Сергей Александрович Филиппов

ОСНОВЫ РОБОТОТЕХНИКИ НА БАЗЕ КОНСТРУКТОРА LEGO MINDSTORMS NXT. ЗАНЯТИЕ 3. УПРАВЛЕНИЕ МОБИЛЬНЫМ РОБОТОМ

ДВУХМОТОРНАЯ ТЕЛЕЖКА

Главная особенность мобильного робота – способность к маневрированию. В отличие от привычного автомобиля, оснащенного рулевым управлением и одним двигателем, робот может быть оснащен несколькими моторами, два из которых используются для движения и маневров, соединенные по отдельности с левыми и правыми колесами. В жизни встречаются подобные устройства, например, танки или электрокары.

Двухмоторная тележка – это самая распространенная разновидность мобильных роботов. Тележка может быть с тремя точками опоры, две из которых – ведущие колеса, а третья – волокуша, или свободно вращающееся колесико (рис. 1). Такие модели являются базовыми для наборов 8527 и 9797. Их инструкции по сборке содержатся в прилагаемых к набору книжечках. Если попытаетесь построить такую тележку сами, помните, что центр масс должен находиться не над волокушей, а ближе к ведущим колесам.

Именно по этой схеме построена стандартная тележка из наборов 8527 и 9797 (рис. 2). В инструкциях этих наборов есть небольшие отличия, но суть одна.

Для тех, кто не хочет ограничиваться замысловатыми базовыми конструкциями, рассмотрим простейший пример крепления моторов к контроллеру NXT. От него можно отталкиваться при создании собственных роботов.

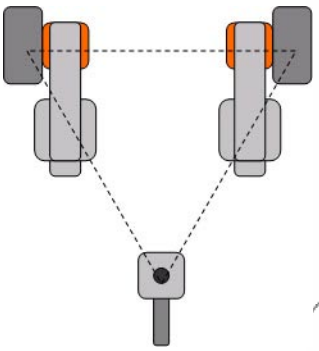
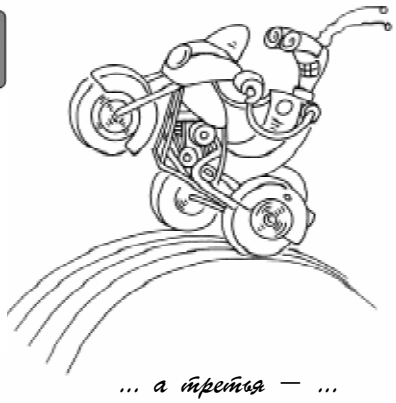


Рис. 1. Схема шасси трехколесной тележки с подвижным третьим колесом



... а третья — ... свободно вращающееся колесико.

Рис. 2. Стандартная основа для робота из набора 9797

Рис. 3. Широкая тележка – простейший вариант

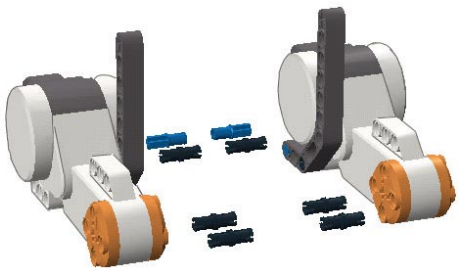


Рис. 5. В зависимости от расположения балок может быть смещен центр тяжести тележки

