СОДЕРЖАНИЕ

ВВЕДЕНИЕ

По мере развития технологий все большее распространение получает беспилотная техника. Эти устройства способны выполнять различные задачи без прямого участия человека. Преимущества беспилотной техники включают снижение риска для жизни и здоровья людей, увеличение точности и эффективности выполнения задач, а также возможность работы в опасных или недоступных для человека местах. Эти устройства и системы могут выполнять различные функции, как в промышленности, так и в других сферах деятельности. Примерами такой техники могут быть беспилотные летательные, подводные и надводные аппараты, роботы, выполняющие сложные задачи на производстве, или автоматизированные машины, выполняющие задачи доставки грузов.

Беспилотная техника часто оснащается различными системами навигации, которые позволяют им перемещаться автономно по окружающей среде. Это может быть реализовано с помощью компьютерного зрения, лидаров, инерциальных измерительных блоков, GPS и других технологий. Какие конкретно средства используются для навигации, зависит от среды, в которой работает беспилотная техника, и других факторов. Для работы с данными, полученными с датчиков, существуют различные методы и алгоритмы, отличающие по сложности реализации. Примером может служить SLAM, использующий лидары или камеры. Датчики и другие системы, используемые в беспилотной технике для навигации, также могут выполнять и другие задачи, например различные исследования окружающей среды.

Одним из важных критериев, по которым различается беспилотная техника, является способ осуществления движения. Например: 1. Сухопутная техника. Она может быть колесной, гусеничной, или иметь другие типы подвижности. 2. Летательная техника. Это могут быть дроны мультироторного типа, летательные аппараты с фиксированным крылом или даже бионические роботы, имитирующие полет животных. 3. Плавающая техника. Включает в себя подводные аппараты, плавучие платформы или роботов, способных перемещаться по поверхности воды. Большой интерес представляют сухопутные роботы, использующие бионические конечности для перемещения. Для подобных систем необходимы алгоритмы, реализующие эффективное движение с использование сложной системы движущихся частей робота.

Темой данного дипломного проекта является разработка программной реализации комплекса сканирования пространства и навигации с использованием роботизированной платформы, в качестве которой выбран робот-паук, и лидаров.

1 ОБЗОР ЛИТЕРАТУРЫ

1.1 Обзор аналогов

В данном подразделе будут рассмотрены несколько существующих реализаций SLAM.

1.1.1 Gmapping

Gmapping (Grid-based FastSLAM Mapping) – это алгоритм картографирования и локализации одновременно, разработанный для роботов с использованием сетки [1]. Он является одним из наиболее популярных и широко используемых алгоритмов SLAM для создания точных карт среды и определения положения робота в реальном времени.

Gmapping основывается на алгоритме FastSLAM, который комбинирует методы фильтрации частиц и построения карты на основе сетки. Он позволяет роботу одновременно определять свое местоположение и строить карту окружающей среды, используя данные из датчиков, таких как лазерный дальномер или сканирующий лазерный датчик.

Основные шаги, выполняемые алгоритмом Gmapping:

1. Инициализация. Алгоритм начинает с некоторой инициализации, устанавливая начальное положение робота и создавая пустую сетку карты.

2. Обработка данных с датчиков. Алгоритм получает данные с датчиков, таких как лазерный дальномер. Эти данные представляют собой измерения расстояния и углов сканирования вокруг робота.

3. Обновление частиц. Gmapping использует метод фильтрации частиц для оценки положения робота. Он создает набор частиц, представляющих возможные положения робота, и обновляет их в соответствии с полученными данными датчиков.

4. Построение карты. Для каждой частицы алгоритм строит локальную карту окружающей среды. Он использует данные сканирования с лазерного дальномера, чтобы определить препятствия и свободные области вокруг робота. Затем эти локальные карты объединяются в глобальную карту с использованием модели на основе сетки. Пример карты, построенной при помощи Gmapping, представлен на рисунке 1.1.

5. Обновление весов частиц. Алгоритм вычисляет веса каждой частицы на основе соответствия между сканированием с лазерного дальномера и картой. Частицы, которые лучше соответствуют данным, получают более высокие веса, тогда как менее соответствующие частицы получают более низкие веса.

6. Перевыборка частиц. Частицы с более высокими весами имеют большую вероятность быть выбранными для перевыборки. При перевыборке генерируются новые наборы частиц, основанные на весах предыдущих частиц. Это позволяет сосредоточиться на наиболее вероятных положениях робота.

7. Обновление оценки положения робота. Используя взвешенные частицы, алгоритм вычисляет оценку положения робота и карту окружающей среды. Эта оценка обновляется с каждым новым набором данных с датчиков.

