**«Санкт-Петербургский государственный электротехнический университет**

**«ЛЭТИ» им. В.И.Ульянова (Ленина)»**

**(СПбГЭТУ «ЛЭТИ»)**

|  |  |  |
| --- | --- | --- |
| **Направление** | 01.03.02- Прикладная математика и информатика | |
| **Профиль** | Разработка программно-информационных  систем | |
| **Факультет** | КТИ | |
| **Кафедра** | МОЭВМ | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | А.А. Лисс |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

БАКАЛАВРА

Тема: Комплексирование данных визуальной одометрии и навигационной системы для беспилотных устройств

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент |  |  |  | М.С. Азаров |
|  |  | *подпись* |  |  |
| Руководитель | к. т. н., доцент |  |  | Д.А. Черниченко |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |

Санкт-Петербург

2024

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой МОЭВМ |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ А.А. Лисс |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | Азаров М.С | | | |  | Группа | 0382 | |
| Тема работы: Комплексирование данных визуальной одометрии и навигационной системы для беспилотных устройств | | | | | | | | |
| Место выполнения ВКР: СПбГЭТУ «ЛЭТИ» кафедра МО ЭВМ | | | | | | | | |
| Исходные данные (технические требования):  Исследовать требования к датчикам при использовании алгоритмов слияния данных визуальной одометрии и навигационной системы | | | | | | | | |
| Содержание ВКР:  Введение, Обзор аналогов, Подготовительная часть, Практическая часть, Обеспечение качества разработки, Заключение. | | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, иллюстративный материал | | | | | | | | |
| Дополнительные разделы: Обеспечение качества разработки, продукции, программного продукта | | | | | | | | |
|  | | | | | | | | |
| Дата выдачи задания | | | Дата представления ВКР к защите | | | | | |
| «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | | | |
|  | | |  | | | | | |
| Студент | |  | | М.С. Азаров | | | |  |
| Руководитель к. т. н., доцент | |  | | Д.А. Черниченко | | | |  |
| *(Уч. степень, уч. звание)* | |  | |  | | | |  |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |  |
| --- | --- | --- |
|  | Утверждаю | Утверждаю |
|  | Зав. кафедрой МОЭВМ | Зав. кафедрой аббревиатура названия кафедры |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ А.А. Лисс | \_\_\_\_\_\_\_\_\_\_\_\_ И.И. Иванов |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | М.С. Азаров |  | Группа | 0382 |
| Тема работы: Комплексирование данных визуальной одометрии и навигационной системы для беспилотных устройств | | | | |

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование работ | Срок выполнения |
| 1 | Обзор литературы по теме работы | 00.00 – 00.00 |
| 2 | Наименование раздела | 00.00 – 00.00 |
| 3 | Наименование раздела | 00.00 – 00.00 |
| 4 | Наименование раздела | 00.00 – 00.00 |
| 5 | Оформление пояснительной записки | 00.00 – 00.00 |
| 6 | Оформление иллюстративного материала | 00.00 – 00.00 |

|  |  |  |
| --- | --- | --- |
| Студент |  | М.С. Азаров |
| Руководитель к. т. н., доцент |  | Д.А. Черниченко |
| *(Уч. степень, уч. звание)* |  |  |

**РЕФЕРАТ**

Пояснительная записка 65 стр., 19 рис., 16 табл., 24 ист.

viSLAM, Гауссовский шум, БПЛА, локализация, IMU, визуальная одометрия, стерео-камеры.

**Объектом исследования** является алгоритмы оценки положения БПЛА на основе датчиков IMU и камер (далее viSLAM)

**Предметом исследования** является влияние качества датчиков и получаемых данных на работу алгоритмов viSLAM

**Цель работы:** Используя различные методы моделирования определить зависимость точности работы алгоритма viSLAM от качества используемых датчиков и получаемых из них данных.

В данной работе рассматривается вопрос о влиянии характеристик датчиков на точность оценки положения БПЛА, полученную с помощью алгоритмы SOTA(state of the art). Для изучения этого вопроса было принято рассматривать алгоритмы объединяющие дынные с камер и IMU(Inertial Measurement Units). Опираясь на это, был выбран алгоритм VINS-Fusion, из-за его доступности и высокой точности среди других алгоритмов SOTA. В основу данных, на которых в дальнейшем исследовался алгоритм, был выбран набор данных EuRoC. Для камер было предложено исследовать зависимость точности алгоритма от разрешения и Гауссовского шума. Для IMU –от кол-ва белого шума и смещения. Дописать полученные результаты.

**ABSTRACT**

Перевод аннотации на английский язык.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 9](#_Toc166440480)

[Актуальность 9](#_Toc166440481)

[Локализация 9](#_Toc166440482)

[SLAM 10](#_Toc166440483)

[Задачи 12](#_Toc166440484)

[Структура 13](#_Toc166440485)

[1. ОБЗОР АНАЛОГОВ 15](#_Toc166440486)

[1.1 Исследование влияния шума камеры на работу ORB-SLAM2 15](#_Toc166440487)

[1.2 Исследование влияния шума камеры глубины на модифицированный RBPF-SLAM 16](#_Toc166440488)

[1.3 Исследование влияния сжатия изображения на точность ORB-SLAM3 16](#_Toc166440489)

[1.4 Исследование влияния изменения разрешения и частоты потокового видео на SVO Pro 17](#_Toc166440490)

[1.5 Вывод 18](#_Toc166440491)

[2. ПОДГОТОВИТЕЛЬНАЯ ЧАСТЬ 19](#_Toc166440492)

[2.1 Выбор алгоритма SLAM 19](#_Toc166440493)

[2.2 Выбор набора данных 21](#_Toc166440494)

[2.2 Выбор исследуемых параметров датчиков/TODO 23](#_Toc166440495)

[3. ПРАКТИЧЕСКАЯ ЧАСТЬ 24](#_Toc166440496)

[3.1 Исследование влияния параметров камеры 24](#_Toc166440497)

[3.2 Исследование влияния параметров IMU 38](#_Toc166440498)

[4. ОБЕСПЕЧЕНИЕ КАЧЕСТВА РАЗРАБОТКИ 39](#_Toc166440499)

[ЗАКЛЮЧЕНИЕ 40](#_Toc166440500)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 41](#_Toc166440501)

[ПРИЛОЖЕНИЕ А 44](#_Toc166440502)

**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

**БПЛА** – беспилотный летательный аппарат.

**IMU** (inertial measurement units) – измерительный блок, который обычно содержит акселерометры, гироскопы (иногда еще магнитометр и барометр) и используется для определения ускорения тела и угловой скорости тела в пространстве. В данной работе подразумевается использование 6-осевого IMU (3-осевой гироскоп + 3-осевой акселерометр)

**МЭМС** - микроэлектромеханические системы , устройства, объединяющие в себе взаимосвязанные механические и электрические компоненты микронных размеров. В данной работе под МЭМС понимают IMU МЭМС.

**SLAM** (simultaneous localization and mapping) – метод построения карты в неизвестном пространстве с одновременным контролем текущего местоположения, ориентации и пройденного пути;

**viSLAM** (visual-inertial SLAM) – визуально-инерциальный SLAM. SLAM который использует данные с камер и IMU.

**Одометрия** - использование данных с датчиков движения для оценки изменения положения с течением времени

**Визуальная одометрия** (**VO** - visual odometry)- процесс определения положения и ориентации робота путем анализа соответствующих изображений с камер.

**VIO** -

**Особые точки** – точки на изображении с уникальными свойствами , благодаря которым их можно детектировать на других изображениях и с других ракурсов.

**Loop Closure** – дословно «закрытие петли», элемент алгоритма SLAM, который уточняет локализацию устройства, используя генерируемую карту местности и повторно встречаемые *особые точки*.

**Местоположение (положение)** – определяется координатами XYZ.

**Ориентация** – определяется тремя углами или кватернионом.

**Поза объекта** – местоположение и ориентация объекта.

**Локализация** – определение *местоположения* и *ориентации* (позы) устройства.

**ATE** (Absolute Trajectory Error)- абсолютная ошибка траектории, метрика определяющая точность работы SLAM алгоритма. Тоже самое что и **APE**(Absolute Pose Error).

**RSME** –

ROS

Топик

Нода

## ВВЕДЕНИЕ

**Актуальность**

Беспилотные летательные аппараты (БПЛА) активно развиваются в последнее время. По итогам 2022 года объем российского рынка беспилотных авиационных систем и услуг с их применением составил около 50 млрд. рублей. А рост с 2018 по 2022 год рынка в среднегодовом выражении увеличился на 27 % [1]. БПЛА имеют широкое применение в различных задачах, таких как: геодезия и картографирование, спасательные операции при стихийных бедствиях, сбор пространственной информации, инспекция зданий, мониторинг дорожного движения и много других задач [2].

**Локализация**

Для качественного выполнения таких задач в автономном режиме, БПЛА необходимо решать проблему навигации. Данная проблема заключается в перемещении из точки A в точку B.

Для решения, данную задачу разбивают на составляющие:

1. Локализация - определение текущего местоположения устройства и его ориентацию;

2. Построение маршрута из A в B;

3. Определение действий чтобы следовать этому маршруту.

Данная работа фокусируется на проблеме *локализации*. Для решения этой задачи используются GPS-трекеры и различные датчики движения: IMU , камеры , event-камеры, лидары , энкодеры, ультразвуковой дальномер и т.д.

Для локализации с помощью датчиков которые не сообщают напрямую позу объекта, используются различны алгоритмы одометрии , такие как визуальная одометрия (использует камеры) , Лидарная одометрия и т.д. .

Локализация делиться на ***относительную*** и ***глобальную***. Глобальная локализация определяет позу объекта без каких-то либо заранее известных данных об положении и ориентации объекта. Относительная же определяет позу объекта, относительно его известного предыдущего положения и ориентации. Применяется, когда нет необходимости в глобальной локализации, либо в условиях, когда ее невозможно применить. Примерами относительной локализации является: визуальная одометрия , лидарная одометрия, локализация на основе IMU и др.

