Отчет по машине БТ-1

Оглавление

[Цель 1](#_Toc477260426)

[Реализация 1](#_Toc477260427)

[Arduino 2](#_Toc477260428)

[Управление двигателями 3](#_Toc477260429)

[Инфракрасные дальномеры 4](#_Toc477260430)

[Компас 5](#_Toc477260431)

[Движение робота 5](#_Toc477260432)

[Общий алгоритм 6](#_Toc477260433)

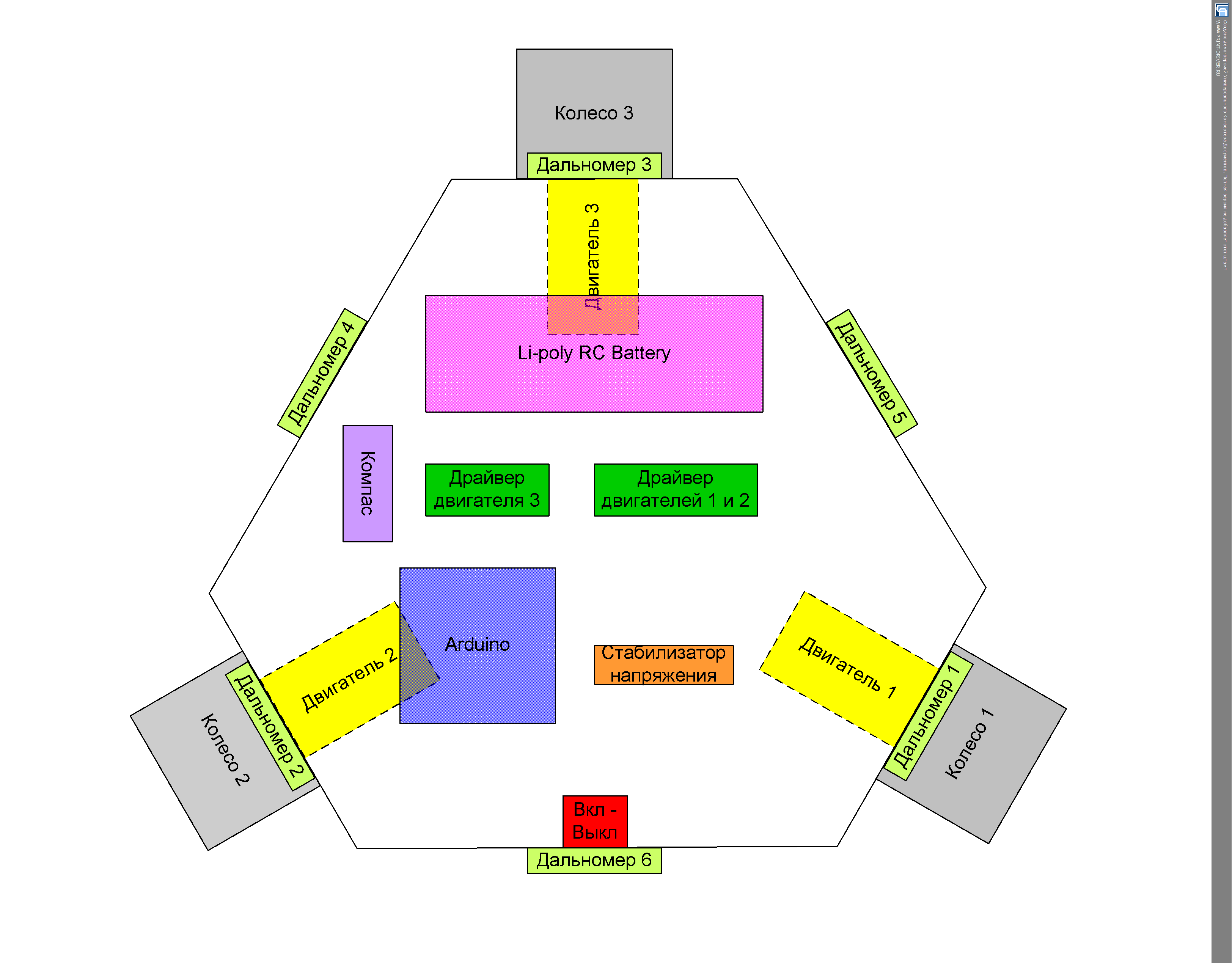
Автор отчета Абаимов Иван Сергеевич.

