Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Системное программирование

Леонид Владимирович Сенюков

Глобальная реконструкция из RGB-D данных

Выпускная квалификационная работа бакалавра

Научный руководитель:
доцент кафедры СП, к. т. н. Ю. В. Литвинов

Консультант:
инженер ООО “Коджент Вижен Рус” Д. А. Корчемкин

Рецензент:
старший преподаватель кафедры информатики, к. ф.-м. н. С. И. Салищев

Санкт-Петербург
2021

SAINT-PETERSBURG STATE UNIVERSITY

Software and Administration of Information Systems
Software Engineering

Leonid Seniukov

Global reconstruction from RGB-D data

Bachelor's thesis

Scientific supervisor:
docent, candidate of Engineering Sciences Yurii Litvinov

Consultant:
engineer at «Cogent Vision Rus» Dmitrii Korchemkin

Reviewer:
senior lecturer, candidate of Physical and Mathematical Sciences Sergey Salishev

Saint-Petersburg
2021

Оглавление

Введение

Высокая точность реконструкции 3D объектов и сцен необходима для работы приложений дополненной реальности и робототехники нового поколения. В сфере робототехники задача генерирования плотных и разреженных моделей статичной области пространства с использованием мобильных роботов и ручных камер с одновременным контролем текущего местоположения и пройденного пути называется SLAM (simultaneous localization and mapping).

Задача реконструкции сцены или объекта по их изображениям является довольно сложной и требует значительных ресурсов. Она может рассматриваться как подзадача SLAM (если выполняется в реальном времени), либо представлять отдельную задачу постпроцессинга, как, например, в алгоритмах SfM (structure from motion). Наличие данных о метрической глубине позволяет разрешить неопределенность масштаба и получить более точные оценки смещений за счет исключения дополнительных степеней свободы.

Особенностью RGB-D камер является предоставление не только трех каналов передачи цвета, но и дополнительного, передающего карты глубин изображений (каждый пиксель задает расстояние до объекта). В последнее время такие камеры стали значительно дешевле, что сделало их более доступными для применения (Microsoft Kinect, MYNT EYE и Intel LiDAR Camera).

Существенным прорывом в области RGB-D реконструкции оказалось появление алгоритмов реального времени [7], [8]. Особенностью данных решений является использование инкрементального подхода к решению задачи, недостатками которого являются, в частности, необходимость использования объемных структур данных для хранения реконструированной к текущему моменту части пространства; а также — накопление ошибок оценки расположения очередного кадра.

По сравнению с инкрементальными подходами, при которых новые данные о сцене появляются в процессе работы алгоритма, при глобальной реконструкции процессы сбора данных и непосредственно

реконструкции разделены во времени, что позволяет при реконструкции использовать все возможные ограничения на взаимное расположение камер. Одним из примеров глобального подхода к решению задачи восстановления структуры объекта из его изображений (structure from motion) для наборов изображений, полученных RGB-камерами, является "усреднение" вращений [17] с последующим "усреднением" подобий [14]. Суть подхода заключается в том, чтобы как можно точнее оценить набор абсолютных перемещений камер из искаженных шумом оценок попарных относительных перемещений между разными точками обзора. Достоинством данного подхода к решению задачи реконструкции является возможность не хранить объемное представление реконструированной части пространства и естественным образом (вследствие использования всех возможных ограничений) избегать накопления ошибки.

В данной работе предлагается распространить идеи усреднения движений на RGB-D случай для решения задачи глобальной реконструкции. Мотивация использования данного подхода состоит в том, что наличие данных о метрической глубине позволяет получить дополнительные ограничения в попарных оценках относительной позы, что может улучшить точность оценки абсолютных поз камер и, как следствие, улучшить качество реконструкции наблюдаемых объектов. Результат будет опубликован в виде open-source библиотеки. Подобное решение могло бы найти применение в области создания высокоточных карт для беспилотных автомобилей, а также моделей пространства для приложений дополненной реальности.

Остальная часть работы устроена следующим образом. В главе 1 описывается цель работы, а также производится постановка основных задач. Затем в главе 2 проводится обзор полезных в рамках данной работы подходов и алгоритмов, а также математических абстракций, применяемых в области. В ее части 2.2 описывается набор данных, подходящих для тестирования разрабатываемого решения, а в 2.6 анализируются существующие решения, решающие задачу проведения RGB-D реконструкции. В главе 3 описывается предлагаемый алгоритм для ре-

шения поставленной задачи, а в главе 4 представлена архитектура решения, реализующего предложенный алгоритм. Далее, в главе 5 описываются проведенные эксперименты, в частности, производится анализ точности полученного решения (5.2) и описываются произведенные замеры времени работы и расходуемой памяти (5.3). Наконец, в главе 6 представлены основные результаты работы.

1. Постановка задачи

Целью данной работы является разработка и реализация алгоритма решения задачи глобальной реконструкции из RGB-D данных.

Для достижения этой цели поставлены следующие задачи:

1. Провести обзор существующих алгоритмов реконструкции.
2. Разработать алгоритм, использующий идеи глобального усреднения движений для задачи RGB-D реконструкции.
3. Реализовать алгоритм.
4. Сравнить результат с существующими алгоритмами RGB-D реконструкции.

2. Обзор предметной области

2.1. Ключевые точки

Часто для оценки перемещения камеры между местами съемки используются соответствия *ключевых точек* изображений. Де-факто "золотым стандартом" для нахождения и сопоставления ключевых точек является алгоритм Scale-Invariant Feature Transform (SIFT) [21] и метод сравнения дескрипторов [2]. Неплохие результаты могут быть достигнуты благодаря использованию SuperGlue [32].

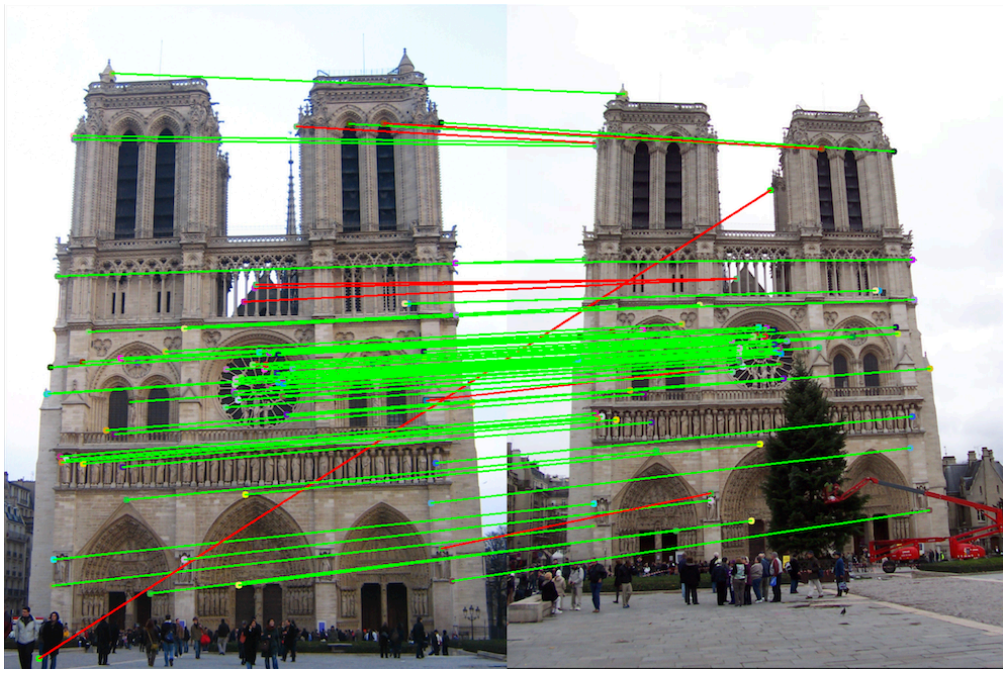


Рис. 1: Выделены истинные (зеленым) и ложные (красным) совпадения ключевых точек [25]

