Санкт-Петербургский государственный университет

Иванов Кирилл Андреевич

Выпускная квалификационная работа

Детекция и ассоциация линий на RGB-изображениях для SLAM-алгоритмов

Уровень образования: бакалавриат

Направление *09.03.04 «Программная инженерия»*

Основная образовательная программа *СВ.5080.2019 «Программная инженерия»*

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

Рецензент:
научный сотрудник, Мюнхенский технический университет, к.ф.-м.н. А. В. Артемов

Санкт-Петербург
2023

Saint Petersburg State University

Kirill Ivanov

Bachelor's Thesis

Line Detection and Assciation in RGB Images for SLAM Algorithms

Education level: bachelor

Speciality *09.03.04 "Software Engineering"*

Programme *CB.5080.2019 "Software Engineering"*

Scientific supervisor:
C.Sc., System Programming chair docent Y. V. Litvinov

Reviewer:
C.Sc., research scientist at Technical University of Munich A. V. Artemov

Saint Petersburg
2023

Оглавление

Введение

Распознавание объектов, выделяющихся на изображениях (так называемых *ориентиров*), является важной частью различных алгоритмов компьютерного зрения. Так, ориентиры активно используются в системах одновременной локализации и построения карты (*Simultaneous Localization And Mapping*, SLAM) — программно-аппаратных комплексах, позволяющих автономному роботу определять свое местоположение в неизвестном окружении, при этом формируя для него карту. При помощи выявления ориентиров (задача детекции) и определения одних и тех же ориентиров на последовательных изображениях (задача ассоциации) можно оценивать траекторию движения системы и, как следствие, уточнять карту окружения и местоположение робота.

Точки являются наиболее популярными ориентирами в системах одновременной локализации и построения карты. Согласно исследованиям [65, 61], многие SLAM-технологии, основанные на точках (например, ORB-SLAM2 [57]), работают эффективно в условиях большого количества уникальных, ярко выраженных ориентиров. В то же время они часто выдают неточные результаты в окружениях с однотонными текстурами. Последние, однако, зачастую содержат большое количество структурных объектов — плоскостей и линий — использование которых может существенно улучшить распознавание геометрии сцены и, как следствие, увеличить точность и устойчивость оценки траектории движения автономной системы. Поэтому в настоящее время началось активное применение таких ориентиров в SLAM-технологиях.

На данный момент подавляющее большинство систем одновременной локализации и построения карты, использующих линии, [62, 61, 93, 65, 76, 99] применяют для детекции алгоритм LSD [94] или его модификации, а для ассоциации — LBD [94] или его вариации. Указанные алгоритмы обеспечивают требуемое быстродействие, однако за последние годы появилось множество детекторов [19, 82, 83, 47, 25, 79] и ассоциаторов [92, 50, 34, 84, 89, 67] линий, которые, согласно экспериментам авторов, имеют лучшее качество распознавания геометрии сцены при сравнимой производительности.

Выбрать оптимальные алгоритмы для использования в системе одновременной локализации и построения карты проблематично во многом из-за отсутствия универсального бенчмарка (метрик и наборов данных) для оценки качества детекции и ассоциации, включающего их тестирование на популярных SLAM-последовательностях. Наличие такого бенчмарка позволит эффективно сравнивать новые алгоритмы с уже существующими, а также выбирать подходящие SLAM-алгоритмы для конкретного окружения.

Таким образом, является актуальной проблема оценки алгоритмов детекции и ассоциации линий для использования в SLAM-системах, которая и будет исследована в рамках данной работы.

1. Постановка цели и задач

Целью работы является реализация бенчмарка для оценки качества и производительности алгоритмов детекции и ассоциации линий в задаче SLAM и сравнение с его помощью существующих алгоритмов. Для достижения цели были поставлены следующие задачи.

1. Провести обзор существующих алгоритмов детекции и ассоциации линий, метрик, а также собрать статистику их использования в SLAM-системах.
2. Подготовить наборы данных, основанные на популярных SLAM-последовательностях и пригодные для оценки детекторов и ассоциаторов линий.
3. Предложить формат унифицированного запуска алгоритмов и реализовать его для рассмотренных алгоритмов.
4. Реализовать библиотеку с метриками детекции и ассоциации линий.
5. Провести экспериментальное исследование существующих алгоритмов детекции и ассоциации линий.

2. Обзор

В данном разделе рассматриваются существующие алгоритмы детекции и ассоциации линий, метрики и наборы данных для их оценки, а также приводится статистика их использования в SLAM-системах. С полным списком статей, рассмотренных в ходе работы, можно ознакомиться на сайте[1].

2.1. Алгоритмы детекции линий

На данный момент существует немало алгоритмов детекции линий: как традиционных, так и основанных на глубоком обучении. На рисунке 1 представлен хронологический обзор существующих детекторов. Рассмотрим далее более подробно основные традиционные подходы детекции сегментов линий, опираясь на классификацию из статьи [98], а также появившиеся за последние годы нейросетевые алгоритмы.

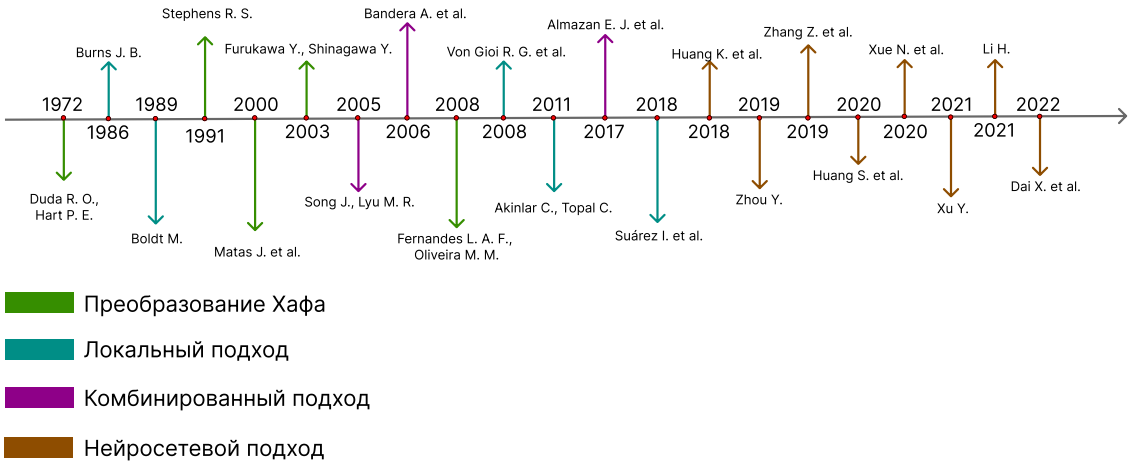


Рис. 1: Хронологический обзор ключевых детекторов линий

2.1.1. Подход, основанный на преобразовании Хафа

Преобразование Хафа [26] заключается в переходе от пространства изображения к пространству параметров и последующем поиске в нем экстремальных значений, которые соответствуют линиям в исходном пространстве, с применением процедуры голосования. Традиционный основанный на проебразовании Хафа алгоритм [13] требует обработки всех пикселей изображения, что существенно сказывается на производительности алгоритма. Для решения этой проблемы были предложены различные вероятностные преобразования Хафа [72, 52, 7, 18, 90], которые позволяют проводить процесс голосования лишь на подмножестве пикселей. Важно также отметить, что

[1] https://amethyst-crane-ddd.notion.site/Line-detection-and-association-a4f42930d06c4e9798c51b8f3eec4c45 (дата обращения: 2023-04-17)

преобразование Хафа в общем случае не предоставляет информацию о концах получаемых линий, таким образом, даже небольшой сегмент будет задетектирован как линия, протяженная на всё изображение. Для решения этой проблемы были предложены алгоритмы [3, 30, 20, 91], анализирующие окрестности вокруг пиковых значений в параметрическом пространстве, которые содержат необходимую информацию о протяженности линии. В целом, данная группа подходов требует подгонки параметров алгоритмов под конкретные окружения, что делает её малоприменимой в современных алгоритмах.

2.1.2. Локальный подход

Алгоритмы подхода «снизу-вверх», также известного как локальный подход, обычно состоят из следующих шагов.

1. Создание первичного множества мелких линейных сегментов.
2. Итеративное связывание мелких сегментов в более крупные на основе анализа соседних элементов.
3. Фильтрация ложных сегментов на основе некоторого критерия.

Алгоритм Бернса [9], предложенный одним из первых в этой группе, сначала строит регионы поддержки линий, далее сравнивает схожесть градиентов соседних пикселей и сцепляет их, если они отличаются менее, чем на пороговое значение. Kahn и др. предложили оптимизацию алгоритма Бернса [29], используя ускоренную процедуру обработки пикселей. Boldt M. и др. [6] предложили использовать символические токены (*symbolic tokens*), представляющие сегменты линий, и связывать их в итеративной манере для получения более крупных сегментов. Авторы исследования [33] показали, что более ранние детекторы работают довольно медленно, склонны распознавать большое количество несущественных сегментов линий, а также многие из них требуют настройки параметров под конкретное окружение. Так, ими был предложен алгоритм LSD, который не имел настраиваемых параметров, включал процедуру фильтрации ложных сегментов и существенно превосходил предшественников в скорости работы. Еще большего ускорения работы детекции линий достигли авторы алгоритма EDLines [1], который, по данным авторов, работает в несколько раз быстрее, чем LSD, сохраняя его преимущества. Достигнуто это было путём создания специализированной процедуры извлечения краёв из изображений, спроектированной специально для работы с сегментами. Авторы исследования [60] разработали алгоритм, предназначенный для детекции линий на открытых пространствах. В работе [10] был предложен новый способ представления сегмента линии, который авторы назвали linelet. Это представление моделирует внутренние свойства линий в виде неориентированного графа, что позволяет построить не только вероятностную структуру проверки

задетектированного сегмента, но и структуру агрегации, обеспечивающую связывание сегментов, имеющих схожие внутренние свойства. В работе [15] авторы исследовали надежность оценок, выдаваемых детекторами, и предложили алгоритм, состоящий из двух независимых модулей: модуля генерации гипотез, который жадно группирует сегменты, предлагая вероятные кандидаты на линии, и модуль вероятностной валидации, решающий, является ли группа сегментов реальной линией. Также недавно был предложен детектор ELSED [77], превосходящий описанные ранее алгоритмы по скорости работы и качеству детекции.

2.1.3. Комбинированный подход

Алгоритмы комбинированного подхода используют как информацию, получаемую из входного изображения, так и из параметрического пространства преобразования Хафа. Так, в работе [71] преобразование Хафа было ускорено при помощи включения в процесс голосования информации о градиенте. В исследованиях [55, 48] были применены процедуры генерации выборки пикселей из массива накопления голосов, которые с высокой вероятностью принадлежат некоторому сегменту и к которым впоследствии был применен алгоритм итеративного связывания в сегменты. Авторы работы [54] предложили двухэтапный детектор: на первом этапе линии обнаруживаются с использованием вероятностного преобразовния Хафа, а на втором — анализируется в области изображения для локализации сегментов, которые генерируют пик в параметрическом пространстве.

