СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Видео и фото тесно вплелись в нашу жизнь. Почти каждый мобильный телефон оснащен камерой. Почти каждая камера умеет записывать видео. Повсеместно распространилась 3D-графика. С развитием возможностей усиливается потребность в "дешевом" построении 3D-сцен. Самый очевидный из таких методов – стереозрение – получение трехмерной картины мира по видеоряду или нескольким изображениям.

Есть и другие возможные применения. В киноиндустрии – для создания спецэффектов и 3D-фильмов. В военном деле – для измерения расстояний. Как правило, камера расположена сбоку от монитора, где отображается лицо собеседника. Таким образом, другой участник смотрит куда-то в сторону. Построение трехмерного изображения лица позволяет синтезировать новое изображение, на котором собеседник смотрит прямо в глаза, что усиливает эффект присутствия.

Совмещение изображений, позволяет человеку получить информацию о расстоянии до объектов по их расхождениям (disparity). Эта идея может быть использована в алгоритмах обработки изображений. Хотя, конечно же, зрение человека активное (то есть параметры оптической системы настраиваются под изображение) – глаза вращаются в глазницах, меняется фокусное расстояние, как правило, построение системы активного зрения сложнее и дороже, чем зрения пассивного.

Система пассивного стереозрения, как правило, включает в себя 2 камеры и большинство алгоритмов решают общую задачу сопоставления 2-х изображений.

Существуют различные классификации алгоритмов сопоставления 2-х. Один из вариантов такой классификации представлен в [1]. Все алгоритмы делятся на локальные (в которых расхождение вычисляется в каждой точке на основе "похожести" окна вокруг этой точки и окна вокруг точек на другом изображении) и глобальные (основанные на ми-

нимизации функционала энергии – мы находим расхождение сразу для всех точек). Глобальные в свою очередь делятся по способу минимизации энергии. Как правило, это динамическое программирование [2, 3, 4, 5] или нахождение минимального разреза графа [6, 7]. Алгоритмы разреза графа называют также двумерными. Они дают довольно точные результаты, но имеют меньшую производительность. Алгоритмы, обрабатывающие строки изображений независимо друг от друга [2, 3], называют одномерными. Они работают быстрее, но подвержены эффекту гребенки, с которым борются с помощью разных ухищрений. Нечто среднее по производительности и качеству из себя представляют алгоритмы оптимизации на поддереве графа, построенного на пикселях изображения.

Многие алгоритмы построены на модели случайных полей Маркова (MRF). Основное предположение в такой модели: расхождение в любой точке зависит только от расхождений соседних точек (как правило, считают, что их 4, хотя можно соседними считать и 8 точек, оставаясь в рамках модели). Минимизация энергии в них производится на основе разрезов графа или распространения доверия [8].

Цель работы – анализ методов применяемых при построении объёмного изображения по стереопаре.

Задачи работы:

- описать термины предметной области и обозначить проблему;
- провести обзор существующих программных решений в области стерео зрения;
- выбрать критерии для их оценки и сравнить;
- выбрать наиболее предпочтительный метод.

1 Анализ предметной области

Как уже упомяналось выше, одна из основных задач стерео зрения – это получение объёмного изображения по нескольким изображениям.

Данную задачу можно разбить на 3 этапа:

- поиск соответствующих точек на изображениях;
- получение трехмерных координат точек;
- построение трехмерной модели.

1.1 Базовые понятия

В данном разделе описаны основные термины данной предметной области.

1.1.1 Проективная геометрия и однородные координаты

В геометрии стерео зрения значительную роль играет проективная геометрия. К проективной геометрии есть несколько подходов: геометрический (подобно Евклидовой геометрии ввести понятие геометрических объектов, аксиом и из этого выводить все свойства проективного пространства), аналитический (рассматривать все в координатах, как в аналитическом подходе к Евклидовой геометрии), алгебраический.

Ниже изложен аналитический подход к проективной геометрии.

Точки проективной плоскости. Рассмотрим двухмерное проективное пространство (которое еще называется проективной плоскостью). В то время как на обычной Евклидовой плоскости точки описываются парой координат $(x,y)^T$, на проективной плоскости точки описываются трехкомпонентным вектором $(x,y,w)^T$. При этом для любого ненулевого числа a, векторы $(x,y,w)^T$ и $(ax, ay, aw)^T$ соответствуют одной и той же

точке. А нулевой вектор $(0,0,0)^T$ не соответствует никакой точке и выкидывается из рассмотрения. Такое описание точек плоскости называется однородными координатами (homogeneous coordinates).

Точкам проективной плоскости можно сопоставить точки обычной Евклидовой плоскости. Координатному вектору $(x,y,w)^T$ при $w \neq 0$ сопоставим точку Евклидовой плоскости с координатами $(x/w, y/w)^T$. Если же $w = 0$, т.е. координатный вектор имеет вид $(x, y, 0)^T$, то говориться, что эта точка в бесконечности. Таким образом, проективную плоскость можно рассматривать как Евклидовую плоскость, дополненную точками из бесконечности.