Примером глобальной *частичной* локализации является локализация на основе GPS, являетсячастичной так как она определяет только положение устройства (долгота, широта, высота), но не его ориентацию. Зачастую для компенсации этого используют датчики IMU, для выяснения ориентации объекта.

GPS широко применим в задачах локализации БПЛА, но к сожалению все еще уязвим к подделке, помехам и воздействию окружающей среды [3]. В ситуациях, когда GPS перестает корректно функционировать, БПЛА может оценить свою позу с помощью относительной локализации, получив позу относительно последней глобальной позы, полученной с GPS. Затем, учитывая эти данные вычислить текущую глобальное позу.

Основной проблемой относительной локализации является накопление ошибки. Под этим подразумевается эффект, при котором чем дольше устройство определяет свою относительную локализацию, тем больше потенциальная ошибка. Именно поэтому так важно иметь максимально точную оценку относительной локализации, а также необходимость иметь способ сбросить накопленную ошибку.

Данная работа рассматривает проблему относительной локализации в условиях отсутствия GPS.

**SLAM**

Для решения задачи относительной локализации применяться SLAM алгоритмы, которые оценивают позу робота и одновременно строит карту местности на основе полученных данных с различных датчиков.

SLAM и одометрия являются связанными понятиями, но не являются синонимами. Одометрия только оценивает позу робота на основе одного датчика (камеры или лидара). SLAM же включает в себя этап одометрии, но помимо этого он строит карту местности(чего не делает одометрия) и также в его задачи входит объединение данные с различных датчиков(например камер и IMU) .

SLAM может опиратся на один датчик , например : vSLAM – visual-SLAM который использует визуальную одометрию для оценки позы, LIDAR SLAM – использует лирад. Либо на множество датчиков : viSLAM – visual-inertial SLAM(IMU+camera), RTAB-MAP (лидар + камера).

Обычно применяется несколько датчиков, для достижения большей точности и компенсирования недостатков каждого. В результате, в обязанности SLAM алгоритма также входит задача объединения данных с разных датчиков.

Построения карты является важным элементом SLAM. Как было сказано ранее, относительная локализация имеет проблему накопления ошибки. Построение карты помогает сохранять информацию об окружающей среде на протяжении всего пути. Затем алгоритм Loop Clouser («закритие петли»), являющийся одним из этапов SLAM, находит повторно встречаемые элементы окружающей среды, что позволяет сбросить накопленную ошибку и уточнить текущую позу объекта.

В последнее время алгоритмы SLAM активно разрабатываться и изучаются в научных кругах.

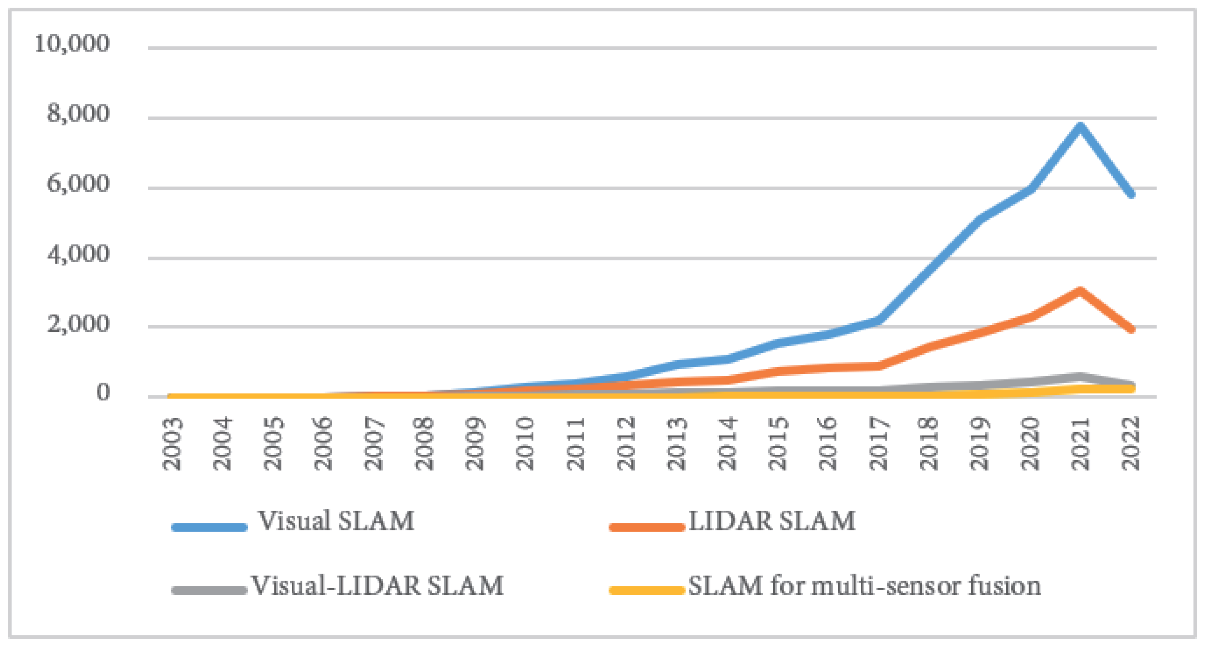


Рисунок 1 – Кол-во цитирований статей связанных с методов SLAM с 2003 по 2022 год в Web of Science

Как видно из рис. 1, количество цитируемых статей, связанных с SLAM методами значительно увеличилось и имеет внушительный масштаб. Спад в 2022 связан с тем что данная статистика собралась в тот же год, и говорит о том что не все статьи еще были опубликованы. [4]

Этих данных достаточно, чтобы доказать, что SLAM сейчас является актуальной темой.

**Цель работы**: Используя различные методы моделирования определить зависимость точности работы алгоритма viSLAM от параметров используемых датчиков и получаемых из них данных.

**Объектом исследования** является алгоритмы оценки положения БПЛА на основе датчиков IMU и камер (далее viSLAM)

**Предметом исследования** является влияние качества датчиков и получаемых данных на работу алгоритмов viSLAM

**Задачи**

Учитывая сказанное, алгоритмы SLAM являются активно исследуемой областью и имеет широкое применение в практических целях так как используются в навигации БПЛА. В связи с этим перед промышленностью встает вопрос, не только в выборе нужного алгоритма SLAM , но и выборе датчиков , для достижения необходимой точности локализации для успешного выполнения задач. С одной стороны, слишком дешёвые датчики вызовут уменьшение точности локализации устройства, в связи с чем выполнение задач будет невозможно. С другой слишком дорогие повлекут избыточные финансовые траты, что не выгодно для бизнеса.

Поэтому в данной работе предпринята попытка помочь с поиском компромисса между качеством датчиков и точностью работы алгоритма. Для достижения цели необходимо решить следующие **задачи**:

* Подготовительная часть
  + Выбрать алгоритм SLAM
  + Выбрать набор данных
  + Выбрать исследуемые параметры датчиков и предложить способы их моделирования
* Практическая часть
  + Смоделировать различные данные соответствующие выбранным параметрам датчиков на основе оригинального набор данных.
  + Выявить зависимости параметров датчиков и точности алгоритма

Данная работа фокусируется на исследовании факторов влияющих на точность алгоритма SLAM. Вопрос связанный с вычислительными нагрузкой в данной работе не рассматривается. В дальнейшем, алгоритму будет предоставлено достаточно вычислительной мощи, чтобы производить максимально точную оценку.

Метод исследования и результаты моделирования, описанные в этой статье, могут быть в дальнейшем использованы при проектировании мобильной роботизированной навигационной системы.

**Структура**

В текущем разделе «1. ВВЕДЕНИЕ» рассказывается об актуальности проблемы рассматриваемой в данной работе и проводиться небольшое поверхностное введение в научную область для лучшего дальнейшего понимания содержания работы. Также ставиться задачи работы , которые в дальнейшем будут выполнены в ходе исследования.

В разделе «2. ОБЗОР АНАЛОГОВ» рассматриваться схожие по теме исследования работы, для выяснения проработанность рассматриваемой области и оценки новизны данной работы.

В разделе «3. ПОДГОТОВИТЕЛЬНАЯ ЧАСТЬ» описывается решение задач связанных с выбором SLAM алгоритма, базы данных и параметрами датчиков и аргументируется почему было принято именно такое решение.

В разделе «4. ПРАКТИЧЕСКАЯ ЧАСТЬ» в соответствии с названием производиться моделирование данных и исследование точности SLAM алгоритма на различных сгенерированных данных .

Наконец в разделе «4. ЗАКЛЮЧЕНИЕ» формулируются заключительные замечания и рекомендации относительно дальнейшего развития.

## 1. ОБЗОР АНАЛОГОВ

Несмотря на активное развитие SLAM в научной литературе, подавляющее большинство статей либо разрабатывают новую систему SLAM, либо создают набор данных для конкретной проблемы (EuRoC, KITTI, TUM-VI и др. [5, 6, 7]), либо фреймворки для удобного создания алгоритмов SLAM (GTSAM, OpenVINS, [8, 9]) или их удобной оценки (EVO[10]). Об этом говорит существования множество даже viSLAM-ов с *открытым* исходным кодом (ORB-SLAM3, SVO, VINS, Okvis, Rovio и несколько других [11, 12, 13, 14, 15]) и различные их вариации и улучшения, созданные за последние 10 лет.

Возможно в связи с тем, что область бурно развивается, и новые алгоритмы регулярно появляться, влиянию датчиков, на которых запускаются данные алгоритмы, уделяется меньшее время. В данном разделе рассматриваться те немногое статьи, которые все-таки фокусируются на этой проблеме.

### 1.1 Исследование влияния шума камеры на работу ORB-SLAM2

В работе [16], изучается влияние различных уровней шумов камеры на работу visual-SLAM. В качестве образца visual-SLAM берётся ORB-SLAM2. Также авторы статьи представляют синтетически сгенерированный набор данных с различными уровнями шума изображения. В ходе создания набора моделируются три вида шума: независимый от освещенности, зависящий от освещенности и смесь этих двух. Также авторы предлагают сравнить работу ORB-SLAM2 на зашумленных данных и на шумоподавленных данных используя быструю и гибкую сверточную нейронную сеть FFDNet[17].