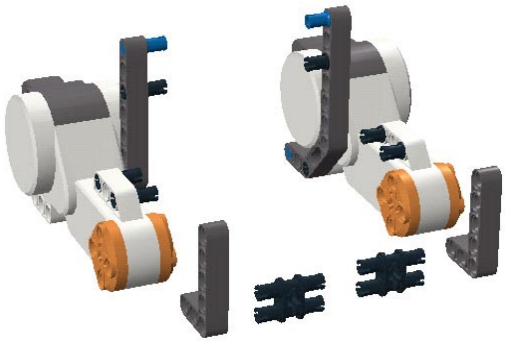


Рис. 6. Дополнительные крепления для придания устойчивости

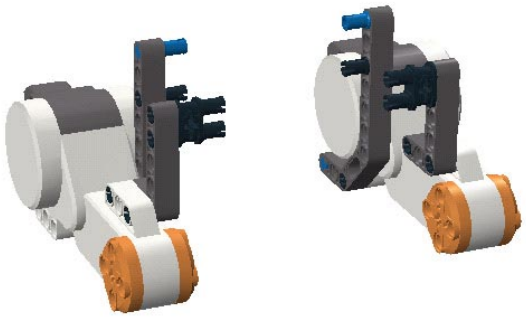


Рис. 7. Все готово для установки контроллера

Рис. 4. Изогнутые балки для крепления моторов

ПРОСТЕЙШАЯ ТЕЛЕЖКА

Для придания устойчивости роботу имеет смысл поставить моторы по двум сторонам от контроллера. Это несколько расширит корпус тележки (рис. 3).

Предлагаемую конструкцию (рис. 4–12) можно делать вдвоем – большая часть деталей устанавливается симметрично. А вот на подключение моторов следует обратить внимание. Для совместимости с алгоритмами, изложенными в этой статье, договоримся, что мотор B – слева, а мотор C – справа по курсу движения. На нашей тележке провода придется подсоединить накрест.

Простейшая конструкция тележки может быть закончена рис. 9, однако в ней есть пара недостатков. Корпус тележки расположен с небольшим наклоном вперед. Если убрать одну втулку из вертикальной оси подвижного колеса, корпус выровняется, но тогда тележка потеряет

Рис. 8. Колеса устанавливаются на 6-модульные оси, втулки предохраняют от нежелательного трения о корпус

Рис. 9. Сзади моторы можно скрепить 15-модульной балкой, но тогда тележка будет слегка наклонена вперед

Рис. 10. Немного усложним конструкцию, подняв заднюю балку

возможность двигаться назад: колесико начнет цепляться за балку. Чуть более сложная, но универсальная конструкция продолжается на рис. 10 (для этого пропускаем рис. 9).

УПРАВЛЕНИЕ БЕЗ ОБРАТНОЙ СВЯЗИ

В задачах управления обычно существуют два объекта: управляющий и управляемый. В простейшем варианте от управляющего объекта поступает команда и управляемый выполняет ее, ничего не сообщая о результате или об изменившихся условиях работы. В этом суть прямой связи (рис. 13).

С точки зрения мобильного робота управляющий объект – это его контроллер с запущенной программой, объект управления – это его колеса и корпус (шасси). Управляющие команды контроллер подает на моторы, при прямой связи руководствуясь показаниями своих внутренних часов – таймера.

Первый класс задач, с которых начинается программирование, – это управление перемещениями робота. Рассмотрим их по порядку. В качестве сред программирования используем Robolab 2.9.4 для начинающих и RobotC для подготовленных программистов.

Рис. 11. Элементы подвижного колеса. Длины осей – 3 и 5 модулей. Колесики должны вращаться свободно

Рис. 12. Корпус такой тележки расположен горизонтально и колесико вращается вокруг вертикальной оси свободно

Рис. 13. Прямая связь в управлении

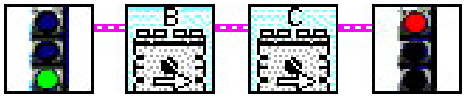


Рис. 14. Включение моторов

Рис. 15. Остановка при попытке начать движение

ДВИЖЕНИЕ В ТЕЧЕНИЕ ЗАДАННОГО ВРЕМЕНИ ВПЕРЕД И НАЗАД

Для движения вперед используются команды управления моторами. Эти команды просто включают моторы. Особенность RCX заключается в том, что после окончания выполнения программы сохраняются все установки в поведении робота. То есть если моторы включены, тележка продолжит движение (рис. 14).

```
task main() // Пример включения
            // моторов на RobotC
{
motor[motorB] = 100; // моторы вперед
motor[motorC] = 100;// с максимальной
                    // мощностью
}
```

Обе команды выполняются практически мгновенно. Если сразу следом за ними выключить моторы, то тележка просто дернется и останется стоять на месте (рис. 15).

```
task main()
{
motor[motorB] = 100;
motor[motorC] = 100;
motor[motorB] = 0; // стоп мотор
motor[motorC] = 0;
}
```

