МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Дальневосточный федеральный университет»
(ДВФУ)
ИНСТИТУТ МИРОВОГО ОКЕНА (ШКОЛА)
Кафедра автоматизации и управления

Данилов Даниил Дмитриевич

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
по направлению подготовки бакалавров

РАЗРАБОТКА ИНФОРМАЦИОННО-УПРАВЛЯЮЩЕЙ СИСТЕМЫ МОБИЛЬНОГО РОБОТА АМУР

по направлению подготовки (специальности) 15.03.06 - Мехатроника и робототехника
профиль «***Мехатроника и робототехника***»

Владивосток
2023

В материалах данной выпускной квалификационной работы не содержатся сведения, составляющие государственную тайну, и сведения, подлежащие экспортному контролю

Уполномоченный по экспортному контролю

_____ _____
подпись *И.О. Фамилия*

«_____» _____ 20____ г.

Защищена в ГЭК с оценкой

_____

Секретарь ГЭК

_____ _____
подпись *И.О. Фамилия*

«_____» _____ 20____ г.

Автор работы _____,
подпись

группа _____

«_____» _____ 20_____ г.

Руководитель ВКР _____
должность, ученое звание

_____ _____
подпись *И.О. Фамилия*

«_____» _____ 20____ г.

Консультант(ы)*

_____ _____
подпись *И.О. Фамилия*

«_____» _____ 20____ г.

Назначен рецензент* _____
ученое звание

_____
фамилия, имя, отчество

«Допустить к защите»

Руководитель структурного подразделения

_____
ученая степень, ученое звание

_____ _____
подпись *И.О. Фамилия*

«_____» _____ 20____ г.

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

студенту _____Данилову Даниилу Дмитриевичу _____
фамилия, имя, отчество (при наличии)

_____15.03.06 - Мехатроника и робототехника; Б11119-
15.03.06мхрб_____
направление подготовки / специальность; группа

1. Тема работы: **Разработка информационно-управляющей системы мобильного робота амур**

2. Срок сдачи студентом законченной работы: **07.07.2023**

3. Исходные данные по работе **техническая документация робота «АМУР», статьи и доклады.**

4. . Содержание работы (перечень подлежащих разработке вопросов):
Создание информационно-управляющей системы мобильного робота.
Настройка системы распознавания человека или объекта по камере.
Реализация движения робота по сплайну с заданной скоростью.

5. Перечень графического материала (с указанием обязательных чертежей):
Отсутствуют

6. Консультанты по работе
Зуев Александр Валерьевич

7. Основные источники информации:
Научные статьи и доклады.

Руководитель ВКР ____А.В. Зуев_____ _____

И.О. Фамилия *подпись*

Задание принял к исполнению _____

дата

Студент _____Д.Д. Данилов_____ _____

И.О. Фамилия *подпись*

КАЛЕНДАРНЫЙ ГРАФИК ВЫПОЛНЕНИЯ РАБОТЫ

№ п/п	Наименование этапов дипломного проекта (работы)	Срок выполнения этапов дипломного проекта (работы)	Примечание
1	Модернизация аппаратной части колесного робота «АМУР»	10.01.2023	
2	Подготовка и настройка программного обеспечения робота	04.02.2022	
3	Разработка информационно-управляющей системы робота	11.04.2022	
4	Работа программного обеспечения при использовании библиотек машинного обучения	01.05.2022	
5	Проведение экспериментальных исследований	06.05.2022	
7	Оформление пояснительной записки	15.06.2022	

Дата выдачи задания _____

Срок представления к защите _____

Руководитель ВКР	_____	Зуев Александр Валерьевич
	(подпись)	(ФИО)
Студент	_____	Данилов Даниил Дмитриевич
	(подпись)	(ФИО)

Аннотация

В работе описано создание информационно-управляющей системы (ИУС) для мобильного колёсного робота АМУР-107 от компании “Сенсорика”.

ИУС реализует движение робота по сплайну, при этом движение робота происходит с заданной скоростью, объезжая препятствия в виде стульев (можно выбрать другое из доступных решений библиотеки машинного обучения MediaPipe), также есть режим поиска человека, где робот двигаясь по траектории сплайна ищет по камере людей (используется библиотека машинного обучения MediaPipe), а затем приближается на заданное расстояние.

До начала работы робот был в неработоспособном состоянии из-за отказа аккумуляторных батарей и сгорании предохранителя электронной части, с программным обеспечением позволяющим взаимодействовать и передавать по локальной сети данные от внешнего ПК к бортовому и наоборот, взаимодействовать бортовому ПК с платой управления “Робокон”. Управление роботом производилось исключительно по кнопкам с внешнего компьютера.

Существовавшее программное обеспечение робота являлось устаревшим и не взаимодействовало с множеством современных библиотек: библиотекой получения видеопотока и обработки изображений - OpenCV, библиотекой машинного обучения для распознавания объектов и поз человека - MediaPipe, библиотекой - Qt для создания приложений, библиотекой socket для обмена по транспортным протоколам.

Первичная задача состояла в переносе с версии Python 2.7 на версию Python 3.9, которую поддерживают все вышеописанные библиотеки на сегодняшний день.

Для реализации обновлённой и расширенной ИУС системы были произведены расчёты колёсной одометрии, основанные только на показаниях энкодеров при этом использовались библиотеки для матричных вычислений NymPY, math. Распознавание объектов было осуществлено с использованием камеры и библиотек машинного обучения MediaPipe. Обработка данных была реализована на внешнем ПК, так как скорость передачи достаточно высокая, а внешний ПК по производительности и графической обработки сильнее бортового.

Введение

Мобильный робот (МР) – это робот, который способен перемещаться, пользуясь колесами, ногами, гусеницами или воздушными винтами для передвижения по ландшафту. В отличие от стационарных роботов, которые ограничены в своей области действия, мобильные роботы способны перемещаться внутри или вне помещений для выполнения различных задач (в данном случае область работы достигает области действия локальной сети АМУР).

Мобильные роботы используются в различных областях, таких как [1,2]:

- промышленность для автоматизации производственных процессов;
- медицина для транспортировки медицинских препаратов и инструментов, а также для выполнения хирургических операций;
- наука и исследования для сбора данных и обследования недоступных мест;
- службы безопасности для разведки и разминирования территорий;
- бытовые нужды, такие как пылесосы-роботы и роботы для газонокосения и уборки снега.

Мобильные роботы оснащаются датчиками и программным обеспечением для навигации и распознавания окружающей среды, которые позволяют им выполнять задачи автономно без участия человека (в данном случае энкодеры, камера, ультразвуковые датчики расстояния).

Колесный МР — это робот, оснащенный колесами для передвижения по различным поверхностям (в данном случае на двух управляемых колёсах и нескольких ведомых).

Для создания эффективной ИУС для мобильного колёсного робота необходимо учитывать множество факторов, включающих в себя задачи, которые должен решать робот, физические характеристики робота, окружающую среду и т.д. Подобная система должна комбинировать в себе аппаратные и программные компоненты, обеспечивающие передвижение робота, его управление, обмен информацией с окружающей средой и другие функции, необходимые для выполнения поставленных задач. Так в данной дипломной работе была реализована управляющая система с дистанционной обработкой данных при этом управление робота происходило без участия человека.

1 Описание существующих подходов к реализации ИУС МР, описание ИУС АМУР-105 и его функциональных возможностей

Главная функция ИУС для всех типов МР — обеспечение их высокоточного движения в разных режимах эксплуатации. Однако различный набор бортовых датчиков, вычислительных устройств и исполнительных механизмов МР, предназначенных для решения многих задач, требует длительной и трудоёмкой разработки ИУС для необходимой точности управления конкретными МР. Сейчас уже созданы стандарты и инструменты, облегчающие построение ИУС. Наиболее известны платформы ROS, Player Project, MS Robotic Studio, URBI, которые являются базой для универсальных ИУС МР. Большинство из них имеет удобные графические средства, позволяющие не только задать конфигурацию ИУС, но и моделировать движение МР, управляемых этой ИУС. Однако ни одна из указанных платформ не содержит средств для выбора аппаратной части ИУС и её структуры, обеспечивающих качественное функционирование МР. Также следует отметить, что ИУС, созданные на базе ROS 1, работают только под управлением операционной системы Ubuntu (и экспериментально — OS X, Android), ROS 2 же расширяет функционал ROS 1 и позволяет работать на операционной системе Windows 10 (но предпочтение до сих пор остаётся за Ubuntu), из всего представленного ROS предполагает наличие производительных бортовых компьютеров. Однако для уменьшения стоимости МР в качестве основных вычислительных узлов ИУС целесообразнее применять дешёвые встраиваемые контроллеры. Проектирование ИУС на платформе Player Project требует проводить обмен данными между элементами всей системы только по протоколу TCP, что существенно ограничивает выбор программной и состав аппаратной частей МР. При использовании MS Robotic Studio необходима поддержка операционной системы Windows. Кроме того, эта программная платформа имеет закрытый исходный код, что усложняет её применение для решения нестандартных задач.

На бортовом нетбуке стоит операционная система Ubuntu 13.04 поэтому, наиболее очевидным решением в текущей задаче построения ИУС мобильного робота было использование платформы ROS, однако при этом необходимо было бы произвести обновление как самого Ubuntu 13.04, так и установку самого ROS с необходимыми библиотеками визуализации и обработки информации, что на данном бортовом компьютере сделать почти невозможно. Объём оперативной памяти на нетбуке ASUS Eee PC 1011CX составляет 1.6 гигабайт, на работу системы при этом для работы ROS минимальные требования составляют от 2х гигабайт (для минимальных настроек и работы с библиотеками технического зрения OpenCV и MediaPipe). Поэтому в данном случае наиболее простой и рациональный способ создания ИУС напрямую используя скрипт Python. Данный подход обеспечивает кроссплатформенность запуска внешнего программного обеспечения. Для которого необходимы лишь библиотеки питон 3.9, а также библиотеки зависимостей технического зрения робота (библиотеки MediaPipe, OpenCv). Обобщенная структура ИУС представлена на рисунке 1.1.

Рисунок 1.1 – Обобщенная структура ИУС

Основными проблемами при построении ИУС данного робота является устаревшая аппаратная часть. Бортовой компьютер расположен в роботе для выполнения и обработки каких-либо трудно затратных работ, с которыми не может справиться микроконтроллер, также для передачи данных по сети используя модуль Wi-Fi. С точки зрения передачи, отправки данных проблем нет и скорость передачи достаточная. Вычислительные возможности при оперативной памяти 1.6 гигабайт достаточно ограничены. Для обработки каких-либо данных с измерительных устройств, например с одометрических или ультразвуковых датчиков скорости обработки будет достаточно, но для обработки изображений даже в один прогон без использования нейросетей уже будут вызывать трудности у данного ПК. Исходя из вышесказанного для решения задач движения робота по сплайну с заданной скоростью и детектирование человека было принято решение по возможности не изменять существовавшее программное обеспечение бортового ПК, а заняться разработкой внешнего программного обеспечения, которое сможет работать с уже существующим.

Все электронные и электромеханические устройства МР размещены в стандартном пластиковом корпусе-кейсе со снимаемой монтажной пластиной в крышке, что обеспечивает лёгкость доступа к элементам системы и удобство транспортировки робота одним человеком. Корпус снабжён закрывающимися на ключ замками, что позволяет ограничить/проконтролировать доступ к компонентам робота. Характеристики мобильного колёсного робота АМУР-105:

- вес интеллектуального герметизированного колёсного и гусеничного шасси – не более 12 кг;
- вес максимальной дополнительной возможной сенсорной нагрузки до 5 кг;
- скорость регулируется от не менее 0,2 до 0,6 м/с;

- запас хода – не менее 0,9 км, автономность – не менее 2 часов;
- предусмотрена возможность подключения захватного устройства для тяжёлых объектов (2 шести-пиновых разъёма установлены на крышке роботов);
- максимальная скорость управляющего радиоканала не менее 2Мб/с на 50м (в условиях прямой радиовидимости);
- исполнение робота: температура эксплуатации в диапазоне не ниже -10ОС и не выше +40ОС, с частичной защитой от атмосферных осадков, в сплошном запираемом корпусе;
- в корпусе робота установлены микро-PC (NetBook) в антишоковом исполнении с малогабаритным LCD-экраном, 1 электронная плата для управления 4 каналами по 4-5А в режиме on-off, 8 каналами ввода дискретных данных, 2 каналами ультразвуковых сенсоров и открытым интерфейсом I2C/USB для расширения состава сенсоров и 2 каналами силового ШИМ-управления по 7-9А;
- на роботе установлены ультразвуковые датчики (2шт.) измерения дальности в диапазоне от не менее 15 см до не более 400 см и аналоговая ТВ камера с управлением от электронной платы (видеосервер в корпусе);
- ТВ-камера подключается по стандарту Ethernet и установлена на 2-х степенном подвесе с углом поворота не менее 210 град;
- программное обеспечение обеспечивает доступ к оборудованию в распределённой среде с Wi-Fi соединениями до 50 метров для решения локомоционных задач супервизорного управления со сбором ТВ и ультразвуковых данных.

Устройство, создающее ЛВС представлено на рисунке 1.2 и состоит из технических компонентов, таких как:

- Стабилизатор напряжения IPON STAB 1000
- Коммутатор D-Link
- Антенна распределения радиосигнала
- Роутер EdgeRouter X

Рисунок 1.2 – Устройство, создающее ЛВС

Внутренний и внешний вид мобильного робота АМУР-105 приведены на рисунках 1.3-1.6.

