RoboCup Junior 2023

КОМАНДА:
МТКП МГТУ им. Н.Э. Баумана
ДИСЦИПЛИНА:
RoboCupJunior Rescue Line

RoboCup Russia Open - RoboCupJunior 2023 Team Description Paper

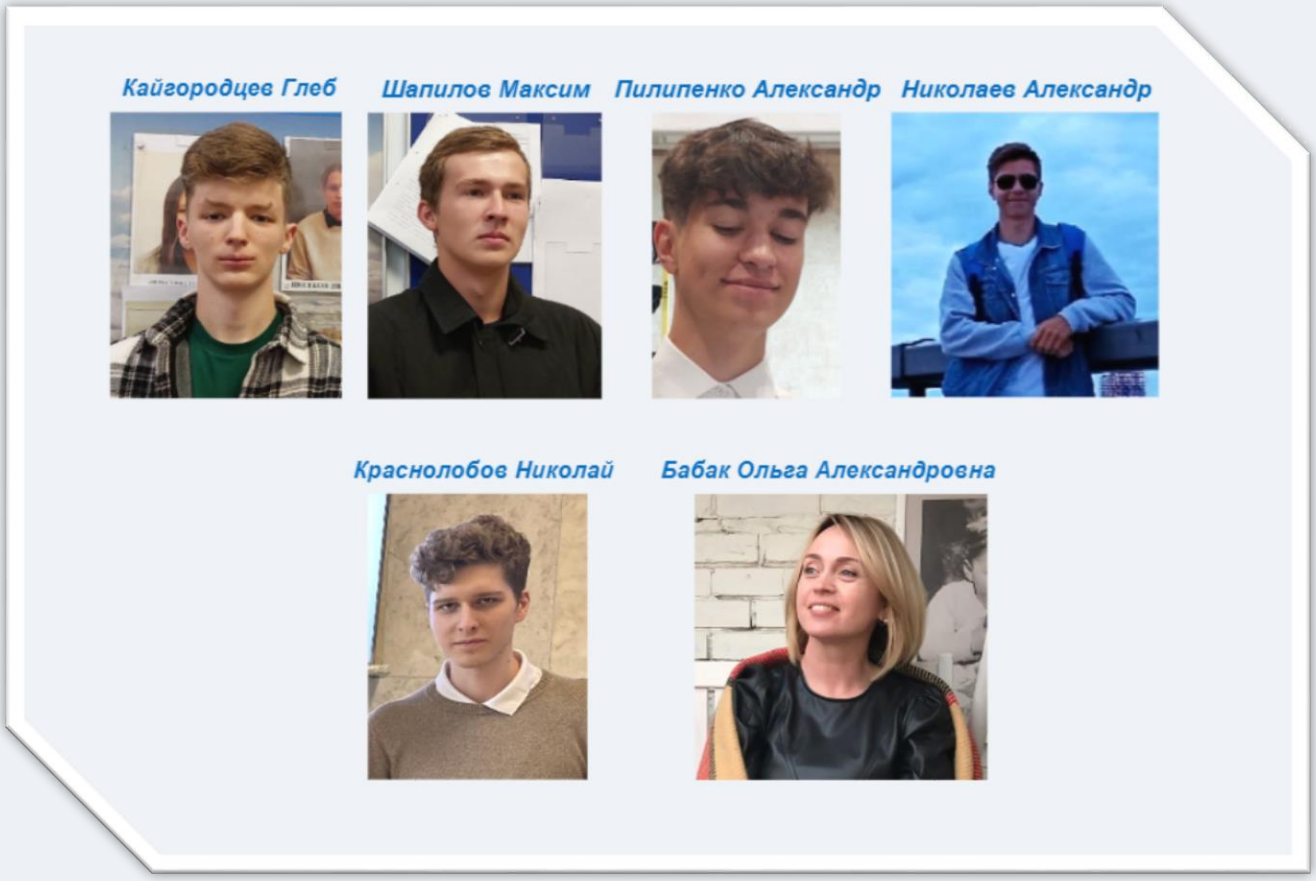
RoboCupJunior League: Rescue Line
Team Name: "Tilt"

Participants Name:

- *Кайгородцев Глеб, glekay@yandex.ru*
- *Шапилов Максим, shapilov.maksim@mail.ru*
- *Николаев Александр, niklexgg@yandex.ru*
- *Пилипенко Александр, 2004_sasha96@mail.ru*
- *Краснолобов Николай, krasnolobovnikolay2004@gmail.com*

Mentor: Бабак Ольга Александровна, babak@bmstu.ru
Institution: МТКП МГТУ им. Н.Э. Баумана

Country (страна): Россия
Date (дата): 11.05.2023



Team Information

Team Name: “Tilt”
Country/Region: РФ, Москва

The Participants Name and their Technical Role:
Member 1: Кайгородцев Глеб – Ведущий конструктор-проектировщик проекта.
Member 2: Шапилов Максим – Конструктор-проектировщик захватывающего устройства.
Member 3: Пилипенко Александр – Ведущий программист проекта, проектировщик электронной части.
Member 4: Николаев Александр – Специалист по технологии 3Д-печати, монтажник.
Member 5: Краснолобов Николай – программист проекта, проектировщик электронной части.

Technical information

Цель нашего проекта – создание макета робота-спасателя с функциями эвакуации пострадавших из аварийных зон. Такой проект с новыми идеями может позволить в будущем попробовать реализовать полномасштабный прототип робота-спасателя, который непосредственно будет оказывать помощь людям в экстренных ситуациях.

Наша команда представляет проект под аббревиатурным названием "В.А.Л.Е.Р.А". Все его преимущества и функции описаны в этой расшифровке:

- *В – вездеходный, это обеспечивается за счет устойчивой колесной базы и повышенного сцепления на колесах робота.*
- *А – автоматический, "В.А.Л.Е.Р.А" является полностью автономной машиной для выполнения задач, непосильных человеку.*
- *Л – легко маневрирующий, в силу своих сервомоторов с крутящим моментом в 10 кг*см робот легко проезжает любые препятствия и сложные участки трассы.*
- *Е – ёмкий, в силу своих небольших размеров, "В.А.Л.Е.Р.А" имеет очень насыщенную начинку внутри (читать подробнее).*
- *Р – робот*
- *А – аптекарь, из-за того, что робот может доставлять медицинский пакет с помощью захватывающего устройства.*

Главными задачами проекта являются:

1. *Выбор элементной базы под проектируемые задачи*
2. *Схемы подключения электроники проекта*
    - *Преобразователи напряжения DC-DC, их функции и назначение*
3. *Спроектировать 3Д-модель основания робота*
4. *Разработать 3Д-модель захватывающего устройства (ковша)*
5. *Обоснованно выбрать вид пластика для печати*
6. *Подготовить 3Д-модели к печати на 3Д-принтере*
7. *Сборка всех частей робота*
    - *Сборка корпуса*
    - *Подключение электроники и датчиков*
8. *Этап программирования робота*
9. *Отладка программы*
10. *Исправление выявленных ошибок*

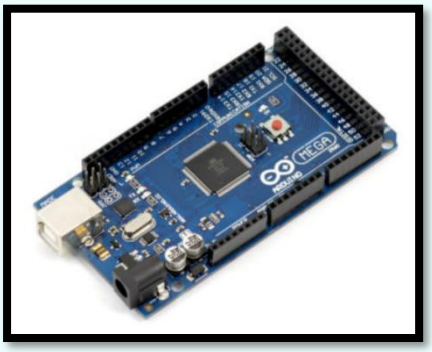
График работы над проектом:

Дата	Реализуемая задача
28.03.2023	Появление идеи участия в соревнованиях по робототехнике
29.03.2023 – 05.04.2023	Ознакомление с регламентом RoboCupJunior и окончательный выбор лиги Rescue Line League
06.04.2023 – 09.04.2023	Создание набросков и идей реализации проекта
10.04.2023 – 12.04.2023	Распределение ролей и определение обязанностей
11.04.2023 – 15.04.2023	Выбор элементной базы
14.04.2023 – 26.04.2023	Печать деталей конструкции корпуса
13.04.2023 – 16.04.2023	Создание пластин и ходовой части робота
13.04.2023 – 19.04.2023	Первичная сборка робота (сборка корпуса и монтаж элементной базы) и начальная отладка кода
20.04.2023 – 24.04.2023	Разработка и создание захватывающего устройства
24.04.2023 – 25.04.2023	Полная сборка и отладка робота
24.04.2023 – 29.04.2023	Подготовка и участие в Московском этапе.
30.04.2023-04.05.2023	Устранение недостатков робота в корпусе и электронной части.
05.05.2023 – 11.05.2023	Программирование и отладка программы.

Ход выполнения задач проекта

1. Выбор элементной базы под проектируемые задачи

Основой нашего проекта стала платформа ArduinoMega 2560 на микроконтроллере ATmega2560. Выбрана она в силу достаточно большого количества своих входов/выходов, необходимых для подключения выводов всех датчиков.



Плата Arduino	Микроконтроллер	Рабочее напряжение (В)	Цифровые входы/выходы	Выходы с ШИМ	Аналоговые входы/выходы
Uno	Atmega328	5	14	6	6
Leonardo	ATmega32u4	5	20	7	12
Nano	ATmega328	5	14	6	8
Mega	ATmega2560	5	54	14	16

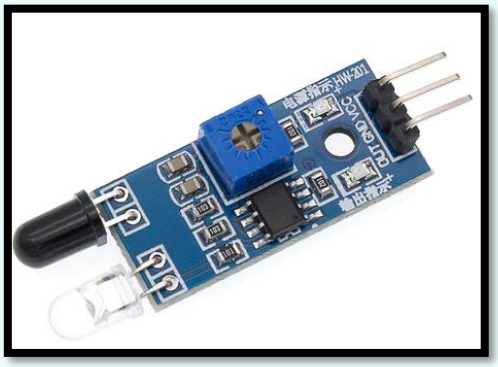
Таблица сравнения входов/выходов Arduino

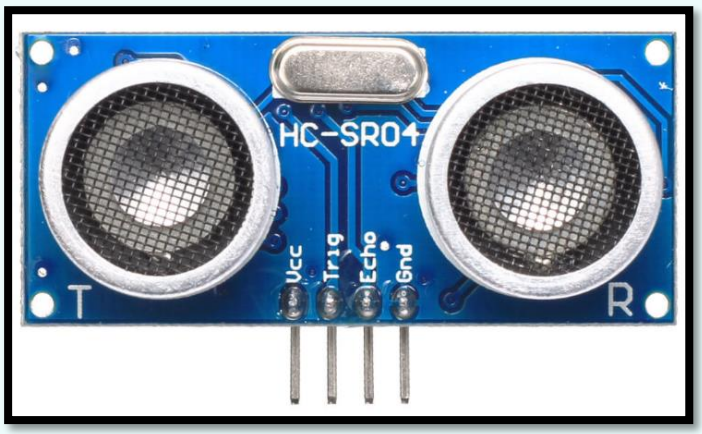
В качестве моторов были выбраны сервоприводы MG995 с крутящим моментом в 10 кг*см с шестеренками из металла

Выбор датчиков основывался на основных задачах, которые должен выполнять робот для участия на соревнованиях RoboCupRussia Open 2023:

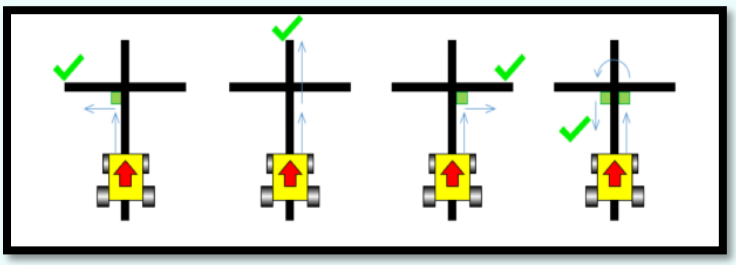
- Езда по линии
- Объезд препятствий
- Определение направления пути на перекрестках
- Эвакуация пострадавших из зоны эвакуации

Для реализации езды по линии были выбраны ИК-датчики с регулированием чувствительности определения отражения. Принцип работы очень простой и надежный – это и есть главные достоинства данного элемента.

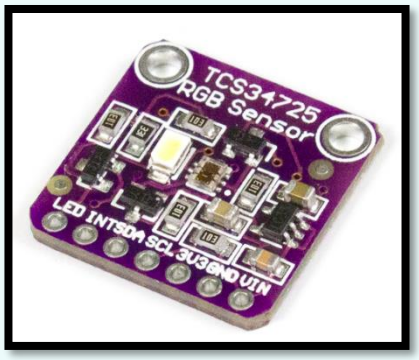




Система объезда препятствий основана на ультразвуковом датчике HC-SR04, который определяет расстояние до объектов по отраженному звуковому сигналу.



Возможные сценарии



TCS34725

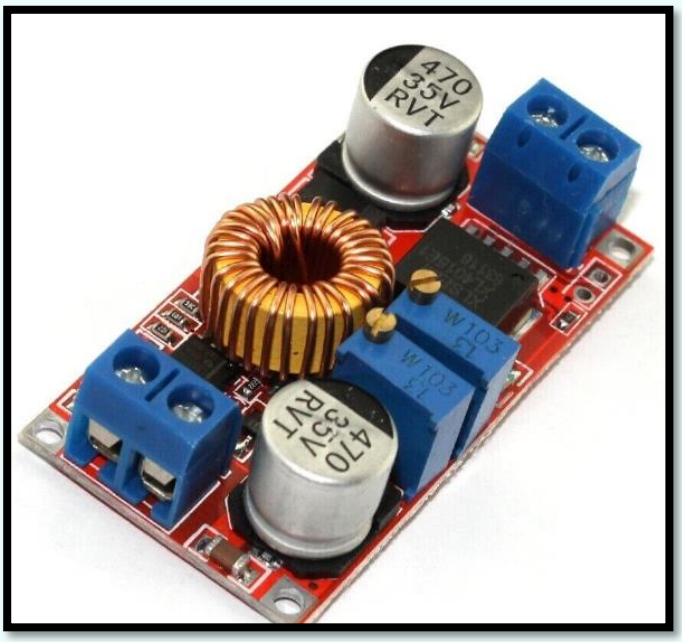
Поворот на перекрестках по регламенту Лиги осуществляется благодаря зеленым квадратам на углах линии. Для решения этой задачи были выбраны датчики цвета TCS34725

Так же для работы системы захватывающего устройства будут использоваться 2 типа сервоприводов MG90S и MG996R с металлическими шестеренками.

MG90S

MG996R

В схеме питания платы и датчиков используется два преобразователя напряжения DC-DC.



2. Спроектировать оптимальную систему питания платы ArduinoMega2560 и драйвера моторов L293D

Электрическая схема драйвера моторов L293D:

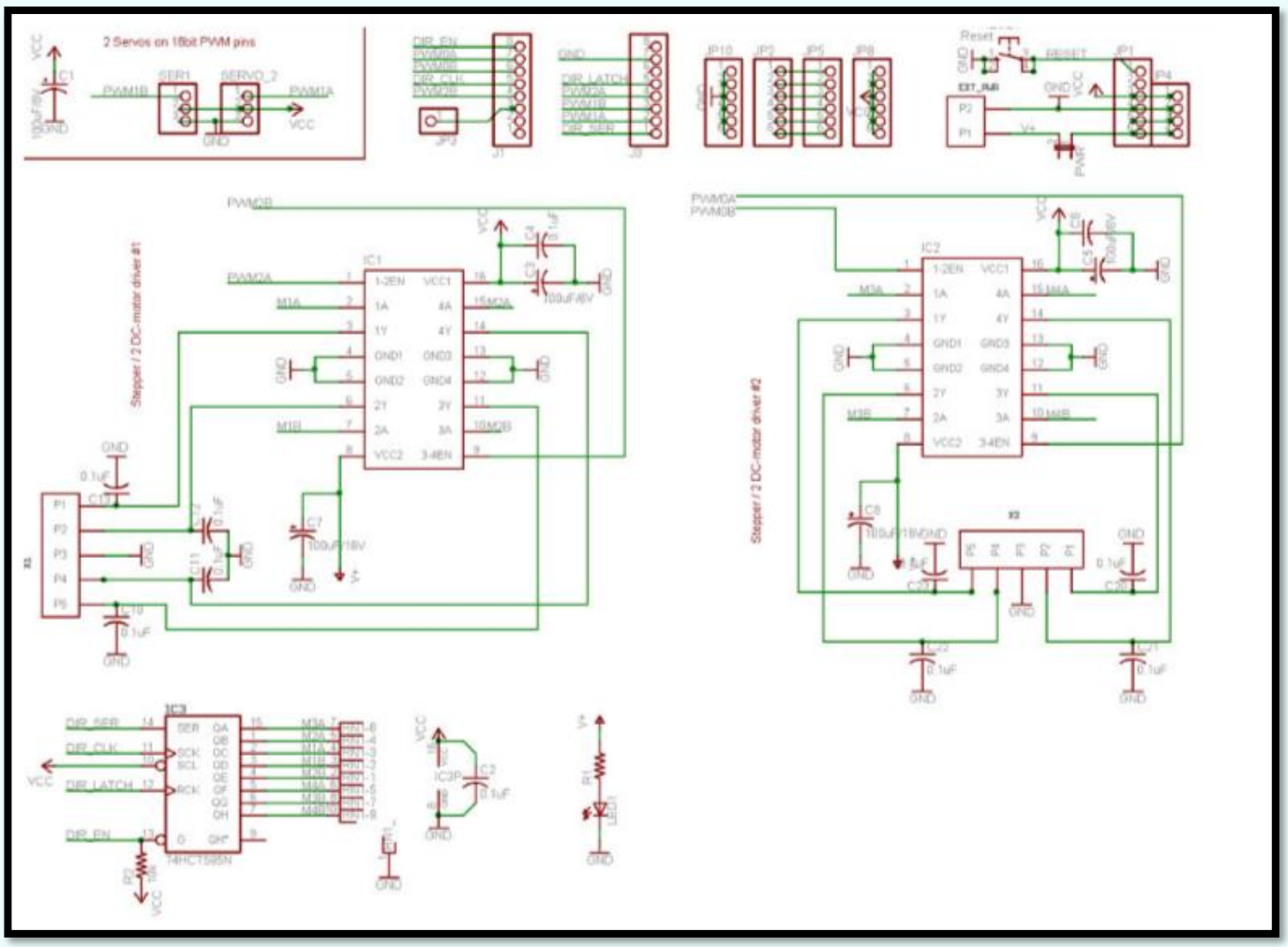
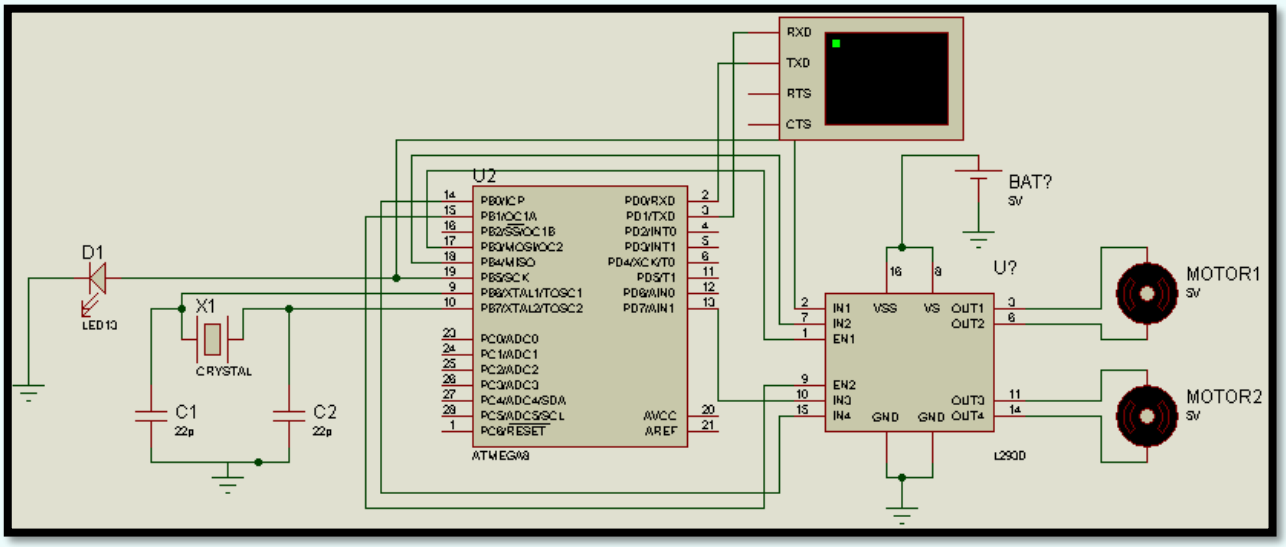


