Robototechnics Department
Physics and Mathematics Lyceum #30

Техническая книга Соревнований First FTC Робот PML-30

Санкт-Петербург
2013

1. Оглавление

1. Оглавление 2
2. Состав команды 3
3. Благодарности 3
4. Введение 3
4.1 Требования к роботу 3
4.2 Цели и задачи 5
5. Основная часть 7
17.11.2012 7
22.11.2012 9
24.11.2012 11
29.11.2012 14
01.12.2012 16
02.12.2012. FTC competition. 17
06.12.2012 20
08.12.2012 22
15.12.2012 24
10.01.2013 26
17.01.2013 31
27.01.2012 33
6. Итоговая модель робота 35
7. Итоговая версия программы 36
7.1 Управляемый период (сырой код, без оболочки): 36
7.2 Автономный период (сырой код, без оболочки): 37

2. Состав команды

ФИО	*Год рождения*	*Место учебы*	*Роль в команде*
Жадковский Александр	1998	Учащийся ФМЛ №30	Программист, оператор №1
Крылов Георгий	1997	Учащийся ФМЛ №30	Инженер, ответственный за техническую книгу
Плеханов Вячеслав	1996	Учащийся ФМЛ №30	Программист, оператор №2, ответственный за модель в Creo
Фокин Иван	1997	Учащийся ФМЛ №30	Инженер
Дмитрий Валерьевич Лузин	1989	Выпускник ФМЛ №30	Руководитель
Антон Владимирович Федотов	1993	Выпускник ФМЛ №30	Руководитель

3. Благодарности

Мы благодарим за предоставленное оборудование и поддержку компанию PTC в лице Дмитрия Орлова, а также ее российского представителя, компанию Ирисофт в лице Ольги Шартуковой.

Также мы благодарим наших руководителей: Дмитрия Валерьевича и Антона Владимировича, а также директора ФМЛ №30 Третьякова А.А. за предоставленное помещение и оборудование.

4. Введение

Робот PML-30 создан для участия в соревнованиях FTC из конструктора Lego Mindstorms Education 9797 и конструктора Tetrix 739144.

Для создания робота использовался один набор Lego и один набор Tetrix. Поэтому у нас строгое ограничение по количеству деталей.

4.1 Требования к роботу

Общие правила для робота	**№**
Робот полностью помещается в *«измерительный куб»* (с внутренними	RG4
Робот НЕ содержит никаких компонентов, которые могут повредить игровое	RG3a и b
Робот НЕ содержит каких-либо вредных материалов.	RG3c
Робот НЕ представляет чрезмерную угрозу, связанную с возможностью	RG3d
Робот НЕ имеет никаких острых краев и углов.	RG3e
Главный переключатель питания установлен правильно. Он легко доступен и	RG5 и
Все аккумуляторные батареи надежно прикреплены к роботу.	RG6
Батарея NXT легко отсоединяется при минимальной разборке робота.	RG7a
USB-порты (на блоке NXT и коммуникационном модуле WiFi Samantha) легко	RG7b
Кнопки NXT контроллера и модуля Samantha легко доступны.	RG7b
Жидкокристаллический дисплей NXT контроллера и светодиоды модуля	RG7c
Электрические компоненты смонтированы таким образом, чтобы они были	RG7d

Робот оснащен держателем для флажка, который надежно удерживает флажок	RG8
Номер команды хорошо виден, как минимум, с двух сторон робота (под углом 180°). Высота цифр составляет 3 дюйма, а ширина штриха равна не менее 1/2 дюйма, причем цвет цифр контрастирует с их фоном. Номера держатся достаточно прочно, чтобы выдержать довольно жесткие условия игры	RG9
Энергоснабжение обеспечивается разрешенными источниками питания.	RG10
Скорость и сила, с которыми робот бросает игровые элементы, не превышают	RG11
Правила, касающиеся деталей роботов и материалов для их изготовления	
Все детали заводского изготовления, из которых сделан робот, взяты из наборов	R1 и R2c
При изготовлении робота не использовались коммерчески доступные узлы и сборные детали, за исключением тех, которые специально разрешены	R2a
Робот не содержит изготовленных заводским способом и/или предварительно отформованных коммерчески доступных деталей из пластмассы или металла.	R2b
Робот не содержит дополнительных механических деталей, за исключением	R2d
Робот имеет только один (1) контроллер NXT, а дополнительные	R3a
Робот имеет один (1) стандартный блок аккумуляторных батарей NXT (переменного или постоянного тока) или шести (6) батареек AA (а не два	R3b
Робот имеет только один (1) стандартный блок аккумуляторных батарей	R3c
В конструкции робота использованы только контроллеры сервоприводов и электродвигателей HiTechnic или MATRIX (их количество не ограничено).	R3e
В конструкции робота использовано не более восьми (8) электродвигателей и не более двенадцати (12) сервоприводов TETRIX или MATRIX, каждый из которых	R3f
Каждый моторный порт NXT (A, B или C) используется для управления не более: (i) одного интерактивного сервомотора NXT, либо (ii) одного приводного XL-мотора, либо (iii) двух приводных E-моторов, либо (iv) двух приводных M-	R3g
Робот имеет только один (1) модуль Samantha и один (1) кабель USB.	R3h
Все сенсоры/датчики прикреплены непосредственно к программируемому блоку NXT, сенсорному мультиплексору HiTechnic или к сенсорному мультиплексору	R3i
Контейнер батареи HiTechnic напряжением 9 В (при наличии такового) используется лишь в качестве составной части сенсорного мультиплексора NXT.	R3j
Макетная плата SuperPro HiTechnic и макетная плата NXT используются в	R3k
В конструкции робота используются только одобренные компанией LEGO	R3l&m
В конструкции робота используются электрические разъемы питания Anderson Powerpole (обжимного типа или разъемы быстрого соединения).	R3n
Провода питания, включая проводку электродвигателей, сервоприводов и	R3o
В конструкции робота используются только светодиоды видимого излучения, источником питания которых служит либо основная аккумуляторная батарея,	R3p
Робот имеет только те электрические компоненты, которые специально разрешены действующими правилами, причем первоначальная конструкция	R3s
Пневматические компоненты LEGO не подвергались модификации с целью	R4
Декоративные элементы робота сделаны с использованием разрешенных деталей или при помощи компонентов, которые не являются функциональными по своей сути. Все украшения и декоративные элементы соответствуют принципам «благородного профессионализма»	R7

Робот PML-30 соответствует всем перечисленным выше требованиям.

4.2 Цели и задачи

Задача данного робота состоит в том, что бы победить соперника, набрав большее количество очков. Соревнование состоит из 2-х периодов:

1. Автономный период.
2. Период с участием операторов роботов.

1. Автономный период: Этот период длится 30 секунд, в течение которого робот должен повесить Автономные кольца на крючки с ИК-датчиками (роботы автономны). За каждое Автономное кольцо на таком крючке альянс получает бонусные 50 очков и право собственности на этот крючок.

2. Период с участием операторов роботов: Оператор, с помощью манипулятора должен заставить робота выполнить следующее:

повесить кольца на крючки вешалки, поместить их на поверхность центральных «ворот» игрового поля или надеть на крючки угловых «ворот». (Вешалка имеет две независимые друг от друга стороны, на каждой из которых находится три ряда крючков – по три крючка в каждом ряду). Кольца любого альянса (т.е. красные и синие) и любого типа (обычные и утяжеленные) могут принести своему альянсу заслуженные очки на любом крючке (т.е., например, все положенные очки за красные кольца, которые висят на стороне вешалки, обращенной к площадке для размещения альянса «синих», будут начисляться без каких-либо ограничений). Команда, сумевшая повесить на тот или иной крючок больше колец, чем ее оппоненты, получает «право собственности» на такой крючок, за исключением случаев, когда на крючке находится автономное кольцо противостоящей команды, дающее ей безоговорочное «право собственности» на такой крючок. Если оба альянса имеют одинаковое количество колец (больше нуля) на одном крючке, то «право собственности» на такой крючок будет принадлежать одновременно обоим альянсам. Кольцо, лежащее на поверхности центральных «ворот» игрового поля, должно полностью находиться в пределах их внешних границ (т.е. никакая часть кольца не может выступать за граничную линию центральных «ворот»). Только утяжеленные кольца могут принести своему альянсу очки, находясь на крючке угловых «ворот». Обычные и автономные кольца, помещенные на угловые «ворота», не засчитываются и не приносят никаких очков.)

Финальный этап: этот этап длится последние 30 секунд периода с участием операторов роботов, за это время робот одной команды может поднять робота своих партнеров по альянсу над поверхностью игрового поля, чтобы заработать бонусные очки за поднятие робота. Роботы, выполняющие подъем, защищены в зоне своего альянса (роботы не автономны).

Для того чтобы получить бонусные очки за поднятие робота, необходимо, чтобы вес поднятого робота полностью опирался на робота-партнера по альянсу без использования каких-либо объектов или элементов игры (т.е. робот, выполняющий подъем, должен самостоятельно удерживать 100 процентов веса поднимаемого робота). Робот, выполняющий подъем, не вправе использовать какие-либо элементы игрового поля (например, стойки, держателя колец и т.д.), для того чтобы поднять своего робота-партнера. Поднятые роботы должны удерживаться над поверхностью игрового поля на протяжении как минимум одной (1) минуты или до тех пор, пока рефери не завершит подсчет очков и бонусных баллов за поднятие робота по итогам матча.

Система подсчета результатов состязания на финальном этапе игры.

Бонусные очки за поднятие робота – роботы, которые подняты и удерживаются над поверхностью игрового поля на высоте не менее 1 дюйма (2,5 см), принесут своему альянсу 30 бонусных баллов. При этом высота измеряется от самой нижней точки робота до поверхности покрытия игрового поля.

