Федеральное государственное бюджетное образовательное учреждение высшего образования «Ярославский государственный университет им. П.Г. Демидова»

На правах рукописи

Антипов Владимир Алексеевич

Повышение точности позиционирования камеры в системе прикладного телевидения с использованием расширенного фильтра Калмана

Специальность 2.2.13
Радиотехника, в том числе системы и устройства телевидения

ДИССЕРТАЦИЯ
на соискание ученой степени кандидата технических наук

Научный руководитель:
доктор технических наук, доцент
Приоров Андрей Леонидович

Ярославль – 2021

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Актуальность темы исследования. На сегодняшний день системы технического зрения широко востребованы в разных областях науки, техники, медицины и промышленности. Техническое зрение может применяться для создания различных радиотехнических систем, примерами которых могут выступать системы прикладного телевидения (СПТ). Системы прикладного телевидения предназначены для передачи и приема изображений в промышленности, науке, образовании, медицине, военном деле, сфере безопасности и других областях.

Многие задачи, которые решают СПТ требуют получения информации о месторасположения того или иного объекта и локализации камеры. С помощью такой информации можно решить такие задачи как обнаружение и оценка параметров движущихся объектов, навигации в пространстве, автоматизирования процессов видеофиксации требуемых событий, исследования труднодоступных или опасных для человека мест. Сложность задачи определения месторасположения объектов и локализации камеры характеризуется множеством самых разнообразных проблем, начиная с неравномерного освещения и заканчивая проблемами различных радиопомех и переотражений от внешних источников сигналов. Также сложность решения этой задачи связана с затруднениями в отдельных случаях реализации каналов связи. Из-за указанных проблем возникает необходимость в создании СПТ, способных решать задачи определения месторасположения объектов в плохо обусловленных и зашумленных средах.

Перспективным, быстро развивающимся и востребованным направлением в СПТ являются одновременная локализация и картографирование (SLAM). Значительный вклад в разработку методов и алгоритмов решения задачи SLAM внесли работы П. Смита, П. Чизмена, В.И. Кобера, Е.Н. Сосновой, С.Л. Зенкевича, В.С. Лемпицкого, А. Девисона, Р. Парра, А. Элизара, Д.В. Солдатовой и др.

Многие соответствующие приложения в робототехнике и техническом зрении требуют быстрого построения трехмерных карт окружающей среды и оценки положения камеры относительно этой карты. Мобильной платформе с камерой нужно, например, знать свое местоположение в окружающей среде для того, чтобы безопасно перемещаться. Эта проблема является традиционной и сложной, поскольку для локализации камеры в окружающей среде требуется трехмерная модель, а для ее построения, в свою очередь, требуется знать месторасположение камеры. Следовательно, и траектория движения камеры, и трехмерная модель должны оцениваться одновременно [1].

Задача SLAM – синхронное определение местоположения и составление карты. Она связана с построением карты неизвестного пространства мобильным устройством во время навигации по строящейся карте. Проблема одновременной локализации и построения карты является фундаментальной задачей при создании автономных мобильных платформ, и алгоритм SLAM в этом случае является базовым методом для многих навигационных систем.

Алгоритмы SLAM позволяют решать следующие задачи:

1) Создание беспилотных транспортных средств и систем помощи водителю. Решение этой задачи обеспечивает повышение мобильности населения, эффективности грузопассажирских перевозок, повышение безопасности дорожного движения, снижение экологической нагрузки на окружающую среду, повышение комфортности для водителей и пользователей транспорта.
2) Поиск и спасение людей. С помощью алгоритмов SLAM беспилотный летательный аппарат (БЛА, БПЛА) способен быстро находить человека и доставлять жизненный важный груз, а также детектировать опасность. Тем самым это дает возможность вовремя обнаружить, оказать первую медицинскую помощь и эвакуировать в безопасное место терпящих бедствие людей.

3) Детектирование и сопровождение объектов. В настоящее время идет широкое внедрение систем видеоаналитики в различные сферы жизни людей. С помощью алгоритмов SLAM можно способствовать решению таких задач, как наблюдение за объектами интереса или посторонними, обнаружение несанкционированного проникновения на охраняемые зоны или появления человека в опасных зонах. Также алгоритмы SLAM могут стать важной составляющей любой интеллектуальной транспортной системы.
4) Исследование труднодоступных мест. Алгоритмы SLAM могут применяться при оперативном получении точной информации о радиационной, химической и биологической обстановке местности для дальнейшего планирования эффективных мер по ликвидации опасности, а также для построения детальных карт при поиске различных ресурсов.
5) Исследование космоса. При исследовании поверхности планет необходимы аппараты, оснащенные автономной навигационной системой, позволяющей им передвигаться по сложному рельефу местности. Такая система должна работать несколько лет, строить точную карту и давать оценку локализации относительно нее, а также должна быть способна рационально использовать энергетические ресурсы аппарата.

Кроме того, существует множество практических применений SLAM как средства получения информации в областях пространства, недоступных людям или опасных для них. На сегодняшний день работы в данной области направлены на повышение точности локализации, улучшение эффективности вычислений и решение проблем объединения данных. Таким образом, задача одновременной локализации и построения карты является крайне важной и актуальной.

Степень разработанности темы. Задача локализации может быть решена с помощью других систем, таких как системы глобального

позиционирования (GPS, ГЛОНАСС и др.) или системы локального позиционирования (локальные беспроводные сети WLAN, Bluetooth-маяки, RFID метки и т.д.), но эти системы либо имеют высокую погрешность, либо неудовлетворительно работают в плохо обусловленных и зашумленных средах.

Основными преимуществами способа локализации с помощью алгоритма SLAM перед другими способами являются:

1) Отсутствие необходимости реализации различных сетей.
2) Отсутствие необходимости знания заранее заданных координат опорных точек и предварительной разметки местности.
3) Способность работать в плохо обусловленных и зашумленных средах.
4) Низкие экономические затраты: для реализации СПТ с алгоритмом SLAM в простейшем случае необходимы всего пара датчиков и одноплатный компьютер.
5) Построение детализированной карты, с помощью которой автономные мобильные платформы могут взаимодействовать с окружающей средой.

Также SLAM имеет следующий недостаток:

1) Ошибка локализации имеет свойство накапливаться. Однако есть, например, алгоритмы замыкания, которые частично решают эту проблему.

На сегодняшний день существует множество различных подходов к задаче SLAM с использованием разных типов сенсоров: лидаров, сонаров, камер. Исследования разработок в данной области показывает, что на сегодняшний день основными методами являются методы на основе расширенного фильтра Калмана, фильтра частиц и графов.

Решения, основанные на расширенном фильтре Калмана, дают более высокую точность построения карты по сравнению с фильтром частиц при одинаковом количестве ориентиров, но имеют квадратичную

вычислительную сложность и высокую чувствительность к соответствию ориентиров.

Алгоритмы SLAM, основанные на фильтре частиц, имеют линейную вычислительную сложность и возможность убирать ошибочные соответствия ориентиров, но ошибки каждой частицы привносят вклад в общую карту и накапливаются с течением времени.

Алгоритмы, использующие графы, представляют карту в виде узлов (месторасположения камеры в определенные моменты времени) и ребер (оценку перемещения между двумя позициями камеры). Это дает линейную зависимость требуемой памяти от количества объектов на карте и делает обновление графа постоянным по времени. Однако данные алгоритмы имеют высокую вычислительную сложность внутри узла, что связано с решением задачи поиска ближайших точек. Кроме того, окончательная оптимизация графа также может потребовать больших вычислительных затрат.

В каждой области, где требуется определить месторасположения камеры, существуют свои требования к точности нахождения месторасположения. Например, для дорожной навигации во многих случаях достаточно точности в 5 метров. С другой стороны, для создания автономных транспортных средств необходима высокая точность определения местоположения, с целью предотвращения аварий и соблюдения правил дорожного движения.

Алгоритмы SLAM являются рекуррентными алгоритмами, у которых текущая оценка месторасположения зависит от предыдущей оценки. Если немного повысить точность оценки месторасположения на каждой итерации, то значительно повысится результирующая оценка. При плохой точности алгоритмы невозможно применять для построения карт крупномасштабных сцен и решать задачи навигации по ним. Таким образом, существует научная проблема, связанная с необходимостью повышения точности построения карты и локализации относительно нее, а также уменьшения вычислительной сложности используемых алгоритмов.

Основной целью работы является повышение точности оценки месторасположения камеры в системе прикладного телевидения.

Для достижения указанной цели в диссертационной работе решаются следующие **задачи:**

1. Разработка алгоритма одновременной локализации и построения карты с использованием камеры и лазерной сканирующей системы.
2. Разработка алгоритма визуальной одометрии с использованием сферической модели камеры.
3. Определение матриц измерения и якобиана для ориентира типа «особая точка» с использованием сферической модели камеры.
4. Улучшение алгоритмов детектирования пространственных ориентиров по данным лазерной сканирующей системы.
5. Исследование алгоритмов одновременной локализации и построения карты на основе расширенного фильтра Калмана.
6. Сравнение визуальных алгоритмов одновременной локализации и построения карты.

Объектом исследования является измерительная система прикладного телевидения в составе автономной мобильной платформы.

Предметом исследования являются алгоритмы цифровой обработки изображений и дальнометрических данных, позволяющие детектировать ориентиры и локализовать камеру в пространстве найденных ориентиров.

Методы исследования. При решении поставленных задач использовались современные методы прикладного телевидения, цифровой обработки сигналов и изображений, технического зрения, статистической радиотехники, математической статистики и линейной алгебры. Для практической реализации алгоритмов применялись современные методы программирования на языке высокого уровня Python 2.7 с использованием библиотеки компьютерного зрения OpenCV 3.0, библиотеки матричных вычислений NumPy, библиотеки для выполнения научных и инженерных расчетов SciPy, фреймворка для программирования робототехнических

систем Robot Operating System Kinetic (ROS Kinetic), трехмерного динамического симулятора Gazebo с возможностью точного и эффективного моделирования робототехнических систем и инструмента визуализации Rviz. Моделирование и экспериментальные исследования предлагаемых алгоритмов выполнялись на мобильной платформе с установленной камерой и лидаром.

Научная новизна

Получены следующие новые научные результаты:

1. Разработан обобщенный алгоритм EKF-SLAM, отличающийся возможностью интеграции нескольких типов датчиков и использования нескольких оценок состояния системы, полученных разным путем, для уточнения параметров системы в расширенном фильтре Калмана. Алгоритм отличается тем, что позволяет рассматривать сложные динамические системы (например, группу мобильных платформ), формировать и обрабатывать локальные карты по заданным критериям.
2. Разработан алгоритм одновременной локализации камеры и построения карты и его модификации с использованием камеры и лазерной сканирующей системы, отличающиеся интеграцией двух типов датчиков в расширенном фильтре Калмана, применением сферической модели камеры и контурного анализа. Алгоритм позволяет строить карту, состоящую из вектора состояния и ковариационной матрицы, двумерную карту проходимости, а также трехмерную карту окружающей среды.
3. Улучшены алгоритмы детектирования пространственных ориентиров по данным лазерной сканирующей системы. Алгоритмы отличаются дополнительным преобразованием данных лидара в комплекснозначный сигнал и его делением на сегменты.
4. Разработан алгоритм построения локальных карт с равномерным использованием ориентиров. Алгоритм отличается от остальных алгоритмов построения локальных карт тем, что учитывает

корреляцию между всеми ориентирами с сохранением линейной вычислительной сложности.

5. Разработан алгоритм реконструкции трехмерной сцены. Алгоритм отличается тем, что в процессе реконструкции используются панорамные изображения, полученные с камеры с объективом типа «рыбий глаз», а также учитывается сферическая модель камеры.

Практическая значимость

1. Предложен обобщенный алгоритм EKF-SLAM, позволяющий рассматривать сложные динамические системы, использовать несколько оценок состояния системы для повышения точности, а также формировать и обрабатывать локальные карты. Это позволяет рационально контролировать точность построения карты, определения месторасположения и вычислительные ресурсы. Таким же способом можно обобщить и другие алгоритмы на базе расширенного фильтра Калмана, применяемые в других областях науки, техники и промышленности.
2. Предложен и реализован на языке высокого уровня алгоритм одновременной локализации и построения карты на основе цифровой обработки телевизионных изображений и данных лазерной сканирующей системы с использованием системы прикладного телевидения. Данный алгоритм позволяет строить траекторию движения мобильной платформы, карту проходимости и трехмерную карту окружающей среды. Ошибка определения месторасположения мобильной платформы разработанного алгоритма составляет 0.88 ± 0.73 м по метрике RPE и 0.09 ± 0.08 м по метрике ATE.
3. Получены результаты исследования применимости различных подходов и особенностей реализации задачи одновременной локализации и построения карты в системе прикладного телевидения.
4. Предложен способ представления данных лидара в комплекснозначный сигнал (контур). Такой способ представления дает возможность

применять корреляционные и спектральные методы обработки сигналов к данным лидара.

Достоверность полученных научных результатов обеспечивается корректным использованием математического аппарата и экспериментальными данными, подтверждающими теоретические выкладки и результаты схожих российских и зарубежных исследований.

Апробация работы. Результаты работы докладывались и обсуждались на следующих научно-технических конференциях:

- Девятая научно-техническая конференция «Техническое зрение в системах управления 2019», Москва, 2019;
- Международная конференция «Системы синхронизации, формирования и обработки сигналов в инфокоммуникациях «СИНХРОИНФО», Ярославль, 2019;
- Международная конференция «Системы синхронизации, формирования и обработки сигналов в инфокоммуникациях «СИНХРОИНФО», Минск, 2018;
- Двадцать третья международная конференция открытой инновационной ассоциации FRUCT, Болонья, 2018;
- Двадцать четвертая международная конференция открытой инновационной ассоциации FRUCT, Москва, 2019.
- Ярославские региональные конференции молодых ученых и аспирантов.

Публикации. По теме диссертации опубликовано 13 научных работ, из них 3 статьи в изданиях из перечня ВАК, 5 статей, индексируемых в SCOPUS, получено Свидетельство о государственной регистрации программы для ЭВМ.

Основные научные положения и результаты, выносимые на защиту

1. Алгоритм одновременной локализации и построения карты на основе цифровой обработки телевизионных изображений и данных лазерной

сканирующей системы для применения в системах прикладного телевидения, обеспечивающий повышение точности позиционирования до 0,88±0,73 м по метрике *RPE* и 0,09±0,08 м по метрике *ATE*, за счет добавления дополнительной оценки состояния системы и пренебрежения избыточной и ошибочной корреляцией между мобильной платформой и ориентирами.

2. Улучшения алгоритмов детектирования пространственных ориентиров по данным лазерной сканирующей системы, обеспечивающие вероятность детектирования ориентира до 90%, за счет применения предварительной обработки данных.
3. Обобщенный алгоритм EKF-SLAM, позволяющий рассматривать сложные динамические системы и применять алгоритмы построения и обработки локальных карт, за счет построения вектора состояния и ковариационной матрицы из соответствующих матриц выбранных объектов системы.

Личный вклад автора. Выносимые на защиту положения предложены и реализованы автором самостоятельно в ходе выполнения научно-исследовательских работ на кафедре инфокоммуникаций и радиофизики Ярославского государственного университета им. П.Г. Демидова.

Структура и объем работы. Диссертация состоит из введения, трех глав, заключения, списка использованных источников и приложения.

Первая глава диссертации посвящена описанию основной проблемы задачи одновременной локализации и построения карты. Приведено описание, основные особенности и отличия алгоритмов SLAM на базе расширенного фильтра Калмана и на базе фильтров частиц. Сравниваются различные типы сенсоров, такие как лидар, сонар и камера. Приводятся принципы работы сенсоров, их основные характеристики, преимущества и недостатки.

Рассматривается такое направление, как визуальный SLAM, в котором в качестве датчика используется камера. Приводится описание популярных алгоритмов: ORB SLAM, LSD-SLAM и RGB-D SLAM.

Вторая глава посвящена разработке алгоритма одновременной локализации и построения карты с использованием камеры с объективом типа «рыбий глаз» и лазерной сканирующей системы, а также его модификации.

На основе проведенного анализа разработан обобщенный алгоритм EKF-SLAM. На основе обобщенного алгоритма разработан EKF-SLAM с использованием камеры с объективом типа «рыбий глаз» и лидара, который состоит из 6 основных этапов:

1) Получение и синхронизация данных;

2) Обнаружение ориентиров в пространстве;

3) Поиск соответствий ориентиров;

4) Оценка месторасположения камеры и ориентиров;

5) Уточнение месторасположения камеры и ориентиров;

6) Построение трехмерной карты.

Все этапы алгоритма подробно описаны. Каждый из этих этапов можно реализовать с помощью ряда различных алгоритмов. Приведено несколько таких алгоритмов для некоторых этапов.

Рассмотрено два типа ориентиров, применяемых в алгоритме: ориентир типа «угол» и ориентир типа «особая точка». Для каждого ориентира подсчитаны матрицы измерения и их якобианы. Показаны две основные проблемы обнаружения ориентиров по данным лидара: недостающие данные и окклюзия. В связи с этим предложены детекторы с использованием контурного анализа.

Предложен алгоритм поиска соответствия ориентиров с помощью камеры. Поскольку многие способы восприятия, такие как зрение, предоставляют богатую информацию о форме, цвете и текстуре, все они

могут быть использованы для поиска соответствия между двумя наборами ориентиров.

Реализованы два алгоритма построения и обработки локальных карт: EKF SLAM с адаптивным диапазоном наблюдения и EKF-SLAM «разделяй и властвуй». На основе проведенного анализа этих алгоритмов разработан алгоритм EKF-SLAM с равномерным использованием ориентиров.

Третья глава посвящена экспериментальным исследованиям предполагаемых алгоритмов. Приводятся результаты исследования по определению оптимальных параметров детекторов обнаружения ориентиров, таких как размер окна, порог.

Рассмотрены две метрики для сравнения работы алгоритмов SLAM: ошибка RPE (относительная ошибка месторасположения) и ATE (абсолютная ошибка траектории).

В заключении изложены основные итоги диссертационного исследования, сформулированы научные и практические результаты диссертации.

Глава 1. АЛГОРИТМЫ ОДНОВРЕМЕННОЙ ЛОКАЛИЗАЦИИ И ПОСТРОЕНИЯ КАРТЫ

1.1. История развития задачи одновременной локализации и картографирования

Задача одновременной локализации и построения карты (SLAM) – метод получения карты неизвестной окружающей среды и траектории движения мобильной платформы. Этот метод был первоначально предложен для решения задач в области робототехники, а именно для создания автономно передвигающихся роботов. Затем метод SLAM стали широко применять в таких приложениях как 3-D моделирование на основе компьютерного зрения, визуализация на основе дополненной реальности (AR) и автомобили с автономным управлением.

Проблема одновременной локализации и построения карты впервые была освещена на конференции по робототехнике и автоматизации «Robotics and Automation Conference», состоявшейся в Сан-Франциско в 1986 году. Ряд исследователей рассматривали применение вероятностных методов в задачах картографирования и локализации. Результатом конференции стало признание того, что задача картографирования является фундаментальной проблемой, которую необходимо было решить. Работы П. Смита, П. Чизмена и Х. Даррент-Уайта показали, что существует корреляция между оценками месторасположения ориентиров на карте и ее рост при увеличении числа наблюдений. Затем в работах Р. Смита, М. Селфа и П. Чизмена было показано, что при движении мобильной платформы, проводящей относительные наблюдения за ориентирами, через неизвестное окружение, все оценки о месторасположении ориентиров обязательно коррелируют друг с другом из-за общей ошибки в предполагаемом местонахождении мобильной платформы. Отсюда следовало, что полное решение задачи одновременной локализации и построения карты с использованием расширенного фильтра Калмана требовало обновлять вектор состояния, состоящее из месторасположений ориентиров и мобильной платформы,

после каждого наблюдения ориентира. Из-за чего возникла квадратичная вычислительная сложность алгоритма, зависящая от числа ориентиров.

Затем исследования были сосредоточены на решениях по уменьшению вычислительной сложности, которые предполагали минимизировать или исключать корреляцию между ориентирами. В результате было выявлено, что решение задачи одновременной локализации и построения карты при уменьшении корреляции ухудшалось, и при росте корреляции, наоборот, улучшалось [2].

Дальнейшие исследования выявили оптимальное количество ориентиров с точки зрения вычислительной сложности и точности построения карты алгоритмом EKF-SLAM в диапазоне от 40 до 50 ориентиров.

После этих исследование появились алгоритмы SLAM, основанные на применении теоремы Байеса и фильтра частиц. Эти алгоритмы используют особенность задачи SLAM, которую вывел Мерфи. Особенность задачи SLAM заключается в том, что оценка месторасположений ориентиров не зависят от траектории движения мобильной платформы, т.е. корреляция между ориентирами возникает только из-за наличия ошибки оценки месторасположения мобильной платформы. Исследование алгоритмов на базе фильтра частиц показало, что вычислительная сложность у них меньше, чем у исследованных ранее алгоритмов SLAM на базе расширенного фильтра Калмана, но плохо применимы для построения крупных карт из-за падения точности.

В настоящее время перспективным направлением является разработка алгоритмов SLAM с использованием камер. Потому что с помощью камер можно получить намного больше информации, чем с помощью других датчиков. Поскольку алгоритмы SLAM используют визуальную информацию, полученную с камер, их называют визуальными алгоритмами SLAM (vSLAM). Алгоритмы vSLAM широко используются в таких областях,

как техническое зрение, робототехника, системы с дополненной реальностью [3].

Исследование алгоритмов vSLAM, показали, что у них существуют такие недостатки, как высокая чувствительность к резким вращениям и движениям, в которых не наблюдается глубина сцены. Также алгоритмы чувствительны к освещению и однородным текстурам.

1.2. Проблема SLAM

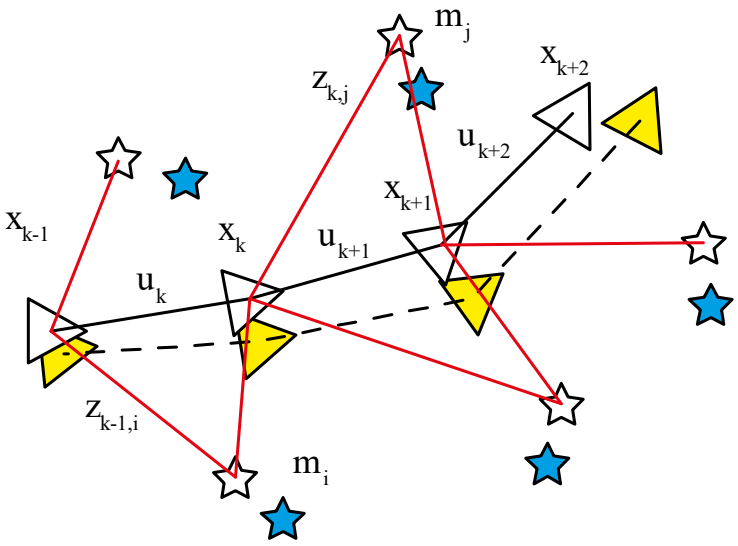


Рис. 1.1. Процесс работы алгоритма SLAM: движение мобильной платформы от положения x_{k-1} под управляющими воздействиями u_k, u_{k+1}, u_{k+2}, наблюдения ориентира m_i обозначены как $z_{k-1,i}$, $z_{k,i}$, $z_{k+1,i}$

Алгоритм SLAM – алгоритм, с помощью которого мобильная платформа может построить карту окружающей среды и одновременно использовать эту карту для определения своего местоположения. В SLAM как траектория движения платформы, так и местоположение всех ориентиров оцениваются в режиме реального времени, без необходимости априорного знания их исходных местоположений.

На основе ряда измерений, полученных с датчика в дискретные моменты времени k требуется вычислить оценку месторасположения

мобильной платформы (рис. 1.1). В этот момент времени отклоняются следующие величины:

x_k – вектор состояния, описывающий месторасположение и ориентацию мобильной платформы;

u_k – вектор управляющего воздействия, примененное в момент времени $k-1$, чтобы перевести робота в состояние x_k в момент времени k;

m_i – вектор, описывающий местоположение i-го ориентира, месторасположение которого предполагается неизменным во времени;

z_{ik} – наблюдение, снятое с мобильной платформы месторасположения i-го ориентира в момент времени k.

$X_{0:k}$ – множество месторасположения мобильной платформы.

$U_{0:k}$ – множество векторов управляющего воздействия.

$Z_{0:k}^{i}$– множество наблюдений i-го ориентира.

m – множество всех ориентиров.

Для решения проблемы SLAM необходимо знать модель движения мобильной платформы и модель наблюдения. Месторасположение робота изменяется по вероятностному закону:

$$P(x_k|u_k, x_{k-1}). \tag{1.1}$$

Модель движения позволяет сделать оценку месторасположения мобильной платформы при условии, что известна предыдущая оценка месторасположения и текущее управляющее воздействие.

Окружающая среда имеет неподвижные ориентиры. Каждый i ориентир имеет координаты m_i. Стоит отметить, что мобильная платформа может воспринимать более одного ориентира одновременно, но из-за математического удобства будем предполагать, что мобильная платформа наблюдает только один ориентир. Это предположение ничему не противоречит, поскольку на каждом временном шаге ориентиры обрабатываются последовательно.

Измерения датчика тоже подчиняется вероятностному закону, часто называемому моделью измерения:

$$P(z_k|x_k, m). \quad (1.2)$$

Вероятностная модель измерения позволяет оценить вероятность конкретного измерения датчика при условии предполагаемого положения мобильной платформы на карте x_k и ориентиров m.

В общем случае проблема SLAM — это проблема определения месторасположения всех ориентиров m и месторасположения мобильной платформы x_k по наблюдениям z_{ik} и управляющему воздействию u_k.

Задача одновременной локализации и построения карты требует вычисления распределения $P(x_k, m\,|Z_{0:k}, U_{0:k}, x_0)$ для всех моментов времени k. Это распределение описывает совокупную оценку местоположений ориентиров и мобильной платформы в момент времени k, учитывая записанные наблюдения и входные данные управляющего воздействия. В общем виде решение задачи SLAM является рекурсивным.

При всем многообразии применяемых методов можно выделить некоторую общую схему работы алгоритма (рис. 1.2):

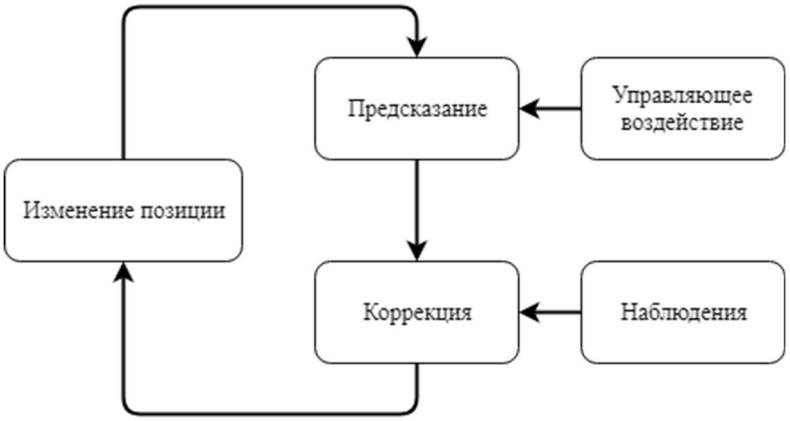


Рис. 1.2. Структурная схема основных этапов процесса одновременной локализации и построения карты

Алгоритм SLAM обычно состоит из двух этапов:

1) Прогнозирование:

$$P(x_k, m\,|Z_{0:k}, U_{0:k}, x_0) = \int P(x_k\,|x_{k-1}, u_k) \times P(x_{k-1}, m\,|Z_{0:k-1}, U_{0:k-1}, x_0)dx_{k-1}. \quad (1.5)$$

2) Коррекция:

$$P(x_k, m \mid Z_{0:k}, U_{0:k}, x_0) = \frac{P(z_k \mid x_k, m) P(x_k, m \mid Z_{0:k-1}, U_{0:k}, x_0)}{P(z_k \mid Z_{0:k-1}, U_{0:k})}. \quad (1.6)$$

Уравнения обеспечивают рекурсивную процедуру для вычисления $P(x_k, m \mid Z_{0:k}, U_{0:k}, x_0)$ для состояния мобильной платформы x_k и карты m в момент k на основе всех наблюдений $Z_{0:k}$ и управляющих воздействий $U_{0:k}$ [2].

1.3. Решения на основе расширенного фильтра Калмана

Решение задачи SLAM методами на основе расширенного фильтра Калмана являются наиболее распространенными. Фильтр Калмана является рекурсивным фильтром, который оценивает вектор состояния динамической системы, используя ряд зашумленных измерений. Алгоритм EKF-SLAM позволяет не только уточнять оценку положения мобильной платформы на карте, но и положение всех обнаруженных ориентиров. При всей своей привлекательности, EKF тем не менее имеет свои недостатки, к которым можно отнести в первую очередь ограничение на количество ориентиров в системе.

1.3.1. Фильтр Калмана

В 1960-х годах появились работы Р. Калмана, посвященные решению задач линейной фильтрации. Затем фильтр Калмана стал предметом многочисленных исследований. Фильтр широко применяется в системах управления и в инженерных приложениях.

Фильтр Калмана – рекурсивный фильтр, оценивающий состояние линейной динамической системы. Состояние системы в текущий момент времени t описывается математическим ожиданием μ_t и ковариацией Σ_t величин системы.

Фильтр Калмана применяется для линейных систем. Линейная система описывается с помощью модели системы и модели измерения. Модель системы описывает каким образом, изменяется система с течением времени.

Модель системы необходима для предсказания состояния системы. Для применения фильтра Калмана необходимо, чтобы рассматриваемая модель перехода системы из одного состояния в другое представлялась в виде линейного уравнения:

$$x_t = A_t x_{t-1} + B_t u_t + \varepsilon_t, \quad (1.7)$$

где x_t и x_{t-1} – вектора состояния в момент времени t и $t-1$, u_t – управляющее воздействие, A_t – матрица эволюции системы, которая воздействует на вектор состояния x_{t-1}, B_t – матрица управления, которая прикладывается к вектору управляющего воздействия u_k, ε_t – нормальный случайный процесс с нулевым математическим ожиданием и ковариационной матрицей R_t.

Уточнение оценки состояния системы осуществляется фильтром с помощью наблюдения. Если осуществлено предсказание системы, то можно вычислить рассогласование между текущим измерением и предсказанным. На основе рассогласования делается корректировка состояния системы. При применении фильтра Калмана предполагается, что измерение моделируется с помощью линейного уравнения:

$$z_t = C_t x_t + \delta_t, \quad (1.8)$$

где C_t – матрица измерений, связывающая истинный вектор состояния и вектор произведенных измерений, δ_t – белый гауссовский шум с нулевым математическим ожиданием и ковариационной матрицей Q_t.

На основании этого фильтрация состоит из двух шагов: предсказание и корректировка, которые чередуются при работе алгоритма.

Во время первого шага происходит предсказание состояния системы на основе оценки состояния предыдущего шага. Данную оценку часто также именуют априорной из-за того, что она дается до выполнения каких-либо измерений и основывается только на математической модели системы.

Второй шаг отвечает за уточнения результата предсказания при помощи полученных измерений.

Предсказание

1) Предсказание состояния системы

$$\bar{\mu}_t = A_t \mu_{t-1} + B_t u_t. \quad (1.10)$$

2) Предсказание ошибки ковариации

$$\bar{\Sigma}_t = A_t * \Sigma_{t-1} * A_t^T + R_t. \quad (1.11)$$

Корректировка

1) Вычисление усиления Калмана

$$K_t = \bar{\Sigma}_t * C_t^T * (C_t * \bar{\Sigma}_t * C_t^T + Q_t)^{-1}. \quad (1.12)$$

2) Обновление оценки с учетом измерения z_t

$$\mu_t = \bar{\mu}_t + K_t * (z_t - C_t(\bar{\mu}_t)). \quad (1.13)$$

3) Обновление ошибки ковариации

$$\Sigma_t = (I - K_t * C_t) * \bar{\Sigma}_t, \quad (1.14)$$

где K_t – коэффициент усиления Калмана.

1.3.2. Расширенный фильтр Калмана

Фильтр Калмана имеет недостаток, заключающийся в том, что он применяется для оценки линейной динамической системы. Многие динамические системы не являются линейными. Т.е. модель системы и модель измерения представляются в виде нелинейных уравнений. Например, мобильная платформа, движущаяся с постоянной поступательной и вращательной скоростью, обычно имеет криволинейную траекторию, которая не может быть описана с помощью линейных уравнений.

