**Реферат**

Выпускная квалификационная работа состоит из 4 глав, 86 стр., 51 рис., 2 табл., 11 источников, 5 приложений.

БЕЗАВАРИЙНОЕ ФУНКЦИОНИРОВАНИЕ; АВТОНОМНОЕ ДВИЖЕНИЕ; МНОГОФУНКЦИОНАЛЬНЫЙ РОБОТИЗИРОВАННЫЙ КОМПЛЕКС; МОНИТОРИНГ ПАРАМЕТРОВ СОСТОЯНИЙ

**Объектом исследования** магистерской диссертации является безаварийной функционирование роботизированного комплекса при автономном движении по маршруту.

**Предметом исследования** магистерской диссертации является прогнозирование возникновения опасных режимов функционирования роботизированного комплекса при локальном движении по траектории.

**Цель работы**: получить модель прогнозирования опасных режимов функционирования МРК при локальном движении по траектории.

В результате была получена модель прогнозирования опасных режимов функционирования роботизированного комплекса при локальном движении по траектории.

Практическая значимость заключается в получении программной реализации модели прогнозирования опасных режимов функционирования роботизированного комплекса при локальном движении по траектории для дальнейших работ по гранту РФФИ №17-47-590128, выполняемого кафедрой ИТАС.

Оглавление

[Список обозначений и сокращений 5](#_Toc485220024)

[Введение 6](#_Toc485220025)

[1 Аналитическая часть 9](#_Toc485220026)

[1.1 Обзор литературы 9](#_Toc485220027)

[1.2 Описание МРК 14](#_Toc485220028)

[1.3 Постановка задачи построения пути 17](#_Toc485220029)

[1.4 Постановка задачи безаварийного движения 19](#_Toc485220030)

[Выводы по разделу 1 22](#_Toc485220031)

[2 Теоретическая часть 23](#_Toc485220032)

[2.1 Планирование пути 23](#_Toc485220033)

[2.2 Метод оптимизации 24](#_Toc485220034)

[2.3 Определение функции поверхности 27](#_Toc485220035)

[2.3.1 Среднее арифметическое 28](#_Toc485220036)

[2.3.2 Полином Ньютона 28](#_Toc485220037)

[2.3.3 Полином Лагранжа 30](#_Toc485220038)

[2.4 Метод определения траектории 31](#_Toc485220039)

[2.5 Вычисление траектории 31](#_Toc485220040)

[2.6 Расчёт веса траекторий 33](#_Toc485220041)

[2.7 Выбор оптимальной траектории 33](#_Toc485220042)

[2.8 Прогнозирование возникновения опасных режимов функционирования МРК 34](#_Toc485220043)

[2.9 Определение характерных значений измеряемых показателей 37](#_Toc485220044)

[2.9.1 Определение общих значений 37](#_Toc485220045)

[2.9.2 Значения энергопотребления 38](#_Toc485220046)

[2.9.3 Определение значений параметров рулевого управления 41](#_Toc485220047)

[2.9.4 Определение значений параметров тормозной системы 42](#_Toc485220048)

[2.9.5 Определение значений параметров подвеса камеры 43](#_Toc485220049)

[2.9.6 Определение неисправностей на основании вибраций 43](#_Toc485220050)

[Выводы по разделу 2 47](#_Toc485220051)

[3 Практическая часть 48](#_Toc485220052)

[3.1 Алгоритм поиска локального пути 48](#_Toc485220053)

[3.2 Алгоритм нахождения интеграла 51](#_Toc485220054)

[3.3 Алгоритм построения траектории 53](#_Toc485220055)

[3.4 Алгоритм избегания наезда на препятствие 56](#_Toc485220056)

[3.5 Алгоритм избегания застревания в яме 58](#_Toc485220057)

[3.6 Алгоритм избегания проскальзывания/пробуксовки 60](#_Toc485220058)

[3.7 Алгоритм сообщения об обрыве электрических цепей 62](#_Toc485220059)

[3.8 Алгоритм сообщения о неисправностях по вибрациям 64](#_Toc485220060)

[3.9 Алгоритм следования платформы по маршруту 66](#_Toc485220061)

[Выводы по главе 3 70](#_Toc485220062)

[4 Экспериментальная часть 71](#_Toc485220063)

[4.1 Определение поверхности 71](#_Toc485220064)

[4.2 Построение траекторий и выбор оптимальной траектории 72](#_Toc485220065)

[4.4 Верификация модели следования по маршруту 74](#_Toc485220066)

[4.5 Верификация модели предотвращения возникновения пробуксовки и проскальзывания 77](#_Toc485220067)

[4.6 Верификация модели сообщения об обрыве электрических цепей 78](#_Toc485220068)

[4.7 Верификация модели сообщения о неисправностях по вибрациям 80](#_Toc485220069)

[Выводы по разделу 4 82](#_Toc485220070)

[Заключение 83](#_Toc485220071)

[Список используемых источников 85](#_Toc485220072)

Приложение А 87

Приложение Б 89

Приложение В 91

Приложение Г 94

Приложение Д 96

# Список обозначений и сокращений

МРК – многофункциональный роботизированный комплекс;

СТЗ – система технического зрения;

ТС – тормозная система;

АЧХ – амплитудно-частотная характеристика;

ГИС – геоинформационная система.

# Введение

**Актуальность.** Большой интерес представляет собой вопрос, касающийся автоматического проложения и последующего передвижения по траектории наземного транспортного средства. Некоторые из ведущих компаний уже рассматривают вопросы о реализации или реализовали на данный момент идею об автоматически передвигающихся автомобилях. Такие компании, как Google, BMW, Volvo реализовали данную идею. Так же частично реализованной идеей является функция автопарковки автомобилей. Функция автопарковки полностью управляет движением автомобиля: поворотом колес, скоростью, торможением.

Так же автоматическое передвижение роботов представляет интерес и в других отраслях, таких как медицина, туризм, сфера продаж. Данные роботы имеют функцию информирования или помощи людям. В работах Кошечкина А.А. [1], Старковой И.В. [3], Андриановой О.Г.[4], Подоплёкина Ю.Ф. [5], Синициной Н.В. [6] вопрос о безаварийном локальном движении рассматриваются с разных точек зрения, что, несомненно, является признаком интенсивного развития данного направления в мире технологий.

Основываясь на этом, можно прийти к выводу, что интерес к данной теме имеется и построение систем, обеспечивающих безаварийное локальное передвижение, имеет коммерческий и научный интерес и является весьма **актуальной** на сегодняшний день.

**Научная новизна** данной выпускной квалификационной работы заключается в реализации модели прогнозирования опасных режимов функционирования МРК при локальном движении по траектории на основании мониторинга данных с датчиков.

**Целью** магистерской диссертации является получение модели прогнозирования опасных режимов функционирования МРК при локальном движении по траектории.

Для достижения поставленной цели в работе были поставлены следующие задачи:

- обзор источников на заданную тему;

- обзор методов нахождения функции поверхности;

- выбор способа расчёта движения МРК по поверхности;

- расчет траектории движения МРК по поверхности;

- создание модели выбора и построения траектории движения МРК;

- определение перечня опасных режимов функционирования МРК;

- определение перечня измеряемых показателей;

- определение значений измеряемых показателей, характерных для аварийных ситуаций;

- создание модели избегания опасных режимов функционирования МРК;

- мониторинг параметров МРК при движении по траектории;

- верификация моделей.

**Объектом исследования** магистерской диссертации является безаварийной функционирование роботизированного комплекса при автономном движении по маршруту.

**Предметом исследования** магистерской диссертации является прогнозирование опасных режимов функционирования роботизированного комплекса при локальном движении по траектории.

**Практическая значимость** заключается в получении модели прогнозирования опасных режимов функционирования МРК при локальном движении по траектории; полученные в ходе выполнения ВКР результаты в дальнейшем будут использованы для работ по гранту РФФИ №17-47-590128, выполняемого кафедрой ИТАС.

В первом разделе приведен обзор литературы на заданную тему, производится описание технических параметров МРК, постановка задачи локального движения по траектории и постановка задачи безаварийного функционирования МРК.

Во втором разделе производится выбор метода оптимизации, выбор способа определения функции поверхности, производится выбор метода расчета траектории, определение методов прогнозирования опасных режимов функционирования МРК.

В третьем разделе осуществляется создание модели построения и выбора оптимальной траектории движения МРК, создание модели прогнозирования опасных режимов функционирования МРК.

В четвертом разделе производится верификация модели построения и следования МРК по траектории, верификация модели избегания опасных режимов функционирования МРК.

В заключении обозначены теоретические и практические выводы по проведенным исследованиям и разработкам модели прогнозирования опасных режимов функционирования МРК при локальном движении по траектории, так же обозначена перспектива развития данной темы.

# 1 Аналитическая часть

## 1.1 Обзор литературы

Вопрос автономного движение наземного транспорта так же можно встретить в различных научных публикациях. Например, в статье [1] робот не использует никаких данных о поверхности, кроме информации «можно идти / нельзя идти», если видит препятствие. Препятствия ищутся в заданном квадрате на прямой следования до конечной точки, если препятствие не «касается» робота и имеется запас для прохождения мимо препятствия, то по данному месту прокладывается траектория. Далее строится траектория движения робота. Если двигаться по прямой, то строится прямая, которая делится на отрезки. Если необходимо огибать препятствия, то строится кривая Безье 2 порядка. В случае кривой Безье, далее производится дискретизация траектории с определенным шагом и производится аппроксимация кусочно-линейными функциями.

А в работе авторов [2] считываются данные с акселерометра и гироскопа. Происходит разбиение осей на сегменты. Затем, по движению объекта по поверхности, собирается описание поверхности для дальнейшей классификации. В дальнейшем, базируясь на классификации данных о поверхности, строится траектория. Вычисления происходят с помощью нейронной сети Хопфилда. В ходе вычислений сетью решается вопрос о дальнейшем движении.

В статье [3] осуществляется построение узлов-препятствий на пути движения. Точки, по которым движется робот, назначает пользователь. Так же назначаются статические препятствия. В силу возможных динамических препятствий, назначаются стоимости узлов, которые соответствуют препятствиям. В стоимость узла включаются наименьшая стоимость приближения к препятствию и эвристическое приближение стоимости пути к цели от текущего положения. Во избежание столкновения с динамическим препятствием: остановиться и продолжить движение после исчезновения препятствия или же изменить скорость, чтобы избежать столкновения. Принятие решения строится на нечеткой логике.

В работе [4] строится заранее опорная траектория, по которой следует робот. Для каждой точки, которую необходимо обогнуть и которая появляется по пути следования по опорной траектории, строится обходная траектория в виде кубического параметризированного сплайна, которые задаются значениями функции в узлах и значениями первых или вторых производных на границах отрезков, что обеспечивает гладкость криволинейной траектории. Каждое колесо имеет своё управление. Путем вычислений определяется скорость движения колес, их угол поворота или торможение.

Вопрос безаварийного движения так же рассматривается во многих работах. В статье [5] безаварийное движение осуществляется с помощью сопоставления векторов текущего движения и требуемого, в случае несовпадения векторов допустимых значений линейных и угловых координат и скоростей происходит переход к ручному управлению платформы.

В статье [6] безаварийное движение рассматривается по отдельности для каждого аварийного случая. Так, для избегания столкновения с каким-либо объектом отслеживаются расстояния до ближайших объектов впереди и позади и по бокам данного, а так же скорость движения данного объекта. Далее происходит вычисление функций принадлежности для входных данных той или иной аварийной ситуации и определяется вероятность аварии. В случае каждой из диапазонов вероятностей заранее определен перечень действий. Вся модель выстроена на нечёткой логике.

В работе [7] предотвращение аварийных режимов функционирование предлагается рассматривать поэлементно. На основании имеющихся элементов платформы осуществляется контроль следующих параметров: наезд на препятствие (тактильный датчик), низкий заряд батареи (датчик заряда батареи), неисправность исполнительных механизмов (энкодеры колес, датчик давления в шинах), продолжительная работа в предельных режимах (датчик тока), недостаточная удаленность от «опасных» объектов. Задача избегания опасных режимов функционирования разбивается на 2 подзадачи: избегание внешних режимов и внутренних.

«Модель прогнозирования есть функциональное представление, адекватно описывающее исследуемый процесс и являющееся основой для получения его будущих значений. В той же кулинарной аналогии модель есть список ингредиентов и их соотношение, необходимый для нашего блюда — прогноза. Если посмотреть внимательно, то быстро выясняется, что понятие «метод прогнозирования» гораздо шире понятия «модель прогнозирования». В связи с этим на первом этапе классификации обычно делят методы на две группы: интуитивные и формализованные (рисунок 1).

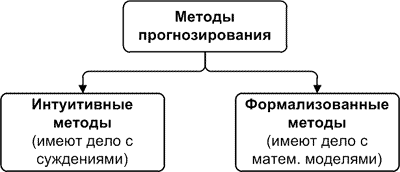


Рисунок 1 – Классификация методов прогнозирования

Интуитивные методы прогнозирования имеют дело с суждениями и оценками экспертов. На сегодняшний день они часто применяются в маркетинге, экономике, политике, так как система, поведение которой необходимо спрогнозировать, или очень сложна и не поддается математическому описанию, или очень проста и в таком описании не нуждается.

Формализованные методы — описанные в литературе методы прогнозирования, в результате которых строят модели прогнозирования, то есть определяют такую математическую зависимость, которая позволяет вычислить будущее значение процесса, то есть сделать прогноз. Классификация методов представлена на рисунке 2.

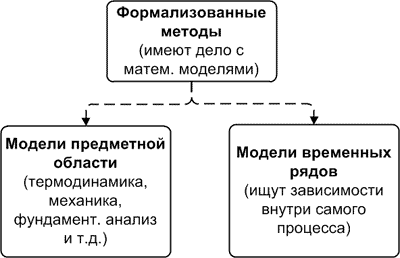


Рисунок 2 – Классификация формализованных методов

Модели предметной области — такие математические модели прогнозирования, для построения которых используют законы предметной области. Например, модель, на которой делают прогноз погоды, содержит уравнения динамики жидкостей и термодинамики. Прогноз развития популяции делается на модели, построенной на дифференциальном уравнении. Прогноз уровня сахара крови человека, больного диабетом, делается на основании системы дифференциальных уравнений. Словом, в таких моделях используются зависимости, свойственные конкретной предметной области. Такого рода моделям свойственен индивидуальный подход в разработке.

Модели временных рядов — математические модели прогнозирования, которые стремятся найти зависимость будущего значения от прошлого внутри самого процесса и на этой зависимости вычислить прогноз. Эти модели универсальны для различных предметных областей, то есть их общий вид не меняется в зависимости от природы временного ряда. Мы можем использовать нейронные сети для прогнозирования температуры воздуха, а после аналогичную модель на нейронных сетях применить для прогноза биржевых индексов. Это обобщенные модели, как кипяток, в которые если бросить продукт, то он сварится вне зависимости от его природы. Модели временных рядов можно разделить на две группы: статистические и структурные (рисунок 3).

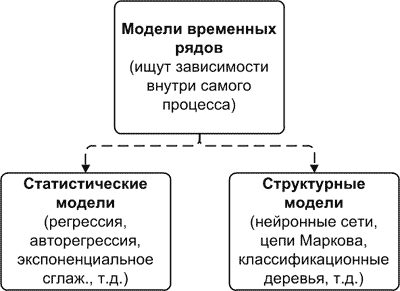


Рисунок 3 – Классификация модели временных рядов

В статистических моделях зависимость будущего значения от прошлого задается в виде некоторого уравнения. К ним относятся:

1. регрессионные модели (линейная регрессия, нелинейная регрессия);

2. авторегрессионные модели (ARIMAX, GARCH, ARDLM);

3. модель экспоненциального сглаживания;

4. модель по выборке максимального подобия и т.д.

В структурных моделях зависимость будущего значения от прошлого задается в виде некоторой структуры и правил перехода по ней. К ним относятся:

1. нейросетевые модели;

2. модели на базе цепей Маркова;

3. модели на базе классификационно-регрессионных деревьев и т.д»[8].

## 1.2 Описание МРК

Внешний вид устройства показан на рисунке 4. Технические характеристики шасси, её физические характеристики и параметры, приведены в таблице 1.

Таблица 1 – Технические характеристики шасси МРК

|  |  |
| --- | --- |
| **Характеристика** | **Значение** |
| Запас хода, не менее | 20 км |
| Скорость перемещения | 2-12 км/час |
| Максимальный преодолеваемый уклон | 15° |
| Высота преодолеваемой ступени | 14 см |
| Габаритные размеры, Ш x В х Д, не более | 800 х 1000 х 1400 мм |
| Вес шасси, не более | 90 кг |
| Вес полезной нагрузки, до | 10 кг |
| Электрическая мощность | 400-600 Вт |
| Колесная формула | 4х2 |



Рисунок 4 – Внешний вид МРК

Электрическая схема подключения представлена на рисунке 5.

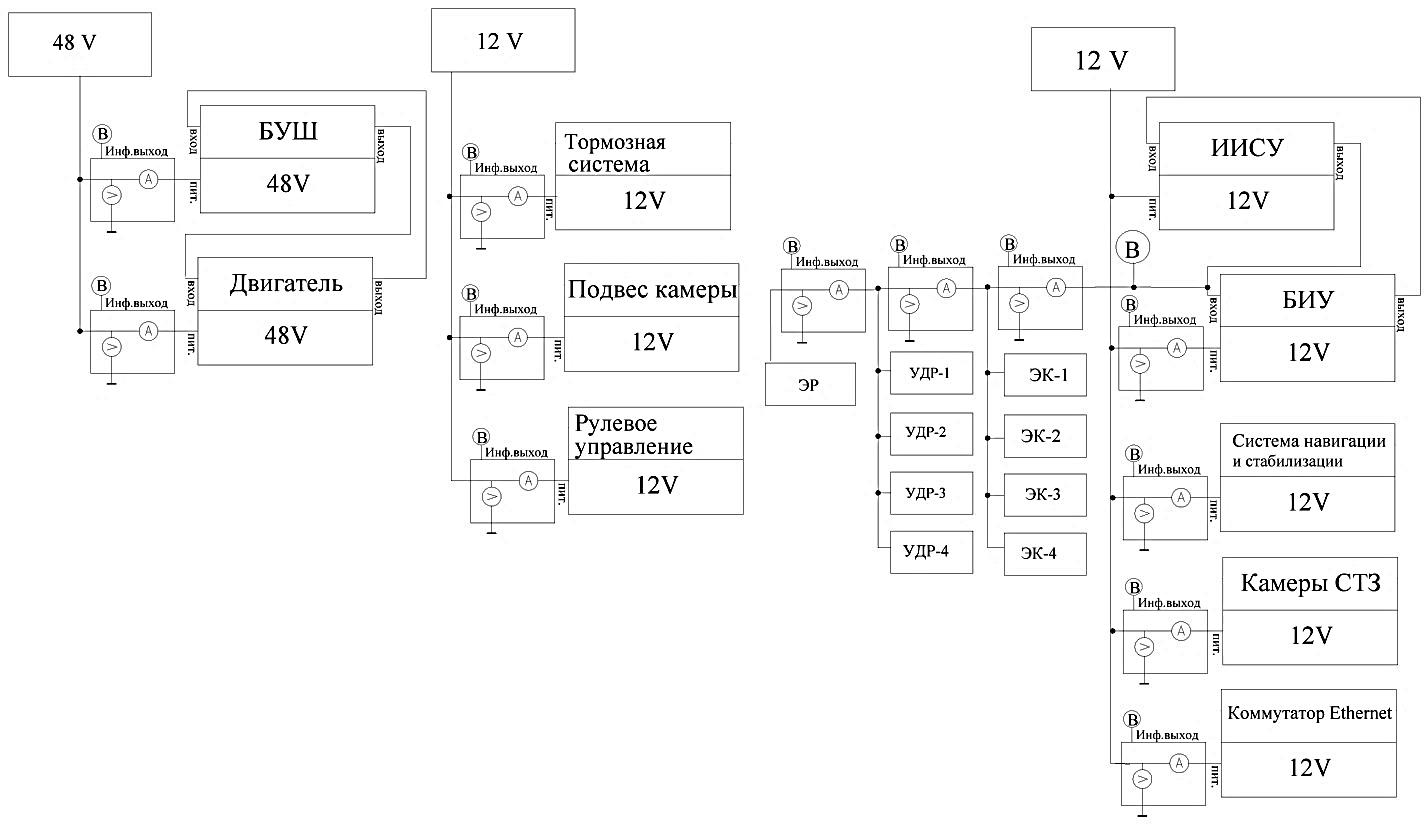


Рисунок 5 – Электрическая схема подключения

Схема рулевой машины представлена на рисунке 6.