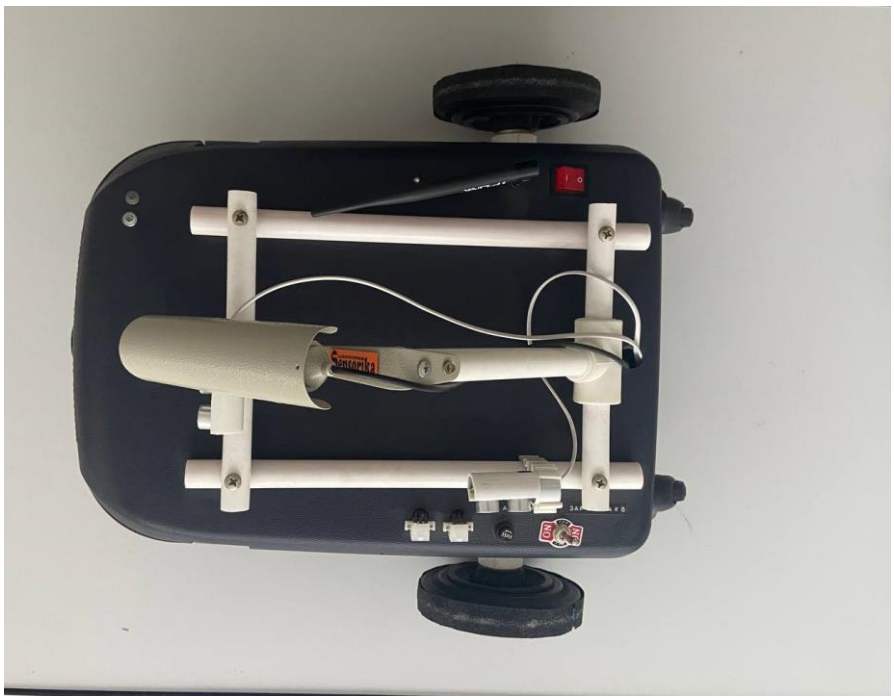

Рисунок 1.3 – Внешний вид мобильного колёсного робота.

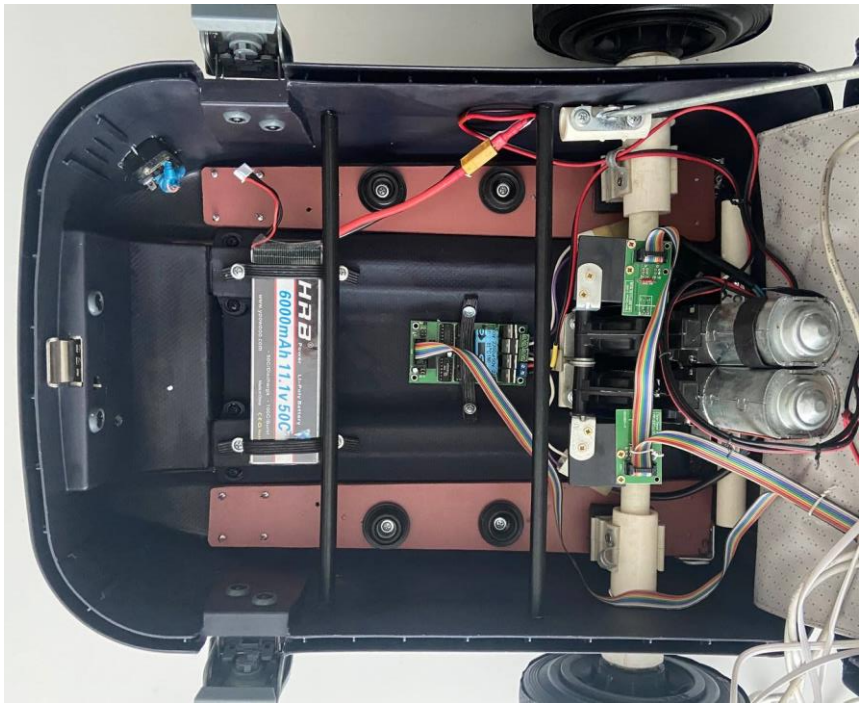

Рисунок 1.4 – Внутренний вид мобильного колёсного робота

Рисунок 1.5 – Внутренний вид мобильного колёсного робота.

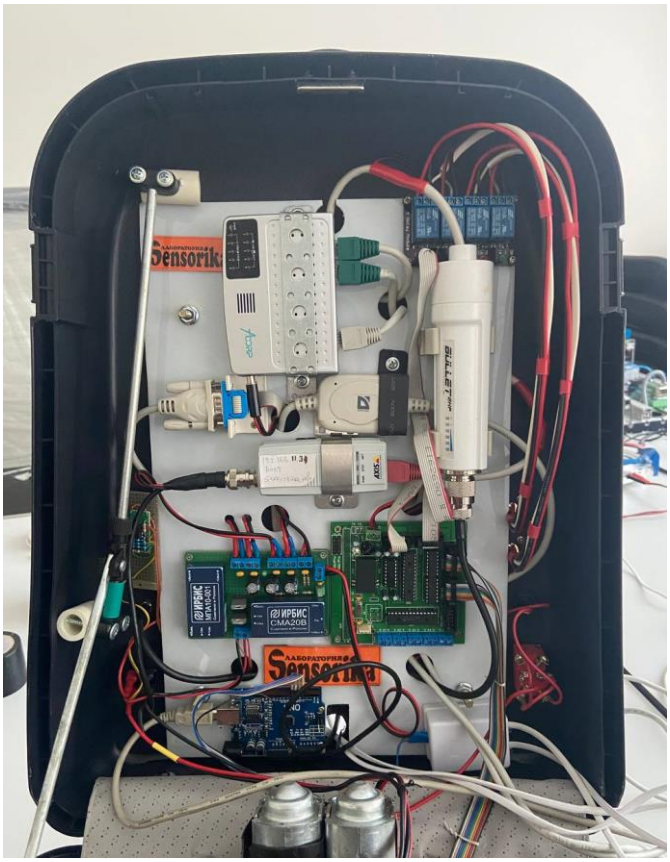

Рисунок 1.6 – Внутренний вид мобильного колёсного робота.

2 Модернизация и анализ аппаратной архитектуры

Данный робот подходит для выполнения различного рода локомоционных задач, таких как поиск объекта, отслеживание передвижения, построения карты местности и т.д. Для реализации некоторых задач будет необходимо расширять аппаратный функционал робота (например, для построения карты местности потребуется лидар). Также данный робот имеет не идеальное быстродействие так как работает с частотой 5 Гц. В данной работе не были произведены глобальные изменения аппаратной части, а только замена неисправных частей, так как основная задача состояла в создании внешней ИУС системы. С точки зрения аппаратной части были произведены следующие изменения: замена перегоревшего предохранителя 10А на электронную часть, замена вышедших из строя кислотных аккумуляторов, замена лампочек, сигнализирующих о разряде аккумулятора, фиксация оптических энкодеров, замена поверхности трения колёс.

2.1 Структурно-функциональная схема мобильного робота и анализ архитектуры

Структурно-функциональная схема мобильного колёсного робота АМУР-105 приведена на рисунке 2.1.

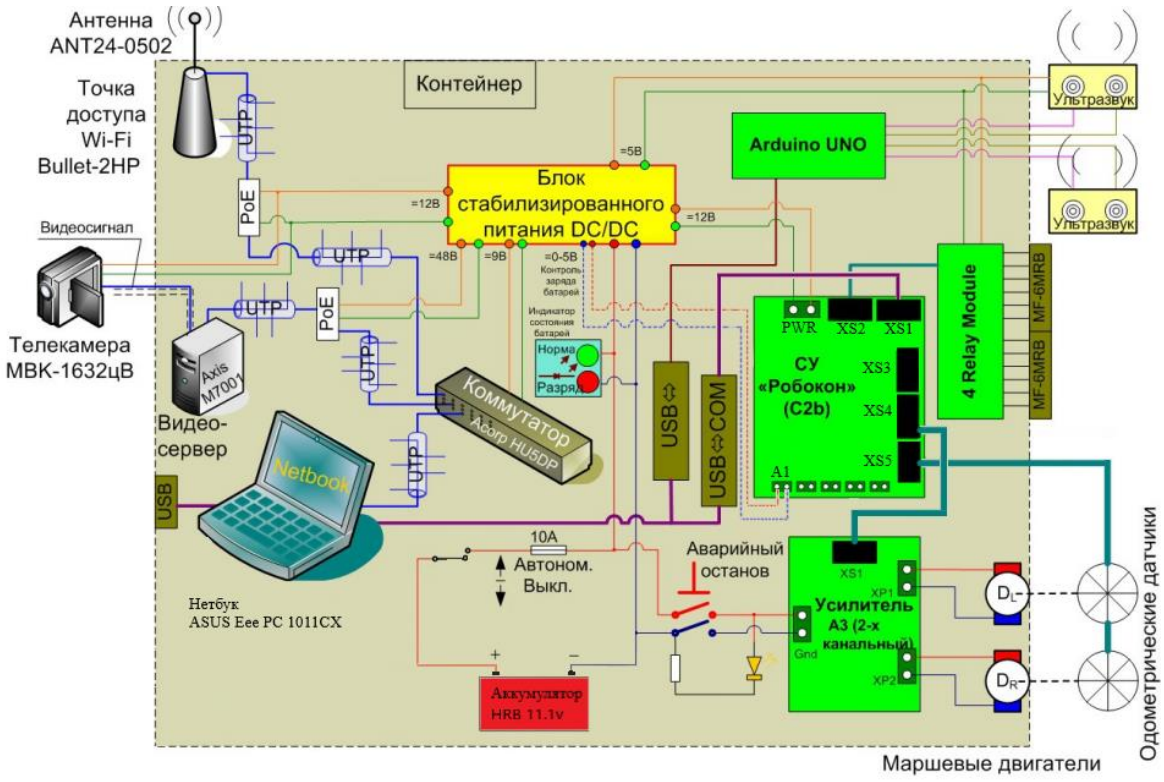

Рисунок 2.1 – Структурно-функциональная схема мобильного колёсного робота АМУР-105

Управляющие и измерительные приборы, размещённые в корпусе мобильного робота:

- Нетбук ASUS Eee PC 1011CX
- Точка доступа Wi-Fi Ubiquiti Bullet-2HP с антенной модели AHT24-0502

- Телекамера МВК-1632цВ
- Видеосервер модели Axis M7001
- Сетевой коммутатор модели Acorp HU5DP
- Микроконтроллеры Робокон и Arduino UNO
- Драйвер двигателей А3
- Одометрические датчики (оптические энкодеры) HOA0901-011
- Ультразвуковые датчики SRF-05
- Систему формирования стабилизированных питающих напряжений (БСП), включающую конверторы напряжения ИРБИС СМВ20В
- Управляющий релейный модуль на 4 канала SRD-05VDC-SL-C
- Аккумулятор литий ионный HRB 6000 mAh 11.1v

Бортовой ПК ASUS Eee PC 1011CX для АМУР-105 предназначен для размещения программного обеспечения (ПО), с помощью которого осуществляется формирование команд управления электронными и мехатронными системами робота и опрос показаний датчиков, установленных на роботе. Питание бортового ПК обеспечивается штатными АКБ. Штатное зарядное устройство бортового ПК подключается к сети 220В. IP адрес в локальной сети Амур: 192.168.11.32. Установлена операционная система – Linux Ubuntu 13.04, что является достаточно старой системой, которая нуждается в обновлении если возникнет необходимость автономной работы робота (например, с обновлёнными внешними библиотеками), в данной же работе обновлять систему не понадобилось, так как всё управление и обработка были на внешнем ПК. Внешний ПК ASUS Eee PC 1011CX вид представлен на рисунке 2.2.

Рисунок 2.2 – Бортовой ПК ASUS Eee PC 1011CX

Точка доступа: Wi-Fi Ubiquiti Bullet-2HP с всенаправленной радиоантенной модели АНТ24-0502 обеспечивает подключение систем МР к ЛВС комплекса по радиоканалу. Питание точки доступа осуществляется стабилизированным напряжением 12В от БСП по

методу PoE. Настройки точек доступа: IP адрес: 192.168.11.30, логин: root, пароль: sensorika.info.

Телекамера МВК-1632цВ – это аналоговая цветная телекамера, которую обычно используют для систем охранной сигнализации. Её стандартный аналоговый видеосигнал преобразуется в сигнал стандарта Ethernet с помощью видеосервера. Питание телекамеры осуществляется стабилизированным напряжением 12В от БСП.

Видеосервер модели Axis M7001 представляет собой одноканальный видеосервер, выполняющий преобразование стандартного аналогового видеосигнала в цифровую форму, сжатие получаемого изображения по выбранному алгоритму со скоростью кадровой развёртки (50Гц) и формирование выходного видеопотока в стандарте Ethernet. Согласно схеме (рис.4), видеопоток без какой-либо дополнительной обработки передаётся по радиоканалу на ПК постов управления, ПО которых (видеоплеер), в свою очередь, преобразует этот видеопоток в растровое изображение. Питание видеосервера осуществляется стабилизированным напряжением 48В от БСП по методу PoE. Настройки видеосервера: IP адрес: 192.168.11.31, логин: root, пароль: sensorika.info. Алгоритм сжатия – M-JPEG

Сетевой коммутатор модели Acorp HU5DP на 5 портов обеспечивает формирование бортовой ЛВС из электронных компонент МР. Питание сетевого коммутатора осуществляется стабилизированным напряжением 9В от БСП.

Микроконтроллер Робокон и Arduino UNO. Функциональная схема платы Робокон представлена на рисунке 2.3. Назначение разъемов ввода вывода (портов контроллера) следующее: порт PWR — порт и разъем питания модуля +5В от БСП; XS1 - XS5 — порты ввода-вывода COM интерфейса, под энкодеры использован порт XS5, под драйвер двигателя использован вход XS1, под реле использован порт XS4, для обмена данными с компьютером и управлением платой использован порт XS3; XS6 — разъем шины ISA8 не использован, A1 – A5 — порты аналогового ввода не использованы, PWR2 — выходной разъем питания 5В, выводит питание на выход модуля не использован.

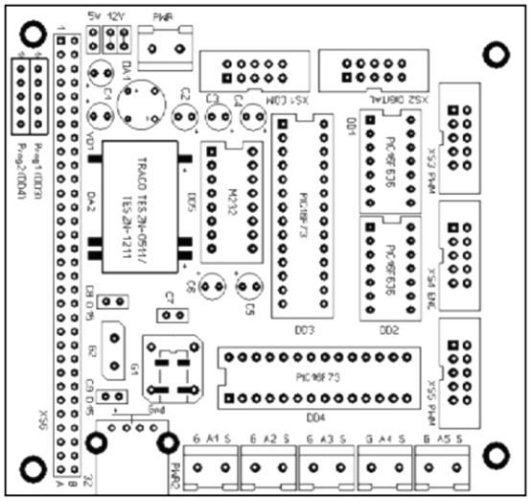

Рисунок 2.3 – Схема платы Робокон

Функциональная схема платы Arduino UNO представлена на рисунке 2.4. Имеет 6 аналоговых портов A0-A5, и 14 цифровых, питается посредством подключения к

компьютеру по USB (A-B) и также по данному кабелю происходит обмен информацией с бортовым ПК. Данная плата необходима для работы с ультразвуковыми датчиками, так как на прямую подключится к плате Робокон не получится (порта для цифровых сигналов нет).