Расширенный фильтр Калмана очень похож на простой фильтр Калмана, за тем исключением, что он может быть использован для оценки параметров нелинейных динамических систем. При применении расширенного фильтра Калмана модель системы описывается с помощью нелинейного уравнения:

$$x_t = g(u_t, x_{t-1}) + \varepsilon_t, \quad (1.15)$$

где $g(u_t, x_{t-1})$ – нелинейная функция, связывающая предыдущее состояние системы с текущим.

И модель измерения определяется с помощью нелинейного уравнения:

$$z_t = h(x_t) + \delta_t, \tag{1.16}$$

где $h(x_t)$ – нелинейная функция, связывающая состояние системы и измерения.

Расширенный фильтр Калмана, также как и классический, состоит из двух основных этапов:

Предсказание

1) Предсказание состояния системы

$$\bar{\mu_t} = g(\mu_{t-1}, u_t). \tag{1.17}$$

2) Предсказание ошибки ковариации

$$\bar{\Sigma_t} = G_t * \Sigma_{t-1} * G_t^T + R_t, \tag{1.18}$$

где $\bar{\mu_t}$ – предсказание состояния системы в текущий момент времени, $g(\mu_{t-1}, u_t)$ – функция предсказания состояния системы, u_t – управляющее воздействие в текущий момент времени, $\bar{\Sigma_t}$ – предсказание ошибки состояния системы в текущий момент времени, G_t – матрица перехода между состояниями, R_t – шум системы.

Корректировка

1) Вычисление коэффициента усиления Калмана

$$K_t = \bar{\Sigma_t} * H_t^T * (H_t * \bar{\Sigma_t} * H_t^T + Q)^{-1}. \tag{1.19}$$

2) Обновление оценки с учетом измерения z_t

$$\mu_t = \bar{\mu_t} + K_t * (z_t - h(\bar{\mu_t})). \tag{1.20}$$

3) Обновление ошибки ковариации

$$\Sigma_t = (I - K_t * H_t) * \bar{\Sigma_t}, \tag{1.21}$$

где K_t – усиление Калмана, H_t – матрица измерения, отображающая отношение измерений и состояний, Q – ковариация шума измерения, z_t – измерение в текущий момент времени, I – матрица идентичности.

1.3.3. Алгоритм EKF-SLAM

Задача SLAM на базе расширенного фильтра Калмана состоит из следующих основных частей:

1) Обнаружение ориентиров в пространстве.

2) Ассоциация данных, т. е. поиск соответствий ориентиров.

3) Оценка месторасположений мобильной платформы и ориентиров.

4) Уточнение месторасположений мобильной платформы и ориентиров.

Применяются различные способы реализации отдельных подзадач в рамках указанной задачи. Таким образом, существует возможность комбинировать различные реализации отдельных алгоритмов и улучшать их по отдельности.

1. Обнаружение ориентиров в пространстве. Основная цель задачи SLAM состоит в том, чтобы получить точную оценку месторасположения мобильной платформы с помощью наблюдения и оценки месторасположения ориентиров. Информация, необходимая для определения месторасположения мобильной платформы, зависит от точности используемого сенсора и точности оценок месторасположения наблюдаемых ориентиров. Геометрическое распределение ориентиров также играет роль. Например, когда нужно ограничить число ориентиров на карте, более выгодно сохранять ориентиры, удаленные друг от друга, и чем те, которые близки друг к другу. Кроме того, оценки ориентиров, которые плохо коррелированы, предоставляют больше информации. Когда две оценки ориентиров полностью коррелированы, то такие ориентиры не дают полезной информации, за исключением геометрической. Еще важнее выбирать ориентиры, таким образом, чтобы была максимальная точность и производительность у алгоритма [4, 5].

2. Ассоциация данных, т.е. поиск соответствий ориентиров. Ориентиры, обнаруженные датчиками аппарата на каждом шаге, включают в себя уже существующие ориентиры на карте, а также новые ориентиры. Если не провести соответствие найденных ориентиров с уже существующими на

карте, то на нее будут добавлены ориентиры-дубликаты, что приведет к неправильной работе алгоритма SLAM.

В большинстве алгоритмов SLAM на базе расширенного фильтра Калмана используют ассоциации ближайшего соседа. В нем для ассоциации каждого наблюдаемого ориентира проводится сравнение расстояния до существующих ориентиров с заранее заданным порогом. Этот простой алгоритм не учитывает взаимосвязь между ориентирами, поэтому вероятность неправильной ассоциации у алгоритма высока.

3. Оценка месторасположения мобильной платформы и ориентиров. Состояние мобильной платформы в произвольный момент времени можно представить с помощью вектора оценки месторасположения X_p и ковариационной матрицы P_p, имеющих вид:

$$X_p = \begin{bmatrix} x_p \\ y_p \\ \theta_p \end{bmatrix}^T, \tag{1.22}$$

$$P_p = \begin{bmatrix} \sigma^2_{x_p x_p} & \sigma^2_{x_p y_p} & \sigma^2_{x_p \theta_p} \\ \sigma^2_{y_p x_p} & \sigma^2_{y_p y_p} & \sigma^2_{y_p \theta_p} \\ \sigma^2_{\theta_p x_p} & \sigma^2_{\theta_p y_p} & \sigma^2_{\theta_p \theta_p} \end{bmatrix}, \tag{1.23}$$

где x_p – оценка координаты x мобильной платформы, y_p – оценка координаты y мобильной платформы, θ_p – оценка ориентации мобильной платформы.

Диагональные элементы являются дисперсией координат, а внедиагональные – показывают меру линейной зависимости оценки координат.

Состояние ориентиров можно представить в виде вектора оценки их месторасположения X_l и ковариационной матрицы P_l имеющих вид:

$$X_l = \begin{bmatrix} x_1 \\ y_1 \\ \dots \\ x_n \\ y_n \end{bmatrix}^T, \tag{1.24}$$

$$P_l = \begin{bmatrix} \sigma^2_{x_1x_1} & \sigma^2_{x_1y_2} & \cdots & \sigma^2_{x_1x_{n-1}} & \sigma^2_{x_1y_n} \\ \sigma^2_{x_1y_1} & \sigma^2_{y_1y_2} & \cdots & \sigma^2_{y_1x_{n-1}} & \sigma^2_{y_1y_n} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma^2_{x_nx_1} & \sigma^2_{y_nx_2} & \cdots & \sigma^2_{x_nx_{n-1}} & \sigma^2_{x_ny_n} \\ \sigma^2_{x_ny_1} & \sigma^2_{y_ny_2} & \cdots & \sigma^2_{x_ny_{n-1}} & \sigma^2_{y_ny_n} \end{bmatrix}, \quad (1.25)$$

где n – количество обнаруженных ориентиров, x_i – оценка координаты x -го ориентира, y_i – оценка координаты y i-го ориентира.

В алгоритме EKF-SLAM карта будет представлять собой вектор состояния X, отражающий оценки положения мобильной платформы и ориентиров, и ковариационной матрицей P, имеющих вид:

$$\mathrm{X} = \begin{bmatrix} X_p \\ X_l \end{bmatrix}^T, \quad (1.26)$$

$$P = \begin{bmatrix} P_p & P_{pl} \\ P_{lp} & P_l \end{bmatrix}, \quad (1.27)$$

где P_{pl} и P_{lp} – ковариационные матрицы, отражающие зависимость между оценкой координат мобильной платформы и оценками положения ориентиров.

Оценка месторасположения мобильной платформы и ориентиров (оценка состояния системы) обновляется на основе состояния системы в предыдущий момент времени X_{k-1}, кинематической модели и одометрических данных.

4. Уточнение месторасположения мобильной платформы и ориентиров. Во время этапа корректировки происходит уточнение месторасположений объектов на карте, используя рассогласования между предсказанными месторасположениями наблюдаемыми ориентирами и полученными месторасположениями ориентиров с помощью датчиков. [6].

1.4. Решения на основе фильтра частиц

Многие популярные решения, основанные на расширенном фильтре Калмана, моделируют апостериорное распределение по картам с использованием унимодального гауссовского распределения. Известно, что EKF плохо масштабируется по отношению к количеству ориентиров на карте, и плохо работает при неправильной ассоциации данных. Решения, основанные на фильтре частиц, моделируют мультимодальное распределение, стоимость обновления которого равна $O(\log(n))$ относительно от количества ориентиров. Стоит также отметить, что фильтр частиц хорошо работает при неправильной ассоциации данных.

Первый, кто применил фильтр частиц в задаче SLAM, был Мерфи в 1991 году. Он первый, кто использовал метод Рао-Блэквелла и показал, что для решения задачи SLAM может быть использован метод выборки. Затем Монтемерло и Трун представили алгоритм FastSLAM. Их основным вкладом была древовидная структура данных, служащая для обновления оценок месторасположения ориентиров, которая требует меньше памяти [7].

Алгоритм FastSLAM состоит из двух основных шагов:

1) Предсказание. Как правило, оценка предсказания происходит с помощью кинематической модели робота.

2) Корректировка. Оценка месторасположения мобильной платформы корректируется с помощью модели наблюдения.

Вектор состояния в алгоритме делится на две части. Первая часть соответствует месторасположению мобильной платформы относительно карты окружающей среды. Вторая часть соответствует карте. Поскольку оценка месторасположения ориентиров не зависит от месторасположения мобильной платформы, то задачу SLAM можно разделить на две части. В первой части используется фильтр частиц для определения месторасположения мобильной платформы. Во второй части используется расширенный фильтр Калмана для определения месторасположения ориентиров. Такой подход имеет низкую вычислительную сложность по

сравнению с EKF-SLAM, но имеет недостаток, заключающийся в том, что точность оценки карты и локализации меньше чем у EKF SLAM.

1.4.1. Фильтр частиц

В фильтре частиц распределение ошибок и шумов представляется не в виде нормального распределения, а в виде распределения частиц (рис. 1.3). Т.е. распределение вероятностей может быть представлено конечным рядом экземпляров состояния (sample):

$$\{x_i, w_i\}, \tag{1.28}$$

где w_j – весовой коэффициент

$$\sum w_i = 1. \tag{1.29}$$

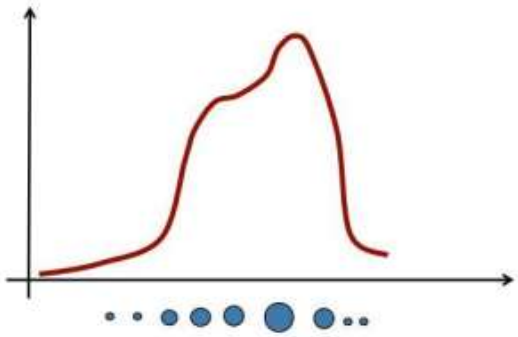


Рис. 1.3. Пример представления распределения в фильтре частиц (круги изображают экземпляры состояния, размер круга – вес экземпляра)

В фильтре частиц оценка вектора состояния x_k определяется с помощью распределения частиц. Распределение частиц получается путем удаления частиц с малым весом и генерации новых частиц для поддержания определенного их количества. Вес частиц определяется с помощью текущего измерения. Если у частицы большой вес это означает, что частица хорошо соответствует текущему измерению. Если у частицы малый вес, то такая частица в дальнейшем не участвует в работе фильтра.

Основные этапы фильтра частиц:

1. Генерация нового распределения частиц из предыдущей плотности вероятности $p(x_k|x_{k-1}^i, z_k)$.

2. Определение значений ненормированных весов частиц.

$$\widetilde{w_k^i} = w_{k-1}^i \frac{p(z_k|x_k^i)p(x_k^i|x_{k-1}^i)}{p(x_k^i|x_{k-1}^i, z_k)}. \quad (1.30)$$

3. Определение нормирующего коэффициента

$$c_k = \sum_{i=1}^N \widetilde{w_k^i}. \quad (1.31)$$

4. Вычисление нормированных весов частиц

$$w_k^i = \frac{\widetilde{w_k^i}}{c_k}. \quad (1.32)$$

5. Нахождение математического ожидания аппроксимируемой плотности вероятности:

$$\widetilde{x_k} = \sum_{i=1}^N x_k^i w_k^i. \quad (1.33)$$

1.4.2. Алгоритм FastSLAM

Оценка положения ориентиров зависит от оценки месторасположения мобильной платформы. Следовательно, решение задачи SLAM может быть представлено следующим образом (рис. 1.4):

$$p(x, m\,|Z_{0:k}, U_{0:k}) = p(x|Z_{0:k}, U_{0:k}) \prod_M p(m_i|x, Z_{0:k}, U_{0:k}). \quad (1.34)$$

Проблема SLAM может быть разложена на M+1 задач. Одна задача – это оценка месторасположения мобильной платформы, другие M задач – оценка месторасположений M ориентиров. Алгоритм FastSLAM использует модифицированный фильтр частиц для оценки месторасположения платформы $p(x_k|z_k, u_k)$. Этот фильтр может обеспечивать хорошую оценку месторасположения, даже при нелинейной кинематике движения. А оценка месторасположения ориентиров $p(m_i|x_k, z_k, u_k)$ определяется с помощью фильтра Калмана, для этого используются отдельные фильтры для каждого отдельного ориентира. Так как оценки месторасположений ориентиров обусловлены оценкой траектории движения, каждая частица в фильтре

частиц имеет свою собственную оценку локальных ориентиров. Таким образом, для N частиц и M ориентиров будет в общей сложности $N * M$ фильтров Калмана, у которых размерность вектора состояния равен 2.

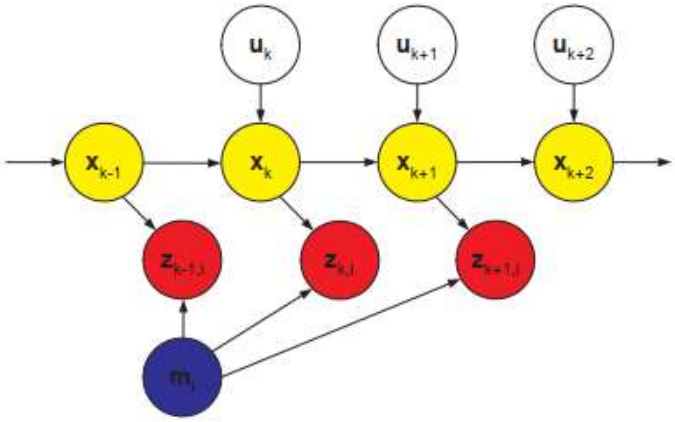


Рис. 1.4. Схема алгоритма FastSLAM.

Отличительной особенностью алгоритма FastSLAM является организация $N * M$ фильтров Калмана. Вместо массива фильтров используется древовидная структура, которая позволяет намного эффективнее обновлять состояния фильтров во время каждого наблюдения.

Алгоритм FastSLAM состоит из 4 основных этапов:

1) *Выборка.* С помощью управляющего воздействия u_k и измерения z_k фильтр Калмана частицы $x_k^{(i)}$ вычисляет оценку месторасположения мобильной платформы. Затем делается выборка из частиц:

$$x_k^{(i)} \sim \pi(x_k | X_{0:k-1}^{(i)}, Z_{0:k}, u_k), \tag{1.35}$$

где π – предсказанная вероятность месторасположения мобильной платформы.

2) *Вычисление весов частиц.* Для каждой частицы вычисляется весовой коэффициент:

$$w_k^{(i)} = w_{k-1}^{(i)} \frac{P(z_k | X_{0:k}^{(i)}, Z_{0:k-1}) P(x_k^{(i)} | x_{k-1}^{(i)}, u_k)}{\pi(x_k^{(i)} | X_{0:k-1}^{(i)}, Z_{0:k}, u_k)}. \tag{1.36}$$

3) *Повторная выборка.* Повторная выборка осуществляется путем удаления частиц с низким значением веса. Затем генерируется новое распределения частиц с добавлением новых недостающих частиц. Для новых частиц задается одинаковый вес $w_k^{(i)} = \frac{1}{N}$.

4) *Оценка карты.* Для каждой частицы выполняется оценка с помощью расширенного фильтра Калмана. [7].

Исследования показали, что оптимальное число частиц в фильтре составляет порядка 100 с точки зрения скорости работы и точности определения месторасположения мобильной платформы [100].

1.5. Решения, основанные на графах

В алгоритмах SLAM, основанных на графах, месторасположения камеры представляются в виде узлов графа. Пространственные ограничения между узлами, которые определяются с помощью наблюдений $\boldsymbol{z}_t$ и вектора управляющего воздействия $\boldsymbol{u}_t$ являются ребрами графа. Т.е. узел графа представляет собой месторасположение камеры, а ребра между узлами представляются ограничениями, которые связывают два последовательных месторасположения камеры. Ограничение представляет собой оценку перемещения между двумя позициями камеры. Перемещение определяется с помощью одометрических данных или с помощью сопоставления наблюдений между двумя позициями. После построения графа ищется такая конфигурация месторасположений, которая является наиболее вероятной. Такой процесс поиска конфигурации называется оптимизацией графа (рис. 1.5).

В решениях, основанных на графах можно выделить три основных этапа:

1) Построение графа.

2) Решение проблемы замыкания цикла.

3) Оптимизация графа.

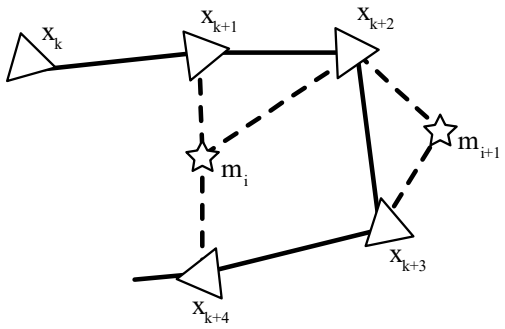


Рис. 1.5. Пример работы алгоритма SLAM, основанного на графах
(Узлы x_k – месторасположение камеры, узлы m_i – месторасположение i ориентира. Граф состоит из двух типов ребер. Сплошными линиями обозначены ребра, которые связывают последовательные месторасположения камеры. Пунктирными линиями обозначены ребра, которые связывают месторасположения камеры с ориентирами, которые наблюдались в этом месте)

При построении графа выделяются ключевые точки на изображении. Затем эти особенности сопоставляются с ключевыми точками, найденные на предыдущем кадре. Ложные пары отбрасываются обычно с помощью алгоритма RANSAC. Относительное движение вычисляется с использованием всех прошедших проверку пар с помощью алгоритмов сопоставления изображений или интерактивных алгоритмов ближайших точек. При использовании этих алгоритмов важно, чтобы перемещение между последовательными узлами был незначительным [93].

Для повышения сходимости алгоритма алгоритм замыкание циклов распознает ранее исследованные области с целью уменьшения накопленной ошибки локализации со временем. Обычно исследованная область определяется с помощью проверки принадлежности предсказанной позиции к некоторой окрестности уже сохраненных ориентиров. После определения исследованной области выполняется повторная локализация, по которой можно сократить накопленную ошибку оценки месторасположения камеры [94].

Цель задачи оптимизации графа – найти конфигурацию параметров или переменных месторасположений камеры и расположение ориентиров,

которые является наиболее вероятной. Различные алгоритмы использую различные критерии: глобальная ошибка по всем кадрам, локальная ошибка по подмножеству кадров и т.п.

Решения, основанные на графах, имеют высокую вычислительную сложность внутри узла из-за задачи поиска ближайших точек, а также окончательная оптимизация графа может потребовать больших вычислительных затрат.

1.5.1. Основные отличия между решениями, основанными на графах, с использованием расширенного фильтра Калмана и фильтра частиц

Подробное сравнение рассмотренных решений задачи одновременной локализации и построения карты приведено в табл. 1.

Таблица 1. Отличия между основными решениями задачи SLAM

	Преимущества	Недостатки
Решения, основанные на расширенном фильтре Калмана.	1) Высокая точность построения карты по сравнению с другими решениями при одинаковом количестве ориентиров.	1) Квадратичная вычислительная сложность. 2) Если соответствие ориентиров неверно, то работа фильтра, скорее всего, будет расходиться.
Решения, основанные на фильтре частиц.	1) Линейная вычислительная сложность. 2) Есть возможность убирать ошибочные соответствия ориентиров, что делает фильтр робастным.	1) Ошибки каждой частицы привносят вклад в общую карту, и накапливается там с течением времени.
Решения, основанных на	1) Линейная зависимость памяти от количества	1) Имеет высокую вычислительную

графах	объектов на карте. 2) Обновление графа является постоянным по времени.	сложность внутри узла из-за задачи поиска ближайших точек. 2) Окончательная оптимизация графа может потребовать больших вычислительных затрат.

1.6. Сенсоры

Как правило, информацию об окружающей среде мобильная платформа получает с помощью датчиков, такие как лазерные и ультразвуковые дальномеры и камеры.

1.6.1. Лидар

Термин «лидар» (Light Detection and Ranging) относится к системам радиолокации, работающим в оптическом диапазоне и использующим в качестве источника излучения лазер. Существуют два основных метода измерения дальности на основе лазера:

1) Время-полетный метод (Time-of-Flight, ToF). Эти системы использует короткие импульсы лазерного излучения, с высокой точностью фиксируя моменты их передачи и приёма откликов (отраженных сигналов), чтобы вычислить расстояния до объектов в окружающем пространстве. Расстояние до точки поверхности объекта, в которой произошло отражение лазерного луча, может быть вычислено по формуле:

$$D = 0{,}5 * c * t\ , \tag{1.37}$$

где c – скорость света, t – полное время прохождения светом пути до точки отражения и обратно, D – искомое расстояние до точки отражения.

2) Метод сдвига фаз. Лазер излучает непрерывный сигнал, к которому затем применяется синусоидальная амплитудная модуляция. В этом случае

время прохождения светом полного пути от передатчика до приемника будет прямо пропорционально сдвигу фаз в излученном и принятом сигналах:

$$t = \frac{\varphi}{2\pi} T, \tag{1.38}$$

где φ – фазовый сдвиг, T – период сигнала

После определения времени t прохождения луча, расстояние D вычисляется по формуле (1.37) [4, 9].

Лазерные сканирующие дальномеры имеют следующие преимущества:

1) Хорошая точность измерения дальности.
2) Угловое разрешение $0.5°$ или $0.25°$ в зависимости от режима работы.
3) Данные с лазерной сканирующей системы могут быть интерпретированы как дальность до препятствия в определенном направлении.

К недостаткам лазерных сканирующих систем относятся:

1) Датчик предоставляет информацию о дальности в определенной плоскости. Если некоторые препятствия лежат выше или ниже такой плоскости, то они не могут быть обнаруженными датчиком.
2) Лазерная сканирующая система является дорогой.
3) Некоторые материалы являются прозрачными для лазерной сканирующей системы, например, стекло.

1.6.2. Сонар

Термин «сонар» (sound navigation and ranging) относится к системе звуковой навигации для определения дальности до объектов посредством отражения звуковых волн. Такой тип датчиков актуален в применении алгоритмов SLAM для автономных подводных аппаратов, которые перемещаются в сложных подводных условиях.

Преимущества:

1) Низкая цена.
2) Высокая точность диапазона. Оценка диапазона обычно находится в пределах 1% от истинного значения.

Недостатки:

1) Слабое эхо.
2) Многократные отражения: звуковая волна может отражаться многократно от одного объекта прежде, чем достигнет приемника, вызывая ложные показания.
3) Перекрестный разговор: когда несколько сонаров посылают одинаковые звуковые волны, тогда может случиться так, что получатель получит звуковую волну от другого датчика.
4) Большой угол разрешения. Благодаря широкому основному лепестку (25°) отражение может происходить откуда угодно вдоль частей сферической поверхности.
5) Ограниченная дальность действия. Обычно датчики могут обнаружить небольшой объект на расстоянии 0,2-10 м.
6) Ограничение скорости звука. В связи с тем, что скорость звука (340 м/с) очень медленная (по сравнению со скоростью света, которая используется в лазерных сканерах) скорость измерения датчика ограничена.

1.6.3. Визуальные сенсоры

На сегодняшний день самым перспективным датчиком, применяемым в задаче SLAM, являются камеры. Это связано с тем, что, в отличие от алгоритмов, основанных на использовании лазерных и ультразвуковых датчиков в качестве основных источников информации об окружающей среде, алгоритмы SLAM могут с помощью камер получать более детальную информацию об объектах окружающей среды.

Преимущества:

1) Большой объем информации
2) Возможность получения трехмерной информации об окружающей среде.
3) Камеры являются пассивными датчиками.

Недостатки:

1) Высокие вычислительные требования для извлечения информации из изображений.
2) Зависимость от освещения.

1.7. Визуальный SLAM

Задача одновременной локализации и картирования широко изучался в течение последних двух десятилетий, и, как следствие, возникло много направлений. Одним из таких направлений является визуальный SLAM, в котором используется в качестве датчиков камера. Использование камер позволяет снизить затраты при разработке автономных систем [10].

Алгоритмы визуального SLAM обычно состоят из четырех этапов:

1) Инициализация.
2) Отслеживание перемещения.
3) Построение карты.
4) Замыкание петли.

При инициализации алгоритма определяется система координат для оценки положения камеры и построения трехмерной карты в окружающей среде. Затем извлекаются данные для построения карты. Существуют два основных подхода для построения карты: прямые методы и методы, основанные в использовании особых точек.

В прямом методе карта представляет собой облако точек соответствующие пикселям изображения, координаты которых вычисляются с помощью карты глубины. В этом случае трехмерная карта получается очень плотной. В методах, основанных в использовании особых точек, карта

представляет собой облако точек соответствующие особым точкам на изображении. В этом случае карта получается разряженной.

После инициализации алгоритма выполняется этап отслеживания. На этом этапе обычно используют алгоритм визуальной одометрии. Чтобы оценить перемещение камеры по изображению относительно карты, используют поиск 2D или 3D соответствий. Поиск 2D соответствия означает поиск соответствий особых точек на изображениях. Поиск 3D соответствий означает поиск соответствий среди облака точек, представляющих карту. Поиск 2D-3D является комбинированным поиском. Затем оценка положения камеры вычисляется из найденных соответствий путем решения задачи «перспектива в n точках» [11, 12].

Построение карты происходит по нескольким изображениям, полученным во время движения камеры. Координаты точек, представляющую карту, описываются с помощью трехмерной параметризации, например, с помощью инверсной глубины [32-34].

Для того чтобы уменьшить ошибку, обычно выполняется оптимизация глобальной карты. Для этого необходимо обнаружить петлю, т.е. распознать, что камера в этом месте ранее находилось. Потом вычисляется накопленная ошибка (разница между текущим предсказанным местом и обнаруженным). Такой метод вычисления ошибки называется замыкание петли. Затем вычисленную ошибку используют в алгоритмах глобальной оптимизации [3, 13-15].

1.7.1. Алгоритм ORB SLAM

Алгоритм ORB-SLAM является одной из самых популярных систем SLAM на основе ключевых кадров и имеет хороший результат оценки траектории движения камеры.

Алгоритм ORB-SLAM включает в себя три потока, которые работают параллельно: трекинг, построения локальной карты и замыкание цикла.

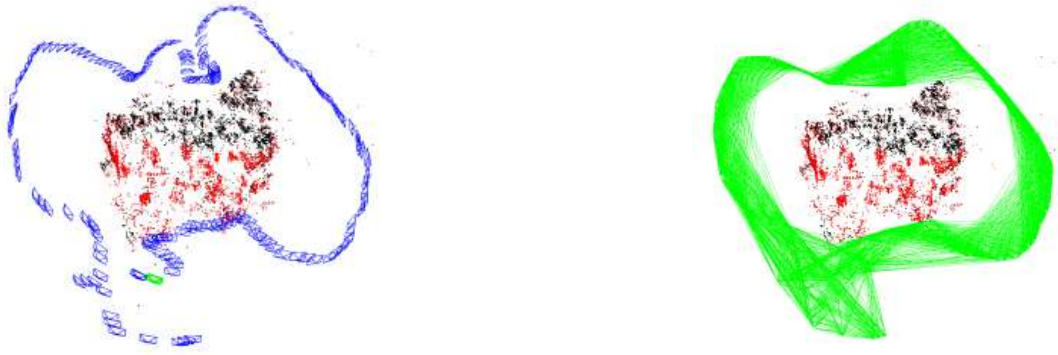
Трекинг отвечает за локализацию камеры на каждом кадре и за принятие решения о том, какой кадр использовать как ключевой кадр. Перед трекингом происходит инициализация карты по первым двум кадрам. В этот момент определяются матрица гомографии и фундаментальная матрица. После инициализации запускается модуль трекинга. Трекинг состоит из четырех этапов [99]:

1) Поиск особых точек ORB.

2) Определение перемещения. Для каждого кадра выполняется поиск соответствия особых точек между текущим кадром и предыдущим, и с помощью алгоритма сопоставления изображений делается оценка месторасположения камеры.

3) Релокация. Если во время трекинга произошел сбой (например, из-за резких движений), то для определения перемещения используется алгоритм распознавания места. Как только удалось сделать оценку положения камеры и провести соответствие между особыми точками, то локальная карта извлекается из графа с ключевыми кадрами, который поддерживается системой (рис. 1.6). Затем с помощью перепроецирования ищутся соответствия между особыми точками, и оптимизируется месторасположения камеры для всех совпадений.

4) Инициализация локальной карты. Локальная карта содержит особые точки текущего кадра и предыдущего ключевого кадра. При большом расстоянии между местом съемки текущего ключевого кадра и местом съемки предыдущим ключевым кадром инициализируется новая локальная карта.

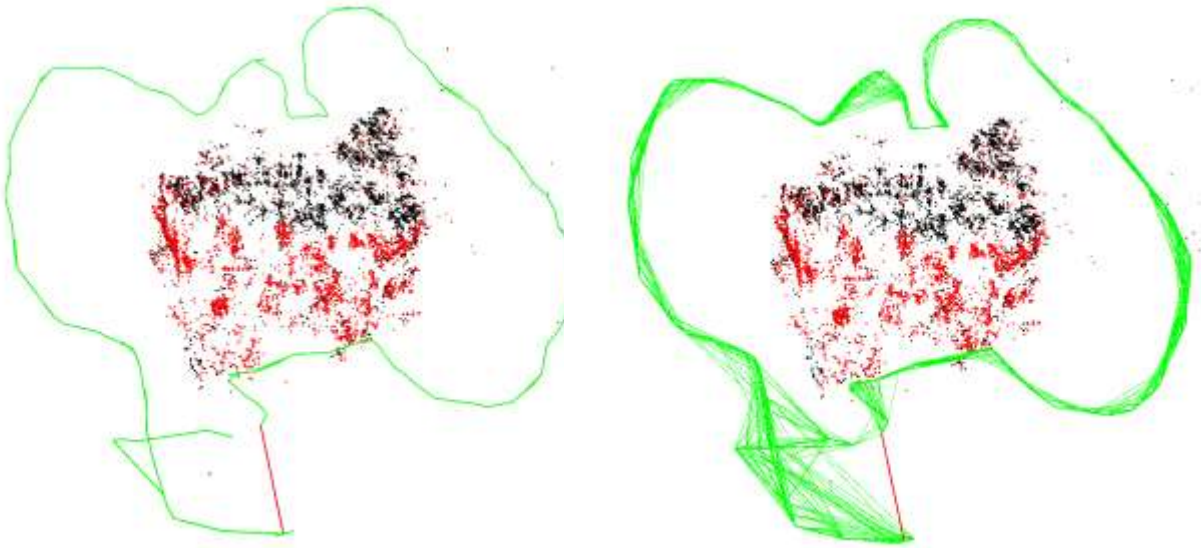


а) б)

Рис. 1.6. Карта окружающей среды и граф: а) ключевые кадры; б) граф с ключевыми кадрами

Модуль построения локальной карты обрабатывает новые ключевые кадры с использованием алгоритма сопоставления изображений для получения оптимальной реконструкции окружающей среды. Для особых точек, у которых не удалось найти соответствия с предыдущим ключевым кадром, выполняется поиск соответствия в других ключевых кадрах в графе. Если количество найденных особых точек в других ключевых кадрах очень большое, то текущий ключевой кадр отбрасывается. Таким образом, осуществляется отбор ключевых кадров. Через некоторые время на основе информации полученной во время трекинга, применяется отбор особых точек – сохраняются особые точки только с наибольшей информативностью [62].

Модуль замыкания цикла с каждым новым ключевым кадром ищет цикл. Если цикл обнаружен, то вычисляется ошибка, накопленная в цикле. Затем оба конца цикла выравниваются и сливаются дублированные особые точки (рис. 1.7). Потом выполняется оптимизация графа месторасположений. Для этого используется алгоритм оптимизации Левенберга-Марквардта [62, 21].