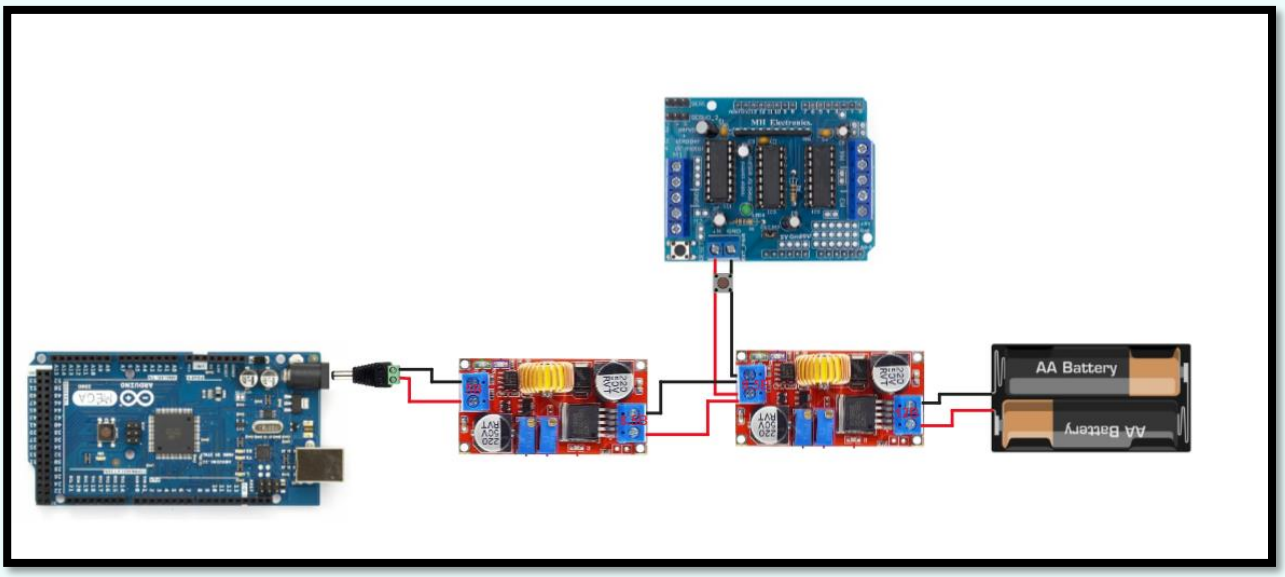
Схема взаимодействия Arduino Mega и L293D:



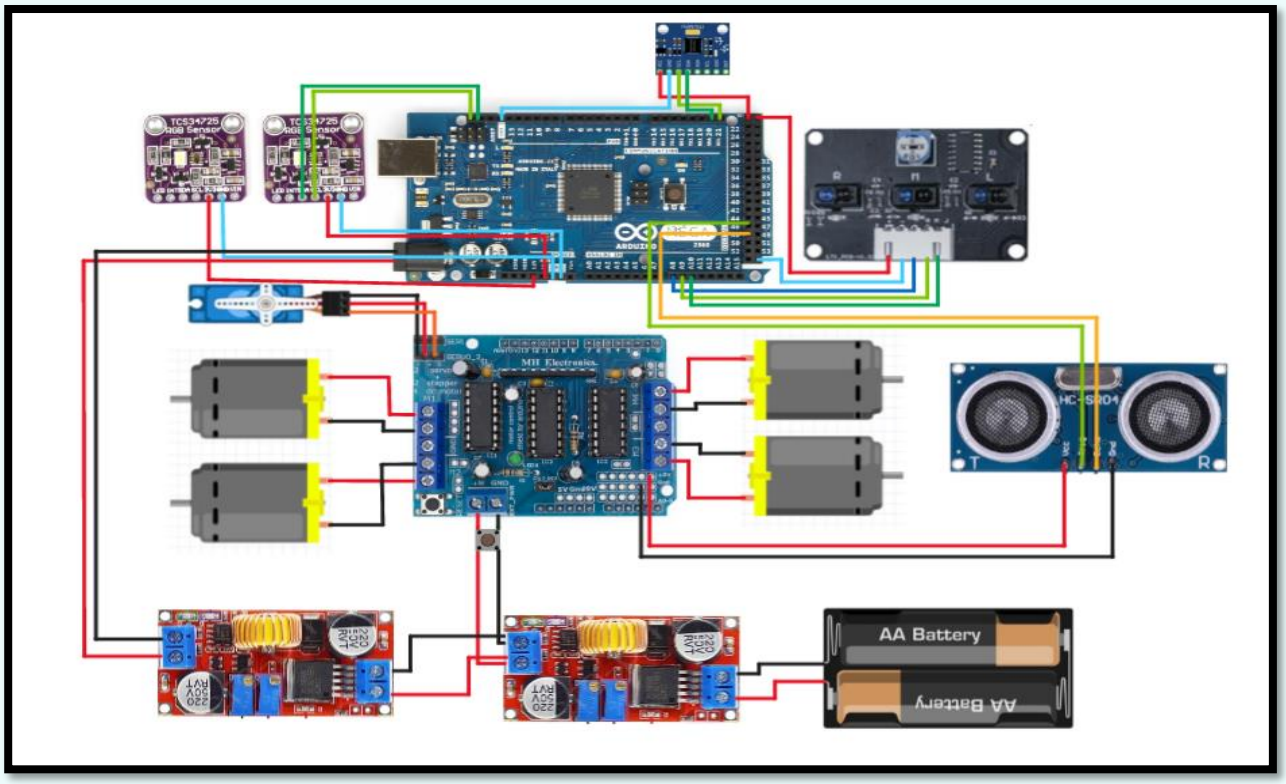
Первоначально схема питания платы ArduinoMega2560 была рассчитана на ДПТ с редуктором 1:48 и была следующей: три батарейки Li-ion 18650 по 4В и два понижающих преобразователя DC-DC:

Первый преобразователь 12В – 8.5В (На драйвер моторов)

Второй преобразователь 8.5В – 5В (На плату Arduino)



Полная схема подключения всей элементной базы:

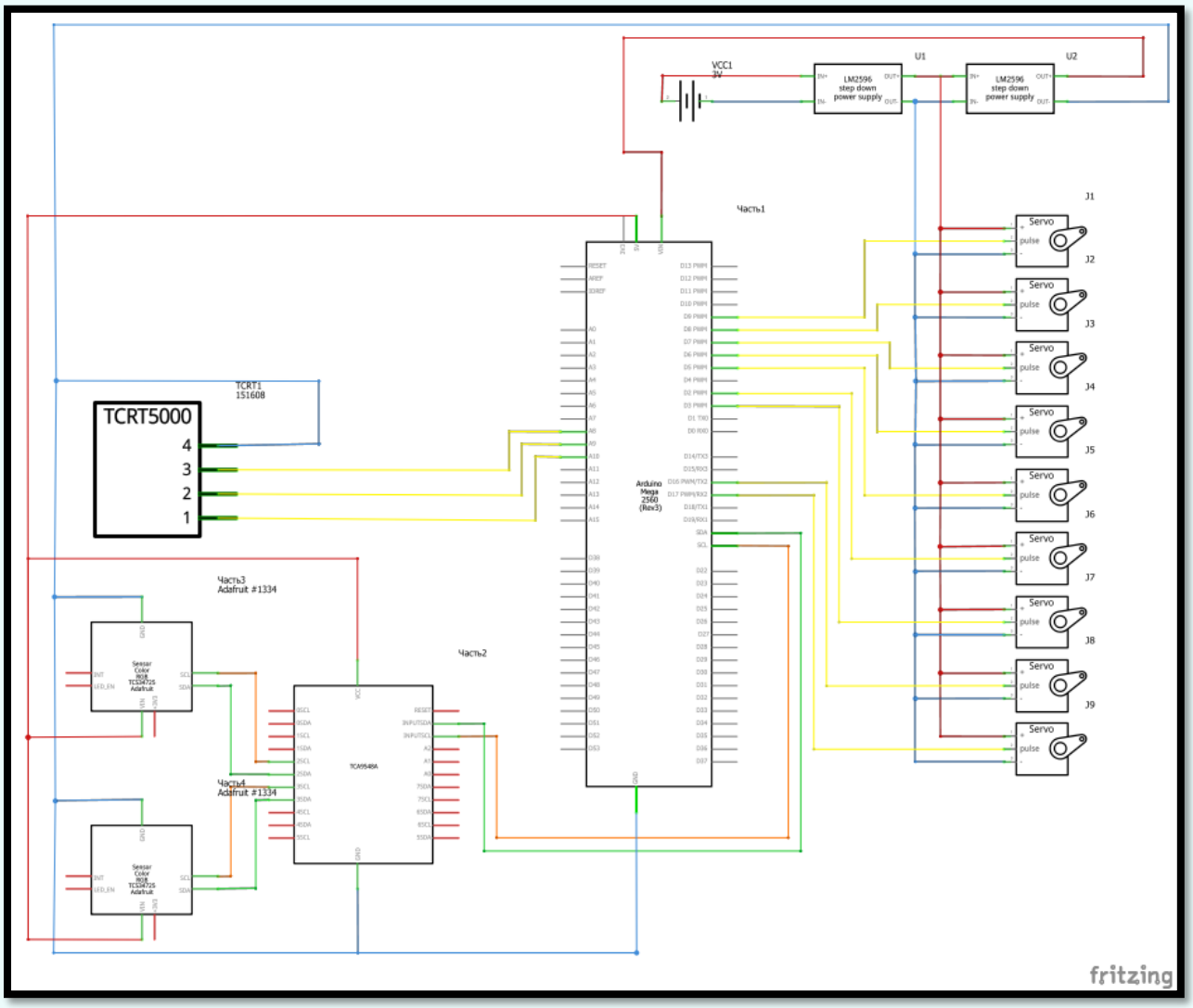


Затем при пуско-наладке была выявлена следующая проблема: Нехватка крутящего момента моторов. Его движения было затруднительно. Решением стала замена ДПТ с редуктором 1:48 на сервоприводы MG996R.

Замена на сервоприводы привела к появлению большого количества электрических соединений со всех датчиков и сервоприводов.

Для решения этой проблемы объединили в одну шину провода питания и провода GND на макетной плате.

Итоговая схема подключения с изменениями выглядит так:



3. Разработка и создание каркаса и крепления шасси автономного робота.

Все работы по проектированию ведутся в Компас-3D. В этапы разработки и изготовления каркаса и крепления шасси входит:

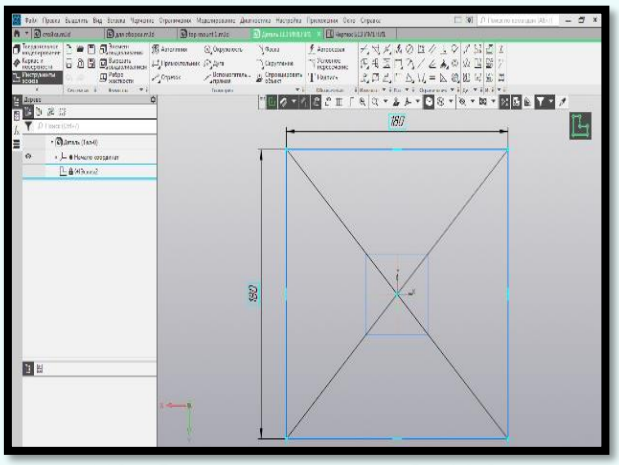
1. Проектирование каркаса
   - Выбор формы пластины
2. Проектирование монтажных точек для электрических и механических компонентов
3. Подготовка пластин к трехмерной печати
   - Разбиение пластин на части
   - Подготовка файлов к печати
4. Шлифовка, доработка деталей корпуса (при необходимости)
5. Сборка каркаса

1 Этап – Проектирование каркаса

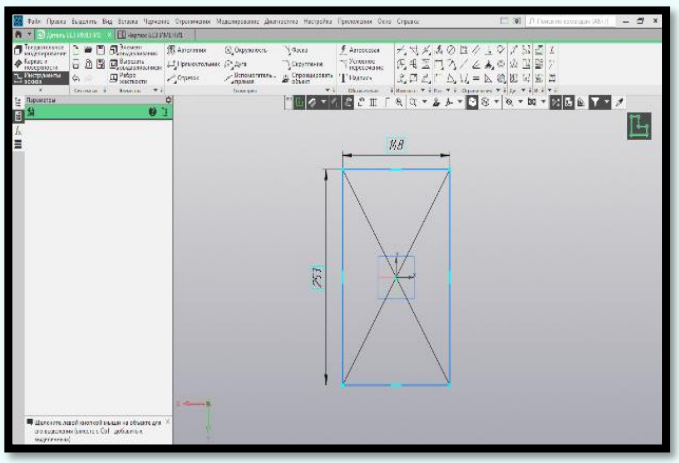
Было принято решение создать каркас робота при помощи двух параллельно соединенных пластин, расстояние между которыми позволяет:

- на нижний уровень (нижняя пластина) разместить все необходимые датчики, управляющее устройство с АКБ;
- на верхний уровень (верхняя пластина) разместить дальномер и захватывающие устройство.

