Научное общество учащихся «Эврика»
Муниципальное автономное образовательное учреждение
«Лицей № 38»
Советского района г.Н.Новгорода

Роботизированная платформа на колесах Бенгта Илона

Выполнил: Павловский Иван, ученик 10 класса

Научный руководитель: Еделев Ю.А., зам. главного инженера КВЦ

Еделев А.Ю., учитель физики

Нижний Новгород

2024 г.

Содержание

Введение

В складских помещениях транспортировка груза осуществляется с помощью погрузчиков или тележек, требующих значительного пространства для совершения маневров, что значительно замедляет и осложняет работу на складах с узкими проёмами.

Для облегчения работы инженерами по всему миру создаются особые тележки и погрузчики, требующие минимального пространства для поворотов. Такое оборудование стоит значительно дороже стандартных аналогов, что увеличит затраты на производство на крупных промышленных заводах и фабриках.

Поэтому насущным становится вопрос об удешевлении и упрощении транспортировки грузов на промышленных складах, для чего необходимо создать такое роботизированное устройство, которое могло бы совершать маневры в тесных пространствах и имело бы дешевую себестоимость.

Цель работы: создать недорогое транспортировочное устройство, имеющее возможность свободного передвижения на малой площади.

Задачи работы:

- рассмотреть существующие устройства для транспортировки грузов
- рассчитать грузоподъемность предполагаемой модели
- разработать и создать собственную роботизированную модель
- провести ряд опытов с данной моделью
- сравнить расчётные характеристики с практическими

1 Стандартные складские транспортировочные устройства

1.1 Складской погрузчик

Обыкновенный складской погрузчик имеет грузоподъёмность от 1000 кг; его длина составляет от 4-х метров, а ширина – от 2-х метров. Для разворота на 180 градусов при непрерывном движении погрузчику требуется пространство 12-ти метров в длину и 5-ти метров в ширину; т.е. площадь разворота составляет порядка 60м^2. Такие условия зачастую отсутствуют на складах, ввиду чего погрузчик тратит лишнее время как на передний, так и на задний ход для совершения разворота на малой площади. Поэтому для перевозки тяжелых грузов чаще используют компактные тележки и платформы.

Рисунок 1. – Складской погрузчик

1.2 Складская тележка (платформа)

Стандартные складские тележки могут иметь габариты от 1м в длину и 0,8м в ширину, при этом достигать высоты 0,6м, что позволяет ей пробираться в самые узкие и низкие места. Благодаря большой погрузочной площади, платформа может перевозить грузы в несколько тонн.

Рисунок 2. – Складская тележка.

2 Существующие альтернативы стандартным тележкам и погрузчикам

2.1 Гусеничная тележка

Гусеничная тележка обладает возможностью вращения вокруг собственной оси, благодаря чему ей не требуется лишнее пространство для поворотов. Во избежание деформации и изгиба корпуса тележки, гусеницы должны находиться внутри корпуса, что делает тележку высокой и уменьшает ее проходимость. Кроме того, гусеницы могут повредить поверхность, по которой передвигается тележка. Также тележка требует двигателя с высоким крутящим моментом для преодоления силы трения покоя.

Рисунок 3. – Гусеничная тележка

2.2 Тележка с четырьмя независимыми приводами

Колеса данной тележки вращаются независимо друг от друга, благодаря четырем двигателям, что дает возможность вращения вокруг своей оси без передвижения. В отличие от гусеничной тележки, такая платформа, оснащённая колесами, не наносит вред покрытию и не требует двигателя с высоким крутящим моментом. Эта тележка является отличным вариантом и не имеет существенных недостатков, но существуют и более совершенные конструкции.

Рисунок 4. – Тележка с четырьмя независимыми приводами

2.3 Тележка с двумя осями вращения колеса

Особенность таких платформ заключается в возможности бесповоротного передвижения в любом направлении. Поэтому, в отличие от предыдущих конструкций, эта тележка легче совершает маневры и передвижение. Однако для функционирования одной такой тележки требуется по два двигателя на колесо, чтобы осуществлять его вращение по двум осям, что существенно увеличивает стоимость тележки.

Рисунок 5. – Платформа с двумя осями вращения колеса

2.4 Платформа с колесами Илона

Платформа с омниколёсами (колесами Илона), благодаря их особенности, имеет возможность при четырёх двигателях вращаться и передвигаться в абсолютно любом направлении. Такая платформа является наилучшим выбором для складских помещений, так как обладает сочетанием высокой эффективности и дешевизны.

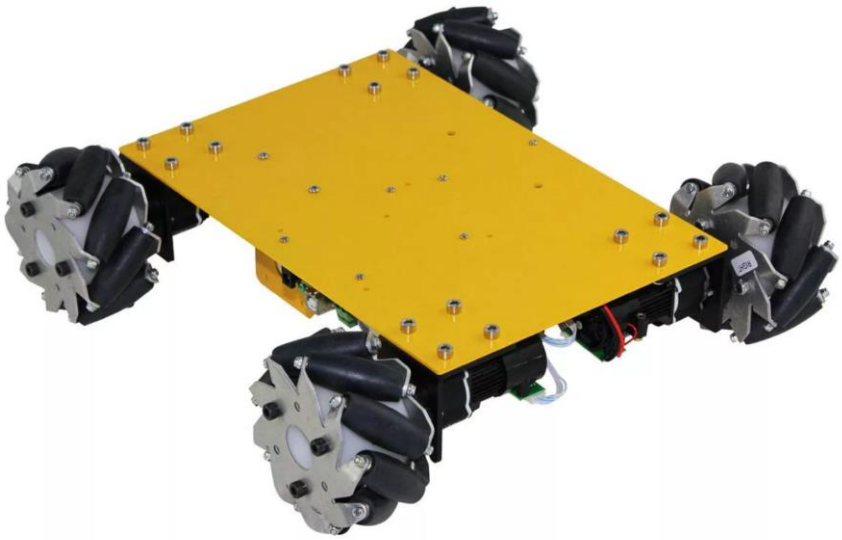


Рисунок 6. – платформа с колесами Илона

3 Выбор и анализ прототипа

Проанализировав существующие аналоги стандартным складским транспортировочным устройствам, я пришел к выводу, что самым эффективным и недорогим из них является платформа на колесах Бенгта Илона, т.к. она имеет высокую проходимость, безостановочно передвигается и вращается в любом направлении и требует всего 4 мотора.