Таким образом, для осуществления движения требуется некоторая задержка перед выключением моторов. Команды ожидания не производят никаких конкретных действий, зато дают возможность моторам выполнить свою часть работы (рис. 16).

```
task main()
{
motor[motorB] = 100;
motor[motorC] = 100;
wait1Msec(1000);   // Ждать 1000 мс
motor[motorB] = 0;
motor[motorC] = 0;
}
```

Движение вперед или назад, очевидно, определяется направлением вращения моторов (рис. 17). Для смены направления не требуется остановка.

```
task main()
{
motor[motorB] = 100;
motor[motorC] = 100;
wait1Msec(1000);
motor[motorB] = -100;
                  // «Полный назад»
motor[motorC] = -100;
wait1Msec(1000);
motor[motorB] = 0;
motor[motorC] = 0;
}
```

В момент смены направления на высокой скорости возможен занос. Плавное

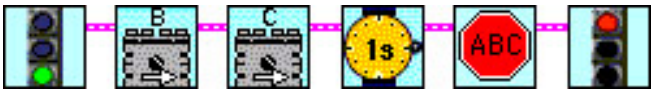


Рис. 16. Правильный порядок управления моторами

Рис. 17. Проехать секунду вперед, секунду назад и остановиться

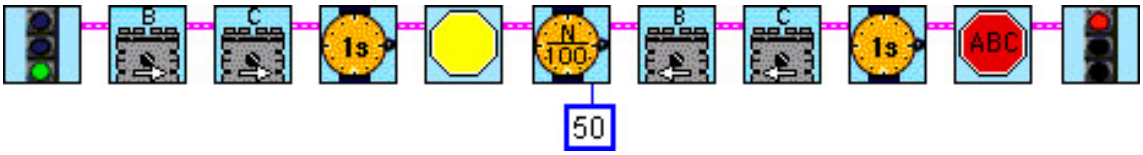


Рис. 18. Перед сменой направления полсекунды ехать по инерции

торможение возможно. Для этого перед подачей команды «назад» с моторов снимается напряжение и робот некоторое время едет по инерции (рис. 18).

Более краткий промежуток, чем одна секунда, задается с помощью команды «N/100» и модификатора. В Robolab 2.9.4 можно задавать время в миллисекундах командой «N/1000».

```
task main()
{
motor[motorB] = 100;
motor[motorC] = 100;
wait1Msec(1000);
// Включить плавающий режим
// управления моторами
bFloatDuringInactiveMotorPWM = true;
motor[motorB] = 0;
motor[motorC] = 0;
wait1Msec(500);
motor[motorB] = -100;
motor[motorC] = -100;
wait1Msec(1000);
// Включить режим «торможения»
bFloatDuringInactiveMotorPWM = false;
motor[motorB] = 0;
motor[motorC] = 0;
}
```

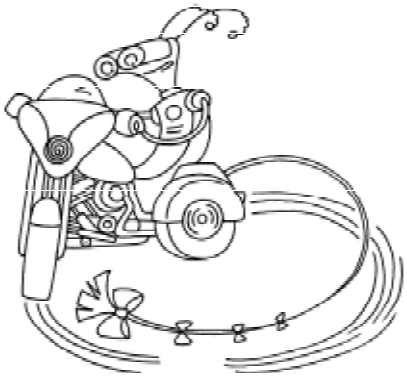
В Robolab обычными командами моторы включаются в плавающем режиме, а в RobotC по умолчанию используется режим «торможения», который позволяет достичь более точного управления. Но и в Robolab существуют «продвинутые» команды управления моторами в режиме торможения да еще с диапазоном мощностей от –100 до 100.

ПОВОРОТЫ

Для выполнения поворота на месте достаточно включить моторы в разные стороны. Тогда робот будет вращаться приблизительно вокруг центра оси ведущих колес со смещением в сторону центра тяжести. Для более точного поворота надо подбирать время в сотых долях секунды (рис. 19). Однако при изменении заряда батареек придется вводить новые параметры поворота.

```
task main()
{
motor[motorB] = 100;  // Моторы в
motor[motorC] = -100; // разные стороны
wait1Msec(300);
motor[motorB] = 0;
motor[motorC] = 0;
}
```

Существует другой тип поворотов. Если один из моторов остановить, а другой включить, то вращение будет проис-



Повороты на месте...