Алгоритм Gmapping имеет несколько преимуществ:

1. Работа в реальном времени. Gmapping способен работать в реальном времени, обновляя карту и оценку положения робота по мере получения новых данных с датчиков.

2. Точность картографирования. Gmapping создает точные карты окружающей среды, используя модель на основе сетки. Он может обнаруживать препятствия и строить подробные и согласованные карты, которые могут быть использованы для планирования пути и навигации.

3. Устойчивость к шуму и неопределенности. Gmapping использует метод фильтрации частиц, который позволяет учитывать шумные данные с датчиков и уменьшать влияние неопределенности на оценку положения робота.

4. Масштабируемость. Gmapping может быть применен к различным типам роботов и средам. Он может работать как на малых, так и на больших площадях, а также адаптироваться к изменениям в окружающей среде.

5. Открытое программное обеспечение. Gmapping является частью пакета программного обеспечения ROS (Robot Operating System) и доступен в открытом исходном коде. Это позволяет разработчикам и исследователям легко использовать и настраивать алгоритм для своих специфических потребностей.

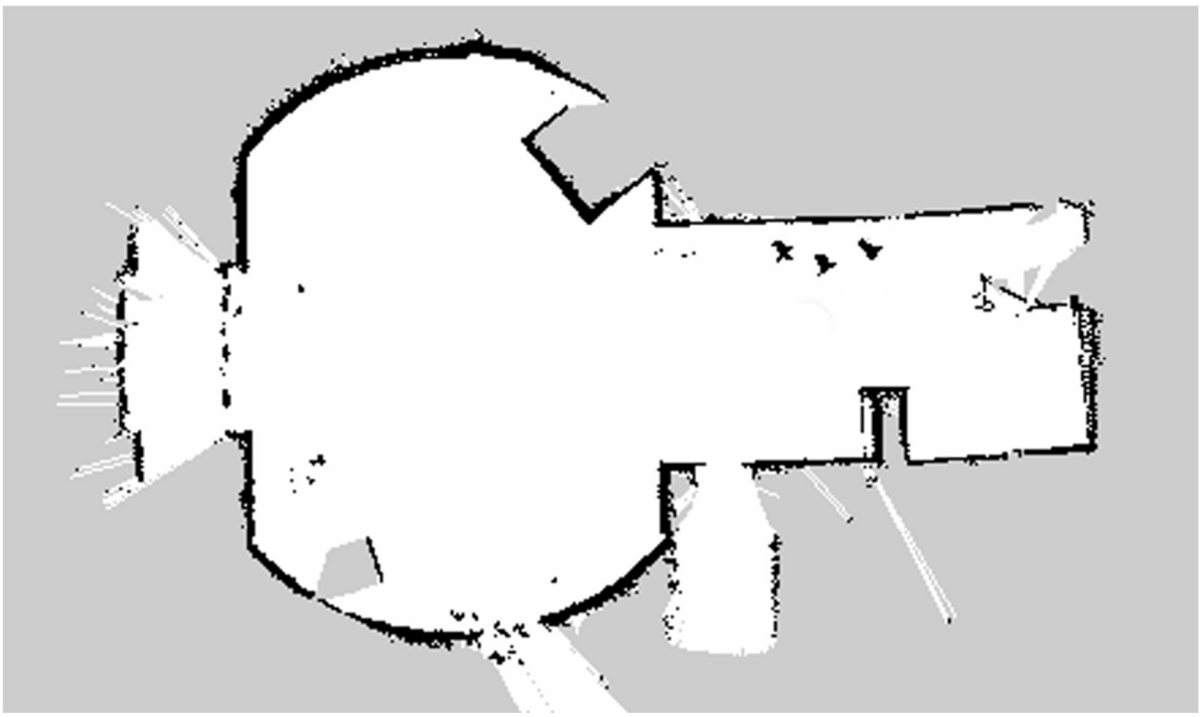

Рисунок 1.1 – Пример карты

Далее представлены некоторые ограничения, имеющиеся у Gmapping:

1. Требования к вычислительным ресурсам. Алгоритм Gmapping требует значительных вычислительных ресурсов для работы в реальном времени, особенно при обработке больших объемов данных с датчиков или при работе на сложных средах.

2. Чувствительность к движению. Gmapping может иметь трудности с точным картографированием и локализацией в случае быстрого движения робота или при наличии динамических объектов в окружающей среде. Это связано с ограничениями обработки данных с датчиков и возможностями модели на основе сетки.

3. Требования к калибровке датчиков. Для достижения наилучших результатов с Gmapping необходимо правильно настроить и калибровать датчики, особенно лазерный дальномер. Неправильная калибровка может привести к неточностям в картах и оценке положения.