С точки зрения RGB-D реконструкции многообещающим выглядит подход, совмещающий информацию о глубине ключевых точек с их координатами в плоскости изображения. В таком случае открывается возможность повысить устойчивость к выбросам (т. н. аутлаерам — измерениям, выделяющимся из общей выборки) у действующих в условиях присутствия выбросов (т. н. робастных) схем оценки параметров благодаря использованию меньшего числа точек для получения минимальных (т. е. получаемых по минимально допустимому числу соответ-

ствий) оценок. В случае RGB-D данных минимальная оценка — оценка перемещения камеры между двумя кадрами по трем сопоставленным ключевым точкам [27], [34]. Использование метрических глубин позволяет оценить кроме относительной ориентации и направления смещения (как в RGB случае) еще и параметр масштаба. После получения начального предположения об относительной позе двух RGB-D камер с использованием ключевых точек можно уточнить оценку, используя итеративные алгоритмы наподобие [4].

2.2. Данные

Для отладки и тестирования был выбран часто используемый для оценки качества работы алгоритмов RGB-D реконструкции датасет [3], содержащий RGB-D реальные данные съемки камерами Microsoft Kinect и Asus Xtion, а также ground-truth данные для поз камер. В датасете представлены данные различного характера: траектории движения камер с замыканием циклов и без; в разных последовательностях кадров, в зависимости от скорости движения камеры, представлены снимки разной четкости; съемка одних последовательностей велась при помощи ручной камеры, других — с использованием камеры, прикрепленной к роботу. Важным преимуществом датасета, например, над набором, выпущенным вместе с `BundleFusion` [7], является его использование авторами других работ [19], [37] по RGB-D реконструкции для оценки точности своих решений. В частности, авторы `BundleFusion` сами использовали его в ходе своих экспериментов. Данный факт облегчает сравнение результатов работы с аналогами, поэтому именно этот датасет был выбран для оценки разработанного алгоритма.

Следует отметить, что RGB кадры и карты глубин в последовательностях предварительно выровнены между собой так, что пиксели на цветном изображении соответствуют пикселям на парной изображению карте глубины без сдвигов. Снимки требуют выравнивания из-за особенности использовавшихся для сбора данных камер: ненулевого смещения между RGB и IR датчиками, из-за которого на RGB сним-

ках исчезают некоторые окклюзии, отображаемые картой глубины, и информация о глубине некоторых пикселей цветного изображения становится неизвестной.

(a) Цветное изображение (b) Карта глубины

Рис. 2: Пример RGB-D данных, соответствующих одному моменту времени [3]

2.3. Представление движений камеры

Относительная поза есть преобразование, соответствующее преобразованию облака точек, видимого с i-ой позы, к облаку точек j-ой позы:

$$\mathcal{T}_{ij}(p) = R_{ij}(p) + t_{ij} \tag{1}$$

Формально, если $SO(3)$ — группа вращений в трехмерном пространстве, $R_0, \dots R_{n-1} \in SO(3)$ и $t_0, \dots t_{n-1} \in \mathbb{R}^3$, то

$$\mathcal{T}_i(p) = R_i(p) + t_i \tag{2}$$

есть преобразование системы координат камеры i к глобальной “мировой” системе координат, причем

$$\mathcal{T}_{ij}(p) = \left(\mathcal{T}_j^{-1} \circ \mathcal{T}_i\right)(p) = R_{ij}(p) + t_{ij} = R_j^{-1}\left(R_i(p) + t_i - t_j\right) \tag{3}$$

2.4. Bundle Adjustment

Bundle Adjustment (BA) [10] — метод одновременного уточнения поз камеры и наблюдаемого облака точек, который предполагает минимизацию некоторой целевой функции.

Рассмотрим наблюдаемую камерой j точку Q_i. Пусть точке соответствует образ в плоскости изображения q_{ji}, а камера j имеет абсолютную позу $\mathcal{T}_j \in SE(3)$ (Евклидово преобразование).

В качестве целевой функции тогда можно использовать

$$\varepsilon_{reproj}\left(\{\mathcal{T}_j\}_{j=0}^{N_c}, \{Q_i\}_{i=0}^{N_p}\right) = \sum_{i=0}^{N_p} \sum_{j \in A_i} \rho\left(\frac{\|e_{ji}\|}{\sigma_{ji}}, \sigma_T\right) \tag{4}$$

Где A_i есть набор индексов камер, обозревающих точку Q_i, $\|e_{ji}\|$ есть ошибка репроекции (расстояние в пикселях) между реальным образом точки q_{ji} на плоскости изображения и проекцией оцениваемого положения этой точки, σ_{ji} — оцененное стандартное отклонение ошибки репроекции, σ_T — устойчивая к выбросам оценка стандартного отклонения среди всех ошибок репроекции, основанная на их медиане [36], ρ — устойчивая к выбросам функция потерь, монотонно зависящая от значения ошибки, например, биквадратная функция (Tukey's biweight function):

$$\rho(x, c) = \begin{cases} \frac{c^2}{6}\left(1 - \left(1 - \frac{x^2}{c^2}\right)^2\right)^3, & \text{если } |x| < c \\ \frac{c^2}{6}, & \text{если } |x| \geq c \end{cases} \tag{5}$$

Минимизацию (4) можно провести, используя итеративный алгоритм нелинейной оптимизации (например, алгоритм, описываемый в [6]).

В [29] предлагается учитывать ограничения на метрическую глубину рассматриваемых точек для Bundle Adjustment в RGB-D случае. Для этого к 2D ограничению (4) добавляется одномерное ограничение на глубину

$$d = (\mathcal{T}^{-1} \cdot p)_z$$

отвечающую координате Z точки p в системе координат камеры, имеющей абсолютную позу $\mathcal{T} \in SE(3)$. Тогда для каждой репроецируемой точки $p = p_j$ к (4) добавляется слагаемое

$$\rho\left(\frac{\|d - u_z\|}{\sigma_{d_{ji}}}, \sigma_{d_T}\right)$$

где u_z — наблюдаемая камерой i глубина точки p, $\sigma_{d_{ji}}$ — оцененное стандартное отклонение ошибки глубины, σ_{d_T} — устойчивая к выбросам оценка стандартного отклонения среди всех ошибок глубины и ρ — монотонная устойчивая к выбросам функция потерь. Также в работе показывается квадратичная оценка зависимости стандартного отклонения ошибки измерения глубины камеры от непосредственно значения измеряемой глубины для камер Microsoft Kinect.

2.5. Глобальные методы реконструкции

Глобальные методы реконструкции могут обрабатывать все кадры одновременно и часто не требуют, чтобы съемка велась средствами одной аппаратуры [9].