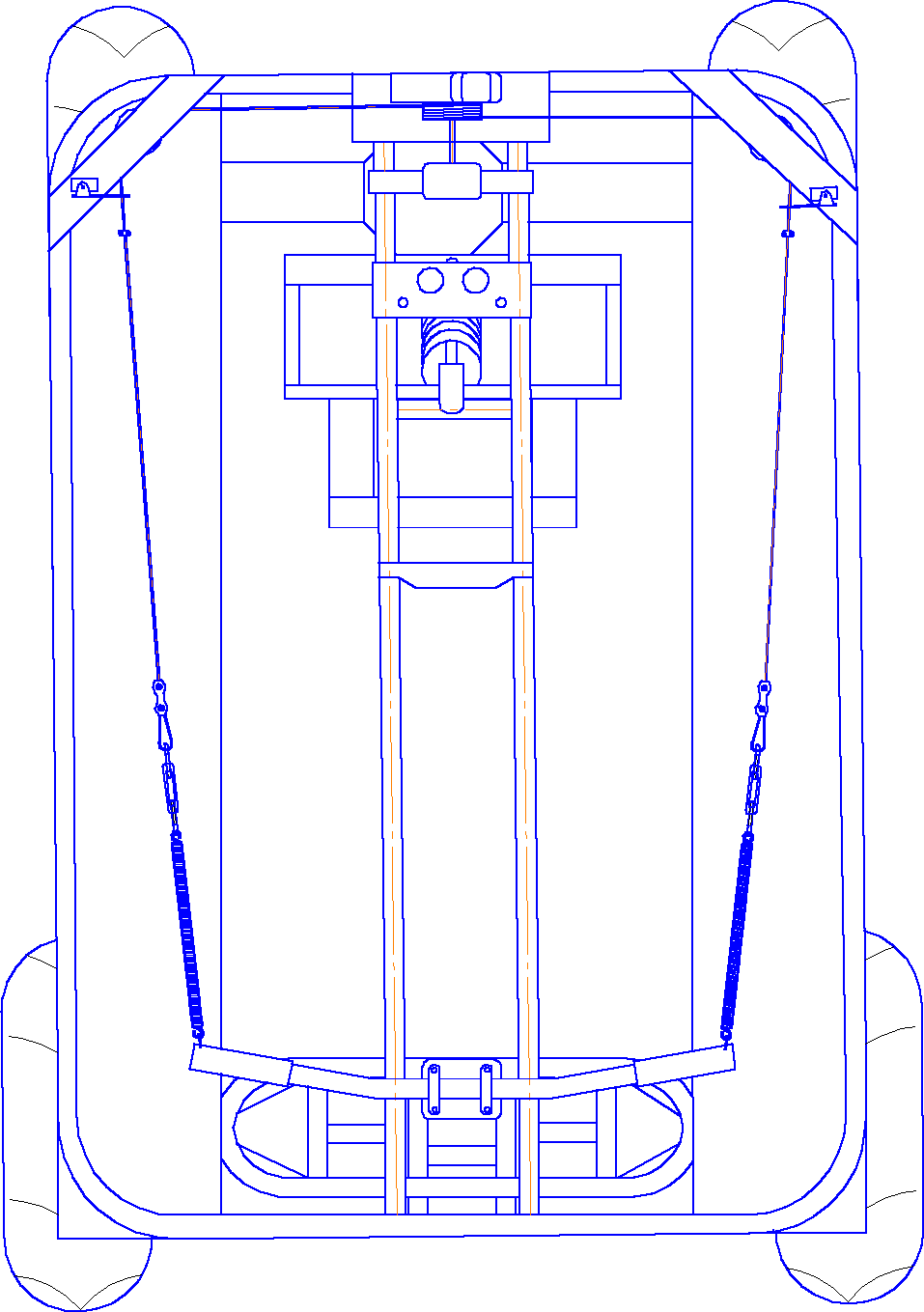


Рисунок 6 – Схема рулевой машины

Условная схема тормозной системы представлено на рисунке 7.

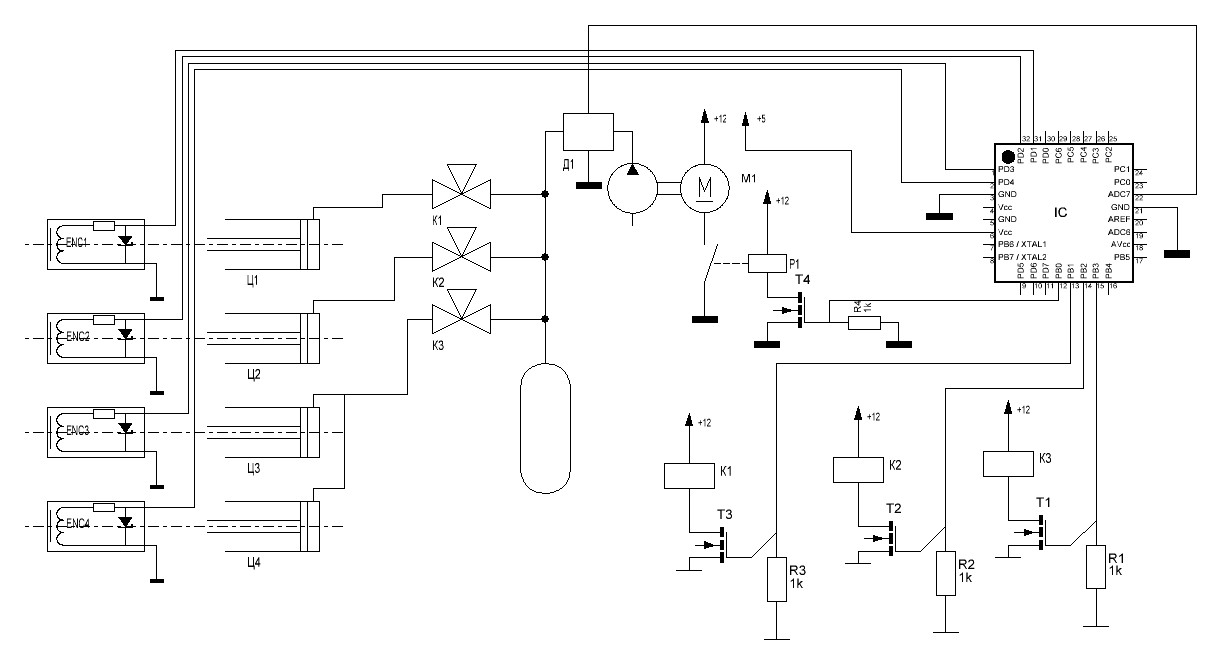


Рисунок 7 – Условная схема тормозной системы

На рисунке 7 условные обозначения следующие: ENC – энкодеры колес, Ц – тормозной цилиндр, К – трёхходовой электромагнитный клапан, М – мотор компрессора, Д – датчик давления, Р – реле управления мотором, IC – микроконтроллер управления ТС.

## 1.3 Постановка задачи построения пути

Разрабатываемый прототип МРК предназначен для эксплуатации в экстремальных условиях внешней среды, в том числе, потенциально опасных для человека. Комплекс предназначен для исследования местности и «подсветки» объектов местности лазерным или иным источником света.

Функционирование МРК происходит на пересеченной местности и заключается в решении поставленных перед ней оператором задач. Задачи МРК представляют собой последовательность точек, которые нужно поочередно посетить с заданной точностью, обеспечив при этом выполнение заданных ограничений на внутренние параметры комплекса и на характеристики движения. Планируемая траектория, в соответствии с которой производится передвижение между указанными в задаче точками, должна быть оптимальной в смысле критерия, заданного оператором МРК. В качестве такого критерия может использоваться либо ожидаемое пройденное аппаратом во время миссии расстояние, либо ожидаемое затраченное время на выполнение миссии.

Местность характеризуется наличием участков различного типа поверхности, различным наклоном относительно горизонтальной плоскости, наличием статических и динамических объектов [9]. Информация о типе, наклоне поверхности и расположении статических объектов известна априори, но может не соответствовать действительности. Распознавание статических и динамических объектов производится во время работы подсистемой технического зрения МРК, после чего результаты распознавания передаются в программу. Процесс планирования траектории должен учитывать наличие объектов и обеспечивать предотвращения столкновения платформы с ними.

Формализацией понятия типа местности в работе является ее непроходимость. Непроходимость должна быть задана для каждой точки поверхности и является интегральным параметром, отражающим характер взаимодействия МРК с местностью на ней. Через непроходимость в систему могут быть переданы сведения о пригодности различных точек поверхности для передвижения платформы. Данный параметр может принимать значения из полуинтервала *P* = [0;∞), причем, предполагается, что меньшим значениям параметра соответствуют более пригодные для передвижения точки.

Пусть в начальный момент времени МРК находится в области A, тогда задача оптимизации будет поставлена следующим образом: найти такую кривую *ab* (рисунок 1), криволинейный интеграл вероятности наличия препятствия по которой будет минимальный. Граничные условия: радиус циркуляции МРК не менее *Rmin*; максимально допустимая вероятность препятствия *Pmax*; координаты конца отрезка должны принадлежать шестиугольнику *B*. На этапе локального планирования использовать допущения, применимые при глобальном планировании, уже нельзя. Так, необходимо учитывать геометрические размеры МРК, различные траектории всех колёс МРК и т.д. Интеграл необходимо брать не только по кривой, но и по некоторой ширине, соответствующей ширине платформы. На рисунке 8 данная область является областью *L*.

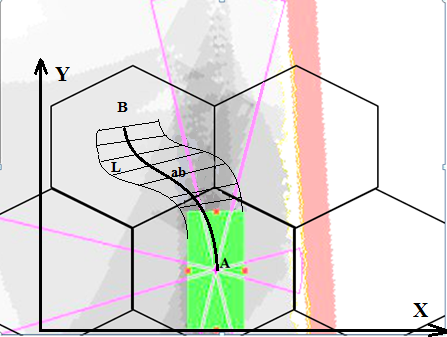


Рисунок 8 – Схема движения МРК по плоскости

Таким образом, задача построения пути сводится к минимизации суммы вероятностей встретить препятствие по пути следования: вдоль траектории движения и по ширине МРК.

## 1.4 Постановка задачи безаварийного движения

Для того чтобы не допустить аварийные ситуации, необходимо осуществлять мониторинг параметров с датчиков. Для того чтобы определить перечень потенциальных угроз корректного функционирования МРК, необходимо разобраться с источниками угроз.

1) Наезд на препятствие. Поскольку препятствия могут быть не только статическими, но и динамическими, имеется реальная угроза возникновения такого рода препятствия на пути следования МРК, не распознанными при передаче с СТЗ и дальномеров.

2) Яма. Поскольку проседание в яму может повлечь за собой поломку частей МРК, необходимо осторожно преодолевать такого рода препятствие. Для начала, определить тип поверхности, по которой происходит движение. При движении по дороге необходимо оценить глубину ямы и в случае допустимой высоты снизить скорость и проехать, иначе – объехать яму. При движении по неасфальтированной дороге яма представляет собой потенциальную угрозу и лучше объехать яму ввиду того, что грунт может просесть и привести к застреванию МРК.

3) Проскальзывание/пробуксовка. Не всегда поверхность, по которой осуществляется движение МРК, идеальна. Поэтому может случиться такое, что платформа просто не сможет ехать по поверхности ввиду плохого сцепления или рыхлости дороги. Так же возможен вариант наезда на препятствие, которое не было определено СТЗ и дальномерами, в этом случае МРК упрется колесами и так же будет осуществляться пробуксовка.

4) Вибрации. По типу и характеру вибраций можно определить состояние работоспособности МРК, возможные неполадки и т.п. Поэтому при превышении или понижении уровня вибраций необходимо вывести сообщение об этом.

5) Обрыв тока. Повреждения, связанные с платформой, могут вывести из строя какие-либо цепи МРК. Отслеживание работоспособности цепей можно осуществить с помощью контроля напряжения. В случае превышения максимального потребления электроэнергии и потребления ниже минимального необходимо вывести сообщение о потенциальном обрыве цепи или коротком замыкании, что, в свою очередь, может помочь избежать перегорания частей цепи.

Методы, с помощью которых удастся избежать опасности, базируются на данных, получаемых с датчиков.

1) Избегание наезда на препятствие. С целью выявления текущей опасности необходимо, чтобы один дальномер не принимал участия в построении матрицы вероятностей встретить препятствие, а выявлял препятствия непосредственно перед МРК. В случае выявления препятствия, необходимо сделать остановку на 10 секунд, чтобы, если препятствие динамическое, оно удалилось, а если препятствие остается, то объехать его. Объезд осуществляется с помощью построения новой траектории движения с новой картой вероятности встретить препятствие.

2) Яма. Для того чтобы не застрять в яме, необходимо иметь данные о параметрах ямы и типе дорожного покрытия. В случае движения по асфальтированной дороге при глубине ямы не более максимально возможного преодолеваемого подъема необходимо снизить скорость и продолжить движение. В случае глубины ямы больше необходимого или же при движении по грунту необходимо объехать яму и продолжить движение. Яма в данном случае расценивается как препятствие и объезжается по схожему алгоритму.

3) Обрыв тока. Для контроля обрыва тока необходимо получать данные с датчиков тока. Отслеживание АЧХ тока и напряжения помогут определить характер аварии, эти данные не должны выходить за установленные рамки. При возникновении обрыва необходимо прекратить движение и сообщить об ошибке.

4) Проскальзывание/пробуксовка. Для выявления этого вида опасности необходимо сравнивать текущее значение оборотов колес с ожидаемым значением, вычисляемым программным путем. В случае расхождения реальных и ожидаемых оборотов колес необходимо либо снизить число оборотов, либо увеличить, в зависимости от разности.

5) Вибрации. Для этого необходимы данные с датчика удара и при превышении или понижении уровня вибраций тут же вывести сообщение о потенциальной неисправности.

Таким образом, задача избегания опасных режимов функционирования МРК сводится к постоянному мониторингу параметров и отслеживанию соответствия поступивших данных допустимым. В случае отклонения от нормы необходимо определить характер несоответствия.

# Выводы по разделу 1

В первом разделе магистерской диссертации произведен обзор литературы на заданную тему, так же приведен обзор технических характеристик МРК, необходимых для построения модели прогнозирования возникновения опасных режимов функционирования МРК при автономном движении по траектории. Так же была выполнена постановка задачи построения пути и постановка задачи безаварийного функционирования МРК с перечислением возможных аварийных ситуаций.

# 2 Теоретическая часть

## 2.1 Планирование пути

Задача планирования пути мобильного робота – задача поиска траектории передвижения робота, проходящей через заданную последовательность точек, характеризующих положение аппарата [9].

Планирование пути ведется в пространстве конфигураций робота, каждая точка (конфигурация) которого однозначно определяет "реальное" состояние робота, и, наоборот, каждому "реальному" состоянию робота соответствует единственная конфигурация. Точки пространства конфигураций представляют собой вектора, компонентами которых могут являться, например, координаты робота и его ориентация в пространстве. В результате планирования пути должна быть сформирована некоторая кривая (траектория) в пространстве конфигураций.

На искомую траекторию могут быть наложены дополнительные ограничения, вызванные физическим смыслом переменных, используемых для определения пространства конфигураций, или иными причинами. Например, положение аппарата в пространстве не может изменяться скачкообразно, поэтому компоненты точек траектории, задающие положение, должны изменяться непрерывно вдоль всего пути [10].

В общем случае может существовать несколько допустимых траекторий движения мобильного робота, и некоторые из них могут быть признаны более предпочтительными, чем другие. Для формализации отношения предпочтительности могут быть использованы различные критерии.

Одним из возможных критериев является ожидаемое время, требуемое на преодоление пути роботом. Такой критерий хорошо сочетается с представлением об эффективном передвижении аппарата, однако существуют сложность на пути к его применению – в ряде случаев (например, при использовании робота в среде со сложным рельефом) неочевиден способ совершения прогноза времени выполнения передвижения вдоль спланированной траектории.

То же относится и к критерию ожидаемого расхода энергии – прогноз расходуемой энергии важен для автономных мобильных роботов, однако для его совершения может быть достаточно сложно получить адекватную модель.

Удобным в использовании, но недостаточно информативным является критерий протяженности спланированного пути. Действительно, такой критерий не позволяет сравнить пути равной длины, проходящие, например, по ровной дороге, покрытой асфальтом, и по бездорожью, что ограничивает применимость этого критерия на практике.

Выбор критерия не ограничивается перечисленными (например, в некоторых работах предлагается максимизация вероятности выживания в условиях поля боя). Для оценки спланированных путей могут также использоваться обобщенные критерии (например, в виде линейной свертки), позволяющие сравнивать пути по нескольким критериям одновременно [11].

## 2.2 Метод оптимизации

Поскольку стоит задача определения оптимального пути, необходимо определить метод оптимизации. Существует большое количество методов, например, минимизация или максимизация, задача максимина или минимакса, метод штрафных функций и другие.

Поиск **минимума** или **максимума** является задачей нахождения глобального экстремума функции. Таким образом, нахождение минимума/максимума находится с учетом значения одной переменной.

Значительная часть алгоритмов условной оптимизации базируется на методах оптимизации безусловной. Как правило, для этого в целевую функцию вводится добавка, характеризующая уровень нарушения ограничений. Эти алгоритмы получили название **методов барьерных штрафных функций**. Поверхность, описываемая штрафной функцией, должна препятствовать выходу траекторий поиска из пространства проектирования.

Первый вариант – увеличение функции происходит при приближении к ограничению изнутри допустимой области, достигая на активном ограничении бесконечной величины. Постоянная штрафная функция, является, по сути, бесконечным барьером. При попытке попадания пробной точки в недопустимую область постоянная штрафная функция принимает «бесконечное» значение (в программных реализациях для этого некоторой переменной присваивается соответствующий признак), после чего принимается решение о том, как действовать дальше.

Недостатком такого алгоритма является невозможность «участия» недопустимых точек в последующем анализе и, соответственно, склонность алгоритмов к зацикливанию (это формальное описание, реальная ситуация более сложна), если минимум лежит на границе.

Второй вариант – внутри и на границе допустимой области «добавка» равна нулю, а затем, при выходе за границу допустимой зоны, она начинает возрастать. По сути, данный вид штрафа – это сумма абсолютных величин невязок нарушенных ограничений. Тем самым на каждое решение, попадающее в «запрещенную область», налагается «штраф».

Таким образом, получается формула (1):

|  |  |
| --- | --- |
|  | (1) |

где *F*(*X*) – исходное значение целевой функции,

*W*(*X*,*p*) – целевая функция с учетом штрафа,

*p* – параметр штрафа,

*ci*(*X*) – вектор нарушенных в точке *X* ограничений,

*I* – число нарушенных ограничений.

Теоретически параметр штрафа должен зависеть от поведения исходной целевой функции. Если, например, она «быстро» убывает в точке на линии ограничения, то «малый» штраф не сможет компенсировать это убывание, и программа продолжит поиск за пределами допустимой области. Если же штраф «слишком» большой, то возникают проблемы, присущие абсолютным штрафам. Поэтому в ситуациях, когда обнаруженный программой условный оптимум лежит на одном или нескольких ограничениях, возможен «небольшой» выход за границы допустимой области.

При огромном количестве частных критериев из-за сложных взаимосвязей в некоторых случаях нелегко добиться приведенных выше соотношений. В этом случае применяют принцип **максимина**, заключающийся в такой вариации значений проектных параметров X, при которой последовательно повышаются те нормированные критерии, численные значения которых в исходном решении оказались наименьшими. Завышение одного критерия неизбежно приводит к снижению значений части остальных критериев. В тоже время проведением ряда операций можно добиться определенной степени уравнивания противоречивых (конфликтных) частных критериев, что и является целью принципа максимина.

Принцип максимина математически формулируется следующим образом: нужно выбрать такой вектор проектных параметров *X*(0) *X* на котором реализуется максимум из минимальных значений частных критериев, т.е. (2):

|  |  |
| --- | --- |
| *F*(*X*(0))= *maxX(0) min*{*fi*(*X*)}*, i*=1*,… n, X*={*x*1*,*..., *xn*} | (2) |

Такой принцип выбора X часто называют принципом «гарантированного результата».

В том случае, если частные критерии fi(X) следует минимизировать, то самым «отстающим» критерием является тот, который принимает максимальное значение. Тогда принцип равномерной компенсации формулируется в виде минимаксной задачи (3)[10]:

|  |  |
| --- | --- |
| *F*(*X*(0)) = *minXi max*{*fi*(*X*)}*, i*=1*,…,n, X*={*x*1*,..., xn*} | (3) |

## 2.3 Определение функции поверхности

Поскольку карта приходит в формате матрицы, а траектория может проходить через точки, которые не заданы в полученной карте, необходимо каким-либо образом определить поверхность, в которой в каждой точке будет находиться значение.

Определение может быть следующими методами: аппроксимации полиномом Лагранжа, среднее арифметическое, аппроксимация полиномом Ньютона.

Вкратце о каждом методе будет рассмотрено ниже. Рисунок исходной карты представлен на рисунке 9. Представление точек в пространстве представлено на рисунке 10.

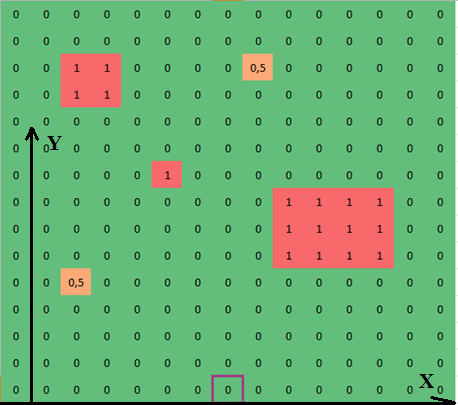


Рисунок 9 – Условная карта вероятностей

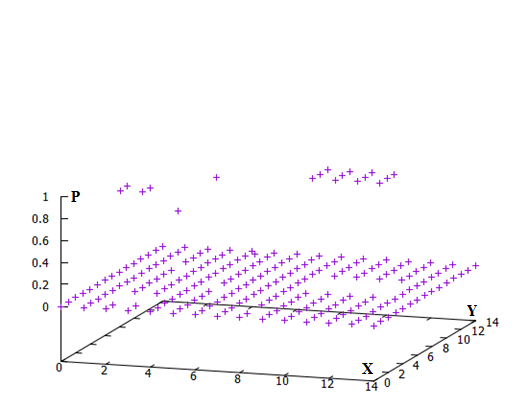


Рисунок 10 – График вероятностей в пространстве

### 2.3.1 Среднее арифметическое

Среднее арифметическое заключается в том, что определяется промежуточное значение между двумя точками по формуле (4):

|  |  |
| --- | --- |
|  | (4) |

Таким образом, найдется значение в средней точке, которое может быть приближенно соответствующим значению необходимой точки.