Кроме того, альянсы могут получить еще по пять (5) бонусных очков за каждый

дополнительный дюйм (2,5 см) высоты, на которую поднят робот, причем максимально допустимая высота над поверхностью игрового поля составляет (61 см).

Цели:

1. Робот должен быть способен брать кольца с крючка.
2. Робот должен управляться с джойстика через модуль Samantha.
3. Создать конструкцию, которая при данных ресурсах позволит вешать кольца на максимально возможную высоту.
4. Робот должен быть способен брать сразу 2 кольца.
5. Робот должен быть способен забраться на ступеньку центральных ворот.
6. Управление роботом должно быть максимально удобным для оператора.
7. Оператор робота должен иметь возможность отличить тяжелое кольцо от лёгкого.
8. В автономном режиме робот должен вешать автономное кольцо на крючок с ИК-датчиком.
9. Модель робота должна быть реализована в Creo. Это нужно для того, что если придется разбирать робота, то его легко можно было восстановить. Также в Creo можно будет посмотреть, как изменится поведение робота, при изменении конструкции, не трогая робота, тем самым уменьшить риск его поломки при тестировании.

5. Основная часть

17.11.2012

1. Дата собрания и время начала:
   Дата собрания: 17.11.2012.
   Время начала: 16:00.

2. Поставленные цели:
   2.1. Собрать конструкцию, которая ездит и захватывает кольца на одной высоте.
   2.2. Написать программу для управления роботом.

3. Реализация целей:
   3.1. Была собрана такая конструкция: мотор поднимает две балки, робот захватывает ими кольцо и прижимает его к верхней перегородке (Рисунок 1).
   3.2. Для быстрого доступа к батарейкам блок NXT был поставлен вертикально (правило RG7a).
   3.3. Передние колёса были заменены на полусферы, потому что колёсам необходимо много места для разворотов.
   3.4. Сначала, для большей жесткости конструкции балки захвата были скреплены между собой посередине. Но крепления были убраны для того, чтобы кольцо лучше держалось в захвате.
   3.5. Для того, чтобы случайно не взять больше 2-х колец (если возьмёшь больше, то будет штраф) была установлена перегородка.

   3.6. С помощью 4 датчиков касания был собран пульт управления и написанна программа управления: 1 датчик - ехать вправо, 2 датчик - ехать влево, 1 и 2 датчик одновременно - ехать вперёд, 3 и 4 датчик одновременно - ехать назад, 3 датчик - опустить балки, 4 датчик - поднять балки.

4. Результат и итоги собрания:
   4.1. Собран двухколёсный робот с захватом.
   4.2. Робот управляется с помощью датчиков касания, берёт кольца на одной высоте, ездит и вешает их обратно.

Получившаяся конструкция:

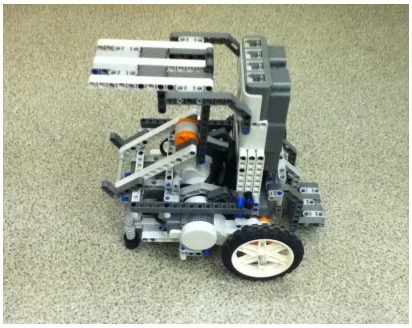


Рисунок 1. Простейший робот для FTC.

5. Идеи для следующих собраний:

5.1. Приобрести джойстик.
5.2. Написать программу управления роботом через джойстик.
5.3. Подключить NXT к ПК через Bluetooth.

22.11.2012

1. Дата собрания и время начала:

Дата собрания: 22.11.2012.
Время начала: 16:00.

2. Поставленные цели:

2.1. Подключить NXT к ПК через Bluetooth.
2.2. Реализовать программу управления джойстиком.

3. Реализация целей:

3.1. NXT был подключен к ПК через Bluetooth.
3.2. Была реализована программа для управления роботом по Bluetooth, через джойстик.
3.3. При использовании робота было очень неудобно захватывать кольца – он “дёргался”. Поэтому было сделано 2 режима управления: на маленьких скоростях и на больших.
3.4. Иногда, когда робот брал кольцо, он переворачивался на спину, чтобы этого не происходило, сзади была установлена подставка.

4. Результат и итоги собрания:

4.1. NXT был подключен к ПК через Bluetooth.
4.2. Реализовано управление роботом с джойстика по Bluetooth. Он берёт кольца на одной высоте, ездит и вешает их обратно.
4.3. Реализовано 2 режима управления: на быстрых скоростях и на медленных.

5. Идеи для следующих собраний:

Получившийся робот ездит и вешает кольца на одну высоту.
Чтобы он вещал кольца на разные по высоте крючки, были идеи:

5.1. С помощью дополнительного мотора поднимать захват на нужную высоту. Но тогда придётся подключать дополнительный блок NXT, а это противоречит правилам (правило R3a).
5.2. Использовать третий мотор не для поднятия балок, а для поднятия коробки для колец по направляющим. Кольца будут приподниматься снизу, удерживаться боковыми стенками коробки и сниматься с крючка за счёт движения робота от крючка (Рисунок 2, 3).

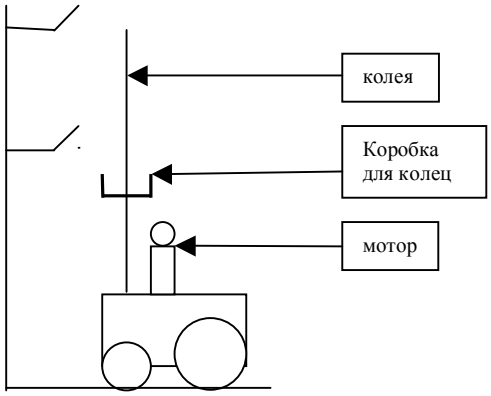


Рисунок 2. Вид сбоку.

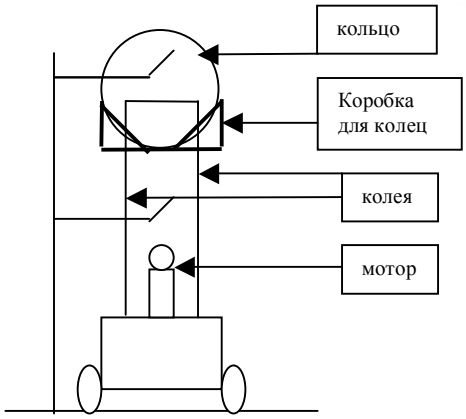


Рисунок 3. Вид спереди во время вешания кольца.

24.11.2012

1. Дата собрания и время начала:

Дата собрания: 24.11.2012.
Время начала: 15:00.

2. Поставленные цели:

(Наша организация приобрела конструктор Tetrix. Теперь мы будем реализовывать робота на базе конструктора Tetrix.)

2.1. Ознакомиться с конструктором Tetrix.
2.2. Построить стандартную тележку Tetrix.
2.3. Придумать, как построить робота для FTC.
2.4. Реализовать лучший проект робота для FTC.
2.5. Реализовать в Creo стандартную тележку Tetrix.

3. Реализация целей:

3.1. Была собрана стандартная тележка Tetrix.
3.2. Идеи сборки робота:

3.2.1. Первоначальная идея - сделать направляющие по которым, как у погрузчика, будет подниматься захват для колец:

3.2.1.1. Идея поднимать захвата с помощью шестерёнок. Но она была отклонена из-за отсутствия специальных осей для шестерёнок:

Рисунок №4. Специальная ось для шестерёнок

3.2.1.2. Идея поднимать захват с помощью верёвки, которая будет наматываться на катушку, которая присоединена к мотору:

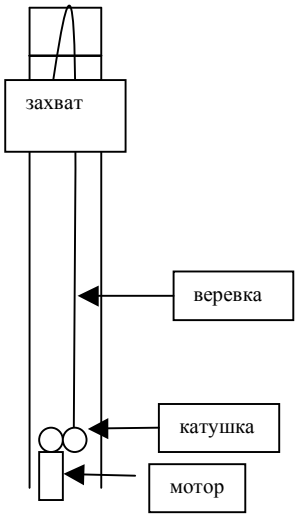


Рисунок 5. Поднятие захвата с помощью верёвки.

Такая конструкция была отклонена, так как он не влезет по размерам (правило RG4). Однако идея поднятия захвата с помощью верёвки была рассмотрена как перспективная.

3.2.2. Идея сделать выезжающие реи:

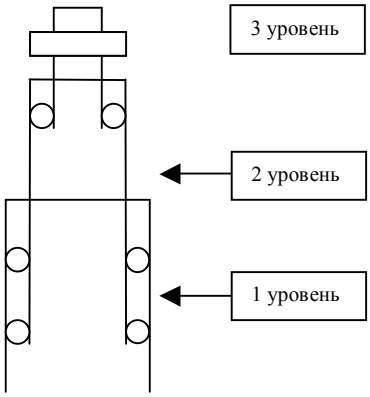


Рисунок 6. Выезжающие реи.

Для того, чтобы доставать до самого верхнего крючка пришлось бы сделать 3 уровня, потому что расстояние от верха стандартной тележки до допустимого более чем в 2 раза меньше расстояния до верхнего крючка. На это ушло бы очень много деталей, а у нас только один набор Tetrix. Поэтому данная конструкция нерациональна и сложна по реализации.

3.2.3. Идея разворота балок на 180º с помощью сервоприводов.

3.3. Была реализована идея разворота балок на 180º с помощью сервоприводов:

Рисунок 7.
Разворот балок с помощью сервоприводов.

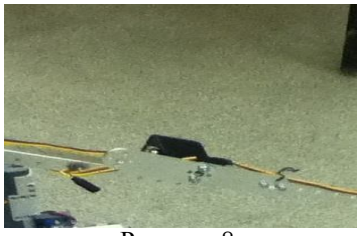


Рисунок 8.
Развернутое положение.