Изначально было два варианта идеи пластин:



Квадрат стороной 180мм



Прямоугольник 148мм х 253мм

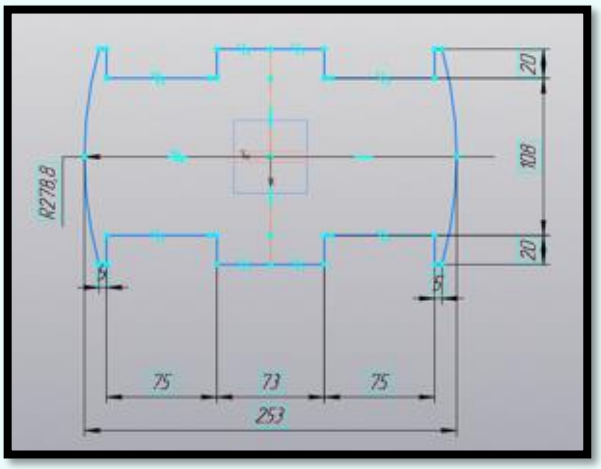
Недостаток – *в ходе работы стало понятно, что квадратная пластина не подходит для нашего проекта:*

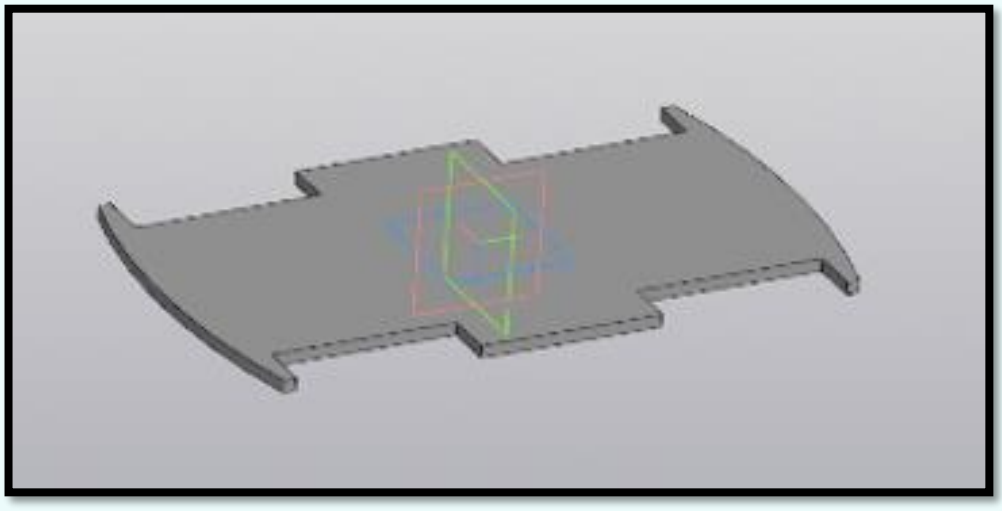
- *невозможно рационально использовать всю площадь*
- *недостаточно места для расположения захватывающего устройства (ковша)*
- *колесная база получается короткой из-за чего робот теряет устойчивость на спусках, подъемах и качелях, что ведет к его опрокидыванию.*

В итоге выбор был в пользу прямоугольной пластины:

Улучшения:

- *Сузили колесную базу за счет прямоугольных вырезов на корпусе.*
- *для уменьшения радиуса поворота и повышения манёвренности в труднодоступных местах сделали скругления коротких сторон пластины.*



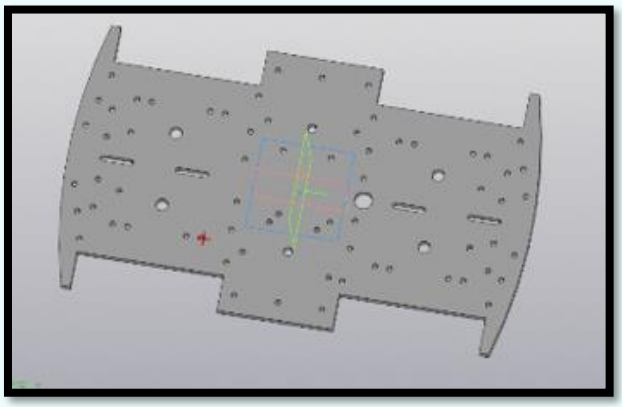


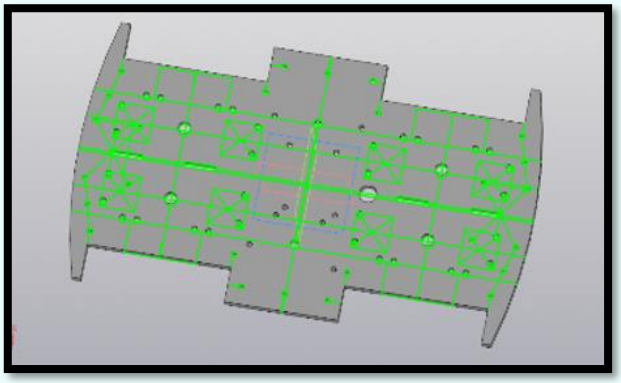
Данные изменения позволили не только рационально использовать место на пластинах, но и сделать робота более устойчивым и манёвренным.

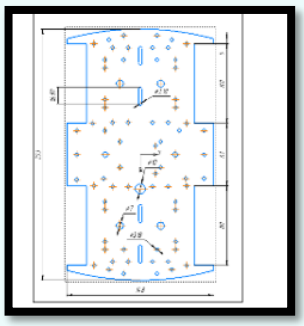
2 Этап – Проектирование монтажных точек для электрических и механических компонентов.

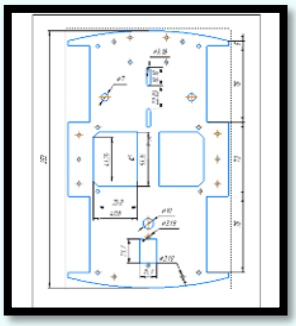
Улучшения:

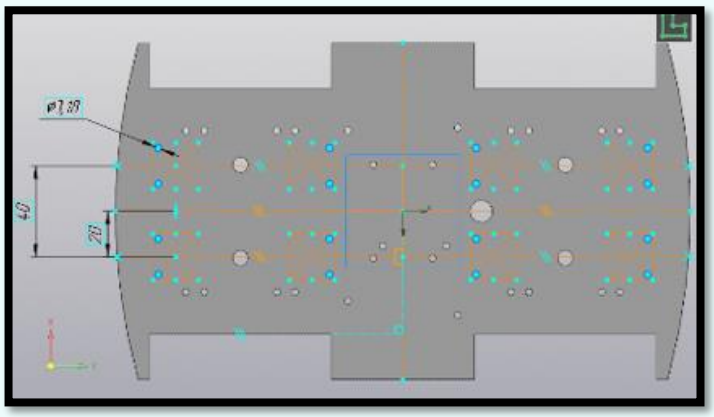
- Крепежные отверстия на верхней и нижней пластинах, d 3.175 мм Технологические отверстия для проводов на верхней и нижней пластинах, d 10 мм и 7 мм
- Отверстия овальной формы на нижней и верхней пластинах для крепления аккумуляторов



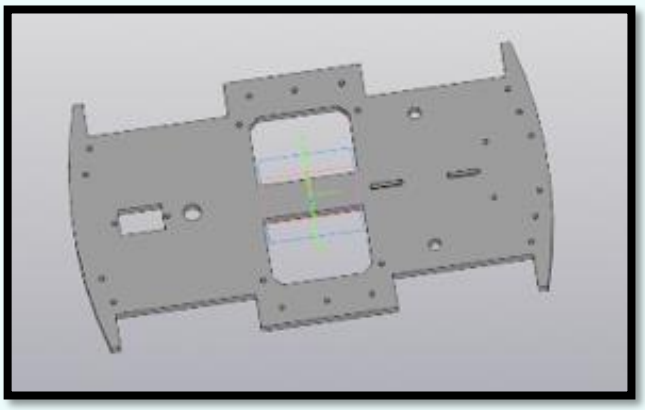


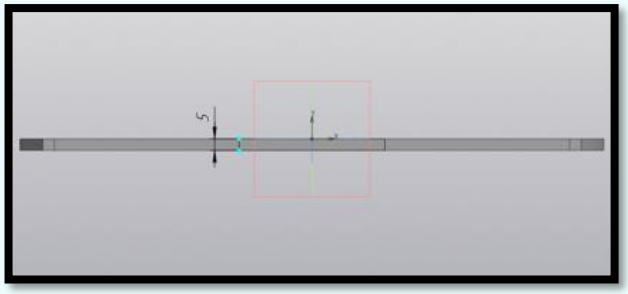






- Отверстия прямоугольной формы (большие) на верхней пластине для облегченного доступа к управляющей схеме
- Отверстие прямоугольной формы (малое) на верхней пластине для сервомотора.

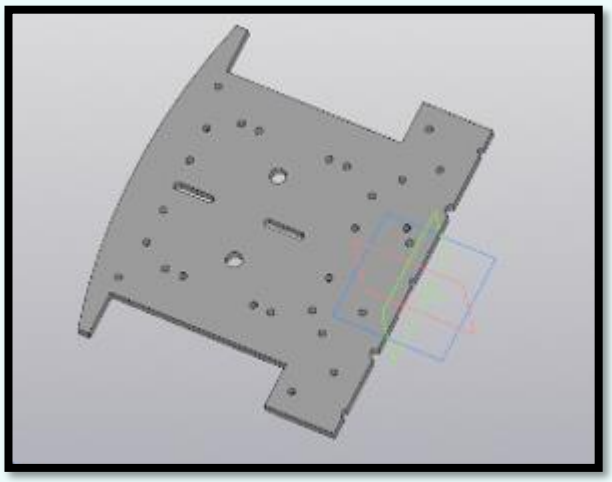


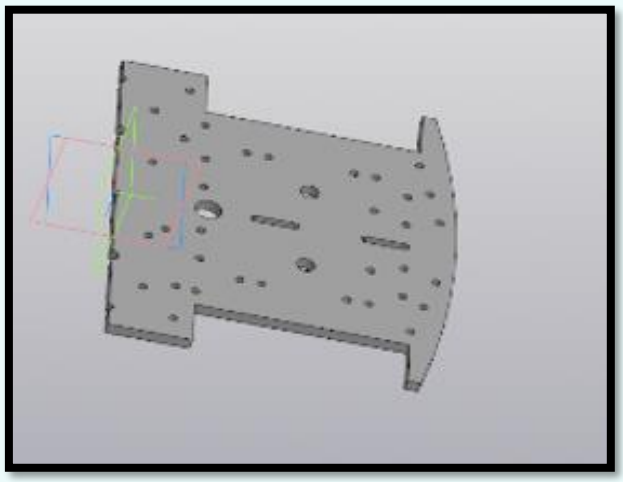


3 Этап – Подготовка пластин к трехмерной печати

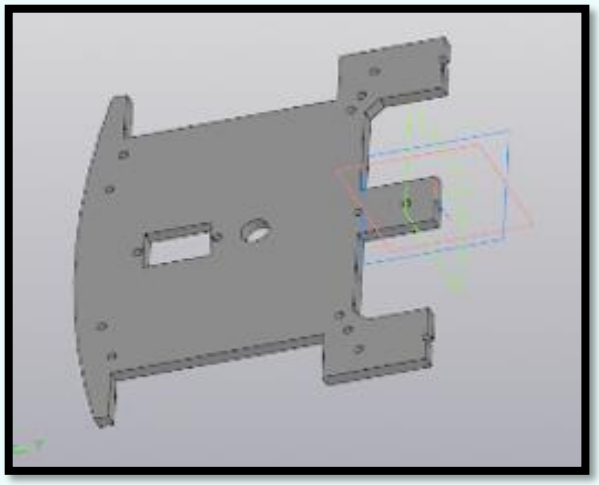
При подготовке к печати каждую пластину потребовалось разделить на две равные части, т.к. рабочее пространство 3д-принтера 200 х 200 х 200 мм не позволяло напечатать их целиком.

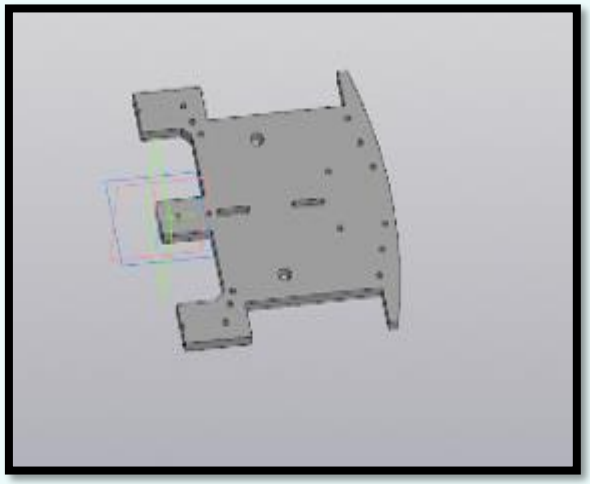
Нижняя пластина





Верхняя пластина

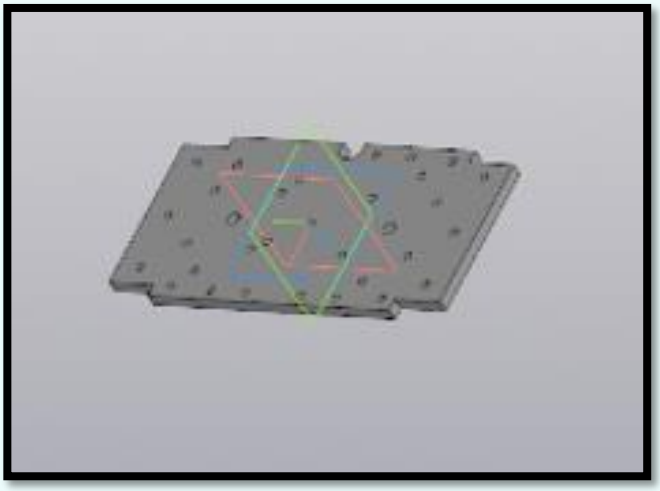


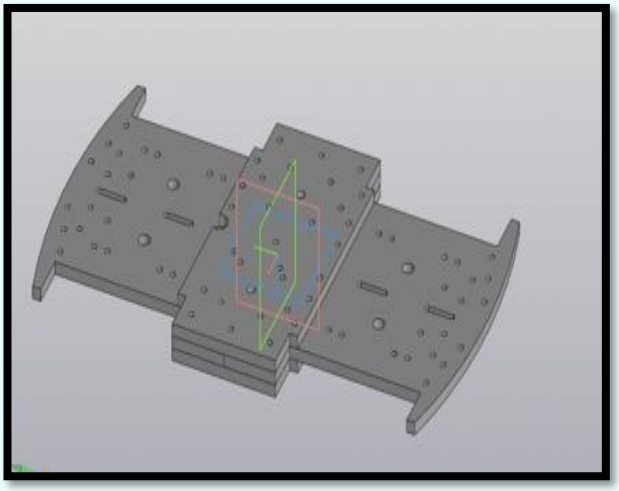


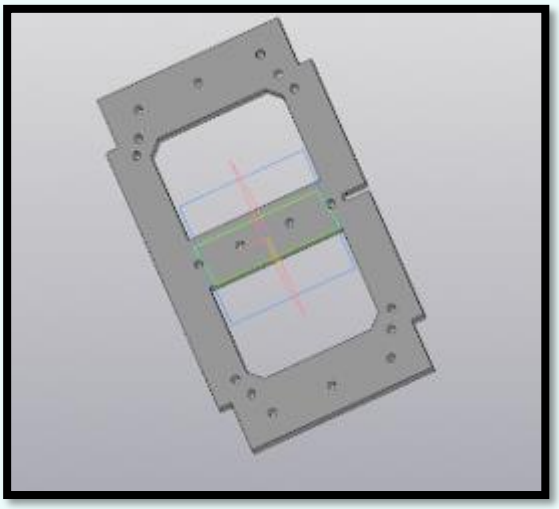
Далее появляется новая задача – разработать способ соединения напечатанных деталей пластин.

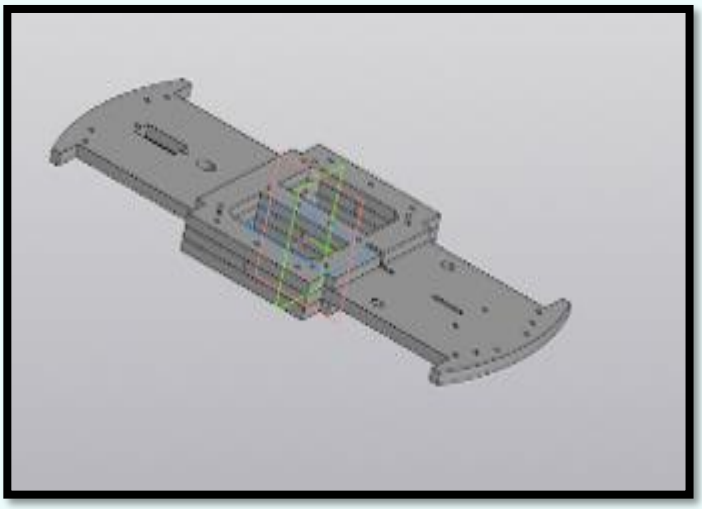
Был спроектирован способ скрепления:

Соединение частей пластин между собой по технологии «Бутерброд»





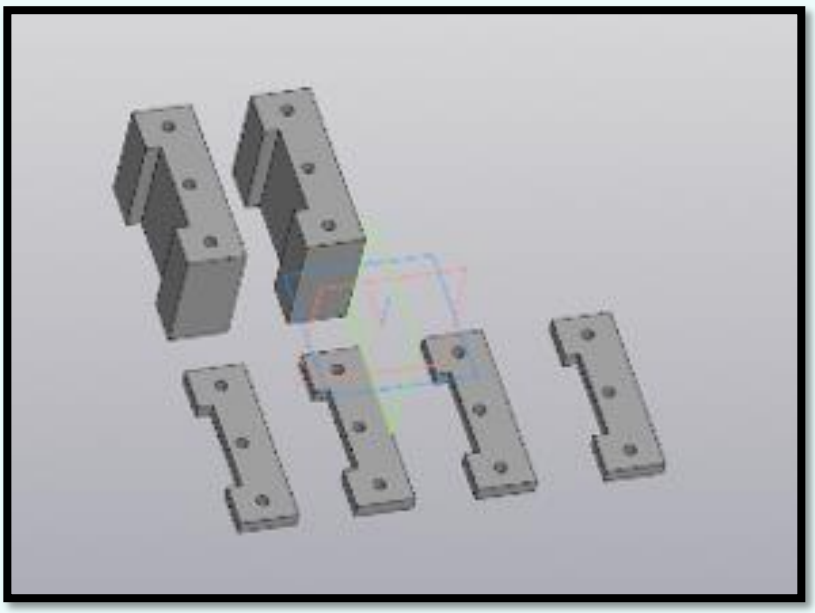




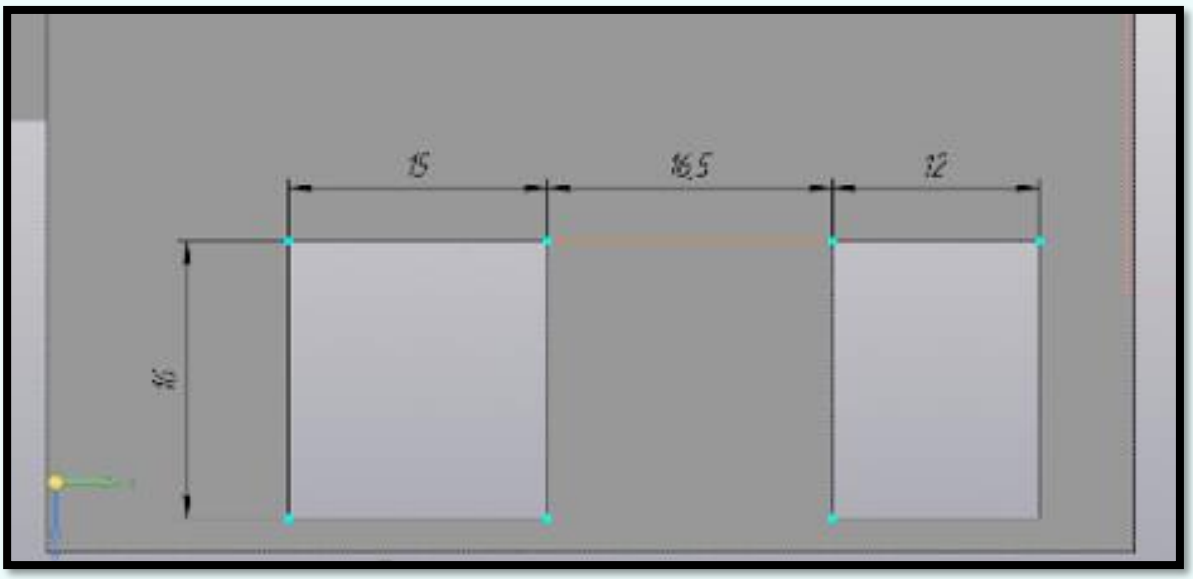
Минусом такого метода является увеличение толщины на 10 мм в местах соединения, что ведет к сложности крепления датчиков. Поэтому был разработан второй вариант скрепления пластин с помощью П-образных соединительных стоек, которые помимо надежного скрепления пластин между собой еще усиливают параллельное соединение верхнего и нижнего уровня.