а) б)

Рис. 1.7. Карта окружающей среды и граф: а) полученная траектория движения (зеленое) и цикл замыкания (красное); б) основной граф

1.7.2. Алгоритм LSD-SLAM

Алгоритм LSD-SLAM использует камеру для создания карты окружающей среды и определения месторасположения камеры на созданной карте. Многие алгоритмы vSLAM полагаются на поиске особенностей или ориентиров в окружающей среде для отслеживания и построения траектории камеры. Но алгоритм LSD-SLAM отличается тем, что использует фотометрическую информацию, то есть использует интенсивность каждого пикселя. Аналогично алгоритмам PTAM и DTAM, LSD-SLAM использует ключевые кадры и по ним вычисляет глубину по множеству пикселей, выполняя множество стереофонических сравнений с последовательными кадрами изображения.

Алгоритм состоит из трех основных модулей: трекинг, оценка глубины и оптимизация карты, как показано на рис. 1.8.

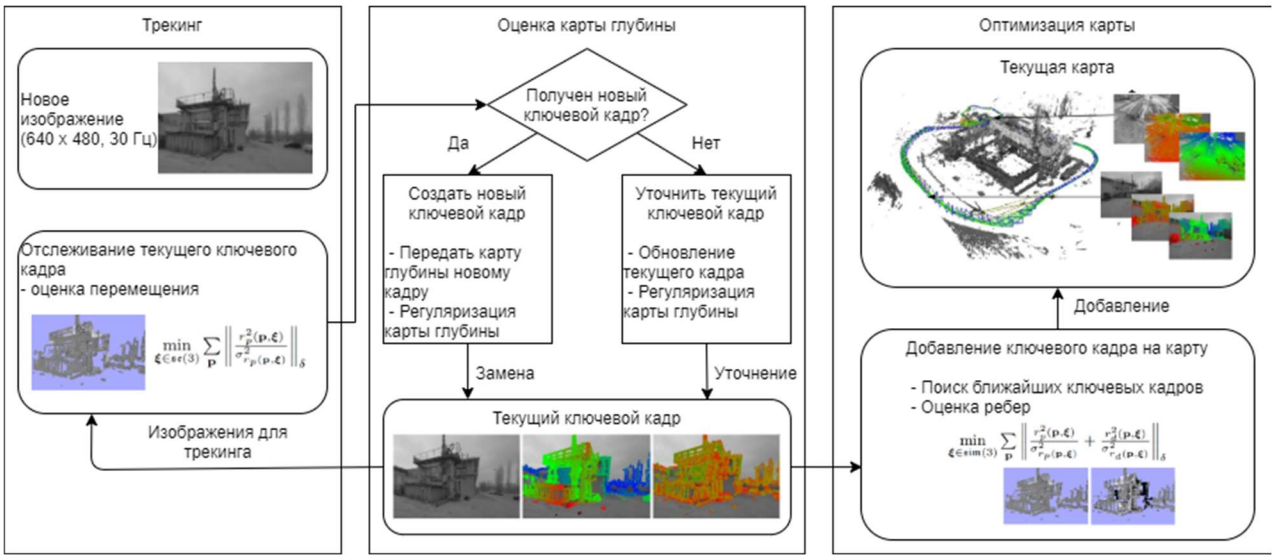


Рис. 1.8. Структурная схема работы алгоритма LSD-SLAM

Трекинг (*сопровождение*). Трекинг непрерывно отслеживает новые изображения с камер. То есть, он оценивает положение камеры относительно предыдущего ключевого кадра.

Первым этапом алгоритма LSD-SLAM является получение изображения с камеры и дальнейшее преобразование в монохромное изображение. Самое первое изображение становится ключевым кадром и инициализируется обратная карта глубины, где для каждого пикселя задается случайное значение с большим средним значением и с большой дисперсией. Последовательные кадры изображения выравниваются по ключевому кадру путем непосредственной минимизации фотометрической ошибки $E(\xi)$ между пикселями для оценки положения камеры относительно ключевого кадра с использованием алгоритма Гаусса-Ньютона. Фотометрическая ошибка относительно текущего ключевого кадра имеет вид:

$$E(\xi) = \sum_i (I_{kf}(p_i) - I(\omega(p_i, D_{kf}(p_i), \xi)))^2, \tag{1.39}$$

где ξ – трехмерное положение камеры относительно ключевого кадра, $p_i - i$ – релевантное положение пикселя изображения, для которого существует глубина, $I_{kf}(p)$ – интенсивность пикселя ключевого кадра, $I(p)$ – интенсивность пикселя текущего изображения, $D_{kf}(p)$ – значение обратной

глубины пикселя p ключевого кадра, $\omega(p_i, D_{kf}(p_i), \xi)$ – проекция пикселя p с соответствующей ему обратной глубиной значения для нового кадра изображения, учитывая преобразование ξ.

Для очередного кадра j и текущего ключевого кадра i ищется ξ путем минимизации фотометрической ошибки:

$$E_p(\xi_{i,j}) = \sum \left\| \frac{r_p^2(p,\xi_{ij})}{\sigma_{r_p}^2(p,\xi_{ij})} \right\|_\delta, \tag{1.40}$$

$$r_p(p, \xi_{ij}) = I_i(p) - I_j(\omega(p, D_i(p), \xi_{ij})), \tag{1.41}$$

$$\sigma_{r_p}^2(p, \xi_{ij}) = 2\sigma_I^2 + \left(\frac{\partial r_p(p,\xi_{ij})}{\partial D_i(p)}\right)^2 V_i(p), \tag{1.42}$$

где σ_I^2 – дисперсия шума интенсивности изображения и $\| \ \|_\delta$ – функция потерь Хьюбера. Минимизация происходит с помощью метода Ньютона-Гаусса с итеративным пересчетом весов (IRLS).

Оценка глубины. Оценка карты глубины происходит в три этапа:

1) Выбор ключевого кадра. Месторасположения камеры выполняется с использованием пирамидальной структуры представления изображения. Оценка месторасположения камеры начинается с самого высокого уровня пирамидальной структуры (изображения с самым низким разрешением), затем оценка уточняется по следующему уровню пирамидальной структуры (изображения с более высоким разрешением) и так далее для каждого уровня. После того как сделана оценка месторасположения камеры принимается решения о том, чтобы текущее изображение стало новым ключевым кадром. Решение определяется на основе расстоянии между месторасположением съемки текущего ключевого кадра и месторасположением съемки текущего изображения.

2) Создание карты глубины. После определения нового ключевого кадра происходит инициализация карты глубины с использованием данных карты глубины предыдущего кадра. После этого происходит оптимизация и нормировка карты глубины и отсечение выбросов.

3) Уточнение карты глубины. Уточнение карты глубины происходит с помощью нескольких изображений, относящиеся к ключевому кадру. Алгоритм уточнения карты глубины оценивает значение глубины каждого пикселя по нескольким изображениям.

Оптимизация карты. Карта, состоящая из набора ключевых кадров и отслеживаемых ограничений, непрерывно оптимизируется в фоновом режиме с помощью алгоритма g2o (General Graph Optimization), осуществляя глобальную оптимизацию карты [22, 23].

1.7.3. Алгоритм RGB-D SLAM

Алгоритм RGB-D SLAM позволяет быстро получать цветные трехмерные модели объектов и сцен в помещении с помощью RGB-D камеры, например, Kinect. Алгоритм использует особые точки, такие как SURF и SIFT для сопоставления пар полученных изображений и использует RANSAC для надежной оценки трехмерного преобразования между ними. Для достижения быстрой обработки текущее изображение сопоставляется только с подмножеством предыдущих изображений. Затем строится граф, узлы которого соответствуют точками обзора камеры, а ребра соответствуют предполагаемым трехмерным преобразованиям. Затем граф оптимизируется с помощью алгоритма HOG-Man, чтобы уменьшить накопленную ошибку.

Структурная схема алгоритма представлена на рис. 1.9. Вычисление оценки траектории делится на два модуля: внешний и внутренний. Внешний модуль вычисляет пространственные преобразования между отдельными изображениями. Внутренний модуль оптимизирует местоположения камеры и строит граф позиций.

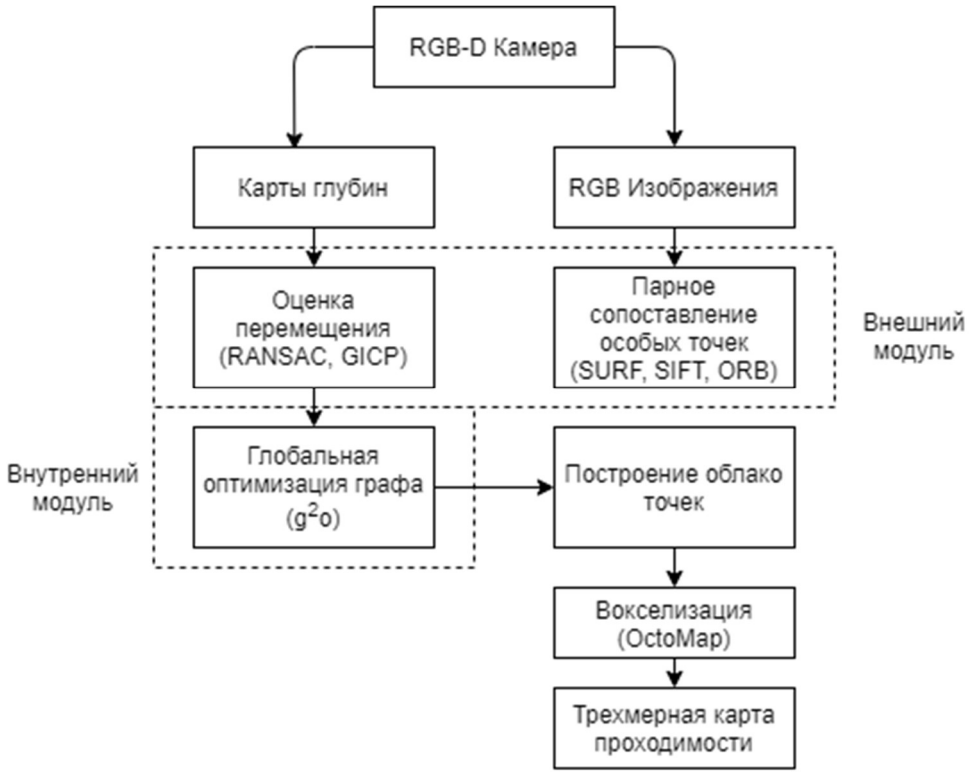


Рис. 1.9. Структурная схема алгоритма RGB-SLAM

Внешний модуль отвечает за вычисление перемещения камеры по полученным данным RGB-D камеры. Внешний модуль использует RGB-изображение, полученное с RGB-D камеры, для определения особых точек и извлечения дескрипторов. В качестве детекторов и дескрипторов могут применяться SURF, SIFT, ORB. После детектирования особые точки проецируются в трехмерном пространстве с помощью данных карты глубины. На этом этапе появляется много выбросов, поскольку визуальные признаки не обеспечивают идеальную надежность. Кроме того, карты глубины и цветные изображения часто не совместимы из-за разных частот получения данных и интерполяции карты глубины. Также особые точки часто лежат на границах объектов, в которых может быть неправильно вычислено значение глубины. Из-за этих причин задача поиска соответствия между особыми точками является сложной. Затем особые точки сопоставляются с ранее извлеченными дескрипторами, и вычисляется преобразование между позициями RGB-D камеры с помощью алгоритма

RANSAC. Следует отметить, что на этом этапе используется карта глубины для формирования плотных облаков особых точек.

Перемещение между двумя позициями камеры представляются в виде ребра в графе позиций. Из-за ошибок оценки перемещения камеры не получается глобально согласованной траектории. Чтобы создать глобально согласованную траекторию, производят оптимизацию графа. Как правило, функция ошибки имеет вид:

$$F(x) = \sum_{(i,j)\epsilon C} e\left(x_i, x_j, z_{ij}\right)^T \Omega_{ij} e(x_i, x_j, z_{ij}), \quad (1.43)$$

где $x = (x_1^T, \dots, x_n^T)^T$- вектор, представляющий координаты x_i, z_{ij} и Ω_{ij}, которые соответственно являются средними значениями и информационной матрицей ограничения, связывающего координаты x_j, т.е. парное преобразование, вычисляемое внешним модулем. $e(x_i, x_j, z_{ij})$ – векторная функция ошибок, которая измеряет, насколько хорошо оценка месторасположений x_i и x_j удовлетворяют ограничению z_{ij}. Когда x_i и x_j полностью соответствуют ограничению, то разность оценки месторасположений точно соответствует предполагаемому преобразованию [24].

1.8. Краткие выводы по главе 1

Описывается проведенный анализ научно-технической литературы по вопросам задачи одновременной локализации и построения карты. Рассмотрены существующие подходы к решению этой задачи: расширенный фильтр Калмана и фильтр частиц. Установлено, что при использовании расширенного фильтра Калмана оценка месторасположения мобильной платформы получается точнее, чем при использовании фильтра частиц, потому что оценка месторасположения используется в вычислениях корреляции между ориентирами. Но существуют и недостатки в использовании расширенного фильтра Калмана, а именно – квадратичная вычислительная сложность, а также проблема в ассоциации данных.

Анализ показал, что расширенный фильтр Калмана в задаче SLAM можно модифицировать путем манипуляций с векторами состояния и ковариации. Такое возможно при условии, что ориентиры, рассматриваемые в динамической системе, являются стационарными и независимыми. Это условие является основой для алгоритмов SLAM на базе фильтра частиц, но оказывается, что его можно также применить и для расширенного фильтра Калмана.

Если ориентиры являются независимыми и дано точное месторасположение мобильной платформы, то корреляция между ними не должна наблюдаться. Как только увеличивается ошибка оценки месторасположения мобильной платформы, то появляется корреляция между ориентирами, за счет которой фильтр подавляет ошибку месторасположения мобильной платформы и ориентиров. Отсюда следует, что, при пренебрежении корреляцией между оценками ориентиров и самими ориентирами, ошибка позиционирования увеличивается, но при этом увеличивается и быстродействие фильтра.

На практике эта ошибка не всегда растет: например, при плохой ассоциации данных возможно пренебрежение ложными ориентирами, а при построении больших карт корреляция между ориентирами может быть избыточной.

Также рассмотрены основные типы сенсоров, используемые в задаче SLAM, которые можно комбинировать. Наиболее точным сенсором является лидар, а более информативными – визуальные сенсоры.

Глава 2. АЛГОРИТМ SLAM С ПРИМЕНЕНИЕМ КАМЕРЫ С ОБЪЕКТИВОМ ТИПА «РЫБИЙ ГЛАЗ» И ЛАЗЕРНОЙ СКАНИРУЮЩЕЙ СИСТЕМЫ

2.1. Общая структура алгоритма SLAM с применением камеры с объективом типа «рыбий глаз» и лазерной сканирующей системы

В последние годы в задаче SLAM стали добавлять вспомогательные датчики. Наиболее часто используемыми вспомогательными датчиками являются инерциальные датчики. Системы SLAM с такими датчиками обычно работают лучше [48-53]. Помимо этого, можно также интегрировать в задаче SLAM камеру и лидар [54-57]. Например, авторы [21] применили визуальный одометр для получения начальных данных для алгоритма сопоставления сканов лидара ICP и получили хорошие результаты. На основании этих исследований в данной работе предложено интегрировать два датчика: камеру и лазерную сканирующую систему. Интеграция предполагает использовать два типа ориентиров, полученных с разных датчиков в расширенном фильтре Калмана при соблюдении условия их стационарности и независимости.

В настоящей работе для сбора данных используется мобильная платформа (рис. 2.1), оснащенная лазерной сканирующей системой и камерой «рыбий глаз» (разрешение 1280x960), закрепленной таким образом, чтобы можно было получать панорамные изображения.

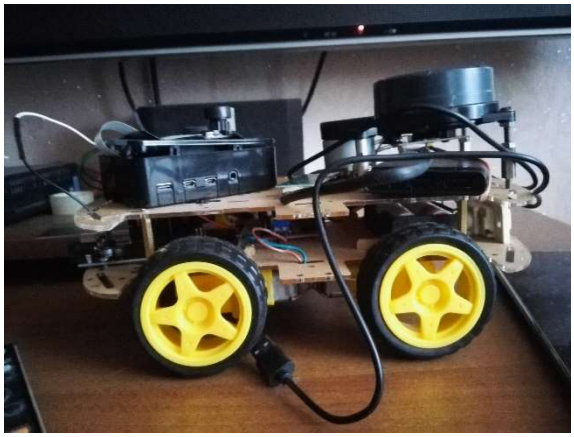


Рис. 2.1. Мобильная платформа для сбора данных

Через определенные промежутки времени мобильная платформа записывает данные лазерной сканирующей системы, показания одометрии и изображения камеры. Затем из этих данных строится трехмерная карта. На рис. 2.2 представлена блок-схема работы соответствующего алгоритма.

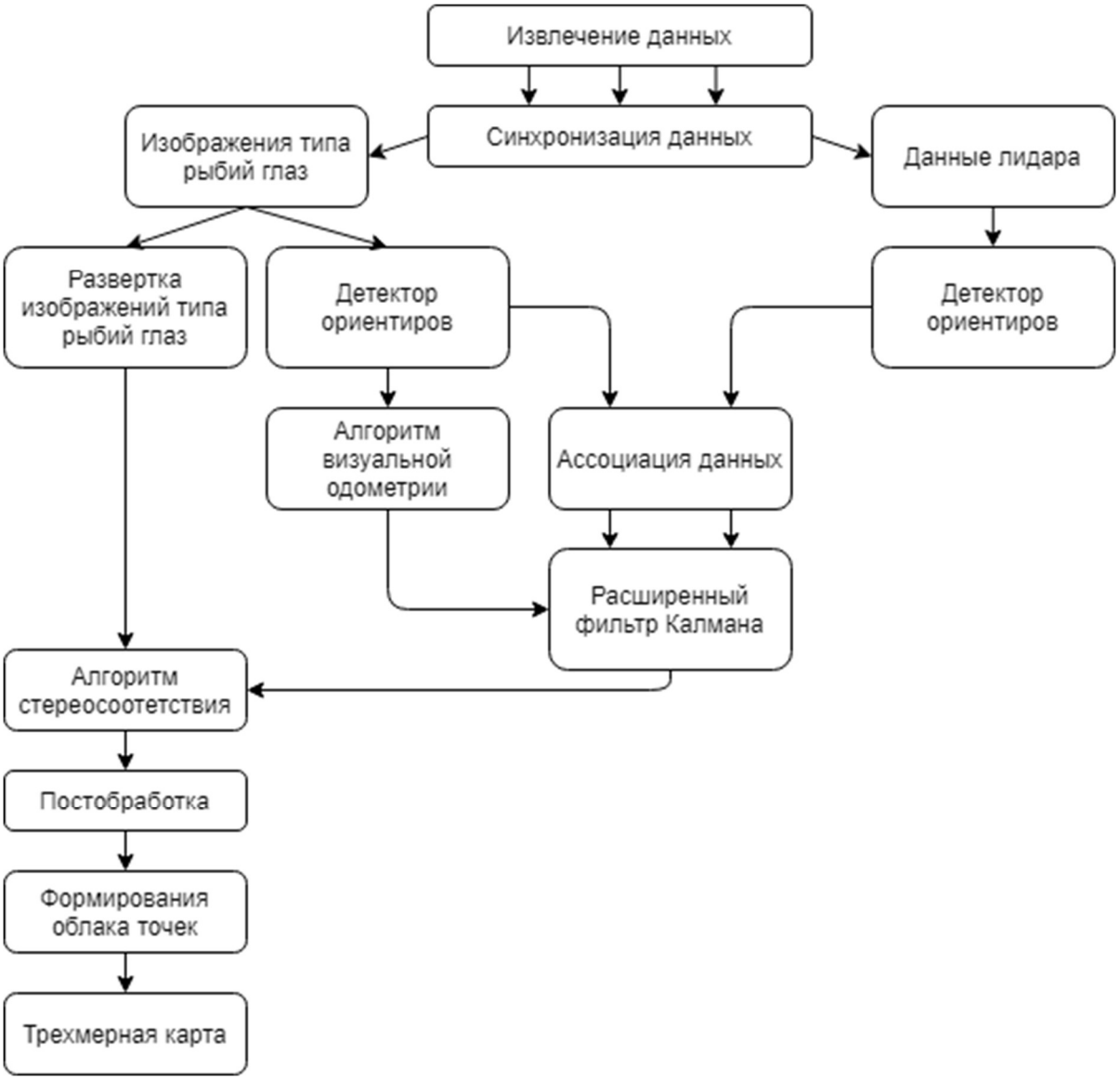


Рис. 2.2. Структурная схема алгоритма обработки данных с мобильной платформы

Алгоритм состоит из шести основных этапов:

1) Получение и синхронизация данных. Мобильная платформа управляется дистанционно человеком-оператором. Программное обеспечение на ней записывает данные с датчиков в файл rosbag. Затем файл обрабатывается алгоритмом, реализованном на платформе ROS. Поскольку датчики имеют разную частоту считывания, полученные данные необходимо

синхронизировать по времени. Берется малый промежуток времени, и если все датчики в этот момент сработали, то полученные данные обрабатываются.

2) Обнаружение ориентиров в пространстве. На этом этапе алгоритм использует ORB детектор для обнаружения особых точек на изображении. Для отслеживания каждого найденного ориентира используются алгоритм Лукаса-Канаде. Для обнаружения ориентиров по данным лидара используется один из следующих детекторов: детектор с использованием согласованной фильтрации, детектор на основе метода масштабного пространства кривизны, детектор Teh-Chin, детектор Wu. Перед использованием детекторов применяется преобразование данных лидара в комплекснозначный сигнал. Без этого преобразования невозможно применять эти детекторы ориентиров.

3) Поиск соответствий ориентиров. На этом этапе ориентиры, найденные на изображении, не требуют поиска соответствий, поскольку используется алгоритм трекинга Лукаса-Канаде. Ассоциация ориентиров, найденных по данным лидара, происходит с помощью технического зрения.

4) Оценка месторасположения камеры и ориентиров. Этот этап связан с этапом предсказания расширенного фильтра Калмана. Для предсказания месторасположения мобильной платформы используется визуальная одометрия. Алгоритм визуальной одометрии учитывает плоскопараллельное движение и сферическую модель камеры.

5) Уточнение месторасположения камеры и ориентиров. Этот этап связан с этапом корректировки расширенного фильтра Калмана. Для уменьшения вычислительной сложности и повышения точности применяется один из алгоритмов построения локальной карты: алгоритм «разделяй и властвуй», алгоритм с адаптационным диапазоном наблюдения и алгоритм с равномерным использование ориентиров.

6) Построение трехмерной карты. Алгоритм построения трехмерной карты строит карту с использованием сферической модели камеры по карте

глубины. Карта глубины вычисляется по двум последовательным изображениям. Координаты съемки изображений вычисляются с помощью алгоритма SLAM.

2.2. Калибровка камеры

2.2.1. Модель камеры обскура

В техническом зрении наиболее популярной моделью камеры является модель с точечным отверстием, или камера-обскура (рис. 2.3). В такой модели некоторая точка в пространстве P проецируется в однородные координаты изображения следующим образом:

$$\begin{bmatrix} X_d \\ Y_d \\ w \end{bmatrix} = M \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} f_x & 0 & C_x \\ 0 & f_y & C_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix}, \quad (2.1)$$

где M – внутренние параметры камеры, f_x, f_y – фокусные расстояния и C_x, C_y – координаты центра изображения.

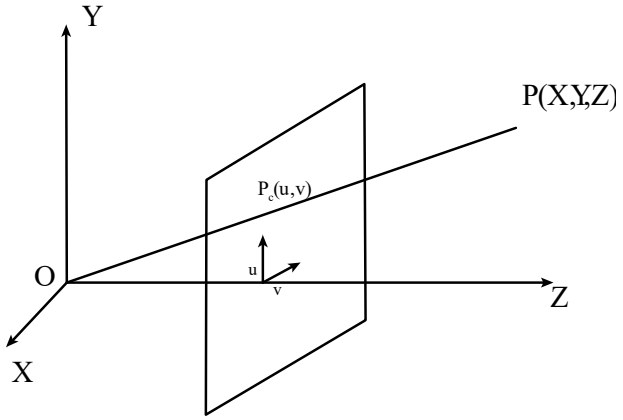


Рис. 2.3. Модель камеры обскура

Однако модель камеры-обскуры плохо применима при широком поле зрения, потому что для этого потребуется изображение бесконечного размера.

2.2.2. Сферическая модель камеры

Поскольку в алгоритме используется лазерная сканирующая система, позволяющая сканировать окружающую среду на 360 градусов, предложено

использовать камеру с объективом типа «рыбий глаз» с охватом 220° и разместить ее так, чтобы получать панорамные изображения на 360 градусов. С применением такой камеры мобильной платформе потребуется делать меньше дополнительных движений и поворотов для построения трехмерной карты окружающей среды. В этом случае модель камеры обскура не подойдет из-за широкого поля зрения камеры, поэтому в рассматриваемом алгоритме используется сферическая модель камеры.

Сферическая модель камеры основана на сферической проекции. Предположим, что есть сфера с единичным радиусом и точка P в пространстве, как показано на рис. 2.4. Точка P' является проекцией точки P на сферу, т.е. точка P' является пересечением поверхности сферы с линией, проведенной от центра сферы Oc к точке P [27]. Таким образом, определяется отображение между пространственными точками и точками на поверхности сферы. Затем эти точки вертикально проецируются на плоскость изображения, и получается круговое изображение, как на рис. 2.4. Тогда точка P' определяется следующим образом:

$$P' = \begin{bmatrix} x \\ y \\ x \end{bmatrix} = \begin{bmatrix} r * sin\theta * cos\phi \\ r * sin\theta * sin\phi \\ r * cos\theta \end{bmatrix}. \quad (2.2)$$

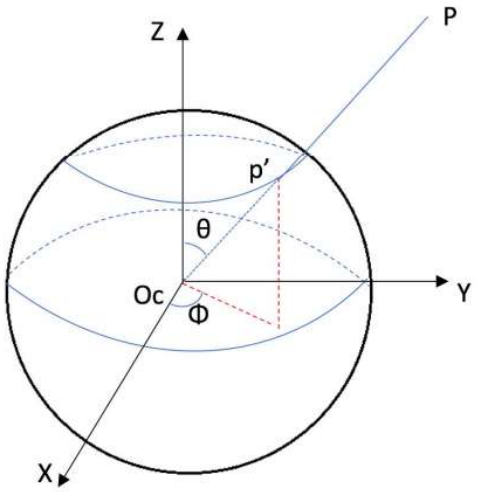


Рис. 2.4. Сферическая модель камеры

Зная координаты точки P' и карту смещения, можно определить координаты точки P [28]:

$$P = \lambda(\theta, \phi)P', \tag{2.3}$$

$$\lambda(\theta, \phi) = b * f/d(\theta, \phi), \tag{2.4}$$

где b – расстояние между точками обзора камеры, d – карта смещения, f – фокусное расстояние.

2.2.3. Калибровка камеры с использованием ее сферической модели

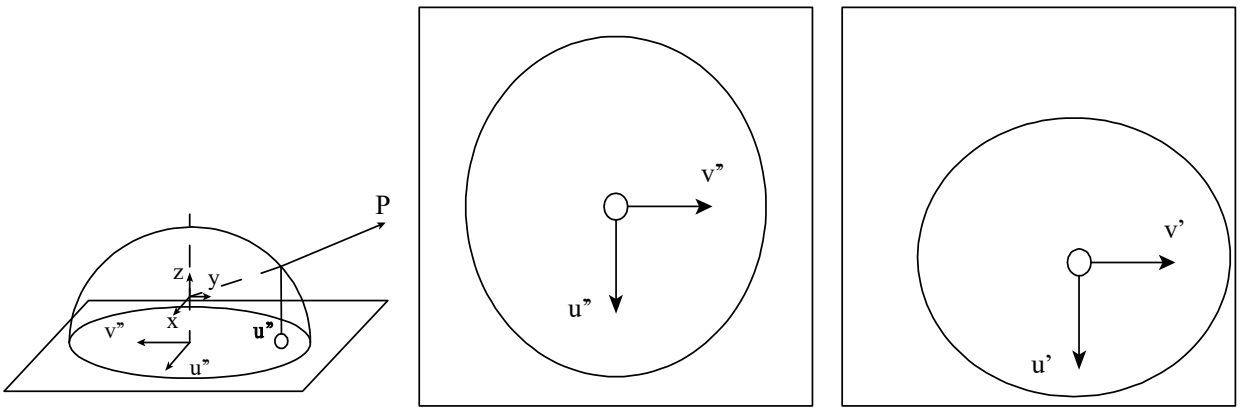
В сферической модели камеры можно выделить две плоскости: плоскость матрицы и плоскость изображения (рис. 2.5). Плоскость матрицы связана с системой отсчета ПЗС-камеры, координаты которой представлены в виде пикселей. Плоскость изображения – плоскость, которая ортогональна оптической оси объектива и находится в оптическом центре камеры. Чтобы перейти из системы, связанной с плоскостью матрицы, в систему, связанную с плоскостью изображения, необходимо применить аффинное преобразование [26]:

$$\begin{bmatrix} u'' \\ v'' \end{bmatrix} = \begin{bmatrix} a & b \\ c & 1 \end{bmatrix} \begin{bmatrix} u' \\ v' \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{2.5}$$

где u'', v'' – координаты проекции пространственной точки P на плоскость изображении, u', v' – координаты проекции пространственной точки P на плоскость матрицы, a, b, c – коэффициенты аффинного преобразования, c_x, c_y – координаты центра изображения.

Пусть функция g определяет отображение между пространственными точками и точками на плоскости матрицы. Тогда можно определить полную модель сферической камеры:

$$P = \lambda g\left(\begin{bmatrix} u'' \\ v'' \end{bmatrix}\right). \tag{2.6}$$



а) б) в)

Рис. 2.5. Координатные системы: а) модели сферической камеры; б) плоскости матрицы (в метрах); в) плоскости изображения (в пикселях)

Под калибровкой камеры понимается нахождение коэффициентов аффинного преобразования a, b, c, центра изображения c_x, c_y и нелинейной функции g, т.е. нахождение преобразования между координатами пикселя на изображении и вектора, указывающего на соответствующую точку в пространстве [29, 30].

Функцию g можно считать симметричной относительно оси вращения, потому что оптика обычно изготавливается с высокой точностью. Функция g может иметь различную форму в зависимости от оптики и используемой конструкции камеры. Но для большинства случаев можно применить полиномиальную функцию:

$$g\left(\begin{bmatrix} u'' \\ v'' \end{bmatrix}\right) = a_0 + a_1\rho + a_2\rho^2 + \cdots + a_n\rho^N, \tag{2.7}$$

где коэффициенты a_i, степень полиномиальной функции N и переменная ρ определяются во время калибровки.

После калибровки, зная полную модель сферической камеры, можно применять различные алгоритмы, такие как визуальная одометрия, трехмерная реконструкция окружающей среды, визуальный SLAM и т.д.

Калибровка камеры производилась с помощью метода Ч. Чжана [27]. На рис. 2.6 и 2.7 представлены полиномиальная функция и ошибка между перепроецированием точки P на плоскость изображения и проекцией точки

$\boldsymbol{u}'$ этой же точки P, построенной с использованием параметров камеры. Точность измерения параметров составила 0,97 пикселей, которая определяется как средняя ошибка перепроецирования.

