Предварительное грубое совмещение изображений по найденным на них прямым линиям для построения мозаик, сверхразрешения и восстановления трехмерных сцен

Д.Б. Волегов*, Д.В. Юрин**
* Московский физико-технический институт (государственный университет)
** Факультет вычислительной математики и кибернетики МГУ им. М.В.Ломоносова
dvolegov@rambler.ru yurin_d@inbox.ru

Аннотация

В работе рассматривается задача грубого совмещения изображений трехмерных сцен по обнаруженным на них прямым линиям. Предложен алгоритм быстрого вычисления преобразования Хафа для прямых линий через преобразование Хартли. Это дает экономию в памяти примерно в два раза и уменьшению времени работы примерно в 6 раз. Предложен алгоритм поиска прямых линий на цветных изображениях. Предложен алгоритм грубого совмещения изображений, использующий информацию о геометрических параметрах прямых линий и о распределении цвета и яркости вокруг прямых линий. Для учета цвета используется аппарат нечеткой логики. Результатом работы алгоритма является плоское проективное преобразование (планарная гомография), приближенно совмещающее изображения. Получена оценка размера окна поиска соответствующих точек после грубого совмещения, что позволяет повысить эффективность работы алгоритмов поиска соответствующих точек и алгоритмов плотного восстановления 3D по нескольким кадрам а также применять известные подходы восстановления формы и движения не только к видео но и кадрам, полученным с фотоаппарата с рук.

Ключевые слова: *совмещение изображений, обнаружение прямых линий, преобразование Радона, преобразование Хафа, преобразование Хартли, характеристические точки, установление соответствий, восстановление трехмерных сцен, нечеткая логика.*

1. ВВЕДЕНИЕ

В настоящей работе рассматривается задача поиска проективного преобразования, грубо совмещающего два изображения в целом. Известно [1], что различные изображения трехмерной сцены связаны эпиполярными ограничениями. В трех ситуациях геометрии съемки существует планарная гомография между изображениями, т.е. взаимно однозначное линейное соответствие между однородными координатами пикселей двух изображений, соответствующих одной и той же точке трехмерного объекта [1,12]:

1) произвольные положения камер, но трехмерная сцена представляет собой плоскость;

2) произвольная трехмерная сцена, камера произвольно вращается и меняет зум при неподвижном оптическом центре;

3) очень удаленная сцена, так, что расстояние до камеры значительно превышает глубину сцены в направлении наблюдения и перемещение камеры, поэтому параллакс точек невелик; эта ситуация часто реализуется при аэрокосмической съемке.

В остальных случаях не существует гомографии, точно переводящей первое изображение во второе. Тем не менее, рассматриваемая постановка задачи представляется многообещающей в качестве предварительного шага при построении мозаик и восстановлении трехмерных сцен по набору изображений. Основная идея основана на следующей аналогии: человек при необходимости сравнения двух изображений стремится изучать их в близком масштабе, расположить их рядом и ориентировать их одинаково, и нет причин при компьютерном подходе решать более сложную задачу.

В случае пренебрежимо малых перспективных искажений (параллакса) для создания мозаик могут использоваться подходы [2-4]. Метод [2] основан на использовании свойств преобразования Фурье (Меллина) относительно трансляции, поворотов и масштабирования. Алгоритм работоспособен при коэффициенте масштабирования вплоть до 1.8 раза [2]. Задача поиска угла поворота и коэффициента масштабирования сведена к поиску трансляции путем перевода изображения амплитуды спектра в логарифмически-полярную систему координат, осуществляемую билинейной интерполяцией. Однако алгоритм плохо работает, если в совмещаемых изображениях присутствуют значительные не перекрывающиеся области. Вопрос о поведении алгоритма при наличии искажений, в том числе перспективных, не исследовался. Интуитивное решение заключается в поиске совмещающего преобразования только по крупным объектам на изображении, т.е. подавлению в спектре высоких частот. Однако, в низкочастотной области слишком мало достоверных данных, так, для объектов с характерным размером порядка половины изображения, вся информация в спектре представлена четырьмя пикселями-соседями нулевой частоты в спектре, которые, при билинейной интерполяции, размазываются по всей оси поворота. Метод успешно применяется в задачах совмещения космических снимков, где перспективные искажения не велики и/или легко компенсируемы, а грубая оценка совмещения изображений может быть получена из телеметрических данных. Подход [3] обычно не позиционируется как метод решения задачи совмещения изображений (Image Registration, Image Matching), однако в ряде случаев может быть с успехом применен. Его достоинством является поиск произвольного аффинного преобразования и достижимость субпиксельной точности. В [2] субпиксельная точность может быть достигнута за счет подбора соответствующего фильтра для поиска максимума изображении корреляции фаз, а также увеличением размера изображения путем дополнения его нулями до следующей степени двойки, что приводит к росту вычислительной сложности более чем в 4 раза. К недостаткам в данном контексте следует отнести то, что алгоритм [3] работает только при малых отклонениях от искомого положения, величина обрабатываемого отклонения определяется полушириной функции Гаусса, используемой при вычислении производных. В отличие от [2] метод не приводит к рассмотрению различных гипотез, а приводит в ближайший локальный минимум функционала. Частично проблема может быть решена при обработке пары изображений по пирамиде детальности. Следует отметить также работу [5], где алгоритм [3] в сочетании с сегментацией используется для решения задачи восстановления трехмерной сцены по стереопаре. Метод [4] имеет более широкий диапазон применения, в частности допустимое различие масштабов может составлять до 6 раз, отсюда ясно, и степень перекрытия изображений может быть невелика, ищется произвольное преобразование планарной гомографии. Суть подхода состоит в следующем. На изображениях ищутся характеристические точки с помощью предложенной в работе модификации детектора Харриса [6], предназначенной для работы по пирамиде детальности, т.е в пространстве переменной детальности (Scale Space) [7]. Для каждой точки вычисляется вектор параметров длины 7, основанных на дифференциальных инвариантах [8,9]. Далее ищется гомография между изображениями с помощью метода RANSAC [10] по парам вероятных соответствий между характеристическим точками на разных изображениях, установленных на основе минимальных расстояний Махаланобиса. Особенностью подхода [4] является применение RANSAC в условиях высокого процента ложных соответствий, зачастую существенно превышающем 50%. В статье [4] результаты проиллюстрированы на паре изображений горной вершины, снятых с различным зумом и при фиксированном положении оптического центра камеры. Хотя в работе это не оговаривается, представляется, что алгоритм будет испытывать трудности, при совмещении изображений искусственных сцен, так как выделяемые особенности будут иметь схожие векторы параметров, например окна в многоэтажном доме, и процент ложных соответствий будет чрезмерно высок.

Задача совмещения изображений тесно связана с проблемами построения мозаик [11], сверхразрешения [12] и восстановления трехмерных сцен [13-21]. Первые две задачи как правило отвечают перечисленным трем ситуациям, когда существует планарная гомография. В

работе [11] рассматривается задача построения мозаики из видеопоследовательности изображений, снятых с помощью камеры произвольно вращающейся относительно неподвижного центра проекции, фокусное расстояние считается постоянным и известным, в составе сцены присутствуют движущиеся объекты. Для пар кадров значительно перекрывающихся по полю зрения вычисляются параметры (сдвиг, поворот) совмещающие изображения путем преобразования Фурье-Меллина аналогично [2]. Таким кадрам соответствует малый поворот камеры относительно оптического центра. В этом приближении оценивается матрица гомографии между изображениями. Свойство конкатенации гомографий [1] (гомография между кадром 0 и N равна произведению матриц междукадровых гомографий), позволяет вычислить матрицы гомографий для любой пары кадров, правда с накоплением ошибки. При этом обнаруживаются перекрывающиеся кадры далеко отстоящие в видеопоследовательности (при повторном проходе камерой области сцены), для таких пар тоже выполняется поиск параметров совмещения методом [2]. Выбирая один из кадров последовательности в качестве базового, матрицы гомографии по отношению к остальным кадрам вычисляются как решение переопределенной системы линейных уравнений, минимизирующее отклонения от оцененных ранее попарных гомографий. Окончательная мозаика строится на базе этих гомографий, выбирая в качестве данных для результирующего изображения фрагменты отдельных кадров, содержащие движущийся объект целиком (или только фон), такая сегментация на фрагменты достигается проведением границ по наименее ярким областям разностных изображений с помощью алгоритма кратчайшего пути Дейкстры [22]. В работе отмечается, что по опыту автора метод [2] сохраняет устойчивость при смещениях до 1/3 размера кадра и повороте до 45°. Выбор метода [2] в противовес методам, основанным на характеристическим точкам, подобным [4] и локальным методам типа [3] мотивируется тем, что наиболее заметные такие особенности обычно возникают на границах движущихся объектов, что приводит к смещенной оценке параметров, интенсивное использование робастных методов на основе RANSAC [10] по-видимому не рассматривалось из за высокой вычислительной сложности, при необходимости построения мозаик по последовательности из сотен изображений. В контексте настоящей работы отметим, что подход [11] основывается на предположении, что в видеопоследовательности последовательные кадры мало отличаются по ориентации камеры и сильно перекрываются. В случае, когда изображения получены не видео, а фотокамерой, подобной априорной информации нет и предлагаемый алгоритм грубого совмещения позволяет выявить сильно перекрывающиеся кадры и расширить алгоритм на набор фото изображений. Забегая вперед отметим, что указанная в [11] проблема обнаружения характеристических особенностей на границах движущихся объектов (что эквивалентно линиям разрыва карты глубин сцены) в настоящей работе частично преодолевается путем характеризации линии с помощью методов нечеткой логики (см. раздел 4.4).

В работе [12] рассматриваются сцены не содержащие движущихся объектов. Для совмещения изображений выбран подход аналогичный [4]. Выбор мотивируется тем, что методы на основе характеристических точек мало чувствительны к изменениям условий освещения, могут обрабатывать изображения, сильно отличающиеся по ориентации камеры, не имеют ограничений на поиск полного набора параметров гомографий и позволяют построить статистически хорошо обоснованную оценку параметров гомографий на основе метода максимального правдоподобия. Кроме того, поиск гомографий может быть выполнен одновременно и согласованно для всех изображений набора. Алгоритм также как и [11] применяется к видеопоследовательности. Попарное совмещение изображений состоит в поиске характеристических точек с помощью [6,4,7], вычислении набора предполагаемых соответствий между точками двух кадров на основе пространственной близости и похожести яркости в окрестности в смысле нормализованной корреляции. Далее производится оценка гомографии методами типа RANSAC по выборкам из 4 точек. Выбирается гомография, с заданной погрешностью правильно описывающая наибольшее количество соответствий характеристических точек. При известной гомографии, производится дальнейших поиск характеристических точек и со-

ответствий в вычисленных окрестностях точек на втором кадре. Гомография уточняется по всем соответствиям классифицированным как верные. Последние два шага повторяются итеративно до стабилизации числа соответствий. Согласованное совмещение всех кадров достигается как и в [11] с использованием свойств конкатенации гомографий, однако отмеченная ранее способность подхода на основе характеристических точек обрабатывать сильно разнесенные кадры заключается в том, что в статистической оценке достоверности совмещения изображений участвует каждая характеристическая точка на всех кадрах, где она оказывается в поле зрения, а не только значительно перекрывающиеся кадры. В контексте настоящей работы отметим, что подход [12] тоже эксплуатирует «близость» изображений в последовательных кадрах таким образом привязан к обработке видео. Предлагаемая в настоящей работе методика позволяет расширить этот подход на случай набора изображений снятых фотоаппаратом, что для задачи сверхразрешения представляется особенно важным, так как пространственное разрешение цифровых фотоаппаратов существенно выше, чем у типовых видео камер и не имеет артефактов типа интерлэйсинга и смаза.

Задачи восстановления трехмерных сцен можно подразделить на методы, дающие плотное восстановление, что наиболее часто встречается в стандартном стерео [13-15] и методы основанные на характеристических особенностях [16-21], что более характерно для задач восстановления формы и движения (shape from motion) и стерео с широкой базой. В задачах стандартного стерео обычно предполагается, что оптические оси двух камер и их горизонталь строго параллельны, так что эпиполярные линии направлены вдоль строк изображения а соответствующие эпиполярные линии находятся в одинаковых номерах строк левого и правого кадра, эпиполюс находится слева [1], так называемое выравненное стерео (Rectified stereo). Если это не так, то изложенная в [23] методика позволяет свести задачу к такой ситуации. В задачах плотного стерео для каждой точки одного изображения соответствующая точка другого изображения ищется в строке с тем же номером в заданном диапазоне смещений, что дает карту смещений (disparity map), которая при известной базе легко может быть пересчитана в расстояния в трехмерном пространстве из элементарных геометрических соображений. Некоторые алгоритмы [13] могут искать двумерную карту смещений, что применимо в случае когда выравнивание выполнено не вполне точно. Однако в любом случае необходимо задать примерные координаты, где искать соответствующую точку на втором изображении и диапазон поиска, причем заниженный диапазон приводит к неверным результатам, а завышенный существенно замедляет работу алгоритма и также может приводить к неверным результатам за счет ложных соответствий. Процедура выравнивания [23] требует оценки фундаментальной матрицы, что требует задания минимум 7 точных соответствий между характеристическими точками на двух изображениях. Однако, как отмечается в [24,25], точность линейных методов невысока, наилучшие результаты, особенно в присутствии ложных соответствий, что неизбежно при автоматической обработке, дают робастные методы на базе RANSAC. Таким образом, желательно наличие порядка или больше сотни соответствий, при по возможности низком проценте ложных соответствий. Аналогичная информация (характеристические точки и соответствия между ними) требуется и для алгоритмов стерео с широкой базой [21] и восстановления формы и движения, например методов факторизации [16-20]. Отметим, что для указанных алгоритмов желательно иметь как можно больше характеристических точек и правильных соответствий между ними, иначе даже при высококачественном восстановлении трехмерных координат этих точек, построенный по ним меш не будет адекватно представлять форму объекта. Тем не менее эти подходы в практическом смысле более перспективны, чем классическое стерео, так как большая база съемки и/или значительное количество кадров обеспечивают существенно более высокую точность восстановления и проводят к адекватному восстановлению формы. Кроме того, исходные изображения для этих алгоритмов могут быть получены с помощью типовых, широко распространенных цифровых фотоаппаратов и видеокамер, в то время, как аппаратура для стерео съемки практически не представлена на широком рынке, за исключением полукустарных устройств соединяющих на одной штанге