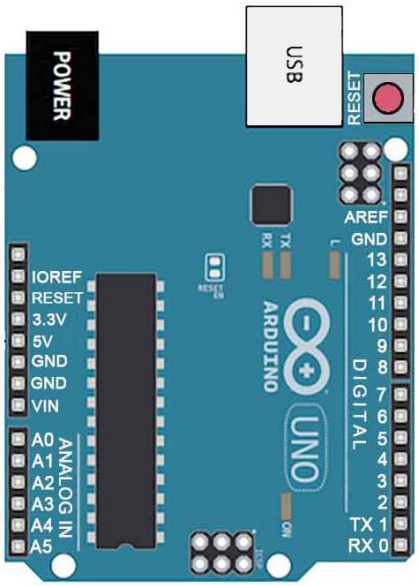

Рисунок 2.4 – Схема платы Arduino UNO

Драйвер двигателей А3 нужен для управления двигателями постоянного тока с платы Робокон. Регулировка скорости вращения происходит путём подачи напряжения. За счёт наличия Н моста движение колёс может обеспечиваться в разных направлениях. Управление происходит по ШИМ сигналу, при этом частота подачи сигнала 5 Гц из-за этого на графике скорости наблюдаются незначительные колебания. Для достижения максимальной плавности движения необходимо уменьшить время такта внешней управляющей программы.

Одометрические датчики (оптические энкодеры) HOA0901-011 отображают информацию о положении колеса из чего находится угол поворота колеса за такт программы и далее рассчитывается навигация робота пространстве. Датчики срабатывают на прозрачные деления кругов, прикреплённых к валам двигателей. Всего 60 делений, следовательно, при проходе одного деления угол поворота колеса составит 6 градусов, при этом ошибка может составлять половину цены деления, то есть 3 градуса.

Ультразвуковые датчики типа SRF05 в количестве 2-х штук установлены на крышке МР. Их показания анализируются микроконтроллером “Arduino UNO”, который, в свою очередь, подключён к нетбуку через USB. Наличие большого количества свободных портов на микроконтроллере позволяет, при желании, расширить состав сенсоров и элементов управления. Нагрузочная способность микроконтроллера определяется стандартом USB (не более 500мА), который обеспечивает нетбук.

Система формирования стабилизированных питающих напряжений (БСП), включает конверторы напряжения ИРБИС СМВ20В, служит для формирования набора стабилизированных напряжений, которая построена на микросборках с заливкой в металлических корпусах. На вход БСП подаётся напряжение 12В постоянного тока от АКБ. Это напряжение в процессе работы МР (особенно в моменты пуска маршевых двигателей) может колебаться от 9В до максимального напряжения заряда АКБ. На выходах БСП

формируется ряд стабилизированных напряжений: 48В (ток нагрузки не более 0,2А), 12В, 9В и 5В (ток нагрузки суммарно не более 1,6А).

Управляющий релейный модуль на 4 канала состоит из 4-х реле типа SRD-05VDC-SL-C (контактная группа on-off, 10А, 30VDC), управление состоянием которых осуществляется от микроконтроллера СУ «Робокон». Все контактные группы реле (4 группы по 3 контакта) выведены на крышку МР через два 6-контактных разъёма MF-6MRB, что обеспечивает возможность подключения захватного устройства.

Литий ионный аккумулятор HRB 11.1v 6000mAh обеспечивает питание всех электронных и электромеханических узлов МР (кроме нетбука и ультразвуковых датчиков), ёмкость 6 А-ч и номинальное напряжением 11.1v при этом максимальное напряжение аккумулятора достигает 12.6v. Полный разряд аккумулятора происходит за 1.5 часа при непрерывной работе маршевых двигателей. Так как в среднем при максимальной скорости робота электронная часть совместно с двигателями потребляет ток 4А. Контроль заряда АКБ осуществляется с помощью специальной пороговой схемы с зелёным и красным светодиодами, которые выведены на верхнюю крышку МР. При работе МР, если напряжение на АКБ составляет более 9,5В, горит зелёный светодиод; в противном случае начинает мигать красный светодиод. Можно организовать контроль заряда АКБ с помощью АЦП любого из 2-х микроконтроллеров («Робокон» или Arduino UNO), измеряя с их помощью напряжения на специальных выводах БСП, диапазон напряжения на которых изменяется в диапазоне 0 – 5В пропорционально напряжению на АКБ. Заряд АКБ следует производить строго с использованием балансировочного зарядного устройства во избежание возгорания из-за разности напряжений на банках АКБ. При этом ток нагрузки устанавливать не больше 3А (половина от величины ёмкости в амперах). Внешний вид аккумулятора представлен на рисунке 2.5.

Рисунок 2.5 – Аккумулятор HRB 11.1v 6000mAh

3 Создание программного и алгоритмического обеспечения

3.1 Обзор существовавшего программного обеспечения

Программное обеспечение АМУР 105 состояло из внешних и бортовых программ, рассмотрим бортовые. Бортовое управление осуществлялось на базе ПК (нетбука) ASUS Eee PC 1011CX с платформой Linux Ubuntu 13.04, основные программы находились в каталоге /home/sensorika/drivers/. При использовании каталогов или работе с ними были необходимы требования системы, ввода пароля: sensorika.info. Программы на бортовом ПК (нетбуке), написанные лабораторией sensorika при поставке этих роботов: zmq_robot.py – содержали реализацию протокола обмена байтами (UDP) данных через локальную вычислительную сеть, engine_pavl.py – содержали основную программу где происходит обмен данными с платой Робокон (считывание данных с энкодеров, подача управления на двигатели, подача управления на дополнительные захватные устройства, считывание данных с ультразвуковых датчиков, установка цветовой палитры светодиодов, получение управления от внешнего управляющего устройства, активирует окно диагностики), test.py – содержало тестовую программу, которая включала: тест соединения с внешним управляющим устройством, тест подачи напряжения на колёса робота, тест получения данных с ультразвуковых датчиков. После загрузки бортового ПК (нетбука), появлялось окно диагностики, где можно было наглядно убедиться в подключении или отключке ультразвуковых датчиков или управляющей платы Робокон (зелёный цвет обозначал подключение, а красный отсутствие соединение с компонентой).

Внешнее программное обеспечение было установлено на ПК (ноутбуке) ASUS и состояло из программ: zmq_robot.py - содержало реализацию протокола обмена байтами (UDP) данных через локальную вычислительную сеть, engine_control.py – управляющая программа, позволяющая управлять роботом с использованием клавиатуры (описание приведено в таблице 1), robotree.py – программа активирующая пользовательское приложение, test.py – содержало тестовую программу подключения к роботу, отправки и получения данных.

Т а б л и ц а 1 – Описание действий по нажатию кнопок

Клавиша	Действие при нажатии
[w]	Подать команду на движение вперёд
[a]	Подать команду на движение влево
[s]	Подать команду на движение вправо
[d]	Подать команду на движение назад

3.2 Обзор модернизированного программного обеспечения

В бортовое программное обеспечение были добавлены файлы: user_manual.txt – содержит руководство пользователя по пуско-наладке робота, move_log.txt – регистрирует данные о движении и состоянии робота (записывается через изменённый файл engine_pavl.py), ultrasound_driver.py - активирует считку данных с ультразвуковых датчиков. Также для быстрого автоматического запуска программы в фоновом режиме при активации компьютера был установлен supervisor (инструкция по отключению и включению есть в файле user_manual.txt). Основные программы на бортовом ПК (нетбуке), такие как engine_pavl.py и zmq_robot.py не обновлялись выше версии 2.7, так как данные

передавались по байтам и на внешнем ПК их также можно было принимать, хотя метод socket на внешнем ПК и другой, также на бортовом ПК (нетбуке) не происходило взаимодействие с внешними или обновлёнными библиотеками.

Из внешнего программного обеспечения были взяты программы: robotree.py, zmq_robot.py и перенесены с версии Python 2.7 на версию Python 3.9, так как возникла необходимость работать с библиотеками MediaPipe, OpenCV, Qt. Было заменено множество названий и методов на аналогичные или доработанные самостоятельно, что при этом функционал вышеприведённых программ не изменился. Далее были добавлены управляющие программы: engine_start.py – программа в которой происходит получение данных с бортового ПК (нетбука) их обработка и отправка управляющего воздействия непосредственно обратно на бортовой ПК (нетбук) робота, camera.py – реализует получение видеопотока с камеры робота по IP, NavigationSys.py – реализует расчёт кинематики робота, PID.py – реализует универсальный ПИД регулятор, odometry.py – позволяет подключится отдельно от основной программы (engine_start.py) для анализа и настройки оптических энкодеров, us_gui.py – позволяет подключится отдельно от основной программы (engine_start.py) для анализа и настройки ультразвуковых датчиков. Действия по нажатию клавиш, приведённых в таблице 1, были расширены и приведены в таблице 2. Программа, отвечающая за движение робота путём нажатия клавиш на клавиатуре – move_keyboard.py.

Т а б л и ц а 2 – Описание действий по нажатию кнопок

Клавиша	Действие при нажатии
[w]	Подать команду на движение вперёд
[a]	Подать команду на движение влево
[s]	Подать команду на движение вправо
[d]	Подать команду на движение назад
[space]	Подать команду остановки
[esc]	Завершить работу программы

Внешняя программа имеет объектно-ориентированную архитектуру. Диаграмма классов, описывающая структуру ИУС, представлена на рисунке 3.1.

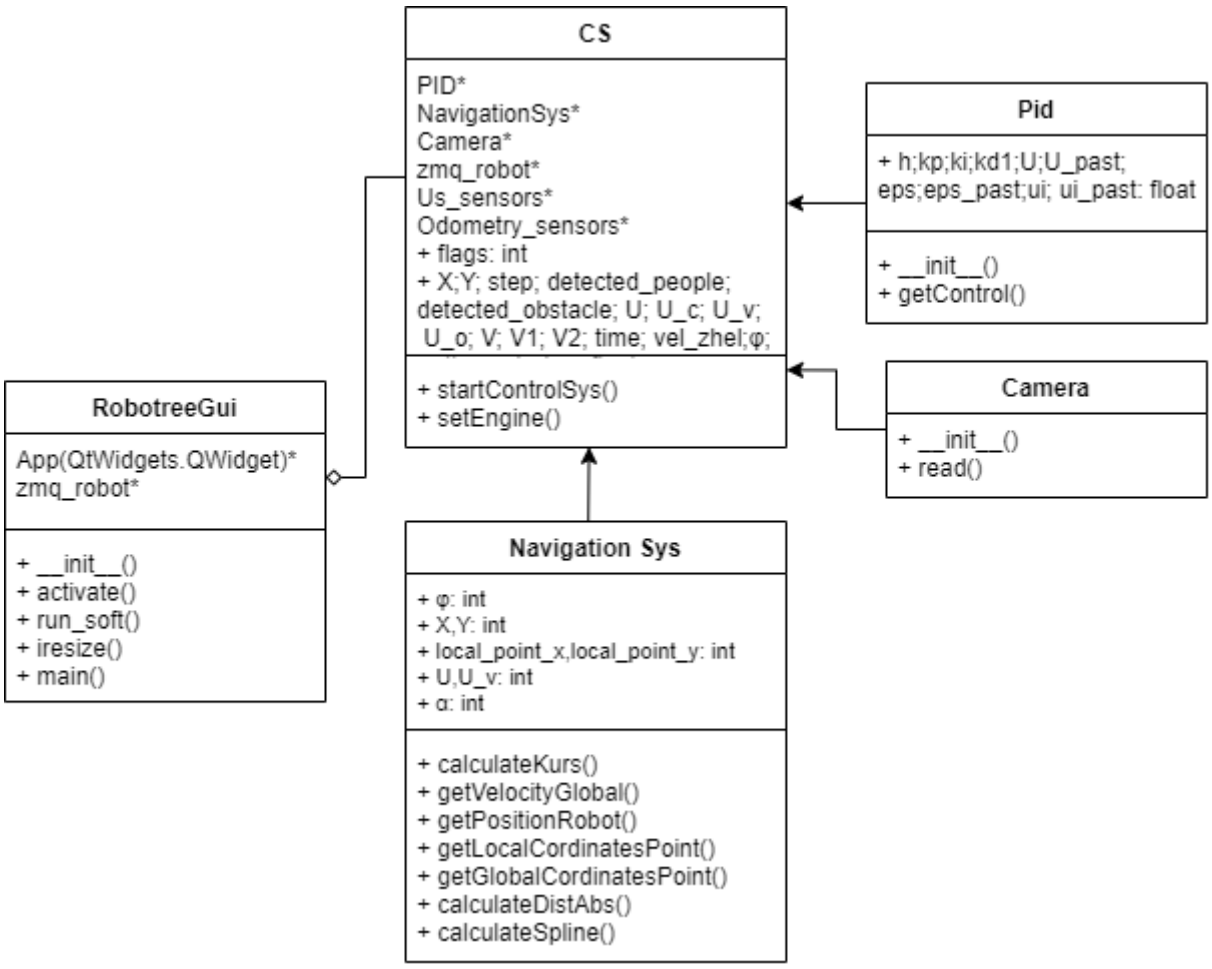

Рисунок 3.1 – Диаграмма классов ИУС

Класс RobotreeGui: данный класс описывает структуру приложения: поля, строки ввода, надписи и т.д. Класс CS: данный класс описывает логику работы всей системы. В нём хранятся переменные, для движения, распознавания объектов, распознавания человека. В нём содержаться объекты классов PID, Camera, NavigationSys, также данный класс запускает работу отдельного потока камеры для чистки буфера. Класс NavigationSys: данный класс содержит в себе методы для расчёта положения робота, его ориентации, методы для расчёта угла курса и метод построения сплайна, получения координат точки в локальной системе координат, метод расчёта расстояния до целевой точки. Класс Camera: данный класс необходим для получения изображения с камеры робота. Данный класс работает на отдельном потоке. Класс PID: данный класс необходим для расчёта управляющих сигналов, на основе входного и желаемого значений. Инициализирующая функция __init__ устанавливает коэффициенты в зависимости от регулятора.