В ходе экспериментов были определены граничные значения шума до которых ATE оставалась стабильной. Что касается шумоподавленных данных, ATE почти во всех экспериментах оставалась стабильной. Другими словами, последовательности с шумоподавлением более надежны и показывали большую точность, чем исходные зашумленные последовательности. Так же было установлено, что для чистой последовательности (без добавления каких либо шумов) среднеквадратичная ошибка (RMSE) ATE после шумоподавления снизилась на 16,75%;

### 1.2 Исследование влияния шума камеры глубины на модифицированный RBPF-SLAM

Авторы статьи [18] изучают влияния случайных шумов, создаваемых камерой глубины съемки ToF, на точность оценки позы мобильного робота на основе усовершенствованного алгоритма RBPF-SLAM. В ходе работы был усовершенствован алгоритм RBPF-SLAM и создана Гауссова модель шумов глубинной камеры, используя которую были сгенерированы зашумленные данные.

В результате были построены графики RMSE траекторий для различных уровней шума . Случайные помехи, создаваемые глубинной камерой ToF, влияют на точность позиционирования мобильного робота. Чем больше шум, тем больше среднеквадратичная погрешность траектории робота. Таким образом когда параметр шума был равен , максимальная среднеквадратичная погрешность траектории робота достигла 0.075м.

### 1.3 Исследование влияния сжатия изображения на точность ORB-SLAM3

В этой работе [19] исследуется влияние сжатия изображений на точность локализации ORB-SLAM3. Для увеличения пропускной способности, данные полученные с БПЛА предлагается сжимать и затем отправлять на внешний сервер, где будет выполняться SLAM. В связи с этим также предлагаются методы по улучшению обработки ORB-SLAM3 сжатых видео и исследуется влияние сжатия на точность модифицированного алгоритма и оригинального.

В результате построены box-диаграммы APE для оригинального ORB-SLAM3 и 4х модификаций на трех сжатых последовательностях, а также для каждого случая были вычислены основные параметры распределения APE такие как: среднее, медиана , минимум/максимум, дисперсия . Опираясь на полученные данные, был сделан вывод, что предложенные модификации повышают устойчивость ORB-SLAM3 к сжатию изображений, обеспечивая более высокую степень сжатия при сохранении желаемой точности локализации.

### 1.4 Исследование влияния изменения разрешения и частоты потокового видео на SVO Pro

В этой статье [20] исследуются характеристики визуальной одометрии при различных сочетаниях разрешения изображения и частоты получения изображений. За пример визуальной одометре берётся алгоритма визуально-инерциальной одометрии SVO Pro Open. В ходе исследования тестируют работу алгоритма на трех различных разрешениях: 848 × 800 пикселей (исходное), 636 × 600 пикселей и 424 × 400 пикселей; и на 4 частототах: 30 (исходная), 25, 20 и 15 Гц. В статье анализируются точность SVO Pro и загрузку ЦП при работе на различных сгенерированных последовательностей.

В результате было выяснено, что для проанализированного промежуточного разрешения (636 × 600 пикселей) может быть достигнут оптимальный компромисс с точки зрения точности локализации, загрузки ЦП и надежности системы. Исследование с изменением частоты показало, что на высоких частотах (25 и 30 Гц) получаются лучшие результаты с точки зрения локализации. Кроме того, из вычислительного анализа было выяснено, что частота 25 Гц обеспечивает значительную экономию вычислительных ресурсов по сравнению с 30 Гц для этого промежуточного разрешения, хотя и с несколько более высокими ошибками перевода. Таким образом, как утверждают авторы, пользователь может найти подходящий компромисс между точностью и ресурсами ЦП в зависимости от доступных вычислительных возможностей.

### 1.5 Вывод

Как можно наблюдать, проблема влияния различных характеристик датчиков на разные алгоритмы SLAM мало рассмотрена в литературе и имеет множество пробелов. Большинство рассмотренных публикаций в основном фокусируется на модификации существующего алгоритма и при экспериментальном исследовании предложенного решения, заодно предоставляют информацию о работе оригинального алгоритма.

Данная работа будет целенаправленно концентрироваться не на исследовании новой модификации, а на свойствах оригинальной модели. Результаты этого исследования могут в дальнейшем применяться в промышленности, при разработке навигационных систем. В ходе работы, будет учтено какие алгоритмы уже были частично исследованы, а какие остались без внимания и учтет это при выборе исследуемого SLAM.

Забегая вперед, для исследований будет выбран алгоритм VINS-Fusion. Как можем видеть, данный алгоритм еще не исследовался с данного ракурса. Другими словами, данная работа действительно представляет научную новизну. Почему был выбран именно VINS-Fusion будет подробно сказано в разделе «3. ПОДГОТОВИТЕЛЬНАЯ ЧАСТЬ».

## 2. ПОДГОТОВИТЕЛЬНАЯ ЧАСТЬ

В данном разделе решаются задачи связанные с выбором алгоритма SLAM, точность которого в дальнейшем будет в различных ситуациях; c выбором набора данных, который ляжет в основу генерируемых данных; и с выбором параметров датчиков. В следующем разделе будет исследоваться, как эти параметры влияют на точность выбранного алгоритма.

### 2.1 Выбор алгоритма SLAM

Так как данная работа нацелена исследовании влиянию параметров датчиков в области локализацию БПЛА, БПЛА имеют характерные особенности связанные с компактными размерами и вычислительной ограниченностью. Поэтому использование лидаров неуместно из-за громоздкости и большой вычислительной нагрузки. Лидары зачастую используются в наземных автопилотах, но на данный момент не применимы в большинстве БПЛА(за исключением очень огромных).

В условиях наружного освещения или при съемке больших сцен использование камер глубины также затруднено.

Благодаря последним достижениям в области проектирования и производства аппаратного обеспечения, появились недорогие и легкие датчики IMU - МЭМС. Благодаря компактности и дешевизне МЭМС датчики стали широко применяться в задачах локализации БПЛА [?]. К сожалению эти датчики имеют большую погрешность и дрейф нуля, в связи с чем зачастую их используют в связке с другими датчиками для компенсирования ошибки.

Такими датчиками может выступить камера. Камера, обладает преимуществами небольшого размера, малого веса, малой мощности и низкой стоимости, а также может предоставлять обширную информацию и проста в использовании. Методы визуального SLAM с использованием камер значительно повысили точность, надежность и эффективность и в последние годы приобретают все большую популярность [22]. На устройстве может быть одна камера – моно или несколько стерео. Для получения большей информации о 3D окружении а также для более точной триангуляции, желательно использовать stereo.

Таким образом, эффективным комплексом датчиков для БПЛА является визуально-инерциальная система, состоящая из недорогого инерциального измерительного блока (IMU) и стерео-камер, которая сочетает в себе преимущества визуальной одометрии (VO) и IMU.

Опираясь на результаты статьи [21], в которой рассматривается и сравниваются множество широко используемых открытых алгоритмов моно и стерео VO/VIO. Результат сравнения можно увидеть на рисуноке 2. Под наши требования подходят VINS-Fusion (VINS-Fusion-imu), Stereo-MSCKF(S- MSCKF), Kimera. Было принято взять **VINS-Fusion** так как он показал лучшую точность среди прочих. (VINS имел наименьшую среднюю RSME по всем видео)

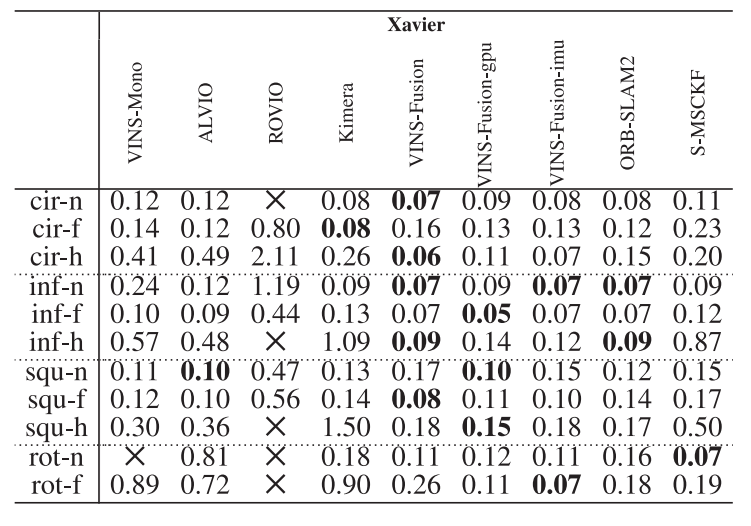


Рисунок 2 – Результат сравнению алгоритмов VO\VIO в статье[21]. По горизонтали располагаются рассматриваемые алгоритмы, по вертикали названия видео данных, на которых тестировались алгоритмы. В ячейках указана RMSE ATE для каждой последовательности.

Также стоит обратить внимания что из раздела «2. ОБЗОР АНАЛОГОВ» видно, что до сих пор, исследования влияние качества датчиков на VINS-Fusion не проводились.

### 2.2 Выбор набора данных

Благодаря активному развитию SLAM алгоритмов, в настоящее время существует множество готовых наборов данных различных датчиков для задач SLAM. Основные из них представлены в рисунке 3.

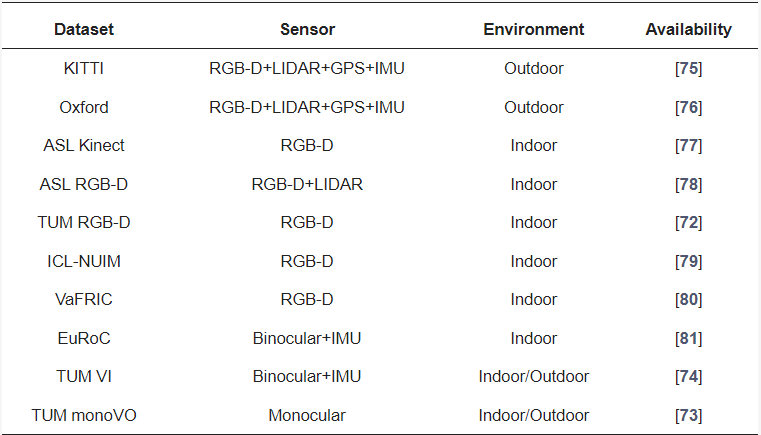


Рисунок 3 – Широко распространенные наборы данных для задач SLAM алгоритмов. Изображение взято из [4].

