется медианным фильтром с окном в три элемента.

Для движения по трассе был выработан следующий алгоритм. Робот сравнивает расстояния до правой и левой стенки, и в соответствии с этим поворачивает колеса в нужное направление. Если впереди робота нет препятствий, скорость увеличивается, но также уменьшается максимально возможный угол поворота колес. Это нужно для того, чтобы на прямых участках робот ехал более прямо. При обнаружении препятствия угол поворота колес увеличивается, и робот притормаживает.

Есть конечно и нерешенные проблемы. Например, робот не знает кривизну поворота, поэтому тормозит перед каждым поворотом.

Посмотреть код проекта можно на GitHub [3].

Литература

- [1] https://en.wikipedia.org/wiki/Servo_%28radio_control%29
- [2] https://www.google.com/url?q=https%3A%2F%2Fsites.google.com%2Fsite%2Fsklipusrobotssystem%2Fhardware%2Fpower-of-robot&sa=D&sntz=1&usg=AFQjCNHdVp_jzQoLmZ8pIL3PrrfXRarkuA
- [3] <https://github.com/sklipus/roborace/tree/master/robots/FreshSRDEDITION>