Недостатком метода является неточность и большая погрешность. Плюс данного метода: простота исполнения.

### 2.3.2 Полином Ньютона

Аппроксимация по формулам Ньютона используется, если интерполируемая функция *f*(*x*) задана в (*n+1*) равноотстоящих узлах, т.е*. xi+*1*-xi=Δxi=h=const*. Общий вид полинома Ньютона (5):

|  |  |
| --- | --- |
| *Pn*(*x*) *= a*0*+a*1(*x-x*0) *+ a*2(*x-x*0)(*x-x*1)*+…+an*(*x-x*0)(*x-x*1)*\*…\**(*x-xn-*1) | (5) |

Из условия интерполяции:

*yi=f*(*xi*)*=Pn(xi),* *i*= следует определение коэффициентов полинома Ньютона через конечные разности (6):

|  |  |
| --- | --- |
| *ai=*, *i*=0,1,2,…*n.* | (6) |

В общем случае (для всех значений функции, определенных в узлах) конечные разности *k*-го порядка имеют вид (7):

|  |  |
| --- | --- |
| Δ*kyi=* Δ*k-1yi+1-* Δ*k-1yi* | (7) |

Определим конечные разности:

Δy0=y1-y0,

Δy1=y2-y1,

…

Δy*i*=y*i+1*-y*i*,

Δ2y0=y1-y0,

Δ2y1=y2-y1,

…

Δ2y*i*=y*i+1*-y*i*.

Найденные конечные разности записываются в виде таблицы:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *x* | *y* | Δ*y* | Δ2*y* | Δ3*y* |
| *x*0 | *y*0 | Δ*y*0 | Δ2*y*0 | Δ3*y*0 |
| *x*1 | *y*1 | Δ*y*1 | Δ2*y*1 | Δ3*y*1 |
| *x*2 | *y*2 | Δ*y*2 | Δ2*y*2 | Δ3*y*2 |

Первая формула Ньютона имеет вид (8):

|  |  |
| --- | --- |
| *Pn*(*x*)=*y0+q* Δ*y*0+ Δ2*y0*+…+ Δn*y0* | (8) |

*q=*, где *x0* – ближайший к точке *х* узел слева.

Если таблица значений функции конечна, то наивысший порядок интерполяционного многочлена равен *n*, где (*n*+1) – количество узлов интерполяции. Если длина таблицы задания функции не ограничена, то порядок интерполирования *n* определяется тем, что каждая конечная разность или их сумма не должна превышать некоторой постоянной величины *Е* (9):

|  |  |
| --- | --- |
|  | (9) |

Эта формула применяется для интерполяции в начале таблицы («интерполяция вперед»), т.к. в этом случае имеется возможность, взяв большее число узлов, увеличить точность интерполирования.

Для интерполяции в конце таблицы («интерполяция назад») используется вторая интерполяционная формула Ньютона (10):

|  |  |
| --- | --- |
| *Pn*(*x*)=*yn+q* Δyn-1+ Δ2*yn-2*+…+ Δn*y0* | (10) |

*q=*, где *xn* – ближайший к точке *х* узел справа.

Оценки погрешностей формул имеют соответственно вид(11), (12):

|  |  |
| --- | --- |
| Rn(x) | (11) |
| Rn(x) | (12) |

Выводы:

1. В формулах Ньютона в случае добавления узла все найденные члены сохраняются и появляется новое слагаемое, представляющее собой не что иное, как поправку к уже вычисленному значению.

2. При интерполяции на малых участках слагаемые в формулах будут расположены в порядке их малости, что облегчает использование формул Ньютона в вычислениях и позволяет судить о точности интерполяции.

3. Степень интерполирующего полинома существенно зависит от шага таблицы (чем меньше шаг, тем график функции более приближен к линейному, что позволяет использовать линейную интерполяцию)

4. Ограниченность применения формул Ньютона связанна с их пригодностью лишь для равноотстоящих узлов.

### 2.3.3 Полином Лагранжа

Интерполяционный полином Лагранжа — многочлен минимальной степени, принимающий данные значения в данном наборе точек. Для n+1 пар чисел (*x0,y0),*(*x1,y1)…* (*xn,yn),*  где все *xj* различны, существует единственный многочлен *L(x)* степени не более *n*, для которого *L(xj)=yj*.

Интерполяционный полином Лагранжа имеет вид(13):

|  |  |
| --- | --- |
|  | (13) |

где *li(x)* – базисные полиномы, определяющиеся по формуле (14):

|  |  |
| --- | --- |
|  | (14) |

*li*(*x*) обладают следующими свойствами:

- являются многочленами степени *n*;

- *li*(*xi*) =1;

- *li(xj)* =0 при *j≠i*.

Отсюда следует, что *L(x)*, как линейная комбинация *li(x)*, может иметь степень не больше *n*, и *L(xi)=yi*.

Оценку погрешности определяют по формуле (15):

|  |  |
| --- | --- |
| |*f(x)-Ln(x)|≤|Ln+1(x)-Ln(x)*| | (15) |

Выводы:

1. Формула Лагранжа может быть использована для произвольно заданных узлов интерполяции.

2. Добавление к уже взятым узлам интерполяции хотя бы одного нового, влечет за собой не только добавление нового базисного полинома, но и необходимость пересчета уже посчитанных базисных полиномов.

## 2.4 Метод определения траектории

Поскольку у МРК имеется множество радиусов циркуляции, обозначим как *Rc*, определение допустимых траекторий может определяться методом перебора, так как число *Rc* не является очень большим. Мощность множества = 41. Значения варьируются от -20 до +20.

Минимальный радиус кривизны равен 3 метрам. При построении кривой необходимо учитывать условие минимального радиуса кривизны, поскольку кривая может иметь радиус менее возможного радиуса поворота и реальная траектория будет отличаться от заданной.

Необходимо помнить, что число траекторий, ведущих в конечную точку или допустимую область, которая задается некоторым числом, ограничено. Поэтому условно разделенное значение *Rc* еще сократится. Именно поэтому метод перебора будет эффективен и прост в реализации.

## 2.5 Вычисление траектории

Для построения траектории наиболее оптимальными могут быть траектории с использованием кривой Безье или же с использованием дуги. Для выбора лучшей траектории необходимо рассмотреть плюсы и минусы обеих траекторий.

Создание траектории – кривой Безье – осуществляется путем построения кривой по начальной и конечной точке, а так же двум векторам направления движения МРК. Формула данной кривой получается следующая (16):

|  |  |
| --- | --- |
| *E* = (1−*t*)3*E*1 + 3(1−*t*)2*tE*2 +3(1−*t*)*t*2*E*3 + t3*E*4 | (16) |

Вместо *E*i нужно подставить координаты *i*-й опорной точки (*xi*, *yi*).

Эти уравнения векторные, то есть для каждой из координат:

*x* = (1−*t*)3*x*1 + 3(1−*t*)2*tx*2 + 3(1− *t*)*t*2*x*3 + *t*2*x*4

*y* = (1−*t*)3*y*1 + 3(1−*t*)2*ty*2 + 3(1− *t*)*t*2*y*3 + *t*2*y*4

Вместо *x*1, *y*1, *x*2, *y*2, *x*3, *y*3, *x*4, *y*4  подставляются координаты трёх опорных точек, и в то время как *t* пробегает множество от 0 до 1, соответствующие значения (*x*, *y*) как раз и образуют кривую. Точки *E*1 и *Е*4 – опорные точки, точки *Е*2 и *Е*3 – управляющие точки.

Так же следует отметить, что направление вектора движения в конечной точке определяется из глобального планирования, поэтому построение вектора в данной подзадаче не рассматривается.

Создание траектории – дуги движения – осуществляется путем построения окружности по вектору, радиусу и конечной точке. Формула данной окружности получается следующая (17):

|  |  |
| --- | --- |
|  | (17) |

Поскольку следует отметить, что направление вектора всегда вперед, поскольку это является одним из критериев, поэтому в формуле вектор не учитывается.

Плюсом построения пути с использованием дуги является простота реализации. Однако, минусов у данного варианта существенно больше:

- пунктом прибытия является некоторая допустимая область, а не конечная точка;

- ввиду движения только по дуге минимизация интеграла встретить препятствие на пути не столь оптимальна, как при использовании кривой Безье;

- исключается возможность маневрирования по поверхности;

- при использовании кривой Безье так же можно получить движение по дуге, но с точным прибытием в нужную конечную точку.

Основываясь на плюсах и минусах методов, наиболее оптимальным выбором будет движение по траектории с использованием кривой Безье.

## 2.6 Расчёт веса траекторий

Траекторию движения необходимо описывать криволинейным интегралом по поверхности. Как было определено ранее, поверхность описывается полиномом Лагранжа.

Функция поверхности: *Pn*(*x*, *y*) = *p*. Где *p* – вероятность встретить препятствие в координатах *x*, *y*.

Интеграл по полиному находится по формуле (18):

|  |  |
| --- | --- |
|  | (18) |

где *l*0 – начало кривой Безье, *lk* – конец кривой Безье. Необходимо помнить, что кривая является траекторией, которая ищется в соответствии с условиями пункта 2.4.

## 2.7 Выбор оптимальной траектории

Поскольку ставилась задача построения оптимального пути с минимальным интегралом вероятностей по поверхности, можно определить, как будет происходить выбор оптимальной траектории. Так как ранее рассматривалась задача максимизации, то необходимо применять именно этот вариант, так как имеется один критерий. Таким образом, задача сводится к минимизации отсутствия препятствия.

## 2.8 Прогнозирование возникновения опасных режимов функционирования МРК

Список опасных режимов функционирования был приведен в п.1.4. Методы, с помощью которых удастся избежать опасности, базируются на данных, получаемых с датчиков.

Модель предсказания режимов работы на основе нейросети: на вход подаётся АЧХ, далее происходит аппроксимация и предсказание следующего значения, тем самым, предотвращая появление опасного режима функционирования. Пусть АЧХ имеет вид (рисунок 11):

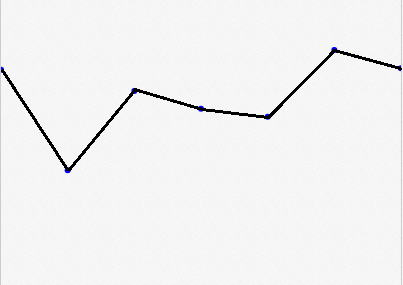


Рисунок 11 – График АЧХ

Аппроксимация данного графика АЧХ с использованием нейросети выглядит следующим образом (рисунок 12):

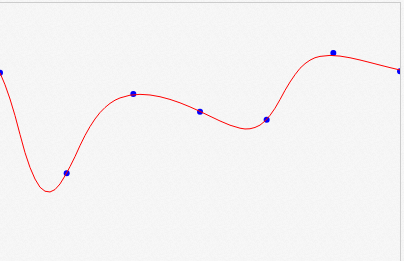


Рисунок 12 – Аппроксимация графика АЧХ

Нейронная сеть, выполняющая аппроксимацию, представлена на рисунке 13:

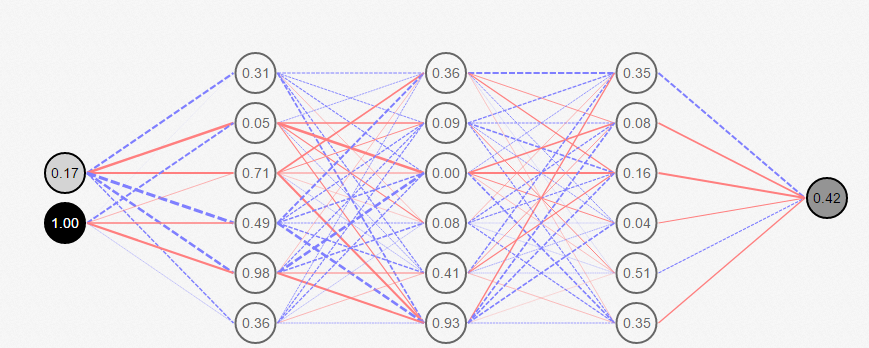


Рисунок 13 – Нейросеть, выполняющая аппроксимацию

На вход данной нейросети подаётся АЧХ некоторого процесса, на выходе получается функция, согласно которой и можно осуществлять прогнозирование возникновения опасных режимов функционирования МРК.

Далее необходимо осуществить распознавание прогнозируемого опасного режима функционирования. Для этого на вход нейросети подаются коэффициенты АЧХ, на выходе получаем значения вероятностей каждой из ситуаций опасных режимов функционирования. Количество нейронов скрытого слоя определяется по формуле (19):

|  |  |
| --- | --- |
|  | (19) |

где k – количество нейронов, n – количество входов, m – количество выходов. В данном случае, количество входов соответствует количеству коэффициентов АЧХ, то есть, 10. Количество выходов соответствует количеству ситуаций опасных режимов функционирования МРК, то есть, 5. Итоговое количество нейронов скрытого слоя . Изображение нейросети с некоторыми коэффициентами АЧХ и вероятностями опасных режимов функционирования представлено на рисунке 14.

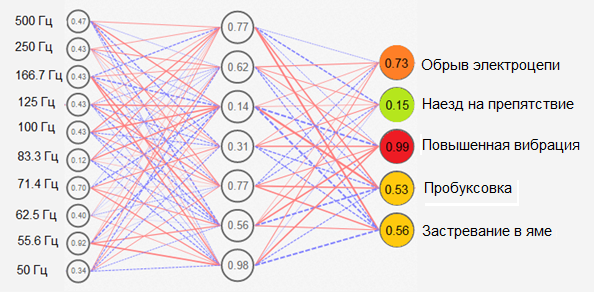


Рисунок 14 – Нейросеть, определяющая прогнозирование вероятностей возникновения опасных режимов функционирования

После определения вероятности возникновения той или иной ситуации опасного режима функционирования необходимо предпринять действия для предотвращения аварии. Действия, предпринимаемые в случае возникновения опасного режима функционирования:

1) Избегание наезда на препятствие. С целью выявления текущей опасности необходимо, чтобы один дальномер не принимал участия в построении матрицы вероятностей встретить препятствие, а выявлял препятствия непосредственно перед МРК. В случае выявления препятствия, необходимо сделать остановку на 10 секунд, чтобы, если препятствие динамическое, оно удалилось, а если препятствие остается, то объехать его. Объезд осуществляется с помощью построения новой траектории движения с новой картой вероятности встретить препятствие и функцией поверхности.

2) Яма. Для того чтобы не застрять в яме, необходимо иметь данные о параметрах ямы и типе дорожного покрытия. В случае движения по асфальтированной дороге при глубине ямы не более максимально возможного преодолеваемого подъема необходимо снизить скорость и продолжить движение. В случае глубины ямы больше необходимого или же при движении по грунту необходимо объехать яму и продолжить движение. Яма в данном случае расценивается как препятствие и объезжается по схожему алгоритму.

3) Обрыв тока. Для контроля обрыва тока необходимо получать данные с датчиков тока. Отслеживание АЧХ тока и напряжения помогут определить характер аварии, эти данные не должны выходить за установленные рамки. При возникновении обрыва необходимо прекратить движение и сообщить об ошибке.

4) Проскальзывание/пробуксовка. Для выявления этого вида опасности необходимо сравнивать текущее значение оборотов колес с ожидаемым значением, вычисляемым программным путем. В случае расхождения реальных и ожидаемых оборотов колес необходимо либо снизить число оборотов, либо увеличить, в зависимости от разности.

5) Вибрации. Для этого необходимы данные с датчика удара и при превышении или понижении уровня вибраций тут же вывести сообщение о потенциальной неисправности.

## 2.9 Определение характерных значений измеряемых показателей

### 2.9.1 Определение общих значений

Высота преодолеваемого препятствия и угла, на который можно осуществить подъем, определены в таблице 1. Угол = 15°, высота = 14 см. На рисунке 15 представлены преодолеваемые МРК препятствия.

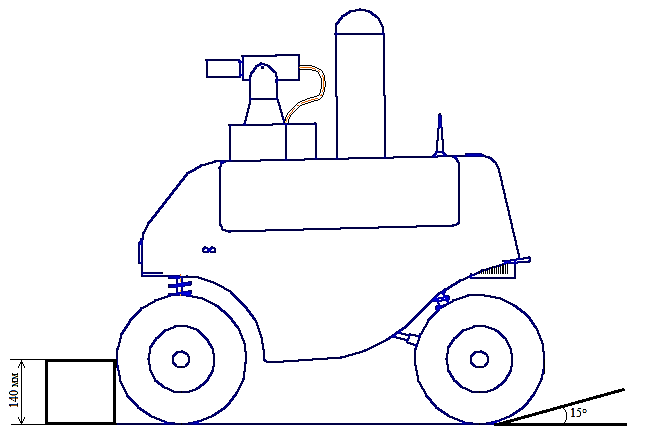


Рисунок 15 – Преодолеваемые МРК препятствия

Для расчета данных электрических цепей необходимо провести замерения каждой из них. Всего в МРК расположено три электрических цепи. Первые две цепи питают шасси, тормоза и привод – 48V, рулевое управление – 12V. Все датчики и электроника подключены к третьей электрической цепи питания 12V.

При повышенном энергопотреблении необходимо сообщать о возможных неисправностях. Данные о нижнем и верхнем уровне энергопотребления получить либо из документации, либо экспериментальным путём.

### 2.9.2 Значения энергопотребления

Экспериментально было установлено, при каких значениях диагностируются три состояния: обрыв питания, нормальная работа и перегрузка. Для этого вычисляется АЧХ тока и АЧХ напряжения. Для этого были проведены измерения значений тока и напряжения для этих состояний. Так же было проведено измерение АЧХ для 10 частот из интервала (0-1000) Гц с выводом двух типов данных: максимального и минимального значений тока или напряжения и разницы между максимальным и минимальным значением тока или напряжения. На рисунке 16 представлены графики измерений тока для трех состояний. На рисунке 17 – измерений напряжения.

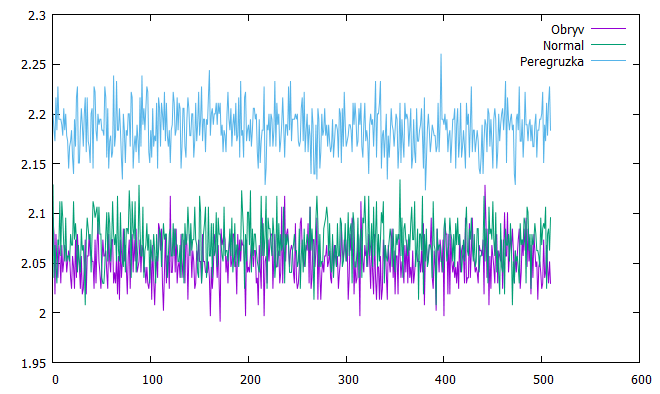


Рисунок 16 – Графики потребления тока при трех режимах работы

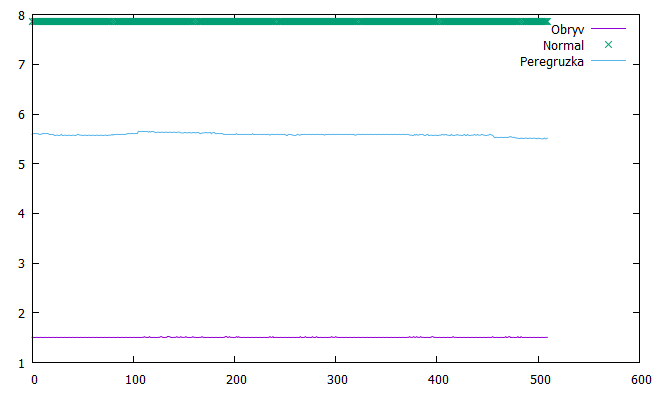


Рисунок 17 – Графики напряжения при трех режимах работы

На рисунках 18 – 19 представлены АЧХ тока и напряжения при трех режимах.