В одной из стоек необходимо было спроектировать технологические отверстия для питания и заливки программы в плату ArduinoMega 2560:

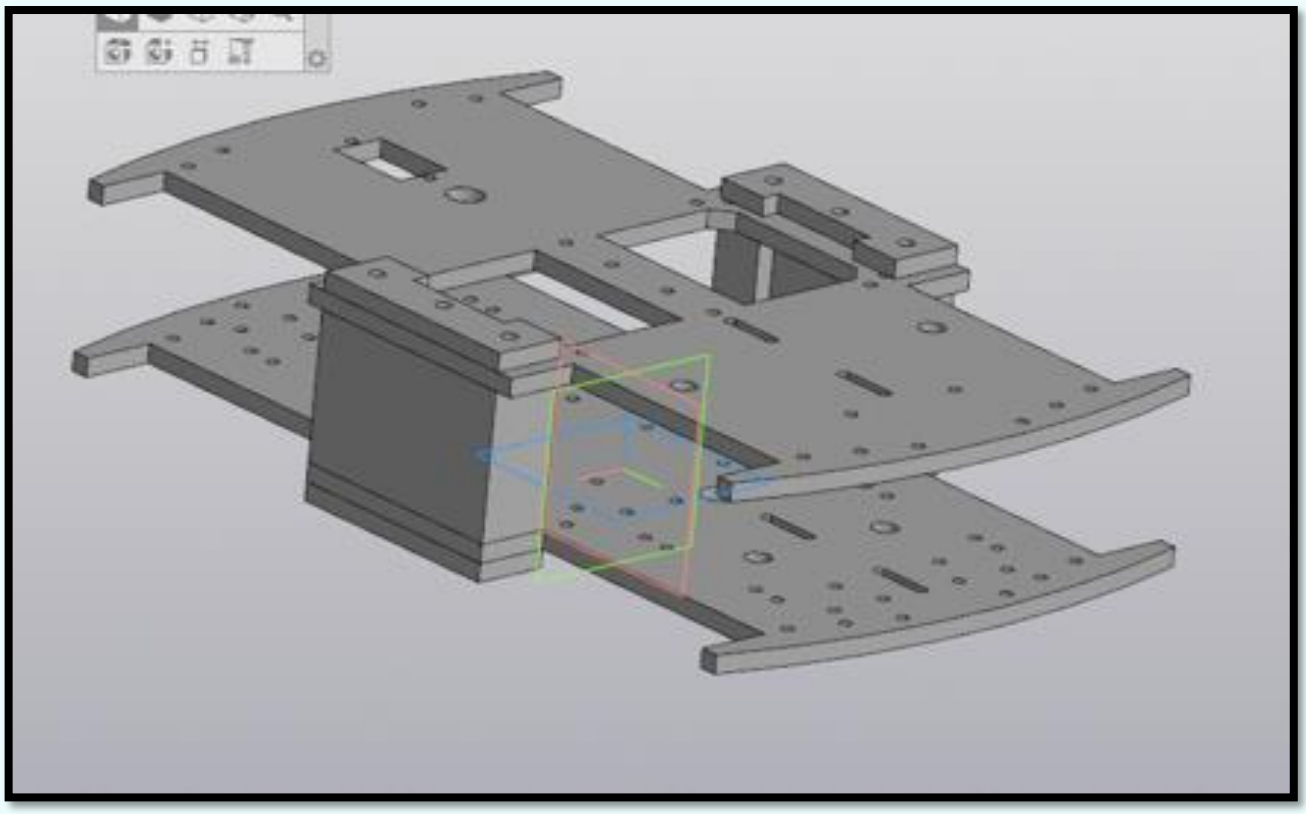
Первоначальный вариант



Вариант с отверстиями под заливку программы в плату.



Итоговый вариант крепления пластин



4. Разработка 3Д-модели захватывающего устройства

Изначально стоявшая задача заключалась в следующем:
Собирать шарики 4,5-5см в количестве 3 штук максимально простым и эффективным способом используя минимальное количество исполнительных механизмов.
Идея появилась, глядя на перевоз контейнеров для мусора.

Появилась идея взять контейнер, который будет опускаться за роботом и в него при помощи клешней собирать шарики – конструкция робота позволяет расположить небольшой контейнер на верхнюю панель робота.
Далее начали попытки разработать крепление чтобы сервомотор располагался на нижней пластине при этом возникли следующие трудности:

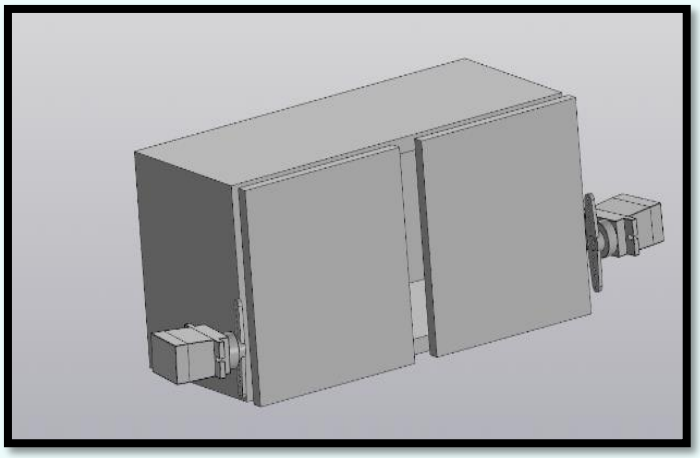
- Большие габариты сервомотора
- Сложность механизма забрасывания контейнера на “В.А.Л.Е.Р.А.”

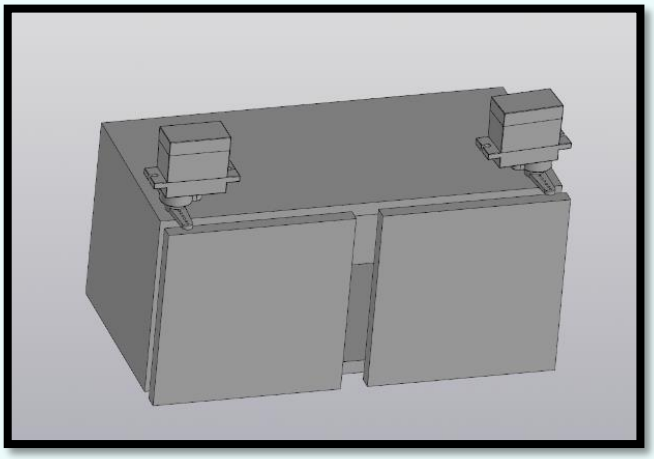
В следствии было принято решение располагать сервомотор, который забрасывает контейнер на верхнюю часть робота. Это позволило упростить механизм забрасывания.
Ещё необходим механизм для вращения контейнера относительно крепежа, чтобы была возможность высыпать шарики в безопасную зону.

Разработка контейнера

Контейнер должен быть максимально маленьких размеров для размещения в себя 3 см и так чтобы шарики из него сами не высыпались. Самое эффективное использование места — это прямоугольник. Он и был взят за основу.

Прототип 1

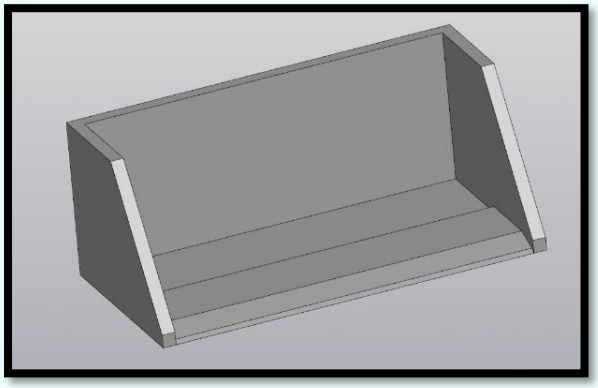


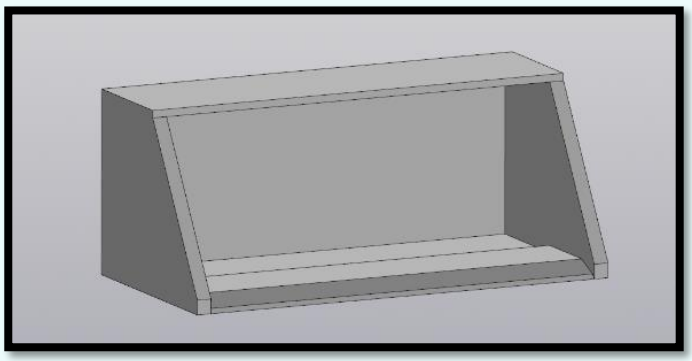


Идея – коробка с двумя дверцами, которые открываются с помощью сервоприводов.

Недостаток – Громоздкость всей конструкции и часть места не используется.

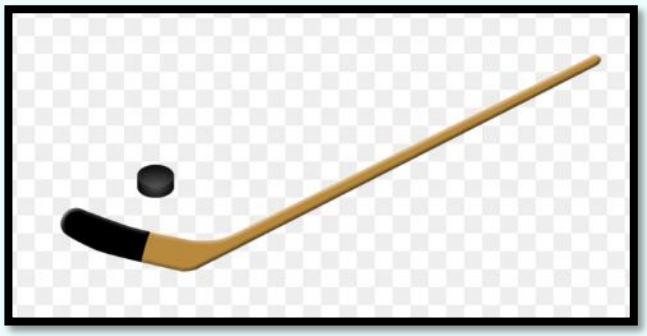
Прототип 2





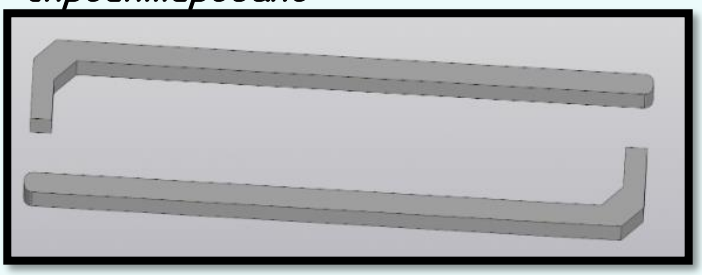
Улучшение – убраны дверцы и оставлено только полезное место. Добавлена сверху крышка, чтобы шарики не вылетали, если контейнер перевернут. В данном виде уже предусмотрен плавный заезд для шариков, а также ступенька возле дальней стенки, чтобы шарики не выкатывались от малейшего движения.

Захват шариков в контейнер



Идея реализации пришла, глядя на хоккейную клюшку.

На основе этого было спроектировано:

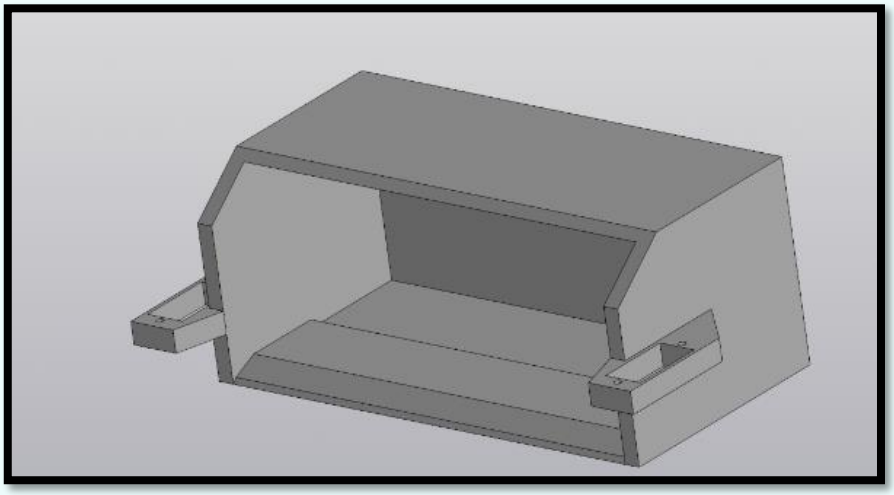


Необходимо было придумать, как расположить сервомоторы, чтобы эти «клюшки» захватывали шарики. Столкнулись с проблемой:

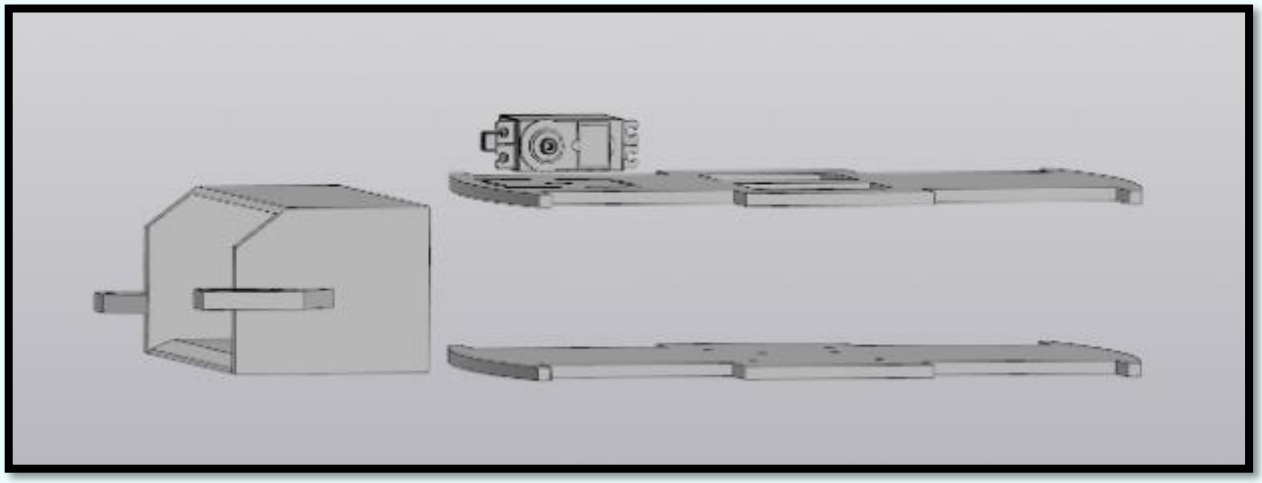
- *Большой скос не позволяет расположить надежные крепления для сервомотора MG90S. Из-за этого контейнер получил новые преобразования:*

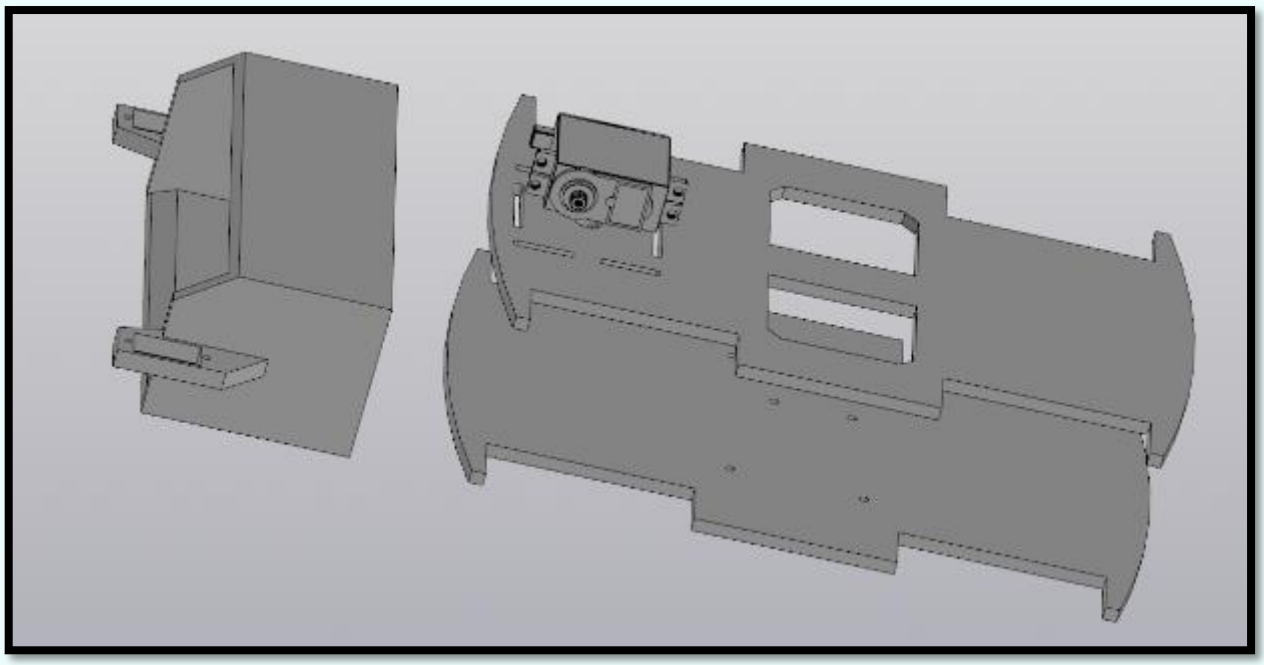
Прототип 3

Улучшения – крепления сервоприводов усилены.