3.4. Поднятие всей конструкции:

Из-за того, что у нас сервоприводы заняты, для поднятия всей конструкции на вертикальную балку были установлены 2 мотора Lego, поднимающие конструкцию.

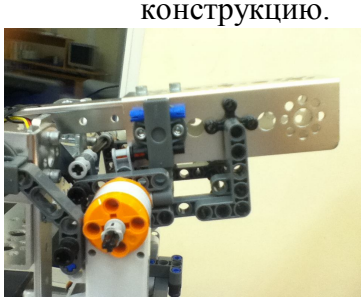


Рисунок 9.
Моторы Lego вид сбоку.

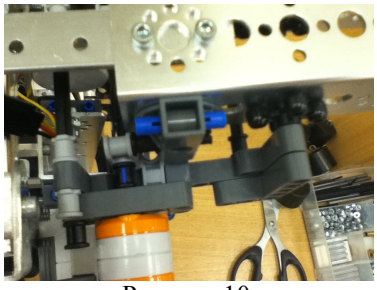


Рисунок 10.
Моторы Lego вид сверху.

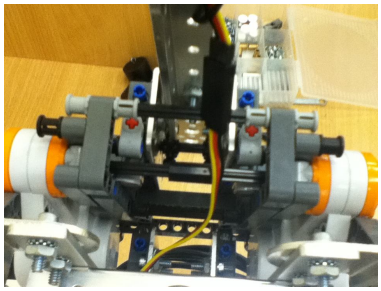


Рисунок 11
Моторы Lego вид спереди.

3.5. Таким образом, у конструкции образовалось 4 уровня:

1 уровень – вертикальная закреплённая балка с моторами Lego.

2 уровень – балка, непосредственно поднимаемая моторами Lego, с первым сервоприводом на конце.

3 уровень – балка, поднимаемая первым сервоприводом, со вторым сервоприводом на конце.

4 уровень – захват – коробка для колец, соединённая с сервоприводом балкой Lego.

3.5. В Creo была реализована стандартная тележка Tetrix.

4. Результат и итоги собрания:

4.1. Была собранна основа робота.

4.2. На вертикальной балке были закреплены моторы Lego.

4.3. Из сервоприводов и деталей конструктора Tetrix была собрана конструкция, которая позволяет достать кольца с верхнего крючка.

4.4. В Creo была реализована стандартная тележка Tetrix.

5. Идеи для следующих собраний:

5.1. Реализовать рисунок робота в Creo.

5.2. Сделать вешалку.

29.11.2012

1. Дата собрания и время начала:

Дата собрания: 29.11.2012.
Время начала: 16:00.

2. Поставленные цели:

2.1. Протестировать получившуюся конструкцию.
2.2. Устранить выявленные недочёты.
2.3. Сделать вешалку.
2.4. Реализовать модель в Creo.

3. Реализация целей:

3.1. Была сделана вешалка.
3.2. Из нижнего положения моторы никак не могли поднять конструкцию, чтобы конструкция не опускалась ниже места, с которого моторы поднимают её (робот ещё влезает в размеры), мы установили специальную подставку.
3.3. При сборке детали Lego и Tetrix плохо совмещались из-за не пропорциональности и конструкция сильно шаталась, чтобы избежать этого были установлены 2 балки по бокам, и конструктор Lego был стянут балками.
3.4. Для быстрого доступа к батарейкам было изменено положение NXT.
3.5. При выпрямлении манипулятора робота провода не доставали до верхнего мотора, поэтому мы переставили блок на вертикальную балку.
3.6. При проверке робота оказалось, что моторам не хватает мощности, чтобы поднимать манипулятор, поэтому была разработана система противовесов.
3.7. В результате робот оказался больше допустимых размеров, чтоб уменьшить его длину мы “спрятали” коробку для колец под балкой манипулятора:

Рисунок 12. Захват для колец.

3.8. При складывании и раскладывании конструкции балки сильно ударились друг об друга и перебили провод. Он был починен и примотан к балке.

3.9. В результате короткого замыкания из строя вышел один сервопривод.

4. Результат и итоги собрания:

4.1.. Один сервопривод вышел из строя.
4.2.. Была сделана вешалка.

Получившаяся конструкция:

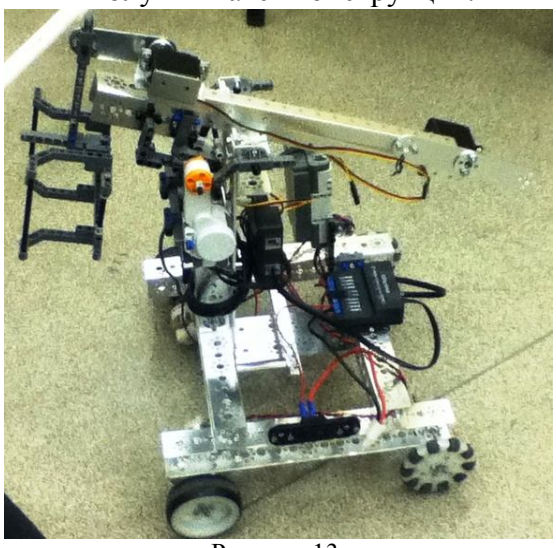


Рисунок 13.
Сложенное положение.

Рисунок 14.
Развёрнутое положение.

5. Идеи для следующих собраний:

5.1. Сделать придаточное соотношение для моторов Lego.
5.2. Заменить сгоревший мотор или изменить конструкцию.
5.3. Приобрести кольца, чтобы можно было тренироваться.

01.12.2012

1. Дата собрания и время начала:

Дата собрания: 1.12.2012.
Время начала: 16:00.

2. Поставленные цели:

2.1. Заменить сгоревший мотор на мотор Lego.
2.2. Протестировать его.
2.3. Если он не будет работать, то изменить конструкцию.
2.4. Подготовиться к соревнованиям First – Russia..

3. Реализация целей:

3.1. Вместо сгоревшего сервопривода был установлен мотор Lego. Но ему не хватило мощности поднять конструкцию.
3.2. Была убрана балка между захватом для колец и сгоревшим сервомотором.
3.3. Захват был установлен на место сгоревшего сервомотора.
3.4. Теперь захват находится в другой стороне от моторов Lego, и они перестали поднимать конструкцию. Для решения этой проблемы были подвешены дополнительные грузы для противовеса.

4. Результат и итоги собрания:

4.1. Был убран сгоревший сервопривод и 3 уровень (смотри пункт 3.5. от 24.11.2012)
4.2. Были подвешены дополнительные грузы.

Получившаяся конструкция:

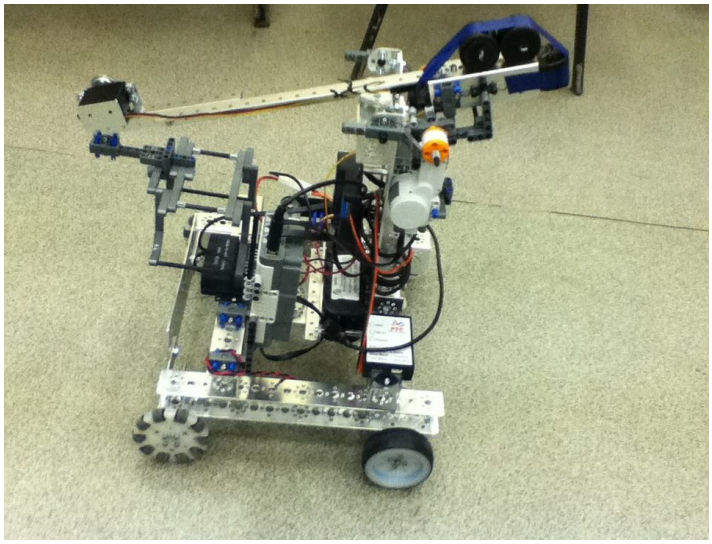


Рисунок 15.
Сложенное положение.

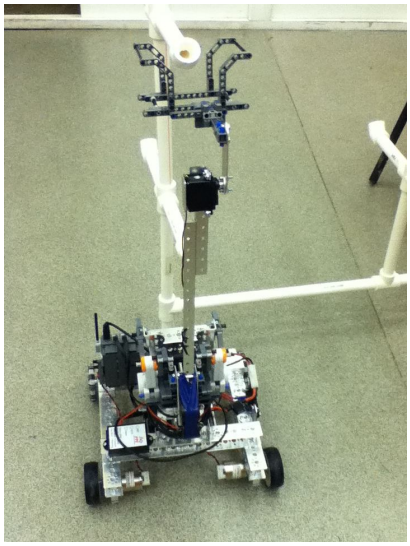


Рисунок 16.
Развёрнутое положение.

02.12.2012. FTC competition.

В Москве прошло соревнование First Russia FTC.

1. Результат:

1.1. По итогам матчей команда заняла 5 место в рейтинге из 9.
1.2. Техническая книга заняла одно из последних мест из-за её недоработанности .