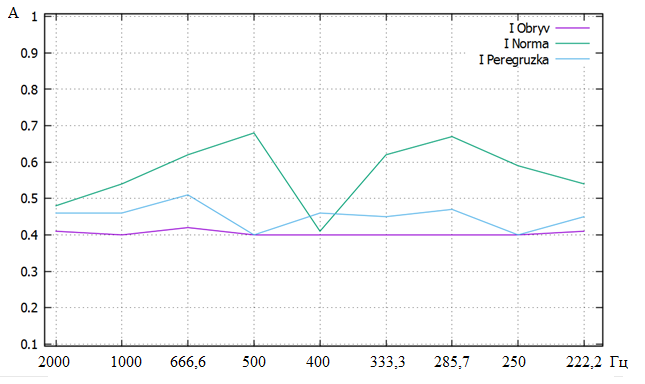


Рисунок 18 – АЧХ тока при трех режимах

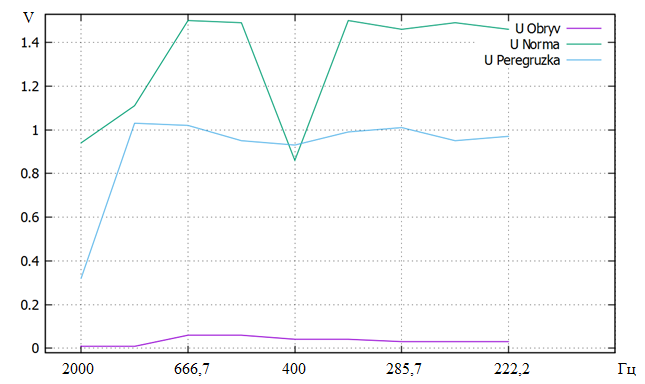


Рисунок 19 – АЧХ напряжения при трёх режимах

Сверяясь с данными графиками во время работы МРК, можно отметить повышенные и пониженные режимы работ, тем самым определив неисправность.

Для получения перечня всевозможных аварийных ситуаций можно воспользоваться таблицей с перечнем данных по параметрам, характерным при работе рассматриваемой подсистемы. Рассматриваемые подсистемы: тормозная система, рулевое управление и подвес стереокамеры.

### 2.9.3 Определение значений параметров рулевого управления

При работе рулевого управления перечень параметров, по которым можно идентифицировать аварийную работу, это:

1. Данные датчика поворота колес.

2. Потребление электрического тока.

3. Наличие вибрационных явлений.

4. Курс движения.

5. Срабатывание концевого датчика.

Пункты 4 и 5 являются необязательными, поскольку имеют значение в небольшом перечне аварийных ситуаций, поэтому основными являются пункты 1-3. В таблице 2 приведен перечень возможных значений по каждому из пунктов. Каждая аварийная ситуация будет характеризоваться строкой из пяти цифр, заключенных в квадратные скобки, каждое из чисел которого определяет значение параметра: [P1, P2, P3, P4, P5]

Таблица 2 – Принимаемые значения параметров рулевого управления

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № п/п | Возможные значения | | | |
| 1 | 1 | 0 |  |  |
| 2 | -1 | 0 | +1 |  |
| 3 | -1 | 0 | +1 |  |
| 4 | -1 | 0 | +1 |  |
| 5 | 1 | 2 | 0 | -1 |

В таблице 2 обозначения -1, 0 и +1 означают меньше нормы, норма и больше нормы соответственно. В параметре 5 1 и -1 – срабатывание правого и левого датчиков соответственно, 0 – отсутствие срабатываний обоих датчиков, 2 – срабатывание двух концевых датчиков.

При комбинации [1, -1, -1, 0, 0] определяется соскок рулевого механизма, в этом случае необходимо прекратить движение и сообщить о поломке. При комбинации [1, +1, 0/+1, +1, 0] определяется неполадка концевого датчика, в этом случае необходимо остановиться и повернуть колеса в крайние положения 3 раза, в случае повторения этой комбинации как минимум 2 из 3 раз сообщить о поломке датчика. При комбинации [1, 0, 0, -1, \*], где \* - любое из возможных значений, принимаемых параметром, можно определить отсутствие возможности поворота колес, тем самым установить нахождение МРК в колее, решением в данном случае будет изменение маршрута движения. При комбинации [1, +1, +1, -1, \*] можно определить то, что заклинило поворотный механизм, в этом случае необходимо отъехать назад и изменить направление, в случае повторения данной комбинации – прекратить движение и сообщить о поломке.

### 2.9.4 Определение значений параметров тормозной системы

При определении аварийных ситуаций тормозной системы можно опираться на следующие параметры:

1. При включении питания подача тока на компрессор.

2. Изменение скорости при торможении.

Оба параметра характеризуются значениями +1, 0, -1, т.е., больше нормы, норма и меньше нормы соответственно.

При комбинации [+1, 0] можно заметить потребление тока выше нормы и диагностировать неполадки с токопотреблением и сообщить об ошибке. При комбинации [0, +1] можно диагностировать слишком большое давление и сообщить об этом. При комбинации [0, -1] можно установить недостаточное давление и так же сообщить об этом. При комбинации [-1, -1] можно определить недостаточное питание, идущее на компрессор, в этом случае необходимо снизить скорость до минимально возможной и осуществлять движение, избегая резких спусков, а так же сообщить о проблеме.

### 2.9.5 Определение значений параметров подвеса камеры

Поскольку невозможно осуществить построение траектории движения МРК без данных, полученных от СТЗ, очень важно вовремя установить поломку подвеса камеры. В случае отсутствия поворота стереопары необходимо определить, в каком направлении остановилась камера. Если направление камеры вперед, то можно продолжить движение, сообщив о поломке. Если же камера направлена не вперед, то необходимо прекратить движение, сообщив о поломке, поскольку в этом случае движение чревато поломкой МРК.

### 2.9.6 Определение неисправностей на основании вибраций

Ввиду того, что характер и частота вибраций могут сообщить о наличии каких-либо неисправностей или же являться признаком корректной работы, мониторинг уровня вибраций может спрогнозировать или диагностировать аварийный режим функционирования МРК. У элементов МРК имеются собственные вибрации, а так же вибрации, создаваемые при движении по поверхности, таким образом, необходимо определить значения, при которых будет диагностироваться состояние покоя, работы или пробуксовки.

Графики вибраций при трёх состояниях показаны на рисунках 20 – 22. На графиках показаны изменения координат по осям X, Y, Z.

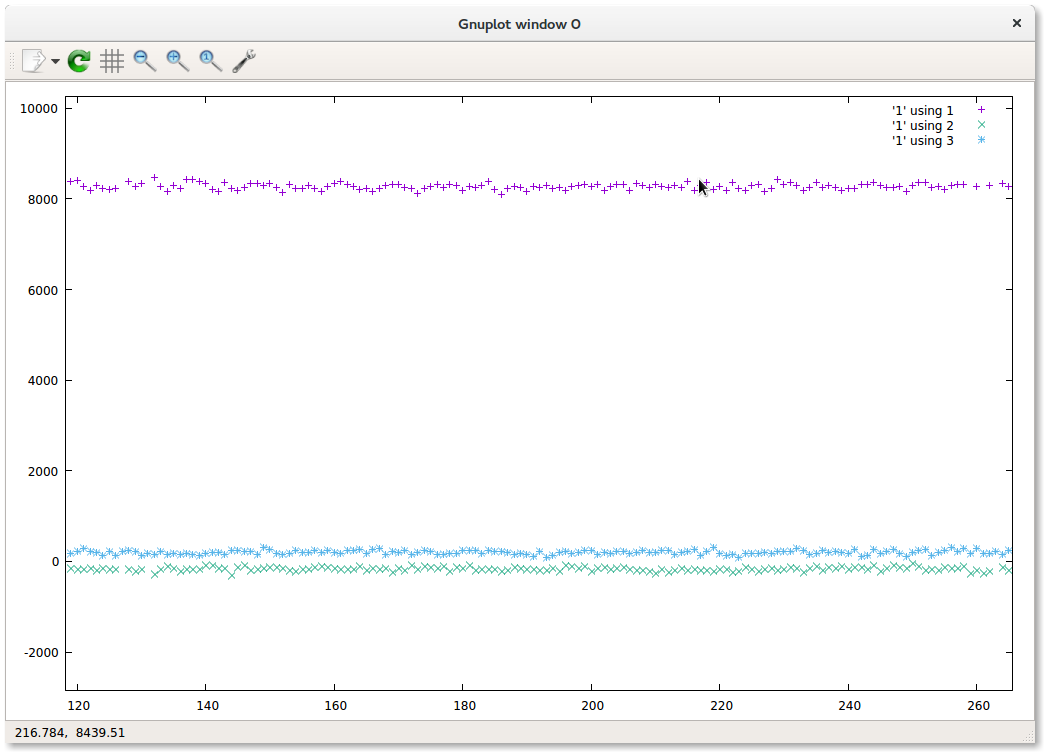


Рисунок 20 – Вибрации в состоянии покоя

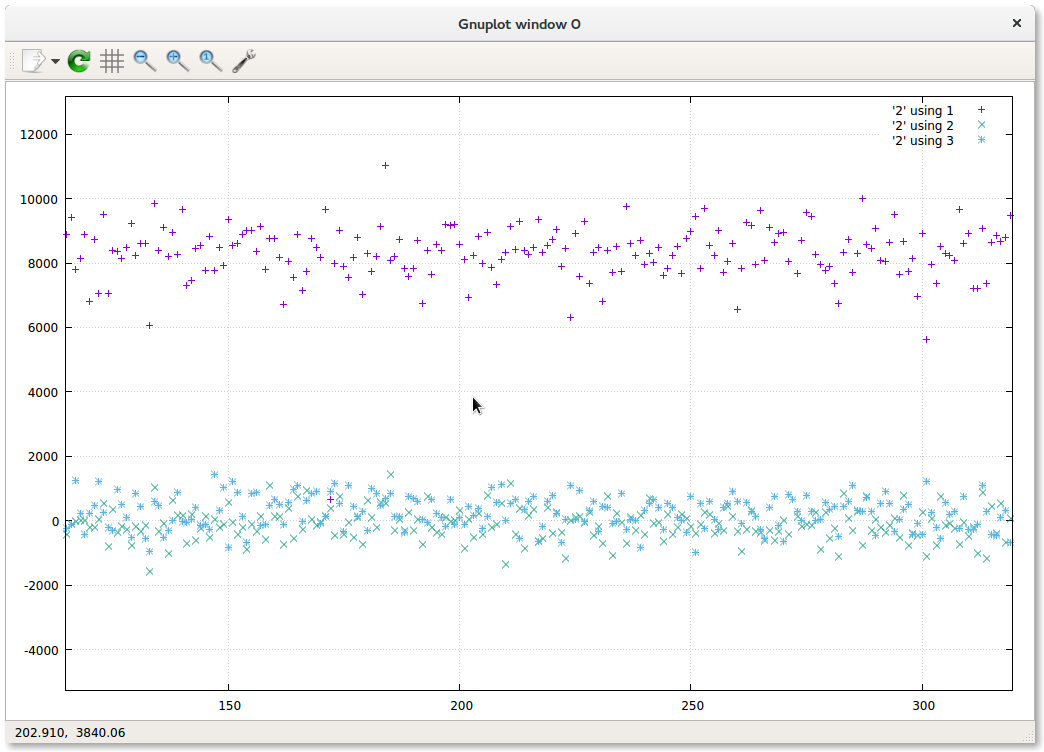


Рисунок 21 – Вибрации в состоянии нормальной работы



Рисунок 22 – Вибрации при пробуксовке

Для определения АЧХ вибраций необходимо произвести разделение по частотам и вычислить амплитуду. Частоту 500Гц необходимо разделить на 9 частей и произвести вычисление АЧХ. На рисунках 23-25 представлены АЧХ вибраций для трех состояний: состояние покоя, нормальной работы и пробуксовки.

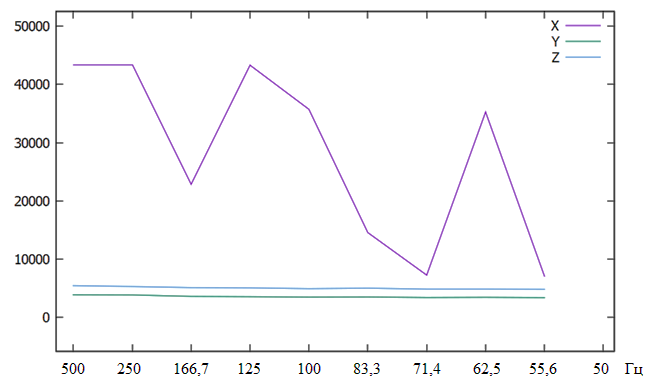


Рисунок 23 – АЧХ режима пробуксовки

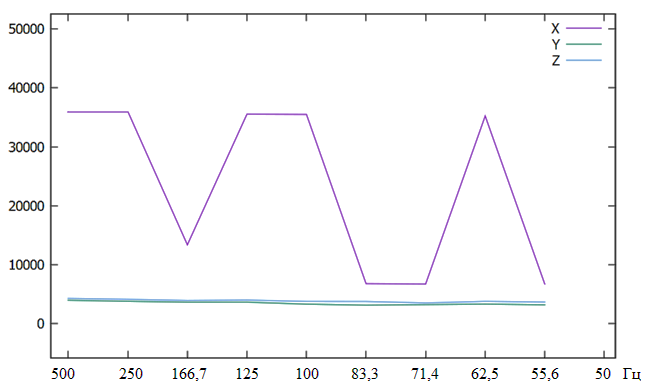


Рисунок 24 – АЧХ состояния нормальной работы

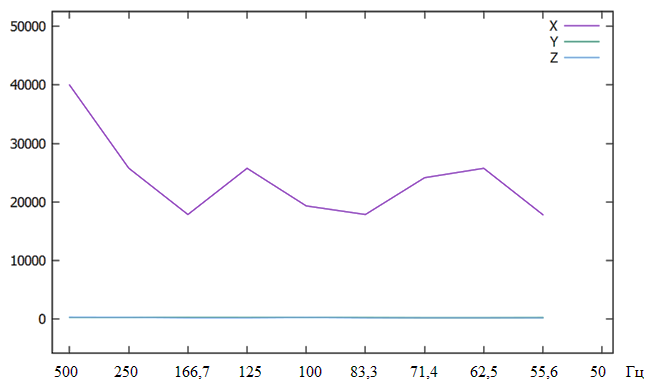


Рисунок 25 – АЧХ состояния покоя

Полученные АЧХ вибраций в дальнейшем будут считаться эталонными значениями при определении состояния платформы.

# Выводы по разделу 2

Во втором разделе магистерской диссертации произведен выбор метода оптимизации, так же дано математическое описание построения траектории. Так же был определен метод получения функции поверхности. Экспериментальным путём были получены данные, характерные для различных режимов функционирования МРК, на основании этих данных были сформированы методы определения неисправностей.

# 3 Практическая часть

Для того чтобы определить наиболее оптимальный путь, необходимо следовать алгоритму:

1. Определить точки начала и конца траектории;

2. Определить дискретность угла поворота колёс;

3. Построить допустимые траектории движения в виде кривых Безье;

4. Взять криволинейный интеграл по каждой из траекторий, где подынтегральным выражением будет выступать вероятность встретить препятствие;

5. С учетом ширины платформы проинтегрировать соответствующую прилежащую поверхность *L*;

6. Полученное значение сравнить с максимально допустимым, в случае превышения этого значения сообщить о том, что локальный путь не может быть построен;

7. Если на области *L* имеется точка, которая резко превышает среднее значение точек, то необходимо сообщить об ошибке и выбрать другой глобальный маршрут.

## 3.1 Алгоритм поиска локального пути

Алгоритм поиска оптимальной траектории движения МРК по поверхности представлен на рисунке 26.

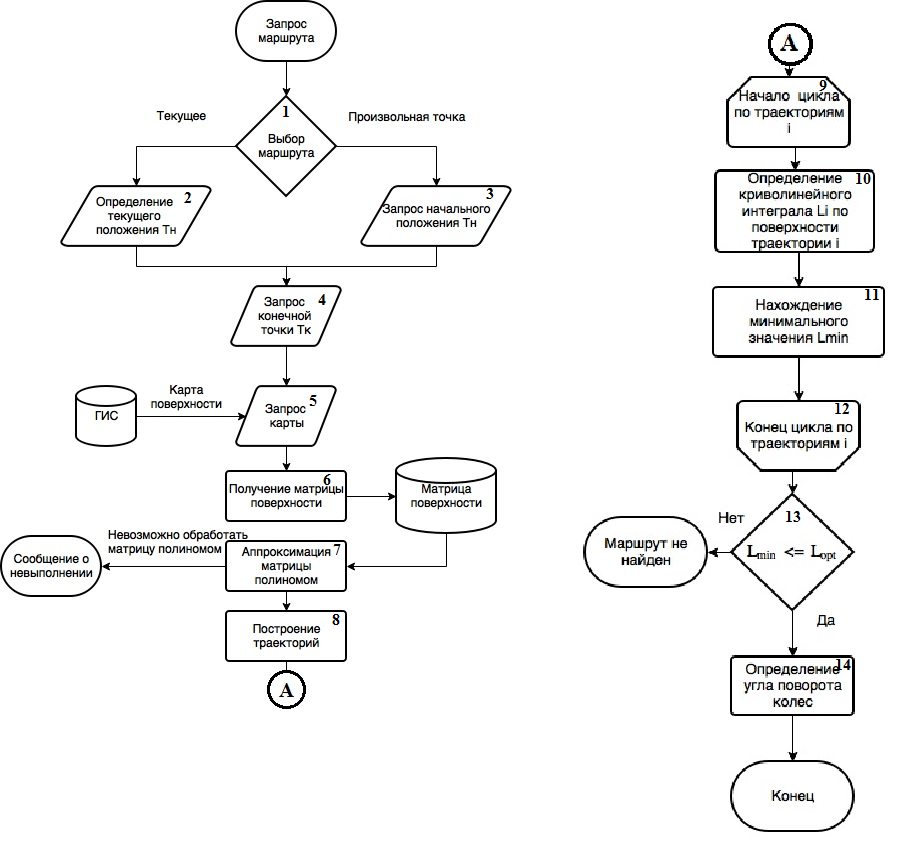


Рисунок 26 – Блок-схема алгоритма нахождения оптимального пути

1, 2, 3. Определение местоположения.

Входные данные: данные о местоположении.

Выходные данные: определение текущего местоположения, определение переменной *Тн*.

Действия: Запись в переменную *Тн* переданных координат местоположения.

4. Запрос конечной точки.

Входные данные: Ввод конечной точки.

Выходные данные: сохраненные координаты конечной точек.

Действия: определение соответствия координат конечной точки. Запись координат конца в переменную *Тк*.

5. Запрос карты.

Входные данные: карта поверхности от ГИС.

6. Получение матрицы поверхности.

Входные данные: карта с вероятностями встретить опасность.

Выходные данные: массив *V* с вероятностями в элементах, записанный в файл. Пример массива:

|  |  |  |
| --- | --- | --- |
| 1 | 0 | 0 |
| 1 | 1 | 0 |
| 1 | 1 | 0 |

Где номер строки и столбца массива соответствуют координатам точки (*x*,*y*), а параметр вероятности – значение в элементе массива.

Действия: преобразование в соответствии с п.3.1.

7. Аппроксимация матрицы полиномом.

Входные данные: массив *V*.

Выходные данные: аппроксимированная матрица.

Действия: обработка массива методом Лагранжа. Данный метод аппроксимации имеется в языке программирования.

Если невозможно аппроксимировать, то вывести сообщение о невозможности исполнения действия и завершение выполнения программы.

8. Построение траекторий.

Входные данные: начальная и конечная точки, вектор направления в конечной и начальной точке.

Выходные данные: список точек на прямых векторов, записанных в массив *D*. В общем виде массив выглядит:

|  |  |  |  |
| --- | --- | --- | --- |
| Точки на прямых векторов 1 | Точки на прямых векторов 2 | … | Точки на прямых векторов *n* |
| Радиусы поворотов на кривой | Радиусы поворотов на кривой | … | Радиусы поворотов на кривой |
| Интеграл кривой 1 | Интеграл кривой 2 | … | Интеграл кривой *n* |

Действия: построение кривых.

Данное действие рассматривается более подробно в следующих пунктах.

9. Начало цикла по траекториям *i*.

10. Определение криволинейного интеграла *Li* по поверхности траектории i.

Входные данные: номер траектории, граничные ограничения интеграла.

Выходные данные: число *Li*.

Действия: определение криволинейного интеграла по поверхности вдоль траектории *i*. Данное действие подробней рассматривается в следующих пунктах.

11. Нахождение оптимального значения *Lmin*

Входные данные: *Lmin*, *Li*.

Выходные данные: Минимальное значение.

Действия: сравнение.

12. Конец цикла по траекториям *i*.

13. *Lmin*<=*Lopt*

14. Определение точек на прямых векторов.

Входные данные: траектория движения.

Выходные данные: точки построения векторов направления МРК.

Действия: определение точек в соответствии с данными из массива *D*.

## 3.2 Алгоритм нахождения интеграла

Алгоритм нахождения интеграла представлен на рисунке 27. Формула интеграла следующая:

Где *l*0 – начало кривой Безье, *lk* – конец кривой Безье, – половина ширины МРК.

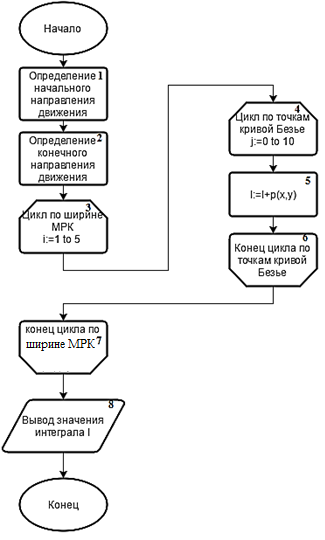


Рисунок 27 – Блок-схема нахождения интеграла поверхности движения

1. Определение начального направления движения.

Входные данные: данные о начальном направлении из глобального маршрута.

2. Определение конечного направления движения.

Входные данные: данные о конечном направлении из глобального маршрута.

