МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ
Кафедра компьютерных технологий и систем

МЕЩЕНКО Егор Андреевич

РАЗРАБОТКА И ОПТИМИЗАЦИЯ АЛГОРИТМОВ ДВИЖЕНИЯ РОБОТА ПО ЛИНИИ В МОБИЛЬНОЙ РОБОТОТЕХНИКЕ

Дипломная работа

Научный руководитель:
кандидат педагогических наук,
доцент А.А. Францкевич

Допущена к защите
«____»_____20___г.
Заведующий кафедрой компьютерных технологий и систем
доктор педагогических наук, кандидат физико-
математических наук, профессор В.В. Казачёнок

Минск, 2025

ОГЛАВЛЕНИЕ

РЕФЕРАТ

Структура и объём дипломной работы

55 страниц, 18 рисунков, 3 приложение, 6 формул, 11 источников

Ключевые слова: ARDUINO, LEGO MINDSRTOM, C\C++, МОБИЛЬНЫЙ РОБОТ, ПИД РЕГУЛЯТОР, П РЕГУЛЯТОР, Д РЕГУЛЯТОР, OV7670, PYTHON, OPENCV, DEEPSEEK.

Текст реферата

Объект исследования – алгоритмы движения по линии в мобильной робототехнике, методы исследования и оптимизации алгоритмов, построение реальных практических условий на различных частично готовых решениях для мобильных роботов, алгоритмы распознавания заданной линии соответствующей будущей траектории мобильного робота.

Цель исследования – разработка нескольких вариантов мобильных роботов и тестирование различных алгоритмов, на базе разработанных роботов.

Методы исследования – изучение существующих мобильных роботов и алгоритмов, которые можно протестировать на реализованных роботах.

Полученные результаты – мобильный робот способный проходить маршрут, заданный линией, выделяющейся на фоне, в окружающей среде. Его главное отличие в уникальности и способности приспосабливаться к различным условиям. Также у нас получились несколько алгоритмов, протестированных на мобильном роботе.

Область возможного практического применения – областью использования может быть образование, логистика, развлекательные цели. В логистических целях, мобильных роботов можно использовать на складах, как это делает Amazon. В образовательных целях, данный робот может выступать в качестве ассистента на котором можно тестировать свои алгоритмы.

РЭФЕРАТ

Структура і аб’ём дыпломнай працы

55 старонок, 18 малюнкаў, 3 дадатка, 6 формул, 11 крыніц

Ключавыя словы: ARDUINO, LEGO MINDSRTOM, C\C++, МАБІЛЬНЫ РОБАТ, ПІД РЭГУЛЯТАР, П РЭГУЛЯТАР, Д РЭГУЛЯТАР, OV7670, PYTHON, OPENCV, DEEPSEEK.

Змест працы

Аб'ект даследавання - алгарытмы руху па лініі ў мабільнай робататэхніцы, метады даследавання і аптымізацыі алгарытмаў, пабудова рэальных практычных умоў на розных часткова гатовых рашэннях для мабільных робатаў, алгарытмы распазнання зададзенай лініі адпаведнай будучай траекторыі мабільнага робата.

Мэта даследавання – распрацоўка некалькіх варыянтаў мабільных робатаў і тэсціраванне розных алгарытмаў, на базе распрацаваных робатаў.

Метады даследавання - вывучэнне існуючых мабільных робатаў і алгарытмаў, якія можна пратэставаць на рэалізаваных робатах.

Атрыманыя вынікі - мабільны робат здольны праходзіць маршрут, зададзены лініяй, якая вылучаецца на фоне, у навакольным асяроддзі. Яго галоўнае адрозненне ва ўнікальнасці і здольнасці прыстасоўвацца да розных умоў. Таксама ў нас атрымаліся некалькі алгарытмаў, пратэставаных на мабільным робаце.

Вобласць магчымага практычнага прымянення - вобласцю выкарыстання можа быць адукацыя, лагістыка, забаўляльныя мэты. У лагістычных мэтах, мабільных робатаў можна выкарыстоўваць на складах, як гэта робіць Amazon. У адукацыйных мэтах, дадзены робат можа выступаць у якасці асістэнта на якім можна тэставаць свае алгарытмы.

SUMMARY

Structure and scope of the diploma work

55 pages, 18 figures, 3 appendixes, 6 formulas, 11 reference

Keywords: ARDUINO, LEGO MINDSRTOM, C\C++, MOBILE ROBOT, PID REGULATOR, P REGULATOR, D REGULATOR, OV7670, PYTHON, OPENCV, DEEPSEEK.

Summary text

The object of the research – line driving algorithms in mobile robotics, methods of research and optimization of algorithms, construction of real practical conditions on various partially ready solutions for mobile robots, algorithms for recognition of a given line corresponding to the future trajectory of a mobile robot.

The aim of the research – to develop several variants of mobile robots and testing different algorithms, based on the developed robots.

Research methods – study of existing mobile robots and algorithms that can be tested on implemented robots.

The results obtained – a mobile robot capable of taking a route given by a line that stands out in the background, in the environment. Its main difference is its uniqueness and ability to adapt to different conditions. We also have several algorithms tested on the mobile robot.

Area of possible practical application – the area of use can be education, logistics, entertainment purposes. For logistical purposes, mobile robots can be used in warehouses, as Amazon does. For educational purposes, the robot can act as an assistant on which you can test your algorithms.

ВВЕДЕНИЕ

50 лет назад, изучение мобильных роботов являлось очень дорогим занятием, которое могли позволить себе лишь лучшие университеты мира. Так в 1970 году в Стэнфордском университете под руководством математика Дж. Маккарти появился мобильный робот, известный как Стэнфордская тележка. Робота снабдили независимыми электроприводами на каждое колесо. Робот мог ориентироваться в пространстве 4 способами. Он был оснащен дальномером, чтобы определять расстояния до препятствия, камеру для различных взаимодействий с окружающей средой, еще у него были датчики осязаний, при соприкосновении с которыми, робот останавливался и пытался объехать препятствие, еще у него была система навигации, подсчитывающая пройденное расстояние. В 1979 году робот был усовершенствован и теперь он мог перемещаться по заданной белой линии. Сейчас же создать своего робота, способного выполнять простейшие команды, может любой человек, имеющий доступ в сеть Интернет и пару долларов на своем счету.

В данной дипломной работе будут рассмотрены различные алгоритмы движения робота по линии, с использованием существующих способов определения заданной линии в пространстве. Для этого в рассмотрение будет введена различная литература, а также будут представлены определения, соответствующие тематике дипломной работы.

ГЛАВА 1
ОБЗОР ЛИТЕРАТУРЫ И ИССЛЕДОВАНИЕ СУЩЕСТВУЮЩИХ РЕШЕНИЙ

1.1. Основные понятия и определения в области мобильной робототехники и алгоритмов движения по линии, их классификация

В области мобильной робототехники и алгоритмов движения по линии введем несколько ключевых понятий и определений.

Мобильный робот – это автономное или управляемое устройство, способное передвигаться в окружающей среде. Такой робот обычно имеет колеса либо гусеницы, в некоторых случаях имеет конструкцию шагохода. Мобильный робот обычно выполняет действия согласно интегрированной в него базы знаний.

Движение по линии – это процесс перемещения и ориентации мобильного робота вдоль определенной линии на поверхности, заданной ранее.

Датчики распознания линии – это устройства, предназначенные для определения положения робота относительно линии. Обычно используют следующие датчики:

1. Оптический датчик. Их еще называют фотоэлектрическим датчиком, бесконтактным датчиком физической величины. Оптический датчик состоит из приемника и источника светового излучения. Вообще такую структуру имеют большинство приборов измерения. Источник же светового излучения у оптического датчика состоит из следующих компонентов:

– Корпуса, отвечающего за защиту элементов конструкции излучателя, а также предупреждает о повреждениях, возникающих из-за механического воздействия, изготавливается из латуни или полиамида.

– Генератора, формирующего электрические импульсы, поступающие на излучатель.

– Излучателя, представленного в виде компактного светодиодного механизма, который испускает световой поток в заданном диапазоне.

– Системы оптики, отвечающей за направление, в котором испускается световое излучение.

– Индикатора, отвечающего за готовность датчика к работе. Приемник состоит из следующих компонентов:

– Оптики, отвечающей за прием и передачу светового луча к преобразователю.

– Фотоприемника, трансформирующего световое излучение в электрический сигнал.

– Усилителя, увеличивающего интенсивность сигнала до значения, которое может обработать прибор.

– Порогового элемента, регулятор крутизны фронта сигнала переключения.

– Электронного ключа, предупреждающего о возникновении коротких замыкании и перегрузок.

– Индикатора цвета, указывающего, в зависимости от цвета, состояние датчика. Например: отсутствие свечения – означает что сигнала нет, зеленый цвет – прибор активирован после получения сигнала, интенсивность которого соответствует заданным параметрам, желтый и красный – показывают увеличение уровня сигнала.

Датчик такого вида улавливает электромагнитные волны от излучателя в видимом, инфракрасном и ультрафиолетовом диапазонах, после чего оптический сигнал преобразуется в электрический. Применяется он для определения перемещения, положения, текстуры объекта, для контроля температуры, измерения расстояния. Также такой датчик способен реагировать на водяной пар, дым, непрозрачные и полупрозрачные объекты.

2. Инфракрасные датчики. Датчик (детектор) движения – инфракрасный (тепловой) датчик, который обнаруживает перемещение живых объектов и управляет освещением. В датчике движения используется в качестве сенсора пироэлектрический датчик, принцип работы которого основан на повышении напряжения на его выходе при повышении уровня инфракрасного излучения по сравнению с фоновым. Такой датчик состоит из двух элементов:

– Светодиода, служащего источником инфракрасного излучения.

– Фотодиода, обладающего высокой чувствительностью

Инфракрасные датчики подразделяются на разные типы в зависимости от применения и, таким образом, используются в некоторых типичных приложениях. Например, датчики скорости используются для синхронизации скорости нескольких двигателей; датчики температуры используются для промышленного контроля температуры; Датчики используются для систем автоматического открывания дверей; ультразвуковые датчики используются для измерения расстояния.

Инфракрасные датчики используются в различных сенсорных проектах, а также используются для измерения температуры различных электронных устройств.

В нашем случае, мы будем использовать такой для определения линии. Обычно черный цвет поглощает все падающее на него излучение, а белый отражает все падающее на него излучение. На основе этого принципа может быть выполнено вторичное позиционирование пары датчиков. Инфракрасные светодиоды и фотодиоды расположены рядом. Когда ИК-передатчик излучает ИК-излучение, поскольку между передатчиком и приемником нет прямой линии контакта, испускаемое излучение должно отражаться обратно на фотодиод после столкновения с чем-либо. Поверхность объекта можно разделить на два типа: отражающая поверхность и неотражающая поверхность. Если поверхность объекта имеет отражающую природу, т. е. она белого или другого светлого цвета, то большая часть падающего на нее излучения будет отражаться обратно и достигать фотодиода.

В зависимости от интенсивности отраженного назад излучения, если поверхность предмета по своей природе неотражающая, т. е. черная или иная темная, она будет поглощать почти все падающее на нее излучение. Поскольку отраженного излучения нет, на фотодиод не падает излучение, и сопротивление фотодиода остается высоким, не позволяя току течь. Эта ситуация аналогична полному отсутствию объектов.

3. Проводящие датчики. Обычные лабораторные измерители электропроводности используют потенциометрический метод и четыре электрода. Часто электроды имеют цилиндрическую форму и расположены концентрически. Электроды обычно изготавливаются из платинового металла. К внешней паре электродов подается переменный ток. Измеряется потенциал между внутренней парой. Проводимость в принципе можно определить, используя расстояние между электродами и площадь их поверхности, используя закон Ома, но, как правило, для точности используется калибровка с использованием электролитов с хорошо известной проводимостью.

Промышленные датчики электропроводности часто используют индуктивный метод, преимущество которого заключается в том, что жидкость не смачивает электрические части датчика. Здесь используются две катушки с индуктивной связью. Один из них – это катушка возбуждения, создающая магнитное поле, и на него подается точно известное напряжение. Другой образует вторичную катушку трансформатора. Жидкость, проходящая через канал в датчике, образует один виток во вторичной обмотке трансформатора. Индуцированный ток является выходным сигналом датчика.