два отдельных фотоаппарата и стереообъектива фирмы Loreo (www.loreo.com), для которых, из-за конструктивных особенностей, стабильность калибровки стерео сомнительна. Алгоритмы [16-20], кроме того, требуют, чтобы каждая характеристическая точка была прослежена по всей последовательности кадров, что практически приводит к необходимости отбрасывать точки не удовлетворяющие этому условию. Таким образом, задачи восстановления трехмерных сцен на основе характеристических особенностей предъявляют весьма жесткие требования к алгоритмам обнаружения характеристических точек и установления между ними взаимнооднозначных соответствий. Как отмечалось выше, наиболее продвинутые алгоритмы обнаружения характеристических точек основаны детекторе Харриса [6,4] в пространстве переменных разрешений [7] и использовании дифференциальных инвариантов [8,9]. Однако этого недостаточно для достоверного обнаружения достаточного количества таких точек и междукадровых соответствий. В задачах восстановления трехмерной сцены сопоставление этих особенностей также осложняется тем, что особые точки трехмерных объектов наблюдаются с существенно различных положений и ориентаций камеры. В работе [26] предлагается адаптировать для каждой характеристической точки на каждом кадре форму окна, по которому вычисляются дифференциальные инварианты. Для обнаружения характеристических точек используется детектор Харриса [7,4], однако в отличие от [27,28] обнаружение точек идет независимо на всех разрешениях, выбирается предопределенное число точек дающих максимальное значение отклика в детекторе Харриса и все они участвуют в поиске соответствий. Мотивируется это тем, что при прохождении по переменному разрешению для особенностей типа уголков может отсутствовать явно выраженный максимум. Рассматриваются все возможные пары соответствий, в качестве меры похожести векторов параметров выбрано расстояние Мехалонобиса, вводится мера неоднозначности соответствия как отношение расстояния между соответствующей парой и кратчайшего расстояния от одного из сравниваемых векторов к ближайшему не соответствующему. Высокий процент верных соответствий достигается путем отбрасывания сомнительных. В [27] этот подход получил дальнейшее развитие. Предложен итеративный алгоритм, который одновременно адаптирует положение, масштаб и форму окрестности для детектирования и описания дифференциальными инвариантами характеристических точек аффинно-инвариантным образом. Использованы дифференциальные инварианты вплоть до четвертого порядка, инвариантность относительно изменения условий освещения и наблюдения достигается тем, что в результирующий вектор параметров из 12 компонент входят отношения дифференциальных инвариантов высших порядков к первому. В заключении используется робастная оценка на основе RANSAC для отсева ложных соответствий при преобразовании описываемом гомографией или фундаментальной матрицей. Заметим, что при съемке искусственных структур, многие хорошо выделяемые особенности будут иметь схожий вектор инвариантов. В [29] рассматриваются цветные изображения, дифференциальные инварианты дополнены компонентами, представляющими цветовую структуру в окрестностях характеристической точки, что положительно сказывается на достоверности соответствий. Предложен инкрементальный алгоритм ограниченного поиска соответствий. Его основная идея состоит в том, чтобы максимально ограничить диапазон поиска соответствующей точки по плоскости изображения. Если известен некоторый достоверный набор соответствий, то выполняется триангуляция Делоне на одном из изображений и переносится на второе. Таким образом, на двух изображениях построены соответствующие треугольники. Если также известна фундаментальная матрица, то для каждой характеристической точки одного изображения, соответствие ищется на втором в пределах соответствующего треугольника и эпиполярной линии (по видимому, в пределах полосы неопределенности около эпиполярной линии [30]). После каждой найденной точки триангуляция дополняется, фундаментальная матрица также уточняется. В качестве меры различия между двумя характеристическим векторам используется Евклидово расстояние, однако предварительно компоненты вектора преобразуются так, чтобы диапазон изменения каждой компоненты был одинаков. В работе отмечается, что проблемой предложенного алгоритма является отсутствие пространственных

ограничений на область поиска при инициализации алгоритма. Эту проблему предлагается решать выбирая в начале алгоритма наиболее сильные характеристические точки и наиболее близкие соответствия. Фактически, алгоритм [29] это эффективный метод доведения количества парных соответствий до желаемого количества, когда изначально установлены надежные соответствия и эпиполярная геометрия, т.е для получения адекватного меша, как упоминалось ранее.

Таким образом, из анализа литературы можно сделать вывод, что задача предварительного ограничения района поиска соответствия в плоскости изображения весьма актуальна, особенно при обработке изображений, полученных не как видеопоследовательность, а как набор снимков цифровым фотоаппаратом с рук. Предлагаемый в настоящей работе алгоритм позволяет решить эту проблему в случае сцен содержащих искусственные объекты, т.е прямые линии. В этой связи следует упомянуть подход [31], где вместо характеристических точек (уголков), обнаруживаемых детектором Харриса используются небольшие области (сегменты) на изображении, резко отличающиеся своими цвето-яркостными характеристиками от ближайшего окружения. Заметим, что таких объектов на изображении может быть недостаточно много для полного восстановления трехмерной сцены, но в этом случае на их основе можно выполнить предварительное грубое совмещение изображений аналогично предлагаемому в настоящей работе.

2. ПОИСК ПРЯМЫХ ЛИНИЙ НА ИЗОБРАЖЕНИИ

Распространенным способом поиска параметрических кривых на изображениях является преобразование Хафа [32]. Его суть в том, что из множества точек изображения выбираются подмножества точек, по которым вычисляются параметры кривой, проходящей через эти точки. С каждой точкой параметрического пространства связан счетчик, показывающий для скольких подмножеств был вычислен этот набор параметров. Параметры с максимальным значением счетчика определяют кривые, найденные на изображении.

Число подмножеств, которое необходимо перебрать экспоненциально возрастает с числом параметров кривой, что приводит к большому количеству вычислений. В связи с этим разработано множество приближенных способов вычисления преобразования Хафа, основанных на переборе не всех подмножеств точек, а только некоторой их части [44], [45], [46].

Прямая линия на изображении задается двумя параметрами, которые могут быть определены для каждой пары точек. Для изображения из N точек число всевозможных пар точек составляет $N(N-1)/2$. Полный перебор всех пар точек имеет сложность $O(N^2)$, что неприемлемо для типичных изображений ($N \sim 10^6$).

В случае прямых линий преобразование Хафа эквивалентно преобразованию Радона [33]. Подробнее о преобразовании Радона см. в разделе 2.4. Для преобразования Радона разработаны методы вычисления со сложностью $O(N \log N)$ [34,35].

В [34] предложен алгоритм вычисления преобразования Радона через двумерное преобразование Фурье. В зависимости от реализации алгоритм предъявляет различные требования к памяти $kN, \quad k = 2 \div 5$.

Ранее авторами был предложен алгоритм [35] вычисления преобразования Радона через двумерное преобразование Хартли. Требования к памяти в два раза меньше, чем в [34]. Преимущество преобразования Хартли перед Фурье в том, что первое является действительным, и требует для своего вычисления меньшего объема памяти и, как следствие обработки меньшего числа данных, меньшего времени вычисления.

В работах [36,37] приведён алгоритм вычисления интегралов от изображения вдоль прямых линий. Это может быть использовано для вычисления преобразования Радона изо-

бражения. Для изображения из N точек вычисляются $O(N\log N)$ интегралов вдоль 'базисных' прямых, затем показывается, что интеграл вдоль любой прямой может быть выражен через сумму не более чем $O(\log N)$ интегралов вдоль 'базисных' прямых. В процессе работы алгоритма строится граф, содержащий N вершин и $16N\left(\frac{\log N}{2}+1\right)$ ребер, что предъявляет чрезмерные требования к памяти (на порядок больше, чем в [34,35]).

В настоящей работе используется алгоритм вычисления преобразования Радона через преобразование Хартли на основе [35], доработанный вариант которого приводится в разделах 2.4, 4.3

2.1 Преобразование Хартли

Напомним, некоторые свойства преобразования Хартли [38].

Определяется функция cas:

$$\mathrm{cas}(x) \equiv \sin(\mathrm{x}) + \cos(\mathrm{x}) = \sqrt{2}\sin(x+\frac{\pi}{4}) \tag{1}$$

Преобразование Хартли $H(\vec{\xi})$ функции $f(\vec{\mathbf{x}})$, где $\vec{\mathbf{x}},\vec{\xi} \in R^n$ очень похоже на преобразование Фурье и определяется следующим образом [38]:

$$H(\vec{\xi}) = \int f(\vec{\mathbf{x}})\mathrm{cas}(2\pi\vec{\mathbf{x}}^T\vec{\xi})\mathrm{d}\vec{\mathbf{x}} \tag{2}$$

Прямое и обратное преобразования Хартли совпадают:

$$f(\vec{\mathbf{x}}) = \int H(\vec{\xi})\mathrm{cas}(2\pi\vec{\mathbf{x}}^T\vec{\xi})\mathrm{d}\vec{\xi} \tag{3}$$

Преобразование Хартли может быть выражено через преобразование Фурье следующим образом:

$$H(\vec{\xi}) = \mathrm{Re}(F(\vec{\xi})) + \mathrm{Im}(F(\vec{\xi})) \tag{4}$$

где $F(\vec{\xi})$ есть преобразование Фурье функции $f(\vec{\mathbf{x}})$:

$$F(\vec{\xi}) = \int f(\vec{\mathbf{x}})\exp(2\pi i\vec{\mathbf{x}}^T\vec{\xi})\mathrm{d}\vec{\mathbf{x}} \tag{5}$$

Обратное преобразование Фурье:

$$f(\vec{\mathbf{x}}) = \int F(\vec{\xi})\exp(-2\pi i\vec{\mathbf{x}}^T\vec{\xi})\mathrm{d}\vec{\xi}$$

Знак суммы в (4) зависит от знака в экспоненте в (5). Определение (5) отличается от [38] и соответствует [39], так как для сравнения производительности преобразования Хартли и Фурье использовалась реализация быстрого преобразования Фурье из [39].

Одномерное дискретное преобразование Хартли (ДПХ) вектора $\vec{\mathbf{f}} = (f_0,\cdots f_{N-1})^T$ есть вектор $\vec{\mathbf{h}} = (h_0,\cdots h_{N-1})^T$ [38]:

$$\begin{aligned} h_i &= \frac{1}{N}\sum_{j=0}^{N-1} f_j \mathrm{cas}\frac{2\pi ij}{N}, \quad i = 0\ldots N-1 \\ f_j &= \sum_{i=0}^{N-1} j_i \mathrm{cas}\frac{2\pi ij}{N}, \quad j = 0\ldots N-1 \end{aligned} \tag{6}$$

Алгоритм вычисления одномерного преобразования Хартли приведен в [38]. Он аналогичен алгоритму быстрого преобразования Фурье (БПФ) и имеет сложность $O(N\log N)$.

Согласно [38] двумерное ДПХ квадратной $N \times N$ матрицы $\mathbf{F}$ есть квадратная $N \times N$ матрица $\mathbf{H}$:

$$H_{i,j} = \sum_{k=0}^{N-1}\sum_{l=0}^{N-1} F_{k,l}\,\mathrm{cas}\frac{2\pi(ik+jl)}{N}, \quad i,j = 0\ldots N-1 \tag{7}$$

В отличие преобразования Фурье ядро преобразования Хартли не сепарабельно ($\mathrm{cas}\frac{2\pi(ik+jl)}{N} \neq \mathrm{cas}\frac{2\pi ik}{N}\mathrm{cas}\frac{2\pi jl}{N}$). Ниже приведены выкладки [47], согласно которым двумерное преобразование Хартли получается одномерным преобразованием Хартли строк, затем столбцов (10) и последующим линейным преобразованием элементов (9). Из (1) пользуясь свойствами четности функций sin() и cos() легко получить:

$$\mathrm{cas}(a+b) = \frac{1}{2}\left[\mathrm{cas}a\mathrm{cas}b + \mathrm{cas}(-a)\mathrm{cas}b + \mathrm{cas}a\mathrm{cas}(-b) - \mathrm{cas}(-a)\mathrm{cas}(-b)\right] \tag{8}$$

Используя (8), формула (7) может быть переписана в виде

$$\begin{aligned} &H_{i,j} = \frac{1}{4}\left[H'_{i,j} + H'_{p,j} + H'_{i,q} - H'_{p,q}\right] \\ &p = (N-i) \bmod N, \quad q = (N-j) \bmod N \end{aligned} \tag{9}$$

$$H'_{i,j} = \sum_{k=0}^{N-1}\mathrm{cas}\frac{2\pi ik}{N}\sum_{l=0}^{N-1} F_{k,l}\,\mathrm{cas}\frac{2\pi jl}{N} \tag{10}$$

Запись $a \bmod b$ означает вычисление остатка от деления a на b.

В формуле (7) постоянной составляющей сигнала (нулевой частотой) является элемент $H_{0,0}$ (левый верхний угол). В дальнейшем потребуется переместить нулевую частоту в центр изображения. Для этого необходимо переставить квадранты исходного изображения $\mathbf{F}$ и квадранты $\mathbf{H}$ (рис. 1а):

$$\begin{aligned} &H^0_{i,j} = H_{r,s} \\ &r = (i + \frac{N}{2}) \bmod N, \quad s = (j + \frac{N}{2}) \bmod N \end{aligned} \tag{11}$$

Преобразование Хартли с нулевой частотой в центре обозначается через $H^0_{i,j}$.

Перестановку (11) квадрантов и преобразование (9) можно объединить (рис. 1б):

$$\begin{aligned} &H^0_{i,j} = \frac{H'_{r,s} + H'_{p,s} + H'_{r,q} - H'_{p,q}}{4} \\ &p = (\frac{N}{2} - i) \bmod N, \quad q = (\frac{N}{2} - j) \bmod N \\ &r = (\frac{N}{2} + i) \bmod N, \quad s = (\frac{N}{2} + j) \bmod N \end{aligned} \tag{12}$$

В случае, когда нет ограничений по памяти и результат можно записывать в новый массив, способ (12) экономит количество операций, для вычислений inplace – целесообразно использовать (9),(10) с последующей перестановкой квадрантов. ДПХ столбцов осуществляется путем транспонирования изображения, вычислением ДПХ строк и транспонированием. Это обусловлено тем, что вычисление ДПХ требует $\log N$ раз обратиться к каждому элементу

изображения; в текущей реализации изображение хранится построчно, и за счет более компактного расположения данных доступ к элементам строки быстрее, чем к элементам столбца.

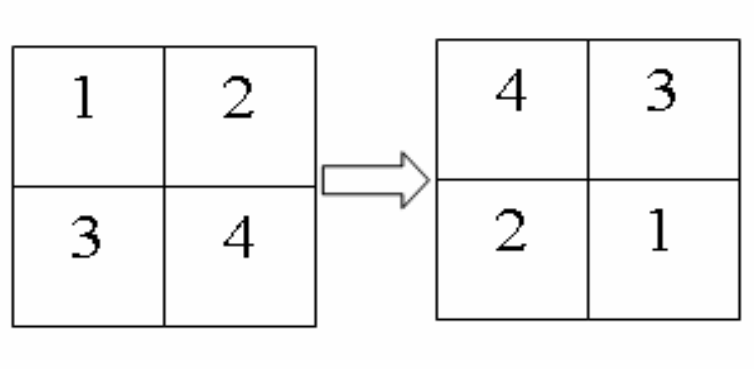

Рис. 1а. Перестановка квадрантов

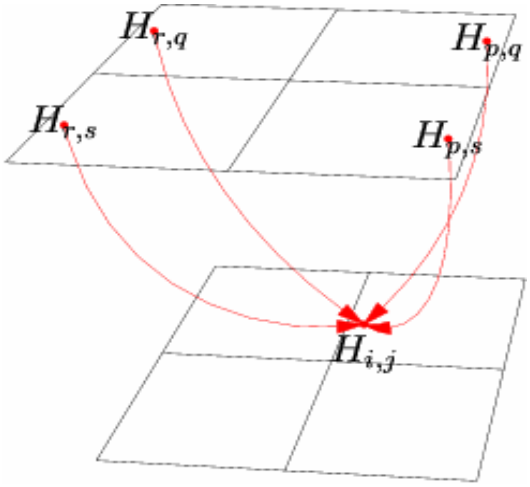

Рис. 1б. Преобразование элементов согласно (12)

Алгоритм 1. Вычисление двумерного ДПХ матрицы $\mathbf{F}$ (inplace).