3. Цикл по ширине МРК *i*:=1 to 5.

Входные данные: *Тк*, *Тн*, управляющие точки.

Выходные данные: формула кривой Безье с учетом текущих опорных и управляющих точек.

Действия: расчёт траектории с учетом измененных точек в зависимости от значения переменной *i*. В общем виде координаты точек изменяются следующим образом:

*x*1=(*i*-3)/2+*x*, *y*1=(3-*i*)/2+*y*.

4. Цикл по точкам кривой Безье *j*:=0 to 10.

5. *I*:=*I*+*p*(*x*,*y*)

Входные данные: переменная *I* типа *integer*, функция поверхности.

Выходные данные: переменная *I* типа *integer*.

Действия: суммирование значений вероятностей *p*(*x*,*y*) по всей траектории. Для того чтобы вычислить значение *p*(*x*,*y*), необходимо вычислить значение функции в текущей точке кривой.

6. Конец цикла по точкам кривой Безье.

7. Конец цикла по ширине МРК.

8. Вывод значения интеграла *I*.

Действия: запись значения интеграла в массив *D*

## 3.3 Алгоритм п**о**строения траектории

Алгоритм построения траектории движения роботизированного комплекса представлена на рисунке 28.

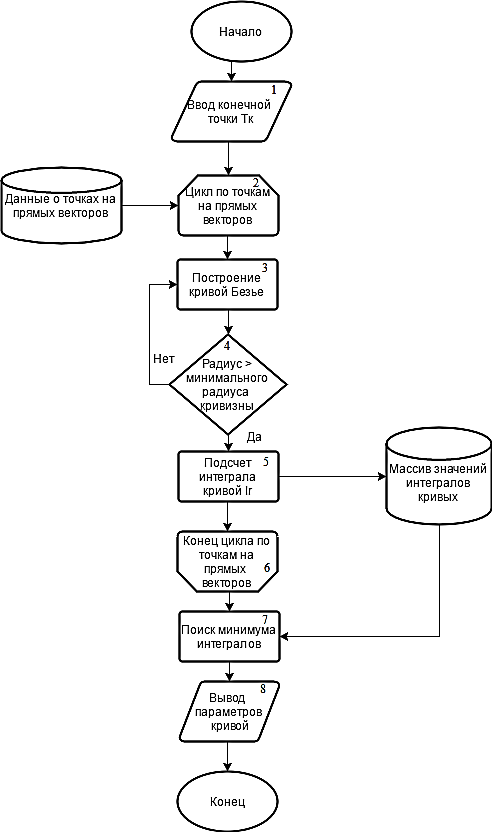


Рисунок 28 – Построение траектории движения

1. Ввод конечной точки *Тк*.

Входные данные: точка *Тк* из общего алгоритма.

2. Цикл по точкам на прямых векторов.

Входные данные: массив D.

3. Построение кривой Безье

Входные данные: *Тн*, *Тк*, направление вектора движения МРК.

Выходные данные: формула получившейся кривой.

Действия: построение по формуле

*E* = (1−*t*)3*E*1 + 3(1−*t*)2*tE*2 +3(1−*t*)*t*2*E*3 + t3*E*4

Вместо *E*i нужно подставить координаты *i*-й опорной точки (*xi*, *yi*).

Эти уравнения векторные, то есть для каждой из координат:

*x* = (1−*t*)3*x*1 + 3(1−*t*)2*tx*2 + 3(1− *t*)*t*2*x*3 + *t*2*x*4

*y* = (1−*t*)3*y*1 + 3(1−*t*)2*ty*2 + 3(1− *t*)*t*2*y*3 + *t*2*y*4

Вместо *x*1, *y*1, *x*2, *y*2, *x*3, *y*3, *x*4, *y*4  подставляются координаты четырёх опорных точек, и в то время как *t* пробегает множество от 0 до 1, соответствующие значения (*x*, *y*) как раз и образуют кривую. Точки *E*1 и *Е*4 – опорные точки, точки *Е*2 и *Е*3 – управляющие точки.

4. Радиус >= минимального радиуса кривизны

Входные точки: три точки в местах перегибов кривой

Выходные данные: радиус в привязке к точкам на векторах

Действия: находим длины сторон треугольника:

*a* =

*b* =

*c* =

По формуле Герона находим площадь треугольника:

*p* =

*S* =

Вычисляем радиус по формуле описанной окружности:

*R* =

5. Подсчет интеграла кривой Безье *I*

Входные данные: управляющие точки на векторах, точки начала и конца *Тк* и *Тн*.

Выходные данные: значение интеграла.

Действия: подсчет в соответствии с п.3.3.

Запись значения в массив *D*.

6. Конец цикла по точкам на прямых векторов.

7. Поиск минимума интегралов.

Входные данные: массив *D*.

Выходные данные: столбец с минимальным значением интеграла.

Действия: поиск минимума в строке значений интегралов.

8. Вывод параметров кривой.

Входные данные: массив *D*, найденный минимум интеграла.

Выходные данные: вывод параметра нужных точек на прямых векторов с радиусами.

Действия: вывод из массива *D* соответствующего значения столбца с минимальным значением интеграла.

## 3.4 Алгоритм избегания наезда на препятствие

Алгоритм избегания наезда на препятствие представлен на рисунке 29.

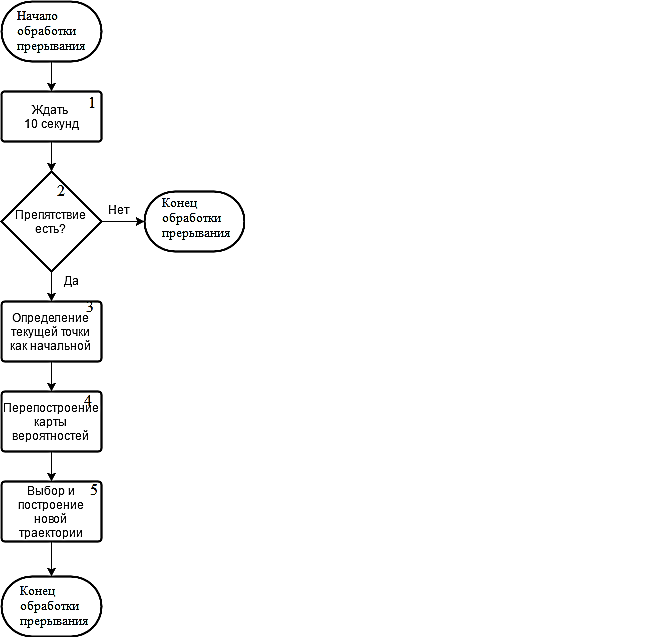


Рисунок 29 – Алгоритм избегания наезда на препятствие

1. Ждать 10 секунд.

Выходные данные: команда тормозной системе об останове на 10 сек.

2. Препятствие есть?

Входные данные: данные дальномера в виде расстояния до ближайшего препятствия.

Выходные данные: *TRUE* или *FALSE*.

Действия: сравнение расстояния до ближайшего препятствия с минимально допустимым расстоянием до препятствия.

3. Определение текущей точки как начальной.

Входные данные: текущее местоположение в виде точки на координатной плоскости (*X*,*Y*).

Выходные данные: начальная точка построения маршрута.

Действия: задание текущей точки как начальной.

4. Перепостроение карты вероятностей.

Входные данные: карта вероятностей встретить препятствие.

Выходные данные: определение функции новой поверхности.

Действия: аппроксимация карты вероятностей полиномом Лагранжа.

5. Выбор и построение новой траектории.

Входные данные: начальная и конечная точки, функция поверхности.

Выходные данные: новая траектория движения МРК.

Действия: нахождение оптимальной траектории движения МРК по местности.

## 3.5 Алгоритм избегания застревания в яме

Алгоритм избегания застревания в яме представлен на рисунке 30.

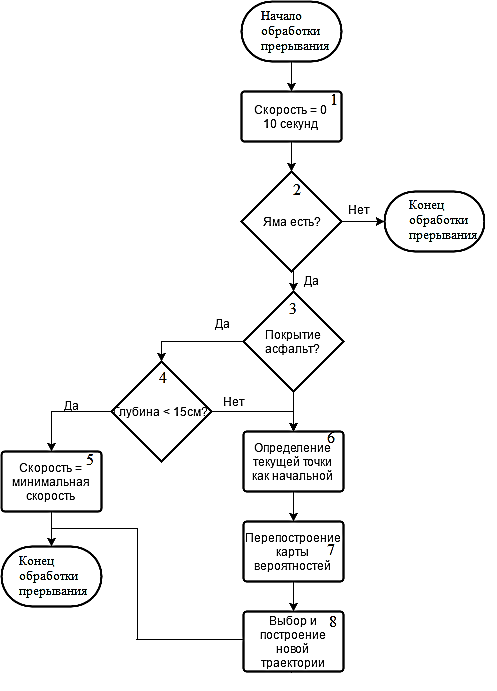


Рисунок 30– Алгоритм избегания застревания в яме

1. Скорость = 0 10 секунд.

Выходные данные: команда остановки на 10 секунд.

2. Яма есть?

Входные данные: данные СТЗ о поверхности перед МРК.

Выходные данные: *TRUE* или *FALSE*.

3. Покрытие асфальт?

Входные данные: информация от геоинформационной системы о типе покрытия дороги.

Выходные данные: *TRUE* или *FALSE*.

4. Глубина < 15 см?

Входные данные: данные с СТЗ и дальномеров о яме.

Выходные данные: *TRUE* или *FALSE*.

5. Скорость = минимальная скорость.

Выходные данные: необходимая скорость.

Действия: команда двигателю с сообщением скорости = 5 км/ч.

6. Определение текущей точки как начальной.

Входные данные: текущее местоположение в виде точки на координатной плоскости (*X*,*Y*).

Выходные данные: начальная точка построения маршрута.

Действия: задание текущей точки как начальной.

7. Перепостроение карты вероятностей.

Входные данные: карта вероятностей встретить препятствие.

Выходные данные: определение функции новой поверхности.

Действия: аппроксимация карты вероятностей полиномом Лагранжа.

8. Выбор и построение новой траектории.

Входные данные: начальная и конечная точки, функция поверхности.

Выходные данные: новая траектория движения МРК.

Действия: нахождение оптимальной траектории движения МРК по местности.

## 3.6 Алгоритм избегания проскальзывания/пробуксовки

Алгоритм проскальзывания и пробуксовки представлен на рисунке 31.

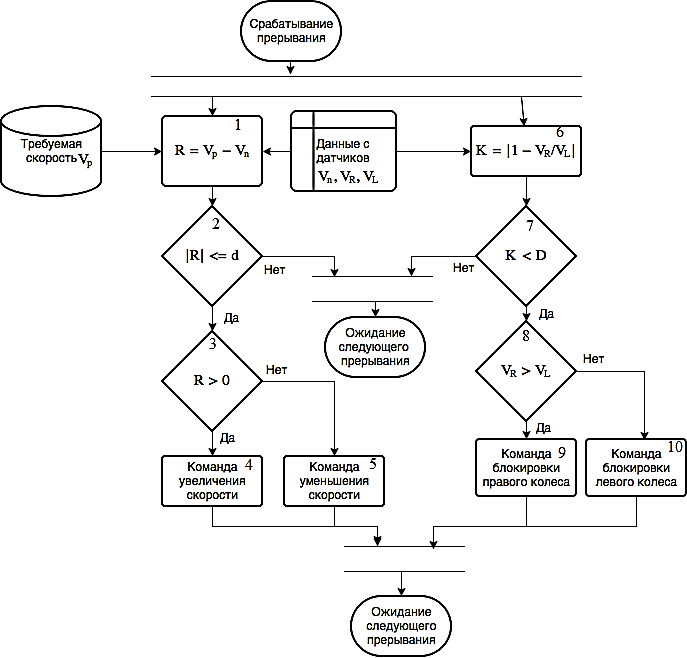


Рисунок 31 – Алгоритм избегания проскальзывания/пробуксовки

1. R = Vp – Vn

Входные данные: требуемая скорость, определенная программой *Vp*, реальная скорость с датчиков *Vn*.

Выходные данные: разница требуемой и реальной скоростей *R*.

Действия: *R* = *Vp* – *Vn*.

2. |R| <= d.

Входные данные: значение разницы требуемой и реальной скоростей *R*, значение максимально возможной разницы между скоростями *d*.

Выходные данные: *TRUE* или *FALSE*.

3. R > 0.

Входные данные: значение разницы скоростей *R*.

Выходные данные: *TRUE* или *FALSE*.

4. Команда увеличения скорости.

Выходные данные: команда увеличения скорости на значение *R*.

5. Команда уменьшения скорости.

Выходные данные: команда уменьшения скорости на значение *R*.

6. K = |1 – VR/VL|

Входные данные: данные с датчиков скоростей правого и левого колеса.

Выходные данные: значение коэффициента различия скоростей *K*.

Действия: *K* = |1 – *VR*/*VL*|.

7. K < D

Входные данные: значение коэффициента *K*, значение максимально возможного коэффициента различия скоростей *D*.

Выходные данные: *TRUE* или *FALSE*.

8. VR > VL

Входные данные: значения скоростей колес *VR* и *VL*.

Выходные данные: *TRUE* или *FALSE*.

9. Команда блокировки правого колеса.

Выходные данные: команда тормозной системе о блокировании движения правого колеса.

10. Команда блокировки левого колеса.

Выходные данные: команда тормозной системе о блокировании движения левого колеса.

## 3.7 Алгоритм сообщения об обрыве электрических цепей

Алгоритм оповещения об обрыве электрических цепей представлен на рисунке 32.

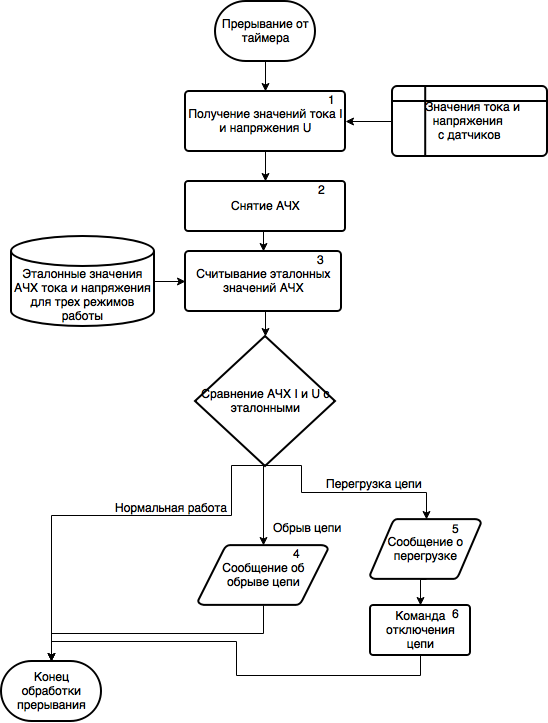


Рисунок 32 – Алгоритм сообщения об обрыве цепей

1. Получение значений тока I и напряжения U

Входные данные: обработанные значения с датчиков тока *I* и напряжения *U*.

2. Разбиение на частоты I и U

Входные данные: значение *U* и *I*.

Выходные данные: *АЧХ I, АЧХ U*.

Действия: вычисление максимального и минимального значений тока и напряжения на частотах [1-1\*2000;*n*-1\*2000]Гц.

3. Сравнение АЧХ I и U с эталонными

Входные данные: значения *U, I*, эталонные значения моделей нормальной работы *Unorm* и *Inorm*, обрыва цепи *Uobr*и *Iobr* и перегрузки *Upere* и *Ipere*.

Выходные данные: тип состояния функционирования.

Действия: суммирование по частотам сумм разностей АЧХ эталонных и АЧХ *U* и *I*, определение состояния функционирования с некоторой погрешностью ω: , где k – номер состояния функционирования, i – номер частоты, n – количество частот.

4. Сообщение об обрыве цепи.

Выходные данные: вывод сообщения об обрыве цепи.

5. Сообщение о перегрузке.

Выходные данные: вывод сообщения о чрезмерном потреблении напряжения.

6. Команда отключения цепи.

Выходные данные: команда об отключении цепи с чрезмерным потреблением напряжения.

## 3.8 Алгоритм сообщения о неисправностях по вибрациям

По частотам сумма не должна превышать некоторого значения погрешности. Алгоритм определения режима работы по вибрационным данным представлен на рисунке 33.

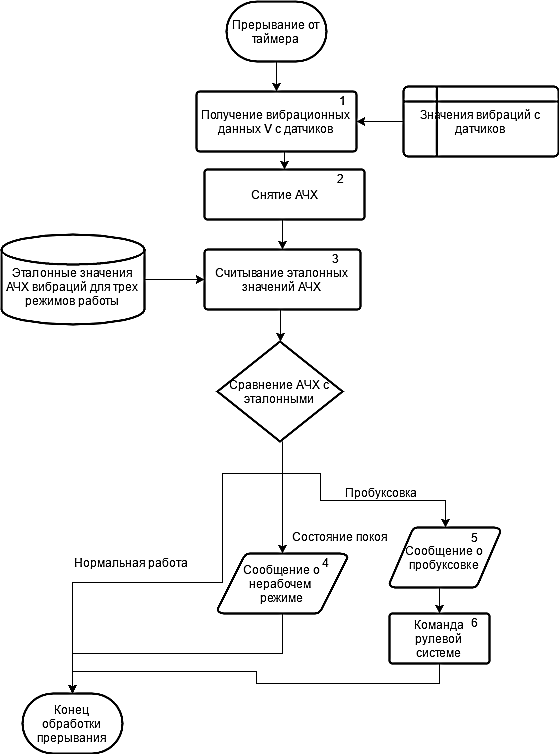


Рисунок 33 – Алгоритм определения режима работы по вибрациям

1. Получение вибрационных данных V с датчиков

Входные данные: обработанные значения с датчиков вибраций *V*.

2. Разбиение V на частоты

Входные данные: значение *V*.

Выходные данные: *АЧХ V*.

Действия: вычисление максимального и минимального значений тока и напряжения на частотах [1-1\*500;*n*-1\*500]Гц.

3. Сравнение АЧХ V с эталонными

Входные данные: значение *V*, эталонные значения моделей нормальной работы *Vnorm*, состояния покоя *Vrest* и пробуксовки *Vbuk*.

Выходные данные: тип состояния функционирования.

Действия: суммирование по частотам сумм разностей АЧХ эталонных значений и АЧХ *V*, определение состояния функционирования с некоторой погрешностью ω: , где k – номер состояния функционирования, i – номер частоты, n – количество частот.

4. Сообщение об обрыве цепи.

Выходные данные: вывод сообщения об обрыве цепи.

5. Сообщение о перегрузке.

Выходные данные: вывод сообщения о чрезмерном потреблении напряжения.

6. Команда отключения цепи.

Выходные данные: команда об отключении цепи с чрезмерным потреблением напряжения.

## 3.9 Алгоритм следования платформы по маршруту

Алгоритм следования платформы по выбранному маршруту представлен на рисунке 34.

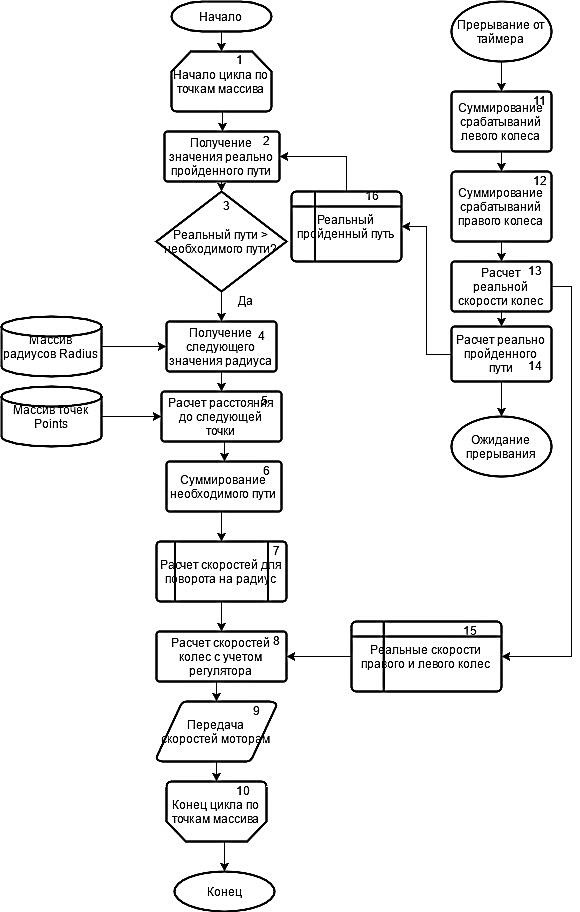


Рисунок 34 – Алгоритм следования платформы по маршруту

1. Начало цикла по точкам массива.

Входные данные: двумерный массив точек *Points* типа *float*.

Выходные данные: число циклов *i*.

2. Получение значения реально пройденного пути.

Входные данные: значение реального пройденного пути *path* типа *float*.

3. Реальный путь > необходимого пути?

Входные данные: *path*, *all*\_*way*.

Выходные данные: *TRUE*, *FALSE*.

4. Получение следующего значения радиуса.

Входные данные: массив *Radius*, *i*.

Выходные данные: присваивание переменной *rad* значение радиуса из массива радиусов *Radius* значения элемента *i*+1.

5. Расчет расстояния до следующей точки.

Входные данные: массив *Points*, *i.*

Выходные данные: значение переменной расстояния до следующей точки *way* типа *float*.