2.1.4. Нейросетевой подход

Благодаря развитию глубокого обучения в последнее десятилетие появилось множество нейросетевых алгоритмов для детекции сегментов линий. Также была сформулирована задача парсинга каркаса [39], заключающаяся в нахождении набора сегментов на изображении, а также точек их сочленения. Начиная с работы Huang K. и др. [39], стали появляться различные нейросетевые детекторы линий. Так, в работе [37] проблема детекции линий была сведена к проблеме раскраски регионов на изображении. Zhang Z. и др. [64] предложили использовать граф для парсинга каркаса, в котором узлы обозначают точки сочленения, а ребра между ними — сегменты линий. В работе [95] был предложен нейростетевой детектор, работающий в сквозной (*end-to-end*) манере. Lin Y. и др. [45] предложили использовать обучаемый модуль, выполняющий преобразование Хафа, что позволило достичь увеличения точности выдаваемых результатов. В статье [79] был предложен новый трехточечный (*tri-point*) способ представления линий. Авторами работы [32] была рассмотрена применимость графовых нейронных сетей в задаче детекции линий, в результате был разработан детектор, обеспечивающий возможность работы в реальном времени. В работе [25] бы-

ла предложена компактная параметризация сегментов линий, что позволило достичь относительного превосходства в точности оценки линий над другими нейросетевыми детекторами, а также достичь сопоставимой с LSD скорости работы, что обеспечило применимость этого алгоритма в системах реального времени. Авторы исследования [47] предложили использовать модули кодирования и декодирования в нейросетевой архитектуре детектора, что позволило достичь улучшения качества детекции относительно предыдущего алгоритма, который, тем не менее, работает в несколько раз быстрее. В статье [83] была рассмотрена проблема детекции линий на изображениях, полученных камерами различных видов, а также было предложено представление сегментов линий в виде кривых Безье, что позволило разработать нейросетевой детектор, способный работать с изображениями, захваченными камерами типа «рыбий глаз», а также сферическими и традиционными камерами. Авторы работы [82] предложили нейросетевой детектор с облегченной архитектурой, позволяющий достигать высокой скорости работы на мобильных устройствах, однако с некоторыми потерями точности детекции. В исследовании [81] также рассматривались вопросы ускорения работы нейросетевых детекторов, так, авторами был предложен алгоритм, основанный на классическом LSD, в который была встроена легковесная сверточная нейронная сеть. Как показывают эксперименты, предложенная реализация превосходит в скорости работы как описанные ранее нейросетевые алгоритмы детекции, так и LSD. В работе [19] был предложен детектор, построенной на полностью сверточной нейронной сети, что позволило отказаться от вычислительно тяжелого модуля генерации гипотез, который использовался в большинстве описанных ранее работ, и принесло существенный прирост скорости вычисления.

2.2. Алгоритмы ассоциации линий

Рассмотрим теперь основные подходы ассоциации линий, основываясь на классификации, представленной в работе [49]. На рисунке 2 представлен хронологический обзор существующих ассоциаторов.

2.2.1. Подход, основанный на дескрипторах линий

Дескриптор сегмента линии — это агрегатор локальной структуры в окрестностях сегмента, представленный, как правило, в виде вещественного вектора фиксированной размерности. Ассоциация между сегментами устанавливается, если расстояние между соответствующими дескрипторами меньше заданного порога. Во множестве ранних работ [56, 68, 4, 69, 53] использовался данный подход для построения алгоритмов ассоциации, дескриптор же мог включать информацию о градиенте, интенсивности и цвете в окрестностях линий. Эти алгоритмы, однако, не были устойчивыми к изменению освещенности и масштаба, а также могли выдавать неточные результаты в

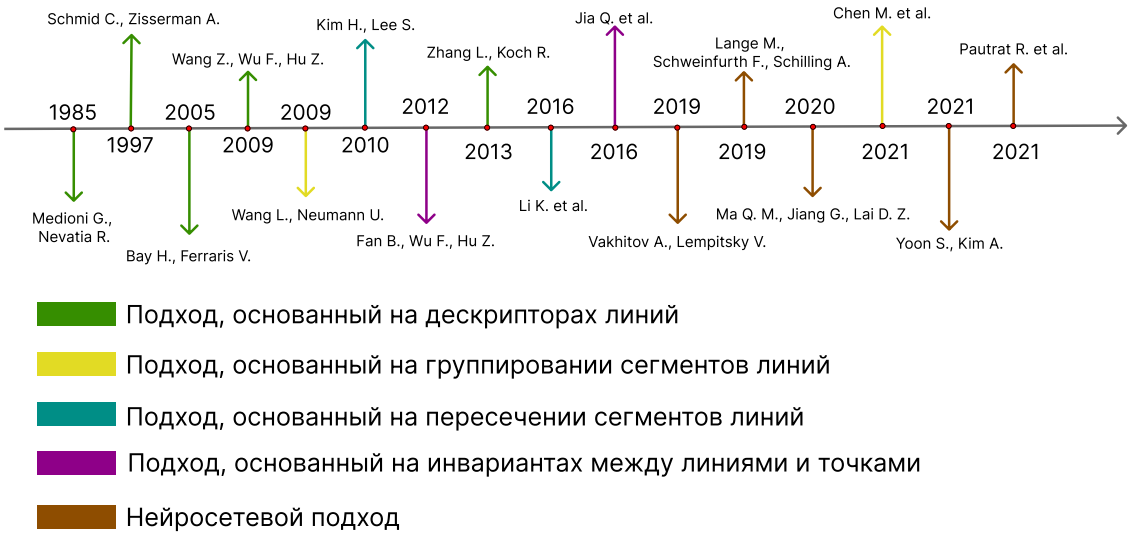


Рис. 2: Хронологический обзор ключевых ассоциаторов линий

окружениях с малым количеством текстур. Многие из последующих работ были нацелены на решение описанных проблем. Так, авторами исследования [5] был предложен алгоритм, который выдавал большое количество соответствий даже в условиях однотипности окружения и использовал итеративную процедуру увеличения множества потенциальных кандидатов на соответствие. В работе [88] был предложен алгоритм MSLD, основанный на дескрипторах, не требующих априорных знаний о сцене, что позволило получить более надежные результаты в условиях недостаточной освещенности. Впоследствии были предложены ассоциаторы, оптимизирующие предыдущий в скорости работы [24] и в точности выдаваемых соответствий [85]. В работе [94] был предложен дескриптор LBD, а проблема нахождения соответствий была выражена в виде задачи нахождения паросочетания в графе отношений между рассматриваемыми сегментами линий.

2.2.2. Подход, основанный на пересечении сегментов линий

Алгоритмы данной категории сопоставляют сегменты линий попарно, основываясь на точках пересечения линий, содержащих эти сегменты. Так, в работе [31] авторы предложили использовать компланарность пар пересекающихся линий для отделения истинных соответствий от ложных. В работе [44] был предложен дескриптор типа «луч-точка-луч», агрегирующий информацию о линиях и точке их пересечения. Li K. и др. [22] предложили иерархический двухэтапный алгоритм, сопоставляющий дескрипторы типа «луч-точка-луч» на первом этапе, и дескрипторов отдельных линий — на втором. Данный ассоциатор, как показали эксперименты, способен выдавать осмысленные результаты в условиях недостаточной освещенности и малого количества текстур.

2.2.3. Подход, основанный на инвариантах между линиями и точками

Ассоциаторы, использующие инварианты между компланарными точками и сегментами линий, обычно сначала устанавливают соответствия между точками, используя известные алгоритмы, например, ORB [59]. После этого происходит поиск сегментов линий, удовлетворяющих инвариантам, которые рассматриваются как истинные соответствия. На данный момент существует два вида инвариантов между компланарными точками и отрезками линий: аффинный и проективный. В работе [16] Fan B. и др., используя алгоритм на основе аффинного инварианта, смогли достичь сравнимые с MLSD результаты ассоциации, а в последующей их работе [17] был предложен проективный инвариант и была увеличена надежность получаемых соответствий. Дальнейшее развитие данный подход нашел в работах [58, 46]: в них авторы предложили дополнительно оценивать гомографию на основе инвариантов, что позволило существенно увеличить количество получаемых соответствий.

2.2.4. Подход, основанный на группировании сегментов линий

Алгоритмы, использующие данный подход, основаны на создании групп сегментов линий по некоторому критерию с последующим их сопоставлением. Так, Wang L. и Neumann U. [87] предложили формировать группы из линейных сегментов, лежащих рядом, на основе схожести средних величин градиента всех пикселей сегментов. Было установлено, что одна и та же группа линий с высокой вероятностью появится на последовательных изображениях. Также авторами было экспериментально показано, что предложенный метод превосходит алгоритмы, основанные на дескрипторах отдельных линий, в условиях однотипности текстур. Авторы работы [23] предложили разбивать все задетектированные сегменты линий на три иерархические группы, после чего устанавливать соответствия последовательно для каждой группы, используя информацию, полученную из ассоциаций предшествующей группы.

2.2.5. Нейросетевой подход

За последние годы, помимо детекторов, появились также нейросетевые ассоциаторы. В таких алгоритмах наиболее популярны нейросетевые архитектуры, построенные на основе сверточных слоев. Так, в работе Вахитова А. и Лемпитского В. [84] сверточная нейронная сеть была применена для построения дескрипторов отдельных линий, а также было показано, что предложенный алгоритм превосходит LBD в качестве сопоставления линий в задаче SLAM. M. Lange и др. в работе [35] предложили ассоциатор линий DLD, построенный на основе ResNet [11] — популярной в компьютерном зрении нейронной сети. Впоследствии этими же авторами была предложена оптимизированная версия предыдущего алгоритма — WLD [34]. Как показывают эксперименты, обе модели существенно превосходят классические подходы с

точки зрения качества ассоциации линий. Хотя дескрипторы линий на основе сверточных нейронных сетей показывают неплохие результаты в изменяющихся условиях, они, тем не менее, как указывают авторы работы [92], имеют архитектурный недостаток, связанный с абстрагированием переменной длины линии путем создания дескрипторов фиксированной размерности. Так, ими был предложен алгоритм ассоциации, вдохновленный моделями обработки естественного языка, в которой линия была представлена как последовательность точек (слов). Авторы статьи [67] предложили представление дескриптора линии как последовательности дескрипторов точек, мотивированные успехами последних работ в этой области. Было установлено, что предложенный алгоритм выдает сравнимые с WLD результаты ассоциации, однако существенно превосходит последний в устойчивости к различным искажениям изображений. Еще одним направлением развития ассоциаторов на основе глубокого обучения стало применение графовых сверточных сетей (*Graph Convolution Networks*, GCN), описанное в работе [50], где задача ассоциации сегментов линий была сведена к транспортной задаче.

2.3. Системы SLAM, использующие линии

Применение линий в качестве ориентиров началось еще в использующих фильтр Калмана SLAM-системах [70, 28, 43, 74], которые, однако, в настоящее время активно не используются — их заменили более современные графовые SLAM-технологии. Поскольку основным фокусом работы являются алгоритмы детекции и ассоциации линий, подробный обзор использующих линии SLAM-систем не проводится, однако приведем таблицу 1, показывающую, какие ассоциаторы и детекторы используются в существующих работах. Нетрудно заметить, что наиболее популярным детектором является LSD, а ассоциатором — LBD.

Публикация	Детектор линий	Ассоциатор линий
Zhang G. et al. [8]	FLD [60]	Предложенный
Pumarola A. et al. [62]	LSD	LBD
Zuo X. et al. [66]	LSD	LBD
Liu Y. et al. [73]	LSD	LBD
Wang R. et al. [27]	LSD	LBD
Lee T. et al. [42]	LSD	—
Gomez-Ojeda R. et al. [61]	LSD	LBD
Fu Q. et al. [99]	LSD	LBD
Lee S. J., Hwang S. S. [41]	LSD	LBD
Zhang F. et al. [36]	LSD	Предложенный
Qian K. et al. [86]	LSD	Предложенный
Li Y. et al. [76]	LSD	LBD
Fu Q. et al. [63]	LSD (мод.)	LBD
Li Y. et al. [65]	LSD	LBD
Lee J., Park S. Y. [40]	LSD	LBD
Yunus R., Li Y. [93]	LSD	LBD
Shu F. et al. [75]	LSD	LBD

Таблица 1: Использующиеся в SLAM-системах детекторы и ассоциаторы линий.

2.4. Метрики

В данном подразделе описываются популярные метрики детекции и ассоциации линий.

2.4.1. Метрики детекции линий

Существует достаточно большое количество метрик для оценки работы алгоритмов детекции как с точки зрения качества выдаваемых результатов, так и с точки зрения скорости их работы. Так, для оценки производительности можно использовать среднее время работы алгоритма и среднее количество обрабатываемых кадров в секунду. Оценивать качество детекции, в свою очередь, можно на основании простых количественных метрик — средней длины детектируемых сегментов и среднего их количества на изображениях. Несомненным преимуществом таких метрик является отсутствие необходимости в референсных значениях, однако, используя их, трудно оценить качество детектора, поэтому далее такие метрики рассматриваться не будут.