Другой способ – использовать четырехэлектродные датчики электропроводности, изготовленные из коррозионностойких материалов. Преимуществом четырехэлектродных датчиков проводимости по сравнению с

индуктивными датчиками является компенсация масштабирования и возможность измерения низкой (ниже 100 МКС / см) проводимости (функция, особенно важная при измерении содержания плавиковой кислоты, близкого к 100%)..

4. Камеры. Такие датчики используют для получения и обработки изображения окружающей среды. Обычно мобильный робот на основе своего обучения сам определяет окружающую среду и может двигаться по различным типам линий, заданных различными способами.

5. Ультразвуковые датчики. Ультразвуковой датчик применяется для обнаружения и определения расстояния до объекта, а также для контроля их движения. Передатчик излучает звуковые колебания, частота которых превышает 20 кГц. Они в виде волн «прошивают» пространство, и, встречаясь с твердыми предметами, отражаются от них и попадают в приемник датчика. Электронная схема подсчитывает расстояние до объекта согласно следующей формуле (1.1):

$$R = \frac{V}{2} \times t, \tag{1.1}$$

где R – искомое расстояние,

t – промежуток времени между отправкой и приемом ультразвуковой волны.

V – Скорость звука.

В конструкции излучателя современных датчиков используются следующие типы компонентов:

- Магнитострикционные – ультразвуковые колебания формируются за счет быстрого изменения размеров ферромагнетника, размещенного в переменном магнитном поле. Его плюсы: долгий срок службы (ресурс не меньше 10 тысяч часов) и высокий КПД, достигающий 80%. Есть и минусы в виде достаточно сложного устройства и быстрого нагрева, из-за чего нужно водяное охлаждение.

- Пьезоэлектрические – гораздо проще в строении, если сравнивать с предыдущим видом, так как волны формируются в процессе быстрого изменения размеров мембраны в переменном электрическом поле. Сама мембрана изготовлена из диэлектрического материала. Также такие передатчики отличаются компактностью, небольшим весом и возможностью излучение ультразвука разной частоты. Существенный минус один – достаточно низкая мощность.

В большинстве датчиков стоят пьезоэлектрические излучатели. Приемник работает благодаря аналогичному эффекту, который действует в обратном направлении. Когда мембрана начинает колебаться под влиянием отраженного ультразвука, в окружающем ее поле появляется ток.

Датчики, приведенные выше, могут использоваться для определения линии в пространстве. Каждый такой датчик имеет свои плюсы и минусы. Поэтому их выбирают в соответствии с поставленной задачей. Чаще эти датчики комбинируют друг с другом, для повышения точности работы устройства, использующего их.

Алгоритмы движения по линии – это методы и процессы, которые позволяют роботу следовать за линией. Они могут быть основаны на различных принципах, таких как пропорционально-интегрально-дифференциальное управление, нейронные сети, и др.

Алгоритмы движения по линии можно разбить на условных 4 класса:

– Алгоритмы машинного обучения. Такие алгоритмы включают в себя использование нейронных сетей, классификаторов и много другого. Так же таким роботам дают возможность с дальнейшим самообучением для адаптации к другим окружающим средам.

– Детерминированные и стохастические алгоритмы. Детерминированные алгоритмы четко прописаны и включают в себя строго определенные шаги и условия для движения по линии. Стохастические же в свою очередь могут учитывать любые случайные факторы или неопределенности.

– Алгоритмы на основе обратной связи. Такие алгоритмы используют информацию об отклонениях от желаемой траектории для корректировки движения и удержания робота на линии.

– Алгоритмы оптического потока. Такие алгоритмы используют информацию об изменении положения пикселей на изображении для определения действий, происходящих в окружающей среде, для корректировки маршрута и удержания робота на линии.

1.2. Исследование необходимых источников по мобильной робототехнике и алгоритмам движения по линии

Для начала рассмотрим литературу по мобильной робототехнике и алгоритмам движения по линии.

Известнейший источник в данном направлении: Овсяницкая Л. Ю., Овсяницкий Д.Н., Овсяницкий А. Д. Алгоритмы и программы движения робота по линии Lego Mindstorms EV3 – 2015.

В данном источнике описаны основные аспекты разработки личного мобильного робота на базе Lego Mindstorm EV3, с использованием различных датчиков. Автор приводит примеры реализации алгоритмов движения робота по линии с использованием датчиков цвета EV3 и NXT, а также описывает плюсы и минусы их использования. В источнике можно узнать о различных регуляторах, о том, как они работают и зачем они нужны.

Автор описывает различные вариации расположения датчиков на роботе, с указанием недостатков каждого выбора. Вот 5 вариантов расположения датчиков из источника:

- Один датчик (рисунок 1.1.а), расположенный с любой стороны линии. В этом случае робот может двигаться вдоль линии, совершая повороты. Имея один датчик можно реализовать многие алгоритмы, в том числе и ПИД-регулятор. Наличие одного датчика не дает возможности для определения перекрестков и инверсий, есть шанс потерять линию.

- Два датчика (рисунок 1.1.б), расположенных по обе стороны от линии. В этом случае робот почти не теряет линии и уже может определять перекрестки, еще робот начинает двигаться плавно.

- Три датчика (рисунок 1.1.в), два - расположены с обеих сторон линии, третий вынесен в сторону и немного вперед, в этом случае робот может определять и подготавливаться к поворотам на острый угол, находить какие-либо сторонние препятствия, определять линии штрих-кода и подсчитывать перекрестки.

- Три датчика (рисунок 1.1.г), два с обеих сторон линии, а третий вынесен немного вперед и находится над линией. Такая конструкция уступает по функциональности предыдущей, но в этом случае мы можем определять сложные участки трассы и принимать какие-либо меры по прохождению их.

- Четыре датчика (рисунок 1.1.д), данное количество датчиков обычно используют для скоростных алгоритмов прохождения очень сложных трасс с насыщенными препятствиями различного рода.

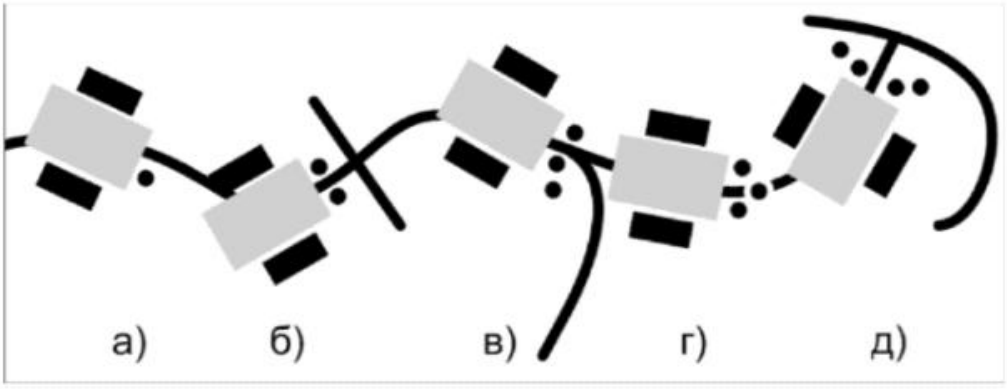

Рисунок 1.1 – Виды расположения датчиков на роботе

Таким образом при создании своего робота на базе Lego Mindstorm EV3, опираясь на этот источник, мы можем построить своего робота и создать алгоритм для его движения по линии, с учетом поставленной нам ранее задачи [1].

Для решения поставленной задачи нужно рассмотреть различные алгоритмы, интереснейшим из этих направлений является определение траектории по линии с использованием камеры, как правило веб-камеры. Один из источников в котором об этом рассказывается, является «Алгоритм автоматического удержания колесного робота на визуально заданной линии – 2016» написанный Сотникова М.В.

Данный источник предлагает рассмотреть задачу синтеза управления, обеспечивающего автоматическое удержание полноприводного колесного робота на визуально заданной линии. Для ее решения предлагается использовать камеру, относительно которой возникает два случая. Первый, когда камера не имеет шести степеней свободы и не может свободно перемещаться в пространстве конфигураций. Второй, когда камера имеет шесть степеней свободы и может свободно перемещаться в пространстве конфигураций.

По ходу решения поставленной задачи применяются различные интересные решения, такие как использование виртуальной камеры, отслеживание изменения нормы отклонения от заданного маршрута. Приведены решения этой задачи на различных ее этапах.

По итогу мы получаем алгоритм управления, базирующийся на использовании многоцелевой структуры, обеспечивающий достижение цели управления с компенсацией постоянных возмущений [5].

Для поиска общего решения нужно обобщенное решение по управлению и конфигурации мобильного робота, способного распознавать линию. Одним

из интереснейших источников в данном направлении является Aleena Johny, Lentin Joseph. Robot Operating System (ROS) for Absolute Beginners.

Автор начинает повествование с введения в мир мобильной и сервисной робототехники, где ROS (Robot Operating System) является каркасом (фреймворком) для создания, тестирования и развертывания программ для роботов. Автор описывает зачем нужен ROS и почему он стал стандартом в индустрии, поясняет основные принципы ROS: узлы, топики, сервисы, сообщения и параметрическое управление. Автор дает основы Linux и Python для полного понимания описываемой информации и подготовки читателя. Источник ориентирован на новичков хоть как-то знакомых с программированием. Автор советует использовать Ubuntu и ROS желательно версии Kinetic (подойдут и более ранние версии), и предлагает установку ROS с нуля, включая настройку среды разработки.

Рассмотрим по подробнее описанные в источнике фундаментальные концепции ROS. Про узлы рассказывается как об отдельных исполняемых модулях, которые обмениваются данными. В источнике демонстрируется, как писать простые узлы на Python. Про топики и сообщения в источнике рассказывается с точки зрения pub/sub механизма. Автор рассказывает о создании собственных типов сообщений. Примерная схема работы pub\sub модели: узел публикует координаты робота, второй — подписывается и визуализирует их. Про сервисы и действия автор рассказывает с точки зрения сравнения одноразового обмена сервисов и длительных операций действий. Пример, показанный в источнике: робот может получить команду "поверни" через сервис, или "двигайся до цели" через действие. В этой части упор сделан на понимание архитектуры ROS, а не только на код. Примеры просты и эффективны — без сложной математики.

В источнике приводится пример реализации систем через имуляцию в Gazebo. Про Gazebo имеется простое объяснение как запускать симуляцию мобильного робота, как подключить различные плагины, как визуализировать с RViz и написать скрипты управления движением. Так же имеются примеры проектов Line-following robot — модель с камерой, визуальной навигацией. Obstacle avoidance robot — с использованием ультразвукового или лазерного сканера. Voice-controlled robot — с использованием микрофона и Google Speech API [10].

Последним источником стал «Michael Margolis. Make an Arduino-Controlled Robot».

Источник «Make an Arduino-Controlled Robot» — это крутое чтиво для всех, кто хочет построить собственного мобильного робота, используя

Arduino. Автор организовал источник таким образом, чтобы даже начинающий робототехник любитель смог собрать, запрограммировать и протестировать полноценного мобильного робота.

Основная платформа, на которой базируется большинство проектов в источнике, — это Arduino Uno и Leonardo, совместно с моторным шилдом Adafruit Motor Shield v1. Также используются стандартные компоненты: колеса, шасси, сервомотор, ИК-датчики, ультразвуковые датчики, батарейный отсек, переключатели. В источнике приводится два варианта базового робота: двухколесный и четырехколесный. Первый — маневренный, второй — устойчивый. Скетчи приводимые автором написаны на стандартном Arduino C/C++. Автор подробно объясняет, как установить среду разработки, подключить плату, загрузить код. Примеры кода идут с комментариями, но без глубокого погружения в синтаксис [11].

1.3. Исследование существующих алгоритмов движения по линии и мобильных роботов, использующих их. Выбор подходящих составляющих

В мире существует множество различных алгоритмов в зависимости от того, что использует робот для взаимодействия с внешней средой. Например, если у робота есть камера, то он способен ездить по различным ориентирам с достаточно высокой точностью. Если же робот наделен датчиками линии, то он может использовать различные регуляторы в зависимости от того сколько датчиков он использует. Например, мы можем установить на робота 4 и более датчиков, благодаря чему мы сможем тестировать алгоритмы на одном датчике, двух, трех и тд. В мире выделяют 5 видов регуляторов, сейчас мы рассмотрим концепции таких алгоритмов и их применение на различных мобильных роботах.