3.1 Колесо Илона Бенгта

3.1.1 Конструкция и области применения колеса

Колесо Илона представляет собой круглый каркас, по длине окружности которого расположены валики, которые соприкасаются с поверхностью земли. Валики закреплены под углом 45 по отношению к оси вращения колеса, что позволяет платформе, благодаря регулированию скорости и направления каждого колеса, передвигаться в абсолютно любом направлении.

Такое колесо, изобретенное в 1973 году, не нашло широкого применения, несмотря на то, что в 1980-х ВМС США выкупили у шведского изобретателя Бенгта Илона патент на эту деталь. Сейчас лишь на некоторых складах эксплуатируются дорогостоящие платформы с подобными колесами.

Рисунок 7. – Колесо Илона

3.1.2 Движение платформы с колесами Илона Бенгта

Благодаря особенностям конструкции колёс, платформа имеет не только возможность передвижения в любом направлении, но и возможность выбора расположения вертикальной оси при вращении на месте, что позволяет максимально свободно передвигаться в условиях узких пространств.

Колеса Илона позволяют закрепить ось вращения тележки не только по её центру и около одного из колеса, но и в любой точке платформы.

Все способности этого колеса, открывают огромный потенциал для передвижения любого транспорта, и очевидно, что в скором будущем эта технология будет внедрена в широкий спектр транспорта специального назначения (внедорожник, марсоход, перронный автобус, погрузчик и т.д.)

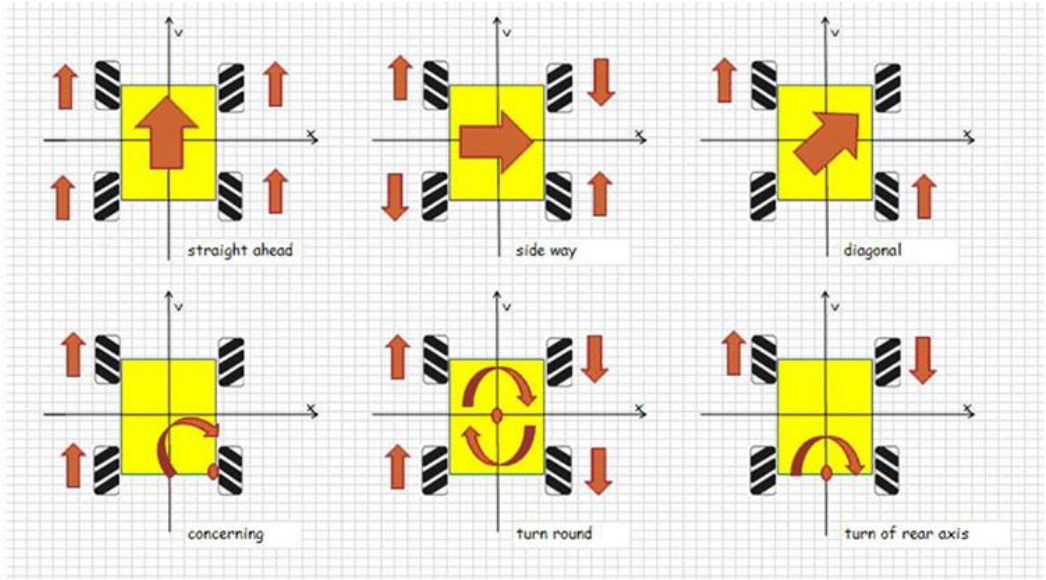


Рисунок 8. – Возможности перемещения транспорта на колесах Илона

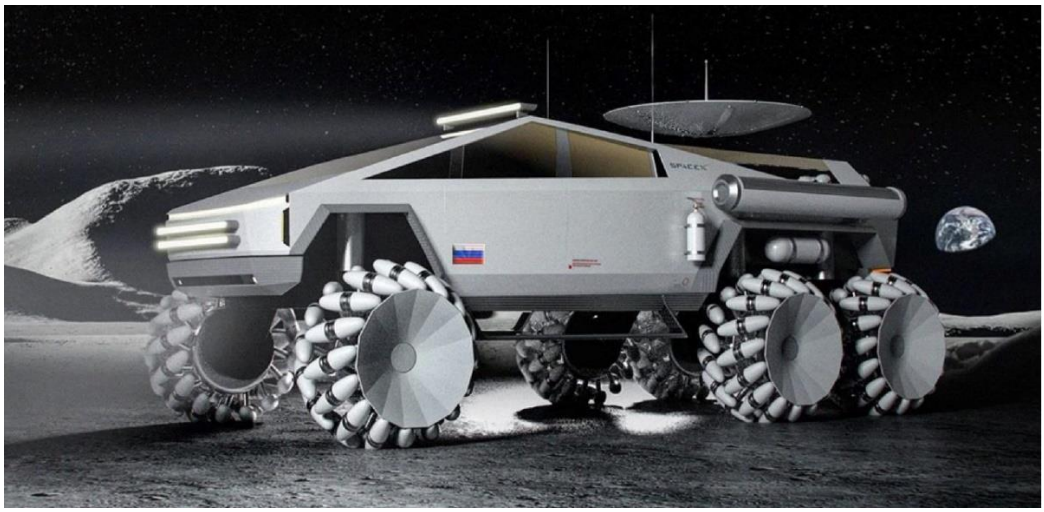


Рисунок 9. – Виртуальная модель лунохода на колесах Илона Бенгта

3.1.3 Теоретическое обоснование движения платформы

Для изучения движения платформы я изучил, что при вращении колеса, сила с которой оно отталкивается от поверхности направлена под углом 45 к оси вращения. На основании этого я получил равнодействующие движущие силы для каждого случая векторным методом. Также были получены теоретические моменты сил при вращении платформы. На данный момент ведется работа над анализом потерь при движении платформы на таких колесах.