Перейти от однородных координат $(x, y, w)^T$ к обычным Евклидовым можно путем деления координатного вектора на последнюю компоненту и последующего ее отбрасывания $(x,y,w)^T$ $(x/w,y/w)^T$. А от Евклидовых координат $(x,y)^T$ перейти к однородным можно за счет дополнения координатного вектора единичкой: $(x,y)^T$ $(x,y,1)^T$

Прямые на проективной плоскости. Любая прямая на проективной плоскости описывается, подобно точке, трехкомпонентным вектором $l = (a,b,c)^T$. Опять же вектор, описывающий прямую, определен с точностью до ненулевого множителя. При этом уравнение прямой будет иметь вид: $l^T x = 0$.

В случае, когда $a^2 + b^2 \neq 0$ мы имеем аналог обычной прямой $ax + by + c = 0$. А вектор $(0,0,w)$ соответствует прямой лежащей в бесконечности.

Трехмерное проективное пространство. По аналогии с проективной плоскостью, точки трехмерного проективного пространства определяются четырехкомпонентным вектором однородных координат $(x,y,z,w)^T$. Опять же для любого ненулевого числа a, координатные вектора $(x,y,z,w)^T$ и $(ax,ay,az,aw)^T$ соответствуют одной и той же точке.

Как в случае проективной плоскости, между точками трехмерного Евклидова пространства и трехмерного проективного пространства можно установить соответствие. Вектору однородных координат $(x,y,z,w)^T$

при $w \neq 0$ соответствует точка Евклодова пространства с координатами $(x/w,y/w,z/w)^T$. А про точку с вектором однородных координат вида $(x,y,z,0)^T$ говорят, что она лежит в бесконечности.

Проективное преобразование. (homography, projective transformation — в англ. литературе). С геометрической точки зрения, проективное преобразование — это обратимое преобразование проективной плоскости (или пространства), которое переводит прямые в прямые. В координатах, проективное преобразование выражается в виде невырожденной квадратной матрицы H, при этом координатный вектор x переходит в координатный вектор x' по следующей формуле: $x' = Hx$.

1.1.2 Модель проективной камеры

Современные CCD-камеры хорошо описываются с помощью следующей модели, называемой проективной камерой (projective camera, pinhole camera). Проективная камера определяется *центром камеры, главной осью* — лучом начинающимся в центре камеры и направленным туда, куда камера смотрит, *плоскостью изображения* — плоскостью на которую выполняется проецирование точек, и системой координат на этой плоскости. В такой модели, произвольная точка пространства X проецируется на плоскость изображения в точку x лежащую на отрезке CX, который соединяет центр камеры C с исходной точкой X (см. рис. 1.1).

Формула проецирования имеет простую математическую запись в однородных координатах:

$x = PX$

где X — однородные координаты точки пространства, x — однородные координаты точки плоскости, P — матрица камеры размера 3x4.

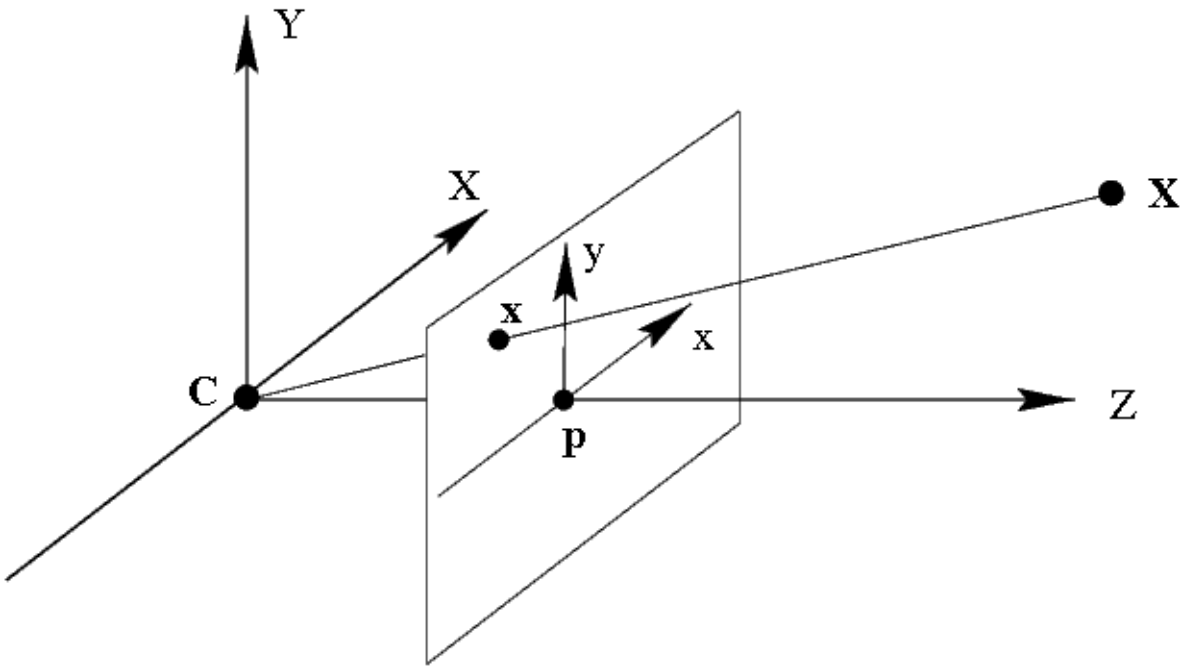

Рисунок 1.1 — **Модель камеры.** C — центр камеры, Cp — главная ось камеры. Точка X трехмерного пространства проецируется в точку x — на плоскости изображения.