# Цель

При включении робот должен двигаться на север, ориентируясь по компасу и обходя примитивные препятствия, состоящие из стен, которые робот может зафиксировать с помощью дальномеров.

# Реализация

Примерная схема: (см. приложение «примерная схема.vsd»)



Робот выполняет поставленную задачу. Робот снабжен дополнительным оборудованием, так как использовался (и может использоваться) для реализации других целей.

Робот представляет из себя шестиугольную платформу, выполненную из фанеры, на которую установлены: arduino, драйверы, система питания, аккумулятор на 11,1 вольт, три электродвигателя и драйвера к ним, компас и 6 инфракрасных дальномеров (далее датчики). Двигатели расположены по трем сторонам на 120 градусов относительно друг друга, и на их осях закреплены омниколеса. 3 датчика расположены над двигателями на краю платформы, остальные три закреплены между ними так же на краю платформы. Компас закреплен над платформой на высоте 17 см для уменьшения возможных помех от остальной электроники. Для удобства у робота есть выраженный перед: со стороны, где расположена кнопка включения.

Схема прилагается: (см. приложение «схема соединений.vsd»)



# Arduino

На робота установлен контроллер Arduino-Nano(Плата) ATmega328(Процессор). Для работы с ним на компьютер нужно установить программу Arduino, подключить контроллер через USB порт, в разделе «Инструменты->Порт» установить нужный входной порт, в том же разделе «Инструменты» поставить соответствующий порт, процессор, а так же программатор mkll. При создании программы в скетч автоматически добавляется библиотека «arduino.h».

# Управление двигателями

Двигатели управляются через специальные драйвера. Каждый двигатель подключен к трем выходным портам платформы. Каждый двигатель пронумерован: 1 находится левее носа робота, второй – правее, а третий - сзади. Первый (out1\_1)отвечает за наличие вращения против часовой стрелки, третий (out1\_2) – наличие вращения по часовой, а второй (PWM\_1) – за скорость вращения (от 0 до 255). Для отправки сигнала по первому и третьему каналу следует использовать команду digitalWrite(номер порта, константы HIGH или LOW). Для отправки сигнала на второй порт следует использовать команду analogWrite(номер порта, скорость вращения).

Приведен пример программы для управления одним из двигателей Go1(скорость вращения от -255 до 255). Двигатель будет крутиться в против часовой стрелки при отрицательном параметре (в данном примере; для двигателя 2 и 3 наоборот)

#define out1\_1 2 // in1

#define PWM\_1 3 // in2 PWM

#define out1\_2 4 // in3

void Go1(int speed)

{

if(speed==0)

{

digitalWrite(out1\_1, HIGH);

digitalWrite(out1\_2, HIGH);

analogWrite(PWM\_1, 255);

}

else

if(speed>0)

{

digitalWrite(out1\_1, HIGH);

digitalWrite(out1\_2, LOW);

analogWrite(PWM\_1, speed);

}

else

{

digitalWrite(out1\_1, LOW);

digitalWrite(out1\_2, HIGH);

analogWrite(PWM\_1, -speed);

}

}

Аналогичные функции написаны для двух других двигателей.

Для удобства управления созданы дополнительные программы для управления роботом. Наличие омниколес позволяет роботу свободно перемещаться линейно вдоль шести направлений (0, 60, 120, 180, 240, 300) с помощью команд goFwd, goBack, goFwdLeft, goFwdRight, goBackLeft, goBackRight (все, как и последующие, с параметром скорости от -255 до 255), вращаться на месте , goLeft, goRight, ехать прямо заворачивая направо или налево goFR и goFL, остановиться – robotStop() без параметра. Вот пример одной из них:

void goFwd(int speed)

{

Go1(speed);

