УПРАВЛЕНИЕ ОБРАЗОВАНИЯ И ПО ДЕЛАМ МОЛОДЕЖИ
ГОРОДА НАБЕРЕЖНЫЕ ЧЕЛНЫ
МУНИЦИПАЛЬНОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ДОПОЛНИТЕЛЬНОГО ОБРАЗОВАНИЯ ДЕТЕЙ
«ЦЕНТР ДЕТСКОГО ТЕХНИЧЕСКОГО ТВОРЧЕСТВА №5»

Инновационные образовательные технологии
Внедрение робототехники в образовательном учреждении

Сборник №3

«Уже в школе дети должны получить
возможность раскрыть свои способности,
подготовиться к жизни в высокотехнологичном
конкурентном мире»

Д.А. Медведев

г. Набережные Челны
2013 г.

Сборник содержит рекомендации из опыта работы специалистов по направлению «Робототехника» при изучении модулей «механические передачи» и «решение задач для робота».

Сборник адресован педагогам дополнительного образования, учителям физики, информатики, математики, технологии.

Авторы-разработчики:

Хазиева М.Р., директор муниципального автономного образовательного учреждения дополнительного образования детей «Центр детского технического творчества №5».

Гиниятова Р.М., заведующий инструктивно-методическим отделом МАОУ ДОД «Центр детского технического творчества №5».

Мартынов С.Д., педагог дополнительного образования МАОУ ДОД «Центр детского технического творчества №5».

Ответственный редактор:

Ретивых О.В., заместитель директора по учебно-воспитательной работе муниципального автономного образовательного учреждения дополнительного образования детей «Центр детского технического творчества №5».

Технический редактор:

Хайбина Р.Р., педагог-организатор муниципального автономного образовательного учреждения дополнительного образования детей «Центр детского технического творчества №5».

СОДЕРЖАНИЕ

Аннотация

Высокопрофессиональные специалисты, обладающие знаниями в области робототехники, необычайно востребованы. Современная промышленность – это в первую очередь автоматизация процессов производства – готовить таких специалистов, с учетом постоянного роста объемов информации, необходимо со школьной скамьи.

Всё направление учебной деятельности под условным названием «робототехника» основано на ставших доступными образовательными конструкторами Lego, которые позволяют собирать из отдельных частей и деталей работающие приборы и механизмы.

Одним из вариантов интеграции робототехники в учебный процесс общеобразовательного учреждения являются уроки информатики, на которых программируются роботы, собранные из образовательных робототехнических конструкторов. Уже первые уроки робототехники дадут ученику, и учителю, понимание разницы между виртуальным и реальным миром. Нельзя забывать о том, что:

– во время появления наборов LRGO MINDSTORMS 2.0 в образовательном учреждении учитель и ученики оказываются в информационном вакууме, т.к. источники информации по теме «робототехника» в сети на русском языке исчисляются единицами;

– четко обозначилось не соответствие между необходимостью включения робототехники в образовательный процесс для приобретения учащимися образовательных результатов, востребованных на рынке труда, и не разработанностью этих вопросов в педагогической науке;

– требования времени и общества к информационной компетентности учащихся постоянно возрастают. Ученик должен быть мобильным, современным, готовым к разработке и внедрению инноваций в жизнь.

При организации учебных занятий по направлению «робототехника» учитель сталкивается с такими проблемами как:

– отсутствие методологической базы;

– недостаточное материально-техническое обеспечение (результат – слабое практическое применение учениками знаний курса «робототехники» для разработки и внедрения инноваций в дальнейшей жизни).

В системе дополнительного образования детей направление «робототехника» выделяется как один из компонентов учебного процесса и у педагога дополнительного образования больше возможности по реализации программы и подготовке воспитанников для участия в мероприятиях различного уровня. При этом не надо забывать об обязательном использовании в образовательном процессе межпредметных связей.

Это одна из наиболее сложных методических задач. Она требует знаний содержания программ и учебников по общеобразовательным

предметам. Реализация межпредметных связей в практике обучения робототехнике предполагает сотрудничество педагога дополнительного образования с учителями математики, физики, технологии, информатики. Посещения открытых уроков, совместного планирования уроков и т.д.

Методика творческой работы педагога по направлению «робототехника» включает ряд этапов:

1. Разработка рабочей программы.
2. Изучение опорных тем из программ и учебников по математике, физике, технологии, информатике; чтение дополнительной научной, научно-популярной и методической литературы.
3. Поурочное планирование с использованием межпредметных связей.
4. Разработка средств и методических приемов реализации межпредметных связей на конкретных уроках;
5. Разработка методики подготовки и проведения комплексных форм организации обучения.
6. Разработка приемов контроля и оценки результатов.

В своей работе при реализации программы робототехника я столкнулся с тем, что на занятия приходят дети, которые дома или в других кружках собирали роботов по готовым инструкциям и схемам. Программировали роботов, используя информацию, взятую с Интернет. Когда перед воспитанниками ставилась конкретная задача, они не могли решить ее без готовых инструкций. Исходя из этого, приступая к обучению воспитанников основам робототехники, я выделил два основных этапа:

– сборка механических передач,

– программирование.

В данном сборнике приведены методические рекомендации для начинающих свою деятельность педагогов в области робототехники по изучению следующих модулей:

– механические передачи (расчет, сборка и применение механических передач),

– решение задач для робота.

Педагог дополнительного образования
МАОУ ДОД «ЦДТТ №5» **С.Д. Мартынов**

Механические передачи

Обучаясь по программе «робототехника», воспитанник впервые встречается с решением комплексных инженерных задач, начинает знакомиться с инженерным образом мыслей и приобщается к инженерному творчеству в процессе проектирования механизмов и конструирования их узлов и деталей. При этом самым трудным для начинающего является приобретение навыка при переходе от реальной машины к ее идеализированной расчетной схеме и обратно.

Цель: изучение принципов механики и действия простейших механизмов.

Задачи:

– знакомство с простейшими механизмами (рычаг, наклонная плоскость, винт, колесо, ось);

– приобретение навыков конструирования, проектирования;

– развитие логического мышления и пространственного воображения;

– расширение кругозора в познании окружающего мира

Полученные знания и умения помогут воспитанникам в понимании принципов работы механических устройств, будут полезны при изучении начального курса школьной физики, а также создание «привлекательного имиджа» *механики* у школьников.

При организации учебных занятий необходимо обратить внимание воспитанников на вопросы устройства и анализа действия механизмов, важных для понимания физической природы явлений.

Все технические устройства используют различные силы природы, заставляя их служить обществу.

Каждое устройство, в зависимости от выполняемых функций состоит из различных частей. Практически все машины и аппараты имеют механическую часть, которая нередко оказывается самой громоздкой, тяжелой и дорогой.

В свою очередь, механическая часть подразделяется на несущую конструкцию, представляющую собой раму, каркас или корпус, и совокупность движущихся тел, связанных определенным образом между собой и с несущей конструкцией. Самую многочисленную группу механических машин образуют такие, которые построены из одних только твердых тел. Машины, в которых не происходит преобразования энергии, предназначенные только для выполнения полезной механической работы и построенные только из одних твердых тел, называются *механизмами*.

Для того чтобы построить из твердых тел механизм, эти тела нужно соединить друг с другом так, чтобы все их относительные движения, кроме необходимых, стали невозможны. Такие подвижные соединения соприкасающихся твердых тел называют *кинематическими парами*, а

образующие их тела - *звеньями*. Система звеньев, подвижно связанных между собой, образует кинематическую цепь. Механизм представляет собой кинематическую цепь, используемую для осуществления требуемого движения.

По условиям монтажа и производства *звенья* не изготавливают как монолит, а собирают из отдельных неподвижно соединенных *деталей.* Последние ради ускорения процесса сборки предварительно соединяют *в сборочные единицы (узлы), комплекты и подкомплекты.* Детали и узлы должны быть сконструированы так, чтобы их можно было изготовить высокопроизводительными методами, т.е. они должны быть *технологичными.*

Таким образом, весьма важно, чтобы в разных машинах применялись в возможно большем количестве однотипные детали и узлы, что дает возможность организовать их массовое производство на специализированных заводах самыми совершенными технологическими методами. Все это объясняет современную тенденцию к расширению использования стандартных деталей и узлов.

Несмотря на то, что механизмы используют в различных областях инженерной деятельности, с их помощью решают сходные задачи. Поэтому различные машины часто содержат большое число однотипных деталей и простейших механизмов (это видно на примере набора ЛЕГО). Изучая конструкцию механизма, следует постоянно помнить о требованиях технологии изготовления. Центральное место занимает проблема устройства и работоспособности кинематических пар, составляющих основу всего механизма.

Зубчатые передачи

Важнейшей частью почти каждого робота является механическая передача. Передача необходима для того, чтобы передать крутящий момент с вала двигателя на колеса или другие движущиеся части робота.

Создавая модели роботов, учащиеся применяют различные виды механических передач. На практических примерах закрепляют понятия рычага в физике и реальными рычагами, встречающимися в природе. Поэтому при изучении данного раздела необходимо четко усвоить определения, характеристики и виды простых механизмов – рычаг, условия равновесия тела, имеющего неподвижную ось вращения.

Зубчатая передача – механизм, в котором два подвижных звена являются зубчатыми колесами, образующими с неподвижным звеном вращательную или поступательную пару. В большинстве случаев зубчатая передача служит для передачи вращательного движения. В некоторых механизмах эту передачу применяют для преобразования вращательного движения в поступательное.

Зубчатые передачи – наиболее распространенный тип передач в современном машиностроении и приборостроении; их применяют в широких

диапазонах скоростей и мощностей.

Еще до нашей эры люди начали применять рычаги в строительном деле. Например, при постройке пирамид в Египте. Рычагом называют твердое тело, которое может вращаться вокруг некоторой оси. Рычаг – это необязательно длинный и тонкий предмет. Например, колесо – тоже рычаг, так как это твердое тело, вращающееся вокруг оси.

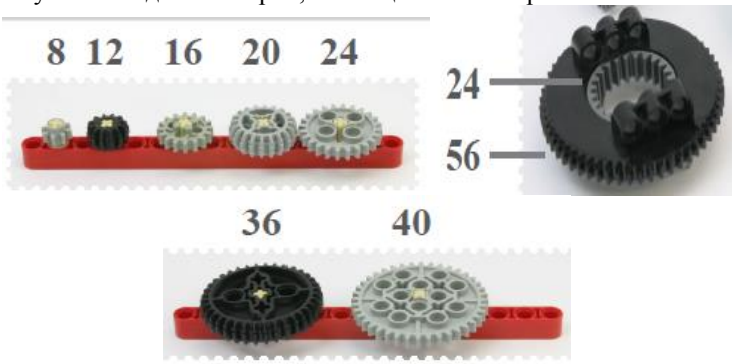
При помощи рычага можно маленькой силой уравновесить большую силу. Рассмотрим, например, подъем ведра из колодца. Рычагом является колодезный ворот – бревно с прикрепленной к нему изогнутой ручкой. Ось вращения ворота проходит сквозь бревно. Меньшей силой служит сила руки человека, а большей силой – сила, с которой ведро и свисающая часть цепи тянет вниз. При равновесии рычага плечо меньшей силы во столько раз больше плеча большей силы, во сколько раз большая сила больше меньшей:

Это можно отобразить в формуле – d1:d2 – отношение плеч сил F2:F1 – обратное отношение сил. Выведение соотношения между приложенными к рычагу силами, плечами этих сил, скоростями точек приложения сил. Выход на золотое правило механики и применение его в робототехнике.

При изучении темы «Передаточное отношение» у воспитанников возникают трудности по применению теоретических знаний на действующих моделях.

Порядок следующий:

Изучение видов шестерен, имеющихся в наборе ЛЕГО.



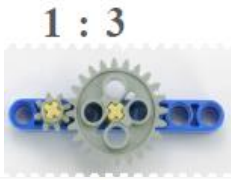
Разница в размере шестеренок влияет на угловую скорость вращения ведомой оси. Ведущая меньше ведомой – скорость уменьшается. Ведущая больше ведомой – скорость увеличивается.

Далее переходим к понятию передаточное отношение.

При всякой передаче существенную роль играет особая величина – передаточное отношение (а также передаточное число), которое надо научиться рассчитывать. Для этого необходимо знать число зубчиков на шестеренках при

зубчатой или цепной передаче и диаметр шкивов при ременной передаче.

Передаточное число – это отношение числа зубьев ведомой шестерни к числу зубьев ведущей.



На практике это выглядит следующим образом. Если одна (ведомая) шестерня имеет 24 зуба, а другая (ведущая) — 8, то передаточное число данной пары равно 3 (24:8).

$$i = i_{12} = \frac{\omega_1}{\omega_2} = \frac{n_1}{n_2}$$

n1 и n2 – отношение частоты вращения ведомого (n1) и ведущего элемента (n2) механической передачи. w1 – угловая скорость ведомого элемента, w2 – угловая скорость ведущего элемента механической передачи.

Следует обязательно обратить внимание учащихся, что направление вращения ведомой шестерни противоположно направлению вращения ведущей и если ведущая шестерня меньше ведомой – скорость будет уменьшаться, а когда ведущая шестерня больше ведомой – скорость будет увеличиваться. Увеличение скорости приводит к потере мощности и наоборот увеличение мощности ведет к потере скорости. Характеристика **передаточное отношение** применима как к механической передаче с одной ступенью (одной

кинематической парой) так и к механическим передачам с множеством ступеней при этом **передаточное отношение** всей механической передачи будет равно произведению всех передаточных отношений всех ступеней

3 – ступени

При выполнении творческих работ, таких как подъемный кран, робот-администратор, манипулятор и др., необходимо зафиксировать робота в определенном положении. В таком случае необходимы знания о таком виде передачи – как червячная.