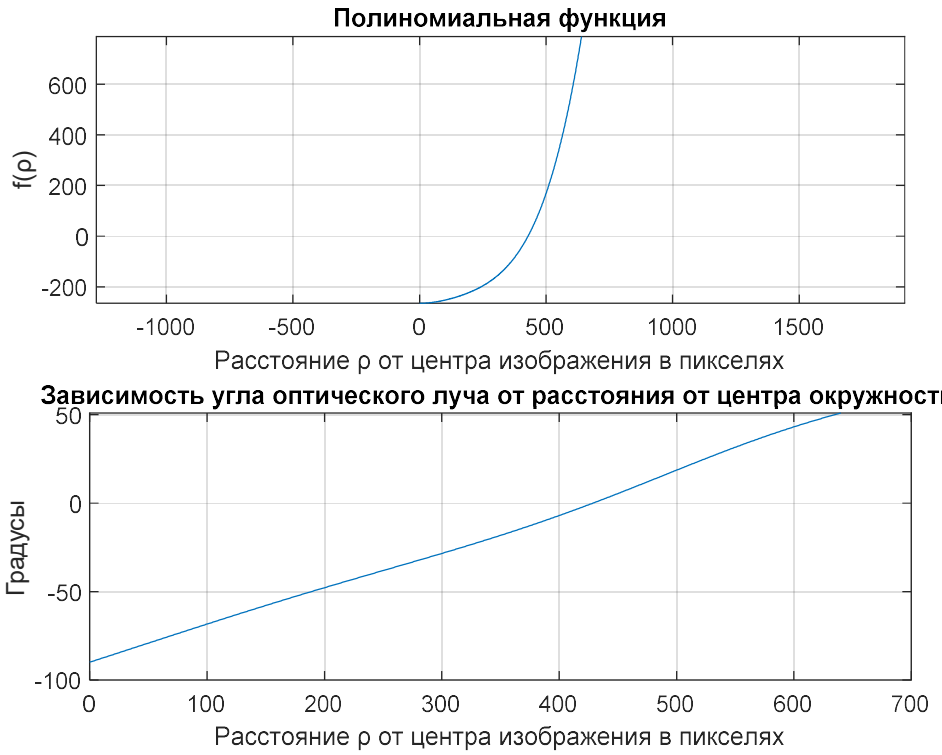
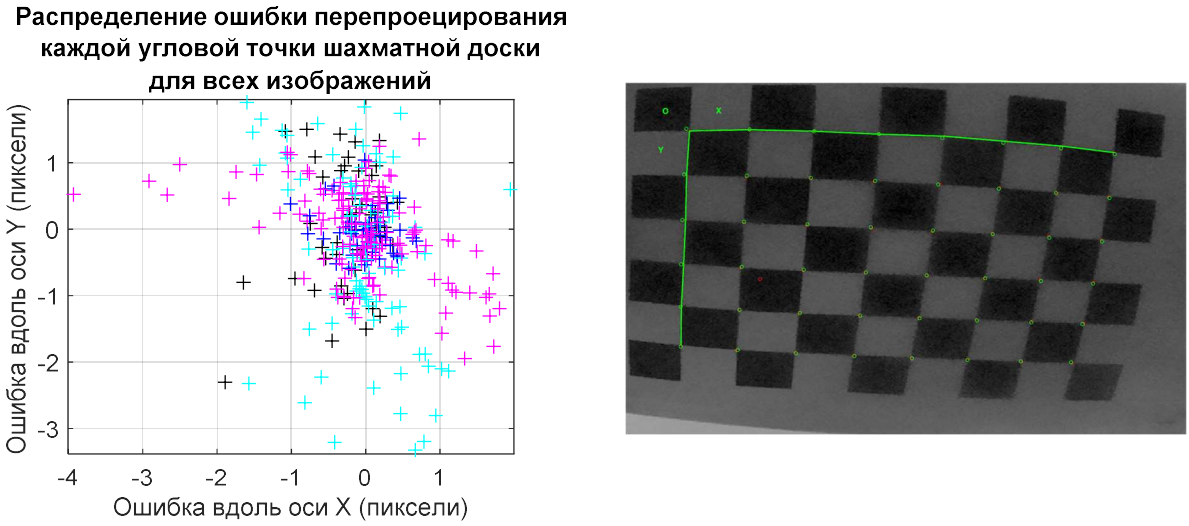


Рис. 2.6. Полученная полиномиальная функция в процессе калибровки камеры



а) б)

Рис. 2.7. а) Распределение ошибки перепроецирования; б) Перепроецирование каждой угловой точки шахматной доски

В итоге после калибровки будут два преобразования:

1) Проецирование трехмерной точки на изображение:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a & b \\ c & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} \frac{f(\rho)}{\sqrt{x^2+y^2}} + \begin{bmatrix} c_x \\ c_y \end{bmatrix}, \tag{2.8}$$

$$f(\rho) = \beta_0 + \beta_1\rho + \beta_2\rho^2 + \cdots + \beta_n\rho^N, \tag{2.9}$$

$$\rho = arctg(\frac{z}{\sqrt{x^2+y^2}}), \tag{2.10}$$

где x, y, z – координаты трехмерной точки в пространстве, u, v – координаты проекции трехмерной точки на изображении, a, b, c – коэффициенты аффинного преобразования, c_x, c_y – координаты центра изображения, $f(p)$ – полиномиальная функция, β_i – коэффициенты обратной полиномиальной функции.

2) Обратное проецирование точки на изображении на сферу с единичным радиусом:

$$\begin{bmatrix} x_p \\ y_p \end{bmatrix} = \frac{1}{\det(A)} \begin{bmatrix} 1 & -b \\ -c & a \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}, \tag{2.11}$$

$$z_p = \alpha_0 r + \alpha_1 r^1 + \cdots + \alpha_n r^n, \tag{2.12}$$

$$r = \sqrt{x_p^2 + y_p^2}, \tag{2.13}$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \frac{1}{\sqrt{x_p^2+y_p^2+z_p^2}} \begin{bmatrix} x_p \\ y_p \\ z_p \end{bmatrix}, \tag{2.14}$$

$$A = \begin{bmatrix} a & b \\ c & 1 \end{bmatrix}, \tag{2.15}$$

где x, y, z – координаты трехмерной точки на единичной окружности, вектор, проведенный из центра матрицы к этой точке, указывает на исходную точку в пространстве, u, v – координаты проекции трехмерной точки на изображении, a, b, c – коэффициенты аффинного преобразования, x_p, y_p, z_p – не нормированные координаты трехмерной точки, a_i – коэффициенты обратной полиномиальной функции, r – радиус от центра изображения до проекции трехмерной точки на изображении.

2.3. Визуальная одометрия

Визуальная одометрия – оценка движения камеры посредством снятых последовательных изображений.

Для определения перемещения с помощью алгоритма визуальной одометрии необходимо выполнить следующие действия:

1) Исправление дисторсии на изображениях. На этом этапе необязательно исправлять все изображение – достаточно исправлять найденные особые точки.
2) На предыдущем изображении ищутся особые точки. На текущем изображении находятся соответствующие им особые точки.
3) Из анализа положения особых точек на предыдущем и текущем изображении вычисляется матрица гомографии.
4) На основе дополнительной информации вычисляется масштабный коэффициент.

Рассмотрим эти шаги подробнее.

1) Исправление дисторсии. Все изображения проходят процедуру предобработки, заключающейся, главным образом, в удалении искажений с помощью параметров, полученных во время калибровки. Но при использовании широкоугольных камер не всегда выгодно исправлять дисторсию на всем изображении, потому что область интереса может находиться на краю изображении, где при исправлении дисторсии изображение сильно портится. В этом случае можно выполнить основные шаги алгоритма без исправления дисторсии при условии, что результат выполнения этих шагов сильно не зависит от дисторсии, а после, при установлении соответствия двумерных координат точки на изображении и трехмерных координат в пространстве, произвести исправление дисторсии.

2) Поиск особых точек. Под особой точкой понимается точка на изображении, которая обладает характерной окрестностью, позволяющей выделить ее среди всех других точек изображения. Особая точка имеет ряд признаков, существенно отличающих ее от множества соседних с ней точек

объекта текущего изображения. Особая точка может представлять характерные части на изображениях объекта, например, углы, вершины геометрических фигур, небольшие окружности и круги, края плоскостей, резкие цветовые переходы, перепады яркости или контрастности. Фактически, она может быть изолированной точкой локального максимума или минимума линии интенсивности или представлять собой наиболее простой элемент дискретного представления функции описания объекта.

Для работы с особыми точками, в частности для установления их соответствий, вычисляются дескрипторы – вектора признаков окрестностей точек. Основную идею сопоставления изображений с использованием особых точек можно представить следующим образом: между выделенными на сравниваемых изображениях точками с помощью их дескрипторов устанавливаются соответствия, и на основе полученных пар особых точек оценивается связывающее их преобразование. Заметим, что соответствия не всегда устанавливаются правильно. Характерные окрестности разных особых точек могут определять похожие дескрипторы, что приводит к установлению ложных соответствий точек (выбросов).

В данной работе для поиска особых точек используется метод ORB.

3) Вычисление матрицы гомографии. Матрица гомографии показывает связь между двумя изображениями плоского объекта. Имея набор точек на одном изображении и сопоставленный ему набор точек на другом, можно найти между ними соответствие в виде матрицы гомографии H. Поскольку гомография является проективным преобразованием плоскости, то ее можно рассматривать как совокупность следующих операций: параллельный перенос, поворот, масштабирование, аффинное преобразование. Обычно матрицу гомографии вычисляют с помощью методов DLT, MLE или RANSAC.

4) Вычисление масштабного коэффициента. Результатом работы алгоритма будет являться величина поворота и вектор движения. После получения вектора переноса необходимо вычислить масштабный

коэффициент. Так как масштаб окружающего мира не может быть определен при помощи одного лишь изображения – необходимы дополнительные источники данных.

2.3.1. Вычисление пройденного пути и угла курса

Для определения перемещения мобильной платформы методом визуальной одометрии разработан алгоритм, представленный на рис. 2.8. Отличие этого алгоритма от остальных подобных ему заключается в том, что исправление дисторсии происходит после сопоставления ключевых точек, а не до него. Это связанно с тем, что ключевые точки могут находиться на краю изображения, где происходит сильное искажение после исправления дисторсии (рис. 2.9).

Рис. 2.8. Структурная схема алгоритма определения перемещения методом визуальной одометрии

В случае использования визуального одометра на мобильной платформе можно считать, что она движется на плоскости ($\Delta Z = 0$, углы

крена и тангажа постоянны и боковое смещение отсутствует). Тогда матрицу гомографии можно представить в виде:

$$\boldsymbol{H} = \boldsymbol{R} + \frac{\boldsymbol{Tn^T}}{h} = \begin{bmatrix} cos\Delta\theta & -sin\Delta\theta & 0 \\ sin\Delta\theta & cos\Delta\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} + \frac{1}{h}\begin{bmatrix} \Delta t_1 \\ \Delta t_2 \\ 0 \end{bmatrix} * \begin{bmatrix} 0 \\ 0 \\ -1 \end{bmatrix}^T = \begin{bmatrix} cos\Delta\theta & -sin\Delta\theta & \frac{-\Delta t_1}{h} \\ sin\Delta\theta & cos\Delta\theta & \frac{-\Delta t_2}{h} \\ 0 & 0 & 1 \end{bmatrix}, \quad (2.16)$$

где $\Delta\theta$ угол поворота камеры относительно оси z, Δt_1 и Δt_2 – перенос камеры, h – высота, на которой камера находится.

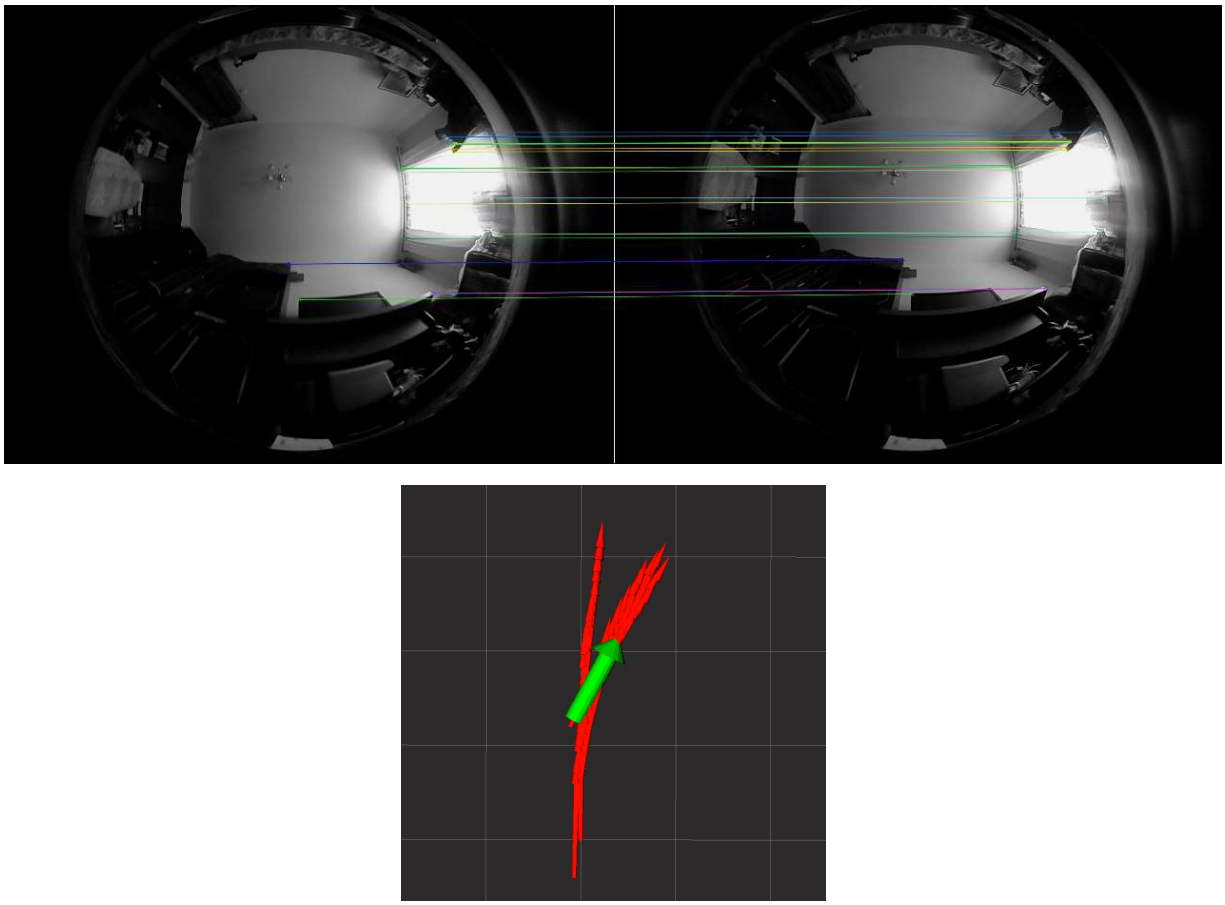


Рис. 2.9. Пример работы алгоритма визуальной одометрии: а) результат поиска пар особых точек; б) результат работы одометрии (зеленой стрелкой показано текущее месторасположение и направление мобильной платформы, красными стрелками – месторасположение в предыдущие моменты времени)

Уравнение описывает евклидово преобразование на плоскости изображения, которое является частным случаем движения камеры (матрицы

гомографии). Евклидово преобразование имеет только три степени свободы и позволяет более стабильно оценивать движение, чем в общем случае (то есть восемь степеней свободы), когда движение ограничено тем, что находится на плоскости земли. Однако из-за неизбежных вибраций, которым подвергается камера во время движения мобильной платформы, форма H может немного отличаться от вышеприведенного вида [30].

Матрица гомографии ищется с помощью метода максимального правдоподобия. Оценка максимального правдоподобия может быть получена путем минимизации следующей функции:

$$E = \sum_i (M(H * (J(x_{i1})) - x_{i2}))^2, \quad (2.17)$$

где x_{i1} – i особая точка на первом изображении, x_{i2} – i особая точка на втором изображении, M – функция проекции трехмерной точки на изображение, J – функция преобразования точки на изображении в точку, находящуюся в трехмерном пространстве.

2.4. Ориентиры

Измерение абсолютного положения предоставляет информацию о местоположении независимо от предыдущих его оценок. Это значит, что ошибка положения не растет неограниченно, как в случае с методами относительного положения. Методы для получения абсолютных измерений можно разделить на методы, основанные на использовании ориентиров или маяков, и методы, основанные на использовании карт.

Ориентир – характерный, хорошо видимый на местности неподвижный объект, с помощью которого легко ориентироваться. Показания датчиков анализируются на наличие ориентиров. Как только ориентиры обнаружены, они сопоставляются с априорно известной информацией об окружающей среде для определения положения.

Типы ориентиров:

1) Активные ориентиры.

2) Пассивные ориентиры.

3) Искусственные ориентиры.

4) Естественные ориентиры.

Требования к ориентирам:

1) Повторная наблюдаемость – ориентиры должны быть просматриваемыми с разных положений и под разными углами.

2) Уникальность – ориентиры должны быть отличимыми друг от друга.

3) Многочисленность в окружающей среде.

4) Стационарность.

Улучшение алгоритма EKF-SLAM состоит в реализации возможности подстановки подматриц соответствующих объектов. Такая реализация позволяет рассматривать сложные динамические системы, а также формировать и обрабатывать локальные карты, состоящие из значимых ориентиров.

В данной работе рассматривается система, состоящая из мобильной платформы, ориентиров, полученных по изображениям камеры, и ориентиров, полученных по данным лидара. Необходимые для вычисления матрицы формируются путем подстановки подматриц соответствующего типа объекта системы. Например, при формировании матрицы измерения сначала ставится подматрица измерения для мобильной платформы, далее ставится подматрица измерения соответствующего типа первого ориентира, потом подматрица измерения для соответствующего типа второго ориентира и т.д. Конечный результат матриц состояния и ковариации представлен на рис. 2.10.

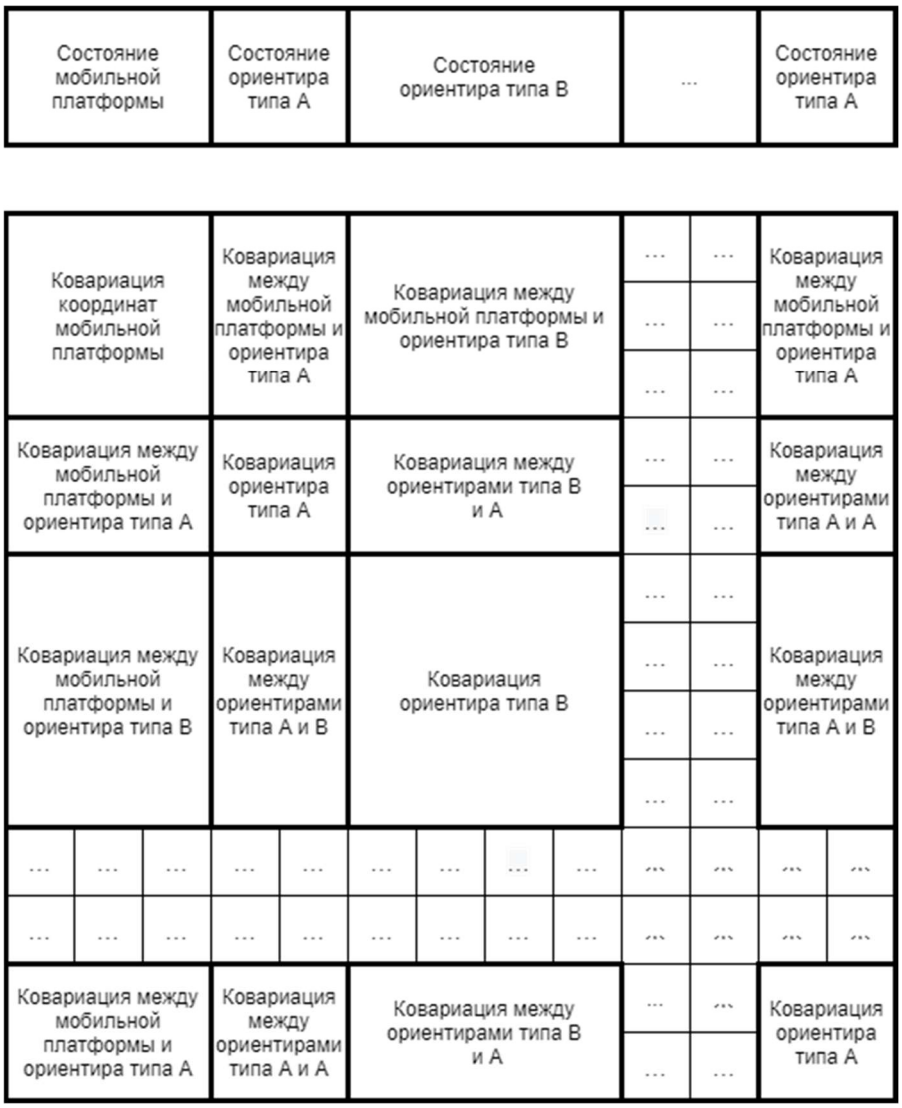


Рис. 2.10. Матрицы состояния и ковариации системы

2.4.1. Ориентир типа «особая точка»

Для инициализации координат особой точки авторы [32-34] предлагают использовать кодирование обратной глубины, где координаты особой точки в обратном кодировании глубины представляет собой 6-мерный вектор:

$$\boldsymbol{y}_i = (x_{c,i} \quad y_{c,i} \quad z_{c,i} \quad \theta_i \quad \varphi_i \quad \rho_i)^T, \tag{2.18}$$

где $x_{c,i}, y_{c,i}, z_{c,i}$ – трехмерные координаты позиции оптического центра камеры во время первого наблюдения, θ_i и φ_i – зенитный и азимутальный углы ориентира относительно системы координат камеры, ρ_i – инверсная глубина (рис. 2.11).

Трехмерные координаты вычисляются следующим образом:

$$\boldsymbol{x}_i = \begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} = \begin{pmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{pmatrix} + \frac{1}{\rho_i} m(\theta_i, \varphi_i), \tag{2.19}$$

$$m(\theta_i, \varphi_i) = (sin\theta cos\varphi \quad sin\theta sin\varphi \quad cos\theta)^T, \tag{2.20}$$

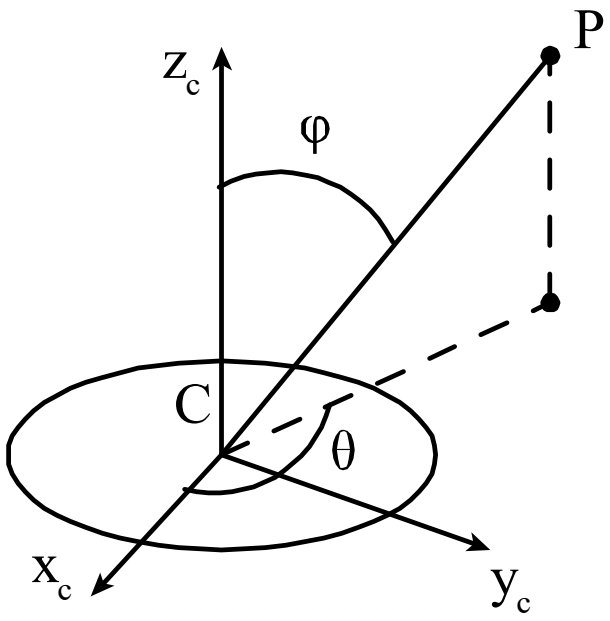


Рис. 2.11. Координатная система камеры. Оптический центр камеры обозначен как C, θ и φ – зенитный и азимутальный углы

Результатом применения функции $m(\theta_i, \varphi_i)$ является единичный вектор, направленный от оптического центра камеры на ориентир $\boldsymbol{y_i}$. Умножение этого вектора на величину глубины $d_i = \frac{1}{\rho_i}$ и добавление его к позиции камеры во время первого наблюдения $(x_{c,i} \quad y_{c,i} \quad z_{c,i})^T$ приводит к получению трехмерной позиции ориентира.

Наблюдение $\boldsymbol{h}_i^C$ имеет вид (рис. 2.12):

$$\boldsymbol{h}_i^C = \boldsymbol{h}_{x,y,z,i}^C = \boldsymbol{R}^{CW} \left(\begin{pmatrix} x_i \\ y_i \\ z_i \end{pmatrix} - \boldsymbol{r}^{WC} \right), \tag{2.21}$$

$$\boldsymbol{h}_i^C = \boldsymbol{R}^{CW} \left(\rho_i \left(\begin{pmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{pmatrix} - \begin{pmatrix} x_r - dcos\alpha \\ y_r + dsin\alpha \\ z_r \end{pmatrix} \right) + m(\theta_i, \varphi_i) \right), \tag{2.22}$$

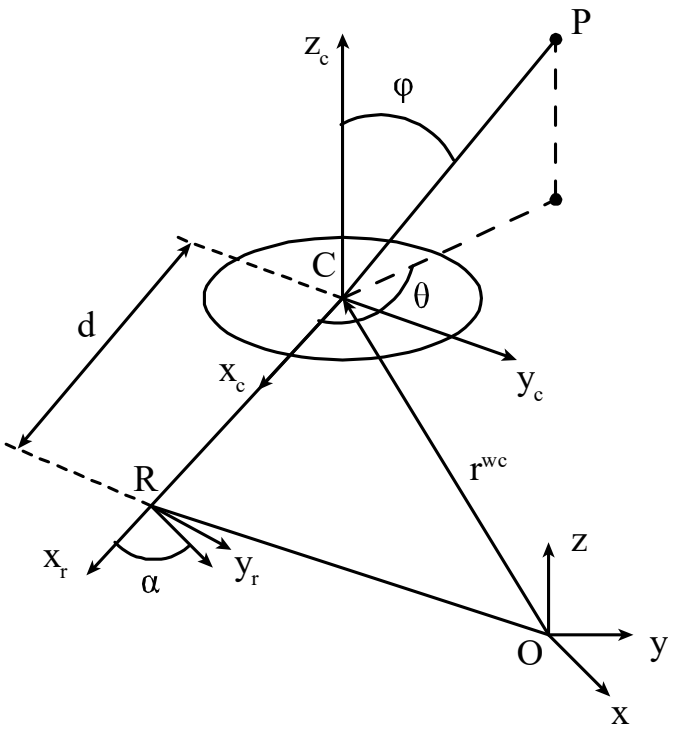


Рис. 2.12. Ориентир типа «особая точка»

где x_r, y_r и z_r – координаты центра мобильной платформы, α – угол направления мобильной платформы, d – расстояние от центра мобильной платформы до камеры.

Матрица якобиана является матрицей размера 2×9. Ее можно представить как:

$$\frac{\partial h(\boldsymbol{f_i})}{\partial \boldsymbol{f_v}} = \left(\frac{\partial h(\boldsymbol{f_i})}{\partial \boldsymbol{r}^{WC}} \quad \frac{\partial h(\boldsymbol{f_i})}{\partial \boldsymbol{q}^{WC}}\right), \tag{2.23}$$

где $\frac{\partial h(\boldsymbol{f_i})}{\partial \boldsymbol{r}^{WC}}$ имеет размерность 2×3, $\frac{\partial h(\boldsymbol{f_i})}{\partial \boldsymbol{q}^{WC}}$ имеет размерность 2×6, $\boldsymbol{f_i}$ – координаты наблюдаемого ориентира, $\boldsymbol{r}^{WC}$ – расстояние от оптического центра камеры до системы координат, $\boldsymbol{q}^{WC}$ – ориентация камеры относительно системы координат. Матрицы якобиана в этом случае будут иметь вид:

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \boldsymbol{f}_i} = \left(\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial x_r} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial y_r} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \alpha} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial x_{c,i}} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial y_{c,i}} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial z_{c,i}} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \theta_i} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \varphi_i} \quad \frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \rho_i}\right), \tag{2.24}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial x_r} = \rho_i \boldsymbol{R}^{CW} \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix}, \tag{2.25}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial y_r} = \rho_i \boldsymbol{R}^{CW} \begin{pmatrix} 0 \\ -1 \\ 0 \end{pmatrix}, \tag{2.26}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \alpha} = \rho_i \boldsymbol{R}^{CW} \begin{pmatrix} -dsin\alpha \\ -dcos\alpha \\ 0 \end{pmatrix} + \begin{pmatrix} -sin\alpha & -cos\alpha & 0 \\ cos\alpha & -sin\alpha & 0 \\ 0 & 0 & 0 \end{pmatrix} \left(\rho_i \left(\begin{pmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{pmatrix} - \begin{pmatrix} x_r - dcos\alpha \\ y_r + dsin\alpha \\ z_r \end{pmatrix} \right) + m(\theta_i, \varphi_i) \right), \tag{2.27}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial x_{c,i}} = \rho_i \boldsymbol{R}^{CW} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \tag{2.28}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial y_{c,i}} = \rho_i \boldsymbol{R}^{CW} \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \tag{2.29}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial z_{c,i}} = \rho_i \boldsymbol{R}^{CW} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \tag{2.30}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \theta_i} = \boldsymbol{R}^{CW} \begin{pmatrix} cos\theta_i cos\varphi_i \\ cos\theta_i sin\varphi_i \\ -sin\theta_i \end{pmatrix}, \tag{2.31}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \varphi_i} = \boldsymbol{R}^{CW} \begin{pmatrix} -sin\theta_i sin\varphi_i \\ sin\theta_i cos\varphi_i \\ 0 \end{pmatrix}, \tag{2.32}$$

$$\frac{\partial \boldsymbol{h}_{\rho,i}^{C}}{\partial \rho_i} = \boldsymbol{R}^{CW} \left(\begin{pmatrix} x_{c,i} \\ y_{c,i} \\ z_{c,i} \end{pmatrix} - \begin{pmatrix} x_r - dcos\alpha \\ y_r + dsin\alpha \\ z_r \end{pmatrix} \right), \tag{2.33}$$

$$\boldsymbol{R}^{CW} = \begin{pmatrix} cos\alpha & -sin\alpha & 0 \\ sin\alpha & cos\alpha & 0 \\ 0 & 0 & 1 \end{pmatrix}. \tag{2.34}$$

Приведённая матрица якобина является матрицей для трехмерных координат ориентира. Для применения в расширенном фильтре Калмана ее необходимо преобразовать к матрице для координат ориентира относительно изображения. Для этого надо спроецировать трехмерные координаты точки на изображение согласно формулам 2.8-2.10 и вычислить матрицу якобиана.

Согласно правилу нахождения производной сложной функции, нужная матрица якобиана будет иметь вид:

$$H = \frac{\partial h(\boldsymbol{f_i})}{\partial(u_i,v_i)} * \frac{\partial \boldsymbol{h}^C_{\rho,i}}{\partial \boldsymbol{f}_i}, \tag{2.35}$$

где $\frac{\partial h(\boldsymbol{f_i})}{\partial(u_i,v_i)}$ – производные координаты ориентира на изображении относительно трехмерных координат ориентира:

$$\frac{\partial h(\boldsymbol{f_i})}{\partial(u_i,v_i)} = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \end{pmatrix}, \tag{2.36}$$

$$\frac{\partial u}{\partial x} = \left(\frac{a}{\sqrt{x^2+y^2}} - \frac{x(ax+by)}{(x^2+y^2)^{\frac{3}{2}}}\right) * \sum_{n=0}^{N} \beta_n p^n - \frac{ax+by}{\sqrt{x^2+y^2}} \sum_{n=1}^{N} \frac{nxz\beta_n p^{n-1}}{(1+\frac{z^2}{x^2+y^2})(x^2+y^2)^{\frac{3}{2}}}, \tag{2.37}$$

$$\frac{\partial u}{\partial y} = \left(\frac{b}{\sqrt{x^2+y^2}} - \frac{y(ax+by)}{(x^2+y^2)^{\frac{3}{2}}}\right) * \sum_{n=0}^{N} \beta_n p^n - \frac{ax+by}{\sqrt{x^2+y^2}} \sum_{n=1}^{N} \frac{nyz\beta_n p^{n-1}}{(1+\frac{z^2}{x^2+y^2})(x^2+y^2)^{\frac{3}{2}}}, \tag{2.38}$$

$$\frac{\partial u}{\partial z} = \frac{ax+by}{\sqrt{x^2+y^2}} \sum_{n=1}^{N} \frac{n\beta_n p^{n-1}}{(1+\frac{z^2}{x^2+y^2})(x^2+y^2)^{\frac{3}{2}}}, \tag{2.39}$$

$$\frac{\partial v}{\partial x} = \left(\frac{c}{\sqrt{x^2+y^2}} - \frac{x(cx+y)}{(x^2+y^2)^{\frac{3}{2}}}\right) * \sum_{n=0}^{N} \beta_n p^n - \frac{cx+y}{\sqrt{x^2+y^2}} \sum_{n=1}^{N} \frac{nxz\beta_n p^{n-1}}{(1+\frac{z^2}{x^2+y^2})(x^2+y^2)^{\frac{3}{2}}}, \tag{2.40}$$

$$\frac{\partial v}{\partial y} = \left(\frac{1}{\sqrt{x^2+y^2}} - \frac{y(cx+y)}{(x^2+y^2)^{\frac{3}{2}}}\right) * \sum_{n=0}^{N} \beta_n p^n - \frac{cx+y}{\sqrt{x^2+y^2}} \sum_{n=1}^{N} \frac{nyz\beta_n p^{n-1}}{(1+\frac{z^2}{x^2+y^2})(x^2+y^2)^{\frac{3}{2}}}, \tag{2.41}$$

$$\frac{\partial v}{\partial z} = \frac{cx+y}{\sqrt{x^2+y^2}} \sum_{n=1}^{N} \frac{n\beta_n p^{n-1}}{(1+\frac{z^2}{x^2+y^2})(x^2+y^2)^{\frac{3}{2}}}. \tag{2.42}$$

2.4.2. Ориентир типа «угол»

Рассмотрим ориентир типа «угол» (рис. 2.13). Уравнение измерения для таких ориентиров выглядит следующим образом:

$$z_t = \begin{bmatrix} r \\ \alpha \end{bmatrix} = \begin{bmatrix} \sqrt{(x_m - x_l)^2 + (y_m - y_l)^2} \\ atan\frac{y_m - y_l}{x_m - x_l} - \theta \end{bmatrix} + Q, \quad (2.43)$$

где z_t – текущее измерение $x_l = x + dcos\theta$, $y_l = y + dsin\theta$, x_m, y_m – координаты ориентира, x_l, y_l – координаты лидара, x, y, θ – координаты мобильной платформы, r – расстояние от лидара до ориентира, α – угол между лидаром и ориентиром, Q– некоррелированный белый гауссовский процесс с нулевой средней и постоянной ковариацией:

$$Q = \begin{bmatrix} \sigma_r^2 & 0 \\ 0 & \sigma_\alpha^2 \end{bmatrix}. \quad (2.44)$$

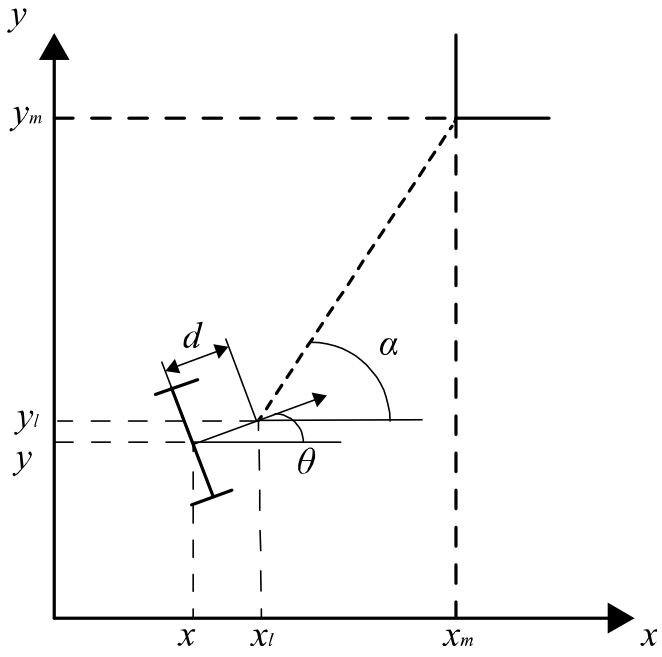


Рис. 2.13. Ориентир типа «угол»

Матрица якобиана измерения для ориентира типа «угол» имеет вид:

$$H_t = \begin{bmatrix} \frac{dr}{dx} & \frac{d\alpha}{dx} \\ \frac{dr}{dy} & \frac{d\alpha}{dy} \\ \frac{dr}{d\theta} & \frac{d\alpha}{d\theta} \\ \frac{dr}{dx_m} & \frac{d\alpha}{dx_m} \\ \frac{dr}{dy_m} & \frac{d\alpha}{dy_m} \end{bmatrix} =$$

$$\begin{bmatrix} -\frac{\Delta x}{\sqrt{q}} & \frac{\Delta y}{q} \\ -\frac{\Delta y}{\sqrt{q}} & -\frac{\Delta x}{q} \\ \frac{d}{\sqrt{q}}(\Delta x sin\theta - \Delta y cos\theta) & -\frac{d}{\sqrt{q}}(\Delta x sin\theta - \Delta y cos\theta) - 1 \\ \frac{\Delta x}{\sqrt{q}} & -\frac{\Delta y}{q} \\ \frac{\Delta y}{\sqrt{q}} & \frac{\Delta x}{q} \end{bmatrix}, \quad (2.45)$$

где $\Delta x = x_m - x_l;\ \Delta y = y_m - y_l; q = (x_m - x_l)^2 + (y_m - y_l)^2$.