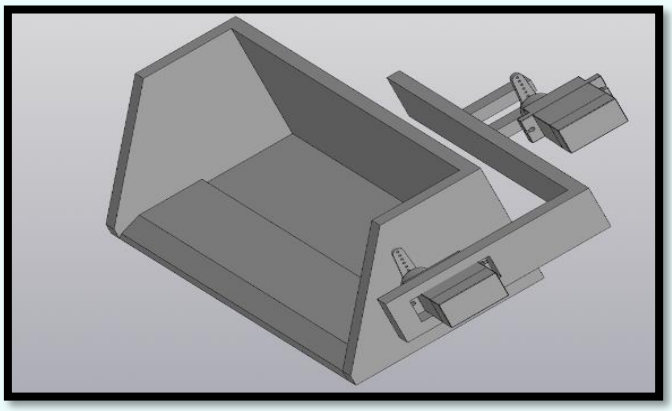
Примерка контейнера рядом с макетом пластин:





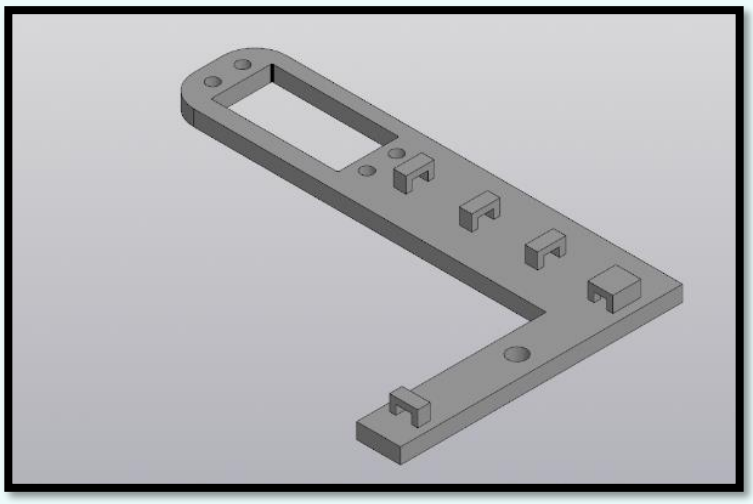
Разработка крепления сервопривода с ковшом

Прототип крепления 1



Недостатки – схема крепления вышла слишком громоздкой. Из-за этого перешли на Г- образное крепление под сервомоторы MG996R.

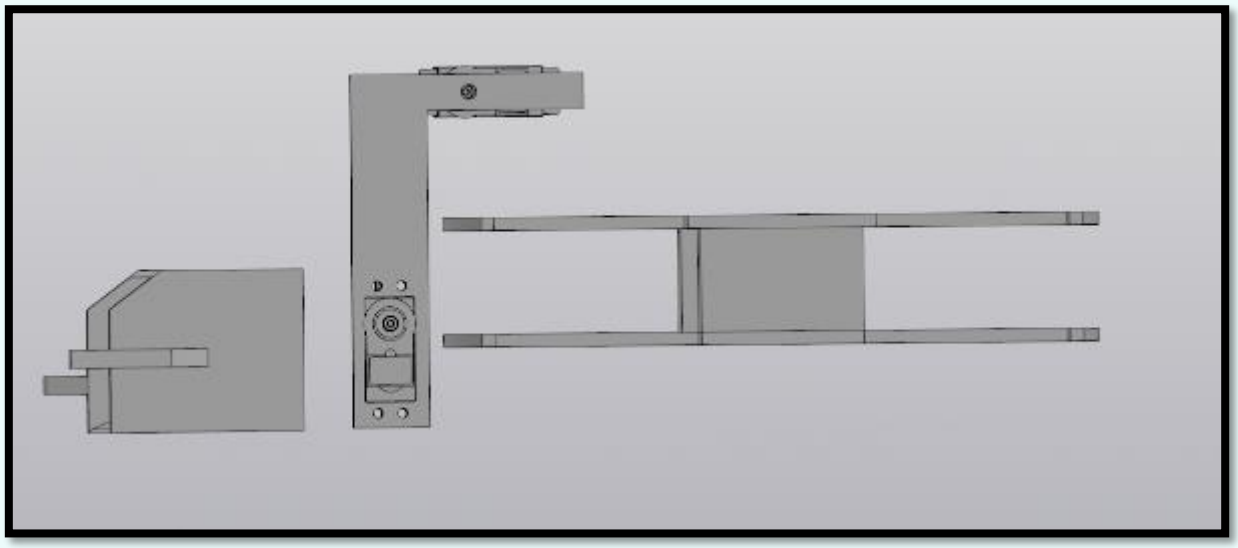
Прототип крепления 2



Улучшения – снижение громоздкости, появление кабель каналов.

Недостаток - верхняя часть крепежа при вращении будет упираться в пластину.

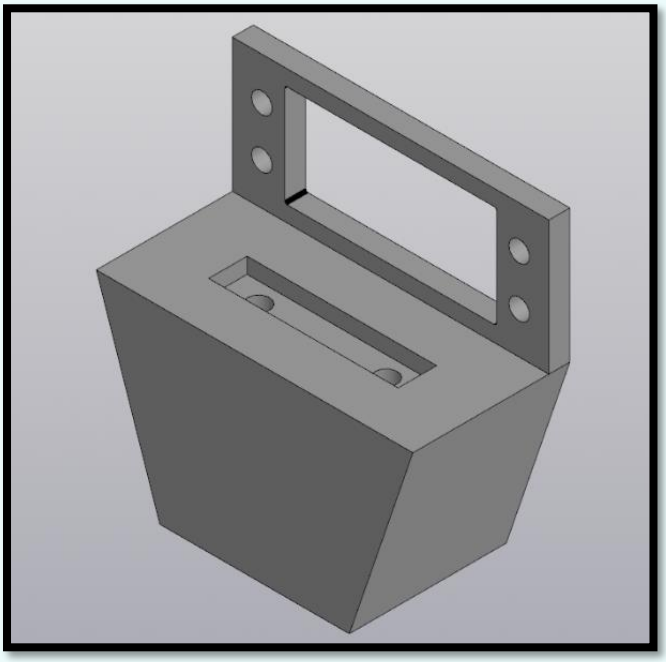
Прототип крепления 3



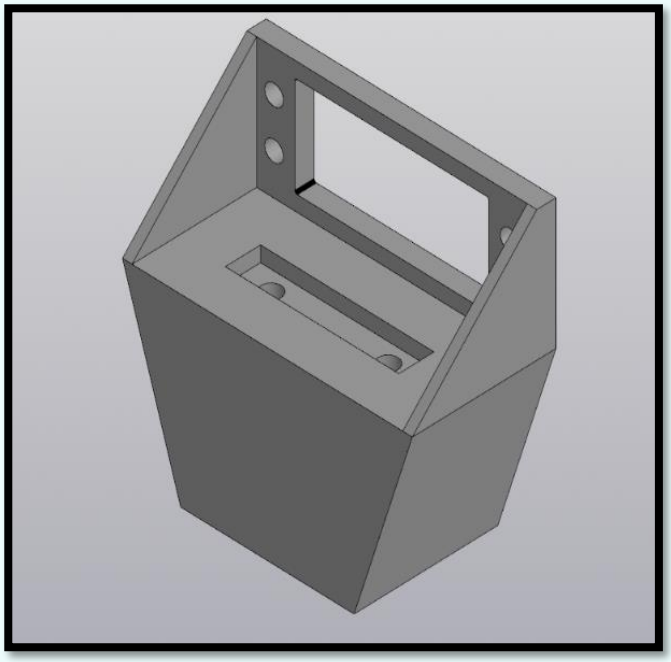
Улучшение – принято решение поднять сервомотор, а не проделывать отверстие в пластине, потому что такое отверстие очень сильно ослабит конструкцию.

Крепление сервомотора на пластину:

Преимущества такой подставки – углубление для шляпок болтов. Для усиления конструкции добавились ребра жесткости.

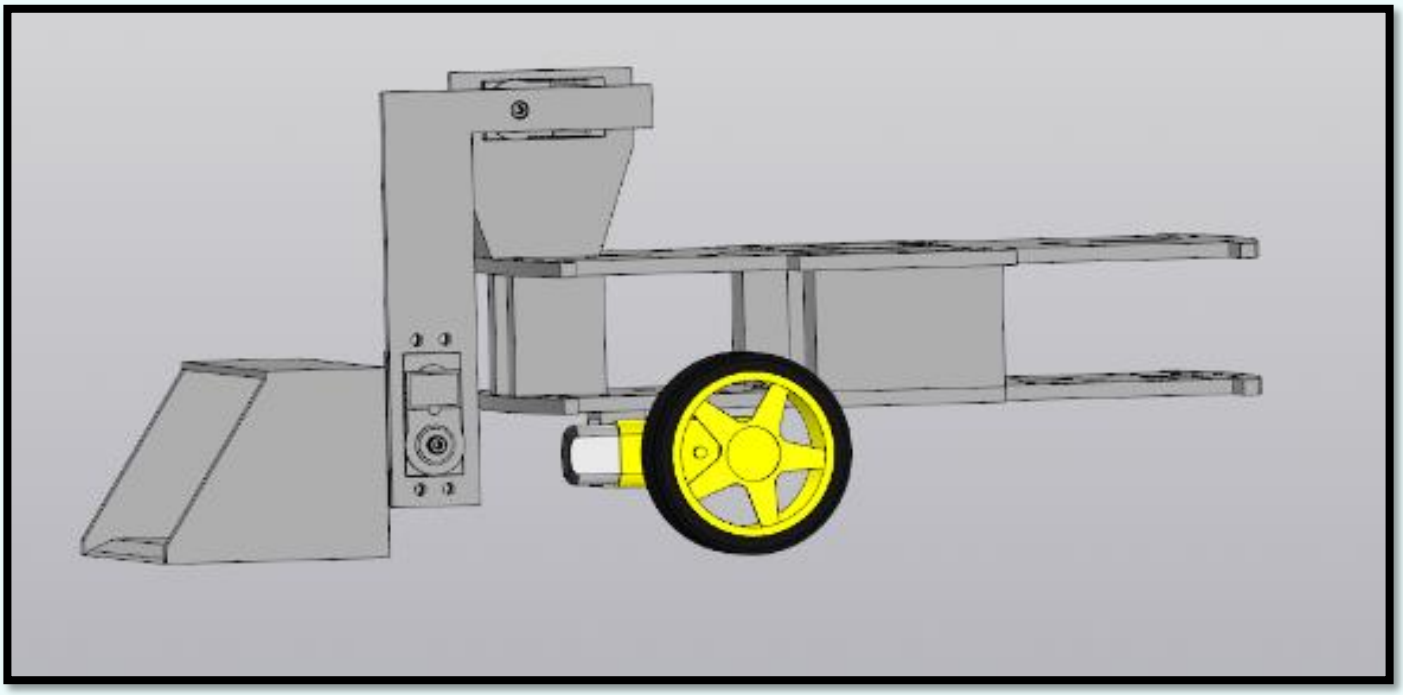


Версия 1

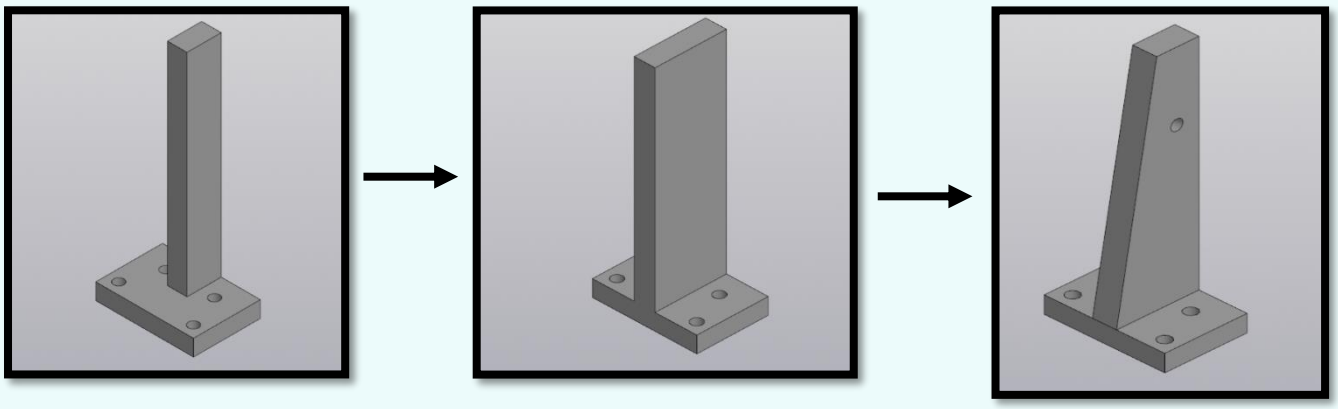


Укрепленная версия 2

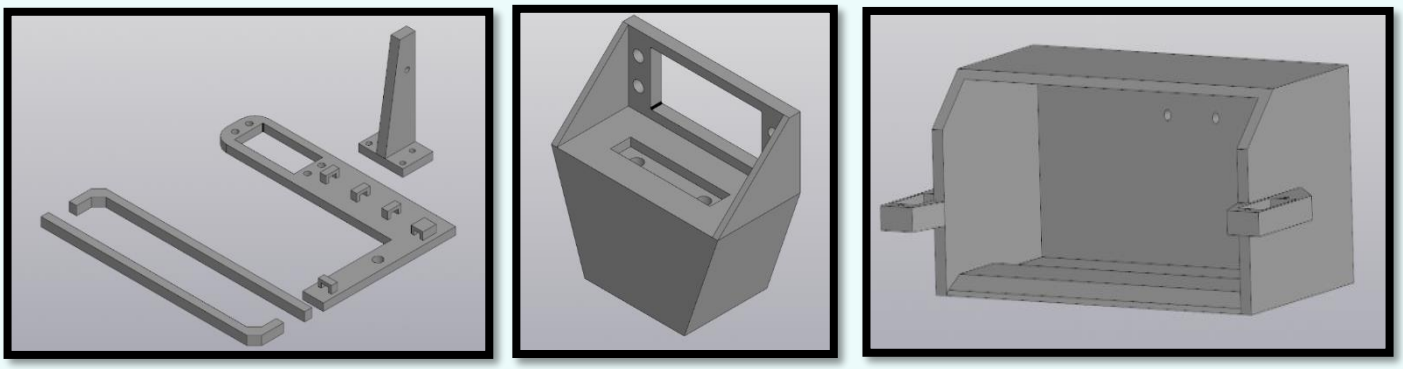
На данном этапе конструкция выглядит следующим образом:



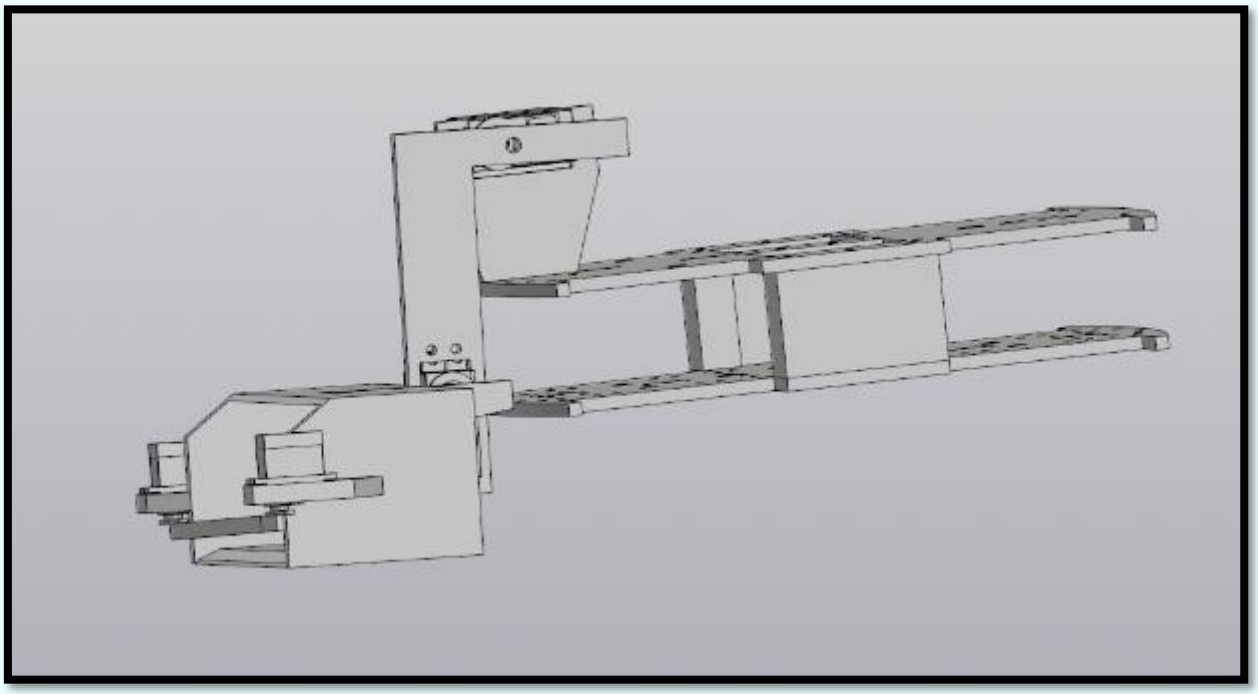
Далее необходимо придумать, как вращать контейнер относительно Г-образного крепежа:

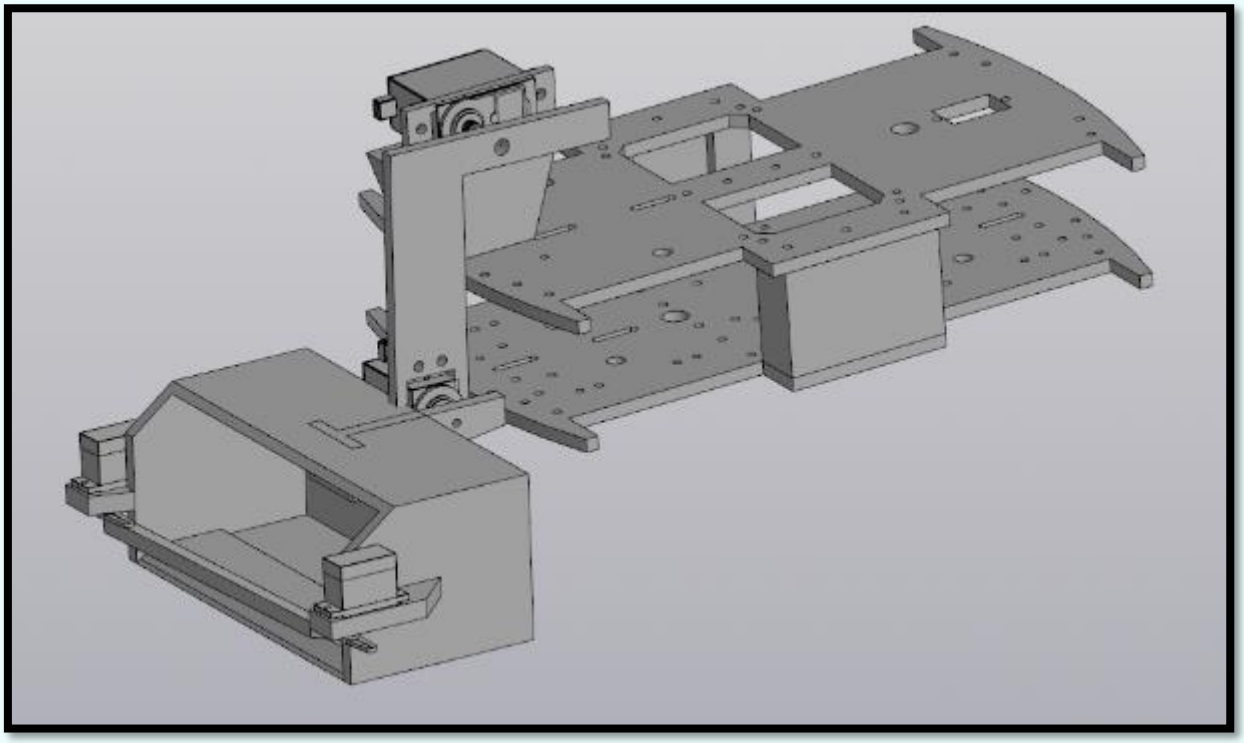


На печать были отправлены итоговые файлы:



Полноценная 3Д-модель захватывающего устройства:





5. Выбор вида пластика на печать

Для вариативности деталей лучше всего подходит 3D-печать. Существуют различные материалы для 3D печати. В нашем случае наиболее оптимальным является пластик. Выбран был PLA-пластик, так как он уже был в наличии и подходит под наши задачи.