Червячная передача – это зубчато-винтовая передача движение, в которой осуществляется по принципу винтовой пары. Червячные передачи применяют для передачи вращательного движения между валами, у которых угол скрещивания осей обычно составляет 90^0.

Редуктор

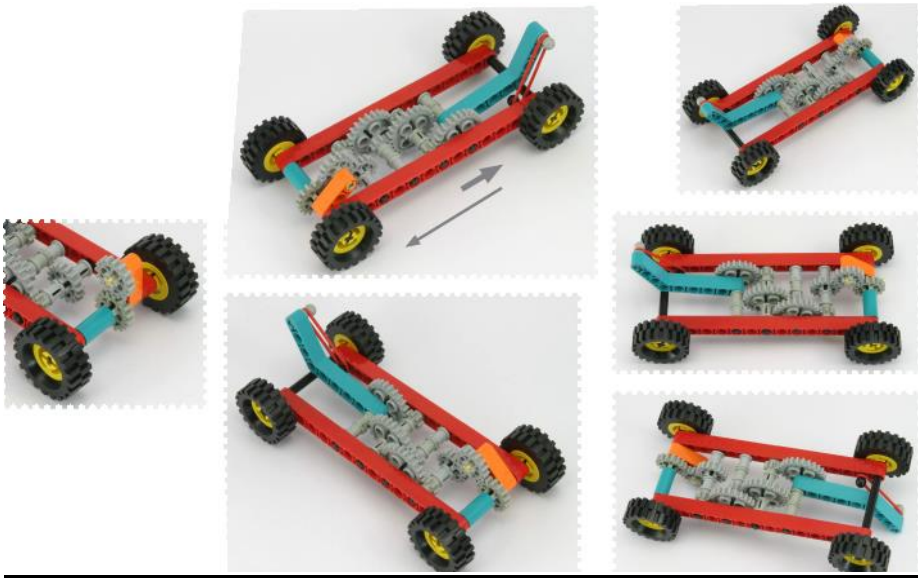
Этот механизм используется совместно с двигателями для преобразования и передачи крутящего момента.

Редуктор – механизм, передающий и преобразующий крутящий момент, с одной или более механическими передачами. Основные характеристики редуктора – КПД, передаточное отношение, передаваемая мощность, максимальные угловые скорости валов, количество ведущих и ведомых валов, тип и количество передач и ступеней.

Редуктор чаще служит для понижения частоты вращения и повышения крутящего момента вместе с тяговой силой.

При прохождении данной темы, необходимо выделить больше часов, т.к. при создании редуктора для своей модели, у учащихся есть затруднения (недостаточно практики).

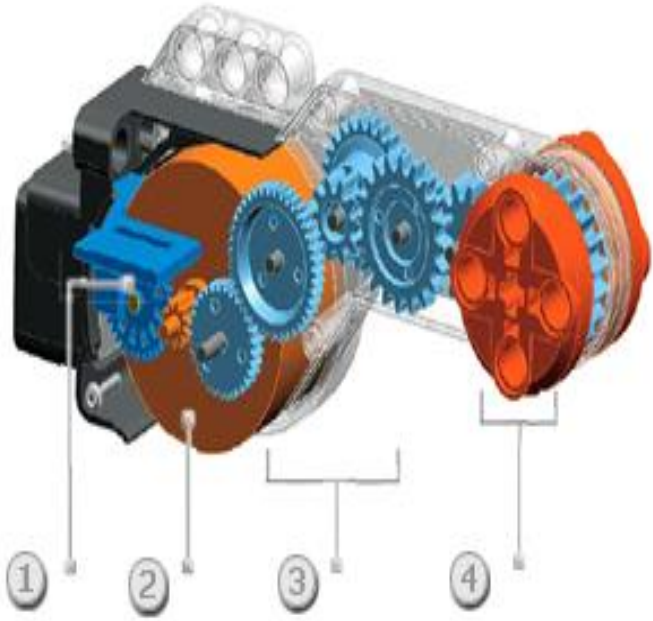
Примеры использования редуктора в тележке на резиномоторе.



Способы крепления шестерен к двигателю

Конструкция сервомотора

1. Тахометр датчика вращения.
2. Электромотор.
3. Встроенный редуктор.
4. Ступица колеса с отверстием под ось.

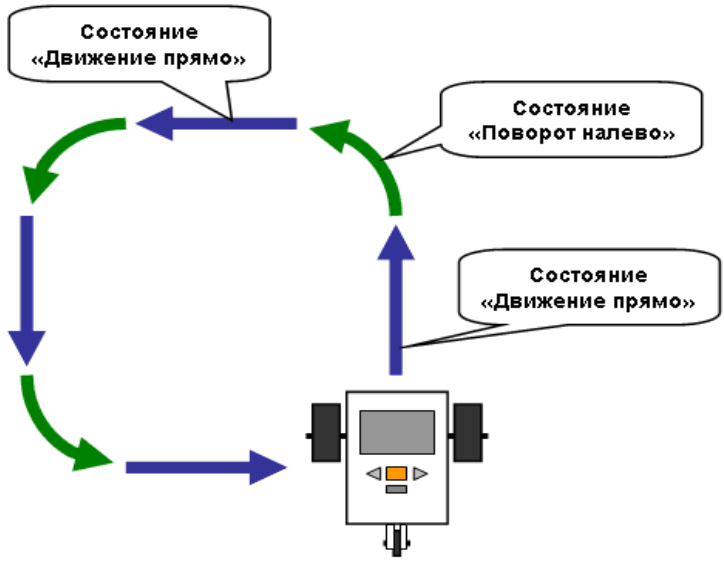


Методические рекомендации по решению задач для робота

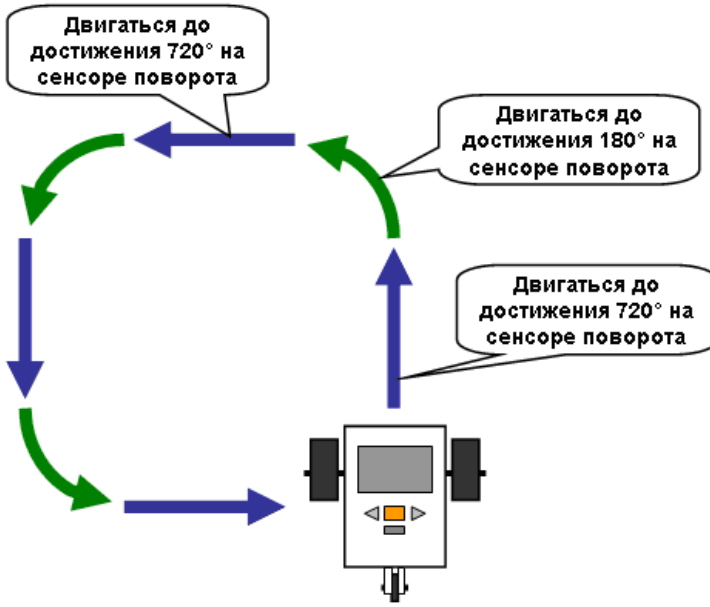
Что можно сказать про робота, который выполняет какую-то программу?

Если, например, в момент начала наблюдения, робот едет? Ответ, лежащий на поверхности, – робот двигается. Но, очевидно, этот же ответ не подойдет как общий для того момента времени, когда робот стоит. Поэтому, неплохим вариантом становится фраза «робот находится в каком-то состоянии».

Таким образом, поведение робота – это череда его состояний. Например, движение по квадрату – сначала робот в состоянии движения прямо, затем, в состоянии поворота налево, которое сменяется опять, состоянием движения прямо и т.д.

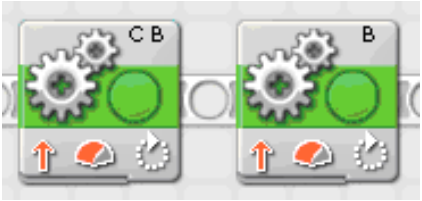


Что заставляет робота изменить свое состояние? Если рассматривать пример про движение по квадрату, то смена одного состояния на другое происходит после прохождения всем роботом определенной дистанции, в случае движения прямо, или после прохождения определенной дистанции одним колесом, в случае поворота. Здесь, в обоих случаях, пройденная дистанция определяется сенсором поворота оси двигателя. Иначе можно сказать, что робот сменил направление движения (состояние) после наступления определенного события на сенсоре поворота, в данном случае событие – сигнал от сенсора о достижении определенного угла поворота оси.



Теперь, если взглянуть на всю программу в целом, то она будет выглядеть как изменение состояния робота в зависимости от наступившего события.

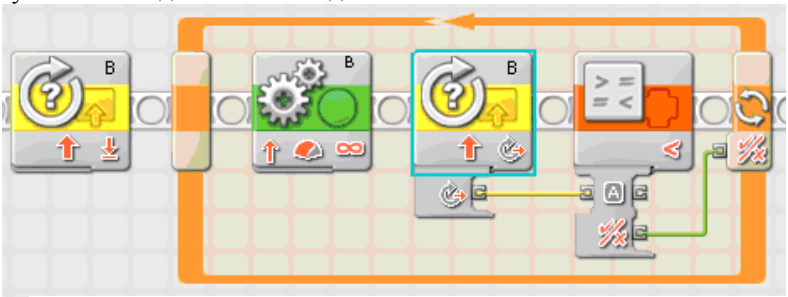
Как же эта диаграмма будет отображена в программу на NXT-G? Например, так:



Как видно, программа состоит из блоков движения. Но где же тогда в данной программе обработчики событий? Дело в том, что каждый блок движения, в котором явно задается количество необходимого движения в градусах, поворотах или секундах, содержит уже обработчик, направленный на определение возникновения перечисленных выше событий:

Иначе, такой блок можно было бы описать целой последовательностью: «Ехать бесконечно до тех пор, пока ось мотора не сделает поворот в 720 градусов»:

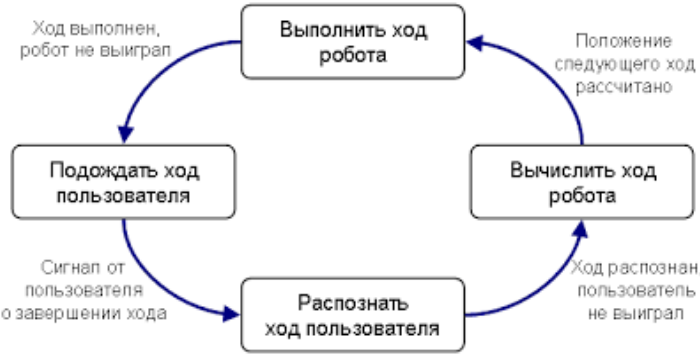
Или более сложный вариант, раскрывающий, что же действительно подразумевается под блоком «ожидание»:



Как показывает этот довольно простой пример, даже самая простая программа состоит из череды состояний и ожиданий событий. Любая. Будь то просто вывод на экран: состояние вывода на экран букв текста, пока не

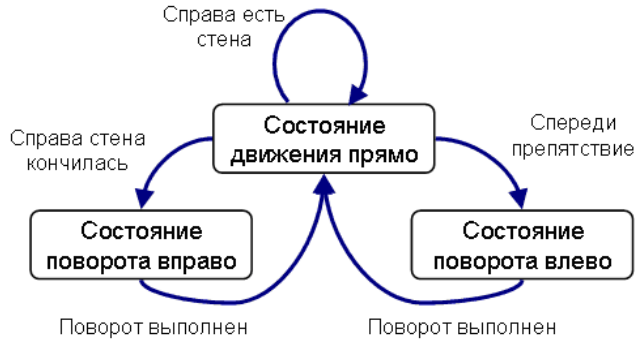
наступит событие «конец текста», или сложная программа игры в крестики-нолики.

Причем со сложными программами дело обстоит гораздо интереснее. Сложную программу можно сначала разбить на большие блоки состояние-действие, которые бы сменяли друг друга по наступлению каких-либо событий. В свою очередь, каждый блок можно было бы разделить на маленькие подблоки со своими событиями. И так далее. Пример, все те же «Крестики-Нолики». Крупные блоки:



Тогда блок «Выполнить ход робота» возможно представить в виде подблоков:

Здесь уже каждый подблок может быть представлен в виде конкретных действий языка NXT-G. Важно также отметить, что переходы из одного и того же состояния, при разных событиях, могут быть разные. Например, робот ищет выход из лабиринта с помощью обхода по принципу правой руки.



Как видно из схемы, существуют три разных события, на которые может реагировать робот в состоянии движения прямо:

- если справа все еще есть стена, продолжать движение прямо;
- если справа стена закончилась, перейти в состояние выполнения разворота вправо;
- если робот уперся в препятствие, перейти в состояние выполнения разворота влево.

Вывод: каждая программа состоит из череды состояний-действий и событий, определяющих чередование состояний. В то время как маленькие программы сразу состоят из неделимых состояний-действий, большие программы могут быть описаны в виде больших блоков – макро-состояний, переход к которым выполняется тоже при определении возникновения какого-то события. Макро-состояния в свою очередь могут разбиваться на подблоки, «связанные» своими состояниями и т.д.

Задание для робота и решение задачи

В середине круга, ограниченного черной линией, находятся две полные 330мл. банки содовой. Ваша задача – создать и запрограммировать робот таким образом, чтобы он вынес банки из круга, после чего проиграл мелодию и остановился.