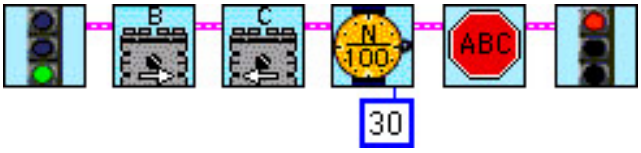


Рис. 19. Поворот на месте

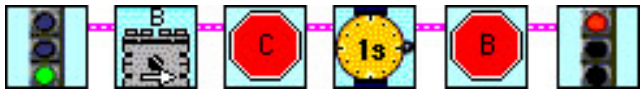


Рис. 20. Плавный поворот

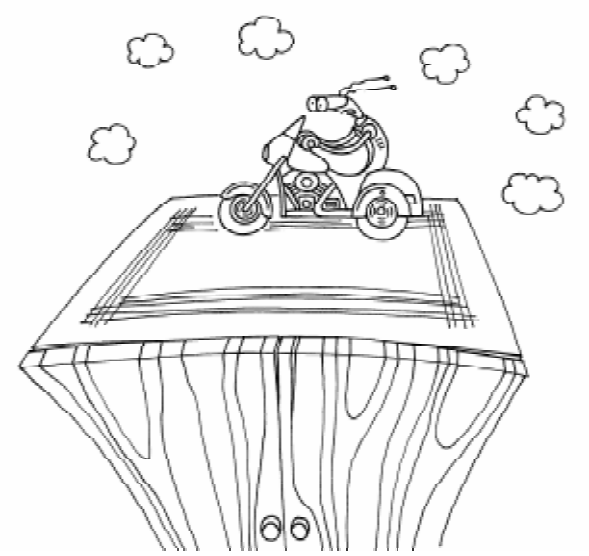
Рис. 21. Движение по многоугольнику с плавными поворотами

ходить вокруг стоящего мотора. Поворот получится более плавным (рис. 20).

```
task main()
{
motor[motorB] = 100;
motor[motorC] = 0;
wait1Msec(1000);
       // вращается только мотор B
motor[motorB] = 0;
}
```

ДВИЖЕНИЕ ПО КВАДРАТУ

Используя полученные знания управления моторами, можно запрограммировать движение по квадрату или другому многоугольнику с помощью цикла или безусловного перехода (рис. 21).



...движение по квадрату с помощью безусловного перехода...

```
task main()
{
while (true){
  motor[motorB] = 100;
  motor[motorC] = 100;
  wait1Msec(1000);
  motor[motorC] = 0;
  wait1Msec(1000);
  motor[motorB] = 0;
}
}
```

Уточнив длительность поворотов и количество повторений, научим тележку объезжать квадрат по периметру один раз (рис. 22). Для точности поворотов снизим мощность моторов примерно вдвое. Задержки придется подобрать самостоятельно.

```
task main()
{
for(int i=0;i<4;i++){
        // Цикл выполняется 4 раза
  motor[motorB] = 50;
  motor[motorC] = 50;
  wait1Msec(1000);
  motor[motorC] = -50;
  wait1Msec(400);
  motor[motorB] = 0;
}
}
```

УПРАВЛЕНИЕ С ОБРАТНОЙ СВЯЗЬЮ

ОБРАТНАЯ СВЯЗЬ

Появление обратной связи в системе означает то, что управляющий объект начинает получать информацию об объекте управления (рис. 23).

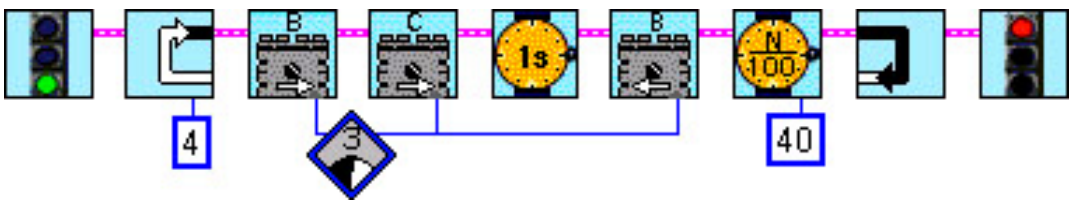


Рис. 22. Для поворота на 90 градусов длительность придется подобрать самостоятельно

Обратная связь осуществляется с помощью датчиков, прикрепленных, например, на корпус робота.

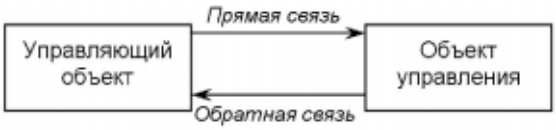


Рис. 23. Управление с обратной связью

ТОЧНЫЕ ПЕРЕМЕЩЕНИЯ

Чтобы поворот не зависел от заряда батареек, можно воспользоваться встроенным в двигатели датчиком оборотов, «энкодером», который позволяет делать измерения с точностью до 1 градуса. Для более эффективного управления задействуем в Robolab «продвинутые» команды, считая что при повороте тележки на 90° левое колесо поворачивается на 250° вокруг своей оси (рис. 24).

```
task main()
{
nMotorEncoder[motorB]=0;
          // Инициализация энкодера
motor[motorB] = 100;
motor[motorC] = -100;
// Пустой цикл ожидания
//показаний энкодера
while(nMotorEncoder[motorB]<250);
motor[motorB] = 0;
motor[motorC] = 0;
}
```

Теперь читателю нетрудно будет самостоятельно построить алгоритм движения по квадрату с использованием датчика оборотов.

ПУТЕШЕСТВИЕ ПО КОМНАТЕ

Естественно, нормальная среда обитания для робота, построенного из школьного или домашнего конструктора, – это комната с мебелью. И для начала неплохо было бы научиться путешествовать по ней, по возможности не натыкаясь на предметы и не застревая.

Подходящая конструкция для такого робота – это трехколесная тележка с установленным ультразвуковым датчиком наверху (рис. 25). Этот датчик следует расположить строго горизонтально относи-

