Министерство науки и высшего образования Российской Федерации
ФГАОУ ВО «Волгоградский государственный университет»
Институт Математики и информационных технологий
Кафедра компьютерных наук и экспериментальной математики

УТВЕРЖДАЮ:
Зав. кафедрой *КНЭМ*
Клячин В.А.
«01» сентября 2021 г.

ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ на УЧЕБНУЮ ПРАКТИКУ, НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ РАБОТА (ПОЛУЧЕНИЕ ПЕРВИЧНЫХ НАВЫКОВ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЫ) на 2021 - 2022 год

Студент	Курбанов Эльдар Ровшанович (ФИО)	МОСм-201 (группа)
Руководитель практики от ВолГУ	Клячин В.А. (ФИО)	зав. кафедрой КНЭМ, проф., д.ф.-м.н. (должность, ученое звание и степень)
Ответственный за организацию практики от кафедры	Клячин В.А. (ФИО)	зав. кафедрой КНЭМ, проф., д.ф.-м.н. (должность, ученое звание и степень)
Место прохождения практики	Лаборатория «Математического и программного обеспечения ЭВМ» кафедры КНЭМ ИМИТ ФГАОУ ВолГУ (наименование учреждения, структурного подразделения)	
Сроки прохождения практики	с «01» сентября 2021 г. по «30» декабря 2021 г.	

1. Содержание и задания практики:

№ п/п	Этапы практики	Содержание работы и задания этапов	Количество часов	Календарные сроки проведения	Форма отчетности
1	Подготовительный этап	Решение организационных вопросов	24	01.09.2021-03.09.20.21	Собеседование

2	Ориентировочный этап	Постановка задачи, выбор методов решения.	200	04.09.2021-14.10.2021	Собеседование, письменный отчёт (часть)
3	Основной этап	Определение проблемы, объекта и предмета исследования, постановка исследовательской задачи; разработка инструментария исследования, использование интерактивных и проектных технологий; сбор и обработка полученных данных с использованием информационных и компьютерных технологий.	400	15.10.2021-27.12.2021	Письменный отчёт (часть).
4	Заключительный этап	Подготовка отчета по практике. Представление научно-исследовательской работы.	24	28.12.2021-30.12.2021	Письменный отчёт (оформление) о результатах НИР; представление НИР

2. Планируемые результаты практики:
студент должен знать: основы программирования и языков программирования, организации баз данных, системного программирования и компьютерного моделирования, соблюдения информационной безопасности; фундаментальные принципы прикладного и системного программирования.
студент должен уметь: использовать основы программирования и языков программирования, организации баз данных, системного программирования и компьютерного моделирования, соблюдения информационной безопасности в профессиональной деятельности; использовать знания в области прикладного и системного программирования в профессиональной деятельности.
студент должен владеть умениями: применения основ программирования и языков программирования, организации баз данных, системного программирования и компьютерного моделирования, соблюдения информационной безопасности при решении конкретных задач; разработки ПО.

Студент	_____ (подпись)	Курбанов Э.Р. (расшифровка подписи)
Руководитель практики от ВолГУ	_____ (подпись)	Клячин В.А. (расшифровка подписи)
Ответственный за организацию практики от кафедры	_____ (подпись)	Клячин В.А. (расшифровка подписи)

Министерство науки и высшего образования Российской Федерации
ФГАОУ ВО «Волгоградский государственный университет»
Институт Математики и информационных технологий
Кафедра компьютерных наук и экспериментальной математики

УТВЕРЖДАЮ:
Зав. кафедрой *КНЭМ*
Клячин В.А.
«01» сентября 2021 г.

ОТЧЕТ О ПРОХОЖДЕНИИ УЧЕБНОЙ ПРАКТИКИ, НАУЧНО-ИССЛЕДОВАТЕЛЬСКАЯ РАБОТА (ПОЛУЧЕНИЕ ПЕРВИЧНЫХ НАВЫКОВ НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЫ)
на 2021 - 2022 учебный год

Студент	Курбанов Эльдар Ровшанович (ФИО)	МОСм-201 (группа)
Руководитель практики от ВолГУ	Клячин В.А. (ФИО)	зав. кафедрой КНЭМ, проф., д.ф.-м.н. (должность, ученое звание и степень)
Ответственный за организацию практики от кафедры	Клячин В.А. (ФИО)	зав. кафедрой КНЭМ, проф., д.ф.-м.н. (должность, ученое звание и степень)
Место прохождения практики	Лаборатория «Математического и программного обеспечения ЭВМ» кафедры КНЭМ ИМИТ ФГАОУ ВолГУ (наименование учреждения, структурного подразделения)	
Сроки прохождения практики	с «01» сентября 2021 г. по «30» декабря 2021 г.	