2.4.3. Трекинг ориентиров

Для отслеживания каждого ориентира, найденного на изображении, используются алгоритм Лукаса-Канаде (рис. 2.14). Алгоритм Лукаса-Канаде широко используется в компьютерном зрении для вычисления оценки оптического потока. Оптический поток – это изображение видимого движения объектов, поверхностей или краев сцены, получаемое в результате перемещения камеры относительно сцены. Алгоритм решает основные уравнения оптического потока для всех пикселей в их окрестностях по методу наименьших квадратов. В этом методе предполагается, что оптический поток должен быть одинаков для всех пикселей в локальной окрестности рассматриваемого пикселя p [59]. Алгоритм Лукаса-Канаде предполагает, что смещение содержимого изображения между двумя последовательными кадрами небольшое и приблизительно постоянное в окрестностях точки p:

$$\begin{cases} I_x(p_1)V_x + I_y(p_1)V_y = -I_t(p_1) \\ I_x(p_2)V_x + I_y(p_2)V_y = -I_t(p_2) \\ \dots \\ I_x(p_n)V_x + I_y(p_n)V_y = -I_t(p_n) \end{cases}, \tag{2.46}$$

где $p_1, p_2, \dots, p_n$ – пиксели внутри окна, $I_x(p_n), I_y(p_n), I_t(p_n)$ – частные производные изображения I по координатам x, y и времени t, вычисленные в точке p_n, V_x, V_y – вектор оптического потока.

Это выражение можно переписать в матричной форме $Av = b$, где:

$$A = \begin{bmatrix} I_x(p_1) & I_y(p_2) \\ I_y(p_2) & I_y(p_2) \\ \vdots & \vdots \\ I_x(p_n) & I_y(p_n) \end{bmatrix}.$$

$$v = \begin{bmatrix} V_x \\ V_y \end{bmatrix}.$$

$$b = \begin{bmatrix} -I_t(p_1) \\ -I_t(p_2) \\ \vdots \\ -I_x(p_n) \end{bmatrix},$$

Полученную систему уравнений решают с помощью взвешенного метода наименьших квадратов:

$$V = (A^T W A)^{-1} A^T W b, \tag{2.47}$$

или

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i w_i I_x(p_n)^2 & \sum_i w_i I_x(p_n) I_y(p_n) \\ \sum_i w_i I_x(p_n) I_y(p_n) & \sum_i w_i I_y(p_n)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i w_i I_x(p_n) I_t(p_n) \\ -\sum_i w_i I_y(p_n) I_t(p_n) \end{bmatrix} \tag{2.48}$$

где w_i – весы, значение которых определяется с помощью гауссовской функции с учетом расстояния между p_i и p.

Рис. 2.14. Пример работы трекинга Лукаса-Канаде (зеленым отображены отслеживаемые точки, красным – их траектории)

Впоследствии появились модификации данного алгоритма Томаши-Канаде (Tomasi-Kanade) и Ши-Томаши-Канаде (Shi-Tomasi-Kanade). Трекер Ши-Томаши-Канаде учитывает аффинные искажения окрестных точек.

2.5. Детекторы ориентиров

При детектировании ориентиров по данным лазерной сканирующей системы возникают две основные проблемы (рис. 2.15):

1) Недостающие данные. Полная форма окружающей среды не является видимой для лазерной сканирующей системы, что может привести к ложным обнаружениям ориентиров на границе видимости.

2) Окклюзия. Объект переднего плана может перекрывать часть фонового объекта. Из-за этого передний объект может слиться с фоновым объектом, и при этом образуются резкие границы [36].

Для нахождения ориентиров предложено использовать контурный анализ. Идея подхода заключается в отказе обработки каждой точки, полученной от лазерного дальномера, и переходе к обработке лишь контура. Контур представляется в виде комплекснозначного сигнала – такая модель дает в большей степени использовать методы, инвариантные к переносу, повороту и изменению масштаба.

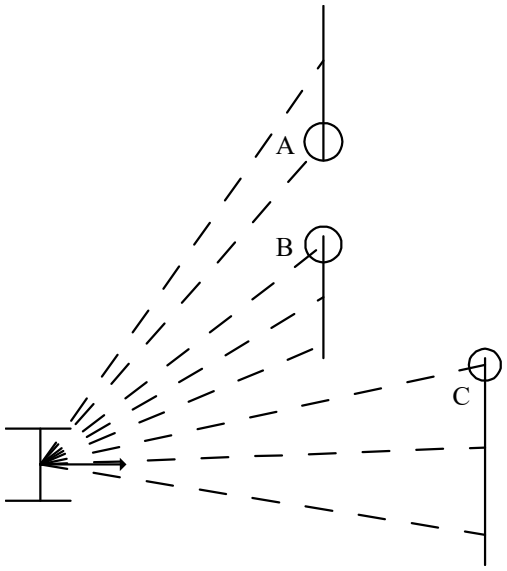


Рис. 2.15. Сценарий детектирования ориентиров

(Направление, с которого просматривается поверхность, является важным для детектирования ориентиров. В случае А должен детектироваться ориентир, в случае В ориентир не может быть четко определенным. В случае С, тень от объекта переднего плана может вызвать появление ложной границы на фоновом объекте)

Перед использованием детектора ориентиров данные лазерной сканирующей системы подвергаются следующим преобразованиям:

1) Представление данных лазерной сканирующей системы в полярный код:

$$C(n) = r(n) * \cos\big(\alpha(n)\big) + i * r(n) * \sin\left(\alpha(n)\right), \quad (2.49)$$

где $r(n)$ – расстояние от лазерной сканирующей системы до объека, $\alpha(n)$ – текущий угол сканирования.

2) Представление полярного кода в разностный код:

$$C(n) = \gamma(n+1) - \gamma(n). \quad (2.50)$$

3) Деление полученного контура на фрагменты по заданному порогу.

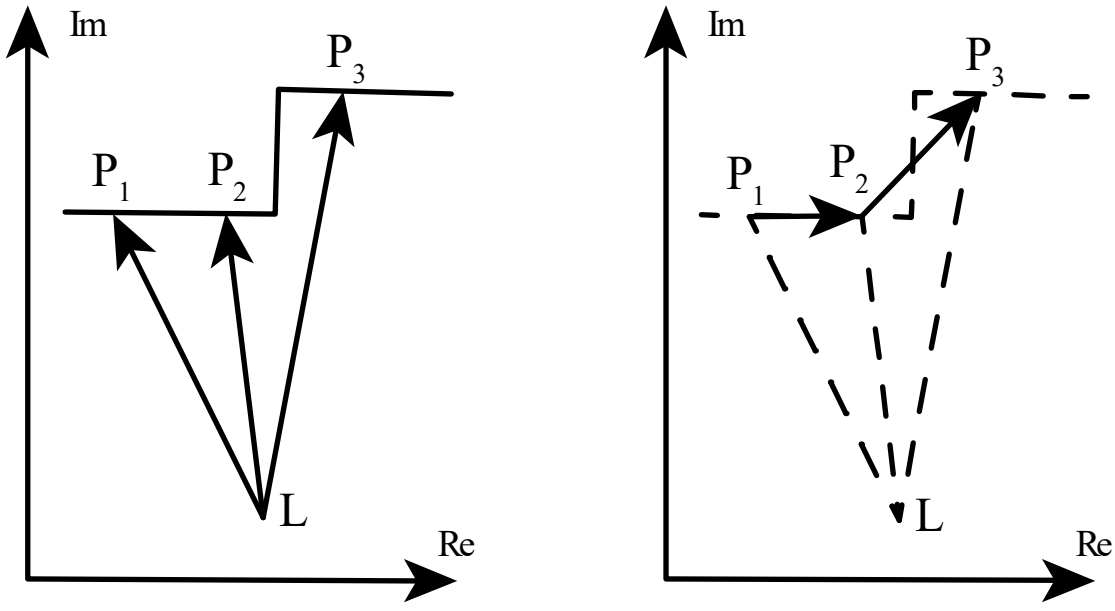
4) Эквализация разностного кода.

В течение первых двух шагов происходит формирование контура видимой части окружающей среды. Третий шаг необходим для решения проблемы окклюзии. На четвертом шаге происходит обработка контурного кода, необходимая для применения методов, инвариантных к переносу, повороту и изменению масштаба [37].

2.5.1 Эквализация разностного кода

Эквализацией кода контура называется процедура, при которой контур делится на заданное количество одинаковых по длине элементарных векторов.

После представления данных лидара в полярный код, а из него в разностный код, полученный контур будет иметь элементарные вектора разного размера (рис. 2.16), потому что расстояние между точками зависит от места наблюдения (месторасположения лидара) в окружающей среде. С помощью такого представления невозможно применять многие алгоритмы контурного анализа, потому что они требуют, чтобы элементарные вектора имели одинаковую длину.



а) б)

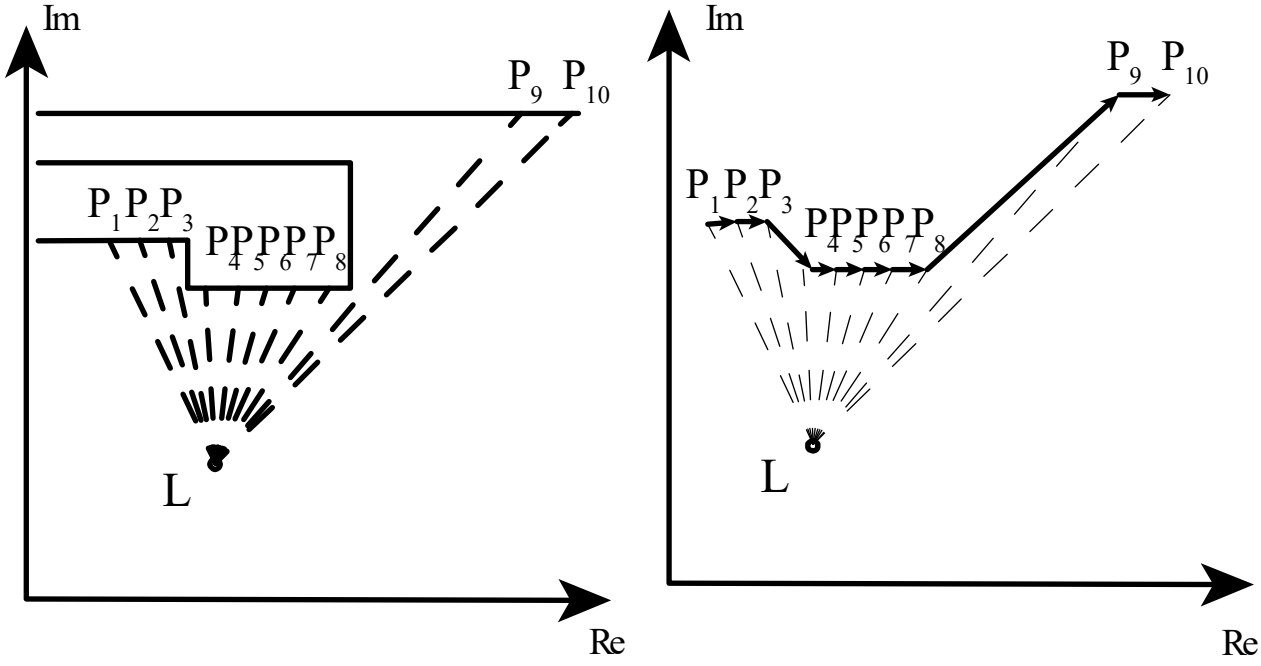
Рис. 2.16. Примеры представления данных лидара в различные коды: а) полярный код; б) разностный код

(Точка L представляет собой месторасположения лидара, точки P_i обозначают точки попадания в препятствия, набор векторов LP_i обозначает полярный код, а набор векторов P_iP_{i+1} – разностный код)

Перед тем как провести операцию эквализации исходного контура Г, необходимо решить две проблемы: недостающие данные и окклюзия. Для решения этих проблем предложено делить весь контур на сегменты, затем каждый сегмент подвергать операции эквализации и применять детекторы ориентиров, которые рассматривают окрестность рассматриваемой точки.

Было замечено, что из-за этих проблем возникают относительно длинные вектора (рис 2.17). В тех местах контура Г, где вектора имеют относительно маленькую длину, можно считать, что эти участки контура имеют достоверную геометрическую информацию об окружающей среде. И наоборот, если вектора имеют относительно большую длину, то, скорее всего, в этих местах наблюдаются пропуск информации и эти участки лучше не рассматривать. На рис. 2.17 (а) представлен лидар и окружающая среда, а на рисунке 2.17 (б) представлен полученный разностный код. Вектор P_8P_9 имеет очень большую длину, что связанно с проблемой недостающих данных, кроме того, по векторам P_7P_8 и P_9P_{10} нельзя четко детектировать ориентир. Поэтому контур Г делится на сегменты, путем выбрасывания длинных векторов. Начало сегмента контура будет являться концом одного из длинных векторов, а конец сегмента будет являться началом следующего такого длинного вектора. Опытным путем определено, что лучше откидывать все вектора, которые имеют длину больше 0.5 м. Кроме того, после преобразования данных лидара в разностный код, началом контура Г будет являться первая точка, полученная лидаром. Начало контура Г необходимо совместить с концом одного из таких длинных векторов, иначе начало

контура Г будет являться точкой разрыва сегмента, что повлечет собой деление этого сегмента на две части.



а) б)

Рис. 2.17. Пример представления данных лидара в разностный код при возникновении проблем, связанных с недостающими данными и окклюзией: а) лидар и окружающая среда; б) полученный разностный код

(Точка L представляет собой месторасположения лидара, точки P_i обозначают точки попадания в препятствия. Набор векторов P_iP_{i+1} обозначают разностный код)

Как было ранее сказано, у полученных сегментов контура в районе их начала и конца нельзя четко детектировать ориентир. Для решения этой проблемы предложено не детектировать ориентиры на этих областях. Поскольку алгоритмы детектирования ориентиров имеют окно, то достаточно не применять подходы для решения проблемы граничных условий. В этом случае отступ от концов сегментов контура будет равна половине размера окна. Если алгоритм сам ищет размер окна, то в этом случае рассматриваем только тот размер окна, который больше заданного. Таким образом решаются проблемы недостающих данных и окклюзии, потому что детекторы будут рассматривать только те участки контура, где

наблюдается достоверная геометрическая информация об окружающей среде.

Проблемы недостающих данных и окклюзии можно решить не только с помощью этого способа. Из начальных условий окружающая среда является стационарной, а, следовательно, участки контура, которые соответствуют достоверной геометрической информации об окружающей среде, с течением времени не должны меняться. Участки контура, возникающие из-за проблемы недостающих данных, меняются с течением времени при изменении точки месторасположения лидара. Учитывая это, можно применить взаимокорреляционную функцию со взвешенным окном для текущего контура и предыдущего. И рассматривать только те участки контуров, у которых оценка степени корреляции является высокой. Взвешенное окно необходимо для того, чтобы рассматривать не контуры целиком, а только локальные участки.

Алгоритм эквализации состоит из 4 шагов:

1) Определение длины элементарных векторов конечного контура $Г_{экв}$.

2) Определение остатка вектора исходного контура Г, входящего в элементарный вектор.

3) Определение количества полных векторов исходного контура Г, входящих в элементарный вектор.

4) Определение не использованной части вектора исходного контура Г.

Определение длины элементарных векторов конечного контура $Г_{экв}$. Длина элементарных векторов конечного контура $Г_{экв}$ не должна быть очень маленькой или очень большой. При маленькой длине исходный контур Г поделится на большое количество элементарных векторов, что приведет к увеличению потребности к вычислительным ресурсам. Количество элементарных векторов в этом случае будет избыточным. При большой длине элементарных векторов контура $Г_{экв}$ теряется исходная информация, как в случае дискретизации сигнала, когда не выполняется теорема Котельникова. Поэтому выбор длины элементарных векторов является

важной задачей. Если брать фиксированное количество элементарных векторов, то из-за разных точек обзора длина всего исходного контура будет меняться, а, следовательно, будет меняться длина элементарных векторов. Она может быть слишком маленькой, поэтому конечный контур в большей степени будет содержать больше шума и меньше геометрической информации об окружающей среде. Или длина может быть очень большой, в этом случае может произойти потеря геометрической информации. Кроме того, некоторые алгоритмы рассчитаны на определенную длину элементарных векторов. Поэтому вариант с фиксированным числом элементарных векторов конечного контура $\Gamma_{экв}$ не подходит. Из-за этого предлагается алгоритм с адаптивным поиском количества элементарных векторов. Алгоритм поиска количества элементарных векторов не зависит от количества точек, полученных лидаром. Алгоритм учитывает длину исходного контура и вычисляет длину элементарных векторов, близкую к заданной длине. Заданная длина должна быть выбрана такой, чтобы не было больших потребностей к вычислительным ресурсам, и достаточной, чтобы содержать геометрическую информацию. На основании этого достаточно выбрать значение равным нескольким сантиметрам, в данной работе опытным путем определено значение, равное 5 см.

Количество элементарных векторов определяется как:

$$m = round\left(\frac{1}{d}\sum_{n=0}^{k-1}|\gamma(n)|\right), \tag{2.51}$$

где m – количество элементарных векторов контура $\Gamma_{экв}$, d –фиксируемая длина, $\gamma(n)$ – вектора исходного контура, k – количество векторов исходного контура.

Длина элементарных векторов в этом случае определяется как:

$$l = \frac{1}{m}\sum_{n=0}^{k-1}|\gamma(n)|. \tag{2.52}$$

Новый эквализованный сегмент состоит из множества элементарных векторов $\varepsilon(r)$. Каждый элементарный вектор $\varepsilon(r)$ может состоять из остатка

вектора $\gamma(n)$ или полных элементарных векторов $\gamma(n+1) \ldots \gamma(n+t-1)$ или использованной части $\gamma(n+t)$ (рис. 2.18) [37].

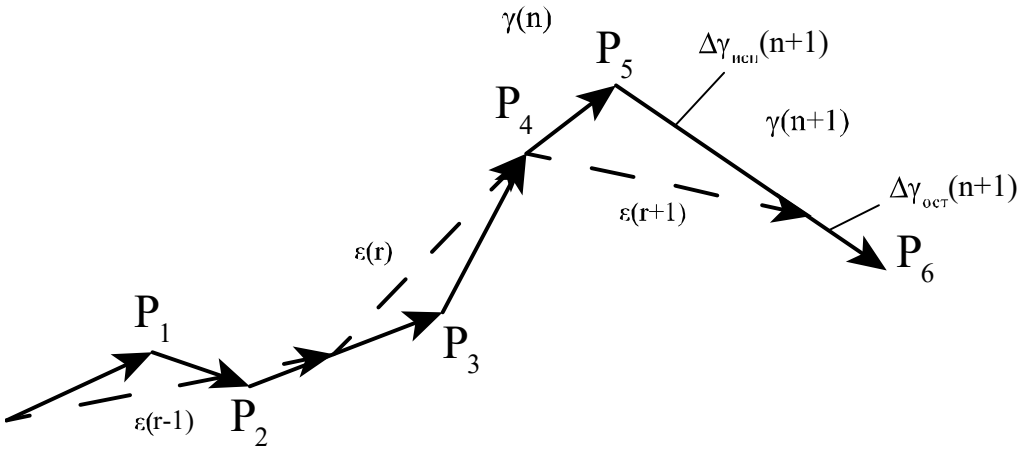


Рис. 2.18. Эквализация сегментов исходного контура Г

Определение остатка вектора исходного контура Г*, входящего в элементарный вектор.* На каждом -м шаге эквализации вначале проверяется условие

$$\left|\Delta\gamma_{\text{ост}}^{(r)}(n)\right| \geq \varepsilon. \tag{2.53}$$

Если это условие выполняется, то остаток меньше чем длина элементарного вектора $\varepsilon(r)$, и в этом случае определяется остаток $\Delta\gamma_{\text{ост}}(n-1)$ для следующего, $(r+1)$-го шага эквализации:

$$\Delta\gamma_{\text{ост}}^{(r+1)}(n) = \Delta\gamma_{\text{ост}}^{(r)}(n) - |\varepsilon|\frac{\Delta\gamma_{\text{ост}}^{(r)}(n)}{\left|\Delta\gamma_{\text{ост}}^{(r)}(n)\right|}. \tag{2.54}$$

Определение количества полных векторов исходного контура Г*, входящих в элементарный вектор.* Если же условие (2.53) не выполняется, то проверяется значение t, при котором начинает выполняться условие

$$\left|\Delta\gamma_{\text{ост}}^{(r)}(n)\right| + \sum_{j=1}^{t}|\gamma(n+j)| \geq \varepsilon, t = 1,2,\ldots \tag{2.55}$$

Определив значение t, можно определить модуль используемого вектора $\Delta\gamma_{\text{исп}}(n+t)$:

$$|\Delta\gamma_{\text{исп}}(n+t)| = |\varepsilon| - \left|\Delta\gamma_{\text{ост}}^{(r)}(n)\right| - \sum_{j=1}^{t-1}|\gamma(n+j)|. \quad (2.56)$$

Зная модуль вектора $\Delta\gamma_{\text{исп}}(n+t)$ можно определить и сам вектор.

$$\Delta\gamma_{\text{исп}}(n+t) = \gamma(n+t)\frac{|\Delta\gamma_{\text{ост}}(n+t)|}{|\gamma(n+t)|}. \quad (2.57)$$

Определение не использованной части вектора исходного контура Г. Остаточный вектор $\Delta\gamma_{\text{ост}}^{(r+1)}(n+t)$ на $(r+1)$ шаг эквализации будет иметь вид:

$$\Delta\gamma_{\text{ост}}^{(r+1)}(n+t) = \gamma(n+t) - \Delta\gamma_{\text{исп}}(n+t). \quad (2.58)$$

Тогда элементарный вектор $\varepsilon(r)$ для -го шага эквализации будет иметь вид:

$$\varepsilon(r) = \Delta\gamma_{\text{ост}}^{(r)}(n) + \Delta\gamma_{\text{исп}}(n+t) + \sum_{j=1}^{t-1}\gamma(n+j). \quad (2.59)$$

2.5.2. Поиск ориентиров типа «угол» с помощью согласованной фильтрации

Для обнаружения ключевых особенностей на контуре и формирования множества таких особенностей можно использовать фильтры, согласованные с классом форм.

Согласованные фильтры обеспечивают образование количественной меры схожести между фильтруемым контуром и эталонной формой. Здесь применяется фильтр, согласованный с эталонной формой типа «угол». Фильтр выделяет фрагмент, который состоит из двух прямолинейных отрезков, составляющих угол $\Delta\varphi$. Для этого второй прямолинейный отрезок подвергается повороту на угол $\pi - \Delta\varphi$, т. е. каждый элементарный вектор этого отрезка умножается на $-e^{-i\Delta\varphi}$. В результате обе стороны угла образуют один прямолинейный отрезок. Когда окно фильтра равно $2s$, то результат фильтрации имеет вид:

$$|\eta_n(m)| = \frac{1}{2\sqrt{s}}\left|\frac{\sum_{n=0}^{s-1}\nu(n+m)}{\sqrt{\sum_{n=0}^{s-1}|\nu(n+m)|^2}} - \frac{\sum_{n=s}^{2s-1}\nu(n+m)}{\sqrt{\sum_{n=s}^{2s-1}|\nu(n+m)|^2}}\right|, \quad (2.60)$$

где $|\eta_{\text{т}}(m)|$ – модуль выходного нормированного сигнала фильтра, s – квадрат нормы эталонного фрагмента, $\nu(n)$ – элементарные вектора контура.

После фильтрации в каждом полученном фрагменте выделяется локальный максимум модуля выходного нормированного сигнала фильтра. Каждый такой локальный максимум считается особой точкой [37].

2.5.3. Алгоритм Teh-Chin

Также можно использовать подходы, разработанные для обнаружения доминирующих точек. Доминирующие точки являются точками, обладающими высокой кривизной. Доминирующие точки используются в задачах сжатия данных и при выделении особенностей контура объекта.

Обычно в методах обнаружения доминирующих точек выделяют два основных этапа:

1) Оценка кривизны.
2) Поиск локальных максимумов.

Для оценки кривизны важно определить область поддержки. Областью поддержки для точки кривой p_i, размера k называется последовательность $D(p_i) = (p_{i-k}, \dots, p_{i-1}, p_i, \dots, p_{i+k})$ (рис. 2.19). Если длина области поддержки велика, то пропускаются некоторые доминирующие точки, если область мала, то появляются избыточные точки. Необходимо для каждой точки определять область поддержки, т. к. каждая точка имеет свои локальные геометрические особенности. Одним из адаптивных методов поиска области поддержки является метод Teh-Chin. В этом методе используется отношение перпендикуляра d_{ik} и длины хорды l_{ik}:

$$r_{ik} = \frac{d_{ik}}{l_{ik}}. \tag{2.61}$$

В качестве оценки кривизны используется мера k – косинус угла

$$cos_{ik} = \frac{\overrightarrow{a_{ık}} * \overrightarrow{b_{ık}}}{|\overrightarrow{a_{ık}}||\overrightarrow{b_{ık}}|} \tag{2.62}$$

где $\overrightarrow{a_{ık}} = (x_{i-k} - x_i, y_{i-k} - y_i)$, $\overrightarrow{b_{ık}} = (x_{i+k} - x_i, y_{i+k} - y_i)$.

Точки с локальным максимальным значением кривизны рассматриваются как доминирующие точки.

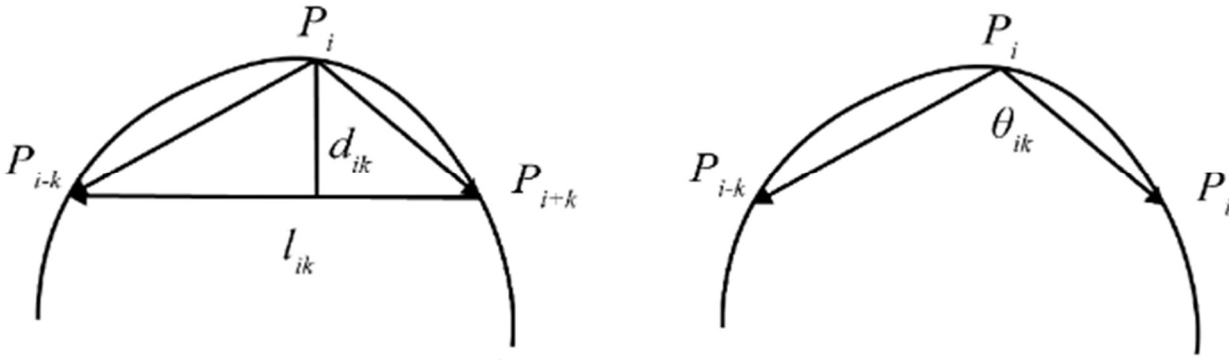


Рис. 2.19. Область поддержки и угол в точке p_i

Ученые Teh и Chin использовали отношения перпендикуляра и длины хорды для определения области поддержки для каждой точки. Область поддержки ищется путем итеративного увеличения, начиная с $k = 1$. При этом k увеличивается до тех пор, пока выполняется одно из условий $l_{i,k} \geq l_{i,k+1}$, тогда k определяет длину области поддержки в точке p_i:$r_{i,k} \geq r_{i,k+1}$ для $d_{i,k} > 0$, $r_{i,k} \leq r_{i,k+1}$ для $d_{i,k} < 0$.

В качестве значения кривизны используется непосредственно величина k – косинуса угла [36].

2.5.4. Алгоритм WU

Метод Teh-Chin используют первый локальный максимум значения кривизны для определения области поддержки. Однако такой локальный максимум может быть вызван наличием шума на кривой, т. е. алгоритм Teh-Chin не является надежным в присутствии шума. Чтобы решить данную проблему Wu предложил выбирать область поддержки такую, чтобы она имела глобальное максимальное значение кривизны, т. е. определять область поддержки между K_{min} и K_{max}. Пусть k_i – лучшая длина области поддержки в точке p_i. Тогда k_i можно определить следующим образом:

$k_i = k$, если $cos_{ik} = \max\{cos_{ij} | j = K_{min}, \dots, K_{max}\}$, для $i = 1, 2, 3, \dots, n$.