Также необходимо отметить диаграмму состояний робота, которая приведена на рисунке 3.2.

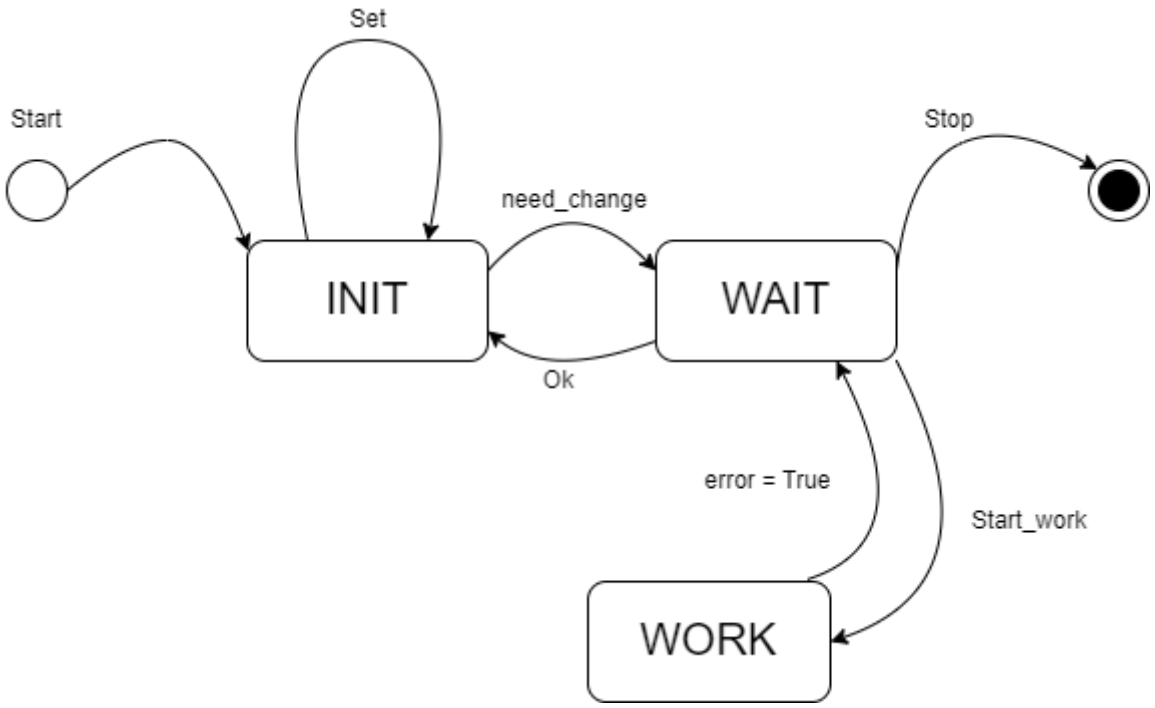

Рисунок 3.2 – Диаграмма состояний ИУС

После получения команды start, система переходит в состояние INIT, в котором происходит опрос всех возможных ip адресов локальной сети АМУР для выявления роботов готовых к работе. Далее идёт состояние ожидания WAIT это состояние выбора конкретного робота или выход из программы, если ожидание слишком велико, при этом робот в данном состоянии не двигается. WORK - состояние является основным для ИУС, и в нем происходят все действия и расчеты, связанные с работой МР, т.е. состояние, когда система выполняет работу по управлению роботом. В данном состоянии считываются сигналы датчиков, вычисляется местоположение препятствий, осуществляется формирование выходных управляющих сигналов для движителей. При получении ошибки, система переходит в состояние ожидания и в последствии останавливается.

Текущее программное обеспечение позволяет роботу выполнять различные задачи, такие как: подъезд к целевой точке по прямой с заданной скоростью, движение по сплайну с заданной скоростью, распознавание человека и подъезд к нему, обнаружение препятствий и их объезд. Текущее программное обеспечение используется на ноутбуке Honor MagicBook 15 с операционной системой Windows, но также может быть установлено на любой ПК с версией Python 3.9 и средой разработки Microsoft Visual Studio начиная с любой версии 2019 года.

3.3 Реализация алгоритма расчёта навигации

Для определения положения робота необходимо знать ориентацию и смещение робота относительно некоторой базовой системы координат, скорость робота можно вычислить как линейное смещение делёное на промежуток времени.

Начальное положение робота фиксирует глобальную систему координат (АСК) и относительно неё строится вся траектория. Связанная система координат (ССК) тесно связана с корпусом робота. В начальный момент времени АСК и ССК совпадают.

Направление осей АСК и ССК в начальный момент времени приведены на рисунке 3.3, направление осей АСК и ССК в произвольный момент времени приведены на рисунке 3.4.

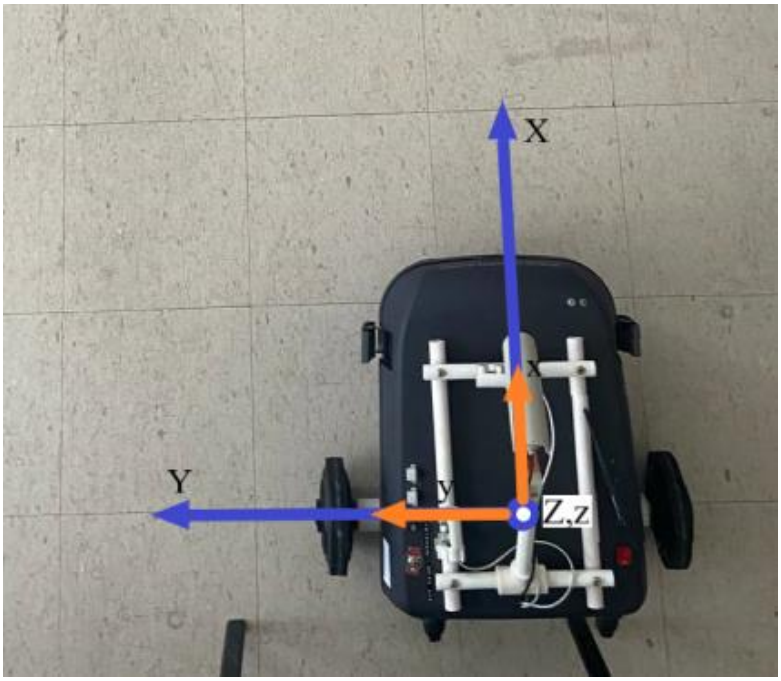

Рисунок 3.3 – Положение ССК и АСК в начальный момент времени

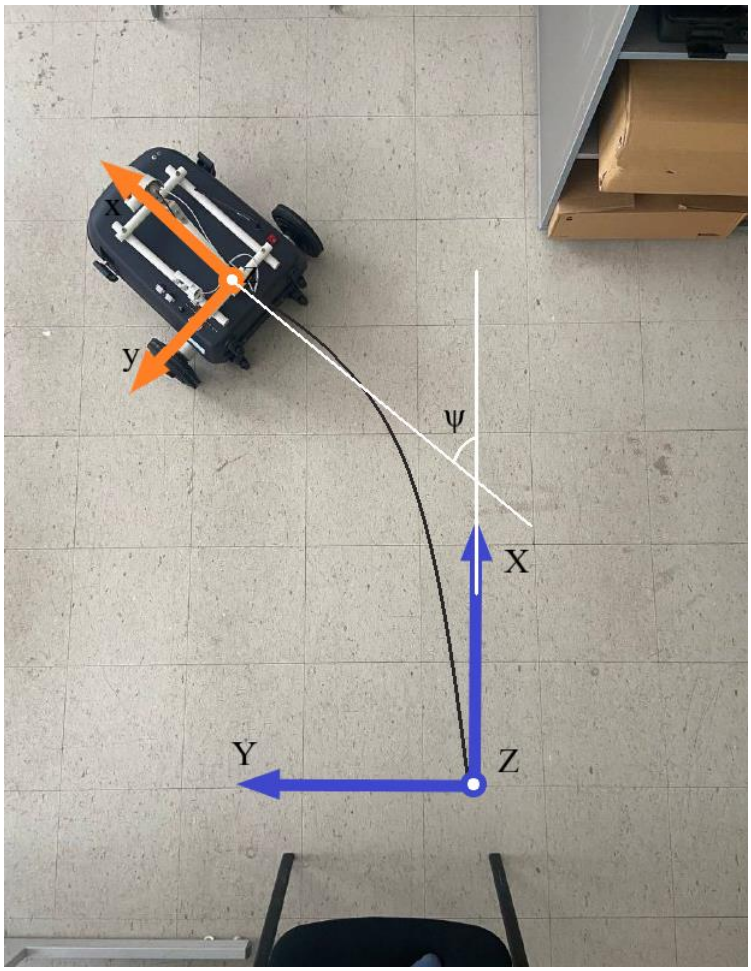

Рисунок 3.4 – Положение ССК и АСК при движении к произвольной точке

Угол поворота ССК относительно АСК вокруг оси Z – угол ψ. Найти данный угол возможно решив систему уравнений (2), пояснение к ней отображены на рисунке 3.5. Также важно отметить, что движение робота происходит только вперёд, при движении на месте вокруг своей оси или назад необходимо будет использовать другие расчёты.

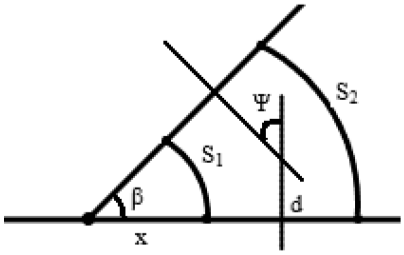

Рисунок 3.5 – Пояснение к нахождению угла ССК относительно АСК

$$S_i = \varphi_i R; \tag{1}$$

$$\begin{cases} S_1 = \beta x \\ S_2 = \beta(x + d) \end{cases}; \tag{2}$$

где:

- S_i – длина произвольной дуги в [м];
- S_1 – длина дуги от левого колеса в [м];
- S_2 – длина дуги от правого колеса в [м];
- φ_i – угол поворота колеса в [рад];
- x – расстояние от точки вращения до левого колеса в [м];
- β – угол поворота оси колёс робота относительно точки вращения в [рад];
- d – расстояние между колёсами робота в [м];
- ψ - угол поворота ССК относительно АСК по оси Z [рад].

Решая данную систему, находим угол β, который будет равен искомому углу ψ.

$$\psi = \beta = \frac{S_2 - S_1}{d}; \tag{3}$$

После нахождения угла ψ необходимо определить модуль скорости робота. На рисунке 3.6 изображено пояснение к системе (5).

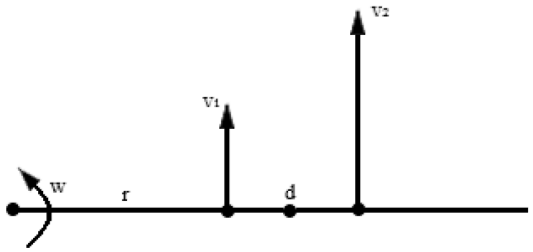

Рисунок 3.6 – Пояснение к нахождению модуля скорости

$$v_i = \frac{S_i}{h}; \tag{4}$$

$$\begin{cases} w = \dfrac{v_1}{r} \\ w = \dfrac{v_2}{r+d} \\ w = \dfrac{V}{r+\dfrac{d}{2}} \end{cases}; \tag{5}$$

где:

- S_i – длина произвольной дуги в [м];
- v_i – линейная скорость колеса в [м/с];
- v_1 – линейная левого колеса в [м/с];
- v_2 – линейная правого колеса в [м/с];
- V – линейная скорость центра робота (модуль скорости) в [м/с];
- d – расстояние между колёсами робота в [м];
- r – расстояние от точки вращения до левого колеса в [м];
- w – угловая скорость вращения оси, проходящей через колёса и центр робота в [рад/с];
- h – промежуток времени или шаг программы [с].

Решая систему (5) получаем модуль скорости робота в АСК:

$$V = \frac{v_1 + v_2}{2}; \tag{6}$$

Тогда для определения координат робота в АСК можно использовать следующие уравнения:

$$x(t) = x(t-1) + V cos(\psi) h; \tag{7}$$

$$y(t) = \mathrm{y(t-1)} + V sin(\psi) h; \tag{8}$$

В процессе работы определение желаемой точки в ССК необходимо для расчёта угла курса на каждой итерации, для это используется матрица поворота относительно оси Z. При этом добавив вектор линейного смещения можно получить матрицу перехода из АСК в ССК и наоборот. Для получения координат точки в ССК:

$$A = \begin{bmatrix} cos(\psi) & -sin(\psi) & 0 & X \\ sin(\psi) & cos(\psi) & 0 & Y \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \tag{9}$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = A^{-1} \begin{bmatrix} x_1 \\ y_1 \end{bmatrix}; \tag{10}$$

где:

- x, y – координаты робота в ССК [м];

- x_1, y_1– координаты робота в АСК в [м];
- A – матрица перехода.

Далее необходимо произвести расчёты угла курса при этом важно ограничить области, чтобы не возникало неопределённостей.

$$\alpha = \begin{cases} 0 \text{ при } x = 0 \text{ и } y = 0 \\ arctg(y/x) \text{ при } (x > 0 \text{ и } y > 0) \text{ или } (x > 0 \text{ и } y < 0) \\ 180 + arctg(y/x) \text{ при } x < 0 \text{ и } y > 0 \\ -180 + arctg\left(\frac{y}{x}\right) \text{ при } x < 0 \text{ и } y < 0 \\ 180 \text{ при } y = 0 \text{ и } x < 0 \\ 90 \text{ при } x = 0 \text{ и } y > 0 \\ 0 \text{ при } y = 0 \text{ и } x > 0 \\ -90 \text{ при } x = 0 \text{ и } y < 0 \end{cases}; \quad (11)$$

Далее необходимо настроить ПИД регулятор по курсу и по скорости. Формула ПИД регулятора – формула (12).

$$u(t) = K_p e(t) + I(t-1) + K_i e(t) + K_d\big(e(t) - e(t-1)\big)/h; \quad (12)$$

где:

- K_p, K_i, K_d – пропорциональный, интегральный, дифференциальный коэффициенты;
- $e(t)$ – ошибка на текущем шаге;
- $e(t-1)$ – ошибка на предыдущем шаге;
- $I(t-1)$ – значение интегральной компоненты на предыдущем шаге;
- h – промежуток времени или шаг программы [с];
- $u(t)$ – текущее значение управляющего сигнала.