В целом, Gmapping является мощным алгоритмом картографирования и локализации SLAM, который обеспечивает точные карты окружающей среды и оценку положения робота в реальном времени. Он широко используется в робототехнике и автономных системах для выполнения задач навигации, планирования пути и взаимодействия с окружающей средой.

1.1.2 Google Cartographer

Google Cartographer – это библиотека и инструмент для создания 2D и 3D карт среды и выполнения одновременной локализации. Разработанная командой Google, Cartographer предоставляет мощные инструменты для робототехники и автономных систем, позволяя им строить детальные и точные карты окружающей среды и определять свое местоположение в реальном времени.

Вот основные компоненты и функции, предоставляемые Google Cartographer:

1. Картирование. Cartographer использует данные с датчиков, таких как лазерные дальномеры или камеры, для построения карты окружающей среды. Он использует методы на основе сетки для представления карты и может работать как в 2D, так и в 3D пространствах. Cartographer позволяет создавать детальные и точные карты, обнаруживать препятствия и другие объекты в окружающей среде.

2. Локализация. Кроме картографирования, Cartographer также выполняет одновременную локализацию, определяя местоположение робота в реальном времени. Он использует данные с датчиков и сопоставляет их с картой окружающей среды для определения положения робота с высокой точностью. Cartographer поддерживает как локализацию на основе признаков, так и локализацию на основе сканирования.

3. Интеграция с датчиками. Cartographer может работать с различными типами датчиков, включая лазерные дальномеры, камеры и инерциальные измерительные блоки (IMU). Он предоставляет гибкие возможности

интеграции с различными моделями и производителями датчиков, позволяя получать высококачественные данные для картографирования и локализации.

4. Поддержка различных платформ. Cartographer разработан для работы на различных платформах, включая настольные компьютеры, мобильные роботы и автономные автомобили. Он имеет модульную структуру, что позволяет легко настраивать и настраивать его для конкретных потребностей и аппаратных сред.

5. Работа в реальном времени. Cartographer способен работать в режиме реального времени, обновляя карты и локализацию по мере получения новых данных с датчиков. Это делает его подходящим для задач навигации в реальном времени и планирования пути.

6. Синхронизация между роботами. Cartographer поддерживает многоагентную систему, что означает, что несколько роботов могут работать совместно для построения согласованных карт. Роботы могут обмениваться данными о картах и положении, что позволяет им эффективно координировать свои действия.

7. Автоматическое выравнивание карт. Одной из особенностей Cartographer является его способность автоматически выравнивать карты разных фрагментов одной общей карты. Это позволяет роботам создавать единое и последовательное представление окружающей среды, даже если они перемещаются по разным областям.

8. Гибкая настройка и расширение. Cartographer предоставляет гибкую архитектуру, которая позволяет настраивать систему и добавлять собственные модули и алгоритмы. Разработчики могут адаптировать Cartographer под свои конкретные потребности и интегрировать его с другими компонентами своей робототехнической системы.

9. Инструменты визуализации. Cartographer предоставляет инструменты визуализации, которые позволяют визуально представить текущую карту и оценку положения робота. Он поддерживает различные форматы визуализации, включая 2D и 3D визуализацию, что упрощает анализ и отладку результатов SLAM.

10. Открытое программное обеспечение. Cartographer является проектом с открытым исходным кодом, доступным в рамках инициативы открытого ПО Google. Это позволяет разработчикам и исследователям свободно использовать, модифицировать и распространять библиотеку, а также вносить свои вклады в ее развитие.

Основная особенность Google Cartographer заключается в том, что он состоит из локального SLAM и глобального SLAM. Локальный SLAM создает отдельные небольшие части карты. Затем, глобальный SLAM алгоритм ищет совпадения между частями карт, а также между данными сенсоров и всеми картами, созданными при помощи локального SLAM в параллельных процессах. В результате, глобальный SLAM формирует единую карту окружающего пространства [2]. Схема работы алгоритма Google Cartographer представлена рисунке 1.2.