2. Проблемные моменты и их решения:

2.1. Техническая книга была неправильно структурирована. Техническая книга будет вестись наиболее подробно.
2.2. Робот не помещался по длине в допустимые габариты (правило RG4). Был перевешен задний груз.
2.3. Грузы были примотаны изолентой, это запрещается правилами. Грузы перекрепили специальными крепежами.
2.4. Батарея была прикреплена специальными крепежами.
2.5. USB провод был подключён к модулю Samantha так, что при столкновении мог повредить её. Посередине робота была установлена балка и на неё перемещена Samantha.
2.6. Спереди робота поместили пластину, прикрывающую сервоконтроллеры.
2.7. На пластину была наклеена табличка с названием команды.
2.8. Отсутствовала программа автономного периода, она была реализована с функцией wait for start.
2.9. Управление роботом было реализовано для джойстика …, а на соревнованиях был джойстик …, поэтому кнопки располагались неудобно. Значение кнопок было изменено, также решили приобрести такой же джойстик.
2.10. Моторы Lego не смогли поднять манипулятор вместе с кольцом, поэтому робот мог достать только до самых нижних крючков. Решено реализовать передаточное соотношение.
2.11. В процессе матча при разворачивании из сложенного положения захват зацепился за переднюю пластину, оператор попробовал отцепить захват, в результате чего робот перевернулся. Пластина была перевешена ниже.
2.12. Робот мог забираться на ступеньку центральных ворот только из положения, когда он находился под углом к ступеньке, из-за того что колеса были параллельны ступеньке. Для решения этой проблемы возникла идея поставить передние колёса не параллельно корпусу, а развернув их на 45º, во внутреннюю сторону.
2.13. Оказалось, что при сборке робота, задние колёса собраны не симметрично – ось вращения находится не посередине мотора. Решено переставить моторы так, чтобы ось вращения была в самой нижней точке для того, чтобы ступенька не задевала мотор.

3. Методы и технические решения соперников:

3.1.

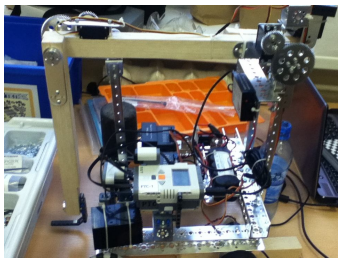


Рисунок 17.

В этой конструкции используются деревянные бруски – это позволяет сэкономить детали. На сервоприводы установлено передаточное соотношение. Также спереди установлены противовесы.

3.2.

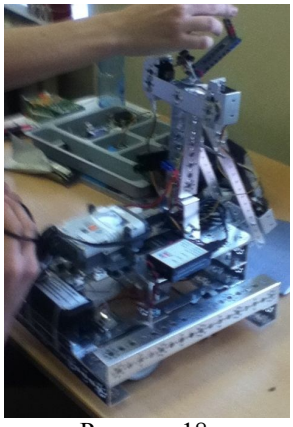


Рисунок 18.

В этой конструкции ведущие колёса установлены в шахматном порядке, что бы лучше поворачивать.

3.3.

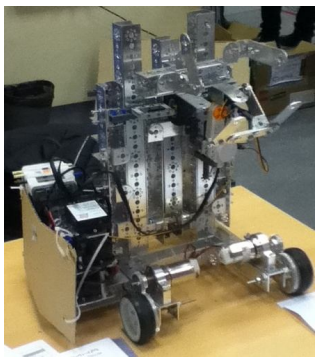


Рисунок 19.

Данная конструкция задействует очень много деталей и нерезультативна.

3.4.

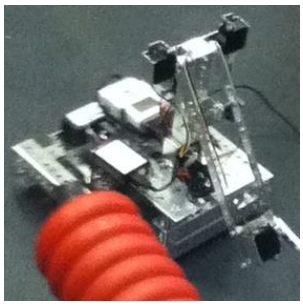


Рисунок 20.

Конструкция поднимается с помощью двух сервомоторов – это не оптимально, лучше задействовать один сервопривод с передаточным соотношением, а второй использовать для поднятия следующего уровня (как на рисунке 17).

4. Тактика:

4.1. Наша тактика до начала соревнований заключалась в том, чтобы вешать кольца по одному (2 мы не поднимем точно) в ряд на средних по высоте крючках, сначала с одной стороны, потом с другой. Если альянс противников будет вешать кольца на средние по высоте крючки, то мы будем вешать вторые кольца, чтобы не дать получить им право собственности на средние по высоте крючки. Если альянс противников на будет этого делать, то тогда мы будем вешать по два кольца на нижние крючки в линию.

4.2. Перед началом соревнований оказалось, что моторы Lego не могут поднять захват вместе с кольцом, поэтому тактика была изменена: брать по два кольца и вешать их на нижние крючки. Брать по одному кольцу легче и быстрее, но так как роботы участников могли вешать только на самые нижние вешалки, было решено брать по два кольца, чтобы обладать правом собственности.

4.3. Тактика соперников в периоде с участием операторов роботов (наиболее результативная) состояла в том, чтобы брать по одному кольцу и вешать их на разные крючки, собирая линию.

4.4. Никто из участников соревнования не отличал тяжёлые кольца от лёгких. Ни у одного робота не было датчиков для определения веса кольца.

4.5. Никто из участников соревнования не поднял союзника по альянсу над полом. Не было ни одной попытки.

4.6. В автономном периоде только один участник повесил автономное кольцо на крючок. У него не было ИК-датчика, он вешал его на строго запрограммированный крючок. Остальные роботы либо оставались неподвижными, либо проезжали некоторое расстояние вперёд.

06.12.2012

1. Дата собрания и время начала:

Дата собрания: 6.12.2012.
Время начала: 16:00.

2. Поставленные цели:

2.1. Была восстановлена конструкция. (При транспортировке робот был частично разобран.)
2.2. Переставить задние моторы согласно пункту 2.13 от 2.12.2012.
2.3. Изменить положение передних колёс согласно пункту 2.12 от 2.12.2012.

3. Реализация целей:

3.1. Была восстановлена конструкция (рисунок 15, пункт 4 от 2.12.2012)
3.2. Задние моторы были переставлены,
согласно пункту 2.13 от 2.12.2012.

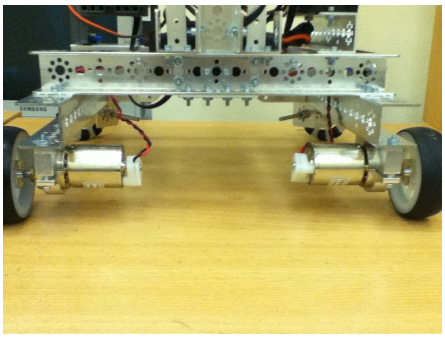


Рисунок 21.
Задние моторы.

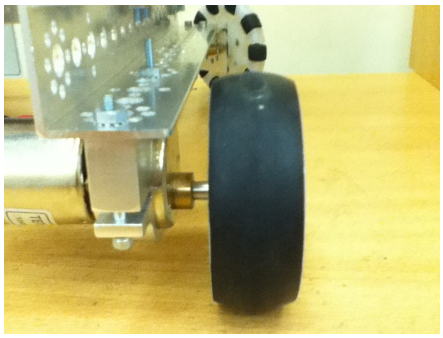


Рисунок 22.
Задний мотор.

3.3. Передние колеса были повёрнуты согласно пункту 2.12 от 2.12.2012.

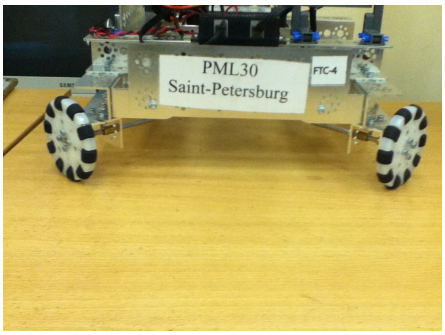


Рисунок 23.
Передние колёса.

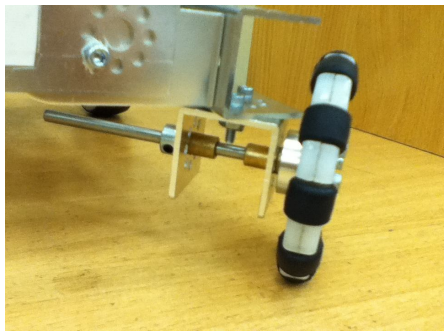


Рисунок 24.
Задние колёса.

3.4. В таком положении робот занимает больше места. Было проверено, что робот влезает в допустимые габариты - его ширина теперь 36 см.
3.5. Что бы сделать передаточное соотношение была разобрана верхняя часть.

4. Результат и итоги собрания:

4.1. Верхняя часть бала разобрана.
4.2. Передние колёса были повёрнуты.
4.3. Ось вращения задних колёса была перемещена.

Получившаяся конструкция:

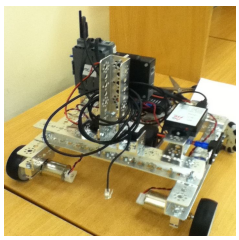


Рисунок 21. Без верхней части.

5. Идеи для следующих собраний:

5.1. Идеи реализации придаточного соотношения:

5.1.1. Сначала для реализации придаточного соотношения были взяты шестерёнки Lego, но потом были выбраны шестерёнки Tetrix, так как они более надёжны и коэффициент передаточного соотношения у них больше, чем у шестерёнок Lego.
5.1.2. Идея продеть между моторами ось, насадив на неё детали-переходники, с прикреплёнными к ним шестерёнками.
5.1.3. Было решено вместо 2-х моторов установить один, прикреплённый к вертикальной балке. Если он будет не справляться, то по бокам будут установлены ещё два мотора.

08.12.2012

1. Дата собрания и время начала:

Дата собрания: 8.12.2012.
Время начала: 16:00.

2. Поставленные цели:

2.1. Сделать передаточное соотношение.
2.2. Изменить программу под нужное число моторов Lego.

3. Реализация целей:

3.1. На вертикальную балку были установлены большие шестерёнки.

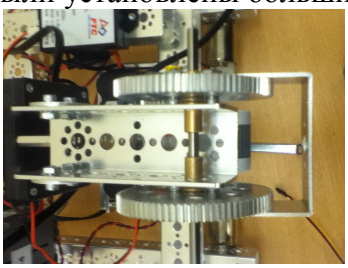


Рисунок №22. Большие шестерёнки.

3.2. Снизу был установлен мотор Lego, с маленькими шестерёнками. Из-за того, что размеры деталей Lego не пропорциональны Tetrix, детали-переходники на оси пришлось устанавливать не симметрично, иначе шестерёнки не будут совпадать.