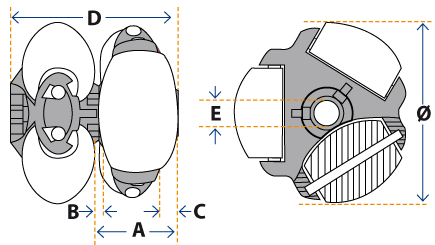
Go2(speed);

Go3(0);

}

Омниколесо – колесо с еще одной степенью свободы: оно может производить свободное вращение вдоль оси, что позволяет роботу перемещаться подобным образом (то есть вдоль оси одного из колес). В примере двум передним колесам отдается команда вращаться вперед. Оси колес расположены друг к другу под углом 120 градусов. При одинаковой скорости вращения остается только составляющая, направленная прямо, так как колеса расположены симметрично относительно друг друга. Третье колесо прокручивается по направлению движения за счет еще одной степени свободы. Аналогичным образом движение выполняется и в остальных направлениях. Внимание! Так как двигатели работают не с одинаковой силой, робот не двигается строго вдоль прямой.

Схема омниколеса:



# Инфракрасные дальномеры

Для определения препятствий на робота были установлены инфракрасные дальномеры SHARP 2Y0A21 (http://www.sharpsma.com/webfm\_send/1208), работающие на расстояниях от 10 до 80 см. Три из них установлены над двигателями на краю платформы и помечены соответственными цифрами, другие три ровно между ними так же на краю платформы и помечены цифрами 4, 5, 6. Эти датчики являются аналоговыми, и подключаются к соответственным портам. Пример работы с аналоговыми датчиками есть в разделе встроенных примеров «Файл->Примеры->03.Analog->AnalogOutSerial». Аналогично с этим примером происходит считывание данных с датчиков:

//Объявлены общие переменные

const int analogInPin1 = A7;

int sensorValue1 = 0;

int outputValue1 = 0;

//Получение аналоговых данных и их обработка

sensorValue1 = analogRead(analogInPin1);

outputValue1 = map(sensorValue1, 0, 1023, 0, 255);

analogWrite(analogOutPin, outputValue1);

Датчик может определить препятствие, только если оно расположено прямо напротив него. Чем дальше препятствие, тем меньше полученные данные (от 0 и больше). 15 см, это примерно 60 единиц. Следует учесть, что датчики могут отличаться по последним трем знаком, и это может влиять на показания.

Данные обновляются каждый цикл. Для уменьшения ошибок создано 6 массивов чисел (по одному для каждого датчика), размер которого обозначается числом nn, равным 20. Это оптимальный размер для данного массива, полученный экспериментально. После получения данных запускается функция writedown(), которая записывает последние 20 обновлений в каждый массив для каждого датчика, и записывает среднее арифметическое в переменные outputValue, с которыми робот продолжает работать.

# Компас

На робота установлен компас LSM303D 3D (https://www.pololu.com/product/2127). Он подключается к интерфейсу I2C. Для его использования скачена и установлена библиотека LSM303.h (https://github.com/pololu/lsm303-arduino). Для работы с компасом в скетче нужно подключить библиотеки LSM303.h и Wire.h, после чего завести объект класса LSM303. Подробно можно посмотреть в примерах, приложенных к библиотеке. В примере, приведенном ниже, compass – объект класса LSM303.

Для начала работы с компасом в функции begin следует написать:

Wire.begin();

compass.init();

compass.enableDefault();

compass.m\_min = (LSM303::vector<int16\_t>){-32767, -32767, -32767};

compass.m\_max = (LSM303::vector<int16\_t>){+32767, +32767, +32767};

Чтобы прочитать данные, нужно, чтобы компас их получил с помощью команды compass.read(), а после чего получить его показания в градусах с точностью до сотой доли с помощью функции compass.heading(), возвращающей переменные типа float. Для робота была специально определена переменная pointer, которая определяла направление, нужное роботу, и написана функция orientation(), приведенная ниже. Функция определяет показание компаса, их отличие от переменной pointer и возвращает разницу в виде числа типа float от -180 до 180. Функция возвращает отрицательное значение если полученное направление отличается от требуемого по часовой стрелке, и положительное, если против.

float orientation()

{

float head;

compass.read();

head = compass.heading();

float dif=pointer-head;

if(dif<=-180)dif=360+dif;

if(dif>=180)dif=-(360-dif);

return dif;

}

Данный компас так же может быть использован в качестве акселерометра и магнитометра. Для этого следует смотреть соответствующие скетчи, прилагаемые к библиотеке.