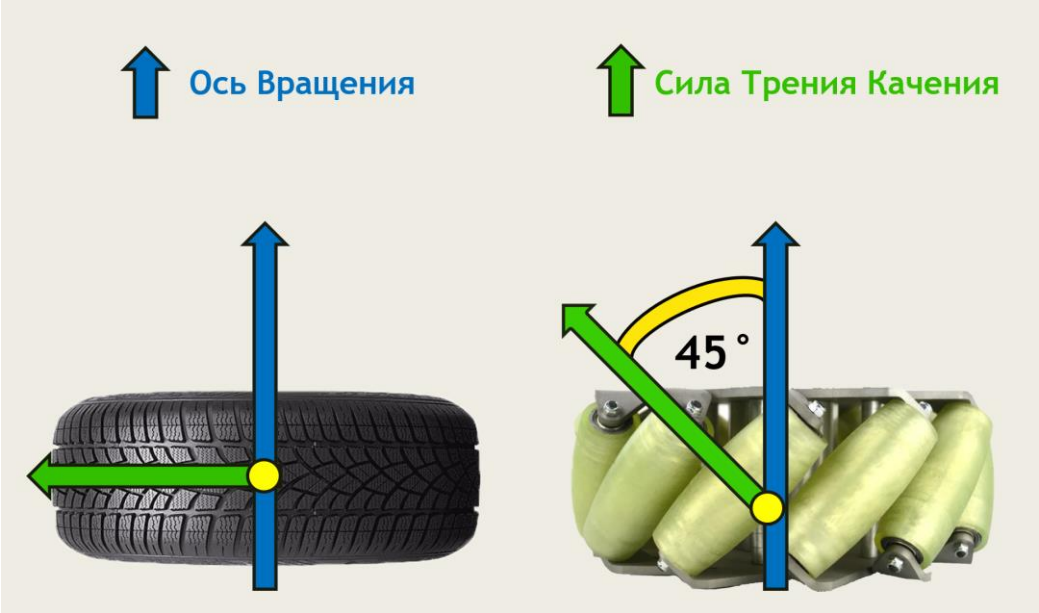


Рисунок 10. – Сравнение силы трения качения, действующей на обычное колесо и колесо Илона при вращении

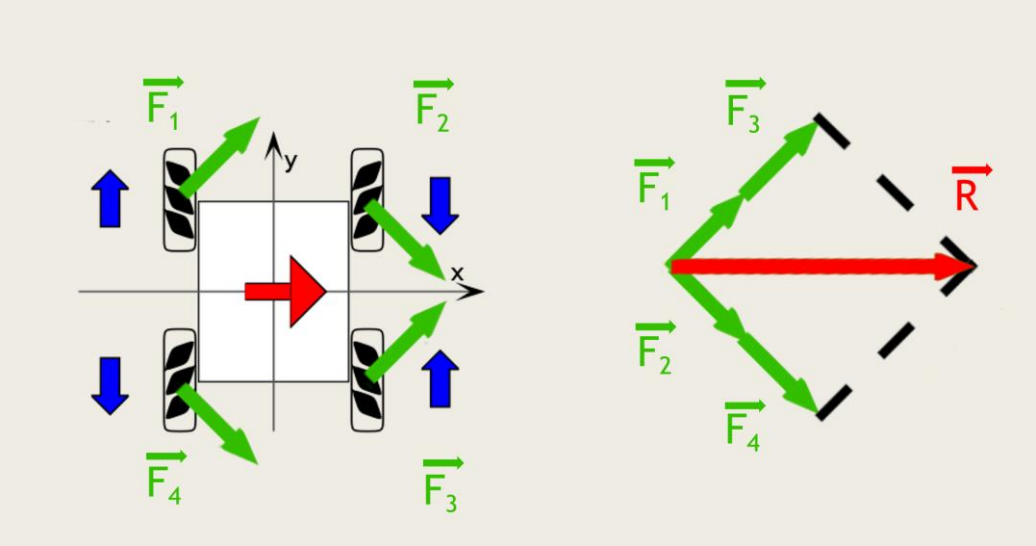


Рисунок 11. – Нахождение равнодействующей силы

4 Моделирование и конструирование собственного устройства

4.1 Каркас платформы

В качестве основы для платформы были выбраны металлические профили, которые позволяют защищать платформу от прогибов и выдерживать большую нагрузку (**9.1**). Профили были соединены винтами так, что каркас имел длину 80 см и ширину 60 см, что открывает для тележки возможность проехать в самые узкие и тесные места склада. По середине конструкции был установлен еще один профиль, для того, чтобы предотвратить деформацию точек платформы, на которые груз оказывает наибольшее давление. Сверху профили закрыты листом фанеры толщиной 10 мм, которая также закреплена винтами.