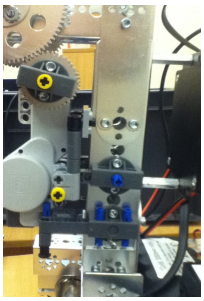


Рисунок 23.
Один мотор Lego вид справа.

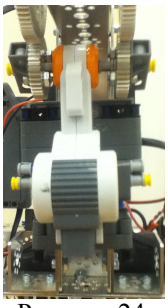


Рисунок 24.
Один мотор Lego вид сзади.

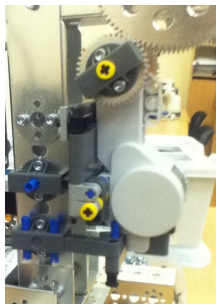


Рисунок 25.
Один мотор Lego слева.

3.3. К получившейся конструкции была прикреплёна рейка, на конце с захватом для колец.
3.4. Система была протестирована, и оказалось, что один мотор не может поднять конструкцию. Тогда был установлен второй мотор сбоку.
3.5. Система с двумя моторами не работала, поэтому был установлен третий мотор:

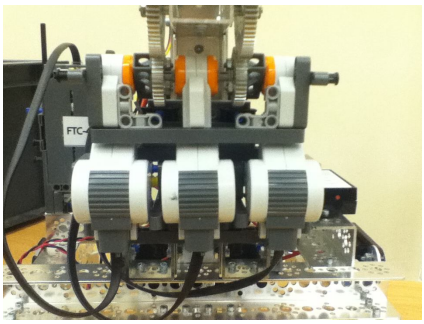


Рисунок 26
Три мотора Lego вид сзади.

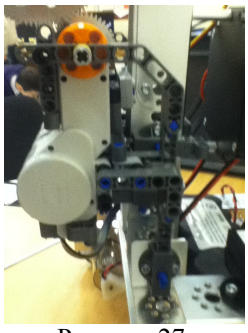


Рисунок 27.
Три мотора Lego вид сбоку.

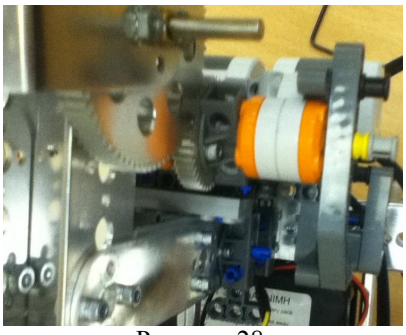


Рисунок 28.
Три мотора Lego вид сверху.

3.6. Три мотора не поднимали конструкцию, и так как мы не можем установить больше моторов, то было решено установить противовес. В качестве противовеса были взяты колёса из набора Tetrix.

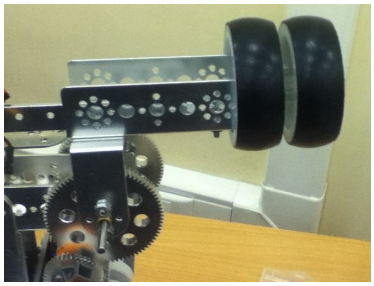


Рисунок 29. Противовес.

С противовесом моторы поднимали и опускали конструкцию.

3.7. Была измерена длина и высота робота. Оказалось, что он не помещается в параметры.

3.8. Балка, на которой закреплён захват, была сдвинута так, чтобы длина робота была меньше 18 дюймов (правило RG4). Теперь его длина составляет 46 см. то есть робот помещается в параметры.

3.9. Было проверено, что со сдвинутой балкой робот может брать кольца со среднего по высоте крючка.

3.10. Была реализована программа, по управлению роботом через Bluetooth, с тремя Lego моторами.

4. Результат и итоги собрания:

4.1. Было установлено 3 мотора Lego.
4.2. Было реализовано передаточное соотношение.
4.3. Был установлен противовес, и конструкция заработала.
4.4. Была реализована программа, по управлению роботом.

5. Идеи для следующих собраний:

5.1. Решено изменить захват, поставив посередине перегородку, что бы каждое кольцо находилось в своем отсеке.

15.12.2012

1. Дата собрания и время начала:

Дата собрания: 15.12.2012.
Время начала: 16:00.

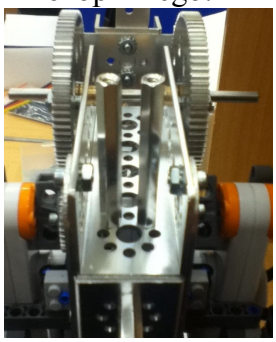
2. Поставленные цели:

2.1. Изменить захват.
2.2. Реализовать программу для управления роботом через новый джойстик.

3. Реализация целей:

Был приобретён джойстик Logitech F-310, который используется на соревнованиях.

3.1. Для того, что бы робот мог повесить кольцо на нижний крючок, не используя сервомоторы Lego, (сложно привести их в нужное положение, в частности это нужно для автономного периода), была установлена подставка для балку, положение которой изменяют моторы Lego:



Рисуок 30. Подставка под балку.

3.2. По краям захвата с обеих сторон были установлены перегородки.
3.3. Захват для колец был закреплён более прочно, также была удлинена балка Lego, которая соединяла захват с сервоприводом.

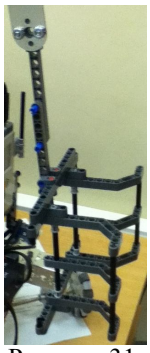


Рисунок 31.
Захват. Вид сбоку.

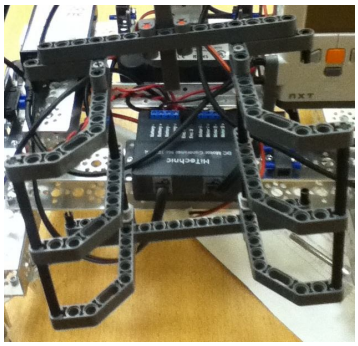


Рисунок №32.
Захват. Вид спереди.

3.4. Была реализована программа управления роботом с моторами Lego и новым джойстиком.

4. Результат и итоги собрания:

4.1. На нижний крючок робот вешает кольцо, не используя моторы Lego.

4.1. Каждое кольцо находится в своём отсеке.

5. Идеи для следующих собраний:

5.1. Появилась идея приобрести и установить датчики веса, что бы зарабатывать дополнительные очки. Пока у нас нет датчиков веса, решили пока поставить датчики касания, а потом заменить их.

5.2. Решено приобрести второй такой джойстик и реализовать программу для двух операторов, что бы было легче управлять им.

5.3. Создать рисунок изменённого робота в Creo.

10.01.2013

1. Дата собрания и время начала:

Дата собрания: 10.1.2013.
Время начала: 16:00.

2. Поставленные цели:

2.1. Установить два датчика касания на захват для колец.
2.2. Подключить их к NXT.
2.3. Написать программу для отличия тяжелого кольца от лёгкого.

3. Реализация целей:

3.1. Были закреплены 2 датчика касания. Чтобы кольцо попадало на датчик сверху, была закреплена деталь.

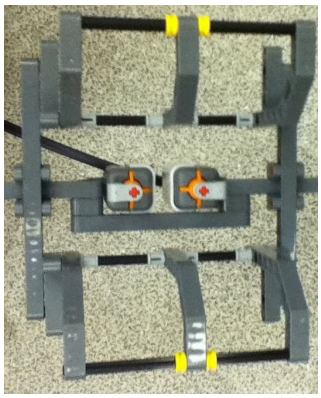


Рисунок 33.
Датчики касания. Вид сверху.

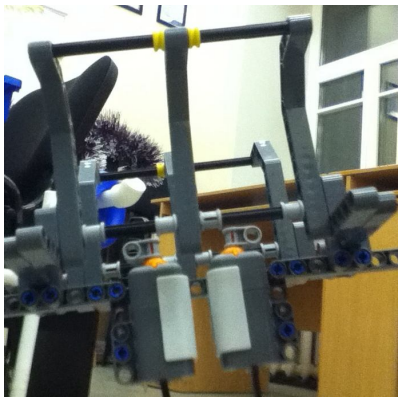


Рисунок 34
Датчики касания. Вид сбоку.

3.2. Блок NXT был перевёрнут входами для датчиков наверху, чтобы провода доставали до датчиков. Однако в самом верхнем положении провода не доставали.

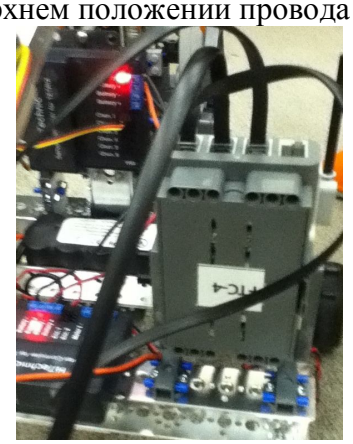


Рисунок 35. Перевёрнутый NXT.

3.3. Провод к сервомотору Tetrix был прикреплён стяжками к подвижной балке.

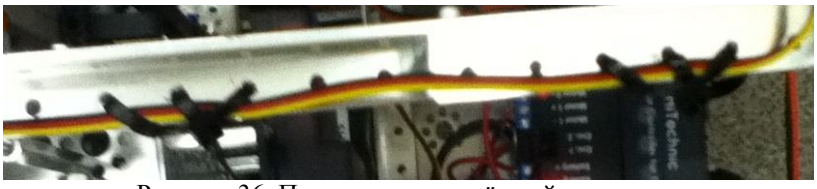


Рисунок 36. Провод, прикреплённый стяжками.

3.4. Робот не влезал по размерам в длину в сложенном положении из-за захвата, поэтому захват был повёрнут на 90º относительно сервопривода.