В Вашем распоряжении находится один набор LEGO MINDSTORMS NXT 2.0, который, по неизвестным причинам, не содержит никаких других датчиков, кроме датчика касания, датчика цвета, ультразвука.

С чего начать?

Планирование – основной принцип, если Вы хотите получить хороший результат. Исходной точкой должно быть продумывание и определение шагов, которые необходимо предпринять. Не забывайте, что любая задача с вовлечением в нее роботов состоит из двух разных, хотя и тесно связанных,

частей: аппаратной (робот непосредственно) и программной (программа). Каждая из этих частей обусловливает другую, потому в нашем случае использования метода проб и ошибок нужно принимать во внимание обе части.

Имеем следующие задачи:

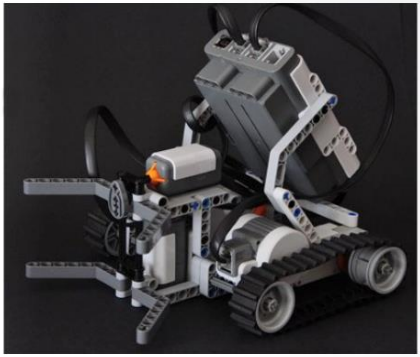
1. Разработка и конструирование робота.
2. Написание программы (с использованием NXT-G путем соединения блоков). Для этого сначала необходимо написать алгоритм, а затем уже преобразовать его в программу.
3. Тестирование программы.
4. Внесение коррективов.

Робот

Необходимо помнить об ограничениях (в конструировании и программировании) этой задачи и возможностей робота. Они следующие:

Ограничения

- Робот может использовать только датчик касания и датчик цвета/света. Нет ограничений на размер или количество деталей, которые он должен переместить и повернуть на игровой площадке.
- Робот должен обнаружить черную линию, ограничивающую область, внутри которой он обязан находиться.
- Робот должен определить, когда сталкивается с одной из банок, и переместить ее за пределы поля.
- Робот должен знать, когда вынес обе банки из поля. При создании и тестировании робота часто возникают сложности, которые требуют модификаций. Например, одна из сложностей, с которой ранее сталкивались, связана с датчиком цвета: он определял все как черное. После нескольких тестов и изменений стало понятно, что проблема заключалась в слишком близком расположении датчика к поверхности пола, таким образом, не было пространства для отражения света в датчик. Перемещение датчика немного выше решило проблему.



Алгоритм

Прежде чем начать комбинировать блоки кода, составляющие программу, было бы неплохо записать ее на нормальном языке, то есть, записать шаги, которые должен сделать робот для выполнения его миссии. Не бывает одного-единственного решения для задачи, люди выполняют одни и те же задачи по-разному. Так после записи программы, подумайте об альтернативных способах достижения требуемой цели более эффективным способом.

В таком случае задачи программирования следующие:

1. Создайте счетчик и обнулите его (благодаря этому сохраняется число банок, которые были уже перемещены за пределы игрового поля).
2. Двигайтесь вперед до тех пор, пока робот не обнаружит черную линию или объект. Если робот обнаружит объект, продолжайте двигаться вперед, пока он не определит черную линию (означает, что объект находится теперь вне игрового поля), добавьте единицу к счетчику.
3. Остановиться, развернуться и вернуться назад.
4. Если счетчик достигает значения 2, проиграть мелодию и остановиться. В противном случае продолжить поиск.

Это только начальная точка для совершенствования программных навыков.

Программа

Теперь Вы знаете, какие шаги должен предпринять робот для выполнения своей миссии, но не рекомендуется сразу запускать всю программу. Это очень полезная стратегия разбивать сложные задачи на более легкие. В этом случае Вы можете разделить программу на две части и объединить их позже.

1. Робот произвольно перемещается в пределах игрового поля, не покидая его.
2. Робот перемещает банку за пределы поля, возвращается назад и проигрывает мелодию.

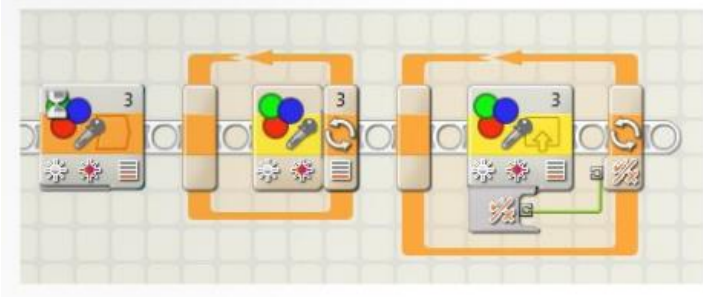
Задание: часть 1

Прежде, чем записать программу, рассмотрим алгоритм:

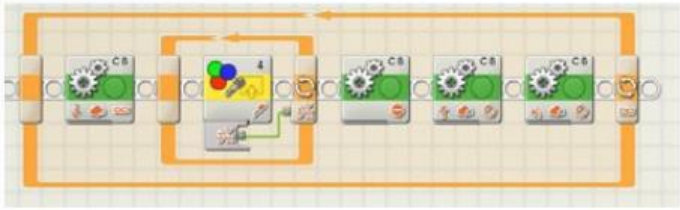
1. Перемещение вперед до тех пор, пока датчик цвета определяет черный цвет.
2. Остановка.
3. Возвращение назад.
4. Поворот.
5. Повтор шагов 1-4.

Ожидание робота, пока датчик не считает черный цвет, можно задать с использованием любой из следующих трех опций. Применение каждой из них

даст тот же самый результат, но третья опция открывает путь к объединению частей задачи.



Программа будет следующей:



Протестируйте программу и удостоверьтесь, что ее поведение соответствует ожиданиям.

Задание: часть 2

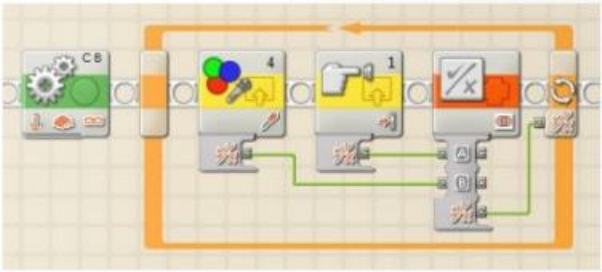
Чтобы сделать задачу проще, предположим, что робот имеет дело с одной из банок (таким образом, ему не нужно ее искать), что требует более легкого алгоритма, например, как этот:

1. Перемещение вперед, пока не произойдет столкновение с банкой.
2. Продолжение перемещения, пока не достигнете черную линию на краю игрового поля.
3. Остановка.
4. Возвращение назад.
5. Проигрывание мелодии.

Полная программа

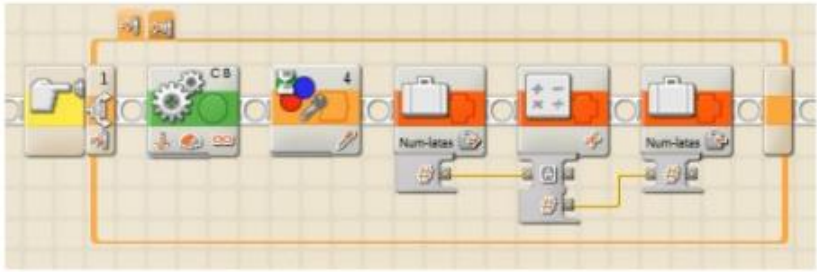
Если две части задачи по отдельности работают, как ожидалось, пора объединить их. Создадим переменную, с помощью которой будем фиксировать число банок, перемещенных за пределы поля. Переменная создается опцией. Определить Переменную в меню Инструменты и задается с начальным значением 0 таким образом:

Сначала проверьте, можно ли контролировать два датчика одновременно и принимать решения на основании их показаний (шаг 2 алгоритма).

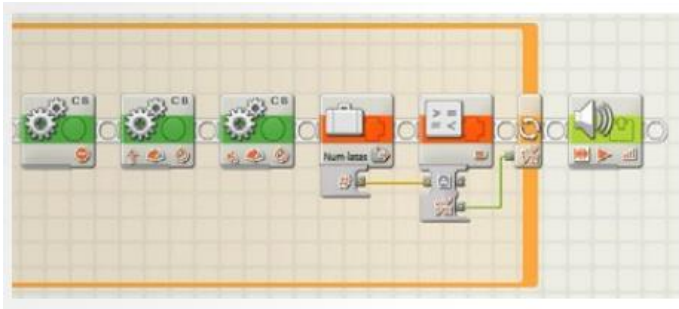


Фрагмент кода на изображении начинается с блока перемещения для приведения робота в движение прежде, чем запустить цикл, который повторяется до тех пор, пока или датчик цвета считывает черный цвет линии, или датчик касания зажат (вступает в контакт с банкой). Цикл непрерывно считывает показания двух датчиков и посредством логической операции (в этом случае OR), комбинирует оба показания, генерируя логическое значение, которое является истинным, если происходит контакт с банкой или обнаруживается черная линия, либо оба фактора имеют место одновременно.

Робот должен знать, достиг ли он границы игрового поля или коснулся банки. С этой целью можно использовать условие, которое будет выполняться только, если датчик касания был активирован. В этом случае робот будет перемещать банку до тех пор, пока не достигнет черной линии, затем добавит единицу к счетчику (шаг 3).



После остановки, возвращения и поворота робота (шаг 4) необходимо проверить, сколько банок было перемещено за пределы игрового поля. Таким образом, программа проверяет значение счетчика и, если оно равно 2, завершает цикл, после чего робот проигрывает мелодию и останавливается (шаг 5).



Тестирование робота

Теперь можно протестировать робота и оценить результаты. Если они не будут соответствовать ожиданиям, то необходимо оценить, нужно ли вносить изменения в конструкцию робота или программу.

Рекомендация! Не делайте одновременно слишком много изменений, поскольку это затрудняет определение того, какие изменения улучшают или ухудшают поведение робота.

Как продолжать???

На данном этапе можно провести следующие модификации:

1. После достижения черной линии робот возвращается назад и поворачивается.
2. При наличии ультразвукового датчика робот не будет перемещаться вслепую. Он в состоянии определить местоположение банок, не касаясь их.

Замените банку на что-то более тяжелое (на большую банку или что-то подобное). Это интересное занятие вроде соревнования по сумо с мертвым весом.

Устройство и применение сенсоров

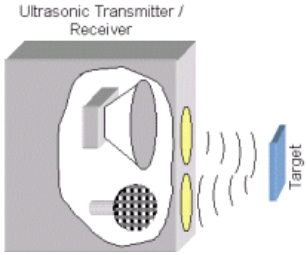
Сенсоры (датчики) являются элементом роботов, предназначенные для измерения, сигнализации, регулирования, управления устройствами или процессами.

Датчики преобразуют контролируемую величину (скорость, освещенность, звук, касание) в сигнал (электрический, оптический,) удобный для измерения.

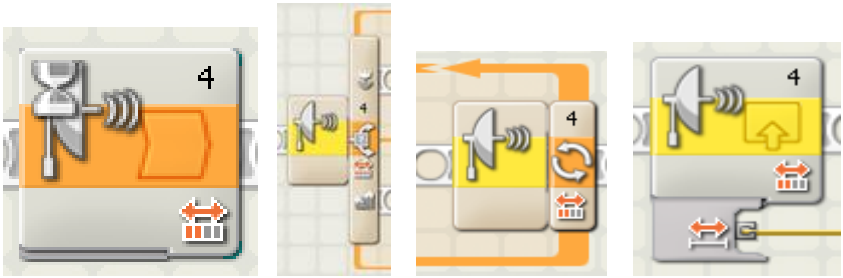
В данном разделе рассмотрим принцип работы сенсора расстояния и освещенности, чаще всего применяемые в ЛЕГО роботах.

Сенсор расстояния

По своему физическому устройству его можно представить как передатчик и приемник: передатчик испускает волну в ультразвуковом диапазоне, волна, отраженная от удаленной поверхности, улавливается приемником.



При написании программ для NXT это означает, что передатчик испускает звуковую волну после того как, программа обратиться к сенсору расстояния.



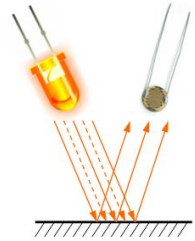
Поскольку путь до препятствия и обратно у звуковой волны займет какое-то время, программа получит данные с сенсора не сразу, а как минимум через это самое время. И в течение него программа будет заблокирована на операции опроса, до тех пор, пока данные с сенсора не придут. Поэтому, чтобы избежать неприятностей с программой, необходимо помнить и учитывать эту задержку.

Сенсор освещенности (или цветовой сенсор) из набора Lego Mindstorms NXT, один из наиболее используемых сенсоров при конструировании и программировании Lego-роботов.

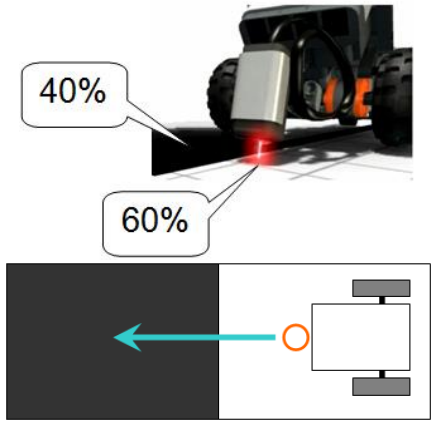
По внутреннему устройству, он не такой сложный как сенсор расстояния. Основным элементом в нем является светочувствительный элемент (фоторезистор или фототранзистор).