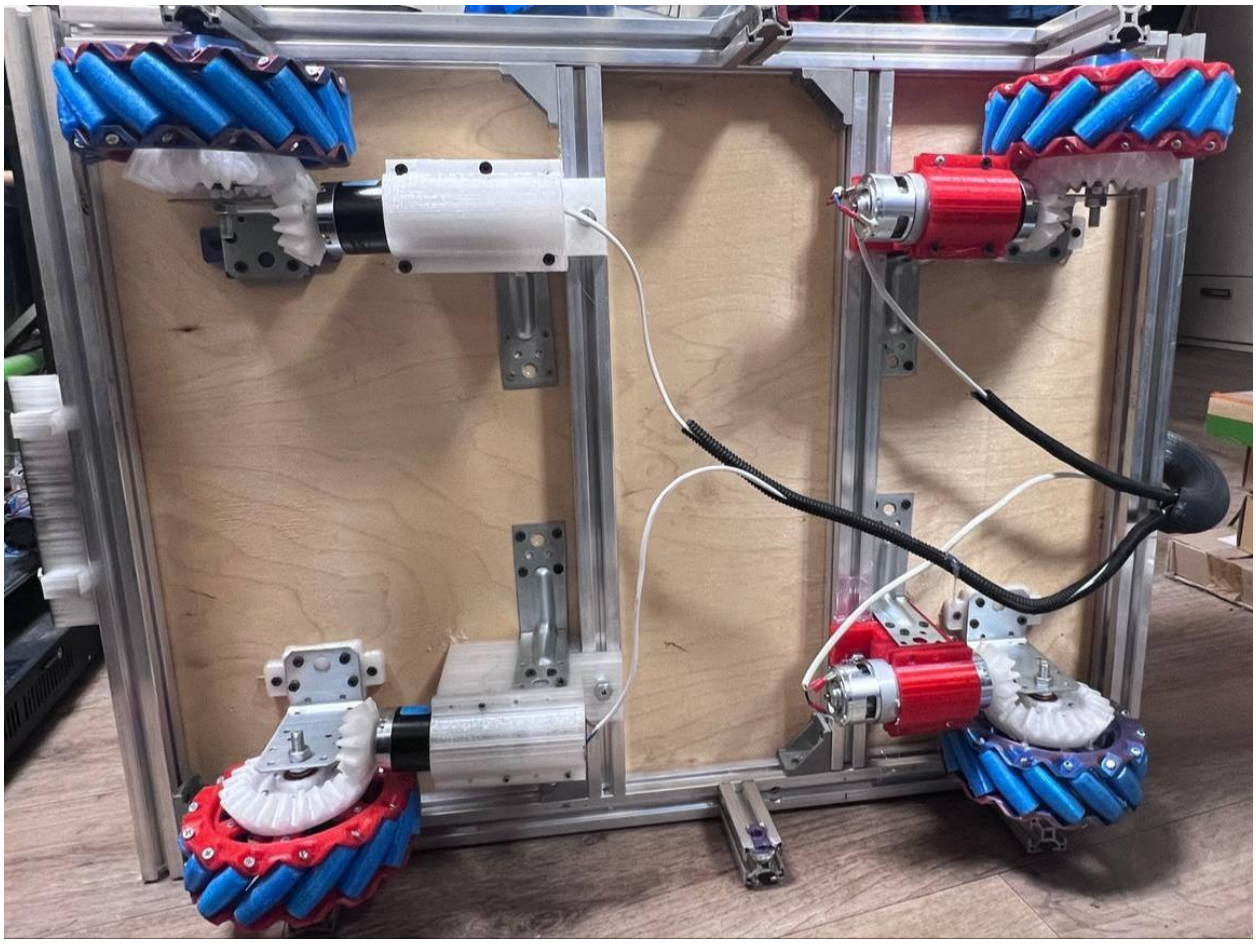


Рисунок 12. – картинка профилей и каркаса

4.2Колесо Илона

4.2.1 3D-модель колеса

Несмотря на то, что готовые модели подобных колес существуют в интернете, нами было принято решение смоделировать колесо самостоятельно, так как колеса грузовой тележки должны выдерживать большой вес, при этом не деформируясь. Колесо было смоделировано в программе Fusion 3D и состоит из 3-х частей: оснований колеса, 14-ти втулок и валиков, крепящихся на втулках. Габариты колеса 15:15:6 сантиметров.

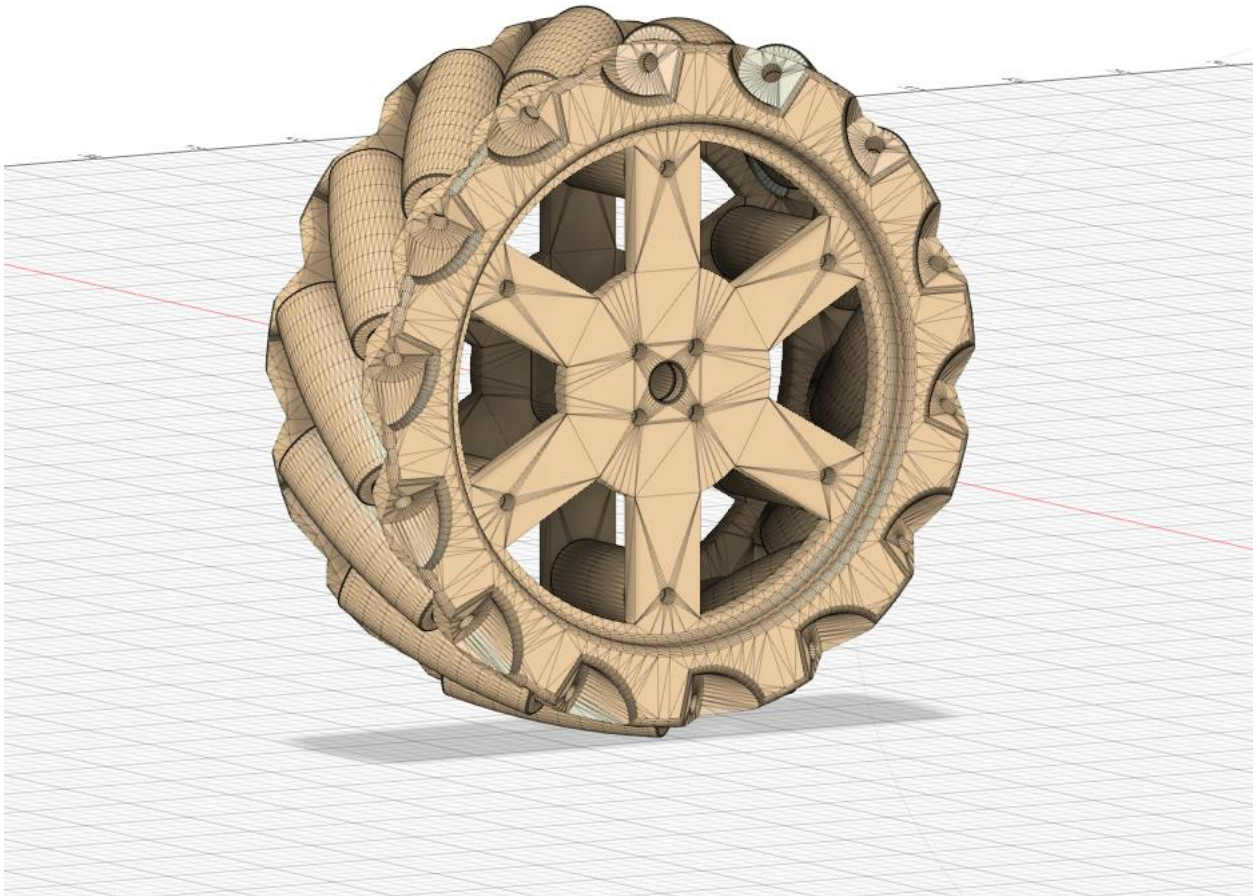


Рисунок 13. – 3D-модель колеса.

4.2.1.1 Выбор материала для каждого элемента колеса.

Для основания колеса и втулок был выбран пластик PLA, который обладает прочностью и дает минимальную усадку, что позволяет сделать колеса практически идентичными и облегчить задачи, связанные с управлением.

Для валиков мы выбрали пластик ТПУ. Он отличается от остальных видов пластика тем, что обладает высокой гибкостью, благодаря чему обеспечит хорошее сцепление с поверхностью и уменьшит давление, оказываемое на колесо. Помимо этого, пластик не боится воды.

Рисунок 14. – характеристики пластика

4.2.1.2 Сборка колеса

Для начала необходимо было нанизать валики на втулки, что осуществлялось с помощью пресса. Через втулки в свою очередь проходили металлические болты, которые соединяли детали с основаниями колеса. Кроме того, основания были скреплены между собой с помощью шести болтов, что помогает колесу распределять вес груза. Собранное колесо имеет массу 400 граммов, и выдерживает груз массой 50 килограммов.

4.3 Двигатели и передача

Для полного функционирования тележки требуется 4 двигателя. Нами были выбраны двигатели 775, на 185 оборотов в минуту с рабочим напряжением 12В. Они были закреплены на профилях с помощью пластиковых конструкций. Они приводят в действия колеса через зубчатые передачи, выполненные также из пластика. Такой вид передачи был выбран как лучшая альтернатива ремням, которые нарушают синхронность вращения колес и усложняют управление.

Рисунок 15. – двигатель.

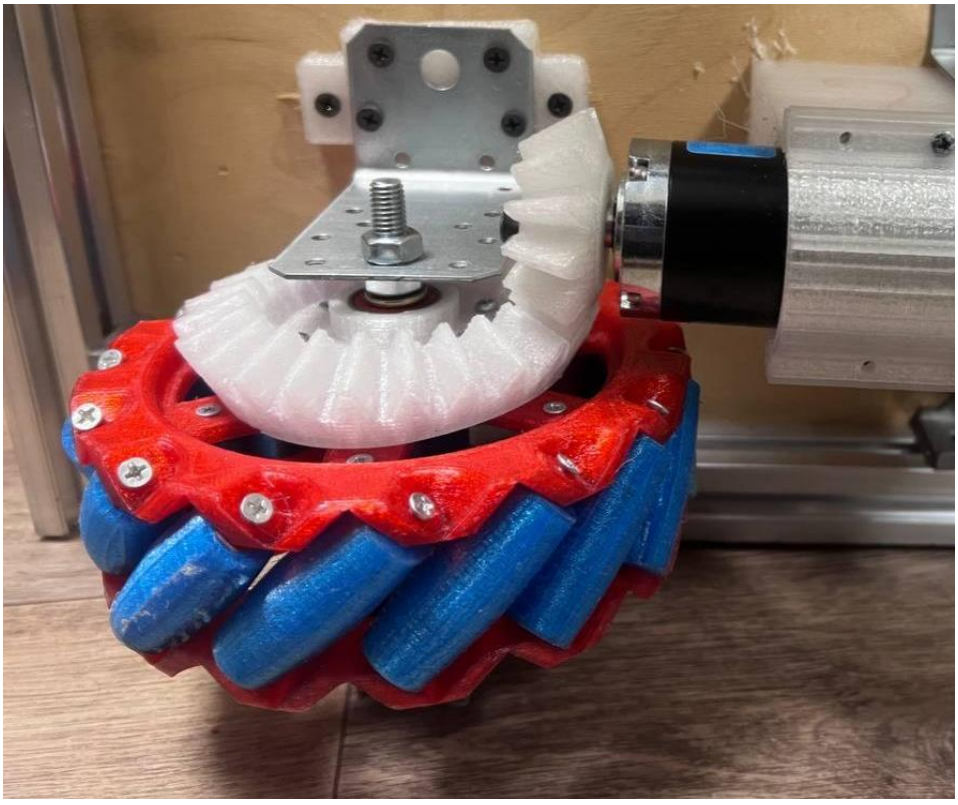


Рисунок 16. – зубчатая передача

5. Управление и электронные компоненты.

5.1 микрокомпьютер

Для управления движением платформы была выбрана плата Arduino Mega, которая имеет 54 цифровых и 16 аналоговых каналов, что позволяет легко подключать все элементы системы управления. Для платы необходим источник тока с напряжением 5В.

Рисунок 17. – плата Arduino Mega

5.2 Источники питания

Для питания платы была использована батарейка маркировки 16850 на 9В; для питания двигателей был выбран аккумулятор AGM на 12В.

Рисунок 18. – батарейка.

Рисунок 19. – аккумулятор

5.3 PWM-контроллеры

Для регулирования скорости каждого мотора были использованы 4 ШИМ-контроллера (Широтно-импульсная модуляция) на 12В, которые изменяют длительность состояний (включен/выключен) в зависимости от входящего напряжения. Благодаря этим контроллерам стало возможным изменять и контролировать скорость вращения каждого двигателя.

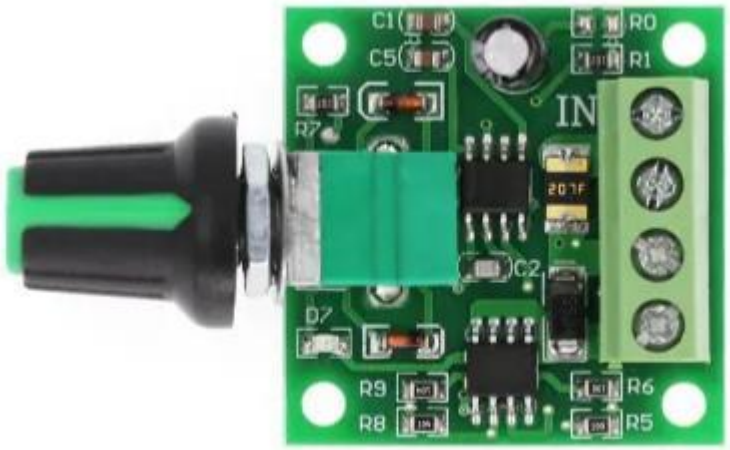


Рисунок 20. – ШИМ-контроллер.