3.5. Теперь начальное положение захвата изменилось, и чтобы захват не зацепился в процессе раскладывания, сервоконтроллер был перемещён на вертикальную балку:

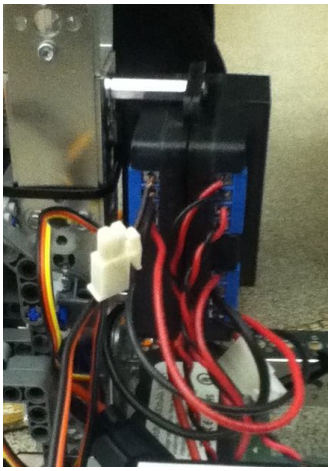


Рисунок 37.
Два блока Tetrix. Вид сбоку.

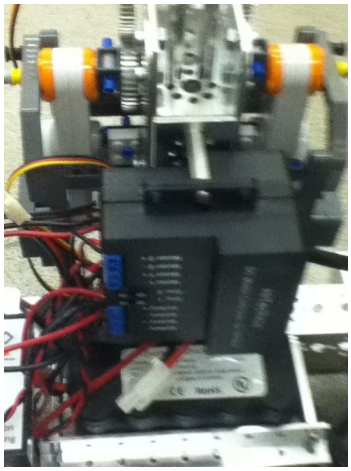


Рисунок 38.
Два блока Tetrix. Вид спереди.

3.6. Блок NXT мешал захвату раскладываться, поэтому он был перемещён.

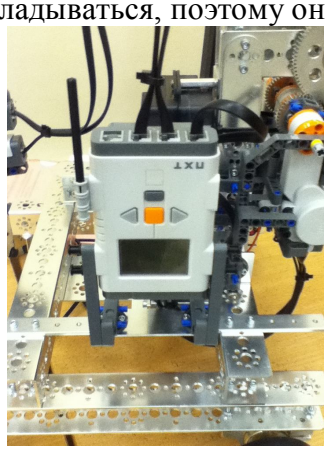


Рисунок 39. Перемещённый блок NXT.

3.7. Блок Samantha был передвинут вдоль балки, на которой установлен, поближе к моторам Lego, что бы не мешать раскладываться захвату.

3.8. Провода от датчиков касания были закреплены между собой стяжками.

3.9. Была написана часть программы, отвечающая за отличие тяжелого кольца от лёгкого. Принцип работы: если нажимается первый датчик касания, то робот издаёт один звук, если нажимается второй датчик касания, то робот издаёт другой звук.

```
...
if(SensorValue(S2)>0)
  PlaySound(soundBeepBeep);
if(SensorValue(S3)>0)
  PlaySound(soundBlip);
```

. . .

3.10. При проверке оказалось, что когда захват неподвижен, датчики касания реагируют только на тяжёлые кольца. Поэтому было решено использовать их как датчики веса.

3.11. Для того, чтобы снимать кольца с крючка и вешать их а нижний крючок не используя сервомоторы Lego, была изменена подставка:

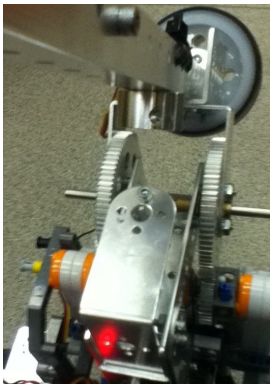


Рисунок 40.
Подставка. Поднятое состояние.

Рисунок 41
Подставка. Опущенное состояние.

Прежнюю подставку пришлось заменить, так как изменился угол поворота захвата.

4. Результат и итоги собрания:

4.1. Когда в захвате находится тяжёлое кольцо, робот издаёт звук.

4.2. Нет никаких препятствий при разворачивании захвата.

4.2. На нижний уровень робот вешает кольцо, не используя моторы Lego.

5. Идеи для следующих собраний:

5.1. Установить спереди грузы, чтобы робот не перевернулся при резкм торможении (основная масса робота сосредоточена в задней части).

5.2. Так как для автономного периода нам понадобится ещё 2 датчика: ИК-датчик и цветовой датчик, для проезда по линии, а вход для датчика остался только один, то для решения этой проблемы было решено приобрести расширитель для датчиков.

12.01.2013

1. Дата собрания и время начала:

Дата собрания: 12.1.2013.
Время начала: 15:00.

2. Поставленные цели:

2.1. Придумать и реализовать программу для автономного периода.
2.2. Придумать и реализовать конструкцию, с помощью которой кольцо для автономного периода будет держаться в захвате.

3. Реализация целей:

3.1. Пока у нас нет ИК-датчика, было решено реализовать программу с сонаром. Сонар был установлен на оси вращения задних колёс.
3.2. Была реализована программа: робот проезжает немного вперёд, поворачивает, едет пока не увидит сонаром препятствие, поворачивается, разворачивает захват и едет вперёд. (смотри рисунок 42). В наборе Tetrix нет датчиков, считывающих обороты колеса, поэтому мы использовали функцию “wait”. Из-за этого поороты каждый раз немножко отличались.
Движение робота в автономном периоде относительно поля из разных позиций:

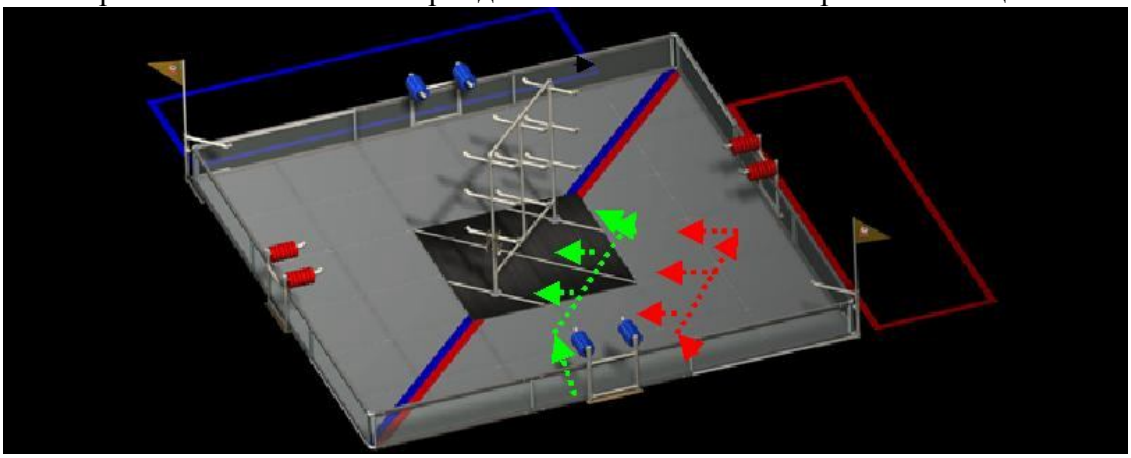


Рисунок 42. Схема движения в автономном периоде.

3.3. Была реализована конструкция, с помощью которой кольцо для автономного периода держится в захвате. Она представляет из себя нитку, привязанную с одной стороны, продетую через кольцо и с другой стороны закреплённой балкой, которая привязана к манипулятору другой ниткой.
При разворачивании кольца нитка дергает за штифт, тот выскакивает и другая нитка перестаёт быть закреплённой с одой стороны.

Рисунок 43.
Нитка. Вид сверху.

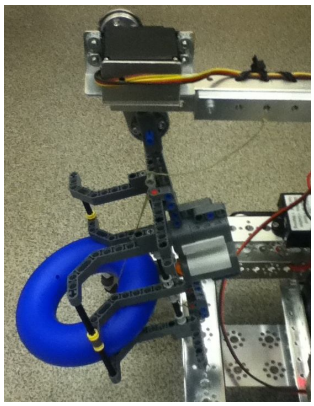


Рисунок 44.
Нитка. Вид сбоку.

3.4. Для того, что бы противовес не ударял по моторам Lego, и для того, чтобы манипулятор нельзя было отвести назад до конца, иначе его высота оказывается меньше, сзади был установлен ограничитель.

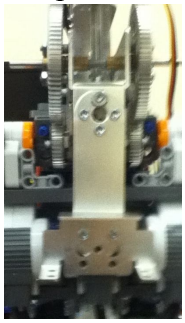


Рисунок 45.
Ограничитель. Вид сзади.

Рисунок 46.
Ограничитель. Вид сбоку.

4. Результат и итоги собрания:

4.1. Реализована программа для автономного периода с сонаром.
4.2. Реализован держатель для автономного кольца.

5. Идеи для следующих собраний:

5.1. Приобрести ИК-датчик.
5.2. Приобрести расширитель для датчиков.
5.3. Написать программу для автономного периода с использованием ИК-датчика и датчика цвета.
5.4. Переставить ведущие колёса вперёд, что бы при поворотах робота вправо и влево, положение захвата изменялось на меньшее расстояние.

17.01.2013

1. Дата собрания и время начала:

Дата собрания: 17.01.2013.
Время начала: 16:00.

2. Поставленные цели:

2.1. Переставить ведущие колёса вперёд.
2.2. Прикрепить сонар над осью вращения ведущих колёс, не используя переходники от Tetrix к Lego (они закончились, а ещё потребуется один для закрепления датчика освещённости).
2.3. Установить датчик освещённости на линии симметрии захвата.
2.4. Сделать конструкцию для того, чтобы робот мог перемещать упавшие кольца на центральные ворота.

3. Реализация целей:

3.1. Ведущие колёса были переставлены вперёд, а передние назад. Также сонар был переставлен над осью вращения передних колёс.

Рисунок 47. Ведущие колёса спереди.

Оказалось, что когда ведущие колёса спереди, то они проскальзывают, потому что большая часть массы робота сосредоточена в задней части. Поэтому колёса были переставлены назад.

3.2. Сонар был закреплён без использования переходника.