1. *Переставить квадранты исходной матрицы (рис. 1а);*
2. *Выполнить одномерные ДПХ строк матрицы* $\mathbf{F}$ *(inplace);*
3. *Транспонировать матрицу (inplace);*
4. *Выполнить (inplace) одномерные ДПХ строк (т.е. бывших столбцов);*
5. *Транспонировать матрицу (inplace); Обозначить полученную матрицу через* $\mathbf{H}'$*;*
6. *Преобразовать элементы* $\mathbf{H}'$ *согласно (9),(10), Результат* $\mathbf{H}^0$ *есть двумерное преобразование Хартли матрицы* $\mathbf{F}$ *с нулевой частотой в элементе* $(N/2, N/2)$.

2.2 Фильтрация изображения

Перед началом работы с изображениями выполнялась калибровка камеры.

Фильтрация изображения состоит в получении контурного препарата.

Калибровка необходима, чтобы устранить искажения камеры (бочку). Для калибровки использовались результаты работы [40].

Радиальные искажения описываются формулой:

$$\begin{aligned} \breve{x} &= x + (x - u_0)(k_1 r^2 + k_2 r^4) \\ \breve{y} &= y + (y - v_0)(k_1 r^2 + k_2 r^4), \\ r^2 &= (x - u_0)^2 + (y - v_0)^2 \end{aligned} \qquad (13)$$

где x, y координаты точки на исходном изображении (с бочкой), а $\breve{x}, \breve{y}$ калиброванные координаты. Перед началом работы были найдены параметры u_0, v_0, k_1, k_2 и в дальнейшем изображения подвергались корректирующему преобразованию.

Для получения контурного препарата использовался алгоритм модуля градиента [41]. Для каждой точки x, y изображения строится матрица $\mathbf{B}$ размером 2×2:

$$\mathbf{B}(x,y) = \sum_i \vec{\mathbf{g}}_i \vec{\mathbf{g}}_i^T, \quad \vec{\mathbf{g}}_i^T = \left(\frac{\partial I_i(x,y)}{\partial x} \quad \frac{\partial I_i(x,y)}{\partial y} \right) \qquad (14)$$

Суммирование производится по цветовым плоскостям $I_i(x,y)$ изображения (одна для серого изображения и три для цветного RGB). Производные изображения вычисляются путем свертки изображения с производной функции Гаусса полушириной χ=2÷3 пикселя. Разность максимального и минимального собственных значений матрицы $\mathbf{B}$ является интенсивностью контурного препарата $C(x,y)$ в точке x,y:

$$C(x,y) = \lambda_1(\mathbf{B}(x,y)) - \lambda_2(\mathbf{B}(x,y)), \quad \mathbf{B}\vec{\mathbf{h}}_i = \lambda_i\vec{\mathbf{h}}_i, \quad \lambda_1 \geq \lambda_2 \tag{15}$$

Выбор $C(x,y)$ в виде (15) оставляет только одномерные особенности. Двумерные особенности типа уголков и пересечений линий соответствуют малым $C(x,y)$.

2.3 Параметризация прямой линии

В настоящей работе используется две параметризации прямой линии на изображении.

2.3.1 ρ,φ -параметризация

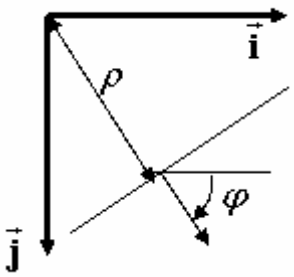

Рис. 2. Параметризация прямой на изображении

Прямая задается двумя параметрами: ρ и φ, где ρ есть расстояние от начала координат до прямой, а φ есть угол между нормалью к прямой и осью абсцисс (рис. 2). Для удобства расстояние ρ считается знаковой величиной. Угол φ лежит в пределах $[0,\pi)$. При таких диапазонах параметров каждой прямой на изображении соответствует единственный набор параметров. Знак ρ выбирается таким, чтобы уравнение прямой имело вид:

$$x\cos\varphi + y\sin\varphi = \rho \tag{16}$$

Расстояние ρ и координаты x,y измеряются в пикселях.

2.3.2 $\vec{\mathbf{m}}$ -параметризация

Прямая задается единичным вектором $\vec{\mathbf{m}}$ в трехмерном пространстве, перпендикулярным к плоскости, проходящей через фокус камеры и прямую на изображении (рис. 3). Связь между вектором $\vec{\mathbf{m}}$ и введенными выше параметрами ρ и φ следующая:

$$\vec{\mathbf{m}} = \frac{w}{\sqrt{w^2+\rho^2}}\left(\cos\varphi \quad \sin\varphi \quad -\frac{\rho}{w}\right)^T, \quad |\vec{\mathbf{m}}| = 1 \tag{17}$$

где w есть угловая разрешающая способность камеры, которой получено изображение и имеет размерность пиксель/радиан.

2.4 Быстрое преобразование Хафа

Для вещественных данных (в том числе изображений) преобразование Хартли вычислительно более эффективно. В настоящем разделе показывается, что все выкладки из [34] могут быть полностью перенесены на преобразование Хартли, что приводит к уменьшению требуемой памяти в два раза и уменьшению времени работы примерно в 6 раз и не приводит ни к каким трудностям или потерям.

Рассмотрим серое изображение $I(x,y)$ (контурный препарат). Как сказано выше, преобразование Хафа для прямых линий эквивалентно преобразованию Радона. Преобразование Радона изображения $I(x,y)$ есть интегральное преобразование следующего вида [33]:

$$R(\rho,\varphi) = \iint I(x,y)\delta(\rho - x\cos\varphi - y\sin\varphi)dxdy \tag{18}$$

$$\int \delta(x)dx = 1 \tag{19}$$

где $\delta(x)$ есть δ-функция Дирака. $R(\rho,\varphi)$ есть интеграл от изображения $I(x,y)$ вдоль прямой с параметрами ρ,φ. Следовательно, если на изображении имеется прямая с параметрами ρ,φ, то величина $R(\rho,\varphi)$ будет большой. В [33] показано, что преобразование Хафа для прямых линий эквивалентно преобразованию Радона.

Пусть $\vec{\mathbf{x}},\vec{\boldsymbol{\xi}} \in R^2$. Для компактности формул введем линейные операторы:

$$\hat{A} f(\vec{\mathbf{x}}) = g(\vec{\boldsymbol{\xi}}) \tag{20}$$

Введем следующие операторы:

Оператор одномерного преобразования Хартли по первой переменной:
$$\left(\hat{H}^{(1)} f(\vec{\mathbf{x}})\right)(\xi_1, x_2) = \int f(\vec{\mathbf{x}})\mathrm{cas}(2\pi x_1\xi_1)dx_1 \tag{21}$$

Оператор двумерного преобразования Хартли:
$$\left(\hat{H}^{(2)} f(\vec{\mathbf{x}})\right)(\vec{\boldsymbol{\xi}}) = \int f(\vec{\mathbf{x}})\mathrm{cas}(2\pi\vec{\boldsymbol{\xi}}^T\vec{\mathbf{x}})d\vec{\mathbf{x}} \tag{22}$$

Оператор перехода к полярным координатам:
$$\left(\hat{P} f(\vec{\mathbf{x}})\right)(r,\varphi) = f(r\cos\varphi, r\sin\varphi) \tag{23}$$

Оператор преобразования Радона:
$$\left(\hat{R} f(\vec{\mathbf{x}})\right)(\rho,\varphi) = \int f(\vec{\mathbf{x}})\delta(\rho - \vec{\mathbf{x}}^T\vec{\mathbf{k}})d\vec{\mathbf{x}}, \quad \vec{\mathbf{k}} \equiv (\cos\varphi,\sin\varphi)^T \tag{24}$$

Теорема 1.

Преобразование Радона функции $f(\vec{\mathbf{x}})$ может быть выражено через преобразование Хартли следующим образом:

$$\left(\hat{R} f(\vec{\mathbf{x}})\right)(\rho,\varphi) = \left(\hat{H}^{(1)}\hat{P}\hat{H}^{(2)} f(\vec{\mathbf{x}})\right)(\rho,\varphi), \tag{25}$$

Доказательство. Учитывая (3), применим оператор $\hat{H}^{(1)}$ к обеим частям (25) :

$$\left(\hat{H}^{(1)}\hat{R} f(\vec{\mathbf{x}})\right)(r,\varphi) = \left(\hat{P}\hat{H}^{(2)} f(\vec{\mathbf{x}})\right)(r,\varphi) \tag{26}$$

Воспользовавшись (19),(24),(21) перепишем левую часть (26) в виде:

$$\left(\hat{H}^{(1)}\hat{R} f(\vec{\mathbf{x}})\right)(r,\varphi) = \int f(\vec{\mathbf{x}})\delta(\rho - \vec{\mathbf{x}}^T\vec{\mathbf{k}})\mathrm{cas}(2\pi\rho r)d\vec{\mathbf{x}}d\rho = \int f(\vec{\mathbf{x}})\mathrm{cas}(2\pi r\vec{\mathbf{x}}^T\vec{\mathbf{k}})d\vec{\mathbf{x}} \tag{27}$$

Раскроем правую часть (26):

$$\left(\hat{P}\hat{H}^{(2)} f(\vec{\mathbf{x}})\right)(r,\varphi) = \hat{P}\left(\int f(\vec{\mathbf{x}})\mathrm{cas}(2\pi\vec{\mathbf{x}}^{\mathrm{T}}\vec{\boldsymbol{\xi}}))d\vec{\mathbf{x}}\right)(\vec{\boldsymbol{\xi}}) = \int f(\vec{\mathbf{x}})\mathrm{cas}(2\pi r\vec{\mathrm{x}}^{\mathrm{T}}\vec{\mathbf{k}}))d\vec{\mathbf{x}} \tag{28}$$

Таким образом, (27) и (28) совпадают ■.

Согласно (25) алгоритм вычисления преобразования Радона изображения $I(x,y)$ следующий:

Алгоритм 2. Быстрое преобразование Хафа через преобразование Хартли.

1. *Выполнить двумерное ДПХ* $H(x,y)$ *изображения* $I(x,y)$ *(см.* ***Алгоритм 1****).*
2. *Представить изображение* $H(x,y)$ *в полярных координатах, обозначив полученное изображение через* $P(r,\varphi)$. *Интенсивность пикселя* (r,φ) *изображения* $P(r,\varphi)$ *равна интенсивности изображения* $H(x,y)$ *в точке* $(r\cos\varphi, r\sin\varphi)$ *(использовать билинейную интерполяцию).*
3. *Вычислить одномерное ДПХ каждой строки изображения* $P(r,\varphi)$. *Согласно Теореме 1, полученное изображение* $R(\rho,\varphi)$ *есть преобразование Радона изображения* $I(x,y)$.

ДПХ требует для простейшей реализации дополнение размеров изображения до степени двойки (дополнение нулями). Для повышения точности определения параметров линий и подавления эффекта Гиббса можно выполнять дополнение нулями до следующей степени двойки, как было предложено в [34]. На рис.3 для тестового изображения (рис. 3а) приведены результаты вычисления преобразования Хафа (Радона), называемого ***синограммой***. На рис.3г. приведена синограмма, вычисленная для серого изображения (рис. 3б), а на рис. 3ж – синограмма вычисленная по контурному препарату (рис. 3в), полученному согласно алгоритму, описанному в разделе 2.2. Ось ρ направлена вправо, ось φ направлена вниз. Начало координат находится в середине самой верхней строки. Прямым линиям на контурном препарате соответствуют яркие точки на синограмме. Затем синограммы подвергаются фильтрации (рис. 3д,з), описанной ниже. Найденные прямые линии приведены на рис. 3е,и.

Поиск максимумов на синограммах рис. 3г,ж затруднителен в силу наличия ярких областей (не точек), не соответствующих прямым линиям. Применив фильтр, имеющий большой отклик на яркие точки, можно добиться их усиления и ослабления ярких областей. Фильтрация осуществляется путем сверки строк синограммы со второй производной функции Гаусса, полуширина которой та же, что и при вычислении производных в контурном препарате (см. раздел 2.2) . Результат применения фильтрации приведен на рис. 3д,з. Можно видеть, что яркие области (не точки) значительно ослабились.

Для второй производной функции Гаусса

$$L(x) = \frac{d^2}{dx^2}\frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{x^2}{2\sigma^2}) \tag{29}$$

преобразование Хартли имеет вид:

$$(\hat{H}^{(1)} L(x))(\xi) = \xi^2 \exp(-\frac{\sigma^2\xi^2}{2}) \tag{30}$$

Справедлива теорема [38], согласно которой преобразование Хартли свертки функций одна из которых обладает определенной четностью (четной или нечетной), равно произведению их преобразований Хартли (аналогично преобразованию Фурье). Так как $L(x)$ является четной, то с учетом фильтрации (25) переписывается в виде:

$$\left(\hat{R} f(\vec{\mathbf{x}})\right)(\rho,\varphi) = \left(\hat{H}^{(1)} \hat{F}_L \hat{P} \hat{H}^{(2)} f(\vec{\mathbf{x}})\right)(\rho,\varphi), \tag{31}$$

Оператор $\hat{F}_L$ выполняет умножение строк изображения на преобразование Хартли функции $L(x)$.

Из (25) легко показать, что свертка строк синограммы с функцией $u_1(x)$ соответствует свертке исходного изображения с центрально-симметричной функцией $u_2(x,y) = u_1(\sqrt{x^2 + y^2})$.

Рис. 3а. Исходное изображение (1224x1632)

Рис. 3б. Яркость исходного изображения (1224x1632)

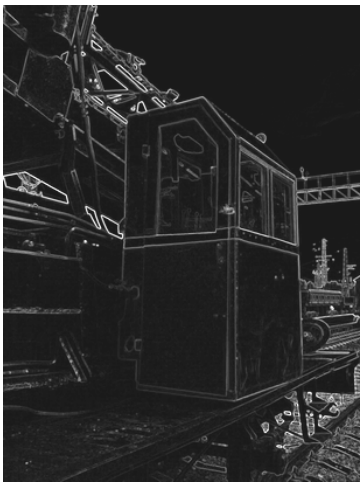

Рис. 3в. Контурный препарат (1224x1632)

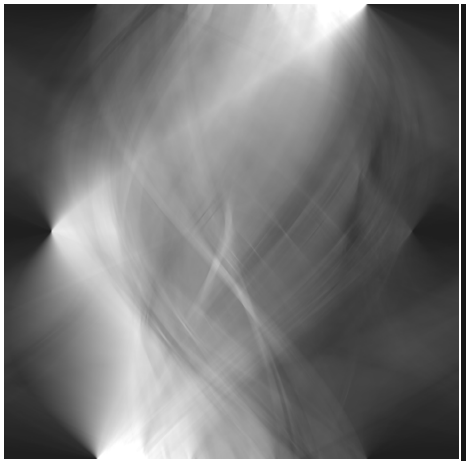

Рис. 3г. Синограмма рис. 3б (2048x2048)

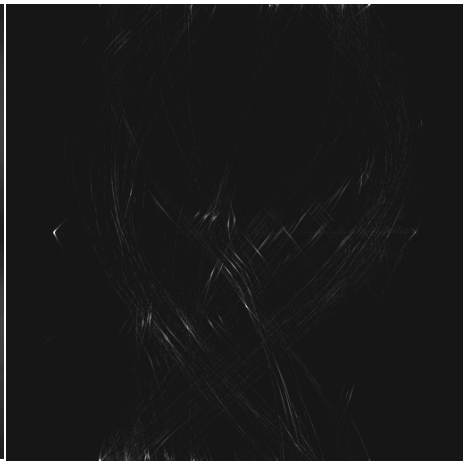

Рис. 3д. Синограмма рис. 3б (2048x2048) после свертки строк со второй производной функции Гаусса

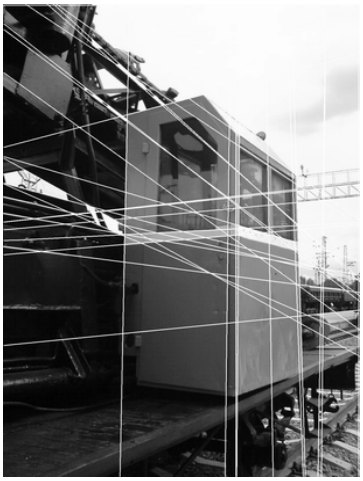

Рис. 3е. Найденные прямые лини на рис. 3б.