Матрица P выражается следующим образом $P = KR[I| - \mathbf{c}] = K[R|\mathbf{t}]$, где K — верхняя треугольная матрица внутренних параметров камеры размера 3x3 (конкретный вид приведен ниже), R — ортогональная матрица размера 3x3, определяющая поворот камеры относительно глобальной системы координат, I — единичная матрица размера 3x3, вектор $\mathbf{c}$ — координаты центра камеры, а $\mathbf{t} = R\mathbf{c}$.

Стоит отметить, что матрица камеры определена с точностью до постоянного ненулевого множителя, который не изменит результатов проецирования точек по формуле $x = PX$. Однако этот постоянный множитель обычно выбирается так, что бы матрица камеры имела вышеописанный вид.

В самом простейшем случае, когда центр камеры лежит в начале координат, главная ось камеры сонаправлена оси Cz, оси координат на плоскости камеры имеют одинаковый масштаб (что эквивалентно квадратным пикселям), а центр изображения имеет нулевые координаты, матрица камеры будет равна $P = K[I|\mathbf{0}]$, где

$$K = \begin{pmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

У реальных CCD камер пикселы обычно незначительно отличаются от квадратных, а центр изображения имеет ненулевые координаты. В таком случае матрица внутренних параметров примет вид:

$$K = \begin{pmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{pmatrix}$$

Коэффициенты $f_{,x}{}_{,y}$ — называются фокусными расстояниями камеры (соответственно общим и вдоль осей x и y).

Помимо этого, в силу неидеальности оптики, на изображениях, полученных с камер, присутствуют искажения-дисторсии (distortion). Данные искажения имеют нелинейную математическую запись:

$$x'' = x'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + 2p_1 x' y' + p_2(r^2 + 2x'^2)$$

$$y'' = y'(1 + k_1 r^2 + k_2 r^4 + k_3 r^6) + p_1(r^2 + 2y'^2) + 2p_2 x' y'$$

где $k1, k2, p1, p2, k3$ — коэффициенты дисторсии, являющиеся параметрами оптической системы; $r2 = x'2 + y'2; (x', y')$ — координаты проекции точки относительно центра изображения при квадратных пикселях и отсутствии искажений; (x, y) — искаженные координаты точки относительно центра изображения при квадратных пикселях.

Дисторсии не зависят от расстояния до объекта, а зависят только от координат точек, в которые проецируются пиксели объекта. Соответственно для компенсации дисторсий обычно выполняется преобразование исходного изображения полученного с камеры. Это преобразование будет одним и тем же для всех изображений, полученных с камеры, при условии постоянства фокусного расстояния (математически — одной и той же матрицы внутренних параметров).

В ситуации, когда известны внутренние параметры камеры и коэффициенты дисторсии говорят, что камера откалибрована.

Об определении трехмерных координат наблюдаемых точек можно говорить, когда есть как минимум две камеры.

Матрицы пары камер, калибровка. Пусть имеются две камеры, заданные своими матрицами P и P' в некоторой системе координат. В таком случае говорят, что имеется пара откалиброванных камер. Если центры камер не совпадают, то эту пару камер можно использовать для определения трехмерных координат наблюдаемых точек.

Зачастую, система координат выбирается так, что матрицы камер имеют вид $P = K[I|0], P' = K'[R'|t']$. Это всегда можно сделать, если выбрать начало координат совпадающее с центром первой камеры, и направить ось Z вдоль ее оптической оси.

Калибровка камер обычно выполняется, за счет многократной съемки некоторого калибровочного шаблона, на изображении можно легко выделить ключевые точки, для которых известны их относительные положения в пространстве. Далее составляются и решаются (приближенно) системы уравнений, связывающие координаты проекций, матрицы камер и положения точек шаблона в пространстве.

Существуют общедоступные реализации алгоритмов калибровки, например, Matlab Calibration toolbox. Так же библиотека OpenCV включает в себя алгоритмы калибровки камер и поиска калибровочного шаблона на изображении.

1.1.3 Эпиполярная геометрия

Перед тем как перейти к описанию собственно метода вычисления трехмерных координат точек, я опишу некоторые важные геометрические свойства, связывающие положения проекций точки трехмерного пространства на изображениях с обеих камер.

Пусть имеются две камеры, как изображено на рисунке 1.2. C — центр первой камеры, C' — центр второй камеры. Точка пространства X проецируется в x на плоскость изображения левой камеры и в x' на плос-

кость изображения правой камеры. Прообразом точки x на изображении левой камеры является луч xX. Этот луч проецируется на плоскость второй камеры в прямую l', называемую эпиполярной линией. Образ точки X на плоскости изображения второй камеры обязательно лежит на эпиполярной линии l'.