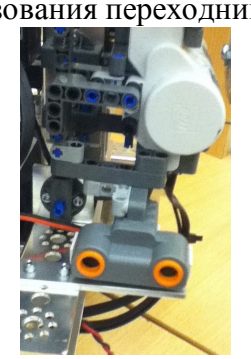


Рисунок 48. Закреплённый сонар.

3.3. Спереди была установлена пластина, которая будет собирать упавшие кольца на центральные ворота.

Рисунок 49. Передняя пластина.

3.4. Посередине робота, за передней пластиной был установлен датик освещённости.

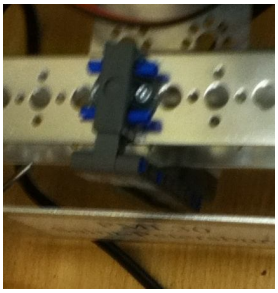


Рисунок 50.
Датчик освещённости. Вид сверху.

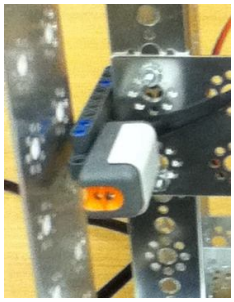


Рисунок 51.
Датчик освещённости. Вид снизу.

4. Результат и итоги собрания:

4.1. Идея перестановки ведущих колёс вперёд была отклонена.
4.2. Сонар был закреплён без использования переходников.
4.3. Спереди была установлена пластина.
4.4. Был закреплён датчик освещённости.

5. Идеи для следующих собраний:

5.1. Реализовать движение по линии.
5.2. Развести управление на два джойстика.
5.3. Приобрести датчик, для считывания градуса, на который повернулось колесо.

27.01.2012

1. Дата собрания и время начала:

Дата собрания: 27.01.2013.
Время начала: 16:00.

2. Поставленные цели:

2.1. Реализовать движение по линии.
2.2. Развести управление на два джойстика.
2.3. Написать автономный период с функцией “nMotorEncoder”, вместо функции “wait”

3. Реализация целей:

3.1. Были сделаны центральные ворота. Но вместо белых линой на черном фоне - чёрные линии на жёлтом фоне.

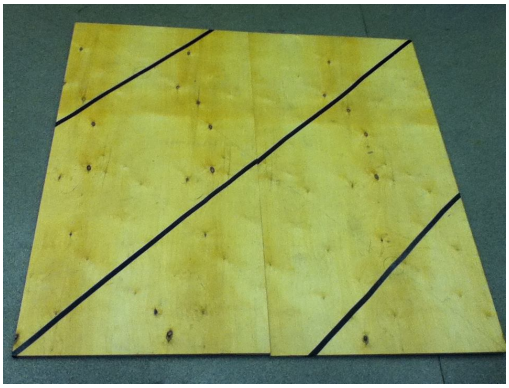


Рисунок 52.
Центральные ворота.

3.2. Когда робот подъезжал к центральным воротам на 45º, то он не мог забраться на ступеньку, потому что ближнее переднее колесо становилось параллельно ступеньке. Было решено установить передние колёса перпендикулярно задним.

Рисунок 53.
Передние колёса, перпендикулярное положение.
Вид спереди.

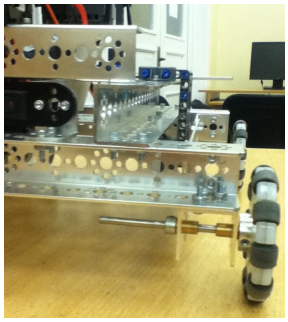


Рисунок 53.
Передние колёса, перпендикулярное положение.
Вид сбоку.

3.3. Управление было разведено на два джойстика. С первого джойстика выполняется управление движением робота - двумя сервомоторами Tetrix. Со второго джойстика выполняется управление захватом – сервоприводом и манипулятором – моторами Lego.
3.4. Начальное расположение робота было изменено на 180º.
3.5. Было реализовано движение по линии.

Алгоритм действий робота в автономном режиме: робот разворачивается на 45º одним колесом, считывает показания датчика освещенности, едет задним ходом пока не сработает Sonar, поворачивает на 90º, разворачивает захват, едет вперёд пока показания датчика освещённости совпадает с первым значением, потом проезжает 10 сантиметров, чтобы забраться на ступеньку, потом ищет линию поворотами, когда находит её то двигается по ней на релейном регуляторе.

3.6. Остановка робота после того, как он повесил кольцо, не была реализована. Предполагается, что робот остановится, когда кольцо перестанет касаться датчика касания.

3.6. Был приобретён датчик, для считывания градуса, на который повернулось колесо. Функция “wait” была заменена на функцию “nMotorEncoder”.

4. Результат и итоги собрания:

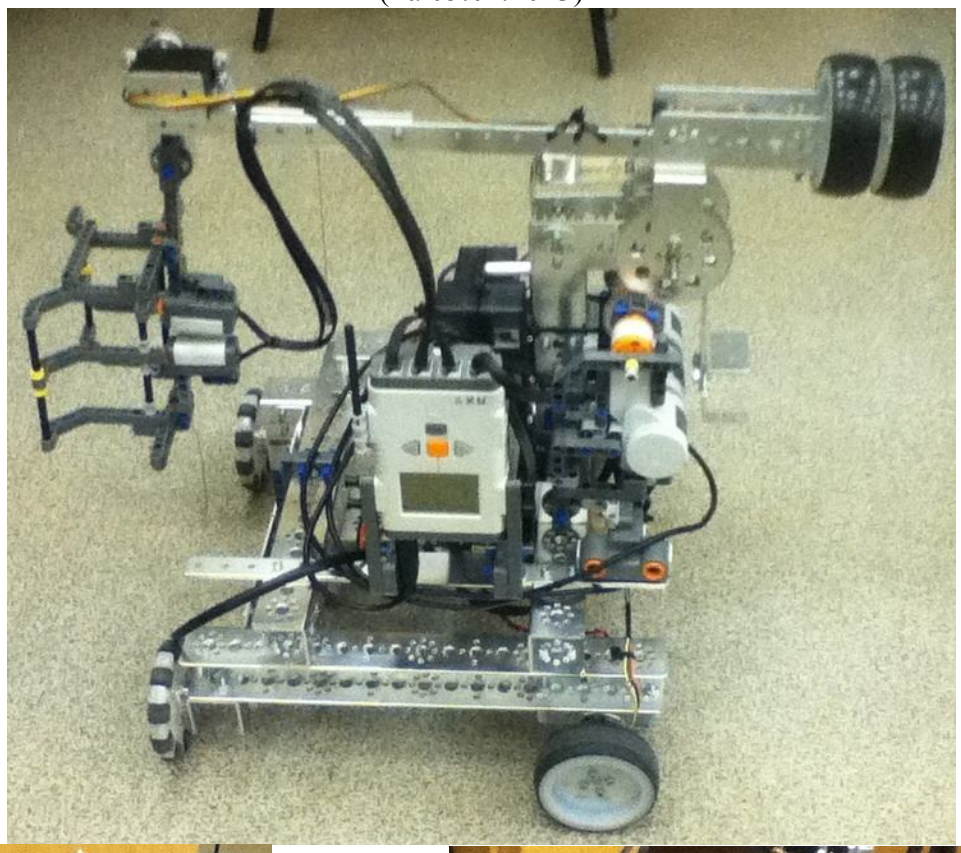
4.1. Передние колёса были повернуты.
4.2. Начальное расположение робота было изменено.
4.3. Было реализовано движение по линии.
4.4. Алгоритм действий робота в автономном режиме был изменён.
4.5. Функция “wait” была заменена на функцию “nMotorEncoder”.
4.6. Управление было разведено на два джойстка.

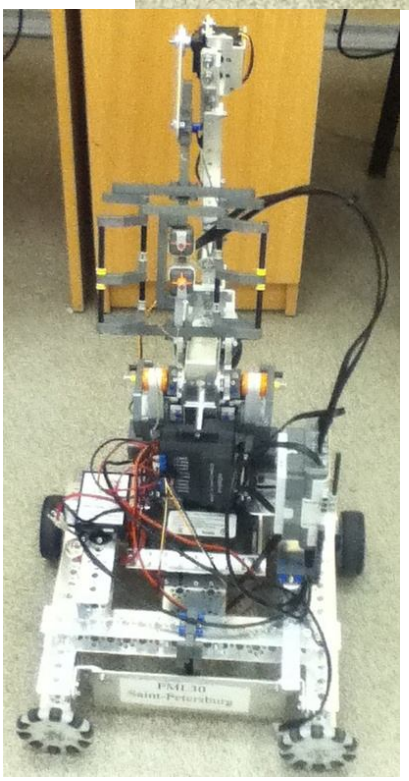
5. Идеи для следующих собраний:

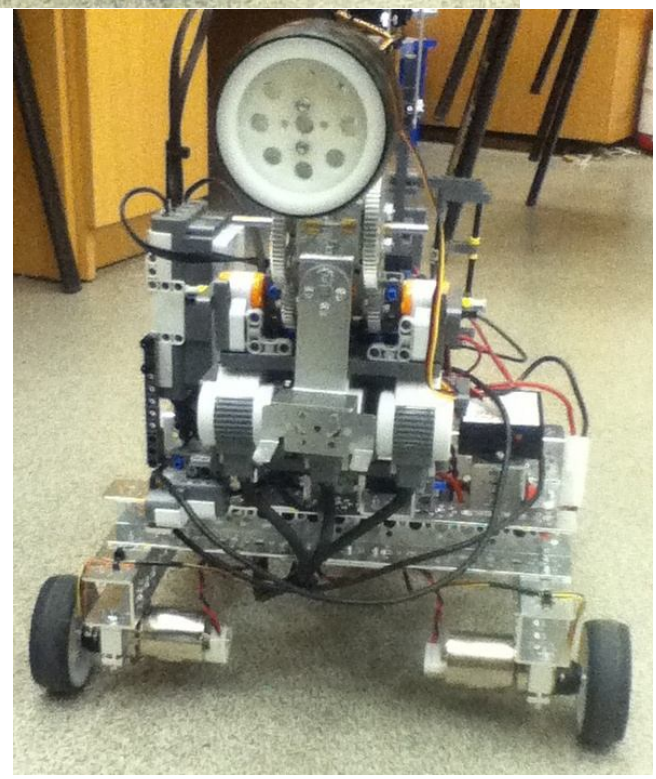
5.1. Реализовать остановку робота, после того, как он повесил кольцо.