5.4 Bluetooth модуль

Для подключения к плате был использован Bluetooth модуль HC-06, который выступает в качестве более надежного аналога радиосвязи.

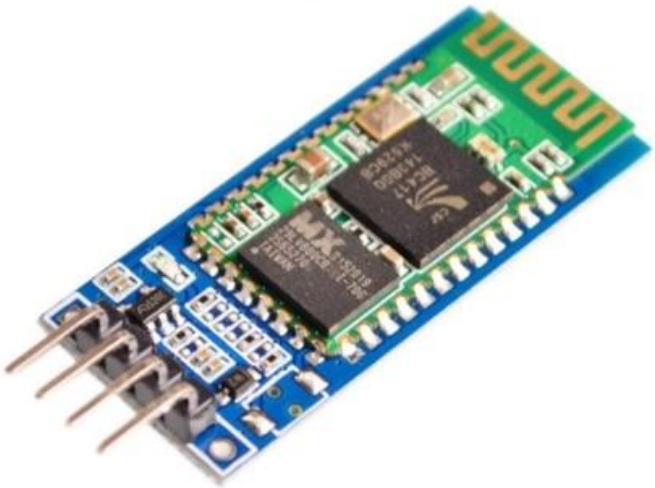


Рисунок 21. – Bluetooth модуль

5.5 Принципиальная схема

Принципиальная схема управления выглядит следующим образом. Плата Arduino Mega передает сигналы на ШИМ-контроллеры, которые в свою очередь определяют направление и скорость вращения каждого колеса в зависимости от входящего сигнала, после чего приводят двигатели в движение.

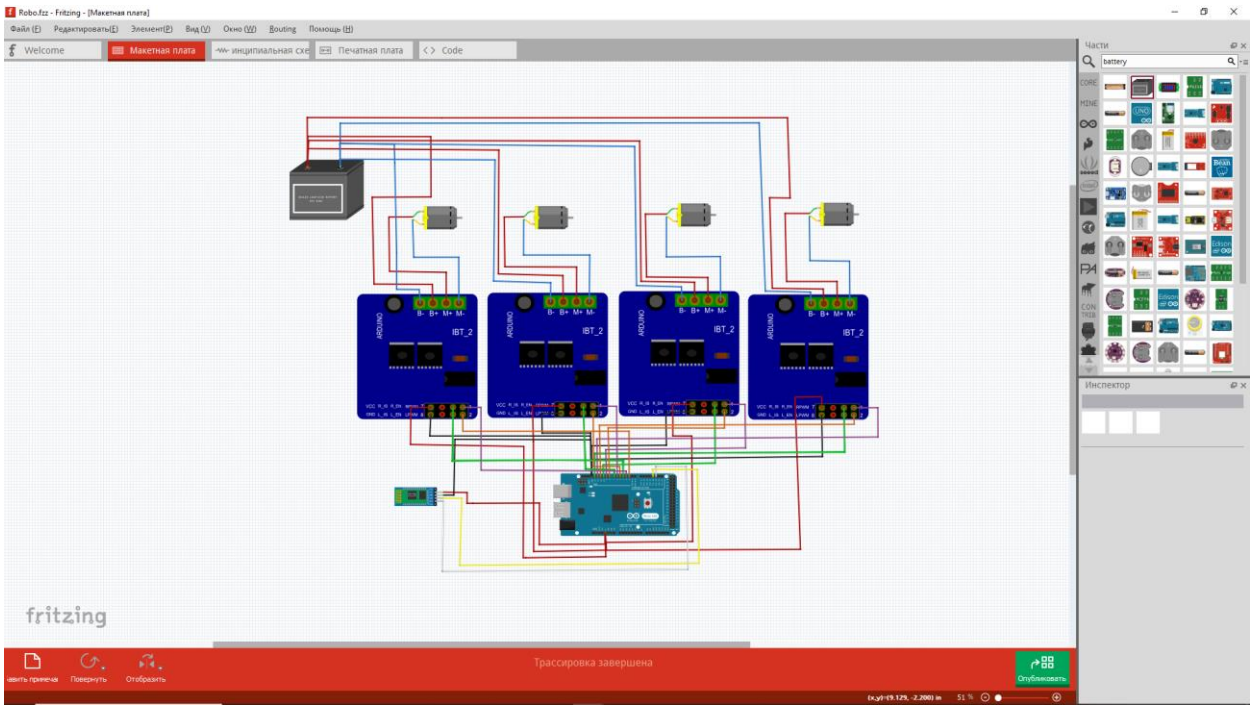


Рисунок 22. – Принципиальная схема

6.1Программирование

6.1.1 объявление переменных и присваивание портов

```
///// Мотор Front Right ////////////
const uint8_t EN1    = 4;        // № вывода Arduino к которому подключены входы драйвера L_EN и R_EN.
const uint8_t L_PWM1 = 2;        // № вывода Arduino к которому подключён  вход  драйвера L_PWM.
const uint8_t R_PWM1 = 3;        // № вывода Arduino к которому подключён  вход  драйвера R_PWM.
///// Мотор Rear Right ///////////////
const uint8_t EN2    = 7;        // № вывода Arduino к которому подключены входы драйвера L_EN и R_EN.
const uint8_t L_PWM2 = 5;        // № вывода Arduino к которому подключён  вход  драйвера L_PWM.
const uint8_t R_PWM2 = 6;        // № вывода Arduino к которому подключён  вход  драйвера R_PWM.
//// Мотор Rear Left //////////////
const uint8_t EN3    = 10;        // № вывода Arduino к которому подключены входы драйвера L_EN и R_EN.
const uint8_t L_PWM3 = 8;        // № вывода Arduino к которому подключён  вход  драйвера L_PWM.
const uint8_t R_PWM3 = 9;        // № вывода Arduino к которому подключён  вход  драйвера R_PWM.
//// Мотор Front Left/////////////
const uint8_t EN4    = 13;        // № вывода Arduino к которому подключены входы драйвера L_EN и R_EN.
const uint8_t L_PWM4 = 11;        // № вывода Arduino к которому подключён  вход  драйвера L_PWM.
const uint8_t R_PWM4 = 12;        // № вывода Arduino к которому подключён  вход  драйвера R_PWM.
```

