Санкт-Петербургский национальный исследовательский университет
информационных технологий, механики и оптики
Факультет информационных технологий и программирования
Кафедра компьютерных технологий

Д.П. Карпов

Сшивка изображений, полученных в результате аэрофотосъёмки

Научный руководитель: Родиков Д.Е.

Санкт-Петербург, 2012

Оглавление

Введение

В настоящее время существенно возросло использование беспилотных летательных аппаратов (БЛА) для осуществления аэрофотосъёмки местности. В результате возникает потребность в автоматизированной обработке результатов съёмки. Одна из распространённых задач обработки — это автоматическое нахождение области перекрытия и совмещение отдельных изображений (сшивка изображений). Данная задача возникает не только при аэрофотосъёмке (АФС), но и при микросъемке медицинских препаратов или микросъемке длинной детали. Полученная в результате сшивки информация об области перекрытия и взаимном расположении фотографий может быть использована для построения точных и актуальных планов местности, анализа каких-либо событий или изменений на местности.

Данная задача принадлежит к задачам компьютерного зрения, которые в настоящее время является одной из быстроразвивающихся отраслей информационных технологий. В рассмотренной задаче мы будет пользоваться методами извлечения структуры из движения (Structure from motion — далее SFM) и методами одновременной навигации и построения карт (Simultaneous localization and mapping — SLAM).

В случае АФС, который рассмотрен далее, задача допускает ряд упрощений, позволяющих увеличить скорость обработки данных. С другой стороны, возникают проблемы, связанные с плохой обусловленностью некоторых подзадач и большой погрешностью в решениях, находимых классическими методами.

В данной работе предложены два подхода, позволяющих ускорить обработку данных, полученных при АФС, а также получена асимптотическая оценка времени работы этих методов. Помимо этого, в данной работе описаны классические этапы решения задачи восстановления структуры пространства по фотографиям и некоторые методы их оптимизации.

Постановка задачи

В данной работе решается задача установки относительного положения изображений, полученных при АФС. При этом считается, что камера, установленная на БЛА, направлена примерно под прямым углом к поверхности земли. В этом случае можно пользоваться предположением, что полученные фотографии можно уложить на плоскость, и что все точки, видимые на них, будут незначительно удалены от этой плоскости по сравнению с высотой и длиной пролёта. Также считается, что внутренние параметры камеры (фокусное расстояние, угол обзора и т.д.) известны заранее.

Так как предполагается наличие сведений о параметрах камер, а также в силу вышеописанных свойств, данная задача сводится к нахождению положения камер относительно некоторой системы отсчёта (например, первой камеры) и положения точек, видимых на фотографии в пространстве. Так как длины пролёта в реальных условиях относительно небольшие, то участок Земли, видимый в данном пролёте, можно считать плоским. Устранение дисторсии, шума и прочих искажений изображения не относится к поставленной задаче. Далее считается, что эти эффекты проявляют себя незначительно.

Для фотографий, полученных в результате АФС, характерны малые области перекрытия изображений и отсутствие избыточности данных о снимаемой поверхности. Это позволяет, с одной стороны, ускорить обработку входных данных, но, с другой, усложняет точное восстановление положений камер и положений точек, видимых на фотографиях, в трёхмерном пространстве.

Далее рассматриваются основные этапы восстановления положения камеры по изображению и описываются модификации алгоритмов на этих этапах, позволяющие оптимизировать решение данной задачи как с точки зрения скорости вычислений, так и с точки зрения точности. Основу решения составляют методы эпиполярной геометрии и SFM, кратко описанные в книге «Structure from motion» [1], а также оптимизация решения (Bundle Adjustment — далее BA), описанная Триггсом в своей статье [2].

В данной работе используются два набора данных: имитация реальной АФС, которая представляет собой набор фотографий из сервиса Google Maps, и набор реальных данных, являющихся результатом заметающего пролёта над прямоугольником местности.

Решение данной задачи разбивается на этапы, каждый из которых рассматривается далее подробно. Также в данной работе предлагается метод существенного уменьшения времени рабо-

ты алгоритма за счёт особой структуры разбиения задачи на подзадачи на этапе нелинейной оптимизации решения.

Поиск точек интереса

Современные методики сшивки фотоснимков основываются на переходе к сокращённому описанию, то есть сведению изображений с многомиллионным числом параметров (цвета пикселей) к набору параметров (примитивов), число которых на несколько порядков меньше. В качестве таких примитивов могут использоваться разреженные точки изображений, контуры, простейшие геометрические фигуры, обладающие особыми свойствами, которые обеспечивают максимальную узнаваемость на других изображениях. Задача сопоставления изображений при таком описании уже может быть решена за обозримое время на современных вычислительных мощностях.

Наилучшими характеристиками в данной задаче обладает метод описания изображения при помощи точек интереса: как правило, это точки экстремума яркости изображения. Найденные точки в последствии потребуются для сопоставления точек на разных фотографиях при помощи специальных структур, описывающих их. На данный момент можно выделить два наиболее эффективных алгоритма поиска точек интереса и построения дескрипторов: SIFT и SURF.

Точки, которые находятся с помощью метода SIFT, устойчивы к растяжению, повороту изображения и частично к изменению точки наблюдения. Для стандартных изображений получается большое число точек, описания (дескрипторы) которых значительно отличаются, что делает возможным поиск по большой базе точек.

SIFT (Scale Invariant Feature Transform) — метод поиска точек интереса, предложенный Давидом Лоу [3, 4]. Основной алгоритм можно разделить на четыре этапа:

1. Нахождение экстремумов по всем шкалам и точкам изображений. Реализуется путём вычисления разности Гауссовых функций, что позволяет находить потенциально интересные точки, которые инвариантны по отношению к растяжению и поворотам.

2. Локализация ключевых точек. В каждой точке, найденной в п.1, строится детализированная модель для уточнения положения ключевой точки и её размера.

3. Добавление ориентации. Одна или несколько ориентаций добавляются к каждой точке на основании направлений градиентов. В дальнейшем все операции производятся над полученным положением, размером и ориентациями точки. Это позволяет получить устойчивость к поворотам и растяжениям изображения.

4. Дескрипторы ключевых точек. Локальные градиенты изображения измеряются по вы-

бранной шкале в районе каждой ключевой точки. Эти данные преобразуются к виду, допускающему значительную степень изменения формы и изменения освещения. Для каждой ключевой точки строится дескриптор размером в 64 (или 128) вещественных чисел.

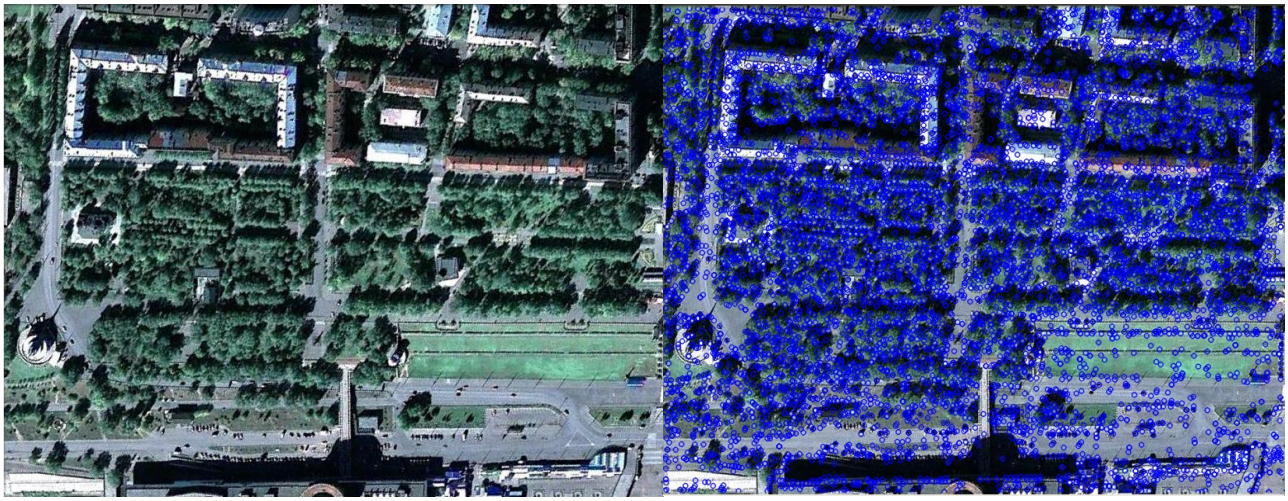

Рис.1 — Слева — пример фотографии, полученной при АФС, справа — результат работы на данной фотографии детектора SIFT

SURF (Speed Up Robust Feature) — детектор точек, предложенный Х. Бэйем. В некоторой степени он использует идеи, применённые в методе SIFT. В то же время, стандартная версия SURF в несколько раз быстрее, чем SIFT [5, 6].

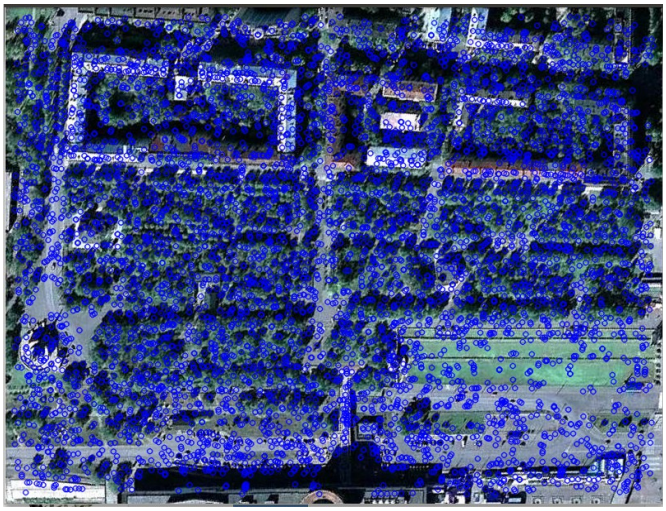

Рис. 2 — Результат работы детектора точек SURF

Метод ищет особые точки с помощью матрицы Гессе. Гессиан функции – симметрическая квадратичная форма, описывающая поведение функции во втором порядке. Матрица этой квадратичной формы образована вторыми частными производными функции. Определитель матрицы Гессе называется определителем Гессе или гессианом. Если все производные существуют, то

$$H(f) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1\, \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1\, \partial x_n} \\ \frac{\partial^2 f}{\partial x_2\, \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2\, \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n\, \partial x_1} & \frac{\partial^2 f}{\partial x_n\, \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

Детерминант матрицы Гессе (т.н. гессиан) достигает экстремума в точках максимального изменения градиента яркости. Он хорошо детектирует пятна, углы и края линий. Гессиан инвариантен относительно вращения. Для каждой ключевой точки считается направление максимального изменения яркости (градиент) и масштаб, взятый из масштабного коэффициента матрицы Гессе.

Градиент в точке вычисляется с помощью фильтров Хаара. После нахождения ключевых точек, SURF формирует их дескрипторы. Дескриптор представляет собой набор из 64 (либо 128) чисел для каждой ключевой точки. Эти числа отображают флуктуации градиента вокруг ключевой точки. Поскольку ключевая точка представляет собой максимум гессиана, то это гарантирует, что в окрестности точки должны быть участки с разными градиентами. Таким образом, обеспечивается дисперсия (различие) дескрипторов для разных ключевых точек.

Возникает естественный вопрос о выборе детектора точек. В табл. 1 рассмотрены результаты эксперимента на восьми изображениях размера 800×600 . В данной работе использовались реализации SIFT и SURF из библиотеки OpenCV.

Таблица 1 — Зависимость времени работы и количества найденных точек от типа детектора

Номер фотографии		1	2	3	4	5	6	7	8
SURF	Время работы, мс	2663	2707	2695	2654	2824	2782	2753	2750
	Число точек интереса	5842	5934	6026	5923	6276	6209	6035	6223
SIFT	Время работы, мс	7288	7150	6564	6654	7331	7370	7165	7025
	Число точек интереса	8452	8332	7517	7579	8559	8784	8392	8297

Таким образом, видно, что время работы SIFT примерно в 2,5 раза больше, чем время работы SURF. Но SIFT при этом находит больше точек. Однако для сравнения качества работы дескрипторов необходимо сопоставить ещё ряд существенных параметров. Прежде всего, важно качество сопоставления полученных точек интереса. В статье Хана делается вывод, что недостатки SURF проявляются на сильно размытых фотографиях, при сшивке разномасштабных фотографий, а также при смене угла обзора. Первые две проблемы обычно не проявляются при АФС (фотографии делаются примерно в одном масштабе, при помощи качественной оптики и с большого расстояния от объектов, что исключает размытие). Третий недостаток несущественен при прямом пролёте, так как точки на фотографиях видны примерно под одним углом. Однако использование SURF при пролёте с поворотами может привести к значительным проблемам (показано в разделе «Сопоставление точек интереса»), которые негативно влияют на качество сшивки. Значит, в общем случае использование SIFT является более надёжным. Но в случае особых наборов данных или приложений, критичных по времени, хорошим решением может стать использование SURF.

Одним из ключевых показателей будет число точек интереса, сопоставленных между несколькими последовательными изображениями. Если после работы детектора точек и функции сопоставления было найдено слишком мало пар точек, то решение может оказаться или неточным, или даже полностью неправильным.

Сопоставление точек интереса

Следующим этапом в нахождении взаимного расположения полученных изображений является сопоставление точек интереса (matching). Необходимо установить, какие точки на разных изображениях соответствуют одной реальной точке в трёхмерном пространстве. На основании этих соответствий задача нахождения взаимного расположения камер сводится к решению системы уравнений.

Поиск близких друг к другу многомерных векторов является одним из самых дорогостоящих с вычислительной точки зрения. Как показано в табл.1, на фотографии 0.5 Мп находится в среднем около 6000 точек интереса, которые и нужно сопоставить с примерно таким же числом точек на другой фотографии. По сути, задача сводится к нахождению ближайшего вектора из некоторого множества к некоторому вектору образцу. Рассматриваемое пространство многомерных векторов можно в данном случае считать евклидовым.

Эффективное решение, хорошо подходящее для данной задачи, предлагает статья «Fast approximate nearest neighbors with automatic algorithm configuration» [7]. В данной статье описан быстрый способ поиска приблизительно ближайшего соседа. Данный метод превосходит по быстродействию линейный поиск. В основе метода лежит использование *k*-мерных деревьев, которые в классическом виде эффективны для малых размерностей пространства. В случайных же *k*-мерных деревьях разбиение на поддеревья происходит по измерению, которое выбирается случайно из D измерений с наибольшей дисперсией. При этом будем использовать *i*-арные деревья, чтобы обеспечить большую гибкость алгоритма. При построении дерева точки рекурсивно разбиваются на кластеры и размещаются в дереве, на каждом шаге кластер делится на *i* новых кластеров. Данный алгоритм зависит от ряда параметров: размерность и арность деревьев, искомая точность аппроксимации ближайшего соседа.

Рис. 3 — Результат сопоставления точек интереса при прямом пролёте

Рис. 4 — Результат сопоставления точек интереса при пролёте с поворотом

Одним из важных вопросов в общем случае работы с детекторами точек является установление пороговых значений: расстояний между векторами-дескрипторами, при котором пару точек на разных фотографиях можно сопоставить одной точке в пространстве. Однако при аэрофотосъёмке этот вопрос не столь актуален, так как дескрипторы точек интереса в данной задаче получаются достаточно уникальными. Но при этом проявляется одна из проблем детектора SURF: меньшая устойчивость его дескрипторов к смене угла обзора (повороту изображения). Из табл. 2 видно, что на фотографиях 2 и 3, а также на фотографиях 4 и 5 (изображены на рисунке 4) число пар точек, найденных детектором SIFT, существенно больше, чем число точек найденных детектором SURF. При этом число пар точек между картинками с прямым пролётом сопоставимы. Таким образом, возникает конфликт «качества» работы алгоритма и его быстродействия. Выбор детектора зависит от типа решаемой задачи. В критичных по времени задачах, та-

ких как сшивка изображений в реальном времени, SURF кажется более приоритетным вариантом. В других задачах использование SIFT может избавить от ряда серьёзных проблем, связанных с недостаточностью статистической выборки.

Основным недостатком SURF в данной задаче является плохая устойчивость к повороту изображения (несмотря на то, что во многих источниках сказано обратное). Далее приведён вычислительный эксперимент, подтверждающий это. В нем рассмотрены две последовательных фотографии, полученные при АФС. При этом точки на фотографиях будут сопоставляться двумя способами: непосредственно на двух последовательных снимках и в случае, когда вторая фотография развёрнута на 90°. Далее построен график зависимости отношения количества сопоставленных пар от пороговых значений в SIFT и SURF. При этом крайние на графиках пороговые значения выбраны так, что число сопоставленных точек при них у SIFT и SURF примерно равны на двух последовательных изображениях. Очевидно, что чем ближе данный коэффициент к единице, тем устойчивей полученные дескрипторы к повороту. Из рисунка 5 следует, что SIFT значительно лучше SURF по этому показателю.

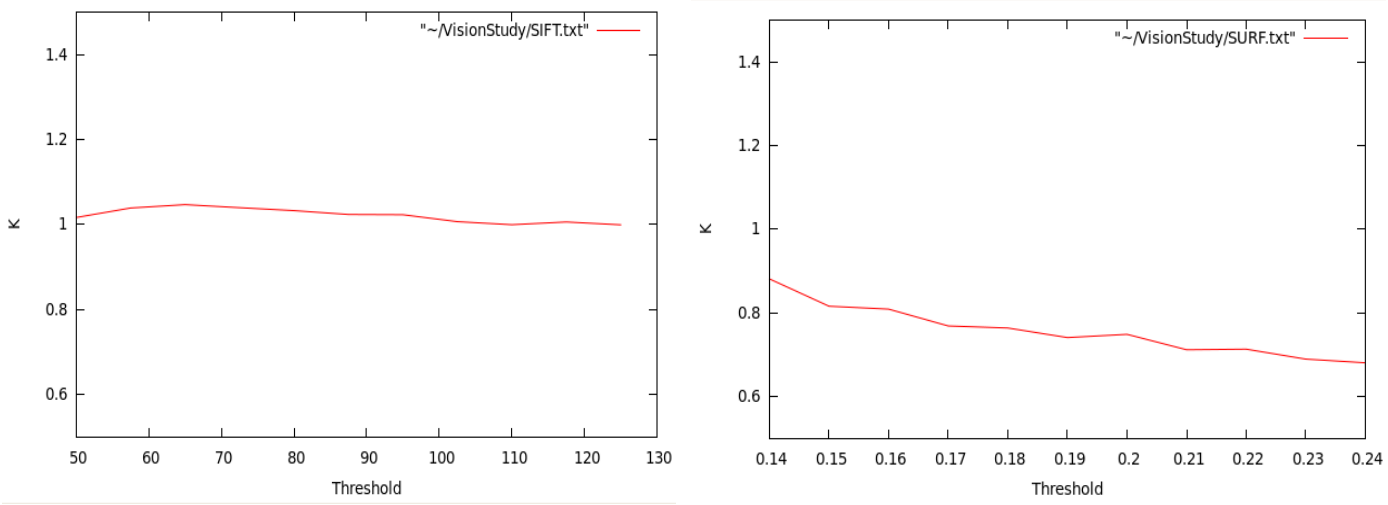

Рис. 5 — Зависимость отношения количества точек от пороговых значений (слева для детектора SIFT, справа — для SURF)

Другой существенной проблемой является наличие различных шумов на изображении. Чтобы теоретически решить задачу о взаимном расположении камер, достаточно восьми пар точек [1]. Однако шум приводит к тому, что решение, полученное в чистом виде по восьми точкам, может быть ошибочным. Для решения этой проблемы приходится использовать статистические методы обработки выбросов. Алгоритмы детекторов и сопоставления позволяют получить до-

статочное число пар точек.

Таблица 2 — Число найденных при сопоставлении «хороших» пар точек.

SIFT							SURF						
Номера фото-графий	1	2	3	4	5	6	Номера фото-графий	1	2	3	4	5	6
1	8451	1351	66	0	0	0	1	5842	1540	27	0	0	0
2	1351	8313	898	6	0	1	2	1540	5954	386	0	0	0
3	66	898	7469	1066	9	0	3	27	386	6034	1251	1	0
4	0	6	1066	7575	907	0	4	0	0	1251	5942	149	0
5	0	0	9	907	8528	1211	5	0	0	1	149	6279	1068
6	0	1	0	0	1211	8693	6	0	0	0	0	1068	620

Использование эпиполярной геометрии

В данном разделе в качестве модели камеры используется стеноп (pinhole camera) — фотографический аппарат без объектива, роль которого выполняет малое отверстие (так как считается, что все нелинейные эффекты, например дисторсия, были устранены, то это возможно). В рамках данной модели получение изображения камерой можно получить в три этапа:

1. Преобразование координат точки из мировых координат в координаты камеры. Пусть $[X_C, Y_C, Z_C, 1]^T$ — обобщённые координаты точки в системе координат камеры, $[X, Y, Z, 1]^T$ — обобщённые координаты точки в мировой системе координат. Тогда это соотношение можно записать в следующем виде $\begin{bmatrix} X_C \\ Y_C \\ Z_C \\ 1 \end{bmatrix} \sim \begin{bmatrix} R\,T \\ 0\;1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$, где R — матрица поворота, которая представляет ориентацию камеры в мировых координатах, а T — координаты центра мировых координат в системе камеры. Эти параметры называются внешними параметрами камеры.

2. Следующее преобразование сопоставляет новым координатам точку на плоскости проекции камеры. Данная плоскость расположена на расстоянии фокусной длины от центра координат камеры. Пусть трёхмерная точка имеет обобщенные координаты $[X_C, Y_C, Z_C, 1]^T$, а двухмерная — $[x, y, 1]^T$. Тогда соотношение между координатами можно записать следующим образом $x = f \cdot \frac{X_C}{Z_C}, y = f \cdot \frac{Y_C}{Z_C}$.

3. Последний этап преобразования: преобразование координат точки на плоскости проекции в координаты точки изображения. $U = [u, v, 1]^T$ — координаты точки на изображении, $X_P = [x, y, 1]^T$. Тогда $U \sim K \cdot X_P$, где K — калибровочная матрица. $K = \begin{bmatrix} \alpha_u & s & u_0 \\ 0 & \alpha_v & v_0 \\ 0 & 0 & 1 \end{bmatrix}$.

K, f — внутренние параметры камеры. Они не меняются от снимка к снимку и в данной задаче их можно считать известными. Значение фокальной длины также можно учесть в матрице K.

Поэтому преобразование можно записать в виде $U \sim P \cdot X$, где $P \sim K \cdot [R\ T]$ - проективная матрица размера 3×4 .

Если две точки на проективной плоскости камеры соответствуют одной и той же трёхмерной точке в пространстве, то это свойство можно выразить математически через основную матрицу *E*. Пусть *x* — проекция точки на одну камеру, а *x′* — на другую. Тогда верно равенство

$x^T \cdot E \cdot x' = 0$, где $E = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \cdot R$, $T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$.

Но $x \sim K^{-1} \cdot u,\ x' = K^{-1} \cdot u'$, где *u*, *u′* — координаты соответствующих точек на изображениях. Поэтому зависимость принимает вид $u'\left(\left(K^{-1}\right)^T \cdot E \cdot K^{-1}\right)u = 0$. Обычно вводится обозначение $F = \left(K^{-1}\right)^T \cdot E \cdot K^{-1}$ — фундаментальная матрица [1].

Вычисление фундаментальной матрицы

Самый простым и известным алгоритмом вычисления фундаментальной матрицы является классический алгоритм восьми точек [8]. Спустя некоторое время алгоритм подвергся существенной критике, утверждалось, что он плохо обусловлен. Были предложены более сложные итеративные алгоритмы. Однако Р. Хартли в своей статье показал, что рядом модификаций можно добиться того, чтобы алгоритм работал сопоставимо с предложенными итеративными [9].

Поиск фундаментальной матрицы осуществляется из уравнения $u'^{T} \cdot F \cdot u = 0$. Для каждой пары точек получаем одно уравнение относительно $F_{i,j}$. Важным свойством матрицы *F* является то, что ее ранг равен двум. То есть решение уравнения имеет одну степень свободы. Чтобы избавиться от этого $F_{3,3}$ считается равным единице и решение ищется по методу наименьших квадратов.

Легко показать, что если $\tilde{u} \sim T \cdot u$, $\tilde{u}' \sim T' \cdot u'$, то $F = T'^{T} \cdot \tilde{F} \cdot T$. Для того, чтобы задача поиска методом наименьших квадратов была хорошо обусловлена, производится процесс нормировки. Центр масс точек при этом переносится в начало координат, а среднее расстояние от начала координат до точек при этом устанавливается равным некоторой величине (как было получено экспериментально, оптимальное среднее расстояние $\sqrt{2}$). Далее для нахождения достаточно применить обратное преобразование, изначально потребовав сингулярность матрицы $\tilde{F}$.

Для того, чтобы избавиться от статистических выбросов, в качестве эффективного решения может быть использован метод RANSAC (RANdom SAmple Consensus). Это итеративный метод, позволяющий при достаточном размере выборки отделить шум в данных от информации о реальных объектах. Помимо этого, RANSAC позволяет эффективно отделить выбросы от элементов, соответствующих модели. В обобщённом виде данный алгоритм можно разбить на несколько этапов:

1. Случайным образом выбирается несколько элементов, для которых предполагается, что они удовлетворяют модели.

2. Вычисляются параметры модели, исходя из выбранных ранее точек.

3. Все остальные элементы данных проверяются в рамках данных параметров модели на принадлежность ей.

4. Если элементов, удовлетворяющих модели, достаточно много, то получен результат. Далее при необходимости можно получить дополнительную информацию: погрешность выбранной модели, число выбросов и т.д.

5. Если элементов, удовлетворяющих модели, мало, то параметры модели пересчитываются.

Таким образом на основании RANSAC легко получить набор точек, удовлетворяющих модели, и получить параметры самой модели, то есть фундаментальную матрицу. Путем обратного преобразования легко получить основную матрицу. А ее, в свою очередь, можно разложить и получить вектор трансляции и матрицу поворота.

Оптимизация решения

Как и для всех вычислительных задач, для данной задачи характерна проблема накопления погрешности. Для того, чтобы избавиться от этого, применяются различные методы оптимизации решения. Наиболее распространенным методом является нелинейная оптимизация, описанная в классической статье Триггсом [2]. На данном этапе задача сводится к минимизации функции суммарной ошибки репроектирования между наблюдаемым и предсказанным положением точек на изображениях, которая выражается в виде суммы квадратов большого количества нелинейных вещественных функций.

Одним из наиболее широко используемых и хорошо зарекомендовавших себя алгоритмов нелинейной минимизации является алгоритм Левенберга-Марквардта, который обладает достаточно быстрой сходимостью в широком диапазоне начальных предположений. При этом у данного метода существует ряд модификаций, направленных на улучшение быстродействия. Эти улучшения учитывают структуру функций характерных именно для данной задачи компьютерного зрения и позволяют существенно улучшить общий алгоритм. Одной из самых эффективных и распространенных модификаций является Sparse Bundle Adjustment (SBA). Асимптотически время работы такого алгоритма SBA оценивается как $O(N^3)$.

Однако, несмотря на все оптимизации, SBA требует существенных вычислительных ресурсов и его оптимизация может значительно улучшить быстродействие алгоритма. В данной задаче не была еще учтена особая структура получаемых фотографий, а именно, связи между фотографиями, характерные для заметающего пролета. В силу больших объёмов данных аэрофотосъёмки, даже небольшие локальные погрешности имеют свойство накапливаться и приводить к существенным отклонениям в местоположении отдельных частей реконструируемой сцены. Поэтому возникает необходимость нелинейной оптимизации не только для всего набора фотографий в совокупности, но и отдельных его подмножеств.

Существует ряд общих подходов для восстановления трехмерных сцен, например, инкрементальный поход [11] или дерево кластеров [12]. При этом асимптотическая сложность по времени первого алгоритма составляет $O(N^5)$, а второго — $O(N^5)$ в худшем случае, $O(N^4)$ в лучшем, где N – число фотографий (различных позиций камер). Однако данные алгоритмы не учитывают специфику данных. Поэтому для задачи аэрофотосъёмки может быть найдено решение с лучшей асимптотикой. Характерное расположение фотографий и их пересечений имеет типичную структуру для так называемого заметающего пролёта (рис. 6). Данный

способ облета местности является самым распространенным, так как позволяет отснять определенный прямоугольник местности, например, для составления фотоплана.

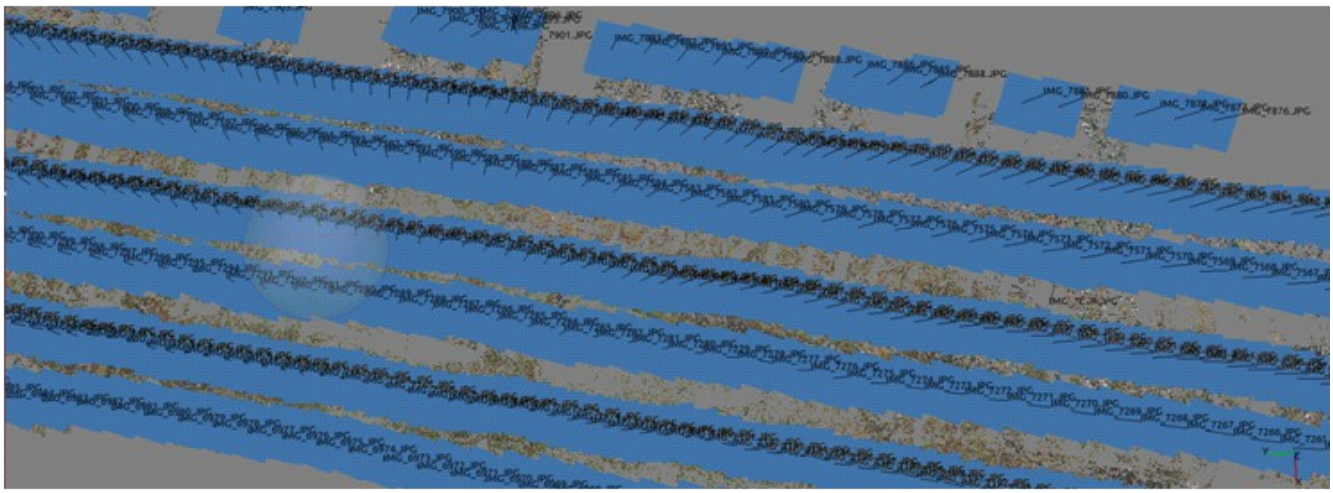

Рис. 6 — Типичное расположение камер при заметающем пролете

Таким образом, можно действовать в предположении, что камеры расположены по прямоугольнику с частичным пересечением их изображений. Пусть данный прямоугольник имеет L фотографий в длину и W в ширину, тогда $N \approx L \cdot W$. При этом фотографии можно разбить на группы так, чтобы две соседние группы имели общие точки. Тогда, во-первых, расположение камер при оптимизации напрямую зависит от положения камер в соседних группах. Во-вторых, если группа размера $w \cdot l$ имеет общие точки с соседней группой, то группа размера $(w+1)\cdot(l+1)$ также обладает этим свойством. Также можно считать, что при АФС на каждой фотографии находится примерно одинаковое точек интереса, равное p. Тогда число точек интереса на N фотографиях равно $p \cdot N$.

Рассмотрим случай, когда фотографии приходят последовательно и формируют группы по l экземпляров. При этом в большинстве случаев можно считать, что $w = 1$ (как правило, такое полётное задание позволяет минимизировать общую длину полета, получив при этом необходимую информацию). Пусть камеры последовательно выравниваются внутри каждой группы, а потом каждая группа выравнивается относительно предыдущей оптимизированной группы размера l и предыдущего (также уже с минимизированной ошибкой) пролета по прямой в другом направлении размера L камер.

Оценим асимптотику предложенного алгоритма, считая l константой. В одной группе оптимизация производится инкрементальным методом, сложность которого $O(p \cdot N^5)$, при этом время работы Bundle Adjustment оценивается как $O(p \cdot N^4)$. Тогда ошибка репроецирования в

рамках одной группы оптимизируется за время $C \cdot p l^5$. Для всех фотографий первого ряда пролёта положение камер оптимизируется только относительно предыдущей группы выполнением ВА для 2*l* камер за время $C_1 \cdot p l^5 + C_2 p \cdot 2l \cdot (2l^3)$. Для фотографий, укладываемых после первого поворота БЛА, производится также выравнивание относительно предыдущего ряда. Но положение группы из *l* фотографий в силу структуры пролёта непосредственно зависит только от части фотографий предыдущего ряда. Их число линейно зависит от размера группы, при этом в большинстве случаев структура пролёта позволяет ограничить его числом $3 \cdot l$. То есть при сшивке фотографий второго и последующего ряда выравнивание фотографий производится внутри группы, а также относительно предыдущей группы и трёх ближайших групп по числу общих точек с предыдущим рядом (то есть всего $l + l + 3 \cdot l = 5 \cdot l$ камер). При этом первая группа в каждом ряду оптимизируется только вместе с 3*l* камерами предыдущего ряда. Таким образом, получаем следующую асимптотическую оценку времени работы:

$$T(N) \sim p \cdot l^5 + \sum_{i=2}^{L/l} (p \cdot l^5 + 2 \cdot p \cdot l \cdot (2l)^3) + \sum_{j=2}^{W} (p \cdot l^5 + p \cdot (l + 3 \cdot l) \cdot (l + 3 \cdot l)^3 + \sum_{i=2}^{L/l} (p \cdot l^5 + p \cdot (2l + 3l) \cdot (2l + 3l)^3))$$

Первое слагаемое *T(N)*, также как и первые слагаемые каждой из сумм, описывает время работы ВА внутри одной группы фотографий. В первой сумме второе слагаемое соответствует времени работы ВА на выравнивание двух соседних последовательных групп по *l* фотографий. Во второй сумме и вложенной сумме значение слагаемых аналогично, только добавляется также выравнивание относительно *L* фотографий предыдущего ряда.

$$T(N) \sim p \cdot l^5 + O(L) + p \cdot (W - 1) \cdot (O(l^5) + (\frac{L}{l} - 1) \cdot O(l^5))$$

Так как *l = const(N)*, то $T(N) = O(W \cdot L) = O(N)$. Данный метод может быть эффективно применён для приложений, требующих обработку фотографий в режиме реального времени, однако при его применении на некоторых наборах данных (например, при плохой обусловленности задачи эпиполярной геометрии) может наблюдаться плохая сходимость решения, полученного данным методом.

Другой подход возможен лишь когда все фотографии и данные о их порядке уже получены. При этом полученное решение является более стабильным (сходимость к нему меньше зависит от обусловленности задачи), но требует больше времени на его нахождение. Рассмотрим в тех же обозначениях прямоугольник, отснятый с БЛА при заметающем пролёте. Пусть выбраны несколько опорных фотографий. Так как в дальнейшем рассматривается асимптотическая оцен-

ка времени работы, то можно считать, что положение камеры и точек, видных на ней, зависят только от соседей первого порядка. На первой итерации относительно них выравниваются все соседние фотографии, включая диагональные границы (можно считать, что камеры видят некоторые квадраты сетки прямоугольника). Далее на каждом шаге оптимизируется положение камер совместно с положением уже выравненных камер группы. При этом две группы могут пересекаться только своим внешним периметром из одной камеры. На схеме (рис. 7) изображён принцип построения групп, внутри которых производится оптимизация для двух камер. Белым отмечены камеры не принадлежащие ни одной из групп, серым — камеры только первой группы, чёрным — только второй, в два цвета покрашены пересечения групп камер.

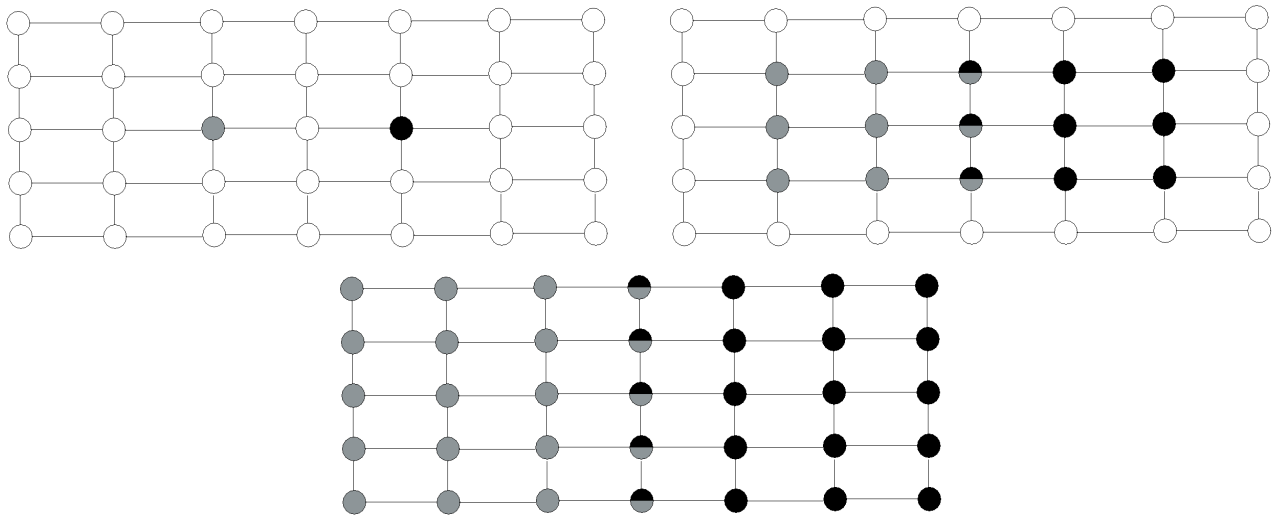

Рис. 7 — Формирование групп камер. Вверху слева — нулевая итерация, вверху справа — первая, внизу — вторая

Когда все камеры оказываются выравнены относительно некоторой точки, полученные группы выравниваются друг относительно друга. Пусть на первый этап требуется время T_1, а на второй — T_2. Пусть в прямоугольнике не остаётся непокрытых камер за *t* итераций. Тогда в нем достаточно изначально разместить не более

$$\left(\frac{W}{2t+1}+1\right)\cdot\left(\frac{L}{2t+1}+1\right)=\frac{N}{(2t+1)^2}+\frac{W+L}{2t+1}+1=H$$ точек. Тогда $T=T_1+T_2$,

$$T_1\sim\sum_{i=1}^{H}\sum_{j=1}^{t} p\cdot(2j+1)^2\cdot((2j+1)^2)^3$$

Первое суммирование в формуле производится по числу всех групп, второе по числу

итераций. Таким образом учитывается время оптимизации каждой группы на каждой итерации в отдельности.

$$T_1 \sim \sum_{i=1}^{H} p \cdot \sum_{j=1}^{t} (2j+1)^8 \le \sum_{i=1}^{H} O(t^9) = O(H \cdot t^9) = O(N \cdot t^7 + (W+L) \cdot t^8 + t^9)$$

После проведения оптимизации внутри каждой группы, остаётся провести последовательное выравнивание групп относительно друг друга. Пусть в одной группе содержится $f(N) = l(N) \times w(N)$ фотографий, тогда всего точек интереса в группе $p \cdot f(N)$. Слияние групп между собой производится согласно оптимизации иерархического метода, описанной Родиковым Д.Е. в магистерской работе [13]. Так как при описанной структуре данных группы фотографии, граничащие по вертикали или по горизонтали, пересекаются, то можно проводить объединение пар групп по вертикальным и горизонтальным границам до тех пор, пока не останется одна группа. Группы, рассмотренные на первом шаге алгоритма, назовём элементарными. Пусть набор данных разбит на одинаковое число элементарных групп по длине и по ширине (то есть схема объединённые группами камеры образуют квадрат), причём число групп по стороне этого квадрата равно 2^S .

На каждом шаге выбирается сторона прямоугольника, относительно которой число групп больше, и эти группы объединяются попарно, через границы, ортогональные выбранной стороне. При этом, так как на каждой итерации число групп по одной из сторон уменьшается в 2 раза. Следовательно, общее число итераций равно $\log_2 W + \log_2 L = \log_2 N$. Так как число групп камер равно степени двойки, то на каждом шаге не будет оставаться не объединённых групп. При этом на *j*-м шаге каждая группа состоит из 2^j элементарных групп, то есть всего в каждой группе содержится по $f(N) \cdot 2^j$ фотографий. Тогда каждая группа, объединяемая на *j*-м этапе, имеет размер $i_j \times k_j$ элементарных групп. При этом очевидны следующие свойства данной последовательности пар: $i_1 \cdot k_1 = 2, i_{j+1} \cdot k_{j+1} = 2 \cdot i_j \cdot k_j$. Следовательно, $i_j \cdot k_j = 2^j$. Так как на каждой итерации оптимизируется $\frac{L}{l(N) \cdot i_j} \cdot \frac{W}{w(N) \cdot k_j} = \frac{N}{f(N) \cdot 2^j}$, то

$$T_2 \sim \sum_{i=1}^{\log_2 N} p \cdot (2 \cdot 2^j \cdot f(N))^4 \cdot \frac{N}{f(N) \cdot 2^j} = p \cdot N \cdot f^3(N) \cdot \sum_{j=1}^{\log_2 N} 2^{3j+4} = O(N \cdot f^3(N) \cdot N^3) = O(N^4 \cdot f^3(N)) \quad .$$

Пусть теперь элементарные группы фотографий образуют квадрат с произвольным числом групп по сторонам, например, $A \times A = N$. Выберем такое максимальное *m*, чтобы было выпол-

нено неравенство $B=2^m \le A$. Ясно, что $B \cdot B \le N$ Тогда, чтобы произвести оптимизацию камер по всему квадрату, достаточно провести четыре оптимизации для квадратов размера *A*, расположенных по углам большого квадрата и одну глобальную оптимизацию по всем группам камер. Таким образом, алгоритм при этом имеет сложность:

$$\tilde{T}_2 \sim 4 \cdot O(B^8 \cdot f^3(B^2)) + (p \cdot f(N) \cdot N) \cdot (f(N) \cdot N)^3 \le 4 \cdot O(N^4 f^3(N)) + O(N^4 \cdot f^4(N)) = O(N^4 \cdot f^4(N)) \quad .$$

Если же набор данных представляет собой вытянутый прямоугольник, то применяется следующий набор действий: разбить более длинную стороны на группы так, чтобы в результате получился набор квадратов; провести оптимизацию положений камер внутри каждого квадрата и объединить группы квадратов иерархическим методом. В случае если часть групп не объединена (так как *k* не обязательно является степенью двух и на каждой итерации может получаться остаток), то на последнем этапе выполняется оптимизация этих остатков с уже полученными группами большего размера. Пусть $L = k(N) \cdot W$, но в случае АФС какая либо зависимость *L* от *W* является неестественной, поэтому можно считать, что $k(N) = const(N)$. Тогда

$$T_2 \sim k \cdot O((W^2)^4 \cdot f^4(N)) + \sum_{j=1}^{\log_2 k} 2^j \cdot O(W^8 \cdot f^4(N)) + \sum_{j=1}^{\log_2 k} (p \cdot 2^j \cdot W^2 + p \cdot 2^{j-1} \cdot W^2)^4 \cdot f^4(N) = O(N^4 \cdot f^4(N)) \quad .$$

Тогда можно выбрать $t \sim const$, то есть выбирать на первом этапе для оптимизации $O(N)$ кластеров камер. Тогда $f(N) = const(N)$ и, следовательно:

$$T = T_1 + T_2 = O(N) + O(N^4) = O(N^4) \quad .$$

Ясно, что наилучшая асимптотическая оценка времени работы достигается при $t = const$, так как любая зависимость $t = f(N)$, где *f* — монотонно возрастающая, приводит как к росту T_1 , так и к росту T_2 . Данный подход является более стабильным с точки зрения сходимости к решению, чем первый метод. В рамках данной задачи он является более эффективным по времени, чем иерархический или инкрементальный подход. Таким образом, данный алгоритм достигает лучшей асимптотической оценки времени работы, что позволяет добиться существенного увеличения производительности приложений, использующих его.

В данном разделе были преложены два алгоритма. Один из них является очень быстрым, но при этом может плохо сходится к решению, другой обладает гораздо лучшей сходимостью, но требует больше времени для обработки данных. Описанные выше алгоритмы используют свойства структуры данных, за счет чего достигается преимущество в быстродействии. Но, с другой стороны, эти алгоритмы привязаны к конкретной структуре входных данных, и существенные отклонения от данных от этой структуры могут привести к негативным последствиям.

Заключение

В данной работе были рассмотрены основные методы, применяемые при восстановлении относительного положения точек съёмки и разреженной структуры пространства. Данные методы в указанном порядке позволяют решить задачу SFM. В дальнейшем достаточно лишь аппроксимировать найденные 3D точки плоскостью и уложить фотографии на эту плоскость. Таким образом, поставленная задача решена. При этом на каждом этапе были выбраны наиболее подходящие методы. Был показан основной недостаток детектора точек SURF, который заключается в неустойчивости найденных им точек интереса к повороту. Однако SIFT при этом имеет другой существенный недостаток — значительно большее время работы. В данной задаче выбор детектора нельзя сделать однозначно, это решение зависит от требований к конкретному приложению. По сути приходится выбирать между надёжностью и быстродействием.

Было указано сочетание методов, позволяющих улучшить обусловленность задачи нахождения матрицы поворота. Кратко описаны методы эпиполярной геометрии.

В данной работе были предложены два способа оптимизации решения. Они позволяют увеличить быстродействие алгоритма по сравнению с общими методами решения задачи SLAM, используя структуру данной задачи. При этом достигается не какое-то увеличение быстродействие алгоритма на конкретных наборах данных, а улучшение асимптотической оценки времени работы алгоритма. При этом первый из описанных методов обладает лучшей асимптотической оценкой, но при его использовании могут возникать проблемы с точностью найденного решения. Второй метод обеспечивает сходимость к решению при достаточно хорошем начальном приближении, но при этом требует большее время для решения задачи.

При этом данную задачу можно развивать далее. Например, существенно увеличить быстродействие может алгоритм, использующий параллельные вычисления. Также важной проблемой является увеличение точности найденного решения.

Литература

[1] Robertson D. P., Cipolla R.. Structure from motion. 2008.

[2] B. Triggs; P. McLauchlan and R. Hartley and A. Fitzgibbon (1999). Bundle Adjustment — A Modern Synthesis. ICCV '99: Proceedings of the International Workshop on Vision Algorithms. Springer-Verlag. pp. 298–372.

[3] D. Lowe. Distinctive Image features from scale invariant keypoints, International journal of Computer Vision, Vol. 60, pp. 91-110, 2004.

[4] Lowe, D.G. 1999. Object recognition from local scale-invariant features. In International Conference on Computer Vision, Corfu, Greece, pp. 1150-1157.

[5] Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool. SURF: Speeded Up Robust Features, Computer Vision and Image Understanding (CVIU), Vol. 110, No. 3, pp. 346-359, 2008

[6] N. Khan, B. McCane, G. Wyvill. SIFT and SURF Performance Evaluation Against Various Image Deformations on Benchmark Dataset. International Conference on Digital Image Computing: Techniques and Applications, 2011.

[7] M. Muja, D. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration, 2009.

[8] H.C. Longuet-Higgins. A Computer Algorithm for Reconstructing a Scene From Two Projections, Nature, vol. 293, pp. 133–135, Sept 1981.

[9] Richard I. Hartley. In Defense of the Eight-Point Algorithm. IEEE Transactions On Pattern Analysis and Machine Intelligence, vol. 19, no. 6, june 1997

[10] Ni K., Steedlyy D., Dellaert F.. Out-of-Core Bundle Adjustment for Large-Scale 3D Reconstruction. 2007.

[11] Klopschitz M., Irschara A., Reitmayr G., Schmalstieg D.. Robust Incremental Structure from Motion. 2010.

[12] Michela Farenzena, Andrea Fusiello, Riccardo Gherardi. Structure-and-motion pipeline on a hierarchical cluster tree. Computer Vision Workshops (ICCV Workshops), 2009

[13] Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики, Факультет информационных технологий и программирования, Кафедра компьютерных технологий «Иерархическое восстановление разреженной структуры пространства и точек съёмки по набору фотографий», Д. Е. Родиков (Магистерская работа)