В режиме измерения окружающей освещенности, количество света, попавшее на светочувствительный элемент, преобразуется в цифровое значение, которое уже используется в программе. Например, с датчиком, работающем в этом режиме, можно собрать робота, который ищет самое освещенное место в комнате.

В режиме измерения отраженного цвета, помимо светочувствительного элемента, активируется светоиспускающий элемент (светодиод). Свет, выпущенный этим элементом, отражается от какой-нибудь поверхности и попадает обратно в светочувствительный элемент.



В зависимости от того насколько светлая отражающая поверхность, в светочувствительный элемент приходит больше света. Это количество света преобразуется в цифровое значение и передается в программу. Чем темнее поверхность, тем меньше света приходит – в программу приходят маленькие значения; чем светлее поверхность, тем больше света приходит – программа оперирует с большими значениями. Допустим, необходимо написать программу, которая перемещает робота из светлой в темную область на следующей карте:



Перед началом программирования, необходимо провести калибровку сенсора освещенности. После чего, измерить, что показывает сенсор на разных частях карты светлой и темной части карты. Пусть после калибровки, показания сенсора будут 30% на темной стороне и70% процентов на светлой. Следовательно, условием, когда можно рассматривать, что робот уже на темной стороне – значение на сенсоре стало меньше 50%.

Тогда программа может быть сформулирована следующим образом:

1. Начать движение прямо (поскольку неизвестно, сколько нужно проехать работу до темной области, нужно задать "бесконечное" движение).
2. Ехать до тех пор, пока значение на сенсоре не станет меньше 50%.
3.Остановиться.

На NXT-G программа будет выглядеть, следующим образом:

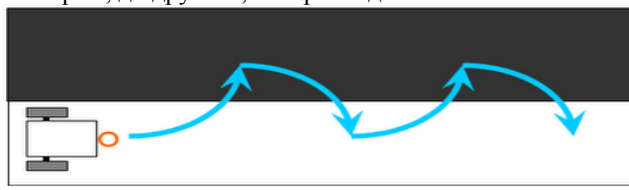
Давайте, усложним программу, чтобы робот возвращался еще и назад, на белую область. Условием, что робот уже на светлой стороне, являются показания сенсора большие 50%.

1. Начать движение прямо
2. Ехать до тех пор, пока значение на сенсоре не станет меньше 50%.
3. Остановиться.
4. Начать движение назад.
5. Ехать до тех пор, пока значение на сенсоре не станет больше 50%.
6. Остановиться.

Теперь, сделаем так, чтобы робот ездил поочередно со светлой стороны на темную, не останавливаясь. Для этого добавим, блок Цикл, а программу, приведенную выше, поместим в него.

У этой программы есть очень интересное свойство: **даже если перед ее запуском робот поставлен на черную область карты, робот сразу же поедет назад.**

Это связано с работой пары блоков Движение и Ожидание. Как только робот поедет (после запуска моторов блоком Движение), первый блок Ожидания сразу же обнаружит, что значение освещенности очень мало (робот уже на черной области), тогда движение вперед остановится и начнется движение назад. Все это произойдет практически мгновенно и создастся ощущение, что робот понял, что ему вперед ехать не надо, и поехал сразу назад. А насколько изменится программа, если условие задачи еще немного изменится? Допустим, карта теперь также состоит из двух частей: темной и светлой, но размеры этих частей увеличились и роботу теперь нужно добраться от одного края карты, до другого, совершая движения «ёлочкой».



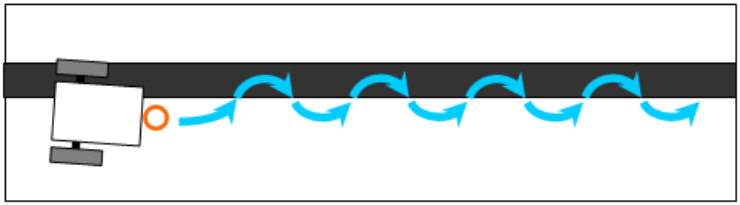
Видно, что характер самого движения не изменился – робот опять должен двигаться поочередно из области одного цвета в другой. Однако, изменилось направление движения, раньше робот у нас ездил вперед и назад, сейчас же движение должно быть все время вперед, только меняется направление поворота. Итак, программа должна состоять из следующих действий:

1. Начать поворот налево.
2. Ехать до тех пор, пока значение на сенсоре не станет меньше 50%.
3. Остановиться.
4. Начать поворот направо.

5. Ехать до тех пор, пока значение на сенсоре не станет больше 50%.
6. Остановиться.
7. Повторить действия, начиная с шага

Поскольку точных размеров поля не известно, нельзя сказать, сколько шагов «елочкой» роботу нужно сделать, поэтому количество повторов (итераций) цикла не ограничивается.

Если кто-то уже попытался применить программы, описанные здесь, к реальному роботу на реальной карте, то тот заметил, что робот останавливается не на середине белой и черной области, а недалеко (почти) границе, где одна область переходит в другую. Поэтому повороты робота длятся крайне маленький промежуток времени и складывается впечатление, что робот не ездит с черной области на белую, а движется по их границе. Все верно! Это поведение умышленно не изменялось, чтобы программа с минимальными изменениями стала работать на следующей карте:

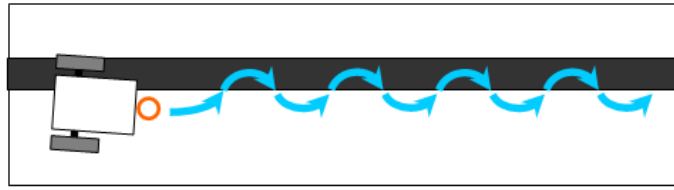


По своей сути ни характер движения, ни его направление не поменялись по сравнению с предыдущей задачей. Робот все также двигается вперед попеременно с черной на белую область, выполняя развороты то вправо, то влево. Единственное, в чем произошло отличие, так это в том, что черная область теперь не большой прямоугольник, а узкая полоска (линия) черного цвета. А поскольку робот будет двигаться вдоль кромки этой линии, то со стороны будет казаться, что он **движется вдоль линии.**

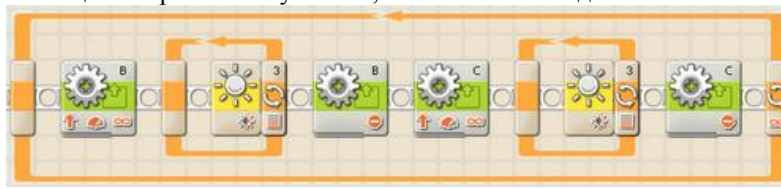
Если попробовать запустить робота с программой, приведенной выше, на новой карте. То, возможно, на некоторых частях линии (например, где линия выполняет резкий поворот) робот будет терять ее. Поэтому внесем небольшие изменения в программу так, чтобы робот выполнял каждый поворот только одним колесом, второе бы в этот момент бездействовало – это сделает «шажки» робота еще более маленькими и уменьшить вероятность потери линии.

Алгоритм движения робота вдоль линии является базовым алгоритмом при изучении робототехники, потому что отражает суть программирования роботов: робот выполняет основную задачу (движение), меняя свое поведение в зависимости от изменений окружающих условий (черные и белые области). Поэтому задачи, основанные на движении вдоль линии, наиболее часто встречаются на всевозможных робототехнических состязаниях и олимпиадах. Естественно, приведенная выше реализация небезупречна. Основной ее недостаток – итоговая скорость движения робота вдоль линии. Можно сказать, что эта реализация базовая – с нее стоит начать изучение этого сложного алгоритма. К тому же, она позволяет довольно быстро (маленькое количество действий в программе) и с минимальным количеством деталей (два мотора и один светочувствительный сенсор) показать, на что способен робот.

Можно напомнить, что при запуске тележки, работающей по этому алгоритму, характер движения будет следующий:

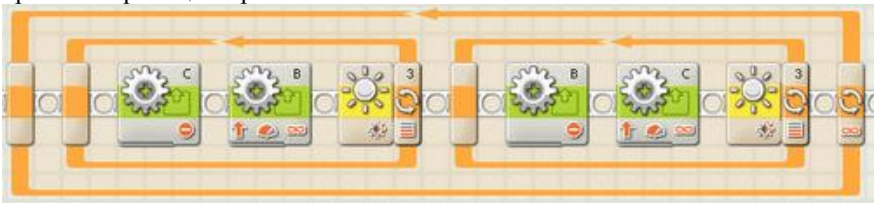


Этот же алгоритм можно записать слегка другим способом. Для этого, вместо каждого блока «ожидание», будет стоять эквивалентный цикл, оканчивающийся при том же условии, что и блок «ожидание».



Затем, чтобы отразить смысл процесса, блоки управления моторами

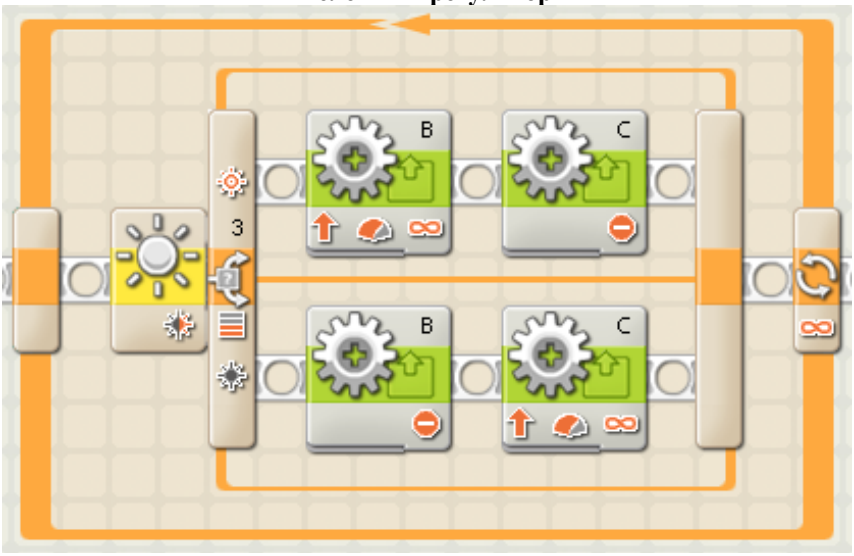
будут перенесены внутрь цикла. Таким образом, становится однозначно понятно, что каждый маленький цикл отвечает за движение с одной или другой стороны от границы черной линии.



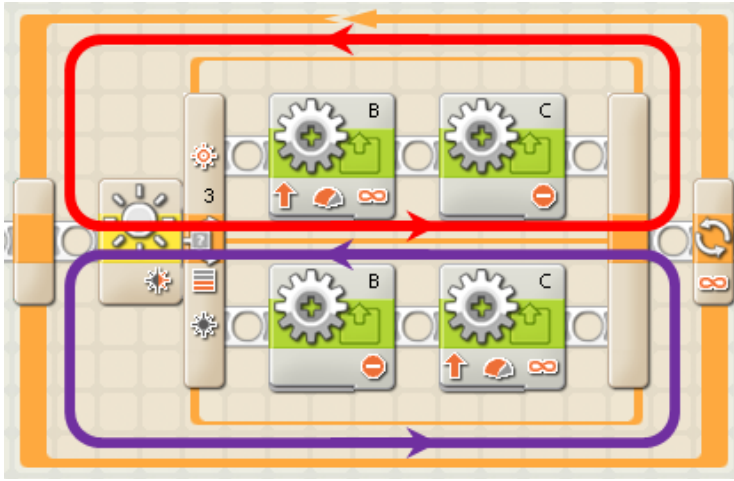
Словами этот алгоритм может быть описан следующим образом: повторяем движение одним мотором до пересечения границы черной линии, после чего повторяем движение уже другим мотором до обратного выезда за границу.

Данная реализация алгоритма может быть легко переделана к более узнаваемой форме – релейный регулятор с двумя состояниями:

Релейный регулятор



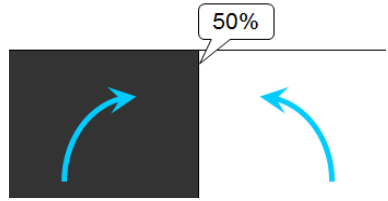
Следует отметить, что изменилась только запись – сам алгоритм остался тот же, причем сохранились не только основные элементы, отвечающие за операции с моторами, но и, при внимательном просмотре, два цикла из стоящего выше примера, с точкой выхода на
границе линии.



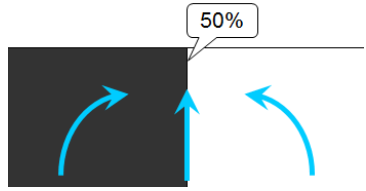
Зачем нужна такая форма записи алгоритма? Дело в том, что от нее очень логично перейти к более сложной теме – пропорциональным регуляторам, о которых будет рассказано в следующих частях.

Робот, двигающийся вдоль линии, по алгоритму, рассмотренному выше, не может похвастаться особой скоростью. Явно это видно на прямых участках, где тележка, вместо того, чтобы ехать вперед двумя колесам, все равно совершает постоянный поиск границы линии, двигаясь, то одним, то другим колесом.