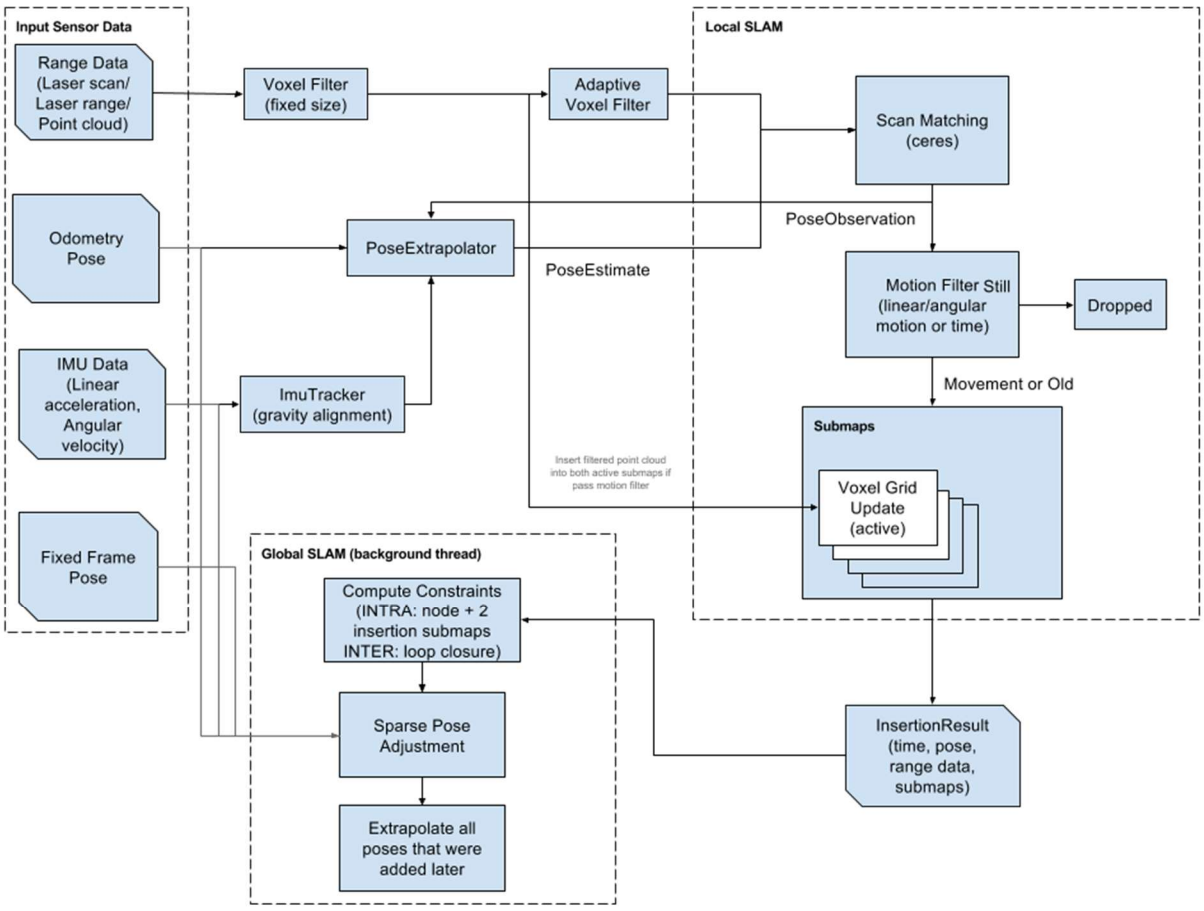

Рисунок 1.2 – Схема работы алгоритма Google Cartographer

Google Cartographer представляет собой мощный инструмент для робототехники и автономных систем, предоставляющий возможности построения детальных карт окружающей среды и определения местоположения в реальном времени. Его гибкость, масштабируемость и поддержка различных платформ делают его привлекательным выбором для разработчиков, которые стремятся реализовать SLAM в своих проектах.

1.1.3 RTAB-Map

Rtabmap (Real-Time Appearance-Based Mapping) – это библиотека и инструмент для одновременной локализации и построения карты в режиме реального времени. Разработанная и поддерживаемая командой IntRoLab (Laboratory of Intensive Robotics) в Монреальском университете, Rtabmap предлагает мощные возможности по картографированию и навигации для робототехники и автономных систем. Используемый алгоритм основан на поиске и сопоставлении соответствий визуальных данных сенсоров с использованием механизма памяти, где хранится база данных визуальных образов в соответствии с данными о местоположении робота [3]. Пример базы данных изображения представлен на рисунке 1.3.

Вот основные компоненты и функции, предоставляемые Rtabmap:

1. Картирование. Rtabmap использует данные с различных датчиков, таких как лазерные дальномеры, камеры и инерциальные измерительные блоки (IMU), для построения карты окружающей среды. Он использует методы на основе признаков для представления карты и может работать как в 2D, так и в 3D пространствах. Rtabmap позволяет создавать плотные карты с высокой детализацией, обнаруживать препятствия, сохранять информацию о среде и определять местоположение робота.

2. Локализация. Rtabmap выполняет одновременную локализацию, определяя местоположение робота в реальном времени. Он использует данные с датчиков, такие как видеопотоки и сканирование окружающей среды, и сопоставляет их с предыдущими данными для оценки положения. Rtabmap поддерживает как локализацию на основе признаков, так и локализацию на основе геометрии.