С другой стороны, существуют качественные метрики оценки детекции [39, 37, 64, 67], которые можно разбить на две большие группы по представлению линии:

метрики, основанные на тепловой карте, и векторные метрики. Рассмотрим далее эти две группы метрик.

Метрики, основанные на тепловой карте (*heatmap metrics*), [39, 51] рассматривают детекцию как задачу классификации каждого пикселя с точки зрения принадлежности какой-либо линии. Под тепловой картой будем понимать булеву матрицу, имеющую такой же размер, что и исходное изображение, и полученную путем растеризации предсказанных или референсных сегментов линий так, что каждый её элемент отражает принадлежность соответствующего пикселя какому-либо сегменту. Для вычисления метрики необходимо построить тепловые карты предсказанных и референсных сегментов линий, после чего определить, какие пиксели на предсказанной тепловой карте являются TP (True Positive), а какие — FP (False Positive). Для этого необходимо решить задачу о назначениях минимального веса, в которой узлами являются пиксели, а весами — расстояния между ними. В рассмотрение попадают лишь те элементы из предсказанной тепловой карты, которые помечены как истинные и расстояние от которых до некоторого референсного пикселя меньше, чем заданный порог d_{max}. Остальные помеченные как истинные пиксели рассматриваются как FP. После решения данной задачи мы имеем матрицу индикаторов, является ли соответствующий пиксель TP или нет, на основании которой можно высчитать такие классические метрики классификации, как точность (precision), полнота (recall), F-мера и усредненная точность (average precision).

Векторные метрики, в отличие от тепловых, учитывают наложение и пересечение сегментов и включают метрики классификации [95] и повторяемости [67]. Для их вычисления необходимо определить некоторую функцию расстояния $d(l_1, l_2)$, где $l_i = (p_i^1, p_i^2)$ — сегмент линии, определяемый граничными точками $p_i^1, p_i^2 \in \mathbb{R}^2$. Можно выделить следующие функции расстояния.

- **Структурное расстояние** [95]:

$$d_s(l_i, l_j) = \min\{||p_j^1 - p_i^1||_2^2 + ||p_j^2 - p_i^2||_2^2, ||p_j^1 - p_i^2||_2^2 + ||p_j^2 - p_i^1||_2^2\}. \tag{1}$$

- **Ортогональное расстояние** [67]:

$$d_\perp(l_i, l_j) = \frac{d_a(l_i, l_j) + d_a(l_j, l_i)}{2}, \tag{2}$$

  где $d_a(l_1, l_2) = \|p_2^1 - p_{l_1}(p_2^1)\|_2 + \|p_2^2 - p_{l_1}(p_2^2)\|_2,\ p_l(\cdot)$ – ортогональная проекция на линию l.

Для вычисления векторных метрик классификации нужно определить для каждого предсказанного сегмента, является он истинным или нет. Задетектированный сегмент линии l_j рассматривается как истинный (TP) тогда и только тогда, когда выполнены следующие условия:

- $\min_{l \in GT} d(l_j, l) \leq d_{max}$, где d_{max} — максимальное допустимое значение расстояния между предсказанным и референсным сегментами, а GT – множество референсных сегментов;

- если существует l_i, что $\arg\min_{l \in GT} d(l_i, l) = \arg\min_{l \in GT} d(l_j, l) = l'$, то l_j должен иметь более высокий уровень уверенности детектора, чем l_i, в случае, когда детектор предоставляет значения степени уверенности для каждого сегмента. Если же детектор таких значений не предоставляет, то должно выполняться неравенство $d(l_j, l') < d(l_i, l')$.

Все остальные предсказанные сегменты помечаются как FP. Важно также отметить, что обычно перед вычислением метрики все сегменты линий масштабируются до разрешения 128×128.

После того, как TP и FN сегменты определены, как и в случае основанных на тепловой карте метрик, вычисляются классические метрики классификации.

Метрики повторяемости отражают долю одних и тех же линий, обнаруженных на разных кадрах. Пусть L_1 и L_2 — наборы линий первого и второго изображений соответственно, а $P_{i \to j}(\cdot)$ — преобразование, отображающее линии i-го изображения на j-е изображение. Пусть $L_1' = P_{1 \to 2}(L_1)$ и $L_2' = P_{2 \to 1}(L_2)$. Также нам необходимо определить следующий индикатор того, что расстояние от линии l до множества линий L меньше порога d_{max}:

$$I_L(l) = \begin{cases} 1, & \text{если } \min_{l_j \in L} d(l, l_j) \leq d_{max} \\ 0, & \text{иначе} \end{cases}, \tag{3}$$

где $d(\cdot, \cdot)$ — либо ортогональное (2), либо структурное (1) расстояние. Теперь мы можем определить метрику *повторяемости*, отражающую долю линий, найденных как на первом, так и на втором кадрах, и *ошибку локализации* — среднее расстояние между такими линиями.

- **Повторяемость.**

$$\text{Rep-}d_{max} = \frac{\sum_{l_i \in L_1'} I_{L_2}(l_i) + \sum_{l_j \in L_2'} I_{L_1}(l_j)}{|L_1| + |L_2|}. \tag{4}$$

- **Ошибка локализации.** Пусть $Dist(L_j, L_i)$ — множество ближайших расстояний от каждой линии в L_j до некоторой линии в L_i в пределах порога d_{max}, которое определяется следующим образом:

$$Dist(L_j, L_i) = \left\{ \min_{l_i \in L_i} d(l_i, l) \;\middle|\; I_{L_i}(l) = 1, l \in L_j \right\}, \tag{5}$$

тогда *ошибка локализации* определяется следующим образом:

$$\text{LE-}d_{max} = \frac{\sum_{d \in Dist(L'_2, L_2) \cup Dist(L'_1, L_1)} d}{|Dist(L'_2, L_2)| + |Dist(L'_1, L_1)|}. \tag{6}$$

Здесь $|\cdot|$ — мощность множества.

2.4.2. Метрики ассоциации линий

Оценить качество ассоциатора линий можно как с использованием только предсказанных, так и с использованием референсных данных. В первом случае для установления истинных соответствий каждый сегмент $l_{1,i} \in L_1$ первого изображения перепроецируются, например, используя известные матрицы гомографии [92] или карты глубины [50, 67], на второе изображение, получая $l'_{1,i} \in L'_1$. Затем пары сегментов $(l_{1,i}, l_{2,j})$ объявляются истинными соответствиями, если $d(l'_{1,i}, l_{2,j}) < d_{max}$, где $d(\cdot, \cdot)$ — функция расстояния, например, одна из описанных ранее (1), (2). В случае оценки на референсных данных точные соответствия между линиями на последовательных кадрах полагаются известными.

После формирования набора истинных соответствий GT появляется возможность отнести каждое предсказанное соответствие к одному из следующих классов:

- **TP**, если соответствие установлено и принадлежит GT набору;
- **FP**, если соответствие установлено и не принадлежит GT набору;
- **TN**, если соответствие не установлено и не принадлежит GT набору;
- **FN**, если соответствие не установлено и принадлежит GT набору.

На основании полученных характеристик, как и в случае детекции, вычисляются классические метрики классификации.

Также выявленные ассоциации линий можно использовать для получения оценки относительной позиции между кадрами (вектора смещения и матрицы вращения одного кадра относительно другого), для которых эти ассоциации были посчитаны. Так, оценить качество ассоциации можно при помощи **ошибок относительной позиции** (relative pose error) [38, 78]. Пусть $T = [\mathbf{R}|\mathbf{t}]$, $\widehat{T} = [\widehat{\mathbf{R}}|\hat{\mathbf{t}}]$ — истинная относительная позиция и её оценка соответственно, где $\mathbf{R}, \widehat{\mathbf{R}} \in SO(3), \mathbf{t}, \hat{\mathbf{t}} \in \mathbb{R}^3$. Также пусть $\Delta T = T\widehat{T}^{-1} = [\Delta\mathbf{R}|\Delta\mathbf{t}]$ — разница между позициями. Тогда мы можем рассчитать следующие показатели.

- **Ошибка смещения**:

$$\epsilon_{trans}(\mathbf{t}, \hat{\mathbf{t}}) = \|\Delta\mathbf{t}\|. \tag{7}$$

- **Ошибка вращения**:

$$\epsilon_{rot}(\mathbf{R}, \widehat{\mathbf{R}}) = \text{rad2deg}\left(\arccos\frac{\text{Tr}(\Delta\mathbf{R}) - 1}{2}\right). \tag{8}$$

2.5. Наборы данных

С развитием SLAM-систем, использующих разные типы датчиков, появилось немалое количество наборов данных, пригодных для их оценки с точки зрения производительности и точности получаемой траектории. Рассмотрим подробнее датасеты, которые упоминаются в работах, посвященных использующим линии SLAM-системам.

- **EuRoC MAV** [14] — набор визуально-инерционных данных, собранный с использованием микролетательного аппарата (*Micro Aerial Vehicle*, MAV). Данные были получены в жилых и промышленных помещениях и содержат синхронизированные стереоизображения, измерения гиростабилизатора и истинные траектории движения.

- **TUM RGB-D** [97] — набор данных, записанный внутри складских и офисных помещений, включающий RGB-изображения и карты глубины, полученные с использованием Microsoft Kinect.

- **ICL-NUIM** [96] — синтетический набор данных, включающий два помещения — жилую комнату и офис. Данные содержат RGB изображения, карты глубины и истинные значения позы камеры.

- **KITTI** [21] — набор данных, записанных на открытых пространствах при помощи установленной на машине системы камер. Точная траектория была получена с использованием LiDAR-сканеров и GPS.

- **OpenLORIS** [2] — набор данных, полученный при помощи зафиксированных на роботе камер в реальных динамических окружениях: кафе, торговых центрах, жилых комнатах. Точная траектория робота была получена при помощи LiDAR-сканеров и системы захвата движений.

- **TartanAir** [80] — набор синтетических фотореалистичных данных, включающий траектории с различными погодными условиями и освещенностью, а также точные позы камеры с различными измерениями, такими как RGB-изображения, карты глубины, облака точек.

Также были найдены следующие два набора данных, содержащие размеченные вручную сегменты линий.

- **ShanghaiTech Wireframe** [39] — набор данных, содержащий изображения различных жилых помещений. Были размечены лишь структурные сегменты линий, такие как потолки, полы, стены, двери, линейные элементы мебели. Линейные с определенного ракурса и текстурные элементы не размечались.
- **YorkUrban** [12] — набор данных, содержащий изображения городской среды, а также предоставляющий ручную разметку.

Последние два набора данных хотя и предоставляют размеченные вручную линии, тем не менее, не имеют последовательных траекторий изображений, что делает их малопригодными для оценки качества ассоциации и оценки SLAM-алгоритмов, использующих линии.

2.6. Вывод

Таким образом, можно заключить, что большинство современных использующих линии SLAM-систем применяют довольно устаревшие алгоритмы детекции и ассоциации — LSD и LBD. В то же время, основываясь на проведенном обзоре, можно заметить наличие алгоритмов, имеющих сравнимую с LSD и LBD скорость работы, но превосходящих их в точности детекции и ассоциации.

Также немаловажным фактором, замедляющим применение современных детекторов и ассоциаторов в SLAM-системах, является отсутствие датасета с разметкой линий на последовательных кадрах, поэтому видится целесообразным создание такого набора данных.

3. Наборы данных

Как уже было указано ранее, отсутствие последовательных датасетов с размеченными линиями затрудняет оценку существующих детекторов и ассоциаторов в задаче SLAM, поэтому было принято решение создать такие наборы данных самостоятельно, основываясь на популярных SLAM-последовательностях. В данной секции будет описан выбор данных для разметки сегментов линий, а также процесс разметки.

3.1. Выбор датасетов

Для разметки сегментов линий были выбраны популярные датасеты ICL-NUIM [96] и TUM RGB-D [97]. Их выбор обусловлен еще и тем, что параллельно данной работе в лаборатории мобильной робототехники Сколтех ведется тестирование алгоритмов плоскостей на этих наборах данных, что впоследствии облегчит создание единого бенчмарка тестирования линий, точек и плоскостей для задачи SLAM.