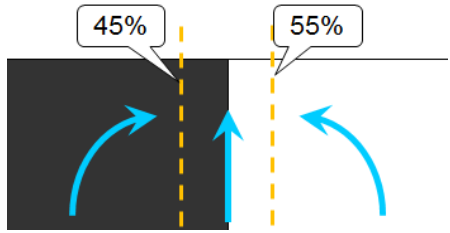
Т.е. очевидно, что если бы в алгоритм добавить движение прямо при определенных условиях, то общая скорость движения значительно бы возросла. Как же это сделать? Ниже представлена диаграмма, показывающая, как изменяется характер движения тележки в зависимости от того, с какой стороны границы находится датчик света. На границе линии датчик показывает 50% освещенности.



Если продолжать мысль о движении прямо, то на диаграмме ее можно выразить, следующим образом.

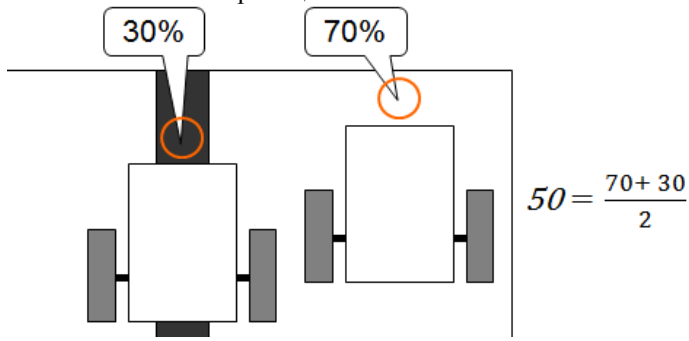


Иными словами, когда тележка находится прямо над границей линии, то она должна двигаться вперед. На практике можно сказать, что ситуация, когда датчик света будет показывать ровно 50%, очень редки, поэтому робот практически никогда не будет двигаться прямо, как бы ровно он не стоял относительно границы. Поэтому необходимо сделать допущение – выбрать небольшой диапазон значений, которые робот будет расценивать как «нахожусь прямо над границей».

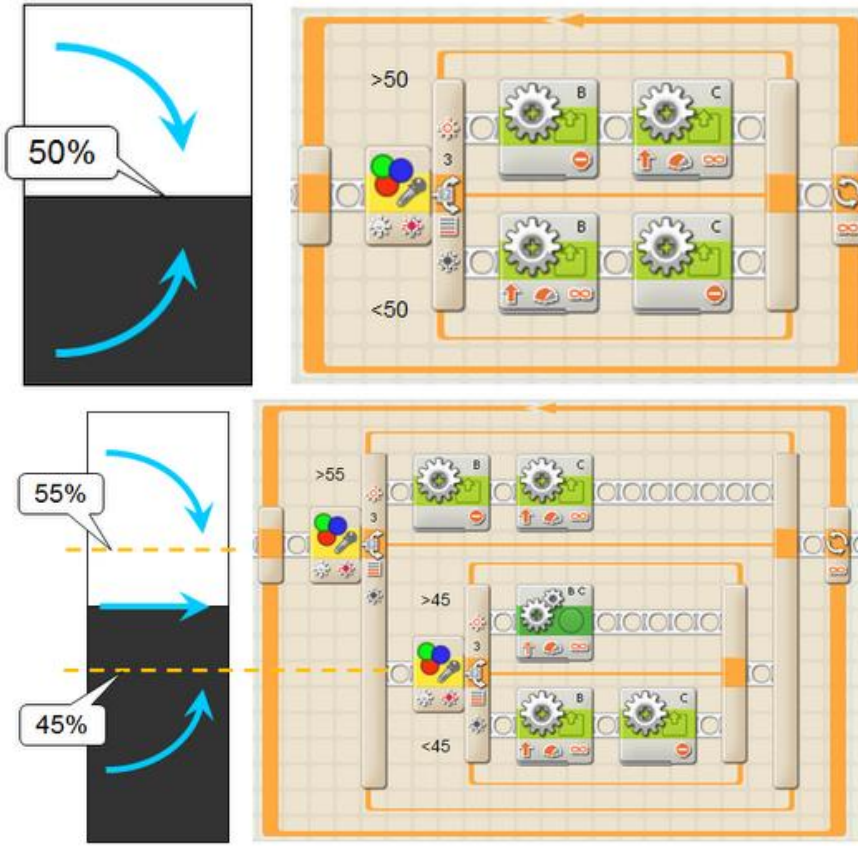


45% и 55% процентов взяты произвольным образом по принципу «чуть-чуть меньше и чуть-чуть больше». Важно помнить, что для каждого отдельного датчика и для каждых новых условий освещенности данные значения будут разные, но все они будут зависеть о того, что показывает датчик на границе линии.

Здесь, кстати, выглядит разумным не терять время, выверяя точно границу, а взять показания датчика на самом темном участке линии и где-нибудь на белом поле. Среднее арифметическое от этих значений и можно будет считать показанием на границе линии.



Как теперь изменится алгоритм поведения робота? Если раньше в нем было ветвление на две ветки, то теперь появляется третья. Легче это представить, если опять нарисовать схемы:



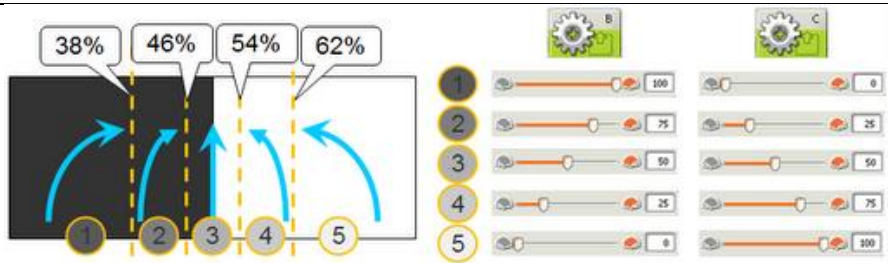
Иными словами в новом алгоритме нижняя часть ветвления в свою очередь тоже ветвится, и таким образом достигается три различных вида движения.

В данном алгоритме для поиска оптимальной скорости перемещения вдоль линии можно изменять мощности на моторах во время поворотов и во время движения прямо. При этом общее правило примерно следующее: скорость движения по прямой меньше скорости поворотов – чтобы робот не успел на большой скорости «вылететь» за трассу, после чего он мог бы не найти линию вообще. Для каждой конфигурации трассы и для каждой конструкции робота данные скорости индивидуальны: если трасса имеет много

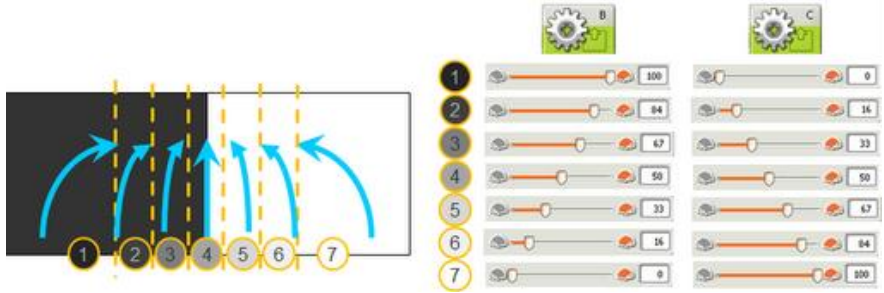
резких поворотов или робот обладает очень большими колесами - мощность на моторы будет меньше, чем в случае когда трасса более предсказуемая или колеса робота значительно меньшего диаметра.

Можно продолжить улучшение и разбить каждый из поворотов на два – в первом случае тележка двигается только одним колесом, а другое стоит; во втором случае оба колеса двигаются, но только скорость одного выше скорости другого.

Таким образом, диапазон всех возможных показаний сенсора разбивается теперь не на три, как в прошлый раз, а на пять участков.



Следующая диаграмма показывает, что будет происходить с мощностью двигателей в алгоритме, диапазон которого разбивается на семь участков.



И ведь так можно продолжать довольно долго - разбивать диапазон все на более мелкие участки: на 9, 11, 13 и т.п. Во всех этих случаях будет наблюдаться следующая тенденция: на одном из моторов мощность будет расти с увеличением показаний на световом сенсоре, в то время как на другом моторе мощность будет уменьшаться.

Иными словами:

- чем больше освещенность, тем больше мощность на моторе X, но меньше - на моторе Y
- чем меньше освещенность, тем меньше мощность на моторе X, но больше - на моторе Y.

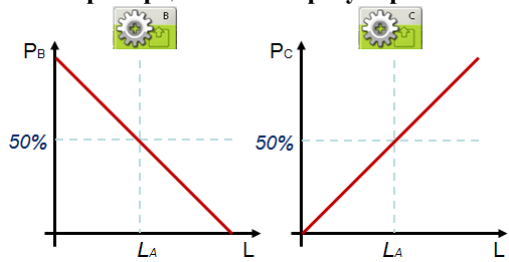
Причем, мощность все время увеличивается (или уменьшается) на

одинаковые пропорции.

- При освещенности 10, мощность будет V
- а при освещенности 20, мощность будет 2V

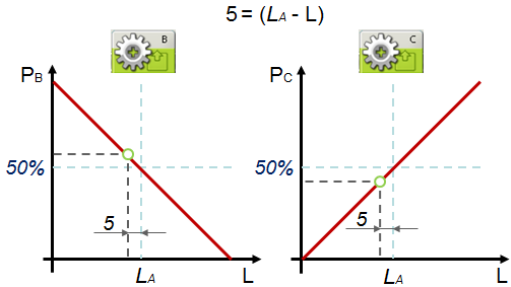
Т.е. мощность ПРОПОРЦИОНАЛЬНО зависит от освещенности.

Пропорциональное регулирование



Как на предыдущих диаграммах, где были явно показаны мощности, так и на графике выше видно, что существует некоторая средняя освещенность (*LA*), при которой на оба мотора подается одинаковая мощность – 50%, т.е. оба колеса едут с одинаковой скоростью. Отклонение от этой средней освещенности (*LA* -L, где L – текущее показание освещенности) будет приводить к изменению мощности на обоих моторах. И как это видно из графиков, чем больше это отклонение, тем больше отличаются мощности моторов от 50%. Именно поэтому, данное отклонение и используется, чтобы определять с какой мощностью должны вращаться моторы.

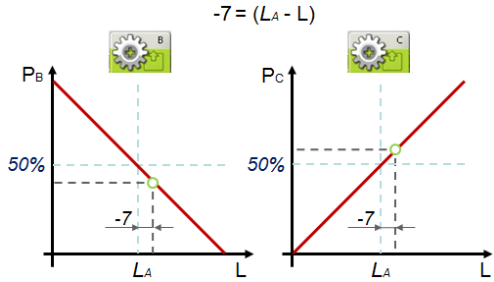
Вот, например, пусть показание на сенсоре показывают освещенность чуть меньше средней.



Тогда мощность на моторе B возрастет на соответствующую величину, а мощность на моторе C, наоборот, на эту величину уменьшится.

$$P_B = 50 + 5 \qquad P_C = 50 - 5$$

Теперь рассмотрим отклонение в другую сторону – показание на сенсоре больше средней освещенности. Формула вычисления отклонения не изменяется.



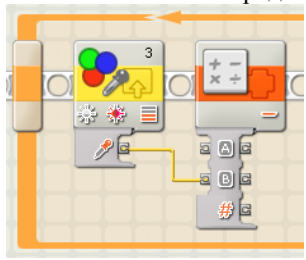
При использовании такого же подхода, как и в предыдущем случае, будет видно, что здесь мощность на моторе B упадет, а на моторе C увеличиться.

$$P_B = 50 + (-7) = 50 - 7 \qquad P_C = 50 - (-7) = 50 + 7$$

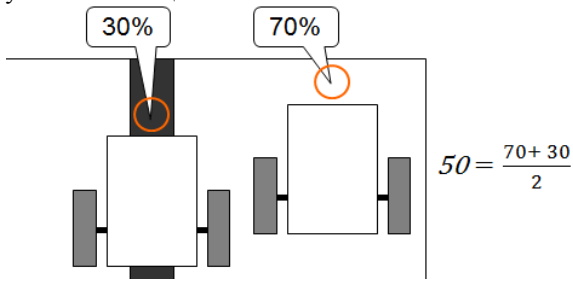
В итоге, получаются следующие формулы, которые позволяют в математической форме записать отношение между освещенностью и мощностью на моторах тележки.

$$P_B = 50 + (L_A - L) \qquad P_C = 50 - (L_A - L)$$

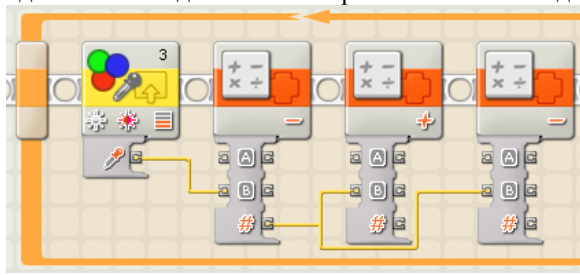
Программа на языке NXT-G будет выглядеть следующим образом: Сначала необходимо посчитать отклонение от средней освещенности.



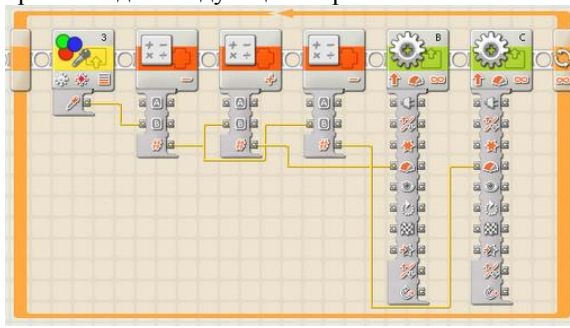
Следует напомнить, что средняя освещенность – это разность показаний на самом темном и самом светлом участках поля. Она будет индивидуальная для каждого робота и каждого поля – это нужно замерять для каждых новых условий освещенности заново.