3. Обнаружение и сопоставление признаков. Rtabmap имеет встроенные алгоритмы для обнаружения и сопоставления признаков на основе изображений. Он может использовать различные дескрипторы, такие как SIFT, SURF или ORB, для идентификации ключевых точек и создания описания сцены. Это позволяет Rtabmap работать с различными типами датчиков и обрабатывать данные с камер и других источников.

4. Граф оптимизации. Rtabmap использует граф оптимизации для улучшения оценки положения и формирования карты. Он строит граф, где узлы представляют собой местоположения робота, а ребра - сопоставления и измерения между ними. Затем Rtabmap выполняет оптимизацию графа, чтобы улучшить точность оценки положения и форму карты.

5. Поддержка различных датчиков и платформ. Rtabmap разработан для работы с различными типами датчиков и платформ, включая мобильные роботы, автономные автомобили и дроны. Он имеет модульную архитектуру, что позволяет легко интегрировать новые датчики и настраивать его для конкретных потребностей.

6. Обнаружение и учет петель. Rtabmap способен обнаруживать петли в окружающей среде и эффективно их учитывать при построении карты. Это позволяет обеспечить более точную локализацию и улучшить качество карты путем устранения ошибок, связанных с повторным посещением областей.

7. Гибридный подход к SLAM. Rtabmap сочетает в себе особенности визуального и геометрического SLAM. Он использует информацию изображений и геометрических данных с датчиков для определения положения робота и построения карты. Это позволяет достичь лучшей точности и надежности в различных сценариях.

8. Встроенная система распознавания объектов. Rtabmap имеет встроенную систему распознавания объектов, позволяющую роботу определять и отслеживать объекты в окружающей среде. Это полезно для задач навигации и взаимодействия с реальными объектами.

9. Визуализация и анализ результатов. Rtabmap предоставляет инструменты визуализации, которые позволяют визуально представить

текущую карту и оценку положения робота. Он также предлагает средства анализа качества карты, такие как графики ошибок локализации и картографии, а также статистику по качеству сопоставления признаков.

10. Расширяемость и открытость. Rtabmap предоставляет API и набор инструментов для расширения его функциональности и интеграции с другими системами. Он является проектом с открытым исходным кодом, что позволяет разработчикам и исследователям свободно использовать и модифицировать его, а также вносить свои вклады в его развитие.