Пропорциональный коэффициент отвечает за влияние на результат, увеличение пропорционального коэффициента повышает быстродействие, но снижает запас устойчивости системы. Интегральный коэффициент позволяет устранять резкий эффект пропорциональной компоненты, но при этом будет нарастать ошибка, что может привести к раскачке системы. Дифференциальный коэффициент пропорционален темпу изменений, увеличение дифференциальной составляющей увеличивает запас устойчивости и быстродействие системы.

Для подбора коэффициентов в данной задаче была выбрана произвольная точка с координатами (3,1). Коэффициенты K_i, K_d обнулялись и производился подбор коэффициента K_p. Далее в зависимости от резкости, время перерегулирования и наличия или отсутствия статистической ошибки эмпирическим путём подбирались коэффициенты K_i, K_d. Для регулятора по углу курса были подобраны коэффициенты $K_p = 2, K_i = 0, K_d = 0.1$, для регулятора по скорости $K_p = 160$, $K_i = 0$, $K_d = 0.03$. Графики угла курса и скорости отображены на рисунках 3.7-3.8.

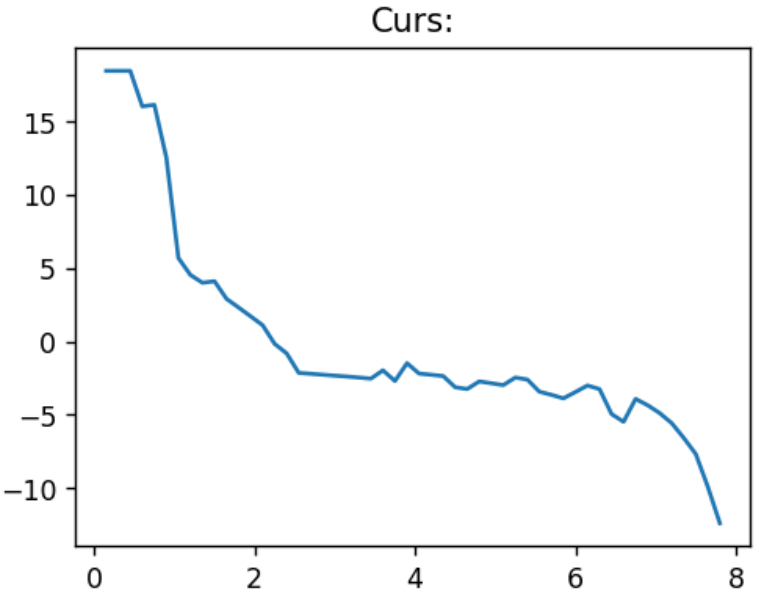

Рисунок 3.7 – Изменение угла курса

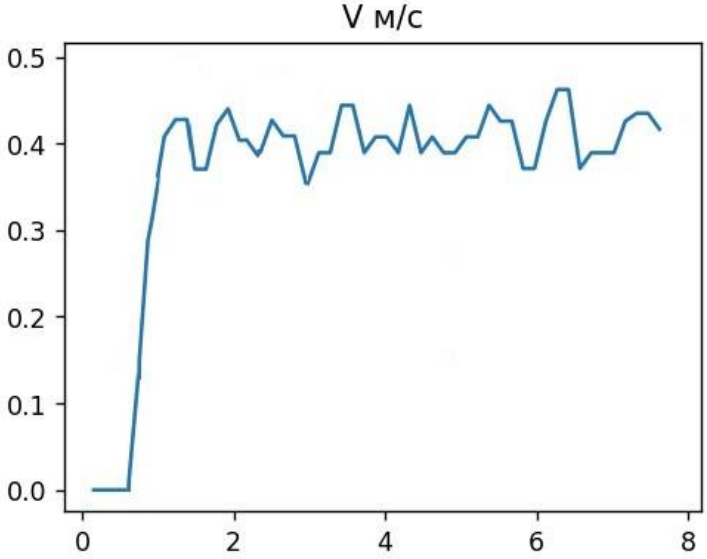

Рисунок 3.8 – Изменение скорости робота

Далее производился выбор сплайна. Сплайн необходим для получения промежуточных точек (интерполяции) и построения непрерывной гладкой траектории движения робота. Расчёт координат сплайна производится методом calculateSpline() с помощью сплайна третьего порядка вида:

$$\begin{cases} S(x) = S_i(x) = a_i + b_i(x - x_{i-1}) + c_i(x - x_{i-1})^2 + d_i(x - x_{i-1})^3 \\ x_{i-1} \le x \le x_i \end{cases}; \quad (13)$$

Для определения коэффициентов a_i, b_i, c_i, d_i на всех n элементарных отрезках необходимо получить $4n$ уравнений. Данные уравнения вытекают из условия прохождения кубического сплайна, таких, как прохождения сплайна через заданные точки и условие гладкости второго порядка, то есть:

$$S_i(x_{i-1}) = y_{i-1}; \quad (14)$$
$$S_i(x_i) = y_i; \quad (15)$$
$$S'_i(x_i) = S'_{i+1}(x_i); \quad (16)$$
$$S''_i(x_i) = S''_{i+1}(x_i); \quad (17)$$

Подставляя полученные выражения, получаем следующие уравнения для определения неизвестных коэффициентов:

$$a_i = y_{i-1}; \quad (18)$$
$$d_i = \frac{c_{i+1} - c_i}{3h_i}; \quad (19)$$
$$b_i = \frac{y_i - y_{i-1}}{h_i} - \frac{h_i}{3}(c_{i+1} + 2c_i); \quad (20)$$

Видно, что коэффициенты b_i и d_i зависят от c_i, составим следующую систему уравнений для коэффициента c_i:

$$h_{i-1}c_{i-1} + 2(h_{i-1} + h_i)c_i + h_i c_{i+1} = 3\left(\frac{y_i - y_{i-1}}{h_i} - \frac{y_{i-1} - y_{i-2}}{h_{i-1}}\right); \quad (21)$$

При свободном закреплении концов сплайна можно приравнять нулю кривизну линии в точках закрепления. Получаемая таким образом функция называется свободным кубическим сплайном. Отсюда следует, что:

$$S'_i(x_0) = b_1 = k_1; \quad (22)$$
$$S'_i(x_n) = b_{n+1} = k_2; \quad (23)$$
$$S''_i(x_0) = 2c_1 = m_1 = 0; \quad (24)$$
$$S''_i(x_n) = 2c_{n+1} = m_2 = 0; \quad (25)$$

Таким образом можно составить систему линейных уравнений, решаемую с помощью метода обратной матрицы, вида $Hc = F$, где:

$$H = \begin{bmatrix} 2(h_{i-1} + h_i) & h_i & 0 & \dots & 0 \\ h_{i-1} & 2(h_{i-1} + h_i) & h_i & \dots & 0 \\ 0 & h_{i-1} & 2(h_{i-1} + h_i) & \dots & 0 \\ \dots & \dots & \dots & \dots & h_i \\ 0 & 0 & 0 & h_{i-1} & 2(h_{i-1} + h_i) \end{bmatrix}; \quad (26)$$

$$c = \begin{bmatrix} c_2 \\ c_3 \\ c_4 \\ \dots \\ c_n \end{bmatrix}; \quad (27)$$

$$F_i = 3\left(\frac{y_i - y_{i-1}}{h_i} - \frac{y_{i-1} - y_{i-2}}{h_{i-1}}\right); \quad (28)$$

Реализация полученного сплайна при входном массиве точек $X = [0,2,3,4,5], Y = [0,2,-3,4,-5]$, отображена на рисунке 3.9.

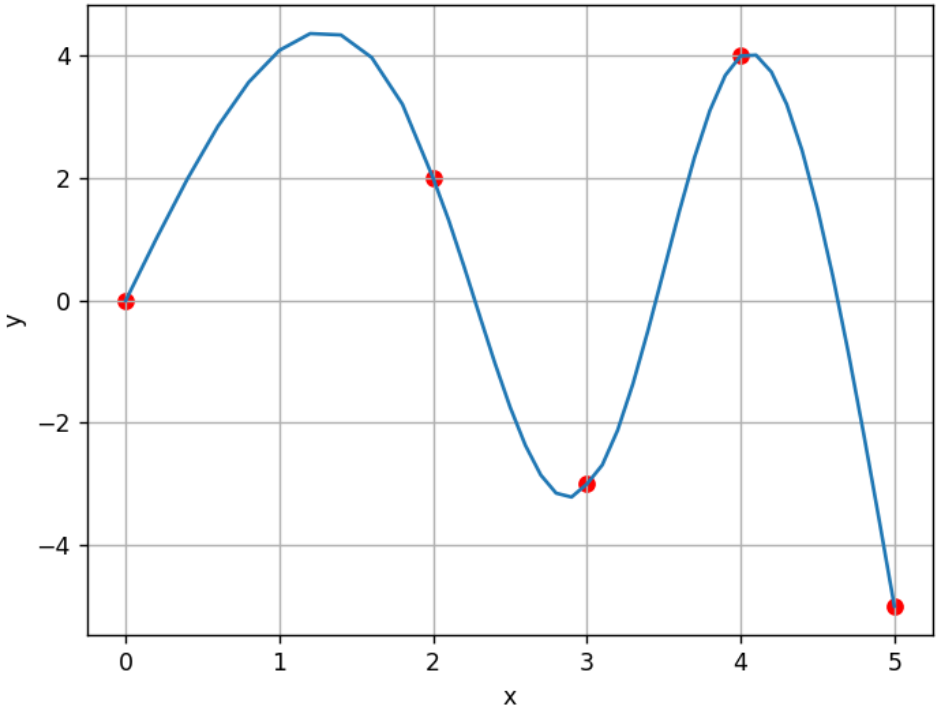

Рисунок 3.9 – График построения кубического сплайна

3.4 Реализация системы технического зрения мобильного робота

При реализации системы технического зрения изначально необходимо было откалибровать камеру и получить с неё изображение. Для получения изображения использовалась библиотека OpenCV. OpenCV – библиотека открытого исходного кода для компьютерного зрения и машинного обучения. Библиотека написана на языках программирования C++ и Python и поддерживает работу на платформах Windows, Linux, macOS, Android и iOS. Она содержит более 2500 алгоритмов для обработки изображений и видео, включая алгоритмы для обнаружения объектов, распознавания лиц, трекинга движения, анализа настроения и т.д. OpenCV имеет широкий круг приложений в сферах автоматизации промышленности, робототехники, медицинской диагностики, безопасности и видеонаблюдения, а также в различных областях искусственного интеллекта, таких как обучение с подкреплением, нейросети и глубокое обучение. Калибровка камеры производилась за счёт изменения фокусного расстояния на самой камере и размера изображения в программе. Для обеспечения высокой скорости обработки изображений был выбран размер изображения 300 на 400 пикселей, изображение с камеры после калибровки представлено на рисунке 3.10.

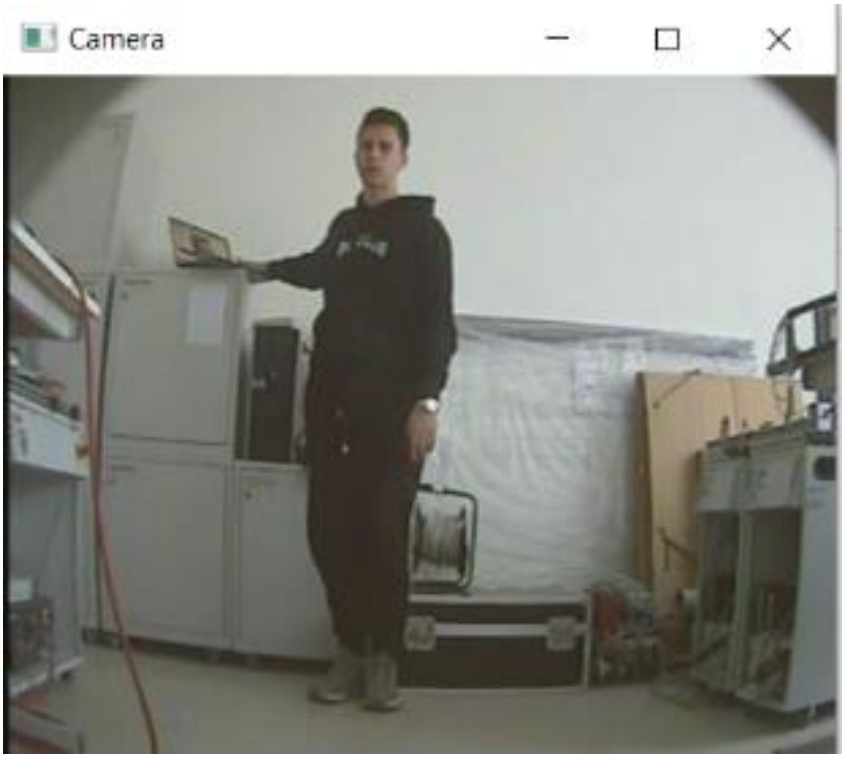

Рисунок 3.10 – Изображение с камеры

Важным аспектом при получении и обработки данных было в чистке буфера, чтобы не возникало замедления входящего потока, и система работала в реальном времени. Реализован данный процесс был путём запуска параллельного потока на считку кадров (в момент обработки кадров считывается текущий, тем самым удаляя из буфера старые кадры). Среднее время между кадрами составляло 0.05с.

Для решения задачи обнаружения человека одной библиотеки OpenCV было недостаточно, так как методы данной библиотеки позволяли лишь распознать лицо человека, а не всё тело. Решения распознавания всего тела или “скелета” человека предлагали две нейронные сети основанные на методах библиотеки машинного обучения TensorFlow такие как: MediaPipe и YOLOv7. Обе библиотеки обеспечивают распознавание тела человека в реальном времени. Была выбрана библиотека MediaPipe из-за ряда её преимуществ необходимых в данной задаче, таких как: высокая производительность при низком разрешении входного кадра, хорошее распознавание удалённых объектов и скорость работы, при этом для решения задачи обнаружения нескольких людей или работе с высоким разрешением входного кадра лучше использовать нейронную сеть YOLOv7.

MediaPipe есть три модели для оценки позы:

- BlazePose GHUM Heavy,
- BlazePose GHUM Full,
- BlazePose GHUM Lite,

эти модели помечены как сложность 0, 1 и 2 соответственно. Для обеспечения скорости обработки была использована модель со сложностью 1.

Для отслеживания человека необходимо знать расстояние до человека и позицию на кадре, поэтому используем ориентиры (позиции какой-либо точки на кадре в координатах X и Y). Библиотека MediaPipe позволяет определить 33 ориентира на теле человека, при этом в данной работе для определения центра тела человека использовались только

ориентиры под номерами 11,12 (ориентиры плеч) отображённые на рисунке 3.11. Изображение тела человека на фоне других предметов отображено с использованием данной библиотеки приведено на рисунке 3.12.

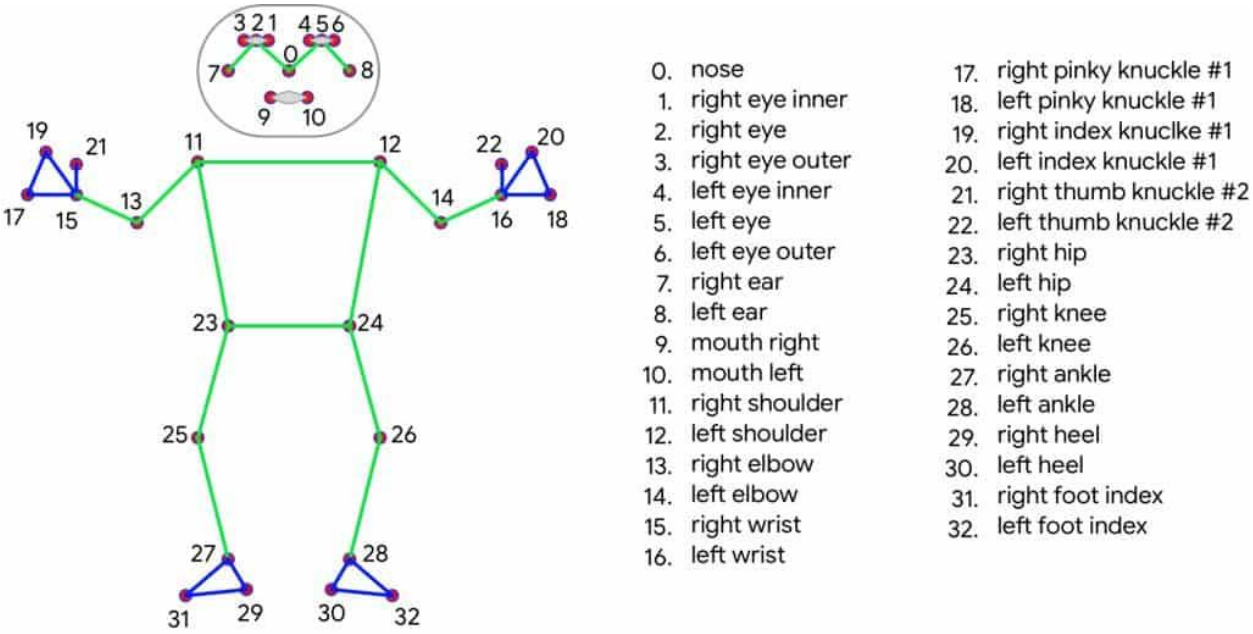

Рисунок 3.11 – Ориентиры на кадре

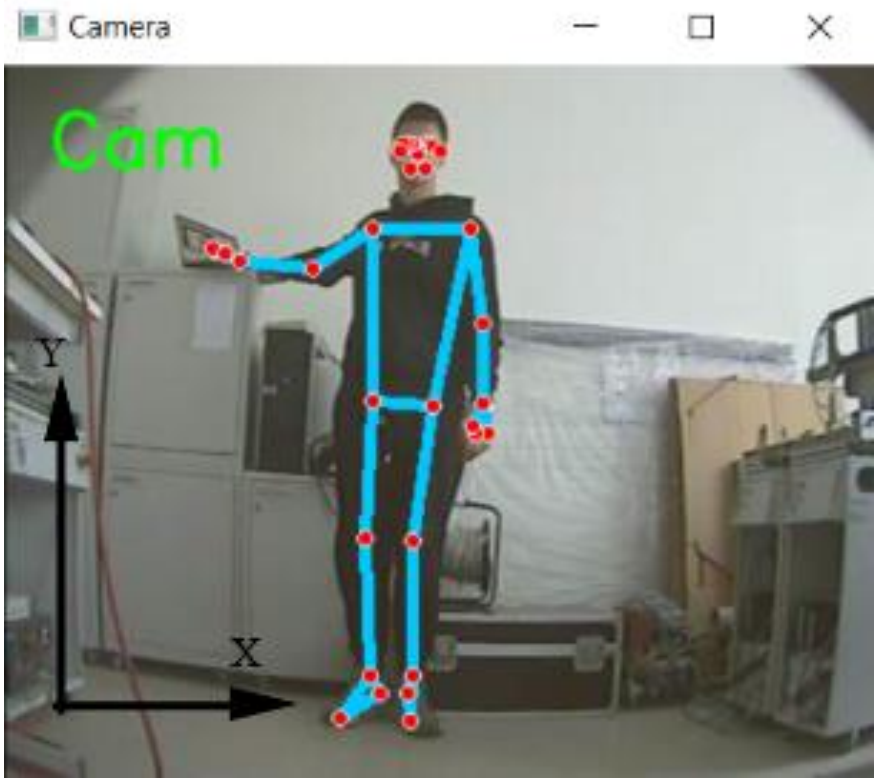

Рисунок 3.12 – Определение тела человека по камере

Каждый ориентир состоит из координаты X и Y, где каждой координате соответствует число от 0 до 1, чтобы узнать положение точки в пикселях по оси Y

необходимо домножить на высоту кадра и, чтобы узнать положение по оси X необходимо домножить на ширину кадра.

Для определения центра робота по оси X будем использовать формулу (29) и длины отрезка плеч формулу (30). Для определения расстояния воспользуемся правилом подобия треугольников (учитываем, что увеличение расстояние линейное). На рисунке 3.13 неизвестное расстояние обозначается Z', для отрезка L расстояние Z измерено вручную, соответственно формула нахождения расстояния Z' – формула (31).

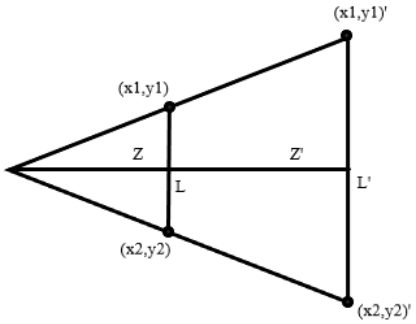

Рисунок 3.13 – Определение расстояния с использованием подобных треугольников

$$\bar{x} = (x_{12} + x_{11})/2; \tag{29}$$

$$L = \sqrt{(x_{12} - x_{11})^2 + (y_{12} - y_{11})^2}; \tag{30}$$

$$Z' = \frac{Z \cdot L'}{L}. \tag{31}$$

Далее в зависимости от положения центра человека посредством подачи управляющего воздействия с ПИД регулятора обеспечивается поворот робота, так чтобы человек находился посередине кадра. Расстояние необходимо для того, чтобы понять, когда начинать выполнение слежки за человеком, а также когда произвести остановку, в данном случае робот начинает распознавать человека на четырёх метрах и останавливается на дистанции одного метра. График управляющего воздействия на колёса при детектировании человека и слежении за ним представлен на рисунке 3.14.

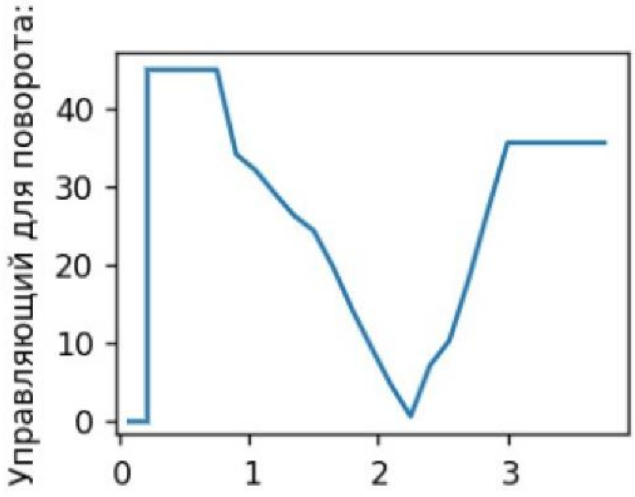

Рисунок 3.14 – График управляющего воздействия при детектировании человека

Для объезда препятствий возможным решением является использование ультразвуковых датчиков. В данной задаче они не применялись, так как препятствием служил стул с четырьмя ножками, который с помощью ультразвуковых датчиков никак робот корректно не сможет объехать. Решение задачи было в использовании той же библиотеки MediaPipe с методами распознавания конкретного объекта в данном случае стула (при необходимости можно поставить любой другой объект из доступных решений MediaPipe). Преимущества данного подхода заключаются в детектировании объекта с различных ракурсов и быстроте обработки кадров. Детектирование положения препятствия (стула) с различных ракурсов представлено соответственно на рисунках 3.15-3.17.

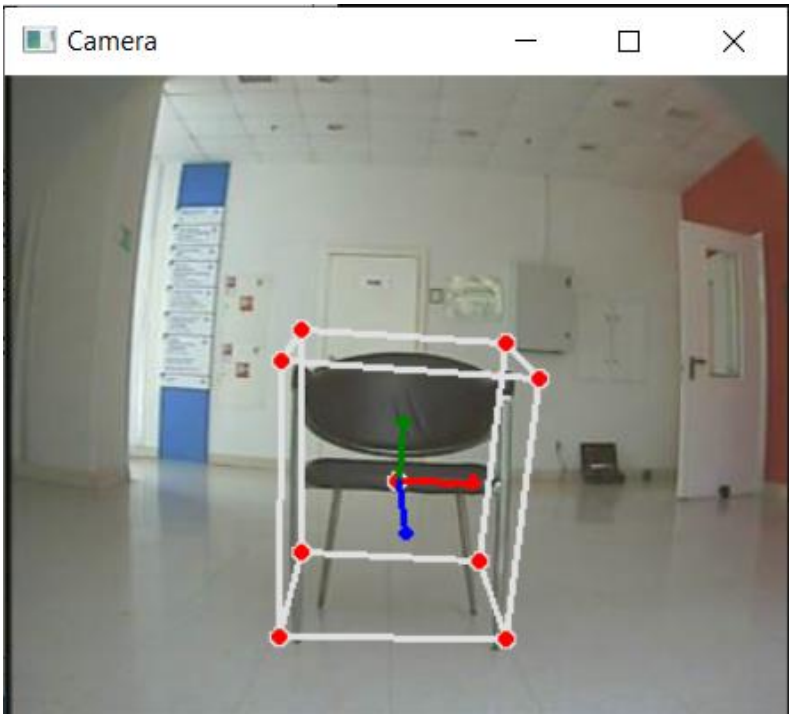

Рисунок 3.15 – Определение положение препятствия (вид спереди)

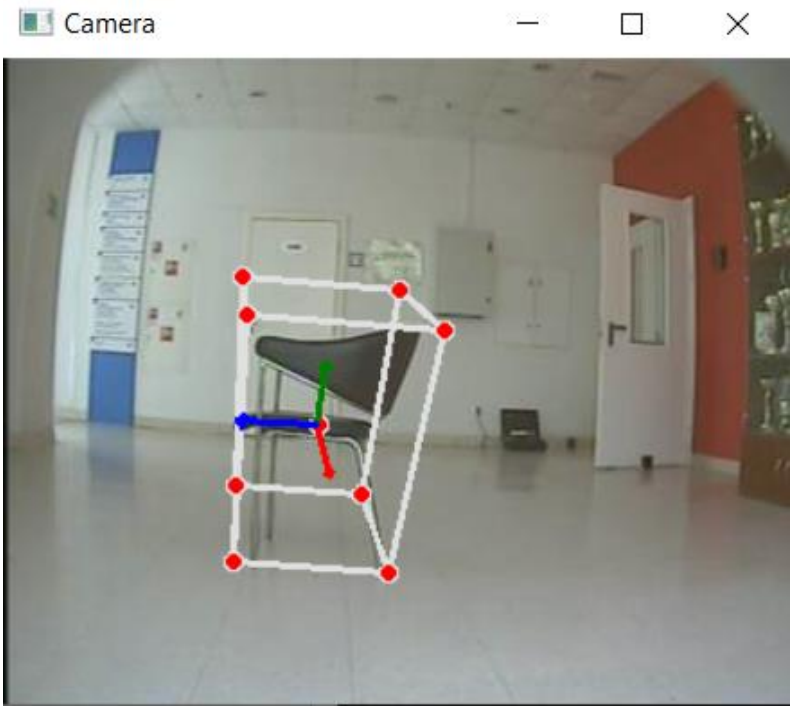

Рисунок 3.16 – Определение положение препятствия (вид сбоку)

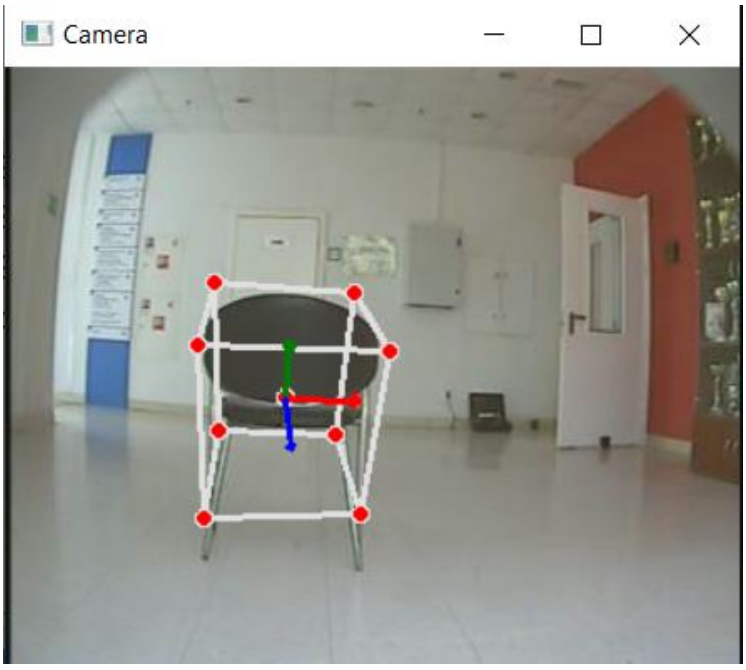

Рисунок 3.17 – Определение положение препятствия (вид сзади)

Детектирование препятствия начинается с расстояния в четыре метра и заканчивается съездом на исходную траекторию движения если при дистанции в два метра от препятствия центр препятствия находится в диапазонах 20% края кадра по оси X. Если при дистанции до препятствия остаётся чуть больше 2м и детектирование не произошло, то происходит остановка робота во избежание столкновения. Алгоритм расчёта управляющего сигнала аналогичный задаче с распознаванием человека, только здесь желаемое значение будет достигнуто, когда координата центра объекта приблизится к краю кадра.