1. Релейный регулятор.

Описание: Рассмотрим на примере двухпозиционного релейного регулятора. Для реализации такого алгоритма нам нужно знать значение датчиков линии на светлом участке (значение white или 0) и на темном участке (значение dark или 1), далее рассчитываем значение «серого» (значение gray, между dark и white), для этого возьмем значение white, прибавим к нему значение dark и умножим на 0.5, тем самым получив значение gray. Алгоритм называется двухпозиционным, потому что у него есть два состояния, это когда значение датчика линии меньше значения gray, и когда больше. Тогда в

зависимости от состояния робот может менять скорость либо на правом двигателе, либо левом. Таков алгоритм нестабилен, и робот зачастую теряет линию из-за того, что не способен двигаться плавно.

Преимущества:

- Простота конструкции

Релейные регуляторы легко реализуются как аппаратно, так и программно, что делает их экономически выгодными.

- Надежность

Простая конструкция обеспечивает высокую устойчивость к отказам, так как меньше компонентов означает меньше точек отказа.

- Отсутствие необходимости в точной математической модели системы

Работает без сложного анализа объекта управления. Достаточно иметь лишь знание о граничных условиях и пороговых значениях.

- Быстрая реакция на значительные отклонения

Мгновенное переключение между состояниями позволяет быстро реагировать на изменения в системе.

- Независимость от малых возмущений

Малые колебания входного сигнала игнорируются, что повышает стабильность.

Недостатки:

- Высокая нестабильность вблизи установившегося значения.

Из-за переключения между состояниями может возникать колебательный режим (постоянное включение/выключение).

- Невозможность плавного регулирования.

Регулятор работает дискретно (вкл/выкл), что делает его менее подходящим для задач, где требуется точное управление.

- Износ реле или исполнительных механизмов.

Постоянное переключение может приводить к быстрому механическому износу или электрическим повреждениям.

- Ограничение применения.

В высокоинерционных системах релейное управление может быть неэффективным или даже вредным. Подходит только для систем с низкой инерционностью и где допустимы скачки управления.

- Энергозатраты.

В некоторых случаях частое переключение может приводить к увеличению энергопотребления.

- Частотная нестабильность.

Частота колебаний, возникающих из-за релейного управления, зависит от параметров системы, что может вызывать нежелательные резонансы.

2. Пропорциональный регулятор.

Описание: Основан на П-пропорциональной составляющей ПИД регулятора. Эта составляющая контролирует управление роботом, за счет силы воздействия на двифгатели(увеличение скорости вращения одного и уменьшения скорости вращения другого) пропорционально силе отклонения робота от линии. Скорость на двигателях рассчитывается относительно параметра, отвечающего за отклонение от линии, умноженного не постоянный параметр Кп, который подбирается в зависимости от окружающей среды. Такой алгоритм позволит плавно ехать роботу по плавно изгибающейся линии, но в случае резкого поворота, могут быть проблемы.

Преимущества:

- Простота реализации

Легко проектируется и реализуется как аппаратно, так и программно.

- Быстрая реакция на изменения ошибки

Регулятор быстро реагирует на изменения входного сигнала и уменьшает ошибку.

- Низкая стоимость

Ввиду своей простоты, П-регуляторы являются экономически эффективным решением.

- Эффективность в системах с малой инерцией

В системах, где отсутствует большая задержка, пропорциональный регулятор способен поддерживать стабильное управление.

- Широкое применение

Применяется во многих областях автоматического управления (электротехника, HVAC-системы, простые роботы).

Недостатки:

- Наличие статической ошибки

П-регулятор не способен полностью устранить ошибку в установившемся режиме, особенно при внешних возмущениях или изменении нагрузки.

- Чувствительность к коэффициенту усиления (K_p)

Малое значение K_p приводит к медленному реагированию регулятора, а слишком большое значение может вызвать перерегулирование и колебания.

- Нестабильность в системах с высокой инерцией

В системах с запаздыванием или большой инерционностью П-регулятор может вызвать неустойчивость и колебания.

- Перерегулирование

Быстрое реагирование на ошибки может привести к перерегулированию, что ухудшает точность и устойчивость системы.

- Ограниченные возможности для точного управления

В сложных системах, где требуется высокая точность и плавность управления, одного П-регулятора недостаточно.

3. Пропорционально-интегральный регулятор.

Описание: Используется для решения проблемы предыдущего регулятора, а именно если у линии есть долгий прямой участок а потом резкий поворот, то робот не успеет на него среагировать. Поэтому было принято решение добавить интегральную составляющую в формулу. По сути, мы используем ту же формулу для нахождения скорости вращения двигателей, только мы добавляем еще интеграл по времени ошибки умноженный на неотрицательный коэффициент Ки, который так же нужно подбирать.

Преимущества:

- Устранение статической ошибки

Интегральная составляющая компенсирует постоянные ошибки, которые остаются при использовании только пропорционального регулятора.

- Улучшенная точность управления

Благодаря интегральной части система достигает более точного установившегося состояния.

- Простота реализации

ПИ-регулятор относительно прост в реализации и настройке по сравнению с ПИД-регулятором.

- Подходит для систем с инерцией

Более эффективно работает в системах с запаздыванием или инерционностью.

Недостатки:

- Подходит для систем с инерцией

Интегральная составляющая реагирует на накопленную ошибку, что приводит к замедлению реакции регулятора.

- Перерегулирование

При неправильной настройке коэффициентов (K_p, K_i) может возникнуть перерегулирование и колебания.

- Чувствительность к настройке коэффициентов

Сложность выбора оптимальных значений K_p и K_i для разных систем.

- Интегральное насыщение

При длительном воздействии ошибки интегральная часть может накапливать избыточное значение (интегральный перегруз), что вызывает затруднение при возвращении системы в нормальный режим.

4. Пропорционально-Дифференциальный регулятор.

Описание: Этот регулятор возник для решения проблемы П регулятора, заключающейся в том, что робот при движении по прямой(слабо извилистой дороге) при использовании П регулятора с маленьким параметром Кп будет ехать ровно, но не сможет среагировать на резкий поворот, а с большим параметром Кп сможет но на ровной линии он будет ехать виляя, что так же может привести к потере линии. Поэтому было принято добавить новое слагаемое Кд. Вычисляемое по средствам вычитания из значения предыдущей ошибки текущей ошибки.

Преимущества:

- Уменьшение перерегулирования

Дифференциальная составляющая замедляет реакцию регулятора при резких изменениях, предотвращая избыточное управление и колебания.

- Быстрое реагирование на изменения

Пропорциональная часть обеспечивает оперативное воздействие на ошибку, а дифференциальная часть помогает компенсировать быстрые изменения.

- Улучшение динамики системы

Снижается время переходного процесса и улучшается устойчивость системы.

- Подходит для систем с малой инерцией

Особенно эффективен в системах, где ошибки меняются быстро и требуется предсказание поведения системы.

Недостатки:

- Не устраняет статическую ошибку

Как и П-регулятор, ПД-регулятор не способен компенсировать постоянное смещение регулируемой величины от заданного значения.

- Чувствительность к шумам

Производная ошибки усиливает высокочастотные помехи, что может привести к нестабильности системы.

- Сложность настройки коэффициентов

Неправильный выбор K_p и K_d может ухудшить стабильность или динамику системы.

- Ограниченность применения

Неэффективен в системах с большими статическими возмущениями, где требуется интегральная составляющая для устранения постоянной ошибки.

3. Пропорционально-интегрально-дифференцирующий регулятор.

Описание: В алгоритме такого регулятора используются основные концепции всех предыдущих регуляторов. Такой регулятор управляет скоростью вращения двигателей, прибавляя параметр u к стандартной средней скорости одного двигателя и вычитая параметр u от стандартной средней скорости другого двигателя. Такой параметр u рассчитывают по особой формуле (1.2). Такой регулятор позволяет минимизировать шанс потери линии, позволяет плавно проезжать маршрут, определенный линией, успевать реагировать на резкие повороты.

$$u(t) = P + I + D = K_p \cdot e(t) + K_i \cdot \int_0^t e(t)\,dt + K_d \cdot \frac{de}{dt}. \qquad (1.2)$$

Преимущества: ограниченность применения, применяется в большинстве систем автоматического управления: от простых до сложных. Точность управления, позволяет добиться минимальной ошибки за счет интегральной составляющей. Улучшение динамических характеристик, дифференциальная составляющая уменьшает перерегулирование и колебания, улучшая переходные процессы. Стабильность при правильной настройке коэффициентов ПИД-регулятор обеспечивает устойчивость системы даже в сложных условиях. Полное устранение статической ошибки, интегральная часть компенсирует постоянные ошибки в установившемся режиме.

Недостатки: сложность настройки, точная настройка коэффициентов (K_p, K_i, K_d) требует знаний о системе и может быть трудоемкой. Чувствительность к шумам, дифференциальная часть усиливает шумы и высокочастотные помехи, что может привести к нестабильности. Интегральное насыщение, интегральная составляющая может накапливать избыточное значение при больших ошибках (интегральный перегруз), вызывая медленное возвращение системы в норму. Не всегда эффективен для систем с большим запаздыванием

В таких системах сложнее настроить параметры для устойчивого управления.

Рассмотрим по подробнее формулу (1.2) и как с ней работать. В данной формуле используются такие составляющие как:

$u(t)$ – наша функция, выходной результат;

P – пропорциональная составляющая;

I – интегральная составляющая;

D – дифференцирующая составляющая;

$e(t)$ – текущая ошибка;

K_p – пропорциональный коэффициент;

K_i – интегральный коэффициент;

K_d – дифференцирующий коэффициент.

Настраивая, указанные выше, параметры мы буем получать то или иное поведение робота, сейчас по подробнее рассмотрим настройку всех параметров.

Есть два похода к настройке ПИД регулятора. Первый – синтез регулятора, то есть вычисление параметров регулятора на основании модели системы. Данный метод позволяет очень точно рассчитать параметры регулятора, но он требует основательного погружения в ТАУ. Второй метод – ручной подбор параметров (коэффициентов). Это метод научных проб и ошибок. Берем готовую систему, меняем один (или сразу несколько коэффициентов) регулятора, включаем регулятор и смотрим за работой системы. В зависимости от того, как ведет себя система с выбранными коэффициентами (недо/пере регулирование) опять меняем коэффициенты и повторяем эксперимент. Такой метод имеет право на жизнь, главное представлять как изменение того или иного коэффициента повлияет на систему (что бы не действовать совсем наугад).

Есть более «оптимизированный» метод подбора коэффициентов – метод Зиглера–Никольса. Метод работает не для любой системы, результаты получаются не самыми оптимальными. Но, зато, метод очень простой и годится для базовой настройки регулятора в большинстве систем. Суть метода состоит в следующем:

1. Выставляем все коэффициенты (K_d, K_i, K_d) в 0.

2. Начинаем постепенно увеличивать значение K_p следим за реакцией системы. Нам нужно добиться, чтобы в системе начались устойчивые колебания (вызванные перерегулированием). Увеличиваем K_p, пока колебания системы не стабилизируются (перестанут затухать).

3. Запоминаем текущее значение K_p (обозначим его K_u) и замеряем период колебаний системы (T_u).

Все, теперь используем полученные значения K_u и T_u для расчета всех параметров ПИД регулятора по формулам (1.3-1.5):

$$K_p = 0.6 \cdot K_u, \tag{1.3}$$

$$K_i = 2 \cdot \frac{K_p}{T_u}, \tag{1.4}$$

$$K_d = K_p \cdot \frac{T_u}{8}. \tag{1.5}$$

Для дискретных регуляторов нужно еще учесть период дискретизации – T [2](умножить на K_i та T , разделить K_d на T).

Для реализации таких алгоритмов можно использовать роботов укомплектованных 1-4 датчиками линии и двумя двигателями.

Рассмотрим оснащенного всем необходимым робота на базе Arduino uno. Программа по управлению таким роботом выглядит следующим образом:

```
a=digitalRead(A2);
b=digitalRead(A1);
if(a==1){
        analogWrite(enA, 100); //Left Motor Speed
        analogWrite(enB, 0);
}else if(b == 1){
        analogWrite(enA, 0); //Left Motor Speed
        analogWrite(enB, 100);
 }
 // Скорость двигателя не должна превышать максимальное значение ШИМ
 //Right Motor Speed
 digitalWrite(in1, HIGH);
 digitalWrite(in2, LOW);
 digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
```

Такой робот (рисунок 1.2) отлично проходит линию не сбиваясь, но очень сильно привязан к состоянию питания, в моем случае это крона, когда она разряжается, двигатели перестают перемещать робота, а датчики дают сбой. Программа для движения по линии использует концепцию двухпозиционного релейного регулятора.