Теперь посчитаем мощность для каждого мотора: в одном случае вычтем из 50, в другом прибавим к 50. 50 в данном случае мощность, при которой колеса двигаются с одинаковой скоростью - тележка едет прямо.



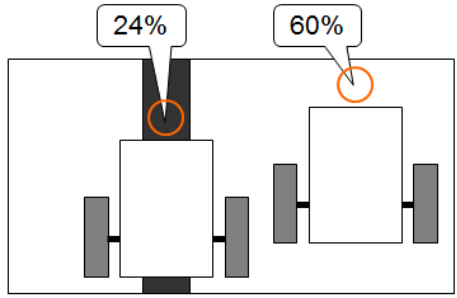
Остается подать вычисленную мощность на каждый из моторов. Вся программа, теперь выглядит следующим образом.



Когда программа будет запущена на реальном роботе, то в большинстве случаев будет заметно, что тележка не ведет себя полностью так, как ожидалось. Ведь судя по диаграммам, приведенным выше, в поведении моторов должен наступать такой момент, когда один из них должен целиком остановиться, в то время как другой должен ехать с максимально возможной скоростью. Этого не происходит, не останавливается (правда это еще зависит от заряда батареи – описываемое поведение справедливо для полностью заряженных батарей). Это приводит к тому, что даже не на очень крутых поворотах робот теряет трассу.

Чтобы разобраться, почему так происходит, можно составить таблицу, показывающую предполагаемую мощность на моторах в зависимости от показаний датчика освещенности.

Для составления таблицы еще раз укажем формулы зависимости мощностей от показаний датчика, а также будем считать, что датчик на самом светлом участке поля (*Lmax*) показывает 60%, а на темном (*Lmin*) – 24%, таким образом получается, что ожидаемая освещенность на границе линии (*LA*) – 42%.

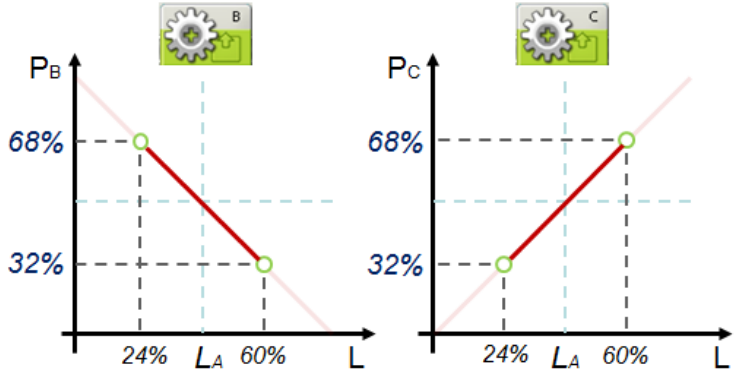


$$42 = \frac{60+24}{2}$$

$P_B = 50 + (L_A - L)$ $P_C = 50 - (L_A - L)$

L	$L_A - L$	B	C
42%	0	50	50
35%	7	57	43
30%	12	62	38
24% (min)	18	68	32
47%	-5	45	55
52%	-10	40	60
60% (max)	-18	32	68

Из этих данных видно, что в то время, как показания на датчике достигают своего минимума или максимума, возможного на этом поле, мощность моторов никогда не доходит до 100%, а также моторы никогда полностью не останавливаются. Вместо этого значения мощности не выходят за пределы диапазона от 32 до 68 процентов.



Следовательно, необходимо придумать механизм (преобразование), который бы позволил при отклонении в -18 или 18 получать терминальные значения мощности (0% и 100%). Для этого, запишем значения мощности в виде минимального и максимального необходимых значений: -50 и 50, с учетом того, что «средняя» мощность 50%. Т.е. при отклонении -18 мощность на моторе B должна упасть до 0% - измениться на -50 от средней мощности, а при отклонении 18 мощность на моторе B должна наоборот возрасти до 100% - измениться на +50 к средней мощности.

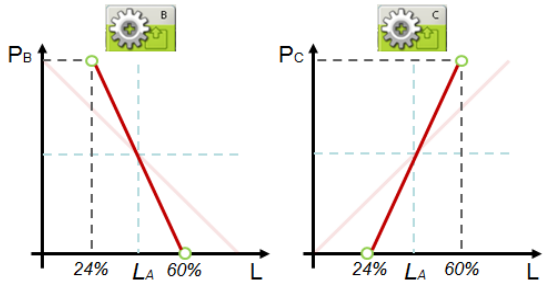
Поэтому преобразовать надо -18 в -50, или увеличить -18 в 2,78 раз, чтобы получить -50, и в тоже самое время для преобразования 18 в +50, надо 18 увеличить в эти же 2,78 раз. Необходимо напомнить, что 18 (или -18) - это отклонение текущей освещенности от освещенности на границе линии - *LA* - L. Таким образом, формулы определяющие зависимость мощности на моторах от освещенности теперь приобретут такой вид:

$$P_B = 50 + (L_A - L) \cdot 2,78 \qquad P_C = 50 - (L_A - L) \cdot 2,78$$

Где *LA* * 2,78 – называют пропорциональной составляющей линейного регулятора управляющего мощностью мотора в зависимости от показаний на датчике освещенности.

За счет этой небольшой модификации, значения мощностей, вычисленных по данным формулам, будут иметь уже вполне ожидаемый вид - изменяться от 0% до 100%. При 24% освещенности (*Lmin*) – колесо B будет вращаться с максимальной скоростью, а при 60% освещенности (*Lmax*) – полностью остановится.

L	L_A - L		
42%	0	50	50
35%	7	61	31
30%	12	83	17
24% (min)	18	100	0
47%	-5	36	64
52%	-10	22	78
60% (max)	-18	0	100



Значение «2,782» называют коэффициентом пропорциональной составляющей линейного регулятора. Он не будет одним и тем же во всех программах, поскольку напрямую зависит от диапазона значений датчика освещенности, которые он может выдавать на конкретном поле при расположении на определенной высоте над поверхностью поля и в зависимости от многих других параметров.

При условиях, подобных текущему, когда средняя мощность должна быть 50%, и моторы должны варьировать свою мощность от 0% до 100% (т.е. изменяться максимум на 50 от средней мощности), формула определения коэффициента пропорционального регулятора будет следующей.

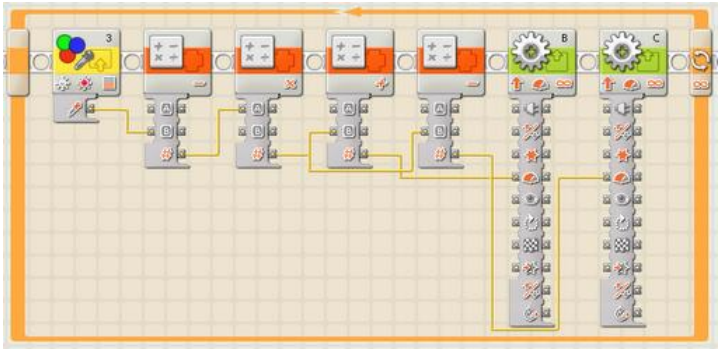
$$K_P = \frac{50}{\frac{L_{max}-L_{min}}{2}} = \frac{50 \cdot 2}{L_{max} - L_{min}} = \frac{100}{L_{max} - L_{min}}$$

Например, если диапазон значений датчика освещенности - от 10 до 80, то коэффициент будет 1,43; если от 40 до 90, то *KP* = 2; если от 30 до 50, то *KP* = 5.

Также следует записать в более общем виде формулу зависимостей мощностей.

$$P_B = 50 + (L_A - L)K_P \qquad P_C = 50 - (L_A - L)K_P$$

Если же говорить о программировании, то новая программа будет отличаться от предыдущей в незначительной степени - добавить лишь один общий блок, выполняющий операцию преобразования с помощью коэффициента пропорциональной составляющей.



Но прежде нужно понять, где же в предыдущих заметках была эта нудная и повторяющаяся работа. Для этого взглянем еще раз на формулы. Первая вычисляет коэффициент пропорциональной составляющей.

$$K_P = \frac{100}{L_{max} - L_{min}}$$

А вторая позволяет рассчитать мощность для каждого мотора.

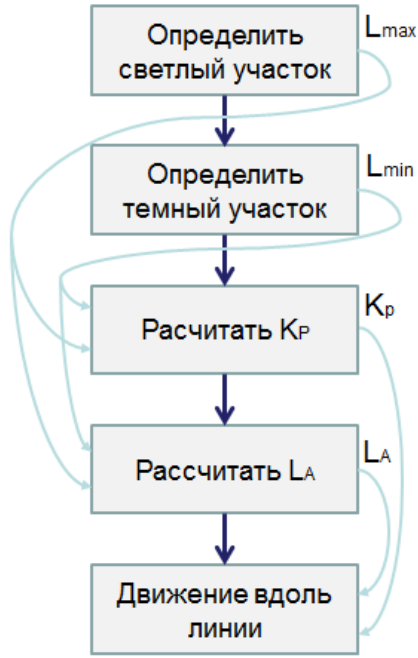
$$P_B = 50 + (L_A - L)K_P \qquad P_C = 50 - (L_A - L)K_P$$

Где, *LA* – предполагаемая освещенность на границе линии, которую можно определить по следующей, третьей формуле.

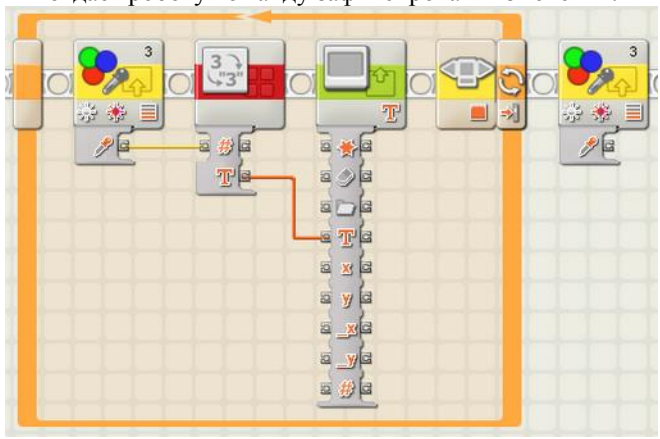
$$L_A = \frac{L_{max} + L_{min}}{2}$$

Можно заметить, что и *KP* и *LA* использующиеся в формуле для мощностей, зависят от уровней освещенности: на белой - *Lmax* и черной - *Lmin* частях поля. А это значит, что каждый раз когда робот попадает в другие условия освещенности, или у него изменяется датчик или его расположение относительно поверхности поля, то и *KP*, и *LA* придется рассчитывать заново, подставлять в программу и перекомпилировать ее. Поэтому вот это очень и хочется возложить на плечи робота - пусть он делает вычисления вместо человека. Как бы тогда хотелось, чтобы устройство функционировало?

Можно предположить, что оно должно помогать определить самый светлый участок на поле (*Lmax*), а потом и самый темный (*Lmin*). Как только эти данные известны, то можно рассчитать и *KP* и *LA*, которые будут затем использованы в пропорциональном регуляторе при движении робота вдоль линии.

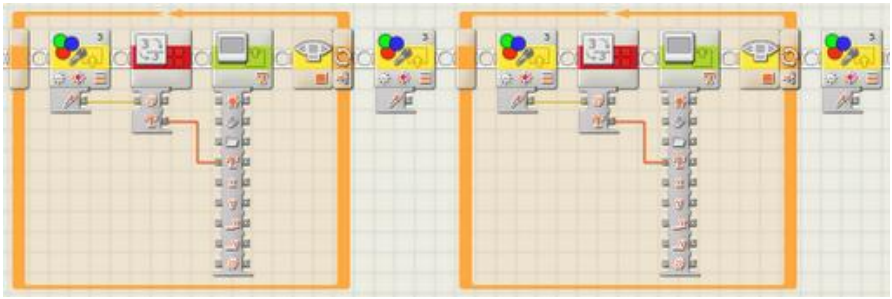


Итак, первый шаг – определить самый светлый участок. Не будем слишком усложнять программу, а воспользуемся традиционным способом, который используется в том числе и на робототехнических состязаниях - ручная калибровка, при которой оператор робота сам выбирает светлый участок поля и отдает роботу команду зафиксировать изменения.



Здесь, в цикле опрашивается датчик и его показания для удобства оператора выводятся на экран, как только фиксируется нажатие оранжевой кнопки на NXT блоке, обозначающее, что оператор сделал свой выбор, завершается цикл и текущее показание датчика еще раз считывается для дальнейшего использования.

Абсолютно такая же структура получиться для определения освещенности на темном участке.



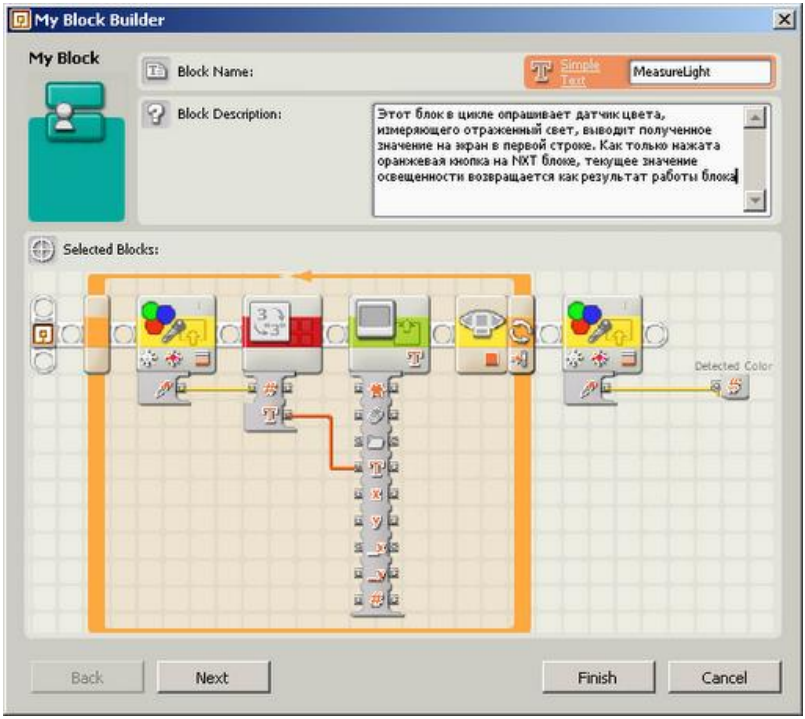
Если попробовать сейчас запустить этот кусок программы, то будет видно, что ожидание второго нажатия почему-то не происходит. На самом деле, второй цикл тоже работает, но в нем исполняется только одна итерация. Связано это с тем что у NXT блока быстродействие лучше, чем у человека. После нажатия на кнопку и выхода из первого цикла, программа сразу же заходит во второй - время перехода очень маленькое - оператор еще не успевает убрать руку с кнопки после нажатия, поэтому второй цикл определяет опять, что кнопка нажата и завершает свою работу.

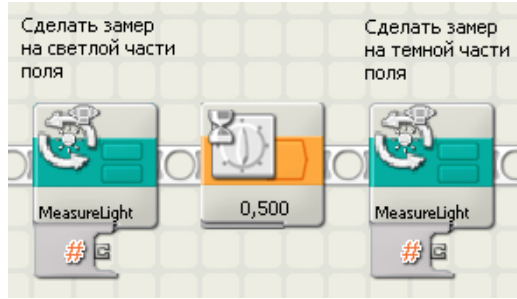
Решить эту небольшую проблему можно двумя способами:

- изменить событие завершения цикла с (нажато) на (нажато и отжато), либо

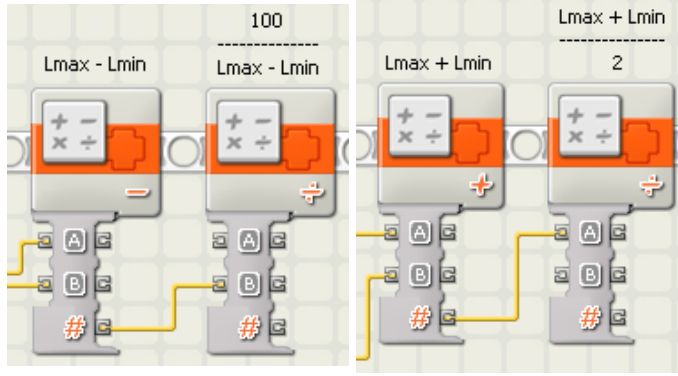
- добавить между двумя циклами небольшую паузу, достаточную для того, чтобы оператор отжал кнопку перед вторым циклом

Поскольку два цикла получились идентичные, то существенно сократить программу позволит использование собственных блоков (MyBlock).

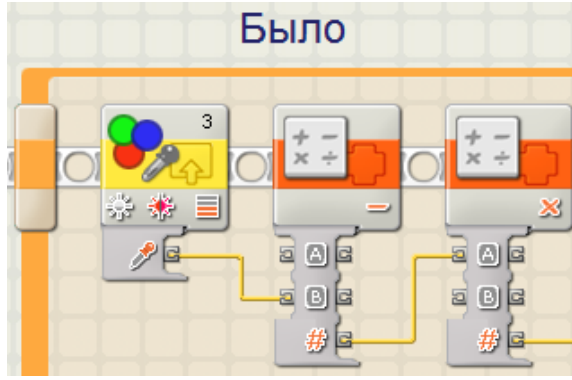


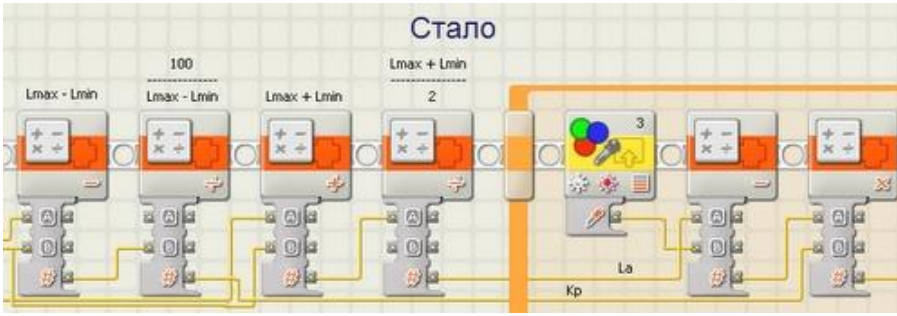


Что же, значения *Lmax* и *Lmin* получены – все готово к тому, чтобы рассчитать коэффициент пропорциональной составляющей регулятора и освещенность на границе линии.

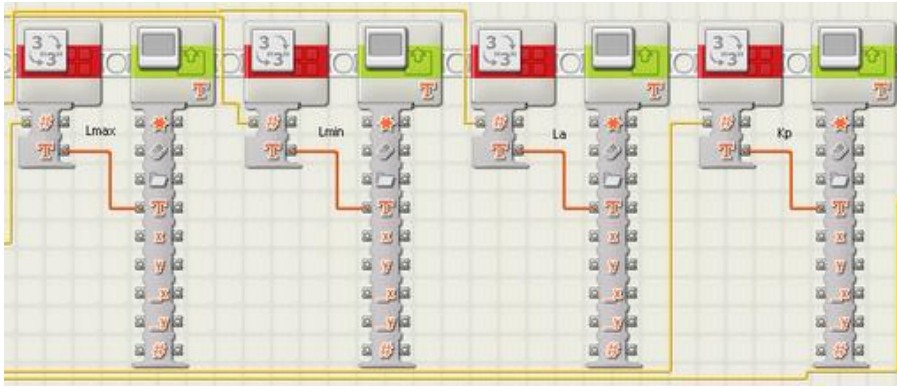


Теперь остается передать результаты вычислений внутрь цикла, выполняющего движение вдоль линии. Вместо жестко вбитых чисел в цикле будут использованы автоматически рассчитываемые значения.

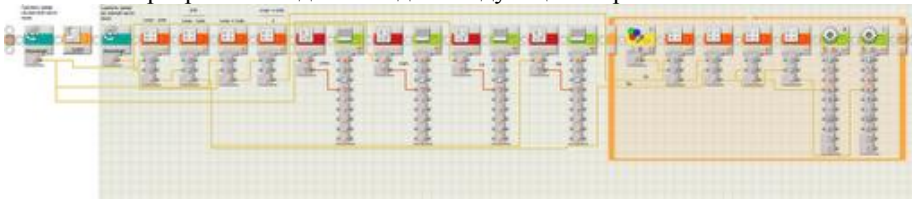




Практика показывает, что в ходе составления программы случаются всякие досадные ошибки при получении минимальной и максимальной яркостей, а равно и при вычислении коэффициента и «средней» освещенности. Поэтому имеет смысл в программе заранее предусмотреть вывод этих значений на экран – это позволит лучше понять, как и почему ведет себя робот в ходе движения вдоль линии.

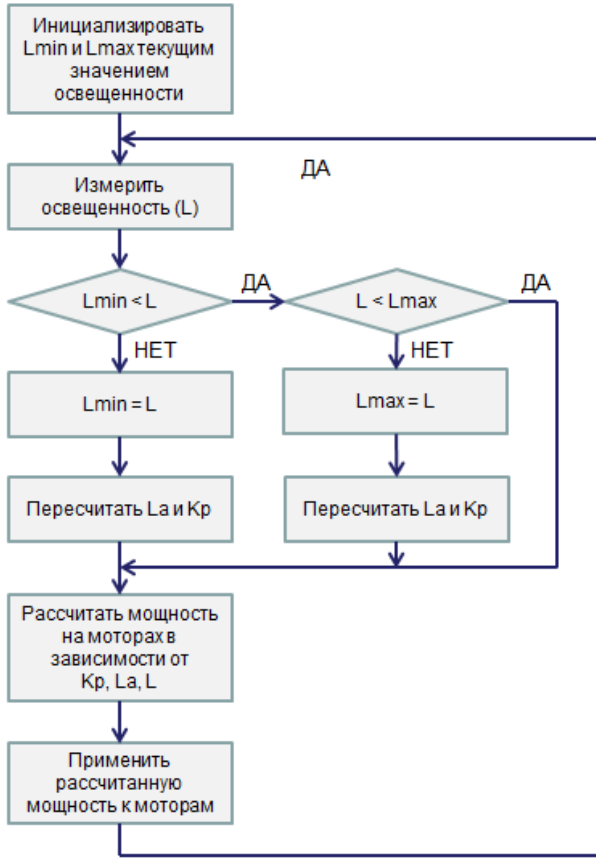


Финальная программа тогда выглядит следующим образом:



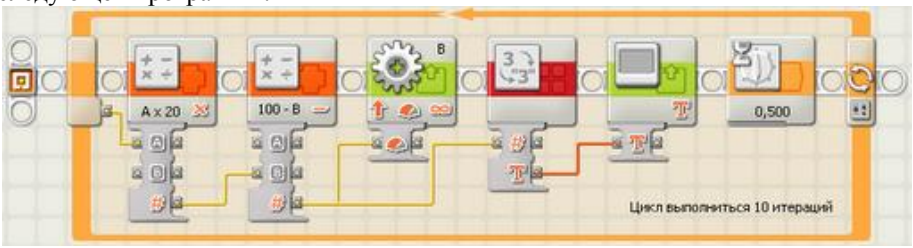
Данная программа позволяет перед запуском робота произвести измерения освещенностей на поле, которые затем использует для вычисления *KP* и *LA*, использующихся пропорциональной составляющей линейного

регулятора в основном цикле движения вдоль линии. В качестве альтернативного подхода можно рассмотреть алгоритм автокалибровки, предложенный одним из преподавателей Нижегородского Института Информационных Технологий. В этом подходе не существует предварительного замера освещенностей. Вместо этого коэффициент пропорциональной составляющей и освещенность на границе линии вычисляются и подстраиваются автоматически по ходу движения робота.



Прежде, чем переходить к исследованию от чего же еще зависит коэффициент, хотелось бы рассмотреть некоторые особенности управления моторами через каналы данных. Данное знание позволит более тщательно подходить к расчету коэффициента пропорциональности или, как это будет показано в конце заметки, сделать программу более устойчивой к изменению освещенности на поле.

Рассмотрим, что происходит с вращением оси мотора в случае работы следующей программы:



В программе в каждой следующей итерации цикла в качестве значения мощности мотора передаются все меньшие числа.

I	I x 20	100 - (I x 20) power
0	0	100
1	20	80
2	40	60
3	60	40
4	80	20
5	100	0
6	120	-20
7	140	-40
8	160	-60
9	180	-80

При запуске программы будет видно: мотор крутится вперед постепенно теряя скорость, затем останавливается и постепенно набирая скорость опять крутиться вперед. Что доказывает – **если в среде NXT-G в качестве мощности подавать отрицательные значения, то ось мотора будет вращаться в прямом направлении (вперед)**, а не в обратном в обратном направлении, как это можно было бы ожидать. Следует заметить, что в других средах программирования LEGO роботов это не так. Что же дает нам эта новая информация?

$$P_B = 50 + (L_A - L)K_P \qquad P_C = 50 - (L_A - L)K_P$$

Из формул выше видно, что в случае неправильно рассчитанного «среднего» значения освещенности LA (предполагаемой освещенности на границе линии), например, когда учитывалось не самое светлое место на поле, значение мощности может стать отрицательным.

