

ФЕДЕРАЛЬНОЕ АГЕНТСТВО НАУЧНЫХ ОРГАНИЗАЦИЙ
ВЫЧИСЛИТЕЛЬНЫЙ ЦЕНТР
ДАЛЬНЕВОСТОЧНОГО ОТДЕЛЕНИЯ
РОССИЙСКОЙ АКАДЕМИИ НАУК

На правах рукописи

Камаев Александр Николаевич

**ПОСТРОЕНИЕ ФОТОГРАФИЧЕСКИХ КАРТ ПОДВОДНОГО ДНА НА
ОСНОВЕ БОЛЬШИХ МАССИВОВ ИЗОБРАЖЕНИЙ**

Специальность 05.13.11 – Математическое и программное обеспечение
вычислительных машин, комплексов и компьютерных сетей

Диссертация на соискание учёной степени
кандидата технических наук

Научный руководитель –
чл.-корр. РАН, д. ф.-м. н. С.И. Смагин

Хабаровск – 2014

СОДЕРЖАНИЕ

Список условных сокращений.....	6
Введение.....	7
Глава 1. Проблемы построения фотографических карт подводного дна.....	15
1.1. Постановка задачи	15
1.1.1. Получение фотоматериала для фотографических карт дна.	15
1.1.2. Использование фотографических карт дна.	18
1.1.3. Специфика задачи построения фотографических карт дна	18
1.2. Анализ методов автоматической сшивки изображений	21
1.2.1. Сопоставление изображений.	21
1.2.2. Калибровка камер.	27
1.2.3. Вычисление положения и ориентации камер.	29
1.2.4. Проекция изображений на одну поверхность.	31
1.2.5. Совмещение изображений.	32
1.3. Применение существующих систем сшивки к подводным изображениям	33
1.4. Основные задачи, требующие решения.....	35
Глава 2. Поиск связанных изображений.....	36
2.1. Постановка задачи	36
2.2. Система координат изображения.....	37
2.3. Визуальные особенности на изображениях.....	39
2.3.1. Компенсация освещения АНПА.	39
2.3.2. Обнаружение особых точек.	43
2.3.3. Построение дескрипторов особых точек.....	46

2.4. Сопоставление особенностей на паре изображений.....	47
2.4.1. Оценки соответствия дескрипторов.....	47
2.4.2. Выявление соответствий.....	48
2.4.3. Фильтрация ложных соответствий.	50
2.5. Алгоритм быстрого поиска связанных изображений.....	53
2.5.1. Необходимость применения быстрых алгоритмов.	53
2.5.2. Списки кандидатов на сопоставление.	54
2.5.3. Поиск похожих дескрипторов особенностей.....	54
2.5.4. Сопоставление пар в больших наборах изображений.	60
2.6. Результаты.....	64
2.6.1. Описание данных.....	64
2.6.2. Компенсация освещения.....	64
2.6.3. Сопоставление пар изображений с малым перекрытием.	65
2.6.4. Поиск связанных изображений.	66
Глава 3. Вычисление положения и ориентации изображений	70
3.1. Постановка задачи	70
3.2. Поиск начального приближения решения.....	70
3.2.1. Начальные значения параметров.	71
3.2.2. Относительное позиционирование изображений.....	72
3.3. Эффективное решение системы линейных алгебраических уравнений ...	75
3.3.1. Общий алгоритм решения.....	75
3.3.2. Минимизация заполнения матрицы системы.	76
3.3.3. Обратный алгоритм Катхилла–Макки.....	77
3.3.4. Метод вложенных сечений.	80

3.3.5. Модификация метода вложенных сечений.	81
3.3.6. Метод параллельных сечений.	82
3.4. Итеративное уточнение решения.....	85
3.4.1. Алгоритм Левенберга–Марквардта.	85
3.4.2. Структура рассматриваемой задачи.....	86
3.4.3. Составление таблицы связи точек.	88
3.4.4. Инициализация координат точек дна.	90
3.4.5. Параметризация.	91
3.4.6. Разреженный алгоритм Левенберга–Марквардта.	93
3.5. Результаты.....	98
3.5.1. Исходные данные для тестирования.....	98
3.5.2. Алгоритмы решения СЛАУ.....	101
3.5.3. Задача поиска приближения планарных координат.	105
3.5.4. Задача уточнения положения и ориентации АНПА.....	106
Глава 4. Визуальное представление фотографической карты дна.....	110
4.1. Постановка задачи	110
4.2. Алгоритм простой сшивки.....	112
4.2.1. Построение аппроксимирующей плоскости.	114
4.2.2. Оценка корректности построенной плоскости.	115
4.2.3. Коррекция плоскости.	117
4.2.4. Функции проекции.....	119
4.3. Сшивка на основе трёхмерной модели дна.....	120
4.3.1. Построение трёхмерной модели рельефа дна.....	121
4.3.2. Процедура проецирования.	122

4.4. Устранение швов на стыках изображений	124
4.5. Результаты	127
4.5.1. Исходные данные для тестирования.....	127
4.5.2. Результат бесшовной сшивки изображений.	129
4.5.3. Трёхмерная модель дна.	134
Заключение	136
Список литературы	139

СПИСОК УСЛОВНЫХ СОКРАЩЕНИЙ

АНПА – автономный необитаемый подводный аппарат

МНК – метод наименьших квадратов

СЛАУ – система линейных алгебраических уравнений

LMS – least median of squares

RANSAC – random sample consensus

RGB – цветовая модель red green blue

SSR – single scale retinex

ВВЕДЕНИЕ

Актуальность темы исследования

Подводное дно – интересный и вместе с тем труднодоступный для изучения объект. Перспективным методом исследования дна является его фотографирование с помощью автономных необитаемых подводных аппаратов (АНПА) [1]. Поскольку глубоко под водой освещение отсутствует, для получения качественных снимков необходимо производить съемку на небольшом удалении от поверхности дна. Это приводит к тому, что каждый снимок охватывает малый участок территории, следовательно, для покрытия больших участков необходимо большое количество снимков. Так как работать с общей фотографической картой исследуемого участка подводного дна значительно удобнее, чем с набором из тысяч разрозненных изображений, актуальной представляется задача сшивки подводных изображений в такие карты. Кроме того задача сшивки подводных изображений тесно пересекается с задачей навигации АНПА по фото или видео информации.

Фотографические карты дна могут быть использованы для самых различных целей, но основные перечислены ниже:

1. Изучение поверхности дна, флоры и фауны, обитающей на дне.
2. Мониторинг подводной инфраструктуры (кабели, трубопроводы, подводные постройки и др.).
3. Поиск затонувших объектов.
4. Навигация АНПА по построенной фотографической карте.

Задача построения фотографических карт дна не нова и имеет ряд решений. Одна из первых систем автоматической сшивки подводных изображений представлена в работе [58]. В данной системе рассматриваются изображения с заранее известным положением. Успешная попытка использования последовательной видеоинформации для расчета траектории аппарата была предпринята позже в работе [17]. Подход, предложенный в данной работе, заключается в обнаружении и сопоставлении особых точек на

последовательных изображениях. Для увеличения производительности процедуры сопоставления используется фильтр Калмана, с помощью которого предсказываются положения особенностей на каждом следующем изображении. Для вычисления траектории (смещения между особыми точками на последовательных изображениях) используется обобщённое преобразование Хафа. В публикациях [67, 88, 2] рассматриваются похожие на [17] подходы. Отличия в основном заключаются в способе установления связи между последовательными изображениями. В [49] описан подход к построению мозаик на основе различных последовательностей изображений, полученных на разных высотах над дном. Мозаика, собранная из изображений, полученных с большой высоты, используется для построения мозаик из изображений, полученных с малой высоты. Очевидно, такой подход хорошо подходит для небольших глубин, на которые проникает свет и позволяет производить съёмку с большой высоты.

Современный подход к построению фотографических карт дна или мозаик состоит из четырёх основных этапов: сопоставление последовательных изображений, сопоставление остальных пересекающихся пар изображений, вычисление движения камер и совмещение изображений, спроецированных на поверхность карты. При этом процесс сопоставления непоследовательных пар использует приближённую структуру движения, полученную на основе сопоставления последовательных пар. Эта структура движения позволяет с некоторой точностью определить места, в которых пересекаются непоследовательные пары изображений. После обнаружения набора дополнительных пар структура движения уточняется, и процесс поиска непоследовательных пересекающихся пар может быть выполнен повторно для уточнённой структуры. Подобный подход реализован в работе [85]. Похожий подход, но использующий дополнительную информацию о положении АНПА из навигационной системы представлен в [42]. Недостатком подхода является то, что сшивка последовательных изображений далеко не всегда определяет

структуру движения с достаточной точностью. Точности рассчитанной структуры может не хватить для обнаружения пересекающихся непоследовательных пар изображений. Более того, иногда последовательную сшивку осуществить невозможно.

Рассмотрим три случая, когда последовательная сшивка изображений невозможна либо неточна. Первый случай – обследование большой зоны, когда между пересечениями своей траектории АНПА проходит большие расстояния. Накапливающаяся при последовательной сшивке ошибка легко может дойти до 10 и более метров, что потребует сравнения большого числа пар изображений между собой для замыкания траектории. Второй случай – недостаточно пересекающиеся последовательные изображения. Когда АНПА идёт быстро или редко выполняет фотографирование, между многими соседними снимками не будет обнаружено соответствий, что приведёт к более быстрому накоплению ошибки. Третий случай – особенности дна. На дне могут быть большие однородные зоны, не содержащие особенностей для сопоставления последовательных снимков.

Чтобы сделать процесс построения фотографических карт возможным для любых последовательностей изображений, независимо от разрывов и пропусков данных, а также от размера исследуемой территории необходим механизм обнаружения положения каждого изображения, не предъявляющий никаких требований к порядку этих изображений. Разработка такого механизма на сегодняшний момент является актуальной задачей, требующей решения большого числа сопутствующих проблем.

Цели и задачи исследования

Основной целью настоящего исследования является разработка и исследование методов построения фотографических карт подводного дна и создание программного средства, реализующего разработанные методы. Для достижения поставленной цели необходимо решить следующий ряд основных задач:

1. Исследование основных подходов к автоматической сшивке изображений и определение этапов, требующих специальных решений, учитывающих специфику подводных изображений.

2. Задача поиска связанных изображений. Решение этой задачи должно учитывать низкую чёткость и плохую освещённость подводных снимков, должно быть работоспособным при малом количестве деталей на снимках и при высокой повторяемости однотипных объектов на дне.

3. Задача ориентирования и позиционирования снимков на основе визуальных особенностей. Для сшивки изображений в единую карту дна необходимо найти положение и ориентацию каждого снимка на карте. Алгоритм поиска положения и ориентации должен быть достаточно быстрым, чтобы справляться с большими объемами входных данных.

4. Задача сшивки изображений. Необходимо разработать эффективную процедуру проецирования каждого изображения на фотографическую карту и смещения цветов проецируемых изображений, позволяющую получить чёткую карту подводного дна.

5. Разработка и тестирование программного средства, реализующего все алгоритмы построения фотографической карты.

Объект и предмет исследования

Объект исследования – методы, алгоритмы и программные средства автоматической сшивки изображений.

Предмет исследования – методы, алгоритмы и программные средства сшивки изображений подводного дна, получаемых с АНПА.

Методы исследования

Теоретической основой диссертационной работы послужили исследования российских и зарубежных учёных в области компьютерного зрения и обработки цифровых сигналов, в частности изображений. Широкое отражения нашло использование методов вычислительной математики,

например методы наименьших квадратов, методы решения систем линейных алгебраических уравнений, методы нелинейной оптимизации и др.

В работе активно используется математическое моделирование, когда реальные физические объекты и процессы заменяются на их математические модели, для которых применяются разрабатываемые алгоритмы.

Все, предлагаемы в работе методы и алгоритмы решения задач, проверяются на основе экспериментов. При этом в качестве исходных данных для экспериментов используются как реальные данные, так и синтетические, в зависимости от конкретной задачи.

Научная новизна

Научная новизна работы выражается в следующем:

1. Предложен алгоритм сравнения пар подводных изображений. В отличие от существующих алгоритмов в нем делается упор на то, что входные изображения обладают низкой чёткостью, малым числом деталей и могут генерировать большое число ложных точечных соответствий. Предложенный алгоритм сравнения использует также данные навигационной системы АНПА, которые могут быть получены с достаточной точностью (курс, глубина, расстояния до дна), что также позволяет повысить его эффективность в сравнении с существующими алгоритмами применительно к задаче построения фотографических карт подводного дна.
2. Предложен быстрый алгоритм поиска соответствующих пар в больших наборах изображений. Основным преимуществом данного алгоритма является то, что он не требует упорядоченного поступления изображений.
3. Предложен алгоритм инициализации нелинейной задачи вычисления положений и ориентаций изображений.
4. Рассматривается метод проецирования изображения на поверхность фотографической карты дна с минимальными искажениями. Предложенный метод позволяет получать четкую бесшовную карту подводного дна даже в

случае перепадов высот рельефа дна, сопоставимых с высотой АНПА над дном.

5. Разработано программное средство «AUV photo stitcher» предназначенное для составления фотографических карт подводного дна на основе больших массивов изображений.

Положения, выносимые на защиту

На защиту выносятся следующие положения:

1. Предложен алгоритм установления связи между парой изображений, учитывающий низкую чёткость, нелинейное изменение освещения, высокую повторяемость однотипных объектов, характерные для подводных изображений и алгоритм для быстрого выявления потенциально связанных пар в больших массивах изображений, не требующий упорядоченности входных изображений и знания их приблизительного положения.
2. Разработан алгоритм инициализации начального значения ориентации и положения снимков для решения нелинейной задачи уточнения этих параметров. Описаны методики эффективного решения задачи поиска начального приближения и нелинейного уточнения положения и ориентации снимков на современных компьютерах.
3. Разработано программное средство «AUV photo stitcher», предназначенное для построения фотографических карт подводного дна на основе больших массивов подводных изображений, получаемых с АНПА.

Практическая значимость полученных результатов

Практическая ценность работы заключается в том, что на базе предложенных в ней алгоритмов можно создавать системы построения фотографических карт подводного дна на основе изображений, поступающих с АНПА после разведывательных миссий. В ходе работы над диссертацией также было разработано программное средство «AUV photo stitcher», которое уже является законченным решением в области построения фотографических карт по подводным изображениям.

Другим возможным применением результатов работы является построение систем навигации АНПА на основе фото либо видеоинформации. Использование алгоритмов описанных в данной работе может стать базой для построения подобных систем.

Апробация результатов работы

Апробация результатов работы проводилась на следующих международных и всероссийских конференциях:

- Всероссийская научно-практическая конференция «Информационные технологии и высокопроизводительные вычисления», 25–27 июня 2013, Хабаровск, Россия.
- 23-я Международная Конференция по Компьютерной графике и Зрению «Графикон» 16-20 сентября 2013, Владивосток, Россия.
- 5-я Всероссийская научно-техническая конференция «Технические проблемы освоения Мирового океана» 2013, Владивосток, Россия.
- 4-я Всероссийская конференция «Математическое моделирование и вычислительно-информационные технологии в междисциплинарных научных исследованиях» 2014, Иркутск, Россия.

Публикации автора по теме диссертации

По теме диссертационного исследования автором опубликованы следующие работы:

- Камаев А.Н. Относительное ориентирование снимков низкой контрастности, полученных с АНПА, на основе точечных особенностей // Учёные заметки ТОГУ. – 2011. – Том 2, №2. – С. 32-42
- Камаев А.Н. Исследование алгоритмов упорядочивания коэффициентов систем линейных алгебраических уравнений, возникающих в задачах компьютерного зрения // Информатика и системы управления. – 2013. – №3. – С. 32-44.

- Камаев А.Н. Автоматическая сшивка изображений, полученных с АНПА при исследовании подводного дна // В трудах конф. ТПОМО-5. – Владивосток 30 сентября – 4 октября 2013г. – С. 474 –479.
- Камаев А.Н. Создание панорамных карт подводного дна на основе больших массивов изображений // В трудах конф. Графикон2013. – Владивосток 16-20 сентября, 2013г. – С. 298 –301.
- Камаев А.Н. Позиционирование изображений морского дна полученных с помощью АНПА // Подводные исследования и робототехника. – 2013. – №2/16. – С. 38-47.
- Камаев А.Н. Свидетельство Роспатента об официальной регистрации программы для ЭВМ AUV photo stitcher. – № 2014613051. – 17.03.2014.

Структура и объем диссертации

Диссертация состоит из списка условных сокращений, введения, основной части, заключения и списка литературы.

Основная часть разбита на четыре главы:

- Глава 1. Проблемы построения фотографических карт подводного дна
- Глава 2. Поиск связанных изображений
- Глава 3. Вычисление положения и ориентации изображений
- Глава 4. Визуальное представление фотографической карты дна

Полный текст диссертации размещён на 148 страницах. В текст работы включены 43 рисунка, 3 таблицы. Список литературы содержит 113 источников, 101 из них на иностранном языке.

ГЛАВА 1. ПРОБЛЕМЫ ПОСТРОЕНИЯ ФОТОГРАФИЧЕСКИХ КАРТ ПОДВОДНОГО ДНА

1.1. Постановка задачи

Фотографической картой подводного дна будем называть набор сшитых между собой фотографий поверхности дна, спроецированных на горизонтальную плоскость. Во многом фотографические карты дна похожи на спутниковые карты земли, предоставляемые сервисами таких компаний, как Google, Яндекс и NASA. Если фотоматериал для построения спутниковых карт получают со спутников, то для построения фотографических карт дна с АНПА [1].

1.1.1. Получение фотоматериала для фотографических карт дна. АНПА имеет на борту камеру, жёстко связанную с его корпусом и направленную вниз – рис. .

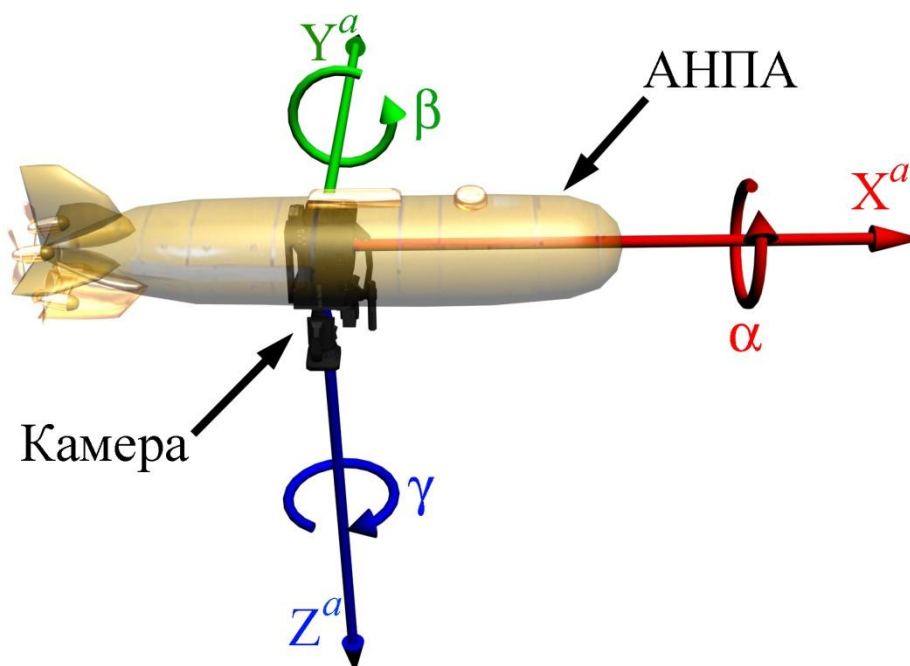


Рис. 1. АНПА и камера.

Локальная система координат, связанная с АНПА также представлена на рис. . Вдоль главной оси аппарата направлена ось X^a , в сторону направлена ось Y^a , а ось Z^a направлена вниз, по направлению взгляда камеры. Такое направление

оси Z^a выбрано для удобства использования стандартной модели перспективной камеры [10], согласно которой камера должна смотреть в направлении оси Z^a . Ориентация АНПА определяется тремя углами вращения: креном – α , дифферентом – β и курсом – γ . Крен определяет поворот АНПА вокруг оси X^a , дифферент вокруг оси Y^a , а курс вокруг оси Z^a .

Для получения набора фотографий, из которых строится карта, аппарату даётся задание многократно пройти над интересующим участком дна, выполняя непрерывное фотографирование. При этом траектория строится так, чтобы каждый новый проход захватывал неисследованные области. Фотографирование при этом выполняется с частотой от 0.5 до 1 Гц. Пример допустимой траектории АНПА представлен на рисунке .

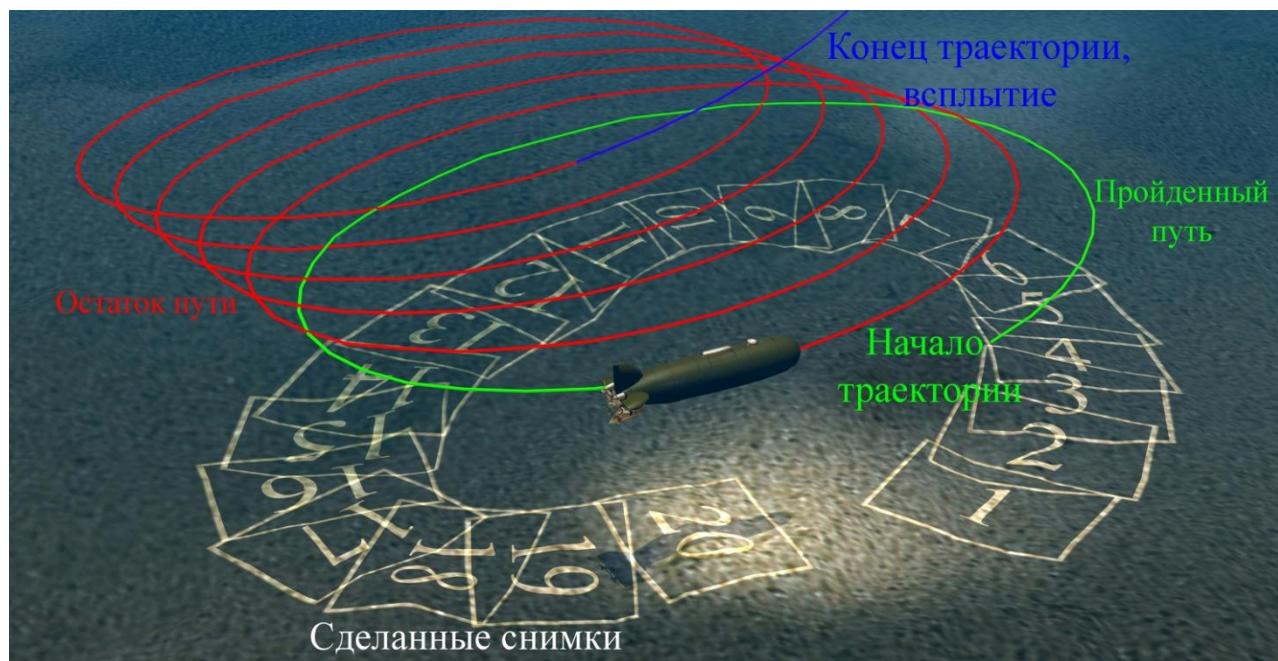


Рис. 2. Возможная траектория АНПА.

В момент выполнения фотографирования аппарат старается двигаться преимущественно плоско. Это означает, что в хороших условиях колебания крена и дифферента аппарат незначительны. Однако из-за неровностей рельефа дна и течений колебания крена и дифферента могут оказаться значительными. АНПА ориентируется за счёт своей навигационной системы. При этом некоторые характеристики, такие как курс γ , высота над дном h , глубина d

могут быть определены достаточно точно, а некоторые, такие как положение АНПА только с очень большой погрешностью.

В первом приближении движение АНПА можно считать плоским, однако для точного построения фотографической карты такой модели движения будет недостаточно, поэтому необходимо будет учитывать все степени свободы АНПА: три степени свободы на перемещение и три на вращение. Тем не менее, примем допущение, что участок дна, охватываемый одной фотографией с АНПА, является достаточно плоским, высота перепада рельефа на таком участке должна составлять не более 40% от общей высоты аппарата над дном – рисунок .

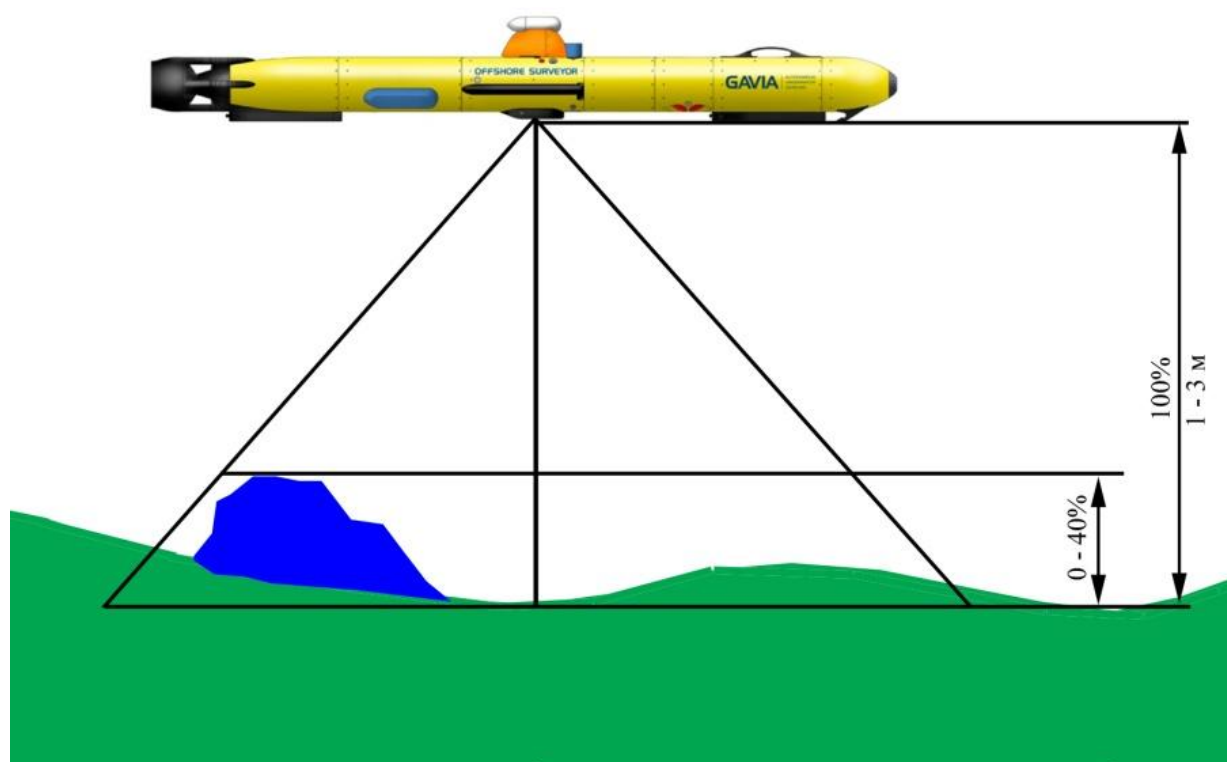


Рис. 3. Перепад рельефа в зоне снимка.

Одна миссия АНПА может длиться до нескольких десятков часов, что при частотах съемки 0.5 – 1 Гц означает получение десятков тысяч фотографий. Данные фотографии должны быть точно позиционированы, ориентированы, сшиты и спроецированы на горизонтальную плоскость – фотографическую карту дна.

1.1.2. Использование фотографических карт дна. Подводное дно является труднодоступным и, вследствие этого, малоисследованным объектом. С активным развитием АНПА появляется возможность опуститься всё глубже и узнать больше. Поскольку организация хорошей связи с АНПА, предполагающей возможность обмениваться большими объемами данных во время выполнения миссии – сложная задача, все получаемые данные хранятся на борту и передаются человеку только после всплытия аппарата. Одним из видов таких данных являются фотографии морского дна, которые, объединённые в фотографические карты, позволяют исследовать рельеф, покрытие дна, морских животных и растений, обитающих на дне.

Кроме чисто исследовательских задач фотографические карты дна могут использоваться для мониторинга технических сооружений. Часто под водой располагаются различные объекты: трубопроводы, кабели, тоннели и другие. За их состоянием необходимо следить, а периодическое получение фотографических карт дна позволяет легко это сделать.

Фотографические карты подводного дна также хорошо подходят для поиска затонувших объектов, особенно, когда обломки этих объектов рассредоточены по большой территории. Другой возможностью применения таких карт является задача навигации АНПА. Готовая карта дна позволяет АНПА точно знать свои координаты в каждый момент времени, когда он идёт над дном и точнее строить свою траекторию. Однако последнее применение справедливо лишь в случаях, когда дно не претерпевает постоянных локальных изменений.

1.1.3. Специфика задачи построения фотографических карт дна

Фотографическая карта подводного дна получается в результате автоматической сшивки изображений. Задачи автоматической сшивки изображений хорошо известны в компьютерном зрении, тем не менее, задача построения фотографических карт подводного дна обладает рядом специфических особенностей, затрудняющих использование стандартных

методов. Специфика задачи основывается на трёх основных факторах – отсутствие света под водой, отсутствие точной навигации и особой структуре дна.

Поскольку под водой нет света, который необходим для фотосъемки, АНПА оснащается осветительными приборами. Эти осветительные приборы не способны достаточно осветить участок дна с большой высоты, поэтому высота АНПА над дном в момент фотографирования h ограничена. Данная высота зависит от мощности осветительных приборов, которая также ограничена ввиду автономности источника питания. На практике, при высоте больше трёх метров качество снимков существенно падает. На рисунке показаны три снимка, сделанные с различной высоты h , если детали на снимках, где $h=1.21$ метров и $h=1.92$ метров хорошо различимы, то снимок, сделанный с высоты $h=2.82$ метров практически однотонно серый.

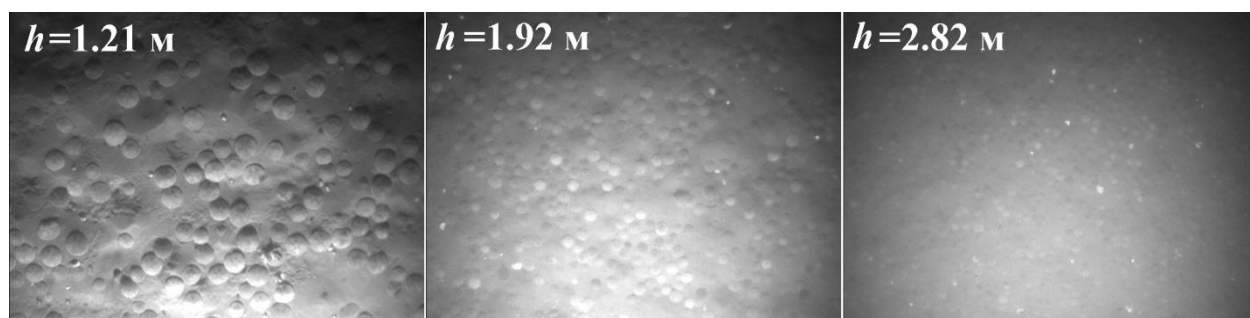


Рис. 4. Зависимость качества снимка от высоты съемки.

Из рисунка видно, что даже изображение, полученное со средней высоты в 1.92 метра, не выглядит резким. Оно нечёткое, слабоконтрастное. Кроме того, на всех снимках хорошо видно, что центральная зона изображения освещена гораздо ярче. Из-за того, что источник света располагается на АНПА и движется вместе с аппаратом, одна и та же область дна на снимках освещается по-разному. На одном снимки часть дна может попасть в зону увеличения освещённости, а на другом в зону спада. Такая нелинейность освещения совместно с низкой чёткостью и слабой контрастностью существенно затрудняют процедуру сопоставления изображений.

Из ограничения высоты съемки h вытекает ещё одна особенность. Один снимок не в состоянии охватить большую площадь поверхности дна. Например, снимок на рисунке , сделанный с высоты $h=1.92$ охватывает всего 2.35 м^2 поверхности. Это приводит к тому, что для покрытия больших площадей требуется огромное число снимков (до нескольких десятков тысяч).

Поскольку точная навигация АНПА в большинстве случаев невозможна, координаты аппарата в момент получения каждого снимка неизвестны. Ошибка в определении координат может достигать десятков длин изображений, что делает эти координаты непригодными для составления фотографических карт. При аэрофотосъемке или съемке со спутников координаты камеры могут быть получены с очень хорошей точностью, тогда как при подводной съемке нужная точность недостижима. Два негативных фактора: необходимость большого числа изображений и неизвестные их координаты – накладываются и существенно усложняют задачу сшивки подводных изображений.

Специфическая структура дна также добавляет трудностей. На дне часто можно встретить большое число однотипных объектов. Это водоросли, ракушки, морские ежи и звёзды, скопления мелких камней и др. Очевидно, что на изображениях эти объекты дают однотипные визуальные особенности, что в совокупности с большим числом изображений с неизвестным положением ещё больше усложняет задачу поиска истинных соответствий.

Специфичным для типичной задачи сшивки изображений в фотографические карты можно назвать характер движения между изображениями. Сшивка изображений, полученных с камеры с шестью степенями свободы, практически не рассматривается. В основном задачи автоматической сшивки применяются к построению панорам, для которых характерны либо чистое вращение камер без перемещений либо плоское перемещение без вращений.

Есть также несколько особенностей, облегчающих задачу построения фотографической карты дна. К таким особенностям относятся известные

параметры камеры АНПА, а также доступные из навигационной системы данные о глубине d , о высоте над дном h и о курсе γ в момент получения каждой фотографии. Эти данные могут быть определены с хорошей точностью любым АНПА. Близкая к плоской траектория движения АНПА и небольшие колебания крена и дифферента в процессе фотографирования также являются благоприятной особенностью рассматриваемой задачи.

1.2. Анализ методов автоматической сшивки изображений

Построение фотографической карты дна – сложный многоэтапный процесс, в котором можно выделить следующие основные этапы, присущие большинству задач автоматической сшивки изображений:

- сопоставление изображений,
- калибровка камер,
- вычисление положения и ориентации камер,
- проекция изображений на одну поверхность,
- совмещение изображений.

Далее проанализируем существующие подходы к решению задач на перечисленных этапах. Рассмотрим, какие этапы построения фотографических карт не требуют специальных решений, а для каких необходима доработка существующих методов либо создание принципиально новых в силу специфики задачи.

1.2.1. Сопоставление изображений. При автоматической сшивке изображений, их сопоставление является ключевой процедурой. В результате сопоставления изображений определяется, связаны ли эти изображения и, если они связаны, определяется содержание связи.

Первый шаг в процедуре сопоставления – определение модели связи изображений. Модель связи изображений показывает, как пиксели одного изображения связаны с пикселями другого изображения. Выбор конкретной модели зависит от вида движения камеры при получении изображений.

Модели, подходящие для комбинаций движений различного вида, а также учитывающие внутренние параметры камеры рассмотрены в трудах [57, 98].

Для сопоставления изображений существует два подхода: попиксельное сопоставление и сопоставление на основе особенностей. Попиксельное, или прямое сопоставление заключается в выборе одного изображения в качестве эталона, и варьировании параметров движения второго изображения с одновременным расчетом ошибки сопоставления для каждой вариации. В итоге выбирается такой набор параметров, который дал минимальную ошибку. В качестве ошибки могут быть выбраны как простой квадрат расстояния между интенсивностями пикселей изображений, так и более устойчивые функции [66, 54, 29, 97, 30]. Также может быть использована модель, учитывающая разную яркость изображений путём ввода коэффициентов смещения и усиления [77, 47, 46, 24]. Другим вариантом функции ошибки является коэффициент нормализованной кросс-корреляции между изображениями.

Перебор всех возможных значений параметров даже в случае, когда движение между изображениями ограничивается двухмерным перемещением – очень долгая процедура. Для её ускорения был предложен вариант сопоставления от худшего к лучшему. При таком варианте сопоставления для каждого изображения строится пирамида изображений, где каждое следующее изображение меньше оригинального. Процедура сопоставления начинается с изображений с маленьким разрешением, а результат сопоставления используется для инициализации ограничений варьирования параметров при сопоставлении изображений большего размера [89, 21, 28]. Для достижения субпиксельной точности при сопоставлении изображений были разработаны различные методы инкрементального уточнения [101].

Попиксельное сопоставление, теоретически может быть использовано при сопоставлении изображений подводного дна при условии, что в качестве неизвестного параметра связи выбрано только смещение одного изображения относительно другого, тогда как другие параметры берутся из навигационной

системы. На практике такой подход будет плохо работать, так как действительная модель связи подводных изображений – модель с шестью степенями свободы (3 степени вращения и 3 перемещения), что вносит в изображения искажения. Эти искажения невелики, ввиду слабых колебаний крена и дифферента аппарата при фотографировании и не слишком резкого изменения рельефа дна, однако достаточны, чтобы сделать попиксельное сопоставление проблематичным. Кроме того, монотонные снимки без характерных особенностей идеально сопоставляются попиксельно, не являясь при этом связанными. За счёт этого генерируется множество ложных связей.

Другой метод основан на особенностях – таких участках изображений, которые каким-либо образом выделяются в своей окрестности и обнаруживаются на различных изображениях одного объекта, полученных с разных точек в различных условиях съемки (обладают свойством повторяемости). Обычно под особенностями подразумевают особые точки, однако в качестве особенностей могут выступать и сегменты линий [113, 26], инвариантные к аффинным преобразованиям области [104], максимально устойчивые зоны [80] и др.

Использовать особенности для сопоставления начали порядка сорока лет назад [55], на протяжении этого времени было разработано немало способов выделения особенностей на изображениях [92, 56, 103]. Важным этапом развития детекторов особенностей стало появление полностью масштабно инвариантных детекторов, где особенности искались в масштабной пирамиде изображений с помощью Difference of Gaussian (DOG) детектора [76] либо детектора углов Харриса [83, 103]. Более быстрый алгоритм поиска инвариантных к масштабу особенностей был предложен в [59]. Таким образом, современные методы позволяют обнаруживать особенности, инвариантные к изменению масштаба, положения и поворота.

Использование особенностей – разумный выбор для сопоставления подводных изображений. Поскольку подводные изображения отличаются своей

низкой чёткостью, простые детекторы, как [83, 103] будут давать много ложных особенностей, обусловленных шумом на изображениях. Многомасштабный DOG детектор – наилучшее решение для подводных снимков, с помощью него можно находить особенности любого масштаба, при этом шум будет автоматически отфильтрован путём выбора начального масштаба при поиске особенностей. Также необходимо учитывать низкую чёткость подводных изображений, что потребует использования низких порогов выявления особенностей.

Для сопоставления изображений необходима процедура сравнения особенностей, для того, чтобы особенности можно было сравнить, необходимо каждой особенности сопоставить некоторый вектор – дескриптор. Чем ниже размерность этого вектора, тем проще сопоставлять особенности, но тем хуже он будет эту особенность описывать. Необходимо найти компромисс, когда размер дескриптора достаточен для идентификации особенности и при этом не слишком большой для использования в операциях сравнения.

В самых простых случаях, когда сопоставляемые изображения отличаются плоским сдвигом, в качестве дескрипторов допустимо использовать яркости окрестности особой точки, а для оценки похожести дескрипторов использовать сумму квадратов разностей интенсивностей либо коэффициент нормализованной кросс-корреляции [82]. В [82] также показано, как достичь инвариантности к повороту, используя среднюю ориентацию градиента яркости, рассчитанную для малой окрестности особой точки. Чтобы обеспечить дескриптору инвариантность относительно масштаба, необходимо находить особые точки, как экстремумы масштабного представления изображения [76, 83, 59].

Наиболее надёжные на данный момент дескрипторы – SIFT дескрипторы [76] и их различные модификации: SIFT на основе анализа главных компонент PCA-SIFT [72], SIFT, использующий информацию о цвете [18, 16]. Дескрипторы SIFT строятся на основе гистограмм ориентаций градиентов

окрестности особых точек и инвариантны к сдвигу, плоскому повороту, масштабу, линейному изменению освещённости. Также дескрипторы SIFT частично инвариантны к аффинным искажениям и нелинейному изменению освещения. В [110] показан алгоритм сопоставления особенностей на основе SIFT дескрипторов, полностью инвариантный к аффинным искажениям. Однако время работы этого алгоритма в разы больше, чем у алгоритма из [76], поэтом, в случае, когда между изображениями не предполагается сильных аффинных искажений, что бывает при построении фотографических карт дна, его использование не оправдано. Кроме SIFT, популярными современными дескрипторами особенностей являются SURF дескрипторы [59], которые значительно уступают SIFT дескрипторам в надёжности сопоставления, однако существенно выигрывают по скорости построения. Оптимальным для подводных изображений представляется использование SIFT дескрипторов.

Для сравнения SIFT дескрипторов обычно используют евклидово расстояние. Самым простым способом сравнения особенностей одного изображения с особенностями другого является сравнение всех возможных пар особенностей и отбрасывании пар со слишком большой разницей. Сравнение подводных изображений потребует использования новых подходов, основанных на быстром поиске в высокоразмерных пространствах. Большое количество изображений, используемое для составления фотографических карт подводного дна, совместно с большой похожестью различных визуальных особенностей, требуют особо внимательного подхода к решению задачи сравнения изображений.

Нелинейное изменение освещения на подводных снимках, обусловленное использованием осветительных приборов, установленных на АНПА, может вносить существенные искажения в SIFT дескрипторы. Поскольку SIFT дескрипторы чувствительны к таким искажениям, необходимо предусмотреть процедуру коррекции освещения каждого снимка.

Обычно среди сопоставленных особенностей присутствует некоторый процент ложных особенностей, поэтому вычисление параметров, описывающих предполагаемое движение между сопоставляемыми изображениями, с использованием метода наименьших квадратов может давать неверные результаты. Современный подход к сопоставлению требует сначала отфильтровать ложные соответствия. Для этого обычно используется процедура RANSAC (RANdom SAmple Consensus) [43] или LMS (least median of squares) [90]. Идеи процедур похожи. Вначале рассчитывается минимальное число k соответствий, необходимых для расчета выбранной модели движения. Затем N раз повторяется процедура извлечения k случайных неповторяющихся соответствий из общего набора соответствий. Для каждой извлечённой группы соответствий рассчитывается модель движения. В процедуре RANSAC вычисляется количество соответствий, удовлетворяющих рассчитанной модели с некоторой выбранной точностью, а в процедуре LMS рассчитывается медиана в массиве ошибок, с которыми соответствия удовлетворяют модели. По итогам N выборок верной признаётся модель, которой удовлетворило максимальное число соответствий в процедуре RANSAC, или модель, давшая наименьшую медиану ошибок в LMS. Варьируя число N можно увеличивать или уменьшать надёжность рассчитанной модели. Все соответствия, удовлетворившие победившей модели в RANSAC или LMS, признаются истинными, а модель пересчитывается с использованием всех истинных соответствий. Если число истинных соответствий по отношению к неистинным оказалось мало, то обычно делается вывод о том, что изображения не связаны.