6. Итоговая модель робота

(на 05.02.2013)







7. Итоговая версия программы

(на 05.02.2013)

7.1 Управляемый период (сырой код, без оболочки):

```
#pragma config(Hubs,  S1, HTMotor,  HTServo,  none,     none)
#pragma config(Sensor, S2,     ,               sensorTouch)
#pragma config(Sensor, S3,     ,               sensorTouch)
#pragma config(Motor,  motorA,          ,             tmotorNXT, openLoop)
#pragma config(Motor,  motorB,          ,             tmotorNXT, openLoop)
#pragma config(Motor,  motorC,          ,             tmotorNXT, openLoop)
#pragma config(Motor,  mtr_S1_C1_1,     motorD,        tmotorTetrix, PIDControl)
#pragma config(Motor,  mtr_S1_C1_2,     motorE,        tmotorTetrix, PIDControl)
#pragma config(Servo,  srvo_S1_C2_1,    servo1,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_2,    servo2,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_3,    servo3,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_4,    servo4,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_5,    servo5,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_6,    servo6,               tServoNone)
//*!!Code automatically generated by 'ROBOTC' configuration wizard               !!*//

#include "JoystickDriver.c"

task main()
{
  float k = 300, n = 0,d = nMotorEncoder(motorA),q=1.582, m, r,f = 0;
  while (true)
  {

    getJoystickSettings(joystick);
       r = f/255 + 172;

    if(r < 1)
       f = -170*255;

    if(r > 254)
       f = 81*255 ;

    r = f/255 + 172;
    servo[servo1] = r;
    d = nMotorEncoder(motorA) ;
    servo[servo1] = r;

    if(joystick.joy1_y1 > 0)
    {
      motor[motorB] = joystick.joy1_y1/q;
      motor[motorC] = joystick.joy1_y1/q;
      motor[motorA] = joystick.joy1_y1/q;
   }

   if(joystick.joy1_y1 < 0)
    {
      motor[motorB] = joystick.joy1_y1/(q*4);
      motor[motorC] = joystick.joy1_y1/(q*4);
      motor[motorA] = joystick.joy1_y1/(q*4);
   }

    if (nMotorEncoder(motorA) > d)
    {
      f += k * (nMotorEncoder(motorA) - d);
    }

    if (nMotorEncoder(motorA) < d)
    {
      f += k * (nMotorEncoder(motorA)-d);
    }

    if (joy1btn(7) > 0)
      f += 30;

    if (joy1btn(8) > 0 )
      f -=30;
```

```
    if (nMotorEncoder(motorA) > d)
    {
      f += k * (nMotorEncoder(motorA) - d);
    }

    if (nMotorEncoder(motorA) < d)
    {
      f += k * (nMotorEncoder(motorA)-d);
    }

    if (joy2Btn(6) > 0)
        {
                motor[motorD] = joystick.joy2_y1 / 10 - joystick.joy2_x2 / 10;
      motor[motorE] = -joystick.joy2_y1 / 10 - joystick.joy2_x2 / 10;
        }
    else
    {
      motor[motorD] = joystick.joy2_y1 / 5 - joystick.joy2_x2 / 5;
      motor[motorE] = -joystick.joy2_y1 / 5 - joystick.joy2_x2 / 5;
    }

    if (joy2Btn(5) > 0)
    {
      motor[motorD] = 0;
       motor[motorE] = 0;
    }

    if (joy2Btn(2) > 0)
    {
        while (joy2Btn(2) > 0)
        {
          motor[motorD] = -30;
          motor[motorE] = 30;

          if (joy2Btn(5) > 0)
         {
           motor[motorD] = 0;
             motor[motorE] = 0;
         }
      }
    }
    if (joy2Btn(4) > 0)
    {
        while (joy2Btn(4) > 0)
        {
        motor[motorD] = 30;
        motor[motorE] = -30;

        if (joy2Btn(5) > 0)
         {
           motor[motorD] = 0;
             motor[motorE] = 0;
         }
      }
    }
    if (SensorValue(S3) > 0)
        PlaySound(soundBlip);

    if (SensorValue(S2) > 0)
        PlaySound(soundBeepBeep);
  }
}
```

7.2 Автономный период (сырой код, без оболочки):

```
#pragma config(Hubs,  S1, HTMotor,  HTServo,  none,     none)
#pragma config(Sensor, S2,     ,               sensorLightActive)
#pragma config(Sensor, S3,     ,               sensorTouch)
#pragma config(Sensor, S4,     sonar,          sensorSONAR)
#pragma config(Motor,  mtr_S1_C1_1,     motorD,        tmotorTetrix, PIDControl)
#pragma config(Motor,  mtr_S1_C1_2,     motorE,        tmotorTetrix, PIDControl)
#pragma config(Servo,  srvo_S1_C2_1,    servo1,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_2,    servo2,               tServoNone)
```

```
#pragma config(Servo,  srvo_S1_C2_3,    servo3,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_4,    servo4,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_5,    servo5,               tServoNone)
#pragma config(Servo,  srvo_S1_C2_6,    servo6,               tServoNone)
//*!!Code automatically generated by 'ROBOTC' configuration wizard               !!*//

void Rotate1 (float L, float d, float a)
{
  long n = nMotorEncoder(motorD) = 0;
        while (n > (-1 * (d * a / L * 8)))
        {
                nxtDisplayCenteredTextLine(3, "%d", n);
    motor[motorD] = -30;
    motor[motorE] = 0;
    n = nMotorEncoder(motorD);

  }

  motor[motorD] = 0;
  motor[motorE] = 0;
}

void Rotate2 (float L, float d, float a)
{
  long n = nMotorEncoder(motorD) = 0;
        while (n > (-1 * (d * a / L * 4)))
        {
                nxtDisplayCenteredTextLine(3, "%d", n);
    motor[motorD] = -30;
    motor[motorE] = -30;
    n = nMotorEncoder(motorD);

  }

  motor[motorD] = 0;
  motor[motorE] = 0;
}
/*void a (float R, float d)
{
  while (nMotorEncoder(motorD) <
  {
        motor[motorD] = -20;
    motor[motorE] = 20;
  }

  motor[motorD] = 0;
  motor[motorE] = 0;
}*/

task main()
{
        float up = 0, k = 0.1, L = 6.9, d = 30.9, light, light2; //L - lenght (wheels), d - diameter of
wheel
        int black = 65;

        servo[servo1] = 172;

         motor[motorD] = 0;
   motor[motorE] = 0;

  Rotate1 (L, d, 45);

  light = SensorValue(S1);

  while(SensorValue(sonar) > 30)
  {
        motor[motorD] = -40;
        motor[motorE] = 40;
  }

  motor[motorD] = 0;
  motor[motorE] = 0;

  Rotate2 (L, d, 90);

  servo[servo1] = 20;
```

```
while ((SensorValue(S2) - light > 45)
{
      motor[motorD] = 40;
      motor[motorE] = -40;
}

motor[motorD] = 0;
motor[motorE] = 0;
nMotorEncoder(motorD) = 0;

while (nMotorEncoder(motorD) < 1200)
{
      motor[motorD] = 40;
      motor[motorE] = -40;
}

 motor[motorD] = 0;
 motor[motorE] = 0;
 nMotorEncoder(motorD) = 0;

 if (SensorValue(S2 > black)
 {
   while ((nMotorEncoder(motorD) < 500) || (SensorValue(S2) > black)
   {
     motor[motorD] = 20;
     motor[motorE] = 0;
   }

   motor[motorD] = 0;
   motor[motorE] = 0;

   if (SensorValue(S2 > black)
   {
     while ((nMotorEncoder(motorD) > 0) || (SensorValue(S2) > black)
     {
       motor[motorD] = -20;
       motor[motorE] = 0;
     }
   }

   motor[motorD] = 0;
   motor[motorE] = 0;
   nMotorEncoder(motorE) = 0;

   if (SensorValue(S2 > black)
   {
     while ((nMotorEncoder(motorE) > -500) || (SensorValue(S2) > black)
     {
       motor[motorD] = 0;
       motor[motorE] = -20;
     }
   }
 }

 motor[motorD] = 0;
 motor[motorE] = 0;

 if (SensorValue(S2 > black)
 {
   while ((nMotorEncoder(motorE) < 0) || (SensorValue(S2) > black)
   {
     motor[motorD] = 0;
     motor[motorE] = 20;
   }
 }

 motor[motorD] = 0;
 motor[motorE] = 0;

 while (true)
 {
   if ((SensorValue(S1) < 62) && (SensorValue(S1) > 58))
   {
     motor[motorD] = -20;
     motor[motorE] = 20;
   }
```

```
    else if (SensorValue(S1) > 51)
    {
      motor[motorD] = 3;
      motor[motorE] = -10;
    }

    else
    {
      motor[motorD] = 10;
      motor[motorE] = -3;
    }

   }


  /*servo[servo1] = 10;
  wait10Msec(20);
  motor[motorE] = -50;
  motor[motorD] = 50;

  while(SensorValue[S3] < 1)
  {
       while(sensorvalue[S2]<35)
       {
     motor[motorD] = 20;
        motor[motorE] = 20;
    }
    up = SensorValue[S2]*k ;
    motor[motorE] = 50 - up;
    motor[motorD] = 50 + up ;
  }*/
}
```