1. Ход выполнения практики

№ п/п	Этап практики	Дата	Описание выполненной работы	Отметки руководителя о выполнении

1	Подготовительный этап	01.09.2021-03.09.2021	Решение организационных вопросов: установочная конференция, знакомство с задачами и программой практики, требованиями к отчетной документации, инструктаж по технике безопасности.	
2	Ориентировочный этап	04.09.2021-14.10.2021	Постановка задачи, выбор методов решения, сбор и предварительная обработка исходных данных, знакомство с методами работы.	
3	Основной этап	15.10.2021-17.10.2021	Изучение и обобщение состояния проблемы в теории и современной отечественной и зарубежной практике.	
		18.10.2021-20.10.2021	Постановка исследовательской задачи. Введение.	
		21.10.2021-31.10.2021	Разработка инструментария исследования, использование интерактивных и проектных технологий.	
		01.11.2021-15.11.2021	Сбор и обработка полученных данных с использованием ИКТ. Описание анализа полученных данных. Глава 1.	
		16.11.2021-30.11.2021	Изучение выбранной технологии. Применение выбранной технологии к поставленной задаче. Глава 2.	
		01.12.2021-24.12.2021	Составление заданий для тестирования. Заключение и выводы.	
		25.12.2021-27.12.2021	Оформление научно-исследовательской работы.	

4	Заключительный этап	28.12.2021-30.12.2021	Подготовка отчета по практике. Представление научно-исследовательской работы.	

Студент _____ Курбанов Э.Р.
(подпись) (расшифровка подписи)

2. Отзывы руководителей проекта

ОТЗЫВ РУКОВОДИТЕЛЯ ПРАКТИКИ ОТ УНИВЕРСИТЕТА

Зачёт по практике принят с оценкой _____ (по 5-балльной шкале) _____ (по 100-бальной шкале)

Ответственный за организацию практики от кафедры «___» _____ 20___ г. _____ (подпись) Клячин В.А. (расшифровка подписи)

Руководитель практики от ВолГУ «___» _____ 20___ г. _____ (подпись) Клячин В.А. (расшифровка подписи)

Приложения[1]

Вступление

В ходе практики я работал над корректным подсчётом пройденного роботом расстоянием. Это позволит роботу самому оценивать его текущее местоположение на карте, выстраиваемой при помощи лазерного сканера LiDAR. Сам робот представляет собой платформу на двух гусеницах и оснащён двумя электродвигателями, драйвером, лазерным сканером и компьютером, управляющий процессом движения. Он изображён на Рисунке ...

Моей задачей стало исправление некорректного подсчёта числа оборотов колеса производимых на ведущей части гусеницы робота. Для успешного построения карты местности при помощи лазерного сканера, изображённого на Рисунке ... роботу необходимо решать задачу локализации в пространстве. Погрешностей в определении местоположения должно быть как можно меньше, они напрямую будут влиять на выстраиваемую карту местности. Будут возникать смещения или ещё хуже - артефакты[2].

На данном роботе возможно реализовать три способа локализации в пространстве:

1) Анализ смещения облака точек;

2) Подсчёт одометрии;

3) Первые два способа вместе, корректирующие показатели друг друга.

Такие способы локализации, как триангуляция на основе заранее установленных радиомаяков и спутниковая связь Глонасс не рассматриваются ввиду требования полной автономности робота.

Практика

Моей задачей является - "исправление расчета оборотов ведущего колеса гусеничного шасси робота". На основе этих оборотов считается фактически пройденное роботом расстояние после применения команды движения в определённую сторону. Обнаружилось, что получаемые значения оборотов отличались от ожидаемых при высокой загруженности управляющего компьютера. Изначально исправлению подлежала только программная часть робота, однако в ходе работы выяснилось, что природа ошибки кроется в операционной системе робота.

Принцип получения показателей пройденного роботом расстояния следующий:

---

[1]Приложения к отчету о прохождении практики: (приводится материалы, указанные в индивидуальном плане на практику в графе «Форма отчетности», например, научно-исследовательская работа, презентации, конспект занятия и т.д.).

[2]объекты на карте, которых в реальности не существует

- Робот включается и инициализирует среду ROS[3];
- Включается система навигации робота, которая требует лазерный сканер и текущее расстояние, пройденное гусеницами;
- Запускается лазерный сканер и происходит инициализация аппаратного интерфейса GPIO с цифровыми электрическими входами;
- Навигационная система по шине I^2C даёт команду драйверам двигателя на движение;
- Датчики Холла, установленные на двигателях робота подают электрический сигнал 3.3 вольт в момент прохождения колесом одного оборота.
- Аппаратный интерфейс GPIO считывает данный сигнал и суммирует все такие обороты;
- На основе новых пройденных роботов подсчитывается пройденное роботом расстояние.

1 этап

Первоначально я посчитал, что причиной расхождения показателей является подвисание программы на каком-либо из циклов в программном коде и при высокой загруженности мы просто не успеваем исполнить код, отвечающей за чтение цифрового сигнала на интерфейсе GPIO. В таком случае вполне возможно мы могли недосчитаться каких-то оборотов колеса и избавление от таких циклов станет решением проблемы.

Т.к. речь идёт о программном коде робота и мы имеем дело с Robot Operating System, оперирующей с входными данными, как с входящими в неё топиками, которые публикуют другие узлы, я нашёл какой узел отвечает за публикацию и суммирование текущих оборотов двигателя. Искать долго не пришлось, но никаких бесконечных циклов в коде узла и библиотеки Jetson.GPIO, которую он использует найдено не было. Каких-либо мест в коде, где исполнение узла могло бы застревать найдено не было.