4 Моделирование и эксперименты

Моделирование работы мобильного колёсного робота АМУР 105 производилось на траектории кубического сплайна с заданной скоростью. Для анализа и вывода служили графики угла курса, скорости робота в АСК, графики угла поворота ССК относительно АСК. Траектория движения робота представлена на рисунке 4.1.

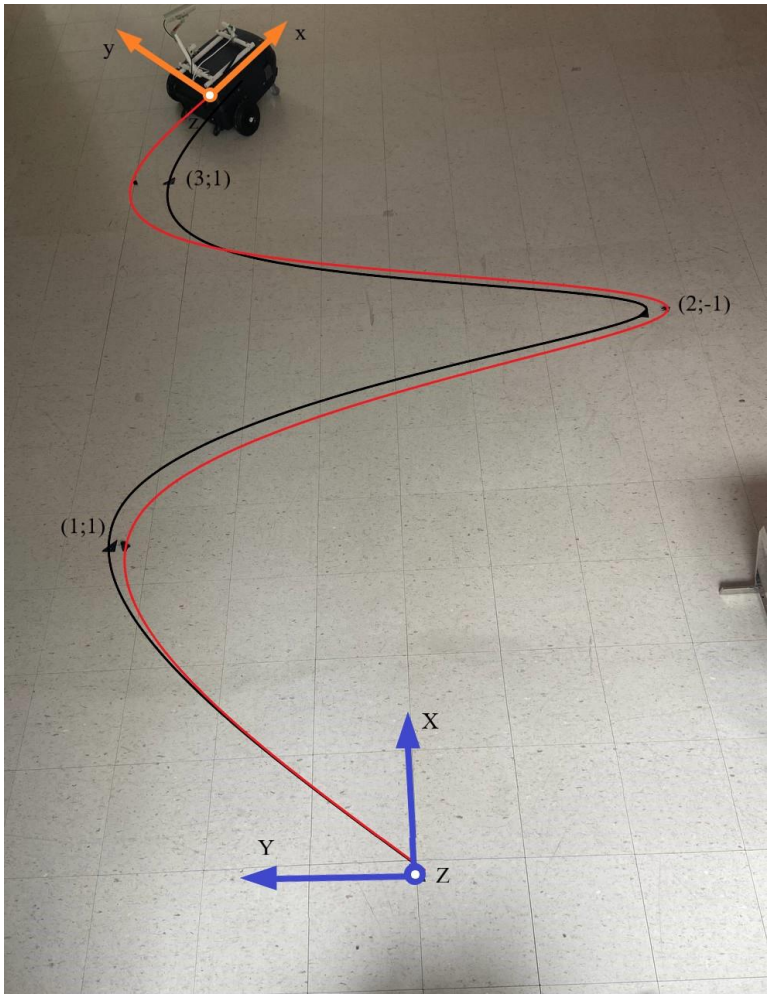

Рисунок 4.1 – Проезд робота по траектории сплайна

На рисунке 22 изображены две траектории: чёрная – желаемая траектория (рассчитывается вычислением координат сплайна), красная – реальная траектория движения робота. Изначально заданные точки для расчёта траектории сплайна $(1,1); (2,-1); (3,1)$. Желаемая скорость 0.35 м/с. Синие оси X, Y, Z обозначают АСК робота, а оранжевые x, y, z ССК робота. Как можно заметить из рисунка 21 реальная траектория схожа с желаемой, но полностью совпадать не может, так как в данном случае присутствует ошибка считывания оборотов энкодеров (может достигать 3 градусов за полный оборот), также не идеальное зацепление с поверхностью в результате которого может возникать проскальзывание колеса (в данном случае если оно есть, то достаточно небольшое, так как траектория схожа с желаемой). Угол курса, скорость движения робота, угол ССК относительно АСК для движения по данной траектории приведены на рисунках 4.2-4.4.

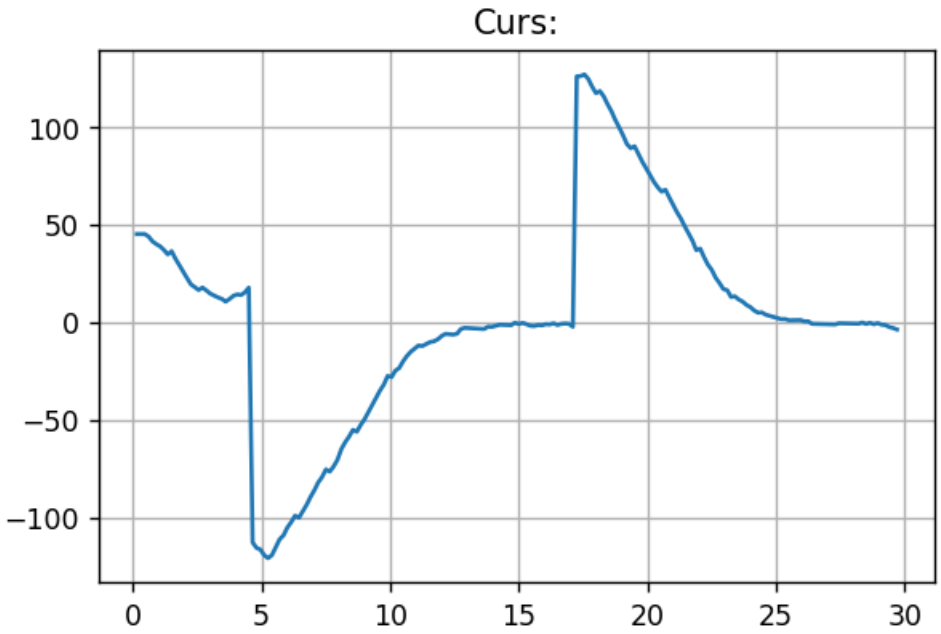

Рисунок 4.2 – График угла курса

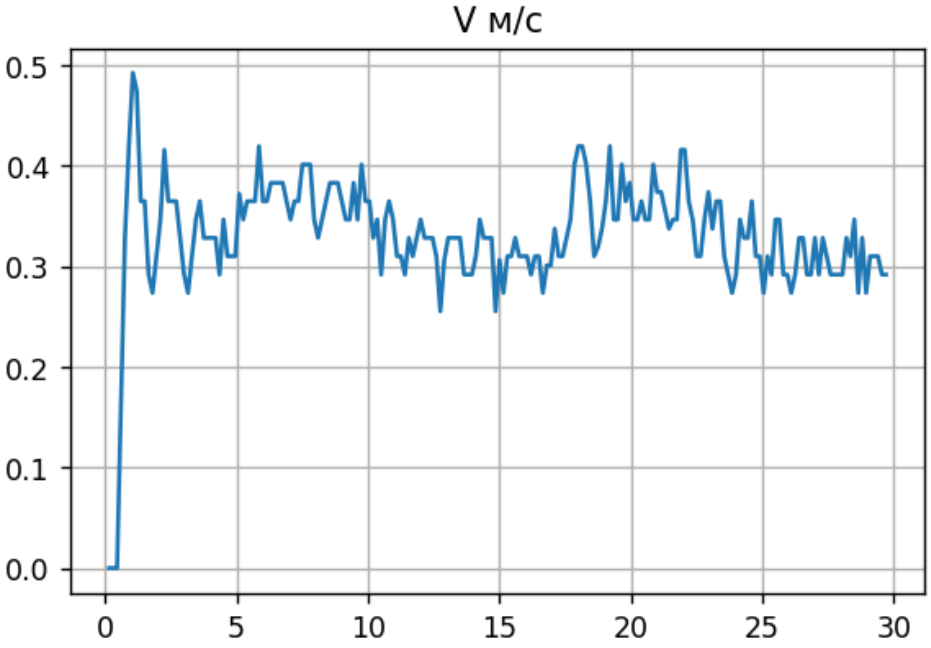

Рисунок 4.3 – График скорости робота

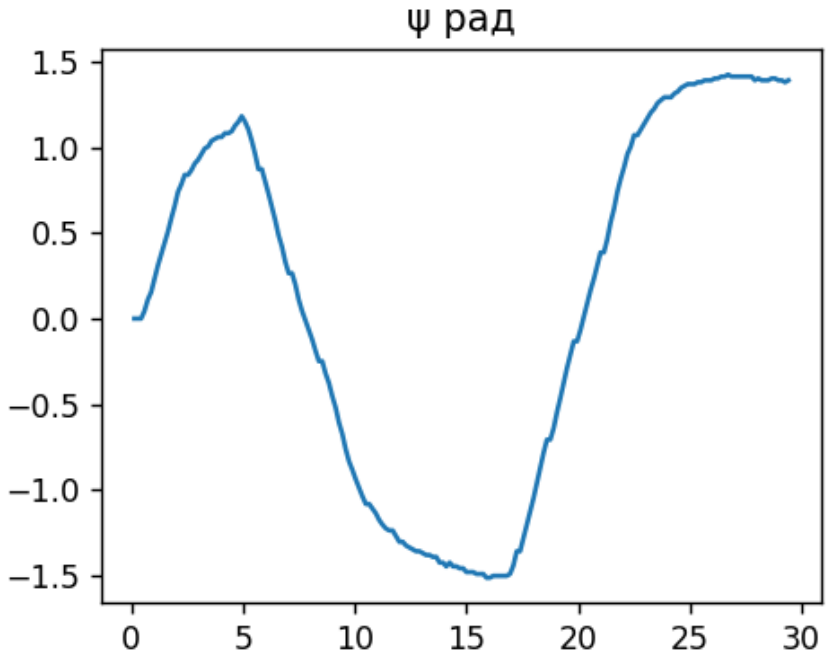

Рисунок 4.4 – График угла ССК относительно АСК

Как видно из графика угла курса робота, в начальный момент его значение составляет 45 градусов в соответствии с направлением на точку с координатами (1,1). Далее происходит плавное изменение до отметки в 10 градусов в момент времени 4.8 секунд (не до нуля т.к. робот попадает в область желаемой точки и происходит переключение). После чего на графике происходит резкий спад, вызванный переключением на следующую точку траектории, далее такое же резкое переключение возникает и в момент времени 17 секунд. Управляющие воздействия обеспечивают плавность поворота на желаемую точку, что свидетельствует о корректной настройке ПИД регулятора угла курса.

Из графика скорости робота в АСК видно, что значение скорости в 0.35 м/с выдерживается (что соответствует правильной настройке ПИД регулятора), но при этом возникают шумы измерений в диапазоне 0.05 м/с, что для такого робота вполне нормально, происходит это из-за достаточно большой ошибки получения данных с энкодеров и низкой частоты на которой работает управляющая система робота.

График угла ССК относительно АСК показывает, что изменение положения ССК корректно. В начальный момент времени угол равен нулю, так как в нулевом моменте времени оси совпадают. В промежутках между точками наблюдается монотонные возрастания или убывания соответственно.

Для проведения испытаний на распознавание человека была выбрана траектория движения к точке с координатами [1.5;0] по прямой, скорость при этом должна поддерживаться около 0.3 м/с. Начало детектирования человека находится на расстоянии трёх метров и заканчивается на одном метре, что и является условием остановки МР. На рисунке 4.5 и 4.6 приведены начальное и конечное изображение детектирования.