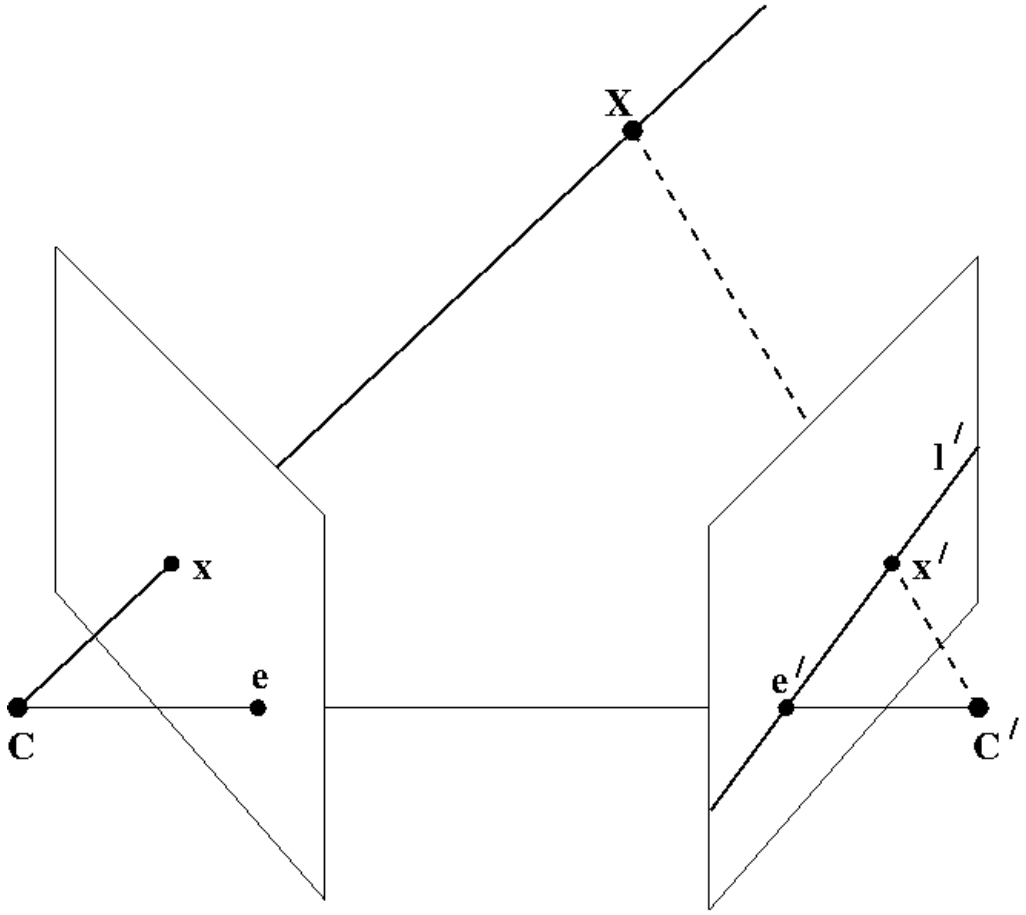

Рисунок 1.2 — Эпиполярная геометрия

Таким образом, каждой точке x на изображении левой камеры соответствует эпиполярная линия l' на изображении правой камеры. При этом пара для x на изображении правой камеры может лежать только на соответствующей эпиполярной линии. Аналогично, каждой точке x' на правом изображении соответствует эпиполярная линия l на левом.

Эпиполярную геометрию используют для поиска стереопар, и для проверки того, что пара точек может быть стереопарой (т.е. проекцией некоторой точки пространства).

Эпиполярная геометрия имеет очень простую запись в координатах. Пусть имеется пара откалиброванных камер, и пусть x — однородные координаты точки на изображении одной камеры, а x' — на изображении второй. Существует такая матрица F размера 3x3, что пара точек x, x' является стереопарой тогда и только тогда, когда:

$$x'^T F x = 0$$

Матрица F называется фундаментальной матрицей (fundamental matrix). Ее ранг равен 2, она определена с точностью до ненулевого множителя и зависит только от матриц исходных камер P и P'.

В случае, когда матрицы камер имеют вид $P = K[I|0], P' = K'[R|t]$ фундаментальная матрица может быть вычислена по формуле:

$$F = K'^{-1^T} R K^T [K R^T t] x$$

где для вектора e обозначение $[e]_X$ вычисляется как

$$[e]_X = \begin{bmatrix} 0 & -e_z & e_y \\ e_z & 0 & -e_x \\ -e_y & e_x & 1 \end{bmatrix}$$

С помощью фундаментальной матрицы вычисляются уравнения эпиполярных линий. Для точки x, вектор, задающий эпиполярную линию, будет иметь вид $l' = Fx$, а уравнение самой эпиполярной линии: $l'^T x' = 0$. Аналогично для точки x', вектор, задающий эпиполярную линию, будет иметь вид $l = F^T x'$.

Помимо фундаментальной матрицы, существует еще такое понятие, как существенная матрица (essential matrix): $E = K'^T F K$. В случае, когда матрицы внутренних параметров будут единичными существенная матрица будет совпадать с фундаментальной. По существенной матрице можно восстановить положение и поворот второй камеры относительно первой, поэтому она используется в задачах, в которых нужно определить движение камеры.

1.1.4 Триангуляция точек

Процесс определения трехмерных координат точки по координатам ее проекций в литературе называется триангуляцией (triangulation).