Рисунок 1.2 – Робот реализованный на Arduino Uno

Далее рассмотрим Мобильного робота на базе Lego Mindstorm EV3. Данная база для робота хороша своей простотой и понятностью в использовании, а также наличием большого количества качественной информации в сети Интернет. Так же Lego Mindstorm EV3 имеет удобную и понятную среду для написания кода, для этого используются различные цветные блоки. Рассмотрим робота, использующего два цветных датчика EV3. Данный мобильный робот прост в создании и для него не сложно составить простой алгоритм (рисунок 1.3.).

Рисунок 1.3 – Алгоритм демонстрирующий работу на двух цветовых датчиках

При необходимости можно посчитать количество перекрестков, так же этот робот с очень низкой вероятностью может потерять линию.

Рисунок 1.4 – Собранный мобильный робот с двумя датчиками

4. Логика нечетких множеств.

Описание: это расширение классической бинарной логики, разработанное для работы с неопределенностью и нечеткими данными. Она позволяет моделировать ситуации, в которых границы между понятиями размыты или неопределенны.

Преимущества: работа с неопределенностью и размытостью, позволяет описывать реальные ситуации, в которых границы между понятиями нечеткие (например, "тепло", "холодно", "быстро", "медленно"), логика нечетких множеств гибко адаптируется под задачи, где невозможно применить строгие математические модели, позволяет переводить неформальные рассуждения человека в математическую модель. Например, "высокая температура" или "средняя скорость". Нечеткая логика может быть легко объединена с традиционными методами управления или алгоритмами машинного обучения. Широко применяется для управления системами, где невозможно создать точную математическую модель (например, климат-контроль, бытовая техника, робототехника). Функции принадлежности и правила нечеткой логики интуитивно понятны и легко интерпретируются.

Недостатки: настройка функций принадлежности и нечетких правил требует глубокого понимания предметной области, часто зависящего от человеческого опыта. Нет строгих математических методов для точной настройки параметров (например, функций принадлежности и весов). При большом количестве переменных и правил система может стать сложной и ресурсоемкой, что приводит к трудностям в обработке. Некорректный выбор функций принадлежности может значительно ухудшить качество модели. Необходимость аппроксимации, точные результаты, как в классической логике, получить невозможно. Результаты всегда являются приближенными. Проблемы с высокоточным управлением, в системах, требующих высокой точности и строгих границ, нечеткая логика может быть менее эффективной, чем традиционные методы. Низкая скорость при больших объемах данных, системы на основе нечеткой логики могут работать медленнее при обработке больших массивов данных.

5. Методы машинного обучения

Описание: Машинное обучение позволяет роботу обучаться следованию линии на основе данных с датчиков (например, изображений с камеры).

Преимущества: способность адаптироваться к сложным условиям (например, изгибы, разрывы линии), эффективно для работы с визуальными данными.

Недостатки: высокие вычислительные затраты, необходимость в обучающих данных и настройке модели, сложность реализации по сравнению с традиционными методами.

Для разработки решений поставленных задач были изучены способы обработки изображения бортовым вычислителем робота, в частности, следующие модули:

1. OV7670.

Камера для записи и передачи картинки (например: VGA с частотой в 30 кадров в секунду размера 640 x 480, но можно получать и в другом формате: QVGA размера 320 x 240, CIF размера 352 x 240, QCIF размера 176 x 144). Модуль рассчитан на входные 3В, поэтому нужно будет установить преобразователи. Данные пикселя такой камеры хранит информацию о его цвете по 8 разрядному параллельному интерфейсу. Если мы будем использовать кодировку RGB, то данные о пикселе будут занимать 2 байта. Данный модуль имеет кучу настроек, среди них можно выбрать формат кадра. RGB кодировка представлена в трех вариантах RGB565, RGB555 и RGB444, цифры означают количество бит на цвет.

Камера использует следующие квитирующие (подтверждающие) сигналы:

- VSYNC: Vertical Sync Output (выход вертикальной синхронизации (для строк) – низкий уровень во время кадра;
- HREF: Horizontal Reference (горизонтальная синхронизация (для колонок) – высокий уровень во время активных пикселов ряда (строки);
- PCLK: Pixel Clock Output (пиксельная синхронизация (тактовый сигнал передачи байта из параллельного порта D0–D7) – независимый генератор синхронизирующих импульсов. Данные правильны на нарастающем фронте.

В дополнение к этому камера оперирует еще следующими сигналами:

- D0-D7: параллельный цифровой 8-битный видеовыход в формате YUV/RGB;
- PWDN: Power Down Mode Selection - включение (лог. 0) и выключение (лог. 1) камеры;
- XCLK: внешнее тактирование (синхронизация);
- Reset: сигнал сброса.

Камера OV7670 синхронизируется с помощью генератора на 24 МГц. Это обеспечивает выход пиксельной синхронизации (PCLK) 24 МГц. Буфер типа FIFO имеет память на 3 Мбит. Генератор тестовых шаблонов формирует шаблон цветовых полос – с 8 полосами, с постепенным уменьшением к серому цвету (fade-to-gray).

Плюсы: дешевизна, настраиваемость, DSP процессор, прямая совместимость с Arduino платой.

Минусы: сложность настройки, необходимость в посреднике.

2. MT9V034C12STM.

Стереосенсоры позволяющие получить изображение 1504 x 480 в 60 кадров в секунду. Для таких стереосенсоров можно заказать плату, к которой можно подключить эти же сенсоры и установить процессор и память для обработки изображения на самом роботе.

Плюсы: больший спектр возможностей.

Минусы: дороговизна, необходимость в сторонних средствах.

Интегрирование указанных модулей в дальнейшем раскроется по подробнее. Рассмотрим каждый модуль, в частности, и выявим их плюсы и недостатки.

Начнем с модуля А (рисунок 1.5). Для использования данного устройства нужен Wifi посредник, который бы передавал поток из камеры на устройство способное обработать изображение и вернуть команды (оператор мобильного робота), ибо процессор платы ардуино не способен на такое. Для корректной передачи изображения я буду использовать регистры процессора, и через десктопное приложение Arduino настроим входные порты платы Arduino.

Рисунок 1.5 - Модуль камеры OV7670

Регистры процессора (микроконтроллера) — блок ячеек памяти, образующий сверхбыструю оперативную память внутри процессора. Есть регистры, используемые самим процессором для вычислений, например, при выборке из памяти очередной команды она помещается в регистр команд, а при выборке переменных они помещаются в регистры операндов. Но в микроконтроллере есть еще и другие регистры – хранящие настройки разных

периферийных устройств. С ними в основном и имеет дело программист. Например, регистр PORTB, хранящий состояние выводов (пинов), или регистр DDR, хранящий настройки, находятся ли выводы (пины) в состояние входа или выхода. В случае с камерой, регистр – это служебная очень быстрая память, например ячейки по 8, 16 бит, к которой можно обратиться по адресу. На стороне камеры работает программа, которая управляет камерой в соответствии с настройками, хранящимися в этих регистрах. Эта программа, в том числе, принимает посылки по I2C – первые 8 бит – адрес, следующие 16 бит – значение регистра. Конфигурировать регистры буду булевыми операциями – AND (&) и OR(|).

Для полноты схемы указанной ниже подойдет любой wifi модуль совместимый с ардуино, способный отправлять и принимать сигналы. Таким образом я буду передавать изображение через wifi модуль на устройство, свой ПК, которое будет анализировать изображение, и передавать на Arduino плату либо соответствующие значения, либо необходимые команды (рисунок 1.6).

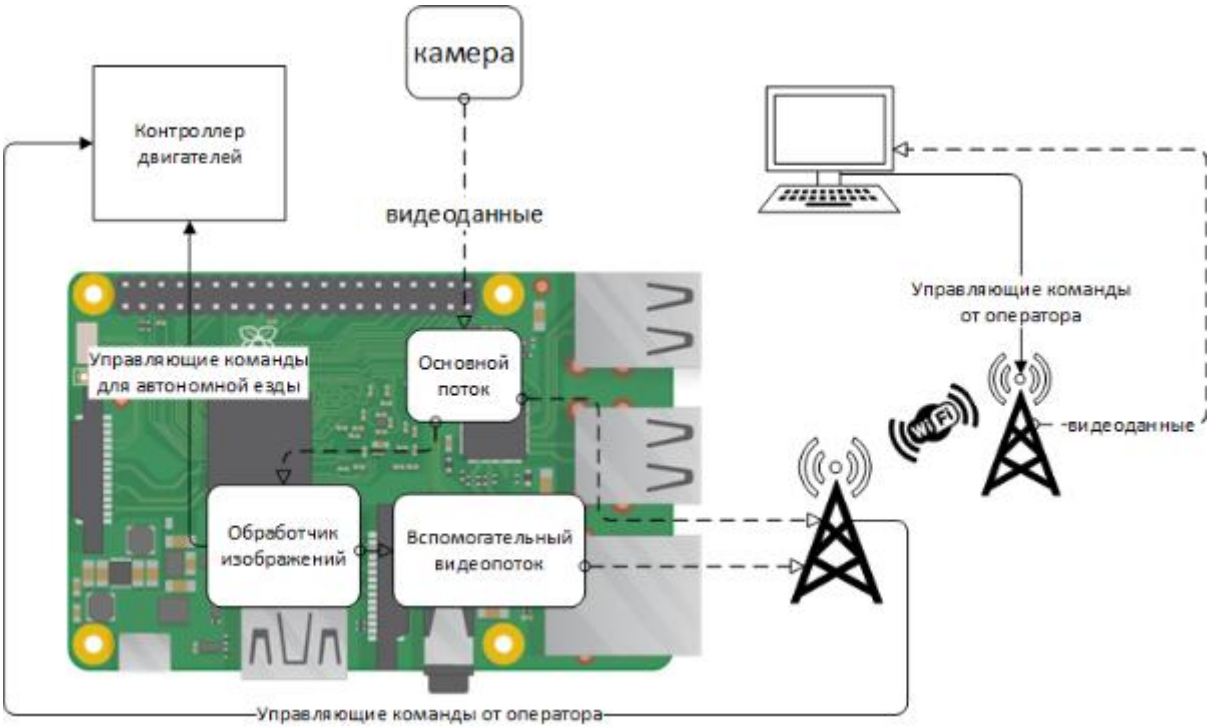

Рисунок 1.6 – Схема работы, которую мы реализуем

Рассмотрим модуль Б (рисунок 1.7). Команда интузиастов из статьи на хабре рассказали об опыте создания платы для стереосенсоров. Они интегрировали специальные процессоры от Интел для двух камер.

Рисунок 1.7 – Стереосенсоры

В мире полно литературы на данную тему, и существует множество решений, способствующих мобильному роботу ориентироваться в пространстве. На данный момент реализованного мобильного робота на базе Arduino и использующего 4 ик датчика найти не удалось, поэтому был выбран подобный робот, с апгрейдом и установкой на него веб камеры. Для достижения цели будут использоваться максимально дешёвые варианты комплектации, и данный выбор обоснован лишь личным интересом автора.

Подробное изучение литературы и существующих алгоритмов дало четкое представление об итоговом алгоритме и о том, что необходимо реализовать различные алгоритмы и привести их анализ.

ГЛАВА 2
РАЗРАБОТКА АЛГОРИТМА И МОБИЛЬНОГО РОБОТА

2.1 Разработка алгоритма движения по линии мобильного робота

Для разработки алгоритма введем окружающую среду, в которой будет двигаться робот. Пусть это будет белое полотно с черной линией, в хорошо освещенном помещении, можно определить как будет двигаться мой робот, я решил использовать 4 датчика линии MH-Sensor_Serial[TSRT-5000], для навигации робота в пространстве. Такой датчик я выбрал, потому что датчик линии выполнен на оптопаре TCRT5000. Установление уже двух аналоговых сенсора линии на днище мобильной платформы, чтобы заставить робота не выезжать за пределы территории, обозначенной контуром или следовать за нарисованной линией. Сенсор способен не только отличать темную поверхность от светлой, но и фиксировать все оттенки серого. Это дает вам возможность точно контролировать процесс перехода границы от черного к белому и поможет роботу не сбиться с пути. Датчик линии выполнен на оптопаре TCRT5000, которая состоит из двух элементов — ИК-светодиода (излучателя) и фототранзистора (приемника) (рисунок 2.2).