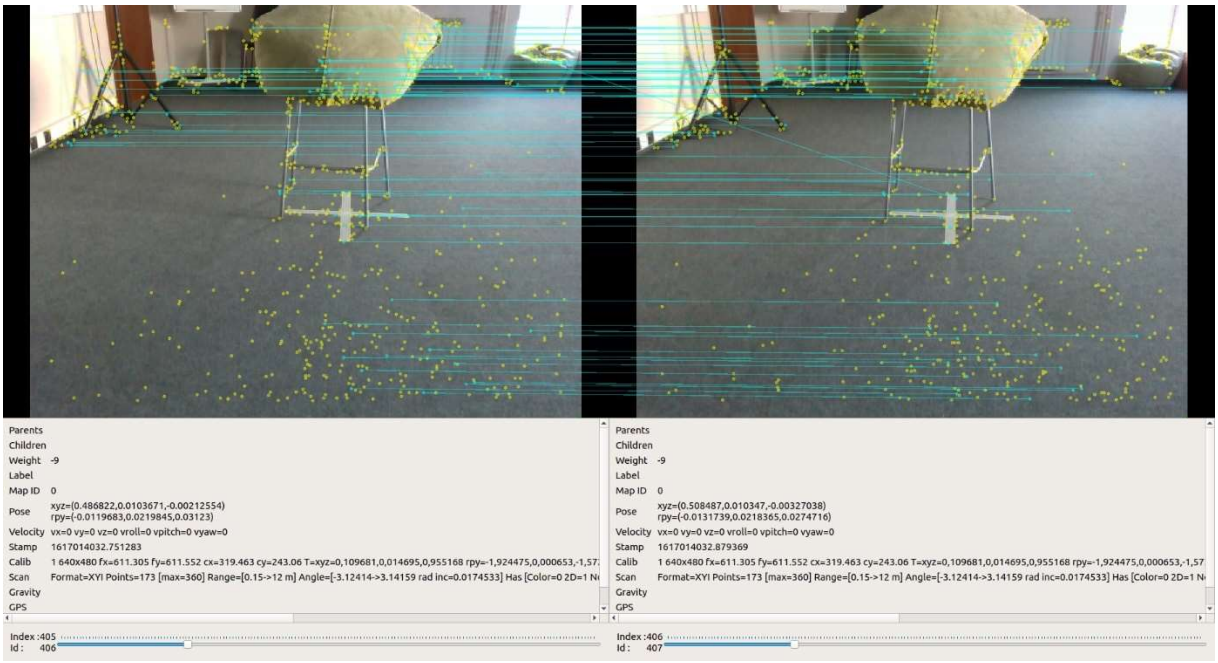

Рисунок 1.3 – База данных изображения Rtabmap

Rtabmap представляет собой мощный инструмент для робототехники и автономных систем, предоставляющий возможности построения детальных карт окружающей среды и определения местоположения в режиме реального времени. С его помощью можно реализовать различные задачи, связанные с картографированием и навигацией, включая автономную навигацию, планирование пути и взаимодействие с окружающей средой.

1.2 Обзор технологий и методов

В этом разделе будут рассмотрены и описаны технологии и методы, которые будут применяться при разработке дипломного проекта.

1.2.1 SLAM

SLAM, или одновременное локализация и построение карты (Simultaneous Localization and Mapping), является одним из ключевых методов в области робототехники и компьютерного зрения. Он позволяет роботу

одновременно определять свое местоположение в неизвестной среде и строить карту этой среды.

Принцип работы SLAM основывается на использовании датчиков и алгоритмов для сбора информации о среде и последующего анализа этой информации. Обычно в SLAM используются два основных типа датчиков: датчики измерения расстояния (например, лазерные сканеры или стереокамеры) и датчики измерения ориентации (например, инерциальные измерители).

С математической точки зрения, SLAM пытается оценить карту и весь путь, пройденный роботом. Таким образом, поза робота рассчитывается только в конце траектории, проделанной роботом [4]. Графическая модель SLAM подхода представлена на рисунке 1.4.

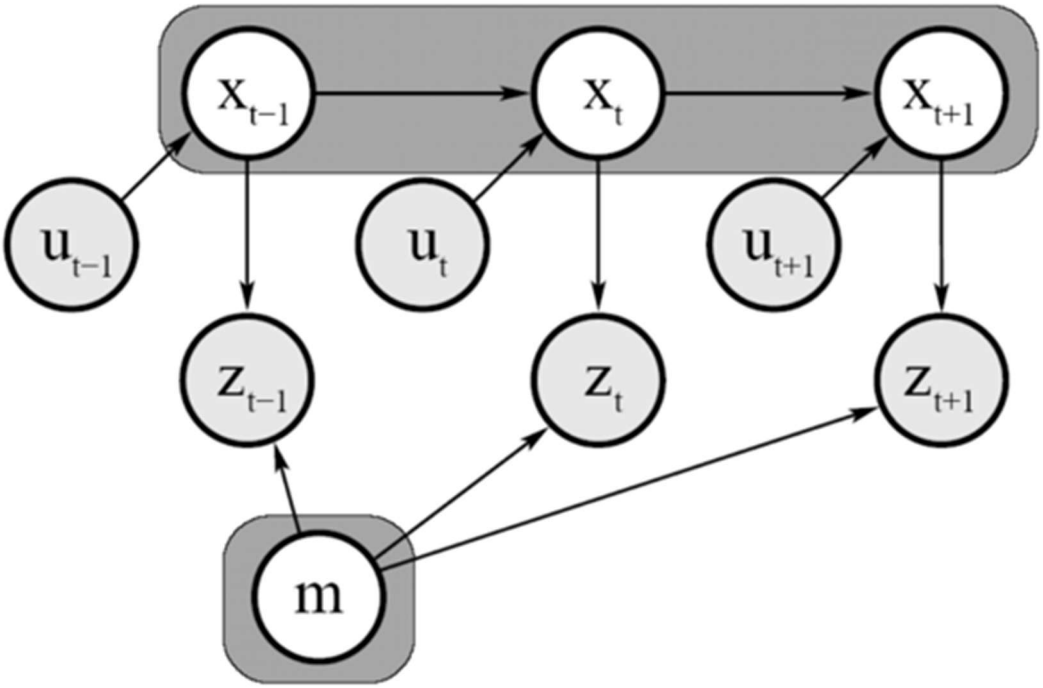

Рисунок 1.4 – Графическая модель SLAM подхода

Процесс SLAM обычно состоит из следующих шагов:

1. Захват данных. Робот собирает данные из своих датчиков, таких как лазерные сканеры, камеры и инерциальные измерители. Лазерные сканеры измеряют расстояния до окружающих объектов, а камеры могут использоваться для извлечения признаков и ориентации. Инерциальные измерители измеряют ускорение и угловую скорость робота.

2. Оценка движения. Используя данные с инерциальных измерителей, робот оценивает свое движение и изменение местоположения с течением времени. Это может включать оценку скорости, ускорения и изменения угла поворота робота.

3. Извлечение признаков. С помощью данных с лазерных сканеров или камер робот извлекает признаки из окружающей среды. Признаками могут быть контуры объектов, особые точки или текстуры. Они используются для сопоставления и определения положения робота относительно них.

4. Ассоциация данных. Следующий шаг состоит в ассоциации данных, то есть соотнесении данных с датчиков с предыдущими измерениями и признаками на карте. Это позволяет определить, какие измерения соответствуют одним и тем же объектам или признакам.

5. Обновление карты. После ассоциации данных робот обновляет карту окружающей среды, добавляя новые признаки и объекты. Карта может быть представлена в виде сетки, графа или другой структуры данных, которая описывает пространственное расположение объектов.

6. Обновление местоположения. С использованием ассоциированных данных и обновленной карты робот обновляет свое текущее местоположение в среде. Этот шаг позволяет роботу уточнить свое местоположение на основе новых данных.

7. Повторение. Весь процесс повторяется снова, начиная с захвата данных. Робот непрерывно собирает данные, обновляет карту и определяет свое местоположение в реальном времени.

SLAM является активной областью исследований, и существует много различных алгоритмов и подходов, позволяющих реализовать этот метод. Некоторые из наиболее распространенных алгоритмов SLAM включают в себя:

1. EKF-SLAM (Extended Kalman Filter SLAM). Этот алгоритм использует расширенный фильтр Калмана для оценки состояния робота и построения карты. Он предполагает линейность моделей движения робота и измерений признаков.

2. FastSLAM. Этот алгоритм использует алгоритм частицы для оценки состояния робота и построения карты. Он разбивает задачу SLAM на набор независимых задач локализации и построения карты для каждого признака на карте.

3. GraphSLAM. Этот алгоритм представляет карту и траекторию робота в виде графа и использует методы оптимизации графов для оценки состояния робота и построения карты.

4. ORB-SLAM. Этот алгоритм комбинирует методы извлечения признаков ORB (Oriented FAST and Rotated BRIEF) с алгоритмом RANSAC (Random Sample Consensus) для решения задачи SLAM в режиме реального времени.

1.2.2 Инверсная кинематика

Инверсная кинематика – это обратная задача к прямой кинематике, которая позволяет определить значения углов, положения и скорости звеньев манипуляционной системы на основе известного положения ее рабочего

органа в пространстве. В данном контексте мы рассмотрим инверсную кинематику роботов.

Инверсная кинематика имеет важное значение в робототехнике, поскольку позволяет управлять движением робота, задавая желаемое положение его рабочего органа. Вместо определения значений углов каждого звена вручную, инверсная кинематика автоматически рассчитывает эти значения с учетом геометрии и ограничений конкретной манипуляционной системы.

Для понимания работы инверсной кинематики с точки зрения геометрии, предположим, что у нас есть манипуляционная система с несколькими звеньями, объединенными в цепь. Каждое звено представляет собой сегмент, связанный с соседними звеньями при помощи сочленений, таких как вращательные или поступательные сочленения.

Главная задача заключается в определении значений углов или длин каждого звена, необходимых для достижения заданного положения рабочего органа (например, конца робота) в трехмерном пространстве.

Процесс решения инверсной кинематики может быть довольно сложным и зависит от конкретной геометрии манипулятора. В общем случае, для решения задачи инверсной кинематики требуется выполнение следующих шагов:

1. Определение координат и ориентации конца робота. Это заданные значения положения и ориентации, которые мы хотим достичь с помощью робота.

2. Анализ геометрии манипулятора. Необходимо изучить структуру и геометрию манипуляционной системы, включая длины звеньев, типы сочленений и ограничения на перемещение каждого звена.

3. Расчет углов или длин звеньев. На основе известных значений положения и ориентации конца робота, а также геометрии манипулятора, нужно вычислить углы или длины звеньев, обеспечивающие требуемое положение.

4. Проверка решения. После расчета значений углов или длин звеньев необходимо проверить, насколько близко полученное решение приближается к желаемому положению и ориентации конца робота. Если требуемая точность достигнута, можно передать вычисленные значения в соответствующие актуаторы робота для выполнения заданного движения.

Важно отметить, что инверсная кинематика может иметь несколько решений или быть неразрешимой в некоторых случаях, особенно когда манипуляционная система имеет ограничения или находится в сингулярных точках. Также могут возникать проблемы совмещения требуемого положения конца робота с физическими ограничениями системы, такими как препятствия или ограничения на движение звеньев.