Действия: вычисления расстояния между двумя точками по формуле .

6. Суммирование необходимого пути.

Входные данные: значение пути до следующей точки *way*.

Выходные данные: значение необходимого пути на текущем значении цикла *all\_way* типа *float*.

Действия: *all\_way* = *all\_way* + *way*.

7. Расчет скоростей для поворота на радиус.

Входные данные: выставленная скорость *trg\_spd*, массив радиусов *Radius*.

Действия: передача радиуса в точке рулевой машине.

8. Расчет скоростей колес с учетом регулятора.

Входные данные: необходимые значения скоростей *left\_way*, *right\_way*, реальные значения скоростей *lft\_spd*, *rgt\_spd* типа *float.*

Выходные данные: значения скоростей с учетом регулирования *lft\_pwm* и *rgt\_pwm* типа *float*.

Действия: получение реальных и необходимых скоростей колес. Затем – вычисление разницы между необходимой и реальной скоростями с записью ошибок в переменные *lft\_err* и *rgt\_err* типа *float*. Далее необходимо вычислить значения *lft\_pwm* и *rgt\_pwm* с помощью ПИ-регулятора по общей формуле , где kp и ki – коэффициенты пропорциональности и интегральные соответственно, – интеграл ошибок с течением времени. В итоге, для левого и правого колес формулы выглядят следующим образом:

9. Передача скоростей моторам.

Входные данные: *lft\_pwm* и *rgt\_pwm*.

Действия: передача скорости аналоговому входу соответствующей стороны платформы.

10. Конец цикла по точкам массива.

Выходные данные: выход из цикла.

11. Суммирование срабатываний левого колеса.

Входные данные: срабатывание прерывания.

Выходные данные: переменная *lft*\_*tck* типа *integer*.

Действия: суммирование срабатываний прерываний левого энкодера.

12. Суммирование срабатываний левого колеса.

Входные данные: срабатывание прерывания.

Выходные данные: переменная *rgt*\_*tck* типа *integer*.

Действия: суммирование срабатываний прерываний правого энкодера.

13. Расчет реальной скорости колес.

Входные данные: *lft*\_*tck*, *rgt*\_*tck*.

Выходные данные: *lft\_spd*, *rgt\_spd* типа *float*.

Действия: *lft\_spd = (lft\_tck\*0,53)/t*, *rgt\_spd = (rgt\_tck\*0,53)/t*, где *t* – время по таймеру.

14. Расчет реально пройденного пути.

Входные данные: *lft\_spd*, *rgt\_spd.*

Выходные данные: переменная пройденного платформой пути *path* типа *float*.

Действия: , где t – время по таймеру.

15. Реальные скорости правого и левого колес.

Входные и выходные данные: *lft\_spd*, *rgt\_spd* типа *float*.

Действия: передача параметров программе.

16. Реальный пройденный путь.

Входные и выходные данные: переменная пройденного платформой пути *path* типа *float*.

Действия: передача параметров программе.

# Выводы по главе 3

В третьей главе выпускной квалификационной работы были рассмотрены алгоритмы выбора траектории движения по поверхности, подробно рассмотрены алгоритмы вычисления интеграла и построения траектории движения. Так же был рассмотрен алгоритм следования платформы по маршруту. Кроме того, были рассмотрены алгоритмы прогнозирования опасных режимов функционирования на основании мониторинга параметров.

# 4 Экспериментальная часть

## 4.1 Определение поверхности

Пусть имеется некоторая поверхность размерами 5x5 точек. Распределение вероятностей представлено на рисунке 35.

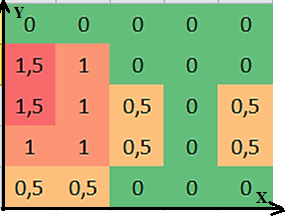


Рисунок 35 – Карта вероятностей

Массив поверхности представлен в виде массива *V* размером 5x5:

*V* =

В трёхмерном пространстве карта вероятностей на основе данных массива *V* будет иметь следующий вид (рисунок 36):

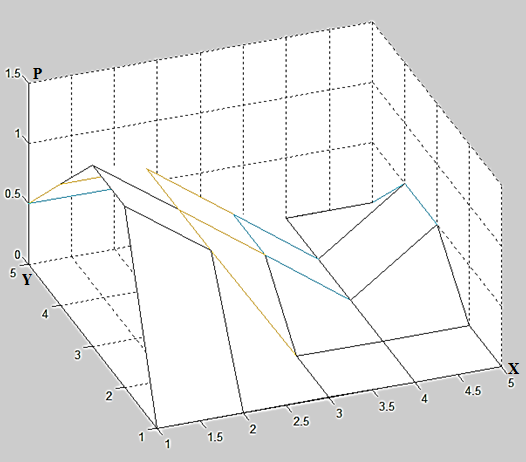


Рисунок 36 – Трёхмерный вид карты вероятностей

Пусть МРК необходимо двигаться из точки *A* в точку *B* (рисунок 37):

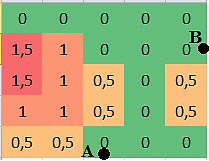


Рисунок 37 – Обозначение начальной и конечной точки движения

Используя аппроксимацию полиномом Лагранжа, получившаяся функция имеет вид:

*L*(*x*,*y*) = ˗1.75+*x*(0.25˗0.125*y*)+(1.875˗0.25*y*)*y*

## 4.2 Построение траекторий и выбор оптимальной траектории

Пусть данные о точках на прямых векторов имеют следующий вид:

|  |  |  |  |
| --- | --- | --- | --- |
| Точка 1 | 3, 2 | 3, 3 | 3, 4 |
| Точка 2 | 5, 3 | 5, 2 | 5, 1 |

Исходя из алгоритма, необходимо построить кривую с заданными точками. Построение кривой осуществляется по формуле:

*E* = (1−*t*)3*E*1 + 3(1−*t*)2*tE*2 +3(1−*t*)*t*2*E*3 + t3*E*4

Вместо *E*i нужно подставить координаты *i*-й опорной точки (*xi*, *yi*).

Эти уравнения векторные, то есть для каждой из координат:

*x* = (1−*t*)3*x*1 + 3(1−*t*)2*tx*2 + 3(1− *t*)*t*2*x*3 + *t*2*x*4

*y* = (1−*t*)3*y*1 + 3(1−*t*)2*ty*2 + 3(1− *t*)*t*2*y*3 + *t*2*y*4

Вместо *x*1, *y*1, *x*2, *y*2, *x*3, *y*3,  *x*1, *y*1 подставляются координаты четырёх опорных точек, и в то время как *t* пробегает множество от 0 до 1, соответствующие значения (*x*, *y*) как раз и образуют кривую. Точки *E*1 и *Е*3 – опорные точки *Тн*, *Тк*, точки *Е*2 и *Е*4 – управляющие точки.

На плоскости данные кривые расположены следующим образом (рисунок 38):



Рисунок 38 – Кривые Безье на плоскости

Итого, имеется 3 варианта кривых, по которым может двигаться МРК. Поскольку задача – найти оптимальный путь, с минимальным значением опасностей на пути, необходимо проинтегрировать значения кривых. Для этого и необходимо использовать функцию поверхности, которая была получена ранее путем аппроксимации.

Координаты начальной точки МРК = (3,1). Координаты конечной точки МРК = (5, 4).

Для каждой управляющей точки необходимо вычислить радиус, чтобы определить, насколько подходит данная кривая для движения по ней МРК.

Программа определения наилучшей траектории на языке *Python* приведена в Приложении А.

Поскольку необходимо выполнить расчет для 3 точек, данные представлены следующим образом:

|  |  |  |  |
| --- | --- | --- | --- |
| **Точка 1** | 3, 2 | 3, 3 | 3, 4 |
| **Точка 2** | 5, 3 | 5, 2 | 5, 1 |
| **Радиусы поворота** | 1.48125874622224  1.13154027516305 2.06382960240465 4.09814317325107 9.79196979433327  -  9.79196979432735 4.09814317325158 2.06382960240465 1.13154027516305 4.72847732164504 | 1.47405130945675  1.65128312453649  1.18976357911347  1.34835822907567  2.57659279787558  -  2.57659279787557  1.34835822907567  1.18976357911348  1.65128312453649 4.26588464606274 | 1.55775581949338  2.38110712421104  0.89124321910364  0.617093917835335  1.12454967399087  -  1.12454967399087  0.617093917835324  0.891243219103653  2.38110712421104  4.05398931088608 |
| **Интеграл** | 41.3435 | 45.1451 | 47.8775 |

Как видно из данных, пара точек (3, 4) и (5, 1) отпадает сразу, поскольку имеется радиус поворота меньше 1. Из двух оставшихся пар точек будет выбрана пара (3, 2) и (5, 3), поскольку вероятность встретить препятствие наименьшая. Далее данный столбец данных передается в качестве опорного для движения МРК.

Путь траектории с оптимальными опорными точками по ширине МРК представлен на рисунке 39.

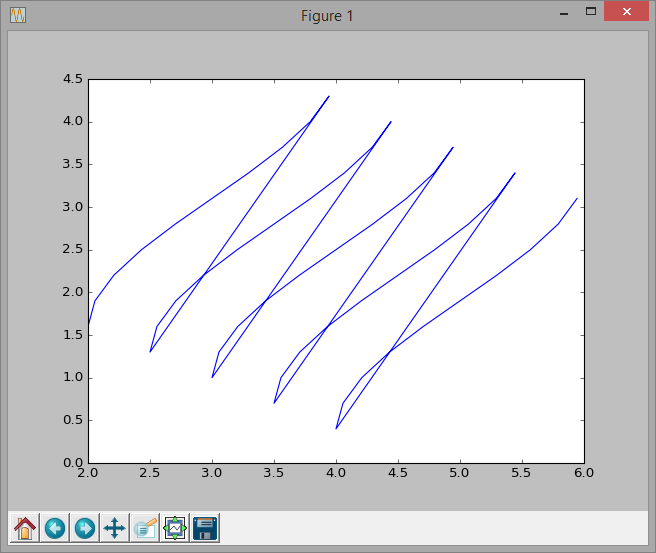


Рисунок 39 – Построение кривых Безье по ширине МРК

## 4.4 Верификация модели следования по маршруту

Ввиду того, что эксперименты необходимо проводить в условиях лаборатории, для данных целей была собрана модель МРК, представленная на рисунках 40 и 41.

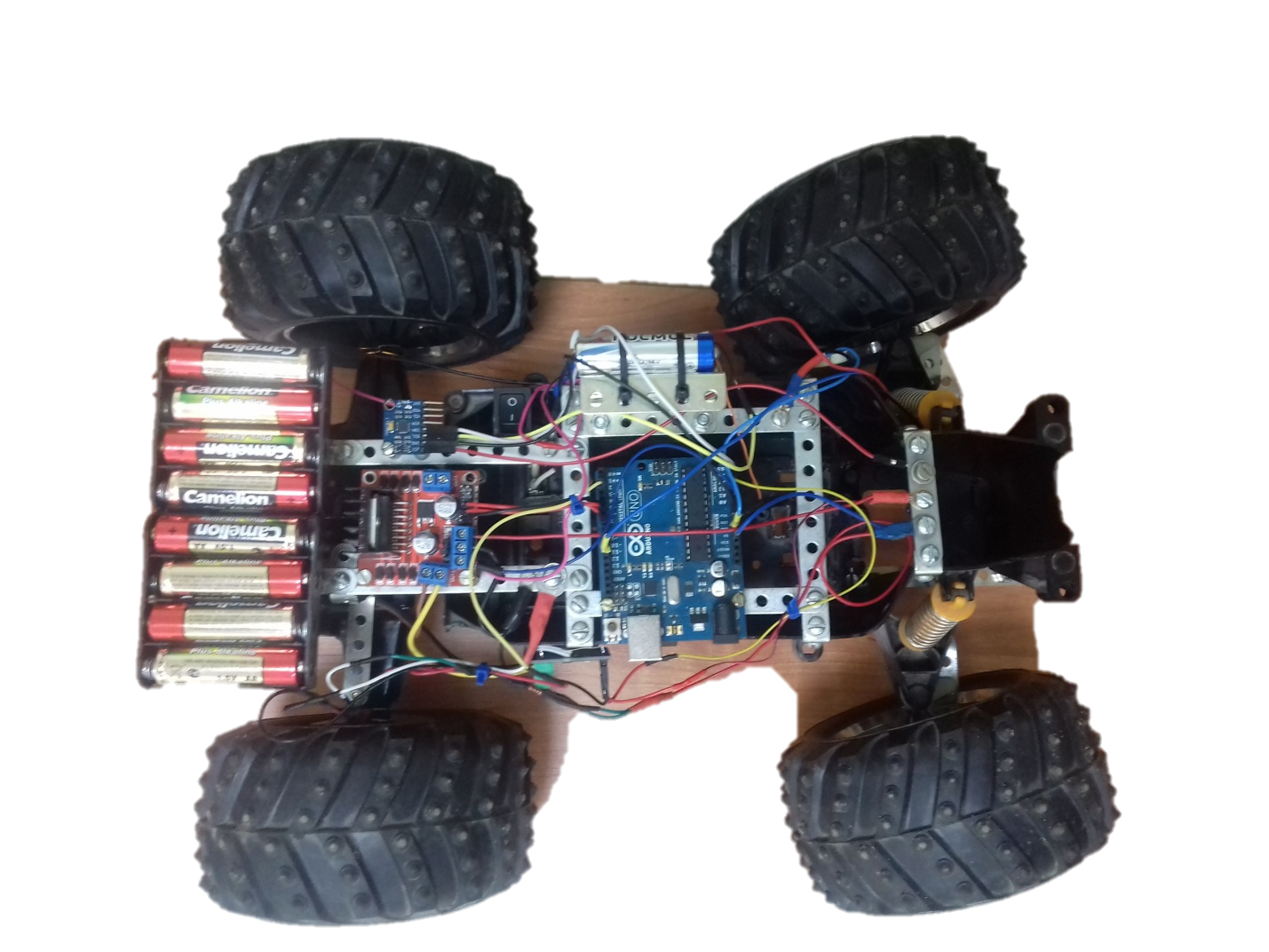


Рисунок 40 – Модель МРК, вид сверху



Рисунок 41 – Модель МРК, вид спереди

По алгоритму, описанному в параграфе 3.9, был написан программный код на языке программирования C. Программный код приведен в Приложении Б.

В результате эксперимента была преодолена траектория, соответствующая исходной. На рисунке 42 представлена исходная траектория и полученная в результате эксперимента. Синим цветом обозначена исходная траектория, чёрным – полученная в результате эксперимента.

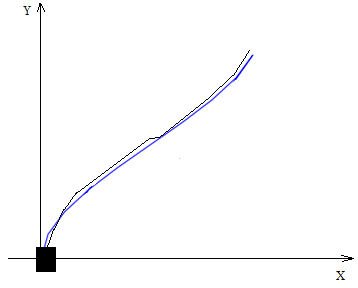


Рисунок 42 – Траектория движения платформы

В итоге, можно сделать вывод, что на этапе прохождения траектории поставленная цель была выполнена, однако, для увеличения точности прохождения необходимо использовать более мощные моторы, поскольку недостаточная мощность сказывалась на проходимости по покрытию.

Кроме того, данный эксперимент может свидетельствовать не только о корректном прохождении траектории, но так же об огибании препятствий и ям, поскольку в основе этих моделей так же лежит следование по маршруту с изменением функции поверхности во время следования. Ввиду отсутствия у модели МРК дальномеров и СТЗ, верификация данной модели так же свидетельствует о правильности исполнения и двух других моделей.

Таким образом, можно сделать вывод о корректности работы сразу трёх моделей: следование по маршруту, избегание препятствий и избегание застревания в яме.

## 4.5 Верификация модели предотвращения возникновения пробуксовки и проскальзывания

Листинг программы избегания пробуксовки/проскальзывания расположен в Приложении В.

Основное отличие модели МРК от МРК при реализации модели предотвращения пробуксовки и проскальзывания заключается в том, что у МРК имеется дифференциал между колесами, тогда как у модели колеса никак не взаимосвязаны. Поэтому для наиболее точной реализации необходимо смоделировать наличие дифференциала на модели. В программе дифференциал имитируется с помощью следующей части:

pwmZ = min(max(0,pwmZ),255);

pwm3 = pwmZ - brake3;

pwm4 = pwmZ - brake4;

pwm3 = min(max(0,pwm3),255);

pwm4 = min(max(0,pwm4),255);

pwm1 = pwm3;

pwm2 = pwm4;

Где pwm1…pwm4 – скорости. Передаваемые на моторы колес, колеса 1 и 2 имеют те же скорости, что и колеса 3 и 4. Таким образом, происходит регулировка между колесами: если скорость одного колеса сильно возрастает, а другое имеет меньшую скорость или не имеет скорости вообще, то скорость сильно вращающегося колеса уменьшается, а скорость колеса с меньшей скоростью увеличивается с целью съезда с препятствия.

Для проведения эксперимента платформа осуществляла движение по каменистой поверхности. Результат эксперимента представлен на рисунке 43.

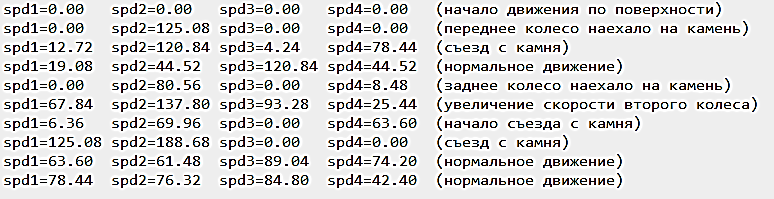


Рисунок 43 – Мониторинг скоростей при пробуксовке

Как видно из результатов эксперимента, платформа преодолевает препятствие путем изменения скоростей колес, тем самым, преодолевая препятствие.

## 4.6 Верификация модели сообщения об обрыве электрических цепей

Листинг программы, сообщающей об обрыве электрических цепей, представлен в Приложении Г.

В ходе эксперимента были получены следующие значения АЧХ тока (рисунок 44) и напряжения (рисунок 45):

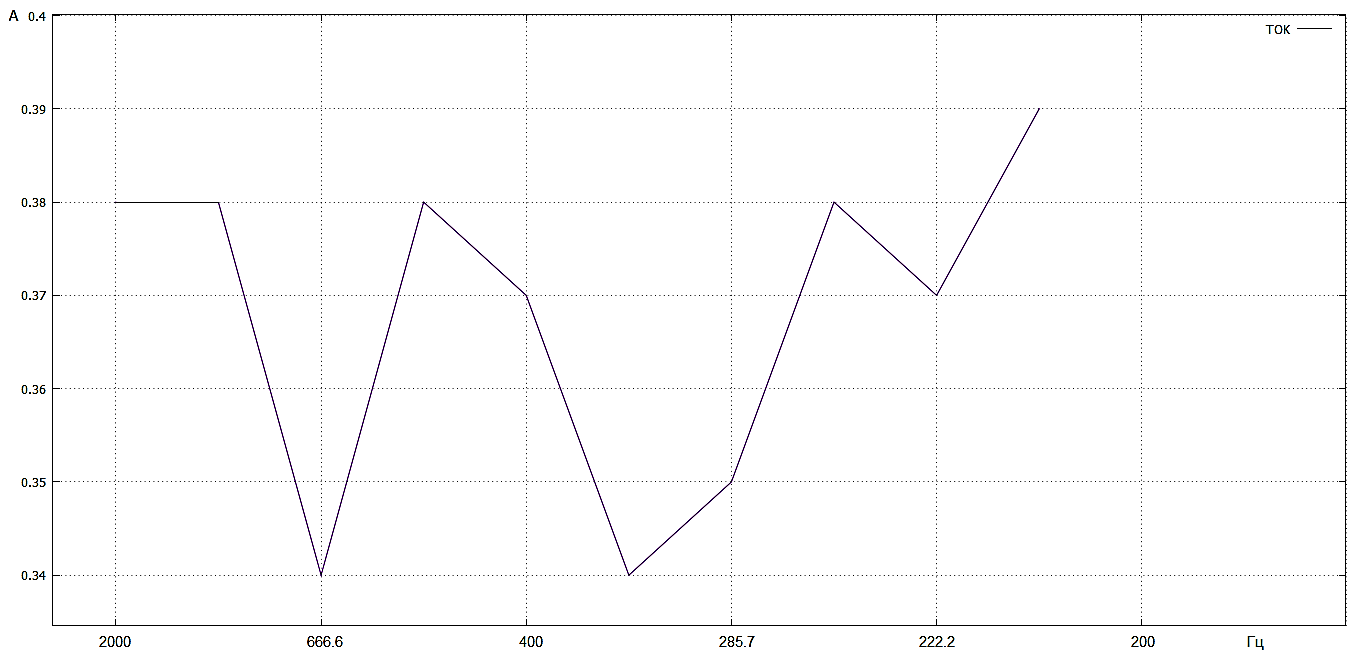


Рисунок 44 – АЧХ тока



Рисунок 45 – АЧХ напряжения

При сравнении фигур площадей под графиком было вычислено, что данный режим соответствует режиму обрыва электрической цепи. На рисунках 46, 47 продемонстрировано сравнение АЧХ полученных данных и АЧХ эталонных значений.