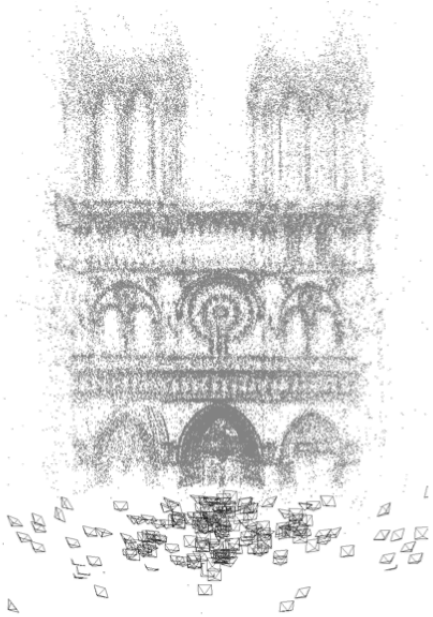


Рис. 3: Пример реконструкции Собора Парижской Богоматери из RGB данных [9]

Можно заметить, что формула композиции движений (3) позволяет выделить ограничение на абсолютные ориентации камер (независящие от смещений) и смещения:

$$R_{ij} = R_j^{-1} R_i \tag{6}$$

$$t_{ij} = R_j^{-1}(t_i - t_j) \tag{7}$$

Из (7) домножением слева на R_j получим

$$R_j t_{ij} = t_i - t_j \tag{8}$$

Уравнения (1), (6), (8) соответствуют 3 стандартным этапам в работе глобальных методов:

1. Оценка относительных поз (1).
2. Усреднение вращений, т. е. оценка R_i из (2) по оценкам R_{ij} из (6).
3. Усреднение смещений, т. е. оценка t_i из (2) по оценкам t_{ij} и R_j из (8).

Реализации глобальных методов реконструкции для RGB камер страдают от сложности оценки метрического масштаба смещений между кадрами [17]. Эта сложность вызвана тем, что ограничения копланарности пар лучей, возникающих в RGB случае, позволяют оценить относительное смещение лишь с точностью до неизвестного масштаба.

Например, стандартным шагом при оценивании движений между RGB кадрами является вычисление *существенной матрицы* [24] (матрицы размера 3×3, имеющей 5 степеней свободы, позволяющей оценить вращение и направление вектора смещения), которая не содержит информации о длине вектора смещения. Существенную матрицу можно вычислить, например, имея 5 сопоставлений точек между кадрами [24].

Отметим, что методы оценки относительной позы не обязательно должны оценивать существенную матрицу. Например, возможно оце-

нивать вращения [20] и проводить отдельную двухточечную оценку направления смещения.

Одним из способов решения задачи оценки смещений в глобальной реконструкции из RGB-изображений является введение дополнительных шагов построения локальных разреженных карт глубин и усреднения масштабов смещений [14].

Заметим, что в случае RGB-D камер доступны плотные карты глубин, что позволяет сразу производить оценку относительных поз с метрическим масштабом смещения.

Именно применение методов глобальных усреднений в контексте RGB-D реконструкции, ровно как и использование всех возможностей, получаемых благодаря наличию карт глубин, представляют наибольший интерес и являются практической целью данной работы.

2.6. Инкрементальные алгоритмы RGB-D реконструкции

Инкрементальные алгоритмы получают на вход поток кадров и карт глубин и могут использовать допущение, что частота съемки достаточна, чтобы изменение ориентации и положения камеры между кадрами было довольно мало.

Одно из последних онлайн инкрементальных решений `BundleFusion` [7], результат работы которого на датасете `BundleFusion/office3` представлен на рисунке 4, разбивает поток кадров на блоки кадров размера $N_{chunk} = 11$ и локально выравнивает облака точек, полученные с этих кадров, а затем глобально оптимизирует блоки между собой. Для этого используется метод реинтеграции поверхности (реконструированная область пространства “дополняется” новыми кадрами с условием поддержания целостности его поверхностей), а глобальная оптимизация основана на методе Ньютона-Гаусса.

Основным недостатком подобных онлайн подходов является необходимость хранить реконструированную часть пространства, а также возникающие проблемы с накоплением ошибки.

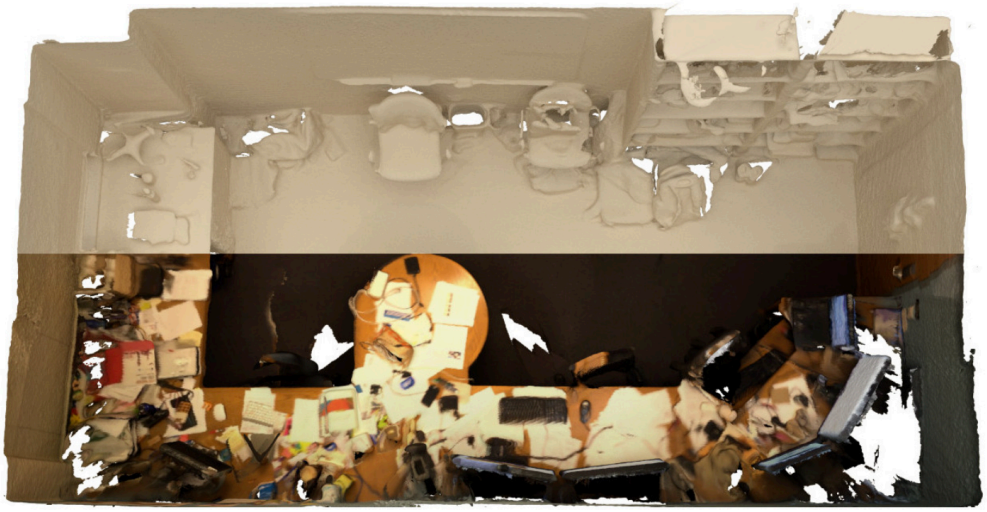


Рис. 4: Пример реконструкции офисного помещения из RGB-D данных алгоритмом `BundleFusion`

Стоит отметить, что для оценки траектории не обязательно поддерживать плотную модель пространства в памяти. Например, система `DVO-SLAM` [19] реализует подход, при котором для попарных оценок относительной позы минимизуется фотометрическая ошибка яркости пикселей в сочетании с ошибкой глубины, а `RGB-D SLAM` [37] использует для тех же целей сопоставления ключевых точек. Вычисление глобальных поз в обеих реализациях получается путем оптимизации графа поз с использованием библиотеки `g2o` [38]. Недостатком такого решения является то, что использование подобной оптимизационной библиотеки общего назначения приводит к возможности получить лишь локально-оптимальное решение, используя для этого предыдущее решение и новые данные о перемещении камеры. Предлагаемый же подход не требует инициализации и приводит к нахождению в некотором смысле глобально-оптимального решения.

Стоит также отметить, что в случае глобального подхода можно с легкостью расширить систему для обработки данных с нескольких камер. Для этого достаточно будет просто объединить данные с них в один фрагмент (при условии известных внутренних параметров).

3. Описание предлагаемого алгоритма

Предлагаемый алгоритм состоит из следующих этапов:

1. Поиск ключевых точек и их сопоставление.
2. Оценка относительных поз камер.
3. Усреднение движений.
    a) Усреднение вращений.
    b) Усреднение смещений.
4. Bundle Adjustment.
5. Создание модели.

3.1. Поиск ключевых точек и их сопоставление

Поиск ключевых точек предлагается производить при помощи алгоритма SIFT. В результате сопоставления найденных при помощи SIFT дескрипторов ключевых точек (вещественных векторов длины 128) можно обнаружить совпадения между ключевыми точками на разных снимках.

В сочетании с информацией о глубине координаты ключевых точек в плоскости изображения позволяют оценить относительные позы камеры, а также выполнить Bundle Adjustment.