Решение о том, связаны изображения или нет, принимается либо после процедур RANSAC, LMS, либо после процедуры сопоставления особенностей. Для несвязанных изображений процедура сопоставления даст мало соответствий. В случае, когда число этих соответствий не превышает минимального числа соответствий, необходимого для расчета модели

движения, хотя бы в два раза, вывод об отсутствии связи можно сделать до процедуры фильтрации ложных соответствий.

Процедура фильтрации ложных соответствий крайне важна для сопоставления подводных изображений, так как для подводных изображений процент ложных соответствий будет очень высок (до 90% для изображений с малой зоной пересечения). В таких условиях необходимы иные надёжные методы фильтрации особенностей, так как процедура RANSAC и LMS эффективны лишь, когда количество ложных соответствий не превышает 50%.

1.2.2. Калибровка камер. Калибровка камеры – это процесс вычисления параметров этой камеры. Бывают внутренние и внешние параметры камеры. Внешние параметры определяются ориентацией и положением камеры, и их определение будет рассмотрено в разделе 1.2.3. Внутренние параметры определяются устройством самой камеры и не зависят от окружающей сцены. Говоря о внутренних параметрах камеры, обычно имеют в виду матрицу калибровки камеры K [10]:

$$K = \begin{pmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}.$$

Здесь f_x, f_y – фокусные расстояния камеры, измеренные в ширинах и высотах пикселя соответственно, c_x, c_y – координаты принципиальной точки (точки пересечения оптической оси камеры и плоскости проекции), а $s = f_y \cdot \tan \alpha$, где α – угол наклона, образующейся между вертикальной стороной пикселя и вертикальной осью плоскости проекции.

Существует несколько способов вычисления внутренних параметров камеры. Один из них – использование изображений со специальными калибровочными объектами. Если есть возможность получить снимок с известными расстояниями от камеры до калибровочных объектов, можно воспользоваться методами [95, 23, 74]. Или воспользоваться изображением разлинованного картона [13]. Другой подход заключается в контролируемом

перемещении плоских калибровочных объектов в пространстве перед камерой [52, 45, 53, 111]. Если на изображении присутствует много объектов искусственного происхождения, имеющих ровные параллельные грани, то пересечения линий, соответствующих параллельным сторонам таких граней, на изображении дадут положения точек на бесконечности, которые можно использовать для вычисления внутренних параметров камеры [22, 87].

Другой способ заключается в вычислении параметров внутренней калибровки из построенной проективной реконструкции. Однако такой способ требует, чтобы было достаточно изображений, имеющих проекции общих точек [57], поэтому применимы в основном в задачах трёхмерной реконструкции, а не сшивки изображений.

Вычисление калибровки также возможно параллельно с вычислением положения и ориентации камер в общей задаче оптимизации [84], но для хорошего решения также требуется большое число изображений с общими точками.

Подходом, обеспечивающим наибольшую стабильность, является расчёт калибровки отдельно с вычислением положения и ориентации камеры. Это позволяет уменьшить количество степеней свободы и без того сложной задачи. При получении подводных изображений с помощью АНПА все фотографии получаются одной камерой с одинаковыми внутренними параметрами, разумно рассчитать эти параметры одним из приведённых методов заранее и использовать как известные в задаче сшивке. Иногда изготовитель камеры декларирует её внутренние параметры, что ещё больше упрощает задачу. В любом случае процедура калибровки камеры АНПА ничем не отличается от калибровки любой другой камеры, поэтому в настоящей работе не исследуется. В дальнейшем будем использовать упрощённый вариант матрицы калибровки, для этого предположим, что принципиальная точка находится в центре изображения, при этом центр изображения примем за начало его системы

координат. Пиксели изображения положим квадратными, то есть $f_x = f_y = f$, а $s = 0$. Тогда матрица внутренней калибровки примет простой вид:

$$\mathbf{K} = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}. \quad (1)$$

Кроме описанных линейных параметров камеры в процессе калибровки могут быть вычислены нелинейные параметры – коэффициенты радиальной дисторсии. Как и линейные параметры, коэффициенты радиальной дисторсии могут быть рассчитаны совместно с положением и ориентацией камер [91], а могут быть получены отдельно с помощью методов, основанных на изображениях прямых линий [31, 71, 41]. Если коэффициенты радиальной дисторсии для камеры АНПА значительны, то на вход системы сшивки следует подавать уже скорректированные изображения, определив коэффициенты дисторсии заранее. В конечном итоге для сшивки подводных изображений по-прежнему может быть использована матрица (1), которая будет справедлива для скорректированных изображений.

1.2.3. Вычисление положения и ориентации камер. Чтобы сшить изображения, необходимо точно знать положения и ориентации камер в момент получения каждого изображения. При этом не обязательно знать глобальные положения и ориентации, достаточно вычислить положения и ориентации камер относительно друг друга. В зависимости от вида движения между кадрами положение или ориентация могут быть частично или полностью известны заранее.

Существует два основных подхода к решению задачи об относительном позиционировании и ориентировании камер. Первый заключается в последовательном добавлении каждого нового изображения к группе изображений, положения и ориентации которых уже известны [100, 91]. При этом положение и ориентация нового изображения может рассчитываться относительно уже имеющихся изображений с использованием линейной или нелинейной схемы метода наименьших квадратов [57], в зависимости от того,

какая модель для описания движения между снимками выбрана. Недостаток такого подхода заключается в накапливании численной ошибки с каждым новым снимком. Если для составления сферической панорамы, когда используется немного изображений, и при круговом замыкании панорамы, возникший в результате ошибки, разрыв может быть «закрыт» [100], то при построении фотографических карт подводного дна из большого числа изображений такой метод оказывается неприемлем. Кроме того сложность задачи при последовательном добавлении снимков может расти, так как в результате добавления новых снимков может потребоваться коррекция ориентаций и положений предыдущих снимков. Для десятков тысяч подводных изображений проведение периодических пересчётов – слишком затратная по времени операция.

Другой метод заключается в одновременном вычислении ориентаций и положений всех камер. Наряду с положениями и ориентациями камер часто вычисляются точные координаты особенностей при сшивке панорам или положения точек пространства в задачах трёхмерной реконструкции. Этот метод получил название *bundle adjustment* и изначально использовался в фотограмметрии [102]. Применение данного метода для построения панорам описано в [93, 38]. Для использования *bundle adjustment* составляется функция общей ошибки, как правило, используется сумма квадратов расстояний между измеренными положениями особых точек и их положениями, получаемыми на основе рассчитанной модели движения. Далее решается задача минимизации ошибки относительно параметров модели движения. *Bundle adjustment* также предполагает решение относительно параметров – трёхмерных координат точек, проекции которых на изображения были найдены [57]. Несмотря на то, что в итоговой задаче минимизации участвует большое число параметров: параметры всех камер и трёхмерные координаты точек, специфическая структура задачи приводит к разреженной структуре матрицы системы уравнений, что позволяет эффективно решать эту задачу [57]. *Bundle adjustment*

применяют для больших задач, состоящих из тысяч изображений и миллионов точек. Данный подход также может быть применён к подводным изображениям. Поскольку десятки тысяч изображений – много даже для bundle adjustment, потребуется извлечь максимум пользы из всех видов разреженности задачи сшивки подводных изображений. Кроме того, для начала работы bundle adjustment потребуется начальная инициализация решения. Алгоритм получения такого решения специфичен для рассматриваемой задачи, поэтому должен быть разработан.

1.2.4. Проекция изображений на одну поверхность. Чтобы из отдельных изображений получить готовую фотографическую карту необходимо спроецировать все изображения на одну поверхность. Выбор способа проецирования специфичен для каждой конкретной задачи. В компьютерном зрении изучены разные способы проецирования для построения различных панорам. Например, для панорам, полученных из одной точки (то есть движение между кадрами определяется исключительно вращением) выбирается сферическая [100] или цилиндрическая проекция [36, 99]. Часто для удобства отображения сферических панорам в качестве поверхности для проецирования берут куб [50, 100], где каждое изображение проецируется на одну или несколько сторон куба, с центром в точке наблюдения. Также для проецирования панорам можно воспользоваться одним из методов, пришедшим из картографии [33]. Или более сложными локально адаптированными проекциями [34, 75]. Фотографическая карта дна – проекция сложного трёхмерного объекта на плоскость и к ней не применимы стандартные методы, которые работают для панорам. Чтобы получать фотографическую карту без искажений, вызванных параллаксом, необходимо разработать собственный алгоритм проецирования отдельных изображений подводного дна на единую плоскость.

Другой задачей, возникающей при проецировании изображений на выбранную поверхность, является корректное масштабирование. Часто

рассчитываемая карта имеет меньшее разрешение, чем совокупность исходных изображений, поэтому прямое проецирование пикселя исходного изображения на пиксель поверхности приведёт к выборочной потере мелких деталей. Чтобы избежать этого, существуют техники, основанные на фильтрации исходных изображений. Например, пирамидальное фильтрование [106], фильтр Гаусса [51], кубическая интерполяция в сочетании с адаптивной фильтрацией [86]. Здесь принципиальных отличий фотографической карты дна от любых стандартных видов панорам не наблюдается.

1.2.5. Совмещение изображений. Этап совмещения изображений – заключительный этап в формировании панорамы. На этом этапе решаются задачи поиска оптимального шва для сшивки, удаления движущихся объектов и смещения цветов пикселей в зонах перекрытия.

Самым простым способом совмещения изображений является усреднение цветов перекрывающихся пикселей. Очевидно, что при такой сшивке движущиеся объекты в зоне перекрытия станут полупрозрачными, а нестыковки на границах изображения будут видны. Самой простой техникой удаления движущихся объектов является выбор не среднего значения пикселя, а медианного [70], что также не устраняет нестыковки на границах изображений.

Более интересной техникой является взвешивание пикселей всех изображений из такого расчёта, что центральные пиксели оказываются самыми тяжелыми, а пиксели, близкие к краям наиболее лёгкими. Т.е. вес каждого пикселя определяется его расстоянием до ближайшей границы изображения [35, 105]. Такой способ взвешивания решает проблему нестыковки краёв, но оставляет проблему двигающихся объектов и смещения цветов. Если веса, определённые как расстояния до краёв изображений возвести в некоторую большую степень, то можно добиться практически полного устранения двигающихся объектов и размытия в зонах перекрытия. Данный метод

отличается своей скоростью и может хорошо подходить для совмещения подводных изображений.

Алгоритм для удаления движущихся объектов, основанный на выделении и удалении регионов различия (RODs – regions of difference) был предложен в [105]. Если необходимо удалять регионы различия с изображений, где присутствует сам движущийся объект, а не фон, то можно оценить разницу пикселей на границах регионов. Для региона различия изображения, содержащего движущийся объект, эта разница будет больше, чем для изображения, где этот объект не виден [60]. Подходы, основанные на удалении движущихся объектов часто неактуальны для подводных изображений. В действительности, вероятность попадания движущегося объекта между АНПА и дном маловероятно, что делает дополнительные вычислительные затраты на удаления таких объектов неоправданными.

Другой подход к вычислению шва и выбору пикселей в зоне перекрытия заключается в минимизации энергии [69]. Энергия строится на основе двух штрафов: штрафа на выбор какого-либо изображения для каждого пикселя и штраф на различие пикселей в области шва. Шов для сшивки может также выбираться вдоль хорошо выраженных краёв в зоне пересечения изображений [96]. Данный подход может быть справедлив для подводных изображений, однако он проигрывает в скорости подходу, основанному на убывании весов каждого пикселя от центра к краю изображения. Между тем, для большого количества изображений логично выбирать более быстрые методы.

1.3. Применение существующих систем сшивки к подводным изображениям

На сегодняшний день существует достаточно много приложений для сшивки изображений, разработанных как большими компаниями, так и одиночными исследователями. Все данные приложения ориентированы на бытовое использование. Как правило, предполагают, что все снимки получены из одной точки, одной камерой или являются кусочками плоского изображения.

Рассмотрим несколько наиболее известных систем для автоматической сшивки изображений: AutoStitch [64] (Autopano Pro [107], Serif Panorama Plus [109] и Calico [108], созданные на базе AutoStitch), Microsoft Research Image Composite Editor [63], разработанные на базе системы Panorama Tools[62]: Hugin – Panorama photo stitcher [61] и PTgui [65].

Все перечисленные программы сшивки изображений прекрасно справляются с задачей составления стандартных панорам, однако попытка применить их к сшивке нескольких снимков подводного дна, представленных на рис. привела к полному провалу. Ни одна программа не обнаружила связей между этими изображениями.



Рис. 5. Пример фотографий морского дна.

Причина кроется в сложных условиях подводной съемки, в результате которых редко удаётся получить изображения с явно выраженными особенностями. Большинство программ сшивки изображений позволяют вручную задавать особенности, но для подводных снимков человеку очень сложно определить соответствия. Только чтобы понять, как связана пара снимков, в зависимости от области перекрытия у человека уходит от нескольких секунд до минуты. А поскольку типичная серия изображений подводного дна может содержать несколько десятков тысяч изображений, ручное вмешательство в процесс построения панорамы становится неприемлемым.

Кроме сложности сопоставления изображений морского дна, движение камеры в процессе съемки сильно отличается от движения, используемого при построении бытовых сферических, цилиндрических или плоских панорам. Это также делает невозможным использование стандартных средства для составления фотографических карт подводного дна.

1.4. Основные задачи, требующие решения

В качестве вывода, перечислим основные задачи, для которых нет стандартных методов решения, и которые требуется решить с учётом специфики подводных изображений:

1. устранение неравномерного освещения, вызванного осветительными приборами АНПА;
2. обнаружение особенностей на слабоконтрастных изображениях;
3. надёжное сопоставление пар изображений, учитывающее большой процент ложных соответствий и использующее дополнительные данные, предоставляемые навигационной системой АНПА;
4. быстрый поиск связанных изображений в наборе из большого числа изображений с ненадёжными особенностями;
5. поиск приближённого положения и ориентации снимков для инициализации нелинейной задачи глобальной оптимизации;
6. эффективное решение глобальной задачи нелинейной минимизации для большого числа параметров;
7. сшивка изображений, максимально уменьшающая нестыковки вызванные эффектом параллакса.

Решение всех приведённых выше задач позволит создать программное средство, пригодное для построения фотографических карт подводного дна из десятков тысяч изображений, получаемых с АНПА в тяжёлых условиях съемки.

ГЛАВА 2. ПОИСК СВЯЗАННЫХ ИЗОБРАЖЕНИЙ

2.1. Постановка задачи

Будем считать, что после выполнения миссии АНПА было получено m изображений подводного дна. При этом из навигационной системы АНПА известны: глубина d_i , курс γ_i , высота над дном h_i в момент получения каждого i -го снимка, $i=1, 2, \dots, m$ (рис. 6).

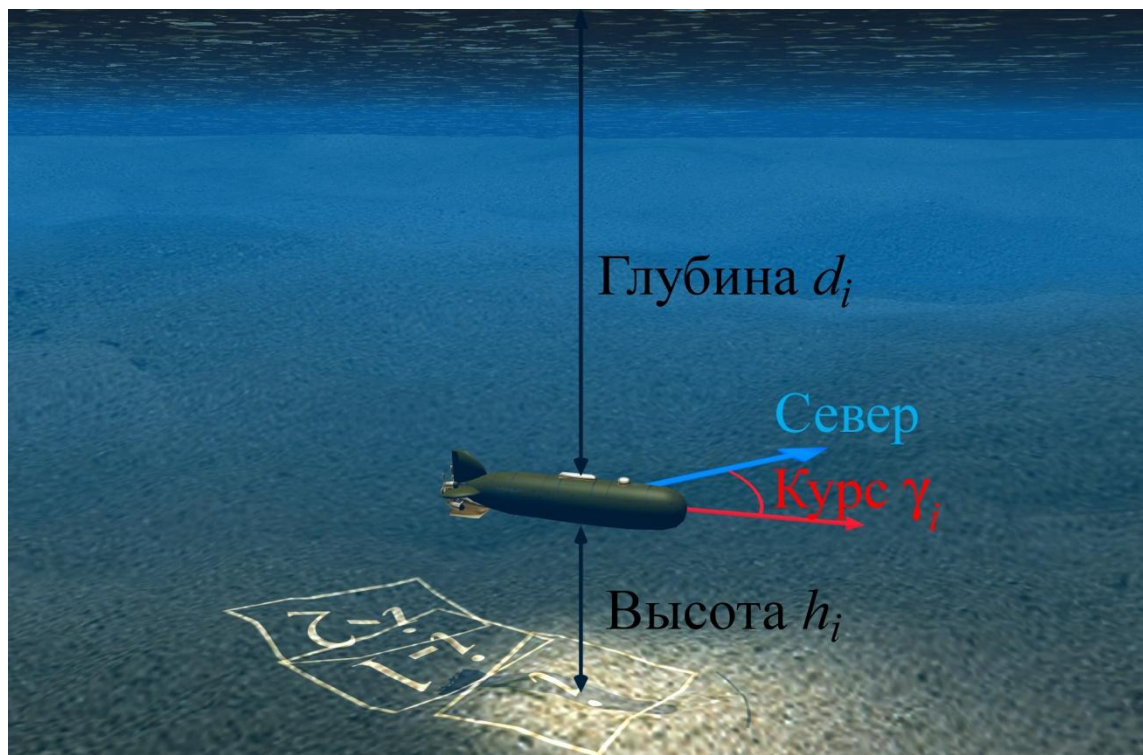


Рис. 6. Данные навигационной системы АНПА.

Необходимо на каждом изображении найти визуальные особенности – точки, выделяющиеся в своей окрестности и имеющие высокую вероятность быть обнаруженными в различных условиях съемки. Пусть на i -ом изображении имеется $\lambda(i)$ точек с координатами $\mathbf{u}_k^i = (u_k^i, v_k^i)$, $k=1, 2, \dots, \lambda(i)$, а в соответствие каждой точке поставлен дескриптор. Тогда задачу поиска связи между двумя изображениями $j, l \in \{1, 2, \dots, m\}$ можно описать, как задачу поиска векторов $\mathbf{p}_n^{jl} = (p_{n1}^{jl}, p_{n2}^{jl})$, таких, что особые точки $\mathbf{u}_{p_{n1}^{jl}}^j$ и $\mathbf{u}_{p_{n2}^{jl}}^l$ являются соответствующими (проекциями одной и той же точки пространства на изображения j и l). Здесь n – номер соответствия для пары изображений j, l .

Если принять общее число соответствий, найденных между изображениями j, l за $\lambda(j, l)$, то $n=1, 2, \dots, \lambda(j, l)$, при этом $p_{n1}^{jl} \neq p_{k1}^{jl}$ и $p_{n2}^{jl} \neq p_{k2}^{jl}$, где $k=1, 2, \dots, \lambda(j, l)$, $k \neq n$. Другими словами, одной точке на изображении j может соответствовать только одна точка на изображении l и наоборот, одной точке на изображении l может соответствовать только одна точка на изображении j .

Если два изображения l и j пересекаются и имеют общие особые точки в зоне пересечения, т.е. $\lambda(j, l) > 0$, то эти изображения считаются связанными, иначе, если $\lambda(j, l) = 0$, изображения l и j считаются несвязанными. Задача поиска связей среди всех изображений состоит в установлении таких пар изображений j, l , которые связаны и в вычислении для них векторов $\mathbf{p}_n^{jl}$.

Решая проблему связи изображений, необходимо учитывать специфику задачи, описанную в разделе 1.1.3. А именно, нелинейные изменения освещённости при фотографировании одной поверхности с разных положений, низкую контрастность снимков, большое количество однотипных особенностей.

2.2. Система координат изображения

Прежде чем находить особые точки, необходимо определиться с системой координат, в которой эти точки лучше представлять. Координаты особых точек выражаются в системе координат, связанной с изображением, на котором эти точки были обнаружены. Для хранения и отображения изображений в компьютерах часто используется система координат, представленная на рис. 7а.

Стандартную систему координат изображения составляют две перпендикулярные оси: X и Y . Ось X параллельна горизонтальным сторонам изображения, а Y – вертикальным. При этом ось X направлена вправо, а ось Y вниз. Началом системы координат является левый верхний угол изображения. Протяженность изображения по оси X составляет w пикселей, а по оси Y – h пикселей.

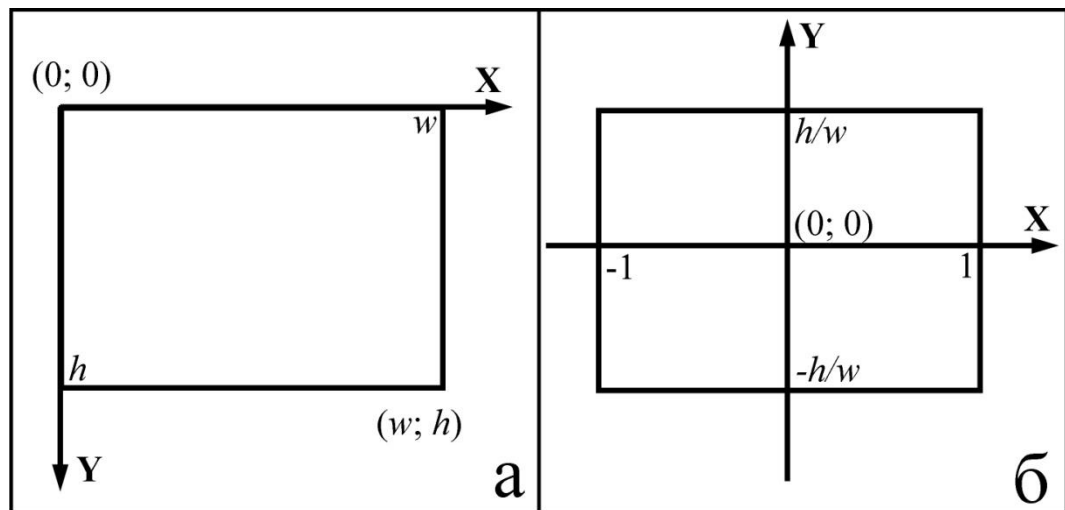


Рис. 7. Системы координат изображений.

Представленная на рис. а стандартная система координат не подходит для рассматриваемой задачи. Основной причиной является то, что для использования упрощённого вида матрицы внутренней калибровки камеры **К** (1) необходимо, чтобы центр системы координат изображения совпадал с положением принципиальной точки. У стандартных камер принципиальная точка находится в центре изображения, поэтому необходимо использовать такую систему координат, в которой начало координат находится в центре изображения. Пример такой системы координат представлен на рис. б.

В последующих разделах многие величины, например ошибки, будут измеряться в пространстве изображений. Поскольку ширина и высота каждого изображения w и h могут быть различными, одно и то же значение ошибки, измеренное в пикселях, может оказаться большим для маленького изображения и маленьким для большого. Чтобы избежать такой двусмысленности будем использовать нормализованную систему координат. В такой системе координат ширина любого изображения будет равняться двум (от -1 до 1), а высота $2h/w$ (от $-h/w$ до h/w), как показано на рис. 7б. Использование такой системы координат будет предполагаться в дальнейшем.

2.3. Визуальные особенности на изображениях

Различные подходы к обнаружению визуальных особенностей рассматривались в разделе 1.2.1. В частности, были упомянуты инвариантные к перемещению, плоскому повороту, линейному изменению освещения и масштабу SIFT особенности и дескрипторы [76]. Именно SIFT особенности, как наиболее надёжные будем использовать для рассматриваемой задачи.

2.3.1. Компенсация освещения АНПА. Так как для описания особенностей предполагается использовать SIFT дескрипторы, необходимо учитывать, что данные дескрипторы состоят из гистограмм градиентов яркостей пикселей окрестности особой точки, распределённых по ориентациям (раздел 2.3.3.). Если внимательно рассмотреть два снимка, представленные на рис. 8, то можно заметить, что они имеют пересечение.

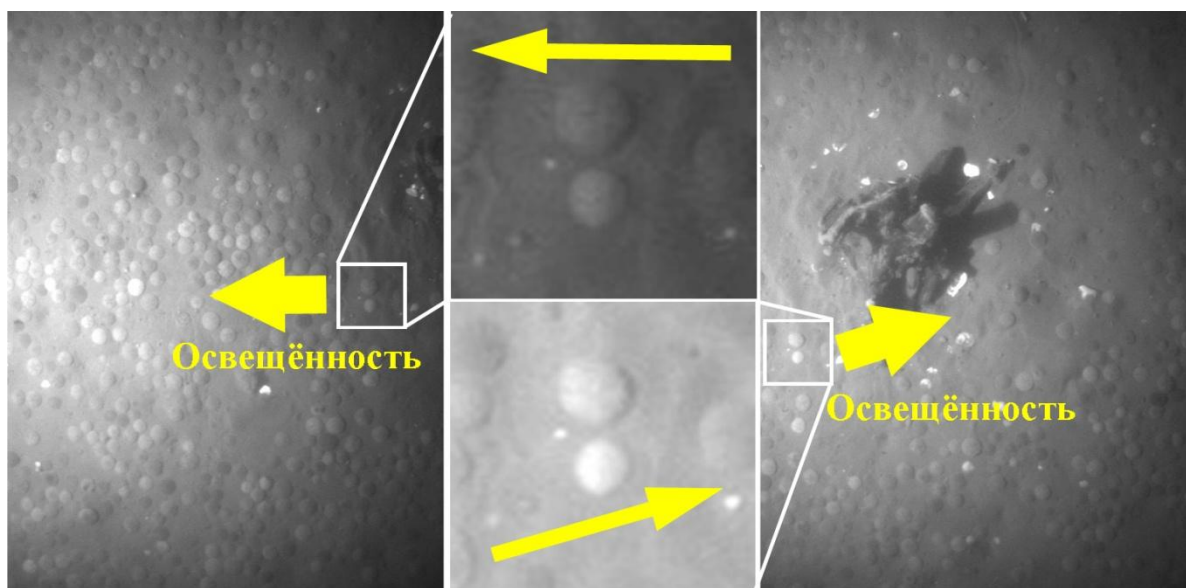


Рис. 8. Освещённость локальных участков изображений.

При этом вектор роста освещённости (изображён стрелками на рис.) в зоне пересечения снимков направлен в разные стороны. Таким образом, на градиенты яркостей пикселей окрестностей всех особенностей в зоне пересечения снимков будут влиять различные векторы освещённости для каждого из двух снимков. Это приведёт к тому, что дескрипторы соответствующих особенностей окажутся непохожими и не позволят

сопоставить особенности. В связи с этим, перед извлечением особенностей и составлением их дескрипторов необходимо скорректировать освещённость на снимках.

Чтобы получить цвет равномерно освещённого дна, необходимо из изображения выделить компоненту освещения, а затем, учитывая данную компоненту пересчитать яркость каждого пикселя. Поскольку имеется необходимость работать с яркостной составляющей изображения, следует перевести изображение в цветовую модель YUV [73]. Компонента $Y(x, y)$ обозначает яркость пикселя с координатами x, y . Компоненты $U(x, y)$ и $V(x, y)$ называются разностными компонентами и используются для хранения информации о цвете.

Как правило, изменения яркости изображения $Y(x, y)$, обусловленные формой объектов имеют высокую и среднюю частоту. Низкочастотное изменение яркости определяется влиянием освещения. Отфильтровав высокие и средние частоты яркостной составляющей изображения, получим карту освещённости $L(x, y)$. Для того чтобы отфильтровать низкие и средние частоты хорошо подойдёт фильтр Гаусса с большим значением σ (2).

$$L(x, y) = G(x, y, \sigma) * Y(x, y). \quad (2)$$

Здесь $*$ – операция свёртки, а $G(x, y, \sigma)$ – функция Гаусса (3).

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}. \quad (3)$$

На рис. 9а показано изображение подводного дна, сделанное АНПА и соответствующая этому изображению карта освещённости $L(x, y)$ с $\sigma=0.1$ (напомним, что используется нормализованная система координат), что составляет 0.05 от ширины изображения – рис. 9б.

Можно использовать различные стратегии коррекции изображения на основе его карты освещённости. Самый простой способ заключается в отнимании компоненты освещённости от яркостной компоненты (4):

$$R(x, y) = Y(x, y) - L(x, y). \quad (4)$$

Здесь $R(x, y)$ – искомое изображение равномерно освещённой поверхности дна. После расчёт по формуле (4) изображение $R(x, y)$ может содержать как отрицательные значения, так и значения, превышающие допустимый диапазон цветов. Чтобы это исправить, необходимо преобразовать $R(x, y)$ в соответствие со следующей формулой:

$$\bar{R}(x, y) = \frac{R(x, y) - \min_{x,y} R(x, y)}{\max_{x,y} R(x, y) - \min_{x,y} R(x, y)}.$$

Заменяв яркостный канал исходного изображения $Y(x, y)$ рассчитанным каналом $\bar{R}(x, y)$ и переведя изображение в цветовую модель RGB, получим изображение со скорректированным освещением – рис. 10а.

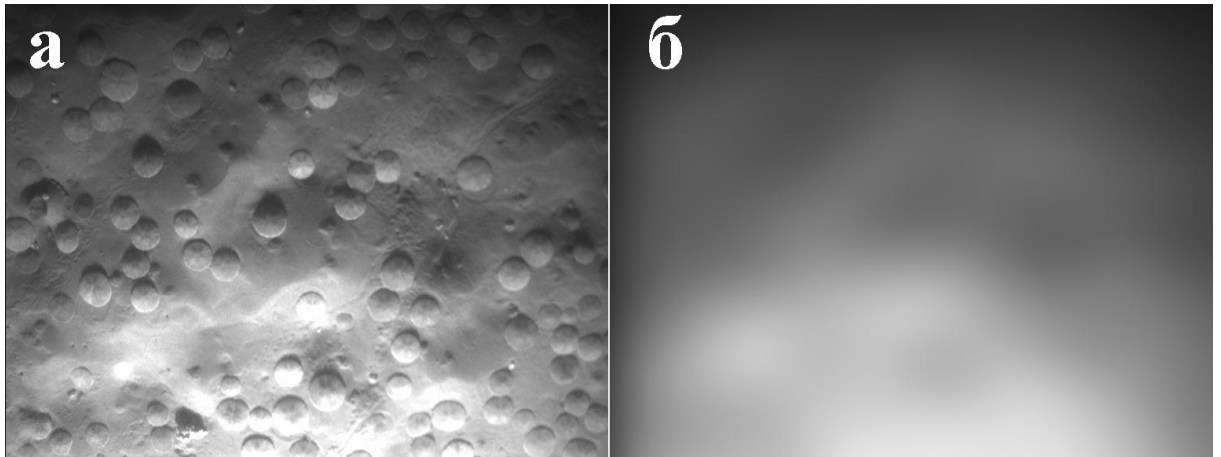


Рис. 9. Изображение и его карта освещённости $L(x, y)$.

Наиболее популярной стратегией коррекции освещения является логарифмическая коррекция (5).

$$R(x, y) = \ln \frac{Y(x, y)}{L(x, y)}. \quad (5)$$

Алгоритм подобной коррекции носит название Single Scale Retinex (SSR) [94, 112] и моделирует способность человека воспринимать цвет поверхности, независимо от освещения. В обработке изображений получила известность другая версия этого алгоритма – Multi Scale Retinex [14, 112], заключающаяся в нахождении нескольких карт освещённости $L(x, y)$, соответствующих различным параметрам σ в функции (3). При этом результирующее

изображение определяется как взвешенная сумма изображений $R(x, y)$ (5), рассчитанных для различных карт $L(x, y)$. В рассматриваемой задаче используется один источник света – АНПА, поэтому нет необходимости в компенсации освещений на разных масштабах и достаточно одной карты $L(x, y)$. Результатом алгоритмы SSR является карта, значения которой могут варьироваться от $-\infty$ до $+\infty$, поэтому $R(x, y)$ следует преобразовать. Если предположить, что допустимый диапазон значений компонент яркости – $[0; 1]$, то $R(x, y)$ можно преобразовать следующим образом:

$$\bar{R}(x, y) = R(x, y)g + 0.5.$$

Здесь g – коэффициент усиления, он выбирается исходя из требуемой контрастности результирующего изображения. В данной работе было выбрано $g = 0.5$. Значения $\bar{R}(x, y)$, не вошедшие в диапазон $[0; 1]$ просто отсекаются по этому диапазону. Результат работы алгоритма SSR представлен на рис. 10б.

В качестве ещё одной методики коррекции изображения можно предложить (6):

$$R(x, y) = \frac{Y(x, y)}{2L(x, y)}. \quad (6)$$

Данный метод не требует последующей коррекции изображений, за исключением простого отсечения значений $R(x, y)$ по диапазону $[0; 1]$. Результат работы метода (6) представлен на рис. 10в.

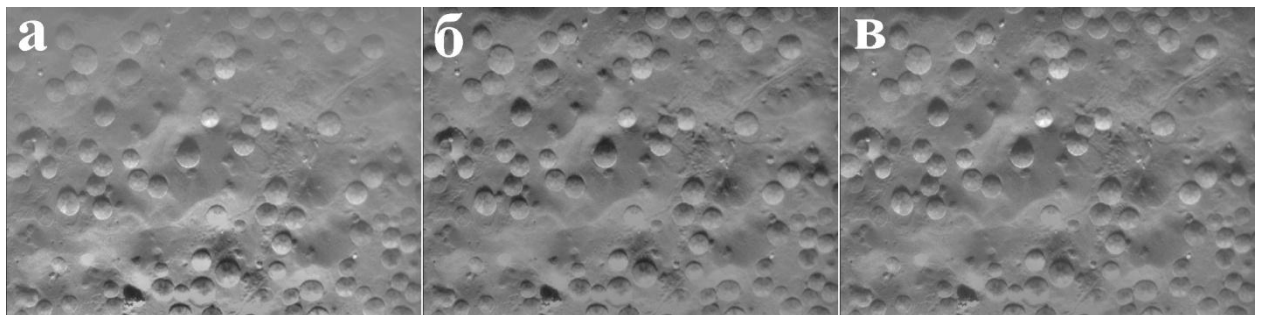


Рис. 10. Коррекция освещения: а - метод (4), б – метод (5), в – метод (6).

Визуально результат работы трёх методик трудно оценить. Все три убирают тени и высветленные участки, которые видны на исходном изображении – рис. а. Наименее привлекательным с точки зрения

человеческого восприятия выглядит метод (4), тогда как методы (5) и (6) визуально слабо различимы. Сравнение эффективности трёх описанных методов применительно к задаче извлечения и сопоставления особенностей приводится в разделе 2.6.2.

2.3.2. Обнаружение особых точек. Поиск особенностей на снимках, сделанных с АНПА, затруднён низким качеством изображений, получаемых в условиях недостаточной освещённости. На рисунке 11 изображена одна и та же зона снимка с разным приближением, на правом изображении хорошо виден шум, величина которого не уступает перепадам яркости, обусловленным объектами на дне. В таких условиях простые детекторы особенностей, как, например, детектор Харриса [56] будет давать большое количество ложных особенностей из-за шума. Реальные особенности будут теряться из-за отсутствия чётких контуров. Наиболее удачным решением для подобных снимков является многомасштабный DoG(Difference of Gaussians) детектор, используемый в [76].

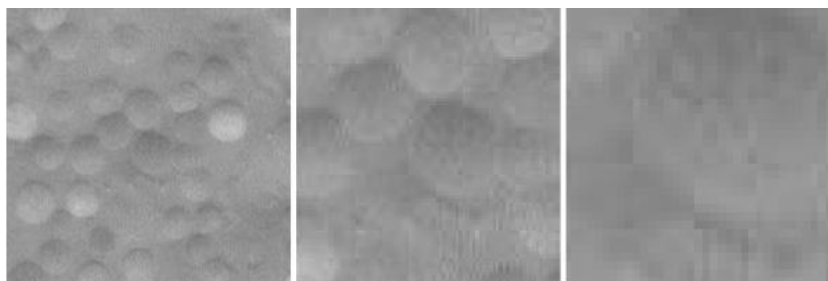


Рис. 11. Шумы на изображениях.

Пусть $G(x, y, \sigma)$ – функция Гаусса, определённая выше (3). Тогда через $L(x, y, \sigma)$ обозначим результат свёртки яркостной компоненты изображения $Y(x, y)$ с функцией $G(x, y, \sigma)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * Y(x, y).$$

Определим функцию разницы гауссианов, как $D(x, y, \sigma_1, \sigma_2)$ согласно формуле (7).

$$D(x, y, \sigma_1, \sigma_2) = L(x, y, \sigma_1) - L(x, y, \sigma_2). \quad (7)$$

Каждому масштабному слою n соответствует функция $D_n(x, y, \sigma_0)$ (8).

$$D_n(x, y, \sigma_0) = D(x, y, \sigma_0 k^n, \sigma_0 k^{n+1}). \quad (8)$$

Где σ_0 – параметр начального размытия по Гауссу, используется для устранения влияния шума на обнаружение особенностей. В [76] в качестве σ_0 рекомендуется выбрать значение 1.6. Коэффициент k – шаг между слоями изображения, чем меньше k , тем точнее масштабная пирамида, но тем больше слоёв необходимо рассчитать для охвата определенного диапазона масштабов. Оптимальными значениями для k являются значения от 1.2 до 1.5.

В оригинальном алгоритме [76] вместо единственной пирамиды, охватывающей все масштабы, предлагается строить серию из нескольких маленьких пирамид – октав, каждая из которых охватывает двукратное изменение масштаба. При этом разрешение изображений в каждой следующей октаве уменьшается вдвое. Такой подход позволяет существенно сократить объем вычислений в случае, когда необходимо охватить большой диапазон изменения масштаба. Однако при использовании этого подхода, точность обнаружения экстремумов с увеличением масштаба существенно падает, несмотря на процедуру уточнения, предложенную в [76]. Учитывая тот факт, что при сшивке подводных изображений разность масштаба между изображениями небольшая из-за ограниченности высоты, на которой способен производить съемку АНПА, а точность обнаружения особенностей играет ключевую роль, разбиение пирамиды на октавы в задаче обнаружения особенностей на подводных снимках нецелесообразно. Четырёхкратного масштаба достаточно, чтобы покрыть как разность масштабов между снимками, так и масштабы обнаружения наиболее важных особенностей. При этом проигрыш в скорости работы алгоритма без использования разбиения на октавы оказывается несущественным, а точность и простота алгоритма сохраняются.

$D_n(x, y, \sigma_0)$ – двумерная функция от координат изображения, при условии, что σ_0 константа, поэтому будем использовать сокращённую запись: $D_n(x, y, \sigma_0) = D_n(x, y)$. Все слои $D_n(x, y)$, $n=0, 1, \dots, N-1$, где N – число масштабных слоёв,

задают дискретную трёхмерную функцию, экстремумы которой определяют положение особых точек, координаты x и y экстремумов соответствуют координатам особой точки на изображении, а слой n – её приближительному масштабу. Количество слоёв N определяется диапазоном масштабов особенностей, который необходимо охватить. Если максимальный масштаб, на котором требуется обнаруживать особенности обозначить через $s_{\max}$, то необходимое количество слоёв можно определить так:

$$N = \left\lceil \frac{\log s_{\max}}{\log k} \right\rceil.$$

Поиск экстремумов осуществляется на слоях $n=1, 2, \dots, N-2$. Точка на слое $n_p \in \{1, 2, \dots, N-2\}$ с координатами u_p, v_p считается экстремумом, если значение функции $D_{n_p}(u_p, v_p)$ больше (меньше) 8-ми соседних значений на слое n_p , 9-ти значений на слое n_p+1 и 9-ти значений на слое n_p-1 . Модуль значения функции $D_{n_p}(u_p, v_p)$ показывает надёжность особой точки. Как правило, для особых точек устанавливается порог на минимально допустимое значение $|D_{n_p}(u_p, v_p)|$, точки со значением ниже порога отбрасываются.

Если экстремум обнаружен на n_p слое, то масштаб особенности в данном экстремуме может быть определён как первая координата вершины параболы, построенной по трём точкам: $(k^{n_p+j}, D_{n_p+j}(u_p, v_p))$, $j=-1, 0, 1$. Вторую координату вершины данной параболы можно использовать как характеристику стабильности экстремума.

В оригинальном алгоритме [76] предлагается в качестве особенностей рассматривать только экстремумы, стабильность которых выше 0.03, однако на многих снимках подводного дна экстремумы с такой высокой стабильностью могут вообще отсутствовать. В зависимости от высоты, с которой сделан конкретный снимок стабильность может быть разной, поэтому в данной работе предлагается выбрать порог стабильности низким, порядка 0.001, и вместе с тем ограничить максимальное количество экстремумов на изображении. Для

этого, после обнаружения всех экстремумов, они сортируются в порядке уменьшения стабильности и последние экстремумы, не вошедшие в допустимое количество, отсекаются. В зависимости от частоты получения снимков АНПА допустимое количество экстремумов может варьироваться от 500 до 1000 на снимок. Чем ниже частота съемки, тем меньше перекрытия изображений и тем больше особенностей необходимо для обнаружения связи.

Результатом применения процедуры поиска особых точек к i -ому изображению, $i=1, 2, \dots, m$ является набор из $\lambda(i)$ точек с координатами $\mathbf{u}_k^i = (u_k^i, v_k^i, s_k^i)$, $k=1, 2, \dots, \lambda(i)$, где дополнительная координата s_k^i - масштаб особенности.

2.3.3. Построение дескрипторов особых точек. Для сопоставления особенностей на изображениях необходимо каждой особенности поставить в соответствие некоторый дескриптор, который бы имел небольшой размер и позволял по возможности уникально идентифицировать особенность. Кроме того дескриптор должен быть устойчив к различным изменениям условий съемки.

Одним из наиболее удачных видов дескрипторов на сегодняшний момент остаются SIFT дескрипторы [76], кодирующие градиенты окрестности особенности, масштабированной в соответствии с масштабом, на котором она обнаружена, и повернутой на угол, соответствующий приоритетному направлению градиентов. Процесс составления данного дескриптора описан в [76].

Одним из слабых мест описанного в [76] алгоритма построения дескриптора является определение ориентации. Ориентация особенности очень чувствительна к самым маленьким изменениям окрестности особой точки. А ошибка в определении ориентации приводит к неправильному построению всего дескриптора. Кроме того, потенциальная возможность наличия множественных ориентаций у особенностей осложняет алгоритм и увеличивает количество особенностей. Преимуществом задачи сшивки изображений,

получаемых с АНПА, является известный курс АНПА в момент получения каждого изображения – γ . Это означает, что шаг определения ориентации из оригинального алгоритма [76] можно пропустить. Вместо приоритетной ориентации следует использовать угол $-\gamma$, что обеспечит инвариантность к плоскому повороту. Точность определения курса γ с помощью навигационной системы АНПА высока, поэтому использование курса аппарата вместо локальных ориентаций особенностей приведёт к упрощению алгоритма и к увеличению его надёжности. Также это избавит от необходимости учитывать множественные ориентации.

SIFT дескриптор особенности является вектором единичной длины, состоящим из 128 чисел с плавающей точкой: 4×4 ячейки по 8 ориентаций в каждой. Для представления чисел с плавающей точкой одинарной точности используется 4 байта памяти. Это означает, что дескриптор одной особенности займёт 512 байт памяти, а в 1 Гб памяти поместится около двух миллионов дескрипторов. Этого достаточно для представления информации о визуальных особенностях для 2-х – 4-х тысяч изображений. Чтобы уменьшить объем дескриптора предлагается для его представления использовать целые однобайтовые числа. В действительности, диапазона одного байта достаточно, чтобы хранить компоненты дескриптора с достаточной точностью.