Пусть,

$$L_{max} = 40 \qquad L_{min} = 20$$

$$L_A = \frac{L_{max} + L_{min}}{2} = \frac{40 + 20}{2} = 30$$

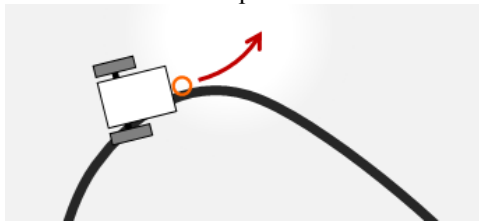
$$K_P = \frac{100}{L_{max} - L_{min}} = \frac{100}{40 - 20} = 5$$

Тогда если при каком-то измерении на поле возникнет ситуация, когда L будет больше замеренного ранее максимального значения, то

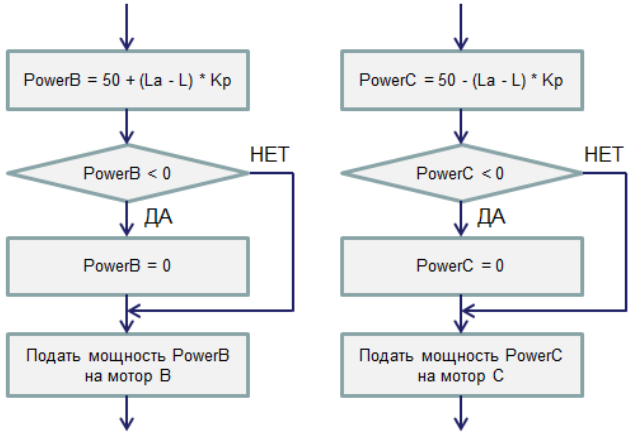
$$L = 47$$

$$P_B = 50 + (30 - 47) \cdot 5 = 50 + (-17) \cdot 5 = 50 - 85 = -35$$

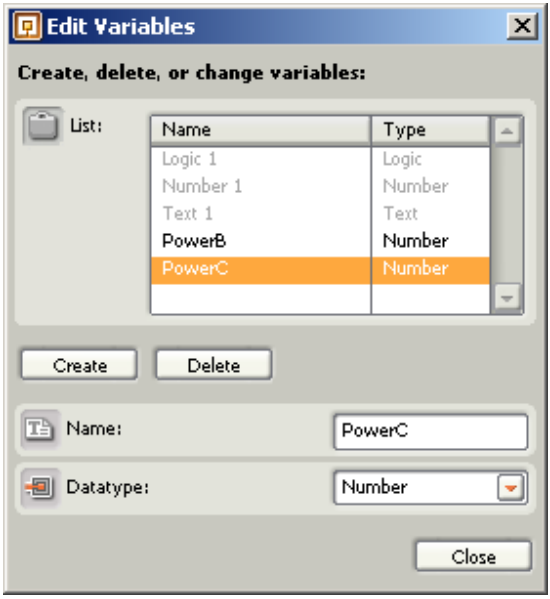
Что приведет к тому, что вместо ожидаемой остановки, колесо начнет двигаться. А поскольку и второе колесо крутиться, причем с максимальной скоростью, то робот может вылететь с трассы.



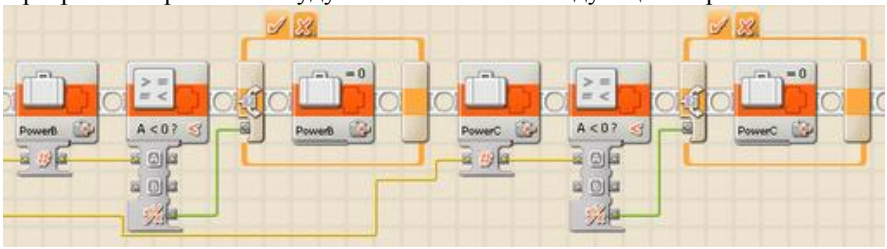
Чтобы избежать такой ситуации внутри цикла управления, после расчета мощностей, устанавливают ограничитель, который бы позволил избежать запуск моторов при непредвиденных значениях освещенности. Для каждого рассчитываемого значения мощности – свой ограничитель.



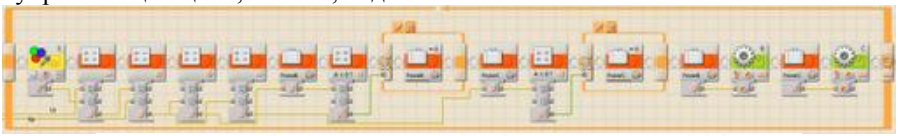
Для удобства операций, значение мощности будет передаваться посредством переменной.



В программе переменные будут использоваться следующим образом:



А управляющий цикл, в итоге, видоизменится:



$$K_P = \frac{50}{\frac{L_{max}-L_{min}}{2}} = \frac{50 \cdot 2}{L_{max}-L_{min}} = \frac{100}{L_{max}-L_{min}}$$

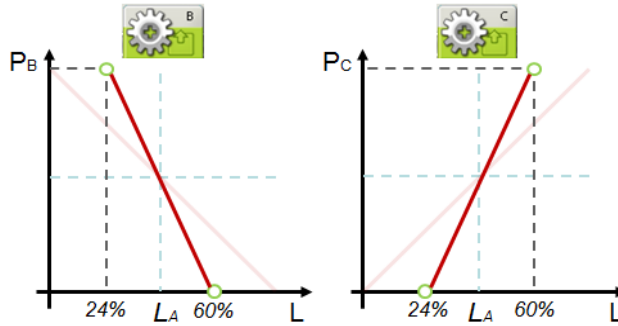
Где, *Lmax* – значения сенсора освещенности на самом светлом участке поля, а

Lmin - значение сенсора на самом темном участке.

Данный коэффициент использовался в формуле, которая определяла мощность подаваемую на каждый мотор при том или ином значении освещенности во время движения.

$$P_B = 50 + (L_A - L)K_P \qquad P_C = 50 - (L_A - L)K_P$$

Значения мощностей, получаемых этими формулами можно отобразить на графике



Иными словами, в заметке #5 было показано, что посредством введения коэффициента пропорциональности, для каждого значения освещенности от *Lmin* до *Lmax* сопоставились значения мощности в диапазоне от 0 до 100%.

Из графика и из формулы выше видно, что мощность колеблется вокруг некоторого среднего значения - 50%. Что по сути обозначает мощность, при которой оба мотора двигаются в одном направлении с одинаковой скоростью, т.е. робот едет прямо, в случае, когда текущее показание освещенности (показание на границе линии) равно *LA*. Такую мощность мы назвали средней мощностью. Легко предположить, что бывают ситуации, при которых средняя мощность может принимать другие значения.

Например, если линия, вдоль которой происходит движение, практически не изменяет своего направления, средняя скорость может быть больше, потому что ситуации, когда робот выскочит за линию и начнет снова ее искать, очень редки.

Другой пример – очень извилистая линия. Здесь средняя скорость может быть наоборот очень низкая, чтобы избегать вылетания с трассы и потери времени на поиск границы линии.

Существуют и другие причины для выбора средней скорости отличной от 50%: при большом диаметре колеса она может быть меньше, чем при маленьком диаметре при большой нагрузке на робота или при севших батарейках скорость придется увеличить, иначе робот будет двигаться слишком медленно и т.д.

Итак, для простоты объяснения рассмотрим оба варианта: один, когда средняя скорость меньше 50%, и второй, когда скорость больше 50%.

Первый вариант: *PA (средняя скорость)* > 50%

Представим, что робот едет по границе линии, а затем съезжает с нее, т.е. сначала освещенность примерно равна *LA*, а затем увеличивается до *Lmax*.

Одно из колес при этом должно полностью остановиться, чтобы движение второго вернуло робота на линию, т.е. мощность на этом колесе должна измениться со значения *PA* до 0.В действительности, если продолжать рассчитывать коэффициент и мощность по старым формулам, то произойдет примерно следующее:

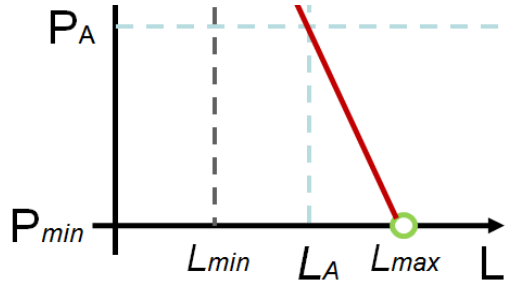
Пусть *LA* = 42, а *Lmax* = 60, а *PA* = 70, тогда *KP* = 2,78.

$$P_B = 70 + (L_A - L) \cdot 2,78$$

L	L_A - L	B
42%	0	70
47%	-5	56
52%	-10	42
60% (max)	-18	20

Из таблицы видно, что, скорее всего колесо не остановится и будет медленно, но вращаться, что может привести к тому, что робот потеряет линию.

Следовательно, значения, использующиеся в формуле вычисления, мощности должны быть изменены таким образом, чтобы обеспечить изменение мощности от *PA* до 0 при изменении освещенности от *LA* до *Lmax*. Иными словами, каждому значению из диапазона от *LA* до *Lmax* должно быть поставлено в соответствие значение мощности от *PA* до 0.



Используя свойства подобных треугольников, можно вывести формулу для коэффициента пропорциональности:

$$K_p = \frac{P_A - P_{min}}{L_{max} - L_A} = \frac{P_A - P_{min}}{L_{max} - \frac{L_{max}+L_{min}}{2}} = \frac{P_A - P_{min}}{\frac{2L_{max}-L_{max}-L_{min}}{2}} = \frac{2 \cdot (P_A - P_{min})}{L_{max} - L_{min}}$$

Теперь, если в формулу подставить значения, приведенные выше, то *KP* станет равным:

$$K_p = \frac{2 \cdot (70 - 0)}{60 - 24} = \frac{140}{36} \approx 3,8$$

А мощность при разных значениях L:

L	L_A - L	B	C
42%	0	70	70
35%	7	97	43
30%	12	117	23
24% (min)	18	140	0
47%	-5	51	89
52%	-10	31	109
60% (max)	-18	0	140

Следует заметить, что в таблице появляются мощности выше 100 - такие значения не требуют никакой специальной обработки – firmware на NXT блоке в таких случаях автоматически будет использовать максимальную возможную мощность - 100%.

Второй вариант: *PA (средняя скорость)* < 50%

Робот сейчас и так едет достаточно медленно, но дело еще усугубляется тем, что при сильном съезде с границы линии робот не спешит возвращаться на нее – особенно это актуально для трассы с крутыми поворотами представленной на картинке выше.

$$P_C = 35 - (L_A - L) \cdot 2,78$$

L	L_A - L	C
42%	0	30
47%	-5	49
52%	-10	63
60% (max)	-18	85

Значения в таблице показывают, что на моторы при заезде на светлую часть трассы (или темную) никогда не будет подаваться максимум энергии. Чтобы добиться подачи максимально возможной энергии на мотор нужно сопоставить каждому значению из диапазона *LA .. Lmax* значение мощности от *PA* до 100. Проделаем преобразования, аналогичные тем, что представлены выше:

$$K_p = \frac{P_{max} - P_A}{L_{max} - L_A} = \frac{2 \cdot (P_{max} - P_A)}{L_{max} - L_{min}}$$

Это приведет к следующим показаниям мощности.

L	L_A - L	B	C
42%	0	35	35
35%	7	60	10
30%	12	78	-8
24% (min)	18	100	-30
47%	-5	17	53
52%	-10	-1	71
60% (max)	-18	-30	100

Видно, что увеличение удалось добиться максимальной мощности в 100%. При этом также начали появляться отрицательные значения мощности. Что, без специальной обработки, может привести к неправильному поведению робота. В двух вариантах для вычисления коэффициента получились две разные формулы, которые отличаются лишь верхней частью делимого. Можно рассмотреть, есть ли какая-нибудь закономерность в этих частях, так чтобы все-таки свести их к единой формуле.

formula	$P_A - P_{min}$	$P_{max} - P_A$
$P_A = 50$	50	50
$P_A = 65$	65	35
$P_A = 80$	80	20
$P_A = 40$	40	60
$P_A = 30$	30	70

Если провести дальнейшую подстановку полученных разностей в формулы для вычисления коэффициента, то можно будет заметить, что "правильному" коэффициенту (тому, который приводит приближение мощности при крайних значениях освещенности к 100 или 0) соответствуют максимальные значения разности. Отсюда математическая запись формулы приходит к следующему виду:

$$K_P = \frac{2 \cdot max(P_{max} - P_A, P_A - P_{min})}{L_{max} - L_{min}}$$

Что по сути обозначает - для вычисления коэффициента пропорциональности линейного регулятора выберите наибольшую разность между максимальной желаемой мощностью и средней мощностью или между средней мощностью и минимальной желаемой мощностью. В формуле умышленно брались *Pmax* и *Pmin* вместо 100 и 0, для того чтобы показать, что в определенных случая желаемые максимальные и минимальные мощности могут отличаться от предельно допустимых, если программист робота точно понимает какого результата он хочет добиться. Но в большинстве случаев, вместо *Pmax* следует использовать 100, а вместо *Pmin* - 0.

На языке NXT-G данное вычисление коэффициента пропорциональности линейного регулятора для движения робота-тележки вдоль границы черной линии может выглядеть следующим образом:

ЗАКЛЮЧЕНИЕ

Уважаемый обладатель сборника Вы ознакомились с курсом «Робототехники» и мы надеемся, что Вы прошли все этапы усвоения системы построения, программирования, анализирования ошибок и их исправления.

Мы будем рады, если наши разработки помогли Вам на данном этапе.

Не останавливайтесь на достигнутом, находите новые пути решения и построения задач. Создавайте оригинальные конструкции, которые будут помогать Вам в дальнейшем развитии.

Успехов Вам!

Список использованной литературы

1. Задачи для факультатива робототехники. Ушаков А.А,//Материалы конкурса ИКТО-2009

2. Дистанционный курс "Основы робототехники". АЛТГПА. - http://www.uni-altai.ru/ifmo/ktoi/dist_ktoi/

3. Челябинский РКЦ. - http://www.rkc-74.ru

4. Дистанционный курс «Конструирование и робототехника» - http://learning.9151394.ru/course/view.php?id=17

5. Институт новых технологий. - http://www.int-edu.ru

6. Образовательная робототехника. Новые горизонты в МК- Марий Эл.- http://mrl.mk.ru/article/2013/06/19/871304-obrazovatelnaya-robototehnika-novyie-gorizontyi.html

http://vaz2101spb.ru/wiki/tjuning/lrc/chto_takoe_peredatochnoe_chislo

http://www.det-mash.ru/index.php?file=cherv_pered

http://www.prorobot.ru/lego/robot_5minutka.php