Тогда областью поддержки является следующая область:

$$D_i = \{p_{i-k_i}, \dots, p_{i+k_i}\}, \tag{2.63}$$

а значение кривизны в точке p_i определяется как среднее значение k – косинус угла [37]

$$cv_i = \frac{1}{k} \sum_{j=1}^{k_i} cos_{ij}. \tag{2.64}$$

2.5.5. Метод масштабного пространства кривизны

Доминантные точки можно также искать, используя теорию масштабного пространства кривизны. Пусть C является кривой, $C = (x(s), y(s))$, где $0 \le s \le L$, L – длина кривой. Функцию кривизны можно представить, как:

$$K_{(x,y)} = \frac{\frac{\partial^2 y}{\partial^2 x}}{\sqrt{[1+\left(\frac{\partial y}{\partial x}\right)^2]^3}}. \tag{2.65}$$

Если обозначить

$$\dot{x} = \frac{\partial x}{\partial s}\, \dot{y} = \frac{\partial y}{\partial s}\, \ddot{x} = \frac{\partial^2 x}{\partial s^2}\, \ddot{y} = \frac{\partial^2 y}{\partial s^2},$$

затем

$$\frac{\partial y}{\partial x} = \frac{\dot{y}}{\dot{x}} \quad \frac{\partial^2 y}{\partial x^2} = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{\dot{x}^3},$$

получим следующую функцию кривизны:

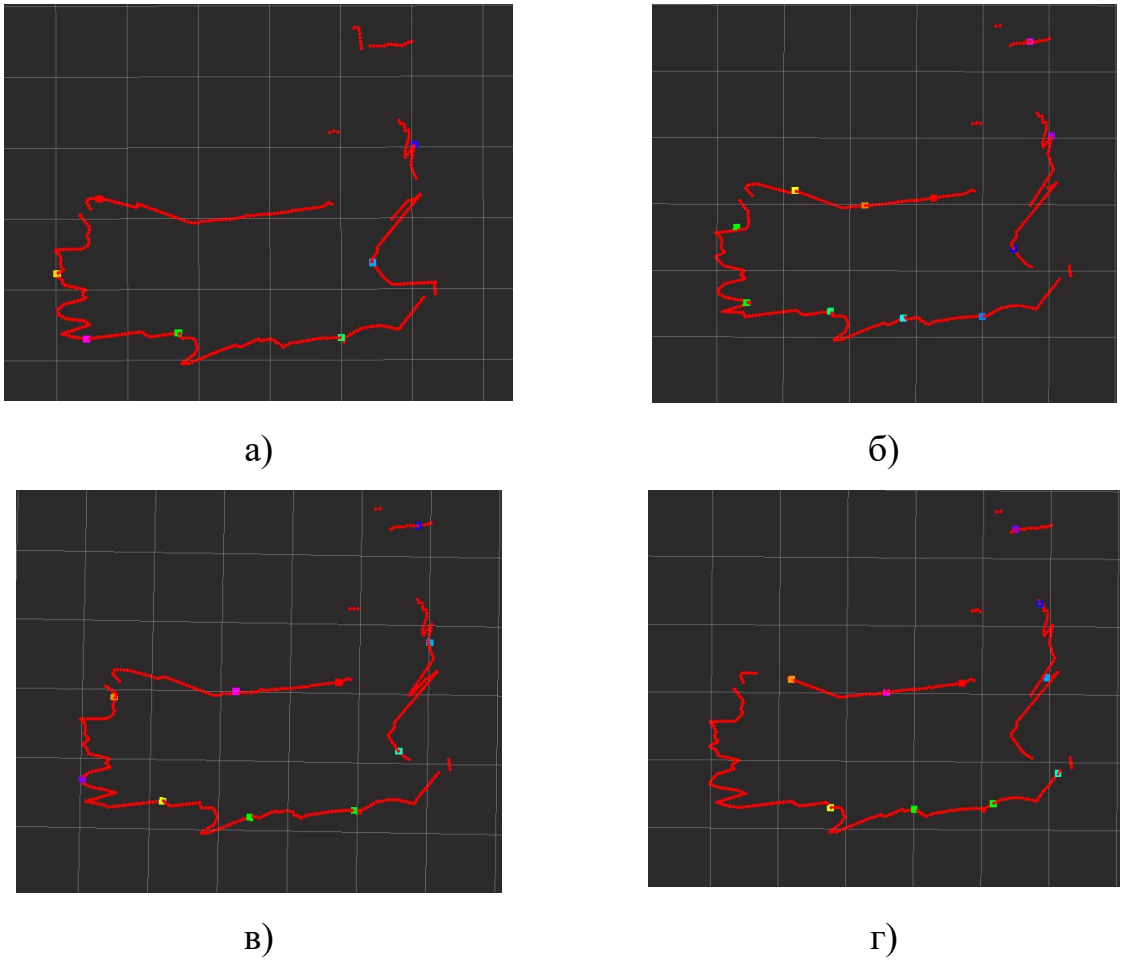
$$K_{(x,y)} = \frac{\dot{x}\ddot{y} - \dot{y}\ddot{x}}{\sqrt{(\dot{x}^2 + \dot{y}^2)^3}}. \tag{2.66}$$

Производные ищутся для результатов сглаживания функций $x(t)$ и $y(t)$, получаемых сверткой с гауссовым ядром, с учетом свойства свертки (правило дифференцирования):

$$\frac{\partial^n f(x) \otimes g(x,\sigma)}{\partial x^n} = f(x) \otimes \frac{\partial^n g(x,\sigma)}{\partial x^n}. \tag{2.67}$$

Изменяя величину σ, можно менять степень детализации за счет сглаживания деталей. В качестве доминантных точек выбираются точки контура, соответствующие экстремумам кривизны.

На рис. 2.20. представлены примеры работы выше перечисленных алгоритмов детектирования ориентиров по данным лидара.



а) б)

в) г)

Рис. 2.20. Примеры работы детекторов ориентиров: а) Teh-Chin; б) алгоритм Wu; в) согласованная фильтрация; г) метод масштабного пространства кривизны

2.6. Усовершенствованные алгоритмы EKF-SLAM

Алгоритм EKF-SLAM успешно используется для построения лишь небольших карт, так как имеет квадратичную вычислительную сложность обновления ковариационной матрицы на каждом шаге. Из-за этого различные улучшения алгоритма EKF-SLAM представляют большой интерес к исследованиям. В настоящей работе рассматриваются два усовершенствованных алгоритма EKF-SLAM: алгоритм EKF-SLAM с адаптационным диапазоном наблюдения и алгоритм EKF-SLAM с разделением и объединением.

2.6.1. Алгоритм EKF-SLAM с адаптационным диапазоном наблюдения

Идея алгоритма EKF-SLAM с адаптационным диапазоном наблюдения состоит в использовании локальной круговой карты для текущей оценки координат мобильной платформы и локализации зоны используемых алгоритмов, с одновременным обновлением глобальной карты (рис. 2.21) [39].

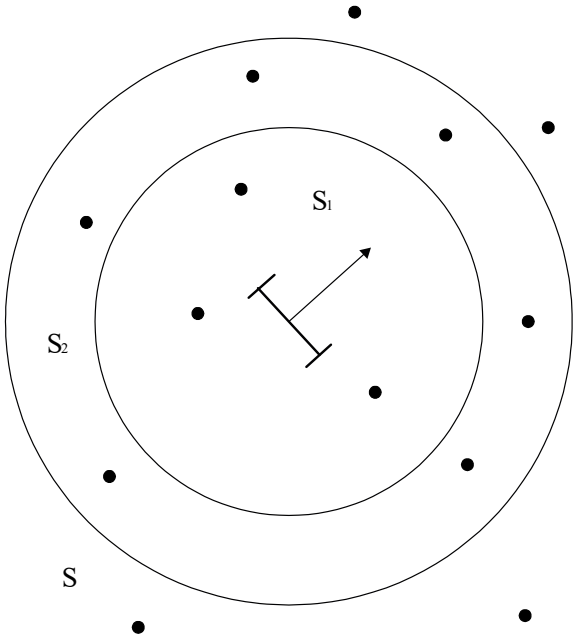


Рис. 2.21. Алгоритм EKF-SLAM с адаптационным диапазоном наблюдения (S – глобальная карта, S_1 – зона наблюдения мобильной платформы, S_2 – зона круговой локальной карты)

Если не менять радиус локальной карты, то возможны следующие проблемы:

1) Число ориентиров в области локальной карты может оказаться слишком малым для уточнения позиции мобильной платформы.

2) Число ориентиров в области локальной карты может оказаться слишком большим, что приведет к уменьшению скорости вычислений.

3) Если диапазон наблюдения является слишком большим, то достоверность наблюдения отдельных ориентиров снижается, что повлияет на точность позиционирования мобильной платформы.

Поэтому радиус локальной карты изменяется по следующим правилам:

1) Если число ориентиров в наблюдаемой области меньше, чем необходимое число для надежной коррекции положения мобильной платформы, и радиус области меньше, чем максимальный радиус, то радиус локальной карты увеличивается.

2) Если число ориентиров больше необходимого числа для хорошей точности алгоритма и радиус локальной карты больше, чем минимальный радиус, то радиус локальной карты уменьшается.

3) Во всех остальных случаях радиус локальной карты не изменяется.

2.6.2. Алгоритм EKF-SLAM с разделением и объединением

Парадигма «разделяй и властвуй» используется для поиска оптимального решения задачи. Задача делится на две или более похожих, но более простых подзадач, затем решают их по очереди и используют их решения для поставленной задачи.

Основная идея алгоритма EKF-SLAM с разделением и объединением заключается в том, чтобы создать последовательность локальных карт определенного размера, а затем объединить их последовательно. Для формирования глобальной карты алгоритм объединяет локальные карты с помощью двоичного дерева (рис. 2.22). Для формирования таких карт выполняется стандартный EKF-SLAM до фиксированного максимального размера. Для получения глобальной карты, полученные m локальных карт объединяются в m / 2 локальных карт двойного размера, которые в свою очередь будут объединяться в m / 4 локальные карты четырехкратного размера, до тех пор, пока окончательная карта размера n не будет результатом объединения 2 карт размером n / 2. Выполнение этого процесса сводится к обходу двоичного дерева и может быть легко реализовано с использованием стека карт [39].

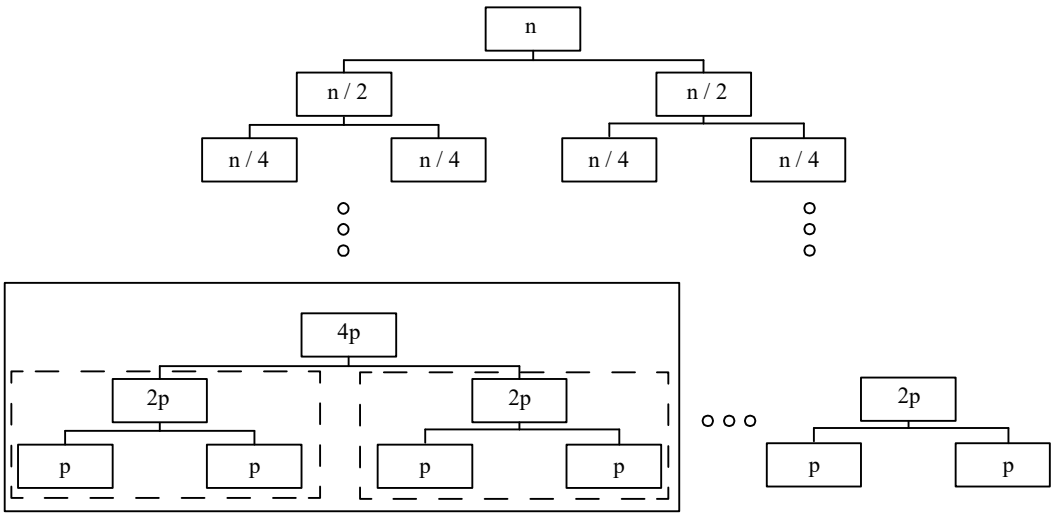


Рис 2.22. Бинарное дерево иерархии локальных карт.

Алгоритм состоит из двух основных шагов:

1) Построение двоичного дерева локальных карт.

2) Построение глобальной карты.

Первый этап выполняется во время движения мобильной платформы. Сначала инициализируется карта, в которой начальное положения мобильной платформы известна и ковариация равна 0, в противном случае карта будет не точна. Затем выполняется стандартный алгоритм EKF-SLAM, до тех пор пока не достигнет определенного размера N. В этот момент вектор состояния и ковариационная матрица карты m_1 будет иметь вид:

$$X_{1..i} = \begin{bmatrix} x_r \\ y_r \\ x_1 \\ y_1 \\ \dots \\ x_i \\ y_i \end{bmatrix}, \tag{2.68}$$

$$P_{1..i} = \begin{bmatrix} \sigma^2_{x_r x_r} & \sigma^2_{x_r y_r} & \sigma^2_{x_r x_1} & \sigma^2_{x_r y_1} & \cdots & \sigma^2_{x_r x_n} & \sigma^2_{x_r y_n} \\ \sigma^2_{x_r y_r} & \sigma^2_{y_r y_r} & \sigma^2_{x_r y_1} & \sigma^2_{y_r y_1} & \cdots & \sigma^2_{y_r x_n} & \sigma^2_{y_r y_n} \\ \sigma^2_{x_1 x_r} & \sigma^2_{x_1 y_r} & \sigma^2_{x_1 x_1} & \sigma^2_{x_1 y_1} & \cdots & \sigma^2_{x_1 x_n} & \sigma^2_{x_1 y_n} \\ \sigma^2_{x_1 y_r} & \sigma^2_{y_1 y_r} & \sigma^2_{x_1 y_1} & \sigma^2_{y_1 y_1} & \cdots & \sigma^2_{y_1 x_n} & \sigma^2_{y_1 y_n} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ \sigma^2_{x_n x_r} & \sigma^2_{x_n y_r} & \sigma^2_{x_n x_1} & \sigma^2_{y_n x_1} & \cdots & \sigma^2_{x_n x_n} & \sigma^2_{x_n y_n} \\ \sigma^2_{x_n y_r} & \sigma^2_{y_n y_r} & \sigma^2_{x_n y_1} & \sigma^2_{y_n y_1} & \cdots & \sigma^2_{x_n y_n} & \sigma^2_{y_n y_n} \end{bmatrix}. \quad (2.69)$$

Затем инициализируется новая карта $m_2 = (X_{i..j}, P_{i..j})$. Здесь существует два варианта инициализации. В первом варианте считать, что начальное положение платформы известно и его ковариационная часть матрицы равна 0. А все наблюдаемые ориентиры считать новыми, которые во время построения глобальной карты объединяться. Во втором варианте считать, что конечное положение мобильной платформы и ориентиры карты m_1 являются начальным условием для построения следующей карты.

Построение глобальной карты состоит из трех шагов:

1) Объединение карт. Карты объединяются последовательно с учетом бинарного дерева, следующим образом:

$$X_{1..j} = \begin{bmatrix} X_{1..i} \\ X_{i..j} \end{bmatrix}, \quad (2.70)$$

$$P_{1..j} = \begin{bmatrix} P_{1..i} & 0 \\ 0 & P_{i..j} \end{bmatrix}. \quad (2.71)$$

2) Шаг обновления. На этом шаге происходит объединение данных между двумя картами (поиск соответствия ориентиров). На этом этапе важно найти ориентиры-дубликаты. Поиск осуществляется не путем перебора каждого ориентира, а с помощью алгоритма «совместная ветвь и граница совместимости» (JCBB). Этот алгоритм ищет в бинарном дереве такую гипотезу H, при которой наблюдается наибольшее количество совместимых ориентиров. В основе гипотезы лежит идея, что вероятность принятия ложной ассоциации пары ориентиров уменьшается по мере увеличения

количества верной ассоциации пар ориентиров. С помощью этого алгоритма вычислительная сложность ассоциации данных меньше чем $O(n^2)$. [95]

3) Шаг трансформации

На шаге трансформации выполняется геометрическое преобразование для каждого элемента карты к единому базису.

2.6.3. Алгоритм EKF-SLAM с равномерным использованием ориентиров

Предыдущие алгоритмы используют геометрическую информацию ориентиров для построения локальных карт. В этом случае используются последние наблюдавшиеся ориентиры и при этом не учитывается корреляция между всеми ориентирами. Для того чтобы учитывать корреляцию между всеми ориентирами с сохранением скорости работы алгоритма, предложено формировать локальную карту, состоящую из наблюдаемых ориентиров, а также ориентиров, которые меньше всего использовали при построении локальной карты. Для каждого ориентира учитывается количество его наблюдений, и мера их использования для построения локальной карты рассчитывается по следующей формуле:

$$s = \frac{o - 0.5 * u}{1 + o}, \tag{2.72}$$

где s – мера использования ориентира o – количество наблюдений ориентира, u – количество использования ориентира в построении локальной карты.

Сначала добавляются наблюдаемые ориентиры, затем проверяется их количество. Если количество ориентиров меньше, чем заданное количество, то вычисляется мера использования для всех ориентиров, и на локальную карту добавляются ориентиры с самым высоким значением меры.

2.6.4. Обобщенный алгоритм EKF-SLAM

Было замечено, что улучшенные алгоритмы можно реализовать таким образом, чтобы можно было применять их единообразно (одним способом).

Это позволяет быстро реализовывать их, а также менять их между собой без переписывания с нуля всего алгоритма EKF-SLAM.

В связи с этим предложено следующее улучшение алгоритма EKF-SLAM, основанного на подстановке подматриц соответствующих объектов рассматриваемой системы (рис. 2.23). Т.е. таким образом можно не только строить карты, состоящие из разных типов объектов, но и единообразно применять различные алгоритмы построения и обработки локальных карт.

Алгоритм состоит из следующих основных этапов:

1) Добавление новых ориентиров на глобальную карту
2) Выбор объектов системы, удовлетворяющих определенным условиям для построения локальной карты. В данном случае это мобильная платформа и ориентиры, удовлетворяющие условиям определенного алгоритма построения локальной карты.
3) Оценка месторасположения мобильной платформы и ориентиров. Этот этап является стандартным для всех алгоритмов EKF-SLAM, но тут он применяется для локальной карты, а не для глобальной.
4) Объединение локальной карты с глобальной картой.

Следующие два этапа являются дополнительными и зависят от реализации алгоритма. Они нужны для синхронизации векторов состояния и матриц ковариации между выбранными объектами системы и обновленными значениями на глобальной карте.

Рис 2.23. Структурная схема обобщенного алгоритма EKF-SLAM

5) Обновление месторасположения мобильной платформы.
6) Обновление выбранных ориентиров.

Для упрощения реализации алгоритма необходимо:

1) Для каждого объекта на карте ввести уникальный идентификатор.
2) Разбить глобальную карту на подматрицы. Каждая подматрица должна соответствовать объекту на карте. Это необходимо для быстрого поиска, извлечения и вставки подматриц по уникальному идентификатору объекта.

3) Вычислить необходимые матрицы для применения расширенного фильтра Калмана для каждого типа объекта независимо от других типов.
4) Ввести операции извлечения и вставки подматриц.
5) Реализовать операцию построения локальной карты из выбранных объектов. Для этого строится вектор состояния и матрица ковариации путем последовательной вставки соответствующих подматриц, извлеченных из вектора состояния и матрицы ковариации глобальной карты.
6) Реализовать операцию построения необходимых матриц для применения расширенного фильтра Калмана. Такие матрицы строятся путем последовательной вставки подматриц, полученных на шаге 3.

Данный алгоритм позволяет также использовать несколько сенсоров. Это позволяет алгоритму эффективнее работать в динамических условиях окружающей среды. Например, с наступлением темного времени суток использование камеры на автономном транспортном средстве становится неактуальным. В таком случае, сбоя работы алгоритма SLAM не произойдет, потому что он будет использовать данные, полученные с других сенсоров.

Стоит отметить, что такой алгоритм не всегда применим. В рассматриваемой системе должны выполняться два условия:

1) Объекты системы должны быть независимыми. В этом случае можно пренебрегать корреляцией между ними, потому что ковариация между двумя независимыми случайными величинами равна 0.

2) У объектов системы значения в векторе состояния не должны меняться со временем. Если это условие нарушается, то при выборке объектов при вычислении у некоторых объектов не учтется изменение значений в векторе состояния, что приведет к неправильной работе расширенного фильтра Калмана. Если в системе имеются некоторые такие объекты, то их следует при вычислении всегда учитывать.

В данной работе рассматриваемая система удовлетворяет этим условиям. Из начальных условий ориентиры являются независимыми и стационарными. Из условий стационарности следует, что их координаты не меняются, а, следовательно, и значения в векторе состояния тоже не меняются. Но мобильная платформа этим условиям не удовлетворяет, поэтому при вычислениях величин, относящихся к мобильной платформе, всегда учитываются.

2.7. Карта смещений

Важной темой исследований в области обработки изображений и компьютерного зрения является согласование стереосоответствия. Проблема стереосоответствия является одной из основных задач для получения карты смещения. Определение карт смещения требуется в таких приложениях, как навигация и управление роботом, машинное зрение, медицинская визуализация и трехмерная реконструкция.

Алгоритмы стереосоответствия могут быть классифицированы на две категории: глобальные и локальные алгоритмы. Глобальные алгоритмы [42] опираются на итеративные схемы, которые выполняют распределения несоответствий на основе минимизации глобальной функции стоимости. Эти алгоритмы дают точные и плотные измерения смещения, но требуют очень больших вычислительных затрат, что делает их непригодными для приложений реального времени.

Локальные алгоритмы [43, 44, 45], также называемые алгоритмами на основе областей, вычисляют значение смещения в каждом пикселе на основе фотометрических свойств соседних пикселей. По сравнению с глобальными алгоритмами, локальные алгоритмы дают значительно менее точные карты смещения, но могут работать достаточно быстро для развертывания во многих приложениях реального времени.

Поскольку два изображения одной и той же сцены снимаются с несколько разных точек обзора с использованием двух камер,

расположенных в одной плоскости, то для большинства пикселей на левом изображении имеются соответствующие пиксели на правом изображении на одной и той же горизонтальной линии. Разница в координатах соответствующих пикселей называется смещением, которое обратно пропорциональна расстоянию между объектами и камерой. Несоответствие может быть определено следующим уравнением:

$$d = \frac{bf}{z}, \tag{2.73}$$

где z – расстояние от точки объекта до камеры (глубина), b – базовое расстояние между левой и правой камерами, а f – фокусное расстояние объектива камеры. На рисунке 2.24 показано, что два изображения объекта получены левой и правой камерами, наблюдающими общую сцену. Нахождение одинаковых точек в двух изображениях таким образом, что совпадающие точки являются одинаковыми проекциями точки в сцене, называется согласованием стереосоответствия и является фундаментальной вычислительной задачей, лежащей в основе определения карты смещения [45].

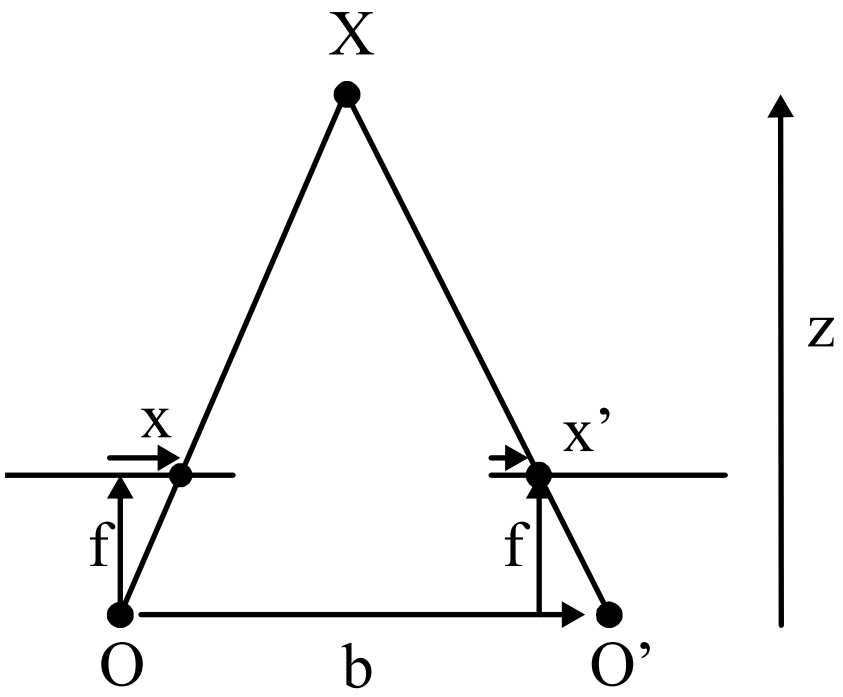


Рис. 2.24. Вычисление расстояния до наблюдаемой точки

Стереосоответствие обычно определяют на основе сопоставления окон пикселей на эпиполярной линии. Для этого изображения выравниваются так, чтобы все эпиполярные линии были параллельны сторонам изображения и из точки с координатами (x, y) соответствующая ей эпиполярная линия задавалась уравнением $x = x_0$. Такой процесс выравнивания изображения называется ректификация. Чтобы определить соответствие пикселя на левом изображении, вычисляется стоимость окна для всех пикселей-кандидатов на правом изображении в пределах диапазона поиска, с использованием таких методов как сумма квадрата разностей или сумма абсолютных разностей:

$$E_{SSD}(p,d) = \sum_{i=-1}^{l} \sum_{j=-k}^{k} [B(x+i, y+j) - M(x+d+i, y+j)]^2, \qquad (2.74)$$

$$E_{SAD}(p,d) = \sum_{i=-1}^{l} \sum_{j=-k}^{k} [B(x+i, y+j) - M(x+d+i, y+j)]^2 . \qquad (2.75)$$

Пиксель на правом изображении, который дает наилучшую стоимость окна, то есть минимальное значение SSD или SAD является наиболее похожим на пиксель левого изображения (рис. 2.25).

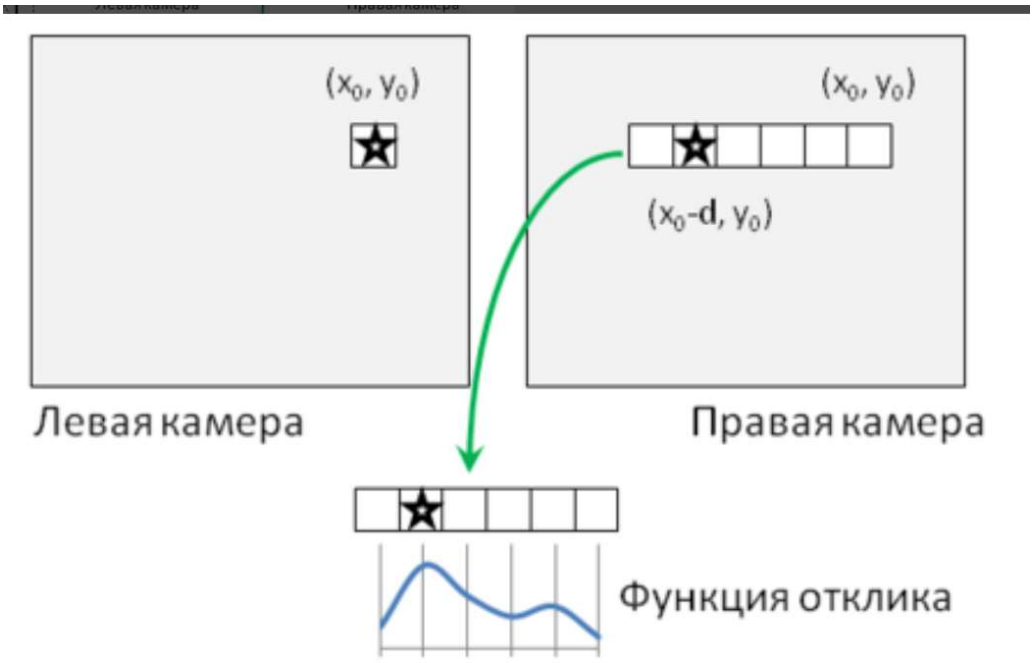


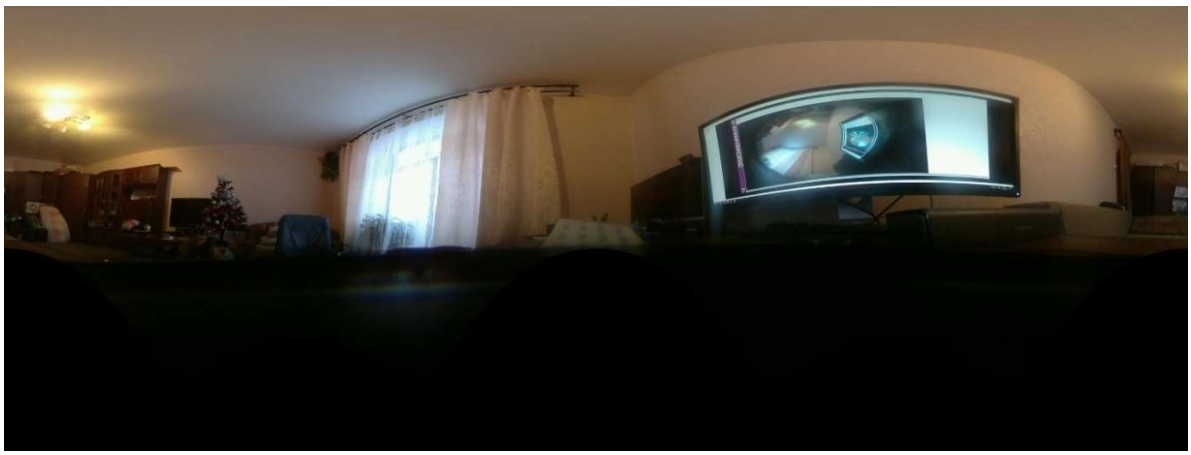
Рис. 2.25. Поиск соответствующего пикселя

2.8. Преобразование изображения в панорамное изображение

Для построения карты глубины по стереопаре изображений необходимо для каждой точки на одном изображении найти парную ей точку на другом изображении (рис. 2.26). А по паре соответствующих точек можно выполнить триангуляцию и определить координаты их прообраза в трехмерном пространстве. Парную точку надо искать на эпиполярной линии. Из-за того, что эпиполярные линии для сферической модели камеры являются большими кругами, то поиск соответствия должен выполняться вдоль больших кругов. Поиск вдоль таких кривых имеет высокую вычислительную сложность по сравнению с поиском по прямым линиям. Но можно ускорить процесс построения карты глубины.

а)

б)



в)

Рис. 2.26. а) исходное изображение; б) эпиполярные линии на исходном изображении; в) панорамное изображение, у которого эпиполярные линии являются горизонтальными прямыми

Идея состоит в преобразовании сферического изображения так, чтобы можно было применить обычный, основанный на корреляции алгоритм построения карты глубины. Для этого формируется облако точек, которое формирует полусферу и панорамное изображение, где каждый пиксель соответствует точке из облака. Для этого используется следующее выражение:

$$u = \cos\left(\frac{i}{m}\pi\right)$$

$$v = \sin\left(\frac{i}{m}\pi\right)\cos\left(\frac{j}{n}2\pi\right)$$

$$s = \sin\left(\frac{i}{m}\pi\right)\sin\left(\frac{j}{n}2\pi\right),$$

где i и j – координаты текущего пикселя, m и n – размеры панорамного изображения.

Зная преобразование отображения трехмерной точки на плоскость изображения, можно провести соответствие между точкой сферического изображения и точкой панорамного изображения и построить, таким образом, панорамное изображение, где эпиполярные лини будут являться прямыми.

2.9. Ассоциация данных

Ассоциация данных происходит не с помощью критерия близкого нахождения наблюдаемого ориентира к прогнозируемому местоположению (геометрический метод), а с помощью технического зрения. Многие способы восприятия, такие как зрение, предоставляют богатую информацию о форме, цвете и текстуре, и все они могут быть использованы для поиска соответствия между двумя наборами ориентиров, полученных по данным лидара.

Основные этапы алгоритма ассоциации данных:

1) Детектирование ориентира по данным лазерной сканирующей системы.

2) Преобразование изображения «рыбий глаз» в панорамное изображение.

3) Поиск области изображения, где находится найденный ориентир

4) Поиск ключевых точек и их дескрипторов методом SIFT в найденной области изображения (рис. 2.27).

5) Сопоставление ориентиров по набору совпавших ключевых точек.