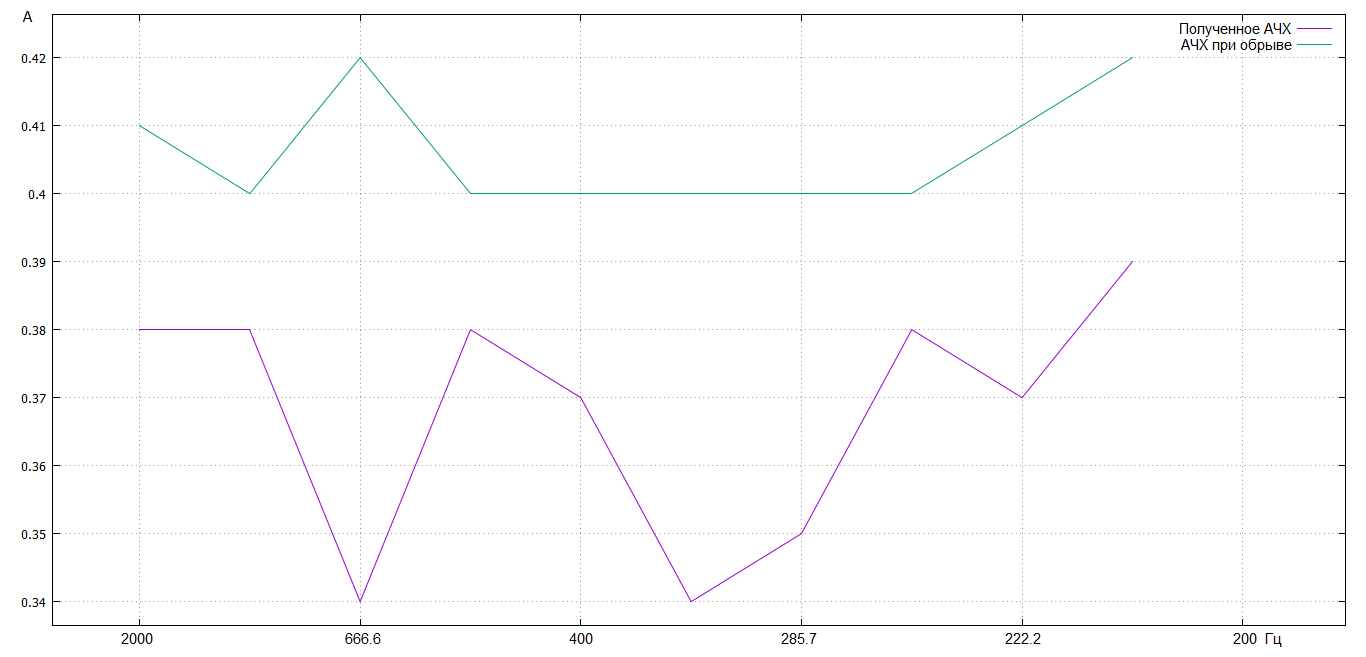


Рисунок 46 – Сравнение графиков АЧХ тока

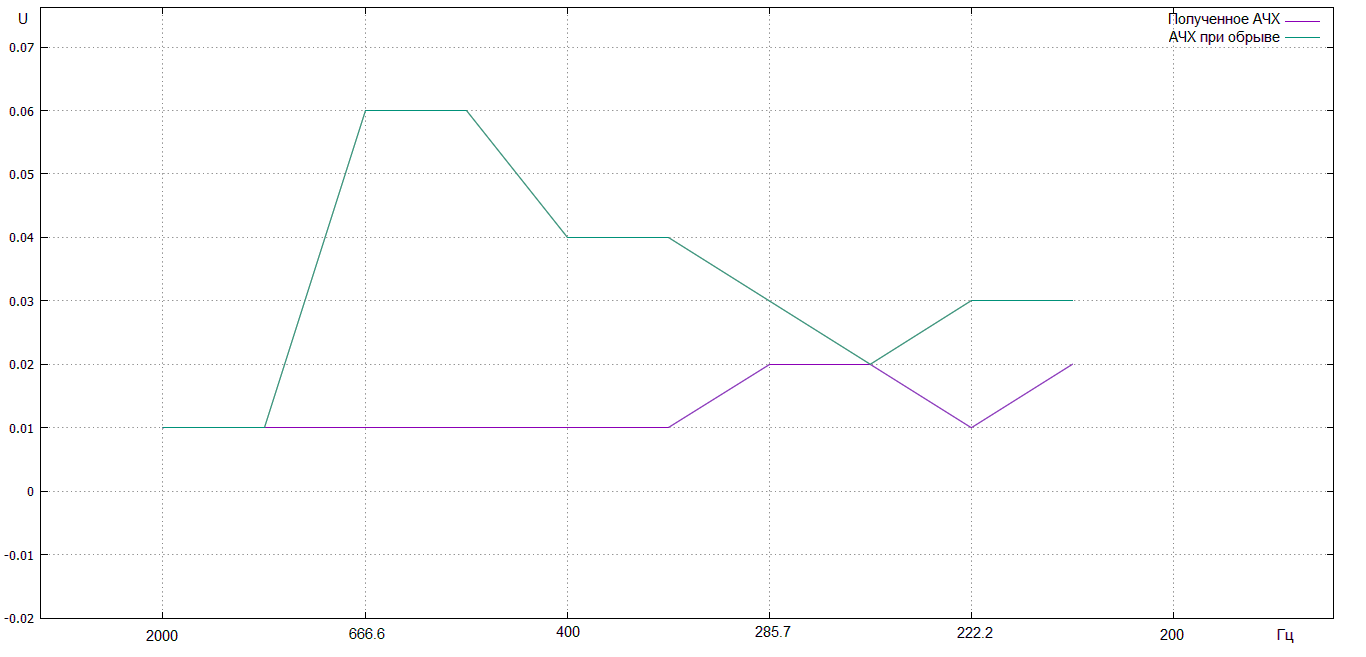


Рисунок 47 – Сравнение графиков АЧХ напряжения

Как видно из графиков, сравнение текущих данных с установленными значениями, характеризующими одно из трёх состояний, выполнено успешно, модель работает корректно.

## 4.7 Верификация модели сообщения о неисправностях по вибрациям

Программный код, реализующий информирование о неисправностях по значению уровня вибраций, представлен в Приложении Д. В ходе эксперимента получены следующие значения АЧХ (рисунок 48):

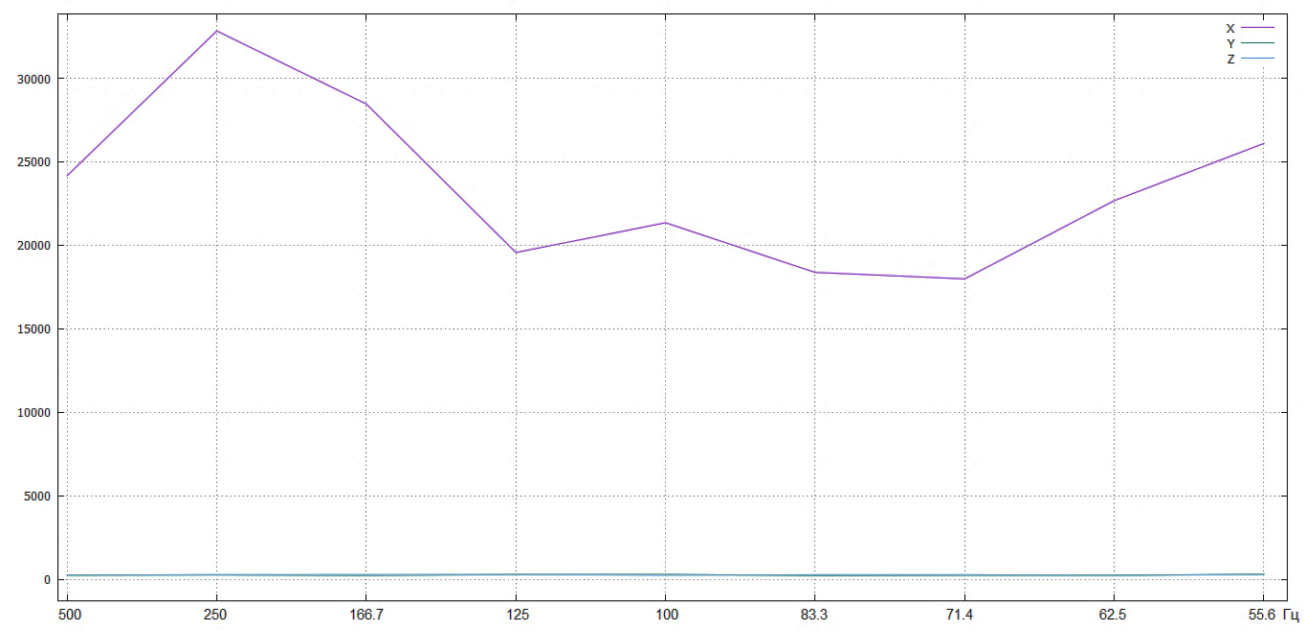


Рисунок 48 – АЧХ полученных вибраций по осям X,Y,Z

При сравнении площадей фигур под графиком было вычислено, что полученные вибрации соответствуют состоянию покоя.

Сравнение графиков АЧХ эталонных значений и полученных приведено на рисунках 49 – 51.

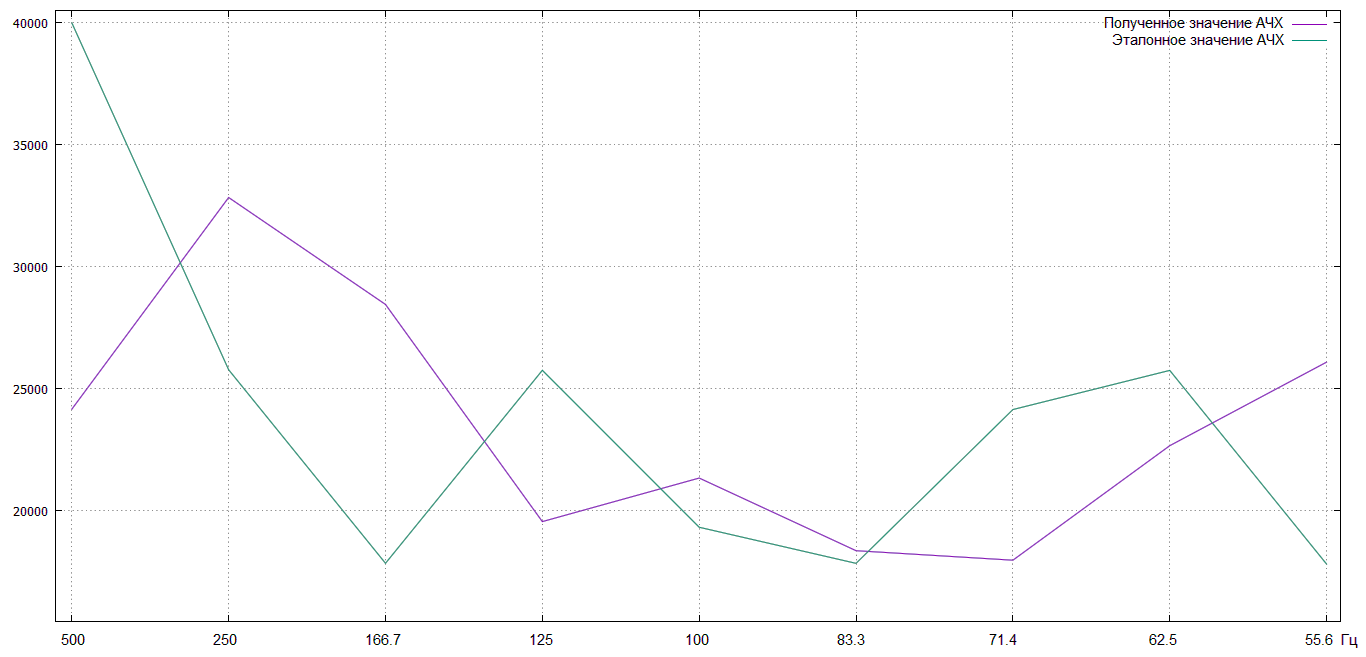


Рисунок 49 – Сравнение АЧХ по оси X

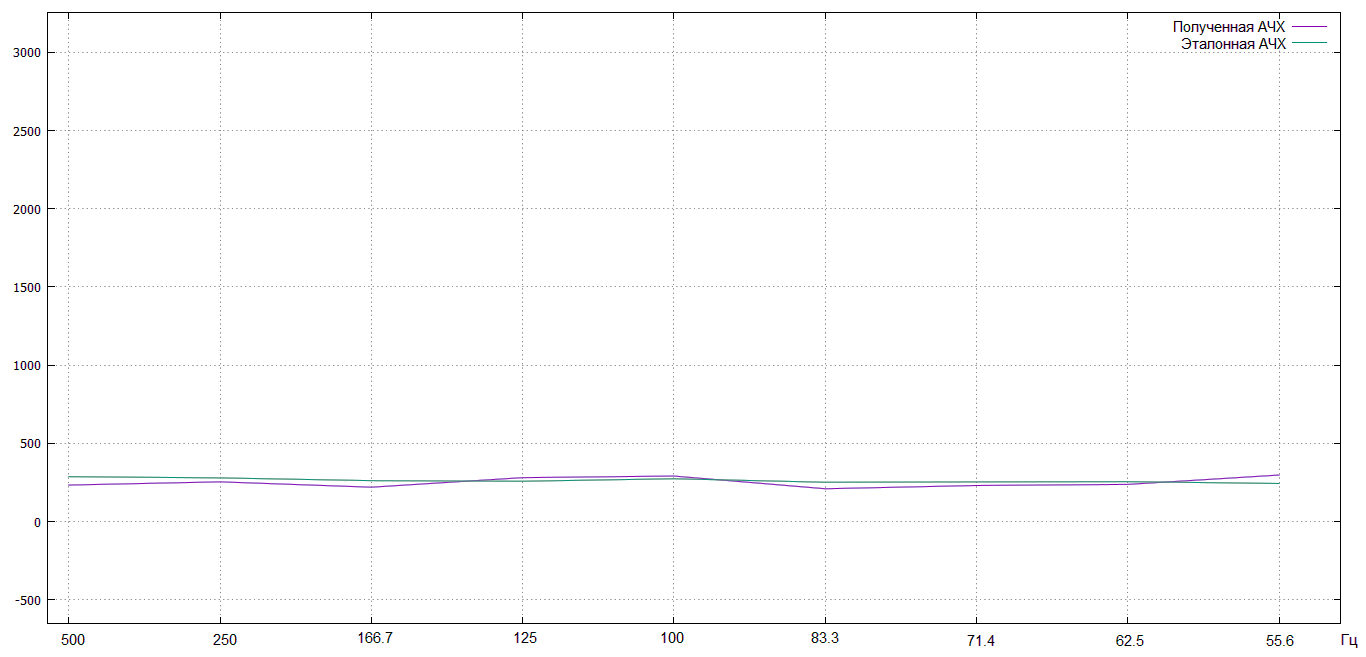


Рисунок 50 – Сравнение АЧХ по оси Y

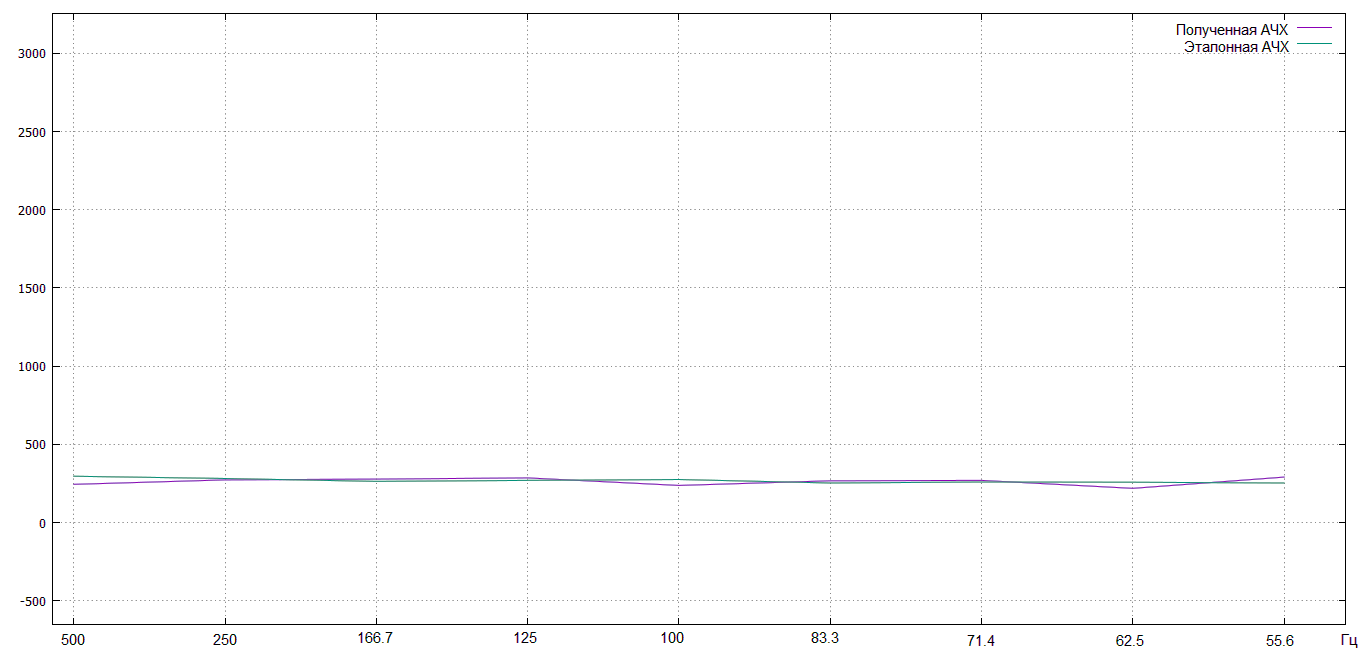


Рисунок 51 – Сравнение АЧХ по оси Z

Как видно из графиков, сравнение текущих данных с установленными значениями, характеризующими одно из трёх состояний, выполнено успешно, модель работает корректно.

# Выводы по разделу 4

В четвертом разделе магистерской диссертации была проведена верификация моделей:

- выбора оптимальной траектории;

- следования по маршруту;

- избегания столкновения с препятствием;

- избегания застревания в яме;

- избегания пробуксовки;

- сообщения об обрыве электрических цепей;

- сообщения о неисправностях по вибрациям.

Все модели работают корректно, поставленную задачу можно считать выполненной.

# Заключение

В ходе выполнения выпускной квалификационной работы были получены следующие результаты:

- осуществлен обзор источников на заданную тему;

- произведен обзор методов нахождения функции поверхности;

- произведен выбор способа расчёта движения МРК по поверхности;

- осуществлён расчет траектории движения МРК по поверхности;

- осуществлено создание модели выбора и построения траектории движения МРК;

- определен перечень опасных режимов функционирования МРК;

- определен перечень измеряемых показателей;

- определены значения измеряемых показателей, характерных для аварийных ситуаций;

- создана модель избегания опасных режимов функционирования МРК;

- осуществлён мониторинг параметров МРК при движении по траектории;

- произведена верификация моделей.

В результате была создана модель прогнозирования опасных режимов функционирования роботизированного комплекса при локальном движении по траектории.

Данный отчет, отражающий основные результаты в ходе выполнения выпускной квалификационной работы, выполнен в соответствии с ГОСТ 7.32-2001. Структура и состав представленного отчета соответствует требованиям, предъявляемым к документации по магистерским диссертациям кафедры «Информационные технологии и автоматизированные системы» Пермского национального исследовательского политехнического университета.

В качестве перспективы развития может быть предложено внедрение экспертной системы для более гибкого определения аварийных режимов функционирования МРК с большим выбором действий в случае возникновения опасности аварийной ситуации.

# Список используемых источников

1. Кошечкин А.А. Принцип построения траектории движения шагающего робота при наличии препятствий / Кошечкин А.А., Лапиков А.Л. В кн.: Вибрационные технологии, мехатроника и управляемые машины, Курск, 2016 г: материалы. Юго-Западный государственный университет, 2016. С.: 281-287.

2. Назарова А.А. Использование методов машинного обучения для построения траектории движения / Назарова А.А., Тарасов В.С. В кн.: Информатика: проблемы, методология, технологии, Воронеж, 2016: материалы XVI Международной научно-методической конференции, 2016. С.: 148-154.

3. Старкова И.В. Поиск траектории движения мобильного робота-гида / Старкова И.В., Абрамов А.И. В кн.: Молодые ученые - ускорению научно-технического прогресса в XXI веке, Ижевск, 2016: сборник материалов IV Всероссийской научно-технической конференции аспирантов, магистрантов и молодых ученых с международным участием, 2016. С.: 235-238.

4. Андрианова О.Г. Кинематическое управление мобильным роботом при движении по полигону с обходом препятствий / Андрианова О.Г., Королькова М.А., Кочетков С.А., Краснова С.А. Материалы конференции «Управление в технических системах» (УТС–2010, Санкт-Петербург). СПб.: ОАО «Концерн «ЦНИИ «Электроприбор», 2010. С.: 356-359

5. Подоплёкин Ю.Ф. Совместное управление движением судна и беспилотного летательного аппарата при посадке / Подоплёкин Ю.Ф., Шаров С.Н. В кн.: Информационно-управляющие системы, Санкт-Петербург, 2014: Информационно-управляющие системы, Санкт-Петербург, 2014. С.: 36-44

6. Синицина Н.В. Нечеткая модель для системы автоматизированного управления автомобилем / Синицина Н.В., Федосова Е.Б. В кн.: XXVIII международная инновационно-ориентированная конференция молодых учёных и студентов МИКМУС-2016, Москва, 2016: Материалы конференции «XXVIII международная инновационно-ориентированная конференция молодых учёных и студентов МИКМУС-2016», 2016. С.: 287-290.