PLA пластик основные плюсы:

- материал прост для печати
- отсутствует усадка при остывании
- отлично липнет к столу
- обладает достаточной прочностью и жесткостью и как следствие хорошо справляется с нагрузками на сжатие, излом и растяжение.

Ещё одним существенным достоинством является пригодность для последующих механических обработок. В ходе сборки возникали сложности с креплением и расположением некоторых элементов. С помощью сверления новых отверстий эту проблему удалось решить в кратчайшие сроки.

К минусам можно отнести маленький диапазон рабочей температуры готового изделия, но для нас это не является существенным фактором.

Вид пластика / Характеристика	ABS	PLA	HIPS	PETG	NYLON	ПОЛИКОРБАНАТ
Предел прочности	40 МПа	65 МПа	32 МПа	53 МПа	40-85 МПа	72 МПа
Жесткость	5/10	7.5/10	10/10	5/10	5/10	6/10
Прочность	8/10	4/10	7/10	8/10	10/10	10/10
Максимальная рабочая температура	98°C	52°C	100°C	73°C	80-95°C	121°C
Коэффициент теплового расширения	90 K^(-1)	68 K^(-1)	80 K^(-1)	60 K^(-1)	95 K^(-1)	69 K^(-1)
Плотность	1,04 гр/см^3	1,24 гр/см^3	1,03-1,04 гр/см^3	1,23 гр/см^3	1,06-1,14 гр/см^3	1,2 гр/см^3
Цена за метр, руб						
Легкость печати	5/10	9/10	6/10	9/10	8/10	6/10

Таблица сравнения популярных видов пластика для печати.

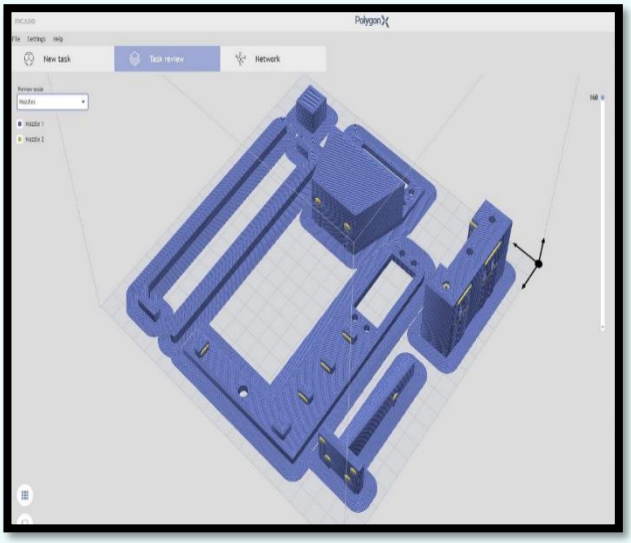
6. Подготовка 3Д-моделей к печати на 3Д-принтере

Ранее уже были подготовлены детали для печати (первая часть и вторая часть) теперь необходимо подготовить их для печати:

После создания 3D модели нужной детали, файл сохранили в формате STL (*.stl). При экспортировании указывали настройки:

- ✓ Максимальное линейное отклонение – 0.001
- ✓ Максимальное угловое отклонение – 4.2

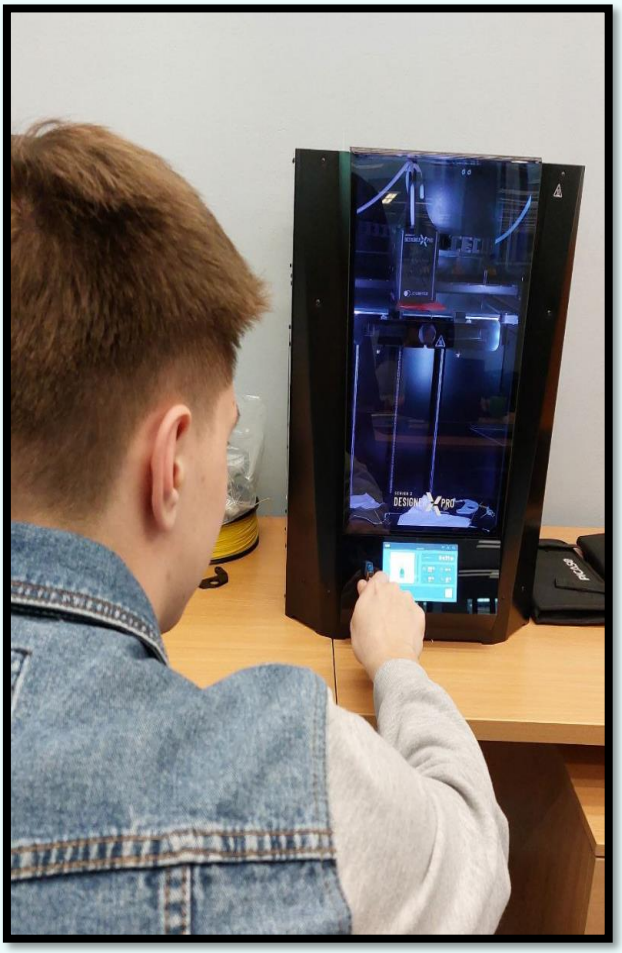
Эти настройки позволяют достичь наивысшего качества детали.
Для того, чтобы перенести данные о 3Д-модели от ПК к принтеру необходимо перевести информацию в G-код с помощью ПО – Слайсер (Оригинальный софт принтера). В основном рабочем поле слайсера выбрали порт подключения и модель принтера и отправили на печать.



В ходе работы нами был использован принтер Picaso 3D Designer X PRO S2. Его основные преимущества от других 3D принтеров с 2-мя экструдерами:

- – активная система управления соплами (поворотный механизм с сервоприводом)
- – использование клапана для перекрытия неактивного сопла во время печати
- – компенсация высоты слоев по оси Z при 2-х компонентной печати с помощью рабочего стола управляемого центральным процессором 3D принтера.

Процесс печати на 3Д-принтере

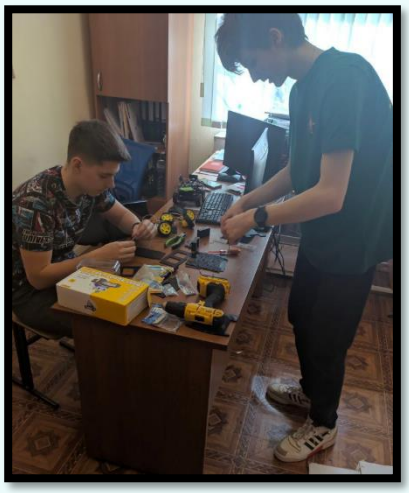


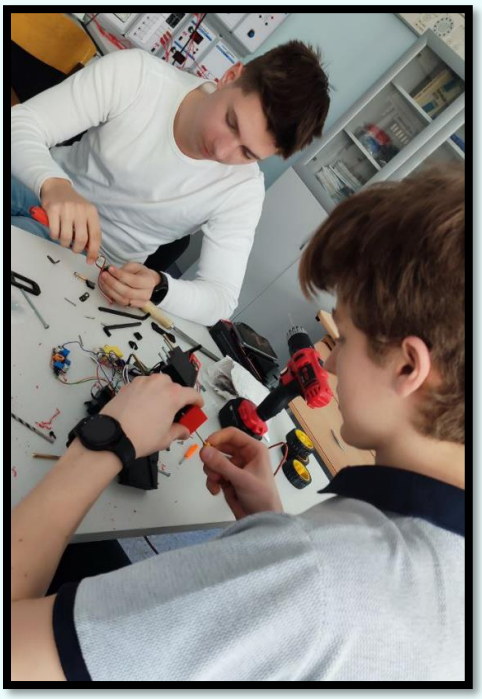
7. Сборка всех частей корпуса.

Процесс сборки всех частей корпуса начался со слесарных работ:

Команда подготавливает шпильки для крепления корпуса робота. Так же в этой мастерской просверливались дополнительные отверстия в корпусе “В.А.Л.Е.Р.А.”.

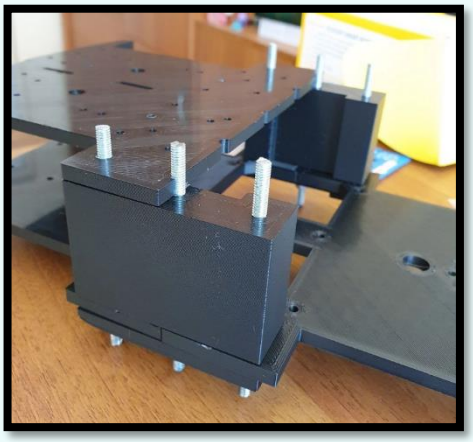
Далее наша команда приступила к сборке корпуса робота:

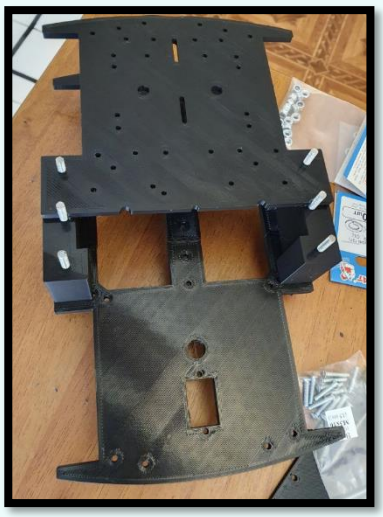


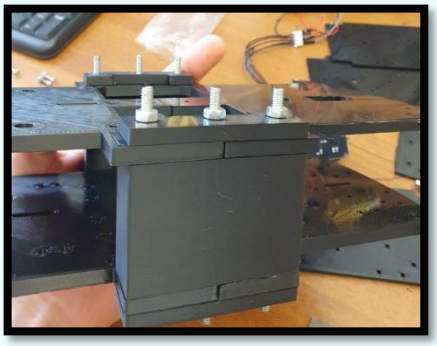


Поэтапная сборка робота:

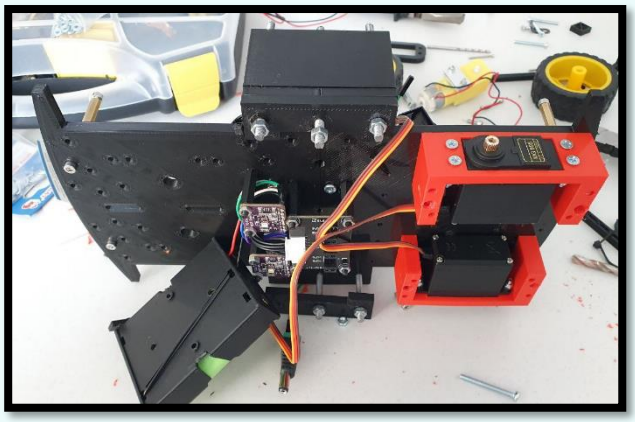
1. Сборка основания

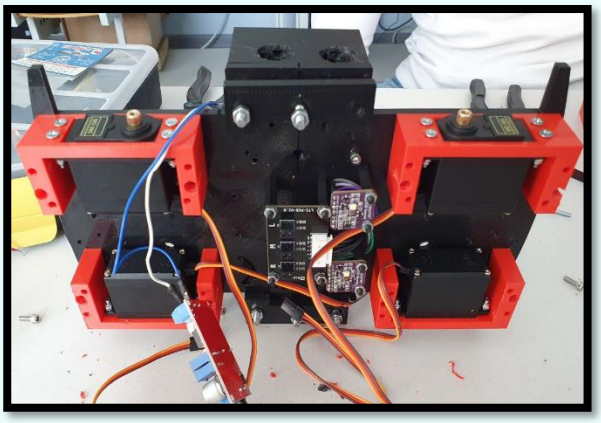


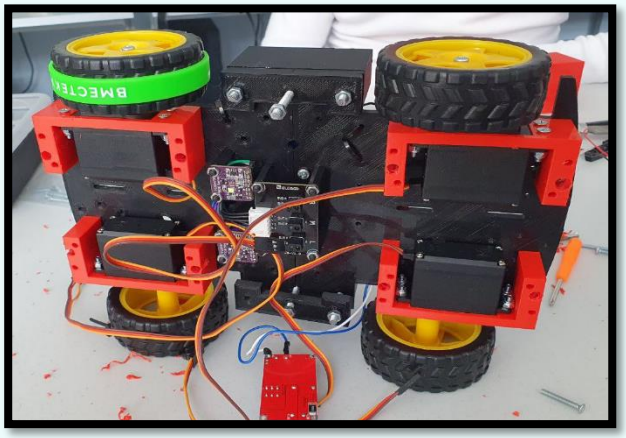


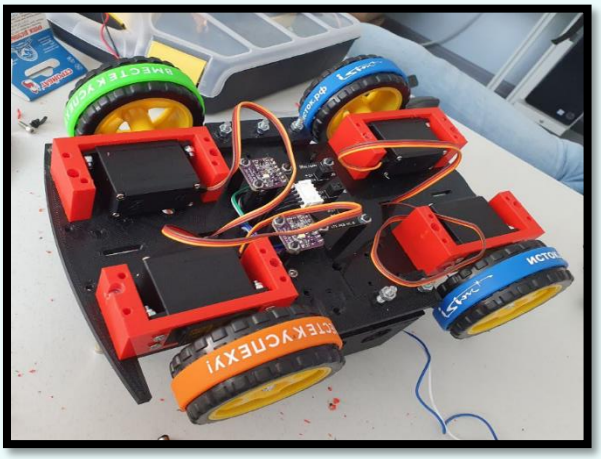


2. Установка элементной базы на корпус

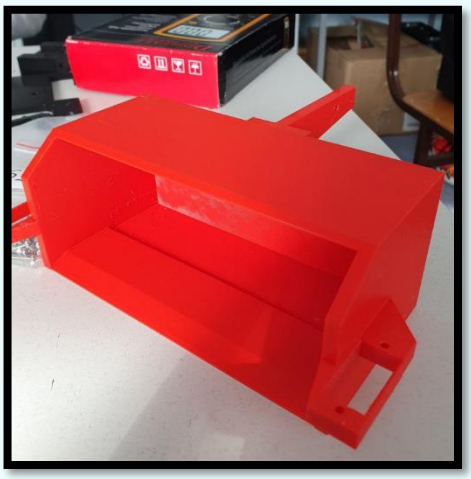


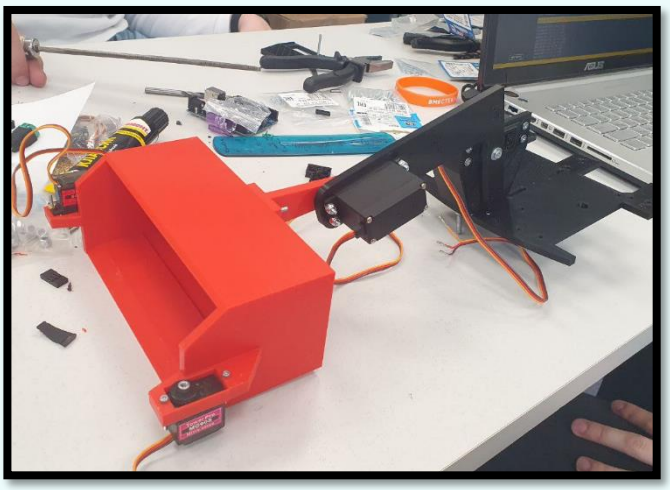


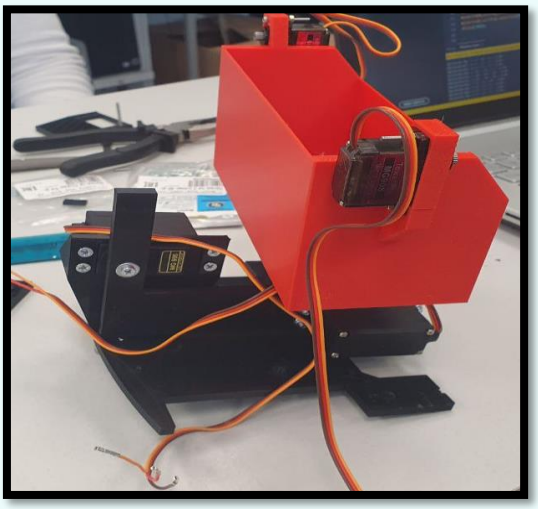


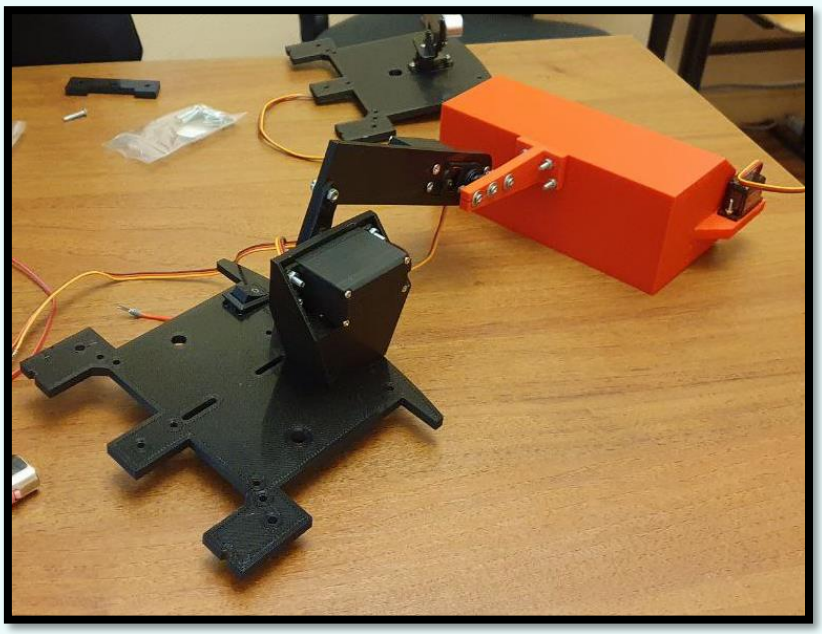


3. Сборка и установка ковша на корпус робота.