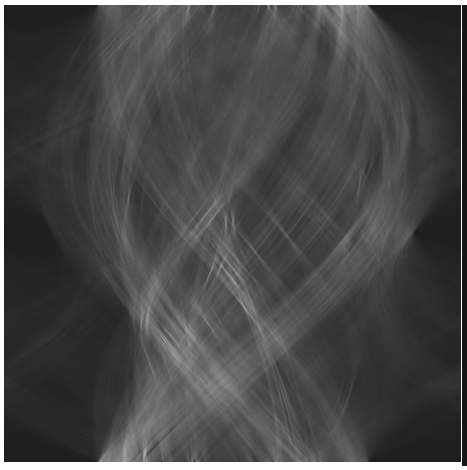

Рис. 3ж. Синограмма рис. 3ж (2048x2048)

Рис. 3з. Синограмма рис. 3з (2048x2048) после свертки строк со второй производной функции Гаусса

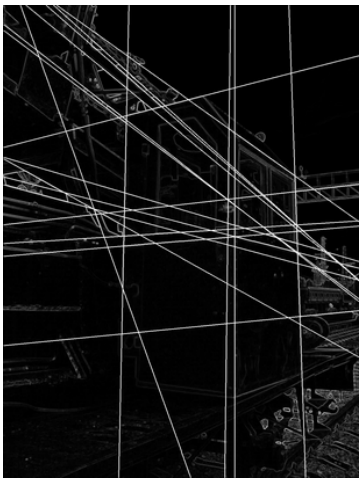

Рис. 3и. Найденные прямые лини на рис. 3в.

В [34] предлагается обходиться без детектора краев, встраивая фильтрацию изображения в вычисление преобразования Радона. Если фильтр является симметричным или антисимметричным, то это же полностью относится и к вычислению преобразования Радона через преобразование Хартли. Такой подход позволяет применить к **серому** изображению только **линейную фильтрацию**. В настоящей работе применялся детектор краев [41], (см. раздел 2.2), что позволяет обнаруживать прямые линии между объектами различающимися как яркостью, так и только цветом. Таким образом в отличие от [34] можно обрабатывать в том числе и цветные изображения. Заметим также, что поскольку на контурном препарате фон черный, а линии изображаются светлым, дополнение нулями до ближайшей степени 2 (или последующих степеней для повышения точности определения параметров линии, как в [34]) можно выполнять непосредственно, не производя вычитания из изображения средней яркости и умножения изображения на функцию окна (Гаусса, Хемминга и т.п.). Кроме этого, возможно применять другие детекторы краев, с целью получения более качественного контурного препарата. На рис. 2е,и найдены почти одни и те же прямые линии. Однако возможны случаи, когда границы между цветными объектами могут быть хорошо найдены только на цветном изображении. В этом случае применение детектора [41] дает значительные преимущества.

Поиск ярких точек после фильтрации осуществляется в два этапа. На первом применяется пороговая обработка и интенсивности пикселей меньше пороговой обнуляются. Результатом является набор ярких «островков» точек. На втором этапе вычисляется центр масс каждого «островка» (весом точки считается ее интенсивность), который и определяет параметры прямой.

Алгоритм 3. Поиск максимумов на синограмме.

1. *Вычислить пороговую интенсивность* I_{th} *по формуле:* $I_{th} = \frac{1}{3} I_{\min} L_{\min}$ *, где* $I_{\min}$ *и* $L_{\min}$ *- минимальные интенсивность и длина линии на контурном препарате, которую необходимо найти. Обнулить точки с интенсивностью меньше пороговой.*
2. *Создать новый кластер. Найти на синограмме точку с максимальной интенсивностью, объявить ее «центром» кластера, поместить ее в список граничных точек кластера и обнулить ее значение на синограмме, чтобы исключить ее из дальнейшего рассмотрения.*
3. *Пока список граничных точек не пуст, для каждой граничной точки* B *:*
    a. *Добавить в список граничных точек тех соседей точки* B *, интенсивность которых больше, чем одна треть от интенсивности «центра» кластера.*
    b. *Обнулить интенсивность точки* B *на синограмме и удалить ее из списка граничных точек.*
4. *Если на синограмме есть точки с интенсивностью больше пороговой, то перейти к п.2.*
5. *Удалить кластеры с числом точек, меньшим чем* $4\sigma^2$ *, где* σ *- полуширина функции Гаусса, используемой для вычисления контурного препарата.*
6. *Поставить в соответствие каждому кластеру прямую с параметрами, соответствующими «центру» кластера.*

Коэффициент одна треть при определении пороговой интенсивности введен для того, чтобы оставить не только максимум, соответствующий прямой линии, но и некоторую окрестность вокруг него на синограмме.

На графике (рис.4) приведено сравнение времени работы алгоритма через преобразование Фурье и через преобразование Хартли. Вычисления выполнялись на машине Intel Pentium 4,

2.8 ГГц, КЭШем 1 Мб, ОЗУ 1 Гб под операционной системой Windows-XP, программа была скомпилирована с помощью Microsoft Visual Studio 2003 (ver. 7.1). Для экспериментальной проверки быстродействия и сравнения производительности предлагаемого алгоритма и алгоритма из [34] бралось изображение размером 8192x8192 пикселей. Строилась последовательность изображений путём уменьшения стороны изображения в два раза. Для каждого изображения вычислялось преобразование Радона с использованием преобразования Хартли и по методике из [34]. Результаты представлены на графике. Видно, что - затраты времени удовлетворяют оценке $O(N \log N), \quad N = M^2$. Предлагаемый алгоритм быстрее, чем [34] в 6 раз, при использованной реализации БПФ [39] и собственной реализации БПХ, которая в основном совпадала с [38], за исключением оптимизации из [38], касающейся уменьшения количества вычислений синуса (вычислительные эксперименты показали, что непосредственное вычисление синуса на процессорах типа Intel P3, P4 зачастую быстрее, чем выполнение умножений и сложений как в [38]). Уменьшение отношения времен работы алгоритмов при малых размерах изображения (128 и 256 пикселей) с 6 до 1.5связано с тем, что изображение целиком помещалось в КЭШе процессора. При дальнейшем увеличении линейных размеров изображения начиная 8192 пикселей также наблюдается увеличение коэффициента свыше 6 из-за свопа памяти, которое начинается именно с алгоритма [34], как требующего большего ее количества. Таким образом можно сказать, что в диапазоне практически интересных размеров изображений предложенный алгоритм быстрее чем [34] примерно в 6 раз, ускорение в основном обусловлено меньшим числом обращений к памяти. Или иными словами изображение обрабатывается предлагаемым алгоритмом немного быстрее, чем изображение с вдвое меньшей шириной и высотой алгоритмом [34], меньший расход памяти позволяет обрабатывать изображения большего размера. Интересно отметить, что аналогичный тест на ноутбуке с процессором AMD показывает еще большее ускорение: примерно в 8 раз против 6.

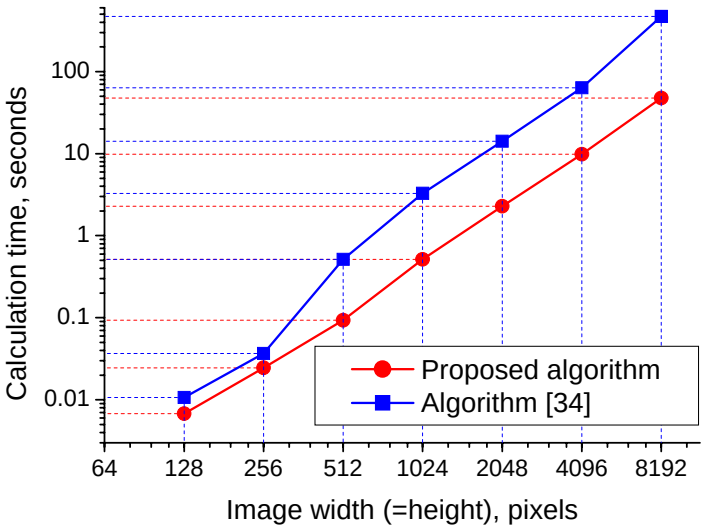

Рис. 4. Время вычисления преобразования Радона через преобразование Хартли и Фурье

3. ПРОЕКТИВНОЕ ПРЕОБРАЗОВАНИЕ (ПП)

В случае если сцена является плоской, существует плоское проективное преобразование $\mathbf{P}$ (матрица 3x3), которое **точно** переводит изображение сцены, полученное первой камерой в изображение полученное второй камеры [1]:

$$\vec{\mathbf{h}}_2 = \mathbf{P}\vec{\mathbf{h}}_1, \tag{32}$$

где $\vec{\mathbf{h}}_1$ и $\vec{\mathbf{h}}_2$ - векторы однородных координат точек на первом и втором изображениях соответственно:

$$\vec{\mathbf{h}}_i = (\tilde{x}_i, \tilde{y}_i, \tilde{z}_i)^T, \quad i = 1,2 \tag{33}$$

Ниже приводится используемая параметризация проективного преобразования, формула связывающая $\mathbf{P}$ со взаимным положением камер и сцены-плоскости, а также формула связывающая параметры прямых на изображениях.

3.1 Параметризация ПП

Рассмотрим модель плоской сцены (рис. 5). Плоская сцена σ снимается двумя камерами с фокусами в точках F_1 и F_2. Нормаль к плоскости σ обозначена через $\vec{\mathbf{n}}$ и направлена от первой камеры. С камерами связаны системы координат $\vec{\mathbf{i}}_1, \vec{\mathbf{j}}_1, \vec{\mathbf{k}}_1$ и $\vec{\mathbf{i}}_2, \vec{\mathbf{j}}_2, \vec{\mathbf{k}}_2$ с началами в фокусах камер. Фокальные плоскости камер обозначены через σ_1 и σ_2. Орты $\vec{\mathbf{i}}_1$, $\vec{\mathbf{j}}_1$ и $\vec{\mathbf{i}}_2$, $\vec{\mathbf{j}}_2$ параллельны плоскостям σ_1 и σ_2, орты $\vec{\mathbf{k}}_1$ и $\vec{\mathbf{k}}_2$ направлены вдоль оптических осей камер. Сдвиг от фокуса первой камеры к фокусу второй камеры обозначен через $\vec{\mathbf{t}}$. Ориентация первой камеры переводится в ориентацию второй камеры ортогональной матрицей $\mathbf{R}$:

$$\mathbf{R}(\vec{\mathbf{i}}_1\ \vec{\mathbf{j}}_1\ \vec{\mathbf{k}}_1) = (\vec{\mathbf{i}}_2\ \vec{\mathbf{j}}_2\ \vec{\mathbf{k}}_2) \tag{34}$$

Плоскость σ удалена от фокуса первой камеры F_1 на расстояние d_1, ($d_1 \geq 0$).

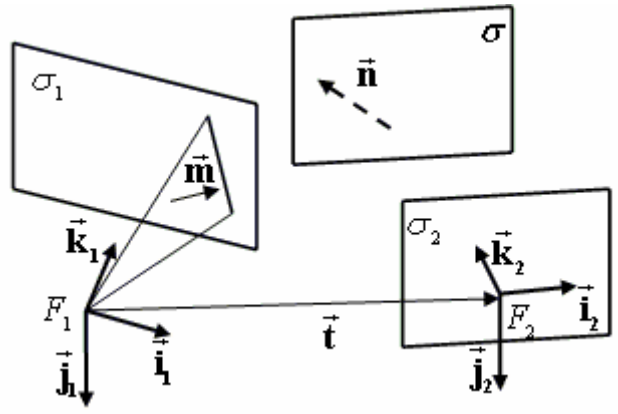

Рис. 5. Плоская сцена.

После небольших выкладок из геометрических соображений можно получить следующее выражение, связывающее однородные координаты соответствующих точек на изображениях σ_1 и σ_2:

$$\mathbf{P} = \mathbf{R}^T\left(\mathbf{I} - \frac{\vec{\mathbf{t}}\vec{\mathbf{n}}^{\mathrm{T}}}{d_1}\right) \tag{35}$$

где $\mathbf{I}$ - единичная матрица 3x3.

3.2 Преобразование параметров прямых

При проективном преобразовании $\mathbf{P}$ изображения образом прямой является прямая. Найдем связь между параметрами ρ_1, φ_1 и ρ_2, φ_2 - образами некоторой прямой сцены на первом и втором изображениях. Перейдем к $\vec{\mathbf{m}}$-параметризации прямой (см. 2.3.2). Уравнение плоскости, проходящей через фокус i-ой камеры и прямую сцены имеет вид:

$$(\vec{\mathbf{m}}_i, \vec{\mathbf{h}}_i) = 0\,, \quad i = 1,2\,. \tag{36}$$

Отсюда, воспользовавшись (32) получаем:

$$\vec{\mathbf{m}}_2 = \frac{\Lambda\vec{\mathbf{m}}_1}{|\,\Lambda\vec{\mathbf{m}}_1\,|}, \tag{37}$$

где

$$\Lambda = (\mathbf{P}^{-1})^T \tag{38}$$

Используя формулу Шермона-Моррисона обращения матрицы [39]:

$$(\mathbf{A} + \vec{\mathbf{u}}\vec{\mathbf{v}}^T)^{-1} = \mathbf{A}^{-1} - \frac{\mathbf{A}^{-1}\vec{\mathbf{u}}(\vec{\mathbf{v}}\mathbf{A}^{-1})^T}{1 + \vec{\mathbf{v}}\mathbf{A}^{-1}\vec{\mathbf{u}}} \tag{39}$$

и учитывая специальный вид (35) матрицы $\mathbf{P}$, перепишем (38) в виде:

$$\Lambda = \mathbf{R}^{\mathrm{T}}\left(\mathbf{I} + \frac{\vec{\mathbf{n}}\vec{\mathbf{t}}^{\mathrm{T}}}{d_1 - (\vec{\mathbf{t}},\vec{\mathbf{n}})}\right) \tag{40}$$

Теорема 2.

При условии, что разрешены лишь сдвиги второй камеры, не превосходящие по модулю доли κ расстояния от первой камеры до плоскости σ:

$$|\vec{\mathbf{t}}| \le \kappa d_1, \tag{41}$$

спектр матрицы Λ ограничен:

$$\frac{1}{1+\kappa} \le |\lambda(\Lambda)| \le \frac{1}{1-\kappa} \tag{42}$$

Доказательство.

Действительно, для любого единичного вектора $\vec{\mathbf{e}}$:

$$\begin{aligned} |\Lambda\vec{\mathbf{e}}| &= \left|\mathbf{R}^{\mathrm{T}}\left(\mathbf{I} + \frac{\vec{\mathbf{n}}\vec{\mathbf{t}}^{\mathrm{T}}}{d_1 - (\vec{\mathbf{t}},\vec{\mathbf{n}})}\right)\vec{\mathbf{e}}\right| = \left|\left(\mathbf{I} + \frac{\vec{\mathbf{n}}\vec{\mathbf{t}}^{\mathrm{T}}}{d_1 - (\vec{\mathbf{t}},\vec{\mathbf{n}})}\right)\vec{\mathbf{e}}\right| \\ \frac{1}{1+\kappa} = 1 - \frac{\kappa}{1+\kappa} &\le \left|\left(\mathbf{I} + \frac{\vec{\mathbf{n}}\vec{\mathbf{t}}^{\mathrm{T}}}{d_1 - (\vec{\mathbf{t}},\vec{\mathbf{n}})}\right)\vec{\mathbf{e}}\right| \le 1 + \frac{\kappa}{1-\kappa} = \frac{1}{1-\kappa} \end{aligned} \tag{43}$$

■.

Чем меньше сдвиг $\vec{\mathbf{t}}$, тем ближе модули собственных чисел к единице. Формула (42) будет использоваться в разделах 4.1.1, 4.2.

4. СОВМЕЩЕНИЕ ИЗОБРАЖЕНИЙ

Задача состоит в нахождении проективного преобразования $\mathbf{P}$, которое переводит набор прямых с параметрами $\rho_{1,i}$, $\varphi_{1,i}$ на первом изображении в набор $\rho_{2,j}$, $\varphi_{2,j}$ на втором изображении.

Сложность задачи состоит в следующем:

1. Соответствия между прямыми линиями на изображениях неизвестны.
2. Количество найденных на изображениях прямых линий различно.
3. Прямые, найденные на первом изображении, могут не быть найдены на втором изображении и наоборот.

Промежуточным результатом работы алгоритма являются набор возможных вариантов совмещения изображений. Каждый вариант применяется к первому изображению и вычисляется разностный кадр между преобразованным первым изображением и вторым (в их общей части).

Лучшим считается преобразование, для которого сумма квадратов интенсивностей пикселей разностного кадра минимальна. При вычислении разностного кадра учитывается только пересекающаяся часть обоих кадров.

4.1 Плоская сцена

4.1.1 Построение функционала

Предлагаемый подход состоит в минимизации функционала F, зависящего от искомых параметров проективного преобразования и от параметров найденных прямых:

$$F = F(\mathbf{P}, \rho_{1,i}, \varphi_{1,i}, \rho_{2,j}, \varphi_{2,j}) \tag{44}$$

Рассматриваемый функционал должен обладать свойством, что его значение на преобразовании $\mathbf{P}$ тем меньше, чем большее число прямых первого изображения (и чем точнее) переводится преобразованием $\mathbf{P}$ в прямые второго изображения. В настоящей работе используется следующий функционал:

$$F = \sum_{i=1}^{N_1} \sum_{j=1}^{N_2} c_{ij} F_{ij} \tag{45}$$

$$F_{ij} = -\exp\left(-\frac{|\vec{\mathbf{m}}_{2,j} \times \Lambda \vec{\mathbf{m}}_{1,i}|^2}{2\sigma_f^{\,2}} \right) \tag{46}$$

Множитель c_{ij} определяет насколько похоже распределение цвета вокруг i-ой прямой на первом изображении на распределение цвета вокруг j-ой прямой на втором изображении, и определяется в разделах 4.3, 4.4

Множитель F_{ij} определяет геометрическую близость i-ой прямой первого изображения (подвергнутого преобразовании $\mathbf{P}$) и j-ой прямой второго изображения. Если i-ая прямая точно переводится преобразованием $\mathbf{P}$ в j-ую прямую, то F_{ij} равно -1.

Параметр σ_f определяет границы параметров прямых, в которых они еще считаются одинаковыми. Согласно (42) модули собственных значений матрицы Λ сосредоточены около единицы, поэтому если угол между векторами $\Lambda\vec{\mathbf{m}}_{1,i}$ и $\vec{\mathbf{m}}_{2,j}$ превосходит σ_f, то $|F_{ij}| << 1$:

$$|\vec{\mathbf{m}}_{2,j} \times \Lambda \vec{\mathbf{m}}_{1,i}| >> \sigma_f \quad \rightarrow \quad |F_{ij}| \approx 0 \tag{47}$$

Типичные значения параметра σ_f лежат в пределах $\sigma = 0.01 \div 0.2$, способы оценки параметра σ_f будут приведены ниже (см. раздел 4.2).

На значение параметра σ_f накладывается ограничение сверху:

$$\sigma_f < \frac{1}{2} \min_{j_1, j_2, j_1 \neq j_2} \sin \phi_{j_1, j_2}, \tag{48}$$

где ϕ_{j_1, j_2} - угол между векторам $\vec{\mathbf{m}}_{2,j_1}$ и $\vec{\mathbf{m}}_{2,j_2}$. Условие (48) гарантирует, для фиксированного i среди слагаемых F_{ij} не более одного слагаемого существенно отлично от нуля. Это условие означает, что **каждой прямой на первом изображении ставится в соответствие не более одной прямой на втором изображении**.

4.1.2 Минимизация функционала

Построенный функционал (45) зависит в общем случае от восьми переменных (см. (35)): три угла поворота матрицы $\mathbf{R}$, две независимых компоненты единичного вектора $\vec{\mathbf{n}}$,

три компоненты вектора $\frac{\vec{\mathbf{t}}}{d_1}$. Этого и следовало ожидать, так как матрица проективного преобразования определена с точностью до числового множителя и может быть произвольной матрицей 3x3. Поиск всех восьми параметров является трудоемкой задачей, и, возможно, излишней, так как весь набор параметров может быть плохо определен. В настоящей работе был выбран набор из четырех самых существенных параметров, которыми и осуществлялась параметризация искомого проективного преобразования.

Вектор $\vec{\mathbf{n}}$ выбирается единичным и направленным вдоль орта $\vec{\mathbf{k}}_1$, так как прямые лучше детектируются на плоскостях, примерно перпендикулярных оптической оси камеры, и в большинстве случаев именно с такими прямыми алгоритму приходится иметь дело. Усредненное направление нормалей к таким плоскостям считается примерно направленным вдоль $\vec{\mathbf{k}}_1$:

$$\vec{\mathbf{n}} = (0,0,1)^T \tag{49}$$

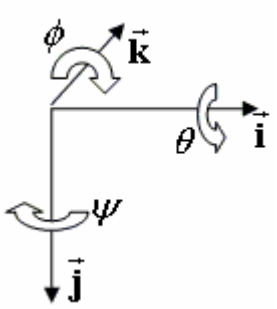

Рис. 6. Поворот камеры описывается тремя углами

Множество поворотов камеры ограничивается поворотом вокруг орта $\vec{\mathbf{k}}_1$ (рис. 6, угол поворота обозначен через ϕ) . Такое ограничение мотивируется тем, что поворот вокруг орта $\vec{\mathbf{i}}_1$ с точностью до членов второго порядка совпадает со сдвигом камеры вдоль орта $\vec{\mathbf{j}}_1$, а поворот вокруг орта $\vec{\mathbf{j}}_1$ со сдвигом вдоль орта $\vec{\mathbf{i}}_1$. Матрица $\mathbf{R}$ имеет следующий вид:

$$\mathbf{R} = \mathbf{R}_\phi = \begin{pmatrix} \cos\phi & -\sin\phi & 0 \\ \sin\phi & \cos\phi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{50}$$

При предположениях (49) и (50) матрица $\mathbf{\Lambda}$ имеет вид:

$$\Lambda(\phi,\vec{\mathbf{t}}) = \begin{pmatrix} \cos\phi & \sin\phi & 0 \\ -\sin\phi & \cos\phi & 0 \\ \frac{\tau_x}{1-\tau_z} & \frac{\tau_y}{1-\tau_z} & \frac{1}{1-\tau_z} \end{pmatrix} \tag{51}$$

где вектор $\vec{\boldsymbol{\tau}}$ определен по формуле:

$$\vec{\boldsymbol{\tau}} = \frac{\mathbf{t}}{d_1} \tag{52}$$

Минимизация функционала (44) происходит в четырехмерном пространстве параметров $(\phi,\vec{\boldsymbol{\tau}})$, где матрица $\mathbf{\Lambda}$ задается формулой (51).

В ходе предварительных численных экспериментов было установлено, что построенный функционал (45) имеет не единственный локальный минимум, и имеет смысл найти не только глобальный минимум, но и другие локальные минимумы со значением, близким к значению в глобальном минимуме. Полученные локальные минимумы определяют параметры искомых проективных преобразований. Исследование нескольких локальных минимумов обусловлено тем, что для сцен из нескольких плоскостей (см. ниже раздел 4.2) точного преобразования не существует и возможно существование нескольких преобразований, переводящих набор прямых на первом изображении в набор на втором. В связи с этим результатом ал-

горитма должно быть не единственное преобразование, а набор гипотез, из которых выбирается лучшая.

Алгоритм 4. Поиск локальных минимумов функционала:

1. *В четырехмерном параметрическом пространстве* $(\phi,\vec{\tau})$ *определить сетку и вычислить в узлах значения функционала.*
2. *Отсортировать узлы по возрастанию значения функционала в них и отобрать предопределенное число узлов (10~100), в которых функционал принял наименьшие значения.*
3. *Спуститься из отобранных узлов в локальный минимум методом сопряженных градиентов [39].*
4. *Удалить копии локальных минимумов, так как из разных начальных точек можно спуститься в один локальный минимум.*

Тонким моментом в приведенном алгоритме является выбор шага сетки. Если шаг слишком велик, то велика вероятность «проскочить» локальный минимум. Если шаг слишком мал, то производится большое число лишних вычислений в силу большой размерности параметрического пространства (при уменьшении шага сетки вдвое по каждой из осей число узлов возрастает в 16 раз!).

Оценка шага может быть получена следующим образом. Рассмотрим два одинаковых изображения с единственной прямой с параметром $\vec{m}$. Введем обозначение:

$$f_0(\phi,\vec{t}) = |\vec{m}\times\Lambda(\phi,\vec{t})\vec{m}|^2 \tag{53}$$

и запишем функционал:

$$F_0(\phi,\vec{t}) = -\exp\left(-\frac{f_0(\phi,\vec{t})}{2\sigma^2}\right) \tag{54}$$

Локальный минимум функционала F_0 находится в точке $(0,\vec{0})$. Индекс 0 у функционала означает, что он исследуется при малых параметрах $\phi,\vec{t}$. После элементарных, но громоздких вычислений получается Гессиан (матрица вторых производных) функционала F_0 в точке $(0,\vec{0})$:

$$\mathbf{H}_0 = \frac{(m_x^2+m_y^2)}{\sigma^2}\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & m_x^2 & m_x m_y & m_x m_z \\ 0 & m_x m_y & m_y^2 & m_y m_z \\ 0 & m_x m_z & m_y m_z & m_z^2 \end{pmatrix} \tag{55}$$

Матрица $\mathbf{H}_0$ имеет следующие собственные значения и собственные векторы:

$$\lambda_1=\lambda_2=\frac{(m_x^2+m_y^2)}{\sigma^2} \quad \vec{h}_1=(1,0,0,0)^T \quad \vec{h}_2=\begin{pmatrix}0\\ \vec{m}\end{pmatrix}$$
$$\lambda_{3,4}=0 \quad \vec{h}_{3,4}=\begin{pmatrix}0\\ \vec{v}\end{pmatrix}, \quad \vec{v}\perp\vec{m} \tag{56}$$

Полученный результат говорит о том, что F_0 увеличивается при повороте камеры и сдвиге вдоль направления $\vec{m}$, но остается неизменным при сдвиге камеры вдоль любого направления, лежащего в плоскости, перпендикулярной вектору $\vec{m}$.

Так как $|\vec{m}|=1$, то согласно (56) :

$$\max(\lambda(\mathbf{H}_0)) \le \frac{1}{\sigma^2} \tag{57}$$

Для пары изображений с N_1 одинаковыми прямыми при выполнении условия (48) справедлива оценка:

$$\max(\lambda(\mathbf{H}_0)) \le \frac{N_1}{\sigma^2} \tag{58}$$

так как при условии (48) членами $F_{ij}, i \neq j$ функционала можно пренебречь.

Для изображений с различными прямыми также справедлива оценка (58), так как если прямые первого изображения неточно переводятся в прямые второго изображения, то значение функционала и собственные значения Гессиана уменьшаются.

Форма гиперповерхности, задаваемой функционалом в параметрическом пространстве, вблизи локального минимума считается параболической:

$$F(\vec{s}) = -N_c + \frac{1}{2}\vec{s}^T \mathbf{H}\vec{s}, \quad \vec{s} = \begin{pmatrix} \phi \\ \vec{t} \end{pmatrix} \tag{59}$$

где N_c - число общих прямых на изображениях.

В предположении $\max(\lambda(\mathbf{H})) \le 1/\sigma^2$, получается, что при $|\vec{s}| < \sigma$ значение функционала ограничено: $|F(\vec{s})| \le N_c/2$.

Таким образом, в п.1 Алгоритма 4 шаг сетки следует выбрать равным σ по всем осям параметрического пространства, а в п.2 спускаться в локальные минимумы только из узлов, для которых $|F| \ge N_c/2$. Так как число общих прямых N_c неизвестно до окончания работы алгоритма, оно полагалось равным $N_c = \min(N_1, N_2)/2$. В случае, если имеется точная априорная информация для N_c, оценка может быть изменена.

4.2 Неплоская сцена

В случае если прямые расположены не на одной плоскости, точного проективного преобразования, переводящего первое изображение во второе не существует. Следовательно, есть ограничение снизу на значение параметра σ_f, так как необходима некоторая толерантность алгоритма к тому, что прямые первого изображения не должны точно перейти в прямые второго изображения. Выразим нижнюю границу параметра σ_f через априорные характеристики сцены, такие как разброс расстояний до плоскостей сцены и разброс ориентаций этих плоскостей.

Рассмотрим сцену, состоящую из нескольких плоскостей σ_i с нормалями $\vec{\mathbf{n}}_i$ и расстояниями до фокуса первой камеры d_i. На плоскостях σ_i расположены прямые линии. Сцена снимается камерами из двух точек. Первая камера переводится во вторую поворотом матрицей $\mathbf{R}$ и сдвигом на вектор $\vec{\mathbf{t}}$. В этом случае для каждой плоскости σ_i преобразование $\mathbf{\Lambda}_i$, переводящее образы прямых на плоскости σ_i на первом изображении в образы тех же прямых на втором изображении, будет своим:

$$\Lambda_i = \mathbf{R}^{\mathrm{T}}\left(\mathbf{I} + \frac{\vec{\mathbf{n}}_i \vec{\mathbf{t}}^{\mathrm{T}}}{d_i - (\vec{\mathbf{t}}, \vec{\mathbf{n}}_i)}\right) \tag{60}$$

Обозначим через $\vec{\mathbf{m}}_{1,ij}$ параметры прямой на первом изображении, которая является образом j-ой прямой, расположенной на плоскости σ_i. Параметры соответствующей прямой на втором изображении обозначим через $\vec{\mathbf{m}}_{2,ij}$. Тогда согласно (37)

$$\frac{\mathbf{\Lambda}_i \vec{\mathbf{m}}_{1,ij}}{|\mathbf{\Lambda}_i \vec{\mathbf{m}}_{1,ij}|} = \vec{\mathbf{m}}_{2,ij}, \quad \forall i, j \tag{61}$$

Для некоторых средних величин (не уточняя, каких именно) $\vec{\mathbf{n}}_0$ и d_0, определим преобразование $\mathbf{\Lambda}_0$:

$$\Lambda_0 = \mathbf{R}^{\mathrm{T}} \left(\mathbf{I} + \frac{\vec{\mathbf{n}}_0 \vec{\mathbf{t}}^{\mathrm{T}}}{d_0 - (\vec{\mathbf{t}}, \vec{\mathbf{n}}_0)} \right) \tag{62}$$

Оценим величину $q(\mathbf{\Lambda_0})$:

$$q(\mathbf{\Lambda}_0) = \max_{i,j} \phi_{i,j}, \tag{63}$$

где $\phi_{i,j}$ - угол между векторами $\vec{\mathbf{m}}_{2,ij}$ и $\mathbf{\Lambda}_0 \vec{\mathbf{m}}_{1,ij}$.

Согласно (62) существует преобразование $\mathbf{\Lambda}_0$ которое удовлетворяет условию, что для любой прямой $\vec{\mathbf{m}}_{1,ij}$ на первом изображении угол между векторами $\mathbf{\Lambda}_0 \vec{\mathbf{m}}_{1,ij}$ и $\vec{\mathbf{m}}_{2,ij}$ не превосходит $q(\mathbf{\Lambda_0})$. Согласно (48), выбрав параметр σ_f порядка $q(\mathbf{\Lambda_0})$ функционал будет принимать примерно одинаковые значения как для идеально плоской сцены так и рассматриваемой сцены из набора плоскостей.

Выразим $q(\mathbf{\Lambda_0})$, задавшись некоторым распределением величин $\vec{\mathbf{n}}_i$ и d_i. Очевидно, что $q(\mathbf{\Lambda_0})$ зависит от конкретного вида распределения $\vec{\mathbf{n}}_i$ и d_i. При самом общем (и весьма грубом) предположении, о том, что

1. векторы $\vec{\mathbf{n}}_i$ равномерно распределены по единичной полусфере (так как вектор $\vec{\mathbf{n}}_i$ направлен от камеры, то он имеет неотрицательную третью компоненту, см. рис. 3)
2. величины $\frac{|\vec{\mathbf{t}}|}{d_i}$ равномерно распределены на отрезке $[0, \varepsilon]$

можно получить следующую оценку на величину $q(\mathbf{\Lambda_0})$:

$$q(\mathbf{\Lambda}_0) \le \frac{\varepsilon}{1-\varepsilon} \tag{64}$$

Объединяя (48) и (64) получаем окончательный диапазон возможных значений σ_f:

$$\frac{\varepsilon}{1-\varepsilon} \le \sigma_f \le \frac{1}{2} \min_{j_1, j_2, j_1 \ne j_2} \sin \phi_{j_1, j_2} \tag{65}$$

Величин ϕ_{j_1, j_2} определяются после (48) и находятся прямым вычислением по определению.

Таким образом, **Алгоритм 4** поиска локальных минимумов требует задания параметра σ_f. Параметр σ_f ограничен снизу и сверху (65). Нижняя граница определяется априорными

параметрами сцены: разбросом нормалей к плоскостям и расстоянием до них. Верхняя граница определяется параметрами найденных на изображениях прямых.

Если верхняя граница окажется меньше нижней, то это значит, что в рамках рассматриваемой модели совмещения, прямые с близкими параметрами (для которых $\phi_{j_1,j_2} < \frac{\varepsilon}{1-\varepsilon}$) считаются одинаковыми. Другими словами, одной прямой на первом изображении может соответствовать более одной прямой на втором изображении. Верхняя граница σ_f должна быть увеличена до значения нижней. При этом увеличившаяся погрешность найденных параметров совмещения связана с грубостью предположения о плоской сцене и только грубоприближенном выполнении отображения (32).

4.3 Цвет вокруг прямых линий

Рассмотрим две прямых линии – первую на одном снимке сцены, вторую на другом. Принимая во внимание распределение цвета вблизи прямых, требуется определить, можно ли считать эти прямые образами одной и той же прямой линии сцены. Ответ на поставленный вопрос не является очевидным. Например, при съемке прямой границы некоторого объекта цвет фона может быть разным на разных снимках и цвета вокруг границы будут совпадать только по одну сторону. Далее, если отражательная способность объекта зависит от направления отражения, то интенсивность вокруг границы на одном снимке будут отличаться от интенсивности на другом. Следовательно, ответом должна быть не дискретная переменная (да/нет), а непрерывная (степень правдоподобия). Естественным математическим аппаратом для этого является **нечеткая логика** [42,43].

Еще одним преимуществом нечеткой логики является «прозрачность» требований в противовес эмпирическим функционалам. Важны именно правила (идеи), а не детальный вид функционала. Достаточно легко дополнить или видоизменить правила, а функционал из них получается формально согласно выбранным определениям нечетких операций.

Учет цвета состоит в выборе коэффициентов c_{ij} из (45). С каждой прямой линией l связано два цветовых дескриптора $\vec{\mathbf{d}}_1$ и $\vec{\mathbf{d}}_2$ – по одну и по другую сторону от нее.

В текущей реализации дескриптор $\vec{\mathbf{d}}$ состоит из трех цветовых компонент: r, g, b:

$$\vec{\mathbf{d}} = (r, g, b) \tag{66}$$

Будем также полагать, что $r, g, b \in [0,1]$.

Каждая компонента является усредненным значением соответствующих цветов пикселей исходного изображения $I(\vec{\mathbf{p}})$ в некоторой области O вокруг прямой по одну из ее сторон (рис. 7).

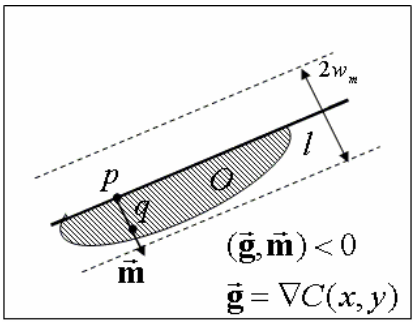

Рис. 7. Область около прямой, по которой усредняется цвет

Пусть $C(\vec{\mathbf{p}})$ обозначает контурный препарат изображения. Область O определяется следующими условиями:

1. Область O содержится в полосе шириной $2w_m = 6\chi$, серединой которой является прямая l. Величина χ есть полуширина функции Гаусса, используемой для вычисления производных при вычислении контурного препарата (см. раздел 2.2).
2. В любой точке области O интенсивность $I(\vec{\mathbf{p}})$ убывает при удалении от прямой.

При усреднении цвета каждая точка $\vec{\mathbf{q}}$ области O имеет вес w_q, который зависит от ее положения. Во-первых, если вблизи точки $\vec{\mathbf{q}}$ есть точка $\vec{\mathbf{p}}$, расположенная на прямой l, в которой велика интенсивность контурного препарата, то вес точки $\vec{\mathbf{q}}$ полагается большим. Во-вторых, точки от l на расстоянии меньшем χ, учитываются с малым весом, так как,

- на границе происходит переход от одного цвета к другому;
- контур локализован с некоторой погрешностью, зависящей от используемого детектора краев.

Обозначим через $\vec{I}(\vec{\mathbf{p}}) = (R(\vec{\mathbf{p}}), G(\vec{\mathbf{p}}), B(\vec{\mathbf{p}}))^T$ вектор из красной, зеленой и синей составляющих цвета пикселя $\vec{\mathbf{p}} = (x, y)$ изображения. Приведем алгоритм вычисления цветового дескриптора:

Алгоритм 4. Вычисление цветового дескриптора

- ✓ $\vec{\mathbf{d}} = 0$, $W = 0$
- ✓ Для каждой точки $\vec{\mathbf{p}}$ прямой l с шагом единица:
  - o $w_p = C(\vec{\mathbf{p}})$
  - o $m \perp l, \quad \vec{\mathbf{p}} \in m$.
    - $\vec{\mathbf{q}}_0 = \vec{\mathbf{p}}$:
    - Для каждой $\vec{\mathbf{q}} \in m$:
      - $\vec{\mathbf{q}}_n = \vec{\mathbf{q}}_{n-1} + \frac{\vec{\mathbf{m}}}{|\vec{\mathbf{m}}|}$:
      - Если $C(\vec{\mathbf{q}}_n) > C(\vec{\mathbf{q}}_{n-1})$ или $|\vec{\mathbf{p}} - \vec{\mathbf{q}}_n| > w_m$, то выход из цикла.
      - $w_q = f_1(w_p) f_2(|\vec{\mathbf{p}} - \vec{\mathbf{q}}_n|)$ /см. (67), (68)/
      - $\vec{\mathbf{d}} = \vec{\mathbf{d}} + w_q \vec{I}(\vec{\mathbf{q}}_n)$
      - $W = W + w_q$
- ✓ $\vec{\mathbf{d}} = \frac{\vec{\mathbf{d}}}{W}$

В текущей реализации эмпирические весовые функции $f_1(w_C)$ и $f_2(t)$ имеют следующий вид:

$$f_1(w_p) = w_p \tag{67}$$

$$f_2(t) = \frac{t^2}{\chi^2 + t^2} \tag{68}$$

4.4 Нечеткая логика

Логические операции

В представленной работе нечеткой логической переменной называется действительное число от нуля до единицы. Используются следующие определения нечетких логических операций:

$$a\ AND\ b \equiv a \cdot b \tag{69}$$

$$NOT\ a \equiv 1 - a \tag{70}$$

где a, b - нечеткие логические переменные. Логическое ИЛИ определяется через известную формулу теории множеств:

$$a\ OR\ b = NOT\ ((NOT\ a)\ AND\ (NOT\ b)) \tag{71}$$

Подставляя (69), (70) в (71) получается:

$$a\ OR\ b \equiv 1 - (1 - a) \cdot (1 - b) \tag{72}$$

Определение (72) обладает свойством «усиления» нечетких утверждений. Например, если $a = b = 0.5$, то $a\ OR\ b = 0.75$, что превосходит каждый из операндов.

Возможны и другие способы определения логических операций, но результат совмещения изображений от этого зависит слабо, что является косвенным признаком правильности алгоритма.

Для вычисления яркости и цвета использовалось цветовое пространство YUV,которое разделяет яркость и цвет, но не приводит к циклическим координатам, как, например, HSV. Переход от компонент цветового дескриптора (66) к компонентам Y,U,V осуществляется по формулам:

$$\begin{aligned} Y(\vec{\mathbf{d}}) &= (\ \ 0.299 \quad\ \ 0.587 \quad\ \ 0.114)^T \vec{\mathbf{d}} \\ U(\vec{\mathbf{d}}) &= (-0.147 \quad -0.289 \quad\ \ 0.436)^T \vec{\mathbf{d}} \\ V(\vec{\mathbf{d}}) &= (\ \ 0.615 \quad -0.515 \quad -0.1)^T \vec{\mathbf{d}} \end{aligned} \tag{73}$$

Обозначим через $\vec{\mathbf{v}}$ вектор цветовых компонент:

$$\vec{\mathbf{v}} = (U, V) \tag{74}$$

Правила определения похожести дескрипторов:

R_1: Если обе интенсивности малы, то дескрипторы похожи («в темноте все кошки серы»).

R_2: Если обе интенсивности не малы и цвета дескрипторов похожи и интенсивности дескрипторов похожи, то дескрипторы похожи.

Ниже приведена таблица функций, каждая из который принимает некоторый набор аргументов и возвращает нечеткую логическую переменную (действительное число от нуля до единицы)

Имя функции	Смысл функции	Формула
$\mathrm{Is}(Y)$	Интенсивность мала	$\max(0,1 - \alpha Y)$
$\mathrm{Ce}(\vec{\mathbf{v}}_1, \vec{\mathbf{v}}_2)$	Цвета дескрипторов похожи	$\max(0,1 - \beta \mid \vec{\mathbf{v}}_1 - \vec{\mathbf{v}}_2 \mid)$
$\mathrm{Ie}(Y_1, Y_2)$	Интенсивности дескрипторов похожи	$\mathrm{Is}(\mid Y_1 - Y_2 \mid)$

Параметр α полагается равным: $\alpha = 1/3\sigma_{noise}$, где σ_{noise} - оценка уровня шума камеры (σ_{noise} для изображения с диапазоном интенсивности от 0 до 255 полученного типовым фотоаппаратом при умеренном освещении $\sigma_{noise} = 1 \div 3$). Параметр β также связан с уровнем шума на

изображении: $\beta = 1/10\sigma_{noise}$. В ходе численных экспериментов было установлено, что при изменении параметров α , β в широком диапазоне, результат изменяется незначительно.

Обозначим условия правил R_1 и R_2 через R_1^c и R_2^c (это действительные числа в диапазоне от нуля до единицы):

$$R_1^c(\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2) = \mathrm{Is}(Y_1)\ AND\ \mathrm{Is}(Y_2) \tag{75}$$

$$R_2^c(\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2) = (NOT\ \mathrm{Is}(Y_1))\ AND\ (NOT\ \mathrm{Is}(Y_2))\ AND\ \mathrm{Ce}(\vec{\mathbf{v}}_1, \vec{\mathbf{v}}_2)\ AND\ \mathrm{Ie}(Y_1, Y_2) \tag{76}$$

Результат объединения правил R_1 и R_2 определим как логическое ИЛИ величин R_1^c, R_2^c и обозначим его через $\mathrm{De}(\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2)$ - степень похожести дескрипторов (действительное число в диапазоне от нуля до единицы):

$$\mathrm{De}(\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2) = R_1^c(\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2)\ OR\ R_2^c(\vec{\mathbf{d}}_1, \vec{\mathbf{d}}_2) \tag{77}$$

Обозначим через $\vec{\mathbf{d}}_1^1$ и $\vec{\mathbf{d}}_1^2$ пару цветовых дескрипторов линий на первом изображении, а через $\vec{\mathbf{d}}_2^1$ и $\vec{\mathbf{d}}_2^2$ пару цветовых дескрипторов прямой на втором изображении.