7. Кычкин А.В. Структурный синтез информационно- измерительной и управляющей системы мобильной платформы / Кычкин А.В., Артёмов С.А., Власов В.А. В кн.: Вестник ПНИПУ №7, Пермь, 2013. С.: 83-95.

8. Чучуева И.И. Классификация методов и моделей прогнозирования [Электронный ресурс] – URL: https://habrahabr.ru/post/177633/ (дата обращения: 10.05.2017).

9. Финн В.К. Искусственный интеллект: Идейная база и основной продукт, 9-я национальная конференция по искусственному интеллекту, Труды конференции, Т.1, М., Физматлит, 2004.

10. Воробьев В. В. Алгоритм поиска и планирования пути для задач групповой робототехники [Электронный ресурс] — URL: http://publications.hse.ru/chapters/105673333 (дата обращения: 16.05.2017).

11. Планирование траектории мобильного робота [Электронный ресурс] — URL: http://robot-develop.org/archives/2554 (дата обращения: 20.05.2017).

**Приложение А**

**Листинг программы построения траектории движения МРК**

import numpy

import math

import matplotlib.pyplot as plt

x1=3

y1=1

x2=3

y2=2

x3=5

y3=3

x4=5

y4=4

B=numpy.zeros([11,2])

C=[]

D=[]

#Считаются коэффициенты q1,q2,q3,q4

i=0

for b in B:

t=i/10

q1=t\*t\*t\*(-1)+t\*t\*3+t\*(-3)+1

q2=t\*t\*t\*3+t\*t\*(-6)+t\*3

q3=t\*t\*t\*(-3)+t\*t\*3

q4=t\*t\*t

b[0]=q1\*x1 + q2\*x2 + q3\*x3 + q4\*x4

b[1]=q1\*y1 + q2\*y2 + q3\*y3 + q4\*y4

i=i+1

print ("Радиусы углов поворота колёс:")

#Вычисления сторон треугольника 0

a=math.sqrt(((B[0,0]-1)-B[1,0]) \*\* 2+((B[0,1]-1)-B[1,1]) \*\* 2)

bb=math.sqrt((B[1,0]-B[2,0]) \*\* 2+(B[1,1]-B[2,1]) \*\* 2)

c=math.sqrt((B[2,0]-(B[0,0]-1)) \*\* 2+(B[2,1]-(B[0,1]-1)) \*\* 2)

#Вычисления периметра и площади треугольника

p=(a+bb+c)/2

S=math.sqrt(p\*(p-a)\*(p-bb)\*(p-c))

#Вычисление радиуса описанной окружности

try:

R=(a\*bb\*c)/(4.0\*S)

except Exception as e:

R = 0

print (R, )

#Вычисление радиусов по всей длине

for i in range(0,9):

a=math.sqrt((B[i,0]-B[i+1,0]) \*\* 2+(B[i,1]-B[i+1,1]) \*\* 2)

bb=math.sqrt((B[i+1,0]-B[i+2,0]) \*\* 2+(B[i+1,1]-B[i+2,1]) \*\* 2)

c=math.sqrt((B[i+2,0]-B[i,0]) \*\* 2+(B[i+2,1]-B[i,1]) \*\* 2)

p=(a+bb+c)/2

S=math.sqrt(p\*(p-a)\*(p-bb)\*(p-c))

try:

R=(a\*bb\*c)/(4.0\*S)

except Exception as e:

R = 0

print (R, )

if R<1: print ("Радиус меньше радиуса циркуляции МРК. Необходимо отбросить данную траекторию.")

#Вычисление последнего радиуса

a=math.sqrt((B[8,0]-B[9,0]) \*\* 2+(B[8,1]-B[9,1]) \*\* 2)

bb=math.sqrt((B[9,0]-(B[9,0]+1)) \*\* 2+(B[9,1]-(B[9,1]+1)) \*\* 2)

c=math.sqrt(((B[9,0]+1)-B[8,0]) \*\* 2+((B[9,1]+1)-B[8,1]) \*\* 2)

#Вычисления периметра и площади треугольника

p=(a+bb+c)/2

S=math.sqrt(p\*(p-a)\*(p-bb)\*(p-c))

#Вычисление радиуса описанной окружности

try:

R=(a\*bb\*c)/(4.0\*S)

except Exception as e:

R = 0

print (R, )

Integral=0

#Вычисление интеграла

for j in range (1,6):

for i in range (0,10):

t=i/10

q1=t\*t\*t\*(-1) + t\*t\*3 + t\*(-3) + 1

q2=t\*t\*t\*3 + t\*t\*(-6) + t\*3

q3=t\*t\*t\*(-3) + t\*t\*3

q4=t\*t\*t

#Расчёт координат с учетом изменения опорных точек

x=q1\*((j-3)/2+x1) + q2\*((j-3)/2+x2) + q3\*((j-3)/2+x3) + q4\*((j-3)/2+x4)

y=q1\*((3-j)\*0.3+y1) + q2\*((3-j)\*0.3+y2) + q3\*((3-j)\*0.3+y3) + q4\*((3-j)\*0.3+y4)

C.append(x)

D.append(y)

Integral=Integral+(-1.75+x\*(0.25-0.125\*y)+(1.875-0.25\*y)\*y)

print (Integral)

plt.plot(C,D)

plt.show()

**Приложение Б**

**Листинг программы следования МРК по траектории**

volatile uint16\_t tme = 0; //счетчик

volatile uint8\_t tck = 0;//переменная срабатываний прерывания

volatile float spd = 0;//переменная скорости

int8\_t trg\_spd = 0;//переменная установленной скорости

float path = 0;//переменная пройденного пути

float pth = 0;//переменная пройденного пути

float koef = 0.1;//переменная преобразования длины маршрута

float DT = 0.25;//переменная приращения

float Points[2][10] = {{3.0, 3.056, 3.21, 3.43, 3.7, 4.0, 4.296, 4.57, 4.79, 4.94}, {1.0, 1.3, 1.6, 1.9, 2.2, 2.5, 2.8, 3.1, 3.4, 3.7}};//массив точек кривой

float Radius[10] = {1.13, 2.06, 4.1, 9.79, 0.1, -9.79, -4.1, -2.06, -1.13, -2.79};//массив радиусов углов поворота в точках кривой

float way = 0;//переменная требуемого участка пути между точками

float all\_way = 0;//переменная требуемого пройденного пути

float rad = 2000 / (100\*Radius[0] - 2.75) + 140;//переменная радиуса

int8\_t step = 0;//счётчик

**void** setup()

{

pinMode(2, INPUT); //Входные данные

pinMode(5, OUTPUT);//Выходные данные

pinMode(6, OUTPUT);

pinMode(7, OUTPUT);

pinMode(8, OUTPUT);

pinMode(9, OUTPUT);

pinMode(10, OUTPUT);//Выходные данные

Serial.begin(9600);

//Таймер

OCR0A = 0xAF;

TIMSK0 |= \_BV(OCIE0A);

//Прерывание энкодера

attachInterrupt(0, tick, CHANGE);

}

**void** tick()//Функция суммирования срабатываний

{

tck++;

}

SIGNAL(TIMER0\_COMPA\_vect) //Функция прерываний

{

if (++tme == int(1000 \* DT))

{

spd = tck \* koef / DT; //преобразование в см/сек

pth += spd \* DT; //Вычисление пройденного пути

path += pth \* DT;//Вычисление реального пройденного пути

tme = tck = 0;

**run**();

}

}

**void** run()

{

analogWrite(11,rad);//Угол поворота

analogWrite(5,trg\_spd);//Скорость

digitalWrite(7, 1);//Направление движения

digitalWrite(6, 0);

}

**void** loop()

{

trg\_spd = 200;//установление скорости

delay(100);

if (abs(all\_way - path) < 15)

{

step++;

if (step>10) trg\_spd=0;//условие ограничения по кол-ву точек

rad = 2000 / (100\*Radius[step - 1] - 2.75) + 140;

if (rad > 180) rad = 180;

if (rad < 100) rad = 100;

way = 100 \* sqrt(pow((Points[1][step - 1] - Points[1][step]), 2) - pow((Points[2][step - 1] - Points[2][step - 1]), 2));

all\_way += way;

}

}

**Приложение В**

**Листинг программы избегания пробуксовки/проскальзывания**

#include <AFMotor.h>

volatile uint16\_t tme = 0;

volatile uint8\_t tck1 = 0; //переменные суммирования

volatile uint8\_t tck2 = 0; //срабатываний энкодера

volatile uint8\_t tck3 = 0;

volatile uint8\_t tck4 = 0;

volatile float spd1 = 0;//переменные реальных скоростей

volatile float spd2 = 0;

volatile float spd3 = 0;

volatile float spd4 = 0;

volatile float err3 = 0;//переменная ошибки

volatile float pwm3 = 0;//переменные скоростей

volatile float pwm4 = 0;//с учетом регулирования

volatile float pwmZ = 0;

volatile float brake3 = 0;//переменные регулирования

volatile float brake4 = 0;//ошибок

int8\_t trg\_spd1 = 0;//переменные требуемых

int8\_t trg\_spd2 = 0;//скоростей колес

int8\_t trg\_spd3 = 90;

float koef = 0.53;//коэффициент преобразования скоростей

float K = 0;//переменная разницы скоростей колес

float k3 = 0.2;//коэффициент регулирования

float DT = 0.25;//коэффициент приращения времени

AF\_DCMotor my\_motor1(1);//инициализация моторов

AF\_DCMotor my\_motor2(2);

AF\_DCMotor my\_motor3(3);

AF\_DCMotor my\_motor4(4);

**void** setup()

{

Serial.begin(9600);

//Таймер

OCR0A = 0xAF;

TIMSK0 |= \_BV(OCIE0A);

//Прерывание колес

attachInterrupt(5, tick3, CHANGE);

attachInterrupt(4, tick4, CHANGE);

attachInterrupt(0, tick1, CHANGE);

attachInterrupt(1, tick2, CHANGE);

//Задание скоростей моторов

my\_motor1.setSpeed(0);

my\_motor2.setSpeed(0);

my\_motor3.setSpeed(0);

my\_motor4.setSpeed(0);

//Задание направления движения моторов

my\_motor1.run(FORWARD);

my\_motor2.run(FORWARD);

my\_motor3.run(FORWARD);

my\_motor4.run(FORWARD);

}

**void** tick1()//Функция прерывания колес

{

tck1++;

}

**void** tick2()//Функция прерывания колес

{

tck2++;

}

**void** tick3()//Функция прерывания колес

{

tck3++;

}

**void** tick4()//Функция прерывания колес

{

tck4++;

}

SIGNAL(TIMER0\_COMPA\_vect) //Функция прерываний

{

if (++tme == int(1000 \* DT))

{

spd1 = tck1 \* koef / DT; //преобразование в см/сек

spd2 = tck2 \* koef / DT; //преобразование в см/сек

spd3 = tck3 \* koef / DT; //преобразование в см/сек

spd4 = tck4 \* koef / DT; //преобразование в см/сек

tme = tck1 = tck2 = tck3 = tck4 = 0;

**run**();

}

}

**void** run()

{

//Регулятор скоростей

err3 = trg\_spd3 - (spd3 + spd4);

pwmZ += k3 \* err3;

//Вычисление скоростей колес

pwmZ = min(max(0,pwmZ),255);

pwm3 = pwmZ - brake3;

pwm4 = pwmZ - brake4;

pwm3 = min(max(0,pwm3),255);

pwm4 = min(max(0,pwm4),255);

pwm1 = pwm3;

pwm2 = pwm4;

//Передача скоростей моторам

my\_motor1.setSpeed(pwm1);

my\_motor2.setSpeed(pwm2);

my\_motor3.setSpeed(pwm3);

my\_motor4.setSpeed(pwm4);

}

**void** loop()

{

//Задание требуемых скоростей

trg\_spd1 = spd1;

trg\_spd2 = spd2;

if (spd3 == 0 & spd4 == 0)

{

K = 1;

}

else

{

//Вычисление разницы скоростей колес

K = min(spd3, spd4) / max(spd3, spd4);

}

if (K < 0.5)

{

if (spd3 > spd4)

{

brake4 = 0;

brake3 = 255;

}

else

{

brake3 = 0;

brake4 = 255;

}

}

else

{

brake4 = brake3 = 0;

}

delay(1000);

}

**Приложение Г**

**Листинг программы сообщения об обрыве электрических цепей**

volatile double I = 0;

volatile double U = 0;

volatile uint8\_t s = 0;

uint32\_t T0 = 0;

volatile float D = 0;

volatile double Iobr = 0;

volatile double Inorm = 0;

volatile double Ipere = 0;

volatile double Uobr = 0;

volatile double Unorm = 0;

volatile double Upere = 0;

volatile double Inow = 0;

volatile double Unow = 0;

float w = 0.05;

//массив полученных значений

volatile float II [10][2];

volatile float UU [10][2];

//массив эталонных значений

volatile uint16\_t III [3][10] = {{0.41, 0.40,0.42,0.40,0.40,0.40,0.40,0.40,0.41},

{0.48,0.54,0.62,0.68,0.41,0.62,0.67,0.59,0.54},{0.46,0.46,0.51,0.40,0.46,0.45,0.47,0.40,0.45}};

volatile uint16\_t UUU [3][10] = {{0.01,0.01,0.06,0.06,0.04,0.04,0.03,0.03,0.03},

{0.94,1.11,1.50,1.49,0.86,1.50,1.46,1.49,1.46},{0.32,1.03,1.02,0.95,0.93,0.99,1.01,0.95,0.97}};

**void** zero ()//обнуление значений

{

for (int i = 1; i <= 9; ++i)

{

if (s % i == 0)

{

II[i-1][1] = 0;

II[i-1][0] = 1023;

UU[i-1][1] = 0;

UU[i-1][0] = 1023;

}

}

}

**void** setup()

{

digitalWrite(7,1);

digitalWrite(6,0);

digitalWrite(5,1);

**zero**();

Serial.begin(9600);

//Таймер

OCR1A = 125;

TIMSK1 |= \_BV(OCIE0A);

}

SIGNAL(TIMER1\_COMPA\_vect) //Функция прерываний

{

TCNT1 = 0;

digitalWrite(13,!digitalRead(13));

I = analogRead(A2) \* 0.0149 - 7.3881;

U = analogRead(A0) \* 0.0055 - 0.7643;

if (s++ > 9) s = 0;

for (int i = 1; i <= 9; ++i)

{

if (s % i == 0)

{

II[i-1][1] = (I > II[i-1][1]) ? I : II[i-1][1];

II[i-1][0] = (I < II[i-1][0]) ? I : II[i-1][0];

UU[i-1][1] = (U > UU[i-1][1]) ? U : UU[i-1][1];

UU[i-1][0] = (U < UU[i-1][0]) ? U : UU[i-1][0];

}

}

}

**void** loop()

{

for (int i = 0; i < 10: ++i)

{

//Вычисление площади фигуры под графиком

//эталонных значений

Iobr += III[0][i];

Inorm += III[1][i];

Ipere += III[2][i];

Uobr += UUU[0][i];

Unorm += UUU[1][i];

Upere += UUU[2][i];

//Вычисление площади фигуры под графиком

//текущих значений

Inow += II[i][1] - II[i][0];

Unow += II[i][1] - II[i][0];

}

if (abs(Iobr - Inow + Uobr - Unow) > 1.5) Serial.print("\nObryv");

else if ((abs(Ipere - Inow + Upere - Unow) > 0.5) & (abs(Ipere - Inow + Upere - Unow) < 1.5)) Serial.print("\nPeregruzka");

else Serial.print("\nNormal");

**zero**();

delay(1000);

}

**Приложение Д**

**Листинг программы определения состояния работоспособности по уровню вибраций**

#include <Wire.h>

int16\_t accX;

int16\_t accY;

int16\_t accZ;

int16\_t XA;

int16\_t YA;

int16\_t ZA;

int16\_t Xprob; int16\_t Xnorm; int16\_t Xpok;

int16\_t Yprob; int16\_t Ynorm; int16\_t Ypok;

int16\_t Zprob; int16\_t Znorm; int16\_t Zpok;

int16\_t rX1; int16\_t rX2; int16\_t rX3;

int16\_t rY1; int16\_t rY2; int16\_t rY3;

int16\_t rZ1; int16\_t rZ2; int16\_t rZ3;

volatile float AX [10][2];

volatile float AY [10][2];

volatile float AZ [10][2];

volatile float XX [10][3] = {{43368.0, 43368,22868.6,43312.8,35739.2,14576.8,7268.0,35272.8,7096.0},

{35949.6,35949.6,13372.0,35535.2,35491.2,6787.2,6722.4,35226.4,6732.8},{39984.0,25782.4,17857.6,25760.0,19332.6,17855.2,24152.8,25755.2,17827.2}};

volatile float YY [10][3] = {{3889.6,3863.2,3612.8,3560.8,3483.2,3506.4,3412.8,3452.0,3401.6},

{3965.6,3788.0,3637.6,3648.8,3305.6,3132.0,3224.8,3308.8,3191.2},{288.0,280.0,262.4,258.4,274.4,252.0,254.4,256.0,244.0}};

volatile float ZZ [10][3] = {{5457.4,5308.8,5120.0,5075.2,4937.6,5023.2,4869.6,4868.8,4824.8},

{4272.0,4125.6,3922.4,4000.8,3766.4,3761.6,3513.6,3772.0,3659.2},{296.8,282.4,264.0,270.4,276.0,253.6,260.0,258.4,253.6}};

volatile uint8\_t s = 0;

uint32\_t tme0 = 0;

**void** setup()

{

digitalWrite(7,1);

digitalWrite(6,0);

digitalWrite(5,1);

Wire.begin();

Serial.begin(9600);

i2cWrite(0x6B,0x00);

//Таймер

OCR1A = 125;

TIMSK1 |= \_BV(OCIE0A);

}

SIGNAL(TIMER1\_COMPA\_vect) //Функция прерываний

{

TCNT1 = 0;

digitalWrite(13,!digitalRead(13));

if (s++ > 9) s = 0;

for (int i = 1; i <= 9; ++i)

{

if (s % i == 0)

{

AX[i-1][1] = (accX > AX[i-1][1]) ? accX : AX[i-1][1];

AX[i-1][0] = (accX < AX[i-1][0]) ? accX : AX[i-1][0];

AY[i-1][1] = (accY > AY[i-1][1]) ? accY : AY[i-1][1];

AY[i-1][0] = (accY < AY[i-1][0]) ? accY : AY[i-1][0];

AZ[i-1][1] = (accZ > AZ[i-1][1]) ? accZ : AZ[i-1][1];

AZ[i-1][0] = (accZ < AZ[i-1][0]) ? accZ : AZ[i-1][0];

}

}

}

**void** loop()

{

uint8\_t\* data = i2cRead(0x3B,14);

accZ = ((data[0] << 8) | data[1]);

accY = ((data[2] << 8) | data[3]);

accX = ((data[4] << 8) | data[5]);

for (int i = 0; i < 10; ++i)

{

//Вычисление площадей фигуры под графиком

//текущих значений координат

XA += AX[i][1] - AX[i][0];

YA += AY[i][1] - AY[i][0];

ZA += AZ[i][1] - AZ[i][0];

//Вычисление площадей фигуры под графиком

//эталонных координат

Xprob += XX[i][0]; Xnorm += XX[i][1]; Xpok += XX[i][2];

Yprob += YY[i][0]; Ynorm += YY[i][1]; Ypok += YY[i][2];

Zprob += ZZ[i][0]; Znorm += ZZ[i][1]; Zpok += ZZ[i][2];

}

rX1 = abs(XA - Xprob); rX2 = abs(XA - Xnorm); rX3 = abs(XA - Xpok);

rY1 = abs(YA - Yprob); rY2 = abs(YA - Ynorm); rY3 = abs(YA - Xpok);

rZ1 = abs(ZA - Zprob); rZ2 = abs(ZA - Znorm); rZ3 = abs(ZA - Zpok);

if ((rX1 + rY1 + rZ1) > 150) Serial.print("\nProbuksovka");

else if (((rX2 + rY2 + rZ2) > 150) & ((rX2 + rY2 + rZ2) < 50)) Serial.print("\nPokoi");

else Serial.print("\nNormal");

delay(1000);

if((millis() - tme0) > 1000)

{

tme0 = millis();

}

}

**void** i2cWrite(uint8\_t registerAddress, uint8\_t data)

{

uint8\_t IMUAddress = 0x68;

Wire.beginTransmission(IMUAddress);

Wire.write(registerAddress);

Wire.write(data);

Wire.endTransmission();

}

uint8\_t\* i2cRead(uint8\_t registerAddress, uint8\_t nbytes)

{

uint8\_t IMUAddress = 0x68;

uint8\_t data[nbytes];

Wire.beginTransmission(IMUAddress);

Wire.write(registerAddress);

Wire.endTransmission(false);

Wire.requestFrom(IMUAddress, nbytes);

for(uint8\_t i = 0; i < nbytes; i++)

{

data[i]= Wire.read();

}

**return** data;

}