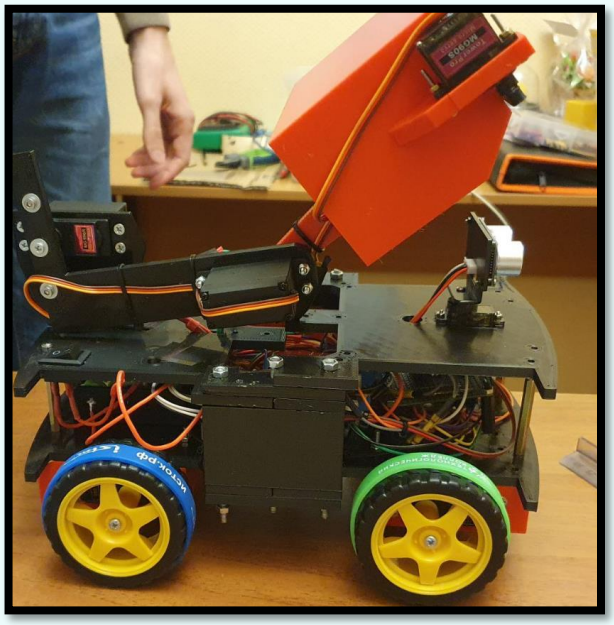


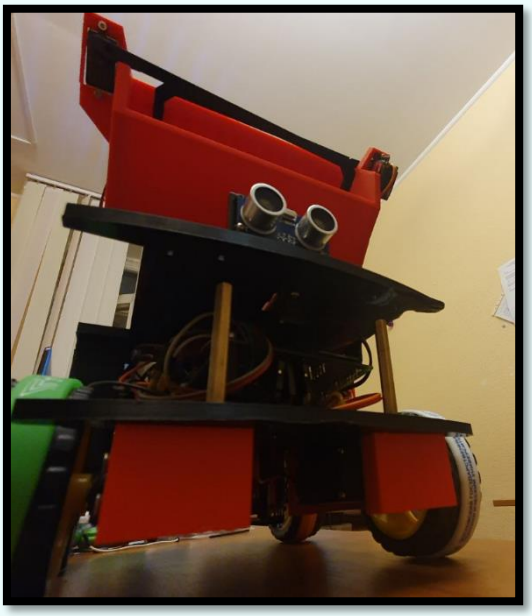


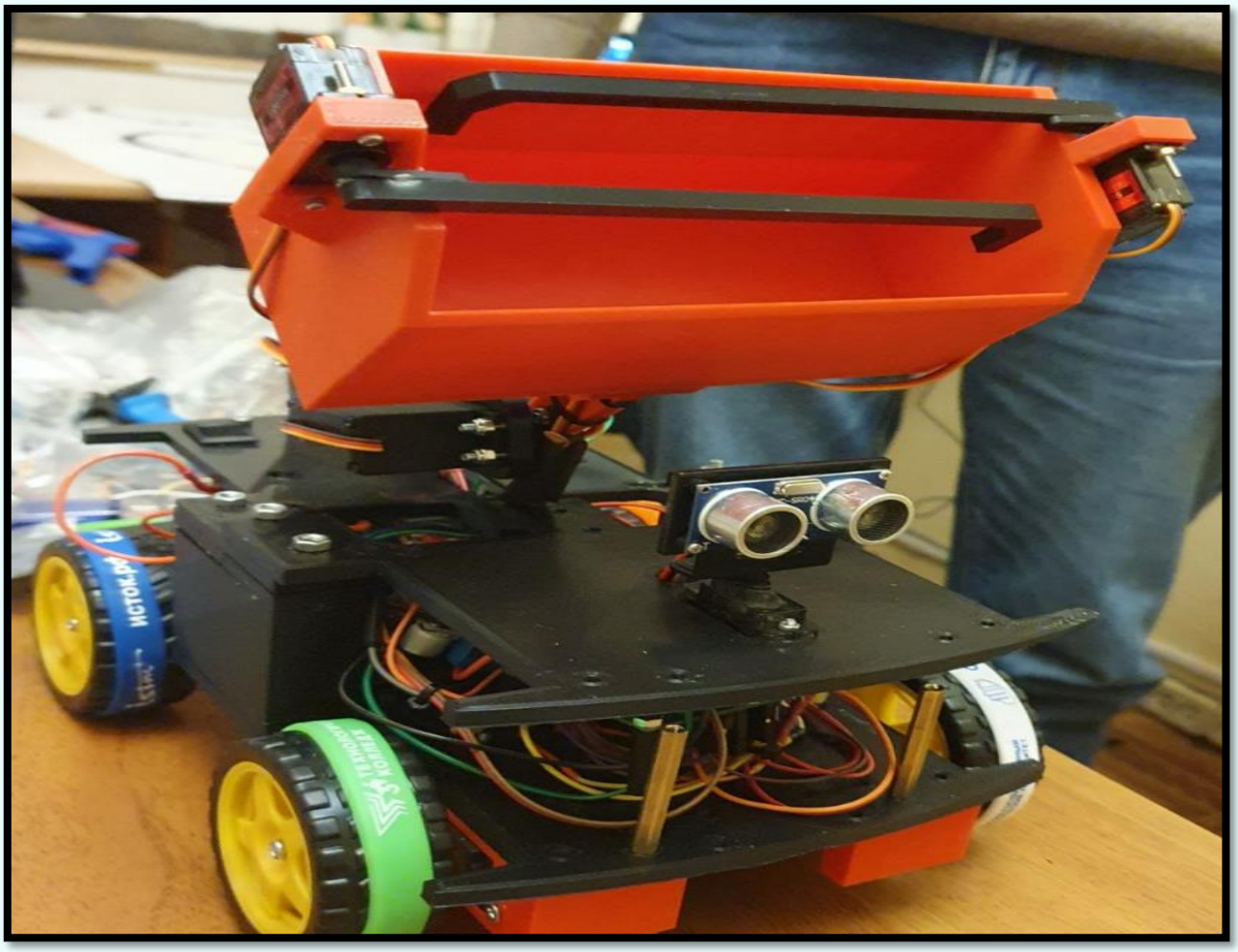




Полная сборка конструкции:







8. Программирование робота

Код программы езды по линии:

Алгоритм действия ПИД регулятора в системе сервоприводов.

1. *Считывание сигнала с датчиков и запись в массив*
2. *Расчет ошибки исходя из показаний датчиков*
3. *Расчет управляющего сигнала ПИД-регулятора*
4. *Расчет скорости моторов.*

```
//-------------------------------------------------------------------------------------------------------------------------------
//Расчет ошибки
//-------------------------------------------------------------------------------------------------------------------------------
  LFSensor[0] = analogRead(sensorL2);
  LFSensor[1] = analogRead(sensorL1);
  LFSensor[2] = analogRead(sensorM);
  LFSensor[3] = analogRead(sensorR1);
  LFSensor[4] = analogRead(sensorR2);

  if((   LFSensor[0]>=950  )&&(LFSensor[1]>=950 )&&(LFSensor[2]>=950)&&(LFSensor[3]>=950 )&&(LFSensor[4]<910 ))  {error = -4;}
  else if((LFSensor[0]>=950 )&&(LFSensor[1]>=950 )&&(LFSensor[2]>=950 )&&(LFSensor[3]<900 )&&(LFSensor[4]<910 ))  {error = -3;}
  else if((LFSensor[0]>=950 )&&(LFSensor[1]>=950 )&&(LFSensor[2]>=950 )&&(LFSensor[3]<910 )&&(LFSensor[4]>=950 ))  {error =- 2;}
  else if((LFSensor[0]>=950 )&&(LFSensor[1]>=950 )&&(LFSensor[2]<910 )&&(LFSensor[3]<910 )&&(LFSensor[4]>=950 ))  {error = -1;}
  else if((LFSensor[0]>=950)&&(LFSensor[1]>=950 )&&(LFSensor[2]<910 )&&(LFSensor[3]>=950 )&&(LFSensor[4]>=950 ))  {error = 0;}
  else if((LFSensor[0]>=950 )&&(LFSensor[1]<910 )&&(LFSensor[2]<910 )&&(LFSensor[3]>=950 )&&(LFSensor[4]>=950 ))  {error =1;}
  else if((LFSensor[0]>=950 )&&(LFSensor[1]<910 )&&(LFSensor[2]>=950 )&&(LFSensor[3]>=950 )&&(LFSensor[4]>=950 ))  {error = 2;}
  else if((LFSensor[0]<910 )&&(LFSensor[1]<910 )&&(LFSensor[2]>=950 )&&(LFSensor[3]>=950 )&&(LFSensor[4]>=950 ))  {error = 3;}
  else if((LFSensor[0]<910 )&&(LFSensor[1]>=950 )&&(LFSensor[2]>=950 )&&(LFSensor[3]>=950 )&&(LFSensor[4]>=950 ))  {error = 4;}

  P = error;
  I = I + error;
  D = error-previousError;
  PIDvalue = (Kp*P) + (Ki*I) + (Kd*D);
  previousError = error;

  int leftFMotorSpeed = 1500 + iniMotorPower - PIDvalue;
  int rightFMotorSpeed = 1500 - iniMotorPower*adj - PIDvalue;
  int leftBMotorSpeed = 1500 + iniMotorPower - PIDvalue;
  int rightBMotorSpeed = 1500 - iniMotorPower*adj - PIDvalue;

   constrain(leftFMotorSpeed, 1400, 1600);
   constrain(rightFMotorSpeed, 1400, 1600);
   constrain(leftBMotorSpeed, 1400, 1600);
   constrain(rightBMotorSpeed, 1400, 1600);

  leftFServo.writeMicroseconds(leftFMotorSpeed);
  rightFServo.writeMicroseconds(rightFMotorSpeed);
  leftBServo.writeMicroseconds(leftBMotorSpeed);
  rightBServo.writeMicroseconds(rightBMotorSpeed);
```

Прохождение пунктирной линии и поворотов на 90 градусов

```
//-------------------------------------------------------------------------------------------------------------------
//Прохождение пунктира
//-------------------------------------------------------------------------------------------------------------------

  if ((error == 0) && (analogRead(sensorM) >= 950)) {
    leftFServo.write(120);
    rightFServo.write(60);
    rightBServo.write(60);
    leftBServo.write(120);

    delay(100);
  }


if(flag == 0){
  if(((LFSensor[2]<900)&&(LFSensor[3]<900)&&(LFSensor[4]<900))||((LFSensor[2]<900)&&(LFSensor[3]<900))){
    leftFServo.write(120);
    rightFServo.write(120);
    rightBServo.write(120);
    leftBServo.write(120);
    delay(400);
  }

    if(((LFSensor[0]<900)&&(LFSensor[1]<900)&&(LFSensor[2]<900))||((LFSensor[0]<900)&&(LFSensor[1]<900))){
    leftFServo.write(60);
    rightFServo.write(60);
    rightBServo.write(60);
    leftBServo.write(60);
    delay(400);
  }
}
```

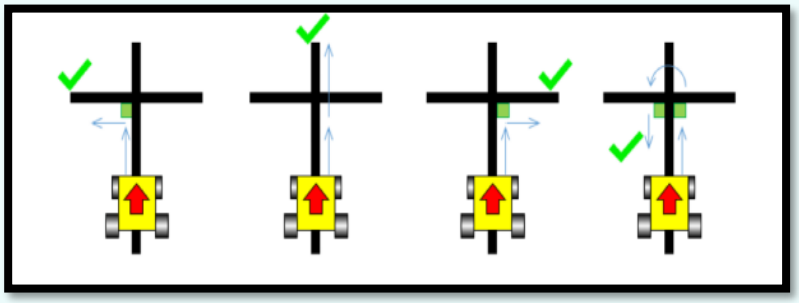
Условия прописаны с флагами, чтобы не сбивать работу других алгоритмов кода.

Детектирование зеленых меток на перекрестках.

Алгоритм основывается на различении зеленого и красного цвета из диапазона

Регламентом лиги предусматриваются следующие взаимодействия с зелеными маркерами:

Для этого были прописаны следующие условия: поворота налево, проезд прямо, поворот направо и разворот на 360 градусов.



Пример кода:

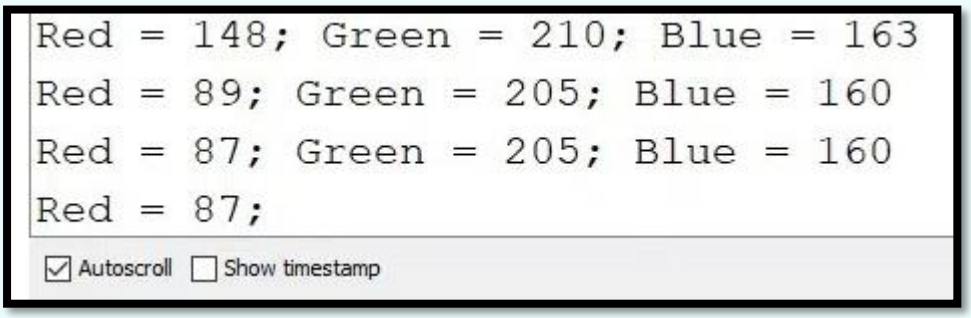
```
//-------------------------------------------------------------------------------------------
//Зеленая метка
//-------------------------------------------------------------------------------------------

   tcaselect(2);
   tcs.getRGB(&red, &green, &blue); // запрашиваем у датчика цветовые компоненты
   delay(20);
   tcaselect(3);
   tcs.getRGB(&red1, &green1, &blue1);
   delay(20);

               if((green>95)&&(green<135)&&(red>40)&&(red<70)){
                     flag = 1;
                     leftFServo.write(90);
                     rightFServo.write(90);
                     rightBServo.write(90);
                     leftBServo.write(90);
                       delay(1000);

                     leftFServo.write(120);
                     rightFServo.write(60);
                     rightBServo.write(60);
                     leftBServo.write(120);

                     delay(200);

                     leftFServo.write(120);
                     rightFServo.write(120);
                     rightBServo.write(120);
                     leftBServo.write(120);
                     delay(1600);
                     flag = 0;
                }

               if((green1>95)&&(green1<135)&&(red1>40)&&(red1<70)){
                     flag = 1;
                     leftFServo.write(90);
                     rightFServo.write(90);
                     rightBServo.write(90);
                     leftBServo.write(90);
                       delay(1000);
```

Детектирование датчиками цвета зеленых маркеров происходило следующим образом:

Сначала идет запрос сигнала с датчика:

```
tcaselect(2);
tcs.getRGB(&red, &green, &blue); // запрашиваем у датчика цветовые компоненты
delay(20);
tcaselect(3);
tcs.getRGB(&red1, &green1, &blue1);
delay(20);
```

Датчик TCS34725 выдает 3 значения: содержание красного (R), зеленого (G), и синего (B) цвета в диапазоне от 0 до 255.



Исходя из этих значений можно прописать условия, при которых датчик будет распознавать зеленую метку.

Эвакуация пострадавших из зоны.