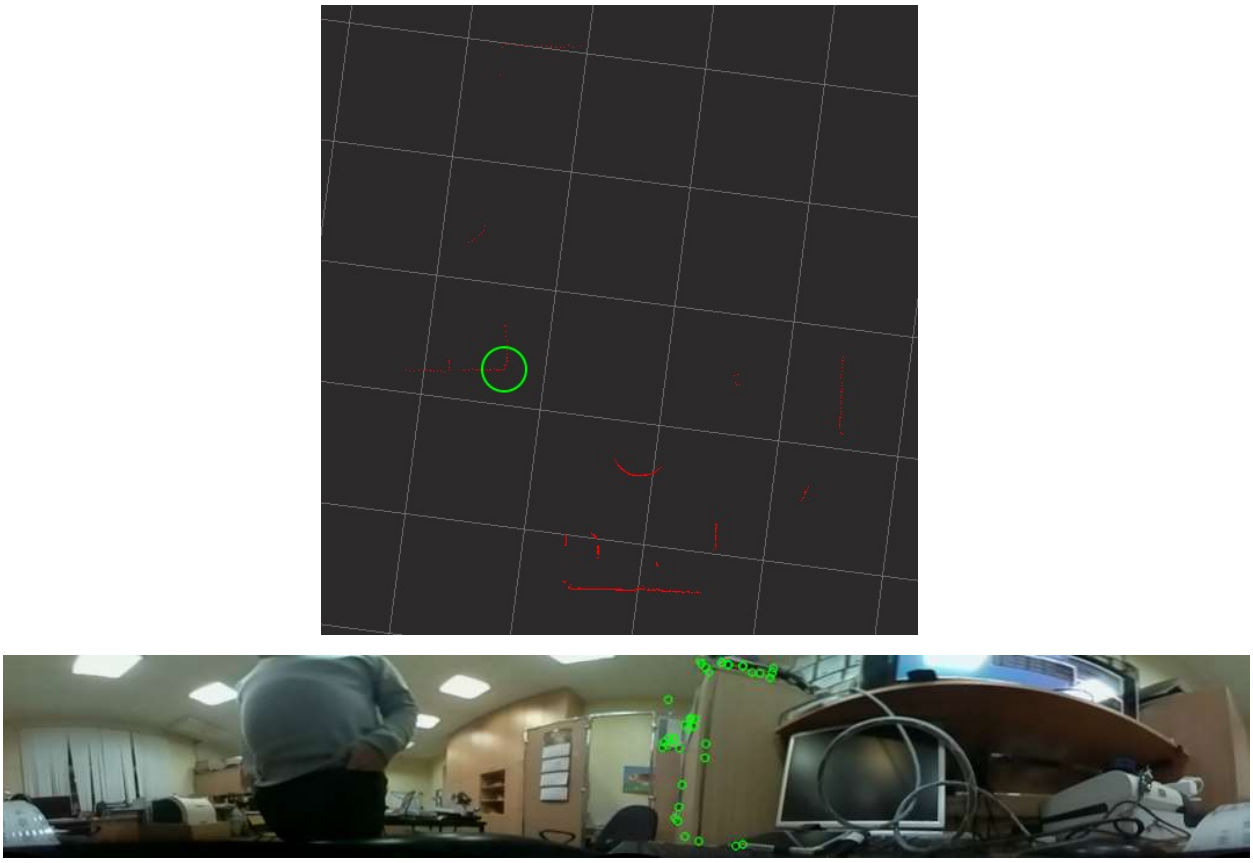


Рис. 2.27. Описание ориентира с помощью SIFT – дескрипторов

2.10. Построение трехмерной карты

Для получения карты смещения используются не две камеры, а одна камера по двум последовательным изображениям, полученным с разных точек обзора. Каждая точка обзора определяется с помощью алгоритма EKF-SLAM. В данной работе применяется локальный алгоритм стереосоответствия, в котором карта смещения определяется на основе сопоставления окон пикселей на эпиполярной линии с помощью суммы абсолютных разностей:

$$E_{SAD}(p,d) = \sum_{i=-1}^{l}\sum_{j=-k}^{k}[B(x+i,y+j) - M(x+d+i,y+j)]^2, \quad (2.76)$$

где B – первое изображение, M – второе изображение.

Точность оценки карты смещения часто страдает от экстремальных сценариев, таких как область без текстур, переэкспонирование, повторяющаяся структура и т. д. Чтобы повысить точность карты смещения, необходима постобработка. На этапе постобработки применяется взвешенная фильтрация наименьших квадратов, поскольку она обеспечивает хорошее сглаживание, сохраняющее края (Рис. 2.28) [2]. Цель фильтрации стереосоответствия может быть выражена в виде минимизации уравнения:

$$\sum_p\left(\left(D'_p - D_p\right)^2 + \lambda\left(a_{x,p}(I)\left(\frac{\partial D}{\partial x}\right)_p^2 + a_{y,p}(I)\left(\frac{\partial D}{\partial y}\right)_p^2\right)\right), \quad (2.77)$$

где D – исходное изображение, D′ – отфильтрованное изображение, p – индекс, определяющий позицию пикселя, $a_{x,p}(I)$ и $a_{y,p}(I)$ – весовые коэффициенты, которые определяются следующим образом:

$$a_{x,p}(I) = \left(\left|\frac{\partial l}{\partial x}(p)\right|^{\alpha} + \varepsilon\right)^{-1}, \quad (2.78)$$

$$a_{y,p}(I) = \left(\left|\frac{\partial l}{\partial y}(p)\right|^{\alpha} + \varepsilon\right)^{-1}, \quad (2.79)$$

где l – канал яркости в логарифмическом масштабе изображения I, параметр α определяет степень резкости границы, ε – константа с малым значением.

Рис. 2.28. Пример работы wls-фильтра

Трехмерная карта строится по карте глубины с использованием сферической модели камеры по формулам 2.11 – 2.14. Пример построения трехмерной сцены представлен на рис. 2.29.

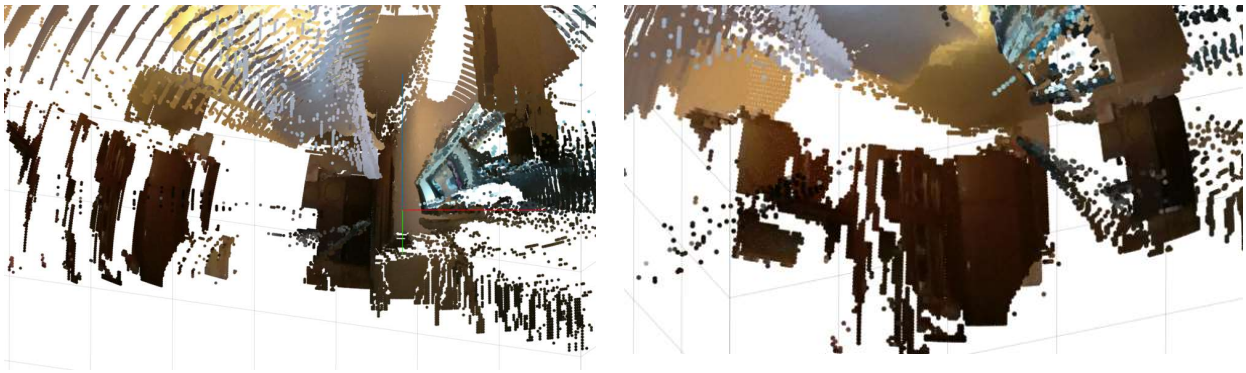


Рис. 2.29. Реконструкция сцены

2.11. Краткие выводы по главе 2

Разработан алгоритм одновременной локализации и построения карты с использованием двух типов сенсоров: лазерной сканирующей системы и камеры с объективом типа «рыбий глаз».

Улучшены детекторы ориентиров по данным лазерной сканирующей системы с использованием контурного анализа. Для нахождения ориентиров данные лазерной сканирующей системы преобразуются в комплекснозначный сигнал. Этот сигнал делится на фрагменты для решения проблем недостающих данных и окклюзии. Предложенное преобразование позволяет применять методы, инвариантные к изменению масштаба, повороту и переносу, а также корреляционный и спектральный анализ.

Рассмотрена сферическая модель камеры. По этой модели найдены внутренние параметры камеры, с использованием которых можно делать реконструкцию окружающей среды по карте глубины. Эта реконструкция происходит по двум последовательным изображениям, месторасположения которых вычисляются с помощью разработанного алгоритма EKF-SLAM. Каждое изображение преобразуется в панорамное изображение для вычисления необходимой для реконструкции карты смещения.

Рассмотрен ориентир типа «особая точка». Для этого типа ориентира рассмотрен метод кодирования обратной глубины. Также вычислена матрица якобиана измерения ориентира с учетом сферической модели камеры.

Рассмотрены улучшенные алгоритмы расширенного фильтра Калмана. Выявлен недостаток алгоритмов построения локальных карт «разделяй и властвуй» и с адаптивным диапазоном наблюдения, заключающийся в том, что в вычислениях используются последние наблюдавшиеся ориентиры. В связи с этим предложен алгоритм с равномерным использованием ориентиров.

На основе анализа улучшенных алгоритмов EKF-SLAM предложен обобщенный алгоритм EKF-SLAM, позволяющий применять различные алгоритмы построения локальных карт, а также рассматривать сложные динамические системы.

Глава 3. ИССЛЕДОВАНИЕ РАБОТОСПОСОБНОСТИ АЛГОРИТМА EKF-SLAM И ЕГО МОДИФИКАЦИЙ

3.1. Методы получения оценки работы алгоритма SLAM

Оценка работы алгоритма SLAM является сложной задачей. Сравнить карту, полученную алгоритмом, с картой местности практически очень сложно. Для этого необходима точная карта местности и метрика, позволяющая сравнить эти карты. Легче сравнить траекторию (последовательность месторасположений $P_1, \dots, P_n$), полученную алгоритмом с истинной траекторией (последовательность месторасположений $Q_1, \dots, Q_n$). Истинную траекторию движения камеры можно получить с помощью маркерной системы захвата движения. Для сравнения траекторий необходимо произвести передискретизацию, т.е. привести последовательности к одинаковой длине и синхронизировать по времени. Для оценки работы алгоритма SLAM используется две основные метрики.

Относительная ошибка месторасположения (RPE). Она измеряет медленное постоянное увеличение ошибки месторасположения камеры (дрейф) в течение фиксированного интервала времени Δt или интервала между отсчетами Δn последовательностей. Метрика RPE определяется на шаге времени i как:

$$E_i = (Q_i^{-1} Q_{i+\Delta n})^{-1} (P_i^{-1} P_{i+\Delta n}). \tag{3.1}$$

Из последовательности *RPE* можно вычислить среднеквадратическую ошибку *RSME* для всех отсчетов i:

$$RSME(E_{1:n}, \Delta n) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \|trans(E_i)\|^2}, \tag{3.2}$$

где $trans$ () – операция извлечения компонентов из матрицы, относящихся к перемещению.

Величина интервала между отсчетами Δn важна при оценке ошибки построения траектории. При Δn = 1 величина $RMSE(E_{1:n})$ будет показывать величину дрейфа в один кадр. При Δn, равном частоте кадров, с которой последовательность записывалась, $RMSE(E_{1:n})$ будет показывать величину дрейфа в секунду. Плохим выбором интервала между отсчетами Δn является n, потому что в этом случае учитываются только ошибки первоначального положения и конечного. Ошибка первоначального положения вносит больше вклада, чем ошибка конечного [97-98], поэтому для сравнения алгоритмов SLAM рекомендуется вычислять *RMSE* для всех возможных интервалов Δn.

Абсолютная ошибка траектории (ATE)

Метрика *RPE* не учитывает совпадение вычисленной траектории и измеренной. Эти траектории могут расходиться. Для решения этой проблемы в метрике *ATE* предложено траектории выравнивать. Для этого нужно найти преобразование S, при котором наблюдается минимальная сумма отклонений вычисленной последовательности месторасположений $P_1, \dots, P_n$ от измеренной последовательности месторасположений $Q_1, \dots, Q_n$. Это может быть достигнуто с использованием метода Хорна [96]. Учитывая это преобразование, абсолютная ошибка траектории на временном шаге i может быть вычислена как [57]:

$$F_i = Q_i^{-1} S P_i. \tag{3.3}$$

Ааналогичным образом вычисляется среднеквадратичная оценка по всем временным индексам i, т.е.:

$$RMSE(F_{1:n}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} \|trans(F_i)\|^2}. \tag{3.4}$$

3.2. Сравнение детекторов ориентиров

3.2.1. Определение оптимальных параметров детекторов

Приводятся результаты исследование по определению оптимальных параметров детекторов обнаружения ориентиров (рис. 3.1), такие как размер окна и порог (табл. 2).

Для оценки качества работы детекторов использовался показатель F-мера:

$$F = 2 * \frac{Precision * Recall}{Precision + Recall}, \quad (3.5)$$

где $Precision$ – точность, $Recall$ – полнота.

$$Recall = \frac{n_{\text{ВО}}}{n_{\text{ВО}}+n_{\text{П}}}, \quad (3.6)$$

$$Precision = \frac{n_{\text{ВО}}}{n_{\text{ВО}}+n_{\text{ЛО}}}, \quad (3.7)$$

где $n_{\text{ВО}}$ – число верно обнаруженных ориентиров, $n_{\text{ло}}$ – число ложно обнаруженных ориентиров, $n_{\text{П}}$ – число необнаруженных ориентиров.

Максимальное значение F-меры детекторов составило 0,95, хуже всего работает метод масштабного пространства кривизны, у которого F-мера составила около 0,6.

Таблица 2. Оптимальные параметры для алгоритмов детектирования ориентиров

	Алгоритм согласованной фильтрации	Алгоритм Wu	Алгоритм Teh-Chin	Метод масштабного пространства кривизны
Параметры	Размер окна 12 Порог 0,85	Размер окна 12 Порог 0,1	Размер окна 12 Порог 0,2	Размер окна 14 Дисперсия 16
F-мера	0,87	0,95	0,87	0,64

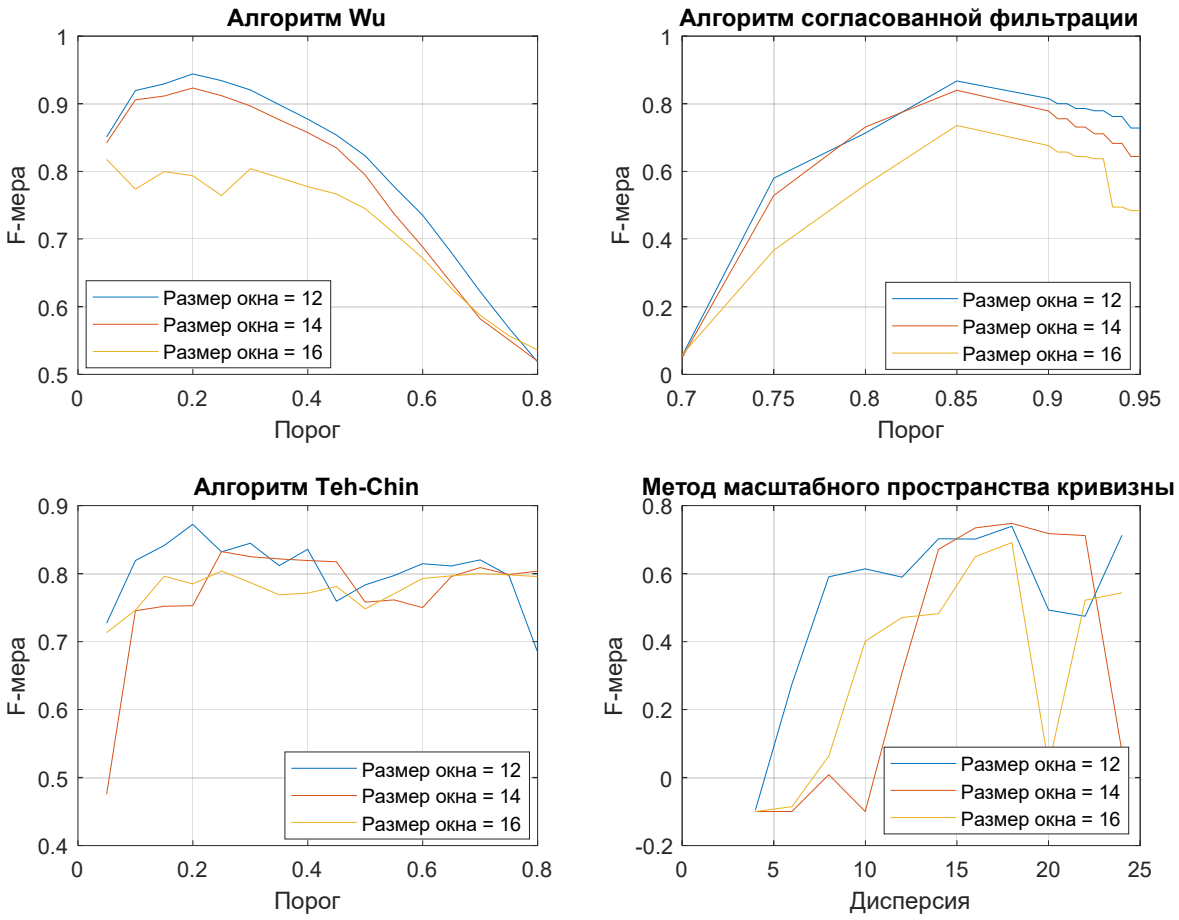


Рис. 3.1. Зависимость работы детекторов от параметров алгоритмов

3.2.2. Изучение влияния шума на работу детекторов

Рассмотрено также влияние шума на работу детекторов обнаружения ориентиров (рис. 3.2). Анализ результатов показывает, что при АБГШ метод согласованной фильтрации и алгоритм Wu работают лучше, чем остальные алгоритмы.

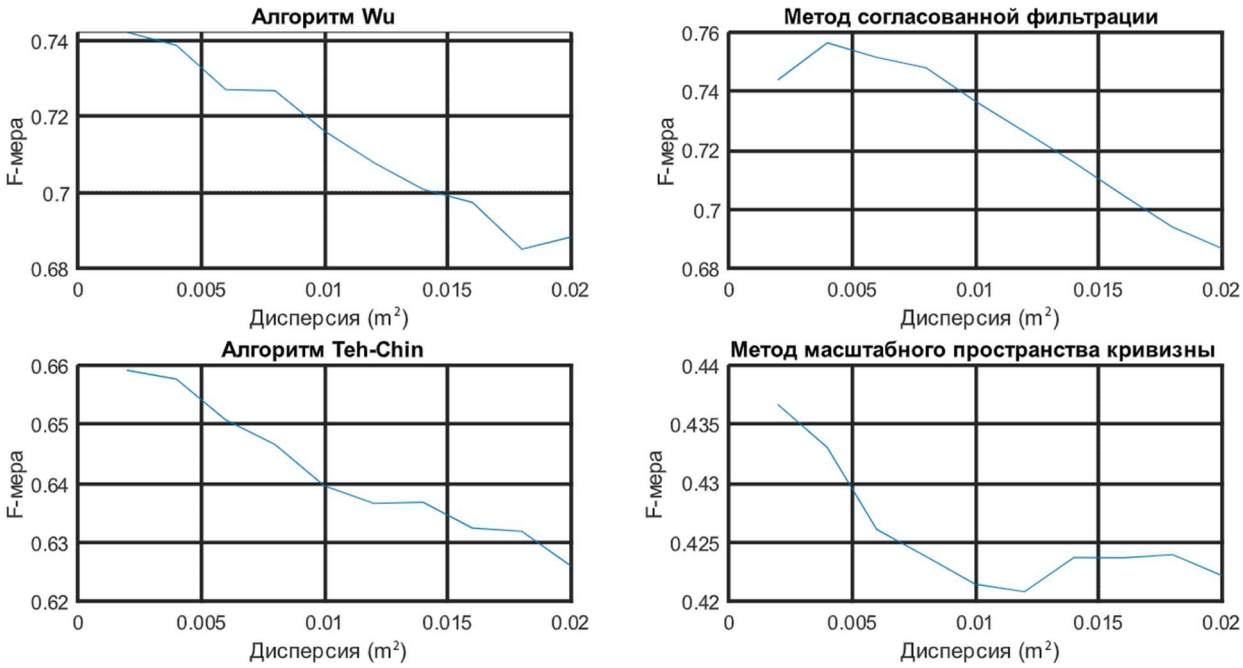
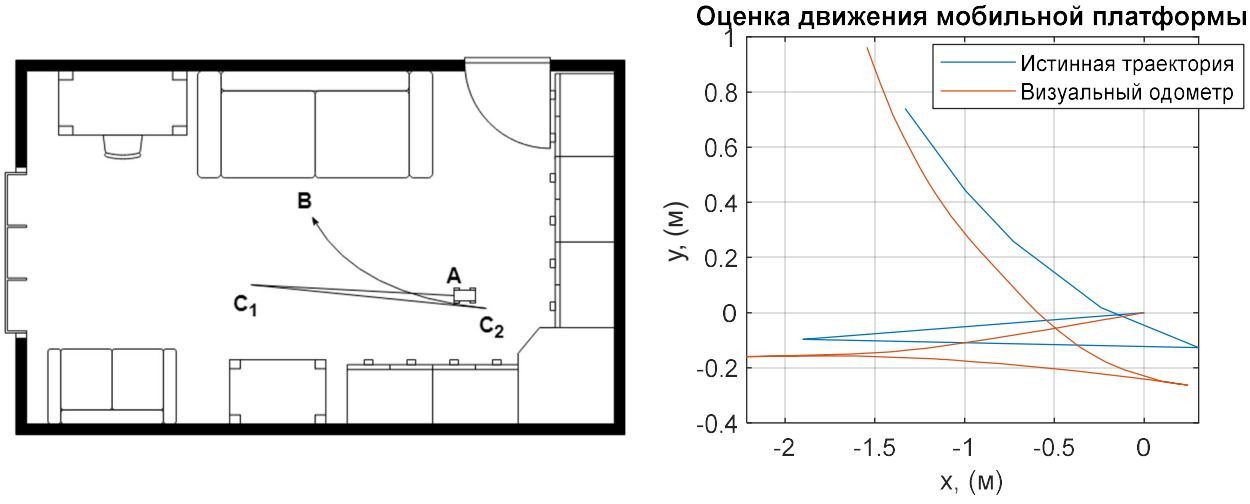


Рис. 3.2. Зависимость работы детекторов от дисперсии АБГШ

3.3. Оценка точности показаний визуального одометра

Экспериментальные исследования проводились в помещении. Оператор с помощью пульта дистанционного управления вел мобильную платформу из точки А в конечную точку В, через условные промежуточные точки С1 и С2. Длина траектории составила 5,9 м (рис. 3.3).



а) б)

Рис. 3.3. Экспериментальные исследования: а) траектория движения мобильной платформы; б) результат работы визуального одометра

Ошибка *RPE* составила 0,45 м, а ошибка *ATE* – 0,05 м.

3.4. Сравнение алгоритма EKF-SLAM с использованием разных видов датчиков

Для сравнения работы алгоритма EKF-SLAM с использованием разных видов датчиков делалось три записи сразу со всеми датчиками в тестовом помещении (рис. 3.4). Кроме того, фиксировалась последовательность месторасположений мобильной платформы во время записи. Далее строилась последовательность оценок месторасположений, полученных при помощи алгоритма EKF-SLAM. Полученные последовательности приводились к общему виду с помощью интерполяции. С помощью этих последовательностей производилась оценка *RPE* и *ATE*. Результаты эксперимента представлены в табл. 3 и на рис. 3.5.

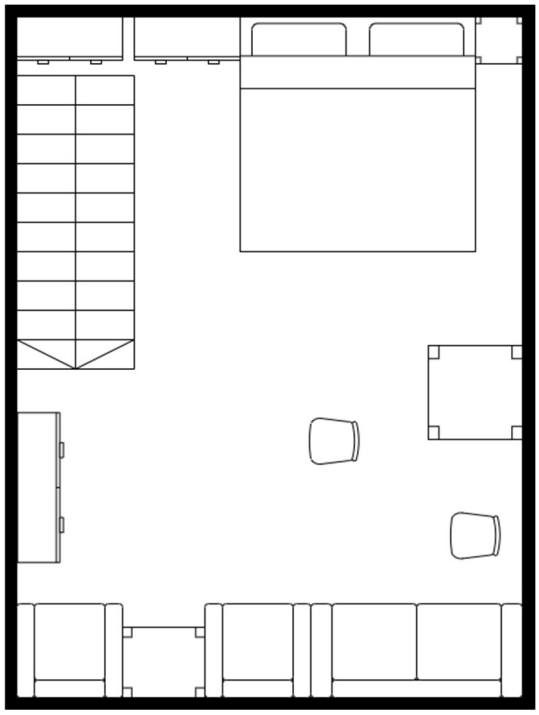


Рис. 3.4. Тестовое помещение

Таблица 3. Оценки *RPE* и *ATE* для разных видов датчиков

Номер заезда	Лидар		Камера		Лидар и камера	
Алгоритм EKF-SLAM						
	RPE (м)	*ATE* (м)	*RPE* (м)	*ATE* (м)	*RPE* (м)	*ATE* (м)
1	0.32	0.13	0.31	0.09	0.31	0.08
2	0.36	0.19	0.36	0.20	0.34	0.16
3	1.98	0.16	1.98	0.17	1.97	0.17
Алгоритм EKF-SLAM «Разделяй и властвуй»						
1	0.33	0.03	0.33	0.03	0.31	0.02
2	0.38	0.01	0.38	0.20	0.31	0.02
3	1.95	0.16	1.95	0.16	1.96	0.16
Алгоритм EKF-SLAM с адаптивным диапазоном наблюдения						
1	0.32	0.02	0.32	0.05	0.32	0.02
2	0.37	0.01	0.38	0.01	0.38	0.01
3	1.98	0.17	2.00	0.18	1.97	0.17
Алгоритм EKF-SLAM с равномерным использованием ориентиров						
1	0.33	0.03	0.34	0.04	0.32	0.02
2	0.41	0.01	0.43	0.01	0.41	0.01
3	2.02	0.13	1.94	0.13	1.91	0.16

Как видно из приведённых результатов, алгоритм с использованием двух датчиков способен точнее восстановить траекторию движения камеры и окружающее пространство. В результате использования двух датчиков точность позиционирования увеличилось на 1,95 % по метрике *RPE* и на 21,26 % по метрике *ATE* в сравнении с использованием только камеры, а также увеличилось на 2,23 % по метрике *RPE* и на 4,8 % по метрике *ATE* в сравнении с использованием только лидара.

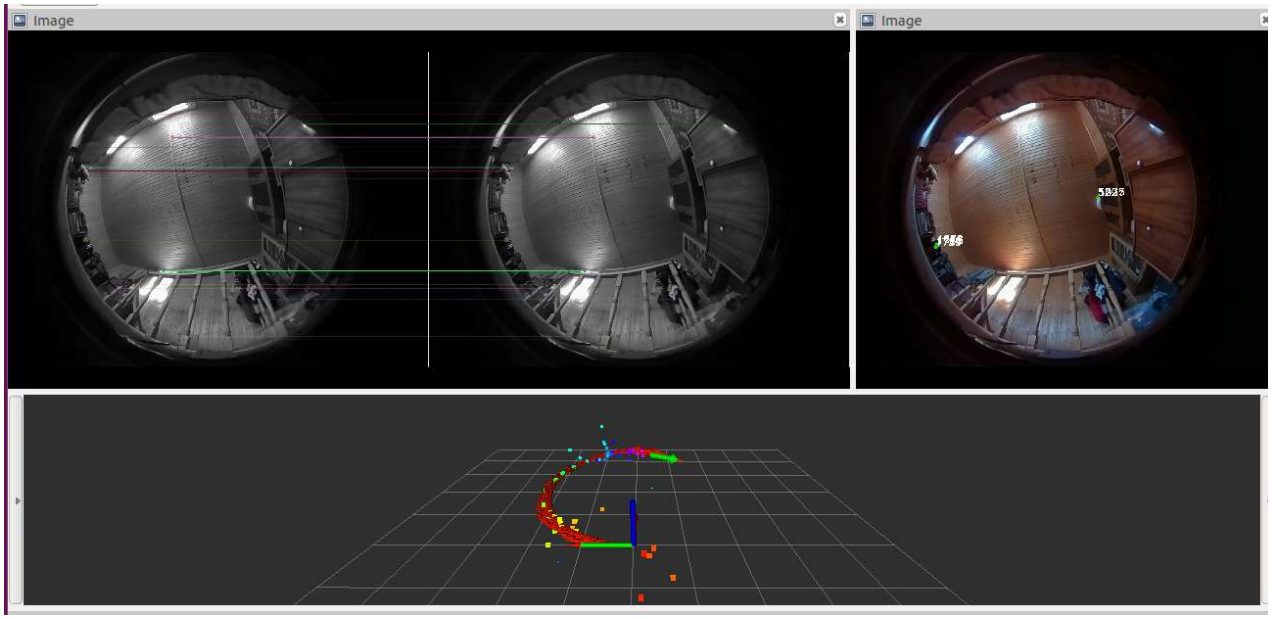


Рис. 3.5. Результат работы алгоритма EKF-SLAM

3.4.1. Исследование зависимости времени работы алгоритмов от количества ориентиров

Исследовано время работы одной итерации каждого алгоритма в зависимости от количества ориентиров (рис. 3.6).

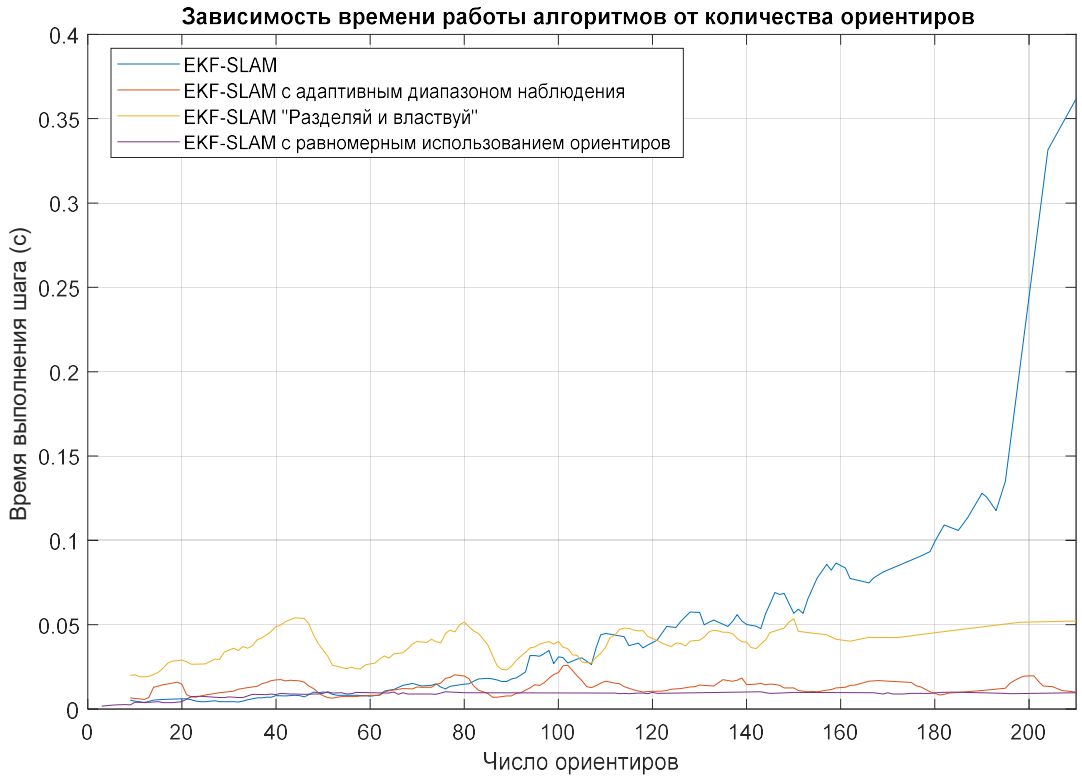


Рис. 3.6. Зависимость времени работы алгоритмов от количества ориентиров

Из графика видно, что алгоритм EKF-SLAM имеет квадратичную вычислительную сложность, а остальные линейную. Также у алгоритма EKF-SLAM «разделяй и властвуй» наблюдается пилообразная форма, это связано с тем, что он во время работы строит последовательно несколько локальных карт фиксированного размера.

3.4.2. Исследование зависимости средней ошибки работы алгоритмов от количества шагов

Исследована точность алгоритмов EKF-SLAM и его улучшенных версий (рис. 3.7).