6.1.2 Движение вперед/назад

Для движения вперед или назад необходимо вращать все 4 колеса в одном направлении.

```
////////////////////// Движение назад ////////////////////////////////////////////////////////////
void back () {
  /////////// Мотор Front Right//////////////
    digitalWrite(L_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN1,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

    digitalWrite(L_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN2,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

  /////////// Мотор Rear Left//////////////
    digitalWrite(L_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN3,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

  ////////// Мотор Front Left//////////////
    digitalWrite(L_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN4,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

}

////////////////// Движение вперед ////////////////////////////////////////////////////
void forward (){
 /////////// Мотор Front Right//////////////
    digitalWrite(L_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN1,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

////////////// Мотор Rear Right ///////////////
    digitalWrite(L_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN2,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

/////////// Мотор Rear Left//////////////
    digitalWrite(L_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN3,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

////////// Мотор Front Left//////////////
    digitalWrite(L_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN4,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);


}
```

6.1.3 движение по диагонали

Движение по диагонали осуществляется при одновременном вращении колес, находящихся в противоположных углах платформы

```
////////////////////////////// Движение по диагонали назад вправо/////////////////////////
void backright (){
 ////////// Мотор Front Left//////////////
    digitalWrite(L_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN4,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

    ////////////// Мотор Rear Right //////////////
    digitalWrite(L_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN2,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);


}
////////////////////////// Движение по диагонали назад влево//////////////////////////////
void backleft () {
   ////////// Мотор Front Right//////////////
    digitalWrite(L_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
    digitalWrite(R_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN1,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

 ////////// Мотор Rear Left//////////////
    digitalWrite(L_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM-
    digitalWrite(R_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
    analogWrite (EN3,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

}

////////////////// Движение по диагонали вперед влево ////////////////////
void forwardleft (){
////////// Мотор Front Left//////////////
    digitalWrite(L_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN4,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

////////////// Мотор Rear Right //////////////
    digitalWrite(L_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN2,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

}
////////////////// Движение по диагонали вперед вправо ////////////////////////
void forwardright(){
   ////////// Мотор Front Right//////////////
    digitalWrite(L_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN1,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

 ////////// Мотор Rear Left//////////////
    digitalWrite(L_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
    digitalWrite(R_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
    digitalWrite(EN3,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

}
```

6.1.4 движение вправо/влево

Для движения платформы вправо необходимо вращать колеса с правой стороны друг на друга, а с левой – друг против друга. Движение влево осуществляется аналогично.

```
////////////////// Движение вправо //////////
void right(){
  ////////// Мотор Front Left//////////////
   digitalWrite(L_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN4,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

   ////////// Мотор Rear Left//////////////
   digitalWrite(L_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN3,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

   ////////// Мотор Front Right//////////////
   digitalWrite(L_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN1,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

   ////////// Мотор Rear Right //////////////
   digitalWrite(L_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN2,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

}

////////////////////// Движение влево ////////////////
void left(){
  ////////// Мотор Front Left//////////////
   digitalWrite(L_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN4,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

   ////////// Мотор Rear Left//////////////
   digitalWrite(L_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN3,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

   ////////// Мотор Front Right//////////////
   digitalWrite(L_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN1,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

   ////////// Мотор Rear Right //////////////
   digitalWrite(L_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN2,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

}
```

6.1.5 Вращение

Чтобы вращать платформу необходимо направить колеса с одной стороны вперед, а с другой – назад.

```
///////////////// Вращение вправо /////////////
void superright(){
  ////////// Мотор Rear Left/////////////
   digitalWrite(L_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN3,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

 ////////// Мотор Front Left/////////////
   digitalWrite(L_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN4,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

   ////////// Мотор Front Right/////////////
   digitalWrite(L_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN1,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

////////////// Мотор Rear Right //////////////
   digitalWrite(L_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN2,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);


//////////////////////// Вращение влево ////////////////////////////////////
void superleft(){
 ////////// Мотор Rear Left/////////////
   digitalWrite(L_PWM3, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM3, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN3,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

////////// Мотор Front Left/////////////
   digitalWrite(L_PWM4, HIGH);  // Устанавливаем логическую 1 на входе драйвера L_PWM
   digitalWrite(R_PWM4, LOW );  // Устанавливаем логический 0 на входе драйвера R_PWM
   digitalWrite(EN4,    HIGH);  // Эта функция выполнит те же действия что и функция analogWrite(EN, 255);

   ////////// Мотор Front Right/////////////
   digitalWrite(L_PWM1, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM1, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN1,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN
 ////////////// Мотор Rear Right //////////////
   digitalWrite(L_PWM2, LOW );  // Устанавливаем логический 0 на входе драйвера L_PWM
   digitalWrite(R_PWM2, HIGH);  // Устанавливаем логическую 1 на входе драйвера R_PWM
   analogWrite (EN2,    255 );  // Устанавливаем 100% ШИМ на входах драйвера L_EN и R_EN

}
```

6.2Приложение

Мобильное приложение было написано с помощью программы MIT APP INVENTOR, которая позволяет легко расположить элементы интерфейса. Программирование осуществляется с помощью готовых блоков.

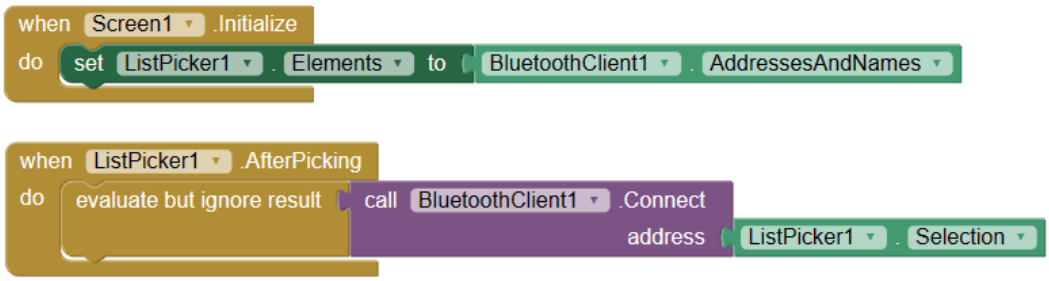
6.2.1 Интерфейс

При запуске приложения на экране появляется кнопки, определяющие направление движения, кнопка подключения к Bluetooth модулю и ползунок, позволяющий изменять скорость движения платформы.