Пусть имеются две откалиброванные камеры с матрицами P_1 и P_2. x_1 и x_2 — однородные координаты проекций некоторой точки пространства X. Тогда можно составить следующую систему уравнений:

$$\begin{cases} x_1 = P_1 X \\ x_2 = P_2 X \end{cases}$$

На практике для решения этой системы применяется следующий подход. Векторно умножают первое уравнение на x_1, второе на x_2, избавляются от линейно зависимых уравнений и приводят систему к виду $AX = 0$, где A имеет размер 4x4. Далее можно либо исходить из того вектор X является однородными координатами точки, положить его последнюю компоненту равной 1 и решать полученную систему из трёх уравнений с тремя неизвестными. Альтернативный способ — взять любое ненулевое решение системы $AX = 0$, например вычисленное, как сингулярный вектор, отвечающий наименьшему сингулярному числу матрицы A.

1.1.5 Карта глубины

Карта глубины (depth map) — это изображение, на котором для каждого пикселя, вместо цвета, храниться его расстояние до камеры. Карта глубины может быть получена с помощью специальной камеры глубины (например, сенсор Kinect является своего рода такой камерой), а так же может быть построена по стереопаре изображений.

Идея, лежащая в основе построения карты глубины по стереопаре очень проста. Для каждой точки на одном изображении выполняется поиск парной ей точки на другом изображении. А по паре соответствующих точек можно выполнить триангуляцию и определить координаты

их прообраза в трехмерном пространстве. Зная трехмерные координаты прообраза, глубина вычисляется, как расстояние до плоскости камеры.

Парную точку нужно искать на эпиполярной линии. Соответственно, для упрощения поиска, изображения выравнивают так, что бы все эпиполярные линии были параллельны сторонам изображения (обычно горизонтальны). Более того, изображения выравнивают так, что бы для точки с координатами (x_0, y_0) соответствующая ей эпиполярная линия задавалась уравнением $x = x_0$, тогда для каждой точки соответствующую ей парную точку нужно искать в той-же строчке на изображении со второй камеры. Такой процесс выравнивания изображений называют ректификацией (rectification). Обычно ректификацию совершают путем ремеппинга изображения и ее совмещают с избавлением от дисторсий. Пример ректифицированных изображений приведен на рисунке 1.3.

Рисунок 1.3 — Пример ректифицированных картинок, и соответствующей им disparity map.

После того как изображения ректифицированы, выполняют поиск соответствующих пар точек. Самый простой способ проиллюстрирован на рисунке 1.4 и состоит в следующем. Для каждого пикселя левой картинки с координатами (x_0, y_0) выполняется поиск пикселя на правой картинке. При этом предполагается, что пиксель на правой картинке должен иметь координаты $(x_0 - d, y_0)$, где d — величина называемая несоответствие/смещение (disparity). Поиск соответствующего пикселя выполняется путем вычисления максимума функции отклика, в качестве которой может выступать, например, корреляция окрестностей пикселей. В результате получается карта смещений (disparity map), пример которой приведен на рис. 1.3.

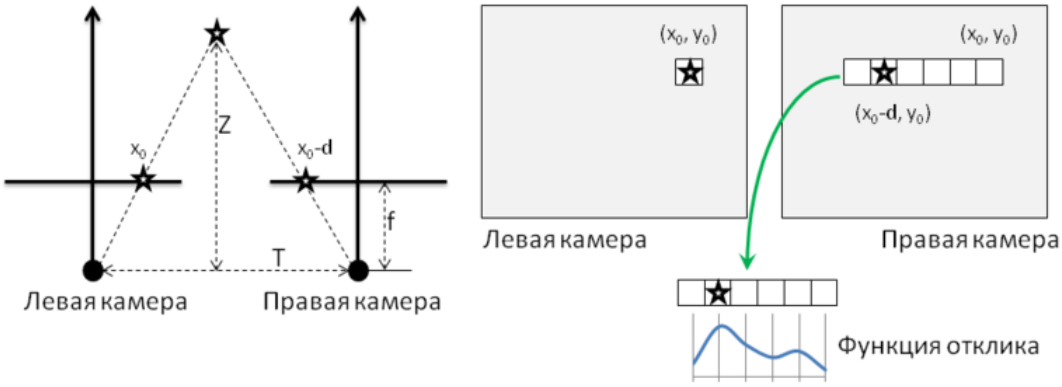

Рисунок 1.4 — Вычисление карты глубины

Собственно значения глубины обратно пропорциональны величине смещения пикселей. Если использовать обозначения с левой половины рисунка 1.4, то зависимость между disparity и глубиной можно выразить следующим способом:

$$\frac{T-d}{Z-f} = \frac{T}{Z} \rightarrow Z = \frac{fT}{d}$$

Из-за обратной зависимости глубины и смещения, разрешающая способность систем стерео зрения, которые работают на основе данного метода, лучше на близких расстояниях, и хуже на далеких.

2 Существующие решения

2.1 Алгоритм RANSAC

Алгоритм RANdom SAmple Consensus (RANSAC), предложенный Фишлером и Боллес [9], представляет собой стабильный метод оценки параметров модели на основе случайных выборок устойчевый к зашумлённости исходных данных (большая доля выбросов).

Часто возникает задача обработки данных, в которой необходимо определить параметры модели, которая должна удовлетворять исходным данным. Все исходные данные можно разделить на два типа: хорошие точки, удовлетворяющие модели, «не-выбросы» или «инлаеры» (англ. inlier) и ложные точки, шумы — случайные включения в исходные данные, «выбросы» или «аутлаеры» (англ. outlier).

RANSAC – это итерационный алгоритм, который пределагает решения, используя минимальный набор данных, необходимых для оценки базовых параметров модели. Как указали Фишлер и Боллес, в отличие от обычных методов выборки, которые используют как можно больше данных для получения начального решения, а затем приступают к сокращению выбросов, RANSAC использует наименьший возможный набор и продолжает расширять его совместимыми точками из исходных данных.

На вход алгоритма поступают:

1) набор исходных данных X;

2) функция M, позволяющая вычислить параметры θ модели P по набору данных из n точек;

3) функция оценки E соответствия точек полученной модели;

4) порог t для функции оценки;

5) количество итераций метода k.

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

Первый этап — выбор точек и подсчёт модели:

• из множества исходных точек X случайным образом выбираются n различных точек;

• на основе выбранных точек вычисляются параметры θ модели P с помощью функции M, построенную модель принято называть гипотезой.

Второй этап — проверка гипотезы:

• для каждой точки проверяется её соответствие данной гипотезе с помощью функции оценки E и порога t;

• каждая точка помечается инлаером или выбросом;

• после проверки всех точек, проверяется, является ли гипотеза лучшей на данный момент, и если является, то она замещает предыдущую лучшую гипотезу.

В конце работы цикла оставляется последняя лучшая гипотеза.

Результатом работы метода являются:

1) параметры θ модели P;

2) точки исходных данных, помеченные инлаерами или выбросами.

Значение параметра t должно быть определено в зависимости от конкретных требований, зависящих от данных, в большинстве случаев, только после экспериментальных оценок. Количество итераций k выбирается достаточно большим, чтобы гарантировать, что по крайней мере один из наборов случайных выборок не содержит выбросов. Определяется оно методом теоретической оценки. Пусть p — вероятность того, что алгоритм RANSAC на некоторой итерации, выбирая n точек, на основе которых строится модель, возьмёт для расчётов из исходного набора данных только инлаеры. В такой ситуации построенная по данным точкам модель, с большой вероятностью будет достаточно точной. Исходя из этого, мы можем использовать вероятность p для оценки точности рабо-

ты алгоритма. Пусть ω – ероятность выбора одного инлаера из общего числа точек, то есть $\omega = I/T$ – количество инлаеров, T – общее число точек. В большинстве случаев доля инлаеров ω неизвестна до начала выполнения алгоритма, но практически всегда можно дать некоторую грубую оценку. Вероятность независимого выбора n инлаеров из исходных данных, в таком случае равна $q = C_I^n/C_T^n = I!(T-n)!/(T!(I-n)!)$, а вероятность того, что хотя бы одна точка из набора выброс, то есть что будет построена некорректная модель – $(1-q)$. Вероятность того, что за k итераций алгоритм ни разу не выберет n инлаеров – $(1-q)^k$, такая ситуация означает, что точная модель не будет построена, а вероятноть этого события равна $(1-p)$. Таким образом

$$1 - p = (1 - q)^k$$

Выразим необходимое нам количество итераций k:

$$k = \frac{log(1-p)}{log(1-q)}$$

Преимуществом алгоритма RANSAC является его способность дать надёжную оценку параметров модели, то есть возможность оценить параметры модели с высокой точностью, даже если в исходном наборе данных присутствует значительное количество выбросов.

Одним из недостатков метода RANSAC является отсутствие верхней границы времени, необходимого для вычисления параметров модели. Если использовать в качестве некоторой границы времени максимальное число итераций, полученное решение может быть не оптимальным, а также существует очень малая вероятность, что ни одна модель не будет соответствовать исходным данным. Точная модель может быть определена с некоторой вероятностью, которая становится больше, чем больше итераций, которые используются. Ещё одним недостатком метода RANSAC является то, что для выполнения алгоритма необходимо задать конкретное пороговое значение. Наконец методом RANSAC можно определить только одну модель для определённого набора данных. Как и для любого подхода, предназначенного для одной модели, существует

следующая проблема: когда в исходных данных присутствуют две (или более) модели, RANSAC может не найти ни одну.

Алгоритм RANSAC часто используется в компьютерном зрении, например, для решения задачи сопоставления изображений и оценки фундаментальной матрицы для определения параметров расположения камеры.

2.2 Алгоритм SIFT

Масштабно-инвариантная трансформация признаков (англ. scale-invariant feature transform, SIFT) это алгоритм компьютерного зрения, используемый для выявления и описания локальных признаков в изображениях. Этот алгоритм был опубликован Дэвидом Лоу в 1999 году и усовершенствован в 2004 году.