Правила определения степени похожести прямых линий:

R_3: Если ($\vec{\mathbf{d}}_1^1$ и $\vec{\mathbf{d}}_2^1$ похожи И $\vec{\mathbf{d}}_1^2$ и $\vec{\mathbf{d}}_2^2$ похожи) ИЛИ ($\vec{\mathbf{d}}_1^2$ и $\vec{\mathbf{d}}_2^1$ похожи И $\vec{\mathbf{d}}_1^1$ и $\vec{\mathbf{d}}_2^2$ похожи), то прямые похожи.

R_4: Если ($\vec{\mathbf{d}}_1^1$ и $\vec{\mathbf{d}}_2^1$ похожи) ИЛИ ($\vec{\mathbf{d}}_1^1$ и $\vec{\mathbf{d}}_2^2$ похожи) ИЛИ ($\vec{\mathbf{d}}_1^2$ и $\vec{\mathbf{d}}_2^1$ похожи) ИЛИ ($\vec{\mathbf{d}}_1^2$ и $\vec{\mathbf{d}}_2^2$), то прямые похожи.

Правило R_3 учитывает случай, когда распределения цвета по обе стороны от прямых совпадают. Правило R_4 учитывает случай, когда совпадают распределения цвета хотя бы по одну сторону от прямых (когда объект с прямыми границами снимается на разном фоне). Очевидно, что если выполняется правило R_3, то выполняется и правило R_4.

Обозначим условия правил R_3 и R_4 через R_3^c и R_4^c (это действительные числа в диапазоне от нуля до единицы):

$$\begin{aligned} &R_3^c(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1, \vec{\mathbf{d}}_2^2) = (\mathrm{De}(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_2^1)\ AND\ \mathrm{De}(\vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^2))\ OR \\ &OR\ (\mathrm{De}(\vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1)\ AND\ \mathrm{De}(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_2^2)) \end{aligned} \tag{78}$$

$$\begin{aligned} &R_4^c(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1, \vec{\mathbf{d}}_2^2) = \mathrm{De}(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_2^1)\ OR\ \mathrm{De}(\vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^2) \\ &OR\ (\mathrm{De}(\vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1)\ OR\ \mathrm{De}(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_2^2)) \end{aligned} \tag{79}$$

Результат объединения правил R_3 и R_4 определим как логическое ИЛИ величин R_3^c, R_4^c и обозначим его через $\mathrm{Le}(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1, \vec{\mathbf{d}}_2^2)$ - степень похожести прямых (действительное число в диапазоне от нуля до единицы):

$$\mathrm{Le}(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1, \vec{\mathbf{d}}_2^2) = R_3^c(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1, \vec{\mathbf{d}}_2^2)\ OR\ R_4^c(\vec{\mathbf{d}}_1^1, \vec{\mathbf{d}}_1^2, \vec{\mathbf{d}}_2^1, \vec{\mathbf{d}}_2^2) \tag{80}$$

Обозначим через $\vec{\mathbf{d}}_{1,i}^1$ и $\vec{\mathbf{d}}_{1,i}^2$ пару цветовых дескрипторов i-ой прямой первого изображения, а через $\vec{\mathbf{d}}_{2,j}^1$ и $\vec{\mathbf{d}}_{2,j}^2$ пару цветовых дескрипторов j-ой прямой второго изображения. Определим c_{ij} из (43) равным степени похожести i-ой и j-ой прямых:

$$c_{ij} = \mathrm{Le}(\vec{\mathbf{d}}^{1}_{1,i}, \vec{\mathbf{d}}^{2}_{1,i}, \vec{\mathbf{d}}^{1}_{2,j}, \vec{\mathbf{d}}^{2}_{2,j}) \tag{81}$$

5. ОЦЕНКА РАЗМЕРА ОКНА ПОИСКА СООТВЕТСТВУЮЩИИХ ТОЧЕК

Предлагаемый подход оценки состоит в следующем.

1. Ввести «расстояние» между прямыми линиями на изображении, зависящее от их взаимного положения. «Расстояние» определяется как усредненное расстояние между точками прямых (см.(82)).
2. Преобразовать параметры прямых первого изображения согласно (37).
3. Для каждой прямой второго изображения найти «ближайшую» преобразованную прямую первого изображения. Таким образом, между прямыми первого и второго изображений устанавливается соответствие.
4. Для каждой прямой второго изображения вычислить «расстояние» до соответствующей преобразованной прямой первого изображения, а также среднее и дисперсию этих величин.
5. Вычислить медианы среднего и дисперсии и составить из этих величин оценку размера окна поиска соответствующих точек на втором изображении (88).

Предлагаемый подход оценивает среднее расстояние между точками прямых линий после грубого совмещения изображений и установления соответствия между прямыми. Оценка применима для случая, когда ближайший к камере объект (образы точек которого испытывают максимальные сдвиги на изображениях) содержит прямые линии. В случае наличия близко расположенного объекта перед камерой без прямых линий оценка будут заниженной. Однако полученную оценку предполагается использовать для задания окна поиска соответствующих точек с целью последующей оценки фундаментальной матрицы. Соответствующие точки на близких объектах могут быть не найдены, что не является критичным, так как фундаментальная матрица может быть оценена и по соответствующим точкам на более удаленных объектах, для которых окно поиска меньше. В дальнейшем, используя эпиполярное ограничение поиск соответствующих точек может быть выполнен вдоль эпиполярных линий с большим диапазоном диспарити, что позволит найти соответствующие точки и на близких объектах.

«Расстояние» между прямыми.

Пусть $\rho'_{1,i}$, $\varphi'_{1,i}$ параметры i-ой преобразованной прямой первого изображения, а $\rho_{2,j}$, $\varphi_{2,j}$ - параметры j-ой прямой второго изображения. Через $C'_1(x,y)$ и $C_2(x,y)$ обозначены преобразованный контурный препарат первого изображения и контурный препарат второго изображения. Расстояние между i-ой и j-ой прямыми определяется как взвешенное знаковое расстояние между точками j-ой прямой и ближайшей точкой преобразованной i-ой прямой в пределах второго изображения; весом является произведение интенсивностей контурных препаратов $C'_1(x,y)$ и $C_2(x,y)$:

$$\begin{aligned} \bar{b}_{ij} &= \int b(x,y) C'_1(x,y) C_2(x,y) dl \\ b(x,y) &= x\cos\varphi'_{1,i} + y\sin\varphi'_{1,i} - \rho'_{1,i} \end{aligned} \tag{82}$$

Интегрирование производится вдоль j-ой прямой второго изображения, уравнение которой имеет вид:

$$x\cos\varphi_{2,j} + y\sin\varphi_{2,j} - \rho_{2,j} = 0 \tag{83}$$

Средний квадрат «расстояния» между прямыми определяется следующим образом:

$$\overline{b}^{2}{}_{ij} = \int b^2(x,y)C'_1(x,y)C_2(x,y)dl \tag{84}$$

Дисперсия расстояния:

$$\sigma^{2}{}_{ij} = \overline{b}^{2}{}_{ij} - (\overline{b}_{ij})^2 \tag{85}$$

Для j-ой прямой второго изображения определяется соответствующая прямая на первом изображении, которая минимизирует $\overline{b}^{2}{}_{ij}$ и это минимальное расстояние обозначается через $\overline{b}^{2}{}_{j}$:

$$\overline{b}^{2}{}_{j} = \min_i \overline{b}^{2}{}_{ij} \tag{86}$$

Обозначим через m^2 медиану величин $\overline{b}^{2}{}_{j}$, а через σ^2 медиану величин $\sigma^{2}{}_{j}$, где $\sigma^{2}{}_{j}$ есть дисперсия расстояния между j-ой прямой и соответствующей ей преобразованной прямой первого изображения:

$$\begin{aligned} m^2 &= \operatorname{median}_j \overline{b}^{2}{}_{j} \\ \sigma^2 &= \operatorname{median}_j \overline{\sigma}^{2}{}_{j} \end{aligned} \tag{87}$$

Размер w_s окна поиска соответствующей точки на втором изображении полагается равным:

$$w_s = \sqrt{m^2 + 3\sigma^2} \tag{88}$$

6. ОБЩИЙ АЛГОРИТМ ПОИСКА ПРОЕКТИВНОГО ПРЕОБРАЗОВАНИЯ

Ниже приведен объединенный алгоритм поиска проективного преобразования, совмещающего изображения

Вход: пара изображений цветных или черно-белых

Выход: проективное преобразование, переводящее первое изображение во второе; преобразованное первое изображение

Алгоритм:

- *Вычислить контурные препараты для обоих изображений (раздел 2.2, (15))*
- *Вычислить преобразование Хафа (Радона) для контурных препаратов (раздел 2.4, Алгоритм 2)*
- *Определить параметры прямых на обоих изображениях (раздел 2.4, Алгоритм 3)*
- *Составить цветовые дескрипторы для прямых линий (разделы 4.3, 4.4, Алгоритм 5)*
- *Найти локальные минимумы функционала (разделы 4.1, 4.2, Алгоритм 4), которые соответствуют различным гипотезам о проективных преобразованиях*
- *Выбрать лучшее проективное преобразование из полученных гипотез (начало раздела 4) и преобразовать им первое изображение.*

7. РЕЗУЛЬТАТЫ

Результаты работы алгоритма приведены на рис.8-11. Можно видеть, что поставленная задача – сближение соответствующих точек - выполнена удовлетворительно. Чем лучше выполняется предположение о том, что сцена плоская, тем лучше сближение соответствующих точек.

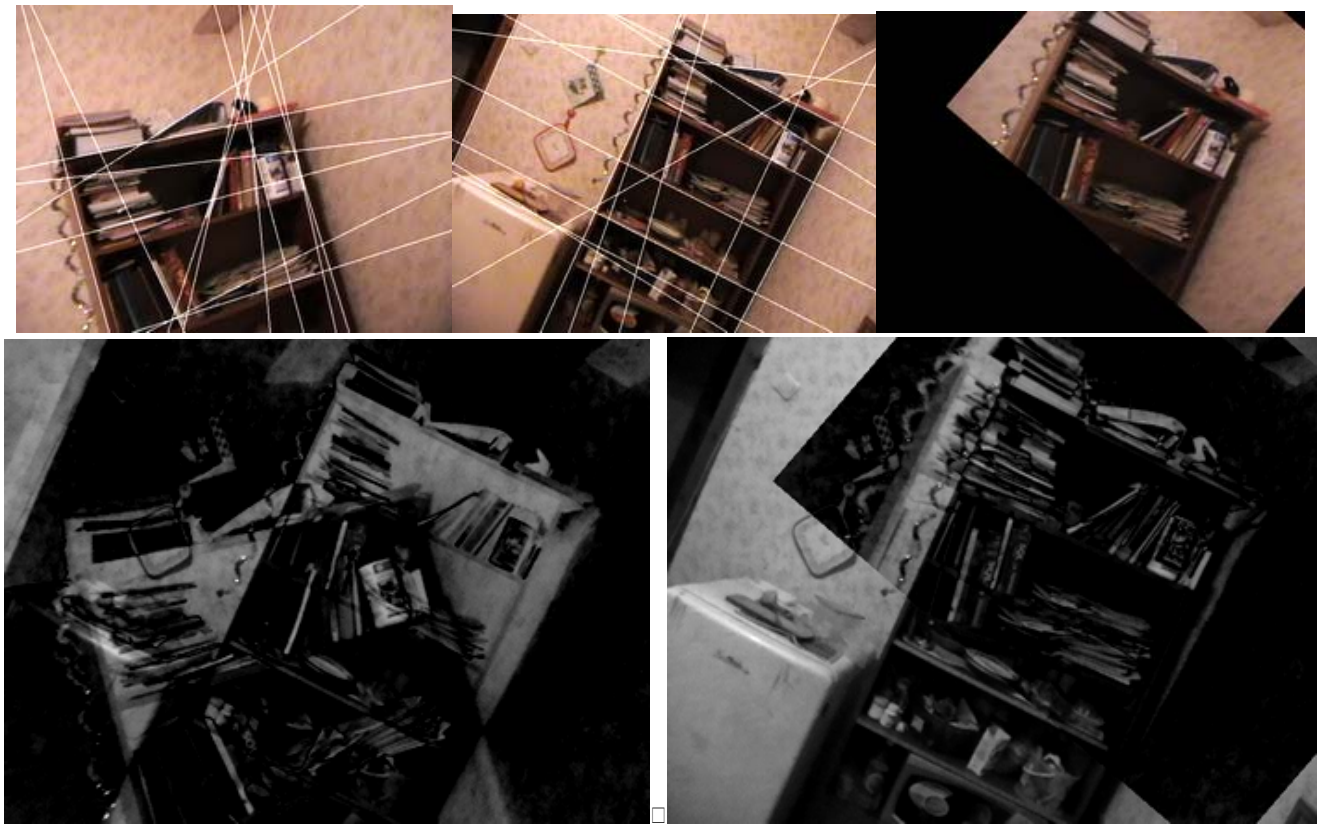

Рис. 8. Сцена «Шкаф».

Верхняя строка: слева - первое изображение, посередине - второе изображение, справа -преобразованное первое изображение. Нижняя строка: слева - разность первого и второго изображений, справа разность второго и преобразованного первого изображения

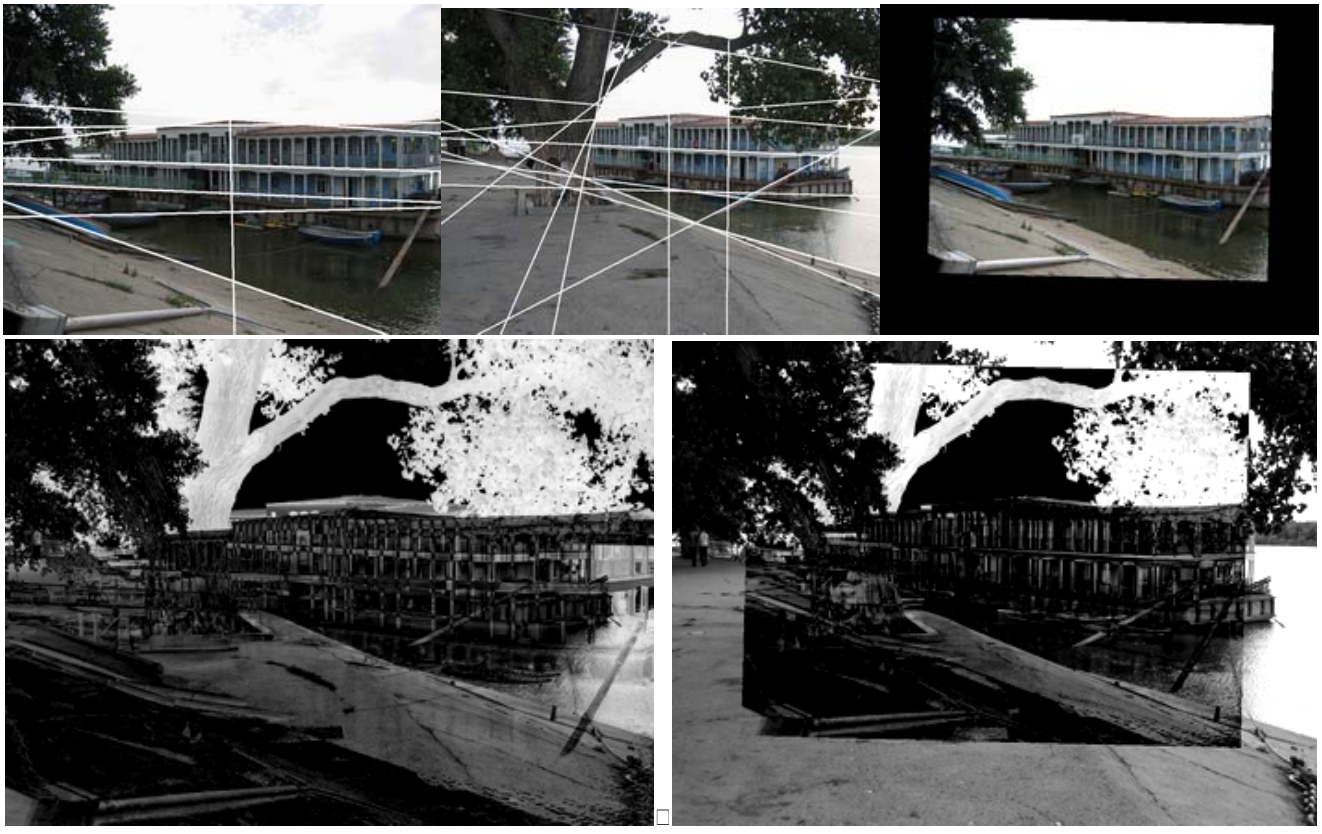

Рис. 9. Сцена «Пристань». Легенда та же, что и на рис. 8

Рис. 10. Сцена «Компьютер». Легенда та же, что и на рис. 8

Рис. 11. Сцена «Сейф». Легенда та же, что и на рис. 8

Ниже приведены времена работы для машины Intel Pentium III, 800 МГц.