Разметка сегментов линий — это довольно кропотливый и трудоёмкий процесс, требующий внимательности разметчика. Поэтому, ввиду ограниченности ресурсов, было принято решение размечать лишь некоторые траектории из указанных датасетов. Так, из ICL-NUIM были выбраны траектории `lr kt2` и `of kt2`, а из TUM RGB-D — `fr3/cabinet` и `fr1/desk`.

3.2. Процесс разметки линий

Для разметки линий необходим инструмент, основным критерием для выбора которого было наличие следующей функциональности:

- отслеживание одной и той же линии на последовательных кадрах;
- возможность выгрузки треков линий на последовательных кадрах, где трек линии — это последовательность координат, которые соответствуют расположению линии на каждом кадре.

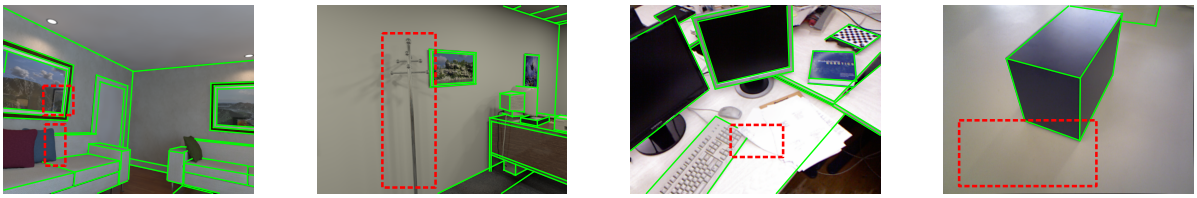
Данная функциональности необходима для составления референсных данных, используемых для оценки ассоциации. Так, были выявлены два обладающих указанной функциональностью инструмента — Scalabel[2] и CVAT[3]. Первый, однако, не имел поддержки общепринятых в компьютерном зрении форматов данных, поэтому для последующей разметки был выбран CVAT.

Для разметки был выделен обученный специалист, взаимодействие которым включало следующие мероприятия:

[2] https://www.scalabel.ai/ (дата обращения: 2022-10-20)
[3] https://www.cvat.ai/ (дата обращения: 2022-10-20)

- консультирование по вопросам разметки: какие сегменты стоит разметить, а какие — стоит игнорировать;
- проверка промежуточных результатов разметки и при необходимости указание возможных улучшений.

Процесс разметки состоял из следующих этапов: определение линий на первом кадре, интерполяция линий на последующих кадрах с использованием функциональности CVAT и корректировка линий на промежуточных кадрах. В итоге с каждой линией был связан её трек. Если линия отсутствовала на сцене, то она исключалась из трека на соответствующих кадрах. Были размечены только структурные сегменты линий, такие как потолки, полы, стены, двери и линейные элементы мебели. Примеры разметки представлены на рисунке 3.



(a) lr kt2 (b) of kt2 (c) fr2/desk (d) fr3/cabinet

Рис. 3: Примеры размеченных линий (выделены зеленым) на траекториях из ICL-NUIM и TUM RGB-D. Красным цветом отмечены элементы, которые не были размечены: отражения, элементы, образующие линии под определенным углом, и тени

3.3. Постобработка данных

Для представления размеченных линий был выбран формат CVAT 1.1, так как он позволяет хранить линии в виде треков, которые сохраняют информацию об ассоциациях на всей последовательности кадров. Также представленные в этом формате линии могут быть легко дополнены или скорректированы средствами CVAT. Тем не менее работать с этим форматом напрямую затруднительно, поскольку метрики ассоциации оперируют с парами соответствий, а метрики детекции — с наборами линий для каждого кадра.

Для упрощения дальнейшего построения оценок метрик были написаны скрипты на Python для преобразования треков линий в данные, пригодные для оценки ассоциации и детекции. Таким образом, полученные наборы данных включают разметку линий для последовательностей в виде треков, скрипт для извлечения данных для оценки ассоциации, а также скрипт для извлечения данных для оценки детекции. Наборы данных размещены в облачном хранилище[4] и снабжены инструкцией по их использованию.

[4] https://disk.yandex.com/d/DLAOGP6FI_27hQ (дата обращения: 2023-04-17)

4. Унификация запуска алгоритмов детекции и ассоциации линий

Для множества детекторов [19, 33, 82, 83, 77, 45, 95, 47, 25, 79, 67, 37, 15, 60, 1, 48, 18, 52, 13, 81] и ассоциаторов [34, 35, 67, 47, 94] были найдены открытые реализации. Их применение в рамках единого бенчмарка, однако, затруднено различием в интерфейсах, а также многочисленными внешними зависимостями и требованием компиляции отдельных модулей. Исходя из выявленных проблем, было решено использовать Docker в качестве средства изоляции окружений отдельных алгоритмов и управления необходимыми зависимостями.

Для решения проблемы различности интерфейсов было принято решение написать адаптеры для каждого алгоритма, используя язык Python, поскольку большинство найденных реализаций написаны на нем. Таким образом, процесс унификации запуска алгоритмов состоял из следующих этапов.

1. Написание базовых адаптеров для нейросетевых и традиционных алгоритмов, которые содержат общий код.

2. Написание адаптеров для отдельных алгоритмов. Поскольку многие детекторы были написаны на языке C++, потребовалось также создать небольшие обвязки, позволяющие запускать их, используя Python.

3. Упаковка адаптированного алгоритма в Docker-контейнер.

Так как выдаваемые детекторами результаты занимают достаточно мало памяти, было принято решение представлять их в виде текстового файла в формате `CSV`, в котором каждая строка имеет вид (x_1, y_1, x_2, y_2), где $(x_1, y_1), (x_2, y_2)$ — координаты концов задетектированного отрезка. Некоторые детекторы также предоставляют набор значений, характеризующих меру неопределенности для каждого сегмента, — список очков (*scores*). Результатом работы таких детекторов на каждом изображении будет являться пара `CSV`-файлов, один из которых содержит сегменты линий, а второй — набранные ими очки. Результатом работы ассоциатора для пары изображений, в свою очередь, является список пар индексов, которые соответствуют линиям на первом и втором изображении и так же представляются в виде `CSV`-файла.

Итоговая схема работы адаптированного алгоритма представлена на рисунке 4. Результаты унификации запуска детекторов были оформлены в виде репозитория на GitHub[5], который включает 20 детекторов и пять ассоциаторов.

[5] https://github.com/prime-slam/line-detection-association-dockers (дата обращения: 2022-12-11)

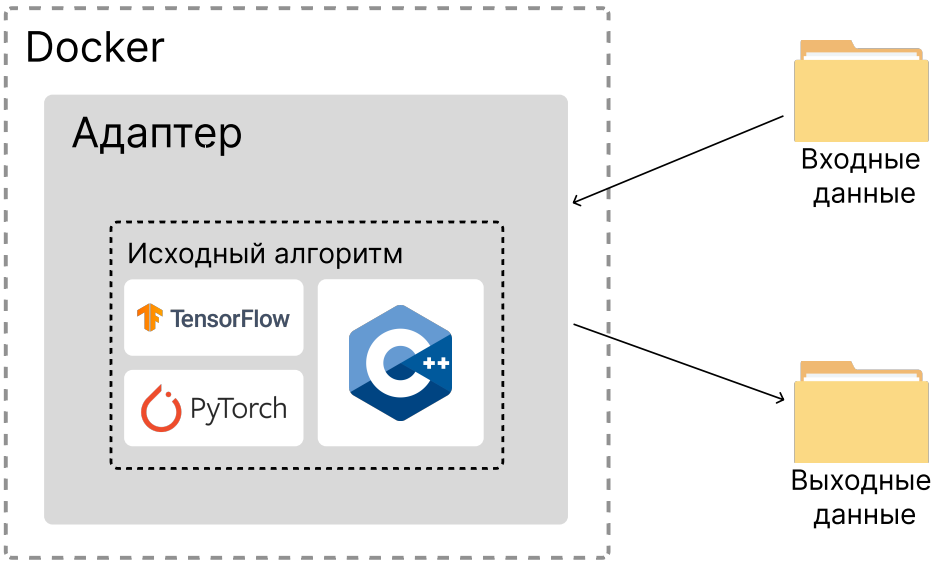


Рис. 4: Схема работы адаптированного алгоритма

5. Библиотека с метриками детекции и ассоциации линий

Поскольку существующие реализации метрик были спроектированы под конкретные алгоритмы и не распространялись отдельно от них, было принято решение реализовать их в виде библиотеки, которая будет описана в данной секции.

5.1. Использованные инструменты

Для реализации библиотеки был выбран язык Python, что объясняется удобством его использования для прототипирования SLAM-систем. Основная функциональность была реализована на основе широко используемых библиотек NumPy[6], SciPy[7] и scikit-image[8]. Также для ускорения работы некоторых метрик был применен параллелизм с использованием библиотеки Joblib[9].

Дополнительно к коду были написаны модульные тесты с применением библиотеки pytest[10] и система непрерывной интеграции, основанная на GitHub Actions и включающая запуск линтера black[11] и тестов.

[6] https://numpy.org/ (дата обращения: 2023-04-17)
[7] https://scipy.org/ (дата обращения: 2023-04-17)
[8] https://scikit-image.org/ (дата обращения: 2023-04-17)
[9] https://joblib.readthedocs.io/en/latest/ (дата обращения: 2023-04-17)
[10] https://docs.pytest.org/ (дата обращения: 2023-04-17)
[11] https://pypi.org/project/black/ (дата обращения: 2023-04-17)

5.2. Архитектура

Реализованная библиотека evolin, как показано на рисунке 5, включает в себя два основных компонента: `Metrics` и `Evaluation`. Рассмотрим их подробнее.

Компонент `Metrics` содержит реализации метрик детекции и ассоциации. Метрики детекции были разбиты на две группы: метрики, основанные на тепловой карте, (`Heatmap`) и векторные метрики (`Vectorized`). Первая группа содержит метрики классификации отдельных пикселей предсказанных линий, а именно усредненную точность (average precision), точность (precision), полноту (precision), F-меру (F-score), а также кривые «точность-полнота» (Precision-Recall curves). Вторая группа содержит аналогичные метрики классификации, а также метрики повторяемости — повторяемость (repeatability) и ошибку локализации (localization error). Метрики ассоциации, в свою очередь, включают в себя метрики классификации (точность, полнота, F-мера), а также метрики оценки относительной позиции.

Компонент `Evaluation` содержит функциональность для вычисления метрик по заданным предсказанным и истинным данным в задачах детекции и ассоциации. Общая для обеих задач функциональность, включающая геометрию, вспомогательные функции и структуры данных, вынесена в отдельный компонент `Common`.

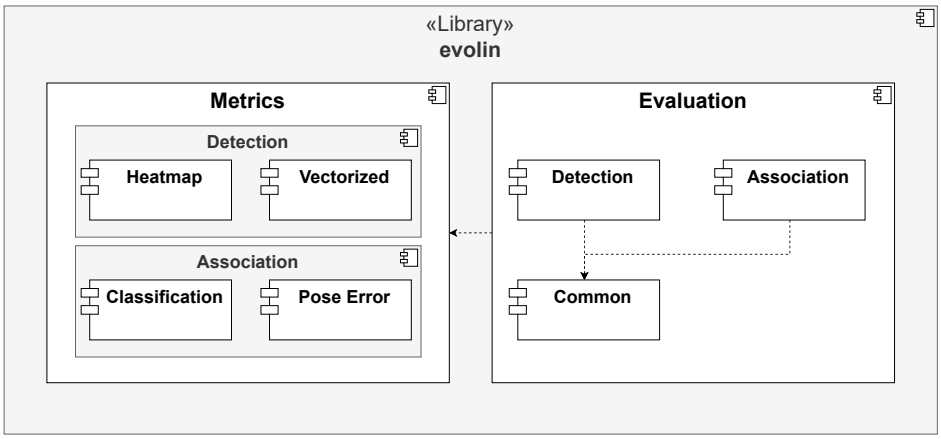


Рис. 5: Диаграмма компонентов библиотеки

5.3. Особенности реализации

Рассмотрим подробнее некоторые детали реализации библиотеки evolin.