Сначала в SIFT извлекаются ключевые точки объектов из набора контрольных изображений [10] и запоминаются в базе данных. Объект распознаётся в новом изображении путём сравнивания каждого признака из нового изображения с признаками из базы данных и нахождения признаков-кандидатов на основе евклидова расстояния между векторами признаков. Из полного набора соответствий в новом изображении отбираются поднаборы ключевых точек, которые наиболее хорошо согласуются с объектом по его местоположению, масштабу и ориентации. Определение подходящих блоков признаков осуществляется быстро с помощью эффективной реализации хеш-таблицы обобщённого преобразования Хафа. Каждый блок из 3 или более признаков, согласующийся с объектом и его положением, подлежит дальнейшей подробной проверке соответствия модели, и резко отклоняющиеся блоки отбрасываются. Наконец, вычисляется вероятность, что определённый набор признаков говорит о присутствии объекта, что даёт информацию о точности совпадения и числе возможных промахов. Объекты, которые проходят все эти тесты, могут считаться правильными с высокой степенью уверенности [11].

Лоу разбивает алгоритм SIFT на следующие четыре шага:

• выявление экстремумов масштабного пространства: поиск ключевых (в рамках алгоритма SIFT) точек;

• локализация ключевых точек: отбрасывание точек с низкой контрастностью или расположенных не вдоль ребёр;

• определение направления: присваивание одного или нескольких направлений для каждой ключевой точки на основе локальных направлений градиента изображения;

• дескрипторы ключевых точек: вычисление дескриптора для каждой ключевой точки.

Ниже перечислены некоторые применения данного алгоритма:

• распознавание объектов;

• определение местоположения и отслеживание техники/роботов;

• объеденение изображений;

• моделирование 3D-сцен, распознавание и трассировка;

• распознавания действий человека;

• анализ человеческого мозга в трёхмерных изображениях Магнитно-резонансной томографии.

2.3 Алгоритм SURF

Алгоритм SURF является улучшение алгоритма SIFT. Стандартная версия SURF в несколько раз быстрее, чем SIFT, и, как утверждают ее авторы [12], более устойчива к различным преобразованиям изображений.

Основным результатом алгоритма SURF являются обнаруженные ключевые точки, которые должны иметь следующие свойства: повторяемость (для обеспечения возможности поиска такой же точки на следующем кадре); инвариантность к повороту и масштабу; уникальность.

Алгоритм метода SURF. Алгоритм работы SURF предусматривает выполнение следующих этапов:

- масштабно-пространственное представление;
- расчет значений гессиана;
- поиск точек локальных максимумов;
- определение точки истинного максимума;
- определение ориентации опорной точки;
- формирование дескриптора опорной точки.

Обнаружение особых точек алгоритмом SURF основано на вычислении детерминанта матрицы Гессе (гессиана H) [7]. Значение гессиана используется для нахождения локального минимума или максимума яркости изображения. В этих точках значение гессиана достигает экстремума. Расчет производных происходит с помощью свертки пикселей изображения с фильтрами, представленными на рисунке 2.1, где белые области соответствуют значению +1, черные – значению -2 (на третьем фильтре – значению -1), серые – нулю [13].

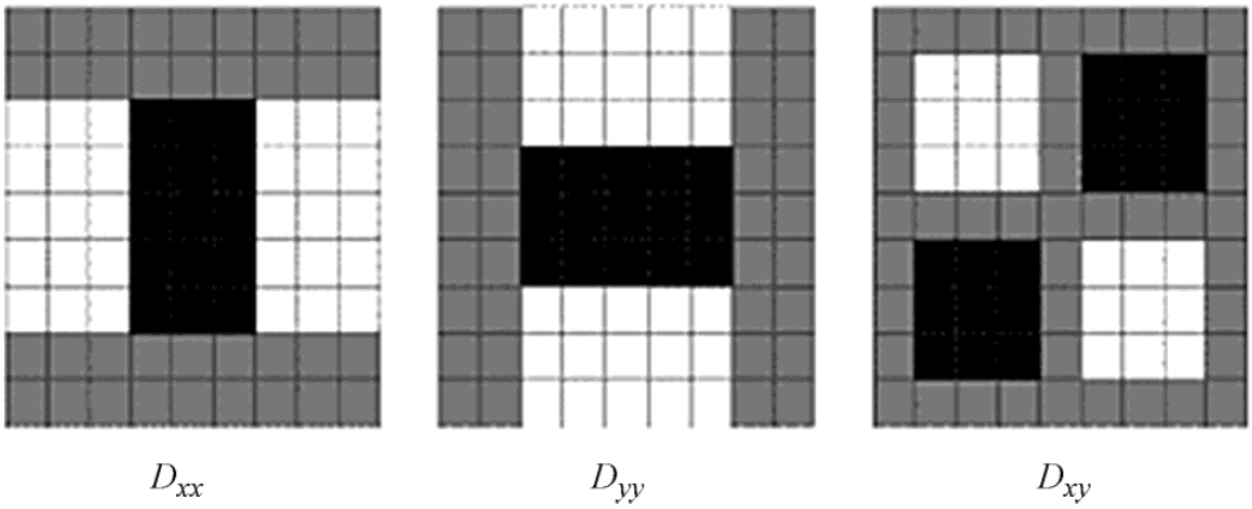

Рисунок 2.1 — Фильтры, используемые в алгоритме

Таким образом, в алгоритме SURF гессиан вычисляется следующим образом:

$$det(h) = D_{xx}D_{yy} - (0{,}9D_{xy})^2,$$

где D_{xx}, D_{yy}, D_{xy} – свертки по фильтрам (см. рис. 2.1); коэффициент 0,9 корректирует приближенный характер вычислений.