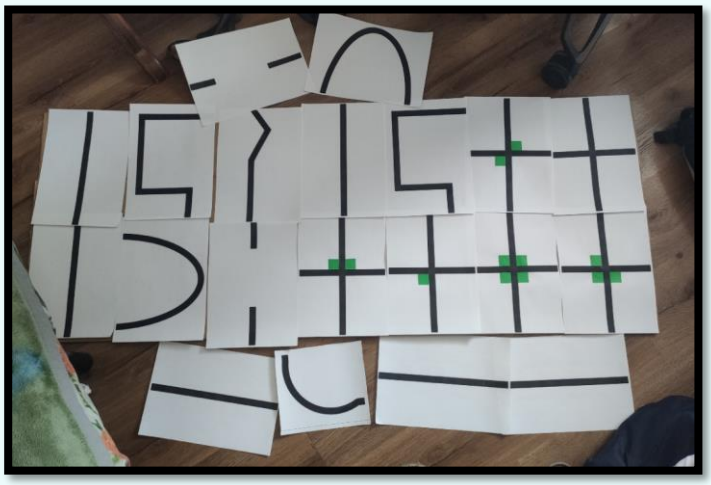
```
void evacuation(){

    leftFServo.write(90);
    rightFServo.write(90);
    rightBServo.write(90);
    leftBServo.write(90);


for (int pos = 0; pos <= 180; pos += 1) {
    mainscoop.write(pos);
    delay(25);
  }

clashRight.write(180);
delay(300);

for (int pos1 = 180; pos1 >= 150; pos1 -= 1) {
    secscoop.write(pos1);
    delay(35);
  }

  delay(5000);

for (int pos2 = 180; pos2 >= 15; pos2 -= 1) {
    clashRight.write(pos2);
    delay(20);
  }

          //поднятие наверх
          for (int pos = 180; pos > 0; pos -= 1) {
              mainscoop.write(pos);
              delay(25);
            }
          for (int pos1 = 150; pos1 <= 180; pos1 += 1) {
              secscoop.write(pos1);
              delay(35);
            }
}
```

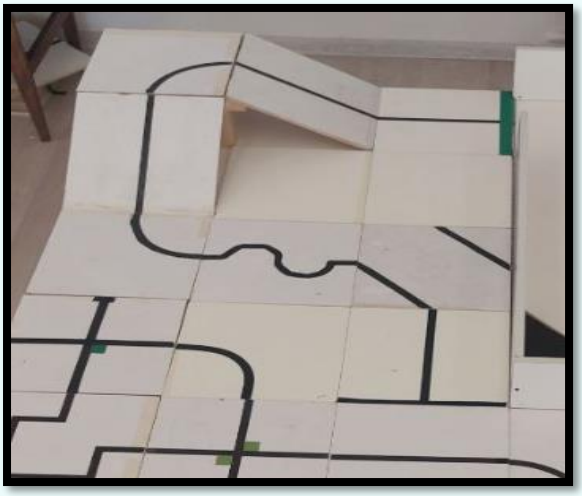
Алгоритм сделан на плавном управлении сервоприводами с помощью цикла for()

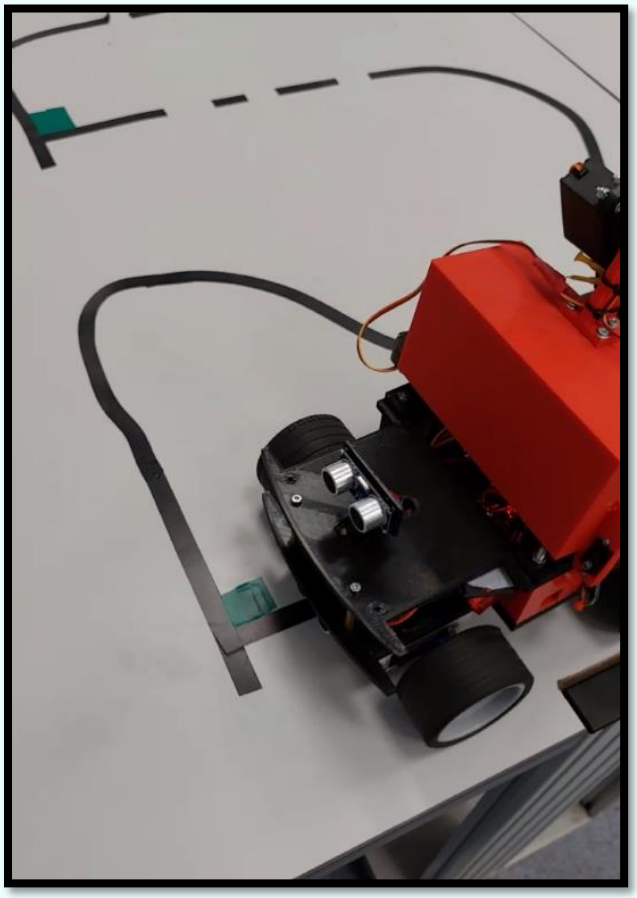
9. Отладка программы.

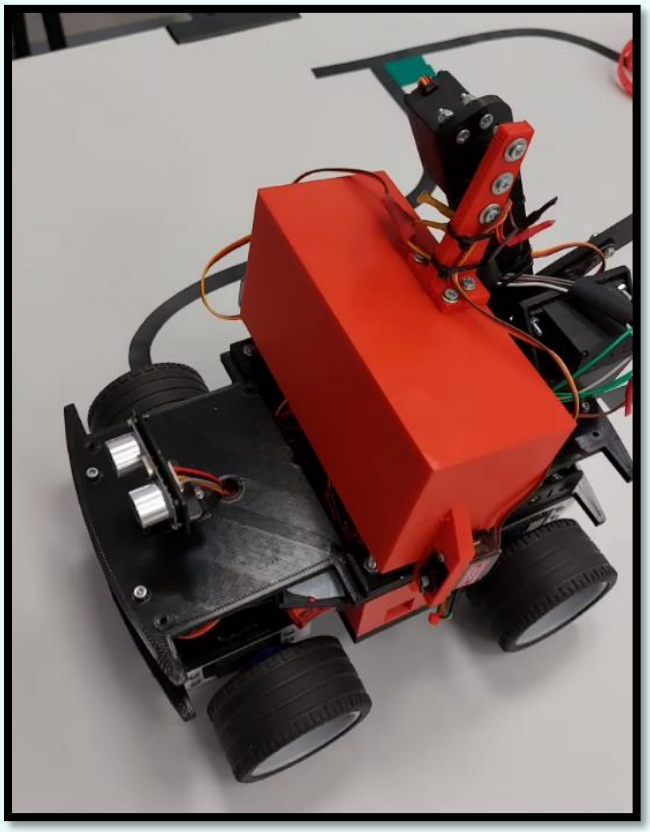
Отдельным и не менее важным этапом работы стала отладка кода программы на реальных условиях.

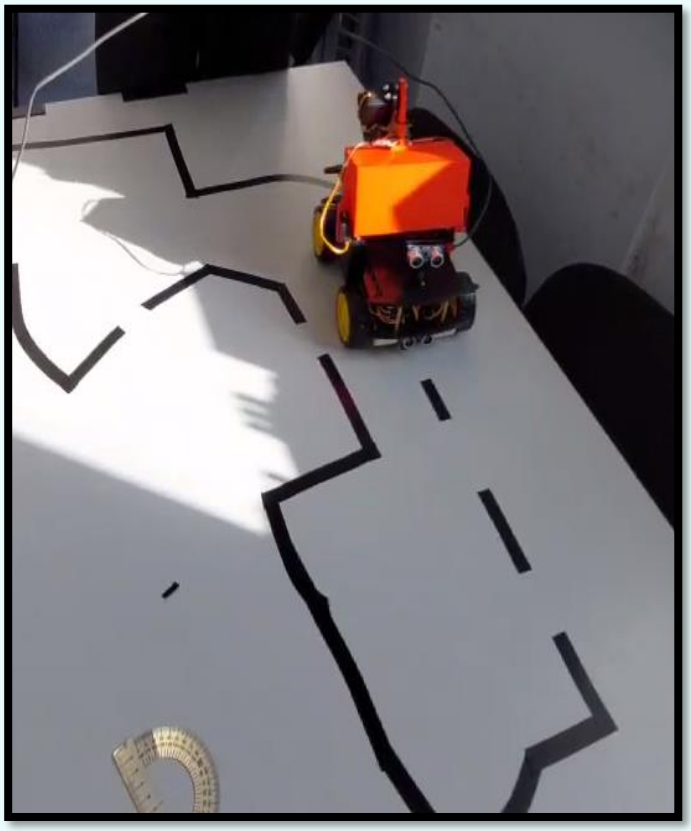
Для тестирования робота было построено множество трасс, препятствий, горок.











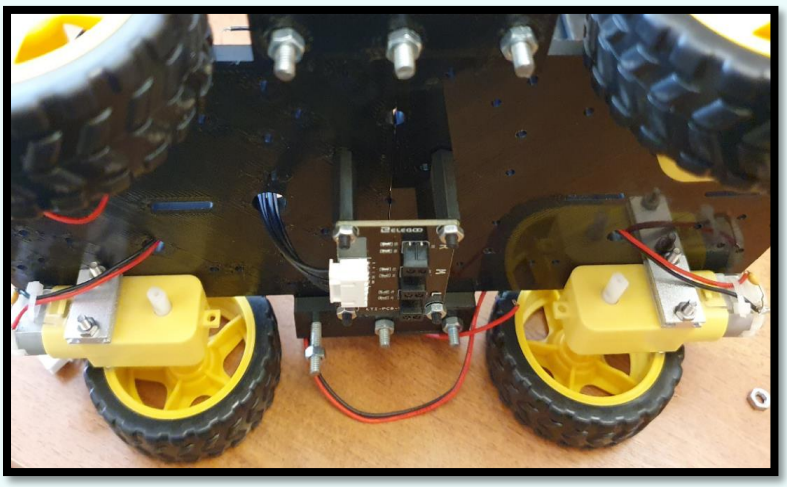
10. Исправление выявленных ошибок.

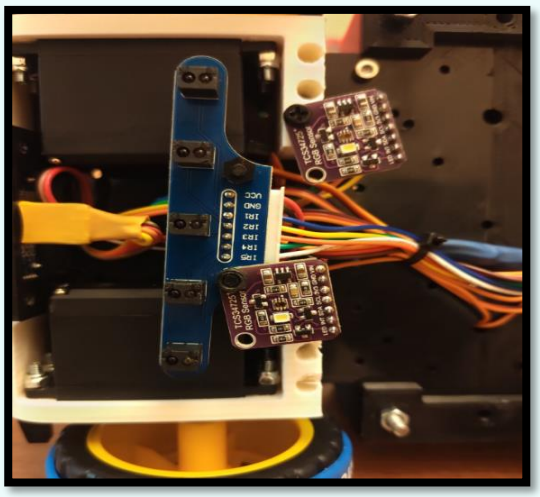
В ходе участия в Московском отборочном этапе Robocup Russia Open 2023 были обнаружены следующие недостатки:

1. ИК-датчик расположенный по центру робота, работал не корректно и мешал проезду через препятствие.
2. Зацеп колес
3. Непрактичная форма свеса для заезда на горку
4. Некорректная работа шин питания и земли на макетной плате.

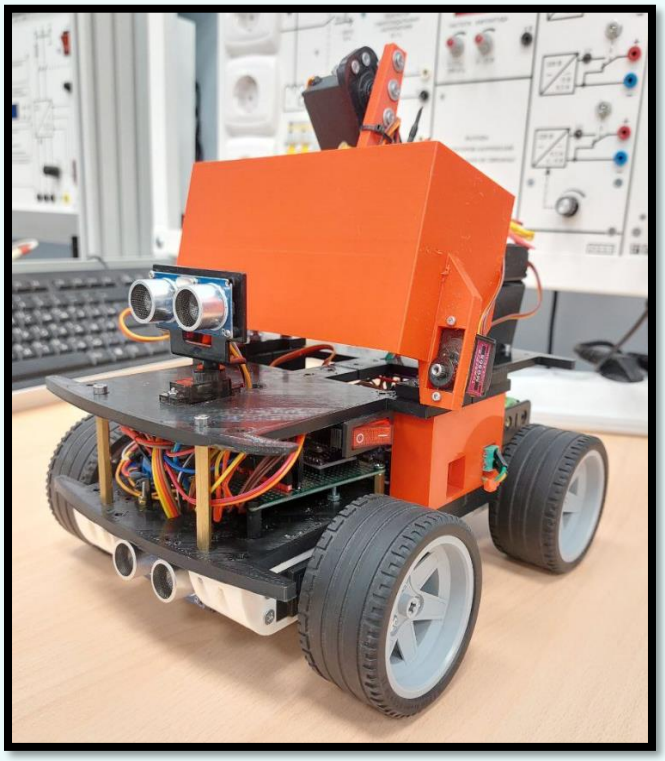
Решение данных проблем:

1. Заменили тип ИК-датчика и его расположение на нижней пластине. Переместили датчик на одну ось с передними колесами, что позволило опустить датчик ниже, при этом не мешая проезду через препятствие высотой до 1.5 см

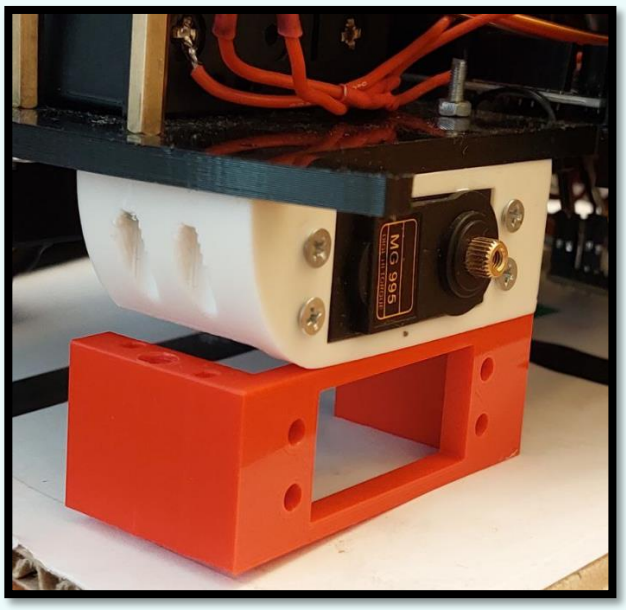


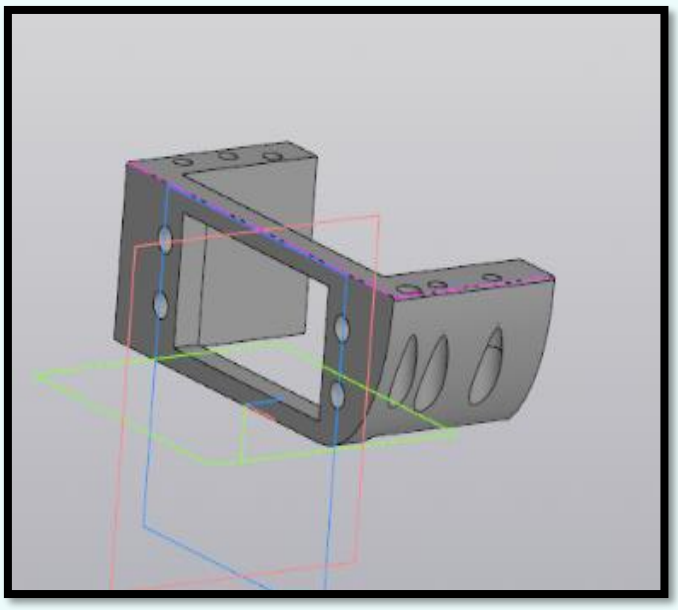


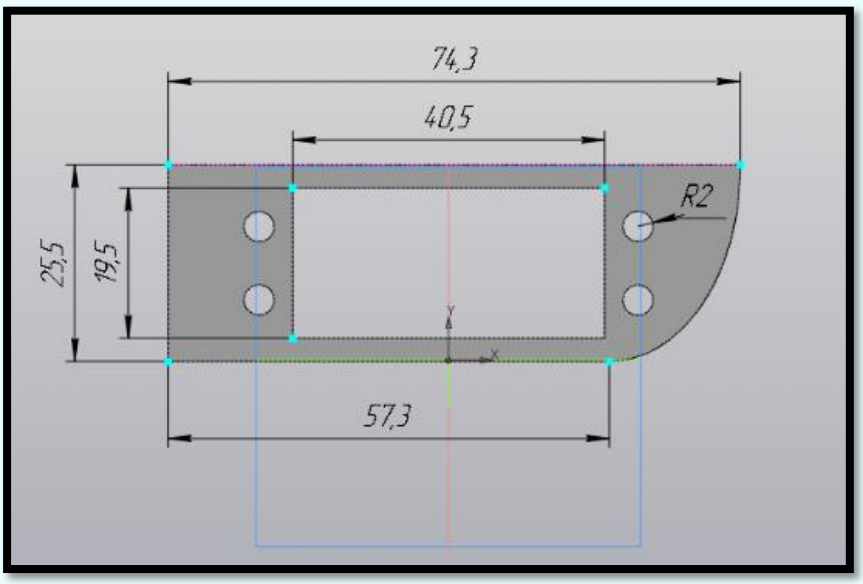
2. Замена предыдущих колес на колеса с большей шириной шины.

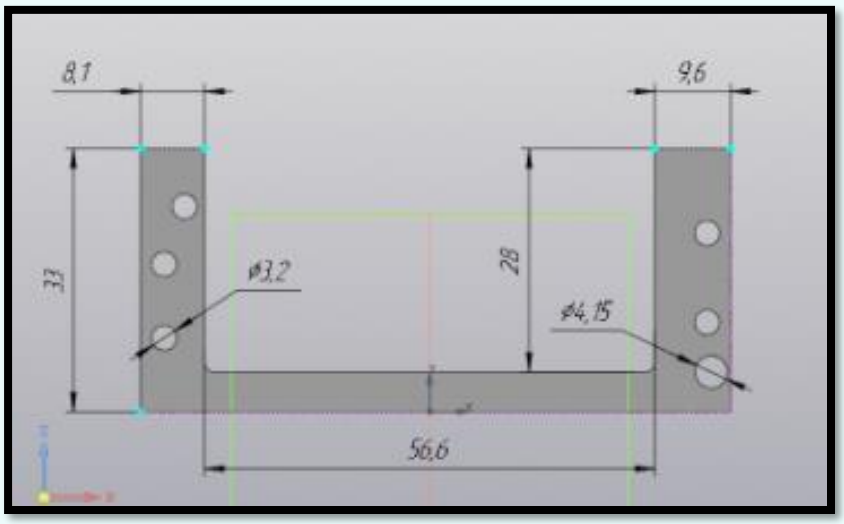


3. Деталь крепления сервомотора была доработана скруглением.









4. Расположение шин питания и земли на макетной плате было изменено, что позволило облегчить доступ к подключению и данная конструкция занимает меньше места.