Компоненты SIFT дескриптора – значения от 0 до 1, но на практике компоненты дескриптора не превышают значения 0.5, поэтому предлагается диапазон чисел с плавающей точкой от 0 до 0.5 отобразить на целочисленный диапазон от 0 до 255. Это даст 4-х кратный выигрыш в памяти и, практически, не повлияет на точность сопоставления.

2.4. Сопоставление особенностей на паре изображений

2.4.1. Оценки соответствия дескрипторов. Дескриптор SIFT можно представить в виде 128-ми мерного вектора. Расстояние между двумя дескрипторами $\mathbf{g}$ и $\mathbf{g}'$ определяется согласно формуле (9). Так как

используются дескрипторы с целочисленными однобайтовыми компонентами (раздел 2.3.3), итоговое расстояние $d(\mathbf{g}, \mathbf{g}')$ делится на 512 – коэффициент масштаба для перехода от дескрипторов с дробными компонентами к дескрипторам с целыми компонентами.

$$d(\mathbf{g}, \mathbf{g}') = \frac{\sqrt{(\mathbf{g} - \mathbf{g}')(\mathbf{g} - \mathbf{g}')}}{512}. \quad (9)$$

Учитывая тот факт, что исходные дескрипторы имели единичную длину, множество значений функции $d(\mathbf{g}, \mathbf{g}')$ (9) лежит в интервале $[0; \sqrt{2}]$. Эта функция является мерой похожести дескрипторов: чем значение $d(\mathbf{g}, \mathbf{g}')$ меньше, тем больше похожи дескрипторы. Можно ввести максимальный порог $d_{\max}$, на значение функции $d(\mathbf{g}, \mathbf{g}')$ для дескрипторов $\mathbf{g}$ и $\mathbf{g}'$, являющихся дескрипторами соответствующих особенностей.

Пусть есть множество всех дескрипторов G , в котором для дескриптора $\mathbf{g}$ были обнаружены два ближайших дескриптора $\mathbf{g}'$ и $\mathbf{g}''$, таких, что $d(\mathbf{g}, \mathbf{g}') < d(\mathbf{g}, \mathbf{g}'')$, $d(\mathbf{g}, \mathbf{g}'') < d(\mathbf{g}, \mathbf{q})$, $\mathbf{q} \in G - \{\mathbf{g}, \mathbf{g}', \mathbf{g}''\}$. Величина $r(\mathbf{g}, \mathbf{g}')$ (10) является оценкой надёжности соответствия дескрипторов $\mathbf{g}$ и $\mathbf{g}'$.

$$r(\mathbf{g}, \mathbf{g}') = \frac{d(\mathbf{g}, \mathbf{g}')}{d(\mathbf{g}, \mathbf{g}'')}. \quad (10)$$

Чем меньше величина $r(\mathbf{g}, \mathbf{g}')$, тем надёжнее можно считать соответствие особенностей с дескрипторами $\mathbf{g}$ и $\mathbf{g}'$. Близость $r(\mathbf{g}, \mathbf{g}')$ к единице означает, что дескриптор $\mathbf{g}$ похож на дескриптор $\mathbf{g}'$ почти так же, как и на дескриптор $\mathbf{g}''$. Поэтому целесообразно установить порог на максимальное значение $r(\mathbf{g}, \mathbf{g}')$ – $r_{\max}$, которое допустимо для дескрипторов соответствующих особенностей $\mathbf{g}$ и $\mathbf{g}'$.

2.4.2. Выявление соответствий. Рассмотрим два изображения с номерами $i, j \in \{1, 2, \dots, m\}, i \neq j$. На этих изображениях найдено $\lambda(i)$ и $\lambda(j)$ особых точек с дескрипторами $\mathbf{g}_k^i$ и $\mathbf{g}_l^j$, $k=1, 2, \dots, \lambda(i)$, $l=1, 2, \dots, \lambda(j)$. Процедуру выявления потенциальных соответствий можно разбить на два шага: сравнение всех дескрипторов и отсеивание повторяющихся пар.

Для сравнения всех дескрипторов будем для каждого $\mathbf{g}_k^i$ -го дескриптора перебирать все дескрипторы $\mathbf{g}_l^j$. В процессе перебора для каждого дескриптора $\mathbf{g}_k^i$ сохраним номер $b(k)$:

$$b(k) = \operatorname{argmin}_l d(\mathbf{g}_k^i, \mathbf{g}_l^j).$$

В результате для каждого k -го дескриптора i -го изображения будет найден $b(k)$ -й дескриптор на j -ом изображении. На основе этого можно составить список соответствующих особенностей $\mathbf{p}_k^{ij} = (k, b(k))$. Очевидно, что для разных значений k могут быть найдены одинаковые значения b . Этого не должно быть в действительности, так как одной особенности одного изображения может соответствовать только одна особенность другого изображения. Поэтому из списка $\mathbf{p}_k^{ij}$ необходимо исключить лишние элементы. Предположим, что в списке $\mathbf{p}_k^{ij}$ были найдены две пары номеров изображений: $\mathbf{p}_{k1}^{ij} = (k1, b(k1))$ и $\mathbf{p}_{k2}^{ij} = (k2, b(k2))$, $k1, k2 \in 1, 2, \dots, \lambda(i)$, для которых справедливо равенство $b(k1)=b(k2)$. Это означает, что только одну пару можно оставить – пару с минимальной оценкой: $d(\mathbf{g}_{k1}^i, \mathbf{g}_{b(k1)}^j)$ и $d(\mathbf{g}_{k2}^i, \mathbf{g}_{b(k2)}^j)$. Обработав все пары подобного вида, получим список из непротиворечиво составленных особенностей. Из оставшихся пар следует исключить пары, для которых $d(\mathbf{g}_k^i, \mathbf{g}_{b(k)}^j) > d_{max}$ или $r(\mathbf{g}_k^i, \mathbf{g}_{b(k)}^j) > r_{max}$. В результате список соответствий примет следующий вид: $\mathbf{p}_n^{ij} = (p_{n1}^{ij}, p_{n2}^{ij})$, $n=1, 2, \dots, \lambda(i, j)$, где $\lambda(i, j)$ – количество оставшихся пар, p_{n1}^{ij} – индекс особенности на i -ом изображении, соответствующей особенности с индексом p_{n2}^{ij} на j -ом изображении.

Чем меньше выбраны значения d_{max} и r_{max} , тем меньше ошибочных соответствий будет найдено. Поскольку для подводных изображений свойственна большая зашумлённость, дескрипторы соответствующих особенностей не будут слишком похожими, а ввиду большого числа однотипных объектов, надёжность соответствий также будет невысокой. В

связи с этим, выбор слишком малых значений для d_{max} и r_{max} приведёт к отсутствию как истинных, так и ложных соответствий на снимках. Наиболее подходящими для исследуемых изображений были следующие пороги: $d_{max}=0.5$, $r_{max}=0.8$. Такие значения порогов позволяют находить достаточно истинных соответствий при их наличии, но также дают много ложных – рис. 12.

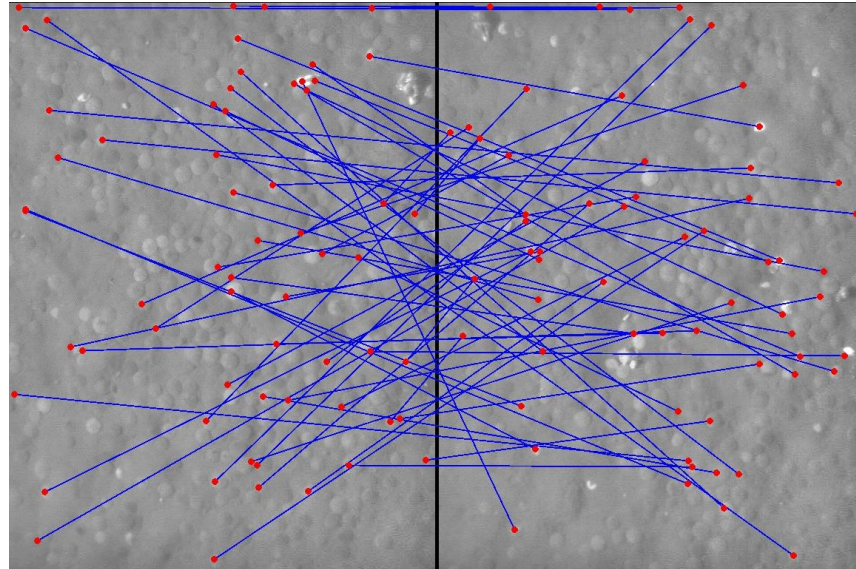


Рис. 12. Набор соответствий.

Количество ложных соответствий зависит от размера зоны пересечения снимков. В случае маленькой зоны пересечения, как на рис. 17, процент ложных соответствий может достигать до 90%.

2.4.3. Фильтрация ложных соответствий. Эффективным методом для отсеивания ложных соответствий является процедура RANSAC (Random Sample Consensus)[43], идея работы которой описана в разделе 1.2.1. Рассчитаем необходимое количество итераций процедуры RANSAC, которое необходимо совершать для вычисления корректной модели связи с вероятностью не менее $1-P_{false}$. Пусть $p_{true} = N_{true} / N$ – вероятность выбрать из набора всех соответствий истинное соответствие. Здесь N_{true} – количество истинных соответствий, а N – общее количество соответствий. В рассматриваемом случае, будем считать, что $0.1 \leq p_{true} \leq 1.0$. Нижняя граница $p_{true}=0.1$ (10% истинных соответствий) будет определять количество выборок K ,

которое необходимо совершить, чтобы получить верную модель с вероятностью ошибки не более P_{false} (11).

$$K = \frac{\log P_{false}}{\log(1 - (p_{true})^M)}. \quad (11)$$

Здесь M – минимальное количество соответствий, необходимых для вычисления модели. Если выбрать $P_{false} = 0.0001$, то для расчета связи с использованием стандартных моделей: фундаментальной матрицы [78, 11] ($M=7$) или гомографии [57] ($M=4$), потребуется $K \approx 10^8$ или $K=10^5$ выборок соответственно. Это слишком большие значения, поэтому в данной работе предлагается использовать специфические особенности задачи.

Поскольку известно, что для каждого кадра может быть достаточно точно определён курс и высота над дном, а движение АНПА в первом приближении можно считать плоским, предлагается использовать следующую модель отображения координат i -го изображения (u_i, v_i) в систему координат j -го изображения (u_j, v_j) (12):

$$\begin{aligned} u_j &= u_i s^{ij} \cos \gamma^{ij} - v_i s^{ij} \sin \gamma^{ij} + t_u^{ij} \\ v_j &= u_i s^{ij} \sin \gamma^{ij} + v_i s^{ij} \cos \gamma^{ij} + t_v^{ij} \end{aligned} \quad (12)$$

Здесь $s^{ij} = \frac{h_i}{h_j}$ – масштаб между изображениями, а $\gamma^{ij} = \gamma_i - \gamma_j$ – угол между изображениями. Вектор $t^{ij} = (t_u^{ij}, t_v^{ij})$ является неизвестным параметром модели. Чтобы найти две компоненты данного вектора, достаточно в (12) подставить одно точечное соответствие. Другими словами, минимальное число соответствий, необходимое для расчёта модели (12) $M=1$. Чтобы рассчитать верную модель с вероятностью ошибки $P_{false} = 0.0001$ нужно совершить всего 90 операций выборок. А поскольку количество соответствий на большинстве изображений не будет превышать 90, можно проверить все возможные соответствия.

После обнаружения наилучшей модели и всех согласующихся с ней соответствий можно пересчитать вектор $t^{ij} = (t_u^{ij}, t_v^{ij})$, используя все хорошие

соответствия. Также можно повторить процедуру сопоставления особенностей, используя в качестве вспомогательной структуры найденную модель. Так как она позволяет с определённой точностью сказать, где должна находиться особенность j -го изображения, соответствующая определённой особенности i -го изображения. Использование модели сужает круг поиска соответствующих особенностей от «каждой с каждой» до «каждой с каждой, входящей в зону погрешности модели». Это приводит к существенному ускорению и уточнению процедуры сопоставления, в результате которой, как правило, находятся новые корректные соответствия, которые не были обнаружены на первом этапе сопоставления. Результат повторного сопоставления с использованием рассчитанной модели для набора пар соответствий на рис. 12 представлен на рис. 13.

В результате отсеивания ложных соответствий мы получим некоторый набор данных, удовлетворяющий рассчитанной модели связи двух изображений. Но часто бывают ситуации, когда поиск производится на двух изображениях, которые не имеют пересечений, т.е. не связаны. Таким образом, следующим этапом, необходимо определить, верно ли наше предположение о наличии связи между изображениями.

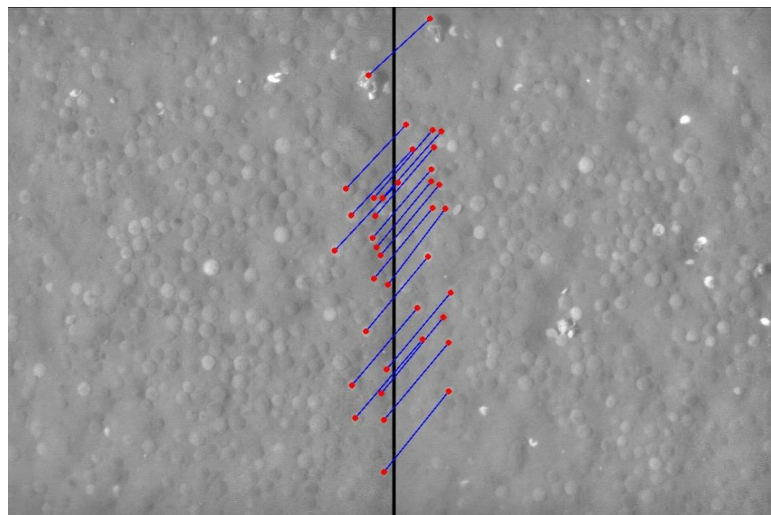


Рис. 13. Истинные соответствия.

Найдя наилучшую модель связи между изображениями можно определить число всех соответствий (включая ложные), находящихся в зоне

пересечения изображений. Обозначим данное число за n_f . Число соответствий, удовлетворяющих нашей модели, обозначим за n_i . Тогда можно считать, что наша модель является истинной, если выполняется условие (13).

$$n_i > \alpha + \beta n_f. \quad (13)$$

Здесь α и β - коэффициенты, которые выбираются практически и зависят от качества исходных данных, а именно от устойчивости особых точек изображений. Для данной работы было выбрано значение $\alpha=4.0$ и $\beta=0.3$. Подробнее о математическом смысле данных коэффициентов можно прочитать в работе [81].

2.5. Алгоритм быстрого поиска связанных изображений

2.5.1. Необходимость применения быстрых алгоритмов. Поиск связанных изображений – самый затратный по времени этап, так как необходимо проводить процедуру сопоставления для очень большого числа пар. Если с АНПА было получено m фотографий, то из них можно составить N пар (14) изображений.

$$N = \frac{m^2 - m}{2}. \quad (14)$$

Если с АНПА было получено 10000 изображений, то для обнаружения связанных пар потребуется выполнить $N \approx 5 \cdot 10^7$ процедур сопоставления. На современном компьютере процедура сопоставления, описанная в разделе 2.4, занимает порядка 0.25 секунд. Это означает, что сопоставление $5 \cdot 10^7$ пар потребуют более 140 суток, а для пар, составленных из 50000 изображений, потребуется уже около 10 лет. Чтобы решить задачу за приемлемое время, необходимо уменьшить количество сопоставляемых пар.

В основе процедуры сопоставления пар изображений лежит процесс сравнения дескрипторов особых точек. Чем больше дескрипторов одного изображения оказались близкими к дескрипторам другого, тем больше вероятность, что для пары будет определена связь. Таким образом, связь следует искать лишь для пар таких изображений, количество близких

дескрипторов особенностей между которыми больше, чем некоторое пороговое число.

2.5.2. Списки кандидатов на сопоставление. Обозначим дескриптор j -ой особенности i -го изображения за $\mathbf{g}_j^i$, $i=1, 2, \dots, m$, $j=1, 2, \dots, \lambda(i)$, где $\lambda(i)$ – количество особенностей на i -ом изображении. Предположим, что существует быстрый способ для дескриптора $\mathbf{g}_j^i$, найти вектор $\mathbf{l}_j^i$, состоящий из номеров изображений, содержащих первые k особенностей с дескрипторами, максимально близкими к дескриптору $\mathbf{g}_j^i$: $\mathbf{l}_j^i = (l_{j1}^i, l_{j2}^i, \dots, l_{jk}^i)$. Подробно этот способ описан в разделе 2.5.3.

Зная $\mathbf{l}_j^i$ можно составить списки предполагаемых кандидатов на сопоставление для каждого изображения. Для этого введём матрицу $\mathbf{W}=(w_{pb})$, $p, b=1, 2, \dots, m$, где w_{pb} – количество похожих дескрипторов между парой изображений p и b . Изначально, все w_{pb} приравниваются к нулю. Далее перебираются все элементы l_{jn}^i , $i=1, 2, \dots, m$, $j=1, 2, \dots, \lambda(i)$, $n=1, 2, \dots, k$ и компоненты матрицы $\mathbf{W}$: w_{i, l_{jn}^i} и $w_{l_{jn}^i, i}$ увеличиваются на единицу. В список предполагаемых кандидатов на сопоставление каждого изображения $\mathbf{S}_i=(s_1^i, s_2^i, \dots, s_\eta^i)$, $i=1, 2, \dots, m$ помещаются η номеров столбцов наибольших элементов из i -ой строки матрицы $\mathbf{W}$.

После вычисления списков $\mathbf{S}_i$, процедуру сопоставления нужно выполнять не для (14), а всего лишь для $N=\eta \cdot m$ пар. Каждое i -ое изображение сопоставляется лишь с изображениями из списка $\mathbf{S}_i$. Влияние параметров η и k на процесс поиска связанных изображений рассмотрен в разделе 2.6.4.

2.5.3. Поиск похожих дескрипторов особенностей. Ключом к быстрому обнаружению потенциально пересекающихся пар изображений является процедура быстрого поиска дескрипторов особенностей, близких к заданному дескриптору. В качестве дескрипторов будем рассматривать 128-мерные вектора – SIFT дескрипторы (раздел 2.3.3), а в качестве меры близости евклидово расстояние между этими векторами (9).

Запрос на поиск выполняется для некоторого дескриптора $\mathbf{q}$ в наборе дескрипторов $G=\{\mathbf{g}_i\}$, $i=1, 2, \dots, n$, где n – количество особенностей, найденных на изображениях:

$$n = \sum_{j=1}^m \lambda(j).$$

Результатом поиска будет k дескрипторов из G , максимально близких к $\mathbf{q}$. Основная проблема данного вида поиска – высокая размерность используемых дескрипторов.

Для поиска в высоко размерных пространствах необходимо организовать элементы множества G таким образом, чтобы возможно было построить способ поиска k дескрипторов максимально близких к $\mathbf{q}$, за время, быстрее линейного.

Можно перечислить следующие наиболее известные способы организации дескрипторов и методы поиска: kd-деревья [44], best bin first [27], локально чувствительно хеширование (LSH) [68], метрические деревья [37], sp-деревья (spill trees) и гибридные sp-деревья [20]. Согласно исследованиям, описанным в статье [20], время поиска в sp-деревьях и метрических деревьях в разы меньше времени поиска в других структурах. Недостатками sp-деревьев является большое количество требуемой памяти и неопределённая глубина дерева, тогда как метрическим деревьям нужно значительно меньше памяти, однако время поиска в них больше. Гибридные sp-деревья являются промежуточным вариантом между sp и метрическими деревьями, позволяющим выбирать между объемом потребляемой памяти и временем поиска.

Метрические и sp-деревья. Приведём описание процесса построения узлов метрического и sp-дерева из статьи [20]. Метрическое дерево – бинарное дерево, узлы которого представляют множества дескрипторов. При этом корневой узел представляет целиком множество G .

Рассмотрим внутренний узел дерева t . Множество точек, представляемое данным узлом $G(t)$, делится на два подмножества $G(t.lc)$ и $G(t.rc)$, представляемые двумя дочерними узлами t : $t.lc$ и $t.rc$. При этом

$G(t)=G(t.lc) \cup G(t.rc)$, также для метрического дерева $G(t.lc) \cap G(t.rc)=\emptyset$. Обратим внимание, что для *sr*-дерева, в отличие от метрического, последнее равенство не всегда справедливо. Узлы самого нижнего уровня содержат некоторое минимальное количество дескрипторов и далее не делятся.

Рассмотрим стратегию деления множества $G(t)$ на множества $G(t.lc)$ и $G(t.rc)$. Сначала необходимо выбрать в $G(t)$ два дескриптора $t.lpv$ и $t.rpv$, расстояние между которыми максимально: $|t.lpv - t.rpv| = \max_{a,b \in G(t)} |a - b|$. Выбор таких точек имеет вычислительную сложность $O(n^2)$, поэтому в качестве точек $t.lpv$ и $t.rpv$ будем брать точки не с максимальным, но с близким к максимальному расстоянием $|t.lpv - t.rpv|$. Для этого выберем случайным образом точку $p \in G(t)$, затем выберем точку $t.lpv$, как самую дальнюю от p , а точку $t.rpv$, как самую дальнюю от $t.lpv$.

Введём вектор $u = t.rpv - t.lpv$. Спроецируем все дескрипторы $G(t)$ на этот вектор. Далее найдём медианную точку A среди проекций. Через точку A построим гиперплоскость L , перпендикулярную вектору u . Плоскость L разделит множество дескрипторов $G(t)$, на два подмножества. Под плоскостью окажутся дескрипторы $G(t.lc)$, а над плоскостью $G(t.rc)$. Рис. 14а. демонстрирует процесс деления узла метрического дерева в двумерном пространстве.

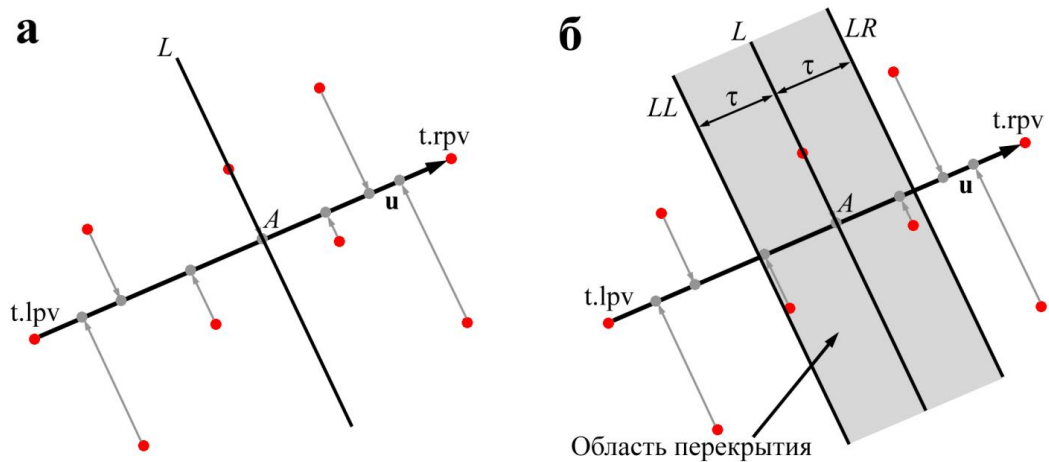


Рис. 14. а. деление узла метрического дерева, б. деление узла *sr*-дерева

Для осуществления процедуры поиска для каждого узла t также понадобится хранить гиперсферу $\mathbb{B}(t.\text{center}, t.r)$, где $t.\text{center}$ – центр гиперсферы, а $t.r$ – её радиус. Основное требование, предъявляемое к данной гиперсфере – это то, что она должна содержать внутри себя все дескрипторы из $G(t)$, при этом $t.r$ должен быть по возможности минимальным.

Процедура построения метрического и sr -дерева отличается способом деления множества $G(t)$ на множества $G(t.lc)$ и $G(t.rc)$. Процесс деления узла в sr -дереве повторяет аналогичный процесс в метрическом дереве до момента вычисления делящей гиперплоскости L . После вычисления L строятся две дополнительные гиперплоскости, параллельные L : LL на расстояние τ от L в направлении, противоположном $\mathbf{u}$ и LR на расстоянии τ в направлении $\mathbf{u}$. Дополнительные гиперплоскости заключают между собой общее для узла $t.lc$ и $t.rc$ пространство дескрипторов. Все дескрипторы, заключенные между LL и LR попадают одновременно и в $G(t.lc)$ и в $G(t.rc)$: $G(t.lc) = \{\mathbf{x} | \mathbf{x} \in G(t), d(\mathbf{x}, LR) + 2\tau > d(\mathbf{x}, LL)\}$, $G(t.rc) = \{\mathbf{x} | \mathbf{x} \in G(t), d(\mathbf{x}, LL) + 2\tau > d(\mathbf{x}, LR)\}$, где $d(\mathbf{x}, P)$ – Евклидово расстояние от дескриптора $\mathbf{x}$ до гиперплоскости P . Процесс деления узла в sr -дереве показан на рис. 14б.

Если в рамках построения одного дерева дескрипторы в одних узлах делятся по правилам метрического дерева, а в других по правилам sr -дерева, то такое дерево называют гибридным sr -деревом [20]. Решение о типе узла принимается в процессе деления. Для этого выбирается параметр $\rho < 1$. Далее, каждый новый узел дерева t делится согласно правилам sr -дерева и если одно из условий $N(t.lc)/N(t) < \rho$ или $N(t.rc)/N(t) < \rho$ не выполняется, то деление отменяется, и узел t делится по правилам метрического дерева. Здесь $N(t)$, $N(t.lc)$, $N(t.rc)$ – количества дескрипторов в узлах t , $t.lc$ и $t.rc$ соответственно. Регулируя параметр ρ можно выбирать между скоростью поиска и занимаемой деревом памятью, чем ближе к единице ρ , тем быстрее идёт поиск, но тем больше памяти занимает дерево. Рекомендуемое в [20] значение $\rho = 0.7$. Именно

гибридным деревьям было отдано предпочтение при реализации процедуры быстрого поиска похожих дескрипторов.

Поиск дескрипторов в гибридном sr -дереве. Рассмотрим процедуру поиска k дескрипторов, максимально похожих на дескриптор запроса $\mathbf{q}$ в гибридном sr -дереве. Дополнительным параметром поиска будет целое число E – максимальное количество листовых узлов, которое следует просмотреть при поиске. Для точного алгоритма поиска $E = \infty$. Для приближительного поиска *best bin first*[27] параметр E может варьироваться от 50 до 200 и более. Чем больше E , тем точнее поиск, но тем медленнее работает алгоритм.

Для поиска будем использовать приоритетную очередь Q , изначально пустую. В данную очередь будем заносить узлы дерева, не имеющие перекрытия. Роль приоритета каждого узла играет наибольшее из двух расстояний: расстояние от дескриптора $\mathbf{q}$ до гипершара, ограниченного гиперсферой $\mathbb{W}$ или расстояние от дескриптора $\mathbf{q}$ до плоскости деления L . Всегда из очереди будет извлекаться узел с минимальным расстоянием. Для обозначения расстояния, назначенного каждому узлу очереди t , будем использовать функцию $d(t)$.

Также для хранения результата поиска введём список P . Этот список предназначен для хранения дескрипторов, близких к $\mathbf{q}$. Изначально список P пуст. Максимальное число элементов в P : $|P|_{\max} = k$. Для P определена величина $P.d = \max_{\mathbf{p} \in P} |\mathbf{p} - \mathbf{q}|$. Если $P = \emptyset$, то $P.d = \infty$. Операция добавления дескриптора $\mathbf{p}$ в P осуществляется следующим образом: если $|P| < k$, то $\mathbf{p}$ добавляется в P , иначе, если $|\mathbf{p} - \mathbf{q}| < P.d$, из P сначала удаляется дескриптор $\mathbf{w} \in P$, такой что $|\mathbf{w} - \mathbf{q}| = P.d$ и только затем добавляется $\mathbf{p}$. В случае если $|P| = k$ и $|\mathbf{p} - \mathbf{q}| \geq P.d$ содержимое P не изменяется.

Алгоритм поиска состоит из двух частей: прямой и обратный поиск.

Алгоритм прямого поиска:

Шаг 1. Присвоить текущему узлу t корневой узел дерева.

Шаг 2. Если узел t – листовой, перейти к шагу 8.

Шаг 3. Если $q < L$ (q лежит под плоскостью деления узла t), перейти к шагу 6.

Шаг 4. Если узел t не имеет области перекрытия ($\tau=0$), поместить $t.lc$ в Q .

Шаг 5. Присвоить $t=t.rc$, перейти к шагу 2.

Шаг 6. Если узел t не имеет области перекрытия ($\tau=0$), поместить $t.rc$ в Q .

Шаг 7. Присвоить $t=t.lc$, перейти к шагу 2.

Шаг 8. Поместить дескрипторы p в список P , $p \in G(t)$, $p \neq q$.

Обратный поиск работает, пока очередь Q не будет пуста и состоит из следующих шагов:

Шаг 1. Присвоить $e=1$

Шаг 2. Извлечь из Q узел t или закончить алгоритм, если Q пуста или $e \geq E$.

Шаг 3. Если $d(t) \geq P.d$ закончить алгоритм.

Шаг 4. Если t – листовой узел, перейти к шагу 10.

Шаг 5. Если $q < L$ (q лежит под плоскостью деления узла t), перейти к шагу 8.

Шаг 6. Если узел t не имеет области перекрытия ($\tau=0$), поместить $t.lc$ в Q .

Шаг 7. Присвоить $t=t.rc$, перейти к шагу 4.

Шаг 8. Если узел t не имеет области перекрытия ($\tau=0$), поместить $t.rc$ в Q .

Шаг 9. Присвоить $t=t.lc$, перейти к шагу 4.

Шаг 10. Поместить дескрипторы p в список P , $p \in G(t)$, $p \neq q$

Шаг 11. Увеличить e на единицу и перейти к шагу 2.

Результатом выполнения прямого и обратного поиска будет список P из k дескрипторов, максимально близких к q . Поскольку номер изображения, к которому принадлежит каждый дескриптор, известен, P легко преобразуется в список номеров изображений, на которых были найдены особенности, с дескрипторами похожими на q . Именно такой список нужно было получить для организации быстрого сопоставления пар в разделе 2.5.2.

Обратим внимание на то, от чего зависит эффективность по времени и памяти описанного алгоритма. На этапе прямого и обратного поиска, один из дочерних узлов просматриваемого узла t помещается в очередь Q , если t не является узлом с перекрытием. Чем больше очередь, тем дольше длится

обратный поиск. С другой стороны, чем больше область перекрытия узла τ , тем больше дубликатов одинаковых дескрипторов появится в дереве и тем глубже оно станет и тем больше потребует памяти. Ограничивающим значением для τ является максимально допустимое расстояние между похожими дескрипторами. Если присвоить τ меньшее значение, то есть риск пропустить близкие к $\mathbf{q}$ узлы. Таким образом, необходимо найти баланс между доступной для построения дерева памятью и желаемой скоростью выполнения алгоритма. Если количество дескрипторов невелико, а памяти много, то имеет смысл использовать много узлов с перекрытием в дереве. Если же дескрипторов много и памяти не хватает, придётся пожертвовать производительностью и отказаться от перекрывающихся узлов.

2.5.4. Сопоставление пар в больших наборах изображений.

Необходимая для дерева память. Вопрос потребляемой памяти для современных ЭВМ по-прежнему остаётся актуальным. Если в 64-разрядных операционных системах приложению может быть выделено достаточно памяти для работы, то в 32-разрядных операционных системах физически невозможно адресовать более 4 Гб оперативной памяти. Более того, операционные системы часто ограничивают количество памяти, выделяемое одному приложению. Обычно, гарантированно можно рассчитывать на доступные 2 Гб. Поэтому, для обеспечения работы процедуры поиска на 32-х разрядных машинах, необходимо следить за объемом памяти, потребляемым деревом поиска.

Рассмотрим информацию, которую обязательно хранить в каждом узле дерева. К такой информации относятся:

1. 129 коэффициентов делящей плоскости L – 132 байта,
2. 128 координата центра узла (центр гиперсферы $\mathbb{B}$) – 128 байт,
3. Радиус гиперсферы $\mathbb{B}$ – 4 байта,
4. Количество дескрипторов (для листовых узлов) – 4 байта,
5. Указатель на индексы дескрипторов (для листовых узлов) – 4 байта,
6. Флаг, является ли узел узлом с перекрытием – 1 байт,

7. Указатели на два дочерних узла – 8 байт.

Напомним, что для хранения координат дескрипторов, которые всегда находятся внутри гиперкуба с единичными сторонами, было предложено использовать однобайтовый целый тип данных. В связи с этим координаты центра узла занимают 128 байт. 129 коэффициентов делящей плоскости состоят из 128-ми размерного вектора – нормали к плоскости, который занимает 128 байт и числа с плавающей точкой – расстоянием от плоскости до начала координат, занимающим 4 байта. В сумме это даёт 132 байта на делящую плоскость. В итоге, общий объем памяти, занимаемый каждым узлом дерева $C_t=281$ байт.

Кроме узлов дерева при поиске в памяти должны храниться массивы дескрипторов и индексов на эти дескрипторы. Использование индексов даёт преимущество в скорости (копирование 4-х байтных индексов быстрее, чем 128-ми байтных дескрипторов) и используемой памяти (узлы с перекрытием дублируют индексы, но не сами дескрипторы). Обозначим размер памяти, занимаемый каждым индексом через $C_{id}=4$, а размер памяти, занимаемый дескриптором через $C_d=128$.

Для хранения дерева из n дескрипторов необходимо C_{mem} (15) байт оперативной памяти.

$$C_{mem} = n \left(\frac{2C_t}{l} + C_{id} + C_d \right). \quad (15)$$

Здесь l – минимальное количество дескрипторов, при котором узел дерева не делится (максимальное количество дескрипторов, представленное листовыми узлами). Выражение (15) получено для случая, когда в дереве отсутствуют перекрывающиеся узлы, то есть для дерева, использующего память максимально эффективно. Выразив из (15) n , получим количество дескрипторов, которое можно представить в виде дерева, имея C_{mem} оперативной памяти (16).

$$n = \frac{C_{\text{mem}}}{2C_t l^{-1} + C_{\text{id}} + C_d}. \quad (16)$$

Имея $C_{\text{mem}} = 2\text{Гб}$ оперативной памяти, при $l=10$, можно построить дерево из $n \approx 11.4$ миллиона дескрипторов, что в свою очередь хватит для представления 10 – 20-ти тысяч изображений. Очевидно, что на 64-разрядной системе ограничения на количество изображений обуславливаются лишь наличием физической памяти, но не ограничением системы.

Разбиение дерева. Проблему сопоставления изображений при $m > 20000$ можно решить и в условиях жёстких ограничений на память, если дана дополнительная информация для каждого изображения. Этой дополнительной информацией являются приблизительные координаты АНПА в момент получения каждого снимка: $\mathbf{x}_i = (x_i, y_i)$, $i=1, 2, \dots, m$. Данные координаты могут быть доступны из навигационной системы АНПА. Также эти координаты можно получить, если построить приблизительную траекторию движения АНПА, используя информацию о скорости и курсе в каждый момент времени.

Зная $\mathbf{x}_i$ можно по пространственному признаку разбить множество всех изображений на несколько подмножеств. Разбиение будем производить вдоль осевой линии области исследования (области покрытия снимками), так как вдоль этой линии область имеет наибольшую протяжённость. Осевая линия проходит через центр масс $\bar{\mathbf{x}} = (\bar{x}, \bar{y})$ точек $\mathbf{x}_i$:

$$\bar{\mathbf{x}} = \frac{1}{m} \sum_{i=1}^m \mathbf{x}_i,$$

под углом θ к оси абсцисс:

$$\theta = \frac{1}{2} \arctan \left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right).$$

Здесь μ_{kp} – дискретный центральный момент:

$$\mu_{kp} = \sum_{i=1}^m (x_i - \bar{x})^k (y_i - \bar{y})^p.$$

Подмножества изображений образуются путём деления множества всех изображений прямыми, перпендикулярными к осевой линии области исследований (рис. 15). Гибридные *sr*-деревья строятся для каждого подмножества поочерёдно. При этом в каждое дерево помещаются не только дескрипторы особых точек с изображений соответствующего подмножества, но и дескрипторы изображений, находящихся на расстоянии меньшем, чем r от границ этого подмножества. Величина r зависит от точности координат x_i . Чем точнее эти координаты, тем меньше может быть величина r . Если максимальная ошибка в координатах x_i составляет p длин участков дна d , охватываемых диагональю одного изображения, то $r \geq d(p+1)$.

Размеры подмножеств изображений выбирается таким образом, чтобы для каждого подмножества хватило памяти для построения дерева и хранения дескрипторов. Запросы на поиск похожих дескрипторов в каждом дереве выполняется исключительно для дескрипторов особенностей изображений подмножества, для которого это дерево было построено, но не для дескрипторов изображений из r окрестности этого подмножества.

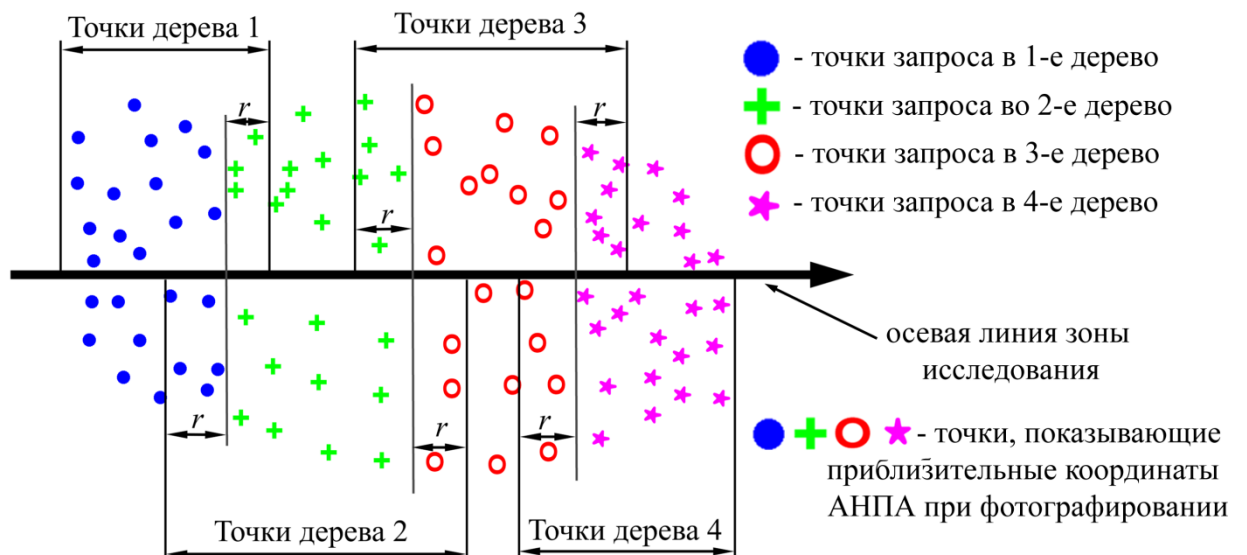


Рис. 15. Деление дескрипторов на подмножества.

Описанное разбиение имеет ряд преимуществ. Вся задача по-прежнему решается в оперативной памяти, нет необходимости хранить дерево на жёстком диске. Размер области перекрытия r практически не влияет на время работы

алгоритма сопоставления, поскольку время построения дерева мало, по сравнению со временем поиска похожих дескрипторов, для всех дескрипторов из подмножества. Каждое подмножество имеет не более двух соседей (а, соответственно зон перекрытия): справа и слева, благодаря тому, что разбиение ведётся вдоль одной оси.

2.6. Результаты

2.6.1. Описание данных. Поскольку процедура сопоставления изображений специфична и в существенной степени зависит от характера изображений, для исследования алгоритмов необходимо использовать данные, полученные непосредственно с АНПА. Практически невозможно синтезировать искусственные данные для исследования алгоритмов поиска связей на изображениях, имеющие аналогичные с реальными данными характеристики. В связи с этим для тестирования был взят набор данных, полученный с АНПА. Снимки сделаны на глубине 34–35 метров, с высоты 1.2–2.8 метров над поверхностью дна. Каждый снимок выполнен с разрешением 768×576 пикселей. Всего в наборе присутствует $m=3592$ изображения. Перекрытия между соседними снимками в наборе слабые, иногда вообще отсутствуют.

2.6.2. Компенсация освещения. В разделе 2.3.1 была рассмотрена процедура компенсации неравномерного освещения с целью увеличения стабильности процедуры сопоставления. Были рассмотрены три способа коррекции освещенности: (4), (5) и (6). Чтобы определить, насколько сильно эти способы влияют на процедуру сопоставления особенностей, четыре раза проведём процедуру сопоставления в исходном наборе изображений. В первый раз не будем использовать коррекцию освещения, а во второй, третий и четвёртый раз используем коррекции (4), (5) и (6) соответственно. В качестве оценки эффективности будем использовать общее число сопоставленных пар изображений – рис. 16.

Из рис. 16 видно, насколько важно применение коррекции освещения для увеличения надёжности сопоставления. Из всех предложенных алгоритмов коррекции освещения наилучшим образом проявили себя методы (5) и (6), дав наибольшее число сопоставленных пар изображений.

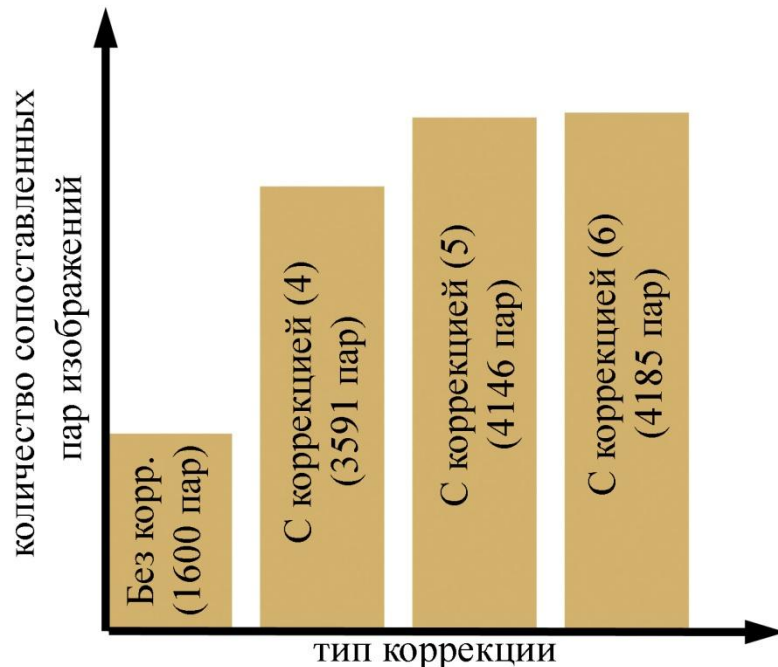


Рис. 16. Эффективность коррекции освещения.

2.6.3. Сопоставление пар изображений с малым перекрытием. При использовании коррекции освещения, задача сопоставления пар изображений является несложной, в случае хорошего перекрытия этих изображений. Часто перекрытие связанных изображений оказывается слишком маленьким. Это может происходить из-за низкой частоты съемки, либо из-за течения, которое может смещать АНПА между кадрами на большее расстояние, чем предполагает его программа. Кроме того, снимки с малым перекрытием содержат меньше избыточной информации, чем снимки с большим перекрытием, что экономит память и машинные ресурсы. Ввиду этого, главным требованием к алгоритму сопоставления пар изображений из раздела 2.4. является способность находить связь между изображениями, имеющими малое перекрытие. Благодаря использованию простой модели связи изображений (12) и алгоритма фильтрации ложных особенностей из 2.4.3 это становится

возможным. На рис. 17 представлена серия последовательно связанных изображений. При этом каждое изображение имеет очень малую зону перекрытия со следующим и предыдущим изображениями. Несмотря на это, изображения были успешно сопоставлены и, между каждой парой изображений, было найдено около 50-ти верных соответствий.

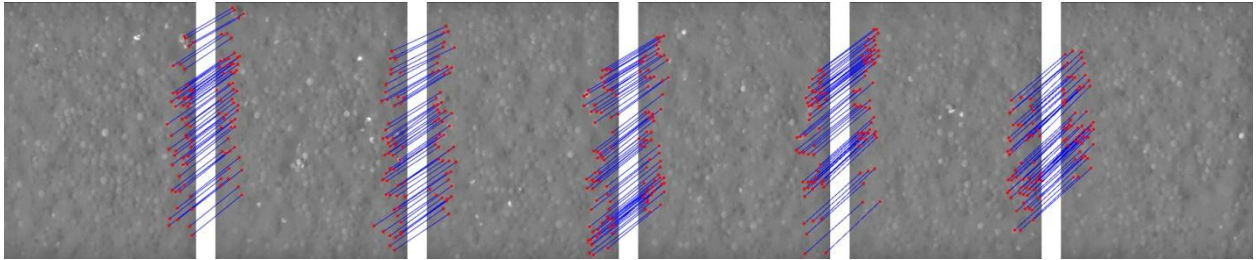


Рис. 17. Сопоставление изображений с малым перекрытием.

Обратим внимание, что для сопоставления подобных изображений человеку требуется несколько десятков секунд, а в большинстве случаев, найти соответствия вручную не представляется возможным. Предложенный алгоритм, в зависимости от количества особенностей на изображениях справляется за 30 – 150 мс, выполняясь на одном ядре процессора Intel Core Q9550, 2.83 ГГц.

2.6.4. Поиск связанных изображений. Быстрый поиск связанных изображений – наиболее длительный и сложный процесс, описанный в разделе 2.5. На качество и скорость его работы влияет целый ряд параметров. Перечислим эти параметры:

1. η – длина списка кандидатов на сопоставление каждого изображения – раздел 2.5.2,
2. E – количество узлов, которое необходимо посетить при поиске в гибридном *sp*-дереве – раздел 2.5.3,
3. k – количество дескрипторов, похожих на дескриптор запроса, которое следует находить – раздел 2.5.3.
4. l – максимальное количество дескрипторов в листовом узле дерева – раздел 2.5.4.

Будем исследовать влияние описанных параметров на всём наборе входных данных из $m=3591$ изображений, на которых были найдены 2362523 особенности.

Определим критерии точности и времени поиска, которые будут использоваться при оценке влияния тех или иных параметров на процедуру сопоставления пар. Для этого введём понятия поиска грубой силой. Данный вид поиска заключается в полном переборе и сопоставлении всех возможных пар изображений. Время, затрачиваемое на такой поиск, как и количество обнаруживаемых соответствующих пар изображений максимально. Поэтому в качестве временной характеристики будем использовать отношение времени поиска грубой силой ко времени поиска описанным алгоритмом. В качестве характеристики точности будем использовать процент найденных пар изображений с помощью описанного алгоритма. За сто процентов примем количества пар, найденных поиском грубой силой.

Рассмотрим влияние параметра k на точность процедуры сопоставления – рис. 18.

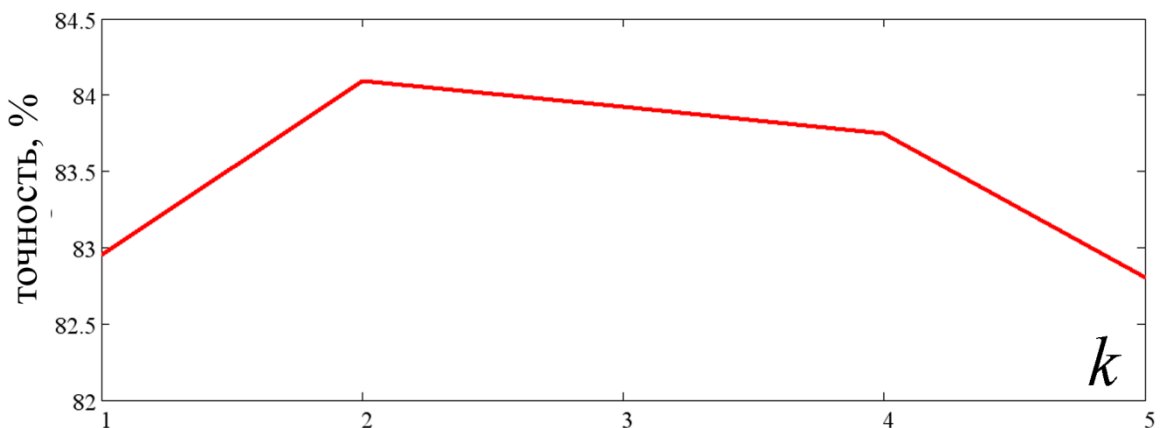


Рис. 18. Зависимость точности сопоставления от k .

Из графика на рис. 18 видно, что максимальная точность достигается при $k=2$. При дальнейшем увеличении k точность падает. Этот факт объясняется средним количеством связей каждого изображения. Чем ближе число k к среднему количеству связей, тем больше вероятность того что в таблицу **W** вносят вклад корректно сопоставленные дескрипторы, а не просто случайно

похожие. Таким образом, параметр k следует выбирать исходя из предполагаемой плотности покрытия дна снимками. Для снимков с малым перекрытием, как в тестируемом наборе данных, выберем $k=2$.

Параметр l влияет как на точность и время поиска, так и на потребляемую память. От величины параметра l , зависит глубина гибридного sp – дерева, так как деление узлов дерева прекращается, когда количество дескрипторов в них не превышает l . Поскольку l – максимальное, а не фактическое количество дескрипторов в каждом листовом узле, зависимость точности и времени сопоставления от l носит ступенчатый характер: рис. 19а, 19б.

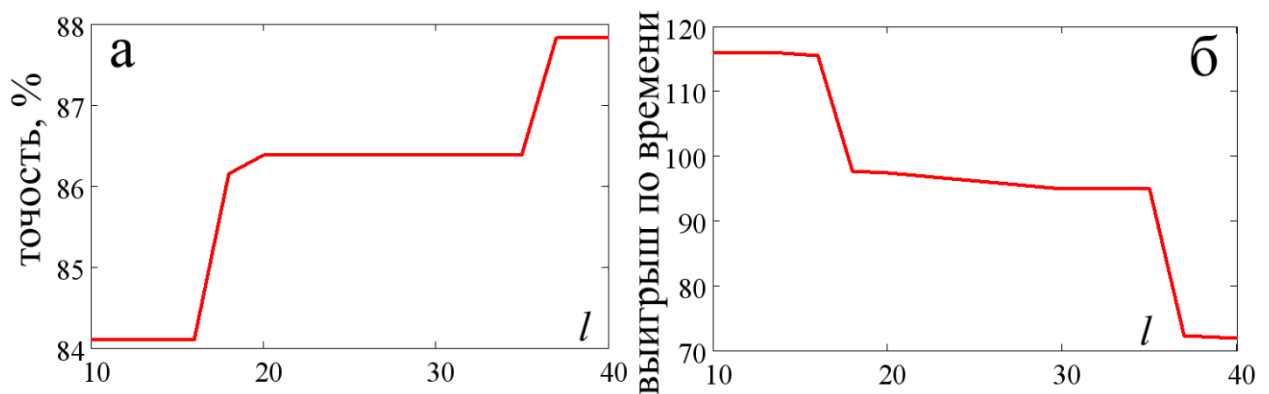


Рис. 19. Зависимость точности и времени сопоставления от l .

Увеличение l приводит к незначительному повышению точности поиска, при этом существенно падает преимущество по времени по сравнению с поиском грубой силой. Вследствие этого предлагается использовать $l=10$. Этого достаточно, чтобы занимаемая деревом память была существенно меньше, чем память, необходимая для хранения дескрипторов. А точность сопоставления будем контролировать другими параметрами, например E и η .

Параметры E и η одинаково влияют на точность и время сопоставления: чем больше эти параметры, тем выше точность и меньше выигрыш по времени и наоборот. Графики зависимости точности и времени сопоставления от E для 3 значений η представлены на рис. 20а и 20б.

На рисунке 20 также присутствует график, соответствующий непостоянному значению η . Для данного графика рассматривается порог t ,

такой, что для изображения p , выбирается такое число $\eta(p)$, что $w_{p,s_{\eta(p)}^p} > t$ (раздел 2.5.2). Увеличивая значение порога t можно повышать точность сопоставления и уменьшать время. Порог t является альтернативой фиксированному параметру η .

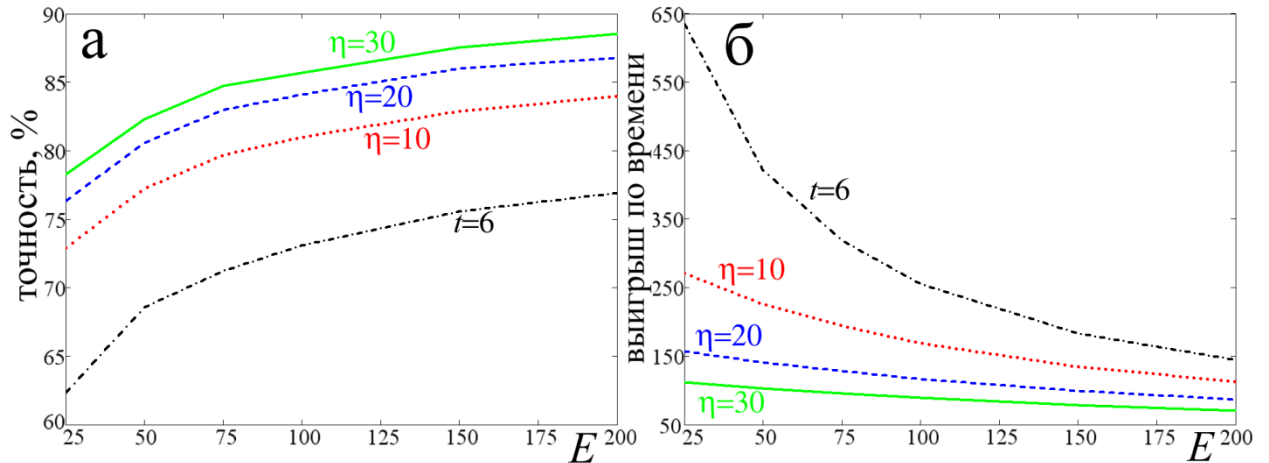


Рис. 20. Зависимость точности и времени сопоставления от E и η .

Стоит отметить, что выигрыш по времени, варьирующийся от 70 до 650 в представленных результатах является характеристикой, зависящей от количества изображений m . Поскольку поиск грубой силы имеет квадратичную вычислительную сложность $O(m^2)$, а описанный алгоритм практически линейную $O(m \cdot \log_2 m)$, выигрыш по времени будет расти с увеличением числа m .

ГЛАВА 3. ВЫЧИСЛЕНИЕ ПОЛОЖЕНИЯ И ОРИЕНТАЦИИ ИЗОБРАЖЕНИЙ

3.1. Постановка задачи

Пусть имеется m изображений, полученных с АНПА в результате выполнения исследовательской миссии. Также после выполнения процесса обнаружения связанных изображений, описанного в главе 2, найдены n пар связанных изображений. Обозначим за $i(l)$ и $j(l)$ номера l -ой связанной пары изображений: $l=1, 2, \dots, n$ $i(l), j(l) \in \{1, 2, \dots, m\}$. В результате процедуры сопоставления для каждой связанной пары изображений был найден вектор $\mathbf{t}_l = \mathbf{t}_{i(l)j(l)}$ (12), задающий связь с точностью до плоского перемещения и набор индексов связанных особенностей: $\mathbf{p}_k^{i(l)j(l)} = (p_{k1}^{i(l)j(l)}, p_{k2}^{i(l)j(l)})$, где $k=1, 2, \dots, \lambda(i(l), j(l))$ (раздел 2.1 и 2.4.2). Напомним, что пара индексов $p_{k1}^{i(l)j(l)}$ и $p_{k2}^{i(l)j(l)}$ означает, что $p_{k1}^{i(l)j(l)}$ -ая особенность $i(l)$ -го изображения с координатами $\mathbf{u}_{p_{k1}^{i(l)j(l)}}^{i(l)} = (u_{p_{k1}^{i(l)j(l)}}^{i(l)}, v_{p_{k1}^{i(l)j(l)}}^{i(l)})$ сопоставлена с $p_{k2}^{i(l)j(l)}$ -ой особенностью на $j(l)$ -ом изображении с координатами $\mathbf{u}_{p_{k2}^{i(l)j(l)}}^{j(l)} = (u_{p_{k2}^{i(l)j(l)}}^{j(l)}, v_{p_{k2}^{i(l)j(l)}}^{j(l)})$.

На основе полученной информации, а также, используя данные, доступные из навигационной системы АНПА необходимо рассчитать точные значения углов поворота АНПА (раздел 1.4.1) в момент получения каждого i -го изображения, $i=1, 2, \dots, m$: α_i (крен), β_i (дифферент), γ_i (курс), а также его координаты: $\mathbf{c}_i = (c_{ix}, c_{iy}, c_{iz})$. Поскольку камера жёстко связана с АНПА, а матрица калибровки камеры $\mathbf{K}$ (1) известна, найденные параметры позволят точно определить положение каждого изображения в панораме.

3.2. Поиск начального приближения решения

Соотношения, связывающие координаты точечных соответствий и искомые положения и ориентации АНПА в момент получения каждого

изображения – нелинейные (раздел 3.4.2). Для решения нелинейной задачи поиска параметров необходимо использовать итеративные методы, которые требуют хорошего начального приближения решения.

3.2.1. Начальные значения параметров. Для представления начального приближения некоторой величины v будем использовать обозначение v^0 . Если координаты АНПА при получении i -го снимка определяются вектором $\mathbf{c}_i = (c_{ix}, c_{iy}, c_{iz})$, а ориентация вектором $\mathbf{r}_i = (\alpha_i, \beta_i, \gamma_i)$, то начальные значения этих параметров обозначим через $\mathbf{c}_i^0 = (c_{ix}^0, c_{iy}^0, c_{iz}^0)$ и $\mathbf{r}_i^0 = (\alpha_{ix}^0, \beta_{iy}^0, \gamma_{iz}^0)$ соответственно, $i=1, 2, \dots, m$.

Начальное значение курса АНПА γ_{iz}^0 известно из его навигационной системы. Поскольку точность определения курса с помощью навигационной системы достаточно высока, его можно непосредственно использовать для инициализации нелинейной задачи поиска параметров. Также навигационная система предоставляет достаточно точное значение глубины АНПА - d_i . Поскольку глубина измеряется в направлении, перпендикулярном водной поверхности, значение глубины может быть использовано для инициализации третьей координаты АНПА: $c_{iz}^0 = d_i$.

Проводя фотографирование дна, АНПА преимущественно сохраняет горизонтальное положение. Это позволяет в качестве начальных значений для крена и дифферента выбрать нулевые значения: $\alpha_i^0 = 0$ и $\beta_i^0 = 0$. Ограничить движение АНПА плоскостью при поиске начального приближения решения – разумный выбор ещё и потому, что это приведёт к более стабильному решению. Попытка вычисления таких параметров, как α_i^0 , β_i^0 и c_i^0 в случае слабосвязанных изображений может привести к «уводу» решения. В качестве примера рассмотрим рис. 21а и 21б. На этих рисунках изображен набор одинаковых изображений с одинаковыми сопоставленными особенностями. При этом и рис. 21а соответствует идеальному совпадению связанных особых точек и рис. 21б. Однако, решение 21а «ушло» в другую плоскость из-за чего на горизонтальной плоскости кажется искажённым.

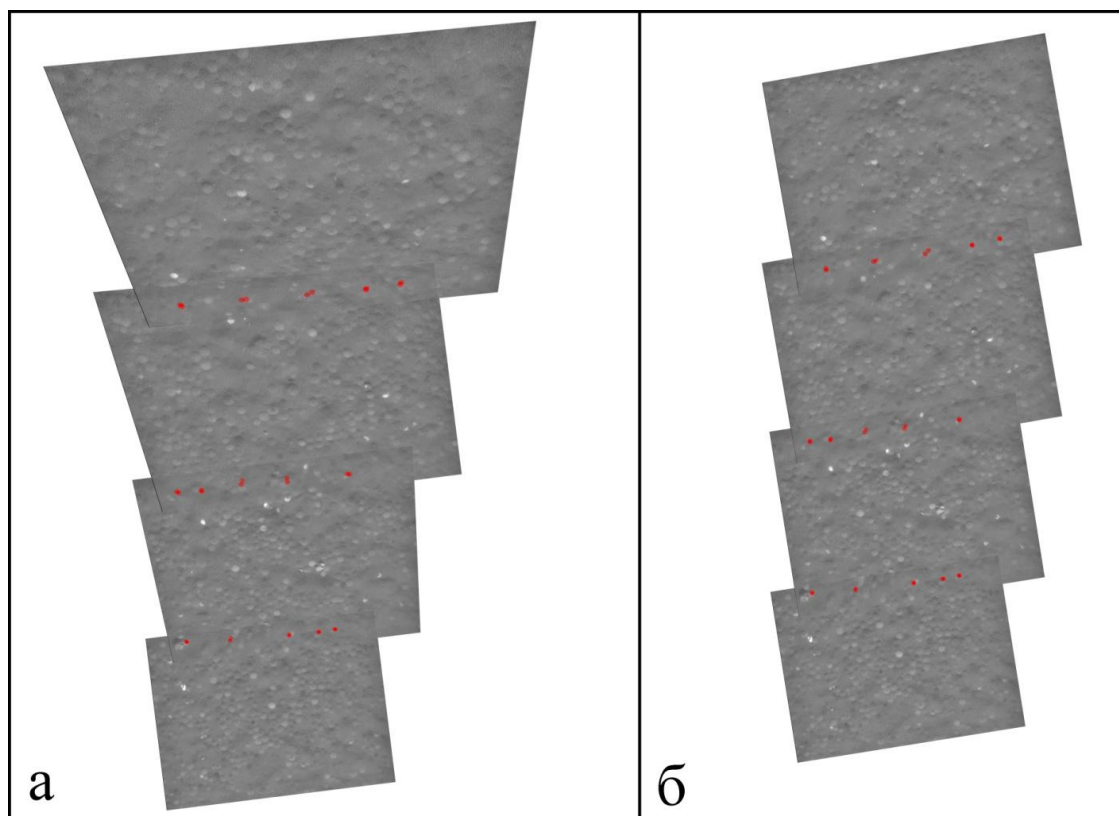


Рис. 21. Пример увода решения в другую плоскость.

Ситуация увода на рис. 21а произошла потому, что зоны пересечения изображений оказались малы или участок дна, представляемый изображениями оказался плоским. В случае если бы параметры α_i^0 , β_i^0 и c_{iz}^0 были фиксированы – рис. 21б, подобного увода бы не произошло. В случае использования нулевых значений для α_i^0 , β_i^0 мы можем получить большую ошибку в согласовании соответствующих точек, чем при расчёте этих величины с помощью линейного метода наименьших квадратов (МНК) [57], однако решение гарантированно останется в горизонтальной плоскости.

Неизвестным осталось начальное приближение для планарных координат АНПА: c_{ix}^0 и c_{iy}^0 . Начальные значения для планарных координат АНПА могут быть найдены на основе известных векторов $\mathbf{t}_l$, $l=1, 2, \dots, n$ – раздел 3.2.2.

3.2.2. Относительное позиционирование изображений. Напомним, что согласно разделу 2.2, было принято решение использовать нормализованную систему координат, когда начало координат располагается в центре изображения, а ширина и высота изображения соответствует 2 и $2w/h$

соответственно, где w и h – ширина и высота изображения, измеренная в пикселях. Таким образом, векторы $\mathbf{t}^{i(l)j(l)}$ определены в нормализованной системе координат изображения $j(l)$, $l=1, 2, \dots, n$.

Для расчёта начального приближения планарных координат c_{ix}^0 и c_{iy}^0 , $i=1, 2, \dots, m$ в системе координат, имеющий реальный масштаб и ориентацию дна, необходимо перевести векторы $\mathbf{t}^{i(l)j(l)}$ в систему координат дна. Для этого необходимо воспользоваться данными из навигационной системы АНПА и сведениями о калибровке камеры $\mathbf{K}$ (1). Вектор $\mathbf{t}^l$ поворачивается на угол, равный курсу АНПА γ_i и масштабируется. Для того чтобы корректно отмасштабировать вектор, необходимо ввести нормализованное фокусное расстояние $\bar{f} = f/w$. Поскольку используются нормализованные координаты изображений, фокусное расстояние в пикселях f (1) не подходит для масштабирования. Зная высоту АНПА над дном h_i , единицу расстояния в нормализованной системе координат i -го изображения можно перевести в масштаб мировой системы координат, домножив на $h_i/\bar{f}$. Таким образом векторы $\mathbf{t}^{i(l)j(l)}$ могут быть переведены в мировую систему координат с помощью формулы (17).

$$\mathbf{d}^l = \mathbf{d}^{i(l)j(l)} = \frac{h_{j(l)}}{\bar{f}} \begin{pmatrix} \cos \gamma_j & \sin \gamma_j \\ -\sin \gamma_j & \cos \gamma_j \end{pmatrix} \mathbf{t}^{i(l)j(l)}. \quad (17)$$

Здесь $\mathbf{d}^l = \mathbf{d}^{i(l)j(l)}$ – векторы, направленные из центра $j(l)$ – го изображения в центр $i(l)$ – го (18).

$$\mathbf{d}^{i(l)j(l)} = \begin{pmatrix} c_{i(l)x}^0 \\ c_{i(l)y}^0 \end{pmatrix} - \begin{pmatrix} c_{j(l)x}^0 \\ c_{j(l)y}^0 \end{pmatrix}. \quad (18)$$

Все n векторов $\mathbf{d}^l$ образуют множество $D=\{\mathbf{d}_l\}$. Множество центров изображений $C=\{\mathbf{c}_i\}$ и множество D образуют граф $G(C, D)$. Далее будем рассматривать случай, когда граф $G(C, D)$ связан. Если граф $G(C, D)$ оказался несвязным, то входной набор изображений представляет несколько различных, непересекающихся участков поверхности дна. В этом случае задачу построения

фотографической карты нужно рассматривать отдельно для каждой компоненты связности графа $G(C, D)$.

Поместим центр m -го изображения в начало системы координат фотографической карты. Тогда $c_{mx}^0 = 0$ и $c_{my}^0 = 0$, а вектор неизвестных $\mathbf{x} = (c_{1x}^0, c_{1y}^0, c_{2x}^0, c_{2y}^0, \dots, c_{m-1,x}^0, c_{m-1,y}^0)$. Уравнения (18) составляют СЛАУ с блочной матрицей $\mathbf{A} = (\mathbf{a}_{li})$, где $i = 1, 2, \dots, m-1$ и вектором свободных членов $\mathbf{b} = (\mathbf{d}^1, \mathbf{d}^2, \dots, \mathbf{d}^n)$. Определим блок $\mathbf{a}_{li}$:

$$\mathbf{a}_{li} = \begin{cases} \mathbf{0}^{2 \times 2}, & i \neq i(l), j(l) \\ \text{diag}(1, 1), & i = i(l). \\ \text{diag}(-1, -1), & i = j(l) \end{cases}$$

Здесь $\mathbf{0}^{2 \times 2}$ – матрица, размерности 2×2 , состоящая из нулевых элементов. Решение системы $\mathbf{Ax} = \mathbf{b}$ даст искомые планарные координаты центров изображений c_{ix}^0 и c_{iy}^0 . В случае, когда $n = m-1$, число уравнений в этой системе совпадает с числом неизвестных и равняется $2(m-1)$, система однозначно разрешима. В общем случае, когда $n > m-1$ система $\mathbf{Ax} = \mathbf{b}$ будет переопределённой. Тогда, вместо точного решения будем искать вектор $\mathbf{x}$, минимизирующий величину e :

$$e = \sum_{l=1}^n \left(\mathbf{d}^l - \begin{pmatrix} c_{i(l)x}^0 \\ c_{i(l)y}^0 \end{pmatrix} + \begin{pmatrix} c_{j(l)x}^0 \\ c_{j(l)y}^0 \end{pmatrix} \right)^2.$$

Вектор $\mathbf{x}$, минимизирующий e можно найти из нормального уравнения $\mathbf{Qx} = \mathbf{A}^T \mathbf{b}$, где $\mathbf{Q} = \mathbf{A}^T \mathbf{A}$ [57]. Матрица $\mathbf{Q} = (\mathbf{q}_{ij})$ – симметричная, положительно определённая и состоит из блоков $\mathbf{q}_{ij}$, $i, j = 1, 2, \dots, m-1$, имеющих размерность 2×2 . Общее число блоков в этой матрице равно $(m-1)^2$. Каждый блок $\mathbf{q}_{ij}$ имеет ненулевые элементы только тогда, когда изображения с номерами i и j непосредственно связаны ($\exists l: i(l)=i, j(l)=j$ или $i(l)=j, j(l)=i$). В большинстве задач компьютерного зрения количество связей между изображениями имеет порядок $2m \div 8m$, что приводит к разреженной структуре матрицы $\mathbf{Q}$.

3.3. Эффективное решение системы линейных алгебраических уравнений

3.3.1. Общий алгоритм решения. В разделе 3.2.2 описана задача, приводящая к решению системы линейных алгебраических уравнений вида $\mathbf{Ax}=\mathbf{b}$, где $\mathbf{A}$ – симметричная, положительно определённая разреженная матрица размерности $N \times N$, $\mathbf{A}=(a_{ij})$, $i, j=1, 2, \dots, N$. Так как матрица $\mathbf{A}$ – симметрична, то для неё существует разложение $\mathbf{A}=\mathbf{LDL}^T$ [48], где $\mathbf{L}=(l_{ij})$ – нижняя унитреугольная матрица, а $\mathbf{D}$ – диагональная. Для выполнения этого разложения будем использовать алгоритм из [48], модифицировав его для учёта разреженности матрицы $\mathbf{L}$. Обозначим номер первого ненулевого элемента i -ой строки матрицы $\mathbf{A}$ через s_i : $a_{ik}=0$, $k < s_i$, тогда $l_{ik}=0$ [98]. Также, для каждой строки j матрицы $\mathbf{A}$ введём множество номеров строк, ниже j -ой строки: $T_j=\{j+1, j+2, \dots, N\}$. Учитывая номера s_i и множества T_j перепишем алгоритм из [48] следующим образом:

Алгоритм 1:

Шаг 1. Присвоить $j=1$.

Шаг 2. Произвести вычисления по формуле:

$$a_{jj} = a_{jj} - \sum_{i=s_j}^{j-1} a_{ji}^2 a_{ii}.$$

Шаг 3. Для всех $i \in T_j$, если $s_i \leq j$, то:

$$a_{ij} = a_{jj}^{-1} \left(a_{ij} - \sum_{k=\max(s_i, s_j)}^{j-1} a_{ik} a_{jk} a_{kk} \right).$$

Шаг 4. Увеличить j на единицу, если $j < n+1$ перейти к шагу 2.

Число итераций на шаге 3 алгоритма 1 напрямую зависит от количества номеров строк в T_j . Очевидно, что номера i , для которых $s_i > j$ могут быть исключены из T_j . Некоторые из рассмотренных далее алгоритмов позволяют заранее исключать из T_j эти номера, упрощая алгоритм. Далее будем называть T_j вертикальной структурой матрицы $\mathbf{A}$.

После выполнения алгоритма 1, из элементов, расположенных на главной диагонали матрицы $\mathbf{A}$ и ниже ее, можно сформировать диагональную и треугольные матрицы разложения $\mathbf{LDL}^T$, где

$$\mathbf{D} = \text{diag}(a_{11}, a_{22}, \dots, a_{NN}),$$

$$l_{ij} = \begin{cases} 0, & j > i \\ 1, & j = i \\ a_{ij}, & j < i \end{cases}, \quad i, j = 1, 2, \dots, N.$$

Разложение $\mathbf{LDL}^T$ позволяет разбить решение исходной системы на три этапа: 1) $\mathbf{Lx}' = \mathbf{b}$ (прямая подстановка), 2) $\mathbf{x}'' = \mathbf{D}^{-1}\mathbf{x}'$, 3) $\mathbf{L}^T\mathbf{x} = \mathbf{x}''$ (обратная подстановка). Решая систему подобным образом, мы получаем преимущество от симметричности (вдвое меньшее число операций для разложения), положительной определенности (нет необходимости выбирать разрешающий элемент) и разреженности её матрицы. Чтобы применение данного алгоритма было максимально эффективным, необходимо упорядочить элементы матрицы $\mathbf{A}$ таким образом, чтобы в начале каждой строки оказалось как можно больше нулей, то есть, чтобы номера s_i были максимальными. Другими словами, необходимо минимизировать заполнение матрицы $\mathbf{A}$. Здесь под заполнением будем понимать величину η (19).

$$\eta = N + 2 \sum_{i=1}^N (i - s_i). \quad (19)$$

3.3.2. Минимизация заполнения матрицы системы. Системы линейных алгебраических уравнений, рассматриваемые в данной работе (раздел 3.2.2, раздел 3.4.6) имеют особенную структуру. Их матрицы состоят из блоков $\mathbf{V}_{ij}$, где $i, j = 1, 2, \dots, t$, где t – количество изображений в задаче. При этом блок $\mathbf{V}_{ij}$ содержит ненулевые элементы только когда изображения с номерами i и j связаны (в зависимости от конкретной задачи изображения пересекаются – раздел 3.2.2, или содержат проекции общей точки пространства – раздел 3.4.6). Чем ближе номера связанных изображений друг к другу (чем меньше величина

$|i-j|$), тем ближе к главной диагонали будут располагаться блоки с ненулевыми элементами и тем меньше будет заполнение η (19) матрицы системы **A**.

Набор изображений и их связи могут быть представлены в виде графа $G(V, E)$, где V – множество номеров изображений (вершины графа), а E – множество связей между этими изображениями (рёбра графа). Используя данное представление, рассмотрим три эвристических алгоритма на графах для перенумерации изображений, приводящих к низкому заполнению η матрицы системы.

3.3.3. Обратный алгоритм Катхилла–Макки. Самым известным алгоритмом перенумерации изображений является алгоритм Катхилла–Макки [40]. Для описания этого алгоритма введем понятие структуры уровней **L**. Структура $\mathbf{L}=(\mathbf{l}_1, \mathbf{l}_2, \dots, \mathbf{l}_n)$ представляет собой вектор размерности n , где n – количество уровней в структуре. Каждый уровень $\mathbf{l}_i = (l_{i1}, l_{i2}, \dots, l_{i\lambda_i})$, $i \in 1, 2, \dots, n$ – вектор размерности λ_i – количества вершин на уровне i . Структура уровней строится на основе вершины $r \in V$ графа изображений $G(V, E)$. Эта вершина целиком заполняет первый уровень структуры **L**: $l_{11}=r$, $\lambda_1=1$. Каждый следующий уровень $\mathbf{l}_i$, $i = 2, 3, \dots, n$ состоит из элементов множества Ψ_i :

$$\Psi_i = \{k \in \bigcup_{j=1}^{\lambda(i-1)} \text{Adj}(l_{i-1,j}) \mid k \notin \Psi_{i-1} \cup \Psi_{i-2}\}.$$

Здесь $\text{Adj}(v)$ – множество всех вершин, смежных с вершиной v , а Ψ_{i-1} и Ψ_{i-2} – множества, составленные из компонент векторов $\mathbf{l}_{i-1}$ и $\mathbf{l}_{i-2}$ соответственно. Множество $\Psi_0 = \emptyset$. Порядок компонент в векторе $\mathbf{l}_i$ для произвольной структуры уровней может быть любым. Однако в алгоритме 2 будет выбран определённый порядок, который позволит упростить алгоритм Катхилла–Макки.

Построение структуры уровней для некоторой вершины r легко реализовать, если хранить граф $G(V, E)$ в виде двух векторов: $\mathbf{d}=(d_1, d_2, \dots, d_{|V|})$ и $\mathbf{C}=(\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_{|V|})$. Каждый элемент d_i хранит количество вершин, связанных с вершиной i , а вектор $\mathbf{c}_i = (c_{i1}, c_{i2}, \dots, c_{id_i})$ хранит номера вершин, связанных с

вершиной i в порядке возрастания их степени (количества рёбер графа, инцидентных этой вершине), $i \in 1, 2, \dots, |V|$. Подобная структура может быть построена из набора рёбер E всего за один проход по ним. Хранение связанных элементов в порядке возрастания их степени необходимо для реализации алгоритма Катхилла–Макки. Используя описанную структуру, приведём алгоритм построения структуры уровней $\mathbf{L}$ на основе вершины r :

Алгоритм 2:

Шаг 1. Инициализировать: $l_{11}=r$, $\lambda_1=1$, $f_i=0$, $i=1, 2, \dots, |V|$, $i \neq r$, $f_r=1$, $n=1$.

Шаг 2. Присвоить: $j=1$, $i=1$.

Шаг 3. Присвоить $p=l_{ni}$.

Шаг 4. Для $k=1, 2, \dots, d_p$: если $f_{c(p,k)}=0$, присвоить: $f_{c(p,k)}=1$, $l_{n+1,j}=c_{pk}$, $j=j+1$.

Шаг 5. Увеличить i на единицу, если $i \leq \lambda_n$ перейти к шагу 3.

Шаг 6. Присвоить: $\lambda_{n+1}=j-1$, если $\lambda_{n+1}>0$, то $n=n+1$ и перейти к шагу 2.

Используя структуру уровней, полученную с помощью алгоритма 2, запишем обратный алгоритм Катхилла–Макки [40] следующим образом:

Алгоритм 3:

Шаг 1. Присвоить: $p=1$, $i=n$.

Шаг 2. Для $j=\lambda_i$, $\lambda_{i-1}, \dots, \lambda_1$ присвоить: $k_{l(i,j)}=p$, $p=p+1$.

Шаг 3. Уменьшить i на единицу, если $i \geq 1$ перейти к шагу 1.

Элементы k_i , $i=1, 2, \dots, |V|$ – новые номера вершин в V .

В качестве вершины r для алгоритма Катхилла–Макки необходимо взять вершину, дающую наиболее длинную и узкую структуру уровней. Такая вершина, как правило, находится на границе графа. Назовём её псевдопериферийной вершиной. Структура уровней $\mathbf{L}$, а также число уровней n зависят от начальной вершины x , для которой данная структура была построена. Рассматривая $\mathbf{L}$ и n как функции вершины x , будем писать: $\mathbf{L}(x)$ и $n(x)$. Псевдопериферийную вершину r можно найти с помощью алгоритма 4.

Алгоритм 4:

Шаг 1. Выбирать в качестве начальной вершины произвольную вершину x .

Шаг 2. Построить структуру уровней $L(x)$.

Шаг 3. Выбрать из $L(x)$ узел r с минимальной степенью.

Шаг 4. Построить структуру уровней $L(r)$.

Шаг 5. Если $n(r) > n(x)$: $x=r$, перейти к шагу 3.

Алгоритм Катхилла–Макки, кроме минимизации заполнения η также уменьшает величину w (20).

$$w = \max(i - s_i + 1), \quad i = 1, 2, \dots, N. \quad (20)$$

Здесь N – число уравнений в решаемой системе. Использование w в алгоритме 1 для вычисления T_j ($T_j = \{j+1, j+2, \dots, \min(j+w-1, N)\}$) даёт существенный прирост производительности за счёт уменьшения числа итераций на шаге 3 этого алгоритма.

На рис. 22 показано графическое представление заполнения матрицы системы, полученной после перенумерации изображений с помощью алгоритма Катхилла–Макки.

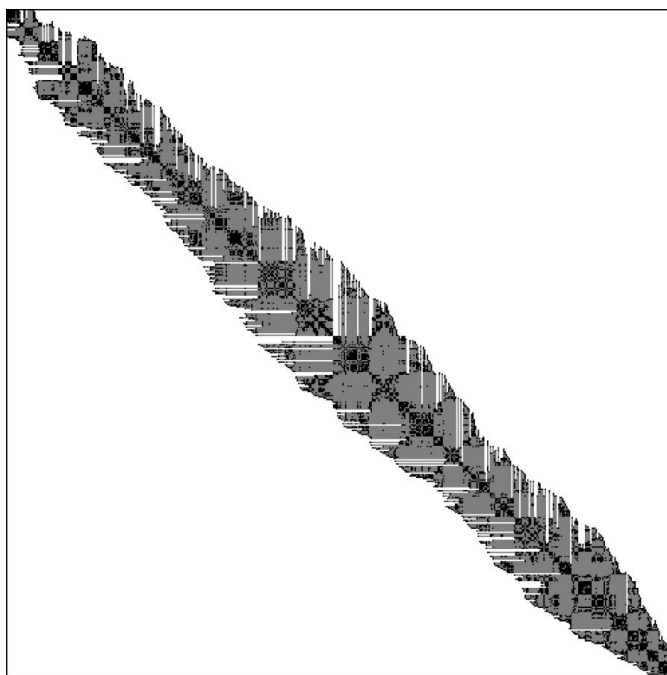


Рис. 22. Обратный алгоритм Катхилла-Макки. Структура матрицы системы.

Белым цветом на рис. 22 показаны нулевые элементы матрицы, чёрным – ненулевые. Серый цвет означает то, что элемент может стать ненулевым после LDL^T разложения матрицы системы.

3.3.4. Метод вложенных сечений. Метод вложенных сечений [4] основан на рекурсивном разделении графа $G(V, E)$ на несвязные компоненты с помощью вершин – разделителей. Алгоритм этого метода можно записать следующим образом:

Алгоритм 5:

Шаг 1. Присвоить $R=V, k=|V|$.

Шаг 2. Выбрать в $G(R, E)$ связную компоненту $G(C, E)$, построить для неё структуру уровней $L(r)=\{I_1, I_2, \dots, I_n\}$, где r – псевдопериферийная вершина.

Шаг 3. Если в $L(r)$ число уровней $n(r) \leq 3$, положить $S=C$ и перейти к шагу 4. Иначе $j=\lceil n/2 \rceil, S=\{y \in I_j \mid \text{Adj}(y) \cap I_{j+1} \neq \emptyset\}$. Здесь $\lceil x \rceil = \min\{n \in \mathbb{Z} \mid n \geq x\}$.

Шаг 4. Пронумеровать узлы разделителя S числами от $k-|S|+1$ до k .

Шаг 5. Присвоить: $R=R-S$ и $k=k-|S|$. Если $R \neq \emptyset$, перейти к шагу 2.

На шаге 2 алгоритма 5 происходит выбор связной компоненты из графа $G(R, E)$. Наилучшим решением является наименьшая по размеру компонента. Такая стратегия выбора позволяет уменьшить разброс элементов вокруг главной диагонали для строк и столбцов матрицы решаемой системы уравнений, соответствующих длинным разделителям.

Алгоритм вложенных сечений не минимизирует величину w , в отличие от алгоритма Катхилла–Макки. Однако при многократном решении задачи с одинаковой структурой можно рассчитать вертикальную структуру матрицы $T(j)$ следующим образом: $T(j)=\{y \in \{j+1, j+2, \dots, n\} \mid s_y \leq j\}$. Использование заранее рассчитанной структуры $T(j)$ существенно упрощает последующее решение задач с однотипной вертикальной структурой с помощью алгоритма 1. Типичный вид матрицы системы, полученной в результате выполнения алгоритма 5, представлен на рисунке 23.

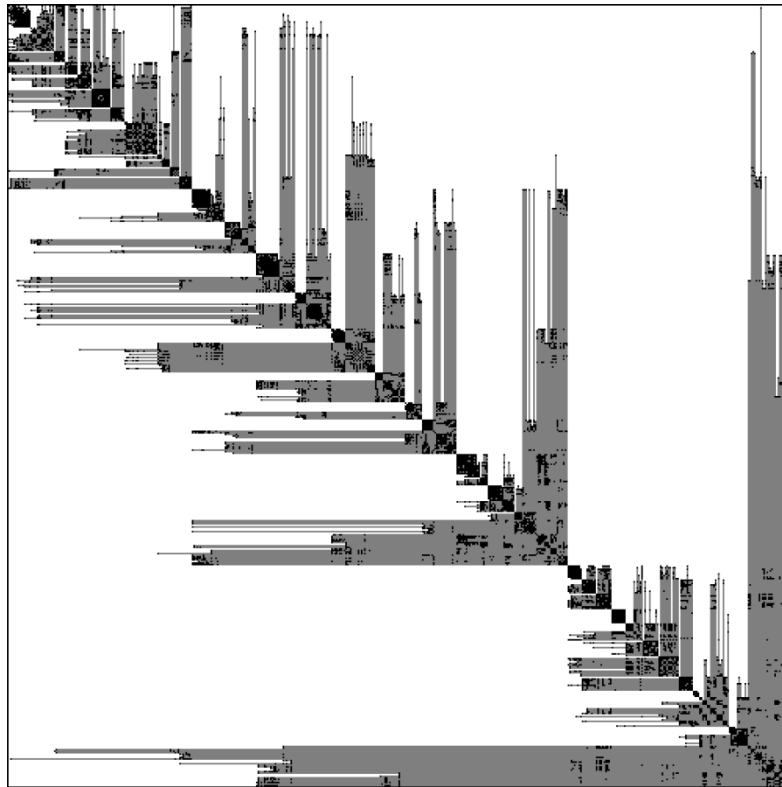


Рис. 23. Метод вложенных сечений. Структура матрицы системы.

3.3.5. Модификация метода вложенных сечений. Наиболее важный шаг в алгоритме 5 – поиск разделителя S . Разделитель на каждой итерации должен состоять из как можно меньшего числа вершин и делить граф на сопоставимые по размеру части.

В алгоритме 5 размер разделителя никак не учитывается. В качестве разделителя на шаге 3 берутся такие элементы центрального уровня j , которые имеют связи с элементами нижнего уровня. Данная стратегия выбора очень проста, но не эффективна. Её можно улучшить, если выбирать в качестве уровня j не центральный, а минимальный уровень из некоторой окрестности центра структуры уровней.

Рассмотрим следующую окрестность уровня j : $\lceil n/2-an \rceil, \lceil n/2-an+1 \rceil, \dots, \lceil n/2+an \rceil$, где a – коэффициент от 0 до 0.5. Из данной окрестности выбирается такой уровень j , который даёт разделитель S минимального размера. При $a=0$ имеем стандартный алгоритм 5. При $a=0.5$ имеем ситуацию, когда на каждой итерации выбирается минимальный уровень из всех уровней в структуре.

Однако, подобная ситуация приводит к неравномерному делению графа и существенно снижает скорость работы алгоритма.

На рис. 24 представлены графики зависимости полученного заполнения матрицы от коэффициента a для двух разных задач. Выигрыш, получаемый при выборе уровня разделителя из некоторого интервала, существенно зависит от характера задачи. Например, на левом графике рисунка 1 при $a=0.35$ заполнение оказалось в 1.4 раза меньше, чем при $a=0$, а на правом графике при $a=0.35$ заполнение уменьшилось в 3 раза.

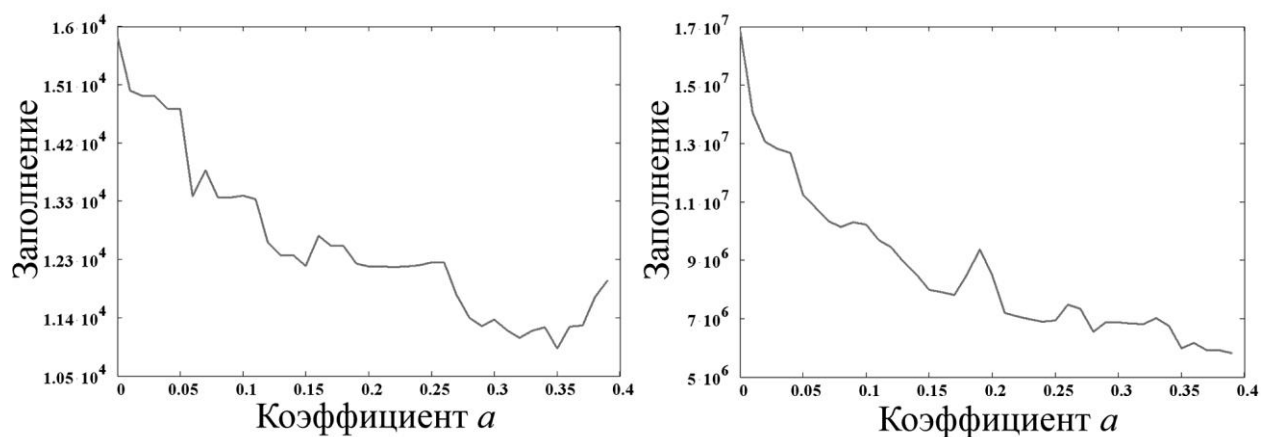


Рис. 24. Зависимость заполнения матрицы системы от коэффициента a .

Для сравнения алгоритмов в разделе 3.5.2 используются две версии алгоритма вложенных сечений: версия с коэффициентом $a=0.35$ и версия, выполняющая алгоритм несколько раз для различных коэффициентов a и выбирающая минимальное получившееся заполнение.

Следует отметить, что модифицированный алгоритм вложенных сечений рассматривается в контексте задач компьютерного зрения, которые характеризуются непостоянным количеством связей между изображениями, что порождает разные по ширине уровни в структуре уровней. Для задач на регулярных сетках предложенная модификация приводит лишь к увеличению времени вычислений, так как практически все уровни будут иметь одинаковую ширину.

3.3.6. Метод параллельных сечений. Метод параллельных сечений заключается в единовременном разделении графа $G(V, E)$ на q компонент

связности R_i , $i=1, 2, \dots, q$ с помощью разделителей S_j , $j=1, 2, \dots, q-1$. Структура уровней L , построенная для графа $G(V, E)$ делится с помощью $q-1$ горизонтальных параллельных разделителей S_j , как это описано в [4]. При этом для компонент связности и разделителей справедливы соотношения (21).

$$\begin{aligned} R_i \cap R_k &= \emptyset, & k &= 1, 2, \dots, q, & k &\neq i \\ S_j \cap S_p &= \emptyset, & p &= 1, 2, \dots, q-1, & |j-p| &> 1 \end{aligned} \quad (21)$$

Нумерация узлов в методе параллельных сечений происходит следующим образом: узлы всех компонент R_i нумеруются отдельно с помощью алгоритма Катхилла–Макки, затем к номеру добавляется число $\sum_{k=1}^{i-1} |R_k|$. Нумерация внутри разделителей S_j произвольна, а для получения глобального номера узла к внутреннему номеру добавляется число $\sum_{i=1}^q |R_i| + \sum_{k=1}^{j-1} S_k$. Подобная нумерация узлов приводит к матрице системы, имеющий вид, представленный на рисунке 25.

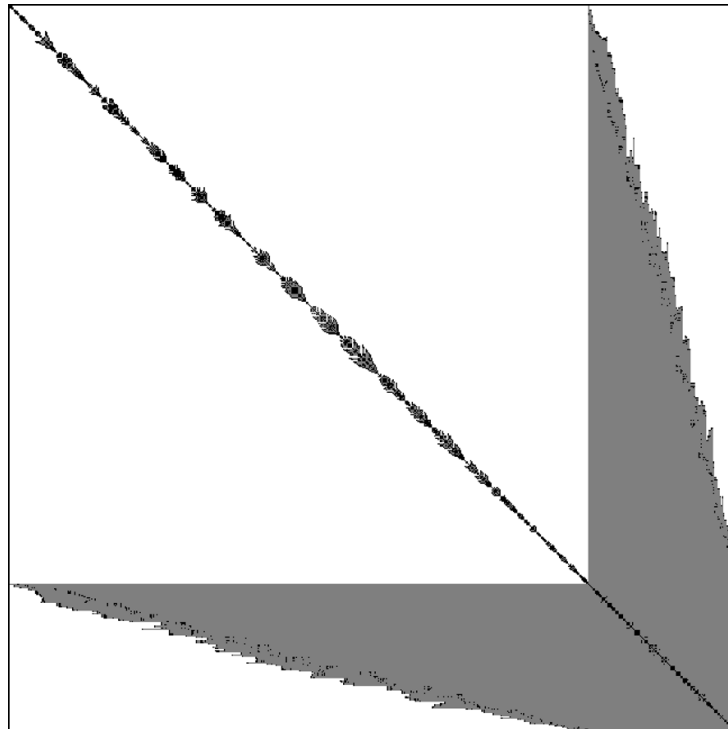


Рис. 25. Метод параллельных сечений. Структура матрицы системы.

Матрица на рис. 25 имеет особенную структуру: она содержит часть отвечающую за разделители и за компоненты связности. Схематически структура матрицы представлена на рис. 26 слева. Матрица состоит из блоков

$\mathbf{B}_{kp}$, $k, p=1, 2, \dots, 2q-1$. Ширина блока $\mathbf{B}_{kp}$ равна $|R_p|$, при $p \leq q$ и $|S_{p-q}|$ при $p > q$. Высота блока $\mathbf{B}_{kp}$ равна $|R_k|$, при $k \leq q$ и $|S_{k-q}|$ при $k > q$.

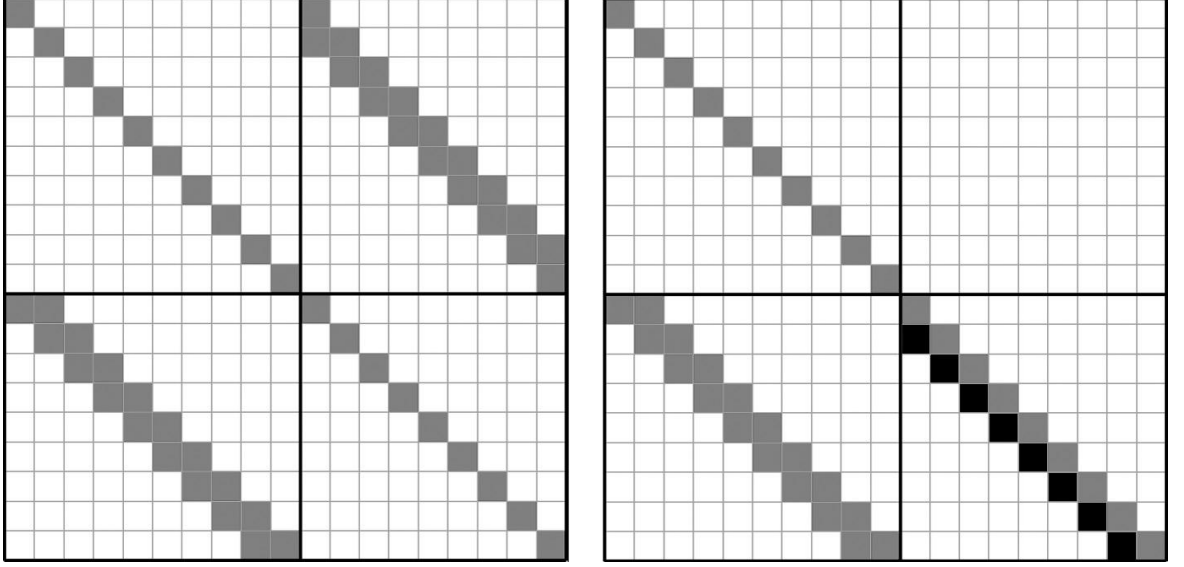


Рис. 26. Метод вложенных сечений: слева – схематическая структура матрицы, справа – структура матрицы $\mathbf{L}$ из $\mathbf{LDL}^T$ разложения.

На основе (21) можно сделать вывод, что блок $\mathbf{B}_{kp}$ содержит ненулевые элементы только в трёх случаях: $k=p$, $0 \leq k-q-p \leq 1$ или $0 \leq p-q-k \leq 1$ (серые блоки на рисунке 26). Подобная структура матрицы не даёт минимального заполнения η (19), но сохраняет большое количество дополнительных пустых блоков после $\mathbf{LDL}^T$ разложения. На рис. 26 справа показана схема матрицы $\mathbf{L}$. Чёрные блоки – блоки, которые стали ненулевыми после $\mathbf{LDL}^T$ разложения.

Кроме блоков с индексами k, p : $p < k$ нулевыми остались дополнительные блоки с индексами i, j : $i > q$, $i-q+1 < j < i-1$, а также блок с индексами $q+1, q$. Если пропускать дополнительные пустые блоки в алгоритме 1, можно существенно уменьшить объем вычислений. Таким образом, говоря о заполнении матрицы системы в случае метода параллельных сечений, будем иметь в виду величину η' (22), учитывающую дополнительные пустые блоки.

$$\eta' = \eta - 2 \sum_{i=1}^{q-1} |S_i| \left(\sum_{j=i+2}^q |R_j| + \sum_{j=1}^{i-2} |S_j| \right). \quad (22)$$

Эффективность применения описанных алгоритмов к задачам относительного позиционирования изображений и к задаче уточнения решения рассмотрена в разделе 3.5.

3.4. Итеративное уточнение решения

3.4.1. Алгоритм Левенберга–Марквардта. Алгоритм Левенберга–Марквардта [79] хорошо зарекомендовал себя в различных задачах компьютерного зрения [57] благодаря своей устойчивости. Он является вариацией метода Гауса–Ньютона.

Предположим, что мы имеем функцию $\mathbf{X} = \mathbf{f}(\mathbf{P})$, где $\mathbf{X}$ – некоторый измеренный вектор в $\mathbb{R}^N$, а $\mathbf{P}$ – вектор параметров в $\mathbb{R}^M$. Измеренный вектор $\mathbf{X}$, аппроксимирующий некоторый истинный вектор $\bar{\mathbf{X}}$, известен. Необходимо найти такой вектор параметров $\hat{\mathbf{P}}$, что $\mathbf{X} = \mathbf{f}(\hat{\mathbf{P}}) - \epsilon$, где $\|\epsilon\|$ – значение минимизируемой ошибки. Неитеративное решение возможно лишь в случае, когда $\mathbf{f}$ – линейна, иначе необходимо выполнить некоторое количество итераций K . При этом должно быть известно начальное значение вектора параметров $\mathbf{P}^0$. Далее оценку необходимо уточнять, основываясь на том предположении, что $\mathbf{f}$ – локально линейна. Пусть $\epsilon^{k-1} = \mathbf{f}(\mathbf{P}^{k-1}) - \mathbf{X}$, $k=1, 2, \dots, K$. Аппроксимируем значение $\mathbf{f}$ около $\mathbf{P}^{k-1}$: $\mathbf{f}(\mathbf{P}^{k-1} + \Delta^k) = \mathbf{f}(\mathbf{P}^{k-1}) + \mathbf{J}\Delta^k$. $\mathbf{J}$ – линейное отображение, представленное матрицей Якоби:

$$\mathbf{J} = \left(\frac{\partial f_i}{\partial p_j} \right), \quad i = 1, 2, \dots, N; j = 1, 2, \dots, M.$$

Необходимо найти вектор $\mathbf{P}^k = \mathbf{P}^{k-1} + \Delta^k$, минимизирующий разность $\mathbf{f}(\mathbf{P}^k) - \mathbf{X}$:

$$\mathbf{f}(\mathbf{P}^k) - \mathbf{X} = \mathbf{f}(\mathbf{P}^{k-1}) + \mathbf{J}\Delta^k - \mathbf{X} = \epsilon^{k-1} + \mathbf{J}\Delta^k.$$

Таким образом, требуется минимизировать $\|\epsilon^{k-1} + \mathbf{J}\Delta^k\|$ по параметру Δ^k , что является линейной задачей. Для расчёта Δ^k необходимо решить нормальное уравнение, которое в методе Левенберга–Марквардта имеет вид (23).

$$(\mathbf{J}^T \mathbf{J} + \lambda \mathbf{I}) \Delta^k = -\mathbf{J}^T \epsilon^{k-1}. \quad (23)$$

Здесь $\mathbf{I}$ – единичная матрица, $\mathbf{J}$ матрица Якоби, рассчитанная для вектора параметров $\mathbf{P}^{k-1}$, а λ – параметр алгоритма. После определения Δ^k , определяем вектор параметров $\mathbf{P}_k$ (24).

$$\mathbf{P}^k = \mathbf{P}^{k-1} + \Delta^k. \quad (24)$$

Основной интерес представляет параметр алгоритма λ . Его начальное значение, как правило, задаётся как 10^{-3} , умноженное на среднее значение диагональных элементов матрицы $\mathbf{J}^T \mathbf{J}$. Если решение (23) Δ^k приводит к уменьшению ошибки, то λ делится на десять и происходит переход к следующему шагу, иначе λ умножается на десять и k -ый шаг повторяется.

Существует несколько стратегий определения количество необходимых шагов K . Алгоритм может работать до тех пор, пока вектор приращения параметров не станет слишком маленьким, пока значение ошибки не достигнет требуемого порога, либо пока ошибка не начнёт уменьшаться слишком медленно. Также может быть установлен жёсткий лимит на общее количество итераций или на максимальное количество повторений одной итерации, не приводящей к уменьшению ошибки.

3.4.2. Структура рассматриваемой задачи. В задаче уточнения положений и ориентаций изображений в качестве вектора измерений $\mathbf{X}$ выступают координаты особых точек, полученные в разделе 2.4. Вклад в задачу будут вносить только те особые точки, которые были сопоставлены с особыми точками на других изображениях. Каждая пара сопоставленных точек с координатами $\mathbf{u}_{p_{k1}}^{i(l)}$ и $\mathbf{u}_{p_{k2}}^{j(l)}$, $l=1, 2, \dots, n$, $k=1, 2, \dots, \lambda(i(l), j(l))$ (раздел 3.1) является проекциями некоторой i -ой точки поверхности дна $\mathbf{X}_i=(x_i, y_i, z_i)$, $i=1, 2, \dots, N$, где N – общее количество точек на дне. Каждая точка $\mathbf{X}_i$ рассматривается тогда, когда известны, по крайней мере, две её проекции. Для удобства введём векторы проекций i -ой точки дна на j -ое изображение: $\mathbf{x}_i^j = (x_i^j, y_i^j)$, $j=1, 2, \dots, m$. Очевидно, что не все $\mathbf{x}_i^j$ известны, так как каждая трёхмерная точка обычно видна лишь на ограниченном числе изображений.

Неизвестные проекции $\mathbf{x}_a^b$, $a \in \{1, 2, \dots, N\}$, $b \in \{1, 2, \dots, m\}$ можно пропускать в решаемой задаче. Известные проекции строятся на основе пар $\mathbf{u}_{p_{k1}}^{i(l)} \text{ и } \mathbf{u}_{p_{k2}}^{j(l)}$, как это показано в разделе 3.4.3.

Таким образом, вектор $\mathbf{X}$ строится на основе координат проекций $\mathbf{x}_i^j$ (25).

$$\mathbf{X} = (\mathbf{x}_1^1, \mathbf{x}_2^1, \dots, \mathbf{x}_N^1, \mathbf{x}_1^2, \mathbf{x}_2^2, \dots, \mathbf{x}_N^2, \dots, \mathbf{x}_1^m, \mathbf{x}_2^m, \dots, \mathbf{x}_N^m). \quad (25)$$

Вектор $\mathbf{X}$ связан с координатами особенностей через координаты точек дна посредством проецирования. Поскольку координаты точек дна $\mathbf{X}_i$ изначально неизвестны, их тоже следует включить в задачу уточнения решения. При этом начальное значение для точек $\mathbf{X}_i$ может быть определено двумя способами, описанными в разделе 3.4.4. Метод одновременного вычисления положения точек $\mathbf{X}_i$ вместе с положением и ориентацией изображений носит название bundle adjustment [57, 102, 93, 38] и широко распространён в компьютерном зрении.

Связь между координатами проекции $\mathbf{x}_i^j$ и трехмерной точки $\mathbf{X}_i$ определяется с использованием матрицы калибровки камеры $\mathbf{K}$ (1), положения и ориентации АНПА согласно (26).

$$\hat{\mathbf{x}}_i^j = \mathbf{K}(\mathbf{R}(\mathbf{r}_j)^T | -\mathbf{R}(\mathbf{r}_j)^T \mathbf{c}_j)(\mathbf{X}_i | 1)^T. \quad (26)$$

Здесь $\mathbf{r}_j = (\alpha_j, \beta_j, \gamma_j)$ и $\mathbf{c}_j = (c_{jx}, c_{jy}, c_{jz})$ – ориентация и положение АНПА. $\mathbf{R}(\mathbf{r}_j)$ – матрица поворота, составленная на основе углов ориентации $\mathbf{r}_j$ (раздел 3.4.5). Вектор $\hat{\mathbf{x}}_i^j = (\hat{x}_i^j, \hat{y}_i^j, \hat{w}_i^j)$ – однородное представление вектора $\mathbf{x}_i^j$:

$$x_i^j = \frac{\hat{x}_i^j}{\hat{w}_i^j}, \quad y_i^j = \frac{\hat{y}_i^j}{\hat{w}_i^j}. \quad (27)$$

Уравнение проекции (26) включает в себя повернутый обращенный вектор $\mathbf{c}_j$. Это связано с тем, что любое перемещение камеры соответствует обратному перемещению сцены, проецируемой на камеру. В частности, поэтому в (26) используется транспонированная матрица $\mathbf{R}(\mathbf{r}_j)^T = \mathbf{R}(\mathbf{r}_j)^{-1}$, так как поворот камеры соответствует обратному повороту сцены. Чтобы

упростить задачу уточнения можно в качестве параметров минимизации выбрать не $\mathbf{c}_j$, а $\bar{\mathbf{c}}_j = (\bar{c}_{jx}, \bar{c}_{jy}, \bar{c}_{jz}) = -\mathbf{R}(\mathbf{r}_j)^T \cdot \mathbf{c}_j$. То есть уточнять не положение камеры, а положение сцены относительно каждой камеры. Начальное значение для $\bar{\mathbf{c}}_j$: $\bar{\mathbf{c}}_j^0 = -\mathbf{R}(\mathbf{r}_j^0)^T \cdot \mathbf{c}_j^0$. Используя (26) и (27) определим функцию отображения $\bar{\mathbf{c}}_j$ и $\mathbf{r}_j$ на $\mathbf{x}_i^j$:

$$f_i^j = \begin{pmatrix} \frac{\bar{f} \cdot (\mathbf{R}(\mathbf{r}_j)^1 \mathbf{X}_i + \bar{c}_{jx})}{\mathbf{R}(\mathbf{r}_j)^3 \mathbf{X}_i + \bar{c}_{jz}} \\ \frac{\bar{f} \cdot (\mathbf{R}(\mathbf{r}_j)^2 \mathbf{X}_i + \bar{c}_{jy})}{\mathbf{R}(\mathbf{r}_j)^3 \mathbf{X}_i + \bar{c}_{jz}} \end{pmatrix}. \quad (28)$$

Здесь $\mathbf{R}(\mathbf{r}_j)^k$ $k=\{1, 2, 3\}$ – k -й столбец матрицы $\mathbf{R}(\mathbf{r}_j)$, а $\bar{f}$ – нормализованное фокусное расстояние камеры АНПА (раздел 3.2.2). Используя (28) определим полный вид векторной функции $\mathbf{f}(\mathbf{P})$ (раздел 3.4.1), где вектор параметров $\mathbf{P}=(\mathbf{r}_1, \bar{\mathbf{c}}_1, \mathbf{r}_2, \bar{\mathbf{c}}_2, \dots, \mathbf{r}_m, \bar{\mathbf{c}}_m, \mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N)$:

$$\mathbf{f}(\mathbf{P}) = (f_1^1, f_2^1, \dots, f_N^1, f_1^2, f_2^2, \dots, f_N^2, \dots, f_1^m, f_2^m, \dots, f_N^m). \quad (29)$$

Более подробно вектор параметров $\mathbf{P}$ рассматривается в разделе 3.4.5. Таким образом, определены все основные сущности алгоритма раздела 3.4.1: вектор измерений $\mathbf{X}$, вектор параметров $\mathbf{P}$ и функция отображения вектора параметров на вектор измерений – $\mathbf{f}(\mathbf{X})$.

3.4.3. Составление таблицы связи точек. При расчёте связей в главе 2 учитывались только связанные особенности на парах изображений. Между тем одна особенность может прослеживаться на большем количестве изображений. Чтобы выявить группы изображений, на которых видна каждая особенность необходимо составить таблицу $\mathbf{T}$. В данной таблице каждой потенциальной трёхмерной точке $\mathbf{X}_i$, $i=1, 2, \dots, N$ координаты которой предполагается в будущем рассчитать, должны быть поставлены в соответствии координаты точек на изображениях $\mathbf{x}_i^j$, $j=1, 2, \dots, m$, содержащих данную трёхмерную точку. Каждый из m столбцов таблицы представляет изображение, а каждая из N строк хранит номера точек в массивах особенностей, найденных на изображениях.

Например, в строке i , столбце j таблицы храниться индекс $k \in \{1, 2, \dots, \lambda(j)\}$ который означает, что $\mathbf{x}_i^j = (u_k^j, v_k^j)$ (обозначения раздела 2.3.2). Поскольку каждая точка $\mathbf{X}_i$ видима лишь на нескольких (минимум двух) изображениях, большинство ячеек таблиц будут пустыми. Это следует учитывать при хранении данной таблицы в памяти.

Заполнить таблицу $\mathbf{T}$ можно пройдя в цикле по всем парам связанных точек изображений $\mathbf{p}_k^l = \mathbf{p}_k^{i(l)j(l)} = (p_{k1}^{i(l)j(l)}, p_{k2}^{i(l)j(l)}) = (p_{k1}^l, p_{k2}^l)$ (раздел 2.1 и 2.4.2), $k=1, 2, \dots, \lambda(i(l), j(l))$ а $l=1, 2, \dots, n$. Для каждого индекса $p_{k1}^{i(l)j(l)}$ и $p_{k2}^{i(l)j(l)}$ проверяются соответствующие точки $\mathbf{u}_{p_{k1}^{i(l)j(l)}}^{i(l)}$ и $\mathbf{u}_{p_{k2}^{i(l)j(l)}}^{j(l)}$ на предмет того, были ли они занесены в таблицу $\mathbf{T}$. Если ни одна из двух точек не заносилась в таблицу, то в таблице создаётся новая строка, которая инициализируется значениями -1, а номер строки сохраняется для обеих точек $\mathbf{u}_{p_{k1}^{i(l)j(l)}}^{i(l)}$ и $\mathbf{u}_{p_{k2}^{i(l)j(l)}}^{j(l)}$. В столбцы $i(l)$ и $j(l)$ сохраненной строки записываются значения $p_{k1}^{i(l)j(l)}$ и $p_{k2}^{i(l)j(l)}$ соответственно. Если одна из точек уже заносилась в таблицу, то для неё храниться номер строки, в которую следует записать другую точку. Например, если точка $\mathbf{u}_{p_{k1}^{i(l)j(l)}}^{i(l)}$ уже была записана в таблицу $\mathbf{T}$, и для неё был сохранён номер η , то в строку η в столбец $j(l)$ заносится значение $p_{k2}^{i(l)j(l)}$, а для точки $\mathbf{u}_{p_{k2}^{i(l)j(l)}}^{j(l)}$ сохраняется значение η и наоборот. Поскольку пары сопоставленных особенностей не повторяются, ситуация, когда обе точки уже есть в таблице, исключена.

Итоговая таблица $\mathbf{T}$, будет иметь N строк, каждая из которых соответствует трёхмерной точке $\mathbf{X}_i$, $i=1, 2, \dots, N$. В столбцах для каждого изображения будет записан номер особенности изображения, которая является проекцией точки $\mathbf{X}_i$. Если точка не видна на каком-то изображении, то в строке, соответствующей этой точке и в столбце, соответствующем этому

изображению будет значение минус один. На основе данных таблицы формируются векторы $\mathbf{x}_i^j$, как показано выше.

3.4.4. Инициализация координат точек дна. Поскольку в задаче уточнения положения и ориентации изображений участвуют не только параметры камер, но и координаты точек дна $\mathbf{X}_i$, $i=1, 2, \dots, N$, для них также необходимо найти начальные значения $\mathbf{X}_i^0 = (x_i^0, y_i^0, z_i^0)$. Для определения $\mathbf{X}_i^0$ можно воспользоваться стандартным DLT алгоритмом [57], однако использование специфики задачи позволяет использовать более простой и надёжный алгоритм.

В задаче сшивки изображений полученных с АНПА известна априорная информация о курсе γ_j , высоте над дном h_j , глубине d_j аппарата в момент получения каждого j -го изображения: $j=1, 2, \dots, m$, а также предполагается малый перепад высот рельефа в зоне, охватываемой одним снимком. Основываясь на этих данных, предложим алгоритм поиска $\mathbf{X}_i^0$.

Пусть i -я точка $\mathbf{X}_i$ видна на $j=j_1, j_2, \dots, j_{m(i)}$ изображениях. Также известна её проекция на каждое из этих изображений $\mathbf{x}_i^j$. Определим координаты трёхмерной точки, соответствующей каждой проекции отдельно. Обозначим через $\mathbf{X}_i^j$ точку поверхности дна, найденную на основе проекции $\mathbf{x}_i^j$ (30).

$$\mathbf{X}_i^j = \frac{h_j}{\bar{f}} \begin{pmatrix} \cos \gamma_j & \sin \gamma_j & 0 \\ -\sin \gamma_j & \cos \gamma_j & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{x}_i^j \\ 1 \end{pmatrix} + \mathbf{c}_j. \quad (30)$$

Здесь $\bar{f}$ – нормализованное фокусное расстояние, а $\mathbf{c}_j$ – координаты АНПА в момент получения каждого изображения. Начальное значение $\mathbf{c}_j^0$ были определены в разделе 3.2.1. Вычислив $\mathbf{X}_i^{0j}$ для всех j -ых изображений, подставив значения $\mathbf{c}_j^0$ в (30) найдём начальное приближение координат точек $\mathbf{X}_i^0$:

$$\mathbf{X}_i^0 = \frac{1}{m(i)} \sum_{j=j_1}^{j_{m(i)}} \mathbf{X}_{i,j}^{0j}. \quad (31)$$

Специальный метод определения $\mathbf{X}_i^0$ проще, быстрее и устойчивее. Несмотря на то, что данный способ даёт большую начальную ошибку при проецировании $\mathbf{X}_i^0$ на исходные изображения, точки, рассчитанные с помощью (30), (31) оказываются ближе к своим истинным положениям, чем те, которые были определены с помощью стандартного DLT метода.

3.4.5. Параметризация. Рассмотрим подробнее вектор параметров $\mathbf{P}$ (раздел 3.4.1) применительно к задаче уточнения положения и ориентации изображений. В 3.4.2 предлагалось в качестве параметров задачи ввести не только положение сцены относительно АНПА $\bar{\mathbf{c}}_j$ и углы ориентации $\mathbf{r}_j$, но и положения точек поверхности дна $\mathbf{X}_i$, $i=1, 2, \dots, N$, $j=1, 2, \dots, m$. Рассмотрим представление этих данных в векторе параметров $\mathbf{P}$.

Проще всего в векторе параметров представляются точки $\mathbf{X}_i$. Каждая точка имеет три степени свободы. Соответственно каждая из трёх компонент, составляющих вектор $\mathbf{X}_i = (x_i, y_i, z_i)$ должна войти в вектор параметров. Другой возможной параметризацией точки пространства, часто используемой в компьютерном зрении, являются однородные координаты точки $\hat{\mathbf{X}}_i = (\hat{x}_i, \hat{y}_i, \hat{z}_i, \hat{w}_i)$. Недостатком однородных координат является то, что для описания трёх степеней свободы точки используется 4 параметра. Поэтому нужно использовать специальные техники, позволяющие поддерживать постоянную длину вектора $\hat{\mathbf{X}}_i$ в процессе минимизации. Преимуществом однородных координат является возможность простого представления точек на бесконечности (с нулевой координатой $\hat{w}_i$). Также при использовании Евклидовых координат одинаковое перемещение точки в плоскости, параллельной плоскости проекции камеры, приводит к разным по модулю ошибкам при разном удалении точки от камеры. Этот недостаток отсутствует при использовании однородных координат. Благодаря тому, что перепад высот

в зоне охватываемой каждым изображением невелик, все токи поверхности дна располагаются на схожем удалении от камеры, что сводит на нет преимущества использование однородных координат. Таким образом, наилучшей параметризацией положения трёхмерных точек для рассматриваемой задачи являются три Евклидовых координаты: x_i, y_i, z_i .

Аналогичным образом представляются координаты положения сцены относительно АНПА: $\bar{\mathbf{c}}_j = (\bar{c}_{jx}, \bar{c}_{jy}, \bar{c}_{jz})$. Использование $\bar{\mathbf{c}}_j$ вместо $\mathbf{c}_j$ существенно упрощает равенство (26). При этом способ перехода от $\bar{\mathbf{c}}_j$ к $\bar{\mathbf{c}}_j$ описан в разделе 3.4.2.

Наибольший интерес представляет параметризация ориентации. В (26) для представления ориентации используется матрица поворота $\mathbf{R}(\mathbf{r}_j)$. Максимально упростить (26) можно, если в качестве параметров для описания ориентации использовать все девять элементов данной матрицы поворота. Однако такая параметризация неэффективна, поскольку поворот АНПА имеет 3 степени свободы, а не 9, как количество элементов матрицы $\mathbf{R}(\mathbf{r}_j)$.

Более эффективным способом параметризации поворота является вектор с эйлеровыми углами $\mathbf{r}_j = (\alpha_j, \beta_j, \gamma_j)$. В этом случае матрицу поворота можно определить следующим образом: $\mathbf{R}(\mathbf{r}_j)$:

$$\begin{aligned}\mathbf{R}(\mathbf{r}_j) &= \mathbf{R}_z(\gamma_j)\mathbf{R}_y(\beta_j)\mathbf{R}_x(\alpha_j), \\ \mathbf{R}_x(\alpha_j) &= \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha_j & \sin \alpha_j \\ 0 & -\sin \alpha_j & \cos \alpha_j \end{pmatrix}, \\ \mathbf{R}_y(\beta_j) &= \begin{pmatrix} \cos \beta_j & 0 & \sin \beta_j \\ 0 & 1 & 0 \\ -\sin \beta_j & 0 & \cos \beta_j \end{pmatrix}, \\ \mathbf{R}_z(\gamma_j) &= \begin{pmatrix} \cos \gamma_j & \sin \gamma_j & 0 \\ -\sin \gamma_j & \cos \gamma_j & 0 \\ 0 & 0 & 1 \end{pmatrix}.\end{aligned}$$

У представления поворота с помощью эйлеровых углов есть недостаток – два близких поворота могут быть представлены различными значениями

эйлеровых углов. Например, повороты $(0, 89^\circ, 0)$ и $(180^\circ, 89^\circ, 180^\circ)$ очень близки, однако в пространстве параметров между ними большое расстояние, что может затруднить процесс минимизации. Но учитывая тот факт, что углы α_j и β_j у АНПА колеблются в небольших пределах, такие ситуации исключены. Параметризация с помощью эйлеровых углов хорошо подходит для рассматриваемой задачи.

3.4.6. Разреженный алгоритм Левенберга–Марквардта. Самым дорогостоящим шагом алгоритма Левенберга–Марквардта является вычисление нормального уравнения (23). Размерность матрицы $\mathbf{J}^T \mathbf{J}$ определяется размерностью вектора параметров $\mathbf{P}$. В вектор параметров входит $3m$ компонент векторов $\mathbf{c}_j$, $3m$ компонент векторов $\mathbf{r}_j, j=1, 2, \dots, m$ и $3N$ компонент векторов $\mathbf{X}_i, i=1, 2, \dots, N$. Таким образом, размерность вектора параметров $6m+3N$ а для решения системы (23) необходимо выполнить $(6m+3N)^3$ операций. Поскольку количество изображений m может достигать нескольких десятков тысяч, а количество точек N нескольких десятков миллионов, система (23) напрямую неразрешима. В связи с этим необходимы специальные методы решения данной системы, учитывающие её разреженность.

Разобьем вектор параметров $\mathbf{P} \in \mathbb{R}^M$ на два вектора параметров: $\mathbf{a}$ и $\mathbf{b}$: $\mathbf{P} = (\mathbf{a}, \mathbf{b})$. Вектор $\mathbf{a} = (\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_m)$, $\mathbf{a}_j = (\mathbf{r}_j, \bar{\mathbf{c}}_j)$, а вектор $\mathbf{b} = (\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_N)$. Тогда матрица Якоби $\mathbf{J}$ будет иметь блочную структуру: $\mathbf{J} = [\mathbf{A}|\mathbf{B}]$, где блоки $\mathbf{A} = (\mathbf{A}_{kj})$ и $\mathbf{B} = (\mathbf{B}_{ki})$, $i=1, 2, \dots, N, j=1, 2, \dots, m, k=1, 2, \dots, N \cdot m$ определены в (32) и (33).

$$\mathbf{A}_{kj} = \begin{pmatrix} \frac{\partial f_m^{d(k)}}{\partial \alpha_j} & \frac{\partial f_m^{d(k)}}{\partial \beta_j} & \frac{\partial f_m^{d(k)}}{\partial \gamma_j} & \frac{\partial f_m^{d(k)}}{\partial \bar{c}_{jx}} & \frac{\partial f_m^{d(k)}}{\partial \bar{c}_{jy}} & \frac{\partial f_m^{d(k)}}{\partial \bar{c}_{jz}} \end{pmatrix}, \quad (32)$$

$$\mathbf{B}_{ki} = \begin{pmatrix} \frac{\partial f_m^{d(k)}}{\partial x_i} & \frac{\partial f_m^{d(k)}}{\partial y_i} & \frac{\partial f_m^{d(k)}}{\partial z_i} \end{pmatrix}. \quad (33)$$

Здесь функции $d(k)$ и $m(k)$ используются для определения используемой функции отображения (28) и определяются следующим образом:

$$d(k) = \left\lfloor \frac{k-1}{m} \right\rfloor, \quad m(k) = ((k-1) \bmod m) + 1.$$

Здесь $x \bmod y$ – остаток от деления числа x на y . Поскольку в реальных задачах на каждом изображении видны не все, а лишь малое число точек, большинство строк из матрицы **J** можно пропустить. Также отметим особую структуру матриц **A** и **B**. Блок $\mathbf{A}_{kj}$ содержит ненулевые элементы только в том случае, когда $d(k) = j$. Это объясняется тем, что проекция $m(k)$ -ой точки на $d(k)$ -ое изображение зависит только от параметров этого изображения $\mathbf{a}_{d(k)}$, но не зависит от параметров l -изображения $\mathbf{a}_l$, $l \neq d(k)$. Соответственно производная $f_{m(k)}^{d(k)}$ по любым параметрам, кроме параметров $\mathbf{a}_j$, $j=d(k)$ будет нулевая. Отсюда следует, что матрица **A** имеет диагональную блочную структуру. Блок $\mathbf{B}_{ki}$ содержит ненулевые элементы только в случае, когда $m(k)=i$. Проекция $m(k)$ -й точки зависит только от координат $m(k)$ -й точки $\mathbf{X}_{m(k)}$, но не зависит от координат l -й точки $\mathbf{X}_l$, $l \neq m(k)$. Производная $f_{m(k)}^{d(k)}$ по любым параметрам, кроме $\mathbf{X}_i$, $i=m(k)$ будет нулевая. Это приводит к матрице **B**, состоящей из нескольких, непересекающихся лестниц – рис. 27.

Нормальное уравнение (23), без учёта слагаемого $\lambda \mathbf{I}$ в блочной записи будет иметь вид (34).

$$\begin{pmatrix} \mathbf{A}^T \mathbf{A} & \mathbf{A}^T \mathbf{B} \\ \mathbf{B}^T \mathbf{A} & \mathbf{B}^T \mathbf{B} \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_a \\ \boldsymbol{\delta}_b \end{pmatrix} = \begin{pmatrix} \mathbf{A}^T \boldsymbol{\epsilon} \\ \mathbf{B}^T \boldsymbol{\epsilon} \end{pmatrix}. \quad (34)$$

Здесь вектор приращения параметров Δ разбит на вектор приращения параметров **a** и вектор приращения параметров **b**: $\Delta = (\boldsymbol{\delta}_a, \boldsymbol{\delta}_b)^T$.

Введём следующие обозначения: $\mathbf{U} = \mathbf{A}^T \mathbf{A}$, $\mathbf{W} = \mathbf{A}^T \mathbf{B}$, $\mathbf{V} = \mathbf{B}^T \mathbf{B}$, $\boldsymbol{\epsilon}_a = \mathbf{A}^T \boldsymbol{\epsilon}$, $\boldsymbol{\epsilon}_b = \mathbf{B}^T \boldsymbol{\epsilon}$. Добавим слагаемое $\lambda \mathbf{I}$ (23): $\mathbf{U}^* = \mathbf{U} + \lambda \mathbf{I}$, $\mathbf{V}^* = \mathbf{V} + \lambda \mathbf{I}$. С учётом новых обозначений полное нормальное уравнение (23) запишется так:

$$\begin{pmatrix} \mathbf{U}^* & \mathbf{W} \\ \mathbf{W}^T & \mathbf{V}^* \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_a \\ \boldsymbol{\delta}_b \end{pmatrix} = \begin{pmatrix} \boldsymbol{\epsilon}_a \\ \boldsymbol{\epsilon}_b \end{pmatrix}. \quad (35)$$

Умножим обе части уравнения (35) на $\begin{pmatrix} \mathbf{I} & -\mathbf{WV}^{*-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix}$:

$$\begin{pmatrix} \mathbf{U}^* - \mathbf{WV}^{*-1}\mathbf{W}^T & \mathbf{0} \\ \mathbf{W}^T & \mathbf{V}^* \end{pmatrix} \begin{pmatrix} \boldsymbol{\delta}_a \\ \boldsymbol{\delta}_b \end{pmatrix} = \begin{pmatrix} \boldsymbol{\epsilon}_a - \mathbf{WV}^{*-1}\boldsymbol{\epsilon}_b \\ \boldsymbol{\epsilon}_b \end{pmatrix}. \quad (36)$$

Благодаря данной операции умножения правый верхний блок в матрице системы (36) нулевой. Запишем верхнюю половину уравнений системы (36):

$$(\mathbf{U}^* - \mathbf{WV}^{*-1}\mathbf{W}^T)\boldsymbol{\delta}_a = \boldsymbol{\epsilon}_a - \mathbf{WV}^{*-1}\boldsymbol{\epsilon}_b. \quad (37)$$

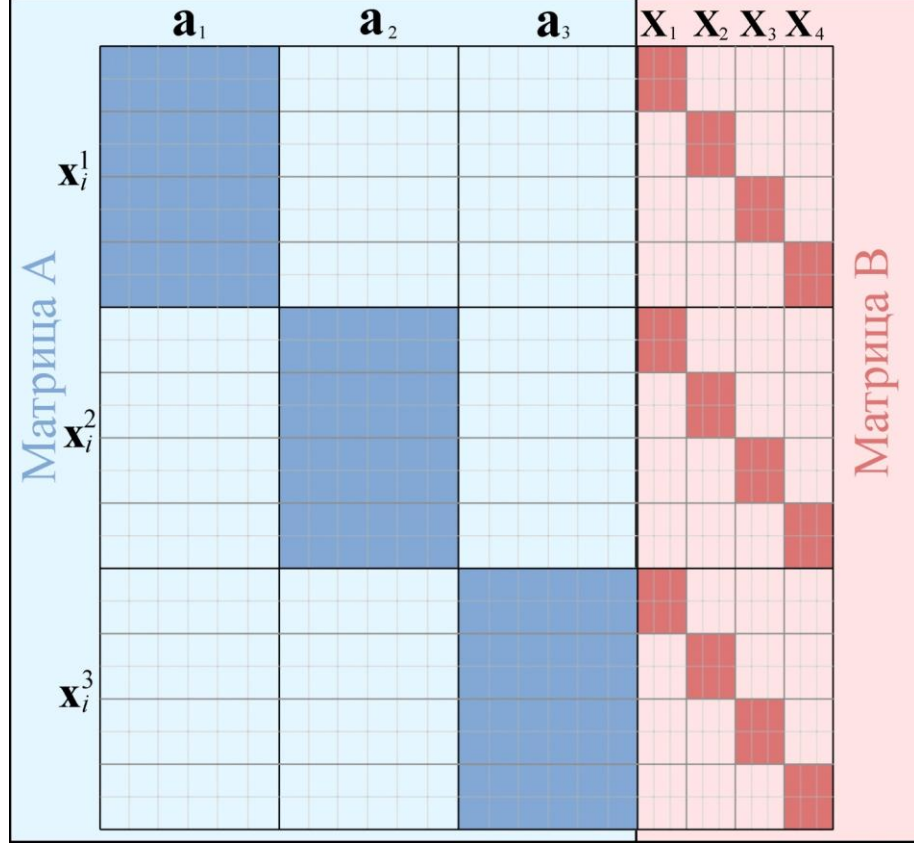


Рис. 27. Структура матриц A и B: $m=3$, $N=4$.

Система (37) может быть разрешена относительно $\boldsymbol{\delta}_a$. Потом вектор $\boldsymbol{\delta}_b$ может быть разрешён путём обратной подстановки (38).

$$\mathbf{V}^*\boldsymbol{\delta}_b = \boldsymbol{\epsilon}_b - \mathbf{W}^T\boldsymbol{\delta}_a. \quad (38)$$

Таким образом, решение самой трудоёмкой задачи алгоритма Левенберга-Марквардта на каждой итерации разбивается на независимое решение для части параметров $\mathbf{a}$ и последующее вычисление параметров части $\mathbf{b}$ с использованием $\mathbf{a}$.

Преимуществом описанного разбиения является особый вид матриц $\mathbf{U}$ и $\mathbf{V}$. Элементы данных матриц получаются путём перемножения строк матриц $\mathbf{A}$

и **B** соответственно. Из особенностей структуры матриц **A** и **B** (рис. 27) следует, что матрица **U** и матрица **V** являются блочно диагональными матрицами: $\mathbf{U}=\text{diag}(\mathbf{U}_1, \mathbf{U}_2, \dots, \mathbf{U}_m)$, $\mathbf{V}=\text{diag}(\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_N)$. Структура нормального уравнения (35), учитывающая особый вид матриц **U** и **V** представлена на рис. 28.

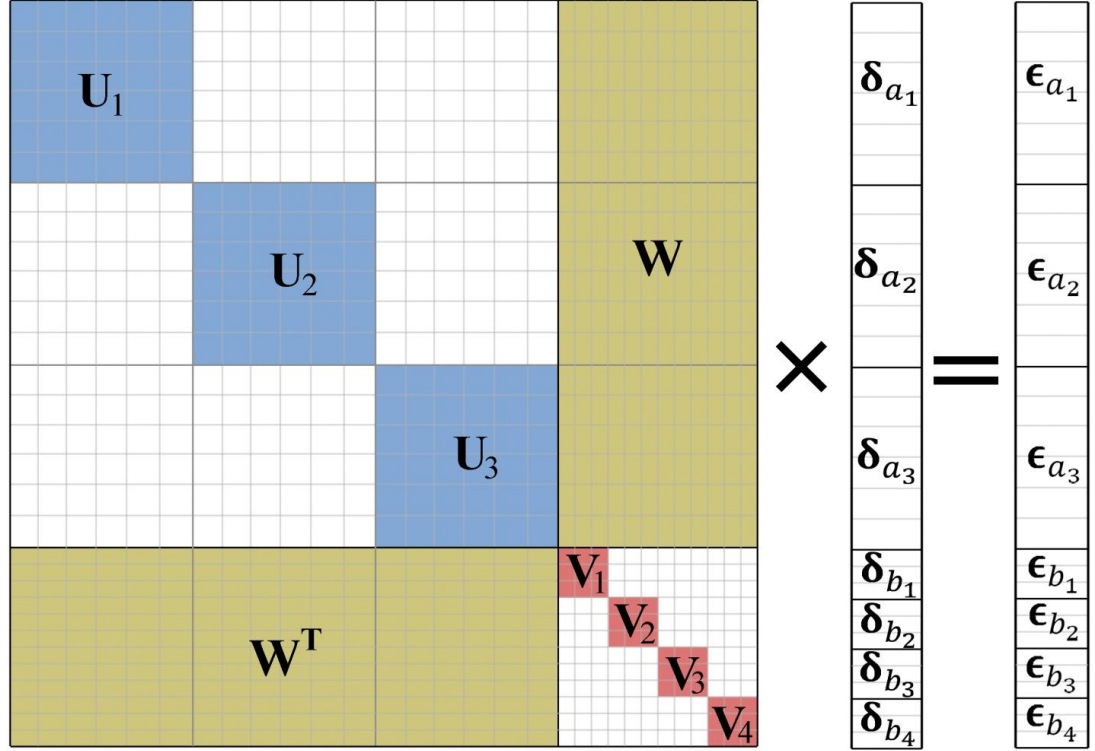


Рис. 28. Структура уравнения (35): $m=3$, $N=4$.

Запишем последовательность вычисления одной итерации алгоритма Левенберга–Марквардта, учитывающего разреженность матрицы **J** и факт того, что на каждом изображении видны лишь небольшое количество точек [57]. Далее индексы $i=1, 2, \dots, N$, $j=1, 2, \dots, m$. Множество $I(j)$ – множество номеров точек $\mathbf{X}_i$, видимых на j -ом изображении, а $J(i)$ – множество номеров изображений, на которых видна i -я точка. С учётом этих обозначений алгоритм вычисления вектора приращения на одной итерации алгоритма Левенберга–Марквардта запишется следующим образом:

Шаг 1: Вычислить матрицы производных:

$$\mathbf{A}_{ij} = \left(\frac{\partial f_i^j}{\partial \alpha_j} \quad \frac{\partial f_i^j}{\partial \beta_j} \quad \frac{\partial f_i^j}{\partial \gamma_j} \quad \frac{\partial f_i^j}{\partial \bar{c}_{jx}} \quad \frac{\partial f_i^j}{\partial \bar{c}_{jy}} \quad \frac{\partial f_i^j}{\partial \bar{c}_{jz}} \right), \quad i \in I(j),$$

$$\mathbf{B}_{ij} = \begin{pmatrix} \frac{\partial f_i^j}{\partial x_i} & \frac{\partial f_i^j}{\partial y_i} & \frac{\partial f_i^j}{\partial z_i} \end{pmatrix}, \quad i \in I(j).$$

Шаг 2: Рассчитать вектор ошибки:

$$\epsilon_{ij} = \mathbf{x}_i^j - f_i^j, \quad i \in I(j).$$

Шаг 3: Вычислить матрицы камер и точек:

$$\mathbf{U}_j = \sum_{k \in I(j)} \mathbf{A}_{kj}^T \mathbf{A}_{kj}, \quad \mathbf{U}_j^* = \mathbf{U}_j + \lambda \frac{\text{Tr } \mathbf{U}}{6m} \cdot \mathbf{E}_{6 \times 6}.$$

$$\mathbf{V}_i = \sum_{k \in J(i)} \mathbf{B}_{ik}^T \mathbf{B}_{ik}, \quad \mathbf{V}_i^* = \mathbf{V}_i + \lambda \frac{\text{Tr } \mathbf{V}}{3N} \cdot \mathbf{E}_{3 \times 3}.$$

Здесь $\mathbf{E}_{a \times b}$ – единичная матрица размерности $a \times b$.

Шаг 4: Рассчитать вспомогательные матрицы и векторы:

$$\mathbf{W}_{ij} = \mathbf{A}_{ij}^T \mathbf{B}_{ij}, \quad i \in I(j),$$

$$\epsilon_{a_j} = \sum_{k \in I(j)} \mathbf{A}_{kj}^T \epsilon_{kj},$$

$$\epsilon_{b_i} = \sum_{k \in J(i)} \mathbf{B}_{ik}^T \epsilon_{ik},$$

$$\mathbf{Y}_{ij} = \mathbf{W}_{ij} \mathbf{V}_i^{*-1}, \quad i \in I(j).$$

Шаг 5: Рассчитать вектор $\delta_a = (\delta_{a_1}, \delta_{a_2}, \dots, \delta_{a_m})$ из системы линейных алгебраических уравнений $\mathbf{S} \delta_a = (\mathbf{e}_1, \mathbf{e}_2, \dots, \mathbf{e}_m)$:

$$\mathbf{S}_{jk} = \begin{cases} - \sum_{l \in I(j) \cup I(k)}^n \mathbf{Y}_{lj} \mathbf{W}_{lk}^T + \mathbf{U}_j^*, & j = k \\ - \sum_{l \in I(j) \cup I(k)}^n \mathbf{Y}_{lj} \mathbf{W}_{lk}^T, & j \neq k \end{cases}, \quad k = 1, 2, \dots, m,$$

$$\mathbf{e}_j = \epsilon_{a_j} - \sum_{i \in I(j)} Y_{ij} \epsilon_{b_i}.$$

Шаг 6: Рассчитать векторы δ_{b_i} :

$$\delta_{b_i} = \mathbf{v}_i^{*-1} \left(\epsilon_{b_i} - \sum_{k \in \mathcal{I}(i)} \mathbf{w}_{ik}^T \delta_{a_k} \right).$$

На шаге 5 рассмотренного алгоритма решается система линейных алгебраических уравнений с матрицей $\mathbf{S}$. Данная матрица по структуре аналогична матрице $\mathbf{Q}$ из раздела 3.2.2, поэтому к ней применимы методики эффективного решения, учитывающие разреженность, рассмотренные в разделе 3.2.3. Используя эти методики становится возможным решение задачи глобального уточнения параметров для десятков тысяч изображений и десятков миллионов точек.

3.5. Результаты

В третьей главе были рассмотрены задачи поиска начального приближения планарных координат и задача уточнения параметров, задающих положение и ориентацию АНПА в пространстве в момент получения каждого снимка. Поскольку обе эти задачи сводятся к решению больших систем линейных алгебраических уравнений, также в главе рассматривался метод эффективного решения этих систем, учитывающий их структуру.

3.5.1. Исходные данные для тестирования. Рассмотрим начальные данные и условия тестирования алгоритмов, описанных в третьей главе. К упорядочиванию изображений, получаемых с АНПА, будем применять четыре алгоритма: обратный алгоритм Катхилла–Макки (ОКМ) (раздел 3.3.3), модифицированный алгоритм вложенных сечений $a=0.35$ (BC1), модифицированный алгоритм вложенных сечений для диапазона a (BC2) (раздел 3.3.5), алгоритм параллельных сечений (ПС) (раздел 3.3.6).

Возьмём шесть различных областей, покрытых изображениями. Две области, показанные на рис. 29а, 29б – фотографии морского дна, полученные с автономного необитаемого подводного аппарата (АНПА). Другие четыре области представляют синтетические данные, сгенерированные для проверки

эффективности рассматриваемых алгоритмов в различных ситуациях, которые могут возникнуть на практике (рис. 29в - 29е).

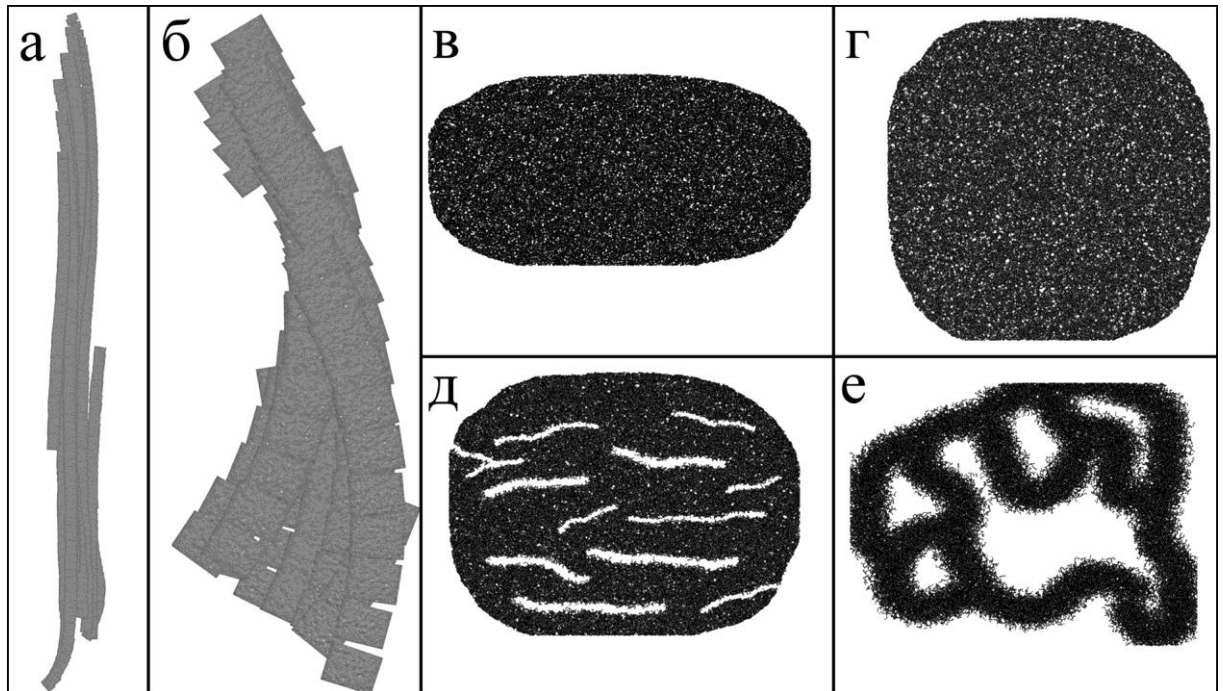


Рис. 29. Набор тестовых областей, покрытых изображениями.

Области на рис. 29 будут использованы для тестирования алгоритмов быстрого решения СЛАУ в задаче поиска начального приближения планарных координат (раздел 3.3). Для тестирования качества работы самого алгоритма поиска начального приближения планарных координат снимков (раздел 3.2) и алгоритма уточнения положения и ориентации АНПА (раздел 3.4) будет использоваться максимальная по размеру область покрытия снимков, доступная на момент написания работы – рис. 30. Это область включает в себя область 29а и содержит ряд дополнительных изображений. Область на рисунке 30 характеризуется наибольшим количеством обнаруженных пар связанных изображений и соответствующих особенностей.

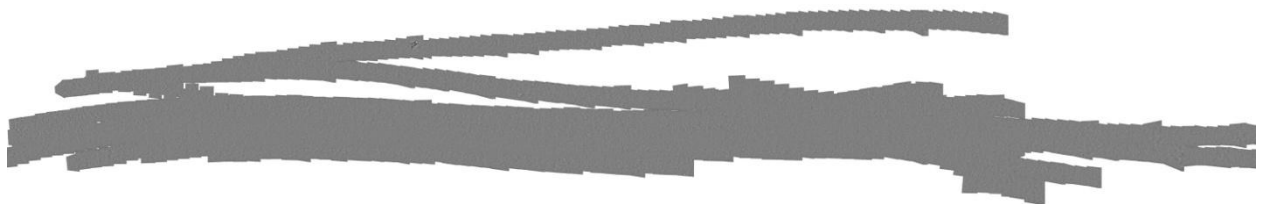


Рис. 30. Тестовая область для алгоритмов вычисления ориентации и положения изображений.

Отметим, что моделирование синтетических тестовых данных для алгоритма уточнения положения и ориентации АНПА затруднено ввиду сложности процесса моделирования распределения точечных особенностей, согласующегося с реальными изображениями морского дна. Поэтому для тестирования алгоритма уточнения синтетические области использоваться не будут.

Таблица 1 содержит подробное описание используемых для тестирования наборов изображений. Каждая из четырёх представленных на рисунке 29 синтетических областей формирует четыре задачи. Два вида задач – lo и hi характеризуются числом изображений m . Для задач типа lo оно низкое: $23000 \div 25000$, а для задач hi – высокое: $90000 \div 100000$. Задачи типа lo используются для тестирования алгоритмов упорядочения и решения систем уравнений. Задачи типа hi используются лишь для тестирования эффективности алгоритмов упорядочения из раздела 3.3.

Другие два вида задач – $weak$ и $strong$. Задачи $weak$ характеризуются малым числом n связей между изображениями. Отношение числа связей к числу изображений в $weak$ -задачах не превышает 3.6.

В общем случае $weak$ -задача соответствует идеальному расположению изображений: они имеют достаточное для решения задачи, но не избыточное перекрытие. Однако на практике часто бывают ситуации, когда одна и та же зона покрывается фотографиями большое количество раз. Это может случиться, когда АНПА погружается в воду, не продвигаясь вперёд, когда АНПА движется медленно, или когда АНПА многократно проходит над одной точкой (точкой схода траекторий). В этом случае рассматриваются $strong$ -задачи. В них отношение числа связей к числу изображений достигает 13.5. Таким образом, для каждой синтетической области рассматриваются четыре задачи: $weak_lo$, $weak_hi$, $strong_lo$ и $strong_hi$.

Таблица 1

Название	Описание	m	n	Рисунок
R1	Сходящийся пучок траекторий АНПА	90	216	Рис. 29б
R2	Вытянутый участок морского дна	647	1904	Рис. 29а
S1_weak_lo	Вытянутый участок территории, полностью покрытый изображениями	24209	62641	Рис. 29в
S1_weak_hi		91748	219517	
S1_strong_lo		25000	252201	
S1_strong_hi		100000	920871	
S2_weak_lo	Округлый участок территории, полностью покрытый изображениями	24266	64268	Рис. 29г
S2_weak_hi		97360	257078	
S2_strong_lo		25000	257576	
S2_strong_hi		100000	1038573	
S3_weak_lo	Участок территории с маленькими зонами, непокрытыми фотографиями	23749	67968	Рис. 29д
S3_weak_hi		92273	233430	
S3_strong_lo		25000	274273	
S3_strong_hi		99979	1228105	
S4_weak_lo	Участок территории с большими зонами, непокрытыми фотографиями	23477	82797	Рис. 29е
S4_weak_hi		91938	329076	
S4_strong_lo		24911	331963	
S4_strong_hi		99390	1335892	
R3	Зона R2 плюс дополнительные связанные изображения	1044	2696	Рис. 30

3.5.2. Алгоритмы решения СЛАУ. При тестировании все алгоритмы выполнялись на одном ядре процессора Intel Core2 Quad Q9550 2.83GHz. В качестве основного критерия эффективности рассмотренных алгоритмов было взято общее время решения системы уравнений. Таблица 2 отображает время решения системы t_s для каждой из описанных областей каждым алгоритмом, а также время упорядочения t_o .

Таблица 2

Название	ОКМ		BC1		BC2		ПС	
	t_o , с	t_s , с	t_o , с	t_s , с	t_o , с	t_s , с	t_o , с	t_s , с
R1	0.001	0.010	0.001	0.015	0.001	0.013	0.001	0.016
R2	0.001	0.109	0.001	0.260	0.047	0.250	0.001	0.150
S1_weak_lo	0.014	3.984	0.156	8.078	5.063	5.469	0.016	11.422
S1_weak_hi	0.032	-	1.079	-	30.235	-	0.062	-
S1_strong_lo	0.015	32.140	0.313	198.485	7.953	152.593	0.032	148.000
S1_strong_hi	0.047	-	1.843	-	61.844	-	0.125	-
S2_weak_lo	0.015	6.797	0.187	8.641	6.187	8.172	0.015	20.156
S2_weak_hi	0.032	-	1.079	-	30.235	-	0.062	-
S2_strong_lo	0.015	65.125	0.296	198.765	7.969	185.937	0.016	197.109
S2_strong_hi	0.078	-	2.047	-	58.250	-	0.140	-
S3_weak_lo	0.016	4.256	0.141	5.922	4.297	5.907	0.015	11.688
S3_weak_hi	0.031	-	0.812	-	27.016	-	0.047	-
S3_strong_lo	0.016	49.391	0.281	85.844	6.812	77.718	0.016	151.812
S3_strong_hi	0.078	-	2.063	-	81.015	-	0.140	-
S4_weak_lo	0.015	4.406	0.125	7.109	3.359	7.094	0.015	11.485
S4_weak_hi	0.047	-	0.922	-	26.781	-	0.062	-
S4_strong_lo	0.015	40.781	0.281	82.454	7.484	76.656	0.016	139.375
S4_strong_hi	0.078	-	2.078	-	55.688	-	0.156	-

Из таблицы 2 видно, что для всех задач обратный алгоритм Катхилла–Макки дал наилучшие результаты, как по времени решения системы, так и по времени упорядочения. Отчасти это объясняется тем, что данный алгоритм не только минимизирует заполнение, но также даёт малую ширину диагонали w (20), что упрощает шаг 3 алгоритма 1 (раздел 3.3.1).

Очевидно, что при наличии информации о вертикальной структуре матрицы системы T , обратный алгоритм Катхилла–Макки лишится данного преимущества. Поэтому, в качестве ещё одного критерия оценки эффективности описанных алгоритмов приведём заполнение матрицы системы

после упорядочения (таблица 3). Для удобства в таблицу 3 для каждой задачи занесено не само заполнение η (19) (η' (22) для алгоритма ПС), а величина $f=\eta/(4m)$ ($f=\eta'/(4m)$ для алгоритма ПС). Эта величина характеризует среднее расстояние между первым ненулевым элементом каждой строки и главной диагональю.

Кроме информации о заполнении, в таблице 3 присутствует информация об оперативной памяти M , требующейся для решения каждой задачи каждым алгоритмом.

Из таблицы 3 видно, что выигрывающий по времени обратный алгоритм Катхилла–Макки далеко не всегда даёт лучшее заполнение. В большинстве weak-задач модифицированные алгоритмы вложенных сечений BC1 и BC2 дали лучшие результаты. Однако на strong-задачах, а также на задачах R1 и R2, для которых характерна очень узкая структура уровней, обратный алгоритм Катхилла–Макки по-прежнему лидирует.

Таблица 3

Название	ОКМ		BC1		BC2		ПС	
	M , Мб	f	M , Мб	f	M , Мб	f	M , Мб	f
R1	0.011	8	0.021	16	0.019	14	0.020	14
R2	0.156	16	0.347	32	0.333	34	0.286	30
S1_weak_lo	52.126	142	66.079	178	42.506	116	86.137	232
S1_weak_hi	334.645	240	250.725	180	171.547	122	510.300	364
S1_strong_lo	196.091	514	468.369	1228	413.666	1084	392.806	1030
S1_strong_hi	1460.042	956	3265.106	2140	3032.128	1988	2550.767	1672
S2_weak_lo	77.374	208	67.960	184	63.648	172	124.098	336
S2_weak_hi	612.313	412	569.415	384	519.744	350	920.646	620
S2_strong_lo	277.973	726	478.717	1256	462.212	1212	449.822	1180
S2_strong_hi	2245.680	1472	3740.140	2452	3645.476	2390	3803.769	2492
S3_weak_lo	53.235	148	45.792	126	44.251	122	83.664	230
S3_weak_hi	322.112	228	210.117	150	177.527	126	513.800	366
S3_strong_lo	241.962	634	309.085	810	296.054	776	396.088	1038
S3_strong_hi	2036.948	1336	3242.107	2126	2713.446	1778	3297.019	2162

Название	ОКМ		BC1		BC2		ПС	
	$M, M6$	f	$M, M6$	f	$M, M6$	f	$M, M6$	f
S4_weak_lo	56.105	156	58.282	162	58.212	162	87.508	244
S4_weak_hi	433.399	310	523.666	374	475.545	340	675.012	482
S4_strong_lo	212.899	560	300.517	790	290.664	764	375.373	986
S4_strong_hi	1976.542	1304	2555.019	1684	2415.144	1592	3526.447	2326

На основе исследования эффективности описанных алгоритмов для рассматриваемых задач, сводящихся к решению линейной (раздел 3.2.2) и нелинейной задач МНК (раздел 3.4), можно сделать следующие выводы. Алгоритм параллельных сечений не оказался лучшим ни для одной из рассматриваемых областей. Кроме того, данному алгоритму необходима особая структуру данных для хранения матрицы системы, а также модификация алгоритма 1, учитывающая эту структуру.

Алгоритм вложенных сечений в исходном варианте [4] также не показал хороших результатов (информация о нём не внесена в таблицы 2 и 3), однако модифицированные алгоритмы вложенных сечений, предложенные в данной работе, оказались самыми эффективными при минимизации заполнения в задачах, где изображения слабо связаны. Данные алгоритмы, можно рекомендовать при необходимости многократного решения систем с одинаковой структурой, например в нелинейных задачах МНК. Вертикальная структура T в таком случае может быть рассчитана один раз и использована для решения всех систем. Это даёт линейную зависимость времени решения этих систем от заполнения η .

Наиболее подходящим для решения рассматриваемых задач показал себя обратный алгоритм Катхилла–Макки. Им следует пользоваться, когда задача не предполагает многократного решения систем уравнений с одинаковой структурой, например в линейных задачах МНК (раздел 3.2.2). Также данный алгоритм даёт наилучшие результаты в задачах с очень плотным покрытием изображениями и в задачах, где область, покрываемая изображениями, узкая.

При необходимости многократного решения задач с однотипной структурой, а особенно, с малым количеством связей, можно рекомендовать модифицированный алгоритм вложенных сечений BC1 или BC2.

Применение алгоритма Катхилла–Макки или модифицированных алгоритмов вложенных сечений совместно с методом решения систем, учитывающим разреженность, позволяет за разумное время решать на современном персональном компьютере задачи компьютерного зрения с десятками тысяч связанных изображений.

3.5.3. Задача поиска приближения планарных координат. Оценка результатов работы алгоритма поиска приближенных планарных координат снимков (раздел 3.2) проводилась на основе области R3 (Таблица 1). Качество работы алгоритма может быть оценено путём анализа ошибок $|e_l|$

$$|e_l| = \sqrt{\left(\mathbf{d}^l - \begin{pmatrix} c_{i(l)x}^0 \\ c_{i(l)y}^0 \end{pmatrix} + \begin{pmatrix} c_{j(l)x}^0 \\ c_{j(l)y}^0 \end{pmatrix} \right)^2}.$$

Здесь использованы обозначения раздела 3.2.2. Расстояние $|e_l|$ показывает, насколько согласуется вектор, соединяющий рассчитанные центры изображений и вектор $\mathbf{d}^l$. На рис. 31 представлена гистограмма распределения связей по ошибкам. Из гистограммы видно, что наибольшая часть связей не соответствует векторам $\mathbf{d}^l$ не более, чем на 0.05 метров, что является хорошим результатом для приближённого вычисления планарных координат. Математическое ожидание измеряемой ошибки составило 0.022 метра, что также является хорошим результатом.

Из гистограммы на рис. 31 также видно, что на наибольший диапазон ошибок (от 0.06 до 0.15) приходится очень малое количество связей – около 100. Это даёт возможность фильтровать связи с большой ошибкой после процедуры поиска приближенных планарных координат. Задача Вычисления приближенных планарных координат может быть решена заново после исключения отфильтрованных связей. Данная процедура приведёт к более

точному и устойчивому решению ценой возможной потери корректных, но слабых связей. Исследования алгоритмов фильтрации ложных соответствий по значению ошибки $|e_l|$ в данной работе не проводилось, но планируется для дальнейших исследований в этой области.

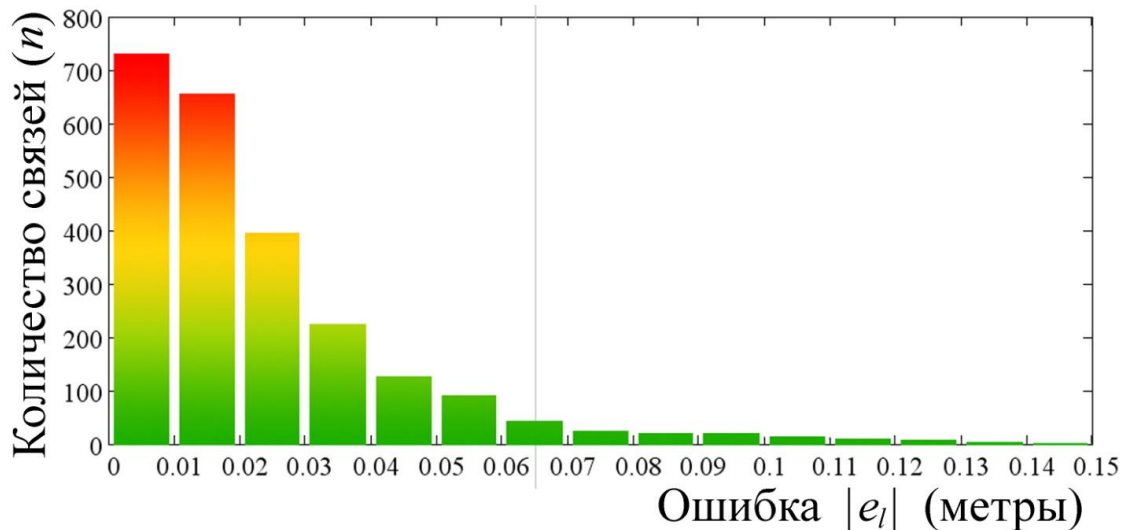


Рис. 31. Гистограмма распределения связей по ошибкам $|e_l|$.

3.5.4. Задача уточнения положения и ориентации АНПА. Для исследования алгоритмов уточнения положения и ориентации АНПА (раздел 3.4) использовалась область R3 (таблица 1) и рассчитанные приближённые планарные координаты снимков (раздел 3.5.3). Также использовалась информация о найденных соответствующих особенностях (глава 2). Всего для 1044 снимков области R3 было обнаружено 332061 проекций связанных особых точек, принадлежащих 157590 точкам поверхности дна.

Для оценки качества рассматриваемого алгоритма будем использовать значение минимизируемой величины $\|\epsilon\|$ (раздел 3.4). График на рис. 32 показывает, как уменьшается данная величина на каждой следующей итерации алгоритма.

Напомним, что величина $\|\epsilon\|$ - есть сумма квадратов расстояний между рассчитанными положениями проекций точек пространства на изображения и положениями особых точек, определёнными в раздел 2.3. Отсюда использование квадратных метров при измерении $\|\epsilon\|$. Из рис. 32 видно, что

величина $\|\epsilon\|$ стремительно уменьшается от значения в 2700 м^2 до очень малого значения всего за 10 итераций.

Для более детальной оценки результата работы алгоритма итеративного уточнения положений и ориентаций АНПА рассмотрим гистограмму (рис. 33) распределение всех 332061 найденных проекций точек пространства по несогласованностям с вычисленной моделью. За несогласованность проекции примем расстояние в метрах между положением данной проекции, полученным в результате обнаружения особенностей (раздел 2.3) и положением, рассчитанным по формуле (28). Поскольку для каждого изображения известна высота над дном в метрах, с которой это изображение получено и фокусное расстояние камеры, измерение расстояний на изображении может проводиться как в метрах, так и в пикселях. Однако, оценка в метрах ближе для человеческого понимания.

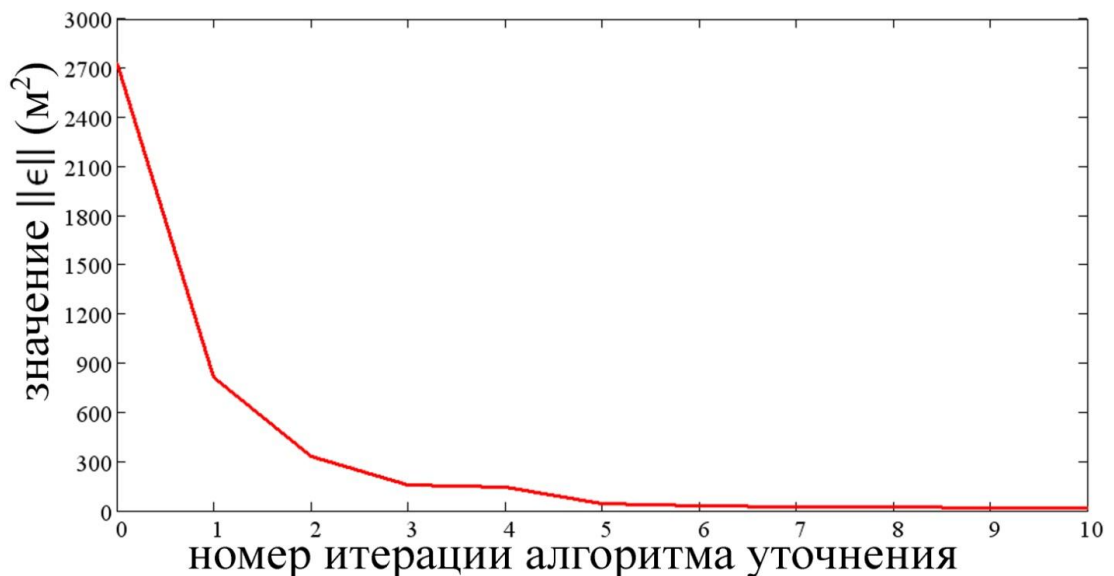


Рис. 32. Уменьшение величины $\|\epsilon\|$ в процессе минимизации.

Из гистограммы на рис. 33 видно, что после применения алгоритма итеративного уточнения, максимальная несогласованность для проекций не превышает 1 см, а средняя несогласованность составляет порядка 3 мм. Большая часть проекций сосредоточена в левой части гистограммы, в зоне малой несогласованности. При среднем размере ширины зоны, охватываемой изображением в 1 – 2 метра, средняя несогласованность в 3 мм вполне

допустима. Изображения, позиционированные и ориентированные с такой точностью, могут быть легко сшиты, чему посвящена четвёртая глава данной работы.

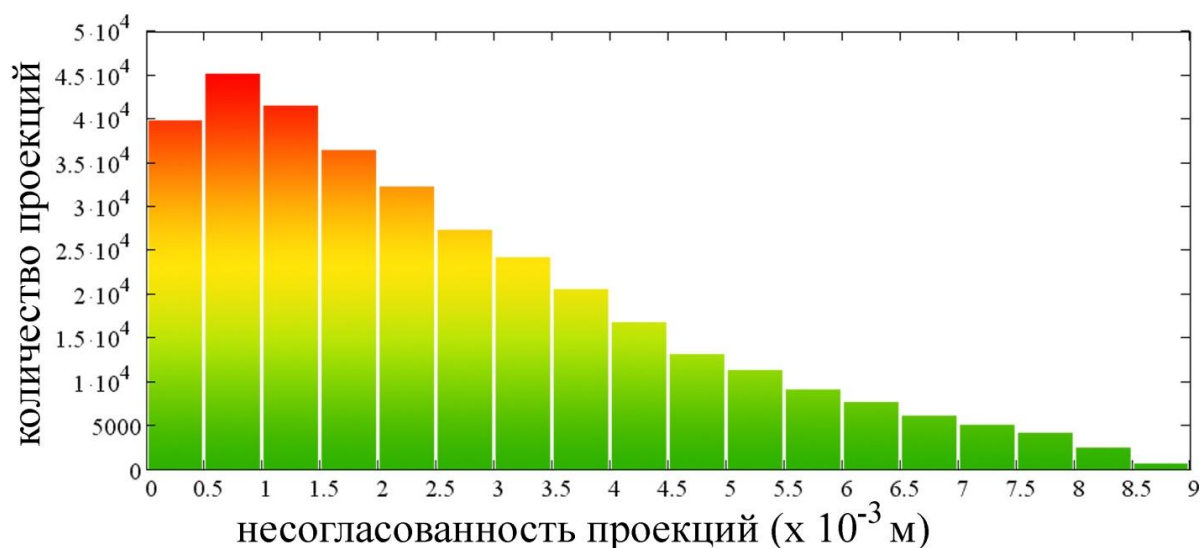


Рис. 33. Распределение проекций по несогласованностям.

Пространственное размещение точек поверхности дна и положения камеры АНПА в момент получения каждого изображения для области R3 представлены на рис. 34. Чёрными точками отображаются положения АНПА в момент фотографирования, а зелёными линиями — конусы камер, показывающие ориентацию камер в момент фотографирования. Красным цветом изображены точки поверхности подводного дна. Рис. 34а показывает все точки и камеры для области R3. На рис. 34б. отображены только точки поверхности без камер, а рис. 34в — увеличенный фрагмент области R3.

Кроме основы для дальнейшей сшивки изображений, результат работы рассматриваемого алгоритма итеративного уточнения может быть применён для построения трёхмерной реконструкции подводного дна.

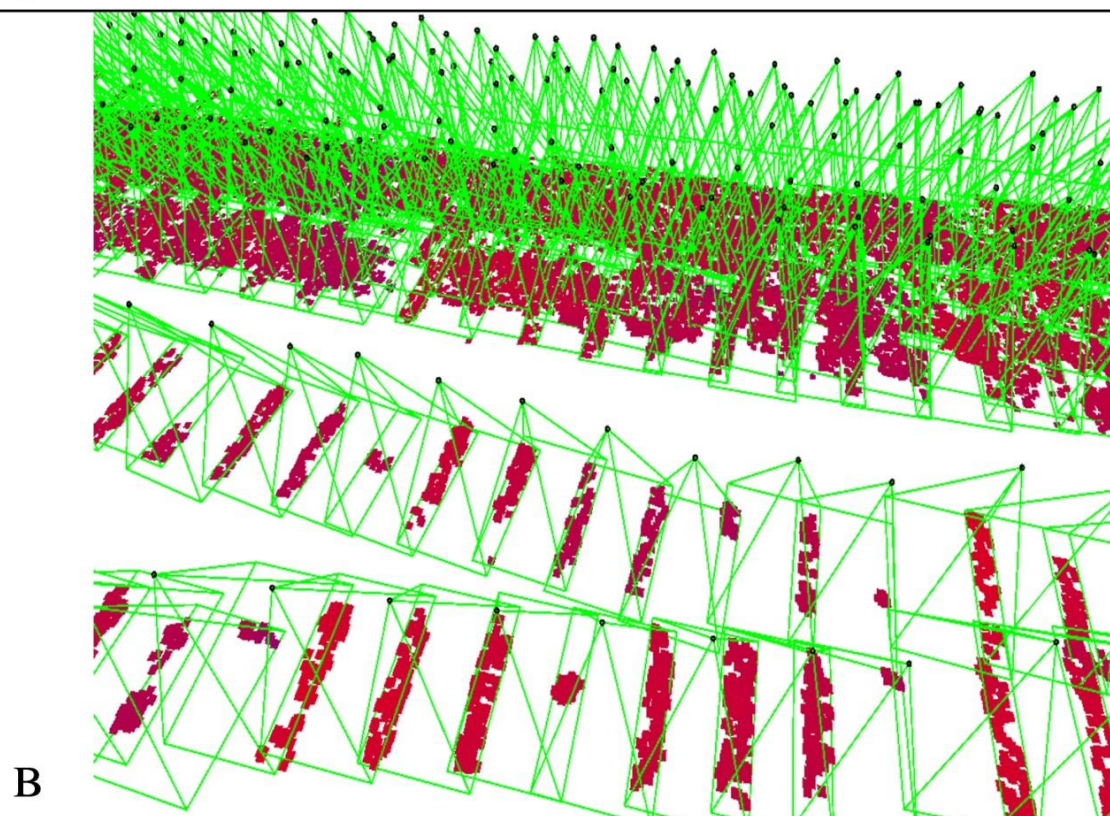
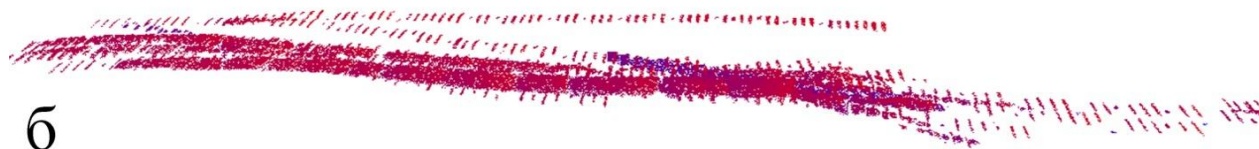
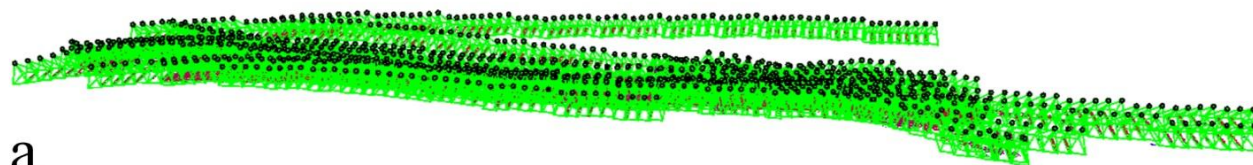


Рис. 34. Общий вид точек и камер, рассчитанных с помощью алгоритма итеративного уточнения.

ГЛАВА 4. ВИЗУАЛЬНОЕ ПРЕДСТАВЛЕНИЕ ФОТОГРАФИЧЕСКОЙ КАРТЫ ДНА

Финальной задачей процедуры построения фотографической карты дна является сшивка и смешение фотографий. В результате решения этой задачи получается бесшовное полотно, содержащее изображения в обследованных зонах. Именно с таким полотном – картой удобно работать исследователям дна. Описанию процедуры сшивки и смешения ориентированных и позиционированных изображений посвящена данная глава. В четвёртой главе рассматривается два варианта организации процесса сшивки: простая плоская сшивка, подходящая для относительно ровных поверхностей дна (раздел 4.2) и сшивка на основе трёхмерной модели дна, подходящая для рельефа с большими перепадами высот (раздел 4.3).

Также в четвёртой главе рассматривается простой и эффективный способ устранения швов на стыках сшиваемых изображений, пригодных для больших массивов входных данных (раздел 4.4).

В разделе 4.5 приведены основные результаты процедуры сшивки и смешения изображений. Описываются исходные данные для тестирования и проводится тестирование алгоритмов из разделов 4.2, 4.3 и 4.4.

4.1. Постановка задачи

Определим набор входных данных для задачи сшивки и смешения изображений. Как и в третьей главе через m обозначим общее количество изображений, а через N рассчитанное в разделе 3.4.3 количество точек пространства $\mathbf{X}_i = (x_i, y_i, z_i)$, $i=1, 2, \dots, N$.

Для решения задачи понадобится таблица $\mathbf{T}$, построенная в разделе 3.4.3. Необходимы будут не столько индексы, хранящиеся в этой таблице, сколько данные о том, на каких изображениях видима каждая i -я точка пространства. Напомним, что если точка $k \in \{1, 2, \dots, N\}$ не видима на p -ом изображении, $p \in \{1, 2, \dots, m\}$, то $T_{kp} = -1$. В случае, когда точка k видна на p -ом изображении

элемент T_{kp} содержит индекс в список экстремумов p -го изображения, который не может равняться -1 : $T_{kp} \neq -1$. Используя данный факт, будем определять видимость каждой точки для каждого изображения.

Кроме таблицы T для сшивки понадобится вектор P (раздел 3.4.2), полученный после работы алгоритма уточнения (раздел 3.4). Данный вектор состоит из координат точек пространства X_i , параметров, характеризующих положение сцены относительно камеры АНПА в момент получения j -го изображения, $j=1, 2, \dots, m$: $\bar{c}_j = (\bar{c}_{jx}, \bar{c}_{jy}, \bar{c}_{jz})$ и параметров, характеризующих ориентацию камеры АНПА: r_j . Напомним, что ориентация задаётся с помощью трёх углов $r_j = (\alpha_j, \beta_j, \gamma_j)$, а матрица поворота, соответствующая этой ориентации $R(r_j)$ была определена в разделе (3.4.5). В главе 4 будем обращаться как к повороту сцены относительно камеры: $R(r_j)$, так и к повороту камеры относительно сцены: $R(r_j)^T$. Аналогично будем использовать положение сцены относительно камеры $\bar{c}_j$ и положение камеры относительно сцены $c_j = -R(r_j)\bar{c}_j$.

Кроме описанных данных будут использованы знания о нормализованном фокусном расстоянии каждого изображения $\bar{f}_j$ (измерения в пространстве изображений по-прежнему проводятся в нормализованной системе координат – раздел 2.2), высоту камеры АНПА в момент фотографирования h_j и размеры выходных изображений в пикселях w_j и h_j . Эти данные понадобятся для масштабирования входных изображений.

Используя описанные исходные данные необходимо спроецировать каждый пиксель каждого j -го изображения на общую плоскость $Z=0$. Плоскость $Z=0$ необходимо поставить в соответствие с изображением – фотографической картой подводного дна. Цвет каждого пикселя данного изображения определить как некоторую взвешенную сумму цветов проекций пикселей входных изображений на плоскость $Z=0$. Коэффициенты взвешенной суммы должны выбираться таким способом, чтобы исключить появление швов между

изображениями на итоговой фотографической карте. Также необходимо разработать способ адекватного хранения и представления фотографических карт большого разрешения.

4.2. Алгоритм простой сшивки

В большинстве случаев рельеф подводного дна имеет плавные перепады высот. Под плавным перепадом будем понимать такой перепад, при котором отклонения по высоте в зоне, ограниченной одним снимком, составляет не более пяти процентов диагонали этого снимка. Рис. 35 демонстрирует пример плавного (рис. 35а) и резкого перепада высот (рис. 35б).

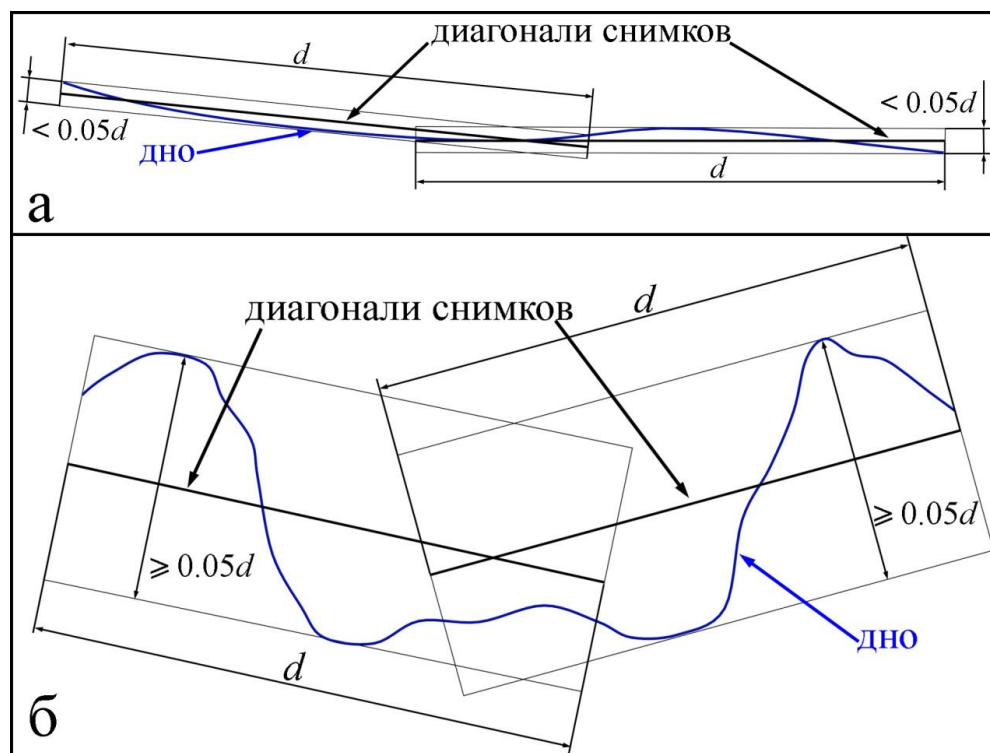


Рис. 35. Плавный (а) и резкий (б) перепад высот дна.

Если известно, что рассматриваемый набор входных изображений описывает дно с плавным перепадом высот, то задачу сшивки можно упростить, приняв участок дна, видимый на каждом изображении за плоскость. Данное упрощение слабо повлияет на результат сшивки. В этом можно убедиться, рассмотрев рис. 36.

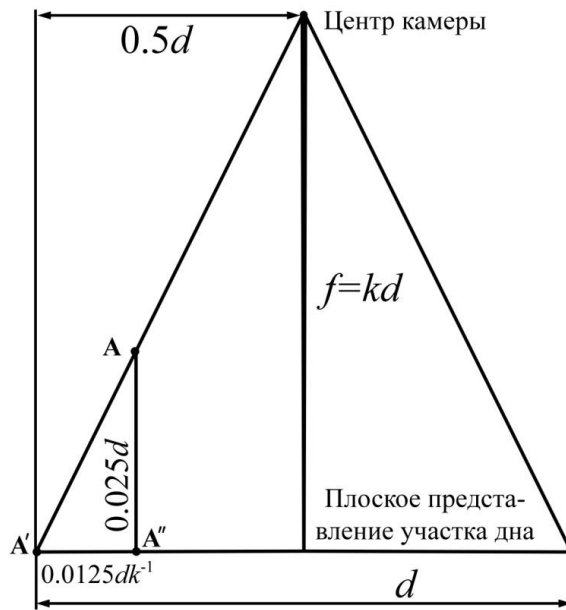


Рис. 36. Примерная ошибка при плоском представлении.

Максимальная ошибка, связанная с плоским представлением участка поверхности, охватываемого изображением, будет наблюдаться в углах изображения (благодаря тому, что расстояние от центра снимка до углов максимально). Рассмотрим подробнее рис. 36. На нём диагональ изображения обозначена за d , а фокусное расстояние камеры указывается в длинах диагонали: $f=kd$, где k – некоторый коэффициент. Согласно условию плавности рельефа, максимально допустимый перепад рельефа составляет $0.05d$. Если плоскость, аппроксимирующая участок рельефа располагается на средней высоте, максимальная высота некоторой точки рельефа A , над аппроксимирующей плоскостью равна $0.025d$ (половина перепада высот). На аппроксимирующую плоскость точка A спроецируется в точку A' , тогда как должна в точку A'' . Расстояние между точками A' и A'' определяет ошибку проецирование и составляет $0.0125dk^{-1}$ для угловых точек изображения. Для обычных камер фокусное расстояние приблизительно равно диагонали изображения, т.е. $k \approx 1$. Таким образом, ошибка, связанная с аппроксимацией участка рельефа плоскостью составляет всего 0.0125 от длины диагонали изображения. Ошибка в 1.25% диагонали мала, однако, уже визуально заметна, но благодаря использованию методики смещения изображений, описанной в

разделе 4.4, вклад пикселей изображений, близких к границам оказывается минимальным. Основным материалом для формирования фотографической карты – центральная зона каждого изображения, где ошибка проецирования будет значительно меньше, чем ошибка на краях.

Алгоритм, принимающий допущение плавности и аппроксимирующий область каждого снимка плоскостью будем называть алгоритмом простой сшивки. Основной задачей данного алгоритма является построение аппроксимирующей плоскости для каждого изображения. В большинстве случаев такую плоскость можно построить, воспользовавшись таблицей T и набором точек пространства $X_i, i=1, 2, \dots, N$. Процедура построения плоскости по этим данным описана в разделе 4.2.1. Для некоторых наборов исходных данных построенная на основе точек X_i плоскость может оказаться некорректной, способ детектирования таких данных представлен в разделе 4.2.2. В случае некорректной плоскости может потребоваться её коррекция (раздел 4.2.3). Другой задачей в алгоритме простой сшивки является определение проецирующей функции. Решение этой задачи представлено в разделе 4.2.4.

4.2.1. Построение аппроксимирующей плоскости. Используя таблицу видимости точек X_i на изображениях $j – T$, составим множества индексов $K_j = \{k | T_{kj} \neq -1 \cap k=1, 2, \dots, N\}, i=1, 2, \dots, N, j=1, 2, \dots, m$. Тогда можно говорить, что на j -ом изображении видны только точки X_k , где $k \in K_j$.

Уравнение плоскости, аппроксимирующей поверхность дна, видимую на j -ом изображении, имеет вид $A_j x + B_j y + C_j z + D_j = 0$. Поскольку все изображения подводного дна, получаются на некоторой, ненулевой глубине, коэффициент уравнения D_j (расстояние до плоскости от начала координат) можно принять за единицу. В этом случае уравнение примет вид (39).

$$A_j x + B_j y + C_j z = -1. \quad (39)$$

Чтобы аппроксимирующая плоскость для j -го изображения наилучшим образом удовлетворяла точкам X_k , необходимо, чтобы квадрат разницы левой и

правой части уравнения (39) при подстановке туда точек $\mathbf{X}_k$ был минимальным. Таким образом, имеем систему линейных алгебраических уравнений (40) для $k \in K_j$.

$$A_j x_k + B_j y_k + C_j z_k = -1. \quad (40)$$

Если $|K_j| < 3$, то система имеет множество решений, в этом случае процесс определения аппроксимирующей плоскости описан в разделе 4.2.3. В общем случае, когда $|K_j| \geq 3$ система (40) имеет одно точное ($|K_j| = 3$) или приближённое решение ($|K_j| > 3$). Введём матрицу $\mathbf{A}$, состоящую из $|K_j|$ строк, каждая строка которой $\mathbf{A}_k = (x_k, y_k, z_k)$. Также введём вектор правых частей $\mathbf{B} = (-1, -1, \dots, -1)^T$ размерности $|K_j|$. Тогда, искомые коэффициенты плоскости A_j, B_j, C_j можно найти, домножив обе части системы (40) на матрицу $\mathbf{A}^T$ (41).

$$\mathbf{A}^T \mathbf{A} (A_j, B_j, C_j)^T = \mathbf{A}^T \mathbf{B}. \quad (41)$$

Система (41) – система с тремя линейными алгебраическими уравнениями и тремя неизвестными, может быть однозначно решена. Решение данной системы, в совокупности с коэффициентом $D_j = 1$ даёт четыре коэффициента уравнения искомой аппроксимирующей плоскости для j -го изображения.

4.2.2. Оценка корректности построенной плоскости. Построенная в разделе 4.2.1 плоскость не всегда подходит для аппроксимации, более того, когда точек, видимых с некоторого j -го, $j=1, 2, \dots, m$ вида меньше трёх, плоскость не может быть построена вовсе. Когда все точки $\mathbf{X}_k$, $k \in K_j$ сосредоточены на малой части площади, покрываемой снимком поверхности, плоскость, полученная в разделе 4.2.1, в большинстве случаев будет неверна. Таким образом, для оценки достоверности рассчитанной аппроксимирующей плоскости необходимо найти площадь покрытия j -го изображения S_j и площадь, на которой сосредоточены все видимые точки $\hat{S}_j$. Тогда коэффициент достоверности рассчитанной плоскости η_j можно определить, как отношение

$\eta_j = \hat{S}_j / S_j$. Значение $\eta_j \geq 0.5$ говорит о том, что плоскость рассчитана с достаточной точностью и не требует корректировки. Значение $\eta_j < 0.5$ указывает на необходимость корректировки (раздел 4.2.3). Далее рассмотрим, как найти площади S_j и $\hat{S}_j$.

Чтобы найти площадь покрытия j -го изображения S_j , определим четыре направления для лучей, идущих от центра камеры до четырёх углов этого изображения:

$$\begin{aligned} \lambda_1 &= \mathbf{R}(\mathbf{r}_j)(1, h_j/w_j, \bar{f}_j)^T, & \lambda_2 &= \mathbf{R}(\mathbf{r}_j)(-1, h_j/w_j, \bar{f}_j)^T, \\ \lambda_3 &= \mathbf{R}(\mathbf{r}_j)(-1, -h_j/w_j, \bar{f}_j)^T, & \lambda_4 &= \mathbf{R}(\mathbf{r}_j)(1, -h_j/w_j, \bar{f}_j)^T. \end{aligned}$$

Для удобства, в обозначении направлений λ_l , $l=1, 2, 3, 4$ опущен индекс j , хотя они и относятся к j -ому изображению. Запишем параметрическое уравнение для лучей $\hat{\lambda}_l$, исходящих из центра j -ой камеры в направлении углов изображения:

$$\hat{\lambda}_l(t) = \mathbf{c}_j + t\lambda_l. \quad (42)$$

Подставив три координаты луча $\hat{\lambda}_l$ в уравнение аппроксимирующей плоскости j -го изображения (39) и разрешив относительно t , найдем такое t_l , которое при подстановке в уравнение (42) даёт координаты точки плоскости, соответствующей углу изображения: $\hat{\lambda}_l(t_l)$. Имея координаты угловых точек найдём площадь S_j , как сумму площадей двух треугольников, по формуле (43).

$$\begin{aligned} S_j &= \frac{1}{2} (|(\hat{\lambda}_2(t_2) - \hat{\lambda}_1(t_1)) \times (\hat{\lambda}_3(t_3) - \hat{\lambda}_1(t_1))| \\ &+ |(\hat{\lambda}_3(t_3) - \hat{\lambda}_1(t_1)) \times (\hat{\lambda}_4(t_4) - \hat{\lambda}_1(t_1))|). \end{aligned} \quad (43)$$

Для нахождения площади, на которой сосредоточены все видимые на j -ом изображении точки, $\hat{S}_j$, необходимо сначала спроецировать все эти точки на аппроксимирующую плоскость. Для этого определим ортонормированный базис $\mathbf{x}_{b_j}$, $\mathbf{y}_{b_j}$, $\mathbf{z}_{b_j}$ таким образом, что вектор $\mathbf{z}_{b_j} = ((A_j, B_j, C_j) / \sqrt{A_j^2 + B_j^2 + C_j^2})$, а направления векторов $\mathbf{x}_{b_j}$, $\mathbf{y}_{b_j}$ произвольны в рамках соблюдения их

перпендикулярности друг другу и вектору $\mathbf{zb}_j$. Векторы $\mathbf{xb}_j$ и $\mathbf{yb}_j$ задают базис плоскости, на которую нужно спроецировать точки $\mathbf{X}_k$: $\mathbf{p}_k=(u_k, v_k)=(\mathbf{X}_k \cdot \mathbf{xb}_j, \mathbf{X}_k \cdot \mathbf{yb}_j)$. Точки $\mathbf{p}_k$ будем использовать для вычисления площади $\hat{S}_j$.

В качестве фигуры, заключающей все точки $\mathbf{p}_k$, выберем выпуклую оболочку. Выпуклую оболочку для набора точек можно построить, воспользовавшись алгоритмом quickhull [25]. Данный алгоритм строит выпуклую оболочку, добавляя в неё новый треугольник на каждом шаге рекурсии, при этом все точки внутри треугольника исключаются, как не попадающие на границу оболочки. За счёт исключения точек, алгоритм имеет вычислительную сложность $O(|K_j|\log(|K_j|))$. Так как сама оболочка не представляет для рассматриваемой задачи интереса, нет необходимости запоминать каждую вершину и каждый треугольник формирующие оболочку. Достаточно на каждом шаге рекурсии аккумулировать значение площади построенного треугольника в переменной $\hat{S}_j$, тогда после завершения алгоритма данная переменная будет содержать искомую площадь.

4.2.3. Коррекция плоскости. В случае, когда коэффициент $\eta < 0.5$ (раздел 4.2.2) необходима коррекция решения, а в случае, когда $|K_j| < 3$ нужен способ решения, отличный от 4.2.1. Необходимость в коррекции решения или поиске нового возникает в случае, когда исходные данные – точки $\mathbf{X}_k$ плохо распределены (либо их недостаточно) и нужно использовать дополнительные предположения для определения решения. Таким дополнительным предположением является перпендикулярность участка рельефа, охватываемого изображением и оси Z камеры, с которой это изображение получено. Такое предположение имеет право на существование, так как АНПА передвигается преимущественно параллельно дну с незначительными углами крена и дифферента, поэтому в большинстве случаев ось Z камеры и плоскости дна близки к тому, чтобы быть перпендикулярными. Особенно это справедливо в случае с плавным перепадом высот рельефа, который рассматривается в задаче простой сшивки.

Вектор, определяющий направление Z_j камеры в момент получения j -го изображения – третий столбец матрицы $\mathbf{R}(\mathbf{r}_j)$. Данный вектор совпадает с нормалью аппроксимирующей плоскости в альтернативном решении:

$$\hat{A}_j = \mathbf{R}(\mathbf{r}_j)_{13}, \quad \hat{B}_j = \mathbf{R}(\mathbf{r}_j)_{23}, \quad \hat{C}_j = \mathbf{R}(\mathbf{r}_j)_{33}.$$

Коэффициенты альтернативной аппроксимирующей плоскости j -го изображения будем обозначать $\hat{A}_j, \hat{B}_j, \hat{C}_j, \hat{D}_j$. Зная первые три коэффициента, найдем четвёртый:

$$\hat{D}_j = |\mathbf{K}_j|^{-1} \sum_{k \in K_j} (-\hat{A}_j x_k - \hat{B}_j y_k - \hat{C}_j z_k).$$

Плоскость $(\hat{A}_j, \hat{B}_j, \hat{C}_j, \hat{D}_j)$ следует использовать в качестве решения тогда, когда $|K_j| < 3$. Для других случаев решение получается комбинированием уравнений (40) и дополнительных уравнений, требующих параллельности вектора нормали аппроксимирующей плоскости (A_j, B_j, C_j) и оси Z_j камеры $(\hat{A}_j, \hat{B}_j, \hat{C}_j)$. Дополнительные уравнения описываются, как приравненное нулю векторное произведение этих векторов:

$$\begin{pmatrix} A_j \\ B_j \\ C_j \end{pmatrix} \times \begin{pmatrix} \hat{A}_j \\ \hat{B}_j \\ \hat{C}_j \end{pmatrix} = \begin{pmatrix} B_j \hat{C}_j - C_j \hat{B}_j \\ C_j \hat{A}_j - A_j \hat{C}_j \\ A_j \hat{B}_j - B_j \hat{A}_j \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Матрица системы дополнительных уравнений определяется следующим образом:

$$\mathbf{M} = \begin{pmatrix} 0 & \hat{C}_j & -\hat{B}_j \\ -\hat{C}_j & 0 & \hat{A}_j \\ \hat{B}_j & -\hat{A}_j & 0 \end{pmatrix}.$$

Общая система, учитывающая дополнительные условия запишется следующим образом:

$$(\mathbf{A}^T \mathbf{A} + \alpha_j \cdot \mathbf{M}^T \mathbf{M})(A_j, B_j, C_j)^T = \mathbf{A}^T \mathbf{B}. \quad (44)$$

Здесь α_j - коэффициент, определяющий вклад дополнительных уравнений в общее решение. Этот коэффициент должен учитывать отношение площадей η_j .

В данной работе коэффициент $\alpha_j = \varepsilon_j s_j \rho_j$, где $s_j = (-5\eta_j + 2.5)^3$ – коэффициент разницы площадей, $\rho_j = \left((A_j^2 + B_j^2 + C_j^2) \sin^2 \beta_j\right)^{-1}$ – нормирующий фактор для дополнительных уравнений, β_j – наименьший угол между прямыми, задаваемыми векторами (A_j, B_j, C_j) и $(\hat{A}_j, \hat{B}_j, \hat{C}_j)$, а ε_j определяется так:

$$\varepsilon_j = \sum_{k \in K_j} (x_k A_j + y_k B_j + z_k C_j + D_j)^2.$$

Такой коэффициент хорошо зарекомендовал себя на практике.

4.2.4. Функции проекции. Когда аппроксимирующая плоскость определена и при необходимости скорректирована, для обеспечения процесса сшивки определим функции проекции. Процесс сшивки требует двух функций: функции прямой проекции и функции обратной проекции. Функция прямой проекции – $\mathbf{f}_j(u, w)$ двум координатам u и w на плоскости j -го изображения ставит в соответствие три координаты (x, y, z) в пространстве. Функция обратной проекции – $\mathbf{f}_j^{-1} = (x, y)$ отображает две координаты на плоскости $Z=0$ в две координаты u, w на плоскости j -го изображения. Функция прямой проекции $\mathbf{f}_j(u, w)$ нужна для определения границ проекции j -го изображения на плоскость Z . Функция обратной проекции используется при определении цвета каждого фрагмента плоскости $Z=0$, которая поставлена в соответствие с фотографической картой дна.

Определим функцию прямой проекции. В разделе 4.2.2 при расчёте площади S_j был описан процесс прямой проекции, в данном разделе запишем данный процесс в формальном виде. Определим две вспомогательные функции, служащие для упрощения записи функции $\mathbf{f}_j(u, w)$:

$$\mathbf{v}_j(u, w) = \begin{pmatrix} u \\ w \\ \bar{f}_j \end{pmatrix},$$

$$t_j(u, w) = \frac{-A_j c_{jx} - B_j c_{jy} - C_j c_{jz} - D_j}{A_j \mathbf{R}(\mathbf{r}_j)_1 \mathbf{v}_j(u, w) + B_j \mathbf{R}(\mathbf{r}_j)_2 \mathbf{v}_j(u, w) + C_j \mathbf{R}(\mathbf{r}_j)_3 \mathbf{v}_j(u, w)}.$$

За $\mathbf{R}(\mathbf{r}_j)_p$, $p=1, 2, 3$, обозначены строки матриц поворота $\mathbf{R}(\mathbf{r}_j)$. Используя вспомогательные функции, определим функцию прямой проекции (45):

$$\mathbf{f}_j(u, w) = \begin{pmatrix} c_{jx} + t_j(u, w)\mathbf{R}(\mathbf{r}_j)_1 v_j(u, w) \\ c_{jy} + t_j(u, w)\mathbf{R}(\mathbf{r}_j)_2 v_j(u, w) \\ c_{jz} + t_j(u, w)\mathbf{R}(\mathbf{r}_j)_3 v_j(u, w) \end{pmatrix}. \quad (45)$$

Функция обратной проекции определяется следующим образом:

$$\mathbf{f}_j^{-1}(x, y) = \begin{pmatrix} \frac{\bar{f}_j \left(\mathbf{R}(\mathbf{r}_j)_{11}x + \mathbf{R}(\mathbf{r}_j)_{21}y + \mathbf{R}(\mathbf{r}_j)_{31}z_j(x, y) + \bar{c}_{jx} \right)}{\mathbf{R}(\mathbf{r}_j)_{13}x + \mathbf{R}(\mathbf{r}_j)_{23}y + \mathbf{R}(\mathbf{r}_j)_{33}z_j(x, y) + \bar{c}_{jz}} \\ \frac{\bar{f}_j \left(\mathbf{R}(\mathbf{r}_j)_{12}x + \mathbf{R}(\mathbf{r}_j)_{22}y + \mathbf{R}(\mathbf{r}_j)_{32}z_j(x, y) + \bar{c}_{jy} \right)}{\mathbf{R}(\mathbf{r}_j)_{13}x + \mathbf{R}(\mathbf{r}_j)_{23}y + \mathbf{R}(\mathbf{r}_j)_{33}z_j(x, y) + \bar{c}_{jz}} \end{pmatrix}. \quad (46)$$

Здесь, для определения координаты z_j использована функция $z_j(x, y)$ (47), использующая аппроксимирующую плоскость.

$$z_j(x, y) = C_j^{-1}(-A_j x - B_j y - D_j). \quad (47)$$

Процесс использования прямой и обратной функции проецирования подробно описывается в разделе 4.4.

4.3. Сшивка на основе трёхмерной модели дна

В случае, когда рельеф дна не удовлетворяет условиям плавного перепада высот, как это изображено на рис. 35б, алгоритм простой сшивки не подходит. Для сшивки изображений дна со сложным рельефом необходимо использовать иной подход. Это же относится и к изображениям, полученным в результате исследования затонувших подводных объектов.

Как правило, в зонах со сложным рельефом АНПА перемещается медленнее, обеспечивая тем самым более плотное покрытие дна снимками. Также для сложного рельефа характерно большее число визуальных особенностей, чем для ровного дна. Оба этих фактора позволяют на основе полученных с АНПА изображений построить не просто фотографическую карту дна, а трёхмерную модель этого дна. А уже из модели можно получить

фотографическую карту, спроецировав каждую точку этой модели на плоскость $Z=0$.

Основным вопросом данного раздела является вопрос построения сеточной модели дна на основе известных точек X_i . Данному вопросу посвящен раздел 4.3.1. Другим вопросом остаётся процедура проецирования точек модели на фотографическую карту дна – раздел 4.3.2.

4.3.1. Построение трёхмерной модели рельефа дна. Полигональная трёхмерная модель состоит из точек и полигонов, определённых в этих точках. Для рассматриваемой задачи набор точек X_i известен и, для получения полноценной модели, необходимо построить набор полигонов с углами в этих точках, описывающих поверхность рельефа дна. Одним из универсальных полигонов является треугольник, именно треугольники чаще всего используются для описания трёхмерных моделей. Процесс построения треугольников на основе известного облака точек называется триангуляцией. Существует множество вариантов триангуляций облака точек при количестве точек большим, чем три. Для оптимального представления поверхности рельефа дна идеально подходят триангуляции, удовлетворяющие условию Делоне [3].

Существует большое число алгоритмов построения триангуляции Делоне, многие из которых описаны в работе [12]. Предпочтение в данной работе отдаётся алгоритмам триангуляции со статическим и динамическим кэшированием поиска. В частности алгоритм с динамическим кэшированием поиска [12] хорошо показал себя для наборов точек подводного дна. Он имеет практически линейную вычислительную сложность и способен справляться с десятками миллионов точек.

Для применения алгоритма триангуляции нужно из набора трёхмерных точек сформировать набор двумерных точек. В задаче построения модели рельефа подводного дна это можно сделать, просто отбросив третью координату z_i каждой точки. Построение триангуляции с помощью алгоритма с

динамическим кэшированием поиска для рассматриваемой задачи начинается с построения суперструктуры – выпуклой фигуры, заключающей внутри все точки. В качестве суперструктуры использовался квадрат, с длиной стороны вдвое превышающей расстояние между самыми удалёнными точками в плоскости $Z=0$ и с центром в геометрическом центре точек. Данный квадрат формировался из двух прямоугольных треугольников с общей гипотенузой. Вершинам этих треугольников присваивались номера, начиная с $N+1$.

После применения алгоритма с динамическим кэшированием поиска индексы вершин, формирующие каждый треугольник, сохранялись в некоторый массив $\mathbf{T}=(\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_n)$, где n – количество построенных треугольников. Каждый треугольник $\mathbf{t}_k=(t_{k1}, t_{k2}, t_{k3})$, где $k=1, 2, \dots, n$ – набор из трёх номеров точек, формирующих треугольник: $\mathbf{X}_{t_{k1}}, \mathbf{X}_{t_{k2}}, \mathbf{X}_{t_{k3}}$. Все треугольники, имеющие хотя бы одну вершину с номером большим, чем N убирались из набора $\mathbf{T}$, чтобы исключить попадание точек суперструктуры в итоговую триангуляцию. Также, чтобы исключить из итоговой модели области с недостаточной детализацией, из набора $\mathbf{T}$ убирались треугольники, у которых одна из сторон оказывалась длиннее одного метра. Данная процедура также позволила избавиться от длинных узких треугольников по краям триангуляции. В результате данных манипуляций получался набор треугольников $\mathbf{T}$, описывающий модель поверхности дна в совокупности с точками $\mathbf{X}_i$.

4.3.2. Процедура проецирования. Функция прямого и обратного проецирования точек модели на плоскость $Z=0$ имеют более сложный вид, чем аналогичные функции в разделе 4.2.4. Однако, особенности данной задачи позволяют без них обойтись.

Функция прямого проецирования $\mathbf{f}_j(u, w)$ нужна исключительно для того, чтобы оценить границы зоны на плоскости $Z=0$, куда проецируется j -ое изображения. В случае, когда на плоскость $Z=0$ проецируется не всё изображения, а только треугольники, содержащие точки $\mathbf{X}_k, k \in K_j$, видимые на j -ом изображении, границы можно определить, как координаты крайней левой,

крайней правой, крайней верхней и крайней нижней точек (направления указаны для вида сверху на плоскость $Z=0$), входящих в треугольники, содержащие точки X_k .

Вместо определения функции обратного проецирования и вычисления цвета точки треугольника, соответствующей пикселю фотографической карты, воспользуемся стандартным алгоритмом рисования треугольников. Поскольку задача рисования треугольников – основная задача трёхмерной графики, самые быстрые алгоритмы рисования треугольников реализованы аппаратно в современных графических ускорителях. Определив зону проецирования для каждого снимка достаточно лишь нарисовать набор треугольников на изображении, аналогичном по размеру, с камерой, смотрящей вдоль мировой оси Z . Это можно сделать, воспользовавшись какой-либо графической библиотекой, например OpenGL. Использование вершинных и фрагментных шейдеров позволяет записывать в изображения не только цвет, но и дополнительные данные, необходимые для корректного смешения цветов, даваемых пересекающимися изображениями.

Для рисования треугольников необходимо знать лишь координаты вершин X_i , набор треугольников T и текстурные координаты, соответствующие каждой вершине. Неизвестные текстурные координаты легко могут быть найдены путём проецирования точек на изображения с помощью функции, описанной в (46), при этом вместо функции $z_j(x, y)$ следует использовать третью координату проецируемой точки.

Использование способа проецирования треугольников на фотографическую карту дна с помощью стандартной графической библиотеки существенно ускоряет и упрощает процесс проецирования. Поскольку каждое изображения может проецироваться отдельно, требования к памяти видеоускорителя остаются незначительными.

4.4. Устранение швов на стыках изображений

Заключительной подзадачей процедуры сшивки является такое смещение входных изображений, спроецированных на плоскость $Z=0$, которое приводит к бесшовной фотографической карте подводного дна. Осложняющим фактором данной задачи является большое количество входных изображений. Напротив, облегчает задачу малое количество движущихся объектов, которые редко попадают в небольшое пространство между АНПА и дном. В отсутствие движущихся объектов эффективной становится методика смещения, описанная в [35, 105], которая и была использована в данной работе.

Предположим, что в точку $P=(x_p, y_p)$ фотографической карты дна проецируются пиксели с изображений $j1, j2, \dots, jn$, где n – количество изображений, на которых видна точка P , а все индексы изображений попадают в интервал $[1; m]$. Соответствующие точке P координаты на изображениях найдем с помощью (46): $(s_{j1}, t_{j1})=f_{j1}^{-1}(x_p, y_p)$, $(s_{j2}, t_{j2})=f_{j2}^{-1}(x_p, y_p), \dots, (s_{jn}, t_{jn})=f_{jn}^{-1}(x_p, y_p)$. Обозначим через $C_j(s, t)$ вектор (обычно вектор, составленный из трёх цветовых компонент: красной, синей и зелёной), описывающий цвет точки j -го изображения с координатами u, w . Тогда (48) определяет цвет P -ой точки – C_p .

$$C_p = \frac{\sum_{l=1}^n k_{jl} C_{jl}(s_{jl}, t_{jl})}{\sum_{l=1}^n k_{jl}}. \quad (48)$$

Здесь k_{jl} , $l=1, 2, \dots, n$ – коэффициент вклада каждого изображения в цвет точки P фотографической карты дна. Этот коэффициент определяется следующим образом:

$$k_{jl} = \left(\min \left(|1 - s_{jl}|, |s_{jl} + 1|, \left| \frac{h_{jl}}{w_{jl}} - t_{jl} \right|, \left| t_{jl} + \left| \frac{h_{jl}}{w_{jl}} \right| \right| \right) \right)^\gamma. \quad (49)$$

Здесь h_{jl} , w_{jl} – высота и ширина изображения jl в пикселях, а γ – коэффициент спада. Идея коэффициента k_{jl} в том, что он имеет максимальное значение в случае, когда точка (s_{jl}, t_{jl}) находится в центре и минимальное, когда точка

находится на границе изображения. Коэффициент γ показывает, насколько резко падает k_{jl} при движении точки (s_{jl}, t_{jl}) к краю изображения. Чем выше данный коэффициент, тем уже получается граница смещения перекрывающихся изображений и тем чётче область перекрытия. Однако при слишком высоких значениях, шов между изображениями начинает выделяться. В данной работе было использовано значение $\gamma = 4$, как наиболее оптимальное.

Использование смещения (48) позволяет формировать фотографическую карту дна последовательно, добавляя в неё новые и новые изображения. Этот процесс можно описать следующим алгоритмом:

Шаг 1. Формируем две карты одинакового размера: фотографическую карту и карту коэффициентов. Изначально обе карты заполняются нулями.

Шаг 2. Пока есть входные изображения, выполняем для каждого из них шаг 3, иначе переходим к шагу 5.

Шаг 3. Определяем границы области, в которую проецируется входное изображение на фотографической карте с помощью функции прямой проекции (45) либо по координатам точек пространства (раздел 4.3.2).

Шаг 4а. Для всех точек из определённой на шаге 3 области находим цвет, даваемый рассматриваемым изображением и соответствующий ему коэффициент (49). Цвет добавляем к фотографической карте, а коэффициент к карте коэффициентов. Переходим к шагу 2.

Шаг 4б. С помощью OpenGL отрисовываем все треугольники, видимые на рассматриваемом изображении в цветовой буфер, а соответствующие коэффициенты помещаем в альфа канал. Добавляем цветовой буфер к фотографической карте, а альфа канал к карте коэффициентов. Переходим к шагу 2.

Шаг 5. Делим цвет каждой точки фотографической карты на значение соответствующего коэффициента в карте коэффициентов.

Шаги 4а и 4б приведённого алгоритма являются альтернативными и соответствуют простой сшивке (раздел 4.2) либо сшивке на основе модели (раздел 4.3).

Реальный размер фотографической карты в пикселях зависит от того, какое разрешения в пикселях на метр мы хотим получить в итоге, и может варьироваться. Из этого вытекают две основные проблемы: согласование размеров входных изображений с размерами фотографической карты и работа с картой в оперативной памяти. Рассмотрим обе эти проблемы.

При получении цвета пикселя изображения $C_{jl}(s_{jl}, t_{jl})$ используются координаты в нормализованной системе - s_{jl}, t_{jl} (раздел 2.2). При этом реальное изображение имеет разрешение в пикселях w_{jl}, h_{jl} . Для получения корректного значения цвета – недостаточно взять ближайший пиксель к координатам:

$$(\bar{s}_{jl}, \bar{t}_{jl}) = \left(\frac{1}{2}(s_{jl} + 1)w_{jl}, \frac{w_{jl}}{2h_{jl}} \left(s_{jl} + \frac{h_{jl}}{w_{jl}} \right) h_{jl} \right).$$

Сначала необходимо согласовать масштаб изображения с масштабом фотографической карты, увеличив либо уменьшив его. Если выбранное разрешение фотографической карты составляет большее число пикселей на метр, чем масштаб изображения, то изображение следует увеличить, иначе уменьшить. Когда масштабы изображения и фотографической карты согласованы, можно выбирать цвет $C_{jl}(s_{jl}, t_{jl})$, как результат билинейной интерполяции по двум осям изображения, на основе значений четырёх ближайших к точке $(\bar{s}_{jl}, \bar{t}_{jl})$ пикселей изображения. Билинейная интерполяция будет происходить автоматически на графическом ускорителе, как результат фильтрации текстуры в случае сшивки на основе модели.

Другая проблема – работа с фотографической картой в оперативной памяти становится существенной, если для карты выбрано большое разрешение. Размер фотографической карты становится настолько большим, что объема оперативной памяти на современном компьютере оказывается недостаточно. Чтобы обойти эту проблему, фотографическую карту

необходимо разбить на блоки с размером, достаточным для размещения в оперативной памяти. При этом все операции, связанные со сшивкой происходят в рамках своего блока и могут выполняться последовательно, не требуя одновременного размещения в памяти более одного блока. Недостатком разбиения является некоторая избыточность вычислений, связанная с тем, что одно и то же изображение может попасть в несколько блоков одновременно. В этом случае, это изображение должно быть несколько раз загружено в оперативную память. Чем меньше размер блока, тем больше будет изображений, попадающих сразу в несколько блоков, поэтому целесообразно выбирать размер блока максимальным, насколько позволяет оперативная память.

4.5. Результаты

В четвёртой главе была рассмотрена задача сшивки изображений. Результатом выполнения этой задачи является бесшовное полотно – фотографическая карта дна. Оценить качество сшивки, а именно чёткость изображения и невидимость швов можно визуально, поэтому данный раздел содержит изображения фрагментов карт, полученных в результате применения описанных в главе 4 алгоритмов.

4.5.1. Исходные данные для тестирования. Для тестирования алгоритмов главы 4 будем использовать два набора фотографий. Один набор – реальные данные с АНПА, образующий карту, приведённую на рисунке 30, другой набор – синтетические данные. Реальные данные получены с участка дна, имеющего плавный перепад высот, а синтетические сгенерированы для рельефа с резким перепадом высот.

Рассмотрим подробнее процесс генерации синтетических тестовых данных. Для получения высоты трёхмерной модели рельефа для каждой точки x, y была использована следующая функция:

$$z(x, y) = \sum_{i=0}^n a_i p(x f_i, y f_i). \quad (50)$$

Здесь $a_i = a_{i-1}/2$, $a_0 = \text{const}$ – амплитуды, при этом начальная амплитуда a_0 выбирается как параметр, определяющий перепад высот рельефа. Величины $f_i = f_{i-1} \cdot 2$, $f_0 = \text{const}$ определяют частоты, при этом базовая частота f_0 выбирается постоянной для алгоритма генерации и определяет частоту наиболее существенных перепадов высот на рельефе. Функция $p(x, y)$ – шумовая функция Перлина [15]. Параметр n определяет детализацию получаемого рельефа, чем больше n , тем большее количество мелких деталей будет содержать рельеф.

Чтобы создать условия для адекватного обнаружения и сопоставления особенностей, каждая точка рельефа должна быть раскрашена в некоторый цвет, при этом общая цветовая картина должна быть уникальной и не содержать повторяющихся шаблонов. Частично данная уникальность обуславливается освещением, рельефа, полученного по формуле (50). С другой стороны уникальность гарантирует процедура выбора цвета для каждой точки. При выборе цвета учитывается высота точки $z(x, y)$. К данной высоте добавляется величина, полученная по формуле (50) для значений a_0 и f_0 , отличных от тех, с которыми построен рельеф. Для добавочной величины используется меньшая амплитуда a_0 и более высокая частота f_0 . Полученная в результате сложения высота является индексом в массив из некоторого количества цветов. При моделировании рельефа в данной работе использовалось шесть различных цветов: жёлтый, коричневый, салатовый, зелёный, серый и белый (от самого низкого к самому высокому).

Построенный описанным методом рельеф представлен на рисунке 37.

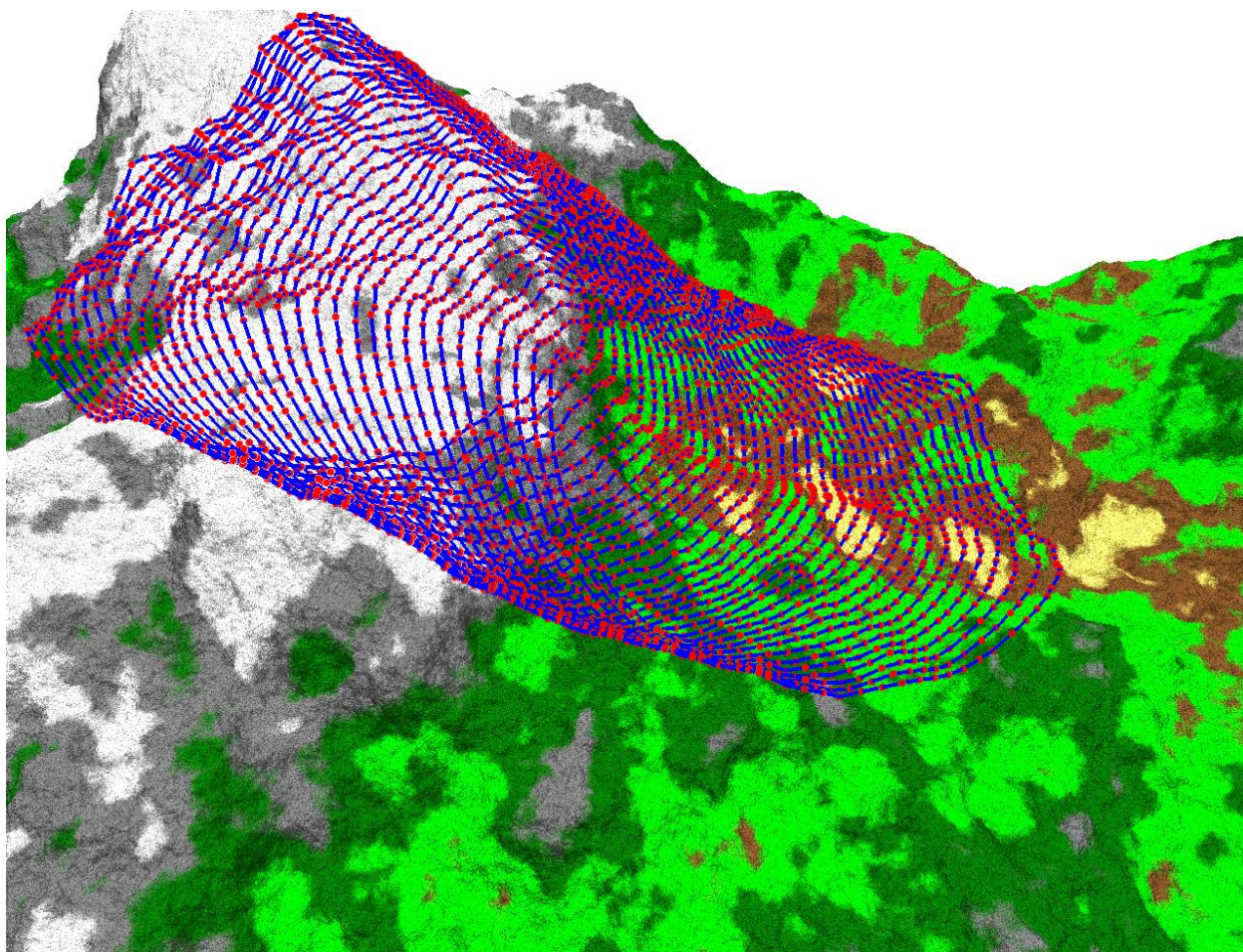


Рис. 37. Синтетические данные для тестирования алгоритмов сшивки.

Также на рисунке показана сгенерированная траектория АНПА, проходящая по области с наибольшим перепадом рельефа. По мере прохода по траектории (обозначена синей линией на рис. 37) АНПА сделал 3318 снимков (красные точки на рис. 37). Модель рельефа была построена с позиции каждого снимка, а каждый снимок был сохранён. К набору из 3318 снимков были применены процедуры поиска особенностей и связей (глава 2) и процедура позиционирования и ориентирования (глава 3). Результат сшивки снимков синтетической модели, с помощью описанных в данной главе методов представлен в следующих разделах.

4.5.2. Результат бесшовной сшивки изображений. В текущей главе было рассмотрено два альтернативных метода сшивки: простая (раздел 4.2) и на основе трёхмерной модели (раздел 4.3). Сравним результаты применения этих методов к изображениям реального дна, представленные на рисунке 38.

Рисунок 38а отражает результат простой сшивки, а рисунок 38б результат сшивки на основе трёхмерной модели. Для обоих методов показаны приближенные фрагменты фотографической карты. На фрагментах не видно швов между изображениями, исключение составляет центральный фрагмент, где у изображений практически отсутствует перекрытие. Оба метода сшивки дают чёткую фотографическую карту и могут быть наложены друг на друга практически пиксель в пиксель. Другими словами, визуальная разница в качестве карт 38а и 38б незаметна. Единственное различие заключается в оформлении краёв зоны. Если алгоритм простой сшивки (рис. 38а) отображает содержимое каждого снимка на фотографическую карту, давая максимально полную картину, то алгоритм сшивки на основе трёхмерной модели (рис. 38б) поступает иначе. При сшивке на основе трёхмерной модели, на фотографическую карту проецируются только треугольники, а не полные зоны изображений. В результате края граничных снимков, не покрытые треугольниками, на фотографической карте не отражаются. Это хорошо заметно по рваной границе фотографической карты на рис. 38б.

Рассмотрим применение алгоритмов сшивки на синтетических данных. В качестве эталона на рис. 39 представим истинную фотографическую карту, полученную при построении модели рельефа на виде сверху в ортографической проекции. Карты, полученные в результате применения алгоритмов раздела 4.3 и раздела 4.2 представлены на рис. 40а и 40б соответственно.

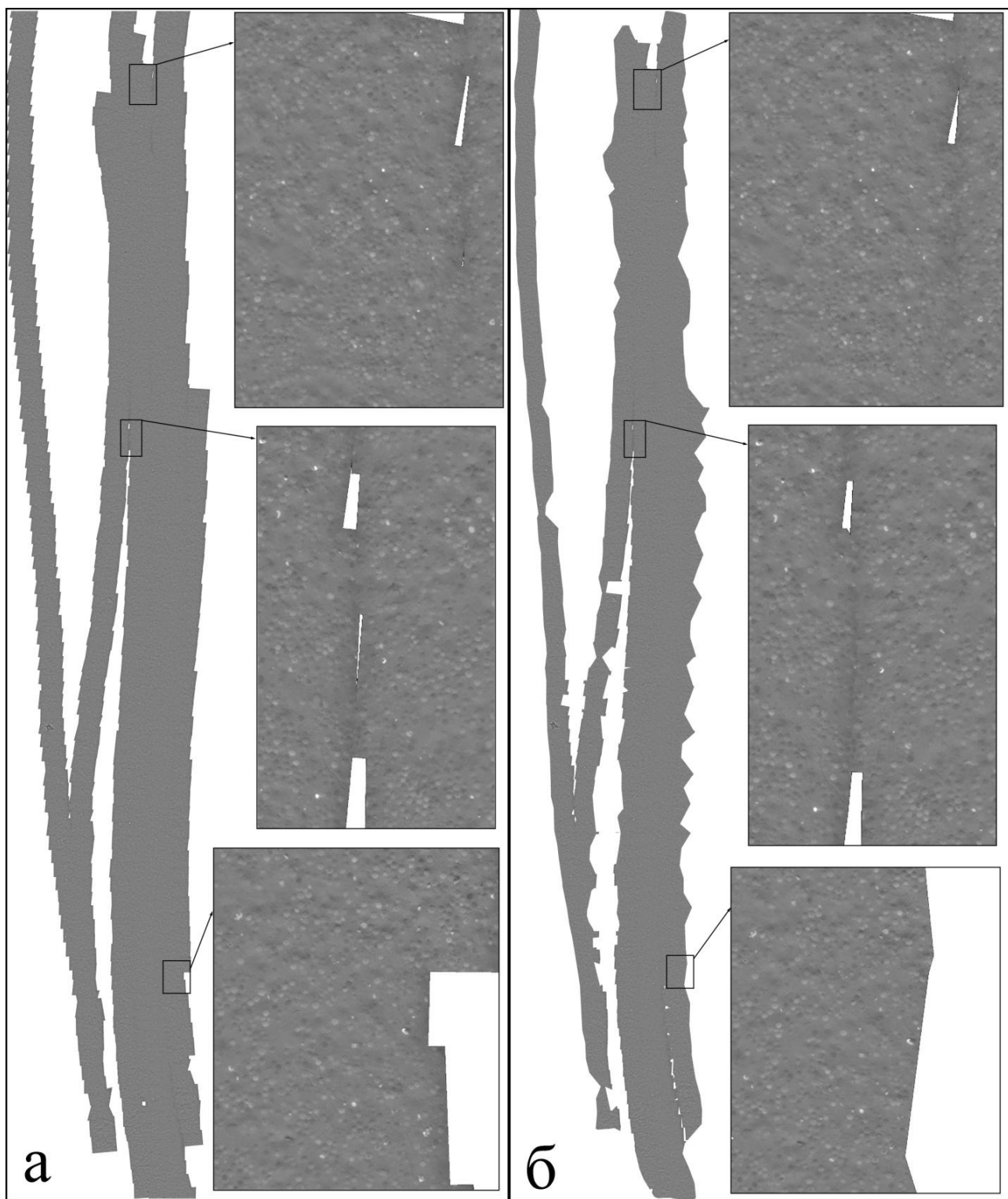


Рис. 38. Применение алгоритмов сшивки к изображениям реального дна.

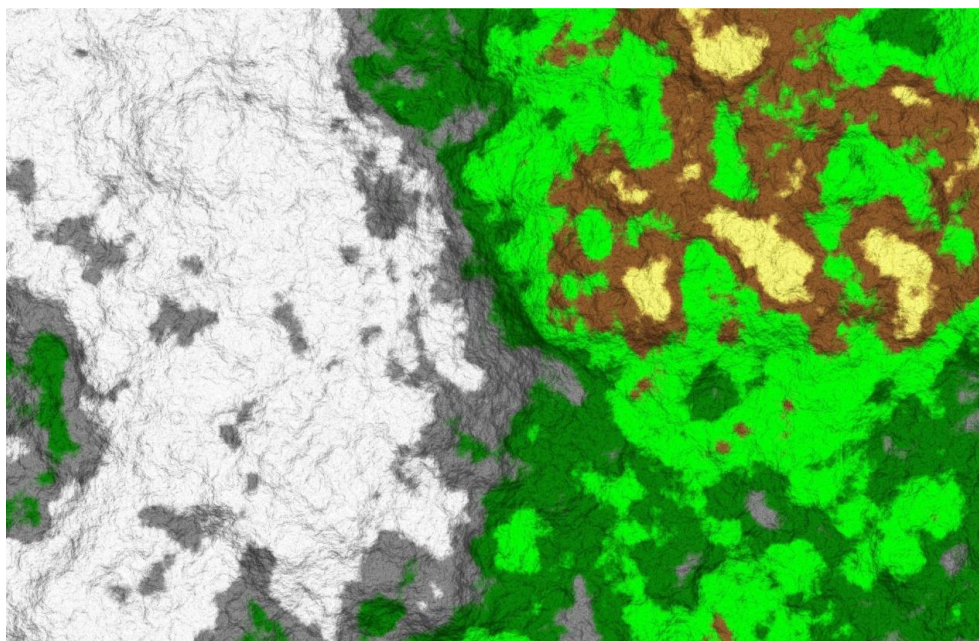


Рис. 39. Эталонная карта синтетической модели дна.

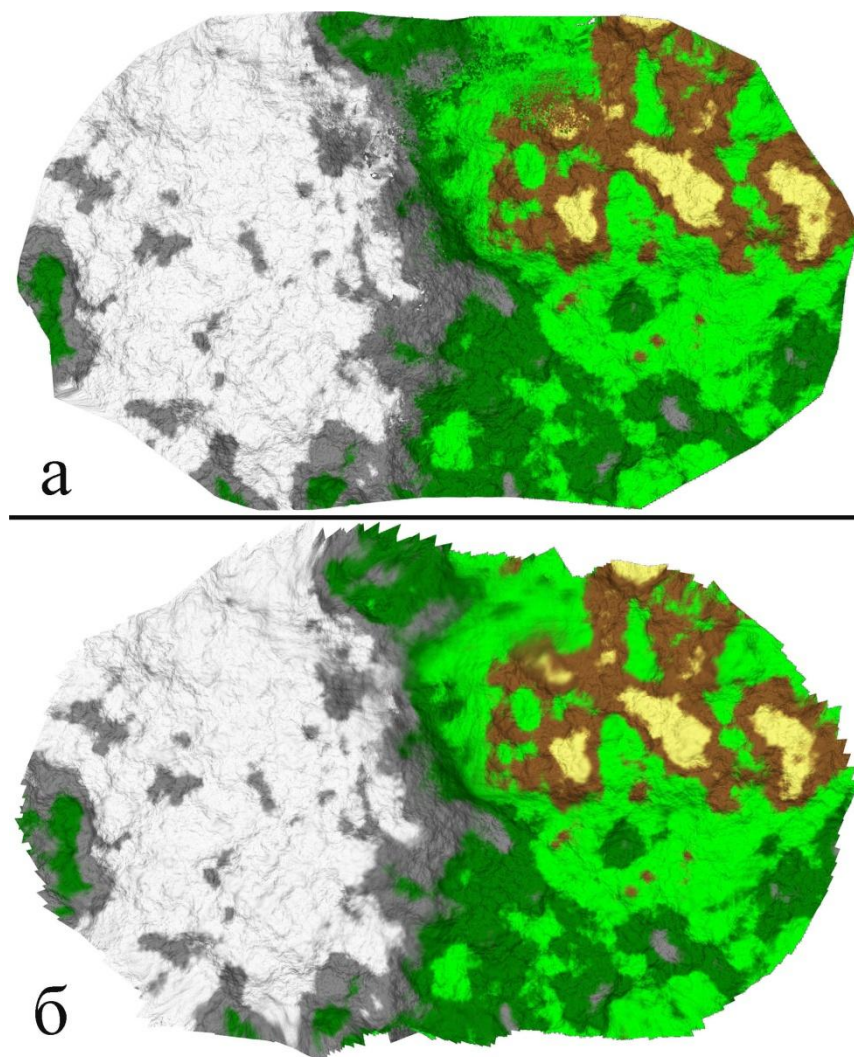


Рис. 40. Применение алгоритмов сшивки на синтетических данных.

На рисунке 40а хорошо заметно, что карта, построенная на основе трёхмерной модели, имеет более высокую чёткость, чем карта, построенная методом простой сшивки (рис. 40б). Нечёткость, даваемая простой сшивкой обуславливается тем, что аппроксимирующие плоскости для каждого изображения не могут с достаточной точностью представить рельеф. В результате, в одну и ту же точку фотографической карты проецируются точки изображений, соответствующие разным точкам рельефа. Это приводит к появлению смазанных зон на карте.

Для демонстрации неприменимости простой сшивки к составлению фотографических карт дна с резкими перепадами высот приведём три увеличенных фрагмента построенных карт – рис. 41.

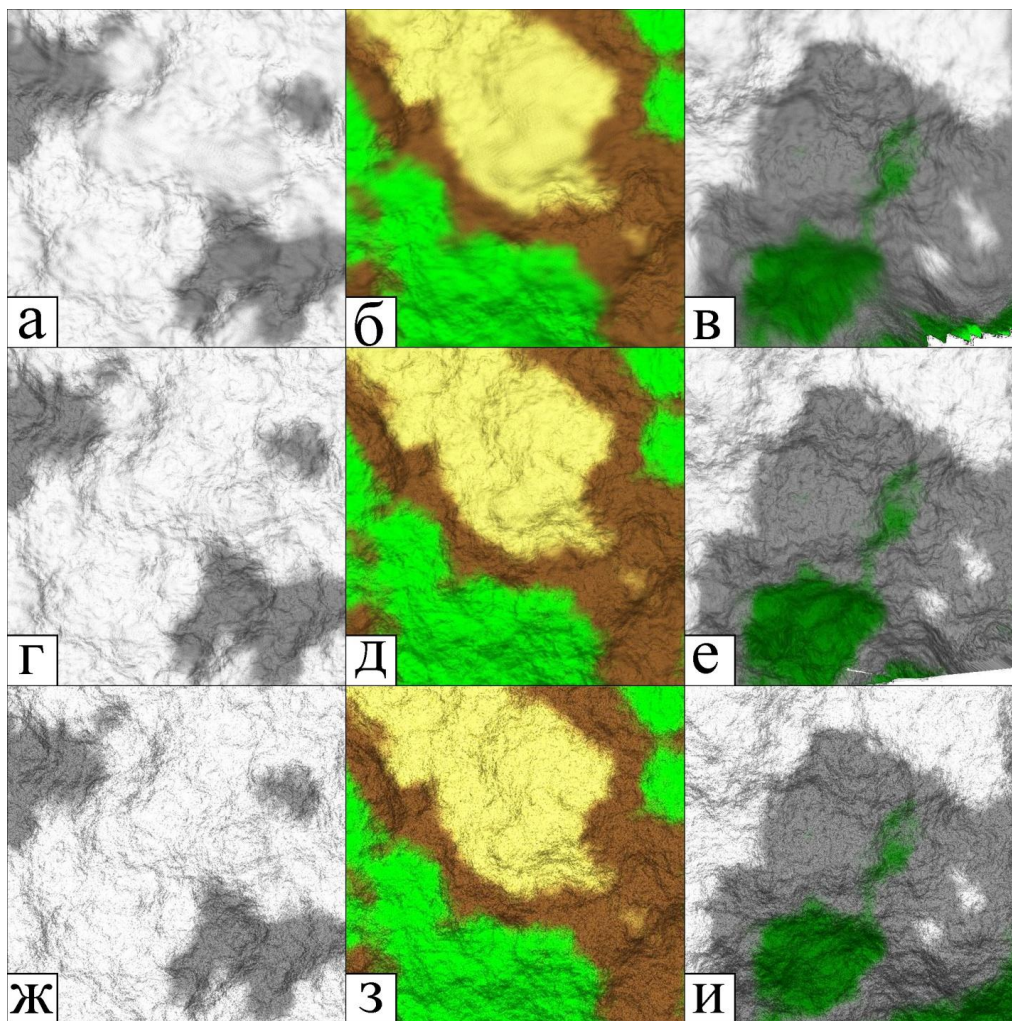


Рис. 41. Фрагменты карты синтетического дна.

На рисунке 41 изображено три столбца фрагментов. Каждый столбец отображает один и тот же фрагмент на карте, полученной методом простой сшивки, на основе трёхмерной модели и эталонной (рис. 39). Фрагменты 41а–в, соответствующие простой сшивке выглядят размазанными и искажёнными. Особенно размазанность выделяется на стыках изображений. Фрагменты 41г–д значительно чётче и идентичны фрагментам эталонной карты 41ж–з.

На основе полученных результатов можно сделать вывод, что простая сшивка имеет преимущества для составления фотографических карт плоских участков дна. Она даёт более полное отображение исходных изображений на фотографическую карту. Для дна с резким перепадом рельефа простую сшивку можно применять лишь в случае, когда требуется карта малого разрешения, тогда как применение сшивки на основе модели является предпочтительным для детальной карты.

4.5.3. Трёхмерная модель дна. Побочным продуктом процедуры сшивки изображений на основе трёхмерной модели является сама трёхмерная модель. Для получения трёхмерных моделей высокой степени детализации необходима процедура плотного сопоставления снимков (каждый пиксель снимка сопоставляется с пикселями связанных снимков), что в случае подводных изображений затруднительно. Этому препятствует большое количество снимков и низкий коэффициент их перекрытия. Однако, грубая модель дна, построенная при получении фотографической карты (раздел 4.3) также может быть полезной в ряде случаев.

На рисунке 42 представлены два вида трёхмерной модели, полученной при построении карты на рисунке 38б. Более интересна модель, полученная в результате построения карты на рисунке 40а. Полученную модель можно сравнить с синтетической моделью, используемой для генерации изображений. На рисунке 43 присутствует как исходная синтетическая модель дна – 43а, так и модель участка, восстановленная в результате построения фотографической

карты – 43б. Очевидно, что модель 43б менее детализирована, чем модель 43а, однако общие очертания рельефа этих моделей полностью совпадают.

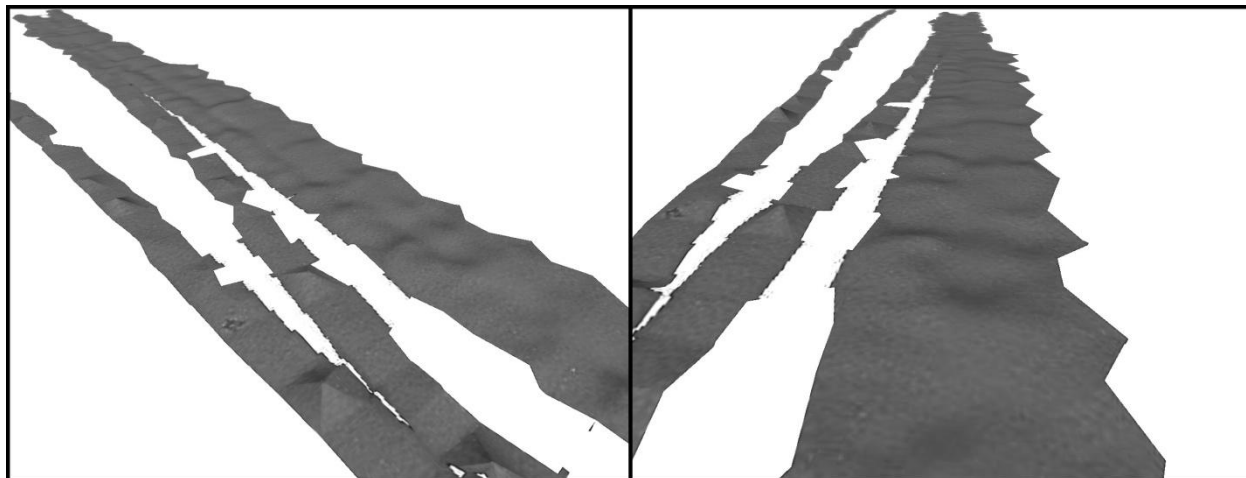


Рис. 42. Трёхмерная модель реального подводного дна.

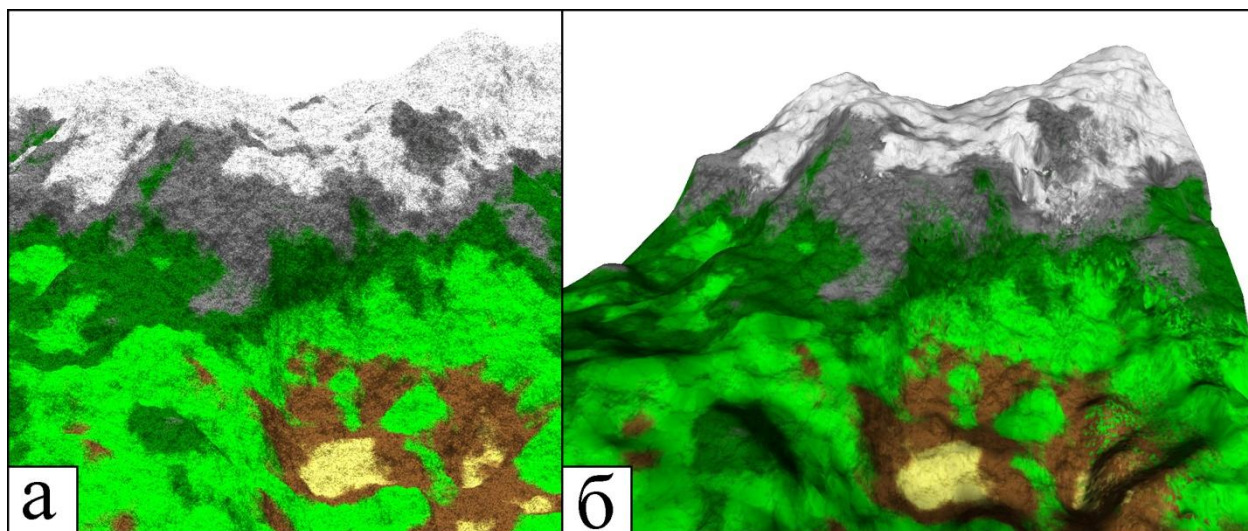


Рис. 43. Трёхмерная модель синтетического подводного дна.

ЗАКЛЮЧЕНИЕ

В диссертационной работе был детально рассмотрен каждый этап процесса построения фотографической карты дна: извлечении и сопоставление точечных особенностей, обнаружение связей между изображениями, приближённое и точное позиционирование и ориентирование изображений, сшивка изображений в единую фотографическую карту. Предложен ряд методов, позволяющих решать проблему построения фотографических карт подводного дна на современных компьютерах для большого объема входных данных. При этом время построения карты сопоставимо со временем получения снимков с помощью АНПА.

В работе рассмотрена важность задачи компенсации нелинейных изменений освещённости снимков при построении дескрипторов особых точек. Предложен собственный метод коррекции освещённости, дающий хорошие результаты.

Предложенный алгоритм установления связей между парой изображений способен быстро обнаруживать связь даже в условиях низкой чёткости снимков и при малом перекрытии. Алгоритм оказался устойчивым к однотипным, повторяющимся объектам, часто встречающимся на дне. Ослабление параметров алгоритма приводят к большему количеству найденных связей, увеличивая при этом долю ложных срабатываний. Тем не менее, для исключения ложных срабатываний к предложенному алгоритму можно добавить стандартную фильтрацию по фундаментальной матрице, проводимую только для пар изображений с обнаруженными и вычисленными связями. Совместно с дополнительной фильтрацией предложенный алгоритм очень устойчив к ложным срабатываниям даже при слабых параметрах поиска.

Алгоритм быстрого поиска связанных пар изображений, применённый в данной работе, позволил существенно уменьшить время поиска связей между изображениями, а главное, избавиться от квадратичной зависимости времени

работы от объема входных данных. Это особенно актуально ввиду большого количества изображений, получаемых после каждой миссии АНПА.

В работе также успешно решена задача поиска начального приближения решения для положений и ориентаций снимков для инициализации нелинейной задачи минимизации. Исследования алгоритмов быстрого решения разреженных систем линейных алгебраических уравнений привели к построению простой и эффективной схеме решения систем, возникающих на этапе поиска начального приближения и нелинейного уточнения положений и ориентаций снимков.

Предложен алгоритм бесшовной сшивки изображений в фотографическую карту дна, способный справляться с большими объемами изображений. Рассмотрено два варианта сшивки: простая сшивка и сшивка на основе трёхмерной модели. Каждый из вариантов хорошо работает в своей области применения: простая сшивка для рельефа с плавным перепадом высот, а основанная на трёхмерной модели для рельефа с резким перепадом высот.

По результатам исследований, проведённых в этой работе, с использованием разработанных алгоритмов было спроектировано программное средство «AUV photo stitcher», выполняющее все процедуры, необходимые для построения фотографических карт дна. Данное программное средство показало свою работоспособность на наборах реальных подводных снимков, полученных АНПА.

К недостаткам разработанной системы можно отнести недостаточно проработанную систему фильтрации ошибок. Различного рода ошибки могут возникать на каждом этапе построения карт, соответственно фильтроваться ошибки также должны на каждом этапе. Не отфильтрованные ошибки могут приводить к некоторым искажениям, либо нечёткостям, заметным на итоговой фотографической карте. Вследствие этого, разработка эффективного блока фильтрации ошибок на всех уровнях является направлением для дальнейшего развития работы.

Кроме непосредственного применения результатов работы для построения фотографических карт подводного дна, существует возможность использования ряда наработок для решения задачи навигации АНПА по фотоизображениям дна в режиме реального времени. Так алгоритм быстрого поиска связанных пар изображений и алгоритм приближённого позиционирования снимков могут стать базовыми алгоритмами для разработки устойчивой навигационной системы. Поскольку современные системы визуальной навигации АНПА полагаются на непрерывную последовательность перекрывающихся кадров, они испытывают затруднения при длительных проходах над дном без визуальных ориентиров, либо на большой высоте, либо длинных проходов без пересечения собственной траектории. В этих случаях современные системы могут накопить слишком большую ошибку, что не позволит своевременно обнаружить пересечение со своей траекторией и скорректировать положение. Использование алгоритма быстрого обнаружения связей позволит устранить данный недостаток навигационной системы. Таким образом, ещё одним перспективным направлением для исследования является применение результатов этой работы для разработки устойчивой системы визуальной навигации АНПА.

СПИСОК ЛИТЕРАТУРЫ

1. Агеев М.Д., Киселёв Л.В., Матвиенко Ю.В. и др. Автономные подводные роботы: Системы и технологии/ под общ. ред. М.Д. Агеева. М.: Наука, 2005. – 400 с.
2. Алгоритмы обработки видеоизображений для решения задач управления и навигации автономных необитаемых подводных аппаратов / А.А. Борейко, А. В. Воронцов, А. А. Кушнерик, А. Ф. Щербатюк // Подводные исследования и робототехника. – 2010. – № 1/9. – С. 29-39.
3. Делоне Б.Н. О пустоте сферы // Изв. АН СССР. ОМОН. – 1934. – №4. С. 793 – 800.
4. Джордж А., Лю Дж. Численное решение больших разреженных систем уравнений: Пер. с англ. – М.: Мир, 1984.
5. Камаев А.Н. Автоматическая сшивка изображений, полученных с АНПА при исследовании подводного дна // В трудах конф. ТПОМО-5. – Владивосток, 30 сентября – 4 октября 2013. – С. 474-479.
6. Камаев А.Н. Исследование алгоритмов упорядочивания коэффициентов систем линейных алгебраических уравнений, возникающих в задачах компьютерного зрения // Информатика и системы управления. – 2013. – №3, с. 32-44.
7. Камаев А.Н. Относительное ориентирование снимков низкой контрастности, полученных с АНПА, на основе точечных особенностей // Учёные заметки ТОГУ. – 2011. – Том 2, №2, С. 32-42.
8. Камаев А.Н. Позиционирование изображений морского дна полученных с помощью АНПА // Подводные исследования и робототехника. – 2013. – №2/16. – С. 38-47.
9. Камаев А.Н. Создание панорамных карт подводного дна на основе больших массивов изображений // В трудах конф. Графикон 2013. – Владивосток 16-20 сентября. – 2013г. – С. 298-301.
10. Конушин. А. Геометрические свойства нескольких изображений. Компьютерная графика и мультимедиа. – 2006. – №4(3).

11. Сидоркина О.С., Юрин Д.В. Методы метапрограммирования в компьютерном зрении: 7-точечный алгоритм и автокалибровка // Graphicon'2009. – Москва, 2009. – С. 321-322.
12. Скворцов А. В. Триангуляция Делоне и её применение. – Томск: Изд-во Томского университета, 2002.
13. A lighting reproduction approach to live-action compositing / Debevec P., Wenger A., Tchou C and others // ACM Transactions on Graphics (Proc. SIGGRAPH 2002). – 2002. – № 21(3). – P. 547-556.
14. A Multiscale Retinex for Bridging the Gap Between Color Images and the Human Observation of Scenes / Daniel J. Jobson, Zia-ur Rahman, Glenn A. Woodell // IEEE Transactions on Image Processing. – 1997. – Vol. 6, № 7. – P. 965-976.
15. A survey of procedural noise functions / Ares Lagae, Sylvain Lefebvre, Rob Cook and others // Computer Graphics forum. – 2010. – vol. 29, Issue 8. – P. 2579–2600.
16. Abhishek Verma, Sugata Banerji, Chengjun Liu. A New Color SIFT Descriptor and Methods for Image Category Classification // IRAST International Congress CACS 2010. – Singapore, 2010. – P. 819-822.
17. Aguirre F, Boucher J. M., Jacq J.J. Underwater navigation by video sequence analysis // Proc. of the International Conference on Pattern Recognition. – Atlantic City, 1990. – Vol 2, P. 537-539.
18. Alaa E., Abdel-Hakim, Aly A. Frag. CSIFT: A SIFT Descriptor with Color Invariant Characteristics // Proc. of the 2006-th CVPR. – New York, 2006. – P. 1978-1983.
19. Allene C., Pons J.-P., Keriven R. Seamless image-based texture atlases using multi-band blending // Proc. of the ICPR 2008. – Tmapa, 2008. – P. 1-4.
20. An investigation of practical approximate nearest neighbor algorithms / T. Liu, A. Moore, A. Gray, K. Yang // Proc. of the conference on NIPS. – Vancouver and Whistler, 2004. – P. 825-832.
21. Anandan P. A computational framework and an algorithm for the measurement of visual motion // International Journal of Computer Vision. – 1989. – № 2(3). – P. 283-310.

22. Antone M., Teller S. Scalable extrinsic calibration of omni-directional image networks // International Journal of Computer Vision. – 2002. – № 49(2-3). – P. 143-174.
23. Atkinson, K. B. Close Range Photogrammetry and Machine Vision. – Whittles Publishing, Scotland, UK. – 1996.
24. Baker S. Lucas-Kanade 20 Years On: A Unifying Framework: Part 3. Technical Report CMU-RI-TR-03-35 // The Robotics Institute, Carnegie Mellon University, 2003.
25. Barber C. Bradford, Dobkin David P., Huhdanpaa Hannu. The quickhull algorithm for convex hulls // ACM Transactions on Mathematical Software. – 1996. – № 22 (4). – P. 469–483.
26. Bartoli A., Coquerelle M., Sturm P. A framework for pencil-of-points structure from-motion // Proc. of the 8-th European Conference on Computer Vision (ECCV 2004). – Prague, 2004. – P. 28-40.
27. Beis J., Lowe D.G. Shape indexing using approximate nearest-neighbor search in high-dimensional spaces // Proc. of the conference on CVPR. – Puerto Rico, 1997. – P. 1000-1006.
28. Bergen J. R., Anandan P., Hanna K. J., Hingorani R. Hierarchical model-based motion estimation // Proc. of the 2nd European Conference on Computer Vision (ECCV'92). – Santa Margherita Liguere, 1992. – P. 237-252.
29. Black M. J., Anandan P. The robust estimation of multiple motions: Parametric and piecewise-smooth flow fields // Computer Vision and Image Understanding. – 1996. – № 63(1). – P. 75-104.
30. Black M. J., Rangarajan A. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision // International Journal of Computer Vision. – 1996. – № 19(1). –P. 57-91.
31. Brown D. C. Close-range camera calibration // Photogrammetric Engineering. – 1971. – № 37(8). – P. 855-866.
32. Brown, M., Lowe, D.G. Invariant features from interest point groups // Proc. of the British Machine Vision Conference. – Cardiff, 2002. – P. 656-665.

33. Bugayevskiy L. M., Snyder J. P. Map Projections: A Reference Manual. – CRC Press. – 1995.
34. Capturing and viewing gigapixel images / Kopf J., Uyttendaele M., Deussen O., Cohen, M. F. // In Proc. of the ACM Transactions on Graphics. – New York, 2007. – № 26(3).
35. Chen C.-Y., Klette R. Image stitching - comparisons and new techniques // Proc. of the Computer Analysis of Images and Patterns. – Ljubljana, 1999. – P. 615-622.
36. Chen S. E. QuickTime VR – an image-based approach to virtual environment navigation // In Proc. of the ACM SIGGRAPH. – Los Angeles, 1995. – P. 29-38.
37. Ciaccia P., Patella M., Zezula P. M-tree: An efficient access method for similarity search in metric spaces // In Proc. of the 23-rd VLDB International Conference. – Athens, 1997. – P. 426-435.
38. Coorg S., Teller S. Spherical mosaics with quaternions and dense correlation // International Journal of Computer Vision. – 2000. – № 37(3) – P. 259-273.
39. Criminisi A., Reid I., Zisserman A. Single view metrology // International Journal of Computer Vision. – 2000. – № 40(2). – P. 123-148.
40. Cuthill E., McKee J. Reducing the bandwidth of sparse symmetric matrices // Proc. of the 24-th National Conference ACM. – New York, 1969. – P. 157-172.
41. El-Melegy M., Farag A. Nonmetric lens distortion calibration: Closed-form solutions, robust estimation and model selection // Proc. of the 9-th International Conference on Computer Vision (ICCV 2003). – Nice, 2003. – P. 554-559.
42. Eustice R. M., Pizarro O., Singh H. Visually augmented navigation for autonomous underwater vehicles // Journal of Oceanic Engineering. – 2008. – Vol 33, №2. – P. 103-122.
43. Fischler M. A., Bolles R. C. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography // Communications of the ACM. – 1981. – № 24(6). – P. 381-395.
44. Friedman J.H., Bentley J.L., Finkel R.A. An algorithm for finding best matches in logarithmic expected time // ACM Transactions on Mathematical Software. – 1977. – Vol 3, №3. – P. 209-226.

45. From accurate range imaging sensor calibration to accurate model-based 3-D object localization / Champleboux G., Lavall'ee S., Szeliski, R., and Brunie, L. //Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'92). – Champaign, 1992. – P. 83-89.
46. Fuh C. S., Maragos P. Motion displacement estimation using an affine model for image matching // Optical Engineering. – 1991. – № 30(7). – P. 881-887.
47. Gennert M. A. Brightness-based stereo matching // Proc. of the 2nd International Conference on Computer Vision (ICCV'88). – Tampa, 1988. – P. 139-143.
48. Golub G.H., Van Loan C.F. Matrix Computations. – The Johns Hopkins University Press, Baltimore, MD, second edition, 1989.
49. Gracias N., Negahdaripour S. Underwater mosaic creation using video sequences from different altitudes // Proc. of the MTS/IEEE OCEANS Conference. – Washington, 2005. – Vol 2, P. 1295-1300.
50. Greene N. Environment mapping and other applications of world projections // IEEE Computer Graphics and Applications. – 1986. – № 6(11). – P. 21-29.
51. Greene, N. and Heckbert, P. Creating raster Omnimax images from multiple perspective views using the elliptical weighted average filter // IEEE Computer Graphics and Applications. – 1986. – № 6(6). – P. 21-27.
52. Gremban K. D., Thorpe C. E., Kanade T. Geometric camera calibration using systems of linear equations // Proc. of the IEEE International Conference on Robotics and Automation. – Philadelphia, 1988. – P. 562-567.
53. Grossberg M. D., Nayar S. K. A general imaging model and a method for finding its parameters // Proc. of the 8-th International Conference on Computer Vision (ICCV 2001). – Vancouver, 2001. – P. 108-115.
54. Hampel F. R. Robust Statistics: The Approach Based on Influence Functions. New York: Wiley, 1986.
55. Hannah M. J. Computer Matching of Areas in Stereo Images // Ph.D. thesis, Stanford University. – 1974.
56. Harris C. and Stephens M. J. (1988). A combined corner and edge detector // Proc. of the Alvey Vision Conference. – Manchester, 1988. – P. 147-152.

57. Hartley R., Zisserman A. Multiple View Geometry in Computer Vision. – Cambridge University Press, 2004. – 672 p.
58. Haywood R., Acquisition of a micro scale photographic survey using an autonomous submersible // Proc. of the OCEANS Conference. – Washington, 1986. – Vol 5. – P. 1423–1426.
59. Herbert B., Andreas E., Tinne Tuytelaars, Luc Van Gool. SURF: Speeded Up Robust Features // Computer Vision and Image Understanding (CVIU). – 2008. – Vol. 110, № 3. – P. 346-369.
60. Herley C. Automatic occlusion removal from minimum number of images // Proc. of the International Conference on Image Processing. – Genova, 2005. – P. 1046-1049.
61. <http://hugin.sourceforge.net/> (дата обращения: 24.03.2014).
62. <http://panotools.sourceforge.net/> (дата обращения: 24.03.2014).
63. <http://research.microsoft.com/en-us/um/redmond/groups/ivm/ICE/> (дата обращения: 24.03.2014).
64. <http://www.cs.bath.ac.uk/brown/autostitch/autostitch.html> (дата обращения: 24.03.2014).
65. <http://www.ptgui.com/> (дата обращения: 24.03.2014).
66. Huber P. J. Robust Statistics. – New York: John Wiley & Sons, 1981.
67. Improved Real-Time Video Mosaicking of the Ocean Floor / S. D. Fleischer, R. L. Marks, S. M. Rock, M. J. Lee // Proc. of the MTS/IEEE OCEANS Conference. – San Diego, 1995. – Vol 3, P 1935-1944.
68. Indyk. P., Motwani R. Approximate nearest neighbours: towards removing the curse of dimensionality // Proc. of the 30-th ACM Symposium on the STOC. – Dallas, 1998. – P. 604-613.
69. Interactive digital photomontage / Agarwala A., Dontcheva M., Agrawala M. and others // Proc of the. SIGGRAPH 2004, Los Angeles, 2004. – № 23(3). –P. 292-300.
70. Irani M., Anandan P. Video indexing based on mosaic representations // Proc. of the IEEE. – 1998. – № 86(5). – P. 905-921.

71. Kang, S. B. (2001). Radial distortion snakes // IEICE Trans. Inf. & Syst. – 2001. – Vol. E84-D, №12. – P. 1603–1611.
72. Ke Y., Sukthankar R. PCA-SIFT: A More Distinctive Representation for Local Image Descriptors // In Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. – Washington, 2004. – Vol 2. – P. 506-513.
73. Keith Jack. Video Demystified. – Newnes. – 2007. – p. 944.
74. Kraus, K. Photogrammetry. – Dummmler, Bonn. – 1997.
75. Locally adapted projections to reduce panorama distortions / Kopf J., Lischinski D., Deussen O., and others // In Proc. of EGSR 2009. – Girona, 2009. – № 28(4).
76. Lowe D. G. Distinctive image features from scale-invariant keypoints // International Journal of Computer Vision. – 2004. – № 60(2). – P. 91-110.
77. Lucas B. D., Kanade T. An iterative image registration technique with an application in stereo vision // Proc. of the 7-th International Joint Conference on Artificial Intelligence (IJCAI-81). – Vancouver, 1981. – P. 674-679.
78. Luong Q.T., Faugeras O.D. The Fundamental Matrix: Theory, Algorithms, and Stability Analysis // International Journal of Computer Vision. – 1996. – № 17(1). – P. 43–75.
79. Marquard. D. An algorithm for least-squares estimation of nonlinear parameters // SIAM J. Appl. Math. – 1963. – № 11, P. 431-441.
80. Matas J. Robust wide baseline stereo from maximally stable extremal regions // Image and Vision Computing. – 2004. – № 22(10), P. 761-767.
81. Matthew Brown, David G. Lowe. Automatic panoramic image stitching using invariant features // International Journal of Computer Vision. – 2007. – Vol. 74, №1. – P. 59-73.
82. Mikolajczyk K., Schmid C. A performance evaluation of local descriptors // Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'2003). – Madison 2003. – P. 257-263.
83. Mikolajczyk K., Schmid C. Scale & affine invariant interest point detectors // International Journal of Computer Vision. – 2004. – № 60(1). – P. 63-86.

84. Moons T., Van Gool L., Vergauwen M. 3D reconstruction from multiple images // Foundations and Trends in Computer Graphics and Computer Vision. – 2010. – № 4(4).
85. Mosaic-based navigation for autonomous underwater vehicles / N.R. Gracias, S. van der Zwaan, A. Bernardino, J. Santos-Victor // IEEE Journal of Oceanic Engineering. – 2003. – Vol 28, №4. – P. 609-624.
86. Optimal texture map reconstruction from multiple views / Wang L., Kang, S. B., Szeliski R., Shum H.-Y. // Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition. – Kauai, 2001. – P. 347-354.
87. Pflugfelder R. Self-calibrating Cameras in Video Surveillance. – Ph.D. thesis, Graz University of Technology. – 2008.
88. Positioning an underwater vehicle through image mosaicking / R. Garcia, J. Batlle, X. Cufi, J. Ama // Proc. of the IEEE International Conference on Robotics and Automation. – Seoul, 2001. – Vol 3, P. 2779-2784.
89. Quam L. H. Hierarchical warp stereo // Proc. of the Image Understanding Workshop. – New Orleans, 1984. – P. 149-155.
90. Rousseeuw P. J. Least median of squares regression // Journal of the American Statistical Association. – 1984. – № 79. – P. 871-880.
91. Sawhney H. S., Kumar, R. True multi-image alignment and its application to mosaicing and lens distortion correction // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1999. – № 21(3). – P. 235-243.
92. Shi J., Tomasi C. Good features to track // Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'94). – Seattle, 1994. – P. 593-600.
93. Shum, H.-Y., Szeliski R. Construction of panoramic mosaics with global and local alignment. // International Journal of Computer Vision. – 2002. – № 48(2). – P. 151-152.
94. Single-Scale Retinex Using Digital Signal Processors / Hines G. D., Rahman Z., Jobson D. J., Woodell G. A. // *Global Signal Processing Expo (GSPx)*. – 2004.

95. Slama C.C. Manual of Photogrammetry. – American Society of Photogrammetry, Falls Church, Virginia, fourth edition. – 1980.
96. Soille P. Morphological image compositing // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2006. – № 28(5). P. 673-683.
97. Stewart C. V. Robust parameter estimation in computer vision // SIAM Reviews. – 1999. – № 41(3). – P. 513-537.
98. Szeliski R. Computer Vision: Algorithms and Applications. – Springer, 2010.
99. Szeliski R. Video mosaics for virtual environments // IEEE Computer Graphics and Applications. – 1996. – № 16(2). – P. 22-30.
100. Szeliski R., Shum H.-Y. Creating full view panoramic image mosaics and texturemapped models // Proc of the SIGGRAPH'97. – Los Angeles, 1997. – P. 251-258.
101. Tian Q., Huhns M. N. Algorithms for subpixel registration // Computer Vision, Graphics, and Image Processing. – 1986. – № 35. – P. 220-233.
102. Triggs B. Bundle adjustment — a modern synthesis // Proc. of International Workshop on Vision Algorithms. – Kerkyra, 1999. – P. 298-372.
103. Triggs B. Detecting keypoints with stable position, orientation, and scale under illumination changes // Proc. of the 8-th European Conference on Computer Vision (ECCV 2004). – Prague, 2004. – P. 100-113.
104. Tuytelaars T., Van Gool, L. Matching widely separated views based on affine invariant regions // International Journal of Computer Vision. – 2004. – № 59(1). – P. 61-85.
105. Uyttendaele M., Eden A., Szeliski R. Eliminating ghosting and exposure artifacts in image mosaics // Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. – Kauai, 2001. – P. 509-516.
106. Williams, L. Pyramidal parametrics // Proc. of the SIGGRAPH '83. – Detroit, 1983. – № 17(3). – P. 1–11.
107. www.autopano.net (дата обращения: 24.03.2014).
108. www.kekus.com (дата обращения: 24.03.2014).
109. www.serif.com (дата обращения: 24.03.2014).

110. Yu Guoshem, Morel Jean-Michel. ASIFT: An Algorithm for Fully Affine Invariant Comparison // Image Processing On Line. – 2011. – Vol. 1.
111. Zhang Z. A flexible new technique for camera calibration // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2000. – № 22(11). – P. 1330-1334.
112. Zhixi Bian, Yan Zhang // Retinex Image Enhancement Techniques – Algorithm, Application and Advantages // EE264 Image Processing and Reconstruction. – 2002.
113. Zoghلامي I., Faugeras O., Deriche R. Using geometric corners to build a 2D mosaic from a set of images // Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'97). – San Juan, 1997. – P. 420-425.