Рисунок 2.1 — Датчик линии MH-Sensor_Serial[TSRT-5000]

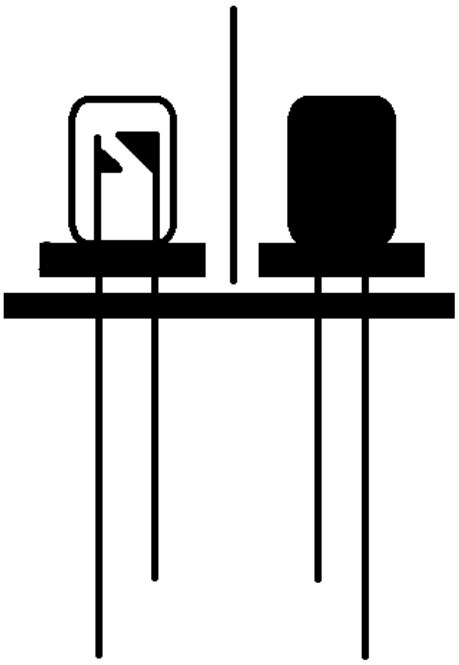

Рисунок 2.2 — Схема расположения приемника и излучателя ИК датчика

Когда светодиод излучает инфракрасный свет, световой поток отражается от поверхности и попадает на фототранзистор, где преобразуется в электрический сигнал. Темный цвет отражает меньше света, светлый — больше. Показания датчика линии зависит от цвета объекта и расстояния сенсора до детектируемой поверхности.

Цвет объекта. Чем белее отражающая поверхность, тем выше уровень напряжение на выходе «S». Чем чернее отражающая поверхность, тем ниже уровень сигнала на выходе «S». Пропасть над столом равносильно максимально черной поверхности.

Расстояние сенсора до детектируемой поверхности. Датчик способен выдавать «адекватные» показания при расстоянии между сенсором и детектируемой поверхностью в диапазоне от 3 до 12 мм. При расстоянии менее 3 миллиметров — перегородка между ИК-излучателем и приемником мешает сенсору принимать отраженный свет. А при расстоянии более 12 миллиметров — отраженный свет рассеивается и не доходит до приемника.

Тогда алгоритм будет содержать логику, описанную далее. калибровка датчиков в окружающей среде, для стабильной ее работы, для этого замеряем значения датчиков на темном и на светлом цвете. Далее взять среднее значение и использовать его при создании программы для определения смещения робота относительно линии.

Интеграция веб-камеры на данном этапе позволит снять вычислительную нагрузку с основного робота, за счета обработки изображения на устройстве помощнике (рисунок 1.6). В одной из версий мобильного робота, выбранного для тестирования на базе Arduino, была установлена веб-камера, передающая в реал-тайм видео на устройство-посредник, где уже происходит обработка видео. Устройство посредник обрабатывает видео покадрово используя OpenCV.

Рассмотрим по подробнее использование OpenCV для анализа изображения. Для анализа будем использовать функционал, представляемый самим OpenCV и язык Python. Для установки используется команда терминала «pip install opencv-python» либо «poetry add opencv-python». Использовать буду python версии 3.12, установленный через pyenv, но можно и вручную, в дополнение нужно установить numpy.

Функционал будем тестировать на кусочках окружения (рисунки 2.3). Первым делом нужно избавить исходные данные от всего лишнего, поможет обычное размытие и преобразование в градации серого (Рисунок 2.4). Для размытия OpenCV имеет реализацию алгоритма Гаусса «cv2.GaussianBlur()» возвращающий MatLike объект импортируемый из opencv.

Рисунок 2.3 – Элементы окружающей среды

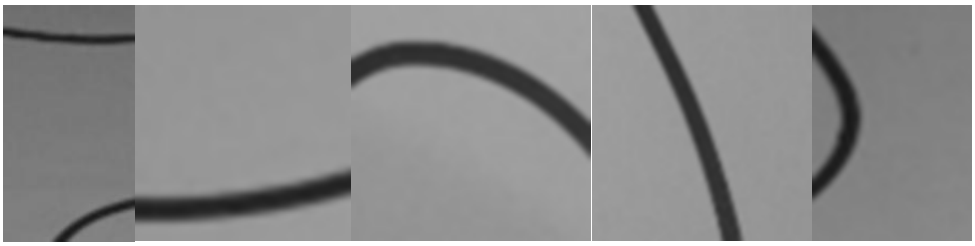

Рисунок 2.4 – Элементы окружающей среды, к которым применён алгоритм Гаусса

Далее нам нужно как-то выделить линию, для этого применим бинаризацию предлагаемую OpenCV «cv2.adaptiveThreshold()» (рисунок 2.5).

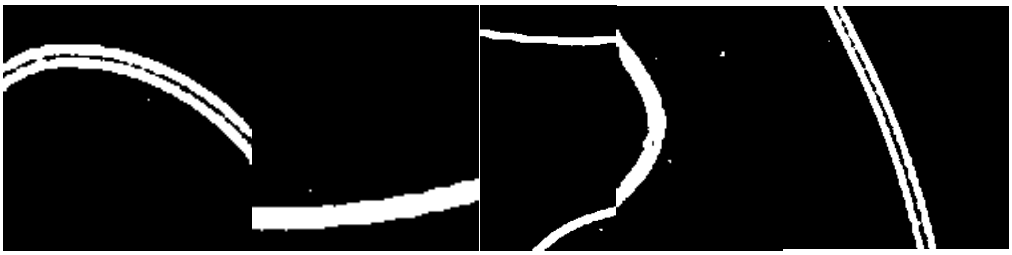

Рисунок 2.5 – Элементы окружающей среды, к которым применён алгоритм бинаризации

Для получения итогового результата нужно выделить границы и опять же можно использовать то, что предлагает OpenCV «cv2.findContours», мы получи массив всех контуров, из них нам нужен максимальный контур, так как в случае движения по линии мы знаем, что самый большой объект (после выделения называемый контуром) это и есть линия по которой мы движемся, так мы обезопасим себя от шума, который не убрался размытием. Определив центр получаемого изображения, нам остается посчитать отклонение, которое мы передадим на нашего мобильного робота. Для этого определим центр выделенного контура. Отклонение посчитаем по простой формуле 2.1.

$$\begin{aligned} x &= contourCenterX - mainImageX, \\ y &= contourCenterY - mainImageY. \end{aligned} \tag{2.1}$$

Таким образом мы получим необходимые для движения по линии мобильного робота корректировки. Робот присылает на потоковое видео, устройство посредник разбивает его на кадры, переводит в оттенки серого и применяет к ним алгоритм Гаусса, далее обрабатываем алгоритмом бинаризации и алгоритмом поиска контуров, для наглядности результат отобразил на исходных данных (рисунок 2.6).

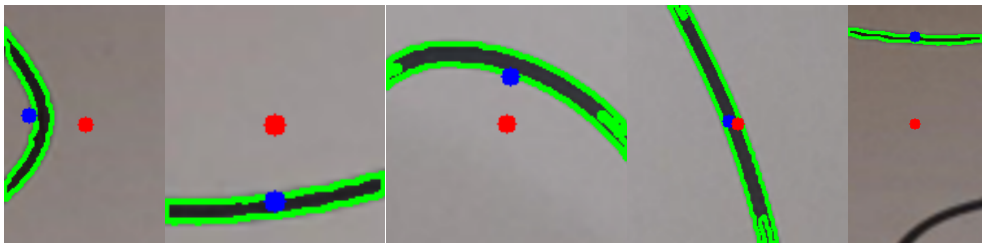

Рисунок 2.6 – Элементы окружающей среды, итоговый вариант обработки изображений, с применением визуализации

Данные действия позволят использовать линию и окружающий мир любого цвета. После получения отклонений, отправляем результат

мобильному роботу, который в свою очередь прогоняет результат через регулятор и создает себе команды для дальнейшего движения.

Для анализа изображений можно использовать стороннее решение, например заранее обученную нейронную сеть, на данный момент можно подключить нейронную сеть описанную на hugging face используя готовые веса и все нужные компоненты, либо уже работающую нейронную сеть, например deepseek, имеющую открытое платное API. Запросы к такому API стоят в районе 0.0001 у.е., вообще стоимость зависит от количества используемых токенов. Выбор пал на deepseek из-за возможности пополнить свой счет без каких-либо проблем (Приложение А). Чтобы запросить координаты отклонений нужно составить промпт (Приложение Б) с указанием роли модели, в качестве роли была выбрана роль «mobile robot». Промпт должен состоять из исходных данных, текста запроса и условий, ожидаемых в ответе от сети. В результате мы получим от работающей нейронной сети координаты отклонений, которые сможем использовать в дальнейшем. Осталось учесть время отклика такой сети, в результате тестирования мы имеем 5 секунд ожидания ответа от сети, таким образом мы не сможем в реалтайм оценивать изображение, но можем оценивать ситуацию каждые 10 секунд, такое решение является простым.

В мире ещё существует Gemini, предоставляющий возможность нескольких бесплатных запросов. Время отклика в среднем 3 секунды, но при этом появляется не приятный момент в виде рейтлимита по запросам. Можно использовать проксирование запросов, но проксирование повысит время отклика, и повысит сложность выполнения поставленной задачи.

Таким образом мы имеем несколько вариантов анализа итогового изображения. Выгодным является анализ через OpenCV, он точен и не имеет задержки больше 1 секунды. Использование нейронной сети подходит под решение в современном мире, но все ещё не совершенно и требует множество ресурсов, как в случае с использованием десктопной нейросети. Поэтому я остановился на решении с OpenCV.

Теперь нам необходимо реализовать регуляторы для наших роботов. Начнем с пропорционально-интегрально-дифференциального управления, чтобы автоматически корректировать свое положение относительно линии. Это может включать регулировку скорости или угла поворота для правильного следования колес по линии.

Корректировка алгоритма в соответствии с данными, полученными в результате нескольких запусков, а именно корректировка параметров K_p, K_d,

K_i. После продолжительной разработки алгоритма ПИД регулятора получится следующий код (Приложение В).

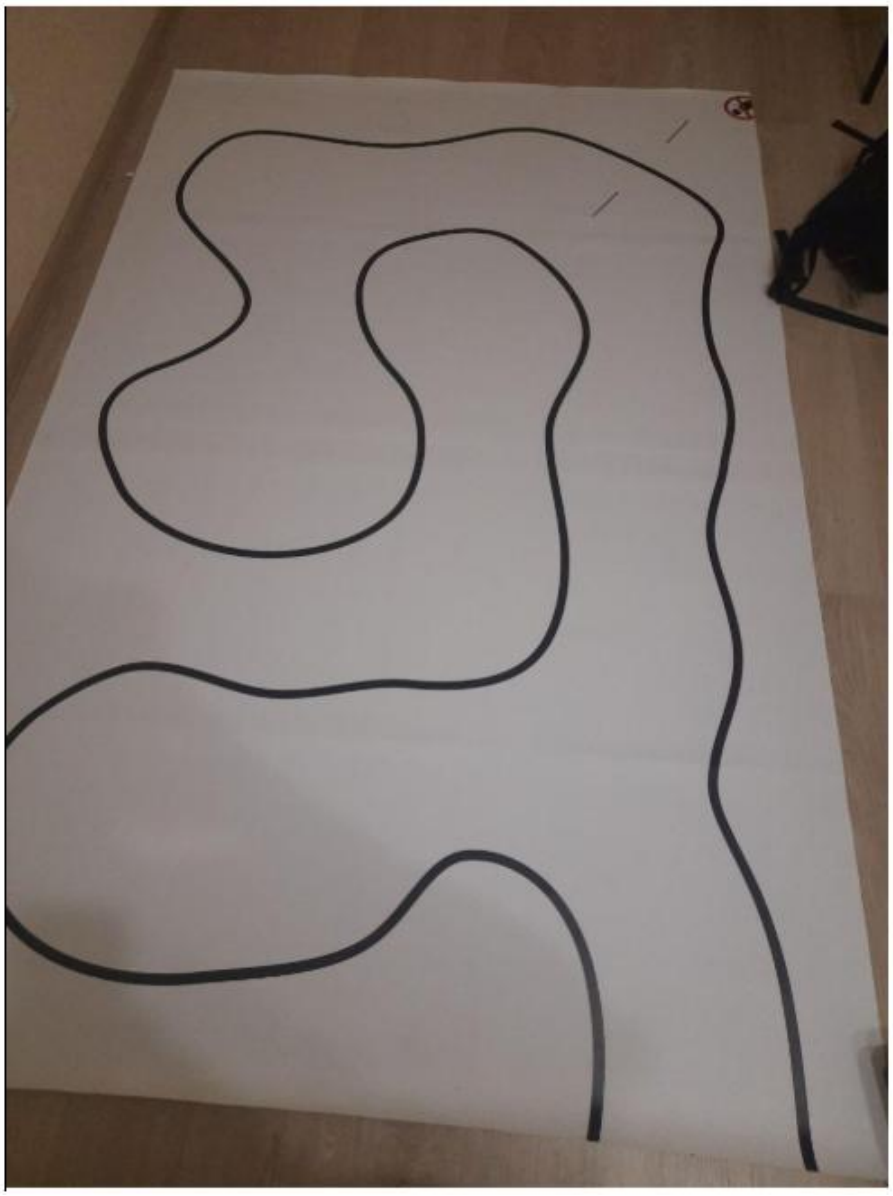

Рисунок 2.7 – Окружающая среда для робота

Для сравнения реализуем и другие регуляторы, для визуального и технического определения оптимального алгоритма в реальных заданных ранее условиях.