Для вычисления тепловых метрик в задаче детекции необходимо решить задачу о назначениях (подробнее описано в подсекции 2.4.1). Для её решения были применены алгоритмы из библиотеки SciPy. Для эффективного вычисления расстояний векторных метрик была использована broadcasting-функциональность[12] библиотеки NumPy.

[12] https://numpy.org/doc/stable/user/basics.broadcasting.html (дата обращения: 2023-04-17)

Как указано в подсекции 2.4.1, для вычисления метрик повторяемости необходимо определить отображение линии с одного изображения на другое. В оригинальной статье [67] для этого использовались синтетические гомографии, но поскольку подготовленные в ходе работы данные содержат информацию о глубине, а также калибровочную матрицу, появляется возможность вычислять перепроекции линий с одного кадра на другой непосредственно. Таким образом, была реализована функциональность для перепроекции линий с использованием карт глубины.

Можно отметить, что вычисление метрик происходит для каждого кадра независимо, поэтому этот процесс был ускорен при помощи параллелизма с использованием библиотеки Joblib.

Для вычисления метрик оценки позиции необходимо сначала вычислить эту позицию. Так, консультантом была написана функциональность для вычисления относительной позиции по соответствиям линий с использованием библиотеки g2o[13].

[13] https://github.com/RainerKuemmerle/g2o (дата обращения: 2023-04-17)

6. Экспериментальное исследование

В данной секции описывается экспериментальное исследование адаптированных алгоритмов детекции и ассоциации, проведенное с использованием реализованных метрик и размеченных наборов данных.

6.1. Цель и вопросы эксперимента

Целью настоящего эксперимента является определение границ производительности и качества каждого детектора и ассоциатора. Для достижения цели были поставлены следующие исследовательские вопросы.

RQ1: Какова точность и производительность алгоритмов детекции и ассоциации на предложенных наборах данных? Данный вопрос направлен на выявление лидирующих алгоритмов по времени работы и по качеству распознавания сегментов линий.

RQ2: Какова точность оценок относительной позиции, полученных с использованием пар «детектор-ассоциатор»? Данный вопрос направлен на исследование качества оценок относительной позиции разными связками из детектора и ассоциатора и выявление наилучшей связки.

6.2. Условия эксперимента

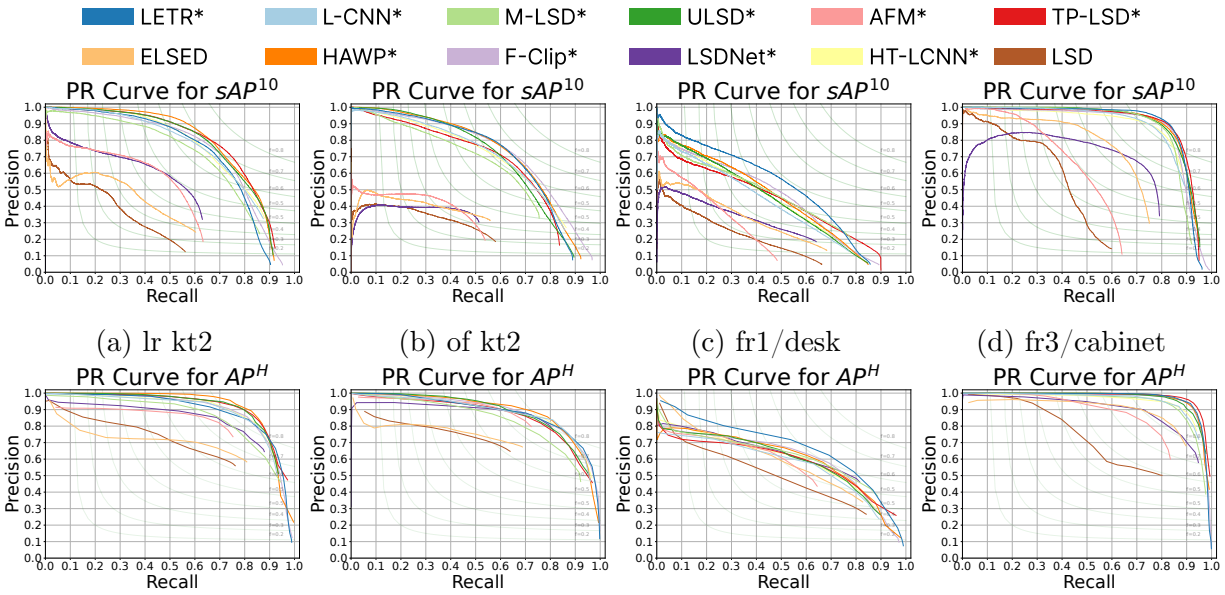
Опишем детальнее тестовый стенд, на котором проводилось экспериментальное исследование.

Характеристики системы

Для проведения эксперимента использовалась машина со следующими характеристиками: 11th Gen Intel(R) Core(TM) i5-11300H @ 3.10GHz (отключенный Turbo Boost), 24 GB RAM, GeForce RTX 3060 Mobile 6 GB, Ubuntu 22.04.

Исследуемые алгоритмы

Для проведения сравнения было использовано 17 алгоритмов детекции: AFM [37], ELSED [77], F-Clip [19], HAWP [25], HT-LCNN [45], L-CNN [95], LETR [47], LSDNet [81], LSWMS [48], M-LSD [82], TP-LSD [79], ULSD [83], FSG [15], SOLD2 [67], LSD [33], EDLines [1], и FLD [60]. Из сравнения были исключены алгоритмы KHT [18], HoughLines [13], поскольку они детектируют лишь протяженные линии, не разбивая их на отдельные сегменты. Также было преведено сравнение всех адаптированных ассоциаторов, а именно DLD [35], WLD [34], LBD [94], SOLD2 [67] и LineTR [47].



(a) lr kt2 (b) of kt2 (c) fr1/desk (d) fr3/cabinet

(e) lr kt2 (f) of kt2 (g) fr1/desk (h) fr3/cabinet

Рис. 6: Кривые Precision-Recall (PR curves) для sAP^{10} (векторная усредненная точность с порогом на структурное расстояние 10 пикселей) и AP^H (тепловая усредненная точность); * — нейросетевой алгоритм

Метрики

Среди векторных метрик для сравнения алгоритмов, предоставляющих меру неопределенности для каждой линии, была выбрана усредненная точность (AP) для расстояний d_s (1) и $d_\perp$ (2) с порогом 5 и 10 пикселей, а также были построены кривые Precision-Recall. Отсутствие меры неопределенности для линий не позволяет вычислить указанные ранее метрики, поэтому для таких алгоритмов сравнение было проведено с использованием точности, полноты и F-меры. Чтобы уменьшить влияние разрешения изображения на получаемые оценки, линии были отмасштабированы до разрешения 128×128 пикселей. Среди метрик, основанных на тепловых картах, были использованы усредненная точность и PR-кривые для оценки алгоритмов, которые измеряют неопределенность линий, и точность, полнота и F-мера — для остальных алгоритмов. Также были вычислены метрики повторяемости для ранее указанных расстояний с порогом 5 пикселей. Для сравнения ассоциаторов были использованы следующие метрики: точность, полнота и F-мера. Сравнение пар «детектор-ассоциатор», в свою очередь, было проведено с использованием метрик оценки относительной позиции: ошибки смещения и вращения.

Наборы данных

Для сравнения были использованы все кадры размеченных траекторий `lr kt2`, `of kt2` из ICL NUIM и `fr3/cabinet`, `fr1/desk` из TUM RGB-D. Метрики повторяемости и оценки относительной позиции не вычислялись на траектории `fr1/desk` из-за

неудовлетворительного качества карт глубины, что не позволяло перепроецировать линии между кадрами.

6.3. Исследование детекторов

Таблица 2: Сравнение детекторов, не предоставляющих меру неопределенности: точность (P), полнота (R) и F-мера (F) для структурного и ортогонального расстояния с порогом 5 пикселей, частота кадров (FPS); * — нейросетевой алгоритм.

Алгоритм	lr kt2							of kt2							fr1/desk							fr3/cabinet						
	d_s			$d_\perp$			FPS	d_s			$d_\perp$			FPS	d_s			$d_\perp$			FPS	d_s			$d_\perp$			FPS
	P^5	R^5	F^5	P^5	R^5	F^5		P^5	R^5	F^5	P^5	R^5	F^5		P^5	R^5	F^5	P^5	R^5	F^5		P^5	R^5	F^5	P^5	R^5	F^5	
EDLines	12.1	**56.2**	19.9	19.2	**89.0**	31.6	**387**	14.7	**50.8**	22.8	24.0	**82.5**	37.1	**325**	4.2	**50.9**	7.8	7.7	**93.2**	14.3	**213**	12.9	**68.0**	21.7	18.0	**94.5**	30.2	**679**
FLD	10.5	45.6	17.0	18.9	82.3	30.7	346	13.1	45.7	20.4	22.7	79.1	35.3	284	4.0	42.9	7.3	8.3	89.9	15.2	180	13.4	54.6	21.5	21.9	89.2	35.2	668
FSG	**28.8**	43.4	**34.6**	**42.1**	63.4	**50.6**	39	**26.7**	28.0	**27.3**	**51.5**	54.0	**52.7**	38	**11.0**	41.6	**17.4**	19.3	73.0	**30.5**	21	**38.7**	63.4	**48.0**	**50.1**	82.2	**62.3**	51
SOLD2*	8.8	24.7	13.0	23.7	66.5	35.0	3	11.5	27.4	16.2	24.9	59.3	35.1	3	6.5	9.3	7.7	**22.4**	32.3	26.5	4	26.5	42.8	32.8	40.0	64.5	49.4	7

Таблица 3: Сравнение детекторов, не предоставляющих меру неопределенности: тепловые точность (P^H), полнота (R^H), и F-мера (F^H); * — нейросетевой алгоритм.

Алгоритм	lr kt2			of kt2			fr1/desk			fr3/cabinet		
	P^H	R^H	F^H	P^H	R^H	F^H	P^H	R^H	F^H	P^H	R^H	F^H
EDLines	52.5	**86.4**	65.3	61.6	**78.2**	68.9	26.5	**91.6**	41.1	57.9	**93.5**	71.5
FLD	53.6	79.3	64.0	60.9	72.6	66.2	28.1	86.4	42.4	66.1	87.6	75.3
FSG	57.4	78.8	66.4	63.6	67.7	65.6	29.6	83.1	43.6	53.0	87.2	65.9
SOLD2*	**74.5**	66.6	**70.3**	**81.1**	67.5	**73.7**	**62.9**	36.2	**45.9**	**87.0**	66.9	**75.6**

Результаты сравнения детекторов на векторных метриках классификации представлены в таблицах 2, A.1 (Приложение A) и на рисунке 6. Среди алгоритмов, предоставляющих меру неопределенности для линий, нейросетевые детекторы (HAWP, F-Clip, LETR, TP-LSD) достигают наилучших результатов и значительно превосходят традиционные алгоритмы, такие как LSD и ELSED. Алгоритмы, не предоставляющие меру неопределенности для линий, в свою очередь, показывают довольно низкие результаты точности (precision) и F-меры на всех наборах данных, однако стоит отметить сравнительно высокие значения полноты (recall) алгоритма EDlines.

На тепловых метриках классификации (представлены в таблицах A.2 (Приложение A), 3 и на рисунке 6) в качестве лидеров снова можно выделить алгоритмы HAWP, F-Clip, LETR и TP-LSD, при этом традиционные алгоритмы по-прежнему им уступают. Также на этих метриках нейросетевой алгоритм SOLD2 лидирует по таким показателям, как точность (precision) и F-мера, среди алгоритмов без меры неопределенности, но алгоритм EDLines снова имеет наивысшую полноту (recall).

По результатам производительности, представленных в таблицах 2, A.1 (Приложение A), можно установить, что традиционные алгоритмы EDLines и FLD существенно

Таблица 4: Сравнение детекторов: повторяемость (Rep-5; больше — лучше) и ошибка локализации (Loc-5, меньше — лучше), порог расстояния — 5 пикселей, шаг между кадрами — 50; * — нейросетевой алгоритм; символ «@» означает, что следующее за ним число было использовано в качества порога на меру неопределенности для фильтрации линий.

Алгоритм	lr kt2				of kt2				fr3/cabinet			
	d_s		$d_\perp$		d_s		$d_\perp$		d_s		$d_\perp$	
	Rep-5↑	Loc-5↓	Rep-5↑	Loc-5↓	Rep-5↑	Loc-5↓	Rep-5↑	Loc-5↓	Rep-5↑	Loc-5↓	Rep-5↑	Loc-5↓
AFM@0.1*	0.21	2.48	0.44	1.79	0.19	2.51	0.40	1.66	0.08	2.49	0.24	2.51
ELSED@0	0.33	2.56	0.51	1.43	0.36	2.44	0.52	**1.28**	0.18	2.91	0.37	1.80
F-Clip@0.4*	0.34	2.24	0.53	1.57	0.34	2.06	0.47	1.41	**0.27**	2.54	0.37	1.91
HAWP@0.8	0.35	**2.16**	**0.55**	1.52	0.35	2.11	0.47	1.41	**0.27**	2.41	0.35	1.82
HT-LCNN@0.98*	0.32	2.22	0.51	1.55	0.31	**2.01**	0.45	1.36	0.23	**2.20**	0.30	1.75
L-CNN@0.98*	0.33	2.27	0.53	1.55	0.31	2.07	0.45	1.40	0.25	2.43	0.33	1.86
LETR@0.7*	0.33	2.45	0.53	1.67	0.35	2.44	0.48	1.66	0.30	2.59	0.39	1.92
LSD@0	**0.43**	2.56	0.52	**1.40**	**0.45**	2.41	0.51	1.31	0.21	2.89	0.35	1.73
LSDNet@0.8*	0.30	2.44	0.50	1.58	0.30	2.33	0.45	1.41	0.20	2.53	0.34	1.90
LSWMS@0	0.28	3.04	0.41	1.71	0.20	3.19	0.31	2.09	0.16	3.28	0.23	2.16
M-LSD@0.4*	0.33	2.42	0.50	1.57	0.35	2.25	0.47	1.51	0.24	2.59	0.32	1.89
TP-LSD@0.3*	0.32	2.30	0.52	1.53	0.36	2.21	0.49	1.43	**0.27**	2.51	0.37	1.89
ULSD@0.9*	0.34	2.25	0.54	1.54	0.32	2.09	0.46	1.42	0.26	2.48	0.35	1.85
EDLines	0.40	2.60	0.51	**1.40**	0.43	2.52	0.51	1.31	0.26	3.00	**0.44**	1.76
FLD	0.39	2.70	0.51	1.42	0.43	2.58	0.50	1.29	0.21	2.97	0.41	1.83
FSG	0.25	2.53	0.52	1.49	0.27	2.48	**0.53**	1.44	0.14	2.66	0.34	**1.72**
SOLD2*	0.38	2.59	0.47	1.51	0.42	2.35	0.47	1.33	0.15	2.39	0.28	1.78

превосходят остальные. Однако стоит отметить, что и среди нейросетевых алгоритмов имеются довольно производительные, как, например, M-LSD.

На метриках повторяемости (таблица 4) традиционные алгоритмы в среднем превосходят нейросетевые. Это, возможно, связано с тем, что более мелкие сегменты, детектируемые традиционными алгоритмами, лучше перепроецируются, чем более протяженные сегменты, которые детектируют нейросетевые алгоритмы. Соответственно, большая доля сегментов может быть корректно распознана как повторяемая.

6.4. Исследование ассоциаторов

Результаты сравнения ассоциаторов по производительности и на метриках классификации представлены в таблице 5. Примечательно, что LineTR имеет самую высокую точность среди представленных алгоритмов, но относительно низкую полноту. SOLD2, в свою очередь, превосходит остальные алгоритмы на метриках полноты и F-меры, однако уступает остальным алгоритмам по производительности. Также стоит отметить, что классический алгоритм LBD показывает сравнительно неплохие результаты на метриках классификации, имея при этом наивысшую производительность.

Таблица 5: Сравнение ассоциаторов: точность (P), полнота (R), F-мера (F) и частота кадров (FPS); * — нейросетевой алгоритм

Алгоритм	lr kt2				of kt2				fr1/desk				fr3/cabinet			
	P	R	F	FPS	P	R	F	FPS	P	R	F	FPS	P	R	F	FPS
LBD	84.8	69.6	76.4	**94**	86.0	73.7	79.4	**80**	36.6	25.2	29.8	**109**	83.3	50.9	63.2	**195**
DLD*	81.8	64.4	72.1	30	78.3	64.7	70.8	16	23.9	16.3	19.4	23	73.6	49.1	58.9	74
WLD*	82.5	66.7	73.8	4	80.3	61.8	69.8	2	35.5	22.7	27.7	3	86.7	63.8	73.5	13
LineTR*	**96.8**	56.7	71.5	14	**95.7**	61.6	75.0	12	**69.5**	6.8	12.5	14	**96.9**	41.2	57.8	19
SOLD2*	88.1	**80.8**	**84.3**	2	89.2	**82.2**	**85.5**	1	42.3	**32.4**	**36.7**	2	94.2	**84.6**	**89.1**	4

6.5. Исследование пар «детектор-ассоциатор»

В таблице 6 представлены результаты сравнения пар «детектор-ассоциатор» на метриках оценки относительной позиции. Для сравнения были выбраны все ассоциаторы и детекторы F-Clip и HAWP, показавшие лучшие результаты на метриках классификации, а также наиболее популярный детектор LSD. Так, можно заключить, что все пары с нейросетевыми ассоциаторами существенно превосходят пары с традиционным алгоритмом LBD. Также стоит отметить, что пара LSD+LineTR показывает лучшие результаты на последовательностях `lr kt2` и `of kt2`, а пара LSD+SOLD2 — на последовательности `fr3/cabinet`. Этот факт опять же может быть связан с тем, что более мелкие сегменты, которые детектирует LSD, лучше перепроецируются.

Таблица 6: Оценки ошибкок позы: медианные ошибки смещения ϵ_{trans} и вращения ϵ_{rot}; прочерк означает, что позу не удалось вычислить в более чем половине случаев; шаг между кадрами — 50; * — нейросетевой алгоритм; символ «@» означает, что следующее за ним число было использовано в качества порога на меру неопределенности для фильтрации линий.

Детектор	Ассоциатор	lr kt2		of kt2		fr3/cabinet	
		ϵ_{trans}	ϵ_{rot}	ϵ_{trans}	ϵ_{rot}	ϵ_{trans}	ϵ_{rot}
LSD@0	LBD	0.452	7.888	0.446	6.986	1.074	28.052
	DLD*	0.206	3.589	0.315	4.585	1.067	32.883
	WLD*	0.383	7.259	0.765	13.199	0.818	22.637
	LineTR*	**0.025**	**0.890**	**0.026**	**0.683**	0.620	17.384
	SOLD2*	0.144	2.902	0.099	1.917	**0.162**	**5.319**
F-Clip@0.4*	LBD	0.790	12.950	0.948	14.130	—	—
	DLD*	0.306	5.321	0.944	13.025	67.799	153.450
	WLD*	0.594	8.396	1.599	25.394	3.114	71.209
	LineTR*	0.061	1.579	0.074	1.517	—	—
	SOLD2*	0.095	2.164	0.295	3.968	0.431	12.772
HAWP@0.7*	LBD	1.993	34.954	2.050	32.943	—	—
	DLD*	1.462	23.861	1.904	27.084	—	—
	WLD*	1.533	25.016	2.425	44.257	—	—
	LineTR*	0.058	1.602	0.079	1.521	—	—
	SOLD2*	0.058	1.383	0.356	4.202	0.517	16.713

6.6. Вывод

Подведем итоги и ответим на поставленные исследовательские вопросы.

RQ1: Какова точность и производительность алгоритмов детекции и ассоциации на предложенных наборах данных? На основании проведенного эксперимента можно заключить, что нейросетевые детекторы (HAWP, F-Clip, LETR, TP-LSD) лучше традиционных распознают структурные элементы, что видно по тепловым и векторным метрикам классификации. Традиционные алгоритмы, в свою очередь, в среднем превосходят нейросетевые на метриках повторяемости. Лидерами по производительности являются классические алгоритмы EDLines и FLD, но и есть также довольно производительный нейросетевой алгоритм M-LSD. Среди ассоциаторов на метриках классификации наилучших результатов достигают нейросетевые алгоритмы LineTR и SOLD2, при этом традиционный алгоритм LBD существенно превосходит остальные по производительности.

RQ2: Какова точность оценок относительной позиции, полученных с использованием пар «детектор-ассоциатор»? Связки нейросетевых ассоциаторов с традиционными детекторами имеют наименьшую ошибку оценки относительной позиции на представленных наборах данных. Явными лидерами являются связки LSD+LineTR и LSD+SOLD2.

Заключение

При выполнении данной работы были достигнуты следующие результаты.

1. В рамках обзора было рассмотрено более 120 алгоритмов детекции и ассоциации линий, выбраны метрики для их оценки, а также на основании собранной статистики были выявлены наиболее популярные в SLAM-системах детекторы и ассоциаторы.

2. Были подготовлены наборы данных для тестирования детекторов и ассоциаторов (с привлечением обученного специалиста и инструмента CVAT), включающие аннотации линий и основанные на популярных SLAM-последовательностях.

3. Предложен формат унифицированного запуска алгоритмов детекции и ассоциации, с его помощью поддержано 20 детекторов и 5 ассоциаторов линий[14], имеющих открытую реализацию.

4. Реализована библиотека с метриками детекции и ассоциации линий (язык Python)[15]. Для библиотеки реализован набор модульных тестов, а также создана система непрерывной интеграции при помощи GitHub Actions.

5. Проведено экспериментальное исследование адаптированных алгоритмов детекции и ассоциации линий. Установлено, что комбинация нейросетевого ассоциатора LineTR и традиционного детектора LSD позволяет достичь наименьшего значения ошибки относительной позы (Relative Pose Error) среди прочих комбинаций детекторов и ассоциаторов.

6. Материалы работы вошли в статью для конференции 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems.

[14] https://github.com/prime-slam/line-detection-association-dockers (дата обращения: 2023-04-18)

[15] https://github.com/prime-slam/evolin (дата обращения: 2023-04-18)

Список литературы

[1] Akinlar Cuneyt, Topal Cihan. EDLines: A real-time line segment detector with a false detection control // Pattern Recognition Letters. — 2011. — Vol. 32, no. 13. — P. 1633–1642.

[2] Are we ready for service robots? the openloris-scene datasets for lifelong slam / Xuesong Shi, Dongjiang Li, Pengpeng Zhao et al. // 2020 IEEE international conference on robotics and automation (ICRA) / IEEE. — 2020. — P. 3139–3145.

[3] Atiquzzaman Mohammed, Akhtar Mohammed W. Complete line segment description using the Hough transform // Image and Vision computing. — 1994. — Vol. 12, no. 5. — P. 267–273.

[4] Automatic line matching and 3D reconstruction of buildings from multiple views / Caroline Baillard, Cordelia Schmid, Andrew Zisserman, Andrew Fitzgibbon // ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery. — Vol. 32. — 1999. — P. 69–80.

[5] Bay Herbert, Ferraris Vittorio, Van Gool Luc. Wide-baseline stereo matching with line segments // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) / IEEE. — Vol. 1. — 2005. — P. 329–336.

[6] Boldt Michael, Weiss Richard, Riseman Edward. Token-based extraction of straight lines // IEEE Transactions on Systems, Man, and Cybernetics. — 1989. — Vol. 19, no. 6. — P. 1581–1594.

[7] Bonci Andrea, Leo Tommaso, Longhi Sauro. A Bayesian approach to the Hough transform for line detection // IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans. — 2005. — Vol. 35, no. 6. — P. 945–955.

[8] Building a 3-D line-based map using stereo SLAM / Guoxuan Zhang, Jin Han Lee, Jongwoo Lim, Il Hong Suh // IEEE Transactions on Robotics. — 2015. — Vol. 31, no. 6. — P. 1364–1377.

[9] Burns J Brian, Hanson Allen R, Riseman Edward M. Extracting straight lines // IEEE transactions on pattern analysis and machine intelligence. — 1986. — no. 4. — P. 425–455.

[10] Cho Nam-Gyu, Yuille Alan, Lee Seong-Whan. A novel linelet-based representation for line segment detection // IEEE transactions on pattern analysis and machine intelligence. — 2017. — Vol. 40, no. 5. — P. 1195–1208.

[11] Deep residual learning for image recognition / Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — P. 770–778.

[12] Denis Patrick, Elder James H, Estrada Francisco J. Efficient edge-based methods for estimating manhattan frames in urban imagery // European conference on computer vision / Springer. — 2008. — P. 197–210.

[13] Duda Richard O, Hart Peter E. Use of the Hough transformation to detect lines and curves in pictures // Communications of the ACM. — 1972. — Vol. 15, no. 1. — P. 11–15.

[14] The EuRoC micro aerial vehicle datasets / Michael Burri, Janosch Nikolic, Pascal Gohl et al. // The International Journal of Robotics Research. — 2016. — Vol. 35, no. 10. — P. 1157–1163.

[15] FSG: A statistical approach to line detection via fast segments grouping / Iago Suárez, Enrique Muñoz, José M Buenaposada, Luis Baumela // 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) / IEEE. — 2018. — P. 97–102.

[16] Fan Bin, Wu Fuchao, Hu Zhanyi. Line matching leveraged by point correspondences // 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition / IEEE. — 2010. — P. 390–397.

[17] Fan Bin, Wu Fuchao, Hu Zhanyi. Robust line matching through line–point invariants // Pattern Recognition. — 2012. — Vol. 45, no. 2. — P. 794–805.

[18] Fernandes Leandro AF, Oliveira Manuel M. Real-time line detection through an improved Hough transform voting scheme // Pattern recognition. — 2008. — Vol. 41, no. 1. — P. 299–314.

[19] Fully convolutional line parsing / Xili Dai, Haigang Gong, Shuai Wu et al. // Neurocomputing. — 2022. — Vol. 506. — P. 1–11.

[20] Furukawa Yasutaka, Shinagawa Yoshihisa. Accurate and robust line segment extraction by analyzing distribution around peaks in Hough space // Computer Vision and Image Understanding. — 2003. — Vol. 92, no. 1. — P. 1–25.

[21] Geiger Andreas, Lenz Philip, Urtasun Raquel. Are we ready for autonomous driving? the kitti vision benchmark suite // 2012 IEEE conference on computer vision and pattern recognition / IEEE. — 2012. — P. 3354–3361.

[22] Hierarchical line matching based on line–junction–line structure descriptor and local homography estimation / Kai Li, Jian Yao, Xiaohu Lu et al. // Neurocomputing. — 2016. — Vol. 184. — P. 207–220.

[23] Hierarchical line segment matching for wide-baseline images via exploiting viewpoint robust local structure and geometric constraints / Min Chen, Shaohua Yan, Rongjun Qin et al. // ISPRS Journal of Photogrammetry and Remote Sensing. — 2021. — Vol. 181. — P. 48–66.

[24] Hirose Keisuke, Saito Hideo. Fast Line Description for Line-based SLAM. // BMVC. — 2012. — P. 1–11.

[25] Holistically-attracted wireframe parsing / Nan Xue, Tianfu Wu, Song Bai et al. // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2020. — P. 2788–2797.

[26] Illingworth John, Kittler Josef. A survey of the Hough transform // Computer vision, graphics, and image processing. — 1988. — Vol. 44, no. 1. — P. 87–116.

[27] Improved point-line feature based visual SLAM method for indoor scenes / Runzhi Wang, Kaichang Di, Wenhui Wan, Yongkang Wang // Sensors. — 2018. — Vol. 18, no. 10. — P. 3559.

[28] Jeong Woo Yeon, Lee Kyoung Mu. Visual SLAM with line and corner features // 2006 IEEE/RSJ international conference on intelligent robots and systems / IEEE. — 2006. — P. 2570–2575.

[29] Kahn Philip, Kitchen L, Riseman Edward M. A fast line finder for vision-guided robot navigation // IEEE Transactions on Pattern Analysis and Machine Intelligence. — 1990. — Vol. 12, no. 11. — P. 1098–1102.

[30] Kamat Varsha, Ganesan Subramaniam. A robust Hough transform technique for description of multiple line segments in an image // Proceedings 1998 International Conference on Image Processing. ICIP98 (Cat. No. 98CB36269) / IEEE. — Vol. 1. — 1998. — P. 216–220.

[31] Kim Hyunwoo, Lee Sukhan. Simultaneous line matching and epipolar geometry estimation based on the intersection context of coplanar line pairs // Pattern Recognition Letters. — 2012. — Vol. 33, no. 10. — P. 1349–1363.

[32] LGNN: A Context-aware Line Segment Detector / Quan Meng, Jiakai Zhang, Qiang Hu et al. // Proceedings of the 28th ACM International Conference on Multimedia. — 2020. — P. 4364–4372.

[33] LSD: A fast line segment detector with a false detection control / Rafael Grompone Von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, Gregory Randall // IEEE transactions on pattern analysis and machine intelligence. — 2008. — Vol. 32, no. 4. — P. 722–732.

[34] Lange Manuel, Raisch Claudio, Schilling Andreas. Wld: A wavelet and learning based line descriptor for line feature matching. — 2020.

[35] Lange Manuel, Schweinfurth Fabian, Schilling Andreas. Dld: A deep learning based line descriptor for line feature matching // 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) / IEEE. — 2019. — P. 5910–5915.

[36] Lap-slam: A line-assisted point-based monocular vslam / Fukai Zhang, Ting Rui, Chengsong Yang, Jianjun Shi // Electronics. — 2019. — Vol. 8, no. 2. — P. 243.

[37] Learning attraction field representation for robust line segment detection / Nan Xue, Song Bai, Fudong Wang et al. // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2019. — P. 1595–1603.

[38] Learning to find good correspondences / Kwang Moo Yi, Eduard Trulls, Yuki Ono et al. // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2018. — P. 2666–2674.

[39] Learning to parse wireframes in images of man-made environments / Kun Huang, Yifan Wang, Zihan Zhou et al. // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. — 2018. — P. 626–635.

[40] Lee Junesuk, Park Soon-Yong. PLF-VINS: Real-time monocular visual-inertial SLAM with point-line fusion and parallel-line fusion // IEEE Robotics and Automation Letters. — 2021. — Vol. 6, no. 4. — P. 7033–7040.

[41] Lee Sang Jun, Hwang Sung Soo. Elaborate monocular point and line slam with robust initialization // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2019. — P. 1121–1129.

[42] Lee Tae-jae, Kim Chul-hong, Cho Dong-il Dan. A monocular vision sensor-based efficient SLAM method for indoor service robots // IEEE Transactions on Industrial Electronics. — 2018. — Vol. 66, no. 1. — P. 318–328.

[43] Lemaire Thomas, Lacroix Simon. Monocular-vision based SLAM using line segments // Proceedings 2007 IEEE International Conference on Robotics and Automation / IEEE. — 2007. — P. 2791–2796.

[44] Li Kai, Yao Jian, Lu Xiaohu. Robust line matching based on ray-point-ray structure descriptor // Asian Conference on Computer Vision / Springer. — 2014. — P. 554–569.

[45] Lin Yancong, Pintea Silvia L, Gemert Jan C van. Deep hough-transform line priors // European Conference on Computer Vision / Springer. — 2020. — P. 323–340.

[46] Line matching based on line-points invariant and local homography / Qi Jia, Xin Fan, Xinkai Gao et al. // Pattern Recognition. — 2018. — Vol. 81. — P. 471–483.

[47] Line segment detection using transformers without edges / Yifan Xu, Weijian Xu, David Cheung, Zhuowen Tu // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2021. — P. 4257–4266.

[48] Line segment detection using weighted mean shift procedures on a 2D slice sampling strategy / Marcos Nieto, Carlos Cuevas, Luis Salgado, Narciso García // Pattern Analysis and Applications. — 2011. — Vol. 14, no. 2. — P. 149–163.

[49] Line segment matching: A benchmark / Kai Li, Jian Yao, Mengsheng Lu et al. // 2016 IEEE Winter Conference on Applications of Computer Vision (WACV) / IEEE. — 2016. — P. 1–9.

[50] Ma QuanMeng, Jiang Guang, Lai DianZhi. Robust line segments matching via graph convolution networks // arXiv preprint arXiv:2004.04993. — 2020.

[51] Martin David R, Fowlkes Charless C, Malik Jitendra. Learning to detect natural image boundaries using local brightness, color, and texture cues // IEEE transactions on pattern analysis and machine intelligence. — 2004. — Vol. 26, no. 5. — P. 530–549.

[52] Matas Jiri, Galambos Charles, Kittler Josef. Robust detection of lines using the progressive probabilistic hough transform // Computer vision and image understanding. — 2000. — Vol. 78, no. 1. — P. 119–137.

[53] McIntosh James H, Mutch Kathleen M. Matching straight lines // Computer Vision, Graphics, and Image Processing. — 1988. — Vol. 43, no. 3. — P. 386–408.

[54] Mcmlsd: A dynamic programming approach to line segment detection / Emilio J Almazan, Ron Tal, Yiming Qian, James H Elder // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. — 2017. — P. 2031–2039.

[55] Mean shift based clustering of Hough domain for fast line segment detection / Antonio Bandera, José Manuel Pérez-Lorenzo, Juan Pedro Bandera, F Sandoval // Pattern Recognition Letters. — 2006. — Vol. 27, no. 6. — P. 578–586.

[56] Medioni Gérard, Nevatia Ramakant. Segment-based stereo matching // Computer vision, graphics, and image processing. — 1985. — Vol. 31, no. 1. — P. 2–18.

[57] Mur-Artal Raul, Tardós Juan D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras // IEEE transactions on robotics. — 2017. — Vol. 33, no. 5. — P. 1255–1262.

[58] Novel coplanar line-points invariants for robust line matching across views / Qi Jia, Xinkai Gao, Xin Fan et al. // European Conference on Computer Vision / Springer. — 2016. — P. 599–611.

[59] ORB: An efficient alternative to SIFT or SURF / Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski // 2011 International conference on computer vision / Ieee. — 2011. — P. 2564–2571.

[60] Outdoor place recognition in urban environments using straight lines / Jin Han Lee, Sehyung Lee, Guoxuan Zhang et al. // 2014 IEEE International Conference on Robotics and Automation (ICRA) / IEEE. — 2014. — P. 5550–5557.

[61] PL-SLAM: A stereo SLAM system through the combination of points and line segments / Ruben Gomez-Ojeda, Francisco-Angel Moreno, David Zuniga-Noël et al. // IEEE Transactions on Robotics. — 2019. — Vol. 35, no. 3. — P. 734–746.

[62] PL-SLAM: Real-time monocular visual SLAM with points and lines / Albert Pumarola, Alexander Vakhitov, Antonio Agudo et al. // 2017 IEEE international conference on robotics and automation (ICRA) / IEEE. — 2017. — P. 4503–4508.

[63] Pl-vins: Real-time monocular visual-inertial slam with point and line features / Qiang Fu, Jialong Wang, Hongshan Yu et al. // arXiv preprint arXiv:2009.07462. — 2020.

[64] Ppgnet: Learning point-pair graph for line segment detection / Ziheng Zhang, Zhengxin Li, Ning Bi et al. // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2019. — P. 7105–7114.

[65] RGB-D SLAM with structural regularities / Yanyan Li, Raza Yunus, Nikolas Brasch et al. // 2021 IEEE International Conference on Robotics and Automation (ICRA) / IEEE. — 2021. — P. 11581–11587.