3.2. Оценка относительных поз камер

В итеративных алгоритмах реконструкции в качестве начального приближения преобразования $\mathcal{T}_{ij}$ (1) зачастую используется тождественное преобразование, а также оценка, полученная на основе кинематической модели и уже реконструированной части пространства. Эта оценка затем уточняется при помощи использования информации, полученной из плотных карт глубин. Ввиду отсутствия данных о реконструированной части пространства и пройденной траектории (в силу возможного

отсутствия траектории как таковой) при глобальной реконструкции, приходится использовать иные методы.

Обозначим за C_i i-ый по порядку цветной кадр, а за D_i — соответствующую ему карту глубины. Тогда входными данными для алгоритма является набор

$$S = \{f_i = (C_i, D_i)\}$$

и каждой точке RGB изображения должна быть сопоставлена определенная глубина. В реальности из-за выравнивания кадров относительно их карт глубин, помех и других причин глубина некоторых пикселей может быть неизвестна. В таком случае будем рассматривать только те ключевые точки, метрическая глубина которых дана. Обозначим за $\{(X_i, Y_i)\}$ пары векторов, соответствующих координатам точек в трехмерном пространстве в системах координат двух камер, чьи спроецированные на плоскость изображения образы $\{(x_i, y_i)\}$ являются найденными сопоставлениями ключевых точек на изображениях C_1 и C_2. Для глобального усреднения движений требуется получить оценки относительной позы некоторых пар камер, т. е. требуется оценить преобразование $\mathcal{T}(p) = R(p) + t$ (вращение R и смещение t), отображающее $\{Y_i\}$ в $\{X_i\}$ "наилучшим образом".

В случае, если в качестве ошибки совмещения облаков точек используется ℓ_2-норма, то существует решение в замкнутой форме для задачи оценки преобразования $\mathcal{T}$ [34] (далее — алгоритм umeyama). Если же требуется минимизировать ошибку репроекции в плоскости изображения, то решение можно получить, например, использованием «Lambda twist» подхода [27]. В отличие от umeyama, однозначно оценивающего преобразование $\mathcal{T}$, в результате применения «Lambda Twist» возможно нахождение до 4 геометрически возможных решений, поэтому для использования в рамках данной работы был выбран именно алгоритм umeyama.

Для повышения устойчивости процесса оценки относительной позы (преобразования $\mathcal{T}$) к выбросам (в том числе к ошибкам сопоставления точек) предлагается использовать Locally Optimized RANSAC (LoRANSAC) [11], в котором umeyama может быть использован и на

шаге решения минимальной задачи (оценка по трем случайно выбранным точкам), и на шаге локальной оптимизации.

Результатом совмещения прообразов ключевых точек является начальное приближение для оценки преобразования $\mathcal{T}$. Оценку можно улучшить, применив одну из вариаций Iterative Closest Point (ICP) [4]. Общая идея состоит в том, чтобы итеративно строить приближения $\mathcal{T}$, но уже на облаках, полученных из плотных карт глубин. Из-за того, что алгоритм может сойтись к локальному минимуму задачи минизации функции ошибки (так как на каждом шаге для сопоставления точек в плотных облаках точек используется текущая оценка $\mathcal{T}$), требуется задать начальное приближение, в качестве которого и предлагается использовать описанную ранее робастную оценку на основе алгоритма umeyama, используемого в LoRANSAC фреймворке.

3.3. Усреднение движений

Усреднение вращений

Усреднение вращений — задача оценки n неизвестных ориентаций (вращений) $R_0, \dots R_{n-1} \in SO(3)$ из искаженных шумом оценок $\hat{R}_{ij}$ относительных ориентаций R_{ij} (6). Стоить отметить, что оценки ориентаций не зависят от оценок смещений между позами. Процесс нахождения глобально-оптимального (в смысле нахождения оценки максимального правдоподобия для задачи (9), где $\kappa_{ij} \geq 0$ — параметры, описывающие модель шума [12] и $\mathcal{E}$ — набор пар индексов (i, j), для которых известны оценки $\hat{R}_{ij}$) решения для задачи усреднения вращений описывается в [30].

$$\max_{R \in SO(d)^n} \sum_{(i,j) \in \mathcal{E}} \kappa_{ij} tr(R_i^{-1} \hat{R}_{ij} R_j) \tag{9}$$

Чтобы уменьшить влияние ошибочно оцененных относительных поз на результат, абсолютные ориентации можно уточнить, используя устойчивую к выбросам функцию потерь и ценовую функцию вида

$$\log_{SO3}(R_i^{-1} \hat{R}_{ij} R_j)$$

Здесь $\log_{SO3} : SO(3) \to \mathfrak{so}(3)$ [22] есть отображение, вычисляющее по элементу группы вращений $SO(3)$ соответствующий ему элемент касательного пространства (вектор поворота) алгебры Ли $\mathfrak{so}(3)$.

Усреднение смещений

При известных R_i и t_{ij} уравнение (8) становится линейным относительно t_i.

Выписав все полученные ограничения, получим $m \leq \frac{n(n-1)}{2}$ уравнений над n неизвестными векторами. Соединив их, получим линейную систему, рассматриваемую в операторной формулировке

$$Ax = b, x = (t_0, t_1, \dots t_{n-1})^\intercal \tag{10}$$

b есть столбик, составленный из m векторов $R_j t_{ij}$

Заметим, что строка матрицы A, соответствующая $R_j t_{ij}$, имеет только два ненулевых элемента: 1 и -1. В таком случае целесообразно применение итеративных методов решения линейных систем, использующих постановку задачи в смысле линейных операторов, требуется лишь добиться того, чтобы матрица системы была матрицей полного ранга. Этого можно достичь, приняв за начало отсчета позицию некоторого кадра, например того, для которого максимально количество успешно оцененных относительных поз.

Для повышения устойчивости к выбросам стоит воспользоваться Iteratively Reweighted Least Squares (IRLS) [5].

3.4. Bundle Adjustment

После получения оценок позиций и ориентаций камер в единой системе координат и обозреваемого облака итоговые оценки можно уточнить, применив Bundle Adjustment (2.4). Сделать это достаточно один раз, в отличие от инкрементальных методов.

3.5. Создание модели

Из данных об абсолютных позах камер с использованием карт глубин и цветных изображений появляется возможность произвести реконструкцию сцены. Для уменьшения объема памяти, требуемого для хранения подобного объекта целесообразно будет заменить облако точек на некоторую выборку из него.

Реализовать подобное усреднение можно, например, сохранив облако в вокселях определенного размера или поддерживая в виде пространственной деревообразной структуры вроде октодерева фиксированной глубины.