2.2 Выбор и настройка мобильного робота и программного обеспечения для тестирования алгоритма

Для реализации двух мобильных роботов была выбрана база Arduino из-за простоты создания роботов, калибровки датчиков и написания алгоритма. Комплектующие были выбраны следующие: плата Arduino Uno (рисунок 2.8), потому что это компактная плата и она имеет как аналоговые порты, так и цифровые. Драйвер для управления двигателями постоянного тока L298N (рисунок 2.9), используется радиолюбителями для многофункционального управления двигателями постоянного тока, модуль камеры (рисунок 1.5) и wifi модуль. Схема драйвер-модуля, состоящая из двух Н-мостов, позволяет подключать к нему один биполярный шаговый двигатель или одновременно два щеточных двигателя постоянного тока. При этом есть возможность изменять скорость и направление вращения моторов. Управление осуществляется путем подачи соответствующих сигналов на командные входы, выполненные в виде штыревых контактов. Также используется куча проводов и система питания робота от 9v кроны. В качестве приложения для создания алгоритма и взаимодействия с роботом будет выбрано приложение Arduino. В результате получится робот с рисунка 1.2. В процессе было принято решение установить на «крышу» веб камеру (рисунок 1.5) с передатчиком для получения новых способов реализации алгоритма, и путей оптимизации.

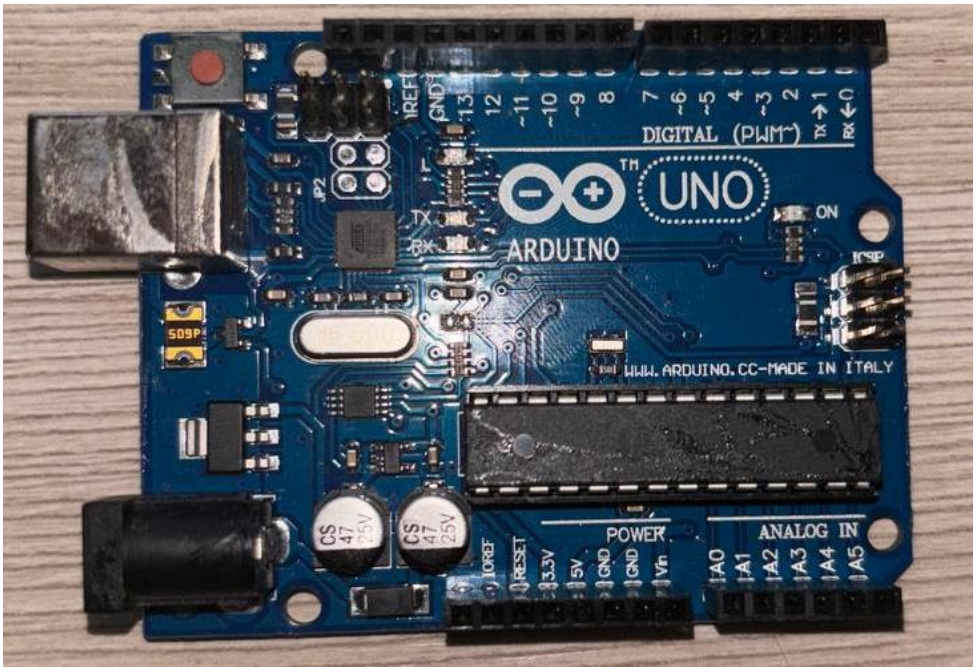

Рисунок 2.8 – Плата Arduino Uno

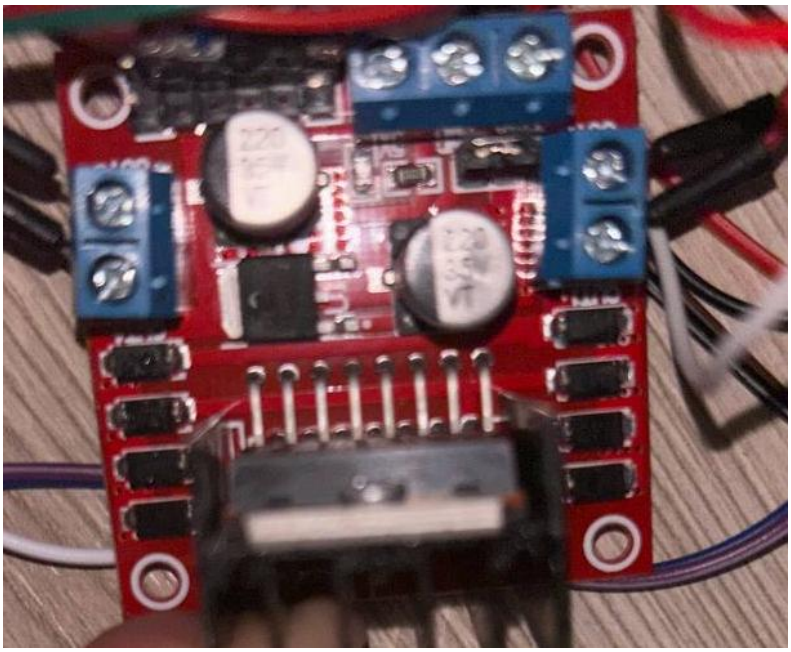

Рисунок 2.9 – Драйвер для управления двигателями постоянного тока L298N

В ходе разработки были трудности с разработкой редуктора двигателя, для стабилизации работы двигателей и повышения их мощности, по средством уменьшения крутящего момента, используя несколько высокотехнологичных шестеренок. После нескольких самостоятельных попыток распечатать редуктор на 3д принтере было принято решения приобрести редуктор с моторчиком и колесом (рисунок 2.10).

Рисунок 2.10 – Электро-моторчик с редуктором и колесом

В результате мы имеем мобильного робота и алгоритм движения мобильного робота, реализованный на основе различных регуляторов, который автоматически подстраивается под положение робота относительно линии.

Далее рассмотрим мобильного робота на базе Lego Mindstorm (рисунок 1.4). Мозгами такого робота является модуль EV3, имеющий 4 порта ввода отмеченных цифрами и 4 порта вывода, отмеченных латинскими буквами. Подключение будет осуществляется через ПК порт. Для отслеживания линии будет использоваться датчик линии (датчик цвета), подключаемый через порт ввода, таких датчиков будем использовать 4. Для движения будут использоваться большие моторы, подключаемые через порты вывода. Опираясь на интуицию, мы можем собрать мобильного lego робота (рисунок 1.4). Алгоритмы для мобильного робота можно реализовать в фирменном приложении собрав схему из блоков кода (рисунок 1.3).

2.3 Тестирование алгоритма и оценка его эффективности. Поиск путей оптимизации

Рассмотрим все алгоритмы по подробнее на каждом из роботов, соберём все плюсы и минусы. Тестирование алгоритмов производилось на карте (рисунок 2.7).

Первым рассмотрим алгоритм на основе П регулятора.

Мобильный робот на базе lego mindstorm справился с трассой за 1 минуту и 42 секунды, реализация алгоритма была очень простой и быстрой. При проезде трассы робот потерял линию пару раз, двигался дергано. При повторном тесте робот показал схожие результаты. При смене освещения ничего не поменялось, так как на результат влияет лишь истинный цвет полотна и линии. По итогу мы имеем лишь несколько путей оптимизации: физический и технический. Физически мы можем либо увеличивать и уменьшать расстояние между датчиками, либо изменять расстояние от датчика до полотна. Технически мы можем изменять П компоненту регулятора, скорость вращения колес. Тем самым мы можем влиять на количество виляний робота, на скорость его движения и скорость реакции на изменение линии.

Плюсы:

- Простая реализация;
- Простое изменения конструкции;

- Быстрое и легкое изменение параметров для оптимизации проделанной работы под поставленную задачу;

- Независимость от освещения.

Минусы:

- Узконаправленное использование;

- Малый спектр возможностей для оптимизации;

- Относительно долгое прохождение линии;

- Множественное виляние робота, от которое почти невозможно избавиться, не отрегулировав ширину линии.

Мобильный робот на базе Arduino без использования камеры справился с прохождением трассы за 1 минуту и 36 секунд, реализация алгоритма требовала базовые навыки программирования на C\C++ подобных языках, поэтому сложно сказать о том, что это было просто. При прохождении трассы робот не терял линию, но двигался очень дергано. При повторном запуске робота в такой же окружающей среде изменений не наблюдалось. При смене освещения ничего не поменялось, так как на результат влияет лишь истинный цвет полотна и линии. Данное решение имеет несколько путей оптимизации, при этом можно отметить, что настройка данного робота очень гибкая, можно с легкостью поменять фактически все. Физически я могу полностью поменять конструкцию робота, от расстояния между датчиками, до расположения центра массы робота и диаметра колес. Из полезных настроек, это диаметр колес, расстояние между колесами, расстояние между датчиками, между датчиками и полотном, вес робота, мощность самих двигателей. С технической точки зрения можно поменять скорость вращения двигателей, последовательность выполнения алгоритма и сама П регуляторная компонента. Изменение этих параметров позволяет настроить робота под различные необходимые задачи.

Плюсы:

- Многозадачность;

- Количество путей оптимизации;

- По сравнению с базой Lego прохождение линии занимает меньше времени;

- Большое количество изменений в корпусе самого робота;

- Независимость от освещения;

- Широкий спектр применения.

Минусы:

- Сложность реализации алгоритма;

- Виляние мобильного робота, от которого почти невозможно избавиться.

Мобильный робот на базе Arduino с использованием камеры и устройства-посредника проходит линию за 1 минуту и 18 секунд. Реализация алгоритма требует навыков по обработке изображений и возможности совмещения работы нескольких сервисов, такой мобильный робот фактически не виляет и не теряет линию благодаря тому, что знает все заранее. Повторный запуск мобильного робота изменений не дает. При ярком освещении робот корректно использует устройство-посредник, если убрать освещение робот будет двигаться по линии так же, как и если бы камера у робота отсутствовала, так как изображение — это просто черный квадрат и робот считает, что находится на линии, из-за чего постоянно теряет линию. Оптимизация такого мобильного робота многогранна, так как появляется новая переменная в виде устройства посредника. Оптимизация зависит от алгоритмов анализа изображения, что дает широкий спектр возможностей. Мы так же можем прикрутить регуляторы к исходным данным и к данным который отправляем роботу, который в свою очередь тоже прогонит их через собственный регулятор. Физические пути оптимизации такие же, как и у предыдущего мобильного робота, т.е. мы можем изменять расстояние между колесами, дорожный просвет, расстояние между датчиками, расстояние между датчиками и дорожным полотном, диаметр колес, масса, различные объекты, например можно добавить светодиодную фару, для корректного использований камеры в случае отсутствия освещения. С технической точки зрения мы можем изменять П регуляторную компоненту и скорость вращения / мощности колес, частоту отправки результатов с камеры, т.е. мы можем либо непрерывно отправлять видео, в байтово-потоковом формате, либо каждые 5 секунд.

