Автономная некоммерческая организация
высшего образования
«Университет Иннополис»

Autonomous noncommercial organization
of higher education
“Innopolis University”

АННОТАЦИЯ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
ПО НАПРАВЛЕНИЮ ПОДГОТОВКИ
09.03.01 – «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»

НАПРАВЛЕННОСТЬ (ПРОФИЛЬ) ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ
«ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»

Тема **Сравнение нелинейных оптимизационных методов для одновременной локализации и картографии**

Выполнил **Макаев Борис Олегович** подпись

Иннополис, Innopolis, 2019

Оглавление

Аннотация

Проблема одновременной локализации и картографии (SLAM) в настоящее время пользуется огромной популярностью; её применение варьируется от дополненной реальности до автономного автомобиля. Улучшение вычислительной мощности является причиной быстрого развития систем SLAM, но все еще остается важной проблема: как повысить вычислительную эффективность SLAM?

Годы исследований показали, что часть SLAM может быть сформулирована с помощью нескольких нелинейных задач оптимизации, и выбор метода решения этих проблем может значительно повлиять на производительность SLAM. Цель данного исследования - проанализировать роль нелинейных методов оптимизации в SLAM. В частности, возникает вопрос: как выбор конкретных методов решения задач оптимизации в SLAM влияет на производительность системы SLAM? Одновременная локализация и сопоставление состоит из двух основных задач: (1) построить и обновить карту среды (например, облако точек), (2) одновременно отслеживать местоположение агента относительно построенной карты.

Библиотека ORB-SLAM была выбрана в качестве конкретной системы интересов; он использует систему оптимизации g2o для решения пяти задач оптимизации с помощью метода Левенберга-Марквардта. Для оценки производительности различных конфигураций задач нелинейной оптимизации для ORB-SLAM было реализовано приложение с графическим интерфейсом пользователя (GUI); Приложение позволяет пользователю указать конфигурацию задач оптимизации в ORB-SLAM и запустить эксперимент, который даст результаты, которые могут оценить производительность текущей конфигурации. Кроме того, функциональность ORB-SLAM была расширена за счет тех же задач оптимизации, которые были сформулированы с использованием инфраструктуры оптимизации Ceres Solver.

Результаты представляют сравнение различных экспериментов с использованием приложения сравнения; они показывают, что различные нелинейные методы оптимизации влияют на производительность всей системы SLAM. Было сделано интересное наблюдение, что среди всех экспериментов в контексте экспериментов, которые были выполнены наиболее быстро с точки зрения среднего / среднего времени отслеживания, это наиболее неточно с точки зрения абсолютной ошибки позы (ошибка между сформированной и эталонной траекторией).

Глава 1

Введение

В настоящее время алгоритмы компьютерного зрения (CV) используются во многих областях робототехники, таких как автономные машины, беспилотные летательные аппараты (БПЛА). Число таких приложений, в которых используется компьютерное зрение, продолжает быстро расти в условиях огромной работы, проделанной множеством исследований. Многие алгоритмы, такие как алгоритмы одновременной локализации и картографирования (SLAM) и визуальной одометрии, были реализованы для того, чтобы позволить роботизированным системам выполнять визуальное отслеживание, оценивать их положение и траекторию относительно мира, строить трехмерную карту местоположения, в котором они находятся. Но возникает большая проблема, как повысить производительность таких алгоритмов; они выполняют тяжелые математические вычисления, и очень важно выбрать правильные методы для решения конкретных задач. В частности, часть проблем SLAM может быть сформулирована с использованием нелинейной оптимизации; такие проблемы включают минимизацию ошибки перепроецирования, настройку пучка и т. д. Различные методы решения задач нелинейной оптимизации могут выполняться по-разному в зависимости от типа задачи, которую он пытается решить; Таким образом, важно сравнить различные методы оптимизации, чтобы выбрать наиболее эффективный.

Этот тезис пытается установить связь между выбором нелинейных методов оптимизации в SLAM и общей производительностью алгоритмов SLAM. В частности, ORB-SLAM [1, 2] был выбран в качестве базовой системы SLAM, которая реализует свои задачи нелинейной оптимизации с использованием инфраструктуры оптимизации графов (g2o). Платформа оптимизации g2o реализует три нелинейных метода оптимизации: Левенберг-Марквардт, Гаусс-Ньютон и Доглег; В этом тезисе делается попытка расширить выбор методов оптимизации для ORB-SLAM путем реализации задач оптимизации с использованием инфраструктуры оптимизации Ceres Solver. Затем можно ответить на конкретный вопрос: влияет ли выбор структуры оптимизации, которая реализует решение задач оптимизации, на производительность ORB-SLAM?

Глава 2

Обзор Литературы

2.1 Нелинейная оптимизация

Нелинейная оптимизация широко используется в естественных науках и технике, особенно в области компьютерного зрения. Существует большое количество проблем, которые можно смоделировать в задаче оптимизации нелинейной формы.

Носедаль и Райт [3] описывают два класса существующих методов нелинейной оптимизации, которые концептуально дополняют друг друга. Методы *Line Search* сначала пытаются найти направление спуска, вдоль которого будет уменьшена целевая функция, а затем вычисляют размер шага, который решает, как далеко следует двигаться в этом направлении. Направление спуска может быть вычислено различными методами, такими как градиентный спуск, метод Ньютона и метод квази-Ньютона. Размер шага может быть определен либо точно, либо неточно. Методы *Trust-Region* аппроксимируют целевую функцию, используя модельную функцию (часто квадратичную) для подмножества пространства поиска, известного как область доверия.

Проблемы наименьших квадратов присутствуют во многих областях приложений, и они являются крупнейшим источником проблем оптимизации без ограничений. В задаче наименьших квадратов целевая функция имеет следующий вид:

$$f(x) = \frac{1}{2}\sum_{j=1}^{m} r_j^2(x), \tag{2.1}$$

где $r_j : \mathbb{R}^2 \Longrightarrow \mathbb{R}$ и обычно упоминается как *остаточный член*. Многие проблемы оптимизации в физике, компьютерном зрении, технике моделируются с использованием методов наименьших квадратов. Преимущество таких методов состоит в том, что такая специальная форма целевой функции f облегчает ее решение, чем общие задачи оптимизации без ограничений.

2.2 Оптимизационные библиотеки

2.2.1 Ceres библиотека

Ceres solver это библиотека нелинейной оптимизации, которая разработана и поддерживается Google. Он написан на C ++ как почти все библиотеки нелинейной оптимизации, потому что он требует высокоскоростных вычислений без больших накладных расходов. В настоящее время решатель Ceres может решать (1) нелинейные задачи наименьших квадратов с ограничениями границ и (2) общие задачи оптимизации без ограничений.

Для решения нелинейных задач наименьших квадратов Ceres реализует как методы области доверия, так и методы *line-search*. Методы *Trust-region* включают популярные методы Левенберга-Марквардта (с точным шагом [4] и неточный шаг [5]) и методы Dogleg. Методы *Line search* могут быть использованы только для задач без ограничивающих условий. К таким методам относятся *Steepest Descent*, *Nonlinear Conjugate Gradient*, *BFGS* и *LBFGS*.

2.2.2 Библиотека g2o

Джорджо Гризетти и соавт.[6, 7] представили свою новую систему оптимизации g2o для решения нелинейной задачи наименьших квадратов в графе или гиперграфе пространства. Они утверждают, что такой способ решения проблемы удобен при попытке оптимизировать решения SLAM или другие системы роботов. g2o реализует два алгоритма для решения нелинейных задач оптимизации наименьших квадратов: Левенберга-Марквардта и Гаусса-Ньютона, а также изгиб Пауэлла. На рисунке **??** мы видим диаграмму классов каркаса оптимизации g2o, которая показывает внутреннюю структуру библиотеки.

2.3 ORB-SLAM

ORB-SLAM

В 2015 году Рауль Мур-Арталь и др. [1] представили свою недавно разработанную функциональную монокулярную систему SLAM. Он реализует такие функции, как отслеживание, картографию, релокализацию и закрытие цикла.

Внешний интерфейс ORB-SLAM имеет функцию извлечения функций ORB, которая работает намного быстрее, чем функции извлечения SIFT, SURF и A-KAZE. Серверная часть состоит из трех потоков, которые работают параллельно: отслеживание, локальное картографирование и замыкание цикла. *Поток отслеживания* локализует камеру с каждым кадром и решает, когда вставить новый ключевой кадр. *Поток локального картографирования* обрабатывает новые ключевые кадры и выполняет локальную

настройку связки для достижения оптимальной реконструкции в окружении позы камеры. Локальное сопоставление также отвечает за отбор избыточных ключевых кадров. *Поток замыкания цикла* ищет циклы с каждым новым ключевым кадром. Если замыкание обнаружено, преобразования подобия должны быть вычислены, чтобы сообщить о дрейфе, накопленном в петле. Затем обе стороны цикла выровнены, а дублированные точки выровнены. После этого выполняется оптимизация графа по ограничениям подобия для достижения глобальной согласованности. ORB-SLAM использует нелинейный метод наименьших квадратов Левенберга-Марквардта, реализованный в g2o, для решения всех задач оптимизации.

2.4 SLAM Датасеты

Как обсуждалось ранее, основная идея алгоритмов SLAM состоит в том, чтобы построить исходную карту (облако точек), затем собрать данные с датчиков (изображения с камер, другие измерения из IMU и т.д.), затем локализовать робота по карте и затем уточнить (оптимизировать) карту соответственно. Иногда исследователи ищут только тестирование своих алгоритмов SLAM вне роботизированной среды; гораздо проще и удобнее тестировать их алгоритмы-прототипы на уже существующих наборах данных. Таким образом, исследователи со всего мира создают различные наборы данных, которые могут содержать (1) последовательности изображений для моно / стереокамеры, (2) показания и измерения с IMU, (3) траекторию наземной истинности для оценки производительности алгоритма SLAM, (4) файлы калибровки. Основная цель создания таких наборов данных для оценки алгоритмов SLAM - предоставить исследователям со всего мира средства оценки алгоритмов друг друга. Это помогает исследователям сравнивать различные алгоритмы SLAM в зависимости от производительности определенного набора данных. Группа компьютерного зрения факультета информатики Технического университета Мюнхена (TUM) предоставляет большое количество наборов данных для монокулярного SLAM, RGB-D SLAM, визуально-инерционного SLAM и т.д. [8]. Например, наборы данных для RGB-D содержат последовательность изображений RGB, данные о глубине, которые были получены с камеры, данные о траектории земной поверхности и данные акселерометра [9]. Проект Технологического института Карлсруэ и Технологического института Тойота в Чикаго (KITTI) предоставляет множество наборов данных, которые были получены с помощью камеры, установленной на автомобиле [10, 11]. Еще один большой набор данных, собранных М. Бурри и соавторами [12] для EuRoC соревнования. Эти наборы данных содержат данные, собранные с Micro Aerial Vehicle (MAV), и включают данные из машинного зала и т.д.

Глава 3

Методология

В этой главе особое внимание будет уделено вопросам проектирования системы и реализации программы для сравнения различных методов нелинейной оптимизации. Подход для сравнения нелинейных методов оптимизации, который был выбран, является подходом изменения существующей системы SLAM. Для успешного решения такой задачи было решено: (1) выбрать хорошую существующую библиотеку SLAM, (2) проанализировать используемые по умолчанию методы нелинейной оптимизации, (3) выбрать и реализовать новые методы нелинейной оптимизации, используя другую платформу оптимизации, (4) разработать простую приложение с графическим интерфейсом, которое дало бы возможность легко настроить конфигурацию методов оптимизации SLAM, (5) сохранить метрики, которые оценивают производительность алгоритма SLAM в некоторой базе данных, и (6) предоставить средство для визуализации метрик, ошибок для определенных конфигураций. Подходы к проектированию этих проблем будут подробно описаны ниже в этой главе.

3.1 Выбор SLAM системы

Как было сказано ранее в главе 2, необходимо выбрать внешний интерфейс, который извлекает и обрабатывает объекты из изображений, и внутренний, который устраняет ошибки перепроецирования и решает другие проблемы оптимизации, такие как глобальные или локальные настройка пакета и т. д. Существует множество различных систем SLAM, экстракторов функций и решений для оптимизации, которые мы можем выбрать. Для этой диссертации в качестве базовой системы SLAM была выбрана библиотека ORB-SLAM2 [2]. Он имеет собственный экстрактор фич ORB, который служит в качестве входной части системы SLAM; Библиотека оптимизации g2o используется как часть серверной части, которая решает проблемы оптимизации.

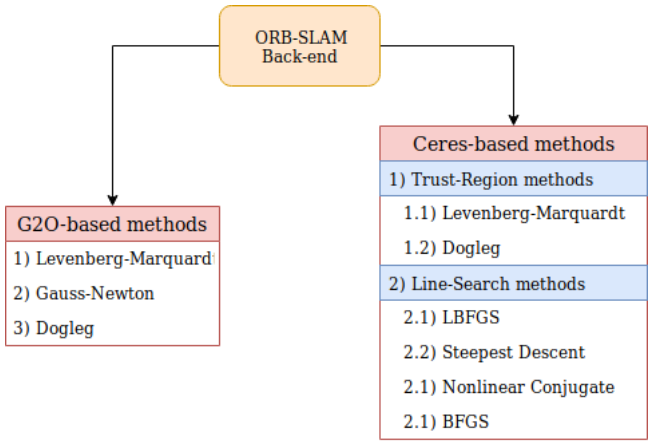


Рис. 3.1: Схема методов внутренней оптимизации ORB-SLAM после расширения

3.2 Расширение оптимизации ORB-SLAM

3.2.1 Выбор Нелинейных Оптимизационных Методов

Как обсуждалось в предыдущей главе, у ceres solver есть 2 типа методов решения задач нелинейной оптимизации: textit Trust-region и textit Line Search. Методы textit Trust-region включают в себя такие методы, как Levenerg-Marquardt и Dogleg. С другой стороны, методы textit Line search включают в себя методы наискорейшего спуска, нелинейного сопряженного спуска, BFGS и LBFGS. Все эти методы являются гибкими с точки зрения конфигурации; Можно легко настроить выбранный алгоритм в соответствии с их потребностями, используя структуру опций языка Си.

Преимущество этой реализации перед g2o состоит в том, что библиотека Ceres дает возможность настроить, как именно будут решаться методы Левенберга-Марквардта или Доглега; Существует несколько типов каждого из этих методов, которые могут по-разному влиять на решение, например, Левенберг-Марквардт с неточным/точным шагом, традиционный Пауелл-доглег или субпространственный доглег. Существует гораздо больше настраиваемых переменных, которые могут повлиять на производительность процесса оптимизации, например, радиус области, количество итераций и другие гиперпараметры, которые можно настроить.

3.2.2 Проектирование Расширений Оптимизации

Рисунка 3.1 показывает конечное количество оптимизационных методов. По-умолчанию, ORB-SLAM использует библиотеку оптимизации g2o. Расширением такой архитектуры является включение библиотеки Ceres в среду ORB-SLAM. Далее для каждой задачи оптимизации, определенной для

ORB-SLAM, соответствующая модель написана с использованием решателя Ceres. Позже параметры библиотек для решения задач оптимизации будут легко скорректированы в приложении сравнения, которое также будет разработано в рамках дипломной работы.

3.3 Дизайн приложения сравнения

Прежде всего, необходимо определить показатель производительности задачи нелинейной оптимизации. На самом деле, в ORB-SLAM может быть несколько показателей эффективности. Наиболее очевидным является время выполнения задачи оптимизации. Довольно просто оценить время выполнения каждой задачи оптимизации, просто поместив таймер в код внутри соответствующих мест, откуда вызывается функция оптимизации. Еще одним критерием эффективности, который можно выбрать, является общая ошибка между наземной истинной траекторией набора данных и восстановленной траекторией. Еще одним важным показателем производительности является потребление ресурсов процессора; разные методы оптимизации используют разные ресурсы процессора для решения соответствующих задач. Вот причина для нахождения идеальных методов, которые были бы наилучшими для наших проблем.

Мотивация для проектирования и разработки такой структуры связана с природой сравнения различных методов нелинейной оптимизации в ORB-SLAM. Есть 5 задач оптимизации, которые определены для решения в ORB-SLAM. Учитывая огромное количество методов оптимизации для каждой из этих проблем оптимизации, возникает проблема - существует довольно много комбинаций различных методов оптимизации, используемых вместе. Если каждый раз компилировать ORB-SLAM с использованием разных методов оптимизации, это займет много времени. Вот почему идея обобщения библиотеки ORB-SLAM для различных методов нелинейной оптимизации и создания приложения с графическим интерфейсом для настройки конфигурации системы SLAM действительно интересна.

На рисунке 3.2 предложена прототип структуры приложения сравнения. Конечный результат должен представлять пользовательское приложение, которое дало бы возможность пользователю тестировать различные конфигурации ORB-SLAM; после постановки экспериментов (параметров и т.д.) их можно запустить, а результаты следует сохранить для исследовательских целей.

3.3.1 Приложение с графическим интерфейсом

Для реализации приложения с графическим интерфейсом были использованы Qt5 QWidgets, которые позволяют проектировать и создавать масштабируемые приложения с графическим интерфейсом. Идея этого приложения с графическим интерфейсом состоит в том, что пользователь может точно указать, как будет решаться задача нелинейной оптимизации. Основная

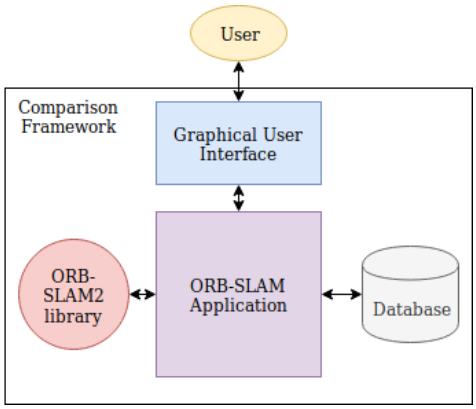


Рис. 3.2: Разработка прототипа высокого уровня для приложения сравнения

конфигурация касается способа решения таких задач; пользователь также может указать расширенные настройки, которые позволяют точно настроить работу алгоритма. Расширенные конфигурации будут отличаться для каждой библиотеки оптимизации g2o и ceres.

Рисунок 3.3 представляет окно запуска, которое дает пользователю возможность запустить ORB-SLAM с определенной конфигурацией. Для каждой задачи оптимизации пользователь может выбрать библиотеку, в которой будут сформулированы эти задачи оптимизации, методы оптимизации, расширенную конфигурацию для задач на основе Ceres и набор данных, в котором ORB-SLAM будет выполнять локализацию и отображение.

Основная идея заключается в том, что после того, как пользователь нажмет кнопку «Отправить задачу», приложение сгенерирует файл JSON из формы, заполненной пользователем. Затем будет запущено приложение ORB-SLAM, которое инициализирует систему SLAM с использованием файла конфигурации JSON. После окончания работы ORB-SLAM приложение с графическим интерфейсом заполняет базу данных информацией о завершенном эксперименте; она будет хранить все результаты, необходимые для визуализации. Кнопка «Эксперименты» открывает новое окно, которое используется для визуализации экспериментов и проверки результатов работы ORB-SLAM с определенной конфигурацией.

На рисунке 3.4 показана вторая страница графического интерфейса, которая предоставляет пользователю несколько возможностей:

1. **Посмотреть результаты** из прошлых экспериментов. Функциональность позволяет также проверять эксперименты и визуализировать их.

2. **Визуализация траектории** эта кнопка запускает инструмент визуализации, который покажет траектории выбранных экспериментов.

3. **APE визуализация** - абсолютная погрешность позы показывает раз-

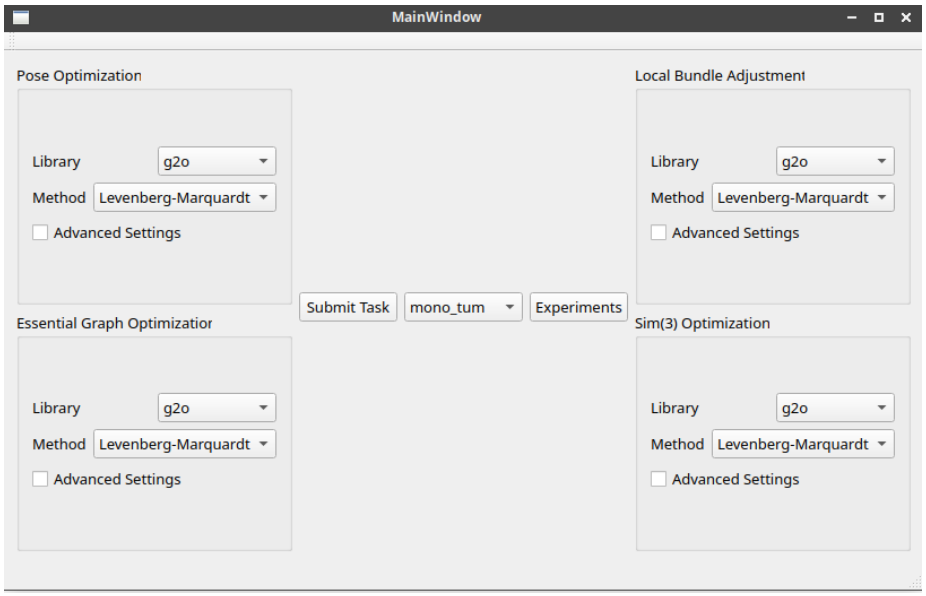


Рис. 3.3: Начальная страница GUI, позволяющая настроить и запустить конкретный эксперимент

ницу мужду сгенерированной траектории и референсной.

4. **Объединенная статистика** Кнопка сохранит несколько сохраненных результатов оценки APE в один файл и отобразит статистику среди выбранных экспериментов.

Все визуализации обрабатываются *evo* [13], библиотекой для визуализации одометрии и SLAM, написанной на Python. Есть 3 важных программы, которые предоставляются библиотекой Evo:

- *evo_traj* - инструмент для визуализации траекторий. Его также можно использовать для сравнения эталонной траектории с созданной.

- *evo_ape* - инструмент для оценки сгенерированной траектории по эталонной траектории на основе определенной метрики: абсолютная ошибка поза.

- *evo_res* - унифицирует несколько результатов расчета метрики и дает хорошие графики для сравнения траекторий.

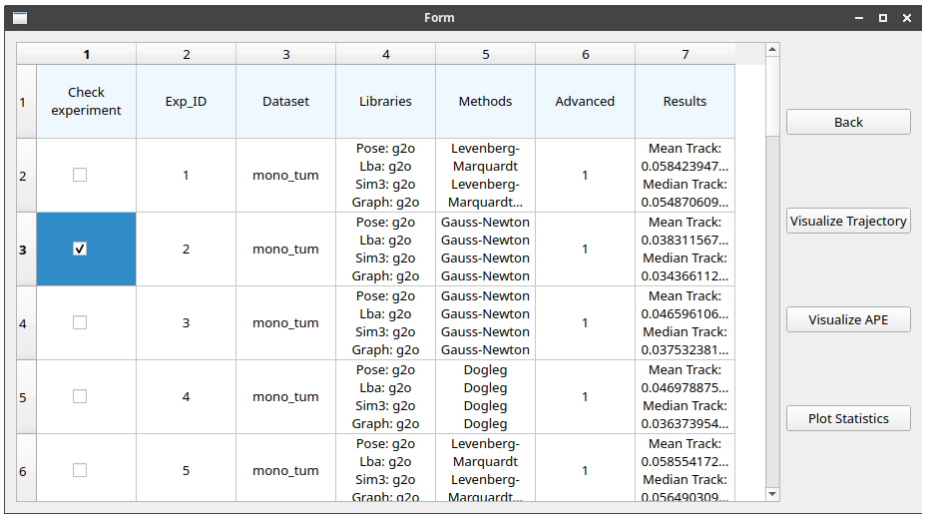


Рис. 3.4: Вторая страница графического интерфейса, которая позволяет пользователю увидеть результаты прошлых экспериментов и визуализировать оценку

3.3.2 ORB-SLAM Приложение

Стандартные приложения ORB-SLAM создаются вместе с библиотекой в процессе компиляции. Каждый пример связан с конкретным набором данных и конкретными настройками датчика (будь то камера стерео, моно или RGB-d). Прежде всего, каждый из этих примеров инициализирует систему SLAM: загружать файлы конфигурации, файлы калибровки, флаги режима камеры; инициализация системы SLAM должна быть расширена для обработки файла JSON с дополнительными конфигурациями для задач оптимизации SLAM. Во время инициализации система SLAM запускает 3 дополнительных потока: один для замыкания контура, второй для LBA, третий для визуализации облака точек, траектории камеры и отслеживаемых точек, наложенных на исходную последовательность изображений. Основной поток системы SLAM для каждого изображения выполняет отслеживание. Каждый из этих примеров реализует небольшие функции, которые помогают обрабатывать соответствующие наборы данных в функцию слежения за SLAM.

3.3.3 ORB-SLAM2 библиотека

Библиотека ORB-SLAM содержит функциональность, которая обеспечивает работу алгоритма SLAM. Чтобы приложение работало, необходимо расширить некоторые функции, например, ORB-SLAM Optimizer должен быть дополнен новыми реализациями для задач оптимизации с использованием библиотеки Ceres Solver, добавить функциональность, которая может ана-

лизировать файл конфигурации JSON и выполнять оптимизацию в соответствии с этим файлом конфигурации JSON.

3.3.4 База Данных

Чтобы выполнить оценку различных конфигураций, должна быть разработана и внедрена некая база данных, в которой будут храниться результаты эксперимента. Было решено разработать реляционные таблицы, которые будут включать метрики, которые будут сравниваться между собой позже. Простой реляционный дизайн базы данных, представленный на рисунке 3.5, показывает, как данные будут сохраняться при выполнении эксперимента (запуск алгоритма SLAM с новой конфигурацией для конкретного набора данных). Позже эти данные могут быть использованы для визуализации в приложении с графическим интерфейсом. Иерархия высокого уровня подчеркивает, какие данные мы хотим хранить; каждая сущность представляет будущую таблицу. *Experiment* table - это основная таблица, которая в основном состоит из ссылочных ключей на другие таблицы; она должен содержать информацию о том, какой датасет мы используем, какой набор методов оптимизации был выбран, какие библиотеки будут использоваться для решения задач оптимизации, какие результаты сохранены (среднее время отслеживания, среднее время отслеживания, абсолютная ошибка позы между эталонной траекторией и сгенерированным траектория). Также стоит упомянуть, что таблица *Experiment* ограничена ссылочными (внешними) ключами других таблиц, что гарантирует, что вставка данных в таблицу экспериментов будет последней операцией. В главе «Реализация» 4 будет представлен окончательный конкретный дизайн базы данных.

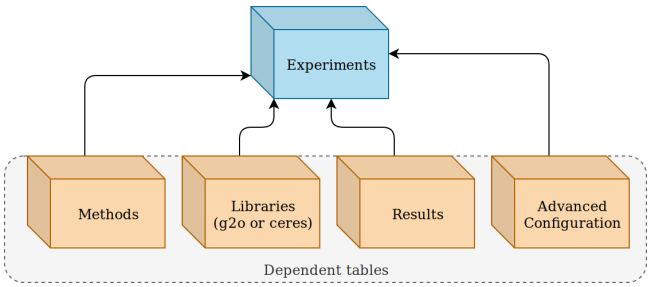


Рис. 3.5: Схема высокого уровня для базы данных, в которой будут храниться результаты экспериментов

3.4 Проектирование экспериментов

Поскольку существует множество комбинаций экспериментов из-за большого количества переменных, которые можно переставить, нужно сформировать только конкретные серии экспериментов. Формирование таких серий

экспериментов, как уже упоминалось выше, должно проводиться на основе экспериментов, которые представляют интерес для исследования. Ниже приведен список таких интересных вопросов, которые помогут сформировать серии экспериментов; Результаты этих экспериментов могут позже ответить на эти вопросы. Также в некоторых экспериментах мы хотим иметь образец, с которым будут сравниваться результаты экспериментов. В случае библиотеки ORB-SLAM2 таким образцом будет исходная конфигурация оптимизатора; Как уже было сказано, исходная конфигурация для каждой задачи оптимизации использует библиотеку оптимизации g2o с набором методов Левенберга-Марквардта для решения каждой из задач нелинейной оптимизации.

1. Для каждого набора данных, какая библиотека в целом работает лучше с точки зрения конкретной метрики (время отслеживания, ошибка траектории и т.д.)?

2. Для одной из четырех проблем оптимизации, какой метод оптимизации работает лучше в области фиксированной библиотеки? (Пример: для задачи оптимизации позы, какой метод оптимизации, Левенберг-Марквардт или Доглег, является лучшим, если оптимизатор использует только задачи оптимизации на основе Ceres?)

3. Насколько точная настройка расширенной конфигурации для методов оптимизации влияет на производительность алгоритма?

Абстрактная схема создания эксперимента показана на рисунке 3.6.

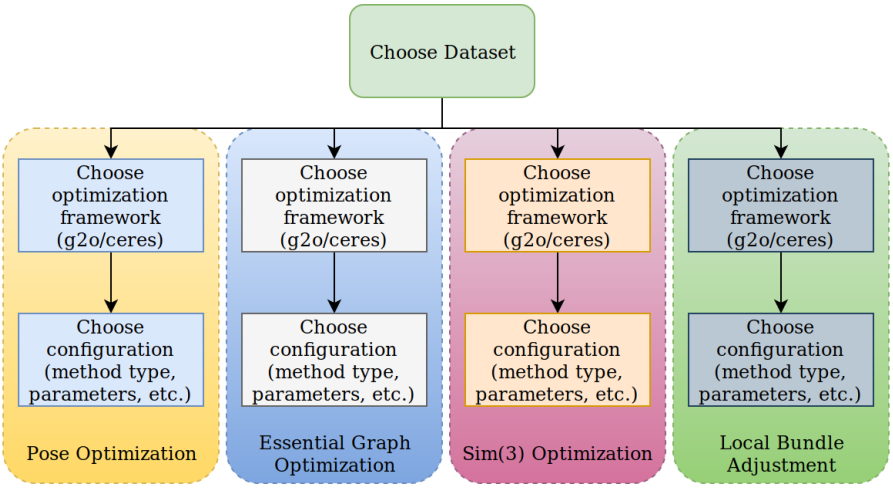


Рис. 3.6: Создание эксперимента

Глава 4

Имплементация

4.1 Настройка системы

4.1.1 Аппаратура

Характеристики аппаратуры представлены в Рисунке 4.1.

MSI GX60-1AC	
Операционная система	Ubuntu 18.04 LTS
CPU	AMD A10-4600m
GPU	Radeon HD7970M
RAM	8 GB DDR3

Таблица 4.1: Характеристики ноутбука MSI GX60

4.1.2 ORB-SLAM Установочный процесс

Перед выполнением любых тестов или изменением кода ORB-SLAM2 необходимо установить следующие зависимости: компилятор C++11 или C++0x; Библиотека Pangolin для визуализации [14]; OpenCV 3.4.5 [15]; Библиотека DBoW2 [16]; G2o [7]; Eigen3 [17]. Затем пакет ORB-SLAM2 должен быть установлен сам [2].

4.1.3 QtCreator Установочный процесс

Чтобы разработать приложение сравнения, необходимо использовать соответствующую среду графического интерфейса. В главе методологии 3 было предложено использовать Qt5 QWidgets. QtCreator - это массивная IDE, которая помогает разработчикам легко создавать приложения с графическим интерфейсом, так как включает в себя режим Designer, которая позволяет создавать приложения QWidget с помощью перетаскивания элементов на

канвас. Тогда все зависимости между формами, кнопками и другими элементами должны быть написаны на C++. Установщик QtCreator можно скачать через их веб-страницу [18].

4.2 Расширяя оптимизацию ORB-SLAM

4.2.1 Обзор кода ORB-SLAM

После установки ORB-SLAM нам нужно найти место, где мы должны изменить код, чтобы протестировать различные методы оптимизации; также код должен быть изменен в определенных местах, чтобы использовать динамически скомпилированную библиотеку ORB-SLAM с приложением сравнения. Интересные файлы: **Optimizer.cpp/.h**. По сути, это место, где все проблемы оптимизации изначально формулируются с использованием библиотеки оптимизации g2o. Как уже обсуждалось, ранее в ORB-SLAM2 были определены пять задач нелинейной оптимизации. Они представлены в виде пяти соответствующих функций в этом файле. Работая над диссертацией, только 4 из них были сформулированы с использованием библиотеки оптимизации Ceres, поэтому в конечном приложении мы будем использовать только четыре проблемы, имплементированных с Ceres: LocalBundleAdjustment, PoseOptimization, OptimizeSim3 и EssentialGraphOptimization.

4.2.2 Имплементация расширений

В главе 3 были предложены модели для реализации методов оптимизации на основе Ceres. Задачи оптимизации на основе Ceres сформулированы почти так же, как в g2o, но они более настраиваемы. Для определения задачи нелинейной оптимизации с использованием библиотеки Ceres должны быть определены соответствующие функции потерь и функции стоимости. Функция потерь обычно называется устойчивым ядром, которое обрабатывает посторонние объекты (outliers). В g2o функция потерь - это либо потеря Хьюбера, либо ее вообще нет. Функции затрат - это ошибки, которые были определены ранее. Для каждой функции дифференциации затрат должен быть предусмотрен модуль. В случае библиотеки Ceres, он предоставляет модуль автоматического дифференцирования.

4.3 Имплементация Приложения Сравнения

GUI разработан и реализован с использованием Qt QWidgets и написан на языке программирования C++. На рисунке 4.1 показана диаграмма UML для приложения сравнения. Были реализованы следующие классы: **MainWindow** - этот класс реализует основной элемент GUI, который позволяет пользователю настраивать параметры для текущей конфигурации

ORB-SLAM, например, установить конкретный метод нелинейной оптимизации, расширенную настройку и т.д. Главное окно имеет две кнопки: первая отправляет задачу в ORB-SLAM для выполнения, вторая переключает окно в окно результатов; **ResultsWindow** - этот класс реализует окно «Результаты», которое позволяет пользователю визуализировать траекторию, абсолютную ошибку позы (APE) и искать предыдущие результаты экспериментов. Имеет 4 кнопки: одну для переключения окна обратно в главное окно и 3 кнопки для визуализации результатов; **AdvancedConfigCeres** - инкапсулирует расширенные настройки для решателя Ceres, которые могут быть установлены пользователем; **CeresTrustRegion** - это часть расширенной конфигурации для Ceres. Он хранит поля для алгоритмов области доверия; **CeresLineSearch** - это также часть расширенной конфигурации для Ceres, которая хранит поля и информацию об алгоритмах поиска строк; **AdvancedConfigG2O** - он должен инкапсулировать расширенные настройки для G2O, но в настоящее время он ничего не хранит, потому что методы оптимизации g2o настраиваются только по имени метода в этом тезисе (Левенберг-Марквардт, Гаусс-Ньютон или Dogleg).

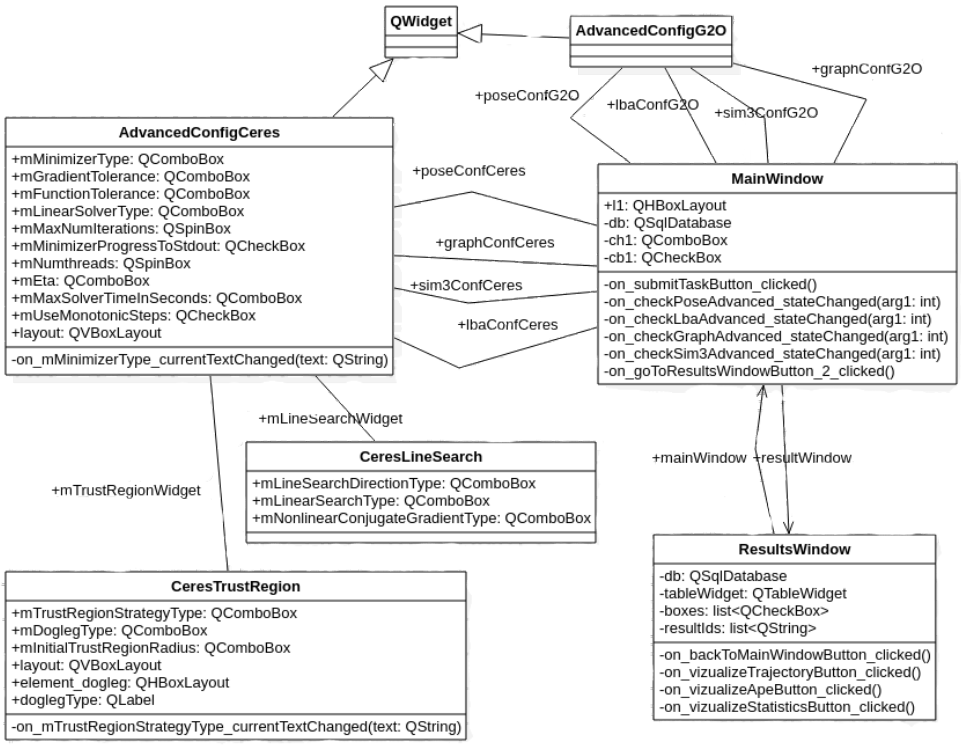


Рис. 4.1: Конечная схема UML для приложения сравнения.

4.3.1 База данных

В предыдущей главе обсуждалась архитектура базы данных высокого уровня; Реализация такой базы данных должна начинаться с выбора системы управления базами данных (СУБД). Окончательный выбор - PostgreSQL, очень мощная СУБД с открытым исходным кодом, которая позволяет разработчику тщательно проектировать и реализовывать необходимую структуру базы данных. Рисунок 4.2 определяет структуру базы данных, которая использовалась при реализации приложения сравнения. Таблица *Libraries* состоит из всех перестановок библиотек для каждого метода.

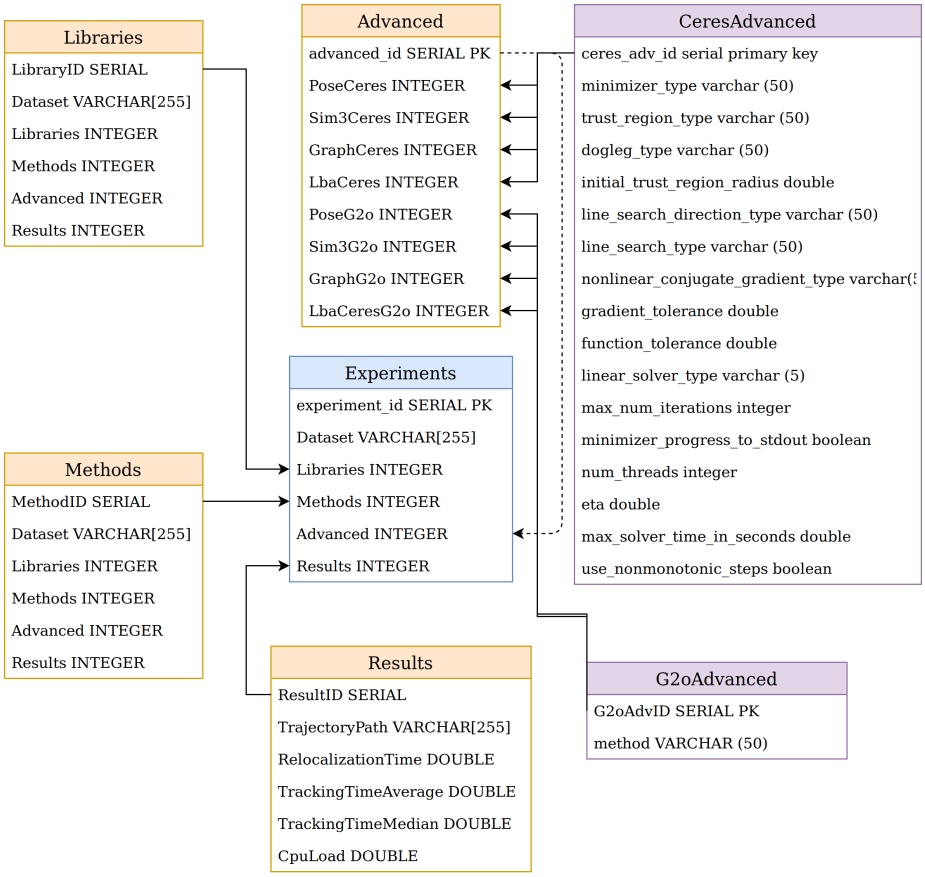


Рис. 4.2: Конечная схема базы данных, которая будет хранит результаты экспериментов.

Код для создания этих таблиц может быть найден в Appendix A.

Глава 5

Результаты и Обсуждение

5.1 Приложение сравнения

Этот раздел покажет пример использования того, как точно использовать приложение сравнения, шаг за шагом. Затем будет запущен пакет экспериментов, чтобы получить репрезентативные результаты, которые впоследствии будут использоваться для оценки различных конфигураций ORB-SLAM; они будут сравниваться с использованием метрик, которая обсуждалась ранее в главе 3, например Потребление ЦП, производительность по времени для различных задач ORB-SLAM в миллисекундах, абсолютная ошибка положения (APE) между сгенерированной траекторией и траекторией наземной истины (также называемой эталонной) и т.д. Этот раздел в основном оценивает само приложение GUI: проблемы проектирования и возможные улучшения будут обсуждаться позже.

5.1.1 Тестирование приложения

В этом подразделе будет описан типичный вариант использования приложения, который показывает все функциональные возможности приложения сравнения. На рисунке 5.1 показаны окна, которые дают возможность настроить каждую из задач оптимизации 4 ORB-SLAM. После установки Gauss-Newton метода оптимизации для проблемы Essential graph optimization для g2o, эксперимент можно начать, нажав кнопку «Отправить задачу».

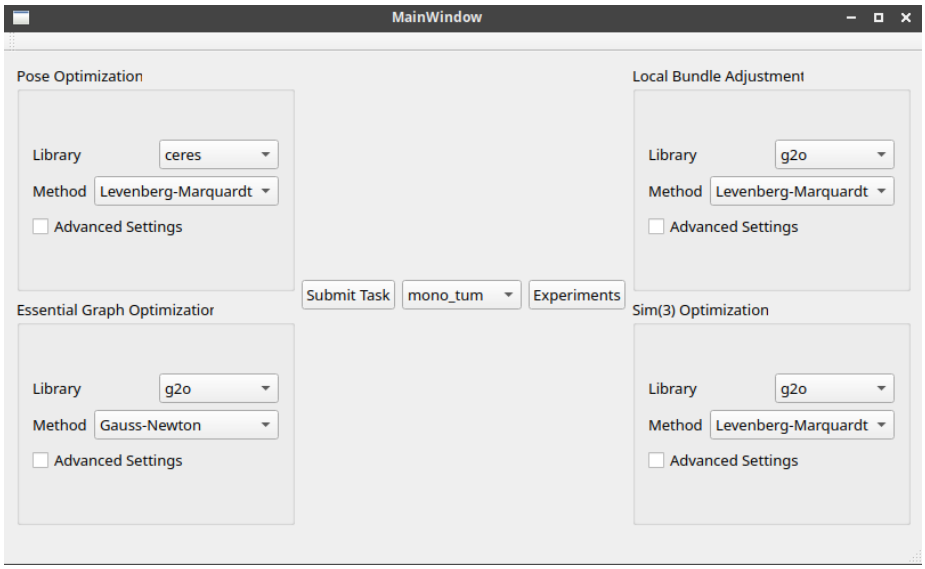


Рис. 5.1: Начальное главное меню. Существенный граф был выбран для решения методом Гаусса-Ньютона g2o.

После нажатия кнопки отправки, приложение сравнения собирает файл json с конфигурацией и запускает процесс приложения ORB-SLAM. Затем ORB-SLAM инициализируется с новой конфигурацией и запускает окна визуализации: сначала для просмотра облака точек и траектории, затем для отображения отслеживаемых точек на текущем изображении. На рисунке 5.2 показаны окна визуализации ORB-SLAM.

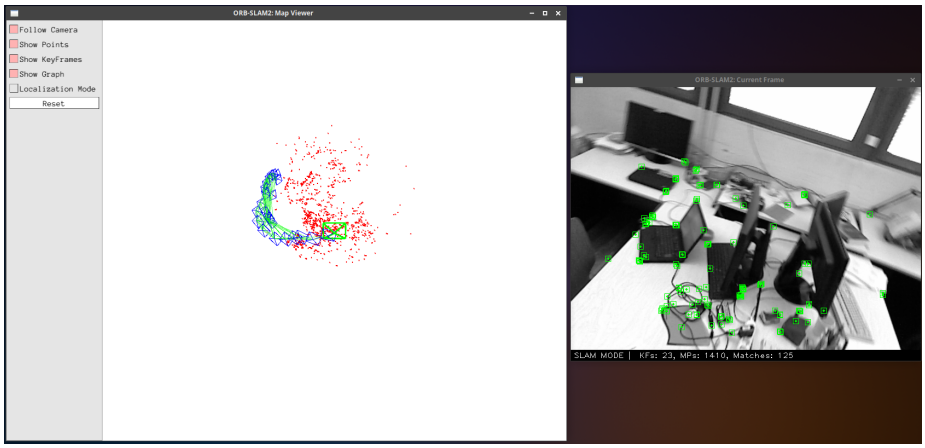


Рис. 5.2: ORB-SLAM визуализация отслеживания траектории и локализации

После того, как ORB-SLAM завершит свою работу, результаты сохраняются в формате json, а затем приложение сравнения сравнивает их и, используя всю информацию об эксперименте, которое оно имело, вставляет всю эту информацию в базу данных. Теперь пользователь может нажать кнопку «Эксперименты» в главном окне, которая приведет его в окно «Результаты».

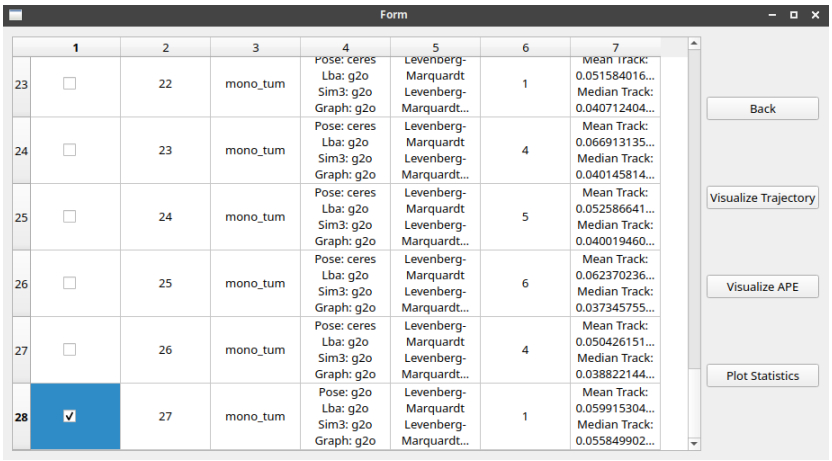
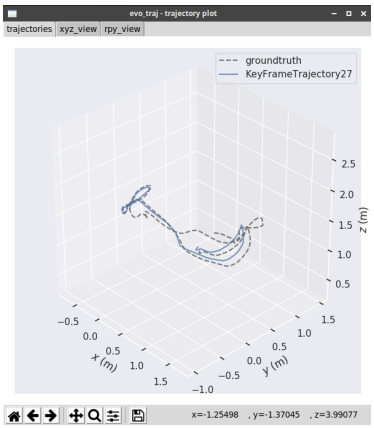
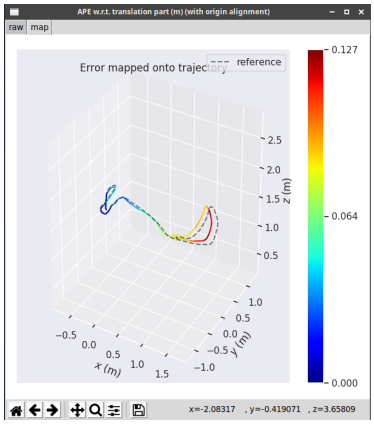


Рис. 5.3: Второе окно, которое показывает все предыдущие результаты и дает возможность проверить эксперименты и визуализировать их результаты

Если пользователь нажимает кнопку «Визуализация траектории» в окне результатов, то будет показан результат запуска *evo_traj* с проверенными экспериментами. В частности, он покажет отмеченные траектории по сравнению с истинной траекторией.



(a) График, который сравнивает сгенерированную траекторию и истинную траекторию



(b) График, показывающий абсолютную ошибку позы между сгенерированной траекторией и эталонной траекторией

Рис. 5.4: Показать график экспериментов

На рисунке 5.5 показано главное окно, в котором указаны расширенные настройки для задачи оптимизации позы на основе Ceres.

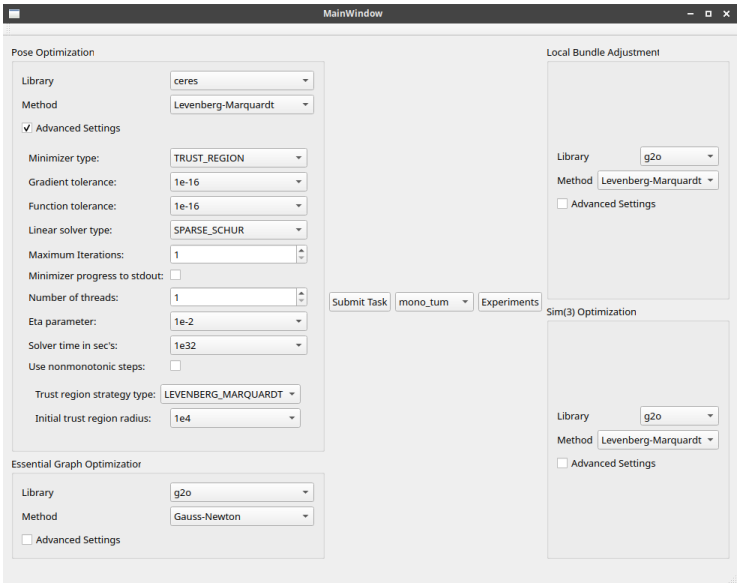


Рис. 5.5: Начальное главное меню. Оптимизация поз решается с помощью расширенных настроек Ceres

id	Graph	Sim(3)	LBA	Pose	Mean (ms)	Median (ms)
1	LM	LM	LM	LM	58	55
2	GN	LM	LM	LM	59	58
3	DL	LM	LM	LM	59	58
4	LM	GN	LM	LM	59	55
5	LM	DL	LM	LM	62	61
6	LM	LM	GN	LM	63	62
7	LM	LM	DL	LM	74	71
8	LM	LM	LM	GN	60	59
9	LM	LM	LM	DL	76	79
10	GN	GN	GN	GN	Track Lost	Track Lost
11	DL	DL	DL	DL	Track Lost	Track Lost

Таблица 5.1: LM - Левенберг-Марквардт; **GN** - Гаусс-Ньютон; **DL** - Доглег; Зеленый цвет означает лучшие результаты с точки зрения медианы и среднего времени отслеживания.

5.2 Запуск экспериментов

Теперь эксперименты должны быть запущены. Как было описано в главе методологии 3, лучше подумать о конкретном вопросе, на который можно было бы ответить с помощью приложения сравнения. Поскольку графики результатов занимают много места, было решено убрать их в главу Приложения B.

5.2.1 Вопрос 1

Вопрос звучит так: «Для mono_tum и для всех задач оптимизации, какая g2o комбинация методов оптимизации является лучшей?». Чтобы попытаться ответить на этот вопрос, было проведено 11 экспериментов. На рисунке 5.1 показаны результаты (медиана/среднее время отслеживания).

Как можно видеть, лучший результат с точки зрения медианы и среднего времени отслеживания был показан в первом эксперименте, который является настройками по умолчанию для ORB-SLAM, когда для всех методов оптимизации установлено значение Левенберга-Марквардта. Но теперь следует визуализировать погрешность абсолютной позы и сравнивать между экспериментами. Результаты для абсолютной ошибки позы для вопроса 1 можно увидеть на рисунках B.1, B.2 и B.3. Эксперимент, который показал лучшие результаты это эксперимент 7 (рисунок B.3 показывает, что он дает лучшие результаты). Эксперимент 7 соответствует конфигурации ORB-SLAM, когда в Local Bundle Adjustment используется метод Dogleg от g2o, а другие методы оптимизации для соответствующих задач установлены на Levenberg-Marquardt.

5.3 Обсуждение результатов

Как можно видеть, лучший результат с точки зрения медианы и среднего времени отслеживания был показан в первом эксперименте, который является настройками по умолчанию для ORB-SLAM, когда для всех методов оптимизации установлено значение Левенберга-Марквардта. Но теперь погрешность абсолютной позы следует визуализировать и сравнивать между экспериментами. Результаты для абсолютной ошибки позы для вопроса 1 можно увидеть на рисунках B.1, B.2 и B.3. Эксперимент, который показал лучшие результаты, - это эксперимент 7 (рисунок B.3 показывает, что он дает лучшие результаты). Эксперимент 7 соответствует конфигурации ORB-SLAM, когда в Local Bundle Adjustment используется метод Dogleg от g2o, а другие методы оптимизации для соответствующих задач установлены на Levenberg-Marquardt. Была проблема с запуском задач оптимизации на основе Ceres - они через некоторое время потеряли трек и не давали желаемой траектории. Эта проблема может быть решена в будущем. Также была проделана работа над созданием приложения сравнения.

Глава 6

Заключение

Этот тезис произвел исследование того, как конфигурация нелинейных методов оптимизации ORB-SLAM влияет на ее производительность. В ходе разработки диссертации были получены следующие существенные результаты:

- **Расширение нелинейных методов ORB-SLAM**: В рамках исследования этого тезиса было решено расширить ORB-SLAM с помощью задач оптимизации на основе Ceres. Это могло бы расширить область исследования, позволив сравнить две структуры оптимизации и решить, какая структура сильнее влияет на производительность системы SLAM.

- **Разработка и реализация приложения для сравнения**: одна из главных частей этого тезиса - это приложение для сравнения. Это позволяет тестировать различные конфигурации ORB-SLAM, например, запустить ORB-SLAM для определенного набора данных с определенной конфигурацией методов нелинейной оптимизации, показывая результаты оценки (среднее / среднее время отслеживания, абсолютная ошибка позы), визуализируя графики. Реализация приложения для сравнения включала в себя разработку и реализацию приложения с графическим интерфейсом на основе Qt, разработку и развертывание базы данных для хранения результатов, расширение библиотеки ORB-SLAM для работы с приложением для сравнения.

- **Выполнение экспериментов**: было проведено несколько экспериментов, чтобы протестировать приложение сравнения и выяснить, какая конфигурация методов нелинейной оптимизации была оптимальной для данного вопроса. Например, для вопроса 1 результаты показали, что конфигурация, в которой локальная настройка пучка решается с использованием метода оптимизации догола, работает лучше, чем другие с точки зрения абсолютной ошибки позы. Конфигурация по умолчанию для задач оптимизации на основе g2o (все проблемы

решаются с помощью Левенберга-Марквардта) является наилучшей с точки зрения среднего / среднего времени отслеживания, но она генерирует неточную траекторию с высокой абсолютной ошибкой поза.

Литература

[1] J. M. R. Mur-Artal and J. Tardos, "Orb-slam: a versatile and ac- curate monocular slam system," *Transactions on Robotics*, vol. 31, pp. 1147–1163, 2015.

[2] J. M. M. M. Raul Mur-Artal, Juan D. Tardos and D. Galvez-Lopez, "Real-time slam for monocular, stereo and rgb-d cameras, with loop detection and relocalization capabilities," https://github.com/raulmur/ORB_SLAM2.

[3] S. J. Wright and J. Nocedal, *Numerical Optimization*. Springer, 2006.

[4] K. Madsen, H. Nielsen, and O. Tingleff, *Methods for nonlinear least squares problems*, 2004.

[5] S. J. Wright and J. N. Holt, "An inexact levenberg marquardt method for large sparse nonlinear least squares," *Journal of the Australian Mathematical Society Series B*, vol. 4, pp. 387–403, 1985.

[6] H. S. K. K. Rainer Kuemmerle, Giorgio Grisetti and W. Burgard, "g2o: A general framework for graph optimization," *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[7] R. Kuemmerle and et al., "g2o: A general framework for graph optimization," https://github.com/RainerKuemmerle/g2o.

[8] T. U. of Munich (TUM), "Datasets for slam," https://vision.in.tum.de/data/datasets.

[9] J. Sturm, W. Burgard, and D. Cremers, "Evaluating egomotion and structure-from-motion approaches using the TUM RGB-D benchmark," in *Proc. of the Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RJS International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[10] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[12] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, 2016. [Online]. Available: http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.abstract

[13] M. Grupp, "Evo: Python package for the evaluation of odometry and slam," https://github.com/MichaelGrupp/evo.

[14] S. Lovegrove, "Pangolin," https://github.com/stevenlovegrove/Pangolin.

[15] "Opencv," https://opencv.org/releases.html.

[16] "Dbow: Bag of words," https://github.com/dorian3d/DBoW2.

[17] "Eigen: Linear algebra library," http://eigen.tuxfamily.org.

[18] Q. D. Frameworks, "Qtcreator ide," https://www.qt.io/download.