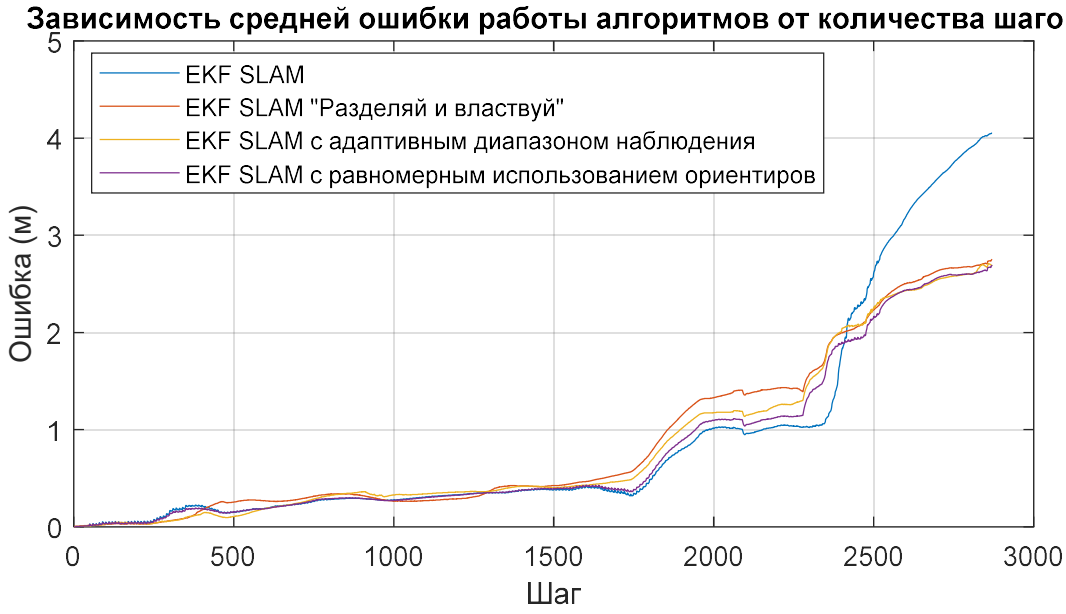


Рис. 3.7. Зависимость средней ошибки при оценке движении мобильной платформы различными алгоритмами EKF-SLAM от количества шагов

Средняя ошибка алгоритма EKF-SLAM составляет 0,89±1,1 м, у EKF-SLAM «разделяй и властвуй» составляет 0,87±0,85 м, у EKF-SLAM с адаптивным диапазоном наблюдения она составляет 0,82±0,83 м, у алгоритма EKF-SLAM с равномерным использованием ориентиров – 0,78±0,81 м. При большом количестве шагов ошибка определения месторасположения мобильной платформы у алгоритма EKF-SLAM по сравнению с другими алгоритмами возрастает. Наименьшую среднюю ошибку имеет алгоритм EKF-SLAM с равномерным использованием ориентиров. Эта ошибка меньше

на 12,35 %, чем средняя ошибка у алгоритма EKF-SLAM, меньше на 10,34%, чем у алгоритма EKF-SLAM «Разделяй и властвуй» и меньше на 4,87%, чем средняя ошибка у алгоритма EKF-SLAM с адаптивным диапазоном наблюдения.

3.4.3. Сравнение визуальных алгоритмов одновременной локализации и построения карты

Полноценно сравнить различные алгоритмы SLAM достаточно трудно, потому что они могут использовать разные датчики. Поэтому сравнить их, используя одни и те же входные данные, не получится. Но можно сравнить приблизительно, используя выше перечисленные метрики.

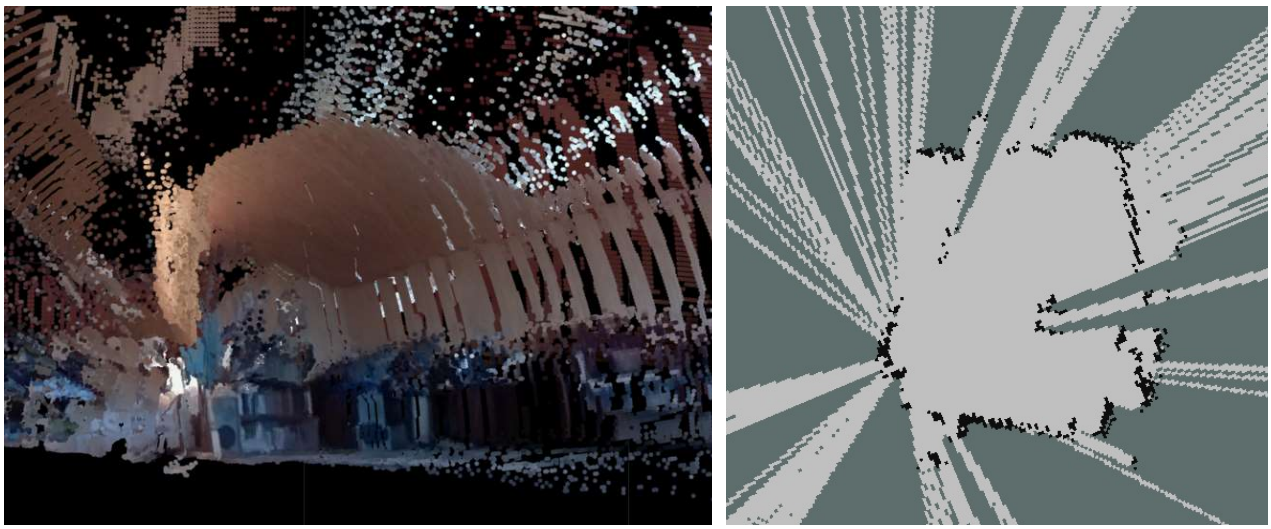
Для сравнения алгоритмов используется метрика *RPE* и полученные результаты из источников 25, 87, 88. Результаты представлены в табл. 4.

Таблица 4. Результаты сравнения алгоритмов

Алгоритм	*RPE* (м)
ORB – SLAM	0,02 ± 0,01
LSD – SLAM	0,37±0,48
RGBD – SLAM	0,1 ± 0,06
EKF-SLAM	0,09 ±0,08

Из таблицы видно, что алгоритм EKF-SLAM точнее работает на 75,68%, чем алгоритм LSD-SLAM, на 10% точнее, чем RGBD-SLAM, и хуже на 77,78%, чем ORB-SLAM.

Стоит отметить, что результатом представленного алгоритма EKF-SLAM является не только оценка траектории движения, карта, представленная в виде вектора состояния и матрицы ковариации, но и трехмерная и двухмерная карты без дополнительной обработки, представленные на рис. 3.8. По ним мобильная платформа сможет автономно передвигаться и взаимодействовать с окружающей средой.



а) б)

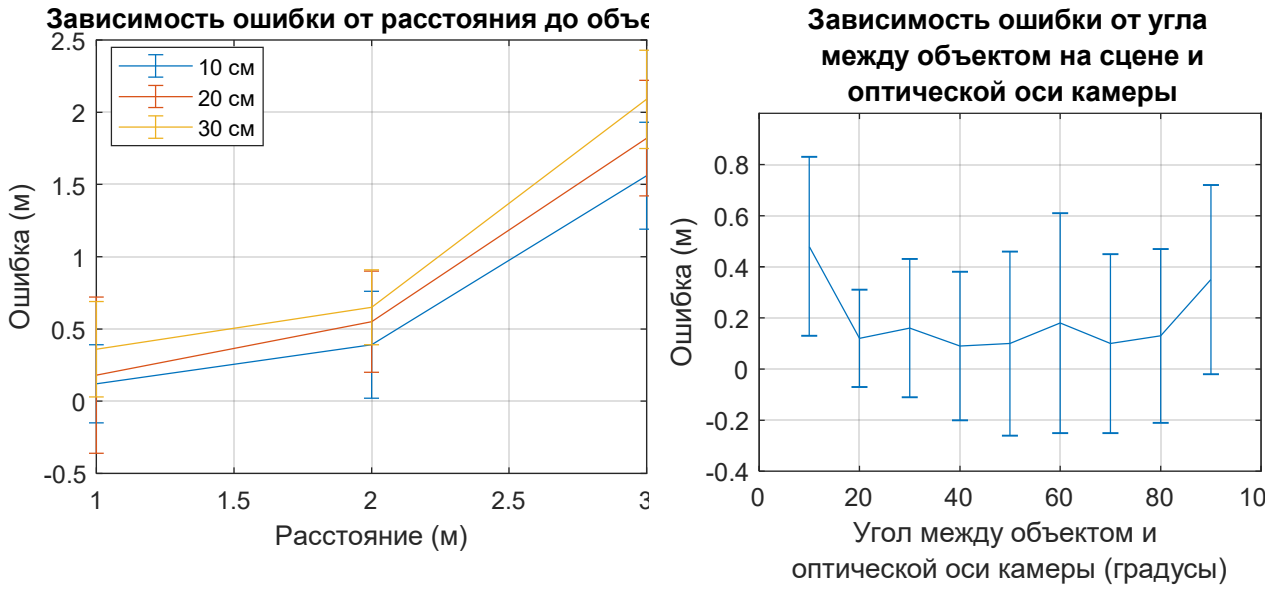
Рис. 3.8. Построенные карты: а) трехмерная; б) двумерная

3.5. Исследование точности восстановленной информации о глубине сцены

Чтобы проверить точность восстановленной информации о глубине из полученной карты глубины, использовалась сцена, состоящая из трех панелей, расположенных от камеры на расстоянии: 1, 2 и 3 м. Расстояние между позициями съемки составляло: 10 см, 20 см, 30 см, разрешение полученных изображений составляло 2592x1944 (рис. 3.9а). Также проведено исследование зависимости точности карты глубины от угла между тестовым объектом на сцене и оптической осью камеры, расстояние между позициями съемки составляло 10 см (рис. 3.9.б).

По графику зависимости ошибки значений карты глубины от расстояния до объекта (рис. 3.9.а) видно, что при увеличении расстояния между позициями съемки точность карты глубины уменьшается. Оптимальное расстояние между позициями съемки должно быть около 10 см.

На графике зависимости ошибки значений карты глубины от угла между объектом на сцене и оптической осью камеры наблюдаются возрастания ошибки при близких значениях углов 0 и 90 градусов.



а) б)

Рис. 3.9. Средняя ошибка измеренного расстояния по карте глубины до тестового объекта на сцене: а) зависимость ошибки от расстояния между позициями съемки; б) зависимость ошибки от угла между тестовым объектом на сцене и оптической осью камеры

Возрастание ошибки при значениях угла, близких к 0 градусов, связано с сильным искажением, полученным при преобразовании обычного изображения в панорамное. При значениях угла, близких к 90 градусам, возрастание ошибки связано с высоким уровнем дисторсии у объектива типа «рыбий глаз».

3.6. Краткие выводы по главе 3

Определены оптимальные параметры детекторов ориентиров с использованием контурного анализа. В исследованиях влияния шума на детектирование ориентиров использовалась модель шумового воздействия АБГШ с диапазоном дисперсии 0–0,02 м². По результатам определено, что метод согласованной фильтрации и алгоритм Wu в среднем показывают F-меру, равную 0,95. Худший результат имеет метод масштабного пространства кривизны.

Исследована работа визуального одометра. Определены ошибки *RPE* и *ATE*, которые составили 0,45 м и 0,05 м.

Исследовано время работы улучшенных алгоритмов EKF-SLAM. Результаты показали, что алгоритм EKF-SLAM имеет квадратичную вычислительную сложность, а улучшенные алгоритмы – линейную, что согласуется с теоретическими выкладками.

Проведено сравнение алгоритма EKF-SLAM с использованием разных типов датчиков. В результате использования двух датчиков точность позиционирования увеличилась на 1,95 % по метрике *RPE* и на 21,26 % по метрике *ATE* в сравнении с использованием только камеры, а также увеличилась на 2,23 % по метрике *RPE* и на 4,8 % по метрике *ATE* в сравнении с использованием только лидара. Точность здесь увеличилась за счет использования двух типов ориентиров. Каждый тип ориентира получен разным путем. Таким образом, появилась дополнительная оценка состояния системы, за счет которой уточняется месторасположение мобильной платформы на этапе корректировки. Во время этого этапа каждый ориентир обрабатывается последовательно и независимо от остальных.

Проведено исследование зависимости ошибки значений карты глубины от расстояния до объекта. Определено оптимальное расстояние между позициями съемки, равное 10 см.

ЗАКЛЮЧЕНИЕ

Основные выводы и результаты диссертационной работы можно сформулировать в следующем виде:

1. Предложен алгоритм одновременной локализации и построения карты с использованием камеры с объективом типа «рыбий глаз» и лазерной сканирующей системы и его модификации. Данный алгоритм позволяет строить траекторию движения мобильной платформы, карту проходимости и трехмерную карту окружающей среды.
2. В результате использования двух датчиков точность позиционирования увеличилось на 1,95% и 21,26% в сравнении с использованием только камеры, и на 2,23% и 4,8% – в сравнении с использованием только лидара по метрикам *RPE* и *ATE*.
3. Предложен метод представления данных лазерной сканирующей системы в виде комплекснозначного сигнала, позволяющего применять алгоритмы обработки телевизионных сигналов, использующих контурный анализ. Данный метод представления позволяет применять алгоритмы, инвариантные к переносу, масштабированию и повороту.
4. Рассмотрен ориентир типа «особая точка». Для этого типа ориентира определены матрицы измерения и якобиана с использованием сферической модели камеры.
5. Проведено исследование детекторов ориентиров с использованием контурного анализа. Численные результаты показывают, что метод согласованной фильтрации и алгоритм Wu в среднем показывают F-меру, равную 0,9. Худший результат имеет метод масштабного пространства кривизны.
6. Исследование времени работы улучшенных алгоритмов EKF-SLAM показывает, что алгоритм EKF-SLAM имеет квадратичную вычислительную сложность, а улучшенные алгоритмы – линейную.

7. Разработан алгоритм визуальной одометрии с использованием сферической модели камеры, учитывающий особенности движения мобильной платформы и камеры с объективом типа «рыбий глаз». Точность работы алгоритма составила 0,45 м и 0,05 м по метрикам *RPE* и *ATE*.
8. Исследование точности работы алгоритмов EKF-SLAM показывает, что при построении маленьких карт точность сравнительно одинакова, но при построении больших карт стандартный алгоритм EKF-SLAM проигрывает улучшенным.
9. Проведен эксперимент по сравнению различных модификаций алгоритма EKF-SLAM. Установлено, что наиболее эффективным с точки зрения скорости и точности является алгоритм EKF-SLAM с равномерным использованием ориентиров. Средняя ошибка у него меньше на 12,35%, чем у алгоритма EKF-SLAM, меньше на 10,34%, чем у алгоритма EKF-SLAM «разделяй и властвуй» и меньше на 4,87%, чем у алгоритма EKF-SLAM с адаптивным диапазоном наблюдения.
10. Сравнение различных алгоритмов SLAM показало, что алгоритм EKF-SLAM работает точнее на 75,68%, чем алгоритм LSD-SLAM, на 10% точнее, чем алгоритм RGBD-SLAM и на 77,78% хуже, чем алгоритм ORB-SLAM.

СПИСОК ЛИТЕРАТУРЫ

1. *Endres F., Hess J., Engelhard N., Cremers D., Burgard W.* An evaluation of the RGB-D SLAM system // IEEE International Conference on Robotics and Automation. 2012.

2. *Durrant-Whyte H., Fellow, Bailey T.* Simultaneous Localisation and Mapping (SLAM): Part 1 The essential Algorithms.

3. *Taketomi T.* Visual SLAM algorithms: a survey from 2010 to 2016 // IPSJ Transactions on Computer Vision and Applications. 2017.

4. *Zunino G.*, Simultaneous Localization and Mapping for Navigation in Realistic Environments.

5. *Dissanayake G., Durrant-Whyte H., Bailey T.* Computationally efficient solution to the simultaneous localization and map building (SLAM) problem // 2006.

6. *Кучерский Р., Манько С.* Алгоритмы локальной навигации и картографии для бортовой системы управления автономного мобильного робота // Известия Южного федерального университета. Технические науки, 2012.

7. *Sim R., Elinas P., Little J.* A Study of the Rao-Blackwellised Particle Filter for Efficient and Accurate Vision-Based SLAM // International Journal of Computer Vision volume. 2007.

8. *Парахневич А., Солонар А., Горшков С.* Фильтрация посредством выборки весовых коэффициентов. Обобщенный фильтр частиц (Particle Filter) // БГУИР, 2012.

9. *Антонов А.* Сканирующие лазерные дальномеры (LIDAR) // Современная электроника, 2016.

10. *Ros G., Ponsa D., Lopez A.* Visual SLAM for Driverless Cars: A Brief Survey // Proceedings of the 2012 IEEE Intelligent Vehicles Symposium Workshops, 2012.

11. *Klette R., Koschan A., Schluns K.* Computer vision: three-dimensional data from images. 1st edn, 1998.

12. *Nister D.* A minimal solution to the generalised 3-point pose problem. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition Vol. 1. pp 560–5671, 2004.

13. *Grisetti G., Kümmerle R., Stachniss C., Burgard W*. A tutorial on graph-based slam. Intell Transp Syst Mag IEEE 2(4):31–43, 2010.

14. *Kümmerle R., Grisetti G., Strasdat H., Konolige K., Burgard W*. g2o: A general framework for graph optimization. In: Proceedings of International Conference on Robotics and Automation. pp 3607–3613, 2011.

15. *Triggs B., McLauchlan P., Hartley R., Fitzgibbon A*. Bundle adjustment a modern synthesis. Vision algorithms: theory and practice. pp. 298–372, 2000.

16. *Дэвисон А., Reid I., Molton N., Stasse O.* "MonoSLAM: Real-time single camera SLAM," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 6, pp. 1052–1067, 2007.

17. *Civera J., Дэвисон А., Montiel J.* "Inverse depth parametrization for monocular SLAM," IEEE Transactions on Robotics, vol. 24, no. 5, pp. 932–945, 2008.

18. *Chiuso A., Favaro P., Jin H., Soatto S*. "Structure from motion causally integrated over time," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 4, pp. 523–535, 2002.

19. *Eade E., Drummond T*. "Scalable monocular SLAM," in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), vol. 1, New York City, USA, June 2006, pp. 469–476.

20. *Klein G., Murray D*. "Parallel tracking and mapping for small AR workspaces," in IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR), Nara, Japan, November 2007, pp. 225–234.

21. *Klein G., Murray D*. "Improving the agility of keyframe-based slam," in European Conference on Computer Vision (ECCV), Marseille, France, October 2008, pp. 802–815.

22. *Шевченко В*. Разработка методики сравнения методов визуальной одометрии для монокулярных камер с использованием открытых баз данных, 2018.

23. *Engel J., Cremers D.* LSD-SLAM: Large-Scale Direct Monocular SLAM, Large-Scale Direct SLAM with Stereo Cameras, 2017.
24. *Karan B.* “Calibration of kinect-type rgb-d sensors for robotic applications,” FME Transactions, vol. 43, no. 1, pp. 47–54, 2015.
25. *Jamiruddin R., Sari A., Shabbir J., Anwer T.* RGB-Depth SLAM Review, 2015.
26. *Song M., Watanabe H.* Robust 3D reconstruction with omni-directional camera based on structure from motion, 2018.
27. *Zhang Z.* A Flexible New Technique for Camera Calibration // Microsoft Research, One Microsoft Way, США, Рэдмонд. 1998.
28. *Li S.* Binocular spherical stereo[J]. Intelligent Transportation Systems, IEEE Transactions on, 2008, 9(4): 589-600.
29. *Scaramuzza D., Martinelli A., Siegwart R.* «A Flexible Technique for Accurate Omnidirectional Camera Calibration and Structure from Motion», Proceedings of IEEE International Conference of Vision Systems (ICVS'06), New York, January 5-7, 2006.
30. *Scaramuzza D.* Omnidirectional Vision: from Calibration to Robot Motion Estimation, ETH Zurich, PhD Thesis no. 17635. PhD Thesis advisor: Prof. Roland Siegwart. Committee members: Prof. Patrick Rives (INRIA Sophia Antipolis), Prof. Luc Van Gool (ETH Zurich). Chair: Prof. Lino Guzzella (ETH Zurich), Zurich, February 22, 2008.
31. *Civera, J., Дэвисон А., Montiel J.* Inverse Depth Parametrization for Monocular SLAM. IEEE Transactions on Robotics, 24(5):pages 932–945, October 2008.
32. *Дэвисон А.* Real-Time Simultaneous Localisation and Mapping with a Single Camera. In ICCV ’03: Proceedings of the Ninth IEEE International Conference on Computer Vision, page 1403. IEEE Computer Society, Washington, DC, USA, 2003. ISBN 0-7695-1950-4.

33. *Дэвисон, А., Reid I., Molton N., Stasse O.* MonoSLAM: Real-Time Single Camera SLAM. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 29(6):pages 1052–1067, 2007. ISSN 0162-8828.

34. *Li Y., Olson E.* Extracting general-purpose features from LIDAR data.

35. *Фурман Я., Кревецкий А., Передреев А., Роженцов А., Хафизов Ф., Егошина И., Леухин А.* Введение в контурный анализ / – М.: 2003. С. 369-379.

36. *Teh C., Chin R.* On the detection of dominant points on digital curves // IEEE Trans. Anal. Mach. Intell, 1989. pp. 859-872.

37. *Wu W.* An adaptive method for detecting dominant points // Pattern Recognition 36, 2003. pp. 2231-2237.

38. *Гэн К., Чулин Н.* Алгоритм навигации беспилотного летательного аппарата на основе улучшенного алгоритма одновременной локализации и картографирования с адаптивным локальным диапазоном наблюдения / Москва, 2017.

39. *Paz L., Jensfelt P., Tardos J., Neira J.* EKF SLAM updates in O(n) with Divide and Conquer SLAM.

40. *Stefano D., Mattoccia S.* "Fast stereo matching for the videt system using a general purpose processor with multimedia extensions" In Fifth IEEE International workshop on Computer Architecture for Machine Perception, Padova, Italy, Sept. 2000.

41. *Stefano D., Marchionni M., Mattoccia S., Neri G.* "A Fast Area-Based Stereo Matching Algorithm" 15th IAPR/CIPRS International Conference on Vision Interface Calgary, Canada, May 2002.

42. *Mozammel M., Chowdhury H., Mondal A, Bhuiyan A.* "3D Imaging Using Stereo Vision", Proc. of the 7th International Conference on Computer and Information Technology (ICCIT'2004), Dhaka, Bangladesh, pp. 307-312.

43. *Uddin M., Chowdhury M., Shioyama T., Mondal A.* "Fast window-based approach for stereo matching", Journal of Science, Jahangirnagar University, Vol. 27, N0. 1, 2004.

44. *Geiger D., Ladendorf B., Yuille A.* “Occlusions and binocular stereo,” International Journal of Computer Vision, vol. 14, pp. 211–226,1995.
45. *Mozammel M., Chowdhury H., Bhuiyan A.* A new approach for disparity map determnination, 2009.
46. *Grisetti G., Stachniss C., Burgard W*. Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters. IEEE Trans. Robot. 2007, 23, 34–46.
47. *Hess W., Kohler D., Rapp H., Andor D.* Real-time loop closure in 2D LIDAR SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 1271–1278.
48. *Chen M.; Yang S.; Yi X.; Wu D.* Real-time 3D Mapping using a 2D Laser Scanner and IMU-aided Visual SLAM. In Proceedings of the IEEE International Conference on Real-time Computing & Robotics IEEE, Kandima, Maldives, 1–5 August 2018.
49. *Tong Q., Peiliang L.; Shaojie S.* VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator. IEEE Trans. Robot. 2018, 34, 1004–1020.
50. *Li M.; Mourikis A.* 3-D motion estimation and online temporal calibration for camera-IMU systems. In Proceedings of the International Conference on Robotics and Automation (ICRA), Karlsruhe, Germany, 6–10 May 2013.
51. *Shi J., He B., Zhang L., Zhang J.* Vision-based real-time 3D mapping for UAV with laser sensor. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Deajeon, Korea, 9–10 October 2016; pp. 9–14.
52. *Lynen S.; Achtelik M., Weiss S., Chli M., Siegwart R.* A robust and modular multi-sensor fusion approach applied to MAV navigation. In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan, 3–8 November 2013.
53. *Oh T., Lee D., Kim H., Myung H.* Graph Structure-Based Simultaneous Localization and Mapping Using a Hybrid Method of 2D Laser Scan and

Monocular Camera Image in Environments with Laser Scan Ambiguity. Sensors 2015, 15, 15830–15852.

54. *López E., García S., Barea R., Bergasa L., Molinos E., Arroyo R., Romera E., Pardo S.* A Multi-Sensorial Simultaneous Localization and Mapping (SLAM) System for Low-Cost Micro Aerial Vehicles in GPS-Denied Environments. Sensors 2017, 17, 802.

55. *Nam T., Shim J., Cho Y.* A 2.5D Map-Based Mobile Robot Localization via Cooperation of Aerial and Ground Robots. Sensors 2017, 17, 2730.

56. *Zhang Z., Zhao R., Liu E., Yan K., Ma Y.* Scale Estimation and Correction of the Monocular Simultaneous Localization and Mapping (SLAM) Based on Fusion of 1D Laser Range Finder and Vision Data. Sensors 2018, 18, 1948.

57. *Sturm J., Engelhard N., Endres F., Burgard W., Cremers D.* "A benchmark for the evaluation of rgb-d slam systems," in Proc. of the International Conference on Intelligent Robot Systems (IROS), Oct. 2012.

58. *Oron S., Bar-Hille A., Avidan S.* Extended Lucas-Kanade Tracking. European Conference on Computer Vision, 2014.

59. *Durrant-Whyte H., Bailey T*. "Simultaneous localization and mapping. Part I," IEEE Robotics and Automation Magazine, vol. 13, no. 2, pp. 99–110, 2006.

60. *Bailey T*., *Durrant-Whyte H.* "Simultaneous localization and mapping (SLAM): part II," IEEE Robotics and Automation Magazine, vol. 13, no. 3, pp. 108–117, 2006.

61. *Buyval, A., Afanasyev, I., Magid, E*. Comparative analysis of ros-based monocular slam methods for indoor navigation, 2007.

62. *Mur-Artal R., Montiel J., Tards, J.* Orb-slam: A versatile and accurate monocular slam system. IEEE Transactions on Robotics, 31(5):1147–1163, 2015.

63. Rublee E., Rabaud V., Konolige K., Bradski, G. Orb: An efficient alternative to sift or surf. In 2011 International Conference on Computer Vision, pages 2564–2571, 2011.

64. *Scaramuzza, D. Fraundorfer, F*. Visual odometry. IEEE Robotics Automation Magazine, 18(4):80–92, 2011.

65. *Zhang J., Singh S*. Visual-lidar odometry and mapping: low-drift, robust, and fast. In 2015 IEEE International Conference on Robotics and Automation (ICRA), pages 2174–2181, 2015.
66. *Castellanos J., Neira J., Tard J*. Limits to the Consistency of EKF-based SLAM, 5th IFAC Symposium on Intelligent Autonomous Vehicles, 2004.
67. *Frese U*. Treemap: An O(logn) Algorithm for Indoor Simultaneous Localization and Mapping, Autonomus Robots, 21(2) pp. 103–122, 2006.
68. *Welch G., Bishop G*. An introduction to the kalman fllter. Technical Report TR 95-041, 2004.
69. *Lucas, B., Kanade, T.* An iterative image registration technique with an application to stereo vision. In: Proccedings of Imageing Understanding Workshop, 1981.
70. Davison A., Reid I., Molton N., Stasse O. Real-time single camera SLAM. IEEE Trans. on PAMI, 2007.
71. *Montiel J., Civera J., Davison A*. Unified inverse depth parametrization for monocular SLAM. In Robotics Science and Systems Conference. Philadelphia., 2006.
72. *Sola J*. Towards Visual Localization, Mapping and Moving Objects Tracking by a Mobile Robot: a Geometric and Probabilistic Approach. PhD thesis, LAAS-CNRS, 2007.
73. *Micusik B*. Two View Geometry of Omnidirectional Cameras. PhD thesis, Center for Machine Perception, Czech Technical University in Prague, 2004.
74. *Garcia R., Sotelo M., Parra I., Fernandez D., Gavilan M*, "2d visual odometry method for global positioning measurement," in IEEE International Symposium on Intelligent Signal Processing, WISP 2007, 2007.
75. *Aulinas J., Petillot Y., Salvi J., Lladó X*. The slam problem: a survey. In: Proceedings of Conference on Artificial Intelligence Research and Development: Proceedings of International Conference of the Catalan Association for Artificial Intelligence. pp. 363–371, 2008.

76. *Fuentes-Pacheco J., Ruiz-Ascencio J., Rendón-Mancha J.* Visual simultaneous localization and mapping: a survey. Artif Intell Rev 43(1):55–81, 2015.
77. *Montemerlo M., Thrun S., Koller D., Wegbreit B. Fastslam*. A factored solution to the simultaneous localization and mapping problem. In AAAI-2002, 2002.
78. *Strasdat H., Montiel J., Davison A.* "Real-time monocular slam: Why filter?" in Proc. IEEE Int. Conf. Robot. Automat., May 2010.
79. *Mouragnon E., Lhuillier M., Dekeyser F., Sayd P.* "Generic and real-time structure from motion using local bundle adjustment," Image and Vision Comput., vol. 27, 2009.
80. *Irschara A., Zach C., Frahm J., Bischof H*. "From structure-frommotion point clouds to fast location recognition," in Proc. IEEE Conf. Comput. Vision and Pattern Recog., 2009.
81. *Geiger A., Roser M., Urtasun R*. "Efficient large-scale stereo matching," in Proc. Asian Conf. Comput. Vision. Berlin, Heidelberg: Springer-Verlag, 2011.
82. *Geiger A., Ziegler J., Stiller C.* "Stereoscan: Dense 3d reconstruction in real-time," in Proc. IEEE Intell. Veh. Symp., 2011.
83. *Konolige K., Agrawal M.* "Frameslam: From bundle adjustment to real-time visual mapping," IEEE Trans. Robot., vol. 24, Oct 2008.
84. *Dissanayake M., Newman P., Clark S., Durrant-Whyte H., Csorba M.* A solution to the simultaneous localization and map building (SLAM) problem. IEEE Transactions on Robotics and Automation, 2001.
85. *Gen Ke Ke, Chulin N.A.* Algorithm of particle data association for SLAM based on improved ant algorithm. Nauka i obrazovanie: nauchnoe izdanie MGTU im. N.E. Baumana, 2015.
86. Caruso D., Engel J., Cremers D. Large-Scale direct SLAM for omnidirectional cameras. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, 2015, pp. 141–148.
87. Cui L., Fei W., A monocular ORB-SLAM in dynamic environments, China, 2019.

88. Antipov V.A., Kirnos V.P., Kokovkina V.A., Priorov A.L. The Object-Oriented Simultaneous Localization and Mapping on the Spherobot Platform // In: Favorskaya M., Jain L. (eds) Computer Vision in Control Systems—6. Intelligent Systems Reference Library, vol 182. Springer, Cham, 2020, pp. 165–175.
89. Антипов В.А., Коковкина В.А., Кирнос В.П., Приоров А.Л., Гурьянов Е.Д. Решение задачи одновременной локализации и картографирования с использованием модификаций расширенного фильтра Калмана // Радиотехнические и телекоммуникационные системы. 2019. № 4. С. 44–51.
90. Kokovkina, V.A., Antipov, V.A., Kirnos, V.P., Priorov, A.L. The algorithm of EKF-SLAM using laser scanning system and fisheye camera // Proceedings of Conference 2019 Systems of Signal Synchronization, Generating and Processing in Telecommunications, SYNCHROINFO 2019. Kazan, Russia, 2019.
91. Kirnos V., Antipov V., Priorov A., Kokovkina V. Landmarks detection by contour analysis in the problem of SLAM // Proceedings of the 24th Conference of Open Innovations Association FRUCT' 24.Moscow, Russia, 8-12 April 2019. P. 156–162.
92. Mur-Artal R., Montiel J. M., Tardos J. D. ORB-SLAM: a Versatile and Accurate Monocular SLAM System // IEEE Transactions on Robotics, 2015.
93. Ортиз-Гонзалез А., Кобер В.И., Карнаухов В.Н., Мозеров М.Г. Алгоритм построения трехмерной карты окружающей среды с использованием камеры глубины. // Информационные процессы. 2019. Т. 4, № 19. С. 355–365.
94. Барамия Д.А., Дьяков М.С., Кузиковский С.А., Лаврентьев М.М. // Автометрия. 2017. Т. 53, № 6. С. 77–83.
95. Neira J., Tardos J. D., Data association in stochastic mapping using the joint compatibility test. // IEEE Transactions on Robotics and automation.
96. Horn B. “Closed-form solution of absolute orientation using unit quaternions,” Journal of the Optical Society of America A, vol. 4, pp. 629–642, 1987.
97. R. Kummerle, B. Steder, C. Dornhege, M. Ruhnke, G. Grisetti, ¨ C. Stachniss, and A. Kleiner, “On measuring the accuracy of SLAM algorithms,” Autonomous Robots, vol. 27, pp. 387–407, 2009.

98. Kelly A., “Linearized error propagation in odometry,” Intl. Journal of Robotics Research (IJRR), vol. 23, no. 2, 2004.
99. Mur-Artal. R., Tardos J. D., ORB-SLAM: Tracking and Mapping Recognizable Features, 2014.
100. Павловский В.Е., Павловский В.В., Технологии SLAM для подвижных роботов: состояние и перспективы // Мехатроника, автоматизация, управление. 2016. Т. 17, № 6.

ПРИЛОЖЕНИЕ 1. Свидетельство о государственной регистрации программы для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2020619283

Программа SLAM с применением камеры с объективом типа "рыбий глаз" и лазерной сканирующей системы

Правообладатель: ***Антипов Владимир Алексеевич (RU)***

Автор: ***Антипов Владимир Алексеевич (RU)***

Заявка № 2020618831
Дата поступления 07 августа 2020 г.
Дата государственной регистрации
в Реестре программ для ЭВМ *14 августа 2020 г.*

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев