Министерство образования и науки Российской Федерации
Санкт-Петербургский политехнический университет им. Петра Великого
Институт прикладной математики и механики
Высшая школа прикладной математики и вычислительной физики

Работа допущена к защите

Руководитель ОП

_____ К.Н.Козлов

_____ 20_____г.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА

ВИЗУАЛЬНАЯ ОДОМЕТРИЯ В УСЛОВИЯХ ИНТЕРВАЛЬНОЙ НЕОПРЕДЕЛЕННОСТИ

по направлению 01.04.02 «Прикладная математика и информатика»
по образовательной программе 01.04.02_02 «Системное программирование»

Выполнил

студент гр. 3640102/90201 Н.С. Коваленко

Научный руководитель

к.ф.-м.н., доцент ВШПМиВФ ИПММ А.Н. Баженов

Консультант

по нормоконтролю Л.А. Арефьева

Санкт-Петербург
2021 год

САНКТ-ПЕТЕРБУРГСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ПЕТРА ВЕЛИКОГО

Институт прикладной математики и информатики

Высшая школа прикладной математики и вычислительной физики

УТВЕРЖДАЮ

Руководитель ОП

_____ К.Н. Козлов

« » _____ 20 ____ г.

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

студенту Коваленко Надежде Сергеевне, гр. 3640102/90201

1.Тема работы: Визуальная одометрия в условиях интервальной неопределенности;

2. Срок сдачи студентом законченной работы: 15 июня 2021;

3. Исходные данные по работе:

- Видео с камер, которые находились на движущихся объектах;
- Набор изображений, которые представляют кадры с видео с камер, которые находились на движущихся объектах;

4. Содержание работы (перечень подлежащих разработке вопросов):

Введение

1. Обзор методов визуальной одометрии
   - 1.1. Применимость визуальной одометрии
   - 1.2. Как сейчас проводится одометрия
   - 1.3. Интервальная неопределенность
   - 1.4. Избавление от выбросов
2. Алгоритм ORB-SLAM2
   - 2.1. Модуль распознавания места на основе DBoW2
     - 2.1.1. Бинарные признаки
     - 2.1.2. База данных изображений
     - 2.1.3. Алгоритм обнаружения петель
   - 2.2. Построение 3-D модели по 2-D изображению
   - 2.3. Описание алгоритма ORB-SLAM2

2.3.1. Моно-, стерео-, RGB изображения

2.3.2. Загрузка системы

2.3.3. Регулировка связки с монокуляром и стерео

2.3.4. Замыкание цикла и полный BA

2.3.5. Вставка ключевого кадра

3. Интервальная регрессия

3.1. Определения

3.2. Постановка задачи

3.3. Алгоритм построения

3.4. Метод центра неопределенности

4. Анализ результатов

4.1. Описание данных

4.2. Результат работы ORB-SLAM2

4.3. Результат работы метода центра неопределенности

Заключение

Список использованных источников

5. Дата выдачи задания: 15 декабря 2020

Руководитель ВКР _____ А.Н. Баженов

(подпись)

Задание принял к исполнению _____

(дата)

Студент _____ Н.С. Коваленко

(подпись)

РЕФЕРАТ

На 44 с., 25 рис., 4 табл.

КЛЮЧЕВЫЕ СЛОВА:ВИЗУАЛЬНАЯ ОДОМЕТРИЯ, ИНТЕРВАЛЬНАЯ НЕОПРЕДЕЛЕННОСТЬ, ЛОКАЛИЗАЦИЯ

Тема выпускной квалификационной работы: ”Визуальная одометрия в условиях интервальной неопределенности.”

Данная работа посвящена исследованию применимости методов решения ИСЛАУ для задачи визуальной одометрии при перемещении роботов-спасателей.

Задачи, которые решались в ходе исследования:

- Поиск и выполнение метода решения задачи визуальной одометрии;
- Поиск датасетов с камер, которые находились на передвигающихся объектах;
- Изучение методов решения ИСЛАУ.

В ходе работы в качестве входных данных был выбран датасет KITTI, состоящий из покадровых изображений видео рядов, снятых с передвижных объектов. Для решения задачи визуальной одометрии был выбран алгоритм `ORB-SLAM2`, который на вход принимает набор изображений, а возвращает в результате координаты построенной карты передвижения объекта. Далее полученные данные были преобразованы и проанализированы для дальнейшего рассмотрения и решения ИСЛАУ, чтобы построить уравнения второго порядка при повороте двигающегося объекта. Был реализован метод центра неопределенности. Данный метод применим для поставленной задачи.

ABSTRACT

44 pages, 25 pictures, 4 tables.

KEYWORDS: VISUAL ODOMETRY,INTERVAL UNCERTAINTY, LOCALIZATION

The theme of the final qualifying work: ”Visual odometry in conditions of interval uncertainty.”

This work is devoted to the study of the applicability of methods for solving ISLAU for the problem of visual odometry when moving rescue robots.

The research set the following goals:

- Search and execution of method for solving the problem of visual odometry;
- Search for datasets from cameras that were on moving objects;
- Search of methods for solving ISLAU.

In the course of the work, the KITTI dataset was chosen as input data, which consists of time-lapse images of video sequences taken from mobile objects. To solve the visual odometry problem, the `ORB-SLAM2` algorithm was chosen, which takes a set of images as input, and returns the coordinates of the constructed object movement map as a result. Further, the obtained data were transformed and analyzed for further consideration and solution by ISLAU in order to construct second-order equations for the rotation of a moving object. The center of uncertainty method was implemented. This method is applicable for the task at hand.

Содержание

Введение

Диссертация посвящена применению методов интервального анализа в визуальной одометрии.

Визуальная одометрия — метод оценки положения и ориентации робота или иного устройства с помощью анализа последовательности изображений, снятых установленной на нем камерой или камерами. Большинство существующих подходов к визуальной одометрии основаны на следующих этапах:

1. Получение входных данных
    - Моно-, стерео- камеры;
2. Корректировка изображения
    - Устранение искажений, шумов;
3. Обнаружение признаков в разных кадрах
    - Сопоставление изображений;
    - Выделение признаков;
    - Построение поля оптического потока;
4. Выявление выбивающихся значений векторов поля оптического потока и их корректировка
5. Оценка движения камеры по скорректированному оптическому потоку

Интервальный анализ — раздел математики, получивший развитие во второй половине XX века. Это математическая дисциплина, предметом которой является решение задач с интервальными неопределённостями и неоднозначностями в данных, возникающими в постановке задачи или на промежуточных стадиях процесса решения, метод которой характеризуется рассмотрением множеств неопределённости как самостоятельных целостных объектов, посредством определения над ними арифметических, аналитических и т.п. операций и отношений.

Во многих практических приложениях, начиная от беспилотных автомобилей и заканчивая промышленным применением мобильных роботов, важно принимать во внимание интервальную неопределенность при выполнении визуальной одометрии, то есть при оценке того, как наше положение и ориентация меняются с течением времени. В частности, одним из важных аспектов этой проблемы является обнаружение несоответствий в данных.

1 Описание роботов спасателей

В современном мире возникают различные аварийные ситуации - техногенные, природные, социальные. Они угрожают безопасности человека. Если случилась чрезвычайная ситуация, при которой спасатели не могут приступить к работе из-за различных факторов, на помощь человеку приходят роботы-спасатели.

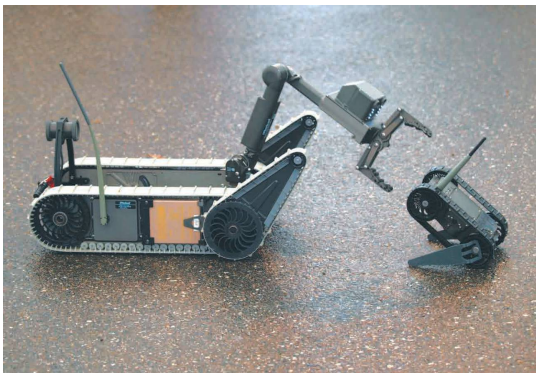
Робот — это автоматическое устройство, которое осуществляет различного рода механические операции и действует по заданному алгоритму без вмешательства человека.

Рассмотрим ситуацию обрушения здания. Человек остался под обломками. Он не может двигаться. Со временем количество воздуха уменьшается. До пострадавшего не могут добраться спасатели, потому что не знают его местоположение. Тогда на помощь приходят роботы, которые могут проникнуть в труднодоступные и опасные места, могут отправить координаты пострадавшего спасателям, а также в их корпусе есть аптечка и дополнительные приспособления для пострадавшего человека.

Система подобных роботов устроена так, что они могут видеть окружающее пространство с помощью различных датчиков и камер. Они могут работать как в дневное, так и в ночное время. Передвижение роботов не будет затруднено из-за дыма, огня, тумана или иных причин потерь видимости, так как для этого их и конструируют, дополняя не только камерами, но и датчиками расстояния, света и другими.

Роботы-спасатели делятся по типу работ: обнаружение пострадавших, устранение завалов, снабжение необходимыми вещами пострадавших.

К первому типу относятся роботы, которые невелики по габаритам, могут проникать в труднодоступные места, видят сквозь стены, могут распознать не только движущегося человека, но и пострадавшего без сознания. Для этого используют также собак, но они не могут проникнуть в такие узкие места, как роботы. Также для собаки подобные задачи опасны для жизни. Примеры таких роботов - Cougar10-LTM, змееподобный Snakebot, представленный на рис. 1.



a)

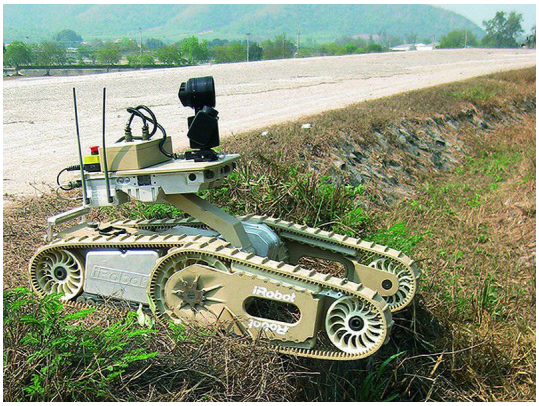
b)

Рис. 1: Примеры роботов-спасателей для обнаружения пострадавших: a) Cougar10-LTM, b) змее-подобный Snakebot.

Ко второму типу относятся роботы, которые необходимы для устранения завалов. Они захватывают крупные глыбы и перемещают их в более подходящие места. У этих роботов крупные размеры, грузоподъемность конечностей может достигать 500 кг. Суммарный вес одного робота примерно 6 тонн. Примером такого робота является Tmsuk T-52 Enryu, который представлены на рис. 2.

Рис. 2: Пример робота-спасателя для устранения завалов.

К третьему виду роботов-спасателей относятся роботы, которые доставляют вещи первой необходимости для пострадавших - аптечку, воду, аппарат для дыхания и другое. Такие роботы еще называются роботами-трансферами. Они небольшого размера, с определенной грузоподъемностью, способны объезжать преграды. К таким роботам можно отнести TARDEC, МРК-35, которые представлены на рис. 3.



a)

b)

Рис. 3: Примеры роботов-спасателей для доставки вещей первой необходимости пострадавшим: a) Tardec, b) МРК-35.

Для своевременной помощи человеку роботу необходимо быстро и корректно перемещаться по неизвестной местности, для этого нужно решить задачу визуальной одометрии.

2 Обзор методов визуальной одометрии

2.1 Применимость визуальной одометрии

Чтобы беспилотные объекты передвигались исправно, нужно чтобы они правильно отслеживали свое положение и ориентацию. Часто для вычисления местоположения можно использовать глобальную систему позиционирования(Global Positioning System - GPS). Однако в случае GPS возможны отключения и сбои из-за высоких зданий, поэтому движение объекта необходимо вычислять шаг за шагом, начиная с последней известной позиции. Соответствующие оценки известны как одометрия [1, 2, 5, 9].

Одометрия часто выполняется путем отслеживания положения относительно твердых объектов, то есть Объектов, которые не двигаются и не изменяются. Такое отслеживание может быть полезно для мобильных роботов в суровых условиях. Например, после стихийного бедствия [3].

2.2 Как сейчас проводится одометрия

В настоящее время одометрия основана на изображениях, а также на использовании лазерных сканеров и аналогичных устройств для измерения расстояния. На основе изображений, полу-

ченных в каждый момент времени t, соответствующее программное обеспечение для обработки изображений определяет местоположение объекта и находит ключевые точки этого объекта [4]. Местоположение этих ключевых точек в момент t затем характеризуется пространственными координатами в специальной системе координат кадра, связанной, например, с текущим местонахождение беспилотного объекта.

Процедура повторяется в следующий момент времени $t' = t + \Delta t$, и различия между координатами соответствующих точек используются для определения поворота и сдвига, связывающего системы координат в моменты времени t и t'.

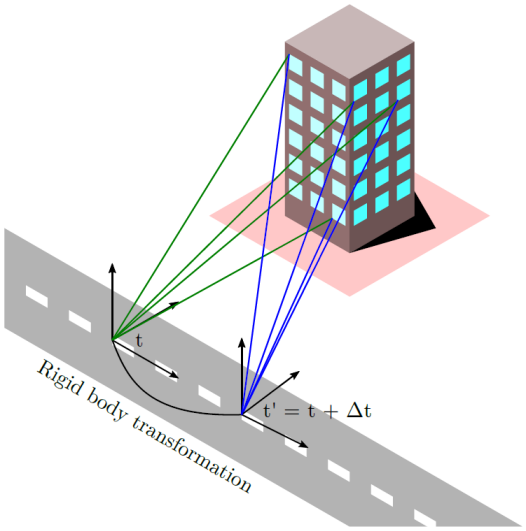


Рис. 4: Общая идея одометрии: соответствующие трехмерные ключевые точки на жестких объектах используется, чтобы найти преобразование твердого тела между двумя последовательными точками.

2.3 Интервальная неопределенность

Измерения никогда не бывают абсолютно точными. Результаты несколько отличаются от фактических значений соответствующих физических величин. В частности, пространственные координаты известны с некоторой неопределенностью. Кроме того, такие временные параметры, как несовершенная синхронизация часов датчика могут приводить к неопределенности пространственных координат [6–8]. Во многих таких ситуациях не известно точных вероятностей различных значений ошибки оценки. Все, что известно, это верхние границы соответствующих ошибок оценки. В этом случае, если оценочное значение координаты равно $\widetilde{x}$, а верхняя граница ошибки оценки равна Δ, то набор возможных значений x представляет собой интервал $x = [\underline{x}; \overline{x}] = [\widetilde{x} - \Delta; \widetilde{x} + \Delta]$. По этой причине этот тип неопределенности известен как интервальная неопределенность.

Неопределенность нужно учитывать при обработке данных. Из-за этой неопределенности результаты обработки данных также известны с неопределенностью. Например, нужно представить не только числовые оценки поворота и сдвига между двумя кадрами, а также необходимо описать насколько точны эти оценки.

Для решения задачи визуальной одометрии был выбран алгоритм ORB-SLAM2.

3 Алгоритм ORB-SLAM2

Метод одновременной локализации и построения карты (Simultaneous Localization and Mapping - SLAM) создает карту неизвестной среды и локализуют объект на карте в режиме реального времени. Среди различных датчиков используют камеру. С помощью нее можно собрать необходимое количество информации об окружающей среде, что позволяет надежно распознать траекторию передвижения и решить поставленную задачу. Поэтому в настоящее время большой интерес вызывают методы Visual SLAM, в которых основным датчиком является камера.

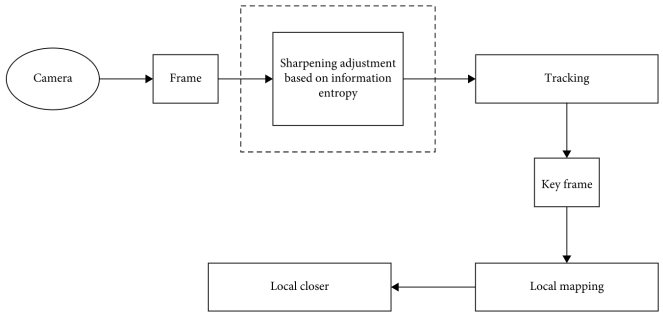


Рис. 5: Схема всех частей алгоритма и их взаимодействия [10]

В данной работе был рассмотрен алгоритм ORB-SLAM2 (Oriented FAST and Rotated BRIEF - ORB) [10]. Этот алгоритм основан на ORB-SLAM [11] и состоит из трех параллельных частей представленных на рис. 5:

- отслеживание для локализации камеры с каждым кадром путем поиска совпадений функций с локальной картой и минимизация ошибки перепроецирования с применением BA (Bundle adjustment --- BA)
- локальное сопоставление для управления локальной картой
- замыкание цикла для обнаружения больших циклов и исправления накопленного отклонения путем выполнения оптимизации графа позиций

3.1 Модуль распознавания места на основе DBoW2

В систему ORB-SLAM2 встроен модуль распознавания места на основе DBoW2 [12] для перенастройки, в случае сбоя отслеживания (например, окклюзии) или для повторной инициализации в уже отображенной сцене, а также для обнаружения петель. Система поддерживает видимый граф [13], который связывает любые два ключевых кадра, находя общие точки, и минимальное остовное дерево, соединяющее все ключевые кадры. Эти структуры графа позволяют извлекать локальные окна ключевых кадров, так что отслеживание и локальное сопоставление работают локально, позволяя работать на больших окружениях, и служат структурой для оптимизации графа позиций, выполняемой при закрытии цикла.

Извлечение локальных функций (ключевых точек и их векторов-дескрипторов) очень дорого выполнять с точки зрения времени вычислений при сравнении изображений. Это часто является узким местом, когда подобные методы применяются в реальном времени. Чтобы преодолеть эту проблему в алгоритме используются ключевые точки FAST [14] и дескрипторы BRIEF [15].

Опишем сущности, используемые в DBoW2.

3.1.1 Бинарные признаки

Ключевые точки FAST — это угловые точки, которые находятся сравнением интенсивности серого некоторых пикселей в круге Брезенхема с радиусом 3. Поскольку проверяются только несколько пикселей, эти точки очень быстро становятся успешными для приложений реального времени. Для каждой ключевой точки FAST мы рисуем квадратную область вокруг них и вычисляем BRIEF-дескриптор.

Дескриптор BRIEF фрагмента изображения — это двоичный вектор, где каждый бит является результатом сравнения интенсивности между двумя пикселями фрагмента. Патчи предварительно сглаживаются гауссовым ядром для уменьшения шума. Зная заранее размер фрагмента S_b, пары пикселей для тестирования выбираются случайным образом в автономном режиме. Помимо S_b, мы должны установить параметр L_b: количество выполняемых тестов (т.е.длина дескриптора). Для точки p на изображении ее краткий дескрипторный вектор $B(p)$ задается следующим образом:

$$B^i(p) = \begin{cases} 1, & \text{если } I(p+a_i) < I(p+b_i), \\ 0, & \text{иначе,} \end{cases} \quad \forall i \in \{\, 1, 2, \ldots, L_b \,\} \tag{3.1}$$

где $B^i(p)$ - i-ый бит дескриптора, $I(\cdot)$ — интенсивность пикселя в сглаженном изображении, a_i и b_i — двумерное смещение i-й тестовой точки относительно центра патча со значением $[-\frac{S_b}{2} \cdots \frac{S_b}{2}] \times$

$[-\frac{S_b}{2} \cdots \frac{S_b}{2}]$, выбранный заранее случайным образом. Этот дескриптор не требует обучения, просто автономный этап для выбора случайных точек, который практически не требует времени.

Каждую координату j этих пар выбираем из выборки распределений $a_i^j \sim N(0, \frac{1}{25}S_b^2)$ и $b_i^j \sim N(a_i^j, \frac{4}{625}S_b^2)$. Для длины дескриптора и размера патча выбираем $L_b = 256$ и $S_b = 48$. Поскольку один из этих дескрипторов является просто вектором битов, измерение расстояния между двумя векторами может быть выполнено путем подсчета количества различных битов между ними (расстояние Хэмминга), которое реализуется с помощью операции `xor`. В данном случае это больше подходит, чем вычисление евклидова расстояния, как это обычно делается с дескрипторами `SIFT` или `SURF`, состоящими из значений с плавающей запятой.

Масштабно-инвариантная трансформация признаков (Scale-invariant feature transform — `SIFT`) — это алгоритм распознавания признаков для выявления и описания локальных признаков в изображениях.

Сначала в `SIFT` извлекаются ключевые точки объектов из набора контрольных изображений и запоминаются в базе данных. Объект распознаётся в новом изображении путём сравнения каждого признака из нового изображения с признаками из базы данных и нахождения признаков-кандидатов на основе евклидова расстояния между векторами признаков. Из полного набора соответствий в новом изображении отбираются поднаборы ключевых точек, которые наиболее хорошо согласуются с объектом по его местоположению, масштабу и ориентации. Определение подходящих блоков признаков осуществляется быстро с помощью эффективной реализации хеш-таблицы обобщённого преобразования Хафа. Каждый блок из 3 или более признаков, согласующийся с объектом и его положением, подлежит дальнейшей подробной проверке соответствия модели, а резко отклоняющиеся блоки отбрасываются. Наконец, вычисляется вероятность, что определённый набор признаков говорит о присутствии объекта, что даёт информацию о точности совпадения и числе возможных промахов. Объекты, которые проходят все эти тесты, могут считаться правильными с высокой степенью уверенности.

`SIFT` может надёжно выделить объекты даже при наличии шума и частичном перекрытии, поскольку описатель признака `SIFT` инвариантен пропорциональному масштабированию, ориентации, изменению освещённости и частично инвариантен аффинным искажениям.

SURF (Speeded up robust features - `SURF`) — это локальный детектор и дескриптор признаков. Его можно использовать для таких задач, как распознавание объектов, регистрация изображений, классификация или 3D-реконструкция. Он частично основан на дескрипторе масштабно-инвариантного преобразования функций (`SIFT`). Стандартная версия `SURF` в несколько раз быстрее, чем `SIFT`, и, по утверждениям ее авторов, более устойчива к различным преобразованиям изображений, чем `SIFT`.

Для обнаружения точек интереса SURF использует целочисленное приближение детерминанта гессенского детектора капель, который может быть вычислен с помощью 3 целочисленных операций с использованием предварительно вычисленного интегрального изображения. Его дескриптор функции основан на сумме отклика вейвлета Хаара вокруг интересующей точки. Их также можно вычислить с помощью интегрального изображения. Неформально капля - это область изображения, в которой некоторые свойства постоянны или приблизительно постоянны; все точки в большом двоичном объекте можно в некотором смысле считать похожими друг на друга. Самый распространенный метод обнаружения больших двоичных объектов - это свертка.

3.1.2 База данных изображений

Для обнаружения повторно посещенных мест используется база данных изображений, состоящая из иерархического набора слов и прямых и обратных индексов, как показано на рис. 6.

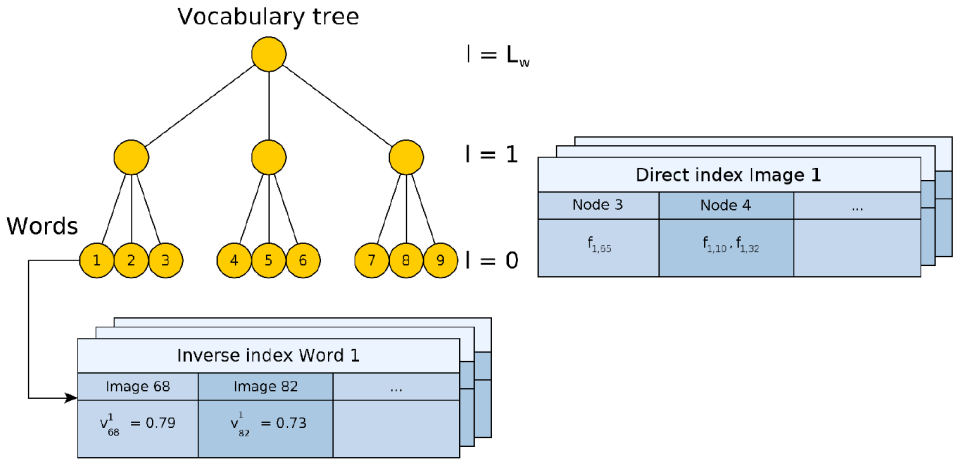


Рис. 6: Пример словарного дерева и прямых и обратных индексов, составляющих базу данных изображений. Словарные слова — это листовые узлы дерева. Обратный индекс хранит вес слов в изображениях, на которых они появляются. В прямом указателе хранятся особенности изображений и их связанные узлы на определенном уровне словарного дерева.

Пакет слов — это метод, который использует визуальный словарь для преобразования изображения в разреженный числовой вектор, позволяющий управлять большими наборами изображений. Визуальный словарь создается в автономном режиме путем дискретизации пространства дескриптора на W визуальных слов. В отличие от других функций, таких как SIFT или SURF, мы дискретизируем пространство двоичных дескрипторов, создавая более компактный словарь.

В случае иерархического набора слов словарь структурирован в виде дерева. Чтобы создать его, необходимо извлечь набор функций из некоторых обучающих изображений, независимо от тех, которые обрабатываются позже. Извлеченные дескрипторы сначала дискретизируются в k_w

двоичных кластеров путем выполнения кластеризации `k-median` с заполнением `k-means++`. Медианы, которые приводят к недвоичному значению, усекаются до 0. Эти кластеры образуют первый уровень узлов в дереве словаря. Последующие уровни создаются путем повторения этой операции с дескрипторами, связанными с каждым узлом, до L_w раз.

В итоге получается дерево с W листами, которые являются словами словаря. Каждому слову присваивается вес в соответствии с его релевантностью в обучающем корпусе, уменьшая вес тех слов, которые очень часты и, следовательно, менее различимы. Для этого используется термин частота — обратная частота документа (`tf-idf`). Затем, чтобы преобразовать изображение I_t, снятое в момент времени t, в вектор набора слов $v_t \in R^W$, двоичные дескрипторы его характеристик проходят дерево от корня к листьям, выбирая на каждом уровне промежуточные узлы, которые минимизировать расстояние Хэмминга.

Чтобы измерить сходство между двумя векторами набора слов v_1 и v_2, вычисляется L_1-оценка $s(v_1, v_2)$, значение которой лежит в $[0 \dots 1]$:

$$s(v_1, v_2) = 1 - \frac{1}{2} \left| \frac{v_1}{|v_1|} - \frac{v_2}{|v_2|} \right| \tag{3.2}$$

Наряду с пакетом слов поддерживается и **обратный индекс**. Эта структура хранит для каждого слова w_i в словаре список изображений I_t, где оно присутствует. Это очень полезно при запросе к базе данных, поскольку позволяет выполнять сравнения только с теми изображениями, которые имеют какое-то общее слово с запрашиваемым изображением. Увеличиваем обратный индекс для хранения пар $< I_t, v_i^t >$, чтобы быстро получить доступ к весу слова в изображении. Обратный индекс обновляется, когда новое изображение добавляется в базу данных, и доступ к нему осуществляется, когда в базе данных выполняется поиск некоторого изображения.

Эти две структуры часто являются единственными, которые используются в подходе пакета слов для поиска изображений. Однако в качестве новшества в этом общем подходе также используется **прямой индекс** для удобного хранения характеристик каждого изображения. Узлы словаря разделяются в соответствии с их уровнем l в дереве, начиная с листьев, с уровня $l = 0$, и заканчивая корнем $l = L_w$. Для каждого изображения I_t сохраняются в прямом индексе узлы на уровне l, которые являются предками слов, присутствующих в I_t, а также список локальных функций f_{t_j}, связанных с каждым узлом. Пользуемся преимуществом прямого индекса и дерева набора слов, чтобы использовать их как средство аппроксимации ближайших соседей в пространстве дескрипторов `BRIEF`. Прямой индекс позволяет ускорить геометрическую проверку, вычисляя соответствия только между теми характеристиками, которые принадлежат одним и тем же словам, или словам с общими предками на уровне l. Прямой индекс обновляется, когда новое изображение добавляется в базу данных, и доступ к нему осуществляется, когда соответствие кандидата получено

и необходима геометрическая проверка.

3.1.3 Алгоритм обнаружения петель

Для обнаружения замыканий цикла используется метод, который состоит из четырех этапов: запрос к базе данных, группировка патчей, временная согласованность, эффективная геометрическая согласованность.

Используем базу данных изображений для хранения и извлечения изображений, подобных любому заданному. При получении последнего изображения оно преобразуется в вектор набора слов v_t. В базе данных выполняется поиск v_t, в результате получается список подходящих кандидатов

$$< v_t, v_{t1} >, < v_t, v_{t2} >, \ldots,$$

связанные с их оценками $s(v_t, v_{t_j})$. Диапазон этих оценок очень зависит от изображения в запросе и распределения содержащихся в нем слов. Затем нормализуем эти оценки с наилучшей оценкой, которую ожидается получить в этой последовательности для вектора v_t, получая нормализованную оценку сходства η [6]:

$$\eta(v_t, v_{t_j}) = \frac{s(v_t, v_{t_j})}{s(v_t, v_{t-\Delta t})} \tag{3.3}$$

Здесь аппроксимируется ожидаемая оценка v_t с помощью $s(v_t, v_{t-\Delta t})$, где $v_{t-\Delta t}$ — вектор пакета слов предыдущего изображения.

Те случаи, когда $s(v_t, v_{t-\Delta t})$ мало (например, когда робот поворачивается), могут ошибочно привести к высоким значениям. Таким образом, пропускаются изображения, которые не достигают минимума $s(v_t, v_{t-\Delta t})$ или необходимого количества функций. Этот минимальный балл зависит от количества изображений, которые можно использовать для обнаружения петель, и правильности полученного результата. Используется небольшое значение, чтобы не допустить отбрасывания действительных изображений. Затем отклоняются те совпадения, чьи $\eta(v_t, v_{t_j})$ не достигают минимального порога, обозначенного α.

Чтобы изображения, близкие по времени, не конкурировали между собой при запросе базы данных, они группируются в **острова** (остров — массив схожих по признакам изображений, близких по времени) и рассматриваются как одно совпадение. Используем обозначение T_i для обозначения интервала, состоящего из временных меток $t_{n_i}, \ldots, t_{m_i}$ и V_{T_i} для острова, который группирует совпадения с записями $v_{t_{n_i}}, \ldots, v_{t_{m_i}}$. Следовательно, несколько совпадений $< v_t, v_{t_{n_i}} >, \ldots, <$

$v_t, v_{t_{m_i}} >$ преобразуются в одно совпадение $< v_t, V_{T_i} >$, если промежутки между последовательными отметками времени в $t_{n_i}, \ldots, t_{m_i}$ малы. Острова также оцениваются по шкале H:

$$H(v_t, V_{T_i}) = \sum_{j=n_i}^{m_i} \eta(v_t, v_{t_j}) \tag{3.4}$$

Остров с наибольшим количеством баллов выбирается как соответствующая группа и переходит к шагу временной согласованности. Помимо предотвращения конфликтов между последовательными изображениями, острова могут помочь установить правильные совпадения. Если I_t и $I_{t'}$ представляют собой замыкание цикла, очень вероятно, что оно будет аналогично $I_{t'\pm\Delta t}$, $I_{t'\pm\Delta t}$,..., производящие длинные острова. Поскольку H вычисляется как сумму баллов, показатель H также способствует совпадениям с длинными островами.

После получения наиболее подходящего острова $V_{T'}$ проверяем его временную согласованность с предыдущими запросами. Соответствие $< v_t, V_{T'} >$ должно соответствовать k предыдущим совпадениям $< v_{t-\Delta t}, V_{T_1} >, \ldots, < v_{t-k\Delta t}, V_{T_k} >$, такие что интервалы T_j и T_{j+1} близки к перекрытию. Если остров проходит временное ограничение, сохраняем только совпадение $< v_t, v_{t'} >$ для $t' \in T'$, которое максимизирует счет, и считаем его кандидатом на закрытие цикла, который должен быть окончательно принят геометрическим этапом проверки.

Геометрическая проверка применяется между любыми парами изображений кандидата на замыкание петли. Эта проверка заключается в нахождении с помощью `RANSAC` (Random Sample Consensus — оценка параметров модели на основе случайных выборок) фундаментальной матрицы между I_t и $I_{t'}$. Чтобы вычислить эти соответствия, необходимо сравнить локальные характеристики изображения запроса с соответствующими характеристиками.

Алгоритм RANSAC:

1. Входные параметры
   (a) Набор исходных данных X
   (b) Функция M, позволяющая вычислить параметры θ модели P по набору данных из n точек
   (c) Функция оценки E соответствия точек полученной модели
   (d) Порог t для функции оценки
   (e) Количество итераций метода k

   Весь алгоритм состоит из одного цикла, каждую итерацию которого можно разделить на два этапа.

2. Первый этап — выбор точек и подсчёт модели

   (a) Из множества исходных точек X случайным образом выбираются n различных точек

   (b) На основе выбранных точек вычисляются параметры θ модели P с помощью функции M, построенную модель принято называть гипотезой

3. Второй этап — проверка гипотезы

   (a) Для каждой точки проверяется её соответствие данной гипотезе с помощью функции оценки E и порога t

   (b) Каждая точка помечается инлаером или выбросом

   (c) После проверки всех точек, проверяется, является ли гипотеза лучшей на данный момент, и если является, то она замещает предыдущую лучшую гипотезу

   В конце работы цикла оставляется последняя лучшая гипотеза.

4. Результат

   (a) Параметры θ модели P

   (b) Точки исходных данных, помеченные инлаерами или выбросами

3.2 Построение 3-D модели по 2-D изображениям

Групповую настройку (Bundle Adjustment — BA) можно определить как проблему одновременного уточнения трехмерных координат, описывающих геометрию сцены, уточнения параметров относительного движения и оптических характеристик камеры (камер), используемой для получения изображений в соответствии с критерием оптимальности, включающим соответствующие проекции изображения всех точек. В режиме реального времени это возможно сделать с помощью `BA`:

1. Наблюдения за функциями сцены (точки карты) среди подмножества выбранных ключевых кадров

2. Сложность растет с увеличением количества ключевых кадров, поэтому их выбор должен избегать ненужной избыточности

3. Сильная сетевая конфигурация ключевых кадров и точек для получения точных результатов, то есть хорошо распределенный набор ключевых кадров, наблюдающих точки с большим количеством совпадений замыкания цикла

4. Первоначальная оценка положений ключевых кадров и местоположения точек для нелинейной оптимизации

5. Возможность выполнять быструю глобальную оптимизацию для закрытия циклов в реальном времени

3.3 Описание алгоритма ORB-SLAM2

3.3.1 Моно-, стерео-, RGB изображения

Метод ORB-SLAM2, основанный на обработке изображений по схожим признакам, предварительно обрабатывает входные изображения для извлечения признаков и ключевых точек, как показано на рис. 7.

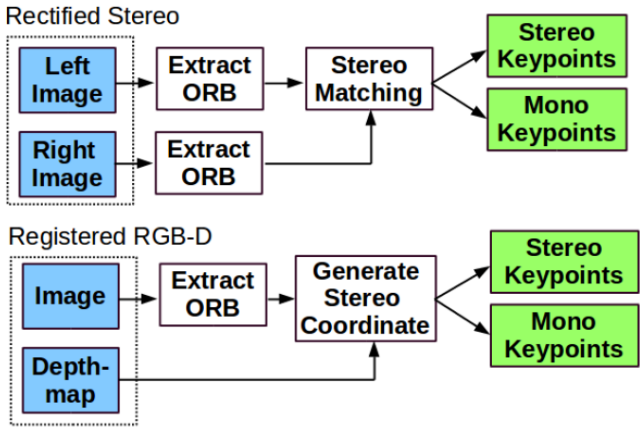


Рис. 7: Схема препроцессинга входных данных.

Затем входные изображения отбрасываются. Все системные операции основываются на функциях. Система не зависит от того, является ли датчик моно, стерео или RGB. Система обрабатывает ключевые точки монокуляра и стерео, которые в дальнейшем классифицируются как близкие или дальние.

Стерео ключевые точки определяются тремя координатами $x_s = (u_L, v_L, u_R)$, где (u_L, v_L) — координаты на левом изображении, (u_R) — горизонтальная координата на правом изображении. Для стереокамер извлекаются признаки ORB из обоих изображений, и для каждого левого ORB производим поиск совпадений в правом изображении. Затем генерируется стерео ключевая точка с координатами левого ORB и горизонтальной координатой правого совпадения, которая субпиксельно уточняется корреляцией патчей. Для камер RGB извлекаются элементы ORB на изображении RGB и

для каждого объекта с координатами $(u_L; v_L)$ преобразуется его значение глубины d в виртуальную правую координату:

$$u_R = u_L - \frac{f_x b}{d} \tag{3.5}$$

где f_x — горизонтальное фокусное расстояние, а b - базовая линия между проекцией структурированного света и инфракрасной камерой.

Неопределенность датчика глубины представлена неопределенностью виртуальной правой координаты. Таким образом, функции стереовхода и входа RGB одинаково обрабатываются остальной частью системы.

Стерео ключевая точка классифицируется как близкая, если связанная с ней глубина менее чем в 40 раз превышает базовую линию стерео/RGB, в противном случае она классифицируется как далекая. Близкие ключевые точки можно безопасно триангулировать из одного кадра, поскольку глубина точно оценивается и предоставляет информацию о масштабе, перемещении и вращении. С другой стороны, дальние точки предоставляют точную информацию о вращении, но более слабую информацию о масштабе и перемещении. Поэтому триангулируем дальние точки, когда они поддерживаются несколькими видами.

Ключевые точки монокуляра определяются двумя координатами $x_m = (u_L; v_L)$ на левом изображении и соответствуют всем тем ORB, для которых не удалось найти стерео совпадение или которые имеют недопустимое значение глубины в случае RGB. Эти точки триангулируются только из нескольких видов и не предоставляют информацию о масштабе, но участвуют в оценке вращения и перемещения.

3.3.2 Загрузка системы

Одним из основных преимуществ использования стереокамер или камер RGB является то, что, имея информацию о глубине только из одного кадра, не нужна конкретная структура для инициализации движения, как в случае с монокуляром. При запуске системы создается ключевой кадр с первым кадром, устанавливаем его позицию в начало координат и создаем начальную карту из всех ключевых точек.

3.3.3 Регулировка связки с монокуляром и стерео

Ограничения системы выполняет BA для оптимизации положения камеры в потоке отслеживания (BA только для движения — motion-only BA), для оптимизации локального окна ключевых кадров и точек в локальном потоке сопоставления (локальный BA) и после закрытия цикла для оптимизации всех ключевых кадров и баллов (полная BA). Для этого Используется метод Левенберга – Марквардта, реализованный в g2o [16].

Motion-only BA оптимизирует ориентацию камеры $R \in SO(3)$ и положение $t \in R^3$, минимизируя ошибку перепроецирования между совпадающими трехмерными точками $X^i \in R^3$ в мировых координатах и ключевыми точками $x^i_(\cdot)$), либо монокуляр $x^i_m \in R^2$ или стерео $x^i_s \in R^3$, где $i \in X$ в наборе всех совпадений:

$$R, t = \arg\min_{R,t} \sum_{i \in X} p\left(\|x^i_{(\cdot)} - \pi_{(\cdot)}(RX^i + t)\|^2_\Sigma\right) \tag{3.6}$$

где p — робастная функция стоимости Хубера, Σ — ковариационная матрица, связанная с масштабом ключевой точки. Функции проецирования $\pi_{(\cdot)}$, монокуляр π_m и выпрямленное стерео π_s, определены следующим образом:

$$\pi_m \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \end{bmatrix} \tag{3.7}$$

$$\pi_s \left(\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \right) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \\ f_x \frac{X-b}{Z} + c_x \end{bmatrix} \tag{3.8}$$

где $(f_x; f_y)$ — фокусное расстояние, $(c_x; c_y)$ — главная точка и b базовая линия. Все это известно из калибровки.

Локальный BA **(Local** BA**)** оптимизирует набор скрытых ключевых кадров K_L и все точки, видимые в этих ключевых кадрах P_L. Все другие ключевые кадры K_F, не входящие в K_L, точки наблюдения в P_L вносят вклад в функцию стоимости, но остаются фиксированными при оптимизации. Определяя X_k как набор совпадений между точками в P_L и ключевыми точками в ключевом кадре k, проблема оптимизации заключается в следующем:

$$\left\{X^i, R_l, t_l | i \in P_L, l \in K_L\right\} = \underset{X^i, R_l, t_l}{\arg\min}(\sum_{k \in K_L \cup K_F} \sum_{j \in X_k} p(E_{kj})) \tag{3.9}$$

Полный BA **(Full** BA**)** — это частный случай локального BA, где все ключевые кадры и точки на карте оптимизированы, за исключением исходного ключевого кадра, который фиксируется, чтобы исключить свободу измерения.

3.3.4 Замыкание цикла и полный BA

Замыкание цикла выполняется в два этапа: во-первых, цикл должен быть обнаружен и подтвержден, а во-вторых, цикл корректируется, оптимизируя граф позиций. В отличие от монокуляра ORBSLAM, где может происходить дрейф масштаба, информация стерео/глубины делает масштаб наблюдаемым, а геометрическая проверка и оптимизация графа позиций больше не требуют рассмотрения дрейфа масштаба и основаны на преобразованиях твердого тела.

В ORB-SLAM2 выполняется полная оптимизация BA после графа позиций для достижения оптимального решения. Эта оптимизация может быть очень дорогостоящей, поэтому выполняется она в отдельном потоке, позволяя системе продолжать создавать карту и обнаруживать петли. Однако при этом возникает проблема объединения выходных данных настройки пакета с текущим состоянием карты. Если во время оптимизации обнаруживается новый цикл, прерываем оптимизацию и продолжаем закрывать цикл, что снова запускает полную оптимизацию BA. Когда полный BA завершается, нужно объединить обновленное подмножество ключевых кадров и точек, оптимизированных полным BA, с необновленными ключевыми кадрами и точками, которые были вставлены во время выполнения оптимизации. Это делается путем распространения исправления обновленных ключевых кадров (т. е. преобразования из неоптимизированной позиции в оптимизированную) на необновленные ключевые кадры через связующее дерево. Необновленные точки преобразуются в соответствии с поправкой, примененной к их опорному ключевому кадру.

3.3.5 Вставка ключевого кадра

ORB-SLAM2 следует аналогично ORB-SLAM, по очень частой вставке ключевых кадров и последующему удалению избыточных. Различие между близкими и дальними стерео точками позволяет ввести новое условие для вставки ключевых кадров, которое может быть критичным в сложных условиях, когда большая часть сцены находится далеко от стереодатчика, как показано на рис. 8. Зеленые точки имеют глубину менее чем в 40 раз превышающую базовую стерео линию, а синие точки находятся дальше. В таких последовательностях важно вставлять ключевые кадры достаточ-

но часто, чтобы количество близких точек позволяло точно оценить трансляцию. Далекие точки помогают ориентироваться в оценке, но не дают информации для перевода и масштаба. В такой среде нужно иметь достаточное количество близких точек для точной оценки трансляции, поэтому, если количество отслеживаемых близких точек упадет ниже τ_t и кадр может создать как минимум τ_c новых близких стерео точек, система вставит новый ключевой кадр. Эмпирически примем, что $\tau_t = 100$ и $\tau_c = 70$ хорошо работают.

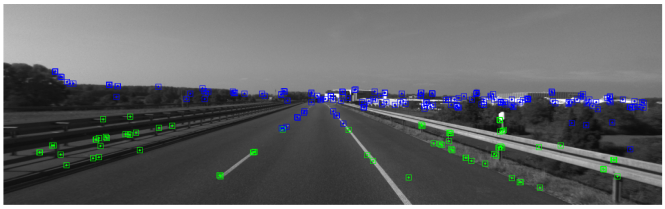


Рис. 8: Точки отслеживания в KITTI 01.

4 Интервальная регрессия

Рассмотрим задачу восстановления функциональных зависимостей по данным, полученным с карт перемещения объектов на поворотах [18].

4.1 Постановка задачи

Пусть y — функция заданного вида:

$$y = f(x, \beta), \tag{4.1}$$

где $x = (x_1, \ldots, x_m)$ — вектор независимых переменных, $\beta = (\beta_1, \ldots, \beta_l)$ — вектор параметров функции. Входным набором данных являются значения переменных x, y. Необходимо вычислить значение параметров $\beta_1, \ldots, \beta_l$, которые соответствуют конкретной функции f из семейства (4.1).

4.2 Линейный случай

Рассмотрим линейный случай [19], где функция (4.1) - линейная функциональная зависимость вида

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_m \cdot x_m \tag{4.2}$$

Результаты измерений неточны. Тогда результат i-го измерения являются интервалы $x_1^{(i)}, x_2^{(i)}, \ldots, x_m^{(i)}, y^{(i)}$. Их точное решение лежит в соответствующих интервалах. Истинное значение x_1 лежит в интервале $x_1^{(i)}$, а значение y находится в интервале $y^{(i)}$. Есть n измерений, таких что индекс i может принимать значения из множества натуральных чисел $\{1, 2, \ldots, n\}$.

Обозначим номер измерения i нижним индексом, который поставим первым при обозначении входов. После этих преобразований полный набор данных будет иметь вид:

$$\begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1m} & y_1 \\ x_{21} & x_{22} & \ldots & x_{2m} & y_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nm} & y_n. \end{pmatrix} \tag{4.3}$$

Нужно найти коэффициенты β_j, $j = 0, 1, \ldots, m$, для которых линейная функция (4.2) наилучшим образом приближала бы интервальные данные измерений (4.3). Интервалы $x_{i1}, x_{i2}, \ldots, x_{im}, y_i$ называются интервалами неопределенности i-го измерения. Также потребуется характеризовать целиком всё множество, ограничиваемое в многомерном пространстве R_{m+1} этими интервалами по отдельным координатным осям.

Брусом неопределённости i-го измерения зависимости называется интервальный вектор-брус $(x_{i1}, x_{i2}, \ldots, x_{im}, y_i) \in R^{m+1}, i = 1, 2, \ldots, n$. Каждый брус неопределённости измерения зависимости является прямым декартовым произведением интервалов неопределённости независимых переменных и зависимой переменной.

Функцию (4.1) или (4.2) можно считать точным решением задачи восстановления зависимости, если её график проходит через все брусы неопределённости данных. В случае точечных данных эта идеальная ситуация почти никогда не реализуется и неустойчива к малым возмущениям в данных. Но в случае данных с существенной интервальной неопределённостью прохождение графика функции через брусы данных (4.3) может реализовываться, и оно устойчиво к возмущениям в данных.

При точечных данных решение задачи восстановления зависимостей получается по следующей общей схеме. Подставляем данные в формулу (4.2) и получаем для каждого отдельного измерения одно уравнение. В результате этой процедуры получаем система уравнений, решив которую, найдём параметры зависимости. В интервальном случае получим уже интервальную систему уравнений. Её решением будет вектор оценки параметров восстанавливаемой зависимости (4.2). Информационное множество задачи получается при этом как множество решений этой интервальной системы уравнений, построенной на основе формулы (4.2) и данных (4.3).

Определение параметров функциональной зависимости производится для того, чтобы затем решение использовать для предсказания значений зависимости в других точках её области определения. Такое предсказание будет осуществляться с некоторой погрешностью, вызванной неопределённостями данных, неоднозначностью самой процедуры восстановления. Эту неопределённость предсказания также необходимо знать и учитывать в нашей деятельности.

Пусть в задаче восстановления зависимостей информационное множество I параметров зависимостей $y = f(x, \beta)$, совместных с данными, является непустым. Коридором совместных зависимостей рассматриваемой задачи называется многозначное отображение T, сопоставляющее каждому значению аргумента x множество

$$T(x) = \bigcup_{\beta \in I} f(x, \beta) \tag{4.4}$$

Значение $T(x)$ коридора совместных зависимостей при каком-то определённом аргументе x («сечение коридора») — это множество $\bigcup_{\beta \in I} f(x, \beta)$, образованное всевозможными значениями, которые принимают на этом аргументе функциональные зависимости, совместные с интервальными данными измерений.

Подставляем в зависимость (4.2) входные параметры $x_1, x_2, \ldots, x_m$ в i-ом измерении при включении полученного значения в интервал y_i, получаем:

$$\beta_0 + \beta_1 \cdot x_{i1} + \beta_2 \cdot x_{i2} + \cdots + \beta_m \cdot x_{im} \in y_i, i = 1, 2, \ldots, n. \tag{4.5}$$

Тогда интервальная система линейных алгебраических уравнений выглядит следующим образом:

$$\begin{cases} \beta_0 + \beta_1 \cdot x_{11} + \beta_2 \cdot x_{12} + \cdots + \beta_m \cdot x_{1m} = y_1 \\ \beta_0 + \beta_1 \cdot x_{21} + \beta_2 \cdot x_{22} + \cdots + \beta_m \cdot x_{2m} = y_2 \\ \vdots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \quad \vdots \\ \beta_0 + \beta_1 \cdot x_{n1} + \beta_2 \cdot x_{n2} + \cdots + \beta_m \cdot x_{nm} = y_n \end{cases} \tag{4.6}$$

у которой интервальность присутствует только в правой части. С другой стороны, (4.5) равносильно

$$\begin{cases} \underline{y}_1 \le \beta_0 + \beta_1 \cdot x_{11} + \beta_2 \cdot x_{12} + \cdots + \beta_m \cdot x_{1m} \le \bar{y}_1 \\ \underline{y}_2 \le \beta_0 + \beta_1 \cdot x_{21} + \beta_2 \cdot x_{22} + \cdots + \beta_m \cdot x_{2m} \le \bar{y}_2 \\ \vdots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \quad \vdots \quad \vdots \\ \underline{y}_n \le \beta_0 + \beta_1 \cdot x_{n1} + \beta_2 \cdot x_{n2} + \cdots + \beta_m \cdot x_{nm} \le \bar{y}_n \end{cases} \tag{4.7}$$

Это система двусторонних линейных неравенств относительно неизвестных параметров $\beta_0, \beta_1, \beta_2, \ldots, \beta_m$, решив которую, можно найти искомую линейную зависимость. Множество решений системы неравенств (4.7) естественно считать информационным множеством параметров восстанавливаемой зависимости для рассматриваемого случая.

Для i-го двустороннего неравенства из системы (4.7) множество решений — это полоса в пространстве R^{m+1} параметров $(\beta_0, \beta_1, \beta_2, \ldots, \beta_m)$, ограниченная с двух сторон гиперплоскостями с уравнениями

$$\beta_0 + \beta_1 \cdot x_{i1} + \beta_2 \cdot x_{i2} + \cdots + \beta_m \cdot x_{im} = \underline{y}_i \tag{4.8}$$

$$\beta_0 + \beta_1 \cdot x_{i1} + \beta_2 \cdot x_{i2} + \cdots + \beta_m \cdot x_{im} = \bar{y}_i \tag{4.9}$$

Множество решений системы неравенств (4.7) является пересечением n штук таких полос, отвечающих отдельным измерениям. Можно рассматривать эти полосы как информационные множества отдельных измерений.

4.3 Квадратичный случай

В нашей задаче при определения траектории движения будут рассматриваться повороты, у которых форма движения - это кривые второго порядка, поэтому будем рассматривать функцию вида:

$$y = \beta_0 + \beta_1 \cdot x + \beta_2 \cdot x^2 \tag{4.10}$$

Тогда общий вид уравнения будет иметь вид:

$$y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \cdots + \beta_l \cdot x_l + \beta_{l+1} \cdot x_{l+1}^2 + \cdots + \beta_m \cdot x_m^2 \tag{4.11}$$

Тогда интервальная система линейных алгебраических уравнений выглядит следующим об-

разом:

$$\begin{cases} \beta_0 + \beta_1 \cdot x_{11} + \beta_2 \cdot x_{12} + \cdots + \beta_l \cdot x_{1l} + \beta_{l+1} \cdot x^2_{1(l+1)} + \cdots + \beta_m \cdot x^2_{1m} = y_1 \\ \beta_0 + \beta_1 \cdot x_{21} + \beta_2 \cdot x_{22} + \cdots + \beta_l \cdot x_{2l} + \beta_{l+1} \cdot x^2_{2(l+1)} + \cdots + \beta_m \cdot x^2_{2m} = y_2 \\ \vdots \quad \vdots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ \beta_0 + \beta_1 \cdot x_{n1} + \beta_2 \cdot x_{n2} + \cdots + \beta_l \cdot x_{nl} + \beta_{l+1} \cdot x^2_{n(l+1)} + \cdots + \beta_m \cdot x^2_{nm} = y_n \end{cases} \tag{4.12}$$

у которой интервальность присутствует только в правой части.

$$\begin{cases} \underline{y}_1 \leq \beta_0 + \beta_1 \cdot x_{11} + \beta_2 \cdot x_{12} + \cdots + \beta_l \cdot x_{1l} + \beta_{l+1} \cdot x^2_{1(l+1)} + \cdots + \beta_m \cdot x^2_{1m} \leq \bar{y}_1 \\ \underline{y}_2 \leq \beta_0 + \beta_1 \cdot x_{21} + \beta_2 \cdot x_{22} + \cdots + \beta_l \cdot x_{2l} + \beta_{l+1} \cdot x^2_{2(l+1)} + \cdots + \beta_m \cdot x^2_{2m} \leq \bar{y}_2 \\ \vdots \quad \vdots \quad \vdots \quad \vdots \quad \ddots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ \underline{y}_n \leq \beta_0 + \beta_1 \cdot x_{n1} + \beta_2 \cdot x_{n2} + \cdots + \beta_l \cdot x_{nl} + \beta_{l+1} \cdot x^2_{n(l+1)} + \cdots + \beta_m \cdot x^2_{nm} \leq \bar{y}_n \end{cases} \tag{4.13}$$

4.4 Метод центра неопределенности

В нашей задаче вектор β - неизвестные параметры, которые необходимо вычислить. И состоит этот вектор из трех элементов - $\beta_0, \beta_1, \beta_2$, так как функция (4.1) представлена следующим образом:

$$y(x) = \beta_0 + \beta_1 \cdot x + \beta_2 \cdot x^2 \tag{4.14}$$

Для набор входных параметров $(y_i, x_{1j}, x_{2j}, x_{nj})$ при $j = 1 \ldots N$ при зафиксированных значениях векторов $x_j = (x_{1j}, x_{2j}, x_{nj})$ удовлетворяет неравенству

$$y_j - \Delta_j \leq f(\beta, x_j) \leq y_j + \Delta_j, \quad j = 1 \ldots N \tag{4.15}$$

где $(y_j - \Delta_j), (y_j - \Delta_j)$ — нижняя и верхняя оценки значения выходной переменной y_j.

Для удобства отсортируем вектор входных параметров x по возрастанию и сместим значения так, чтобы $x_0 = 0$.

Вычислим нижнюю h_{Hi} и верхнюю h_{Bi} границы множества H_i для каждого x_i:

$$h_{Hi} = x_i - \Delta_{max}, \tag{4.16}$$

$$h_{Bi} = x_i + \Delta_{max} \tag{4.17}$$

После этого необходимо определить совместность системы. Вычислим вспомогательные величины для каждого измерения:

$$h_{min} = max_i(h_{Hi}), \tag{4.18}$$

$$h_{max} = min_i(h_{Bi}) \tag{4.19}$$

Сравним эти значения:

- Если $h_{min} > h_{max}$, то выборка несовместна и далее необходимо искать промахи
- Если $h_{min} \leq h_{max}$, то выборка совместна и значение интервала допустимых значений $I = [h_{min}, h_{max}]$

Далее нужно рассчитать оценку центрального действительного значения:

$$x_c = (h_{max} + h_{min})/2 \tag{4.20}$$

и максимальное действительное отклонение:

$$\Delta x = (h_{max} - h_{min})/2 \tag{4.21}$$

4.4.1 Вычисление коэффициента β_0

Рассмотрим тройку наблюдений y_i, y_j, y_k с упорядоченными значениями $x_i < x_j < x_k \quad \forall i = 0, N-3; j = i+1, N-2; k = j+1, N-1$. Для каждой тройки значений $i < j < k$ вычислим допустимые значения коэффициента β_0:

$$\beta_{0H0jk} = h_{H0}, \quad \text{если } i = 0 \quad \forall j = i+1, N-2; k = j+1, N-1 \tag{4.22}$$

$$\beta_{0Hjik} = \frac{x_i \cdot x_j \cdot x_k}{x_j - x_i} \cdot \left(\frac{h_{Hi}}{x_i \cdot (x_k - x_i)} - \frac{h_{Bj}}{x_j \cdot (x_k - x_j)} + \frac{h_{Hk} \cdot (x_j - x_i)}{x_k \cdot (x_k - x_i) \cdot (x_k - x_j)} \right), \quad \forall i = 1, N-3; j = i+1, N-2; k = j+1, N-1 \tag{4.23}$$

$$\beta_{0B0jk} = h_{B0}, \quad \text{если } i = 0 \quad \forall j = i+1, N-2; k = j+1, N-1 \tag{4.24}$$

$$\beta_{0Bijk} = \frac{x_i \cdot x_j \cdot x_k}{x_j - x_i} \cdot \left(\frac{h_{Bi}}{x_i \cdot (x_k - x_i)} - \frac{h_{Hj}}{x_j \cdot (x_k - x_j)} + \frac{h_{Bk} \cdot (x_j - x_i)}{x_k \cdot (x_k - x_i) \cdot (x_k - x_j)} \right), \quad \forall i = 1, N-3; j = i+1, N-2; k = j+1, N-1 \tag{4.25}$$

С помощью значений (4.23) и (4.25) вычисляем:

$$\beta_{0min} = max_{ijk}(\beta_{0Hijk}) \tag{4.26}$$

$$\beta_{0max} = min_{ijk}(\beta_{0Bijk}) \tag{4.27}$$

Сравниваем значения вычисленных величин и определяем совместность по (4.18), (4.19). Далее вычисляем значение коэффициента по формуле (4.20).

4.4.2 Вычисление коэффициента β_1

Рассмотрим тройку наблюдений y_i, y_j, y_k с упорядоченными значениями $x_i < x_j < x_k \quad \forall i = 0, N-3; j = i+1, N-2; k = j+1, N-1$. Для каждой тройки значений $i < j < k$ вычислим допустимые значения коэффициента β_1:

$$\beta_{1Hjik} = \frac{(x_i + x_k) \cdot (x_j + x_k)}{x_j - x_i} \left(-\frac{h_{Bi}}{(x_k^2 - x_i^2)} + \frac{h_{Hj}}{(x_k^2 - x_j^2)} - \frac{h_B k \cdot (x_j^2 - x_i^2)}{(x_k^2 - x_j^2) \cdot (x_k^2 - x_i^2)} \right),$$
$$\forall i = 0, N-3; j = i+1, N-2; k = j+1, N-1 \tag{4.28}$$

$$\beta_{1Bijk} = \frac{(x_i + x_k) \cdot (x_j + x_k)}{x_j - x_i} \left(-\frac{h_{Hi}}{(x_k^2 - x_i^2)} + \frac{h_{Bj}}{(x_k^2 - x_j^2)} - \frac{h_H k \cdot (x_j^2 - x_i^2)}{(x_k^2 - x_j^2) \cdot (x_k^2 - x_i^2)} \right),$$
$$\forall i = 0, N-3; j = i+1, N-2; k = j+1, N-1 \tag{4.29}$$

С помощью значений (4.28) и (4.29) вычисляем:

$$\beta_{1min} = max_{ijk}(\beta_{1Hijk}) \tag{4.30}$$

$$\beta_{1max} = min_{ijk}(\beta_{1Bijk}) \tag{4.31}$$

Сравниваем значения вычисленных величин и определяем совместность по (4.18), (4.19). Далее вычисляем значение коэффициента по формуле (4.20).

4.4.3 Вычисление коэффициента β_2

Рассмотрим тройку наблюдений y_i, y_j, y_k с упорядоченными значениями $x_i < x_j < x_k \quad \forall i = 1, N-3; j = i+1, N-2; k = j+1, N-1$. Для каждой тройки значений $i < j < k$ вычислим

допустимые значения коэффициента β_2:

$$\beta_{2Hjik} = \frac{\frac{h_{Hi}}{(x_k - x_i)} - \frac{h_{Bj}}{(x_k - x_j)} + \frac{h_{Hk} \cdot (x_j - x_i)}{(x_k - x_j) \cdot (x_k - x_i)}}{(x_j - x_i)},$$
$$\forall i = 0, N-3; j = i+1, N-2; k = j+1, N-1 \tag{4.32}$$

$$\beta_{2Bijk} = \frac{\frac{h_{Bi}}{(x_k - x_i)} - \frac{h_{Hj}}{(x_k - x_j)} + \frac{h_{Bk} \cdot (x_j - x_i)}{(x_k - x_j) \cdot (x_k - x_i)}}{(x_j - x_i)},$$
$$\forall i = 0, N-3; j = i+1, N-2; k = j+1, N-1 \tag{4.33}$$

С помощью значений (4.32) и (4.33) вычисляем:

$$\beta_{2min} = max_{ijk}(\beta_{2Hijk}) \tag{4.34}$$

$$\beta_{2max} = min_{ijk}(\beta_{2Bijk}) \tag{4.35}$$

Сравниваем значения вычисленных величин и определяем совместность по (4.18), (4.19). Далее вычисляем значение коэффициента по формуле (4.20).

4.5 Решение задачи о допусках методом распознающего функционала

Поставленную задачу можно решить методом распознающего функционала [17]. Представим систему 4.6 в матричном виде:

$$X \cdot \beta = y \tag{4.36}$$

Распознающий функционал - это функционал трех аргументов - вещественной $n-$мерной точки и двух интервальных $m \times n-$матрицы и $m-$вектора:

$$Tol(\beta, X, y) := \min_{i=1,\ldots,m} (Tol_i(\beta, X, y)) \tag{4.37}$$

где

$$Tol_i(\beta, X, y) := rad(y_i) - |mid(y_i) - X_{i,:}\beta| - \text{ это образующая,}$$

$$X_{i,:} - i\text{-ая строка матрицы } X, i = 1, \ldots, m.$$

5 Анализ результатов алгоритмов

5.1 Описание данных

В качестве исходных данных был рассмотрен набор изображений KITTI. Датасет для решения задачи визуальной одометрии состоит из 22 стерео последовательностей, которые сохранены в формате png без потери качества. 11 последовательностей (00-10) с наземными траекториями для обучения и 11 последовательностей (11-21) без достоверных данных для оценки. Примеры изображений данных приведены на рис. 9.

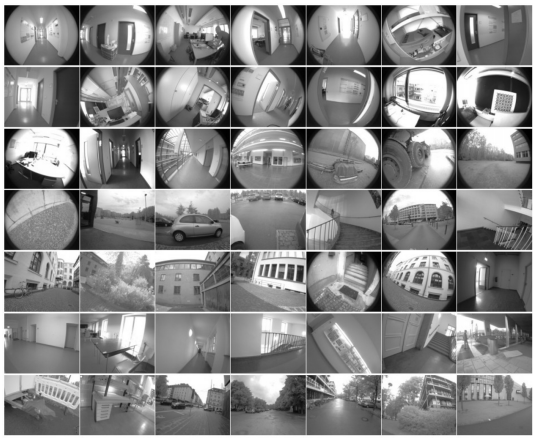


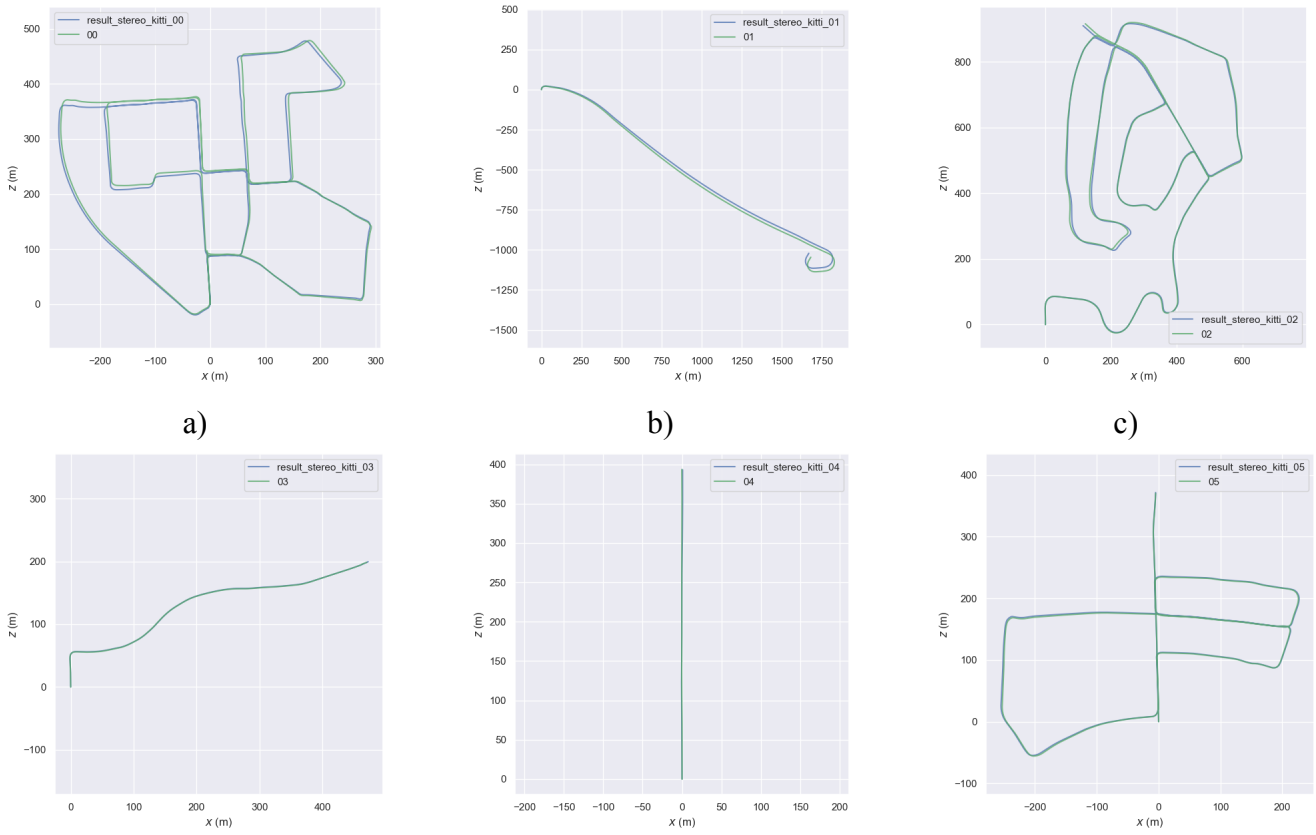
Рис. 9: Примеры изображений из датасета KITTI.

Кроме этого есть ограничение на используемые методы обработки, а именно алгоритм должен быть полностью автоматическим, то есть не должно быть ручной корректировки итоговых или промежуточных результатов.

Алгоритм запускался на ноутбуке с конфигурацией Intel Core i7-1065G7 16Гб оперативной памяти.

5.2 Результаты работы `ORB-SLAM2`

Для начала запустили алгоритм `ORB-SLAM2` на данных `KITTI`, а именно на видео 00-05. Результаты представлены на рис. 10.



a) b) c)

d) e) f)

Рис. 10: Карты, полученные после выполнения алгоритма ORB-SLAM2: a) 00, b) 01, c) 02, d) 03, e) 04, f) 05.

После запуска алгоритма получаем координаты объекта для каждого кадра исходного видео, также получаем координаты ключевых точек, представленные на рис. 11. Ключевые точки - это результат алгоритма поиска позиции объекта. Их количество заведомо меньше числа кадров на видео, потому что в алгоритме кадры группируются в острова и обработка происходит не всех кадров, однако их количества достаточно для корректной работы алгоритма ORB-SLAM2.

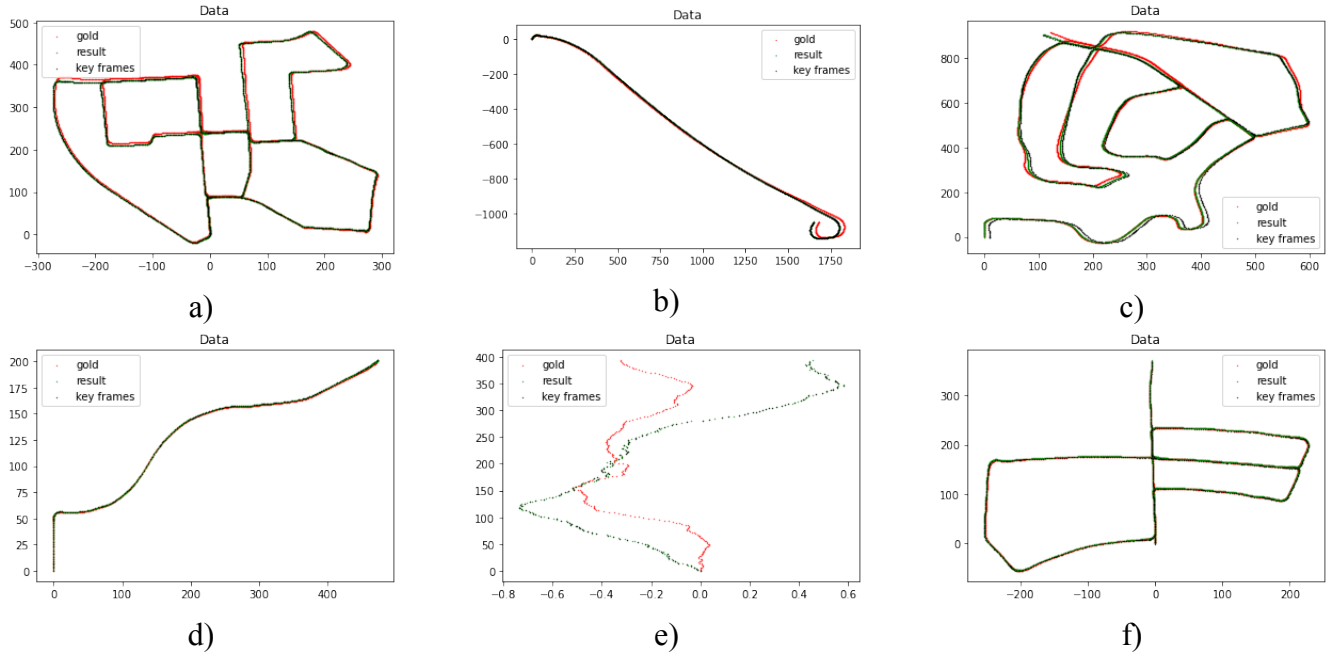


Рис. 11: Точки, полученные после выполнения алгоритма ORB-SLAM2: a) 00, b) 01, c) 02, d) 03, e) 04, f) 05.

Рассмотрим соотношение числа кадров на видео и числом ключевых кадров. Из таблицы (1) видно, что нет четкой зависимости между общим числом кадров и числом ключевых кадров, то есть для одного видео их число отличается в 3,2 раза, для другого видео в 1,05 раз. Чем больше количество кадров на видео, тем больше необходимо ключевых кадров обработать для получения траектории движения объекта. Однако, количество ключевых кадров заранее узнать нельзя.

Номер набора данных	Число изображений	Число ключевых кадров	Разница
00	4541	1388	3,27
01	1101	1044	1,05
02	4661	1734	2,68
03	801	223	3,59
04	271	154	1,76
05	2761	724	3,81

Таблица 1: Количество ключевых точек для обработки видео.

5.3 Поиск поворотов

Далее рассмотрим несколько карт из датасета. Выполним поиск поворотов и их локализацию, то есть из всей карты, всех точек траектории выберем только те, которые входят в рассматриваемый поворот. Выполним поиск 3 поворотов на карте. Результаты представлены на рисунках ниже.

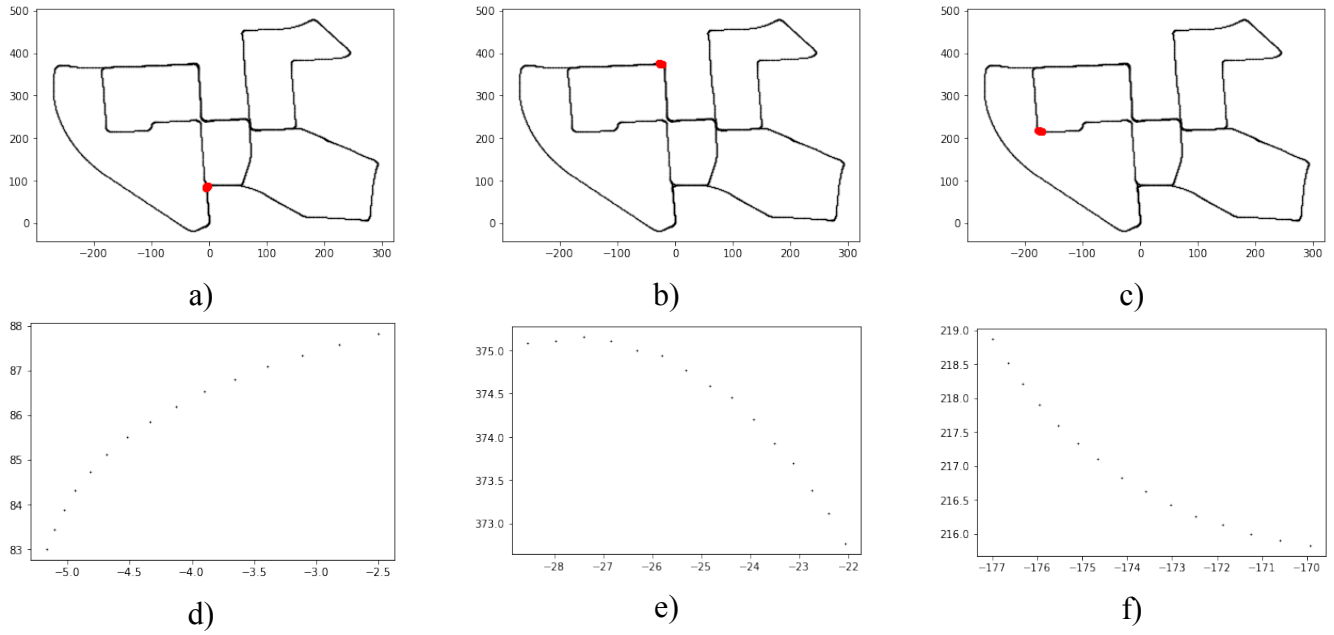


Рис. 12: Поиск и локализация поворотов для набора данных 00: a) 0 поворот, b) 4 поворот, c) 6 поворот, d) локализация 0 поворота, e) локализация 4 поворота, f) локализация 6 поворота.

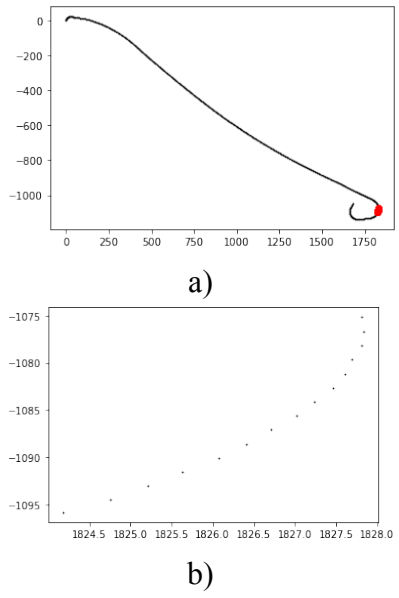


Рис. 13: Поиск и локализация поворотов для набора данных 01: a) 0 поворот, b) локализация 0 поворота.

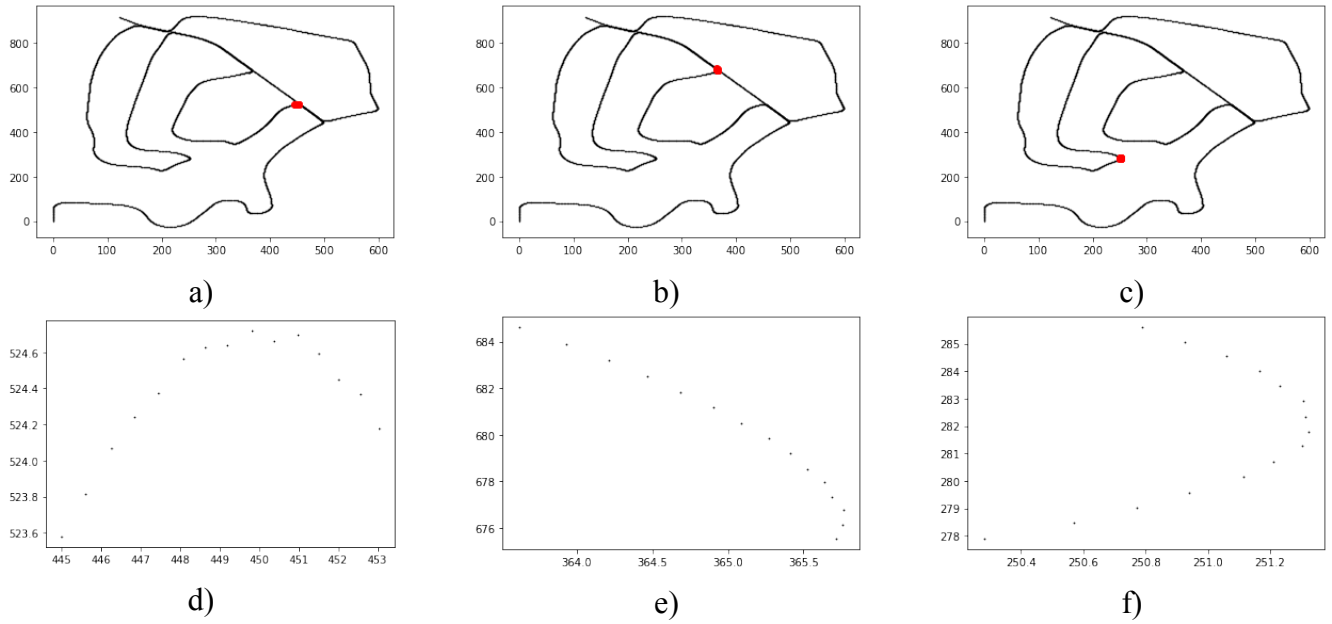


Рис. 14: Поиск и локализация поворотов для набора данных 02: a) 23 поворота, b) 34 поворота, c) 46 поворота, d) локализация 23 поворота, e) локализация 34 поворота поворота, f) локализация 46 поворота поворота.

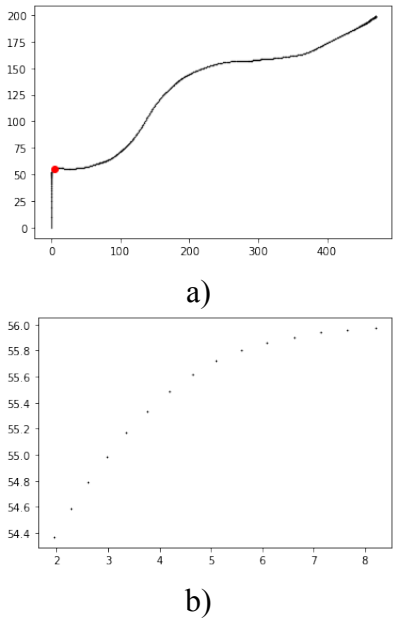


Рис. 15: Поиск и локализация поворотов для набора данных 03: a) нулевой поворот, b) локализация нулевого поворота.

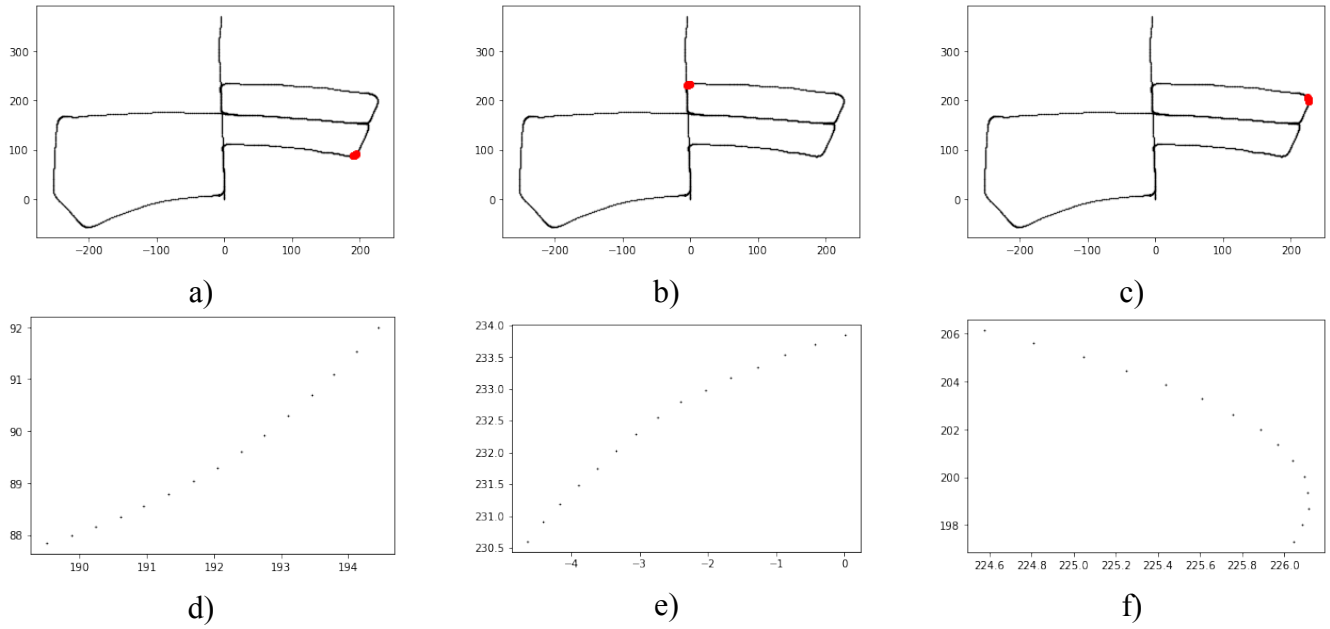


Рис. 16: Поиск и локализация поворотов для набора данных 05: a) 0 поворота, b) 3 поворота, c) 6 поворота, d) локализация 0 поворота, e) локализация 3 поворота поворота, f) локализация 6 поворота поворота.

5.4 Результат поиска уравнения второго порядка

5.4.1 Метод центра неопределенности

После поиска и локализации поворота вычислим уравнения второго порядка, которые описывают это движение с помощью подхода описанного в разделе 4. Ниже представлены полученные результаты.

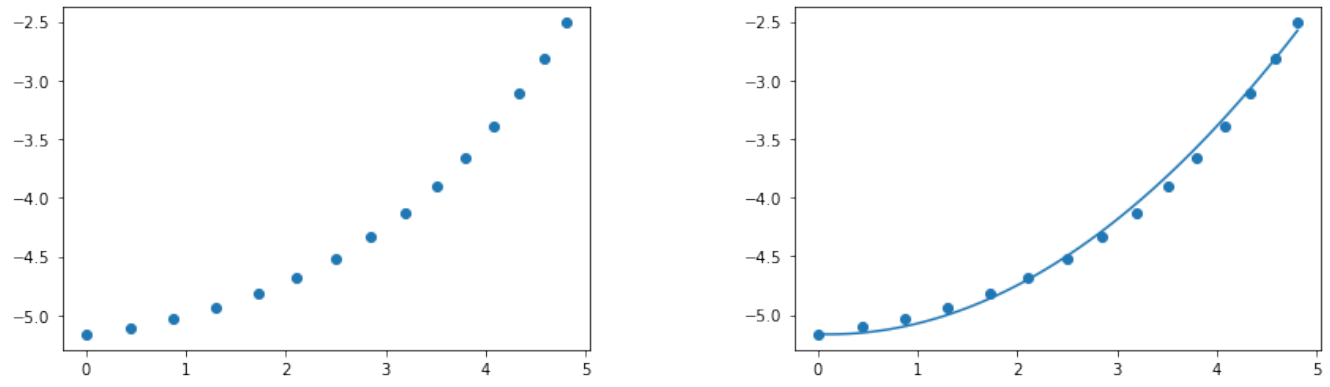
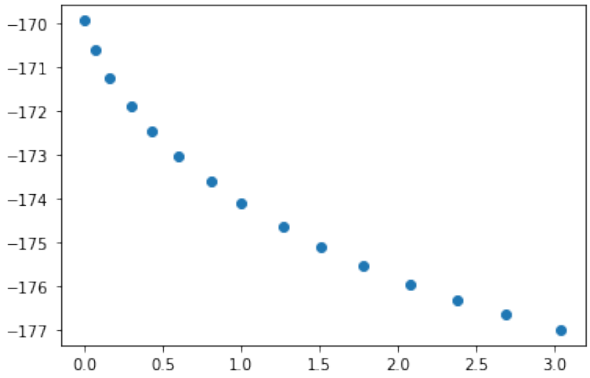
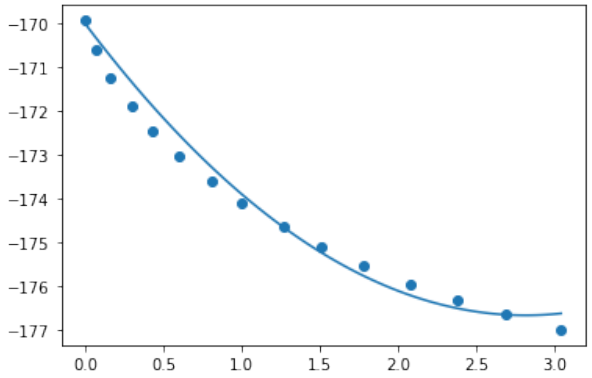


Рис. 17: Набор данных 00 поворот 0: a) исходные точки поворота, b) Уравнение итоговое - $(-5.1635 - 0.0266 \cdot x + 0.1174 \cdot x^2)$.

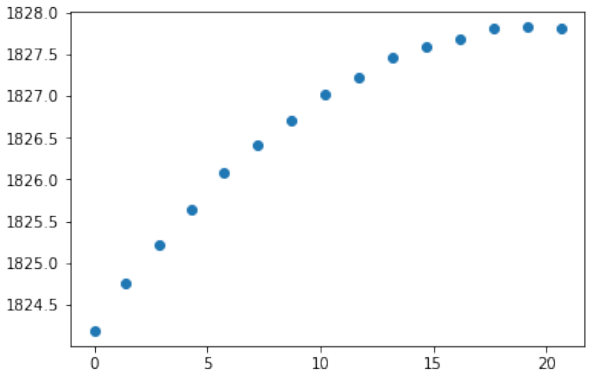


a)

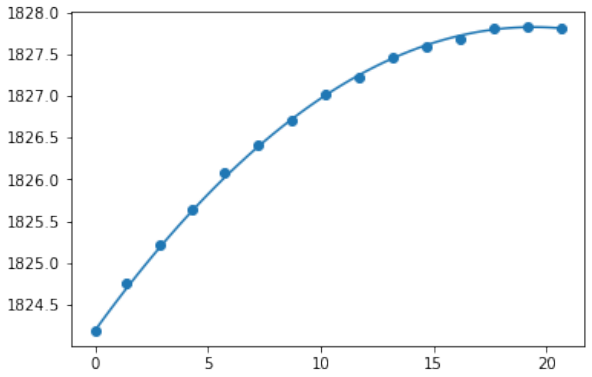


b)

Рис. 18: Набор данных 00 поворот 6: a) исходные точки поворота, b) Уравнение итоговое - $(-170.0094 - 4.7250 \cdot x + 0.8386 \cdot x^2)$.

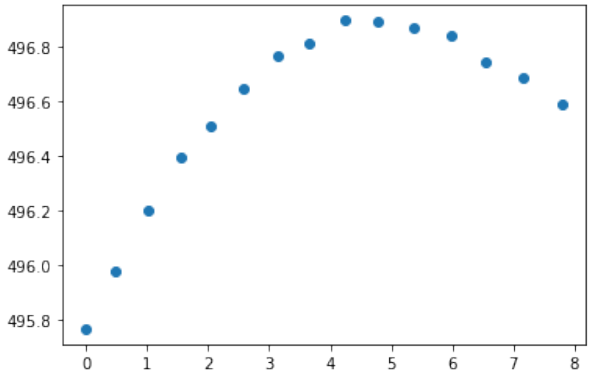


a)

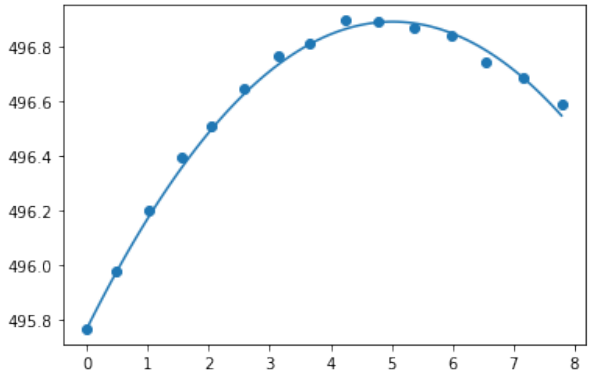


b)

Рис. 19: Набор данных 01 поворот 15: a) исходные точки поворота, b) Уравнение итоговое - $(1824.1830 + 0.3755 \cdot x - 0.0097 \cdot x^2)$.

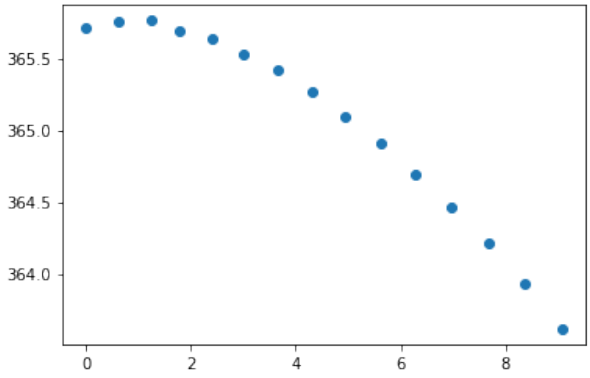


a)

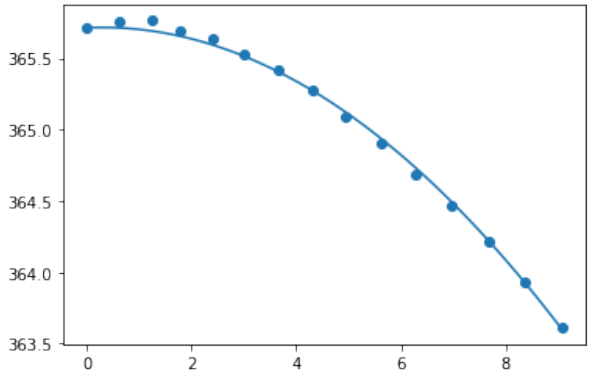


b)

Рис. 20: Набор данных 02 поворот 20: a) исходные точки поворота, b) Уравнение итоговое - $(495.7650 + 0.4487 \cdot x - 0.0447 \cdot x^2)$.

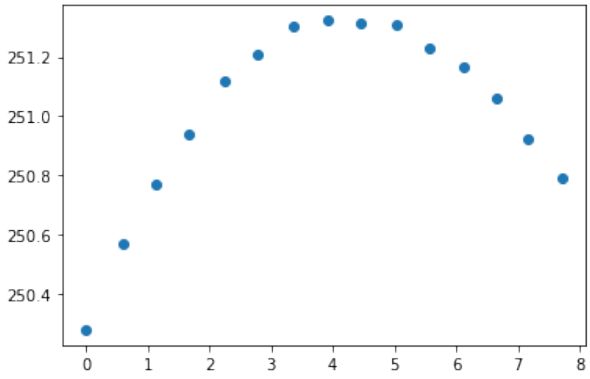


a)

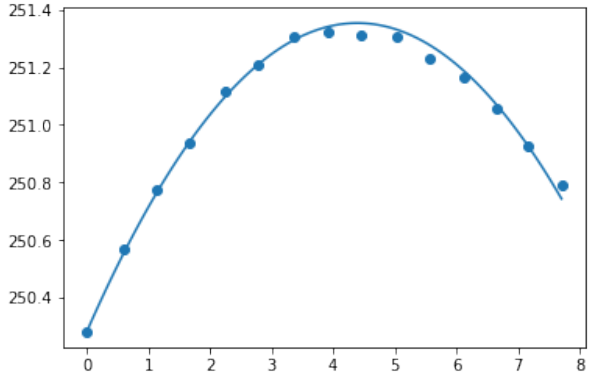


b)

Рис. 21: Набор данных 02 поворот 34: a) исходные точки поворота, b) Уравнение итоговое - $(365.7132 + 0.0160 \cdot x - 0.0274 \cdot x^2)$.

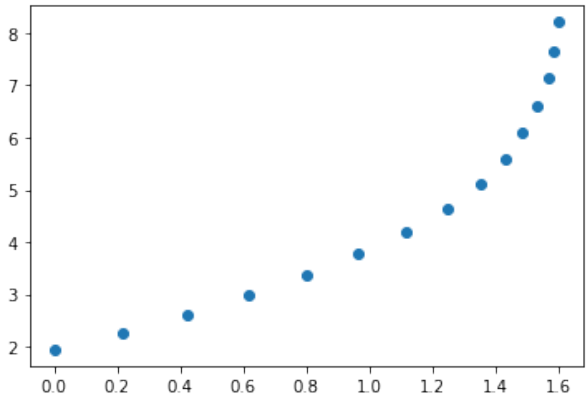


a)

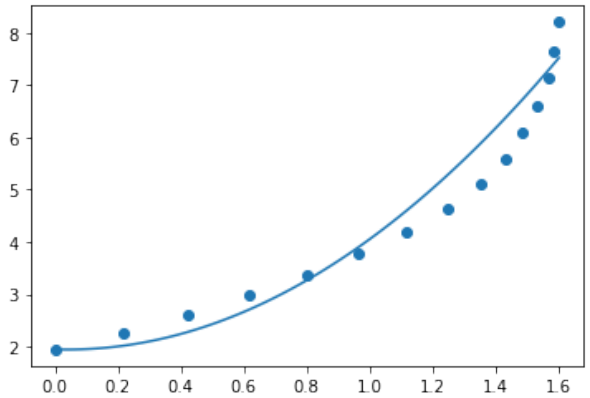


b)

Рис. 22: Набор данных 02 поворот 46: a) исходные точки поворота, b) Уравнение итоговое - $(250.2806 + 0.4887 \cdot x - 0.0556 \cdot x^2)$.



a)



b)

Рис. 23: Набор данных 03 поворот 00: a) исходные точки поворота, b) Уравнение итоговое - $(1.9461 - 0.1738 \cdot x + 2.2796 \cdot x^2)$.

5.4.2 Метод распознающего функционала

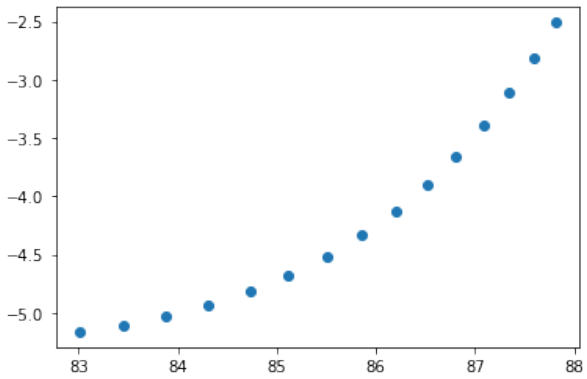
Кроме этого решим задачу с помощью функции `tolsolvty`. Где матрица X — это значения x^0, x^1, x^2 для координаты каждой точки поворота, а вектор y - это интервал допустимых значений y_i для каждой точки поворота, то есть $[h_{Hi}, h_{Bi}]$. На вход функция принимает:

- infA и supA — нижний и верхний концы интервальной матрица ИСЛАУ;
- infb и supb — нижний и верхний концы интервальной вектора правой части ИСЛАУ.

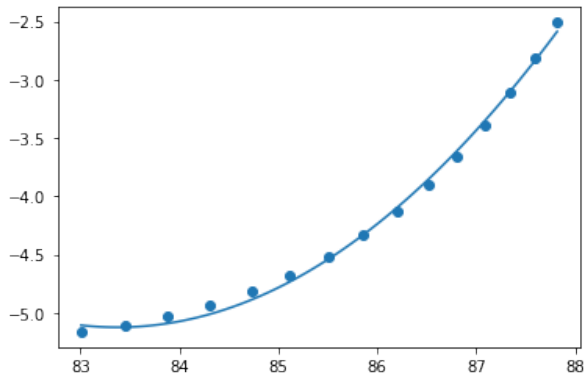
В результате функция возвращает:

- tolmax — максимум распознающего функционала;
- argmax — аргумент максимума распознающего функционала;
- ccode — код останова, который может принимать следующие значения:
  1. останов по изменению значения распознающего функционала;
  2. останов по норме суперградиента распознающего функционала;
  3. останов по аргументу распознающего функционала;
  4. останов по числу итераций программы;
  5. аварийный останов по числу шагов одномерного подъёма по направлению.

Результат выполнения функции `tolsolvty` для каждого набора входных данных представлен в таблице 2.



a)

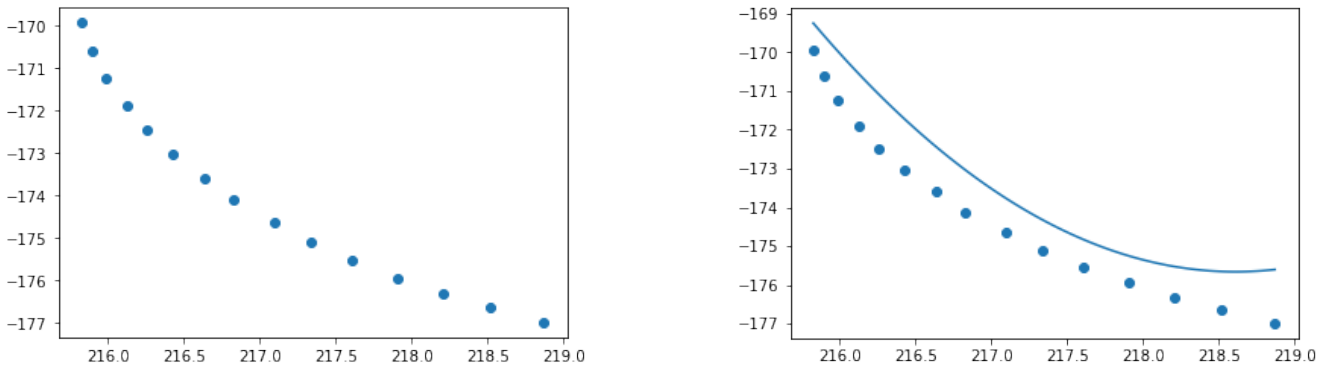


b)

Рис. 24: Набор данных 00 поворот 00: a) исходные точки поворота, b) Уравнение итоговое - $(887.68 - 21.417 \cdot x + 0.1284 \cdot x^2)$.

Набор данных	tolmax	β_0	β_1	β_2
00 0	0.9338	887.68	-21.417	0.1284
00 6	0.6564	39387	-361.95	0.8279
01 15	0.9661	-9450.2	-20.951	-0.0097
02 20	0.9627	-8058.1	38.41	-0.0431
02 34	0.9594	-12268	37.393	-0.0277
02 46	0.9708	-3935.2	29.659	-0.0525
03 00	0.3801	10557	-385.72	3.5242
05 00	0.9107	-1418.1	34.683	-0.1865
05 03	0.8818	16790	-145.98	0.3172
05 06	0.9824	-1019.1	12.507	-0.0314

Таблица 2: Коэффициенты уравнения второго порядка для описания движения объектов на повороте, полученные методом распознающего функционала.



a) b)

Рис. 25: Набор данных 00 поворот 06: a) исходные точки поворота, b) Уравнение итоговое - $(39387 - 361.95 \cdot x + 0.8278 \cdot x^2)$.

5.5 Оценка точности полученных результатов

Для оценки точности вычисляем для каждого поворота L_2 метрику:

$$L_2 = \sqrt{\sum_{i=1}^{n} (\beta_0 + \beta_1 \cdot x_i + \beta_2 \cdot x_i^2 - y_i)^2} \tag{5.1}$$

Результаты представлены в таблице 3 для метода центра неопределенности, а также вычислим количество интервалов, которые проходит найденная кривая второго порядка.

Все интервалы на всех поворотах были пройдены, то есть значение y_i, которое вычислялось

Набор данных	Мера L_2	Пройденные интервалы	Уравнение
00 0	0.2284	15	$-5.1635 - 0.0266 \cdot x + 0.1174 \cdot x^2$
00 6	1.2694	15	$-170.0094 - 4.7250 \cdot x + 0.8386 \cdot x^2$
01 15	0.1062	15	$1824.1830 + 0.3755 \cdot x - 0.0097 \cdot x^2$
02 20	0.0919	15	$495.7650 + 0.4487 \cdot x - 0.0447 \cdot x^2$
02 34	0.1261	15	$365.7132 + 0.0161 \cdot x - 0.0274 \cdot x^2$
02 46	0.0895	15	$250.2806 + 0.4887 \cdot x - 0.0556 \cdot x^2$
03 00	1.8020	15	$1.9461 - 0.1738 \cdot x + 2.2796 \cdot x^2$
05 00	0.3519	15	$189.5095 + 2.0001 \cdot x - 0.2031 \cdot x^2$
05 03	0.3697	15	$-4.6392 + 0.4845 \cdot x + 0.2801 \cdot x^2$
05 06	0.0674	15	$226.043 + 0.1071 \cdot x - 0.0309 \cdot x^2$

Таблица 3: Оценка точности результатов, полученных с помощью метода центра неопределенности.

по найденным уравнениям, лежит внутри интервала H_i. Аналогично проверим результат, полученный с помощью метода распознающего функционала. Оценка представлена в таблице 4.

Набор данных	Мера L_2	Пройденные интервалы	Уравнение
00 0	0.1933	15	$887.68 - 21.417 \cdot x + 0.1284 \cdot x^2$
00 6	4.1157	7	$39387 - 361.95 \cdot x + 0.8278 \cdot x^2$
01 15	0.8999	15	$-9450.2 - 20.951 \cdot x - 0.0097 \cdot x^2$
02 20	0.1185	15	$-8058.1 + 38.41 \cdot x - 0.0431 \cdot x^2$
02 34	1.2033	15	$-12268 + 37.393 \cdot x - 0.0277 \cdot x^2$
02 46	0.6382	15	$-3935.2 + 29.659 \cdot x - 0.052531 \cdot x^2$
03 00	3.3437	9	$10557 - 385.72 \cdot x + 3.5242 \cdot x^2$
05 00	0.2679	15	$-1418.1 + 34.683 \cdot x - 0.18648 \cdot x^2$
05 03	2.6495	15	$16790 - 145.98 \cdot x + 0.31723 \cdot x^2$
05 06	0.1422	15	$-1019.1 + 12.507 \cdot x - 0.0314 \cdot x^2$

Таблица 4: Оценка точности результатов, полученных с помощью метода распознающего функционала.

Для некоторых наборов данных при использовании метода распознающего функционала были достигнуты не все интервалы.

Значение коэффициентов в результате двух разных методов сильно разнятся, так как для метода центра неопределенности была выполнена предобработка входных данных, а именно смещение оси ординат таким образом, что данные по X отсортированы по возрастанию и $x[0] = 0$. Уравнение второго порядка были построены относительно преобразованных данных в методе центра неопределенности. Однако, метод распознающего функционала был выполнен на исходных данных без каких либо изменений.

Заключение

В ходе проделанной работы была рассмотрена задача визуальной одометрии для позиционирования беспилотных объектов, а именно роботов-спасателей разных типов. Были получены и локализованы траектории поворотов из общих карт из открытого набора данных KITTI, а далее решены ИСЛАУ для поиска и прогнозирования дальнейшей траектории движения роботов. Реализованный метод решения ИСЛАУ применим для поставленной задачи для различных поворотов при любом количестве точек траектории.

Список литературы

[1] Graeter J., Wilczynski A., Lauer M. Limo: Lidar-monocular visual odometry //2018 IEEE/RSJ international conference on intelligent robots and systems (IROS). – IEEE, 2018. – С. 7872-7879.

[2] A. Howard, ”Real-time stereo visual odometry for autonomous ground vehicles Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems IROS’08, Nice, France, September 22-26, 2008.

[3] S. P. Kleinschmidt and B.Wagner, ”Visual Multimodal Odometry: Robust Visual Odometry in Harsh Environments Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics SSRR’18, Philadelphia, Pennsylvania, August 6-8, 2018.

[4] D. G. Lowe, ”Distinctive Image Features from Scale-Invariant Keypoints International Journal of Computer Vision, 2004, Vol. 60, No. 2, pp. 91-110.

[5] S. M. Prakhya, L. Bingbing, L. Weisi, and U. Qayyum, ”Sparse Depth Odometry: 3D keypoint based pose estimation from dense depth data Proceedings of the 2015 IEEE International Conference on Robotics and Automation ICRA’15, Seattle, Washington, May 26-30, 2015, pp. 4216-4223.

[6] R. Voges, C.S. Wieghardt, and B. Wagner, ”Finding Timestamp Offsets for a Multi-Sensor System Using Sensor Observations Photogrammetric Engineering and Remote Sensing, 2018, Vol. 84, No. 6, pp. 357-366.

[7] R. Voges, and B. Wagner, ”Timestamp Offset Calibration for an IMU-Camera System Under Interval Uncertainty Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, October 1-5, 2018.

[8] R. Voges, B. Wagner, and V. Kreinovich, ”Effcient Algorithms for Synchronizing Localization Sensors Under Interval Uncertainty Reliable Computing (Interval Computations), 2020, accepted.

[9] J. Zhang and S. Singh, ”Laser-visual-inertial odometry and mapping with high robustness and low drift Journal of Field Robotics, 2018, Vol. 35, No. 8, pp. 1242-1264.

[10] Mur-Artal R., Tardós J. D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras //IEEE Transactions on Robotics. – 2017. – Т. 33. – №. 5. – С. 1255-1262.

[11] R. Mur-Artal, J. M. M. Montiel, and J. D. Tard´os, “ORB-SLAM: a versatile and accurate monocular SLAM system,” IEEE Trans. Robot., vol. 31, no. 5, pp. 1147–1163, 2015.

[12] D. G´alvez-L´opez and J. D. Tard´os, “Bags of binary words for fast place recognition in image sequences,” IEEE Trans. Robot., vol. 28, no. 5, pp. 1188–1197, 2012.

[13] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige, “Double window optimisation for constant time visual SLAM,” in IEEE Int. Conf. Comput. Vision (ICCV), 2011, pp. 2352–2359.

[14] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” in European Conference on Computer Vision, vol. 1, May 2006, pp. 430–443.

[15] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “BRIEF: Binary Robust Independent Elementary Features,” in European Conference on Computer Vision, vol. 6314, September 2010, pp. 778–792.

[16] R. Kuemmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “g2o: A general framework for graph optimization,” in IEEE Int. Conf. on Robot. and Autom. (ICRA), 2011, pp. 3607–3613.

[17] Реализация программы tolsolvty на языке программирования GNU Octave.- URL:http://www.nsc.ru/interval/Programing/OctCodes/tolsolvty.m

[18] Шарая И.А. ”Ограничено ли допустимое множество решений интервальной системы”// Вычислительные технологии.—2004.—т.9, №3.—с. 108—112.— URL:http://www.nsc.ru/interval/sharaya/Papers/ct04.pdf

[19] Воронцова Е.А. ”Линейная задача о допусках для интервальной модели межотраслевого баланса”// Вычислительные технологии.—2017.—т.22,№2.—с.67—84.— URL:http://www.nsc.ru/interval/Library/Thematic/Economics/Vorontsova-JCT-2017.pdf (дата обращения: 01.06.2020).