[66] Robust visual SLAM with point and line features / Xingxing Zuo, Xiaojia Xie, Yong Liu, Guoquan Huang // 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) / IEEE. — 2017. — P. 1775–1782.

[67] SOLD2: Self-supervised occlusion-aware line description and detection / Rémi Pautrat, Juan-Ting Lin, Viktor Larsson et al. // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2021. — P. 11368–11378.

[68] Schmid Cordelia, Zisserman Andrew. Automatic line matching across views // Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition / IEEE. — 1997. — P. 666–671.

[69] Schmid Cordelia, Zisserman Andrew. The geometry and matching of lines and curves over multiple views // International Journal of Computer Vision. — 2000. — Vol. 40, no. 3. — P. 199–233.

[70] Smith Paul, Reid Ian, Davison Andrew J. Real-Time Monocular SLAM with Straight Lines. // BMVC. — Vol. 6. — 2006. — P. 17–26.

[71] Song Jiqiang, Lyu Michael R. A Hough transform based line recognition method utilizing both parameter space and image space // Pattern recognition. — 2005. — Vol. 38, no. 4. — P. 539–552.

[72] Stephens Richard S. Probabilistic approach to the Hough transform // Image and vision computing. — 1991. — Vol. 9, no. 1. — P. 66–71.

[73] Stereo visual-inertial SLAM with points and lines / Yanqing Liu, Dongdong Yang, Jiamao Li et al. // IEEE Access. — 2018. — Vol. 6. — P. 69381–69392.

[74] StructSLAM: Visual SLAM with building structure lines / Huizhong Zhou, Danping Zou, Ling Pei et al. // IEEE Transactions on Vehicular Technology. — 2015. — Vol. 64, no. 4. — P. 1364–1375.

[75] Structure PLP-SLAM: Efficient Sparse Mapping and Localization using Point, Line and Plane for Monocular, RGB-D and Stereo Cameras / Fangwen Shu, Jiaxuan Wang, Alain Pagani, Didier Stricker // arXiv preprint arXiv:2207.06058. — 2022.

[76] Structure-slam: Low-drift monocular slam in indoor environments / Yanyan Li, Nikolas Brasch, Yida Wang et al. // IEEE Robotics and Automation Letters. — 2020. — Vol. 5, no. 4. — P. 6583–6590.

[77] Suárez Iago, Buenaposada José M, Baumela Luis. Elsed: Enhanced line segment drawing // arXiv preprint arXiv:2108.03144. — 2021.

[78] Superglue: Learning feature matching with graph neural networks / Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, Andrew Rabinovich // Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. — 2020. — P. 4938–4947.

[79] TP-LSD: Tri-points based line segment detector / Siyu Huang, Fangbo Qin, Pengfei Xiong et al. // European Conference on Computer Vision / Springer. — 2020. — P. 770–785.

[80] Tartanair: A dataset to push the limits of visual slam / Wenshan Wang, Delong Zhu, Xiangwei Wang et al. // 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) / IEEE. — 2020. — P. 4909–4916.

[81] Teplyakov Lev, Erlygin Leonid, Shvets Evgeny. LSDNet: Trainable Modification of LSD Algorithm for Real-Time Line Segment Detection // IEEE Access. — 2022. — Vol. 10. — P. 45256–45265.

[82] Towards real-time and light-weight line segment detection / Geonmo Gu, Byungsoo Ko, SeoungHyun Go et al. // arXiv preprint arXiv:2106.00186. — 2021.

[83] ULSD: unified line segment detection across pinhole, fisheye, and spherical cameras / Hao Li, Huai Yu, Jinwang Wang et al. // ISPRS Journal of Photogrammetry and Remote Sensing. — 2021. — Vol. 178. — P. 187–202.

[84] Vakhitov Alexander, Lempitsky Victor. Learnable line segment descriptor for visual slam // IEEE Access. — 2019. — Vol. 7. — P. 39923–39934.

[85] Verhagen Bart, Timofte Radu, Van Gool Luc. Scale-invariant line descriptors for wide baseline matching // IEEE Winter Conference on Applications of Computer Vision / IEEE. — 2014. — P. 493–500.

[86] Visual SLAM with BoPLW pairs using egocentric stereo camera for wearable-assisted substation inspection / Kun Qian, Wei Zhao, Kai Li et al. // IEEE Sensors Journal. — 2019. — Vol. 20, no. 3. — P. 1630–1641.

[87] Wang Lu, Neumann Ulrich, You Suya. Wide-baseline image matching using line signatures // 2009 IEEE 12th International Conference on Computer Vision / IEEE. — 2009. — P. 1311–1318.

[88] Wang Zhiheng, Wu Fuchao, Hu Zhanyi. MSLD: A robust descriptor for line matching // Pattern Recognition. — 2009. — Vol. 42, no. 5. — P. 941–953.

[89] Wei Dong, Zhang Yongjun, Li Chang. Robust line segment matching via reweighted random walks on the homography graph // Pattern Recognition. — 2021. — Vol. 111. — P. 107693.

[90] Xu Zezhong, Shin Bok-Suk, Klette Reinhard. Accurate and robust line segment extraction using minimum entropy with Hough transform // IEEE Transactions on Image Processing. — 2014. — Vol. 24, no. 3. — P. 813–822.

[91] Xu Zezhong, Shin Bok-Suk, Klette Reinhard. Closed form line-segment extraction using the Hough transform // Pattern Recognition. — 2015. — Vol. 48, no. 12. — P. 4012–4023.

[92] Yoon Sungho, Kim Ayoung. Line as a Visual Sentence: Context-Aware Line Descriptor for Visual Localization // IEEE Robotics and Automation Letters. — 2021. — Vol. 6, no. 4. — P. 8726–8733.

[93] Yunus Raza, Li Yanyan, Tombari Federico. Manhattanslam: Robust planar tracking and mapping leveraging mixture of manhattan frames // 2021 IEEE International Conference on Robotics and Automation (ICRA) / IEEE. — 2021. — P. 6687–6693.

[94] Zhang Lilian, Koch Reinhard. An efficient and robust line segment matching approach based on LBD descriptor and pairwise geometric consistency // Journal of Visual Communication and Image Representation. — 2013. — Vol. 24, no. 7. — P. 794–805.

[95] Zhou Yichao, Qi Haozhi, Ma Yi. End-to-end wireframe parsing // Proceedings of the IEEE/CVF International Conference on Computer Vision. — 2019. — P. 962–971.

[96] A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM / Ankur Handa, Thomas Whelan, John McDonald, Andrew J Davison // 2014 IEEE international conference on Robotics and automation (ICRA) / IEEE. — 2014. — P. 1524–1531.

[97] A benchmark for the evaluation of RGB-D SLAM systems / Jürgen Sturm, Nikolas Engelhard, Felix Endres et al. // 2012 IEEE/RSJ international conference on intelligent robots and systems / IEEE. — 2012. — P. 573–580.

[98] A review of Hough transform and line segment detection approaches / Payam Rahmdel, Richard A Comley, Daming Shi, Siobhan McElduff. — 2015.

[99] A robust RGB-D SLAM system with points and lines for low texture indoor environments / Qiang Fu, Hongshan Yu, Lihai Lai et al. // IEEE Sensors Journal. — 2019. — Vol. 19, no. 21. — P. 9908–9920.

Приложение А

Таблица А.1: Сравнение детекторов, предоставляющих меру неопределенности: усредненная точность для структурного (sAP) и ортогонального (oAP) расстояния с порогами 5 и 10 пикселей, частота кадров (FPS); * — нейросетевой алгоритм.

Алгоритм	lr kt2					of kt2					fr1/desk					fr3/cabinet				
	sAP^5	sAP^{10}	oAP^5	oAP^{10}	FPS	sAP^5	sAP^{10}	oAP^5	oAP^{10}	FPS	sAP^5	sAP^{10}	oAP^5	oAP^{10}	FPS	sAP^5	sAP^{10}	oAP^5	oAP^{10}	FPS
AFM*	32.2	41.4	54.6	56.8	12	16.0	23.3	42.9	46.2	12	10.4	19.3	28.0	34.3	11	39.0	49.3	66.0	70.7	10
ELSED	23.2	29.3	47.4	48.6	107	14.7	22.9	40.7	42.0	98	16.0	24.4	38.1	41.1	69	56.0	63.8	76.9	77.8	**162**
F-Clip*	68.6	74.0	79.5	80.0	30	**67.5**	**72.4**	**75.5**	**77.2**	29	36.3	42.7	46.7	50.5	29	87.1	87.5	90.5	90.9	29
HAWP*	**73.2**	**77.8**	**82.0**	**82.6**	25	66.3	70.9	73.0	75.0	23	36.1	42.1	45.2	49.1	23	87.7	88.8	89.5	90.2	26
HT-LCNN*	68.2	73.3	76.8	77.2	7	61.6	65.9	68.4	70.3	6	33.1	39.2	41.6	45.9	5	82.8	83.2	84.2	84.6	8
L-CNN*	67.0	72.2	76.0	76.5	17	61.2	66.0	67.7	69.8	11	31.7	37.5	40.0	43.7	8	83.7	84.2	85.3	86.0	25
LETR*	64.8	72.2	78.9	79.6	8	60.5	69.5	72.3	74.4	8	**38.3**	**49.1**	**54.0**	**58.9**	8	87.0	88.8	91.1	**91.7**	8
LSD	17.3	23.1	44.8	46.3	**111**	12.8	19.9	41.6	43.7	**112**	8.3	16.7	31.5	35.6	61	30.9	39.8	60.0	61.6	139
LSDNet*	33.3	42.5	54.4	55.7	13	13.0	19.5	36.2	39.7	13	12.7	22.7	31.8	37.1	13	54.0	60.3	74.4	75.9	13
LSWMS	1.5	3.9	10.7	14.6	20	2.9	8.2	20.5	27.7	25	1.5	5.2	11.1	17.3	20	1.8	3.3	10.3	14.2	19
M-LSD*	60.3	67.8	74.4	75.3	98	51.9	58.3	65.5	68.5	103	33.9	40.8	44.7	48.8	**99**	84.8	86.1	89.4	89.9	111
TP-LSD*	70.1	77.5	81.6	82.2	17	58.1	64.5	68.2	70.4	17	35.3	41.7	44.8	48.5	17	**88.1**	**89.0**	**91.2**	**91.7**	17
ULSD*	71.4	76.4	79.6	80.1	27	62.5	67.3	70.9	72.3	26	34.1	40.4	43.4	47.3	25	87.6	88.9	89.4	90.3	28

Таблица А.2: Сравнение детекторов, предоставляющих меру неопределенности: тепловые усредненная точность (AP^H) и F-мера (F^H); * — нейросетевой алгоритм.

Алгоритм	lr kt2		of kt2		fr1/desk		fr3/cabinet	
	AP^H	F^H	AP^H	F^H	AP^H	F^H	AP^H	F^H
AFM*	62.7	75.6	65.1	75.8	64.3	78.0	41.1	55.7
ELSED	57.1	68.4	52.8	68.0	80.5	82.5	52.2	56.1
F-Clip*	86.5	83.0	87.8	81.0	94.8	92.5	59.4	63.0
HAWP*	**91.7**	**85.7**	89.1	**82.1**	96.5	92.0	59.6	61.6
HT-LCNN*	87.5	82.4	87.2	81.5	91.5	89.3	55.8	60.2
L-CNN*	87.1	82.2	87.1	**82.1**	92.5	89.1	54.3	59.2
LETR*	89.6	82.8	**89.3**	81.7	96.8	93.0	**65.7**	**65.6**
LSD*	58.0	65.1	45.3	64.3	61.0	61.7	44.3	49.3
LSDNet*	77.6	77.8	74.2	80.1	87.3	82.0	54.9	61.0
LSWMS	32.7	57.3	37.6	58.0	39.2	62.8	24.1	46.7
M-LSD*	82.7	77.8	79.8	76.3	92.9	88.6	57.4	62.1
TP-LSD*	90.2	85.1	85.1	79.3	**97.4**	**93.5**	57.2	60.9
ULSD*	88.7	84.7	86.2	79.4	95.3	91.0	56.6	60.1