Время работы алгоритма $t_{a\lg}$ складывается из трех составляющих: время t_{radon} поиска прямых на изображении используя преобразование Радона, время t_{proj} поиска проективных преобразований, время t_{best} отбора лучшей гипотезы:

$$t_{a\lg} = t_{radon} + t_{proj} + t_{best} \tag{89}$$

Время t_{radon} зависит только от размера изображений: $t_{radon} = O(N \log N)$, где N есть число точек на изображении. Для изображения 1024×1024 время t_{radon} составляет около 1 секунды.

Время t_{proj} включает в себя поиск локальных минимумов функционала. Оно не зависит от размеров изображения и определяется числами N_1 и N_2 найденных прямых: $t_{proj} = O(N_1 N_2)$ и диапазоном, в котором ищутся параметры преобразования. Для N_1, N_2 порядка 10-15 прямых и диапазоне параметров, приведенных в **Таблице 1**, время t_{proj} в среднем составляет 100 миллисекунд.

Таблица 1. Диапазон искомых параметров.

Угол поворота камеры, градусы	-180	180
Сдвиг камеры по горизонтали и вертикали, % размера изображения	-70	70
Масштабирование изображения, %	50	200

Время t_{best} зависит от числа найденных гипотез и размера изображений. Так как число гипотез обычно небольшое и фиксированное (3-5), то $t_{best} = O(N)$. Для изображения 1024×1024 время t_{best} порядка секунды.

8. ЗАКЛЮЧЕНИЕ

В настоящей работе предложен алгоритм грубого совмещения изображений, использующий информацию о найденных прямых линиях. Грубое совмещение облегчает установление соответствий между точками. Алгоритм учитывает как информацию о положении прямых, так и о цвете изображения вблизи прямых линий. Для поиска прямых линий предложен алгоритм вычисления преобразования Хафа через преобразование Хартли, что предъявляет меньшие требование к памяти и требует меньшего числа операций, чем известные подходы [34, 36, 37].

Алгоритм имеет следующие преимущества. Во-первых, прямые линии являются хорошо детектируемой особенностью, устойчивой к изменению освещенности сцены и перемещению камеры. Во-вторых, алгоритм способен справляться с ситуацией, когда на изображениях находятся разные прямые линии. В-третьих, промежуточным этапом алгоритма является поиск гипотез о возможных проективных преобразованиях, среди которых затем выбирается лучшая. Это значительно увеличивает вероятность найти «правильное» преобразование.

Получена оценка размера окна поиска соответствующих точек после грубого совмещения, что позволяет повысить эффективность работы алгоритмов поиска соответствующих точек и алгоритмов плотного восстановления 3D по нескольким кадрам. Грубое предварительное совмещение изображений открывает возможность обрабатывать изображения, полученные цифровым фотоаппаратом с рук алгоритмами восстановления формы и движения, которые рассчитаны на обработку видеопоследовательности и используют предположение о «близости» изображений на последовательных кадрах.

9. БЛАГОДАРНОСТИ

Работа выполнена при финансовой поддержке РФФИ, гранты 06-01-00789-а, 05-07-90345-в, 05-07-90390-в. Авторы выражают благодарность проф. Чатвину и доктору Янгу (Department of Engineering and Design,School of Science and Technology,University of Sussex, UK) за любезно предоставленный оттиск статьи [34], доценту ВМиК МГУ к.ф.-м.н Крылову А.С. и зав. кафедрой СИМ МФТИ проф. Клименко С.В. за проявленное к нашей работе внимание и плодотворные обсуждения.

10. СПИСОК ЛИТЕРАТУРЫ

1. R. Hartley, A. Zisserman. Multiple View Geometry in Computer Vision. *Cambridge University Press 2004, 672 p., ISBN: 0521540518.*
2. B. Srinivasa, B.N. Chatterji, "An FFT-based technique for translation, rotation, and scale-invariant image registration", *IEEE PAMI*, Vol 5(8), pp. 1266-1271, August, 1996.
3. Jianbo Shi and Carlo Tomasi, “Good features to track”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94) (Seattle)*, June 1994.
4. Y. Dufournaud, C. Schmid, and R. Horaud, "Matching images with different resolutions", *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR’00), 2000 (Hilton Head Island, SC, USA). -V. 1, -P. 612-618. ISBN: 0-7695-0662-3.* http://citeseer.ist.psu.edu/dufournaud00matching.html
5. S. Birchfield and C. Tomasi. Multiway cut for stereo and motion with slanted surfaces. *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Kerkyra, Greece, September 1999, pp. 489--495. http://citeseer.ist.psu.edu/birchfield99multiway.html.
6. C. G. Harris and M. Stephens, "A combined corner and edge detector," In Proc. 4th Alvey Vision Conf., Manchester, pages 147-151, 1988.
7. Tony Lindeberg: Scale-Space Theory in Computer Vision, Kluwer Academic Publishers, Dordrecht, Netherlands, 1994. См. также http://www.nada.kth.se/%7Etony/earlyvision.html.
8. C. Schmid, R. Mohr. Local Grayvalue Invariants for Image Retrieval. *//IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) 1997, -V. 19, -No. 5, -P. 530-534.* http://citeseer.ist.psu.edu/schmid97local.html
9. L.M.J. Florack, B.M. Ter Haar Romeny, J.J. Koenderink, and M.A. Viergever/ General Intensity Transformations and Differential Invariants. //Journal of Mathematical Imaging and Vision, vol. 4, no. 2, 171-187 (1994). http://www.bmi2.bmt.tue.nl/image-analysis/People/LFlorack/Extensions/Flor94d.pdf
10. M. Fischler and R. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM, -V. 24. -No. 6. –P.381–395, June 1981.
11. James Davis. Mosaics of scenes with moving objects. In Proc. Computer Vision and Pattern Recognition Conf., pages 354--360, 1998. http://citeseer.ist.psu.edu/davis98mosaics.html.
12. D. Capel, A. Zisserman. Computer Vision Applied to Super Resolution. //IEEE Signal Processing Magazine, May 2003. –P. 75—86. http://www.robots.ox.ac.uk/%7evgg/publications/html/index.html.
13. V. Kolmogorov and R. Zabih. Computing visual correspondence with occlusions using graph cuts. In *International Conference on Computer Vision*, Vancouver, Canada, July 2001. http://www.cs.cornell.edu/rdz/Papers/KZ-ICCV01-tr.pdf.
14. Michael H. Lin, Carlo Tomasi: Surfaces with Occlusions from Layered Stereo. //IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003.
15. D. Scharstein and R. Szeliski. A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms. IJCV 47(1/2/3):7-42, April-June 2002. http://cat.middlebury.edu/stereo/.

16. C.J. Poelman, T. Kanade. A Paraperspective Factorization Method for Shape and Motion Recovery. // IEEE Transactions on Pattern Analysis and Machine Intelligence, 1997. –V. 19. – No. 3. –P. 206—218. http://citeseer.ist.psu.edu/poelman92paraperspective.html.
17. Joao Paulo Salgado, Arriscado Costeira. A multi-body Factorization method for motion analysis: //Tese para obtencao do grau de doutor em Engenharia Electrotecnica e de Computadores. /Universitade Technica de Lisboa Instituto Superior Rechnico. Lisboa, Maio de 1995. http://omni.isr.ist.utl.pt/~jpc/pubs.html .
18. Peter Sturm, Bill Triggs. A Factorization Based Algorithm for Multi-Image Projective Structure and Motion: //4th European Conference on Computer Vision, Cambridge, England, April 1996, pp 709-720.
19. Н. В. Свешникова, Д. В. Юрин. Априорный и апостериорный расчет погрешностей восстановления трехмерных сцен алгоритмами факторизации. //Программирование 2004, Т.30, № 5, С. 48-68.
20. N.V. Sveshnikova, D.V. Yurin The Factorization Algorithms: Results Reliability and Application for the Epipolar Geometry Recovery. // In Conference Proceedings. 16-th International Conference on Computer Graphics and Application GraphiCon'2006 July 1—5, 2006 Novosibirsk Akademgorodok, Russia.
21. P. Pritchett, A. Zisserman. Wide Baseline Stereo Matching. //Proceedings of the 6th International Conference on Computer Vision, Bombay, India, Jan. 1998. –P. 754—760. http://www.robots.ox.ac.uk/%7evgg/publications/html/index.html.
22. Т. Кормен, Ч. Лейзерсон, Р. Ривест, К. Штайн. Алгоритмы. Построение и анализ. 2-е издание. Издательство: Вильямс, 2005 г. 1296 стр. ISBN 5-8459-0857-4, 0-07-013151-1
23. R.I. Hartley. Theory and Practice of Projective Rectification. //International Journal of Computer Vision, -V. 35, -No. 2, 12 November 1999, -P. 115-127.
24. Z. Zhang. Determining the epipolar geometry and its uncertainty: A review. Technical Report 2927, INRIA Sophia-Antipolis, France, July 1996. http://citeseer.ist.psu.edu/zhang98determining.html.
25. X. Armangué, J. Pagès, J. Salvi. Comparative Survey on Fundamental Matrix Estimation http://citeseer.ist.psu.edu/483847.html.
26. A. Baumberg. Reliable feature matching across widely separated views. //In Proc. CVPR, pages 774--781, 2000. http://citeseer.ist.psu.edu/baumberg00reliable.html.
27. K. Mikolajczyk, C. Schmid. Scale & Affine Invariant Interest Point Detectors. //International Journal of Computer Vision. –V. 60, -No. 1, –P. 63—86, October 2004. http://www.robots.ox.ac.uk/~vgg/research/affine/det_eval_files/mikolajczyk_ijcv2004.pdf.
28. D.G. Lowe. Distinctive image features from scale-invariant keypoints. //International Journal of Computer Vision 2004, -V. 60, -No. 2, -P. 91-110. http://www.cs.ubc.ca/spider/lowe/pubs.html.
29. V. Gouet, P. Montesinos and D. Pele. A fast matching method for color uncalibrated images using differential invariants. //In Proceedings of the British Machine Vision Conference, Southampton, 1998. http://citeseer.ist.psu.edu/article/gouet98fast.html.
30. G. Csurka, C. Zeller, Z. Zhang, O.D. Faugeras. Characterizing the Uncertainty of the Fundamental Matrix. //Computer Vision and Image Understanding: CVIU 1997, -V. 68, -No. 1, -P.18—36. http://citeseer.ist.psu.edu/csurka95characterizing.html.
31. J. Matas, O. Chum, U. Martin, T Pajdla. Robust wide baseline stereo from maximally stable extremal regions. //Proceedings of the British Machine Vision Conference, -V. 1, -P. 384—393, London, 2002. http://cmp.felk.cvut.cz/~matas/.
32. Princen J.P., Illingworth, J. and Kittler, J.V., “A Formal Definition of the Hough Transform: Properties and Relationships”, *Journal of Mathematical Imaging and Vision*, vol.1, num.1, pp.153-168, 1992

33. Toft P.A., *"The Radon Transform: Theory and Implementation"*, PhD Thesis, Technical University of Denmark, 1996
34. Cheyne Gaw Ho, Rupert C. D. Young, Chris D. Bradfield, Chris R. Chatwin, "A Fast Hough Transform for the Parametrisation of Straight Lines using Fourier Methods", *Real-Time Imaging*, vol.6, num.2, pp.113-127, 2000
35. Volegov D.B., Gusev V.V., Yurin D.V. "Straight Line Detection on Images via Hartley Transform. Fast Hough Transform, In *Conference Proceedings. 16-th International Conference on Computer Graphics and Application GraphiCon'2006*, July 1 - 5, 2006, Novosibirsk, Akademgorodok, Russia.
36. D.L. Donoho and X. Huo, *"Beamlets and multiscale image analysis"*, http://www-stat.stanford.edu/donoho/Reports/ http://citeseer.ist.psu.edu/donoho01beamlets.html
37. David Donoho and Xiaoming Huo, *"Applications of Beamlets to Detection and Extraction of Lines, Curves and Objects in Very Noisy Images"*, http://citeseer.ist.psu.edu/446366.html
38. Брэйсуэлл Р., *"Преобразование Хартли. Теория и приложения"*, Москва "Мир", 1990, ISBN 5-03-001632-5
39. Press et al, "*Numerical recipes in C"*, Cambridge University Press, 1992.
40. Zhengyou Zhang, "A Flexible New Technique for Camera Calibration", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, num. 11, pp. 1330-1334, 2000
41. Di Zenzo S. "A note on the gradient of multi-image", *Computer Vision Graphics Image Process*, Vol. 33. pp. 116-125, 1986
42. Klir G. and Bo Yuan, "*Fuzzy Sets and Fuzzy Logic",* 1995, ISBN 0-13-101171-5
43. Zimmermann H., "*Fuzzy Set Theory and its Applications",* 2001, ISBN 0-7923-7435-5.
44. N. Kiryati ,Y. Eldar and A.M. Bruckstein , *"A Probabilistic Hough Transform "*,Pattern Recognition vol.24, pp. 303-316,1991
45. L. Xu, E.Oja and P. Kultanen , "*A New Curve Detection Method: Randomized Hough Transform (RHT)*",Pattern Recognition Letters ,vol. 11,no. 5 , 1990,pp. 331-338
46. H. Kalvianen and P. Hirvonen ,"*Connective Randomized Hough Transform (CRHT)*", Proceedings of the 9th Scandinavian Conference on Image Analysis, Uppsala, Sweden ,June 1995 ,vol. 2,pp. 1029-1036
47. Thomas Theußl, Robert F. Tobler, Eduard Gröller. The Multi-Dimensional Hartley Transform as a Basis for Volume Rendering. //WSCG 2000 Conference Proceedings, http://citeseer.ist.psu.edu/450842.html, http://wscg.zcu.cz/wscg2000/Papers_2000/W11.pdf.gz.