4. Архитектура предлагаемого решения

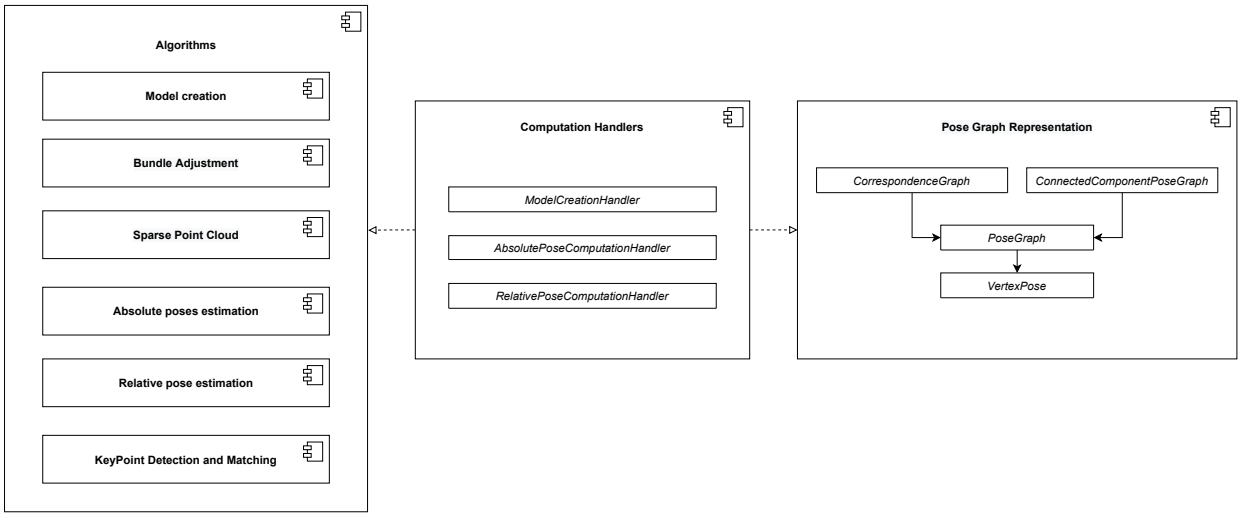


Рис. 5: Предлагаемая структура решения

При использовании представления поз камеры в виде статических структур (`Pose Graph Representation` на Рис. 7) открывается возможность отделить их от структур и алгоритмов, необходимых для осуществления шагов реконструкции (`Algorithms`).

Управление процессом вычисления возложено на специальные классы — контроллеры вычислений (`Computation Handlers`), каждый из которых соответствует глобальному этапу алгоритма.

4.1. Pose Graph Representation

Позицию камеры оказывается удобным представлять в виде вершины графа, в котором ребра соединяют позы, оценка преобразования между которыми известна.

`PoseGraph` является контейнером для информации об оцениваемых положениях камеры и известных относительных позах. Из-за специфики производимых над графом действий пришлось уточнить граф поз до двух различных категорий:
`CorrespondenceGraph` и `ConnectedComponentPoseGraph`.

`CorrespondenceGraph` изначально пуст и инициализируется при чтении датасета. По мере успешной оценки относительных поз в него до-

бавляются ребра. Проблема заключается в том, что для нахождения абсолютных поз требуется связность оцениваемого графа, поэтому из него выделяются компоненты связности, которым соответствуют классы `ConnectedComponentPoseGraph`.

4.2. RelativePoseComputationHandler

За оценку относительных поз отвечает `RelativePoseComputationHandler`, по очереди запускающий следующие шаги:

4.2.1. KeyPoint Detection and Matching

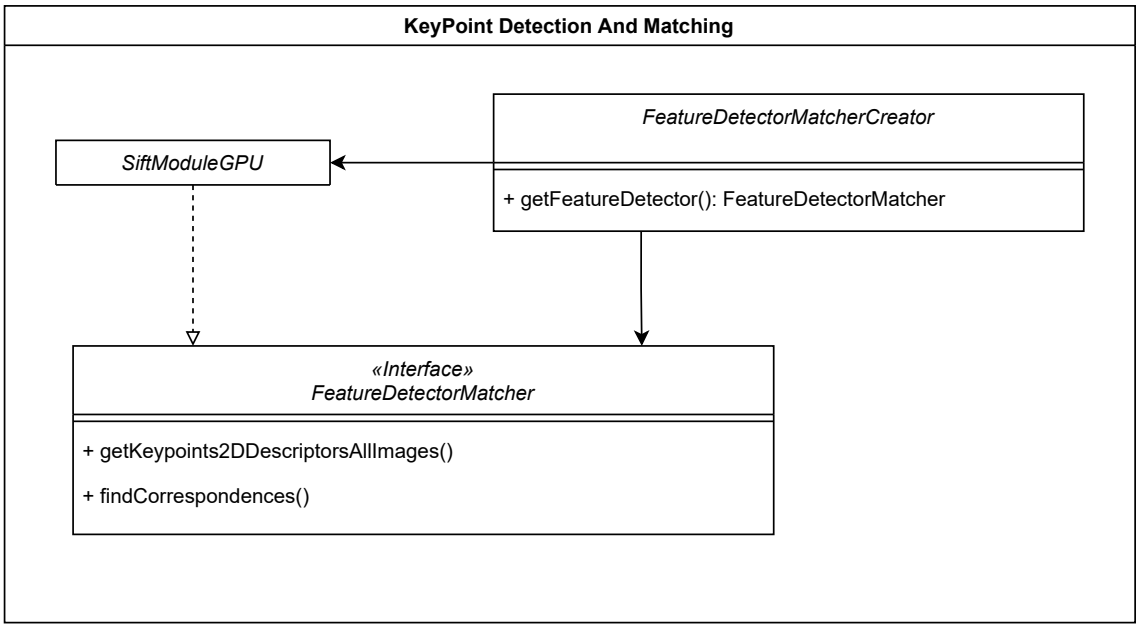


Рис. 6: Нахождение и сопоставление ключевых точек, диаграмма классов

Из соображений эффективности для ипользования была выбрана GPU-реализация SIFT — библиотека `SiftGPU` [31], взаимодействие с которой происходит через класс `SiftModuleGPU`, в которой представлены реализации алгоритма с использованием CUDA C [23], а нахождение сопоставлений ключевых точек между кадрами (матчинг) также производится средствами GPU. Для увеличения точности сопоставле-

ния дополнительно производится нормализация дескрипторов методом `RootSift` [2]

В качестве альтернативы была рассмотрена CUDA реализация `CudaSIFT` [13], но предпочтение было отдано первому варианту, потому что точность сопоставления точек в контексте глобального подхода оказывается важнее скорости работы.

4.2.2. Relative pose estimation

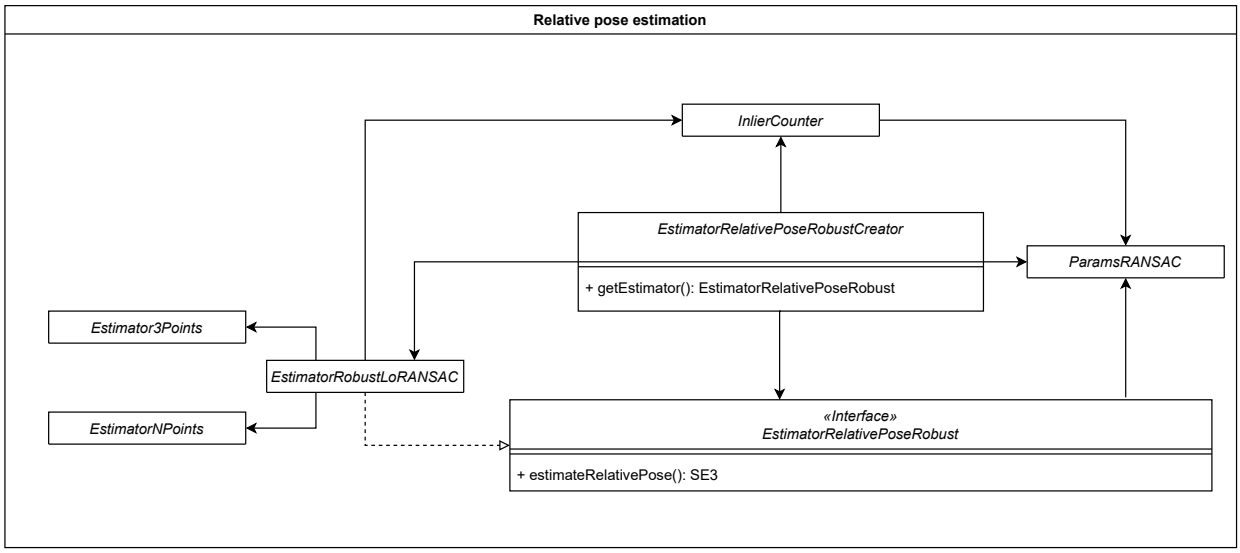


Рис. 7: Устойчивая к выбросам оценка относительной позы, диаграмма классов

Оценка позы происходит через интерфейс `EstimatorRelativePoseRobust`, реализующийся классом `EstimatorRobustLoRANSAC`. Созданием экземпляра класса занимается фабрика `EstimatorRelativePoseRobustCreator`.

Для оценки позы по 3 и N точкам используются классы `Estimator3Points` и `EstimatorNPoints`, инкапсулирующие вызов алгоритма `umeyama` из библиотеки `Eigen` [15].

Сам фреймворк `LoRANSAC` конфигурируется структурой `ParamsRANSAC`, содержащей данные о числе итераций, минимальной доле инлаеров для признания оценки успешной и граничного значения

ошибки сопоставления точки для определения ее как выброса. Подсчет числа инлаеров производится классом `InlierCounter`.

Благодаря тому, что результаты вычисления относительных поз не зависят друг от друга, их можно оценивать параллельно. Из соображений удобства и простоты использования для внедрения в проект многопоточности была выбрана библиотека `Intel TBB` [28] — подобное распараллеливание было осуществлено благодаря использованию конструкции `tbb::parallel_for`. Так как выполняемые задачи одинаковы по сложности исполнения, то загрузка доступных потоков происходит равномерно, что дает оптимальный уровень эффективности.

4.2.3. Relative pose refinement

Уточнение полученной в результате робастной оценки относительной позы происходит при помощи алгоритма ICP. Из-за того, что приходится выравнивать между собой облака точек, полученные из плотных карт глубин, исполнение алгоритма оказывается весьма затратным по времени и памяти. В первую очередь с целью экономии времени требовалось внедрить в систему реализацию алгоритма на GPU. Среди находящихся в открытом доступе реализаций была обнаружена только версия `ICPCUDA` [18], которая и была использована.

Получив попарные оценки относительной позы, можно перейти к оценке абсолютных поз, осуществляемой `AbsolutePoseComputationHandler`.

4.3. AbsolutePoseComputationHandler

Дальнейшие шаги алгоритма, запускаемые этим контроллером:

4.3.1. Rotation Averaging

Для усреднения вращений используется реализация Shonan Rotation Averaging [30], доступная в библиотеке `gtsam` [16].

Устойчивое к выбросам уточнение вращений происходит при помощи библиотеки `ceres-solver` [1].

4.3.2. Translation Averaging

Усреднение смещений производится путем решения разреженной системы линейных уравнений 3.3. Для этих целей оказывается удобным использовать часть библиотеки `Eigen`, специализированную для работы с разреженными системами линейных уравнений и разреженными матрицами.

IRLS требует инициализации начальным приближением решения, с которого начинается итеративный процесс. В качестве такого кандидата можно использовать l_2-решение, получаемое, например, методом Preconditioned Conjugate Gradient (PCG) [35], реализация которого доступна в библиотеке `Eigen`. Взвешенная система для IRLS получается домножением исходной на веса, обратные к вычетам, и ее решение снова сводится к нахождению l_2-решения.

4.3.3. Bundle Adjustment и Sparse Point Cloud

Для реализации Bundle Adjustment требуется иметь представление о совпавших ключевых точках между всеми изображениями. Для создания такого облака используется два класса компонента `Sparse Point Cloud`: `PointClassifier` и `CloudProjector`.

`PointClassifier` отождествляет совпавшие точки между собой, получая на вход данные о попарных совпадениях между ними, таким образом распределяя их по своеобразным классам эквивалентности.

`CloudProjector` вычисляет координаты полученных классов точек в трехмерном пространстве, находя среднее значение по всем оценкам положения представителей конкретного класса.

Непосредственно уточнением поз и координат облака точек занимается класс `BundleAdjuster`, реализующий подход, описанный в [29]. Для нелинейной оптимизации снова использовалась библиотека `ceres-solver`.

4.4. Model Creation

Для вокселизации, экспорта в файл и визуализации плотного реконструированного облака точек используется библиотека для работы с облаками точек `Point Cloud Library` (`PCL`) [26].

5. Эксперименты

5.1. Методология

На каждой из рассматриваемых последовательностей RGB-D данных алгоритм запускался $N = 5$ раз. Данные о скорости работы и затратах памяти затем усреднялись, а среди оцененных траекторий для дальнейшего анализа выбиралась медианная по значению среднеквадратичной абсолютной ошибки смещения ATE RMSE (Absolute trajectory root-mean-square-error)[1].

В результате проведенных экспериментов получены данные о точности оценивания траектории: смещений (в сантиметрах) и ориентаций (в градусах), а также проведены измерения скорости работы и объема потребляемой памяти CPU RAM.

Измерения проводились на платформе, имеющей 2 процессора `Intel Xeon Platinum 8176`, а также 4 видеокарты `GeForce RTX 2080 Ti Rev. A` на трех последовательностях датасета freiburg: `fr1/desk`, `fr2/xyz`, `fr3/office`. Выбор этих последовательностей обусловлен тем, что для них доступны данные о точности аналогов, собранные другими авторами.

Для тестирования предлагаемого решения набор `fr2/xyz` из-за его большой длины (более 3.5 тысяч кадров) был заменен на выборку из него (брался каждый 4 кадр). В результате получилась последовательность длиной 912 кадра, наборы `fr1/desk`, `fr3/office` были оставлены без изменения, их длина составила 573 и 2488 кадра соответственно.

При замерах времени работы и потребляемой памяти непосредственное построение плотной модели не производилось, потому что для оценки траектории оно не требуется, однако плотную модель можно построить в результате дополнительного шага, имея данные об оцененной траектории. Пример такой реконструированной модели приведен на рисунке 8.

[1]Если $t_i \in \mathbb{R}^3$ — точное положение камеры в позе i, а $\hat{t}_i$ — оцененное, то $\mathrm{RMSE}(\{e_i\}) = \sqrt{\frac{1}{n}\sum_{i=1}^{n} e_i^2}$, где $e_i = \mathrm{ATE}(t_i, \hat{t}_i) = \|t_i - \hat{t}_i\|$

Рис. 8: Реконструированная часть пространства для набора `fr2/xyz`

5.2. Точность оценки траектории

Для оценивания качества получаемой траектории использовалось две метрики: *абсолютная ошибка смещения* и *абсолютная ошибка ориентации*.

Абсолютная ошибка смещения может трактоваться как l_2 расстояние между оцененными и точными координатами камеры в $\mathbb{R}^3$, а абсолютная ошибка ориентации — как угол между оцененным $\hat{R}_i$ и точным R_i направлениями оптических центров. Для матричного представления ошибки ориентации $R = {R_i}^{-1}\hat{R}_i \in SO(3)$ формула для вычисления ошибки примет вид:

$$\text{Rotation Error}(R) = cos^{-1}\left(min\left(1, max\left(-1, \frac{tr(R)-1}{2}\right)\right)\right)$$

Оценки точности абсолютных смещений при позиционировании камеры в сравнении с существующими решениями приведены в таблице 1. Результаты точности аналогов взяты из публикации [7].

	fr1/desk	**fr2/xyz**	**fr3/office**
DVO-SLAM	2.1	1.8	3.5
RGB-D SLAM	2.3	**0.8**	3.2
BundleFusion	**1.6**	1.1	2.2
Предлагаемое решение	1.8	1.3	**1.5**

Таблица 1: ATE RMSE, см

Данные о точности оценки абсолютных ориентаций оказались доступны не для всех последовательностей, но имеющиеся данные о точности аналогов взяты из публикации [37]. Сравнение приведено в таблице 2.

	fr1/desk	**fr2/xyz**	**fr3/office**
RGB-D SLAM	2.4	–	–
Предлагаемое решение	2.1	2.2	0.7

Таблица 2: Rotation RMSE, °

Значения стандартного отклонения результатов точности оценки траектории для рассматриваемых метрик приведены в таблице 3.

	fr1/desk	**fr2/xyz**	**fr3/office**
ATE STD, см	1.0	0.6	0.6
Rotation Error STD, °	0.5	0.3	0.3

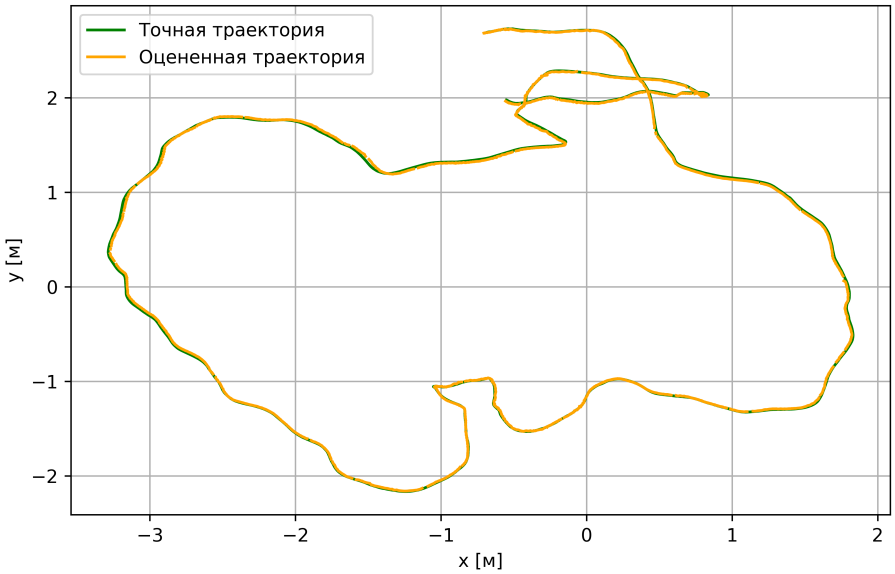
Таблица 3: Стандартное отклонение

Данные о количестве успешно локализированных поз, т.е. тех, для которых была найдена оценка абсолютной ориентации и смещения, приведены в таблице 4.

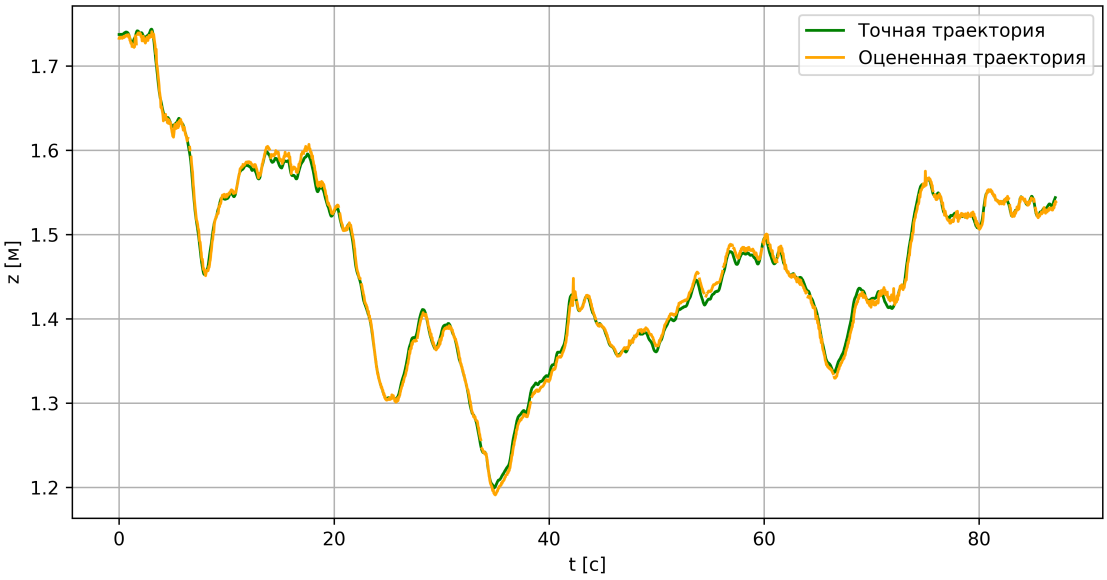
	fr1/desk	**fr2/xyz**	**fr3/office**
Количество локализированных поз	570	912	2481
Общее число поз в датасете	573	912	2488

Таблица 4: Число успешно оцененных абсолютных поз

Оцененная и точная траектории одного из наборов показаны на рисунке 9a. Из-за невозможности изобразить трехмерную кривую на двумерном графике значения координаты Z обеих траекторий в зависимости от времени вынесены отдельно на график 9b.



(a) В плоскости XY



(b) По оси Z

Рис. 9: Сравнение оцененной траектории и точной для набора `fr3/office`

	fr1/desk	**fr2/xyz**	**fr3/office**
Среднее	1633.6	5384.2	14415.2
Стандартное отклонение	450.6	728.4	483.2

Таблица 5: Время работы, с

	fr1/desk	**fr2/xyz**	**fr3/office**
Среднее	5778.2	22760.9	27624.7
Стандартное отклонение	11.8	8.6	131.9

Таблица 6: Затраты памяти CPU RAM (max resident stack size), MB

5.3. Измерение времени работы и потребляемой памяти

В таблице 5 приведены данные о скорости работы реализованного алгоритма, и в таблице 6 — затраты оперативной памяти, потребовавшиеся для оценки траектории.

5.4. Анализ результатов экспериментов

Проведенный эксперимент и сравнение точности оценки траектории показывает, что для датасета `fr1/desk` разработанный алгоритм показывает хоть и не лучший, но сравнимый с аналогами результат. Также заметно, что уменьшение частоты съемки на датасете `fr2/xyz` не привело к значительному увеличению значения среднеквадратичной ошибки на фоне конкурентов, а на датасете `fr3/office` алгоритм показал лучший по значению ATE RMSE результат среди рассматриваемых решений.

С точки зрения времени работы алгоритма существует важное направление, в котором возможно увеличение быстродействия. В представленной реализации алгоритм для набора n изображений оценивает все $\frac{n(n-1)}{2}$ относительных поз. Существуют подходы, упоминающиеся, например, в [33], позволяющие уменьшить число таких сравнений, не реализованные в рамках данной работы.

Увеличение точности возможно путем более тщательного уточнения относительной позы. В частности, в представленной реализации робастная оценка относительной позы уточняется в результате применения

алгоритма ICP только к плотным карт глубин, без учета информации о совпавших ключевых точках. Использование такого рода знаний для минимизации комбинированной (т.е. учитывающей информацию о глубине и соответствии ключевых точек) ошибки репроекции теоретически способно улучшить точность оценивания относительной позы.