В реальных системах решение инверсной кинематики обычно выполняется с использованием численных методов, таких как метод Ньютона-Рафсона или методы оптимизации. Эти методы позволяют находить

численное решение для требуемого положения конца робота, учитывая геометрию и ограничения манипулятора.

1.3.3 Пакетный менеджер Cargo

Cargo – это сборочный и управляющий пакетами инструмент в языке программирования Rust. Он является официальным инструментом для управления зависимостями и сборки проектов на Rust. Cargo обеспечивает простой и эффективный способ создания, сборки и управления проектами [5].

Cargo позволяет легко управлять зависимостями проекта. Он использует файл `Cargo.toml` для указания зависимостей и их версий. Cargo может автоматически загружать и устанавливать зависимости из центрального репозитория пакетов Cargo. Это упрощает добавление сторонних библиотек в проект и обновление зависимостей.

Cargo обеспечивает простой способ сборки проекта. Он автоматически обнаруживает и собирает все файлы исходного кода в проекте. Cargo предоставляет различные команды сборки, такие как `cargo build`, `cargo run` и `cargo test`, которые позволяют собирать, запускать и тестировать проект соответственно. Он автоматически управляет компиляцией и линковкой зависимостей проекта.

Cargo предлагает простой способ создания и управления проектами на Rust. Команда `cargo new` создает новый проект со структурой каталогов, включающей файл `Cargo.toml` и каталог `src` для исходного кода. Cargo также обеспечивает интеграцию с системой контроля версий Git.

Cargo позволяет упаковывать и распространять проекты на Rust в виде пакетов. С помощью команды `cargo package` можно создать архив проекта, содержащий все необходимые файлы для его сборки.

Cargo предлагает встроенную поддержку для модульного и интеграционного тестирования в проектах Rust. С помощью команды `cargo test` можно запустить все тесты в проекте или только определенные тесты.

Cargo является мощным и удобным инструментом для разработки проектов на Rust. Он упрощает управление зависимостями, сборку проекта и распространение пакетов. Богатый функционал и простота использования делают Cargo неотъемлемой частью экосистемы Rust.

2 СТРУКТУРНОЕ ПРОЕКТИРОВАНИЕ

Текст раздела

2.1 Подраздел

Текст раздела

2.2 Подраздел

Текст раздела

2.3 Подраздел

Текст раздела

2.4 Подраздел

Текст раздела

2.5 Подраздел

Текст раздела

3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

3.1 Подраздел

Текст раздела

3.2 Подраздел

Текст раздела

3.3 Подраздел

Текст раздела

3.4 Подраздел

Текст раздела

3.5 Подраздел

Текст раздела

3.6 Подраздел

Текст раздела

3.7 Подраздел

Текст раздела

3.8 Подраздел

Текст раздела

4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

Текст раздела

5 ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

Текст раздела

5.1 Подраздел

Текст раздела

5.2 Подраздел

Текст раздела

6 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Текст раздела

6.1 Подраздел

Текст раздела

6.2 Подраздел

Текст раздела

6.3 Подраздел

Текст раздела

7 ЭКОНОМИЧЕСКАЯ ЧАСТЬ

7.1 Подраздел

Текст раздела

7.2 Подраздел

Текст раздела

7.3 Подраздел

Текст раздела

7.4 Подраздел

Текст раздела

7.5 Подраздел

Текст раздела

ЗАКЛЮЧЕНИЕ

Текст раздела

СПИСОК ИСПОЛЬЗУЕМЫХ ИСТОЧНИКОВ

[1] Gmapping [Электронный ресурс] – Режим доступа: http://wiki.ros.org/gmapping/. – Дата доступа: 09.04.2023.

[2] Google Cartographer ROS [Электронный ресурс] – Режим доступа: https://google-cartographer-ros.readthedocs.io/en/latest/. – Дата доступа: 09.04.2023.

[3] RTAB-Map, Real-Time Appearance-Based Mapping [Электронный ресурс] – Режим доступа: https://introlab.github.io/rtabmap/. – Дата доступа: 09.04.2023.

[4] Autonomous Navigation with Simultaneous Localization and Mapping in/outdoor / E. Pedrosa [и др.]. – Aveiro, Portugal, 2020.

[5] The Cargo Book [Электронный ресурс] – Режим доступа: https://doc.rust-lang.org/cargo/. – Дата доступа: 10.04.2023.

ПРИЛОЖЕНИЕ А
(*обязательное*)
Название приложения

ПРИЛОЖЕНИЕ Б
(*обязательное*)
Название приложения

ПРИЛОЖЕНИЕ В
(*обязательное*)
Спецификация

ПРИЛОЖЕНИЕ Г
(*обязательное*)
Ведомость документов