Плюсы:

- Большая скорость прохождения линии;
- Подготовленность к сложным участкам;
- Широчайший спектр возможностей по оптимизации;
- Большой спектр применения в реальном мире;
- Движение без ошибок;
- Независимость от освещения при наличии светодиодной фары.

Минусы:

- Зависимость от освещения, в случае отсутствия светодиодной фары;
- Высокое энергопотребление, при использовании 9v кроны, заряда

хватит лишь на пару запусков, по сравнению с другими реализациями мобильных роботов.

- Высокая сложность реализации алгоритма;
- Зависимость от устройства посредника и его вычислительных мощностей, что удорожает производство;

Теперь рассмотрим алгоритм на основе ПИ регулятора.

Мобильный робот на базе lego mindstorm справился с трассой за 1 минуту и 34 секунды, реализация алгоритма была очень простой. При проезде заданного маршрута мобильный робот потерял линию несколько раз, двигался плавнее чем на П регуляторе. При повторном тесте робот показал похожие результаты. При смене освещения ничего не поменялось, так как на результат влияет лишь истинный цвет полотна и линии. Для данного робота мы имеем несколько путей оптимизации: физический и технический. Физически мы можем либо увеличивать и уменьшать расстояние между датчиками, либо изменять расстояние от датчика до полотна. Технически мы можем изменять П компоненту регулятора и И компоненту, отвечающую за ошибки, скорость вращения колес. Таким образом мы можем влиять на количество виляний робота, на скорость его движения и скорость реакции на изменение линии.

Плюсы:

- Простая реализация;
- Простое изменения конструкции;
- Быстрое и легкое изменение параметров для оптимизации проделанной работы под поставленную задачу;
- Независимость от освещения.

Минусы:

- Узконаправленное использование;
- Относительно долгое прохождение линии;
- Множественное виляние робота, от которое почти не избавиться, не отрегулировав ширину линии.

Мобильный робот на базе Arduino без использования камеры справился с прохождением трассы за 1 минуту и 32 секунды, реализация алгоритма требовала базовые навыки программирования на C\C++ подобных языках, поэтому сложно говорить о простоте. При прохождении трассы робот не терял линию, но передвигался виляя. При повторном запуске робота в такой же окружающей среде изменений не наблюдалось. При смене освещения ничего не поменялось, так как на результат влияет лишь истинный цвет полотна и линии. Данное решение имеет несколько путей оптимизации, при этом можно отметить, что настройка данного робота очень гибкая, можно с легкостью

поменять фактически все. Физически я могу полностью поменять конструкцию робота, от расстояния между датчиками, до расположения центра массы робота и диаметра колес. Из полезных настроек, это диаметр колес, расстояние между колесами, расстояние между датчиками, между датчиком и полотном, вес робота, мощность самих двигателей. С технической точки зрения можно поменять скорость вращения двигателей, последовательность выполнения алгоритма и сами П и И регуляторные компоненты. Изменение этих параметров позволяет настроить робота под различные необходимые задачи.

Плюсы:

- Многозадачность;
- Количество путей оптимизации;
- Большое количество изменений в корпусе самого робота;
- Независимость от освещения;
- Широкий спектр применения.

Минусы:

- Сложность реализации алгоритма;
- Виляние мобильного робота, от которого почти невозможно избавиться.

Мобильный робот на базе Arduino с использованием камеры и устройства-посредника проходит линию за 1 минуту и 20 секунд. Реализация алгоритма требует навыков по обработке изображений и возможности совмещения работы нескольких сервисов, такой мобильный робот фактически не виляет и не теряет линию благодаря тому, что знает все заранее. Повторный запуск мобильного робота изменений не дает. При ярком освещении робот корректно использует устройство-посредник, если убрать освещение робот будет двигаться по линии так же, как и если бы камера у робота отсутствовала, так как изображение — это просто черный квадрат и робот считает, что находится на линии, из-за чего постоянно теряет линию. Оптимизация такого мобильного робота многогранна, так как появляется новая переменная в виде устройства посредника. Оптимизация зависит от алгоритмов анализа изображения, что дает широкий спектр возможностей. Мы так же можем прикрутить регуляторы к исходным данным и к данным который отправляем роботу, который в свою очередь тоже прогонит их через собственный регулятор. Физические пути оптимизации такие же, как и у предыдущего мобильного робота, т.е. мы можем изменять расстояние между колесами, дорожный просвет, расстояние между датчиками, расстояние между датчиками и дорожным полотном, диаметр колес, масса, различные объекты,

например можно добавить светодиодную фару, для корректного использований камеры в случае отсутствия освещения. С технической точки зрения мы можем изменять П и И регуляторные компоненты, и скорость изменения мощности колес, частоту отправки результатов с камеры, т.е. мы можем либо непрерывно отправлять видео, в байтово-потоковом формате, либо каждые 5 секунд.

Плюсы:

- Большая скорость прохождения линии;
- Подготовленность к сложным участкам;
- Широчайший спектр возможностей по оптимизации;
- Большой спектр применения в реальном мире;
- Движение без ошибок;
- Независимость от освещения при наличии светодиодной фары.

Минусы:

- Зависимость от освещения, в случае отсутствия светодиодной фары;
- Высокое энергопотребление, при использовании 9v кроны, заряда хватит лишь на пару запусков, по сравнению с другими реализациями мобильных роботов.
- Высокая сложность реализации алгоритма;
- Зависимость от устройства посредника и его вычислительных мощностей, что удорожает производство;

Теперь рассмотрим алгоритм на основе ПИД регулятора.

Мобильный робот на базе lego mindstorm справился с трассой за 1 минуту и 24 секунды, реализация алгоритма достаточно проста. При проезде трассы робот почти не терял линию, двигался плавно иногда дергаясь. При повторном тесте робот показал схожие результаты. При смене освещения ничего не поменялось, так как на результат влияет лишь истинный цвет полотна и линии. По итогу мы имеем следующие пути оптимизации: физический и технический. Физически мы можем либо увеличивать и уменьшать расстояние между датчиками, либо изменять расстояние от датчика до полотна. Технически мы можем настраивать П И Д компоненты регулятора, скорость вращения колес. Тем самым мы можем влиять на количество виляний робота, на скорость его движения и скорость реакции на изменение линии.

Плюсы:

- Достаточно простая реализация, учитывая то, что алгоритм основан на ПИД регуляторе;
- Простые изменения конструкции;

- Быстрое и легкое изменение параметров для оптимизации проделанной работы под поставленную задачу;

- Плавное прохождение заданной линии;

- Независимость от освещения.

Минусы:

- Узконаправленное использование;

- Малый спектр возможностей для оптимизации;

Мобильный робот на базе Arduino без использования камеры справился с прохождением трассы за 1 минуту и 16 секунд, реализация алгоритма требовала базовые навыки программирования на C\C++ подобных языках, поэтому сложно сказать о том, что это было просто. При прохождении трассы робот не терял линию и двигался плавно. При повторном запуске робота в такой же окружающей среде изменений не наблюдалось. При смене освещения ничего не поменялось, так как на результат влияет лишь истинный цвет полотна и линии. Данное решение имеет несколько путей оптимизации, при этом можно отметить, что настройка данного робота очень гибкая, можно с легкостью поменять фактически все. Физически я могу полностью поменять конструкцию робота, от расстояния между датчиками, до расположения центра массы робота и диаметра колес. Из полезных настроек, это диаметр колес, расстояние между колесами, расстояние между датчиками, между датчиками и полотном, вес робота, мощность самих двигателей. С технической точки зрения можно поменять скорость вращения двигателей, последовательность выполнения алгоритма и П И Д регуляторные компоненты. Изменение этих параметров позволяет настроить робота под различные необходимые задачи.

Плюсы:

- Многозадачность;

- Количество путей оптимизации;

- По сравнению с базой Lego прохождение линии занимает меньше времени;

- Большое количество изменений в корпусе самого робота;

- Независимость от освещения;

- Широкий спектр применения.

Минусы:

- Сложность реализации алгоритма;

- Шанс потерять линию.

Мобильный робот на базе Arduino с использованием камеры и устройства-посредника проходит линию за 1 минуту и 18 секунд. Реализация

алгоритма требует навыков по обработке изображений и возможности совмещения работы нескольких сервисов, такой мобильный робот фактически не виляет и не теряет линию благодаря тому, что знает все заранее. Повторный запуск мобильного робота изменений не дает. При ярком освещении робот корректно использует устройство-посредник, если убрать освещение робот будет двигаться по линии так же как и если бы камера у робота отсутствовала, так как изображение это просто черный квадрат и робот считает, что находится на линии, из за чего постоянно теряет линию. Оптимизация такого мобильного робота многогранна, так как появляется новая переменная в виде устройства посредника. Оптимизация зависит от алгоритмов анализа изображения, что дает широкий спектр возможностей. Мы так же можем прикрутить регуляторы к исходным данным и к данным который отправляем роботу, который в свою очередь тоже прогонит их через собственный регулятор. Физические пути оптимизации такие же как и у предыдущего мобильного робота, т.е. мы можем изменять расстояние между колесами, дорожный просвет, расстояние между датчиками, расстояние между датчиками и дорожным полотном, диаметр колес, масса, различные объекты, например можно добавить светодиодную фару, для корректного использований камеры в случае отсутствия освещения. С технической точки зрения мы можем изменять П И Д регуляторные компоненты и скорость вращения / мощности колес, частоту отправки результатов с камеры, т.е. мы можем либо непрерывно отправлять видео, в байтово-потоковом формате, либо каждые 5 секунд.

Плюсы:

- Большая скорость прохождения линии;
- Подготовленность к сложным участкам;
- Широчайший спектр возможностей по оптимизации;
- Большой спектр применения в реальном мире;
- Движение без ошибок;
- Независимость от освещения при наличии светодиодной фары.

Минусы:

- Зависимость от освещения, в случае отсутствия светодиодной фары;
- Высокое энергопотребление, при использовании 9v кроны, заряда хватит лишь на пару запусков, по сравнению с другими реализациями мобильных роботов.
- Высокая сложность реализации алгоритма;
- Зависимость от устройства посредника и его вычислительных мощностей, что удорожает производство;

Проанализировав написанное выше, можно сказать, что П и И и Д регуляторные алгоритмы в абстракции достаточно эффективны, также как и различные двойные комбинации. Их реализация проста и выгодна с точки зрения ресурсов, но такие алгоритмы применимы для простых трасс, без перекрестков и разрывов. Для использования в сложных ситуациях больше подходят алгоритмы на ПИД регуляторе, в идеале в связке с веб-камерой.

Рассмотрим по подробнее настройку ПИД регуляторного алгоритма, а точнее его параметров используя участок с резким поворотом. На рисунке 2.11 изображено примерное поведение мобильного робота, в зависимости от используемого регулятора. Для его настройки мы будем использовать принцип настройки по порядку, т.е. сначала мы занулим все коэффициенты и П коэффициент выставим в 1, тем самым получив П регуляторный алгоритм и робот должен будет поехать по темно-красной траектории. Регулируя значение этого параметра, мы должны привести траекторию близкую к предполагаемой. Далее мы настраиваем И коэффициент, приравняв к 0.5, и далее подгоняя его под предполагаемую траекторию, после запоминаем это значение и зануляем его. Проводим настройку Д компоненты пока его движение не станет более стабильным по отношению к обычному П регуляторному алгоритму. Далее возвращаем И компоненту и смотрим на поведение робота. В случае не идеальной поездки по чуть-чуть регулируем коэффициенты И и Д.

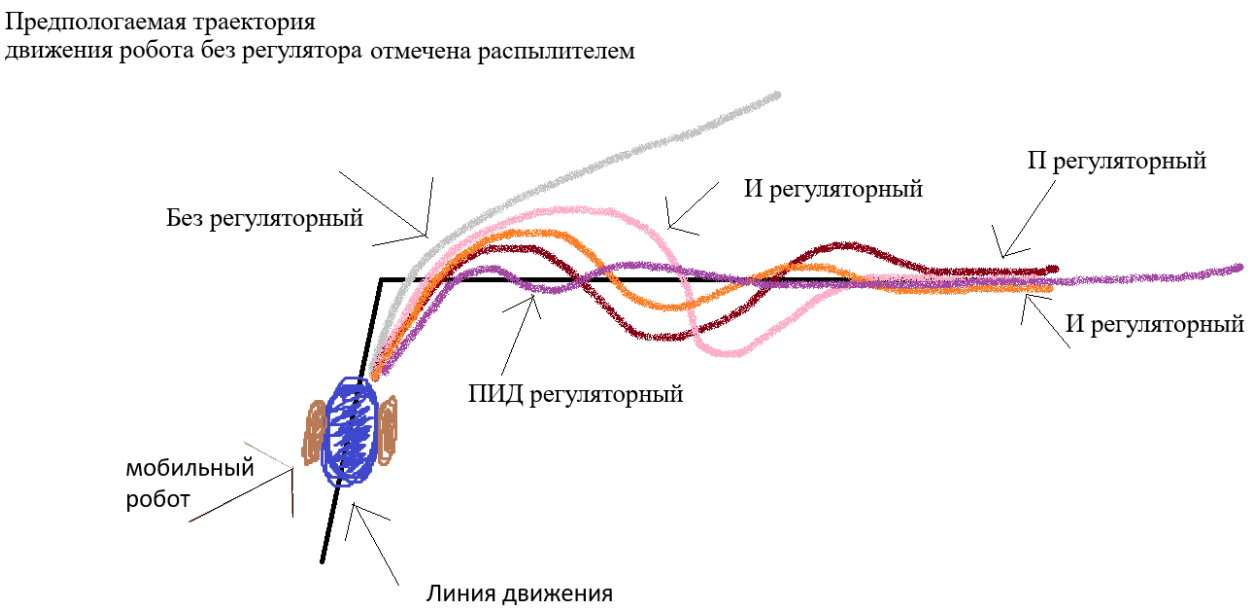

Рисунок 2.11 – Схема приблизительной траектории движения мобильного робота, использующего различные регуляторы

Оптимизация ПИД регуляторного алгоритма означает настройку его параметров таким образом, чтобы траектория мобильного робота была как можно ближе к предполагаемой, т.е. оптимальной.

Для оптимизации алгоритма можно использовать различные подходы. Один из самых эффективных, это ручное изменение параметров и регулировка положения датчиков на роботе. Использование данных, с прошлых поездок, а именно количество ошибок и не точностей, учет уникальности заданной траектории, а также изменение самой конструкции робота под наши нужды.

Выбор пал на базу Arduino и Lego Mindstorm. Lego Mindstorm позволяет быстро и доступно получить необходимый результат, от обычного управляемого робота, до самостоятельного мобильного робота способного ориентироваться в пространстве и удерживать себя на линии следования. При всем при том это достаточно дорогое решение, поэтому стоит остановиться на базе Arduino, в частности на плате Arduino UNO, в дополнении с веб камерой и 4 ИК датчиками.

Итоговое решение вышло дешевым и надежным. Для разработки алгоритма использовался C\C++ подобный язык. Реализованные ПИД, П и Д регуляторные алгоритмы показали свою эффективность в различных случаях, так для простых задач больше подходит П и Д регуляторы, в связи с их простотой, а для сложный трасс, имеющих перекрестки и прерывистые линии лучше использовать ПИД регуляторный в связке с камерой, желательно с обработкой всего видеопотока.

ЗАКЛЮЧЕНИЕ

Дипломная работа посвящена рассмотрению и анализу существующих алгоритмов, способствующих мобильному роботу ориентироваться в окружающем мире. При выполнении дипломной работы были рассмотрены существующие решения в мире разработки собственного мобильного робота и алгоритмов для них. Таким образом были выделены алгоритмы на основе различных регуляторов (ПИД регулятор, П регулятор, Д регулятор), алгоритмы с использованием машинного обучения, и алгоритмы на основе логики нечетких множеств. Для всех алгоритмов были выделены их преимущества и недостатки.

Вторым важным этапом являлась собственная разработка мобильного робота и подходящего для него алгоритма. Были разработаны 3 робота, два на базе Arduino и на базе LegoMindstorm. Два робота использовать ИК датчики для ориентирования в пространстве, и один в дополнение использовал Веб камеру. Уникальностью данной работы является робот, основанный на базе Arduino и использующий 4 ИК датчика имеющий минимальную итоговую стоимость. Под оптимизацией алгоритма подразумевалась настройка параметров ПИД регулятора и настройка самого мобильного робота, например: перемещение оси колес с центра в заднюю часть робота, смещение центра тяжести в результате добавления или удаления грузиков.

Практическая значимость данной дипломной работы заключается в широком спектре возможностей, от логистики, до обучающих и развлекающих целей. Ярким примером являются конкурсы и олимпиады по робототехнике, проводимые в Республике Беларусь, использование похожих технологий для перемещения огромных белазов по карьерам, компания Amazon использует мобильных роботов в качестве автоматизированных работников на своих складах.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Документация Arduino [Электронный ресурс]. — Режим доступа: https://docs.arduino.cc/ . — Дата доступа: 21.10.2024.
2. Метод Зиглера-Никольса настройки параметров регулятора // Компьютеры и программы. Все о компьютерной технике и программном обеспечении. — Режим доступа: https://komp.alleasing.ru/parametr/metod-tsiglera-nikolsa-nastroyki-parametrov-regulyatora.html. — Дата доступа: 25.05.2025.
3. Овсяницкий, Д. Н. Алгоритмы и программы движения робота по линии Lego Mindstorms EV3 / Д. Н. Овсяницкий — Москва: Перо, 2015 — 168 с.
4. Овсяницкая, Л.Ю. Пропорциональное управление роботом Lego Mindstorm EV3 / Л.Ю. Овсяницкая, Д.Н. Овсяницкий, А.Д. Овсяницкий. — Издательство «Перо», 2015 – 188 с.
5. Сотникова, М. В. Алгоритм автоматического удержания колесного робота на визуально заданной линии / М. В. Сотникова — Вестник СПбГУ. 2016 — С. 101-107.
6. Статья о настройке среды разработки Arduino IDE [Электронный ресурс]. — Режим доступа: https://deepbluembedded.com/arduino-getting-started-beginners-guide/#google_v ignette . — Дата доступа: 10.11.2024.
7. Т. Ю. Ким, Г. А. Прокопович Оптимизация коэффициентов ПИД-регулятора системы управления движением мобильного робота по цветоконтрастной линии на основе генетического алгоритма / Т. Ю. Ким, Г. А. Прокопович [Электронный ресурс]. — Режим доступа: https://inf.grid.by/jour/article/view/1170 - Дата доступа: 04.12.2023.
8. Филиппов, С. Основы робототехники на базе конструктора Lego Mindstorms NXT. Занятие 5. / С. Филиппов. – Заочная школа современного программирования. 2010 – 43 с.
9. Deepseek docs [Electronic resource]. – Mode of access: https://api-docs.deepseek.com/. – Date of access: 24.04.2025.
10. Johny, A., Lentin J. Robot Operating System (ROS) for Absolute Beginners / J. Lentin. – Make: Cheerakathil House, Aluva, India, 2018 – 283 p.
11. Margolis, M. Make an Arduino-Controlled Robot / M. Margolis. – Make: O'REILLY, 2012 – 254 p.

```
from openai import OpenAI
import settings

client = OpenAI(api_key="<settings.DEEPSEEK_API_KEY.get_secret_key()>",
base_url="https://api.deepseek.com")

response = client.chat.completions.create(
    model="deepseek-reasoner",
    messages=[
        {"role": "system", "content": "You are a helpful assistant"},
        {"role": "user", "content": "Hello"},
    ],
    stream=False
)

print(response.choices[0].message.content)
```

{"role": "mobile robot", "content": f"image bytes: {image_bytes}, analyse this image and calculate deviation for commands via mobile robot. Send me only deviation X and Y"},

```
float Kp=0.9,Ki=0,Kd=5;
float error=0, P=0, I=0, D=0, PID_value=0;
float previous_error=0, previous_I=0;
int sensor[4]={0, 0, 0, 0};
int initial_motor_speed=100;
int left_motor_speed = initial_motor_speed;
int right_motor_speed = initial_motor_speed;

const int buttonPin = 2;  // the number of the pushbutton pin
const int ledPinRed = 10;
const int ledPinGreen = 11;// the number of the LED pin

// первый двигатель
int enA = 9;
int in1 = 7;
int in2 = 6;

// второй двигатель
int enB = 3;
int in3 = 5;
int in4 = 4;

// variables will change:
int buttonState = 0;  // variable for reading the pushbutton status
bool flag = false;
bool flag_light = false;
void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPinRed, OUTPUT);
  pinMode(ledPinGreen, OUTPUT);
  // initialize the pushbutton pin as an input:
  pinMode(buttonPin, INPUT);
  // motors
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
```

```
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
}

void read_sensor_values()
{
  sensor[0]=digitalRead(A3);
  sensor[1]=digitalRead(A2);
  sensor[2]=digitalRead(A1);
  sensor[3]=digitalRead(A0);
  if((sensor[0]==0)&&(sensor[1]==0)&&(sensor[2]==0)&&(sensor[3]==1))
   error=4;
   else
if((sensor[0]==0)&&(sensor[1]==0)&&(sensor[2]==1)&&(sensor[3]==1))
      error=3;
      else
if((sensor[0]==0)&&(sensor[1]==0)&&(sensor[2]==1)&&(sensor[3]==0))
         error=2;
         else
if((sensor[0]==0)&&(sensor[1]==1)&&(sensor[2]==0)&&(sensor[3]==0))
            error=-2;
            else
if((sensor[0]==1)&&(sensor[1]==1)&&(sensor[2]==0)&&(sensor[3]==0))
               error=-3;
               else
if((sensor[0]==1)&&(sensor[1]==0)&&(sensor[2]==0)&&(sensor[3]==0))
                  error=-4;
                  else
if((sensor[0]==0)&&(sensor[1]==0)&&(sensor[2]==0)&&(sensor[3]==0))
                     if(error==-4) error=-5;
                     else if(error==-4)error=5;
                       else error = 0;
}
void calculate_pid()
{
  P = error;
  I = I + previous_I;
```

```
  D = error-previous_error;
  PID_value = (Kp*P) + (Ki*I) + (Kd*D);
  previous_I=I;
  previous_error=error;
}
void loop() {
 // read the state of the pushbutton value:
 buttonState = digitalRead(buttonPin);
 // check if the pushbutton is pressed. If it is, the buttonState is HIGH:
 if (buttonState == HIGH) {
        if(flag){
        flag = false;
     }
    else flag = true;
    flag_light = true;
    delay(1000);

 }
 if(buttonState == LOW && flag) {
  if(flag_light){
    flag_light = false;
    //prestart to 2 sec
    digitalWrite(ledPinGreen, HIGH);    // включаем светодиод
    delay(1000);                // ждем 1000 миллисекунд (1 секунда)
    digitalWrite(ledPinGreen, LOW);     // выключаем светодиод
    delay(1000);
    digitalWrite(ledPinGreen, HIGH);
    delay(1000);                // ждем 1000 миллисекунд (1 секунда)
    digitalWrite(ledPinGreen, LOW);     // выключаем светодиод
    delay(1000);
    //start to 2 sec
    digitalWrite(ledPinRed, HIGH);
    delay(1000);                // ждем 1000 миллисекунд (1 секунда)
    digitalWrite(ledPinRed, LOW);     // выключаем светодиод
    delay(1000);
    }
    read_sensor_values();
    calculate_pid();
```

```
        // Расчет оптимальной скорости двигателя

        int left_motor_speed = initial_motor_speed-PID_value;
        int right_motor_speed = initial_motor_speed+PID_value;

        // Скорость двигателя не должна превышать максимальное значение
ШИМ
        constrain(left_motor_speed,0,255);
        constrain(right_motor_speed,0,255);
        analogWrite(enA,left_motor_speed); //Left Motor Speed
        analogWrite(enB,right_motor_speed); //Right Motor Speed

        digitalWrite(in1, HIGH);
        digitalWrite(in2, LOW);
        digitalWrite(in3, HIGH);
        digitalWrite(in4, LOW);

      }else{
        digitalWrite(in1, LOW);
        digitalWrite(in2, LOW);
        digitalWrite(in3, LOW);
        digitalWrite(in4, LOW);}
    }
```