6. Результаты

В итоге достигнуты следующие результаты:

1. Произведен обзор глобальных и инкрементальных методов реконструкции.

2. Разработан алгоритм для решения задачи RGB-D реконструкции.

3. Алгоритм реализован (код доступен в репозитории `https://github.com/leoneed03/GDR`).

4. Проведено сравнение точности оценки траектории с аналогами.

Список литературы

[1] Agarwal Sameer, Mierle Keir, Others. Ceres Solver. — `http://ceres-solver.org` (дата обращения 25.04.2021).

[2] Arandjelović Relja, Zisserman Andrew. Three things everyone should know to improve object retrieval // 2012 IEEE Conference on Computer Vision and Pattern Recognition / IEEE. — 2012. — P. 2911–2918.

[3] A Benchmark for the Evaluation of RGB-D SLAM Systems / J. Sturm, N. Engelhard, F. Endres et al. // Proc. of the International Conference on Intelligent Robot Systems (IROS). — 2012. — Oct.

[4] Besl Paul J, McKay Neil D. Method for registration of 3-D shapes // Sensor fusion IV: control paradigms and data structures / International Society for Optics and Photonics. — Vol. 1611. — 1992. — P. 586–606.

[5] Björck Åke. Numerical methods for least squares problems. — SIAM, 1996.

[6] Bundle adjustment in the large / Sameer Agarwal, Noah Snavely, Steven M Seitz, Richard Szeliski // European conference on computer vision / Springer. — 2010. — P. 29–42.

[7] Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration / Angela Dai, Matthias Nießner, Michael Zollhöfer et al. // ACM Transactions on Graphics (ToG). — 2017. — Vol. 36, no. 4. — P. 1.

[8] Cao Yan-Pei, Kobbelt Leif, Hu Shi-Min. Real-time high-accuracy three-dimensional reconstruction with consumer RGB-D cameras // ACM Transactions on Graphics (TOG). — 2018. — Vol. 37, no. 5. — P. 1–16.

[9] Chatterjee Avishek, Govindu Venu Madhav. Efficient and Robust Large-Scale Rotation Averaging // Proceedings of the IEEE

International Conference on Computer Vision (ICCV). — 2013. — December.

[10] Chen Yu, Chen Yisong, Wang Guoping. Bundle Adjustment Revisited // CoRR. — 2019. — Vol. abs/1912.03858. — 1912.03858.

[11] Chum Ondřej, Matas Jiří, Kittler Josef. Locally optimized RANSAC // Joint Pattern Recognition Symposium / Springer. — 2003. — P. 236–243.

[12] Cramér–Rao bounds for synchronization of rotations / Nicolas Boumal, Amit Singer, P-A Absil, Vincent D Blondel // Information and Inference: A Journal of the IMA. — 2014. — Vol. 3, no. 1. — P. 1–39.

[13] CudaSift — URL: `https://github.com/Celebrandil/CudaSift` (дата обращения 06.12.2020).

[14] Cui Zhaopeng, Tan Ping. Global structure-from-motion by similarity averaging // Proceedings of the IEEE International Conference on Computer Vision. — 2015. — P. 864–872.

[15] Eigen — URL: `http://eigen.tuxfamily.org` (дата обращения 06.12.2020).

[16] GTSAM — URL: `https://gtsam.org` (дата обращения 06.12.2020).

[17] Govindu Venu Madhav. Combining two-view constraints for motion estimation // Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001 / IEEE. — Vol. 2. — 2001. — P. II–II.

[18] ICPCUDA — URL: `https://github.com/mp3guy/ICPCUDA` (дата обращения 06.12.2020).

[19] Kerl Christian, Sturm Jürgen, Cremers Daniel. Dense visual SLAM for RGB-D cameras // 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems / IEEE. — 2013. — P. 2100–2106.

[20] Kneip Laurent, Siegwart Roland, Pollefeys Marc. Finding the exact rotation between two images independently of the translation // European conference on computer vision / Springer. — 2012. — P. 696–709.

[21] Lowe David G. Distinctive image features from scale-invariant keypoints // International journal of computer vision. — 2004. — Vol. 60, no. 2. — P. 91–110.

[22] Mäkinen Jari. Rotation manifold SO (3) and its tangential vectors // Computational Mechanics. — 2008. — Vol. 42, no. 6. — P. 907–919.

[23] NVIDIA, Vingelmann Péter, Fitzek Frank H.P. CUDA, release: 10.2.89. — https://developer.nvidia.com/cuda-toolkit (дата обращения 25.04.2021). — 2020.

[24] Nistér David. An efficient solution to the five-point relative pose problem // IEEE transactions on pattern analysis and machine intelligence. — 2004. — Vol. 26, no. 6. — P. 756–770.

[25] Notre Dame SIFT — https://www.cc.gatech.edu/~hays/compvision/proj2/ (дата обращения 06.12.2020).

[26] PCL — URL: https://pointclouds.org (дата обращения 18.04.2021).

[27] Persson Mikael, Nordberg Klas. Lambda twist: An accurate fast robust perspective three point (P3P) solver // Proceedings of the European Conference on Computer Vision (ECCV). — 2018. — P. 318–332.

[28] Pheatt Chuck. Intel® threading building blocks // Journal of Computing Sciences in Colleges. — 2008. — Vol. 23, no. 4. — P. 298–298.

[29] Scherer Sebastian A, Dube Daniel, Zell Andreas. Using depth in visual simultaneous localisation and mapping // 2012 IEEE International Conference on Robotics and Automation / IEEE. — 2012. — P. 5216–5221.

[30] Shonan Rotation Averaging: Global Optimality by Surfing $SO(p)\hat{}n$ / Frank Dellaert, David M Rosen, Jing Wu et al. // European Conference on Computer Vision / Springer. — 2020. — P. 292–308.

[31] SiftGPU — URL: `https://github.com/pitzer/SiftGPU` (дата обращения 06.12.2020).

[32] Sarlin Paul-Edouard, DeTone Daniel, Malisiewicz Tomasz, Rabinovich Andrew. SuperGlue: Learning Feature Matching with Graph Neural Networks. — 2020. — 1911.11763.

[33] Tolias Giorgos, Avrithis Yannis, Jégou Hervé. To aggregate or not to aggregate: Selective match kernels for image search // Proceedings of the IEEE International Conference on Computer Vision. — 2013. — P. 1401–1408.

[34] Umeyama Shinji. Least-squares estimation of transformation parameters between two point patterns // IEEE Transactions on Pattern Analysis & Machine Intelligence. — 1991. — no. 4. — P. 376–380.

[35] Yuan Jin-Yun, Iusem Alfredo Noel. Preconditioned conjugate gradient method for generalized least squares problems // Journal of computational and applied mathematics. — 1996. — Vol. 71, no. 2. — P. 287–297.

[36] Zhang Zhengyou. Parameter estimation techniques: A tutorial with application to conic fitting // Image and vision Computing. — 1997. — Vol. 15, no. 1. — P. 59–76.

[37] An evaluation of the RGB-D SLAM system / Felix Endres, Jürgen Hess, Nikolas Engelhard et al. // 2012 IEEE International Conference on Robotics and Automation / IEEE. — 2012. — P. 1691–1696.

[38] g 2 o: A general framework for graph optimization / Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat et al. // 2011 IEEE International

Conference on Robotics and Automation / IEEE. — 2011. — P. 3607–3613.