В связи с тем, что был выбран VINS-Fusion который является стерео viSLAM, из наборов данных представленных в рисунке 3 нам подходят только EuRoC и TUM VI. Было принято решения взять набор **EuRoC**, для моделирования различных параметров датчиков. Это связано с тем, что по личному опыту он более распространен в литературе, связанной с viSLAM, чем TUM VI.

EuRoC содержит 11 видео последовательностей, снятых в трех помещениях(V1, V2, MH). Характерные особенности каждой видеопоследовательности показаны на рисунке 4.

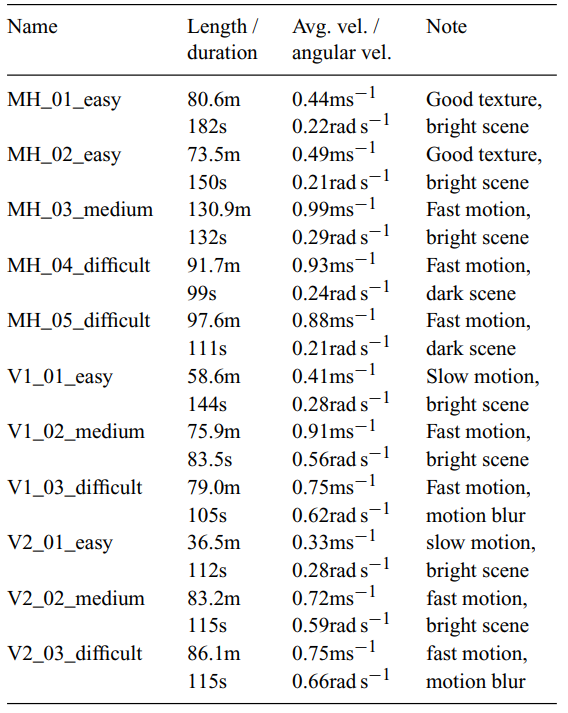


Рисунок 4 – Краткие характеристики видеопоследовательностей EuRoC. Изображение взято из [5].

Также в наборе для каждого видео представлена истинная траектория БПЛА хранимая в .csv формате. Данная траектория получена путем использования системы захвата движения **Vicon** или лазерного трекера **Leica MS50**. Истинная траектория необходима для вычисления ошибки траектории построенной алгоритмом SLAM.

Также стоит упомянуть что разработчиками OpenVINS[9] было обнаружено, что истинная траектория V1\_01\_easy была неточно оценена. В итоге они предоставили более корректный вариант истинной траектории. Подробнее [23]. В связи с этим в дальнейшем в ходе работы будет использоваться обновленный вариант истинной траектории для V1\_01\_easy.

### 2.2 Выбор исследуемых параметров датчиков/TODO

VINS-Fusion использует два типа датчиков: IMU и стерео камеры. В связи с этим, будут моделироваться следующие параметры для этих датчиков.

Для стереокамер:

* Различное разрешение
* Различный уровень шума изображения

## 3. ПРАКТИЧЕСКАЯ ЧАСТЬ

В данном разделе производиться моделирование данных соответствующие различным параметрам датчиков камер и IMU. Исследуется зависимость между параметрами датчиков и точностью VINS-Fusion. На основе проведенных экспериментов и полученных результатов делаться соответствующие выводы.

### 3.1 Исследование влияния параметров камеры

#### 3.1.1 Разрешение

**Моделирование**

Для моделирования различного разрешение потокового видео было решено смоделировать следующие разрешения:

* 480 x 752 – оригинальное
* 432 x 677
* 389 x 609
* 350 x 548
* 315 x 493

Данные значения разрешений были получены уменьшение предыдущего разрешение на 0.9.(Пример: и т.д.)

Из набора данных EuRoC было взято следующие три видео (в скобках указаны сокращения принятые для этого подраздела):

* V1\_01\_easy (V1)
* MH\_03\_medium (MH)
* V2\_03\_difficult (V2)

Данный выбор был обусловлен необходимостью максимально покрыть набор данных, поэтому были взяты последовательности из всех трех комнат, с различными уровнями сложности.

В результате, необходимо смоделировать 15 видео данных для последующего исследования точности VINS-Fusion.

В наборе данных EuRoC данные хранятся в формате bag . Bag это формат файла ROS в котором сохраняются сообщения публикуемые в топиках различными нодами, генерируемый специальными инструментами ROS. Bag файл имеет расширение .bag . В EuRoC в bag файле хранятся данные получаемые из датчиков IMU, камер. Для каждой видеопоследовательности существует свой bag файл.

Поэтому для моделирования, необходимо создать новый bag файл для каждой последовательности изменив в нем только разрешение изображений получаемых с камер.

Для этого была написана программа ***change\_resolution.py*** c использование языка Python и следующих библиотек: rosbag, CvBridge, openCV.

Данная программа создает новые bag файл с различными разрешениями видео внутри, в следующие этапы:

1. C помощью функций библиотеки rosbag, происходит последовательное считывание всех сообщений хранимых в bag файле
2. Если сообщение не принадлежит камерам, то данное сообщение без изменений сохраняется в новых bag файл. В противном случае сообщение передаться в функцию *shrink\_image()* и результат сохраняется в новых bag файл*.*
3. В функции *shrink\_image()* :
   1. происходит конвертация изображение хранимого в сообщении ROS в формат openCV используя библиотеку CvBridge.
   2. Изменение разрешения инструментами библиотеки openCV.
   3. Обратная конвертация измененного изображения с помощью CvBridge.
   4. Возврат нового сообщения

Программа запускается из командной строки с передачей названия изменяемой последовательности, пример см. рисунок 5.



Рисунок 5 – Пример запуска программы *change\_resolution.py*

Таким образом для одной последовательности программа *change\_resolution.py* создает 4 новых bag файла соответствующих различным разрешениям.

Так как необходимо запускать программу *change\_resolution.py* для кадой последовательности , для автоматизации был написан bash скрипт ***chres.sh*** , который автоматически запускал *change\_resolution.py* для нужных последовательностей EuRoC. Пример работы *chres.sh* можно увидеть на рисунке 6.



Рисунок 6 – Пример работы *chres.sh*

Таким образом были созданы 12 последовательностей с разрешениями (432 x 677, 389 x 609, 350 x 548, 315 x 493) для каждой из трех последовательности (V1, V2, MH)

**Запуск VINS-Fusion**

В соответствии с представленной информацией в GitHub репозитории VINS-Fusion , чтобы запустить алгоритм на данных EuRoC, необходимо выполнить 4 команды в различный терманалах. Команды представлены на рисунке 7.

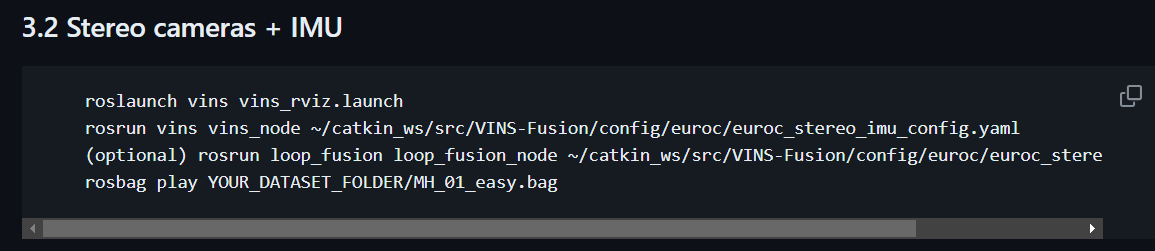


Рисунок 7 – Команды необходимые для запуска VINS\_Fusion

Команда *roslaunch vins vins\_rviz.launch*  запускает ros master и визуализатор rviz в котором отображается построение траектории в ходе работы VINS.

Команда *rosrun vins vins\_node <config.yaml>* запускает основные ноды связанные с визуальной одометрией и построением траектории на основе VO и IMU. В *config.yaml* описываются все переменные алгоритма и параметры используемых датчиков.

Команда *rosrun loop\_fusion loop\_fusion\_node* <*config.yaml>* запускает ноды связанные выполнением Loop Closure. В противном случае алгоритм не будет выполнять уточнение траектории используя метод «закрытия петли». *config.yaml* – совпадает с конфигурационным файлом из предыдущей команды.

*rosbag play ./MH\_01\_easy.bag* – воспроизводит сохранённые данные в bag, имитируя получение данных алгоритмом в реальном времени.

Для запуска VINS использует не только данные из конфигурационного файла *euroc\_stereo\_imu\_config.yaml* но и из зависимых файлов лежащих в папке /config/euroc/, список файлов показан на рисунке 8.

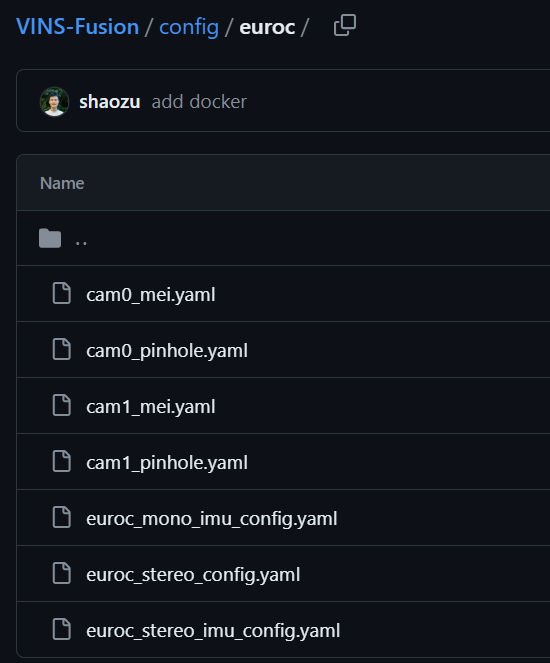
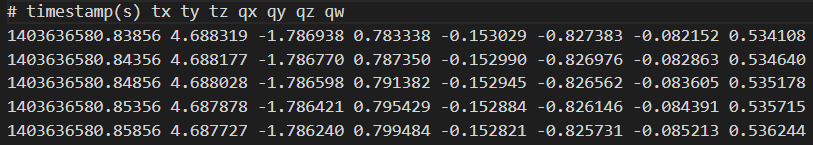


Рисунок 8 – Список конфигурационных файлов для запуска VINS-Fusion на наборе данных EuRoC.

Так как в данных конфигурационных файлахсодержит информацию о параметрах камеры, то для корректного выполнения VINS на различных разрешениях были созданы соответствующие папки с измененными конфигурационные файлами.

Для получения траектории БПЛА в файловом формате из запущенного VINS, была создана программа ***sub.py***. Данная программа создает ноду-подписчика и подписывается на топик, в котором публикуется траектория для отрисовки в rviz. (Название данного топика: «/loop\_fusion/pose\_graph\_path»). Полученные данные сохраняются в файле в формате TUM по пути переданному в программу.

Формат траектории TUM выглядит следующим образом, как показано на рисунке 9. Каждая строка содержит 8 записей, содержащих временную метку (в секундах), положение и ориентацию (в виде кватерниона), причем каждое значение разделено пробелом. Символ # обозначает комментарий.

 Рисунок 9 – Пример того как храниться траектория в формате TUM.

В связи с тем, что VINS необходимо запустить 15 раз были предприняты методы для автоматизации процесса:

1. Для этого первая команда *roslaunch vins vins\_rviz.launch*  была заменена командой *roscore* , для того чтобы запустить только ros master , без запуска графического визуализатора rviz.
2. Для того чтобы была возможность запустить все команды в одном окне консоли, все команды запускаться в фоновом режиме используя в конце символ амперсанда (&)
3. Чтобы работая в фоновом режиме команды не выводили в консоль промежуточные результаты был перенаправлен вывод используя потоки stdout и stderr. В результате команда должна выглядеть следующим образом: *command > /dev/null 2>&1 &*
4. После выполнения работы все процессы связанные с ros уничтожаются, для корректного последующего запуска алгоритма.
5. Написан скрипт ***create\_VINS\_tum.sh***который реализует данные подходы и автоматизирует последовательный запуск алгоритма VINS а также запуск программы ***sub.py*** на всех 15 видео.

В результате работы скрипта ***create\_VINS\_tum.sh*** было создано 15 фалов .tum описывающих траектории полученные при работе VINS-Fusion на последовательностях с различными разрешениями.

**Анализ полученных результатов**

Для анализа и сравнения полученных данных использовался такой инструмент как EVO [10].

EVO имеет следующие консольные команды:

* *evo\_ape* – вычисляет APE(absolute pose error), то же самое что и ATE(absolute trajectory error) между траекторией вычисленной SLAM и истинной траекторией
* *evo\_rpe* - вместо прямого сравнения абсолютных поз, относительная погрешность поз сравнивает движения ("дельта поз")
* *evo\_traj* – инструмент для анализа, построения или экспорта одного или нескольких траектории
* *evo\_res* – инструмент для сравнения одного или нескольких файлов результатов из evo\_ape или evo\_rpe

В наборе EuRoC истинная траектория храниться в формате .csv как было сказано в подразделе «3.2 Выбор набора данных». Траектория построенная VINS храниться в формате .tum . Для согласования форматов был применен инструмент *evo\_traj,* в результате чеговсе истинные траектории были конвертированы из формата .csv в формат .tum

Для описания точности работы VINS-Fusion будет использоваться метрика APE. APE для времени t вычисляется следующим образом:

Где – поза истинной траектории в момент t , - поза вычисленной SLAM траектории в момент t.

После получения APE для каждого момента времени, можно вычислить различные статистики численно описывающие конкретную вычисленную траекторию алгоритмом SLAM.

Примером такой статистики может выступить RMSE:

Для вычисления APE между истинной траекторией и полученной применяется команда *evo\_ape.* Для нее необходимо указать формат сравниваемых данных, нахождение файлов истинной и вычисленной траектории и способ геометрического сопоставления.

Последнее является очень важным, под сопоставлением понимается какие преобразования можно использовать при наложения вычисленной траектории на истинную. Пример различных сопоставлений показан на рисунке 10.

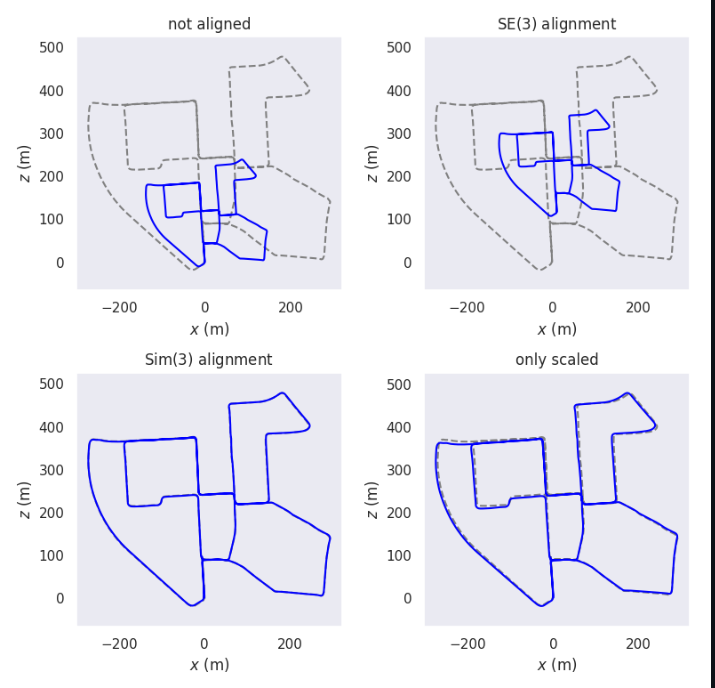


Рисунок 10 – Различные способы сопоставления траекторий. Сверху-вниз , слева-направо: без сопоставления, вращение и перемещение, вращение перемещение и масштаб , только масштаб.

Может казаться, что лучше всего сравнивать траектории без сопоставления, но тогда для оригинального набора данных мы получаем следующий результат , рисунок 11 . Можем видеть что ошибка достаточно большая, хотя очевидно что фигуры похожи с точностью до сдвига. На рисунке 12 показаны те же траектории но с применением сопоставления и как видно траектории почти совпадают.

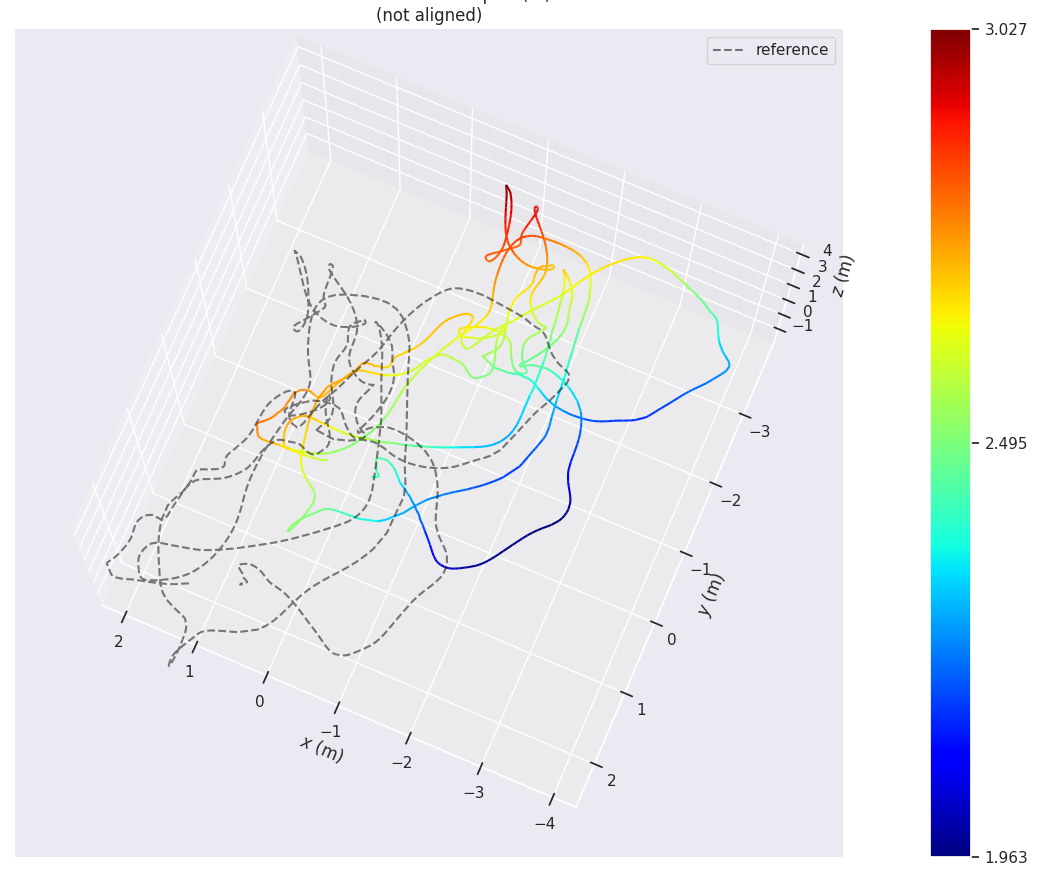


Рисунок 11 – Результат сравнения истинной траекторий и вычисленной VINS без сопоставления. Пунктиром обозначена истинная траектория, цветной линией траектория полученная из VINS. Последовательность V1\_01\_easy.

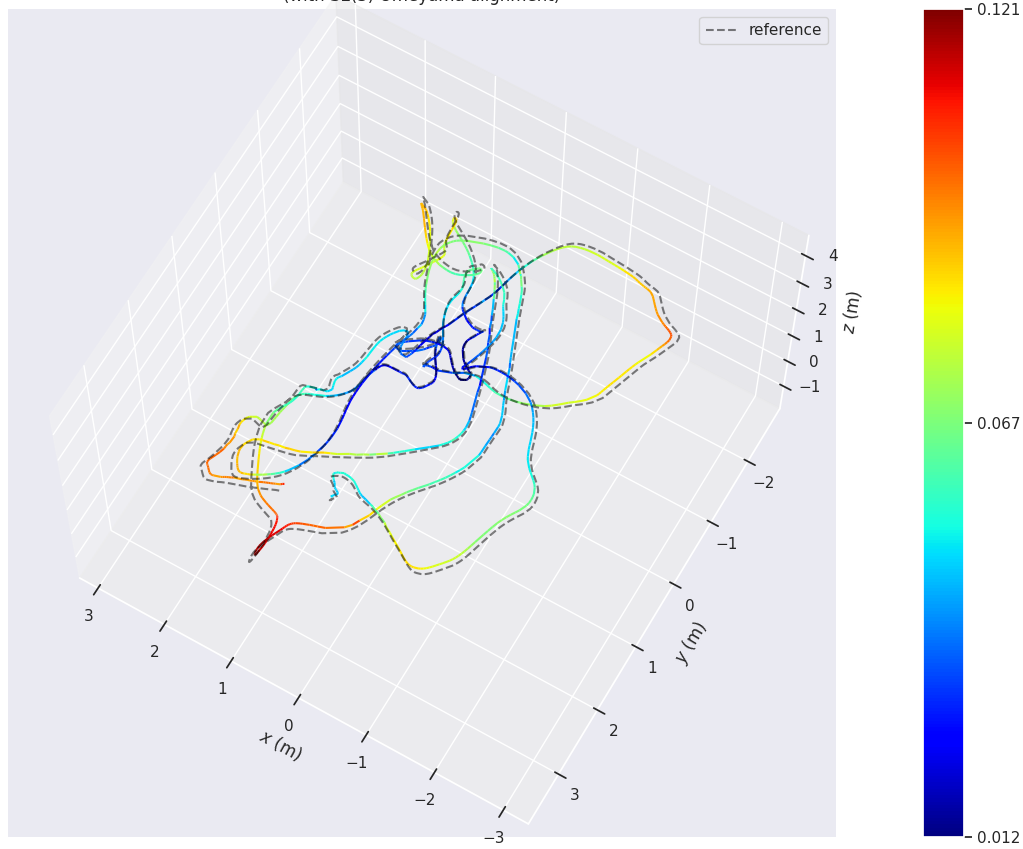


Рисунок 12 – Результат сравнения истинной траекторий и вычисленной VINS с сопоставлением. Пунктиром обозначена истинная траектория, цветной линией траектория полученная из VINS. Последовательность V1\_01\_easy.

Но тогда возникает другая проблема, если да каждой пары траекторий производить различное сопоставление, итоговое результаты будет невозможно соглосовать между собой. Для этого команда *evo\_ape* будет использоваться с флагом *--align\_origin*, который отвечает за простое сопоставление по исходной точке. В данном случае, сопоставление у всех будет одинаковое и можно будет удобно сравнивать дрейф оценки траектории.

В результате, итоговая команда для вычисления APE выглядит следующим образом:

*evo\_ape tum <путь до ист. тр.> <путь до выч. тр.> -p --align\_origin --save\_results results.zip*

Результат которой сохраняется в файл *results.zip*

После выполнения сравнения истинных траекторий с вычисленными VINS, с использованием инструмента *evo\_ape* , было получено множество файлов фалов *results*.zip. Для обработки и сравнения этих файлов используется инструмент EVO *evo\_res,* который соединяет полученные APE из разных последовательностей в один график.

Таким образом был получен график APE для всех разрешений для последовательности V1\_01\_easy, см. рисунок 13.

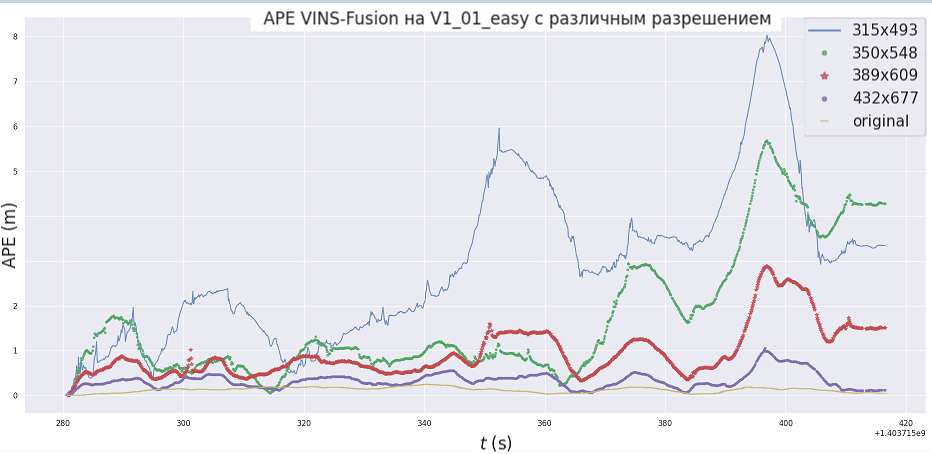


Рисунок 13 – График APE VINS-Fusion для всех разрешений для последовательности V1\_01\_easy.

Также *evo\_res* предоставляет самые популярные статистики APE для каждой последовательности. В таблицу были собраны все полученные статистики для всех 15 последовательностей, см. рисунок 14.

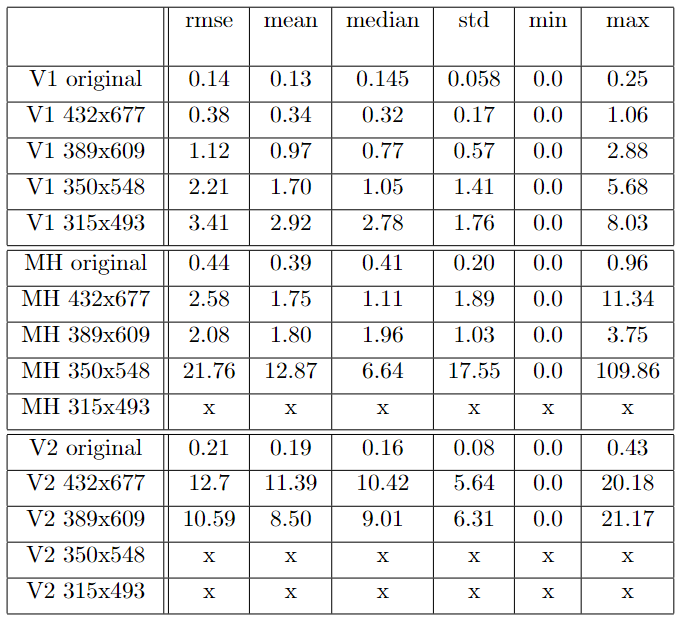


Рисунок 14 – Таблица статистик APE VINS-Fusion для каждой рассматриваемой последовательности данных в этом подразделе. x – обозначено, что алгоритм потерял траекторию и ошибка превысила 100м.(V1 - V1\_01\_easy, V2 - V2\_03\_difficult, MH- MH\_03\_meddium)

В ходе выполнения экспериментов были получены:

1. График APE VINS-Fusion для различных разрешений V1\_01\_easy, чтобы изучить как влияет изменение разрешения на точность алгоритма. Рисунок 13.
2. Таблица статистик APE VINS-Fusion для каждой рассматриваемой последовательности, чтобы оценить изменение точности от разрешения для различных изначальных видеоданных.

Анализ полученных результатов будет произведен далее.

**Вывод**

Как и ожидалось из рисунока 13 видно, что чем меньше разрешение, тем выше получаемая APE, что говорит об ухудшении точности работы алгоритма. Колебания на графики скорее всего связаны с сложными участками траектории такие как повороты.

Из таблицы из рис.14 можно сделать вывод что чем сложнее данные тем лучше нужны датчики, об этом говорит то что на простой последовательности V1\_01\_easy VINS справлялся на всех разрешениях несмотря на растущую ошибку. На сложной V2\_03\_difficult VINS уже не смог корректно стоить траекторию на разрешении 350 x 548 и 315 x 493.

Если говорить о дальнейшем применении полученных данных, то можно разобрать пример выбора разрешения датчиков таким чтобы RMSE APE была не больше 1 м:

* для задач с медленным движением БПЛА и яркой сценой , другими словами условиями близкими к V1\_01\_easy если средняя ошибка должна быть меньше метро тогда лучше использовать камеры с разрешением >389x609 и тд. С другой стороны если средняя ошибка 3 метра а максимальная 8метров на протяжении 80 метров допустима , то можно использовать камеры и с разрешением >315 x 493.
* для задач с быстрым движением как в MH\_03\_medium использование камер с разрешением меньшим или равным 350 x 548 непримемлемо. Для того чтобы достичь средней точности меньше 1 метра , рекомендуется использовать разрешение в районе 752x480.
* Для сложных задач с быстрым движением и размытием подвижных объектов как в последовательности V2\_03\_difficult желательно использовать разрешение камер около 752x480 но никак не ниже , так как ошибка значительно растет с уменьшение разрешения на сложных данных.

#### 3.1.2 Шум

В данном подразделе будет исследоваться влияние шума изображения на точность VINS. Процесс исследования будет проходит аналогичным образом поэтому настолько подробно как в предыдущем разделе описываться не будет .

**Моделирование**

В изображении присутствует много типов шума, таких как шум фотонного выстрела, шум темнового тока, шум с фиксированным шаблоном смещения, шум с фиксированным шаблоном темнового тока, шум последователя источника, шум квантования и шум сброса сенсорного узла. В результате чего шум изображения является интенсивно и пространственно зависимым. Такой шум сложно моделировать и регулярно появляться новые методы генерации более реалистичного шума.

Самым простым способом приближением реального шума принято считать Гауссовский шум. Гауссовский шум моделируется следующим образом:

1. генерируется множество случайных значений соответствующие нормальному распределению с , количество чисел равно кол-ву пикселей в изображении
2. К каждому значению интенсивности пикселя в изображении прибавляется свое случайное число.

Более формальным языком, Гауссовский шум выглядит мледующим образом:

Где,

- зашумленное изображение

– чистое изображение

– нормальное распределение

Именно такой шум будет использоваться в данной работе. Для моделирования различного уровня Гауссовского шума было решено смоделировать шум со следующими значениями :



Из набора данных EuRoC было взято следующие три видео (в скобках указаны сокращения принятые для этого подраздела):

* V1\_01\_easy (V1)
* V2\_02\_medium (V2)
* MH\_04\_difficult (MH)

Данный выбор был обусловлен необходимостью максимально покрыть набор данных, поэтому были взяты последовательности из всех трех комнат, с различными уровнями сложности.

Для генерации данных использовалась программа ***add\_noise\_img.py***, которая работает аналогично *change\_resolution.py* программе описанной подробно в предыдущем подразделе. Отличием является замена функции *shrink\_image* на *add\_noise\_image*, которая собственно и добавляет шум с нужной сигмой.

В результате чего было сгенерировано 15 последовательностей , по 5 различных шумов для 3 оригинальных последовательностей . В сумме с оригинальными данными, набор для исследования влияния шума включает 18 последовательностей.

**Запуск VINS-Fusion**

Тестирование алгоритма VINS производилась аналогичным образом с использованием программы ***create\_VINS\_tum.sh***, за исключение только того, что для данной ситуации не изменялись оригинальный файлы конфигурации. Это связанно с тем что в VINS не передаться никаких априорных данных об интенсивности шума на изображении при запуске.

В результате работы скрипта ***create\_VINS\_tum.sh*** было создано 18 фалов .tum описывающих траектории полученные при работе VINS-Fusion на последовательностях с различным уровнем шума.

**Анализ полученных результатов**

Для анализа и сравнения полученных данных также использовался аналогичным образом инструмент EVO [10].

В результате чего был получен график APE для всех уровней шума для последовательности V1\_01\_easy, см. рисунок 15.

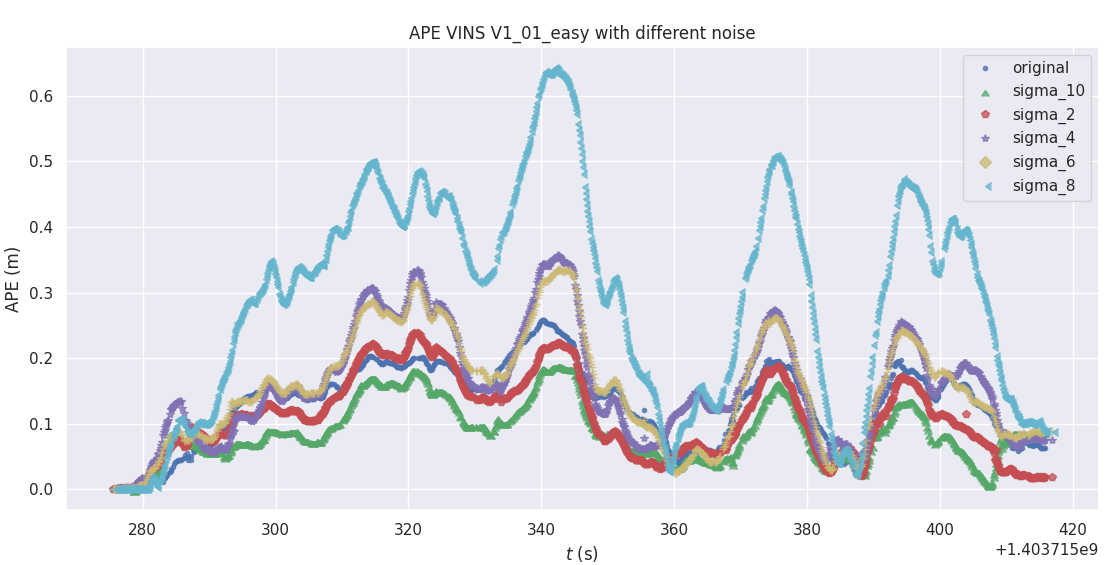


Рисунок 15 – График APE VINS-Fusion для всех уровней шума для последовательности V1\_01\_easy.

Для большей визуальной наглядности, данная информация была представлена в виде box-графика, см. рисунок 16.

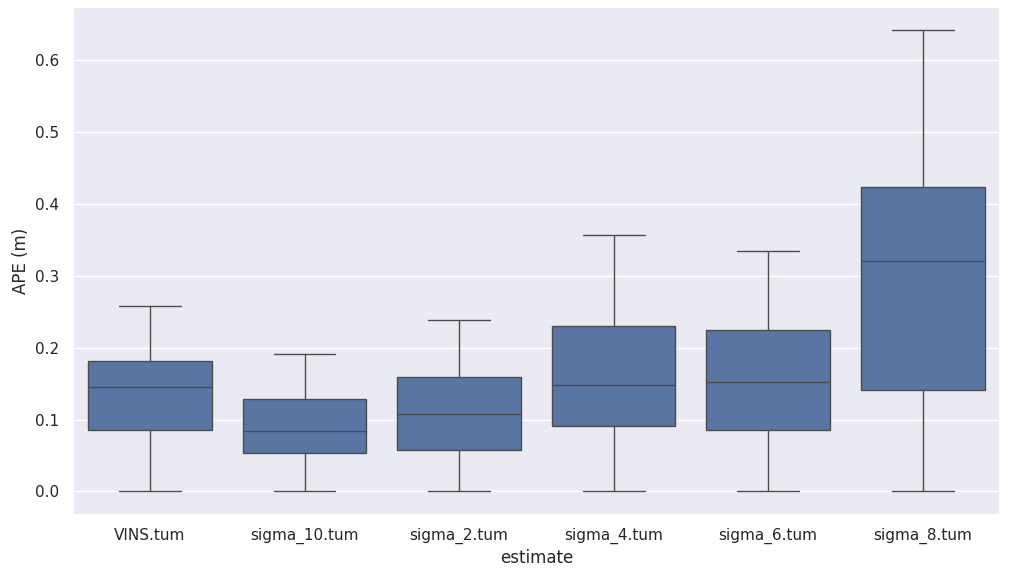


Рисунок 16 – Box-график APE VINS-Fusion для всех уровней шума для последовательности V1\_01\_easy. (VINS.tum – неизмененные данные)

Также была составлена таблица, в которую были собраны все полученные статистики для всех 18 последовательностей, см. рисунок 17.

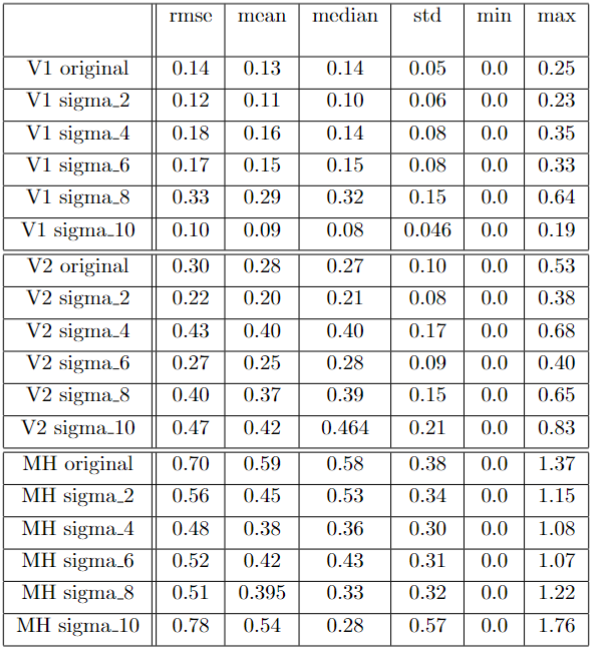


Рисунок 17 – Таблица статистик APE VINS-Fusion для каждой рассматриваемой последовательности данных с различными уровнями шума (V1 - V1\_01\_easy, V2 - V2\_02\_medium, MH- MH\_04\_difficult)

В ходе выполнения экспериментов были получены:

1. График APE VINS-Fusion для различных уровней шума V1\_01\_easy, чтобы изучить как влияет уровень шума на точность алгоритма. Рисунок 15 и 16.
2. Таблица статистик APE VINS-Fusion для всех рассматриваемых последовательности, чтобы оценить изменение точности от уровня шума для различных изначальных видеоданных.

Анализ полученных результатов будет произведен далее.

**Вывод**

Из рисунка 16 видно, что оказывается не сильно влияет на точность алгоритма, т.к. несмотря на увеличение шума APE в среднем оставалась в районе 0.5 метров. Хоть и из рисунка 15 кажется что алгоритм на последовательности с шумом сигма 8 отработал плохо, на самом деле это не так , просто масштаб по APE очень небольшой т.к. почти на всех последовательностях VINS отработал хорошо.

Из таблицы из рис.17 можно сделать вывод что чем сложнее данные тем меньше точность, об этом говорит то что на простой последовательности V1\_01\_easy при максимальном шуме средняя ошибка была 0.1м , а уже на MH\_04\_difficult 0.7м.

Для оценки шума на пользовательском наборе данных существует множество подходов, которые оценивают реальный шум как Гауссовский и помогают вычислить соответствующую σ. Имея эти данные можно предполагать ожидаемую точность VINS на этих данных. Так можно утверждать, что с шумом на пользовательских данных сопоставимый с Гауссовским шумом с уровнем σ=10, алгоритм VINS справляется достаточно неплохо. Даже на сложных данных максимальная ошибка была 1.7м.

## 4. ОБЕСПЕЧЕНИЕ КАЧЕСТВА РАЗРАБОТКИ

В данном разделе изучается вопрос определении характеристик качества разработки SLAM алгоритмов на примере алгоритмаVINS-Fusion и продемонстрированно использование полученных результатов для определения операциональных определений (ОО) качества алгоритма для потребителей.

В роли потребителей выступают предприятия и компании разрабатывающие на коммерческой основе и не только системы автономной навигации. В разработку навигационных систем входит как создание аппаратной части, так и программной.

Для оценки качества программной составляющей, необходимо составить операциональное определение качества и проверить соответствие ПО данному качеству.

Для примера будет рассмотрен элемент навигационной системы – относительная локализация. В роли критерия качества выступит точность алгоритма. Предположим, что у нас уже выбрана используемая аппаратура, пусть будет это стерео-камеры с разрешением 680x430 и какой-нибудь датчик IMU. Необходимо определить является ли созданный алгоритм локализации точным. Для определения этого можно ориентироваться на полученные в данной работе результаты, учитывая что VINS-Fusion является самым точным viSLAM алгоритмом из общедоступных. Таким образом имеем следующее операциональные определение точности предоставленное в рисунке 18.

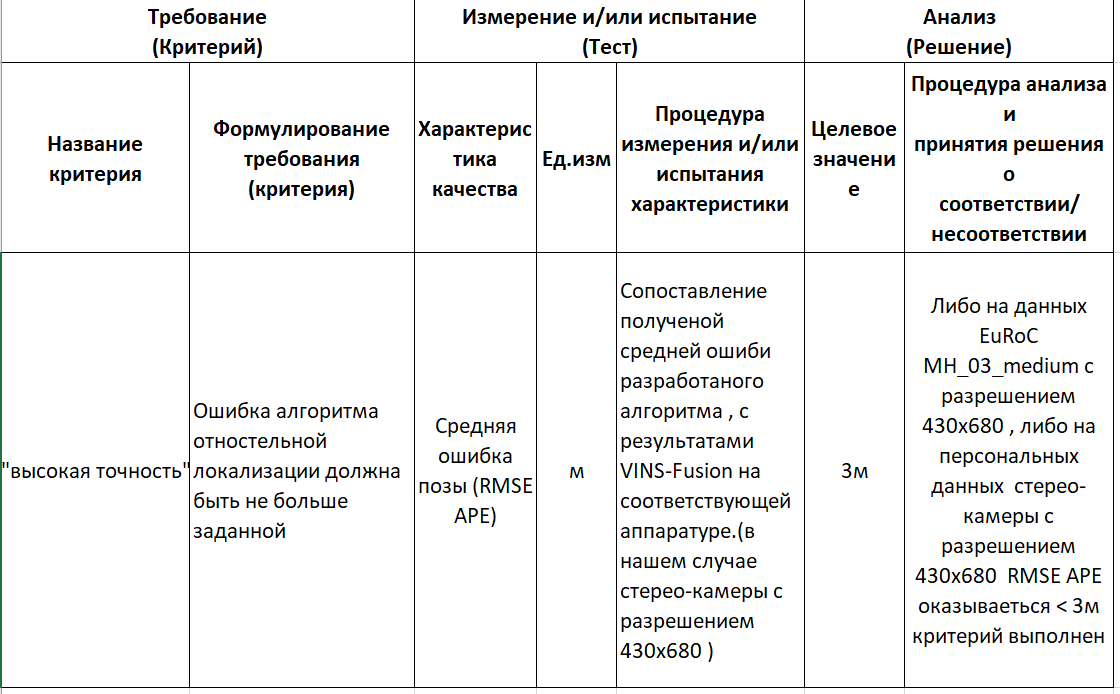


Рисунок 18 – Предложенное операциональное определение точности алгоритма

## ЗАКЛЮЧЕНИЕ

В ходе выполнения работы были решены все поставленные задачи, а именно: был выбран алгоритм SLAM – VINS-Fusion , был выбран набор данных EuRoC, были выбраны параметры датчиков влияние которых в дальнейшем моделировалось и исследовалось. Также были смоделировать данные соответствующие выбранным параметрам датчиков на основе оригинального набор данных. Протестирован VINS-Fusion на сгенерированных данных и построены графики и таблицы отражающие зависимости параметров датчиков и точности алгоритма. Данные этапы были реализованы с помощью таких инструментов как: Python, bash, Opencv, ROS, EVO.

Зависимость разрешения камеры и точности алгоритма VINS-Fusion отражает график (рис.13) APE VINS-Fusion для последовательности V1\_01\_esasy c различным разрешением и таблица (рис.14) основных статистик APE для 15 последовательностей состоящих из 3 последовательностей из EuRoC , каждая из которых была представлена в 5 различных разрешениях. Как было продемонстрированно в выводе подраздела «4.1.1 Разрешение» предоставленные данные помогают выбрать подходящее разрешение камеры для необходимой задачи.

Зависимость уровня шума изображения и точности алгоритма VINS-Fusion отражает график APE VINS-Fusion для последовательности V1\_01\_esasy c различным уровнем шума (рис.15 и 16) и таблица (рис.17) основных статистик APE для 18 последовательностей состоящих из 3 последовательностей из EuRoC , каждая из которых была представлена с 6 различными шумами.

Также в разделе «4. ОБЕСПЕЧЕНИЕ КАЧЕСТВА РАЗРАБОТКИ», было продемонстрированно как можно применять полученные результаты для составления операциональных определений качества для алгоритмов визуально-инерциальной локализации.

В дальнейшем планируется также исследовать влияние следующих параметров на точность VINS:

* Для камеры:
  + - Уровень не гауссовского шума, который будет более приближен к реальному.
    - Частота кадров в секунду
* Для IMU:
  + - Уровень белого шума
    - Уровень смещение нуля (bias)

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Стратегия развития беспилотной авиации Российской Федерации на период до 2030 года и на перспективу до 2035 года. [Электронный ресурс].URL: <http://static.government.ru/media/files/3m4AHa9s3PrYTDr316ibUtyEVUpnRT2x.pdf>
2. Abdulla Al-Kaff, David Martín, Fernando García, Arturo de la Escalera, José María Armingol, Survey of computer vision algorithms and applications for unmanned aerial vehicles. URL: <https://www.sciencedirect.com/science/article/abs/pii/S0957417417306395#preview-section-abstract>
3. Nasser Gyagenda, Jasper V. Hatilima, Hubert Roth, Vadim Zhmud, A review of GNSS-independent UAV navigation techniques. URL: <https://www.sciencedirect.com/science/article/abs/pii/S0921889022000343#preview-section-abstract>
4. Chen, W.; Zhou, C.; Shang, G.; Wang, X.; Li, Z.; Xu, C.; Hu, K. SLAM Overview: From Single Sensor to Heterogeneous Fusion. Remote Sens. 2022, 14, 6033. <https://doi.org/10.3390/rs14236033>
5. M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. Achtelik and R. Siegwart, The EuRoC micro aerial vehicle datasets
6. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The KITTI dataset.
7. Schubert, D.; Goll, T.; Demmel, N.; Usenko, V.; Stückler, J.; Cremers, D. The TUM VI benchmark for evaluating visual-inertial odometry.
8. Dellaert, Frank. “Factor Graphs and GTSAM: A Hands-on Introduction.” (2012).
9. P. Geneva, K. Eckenhoff, W. Lee, Y. Yang and G. Huang, "OpenVINS: A Research Platform for Visual-Inertial Estimation,"
10. Grupp, Michael. "evo: Python package for the evaluation of odometry and slam." (2017).
11. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.M.; Tardós, J.D. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual–Inertial, and Multimap SLAM
12. Forster C., Pizzoli M., Scaramuzza D. SVO: Fast semi-direct monocular visual odometry //2014 IEEE international conference on robotics and automation (ICRA). – IEEE, 2014. – С. 15-22.
13. Qin T., Li P., Shen S. Vins-mono: A robust and versatile monocular visual-inertial state estimator //IEEE Transactions on Robotics. – 2018. – Т. 34. – №. 4. – С. 1004-1020.
14. Leutenegger S. et al. Keyframe-based visual–inertial odometry using nonlinear optimization //The International Journal of Robotics Research. – 2015. – Т. 34. – №. 3. – С. 314-334.
15. Bloesch M. et al. Robust visual inertial odometry using a direct EKF-based approach //2015 IEEE/RSJ international conference on intelligent robots and systems (IROS). – IEEE, 2015. – С. 298-304.
16. Cao L., Ling J., Xiao X. Study on the influence of image noise on monocular feature-based visual slam based on ffdnet //Sensors. – 2020. – Т. 20. – №. 17. – С. 4922.
17. Zhang K., Zuo W., Zhang L. FFDNet: Toward a fast and flexible solution for CNN-based image denoising //IEEE Transactions on Image Processing. – 2018. – Т. 27. – №. 9. – С. 4608-4622.
18. Zhang S. The Research of RBPF-SLAM Accuracy under the Influence of Depth Camera Noises //2020 International Conference on Computing and Data Science (CDS). – IEEE, 2020. – С. 439-442.
19. Wang G. Robust visual SLAM with compressed image data: A study of ORB-SLAM3 performance under extreme image compression. – 2023.
20. Godio S. et al. Resolution and Frequency Effects on UAVs Semi-Direct Visual-Inertial Odometry (SVO) for Warehouse Logistics //Sensors. – 2022. – Т. 22. – №. 24. – С. 9911.
21. Jeon J. et al. Run your visual-inertial odometry on NVIDIA Jetson: Benchmark tests on a micro aerial vehicle //IEEE robotics and automation letters. – 2021. – Т. 6. – №. 3. – С. 5332-5339.
22. Yang N. et al. Challenges in monocular visual odometry: Photometric calibration, motion bias, and rolling shutter effect //IEEE Robotics and Automation Letters. – 2018. – Т. 3. – №. 4. – С. 2878-2885.
23. Getting Started » Supported Datasets | OpenVINS. [Электронный ресурс] URL:<https://docs.openvins.com/gs-datasets.html>

## ПРИЛОЖЕНИЕ А