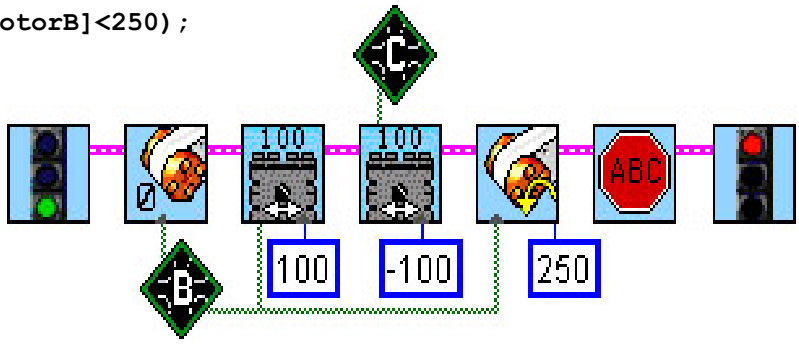


Рис. 24. Точный поворот на месте

Рис. 25. Маленький исследователь из набора 9797 с ультразвуковым датчиком

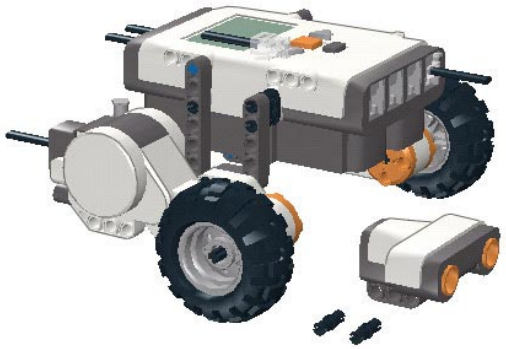


Рис. 26. Крепление датчика расстояния к корпусу тележки

Рис. 27. Датчик должен смотреть строго горизонтально

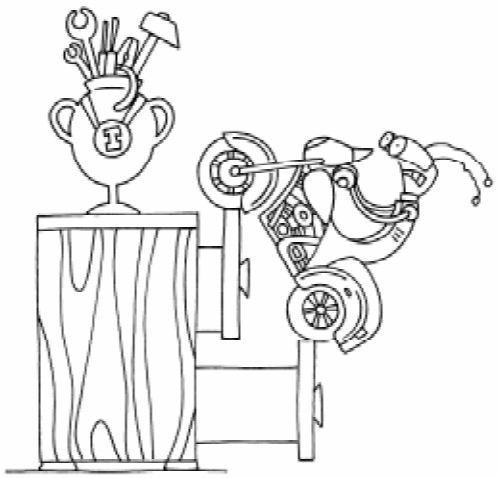
```
task main()
{
while (true){
  motor[motorB] = 100;
  motor[motorC] = 100;
  while(SensorValue[S4]>25);
// пустой цикл ожидания препятствия
  motor[motorB] = -100;
  motor[motorC] = -100;
  wait1Msec(500);
  motor[motorB] = 100;
  wait1Msec(600);
}
}
```

тельно пола, иначе любая соринка может быть воспринята как непреодолимое препятствие или, наоборот, что-то серьезное не будет замечено.

Более простой вариант конструкции (рис. 26–27) можно построить на основе тележки, которая рассматривалась ранее. В предлагаемых алгоритмах датчик подсоединяется на порт 4.

Программа похожа на движение по квадрату с небольшим добавлением: встречая предмет, робот немного отъезжает назад, прежде чем приступить к повороту (рис. 28).

Можно сделать несколько короче, если заменить отъезд назад с поворотом на ме-



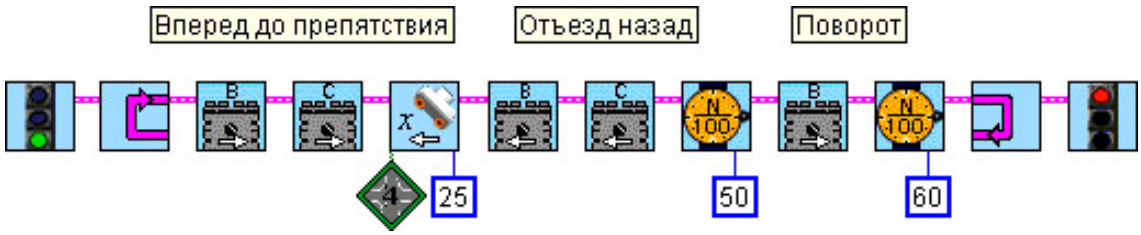


Рис. 28. Алгоритм путешествие по комнате

Рис. 29. Алгоритм путешествие по комнате с плавным поворотом задним ходом

сте одним действием: плавным поворотом задним ходом (рис. 29).

```
task main()
{
while (true){
  motor[motorB] = 100;
  motor[motorC] = 100;
  while(SensorValue[S4]>25);
  motor[motorB] = -100;
  motor[motorC] = 0;
  wait1Msec(700);
}
}
```

Правда, в некоторых условиях такой поворот может привести к небольшой аварии, так что будьте с ним осторожнее. Кстати, и во всех программах следует подобрать свои параметры для расстояния до предметов и длительности поворотов. А внимательный испытатель, конечно, быстро перейдет с временных задержек на более точные датчики оборотов.

Литература

1. *Филиппов С.А.* под ред. *Фрадкова А.Л.* Робототехника для детей и родителей. СПб.: Наука, 2010.

2. *Ананьевский М.С., Болтунов Г.И., Зайцев Ю.Е., Матвеев А.С., Фрадков А.Л., Шиегин В.В.* Под ред. *Фрадкова А.Л., Ананьевского М.С.* Санкт-Петербургские олимпиады по кибернетике. СПб.: Наука, 2006.

3. LEGO Technic Tora no Maki, ISOGAWA Yoshihito, Version 1.00 Isogawa Studio, Inc., 2007 // http://www.isogawastudio.co.jp/legostudio/toranomaki/en/.

4. Сайт подразделения Lego Education: http://www.lego.com/education/.

5. Среда трехмерного моделирования Lego Digital Designer: http://ldd.lego.com/.

6. Среда программирования RobotC: http://www.robotc.net/.

7. Сайт поддержки пользователей Lego Mindstorms, Robolab 2.9.4: http://www.legoengineering.com/.

Филиппов Сергей Александрович, учитель информатики физико-математического лицея № 239, методист.