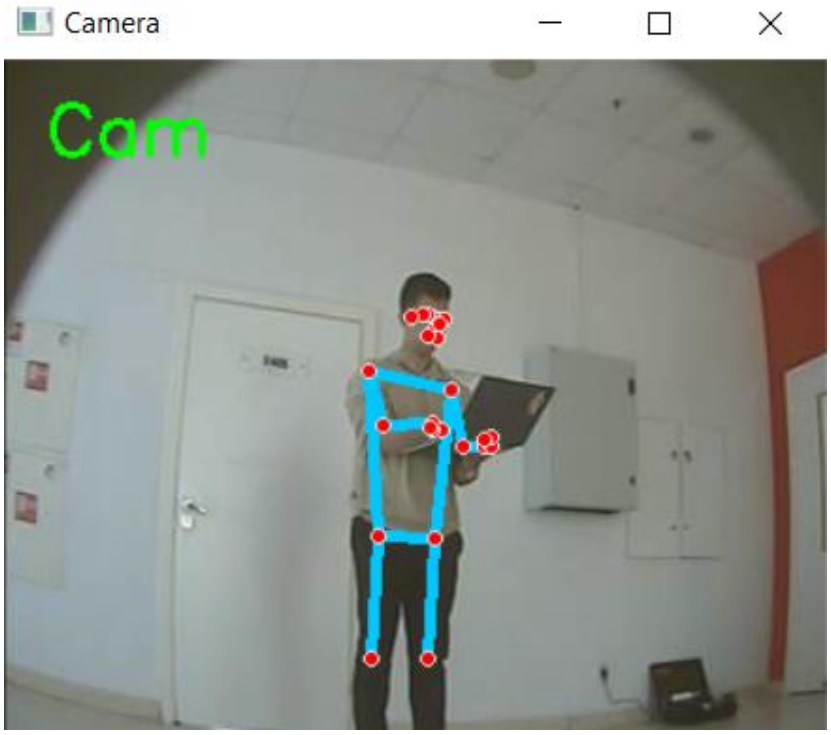

Рисунок 4.5 – Начальное положение детектирования

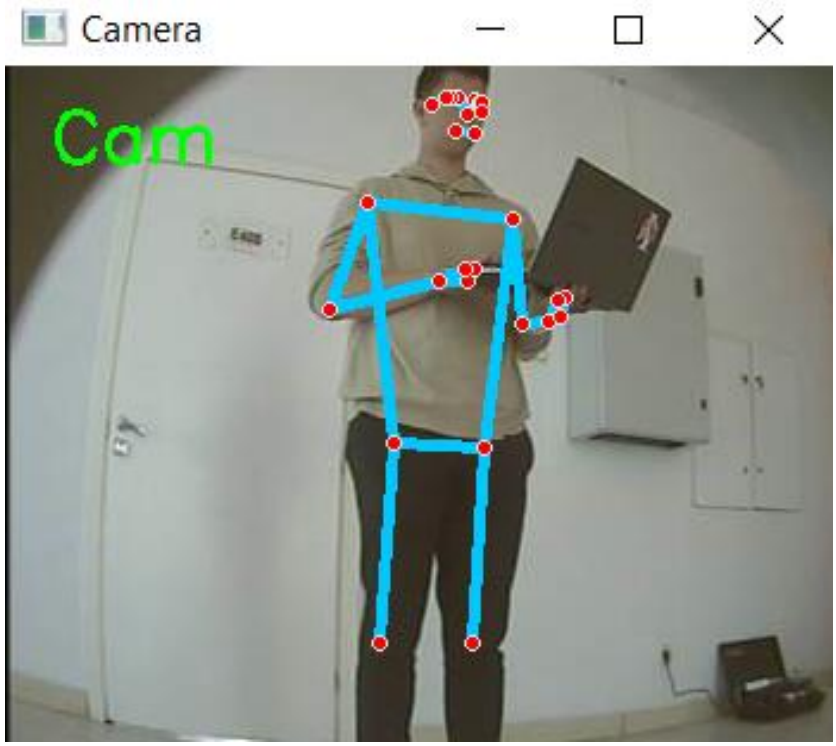

Рисунок 4.6 – Конечное положение детектирования

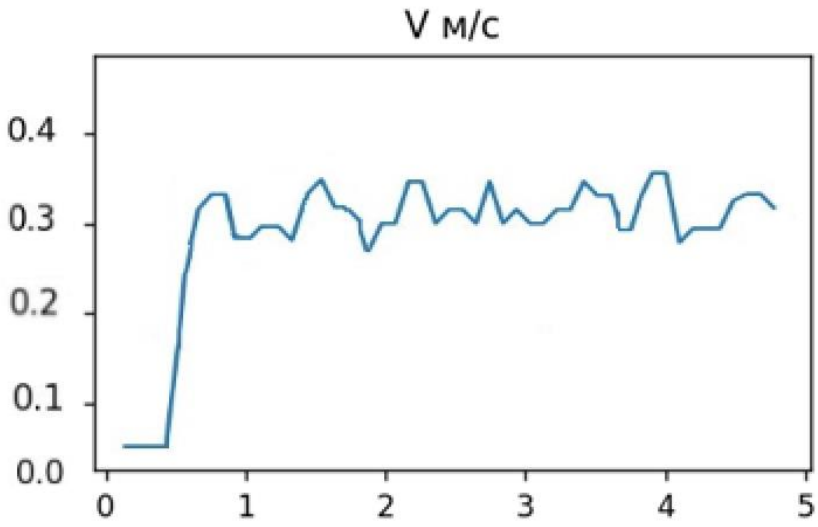

Рисунок 4.7 – График скорости при движении к человеку

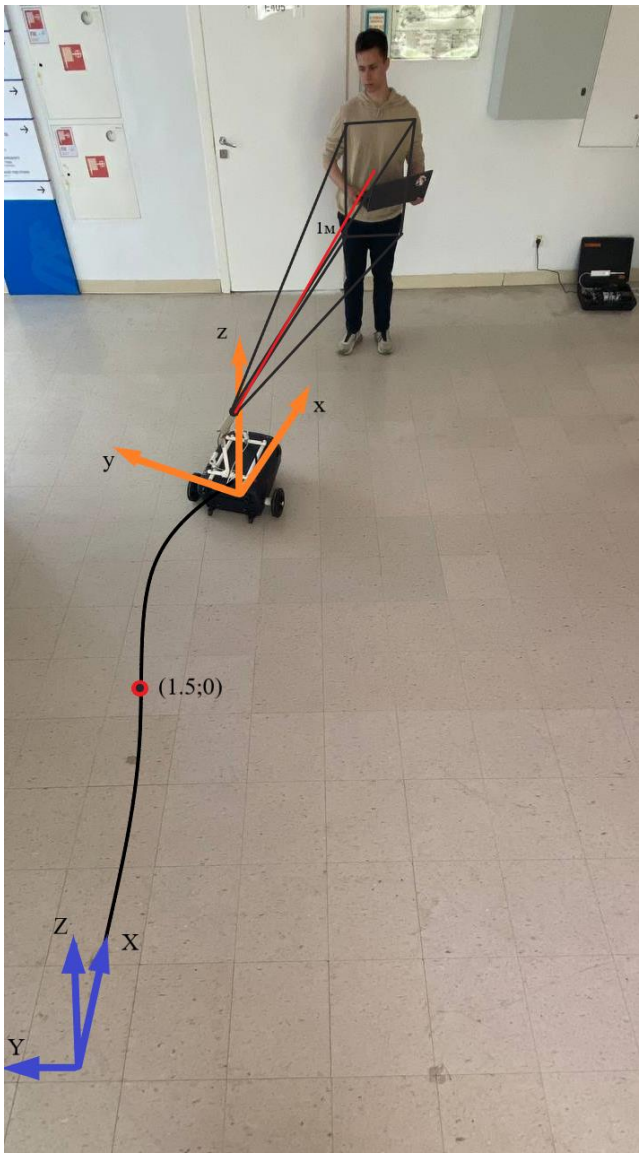

Рисунок 4.8 – Траектория движения робота при детектировании человека

Как видно из графика скорости 4.7 скорость в 0.3 м/с обеспечивается. На рисунке 4.8 представлена траектория движения МР, что тоже соответствует желаемой траектории движения МР при начале детектирования с расстояния 3 метра и остановке при 1 метре.

Далее на рисунке 4.9 представлена траектория движения МР при объезде препятствия в виде стула. Движение соответствует желаемой траектории.

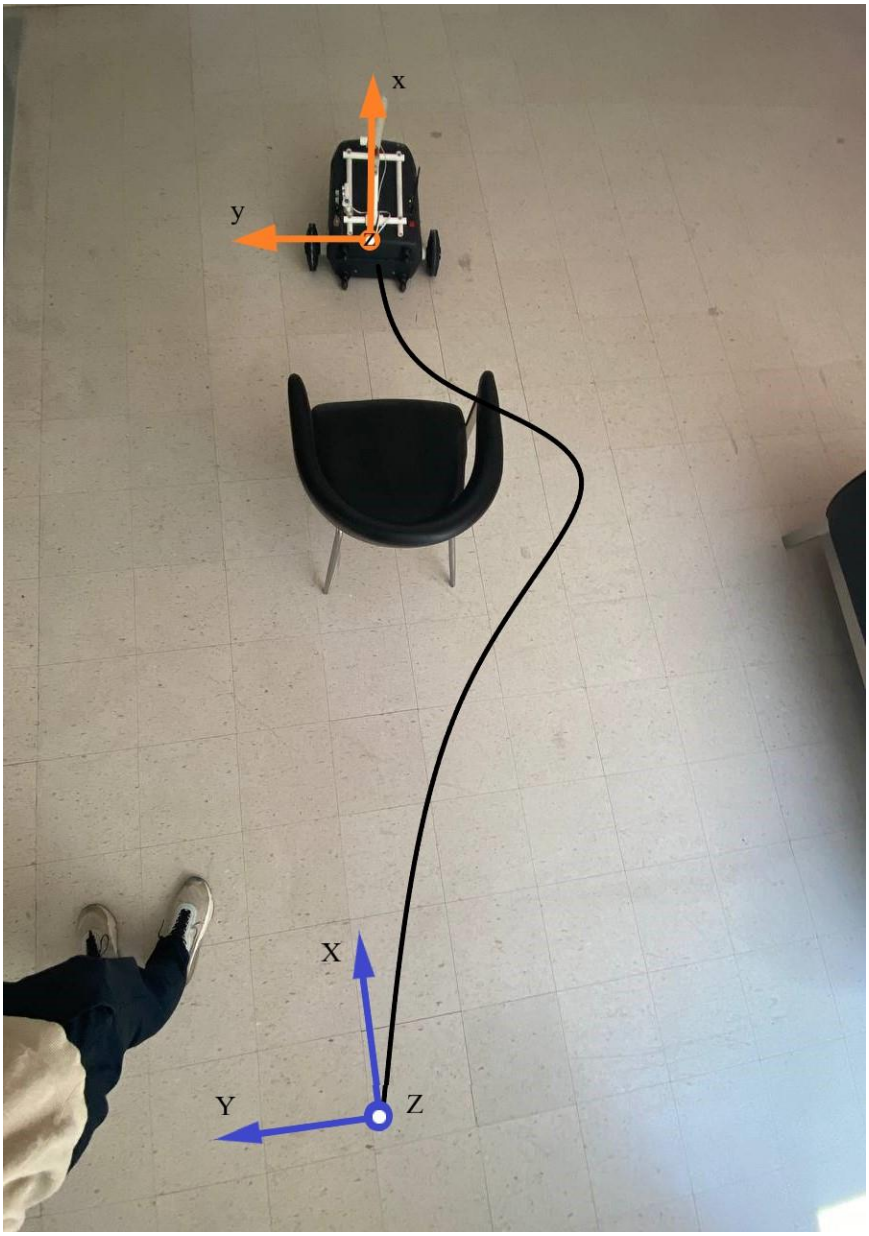

Рисунок 4.9 – Траектория движения робота объезде препятствия

Заключение

Описанная в данной работе управляющая система для мобильного колёсного робота АМУР 105 позволяет решать различные задачи, такие как: поиск объекта, отслеживание передвижения, движение за человеком, движение по сплайну с заданной скоростью и т.д. Данная управляющая система может быть расширена соответствующими захватными устройствами на верхней части робота, а также различными датчиками необходимыми для конкретного вида задач. Преимущество данной ИУС системы является её простота, лёгкость в настройке и перепрограммировании, также возможность к быстрому переносу с одного компьютера на другой. В тоже время данная управляющая система не обеспечивает высокую точность, из-за устаревшей аппаратной части робота.

Список литературы

1. Толстой И.М., Захаров К.С., Кан И.А. “Локализация и навигация мультиагентной робототехнической системы на основе ARUCO-маркеров” // Пятый Всероссийский научно-практический семинар «Беспилотные транспортные средства с элементами искусственного интеллекта» (БТС-ИИ-2019): Труды семинара. – Переславль-Залесский: Российская ассоциация искусственного интеллекта, 2019. – 264 с. С. 39-47.
2. F. Sani and G. Karimian, “Automatic navigation and landing of an indoor AR. drone quadrotor using ArUco marker and inertial sensors,” 2017 International Conference on Computer and Drone Applications (IConDA), Kuching, 2017, pp. 102-107, doi: 10.1109/ICONDA.2017.8270408.
3. Гриняк В.М, Гриняк Т.М., Цыбанов П.А. “Позиционирование внутри помещений с помощью Bluetooth-устройств” // Территория новых возможностей. Вестник Владивостокского государственного университета экономики и сервиса. 2018. Т. 10. № 2. С. 137–147 Система навигации мобильного робота.
4. Карпов, В. Э., М. В. Платонова. "Система навигации мобильного робота." // Информационные средства и технологии: тр. 18-й Междунар. науч.-техн. конф. Москва. 2010.
5. Михайлов, Борис Борисович, Анаид Вартановна Назарова, and Аркадий Семенович Ющенко. "Автономные мобильные роботы-навигация и управление." Известия Южного федерального университета. Технические науки 2. 2016. С. 48-67.
6. MacMillan, N., Allen, R., Marinakis, D., Whitesides, S. “Range-based navigation system for a mobile robot”. // Canadian Conference on Computer and Robot Vision. 2011, pp. 16-23. IEEE.
7. Barber, R., Crespo, J., Gómez, C., Hernámdez, A. C., Galli M. "Mobile robot navigation in indoor environments: Geometric, topological, and semantic navigation." // Applications of Mobile Robots. IntechOpen, 2018.
8. Andersen, Jens Christian. "Mobile robot navigation." // Technical University of Denmark. 2006.
9. Rufus Blas, M., Riisgaard, S., Ravn, O., Andersen, N. A., Blanke, M., Andersen, J. C. “Terrain Classification for Outdoor Autonomous Robots using 2D Laser Scans.” // 2nd International Conference on Informatics in Control, Automation and Robotics. 2005, pp. 347-351.
10. Lim, J., Lee, S., Tewolde, G., Kwon, J. “Indoor localization and navigation for a mobile robot equipped with rotating ultrasonic sensors using a smartphone as the robot's brain.” // IEEE International Conference on Electro/Information Technology. 2015, pp. 621-625.
11. Gini, G. C., Marchi, A. “Indoor robot navigation with single camera vision.” // PRIS, 2, 2002, pp. 67-76.
12. Alves, Paulo, Hugo Costelha, Carlos Neves. "Localization and navigation of a mobile robot in an office-like environment." // IEEE 13th International Conference on Autonomous Robot Systems. 2013, pp. 1-6.
13. Stachniss, C., Burgard, W. “Particle filters for robot navigation.” // Foundations and Trends in Robotics, 3(4). 2014, pp. 211-282.

Оглавление