Приложение А

База Данных

А.1 Запросы для создания базы данных

```
CREATE TABLE libraries(
library_id SERIAL PRIMARY KEY,
pose_optim_lib VARCHAR (30) NOT NULL,
lba_optim_lib VARCHAR (30) NOT NULL,
sim3_optim_lib VARCHAR (30) NOT NULL, graph_optim_lib VARCHAR (30)
    NOT NULL);


 CREATE TABLE methods(
method_id SERIAL PRIMARY KEY,
pose_optim_method VARCHAR (60) NOT NULL,
lba_optim_method VARCHAR (60) NOT NULL,
sim3_optim_method VARCHAR (60) NOT NULL, graph_optim_method VARCHAR
     (60) NOT NULL);

CREATE TABLE results(
result_id SERIAL PRIMARY KEY,
trajectory_path VARCHAR (255) NOT NULL,
tracking_time_average DOUBLE PRECISION NOT NULL,
tracking_time_median DOUBLE PRECISION NOT NULL);

CREATE TABLE advanced_ceres(
ceres_adv_id SERIAL PRIMARY KEY,
minimizer_type VARCHAR (50) NOT NULL,
trust_region_type VARCHAR (50),
dogleg_type VARCHAR (50),
initial_trust_region_radius DOUBLE PRECISION,
line_search_direction_type VARCHAR (50),
line_search_type VARCHAR(50),
nonlinear_conjugate_gradient_type VARCHAR(50),
gradient_tolerance DOUBLE PRECISION NOT NULL,
function_tolerance DOUBLE PRECISION NOT NULL,
linear_solver_type VARCHAR (50) NOT NULL,
max_num_iterations INTEGER NOT NULL,
minimizer_progress_to_stdout BOOLEAN NOT NULL,
num_threads INTEGER NOT NULL,
```

```
35 eta DOUBLE PRECISION NOT NULL,
36 max_solver_time_in_seconds DOUBLE PRECISION NOT NULL,
37 use_nonmonotonic_steps BOOLEAN NOT NULL);
38
39 INSERT INTO advanced_ceres (minimizer_type, trust_region_type,
       initial_trust_region_radius, gradient_tolerance,
       function_tolerance, linear_solver_type, max_num_iterations,
40                 minimizer_progress_to_stdout, num_threads, eta,
       max_solver_time_in_seconds, use_nonmonotonic_steps) VALUES ('
       TRUST_REGION', 'LEVENBERG_MARQUARDT', 1e4, 1e-16, 1e-16, '
       SPARSE_SCHUR', 5, TRUE, 1, 1e-2, 1e32, TRUE) returning
       ceres_adv_id;
41
42 CREATE TABLE advanced_g2o(
43 g2o_adv_id SERIAL PRIMARY KEY,
44 method VARCHAR (50) NOT NULL);
45
46 INSERT INTO advanced_g2o (method) VALUES ('levenberg-marquardt');
47 INSERT INTO advanced_g2o (method) VALUES ('gauss-newton');
48 INSERT INTO advanced_g2o (method) VALUES ('dogleg');
49
50 CREATE TABLE advanced(
51 advanced_id SERIAL PRIMARY KEY,
52 pose_ceres INTEGER REFERENCES advanced_ceres,
53 pose_g2o INTEGER REFERENCES advanced_g2o,
54 lba_ceres INTEGER REFERENCES advanced_ceres,
55 lba_g2o INTEGER REFERENCES advanced_g2o,
56 sim3_ceres INTEGER REFERENCES advanced_ceres,
57 sim3_g2o INTEGER REFERENCES advanced_g2o,
58 graph_ceres INTEGER REFERENCES advanced_ceres,
59 graph_g2o INTEGER REFERENCES advanced_g2o);
60
61 CREATE TABLE experiments(
62 experiment_id SERIAL PRIMARY KEY,
63 dataset_name VARCHAR (60) NOT NULL,
64 libraries INTEGER REFERENCES libraries,
65 methods INTEGER REFERENCES methods,
66 advanced INTEGER REFERENCES advanced,
67 results INTEGER REFERENCES results ON DELETE CASCADE);
68
69 INSERT INTO libraries (pose_optim_lib, lba_optim_lib,
       sim3_optim_lib, graph_optim_lib) VALUES ('g2o', 'g2o', 'g2o', '
       g2o'),
70   ('g2o', 'g2o', 'g2o', 'ceres'),
71   ('g2o', 'g2o', 'ceres', 'g2o'),
72   ('g2o', 'g2o', 'ceres', 'ceres'),
73   ('g2o', 'ceres', 'g2o', 'g2o'),
74   ('g2o', 'ceres', 'g2o', 'ceres'),
75   ('g2o', 'ceres', 'ceres', 'g2o'),
76   ('g2o', 'ceres', 'ceres', 'ceres'),
77   ('ceres', 'g2o', 'g2o', 'g2o'),
78   ('ceres', 'g2o', 'g2o', 'ceres'),
79   ('ceres', 'g2o', 'ceres', 'g2o'),
80   ('ceres', 'g2o', 'ceres', 'ceres'),
81   ('ceres', 'ceres', 'g2o', 'g2o'),
82   ('ceres', 'ceres', 'g2o', 'ceres'),
83   ('ceres', 'ceres', 'ceres', 'g2o'),
```

```
84    ('ceres', 'ceres', 'ceres', 'ceres');
85
86  INSERT INTO methods (pose_optim_method, lba_optim_method,
        sim3_optim_method, graph_optim_method) VALUES ('Levenberg-
        Marquardt', 'Levenberg-Marquardt', 'Levenberg-Marquardt', '
        Levenberg-Marquardt'),
87    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Levenberg-
        Marquardt', 'Gauss-Newton'),
88    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Levenberg-
        Marquardt', 'Dogleg'),
89    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Gauss-Newton', '
        Levenberg-Marquardt'),
90    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Gauss-Newton', '
        Gauss-Newton'),
91    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Gauss-Newton', '
        Dogleg'),
92    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Dogleg', '
        Levenberg-Marquardt'),
93    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Dogleg', 'Gauss-
        Newton'),
94    ('Levenberg-Marquardt', 'Levenberg-Marquardt', 'Dogleg', 'Dogleg'
        ),
95    ('Levenberg-Marquardt', 'Gauss-Newton', 'Levenberg-Marquardt', '
        Levenberg-Marquardt'),
96    ('Levenberg-Marquardt', 'Gauss-Newton', 'Levenberg-Marquardt', '
        Gauss-Newton'),
97    ('Levenberg-Marquardt', 'Gauss-Newton', 'Levenberg-Marquardt', '
        Dogleg'),
98    ('Levenberg-Marquardt', 'Gauss-Newton', 'Gauss-Newton', '
        Levenberg-Marquardt'),
99    ('Levenberg-Marquardt', 'Gauss-Newton', 'Gauss-Newton', 'Gauss-
        Newton'),
100   ('Levenberg-Marquardt', 'Gauss-Newton', 'Gauss-Newton', 'Dogleg')
        ,
101   ('Levenberg-Marquardt', 'Gauss-Newton', 'Dogleg', 'Levenberg-
        Marquardt'),
102   ('Levenberg-Marquardt', 'Gauss-Newton', 'Dogleg', 'Gauss-Newton')
        ,
103   ('Levenberg-Marquardt', 'Gauss-Newton', 'Dogleg', 'Dogleg'),
104   ('Levenberg-Marquardt', 'Dogleg', 'Levenberg-Marquardt', '
        Levenberg-Marquardt'),
105   ('Levenberg-Marquardt', 'Dogleg', 'Levenberg-Marquardt', 'Gauss-
        Newton'),
106   ('Levenberg-Marquardt', 'Dogleg', 'Levenberg-Marquardt', 'Dogleg'
        ),
107   ('Levenberg-Marquardt', 'Dogleg', 'Gauss-Newton', 'Levenberg-
        Marquardt'),
108   ('Levenberg-Marquardt', 'Dogleg', 'Gauss-Newton', 'Gauss-Newton')
        ,
109   ('Levenberg-Marquardt', 'Dogleg', 'Gauss-Newton', 'Dogleg'),
110   ('Levenberg-Marquardt', 'Dogleg', 'Dogleg', 'Levenberg-Marquardt'
        ),
111   ('Levenberg-Marquardt', 'Dogleg', 'Dogleg', 'Gauss-Newton'),
112   ('Levenberg-Marquardt', 'Dogleg', 'Dogleg', 'Dogleg'),
113   ('Gauss-Newton', 'Levenberg-Marquardt', 'Levenberg-Marquardt', '
        Levenberg-Marquardt'),
114   ('Gauss-Newton', 'Levenberg-Marquardt', 'Levenberg-Marquardt', '
```

```
    Gauss-Newton'),
115 ('Gauss-Newton', 'Levenberg-Marquardt', 'Levenberg-Marquardt', '
    Dogleg'),
116 ('Gauss-Newton', 'Levenberg-Marquardt', 'Gauss-Newton', '
    Levenberg-Marquardt'),
117 ('Gauss-Newton', 'Levenberg-Marquardt', 'Gauss-Newton', 'Gauss-
    Newton'),
118 ('Gauss-Newton', 'Levenberg-Marquardt', 'Gauss-Newton', 'Dogleg')
    ,
119 ('Gauss-Newton', 'Levenberg-Marquardt', 'Dogleg', 'Levenberg-
    Marquardt'),
120 ('Gauss-Newton', 'Levenberg-Marquardt', 'Dogleg', 'Gauss-Newton')
    ,
121 ('Gauss-Newton', 'Levenberg-Marquardt', 'Dogleg', 'Dogleg'),
122 ('Gauss-Newton', 'Gauss-Newton', 'Levenberg-Marquardt', '
    Levenberg-Marquardt'),
123 ('Gauss-Newton', 'Gauss-Newton', 'Levenberg-Marquardt', 'Gauss-
    Newton'),
124 ('Gauss-Newton', 'Gauss-Newton', 'Levenberg-Marquardt', 'Dogleg')
    ,
125 ('Gauss-Newton', 'Gauss-Newton', 'Gauss-Newton', 'Levenberg-
    Marquardt'),
126 ('Gauss-Newton', 'Gauss-Newton', 'Gauss-Newton', 'Gauss-Newton'),
127 ('Gauss-Newton', 'Gauss-Newton', 'Gauss-Newton', 'Dogleg'),
128 ('Gauss-Newton', 'Gauss-Newton', 'Dogleg', 'Levenberg-Marquardt')
    ,
129 ('Gauss-Newton', 'Gauss-Newton', 'Dogleg', 'Gauss-Newton'),
130 ('Gauss-Newton', 'Gauss-Newton', 'Dogleg', 'Dogleg'),
131 ('Gauss-Newton', 'Dogleg', 'Levenberg-Marquardt', 'Levenberg-
    Marquardt'),
132 ('Gauss-Newton', 'Dogleg', 'Levenberg-Marquardt', 'Gauss-Newton')
    ,
133 ('Gauss-Newton', 'Dogleg', 'Levenberg-Marquardt', 'Dogleg'),
134 ('Gauss-Newton', 'Dogleg', 'Gauss-Newton', 'Levenberg-Marquardt')
    ,
135 ('Gauss-Newton', 'Dogleg', 'Gauss-Newton', 'Gauss-Newton'),
136 ('Gauss-Newton', 'Dogleg', 'Gauss-Newton', 'Dogleg'),
137 ('Gauss-Newton', 'Dogleg', 'Dogleg', 'Levenberg-Marquardt'),
138 ('Gauss-Newton', 'Dogleg', 'Dogleg', 'Gauss-Newton'),
139 ('Gauss-Newton', 'Dogleg', 'Dogleg', 'Dogleg'),
140 ('Dogleg', 'Levenberg-Marquardt', 'Levenberg-Marquardt', '
    Levenberg-Marquardt'),
141 ('Dogleg', 'Levenberg-Marquardt', 'Levenberg-Marquardt', 'Gauss-
    Newton'),
142 ('Dogleg', 'Levenberg-Marquardt', 'Levenberg-Marquardt', 'Dogleg'
    ),
143 ('Dogleg', 'Levenberg-Marquardt', 'Gauss-Newton', 'Levenberg-
    Marquardt'),
144 ('Dogleg', 'Levenberg-Marquardt', 'Gauss-Newton', 'Gauss-Newton')
    ,
145 ('Dogleg', 'Levenberg-Marquardt', 'Gauss-Newton', 'Dogleg'),
146 ('Dogleg', 'Levenberg-Marquardt', 'Dogleg', 'Levenberg-Marquardt'
    ),
147 ('Dogleg', 'Levenberg-Marquardt', 'Dogleg', 'Gauss-Newton'),
148 ('Dogleg', 'Levenberg-Marquardt', 'Dogleg', 'Dogleg'),
149 ('Dogleg', 'Gauss-Newton', 'Levenberg-Marquardt', 'Levenberg-
    Marquardt'),
```

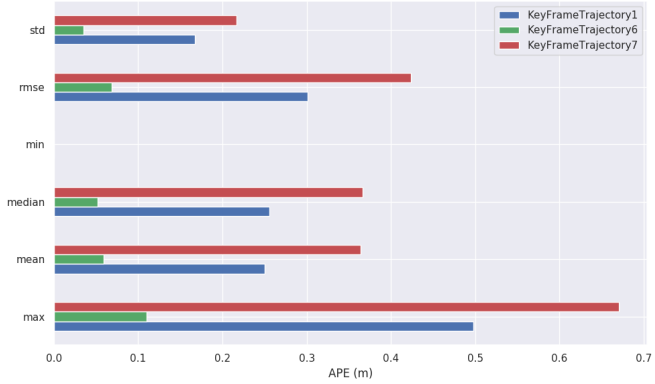
```
150   ('Dogleg', 'Gauss-Newton', 'Levenberg-Marquardt', 'Gauss-Newton')
        ,
151   ('Dogleg', 'Gauss-Newton', 'Levenberg-Marquardt', 'Dogleg'),
152   ('Dogleg', 'Gauss-Newton', 'Gauss-Newton', 'Levenberg-Marquardt')
        ,
153   ('Dogleg', 'Gauss-Newton', 'Gauss-Newton', 'Gauss-Newton'),
154   ('Dogleg', 'Gauss-Newton', 'Gauss-Newton', 'Dogleg'),
155   ('Dogleg', 'Gauss-Newton', 'Dogleg', 'Levenberg-Marquardt'),
156   ('Dogleg', 'Gauss-Newton', 'Dogleg', 'Gauss-Newton'),
157   ('Dogleg', 'Gauss-Newton', 'Dogleg', 'Dogleg'),
158   ('Dogleg', 'Dogleg', 'Levenberg-Marquardt', 'Levenberg-Marquardt'
        ),
159   ('Dogleg', 'Dogleg', 'Levenberg-Marquardt', 'Gauss-Newton'),
160   ('Dogleg', 'Dogleg', 'Levenberg-Marquardt', 'Dogleg'),
161   ('Dogleg', 'Dogleg', 'Gauss-Newton', 'Levenberg-Marquardt'),
162   ('Dogleg', 'Dogleg', 'Gauss-Newton', 'Gauss-Newton'),
163   ('Dogleg', 'Dogleg', 'Gauss-Newton', 'Dogleg'),
164   ('Dogleg', 'Dogleg', 'Dogleg', 'Levenberg-Marquardt'),
165   ('Dogleg', 'Dogleg', 'Dogleg', 'Gauss-Newton'),
166   ('Dogleg', 'Dogleg', 'Dogleg', 'Dogleg');
```

Приложение B

Графики и визуализация

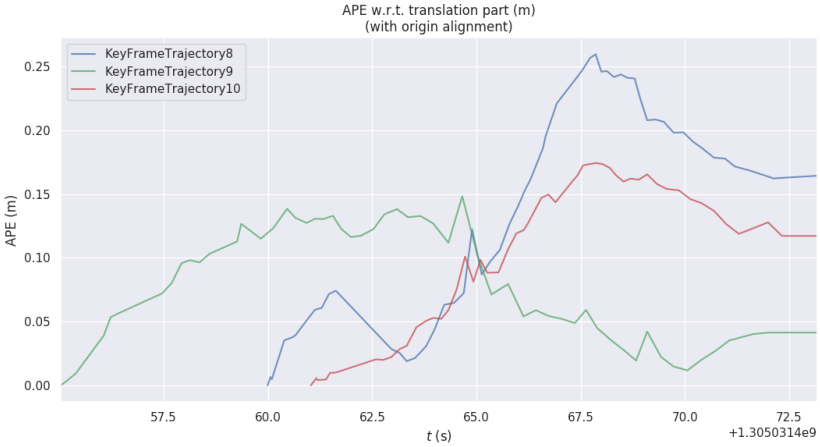
B.1 Вопрос 1: Результаты

(a) Абсолютная ошибка позы: **KeyFrameTrajectory1** - результаты для первого эксперимента (все методы Левенберг-Марквардт); **KeyFrameTrajectory6** - результаты для второго эксперимента (оптимизация графа - Гаусс-Ньютон); **KeyFrameTrajectory7** - результаты третьего эксперимента

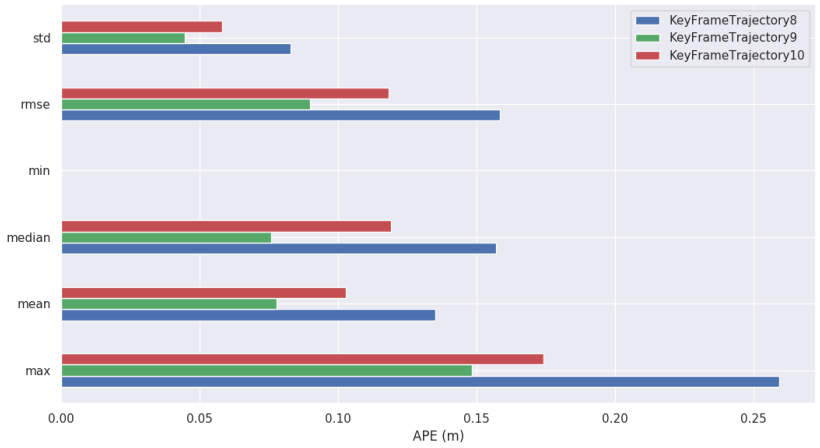


(b) Остальные результаты; квадратическая погрешность лучшая во втором

Рис. B.1: Результаты для 1, 2, 3 эксперементов

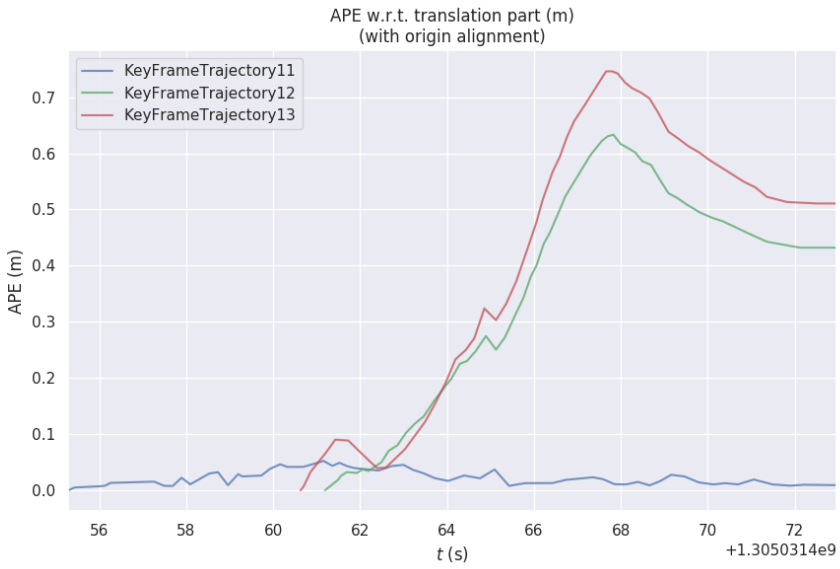


(a) Абсолютная ошибка позы: **KeyFrameTrajectory8** - результаты для четвертого эксперимента (sim(3) - GN); **KeyFrameTrajectory9** - результаты для пятого эксперимента (sim(3) - DL); **KeyFrameTrajectory10** - результаты для шестого эксперимента (LBA - GN)

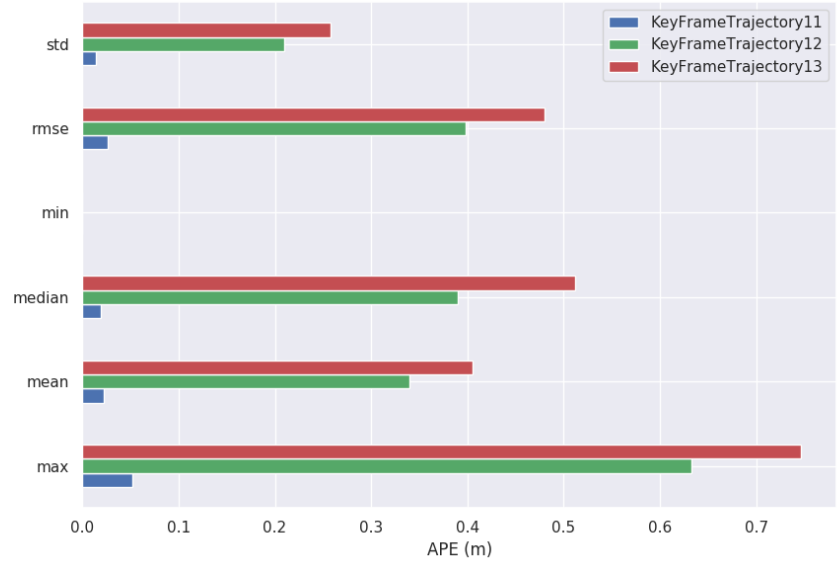


(b) Остальные результаты

Рис. B.2: Результаты для 4, 5, 6 эксперементов



(a) APE: **KeyFrameTrajectory11** - результаты для седьмого эксперимента (LBA - DL); **KeyFrameTrajectory12** - результаты для восьмого эксперимента (Pose - GN); **KeyFrameTrajectory13** - результаты для девятого эксперимента (Pose - DL)



(b)

Рис. B.3: Результаты для 7, 8, 9 эксперементов