Мною была выдвинута идея о том, что такие просчёты со стороны узла напрямую связаны с природой операционной системы Ubuntu, используемой на роботе. Данная ОС не является системой, нацеленной на исполнение команд в режиме реального времени, а это значит, что в момент прохождения ведущим колесом робота датчика Холла мы не можем гарантировать квант времени от операционной системы на исполнение программы нашего узла, а значит не можем и гарантировать подсчёт всех оборотов колеса. Примерная схема такого просчёта представлена на Рисунке ...

Заручившись поддержкой тематических интернет-форумов и своего научного руководителя, я приступил ко второму этапу...

[3]Robot Operating System

2 этап

Выходом из данной ситуации стало бы использование операционной системы реального времени, такой как QNX[4], но это стало не позволительной роскошью для данного робота в следствии отсутствия какой-либо рабочей реализации используемого фреймворка ROS для данной ОС, а также высокая стоимость лицензии.

По названным выше причинам было решено некоторый микроконтроллер, который удовлетворял следующим требованиям:

1) Принимает электрические сигналы в реальном времени без просчётов

2) Способна коммуницировать с Robot Operating System

3) Является компактным и энергоэффективным решением

Под эти требования отлично подошёл микроконтроллер Teensy 4.0 на базе 32 битного ARM процессора NXP MIMXRT1062DVL6A. Схематичное описание и внешний вид микрокомпьютера представлены на Рисунке ...

К нему были подсоединены датчики холла и внешние электропитание 5 вольт. В последствии планируется делегировать на данный микроконтроллер нагрузку, связанную с управлением драйвером электродвигателей робота.

После проверки цепей питания и удостоверившись в корректном прохождении сигналов к микроконтроллеру, я начал реализовывать программную часть.

3 этап

Для реализации программной части необходимо использовать систему разработки Arduino IDE с установленным дополнением TeensyDuino. Это позволяет использовать все библиотеки, доступные для Arduino доступными и для микроконтроллера Teensy 4.0.

Для коммуникации между основным компьютером NVIDIA Jetson Xavier NX и Teensy 4.0 было решено использовать предоставляемый фреймворком ROS инструмент rosserial. Данный инструмент позволяет при помощи Arduino-совместимой библиотеки и подключения по серийному порту наладить полноценную в рамках ROS коммуникацию в режиме реального времени без необходимости вручную описывать взаимодействие между двумя компьютерами.

Идея взаимодействия будет следующая:

1) На основном компьютере запускается ROS, который при помощи rosserial уста-

[4]QNX (произносится «кьюникс», «кью-эн-экс») — POSIX-совместимая операционная система реального времени, предназначенная преимущественно для встраиваемых систем. Считается одной из лучших реализаций концепции микроядерных операционных систем.

навливает соединение с Teensy

2) Микроконтроллер считает количество пришедших электрических сигналов

3) Каждый ROS цикл публикуется количество подсчитанных сигналов

4) Узел на стороне главного компьютера принимает и обрабатывает данные числа для подсчёта местоположения робота

Реализация скетча представлена в Листинге ...

После завершения работы мои проверки не показали расхождений в значении подсчитанных оборотов ведущих колёс робота и я посчитал данную задачу завершённой.

Целью данной работы является создание системы автоматического управления роботом с учётом данных, получаемых от окружающего пространства и прежде всего создание самого тестируемого образца робота и его аппаратной системы управления.

Для достижения поставленной цели необходимо было решить следующие **задачи**:

1) Исследовать предметную область робототехники[5] (аппаратную и программную часть);

2) Изучить существующие известные аналоги (в т.ч. зарубежные) и продумать как сделать робота ещё лучше;

3) Разработать схему управления роботом и соответствующее ПО;

4) Протестировать созданное изделие.

Научная новизна:

1) Впервые в России был сделан робот с одновременным использованием технологии YDLIDAR, движением и распознаванием объектов окружающего пространства на базе платформы NVIDIA Jetson Xavier NX[6];

2) Создана программно-аппаратная база, на основе которой можно сделать робота, выполняющего иной функционал.

Практическая значимость данной работы заключается в том, что была решена задача создания своего собственного алгоритма движения для робота на базе относительно новой и ещё мало изученной платформы Jetson Xavier NX со своим алгоритмом езды и следованием за целевыми объектами.

Методология и методы исследования. При разработке данной системы управления и формирования поведенческой стратегии автономного мобильного робота исполь-

[5]Робототехника не изучалась на протяжении всего курса обучения в университете.

[6]Возможно, это происходит не впервые, но других таких известных случаев не нашлось.

зовались такие методы эмпирического исследования, как наблюдение и эксперимент, а к методам теоретического исследования - анализ и синтез и восхождение от абстрактного к конкретному.

Объем и структура работы. Данная работа состоит из введения, трёх глав и заключения. Полный объём составляет 20 страниц, включая 7 рисунков. Список литературы содержит 20 наименований.

1. Реализация SLAM

1.1. Введение

Одновременная локализация и картографирование (SLAM) — это задача получения карты неизвестной среды с помощью движущегося робота при одновременной локализации робота относительно этой карты. Проблема SLAM касается ситуаций, когда у робота отсутствует датчик глобального позиционирования. Вместо этого он должен полагаться на независимые от внешних устройств датчики для оценки положения робота (например, лазерный сканер, одометрия, инерциальный датчик). Такие датчики со временем накапливают ошибки, что усложняет задачу получения точной карты. В последние годы проблема SLAM привлекла значительное внимание научного сообщества, и появилось множество новых алгоритмов и методов[3].

1.2. Лазерное сканирование при помощи LiDAR

Одновременная локализация и картографирование (SLAM) может обеспечить позиционирование и картирование неизвестных сред в режиме реального времени, а затем реализовать планирование пути и автономную навигацию. В настоящее время датчики восприятия, используемые в мобильных роботах, в основном делятся на две категории: датчики LiDAR и визуальные датчики. Преимущество лазерных сканеров LiDAR по сравнению с визуальными датчиками[7] заключается в том, что они менее подвержены влиянию окружающей среды и напрямую предоставляют данные облаков точек, описывающие геометрическую информацию об окружающей среде. Как правило, облака точек, полученные непосредственно однолинейным лидаром, представляют собой двумерные данные, которых вполне достаточно для решения задачи локализации и картографирования местности[4]. Пример визуализации облака точек можно увидеть на рисунке 1.

[7]такие, как видеокамера

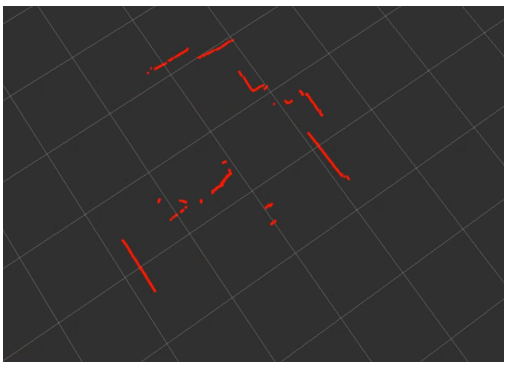

Рис. 1. Облако точек, создаваемое датчиком LiDAR

1.3. Одометрия для SLAM

Одной из наиболее важных проблем в области приложений автономных систем, является самолокализация, то есть самоопределение положения и ориентация движущейся платформы во времени. Для автономной навигации, обхода препятствий и отслеживания объектов платформа должна постоянно сохранять информацию о своем положении и позе. Традиционным методом локализации, который широко используется в автономных платформах, является глобальная система позиционирования (GPS).

В последнее время появилось много исследований по автономным методам одометрии и одновременной локализации и картографированию (SLAM) в качестве популярного примера. Такие методы позволяют рассчитать положение и ориентацию транспортного средства на основе данных, полученных от бортовых датчиков. В отличие от GPS, мы не будем полагаться на внешнюю поддержку спутниковых радиосигналов, которые часто непостоянны и слишком зашумлены для точной локализации. Вместо этого будет использована одометрия, которая использует локальные сенсорные данные для определения положения и ориентации платформы относительно заданной начальной точки[5].

1.4. Фреймворк ROS

Написание программного обеспечения для роботов затруднено, особенно в связи с тем, что масштабы и возможности робототехники продолжают расти. Разные типы роботов могут иметь совершенно разное аппаратное обеспечение, что делает повторное использование кода нетривиальным. Вдобавок к этому сам размер необходимого кода может быть пугающим, поскольку он должен содержать глубокий стек, начиная с программного обеспечения на уровне драйвера и заканчивая восприятием, абстрактными рассуждениями и далее. Поскольку требуемая широта знаний выходит далеко за рамки возможностей любого отдельного исследователя, архитектуры программного обеспечения для робототехники также должны поддерживать крупномасштабные усилия по интеграции программного обеспечения [6].

Фреймворк ROS был создан для облегчения задачи написания ПО для роботов, так как предоставляет готовые решения для коммуникации различных подпрограмм управления роботом, а его большое сообщество в свободном доступе предоставляет различные

пакеты, которые, благодаря, встроенным инструментам и гибкой настройке позволяют построить быстрый прототип проекта.

1.4.1. Терминология ROS

Узлы — это процессы, выполняющие вычисления. ROS разработана как модульная система и обычно состоит из множества узлов. В этом контексте термин «узел» взаимозаменяем с «программным модулем». В данном случае использование термина «узел» связано с визуализацией систем на основе ROS во время выполнения: когда работает много узлов, удобно отображать одноранговые связи в виде графа с процессами в качестве узлов графа и одноранговыми узлами. одноранговые ссылки в виде дуг.

Узлы общаются друг с другом, передавая сообщения. Сообщение представляет собой строго типизированную структуру данных. Поддерживаются стандартные примитивные типы (целочисленные, с плавающей запятой, логические и т. д.), а также массивы примитивных типов и констант. Сообщения могут состоять из других сообщений и массивов других сообщений, вложенных произвольно глубоко.

Узел отправляет сообщение, публикуя его в заданном топике, которое представляет собой просто строку, такую как «odometry» или «map». Узел, который интересуется определенным видом данных, подпишется на соответствующий топик. Для одного топика может быть несколько одновременных публикаторов и подписчиков, и один узел может публиковать и/или подписываться на несколько топиков. Как правило, публикаторы и подписчики не знают о существовании друг друга.

Хотя модель публикации-подписки на основе топиков представляет собой гибкую коммуникационную парадигму, ее «широковещательная» схема маршрутизации не подходит для случаев, когда нам обязательно нужен ответ от подписчика. В ROS существует такая форма связи и она называется сервисы, определяемые строковым именем и парой строго типизированных сообщений: одно для запроса и одно для ответа. Это аналогично веб-сервисам, которые определяются URI и имеют строго типизированные запросы и ответы. В отличие от топиков, только один узел может создать сервису конкретным именем: например, может быть только один сервис под названием «image_classification», точно так же, как может быть только одна веб-служба с любым заданным URI[13].

2. Практическая реализация

В данной секции будет описаны практические аспекты реализации поведенческой стратегии робота и SLAM в частности.

2.1. Аппаратная составляющая

2.1.1. Шасси

Шасси робота[8] представляет из себя самоходную платформу длинной 275мм, шириной 190мм и высотой 95мм. Она состоит из двух пластиковых гусениц, металлической крышки для установки оборудования, а также двух электромоторов, оснащённых энкодерами с номинальным напряжением 9 вольт[7]. Внешний вид шасси робота изображён на Рисунке 2.

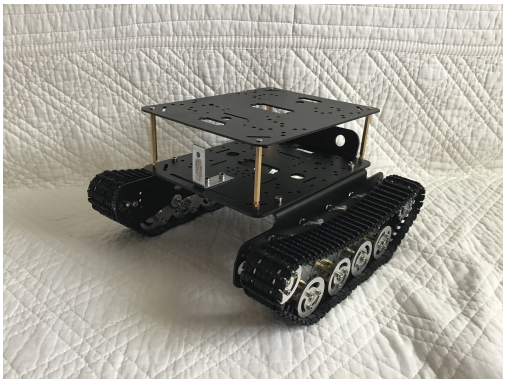

Рис. 2. Внешний вид шасси робота

2.1.2. Вычислительные мощности

Для подсчёта импульсов энкодеров[9] на электродвигателях робота и считывания аналогового значения, выдаваемого звуковым эхолокатором используется Arduino-подобный микрокомпьютер Teensy 4.0 на базе 32 битного ARM процессора NXP MIMXRT1062DVL6A[8]. Схематичное описание и внешний вид микрокомпьютера представлены на Рисунке 3.

Все остальные расчёты и координацию движения робота принимает на себя одноплатный микрокомпьютер Nvidia Jetson Xavier NX на базе шестиядерного 64-разрядного процессора NVIDIA Carmel с архитектурой ARM v8.2 и 8гб оперативной памяти LPDDR4x[9]. Такой высокопроизводительный компьютер нацелен не только на управление роботом и обработку данных лазерного сканирования, но и на цели, описанные в разделе 3. Внешний вид компьютера изображён на Рисунке 4.

2.2. ОС и фреймворк ROS

На компьютер Jetson Xavier NX установлен специальный образ операционной системы Ubuntu 18.04 от компании Nvidia, где по-умолчанию в комплект включён пакет разработчика JetPack SDK, в котором собраны полезные инструменты для оптимизации выполнения нейронных сетей на мобильных устройствах. Единственный дистрибутив

[8]модель TS100 производителя SZdoit
[9]датчики Холла

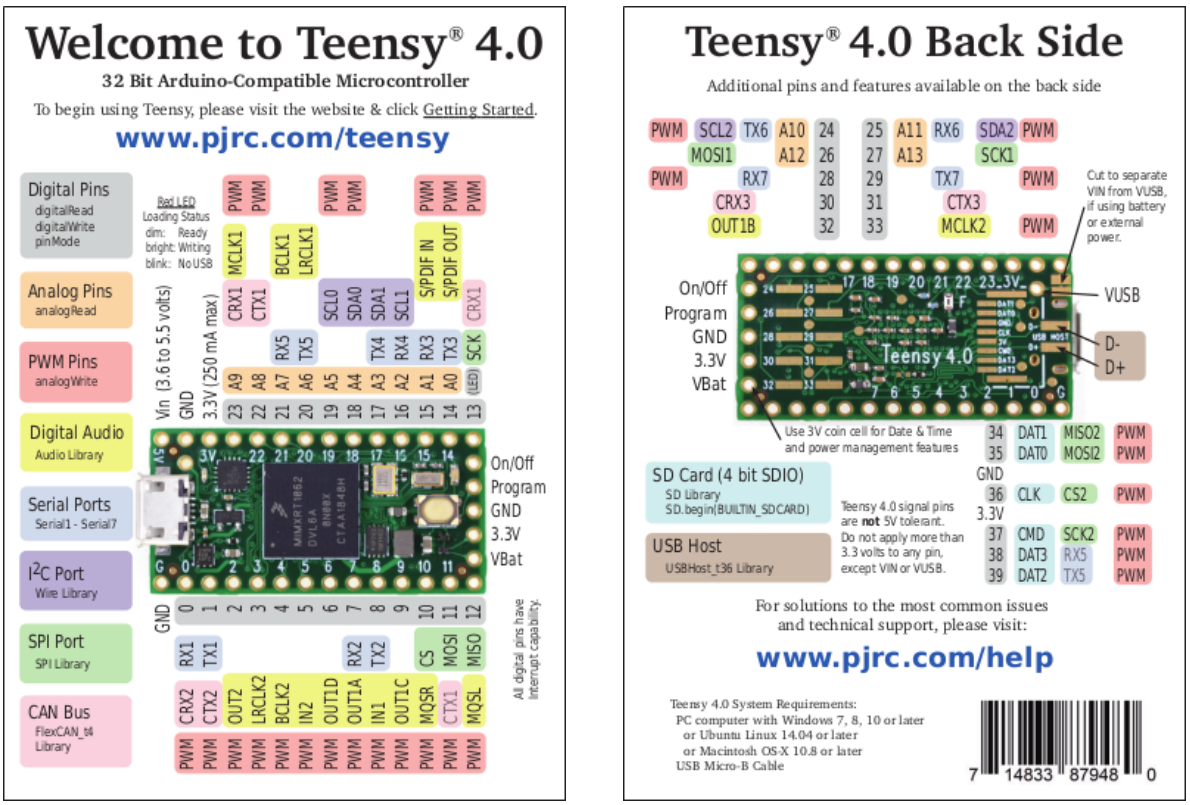

Рис. 3. Описание мирокомпьютера Teensy 4.0

ROS, подходящий под установленную ОС является ROS Melodic, выпущенный в 2018 году.

2.3. Лазерный сканер

Для получения информации об окружающем пространстве используется 360 градусный двумерный лазерный сканер YDLIDAR X4, реализующий технологию LiDAR. Данный лидар способен сканировать наличие препятствий вокруг робота на расстоянии от 12 сантиметров до 10 метров с частотой обновления до 12 герц[10].

2.4. Управление электродвигателями

Для управления двигателями применяется встроенная в компьютер Xavier NX шина управления I2C, которая позволяет отправлять и получать различные команды. К данной шине подключен Grove - I2C Motor Driver V1.3 производителя Seeedstudio. Изображение с описанием драйвера представлено на Рисунке 5.

Данный драйвер может напрямую управлять шаговым двигателем или двигателем постоянного тока. В его сердце находится чип двухканальный драйвер L298N, который может обрабатывать ток до 2 ампер на канал. Для питания двигателя требуется источник питания от 6 до 15 вольт[19].

Шасси робота поставлялось с электродвигателями DT25-370 с номинальным рабо-

Рис. 4. Внешний вид микрокомпьютера NVIDIA Jetson Xavier NX[14]

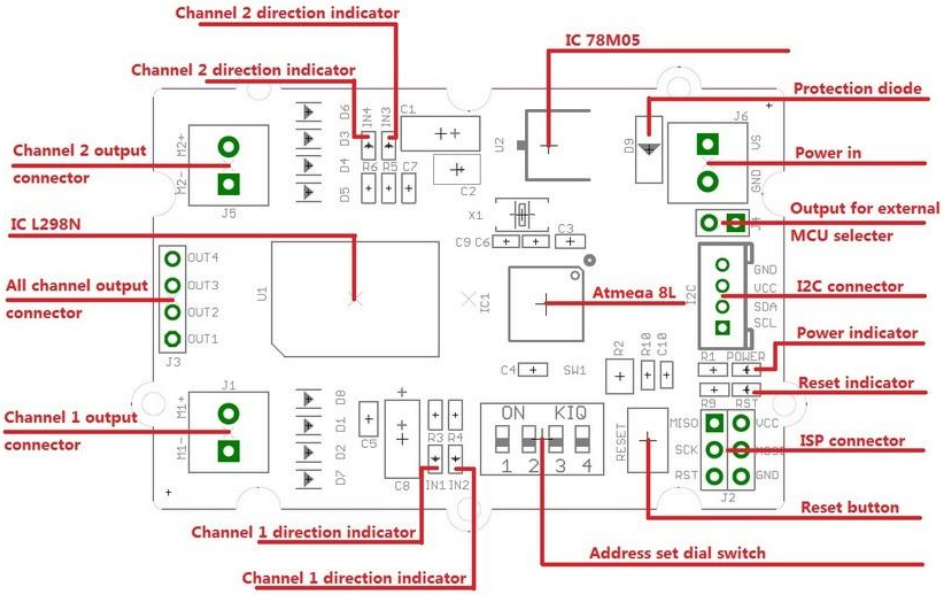

Рис. 5. Описание Grove Motor Driver 1.3[19]

чим напряжением 9 вольт и встроенной одометрией, реализуемой датчиками Холла, подключенными к компьютеру Teensy 4.0[11].

2.5. Программная реализация

2.5.1. Датчики Холла и одометрия

Микрокомпьютер Teensy 4.0, который напрямую подключен к датчикам Холла на электродвигателях робота на основе простого подсчёта импульсов, произошедших за один тик, формирует сообщение следующего формата, приведённого в Листинге 1.

Листинг 1. Формат сообщения diffbot_msgs/Encoders[20]

```
int32[2] ticks
```

Данное сообщение по серийному порту отправляется на основной компьютер Xavier NX, который исходя из текущего времени и подсчитанного количества импульсов с двух двигателей робота формирует сообщение, указанное в Листинге 2, называемое одометрией.

Листинг 2. Формат сообщения nav_msgs/Odometry[15]

```
std_msgs/Header header
string child_frame_id
geometry_msgs/PoseWithCovariance pose
geometry_msgs/TwistWithCovariance twist
```

Благодаря одометрии мы можем локализовать робота на карте, построенной из облака точек при помощи лидара.

2.5.2. Лазерное сканирование и карта

Подключенный к компьютеру YDLidar X4 при помощи серийного порта и драйвера формирует сообщение, в котором содержится облако точек - массив чисел из 720 элементов, в котором есть прямое соответствие текущего положения считывающей головки лидара и расстояние до точки, которую удалось обнаружить. Если точка не была обнаружена, в массив записывается число -1. Формат данного сообщения, представлен в Листинге 3.

Листинг 3. Формат сообщения sensor_msgs/LaserScan[16]

```
std_msgs/Header header
float32 angle_min
float32 angle_max
float32 angle_increment
float32 time_increment
float32 scan_time
float32 range_min
float32 range_max
float32[] ranges
float32[] intensities
```

Благодаря соответствию текущего местоположения робота, которое удаётся вычислить при помощи одометрии и текущего облака точек, мы можем построить карту, в которой отражённые лазером точки станут препятствиями на определённой координате карты, а всё остальное мы объявляем свободным пространством по которому можно безопасно передвигаться. Пример построенной карты приведён на Рисунке 6.

Для построения карты пространства на роботе используется GMapping - специализированная для ROS программа с открытым исходным кодом, которая позволяет строить карту, опираясь на различные сенсоры устройства.

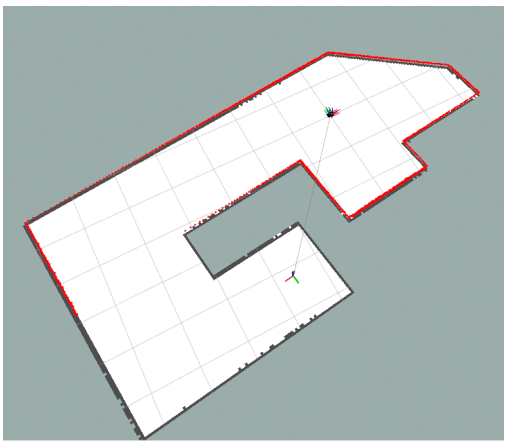

Рис. 6. Пример визуализации карты в программе RVIZ, построенной на основе облака точек, создаваемой LiDAR[17]

2.5.3. Управление двигателями

Для управления двигателями в Robot Operating System реализован специальный формат сообщений, описанный в Листинге . Для того чтобы прослушивать сообщения подобного формата и подачи соответствующей команды по шине I2C используется программный драйвер для Grove Motor Driver, взаимодействующий с системой ROS Hardware Interface, также следящей за тем чтобы команды, поданные на двигатель точно были исполнены.

Листинг 4. Формат сообщения geometry_msgs/Twist Message[18]

```
geometry_msgs/Vector3 linear
geometry_msgs/Vector3 angular
```

Контроль за исполнением осуществляется при помощи подсчёта одометрии робота. Если было выдано задание ехать со скоростью 1 км/ч, то робот не обязательно может поехать с такой скоростью, так как его драйвер откалиброван на определённую мощность двигателей при заданных условиях, но может произойти так, что робот может начать ехать на подъём, что неизбежно повлечёт за собой снижение скорости, так как подающей мощности на электродвигатели больше не будет хватать.

Такие случаи и отслеживаются ROS Hardware Interface: одометрия начнёт отставать от требований некой программы по скорости движения робота и мы начнём давать указание драйверу на то, что нужно увеличить частоту ШИМ модуляции, при помощи которой и управляется скорость движения двигателями и робот начнёт двигаться быстрее. Это принципиально важно чтобы снижать уровень ошибки локализации робота на текущей карте и таким образом повысить точность локализации.

3. Перспективы разработки

В данный момент разрабатывается система управления на обучения с подкреплением(Reinforcement Learning). В качестве входных данных будут использоваться данные

с видеокамеры[10] и облако точек с лазерного сканера YDLidar X4.

Описанные выше входные данные будут передаваться в управляющую нейросеть. Выходными данными нейросети станут управляющие роботом сообщения ROS. Для отработки нейросетевой системы управления на базе обучения с подкреплением будет использоваться симулированная среда Gazebo для ROS.

Заключение

Основные результаты работы заключаются в следующем:

1) Построена программно-аппаратная система, реализующая алгоритм SLAM[11] с использованием базовой библиотеки GMapping;

2) Внедрена система одометрии для подсчёта пройденного роботом расстояния, однако она ещё не откалибрована;

3) Для системы на базе обучения с покреплением подготавливается тестовая среда и создаётся нейросетевая модель управления.

Изображения готового робота можно увидеть на Рисунке 7.

Рис. 7. Готовый робот, выполняющий поиск целевых объектов в доме.

[10] будет определяться наличие или отсутствие целевого объекта в поле зрения видеокамеры

[11] SLAM — Simultaneous Localization And Mapping — Метод одновременной навигации и построения карты[12]

Список литературы

[1] Сергеев, Е. Стратегия новой индустриализации России: автоматизация, роботизация, нанотехнологии. - ЛитРес, 2018. - 200 с. - Текст: непосредственный.

[2] Kurbanov, E. Recognition of Faces, Head Positions, Gender, Age, and Emotions In Real Time Using Deep Convolutional Neural Networks. / CEUR-WS - URL: http://ceur-ws.org/Vol-2500/#paper_4 (дата обращения: 28.12.2021). - Текст: электронный.

[3] Sebastian Thrun and Yufeng Liu and Daphne Koller and A. Ng and Zoubin Ghahramani and Hugh F. Durrant-Whyte. Simultaneous Localization and Mapping with Sparse Extended Information Filters. / The International Journal of Robotics Research, 2004. - с. 693 - 716, т. 23. - Текст: непосредственный.

[4] Chen, Chunxu, Pei, Ling, Xu, Changqing, Danping, Zou, Qi, Yuhui, Zhu, Yifan, Li, Tao. Trajectory Optimization of LiDAR SLAM Based on Local Pose Graph. 10.1007/978-981-13-7751-8_36, 2019 - Текст: непосредственный.

[5] S. A. S. Mohamed, M. Haghbayan, T. Westerlund, J. Heikkonen, H. Tenhunen, and J. Plosila, "A Survey on Odometry for Autonomous Navigation Systems,"IEEE Access, vol. 7, pp. 97466-97486, 2019 - URL:https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8764393 (дата обращения 30.12.2021). Текст: электронный.

[6] Quigley M. et al. ROS: an open-source Robot Operating System //ICRA workshop on open source software. – Т. 3. – №. 3.2. – С. 5, 2009 - URL: http://robotics.stanford.edu/~ang/papers/icraoss09-ROS.pdf (дата обращения 30.12.2021). Текст: электронный.

[7] szdoit Store, Описание товара «Беспроводной металлический радиоуправляемый робот танк шасси амортизирующий автомобильный модуль с системой подвески гусеничная гусеница для Arduino игрушка «сделай сам» / AliExpress - URL: https://aliexpress.ru/item/1005002054452153.html (дата обращения 30.12.2021). - Текст: электронный.

[8] PJRC, Teensy® 4.0 Development Board - URL: https://www.pjrc.com/store/teensy40.html (дата обращения 30.12.2021). Текст: электронный.

[9] Nvidia, Технические спецификации Nvidia Jetson Xavier NX. - URL: https://www.nvidia.com/ru-ru/autonomous-machines/embedded-systems/jetson-xavier-nx/ (дата обращения: 30.12.2021). Текст: электронный.

[10] YDLIDAR, X4 DATA SHEET - URL: https://www.ydlidar.com/Public/upload/files/2021-08-20/YDLIDAR%20X4%20Data%20sheet%20V2.0.pdf (дата обращения 30.12.2021). Текст: электронный.

[11] Do Store, Описание товара DT25-370. - URL: https://aliexpress.ru/item/4000548659629.html (дата обращения: 30.12.2021). Текст: электронный.

[12] noonv, SLAM / RoboCraft - URL: https://robocraft.ru/blog/technology/724.html (дата обращения 30.12.2021). Текст: электронный.

[13] Robinroy Peter, ROS Terminology / Robin Robotics - URL: https://robinrobotic.blogspot.com/2019/06/ros-terminology.html (дата обращения 30.12.2021). Текст: электронный.

[14] NVIDIA, Jetson Xavier NX Developer Kit / NVIDIA Developer - URL: https://developer.nvidia.com/embedded/jetson-xavier-nx-devkit (дата обращения 30.12.2021). Текст: электронный.

[15] Robot Operating System, nav_msgs/Odometry Message - URL: https://docs.ros.org/en/noetic/api/nav_msgs/html/msg/Odometry.html (дата обращения 30.12.2021). Текст: электронный.

[16] Robot Operating System, sensor_msgs/LaserScan Message - URL: https://docs.ros.org/en/noetic/api/sensor_msgs/html/msg/LaserScan.html (дата обращения 30.12.2021). Текст: электронный.

[17] Kiran Palla, 04. Creating Map using Laser Scanner and Gmapping / KiranPalla.com - URL: https://kiranpalla.com/autonomous-navigation-ros-differential-drive-robot-simulation/creating-map-using-laser-scanner-and-gmapping/ (дата обращения 30.12.2021). Текст: электронный.

[18] Robot Operating System, geometry_msgs/Twist Message - URL: https://docs.ros.org/en/lunar/api/geometry_msgs/html/msg/Twist.html (дата обращения 30.12.2021). Текст: электронный.

[19] seeed studio, Grove - I2C Motor Driver V1.3 - URL: https://wiki.seeedstudio.com/Grove-I2C_Motor_Driver_V1.3 (дата обращения 30.12.2021). Текст: электронный.

[20] Franz Pucher, Encoders.msg / GitHub - URL: https://github.com/ros-mobile-robots/diffbot/blob/noetic-devel/diffbot_msgs/msg/Encoders.msg (дата обращения 30.12.2021). Текст: электронный.