При тестировании в условиях лаборатории были выявлены сильные магнитные аномалии, изменяющие показания компаса и дающие погрешность +- 5 градусов. Так же возможно влияние работающих двигателя на показания компаса.

# Движение робота

В скетче есть две основные функции: setup и loop. Первая предназначена для начальных настроек, вторая после выполнения запускается заново бесконечное число раз. В начале такого цикла сначала выполняется считывание данных с датчиков, на основании которых роботом принимается решение по поводу направления движения. В общем доступе заведена переменная типа char, для определения действия робота: w – вперед, x – назад, a – вперед налево, d – вперед направо, z – назад налево, c – назад направо, q – развернуться налево, e – развернуться направо, r – вперед с поворотом направо, l – вперед с поворотом налево. В зависимости от значения этой переменной к концу переменной робот будет выполнять соответствующие движения. После запуска движения, двигатели будут работать самостоятельно.

Движение по направлению не будет производиться, если датчик, направленный вдоль линии движения, или два соседних с ним, зафиксирует препятствие.

Пример:

//Предохранитель от столкновений во время движения.

if(((outputValue5>da2)||(outputValue3>da)||(outputValue1>da)) && moving=='z')

{

robotStop();

}

…

//Определение действия робота, при условии того, что путь свободен.

if(((outputValue5<da2)&&(outputValue3<da)&&(outputValue1<da)))

{

if(moving=='z')goBackLeft(RSPEED);

}

//постоянная RSPEED задает стандартную скорость робота, равную 75.

Препятствие фиксируется при показаниях датчика больших, чем число da, равное 60, если датчик расположен над колесом, или больших чем da2, равное 80, если датчик находится на стороне между колесами. Эти постоянные разные для этих датчиков из-за того, что они расположены на разном расстоянии от центра робота, что должно быть компенсировано на случай, если робот будет поворачиваться.

Робот может смещаться во время вращения. При вращении вокруг своей оси робот не должен совершать движения по направлению к чему-либо, вследствие чего не будет останавливаться при столкновении со стенками. Для этого введено ограничение на вращение, предотвращающее серьезные столкновения с препятствиями: вращение не производится, если на одном из датчиков показания больше величины crit=200.

# Общий алгоритм

Алгоритм движения робота:

mt=’w’;//w – движение вперед с коррекцией,

//o – движение вдоль стены

moving =’w’;//смотреть раздел «Движение робота»

ошибка=+-20 градусов.

Нц

Получить данные с датчиков (outputValue1, 2, 3, 4, 5, 7).

Раз в 2 цикла

( dif = получить отклонение с компаса//смотреть раздел «Компас»

Если ((отклонение больше ошибки) И mt=='w')

То

Если (отклонение вправо) то moving =’q’;//взять левее

Если (отклонение влево) то moving =’e’;//взять правее

Иначе

mt=’w’;

Все

Если (спереди достаточно свободно)И  
 (движение вдоль стены)И(отклонение не больше 30))

то движение вперед;

Все

)

Если (спереди стена)

То

Если (движение вперед) то движение вдоль стены

Если (движение вдоль стены) то moving='q'

Иначе

Если (движение вдоль стены)

То

moving='w'

Если (outputValue2<da-15) то moving='r'

Все

Все

Движение(moving)//смотреть раздел «Движение робота»

Пауза 5 миллисекунд

Кц

Данный алгоритм позволяет объезжать роботу несложные прямоугольные и даже Г-образные препятствия.

Полная реализация на языке си, вставки из которой можно найти в тексте, прилагается в папке “bt1”, в файле “bt1.ino”.

Так же прилагаются результаты полевых исследований (пока нет).