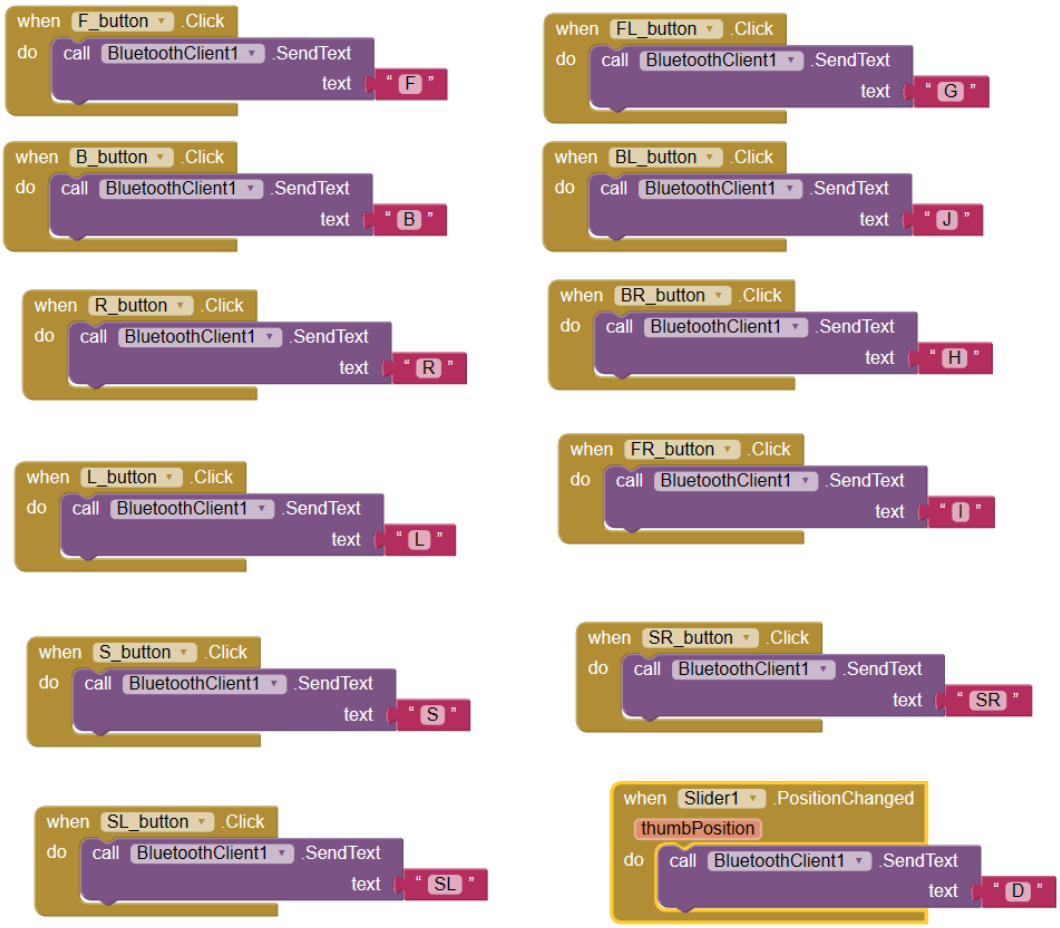
Рисунок 23. – интерфейс

6.2.2 Программирование кнопок

Основной При нажатии на кнопку “Bluetooth” появляется список с устройствами, доступными для подключения.



При нажатии на кнопку Bluetooth модулю отправляется текстовый сигнал, который плата Arduino воспринимает, как команду.



После того, как плата получает сигнал, начинается выполнение команды.

```
void loop() {
  if (Serial1.available()) {
    command = Serial1.read();
    if (command == 'X') {
      state = 1;
    } else if (command == 'x') {
      state = 0;
    }
    if (command == 'B') {
      back();
    } else if (command == 'F') {
      forward();
    } else if (command == 'SR' ) {
      superright ();
    } else if (command == 'SL') {
      superleft ();
    } else if (command == 'R' ) {
      right ();
    } else if (command == 'L') {
      left ();
    } else if (command == 'G') {
      forwardleft ();
    } else if (command == 'I') {
      forwardright ();
    } else if (command == 'H') {
      backright ();
    } else if (command == 'J') {
      backleft ();
    }  else if (command == 'S'){
      stopRobot();
    }
  }
}
```

7 Готовая платформа

Сзади на платформе установлены три контейнера, в первый из которых помещены все электронные устройства, во второй – аккумулятор, а в третий оставлен пустым для запасных батареек или деталей. Собранная платформа имеет грузоподъемность около 80 кг и может развивать скорость до 5 м/с.

Рисунки 24, 25 – готовая платформа

8 Перспективы в работе над проектом

Сейчас идет работа над внедрением машинного зрения, что позволит платформе автоматически определять препятствия на своём пути, ехать по заданной траектории, распознавать нужные предметы и лица. Это сделает работу тележки полностью автономной. Для этой задачи был выбран микрокомпьютер Orange Pi 3 LTS, который обладает достаточной производительностью для выполнения всех вышеперечисленных функций.

Заключение

По итогам выполненной работы удалось создать рабочую платформу, на колёсах Илона, управляемую с телефона. Она может выдерживать до 80кг нагрузки и перемещаться со скоростью около 5 м/с. Платформа занимает малое пространство и способна помещаться в дверные проемы. Технология машинного зрения, над которой ведется работа уже позволяет выводить прямую видеотрансляцию на сервер, делать и сохранять фотографии в сетевую папку.

Список используемых источников и литературы

1) Li Shi gang motion control based on MECANUM wheel mobile robot Tshinghua University 2018.

2) B. Barshan and H. F. Durrant-Whyte. Inertial navigation systems for mobile robots. IEEE Transactions on Robotics and Automation.

3) Трехмерная оценка положения мобильных роботов на основе более гладкого подхода. В материалах Международной конференции IEEE 1999 г. по робототехнике и автоматизации, май 1999 г.