Следовательно, в качестве особых точек выбираются локальные максимумы гессианов, соответствующие локальным максимумам изменения градиента яркости. После нахождения точек локальных максимумов определяется точка истинного максимума гессиана. На этом шаге этап детектирования окончен. Дескриптор представляет собой массив из 64 чисел, которые определяют опорную точку. Дескриптор SURF инвариантен к масштабу и вращению.

Данный алгоритм можно использовать для таких задач, как распознавание объектов, регистрация изображений, классификация или 3D-реконструкция.

2.4 Метод поиска соответствующих точек, использующий DSI

Рассмотрим подробнее одномерный метод поиска соответствующих точек, использующий DSI (*disparity space image*) [4] Пусть s_i и s'_i – соответствующие i-е строки левого и правого изображений, $I(x,y)$ и $I'(x,y)$ – соответствующие функции интенсивности. Тогда параметр DSI_i вычисляется как разность интенсивностей:

$$DSI_i(x,d) = ||I(x,i) - I'(x-d,i)||,$$

где N – ширина картинки; d_{max} – максимальное допустимое расхождение на изображении; при этом выполняются условия $-d_{max} \leqslant d \leqslant d_{max}$ и $0 \leqslant (x+d) \leqslant N$.

Другим способом построения DSI_i является вычисление величины, основанной на корреляции функций интенсивностей вокруг пикселей (x,i) и $(x-d,i)$. При этом размеры окна выбираются адаптивно (например, если точка находится на границе объектов, то окно с центром в этой точке будет плохим, лучше использовать окно, которое полностью принадлежит одному объекту). Далее отбрасываем строки DSI_i, которые соответствуют заведомо невозможным значениям d (например, из условий, что пиксель на левом изображении должен быть правее соответствующего ему на правом, из условия $|d| \leqslant d_{max}$) [14].

Задача сводится к поиску оптимального пути на полученной двумерной матрице. При этом за каждый тип движения назначается определенный штраф. Используются типы движения по горизонтали, вертикали и по диагонали. Последние два типа соответствуют заслоненным областям (присутствующим только на одном из изображений стереопары). Отметим, что если некоторая поверхность на левом изображении имеет ширину 9 пикселей, а на правом – 3 пикселя, то в этой терминологии трем пикселям из изображения слева соответствуют 3 пикселя справа (это могут быть, например, третий, шестой и восьмой), остальные 6 пикселей считаются заслоненными.

Задача решается методом динамического программирования. Если на изображении заранее известны GCP-точки (*ground control points* – точки, положение и соответствие которых мы можем определить достаточно точно), то количество возможных путей сокращается за счет использования следующего ограничения. Путь должен проходить так, чтобы он не противоречил расстановке базовых GCP-точек. Возможны также многозначные GCP-точки, когда точке одного изображения может соответствовать одна точка из небольшого набора вариантов на другом изображении.

3 Сравнение

По скольку алгоритм SURF базируется на методе SIFT, сравним их первыми.

На рисунке 3.1 представлены результаты метода SIFT и его сравнение с результатами методом SURF. Метод SIFT дает больше характерных точек и более точную их локализацию, однако имеет более медленное исполнение.

Рисунок 3.1 — Сравнение результатов методов SIFT и SURF

На рисунке изображены:

- (слева) пример фотографии, полученной при аэрофотосъёмке;
- (справа) результат работы SIFT на данной фотографии;
- (по центру) результат работы SURF на данной фотографии [15].

Время работы SIFT в 2.5 раза больше, чем время работы SURF, однако SIFT при этом находит больше точек [15]. Недостатки SURF проявляются на сильно размытых фотографиях [16, 17], при сшивке разномасштабных фотографий и при смене угла обзора.

В общем случае алгоритм SIFT является более надежным, в то время как SURF - более быстрым.

На таблице 3.1 представлено сравнение рассмотренных методов поиска соответсвующих точек по производительности в реальном времени, точности локализации ключевых точек и по наличию шагу вычисления дескриптора.

Таблица 3.1 — Сравнение алгоритмов поиска соответсвующих точек

Алгоритм	Производ.	Точность	Вычисление дескр.
SIFT	средняя	высокая	да
SURF	высокая	средняя	да
Метод исп. DSI	-	средняя	нет

По сравнению с методом использующим DSI, алгоритмы SIFT и SURF используют базы данных, и поэтому могут найти объект на новом изображении, в то время когда первому методу всегда требуется 2 изображения. Кроме этого, SIFT и SURF позволяют не только представить трёхмерную модель объекта, но и распознать его.

Если же сравнивать алгоритмы SIFT и SURF, то наиболее предподчтиельным оказывается SURF, т.к. за небольшой объём вычислений, с более высокой скоростью получаются почти те же ключевые точки, как у метода SIFT

Алгоритм RANSAC исключен из сравнения, т.к. он решает другую задачу. Алгоритм RANSAC можно использовать в совокупности с одним из выше перечисленных для фильтрации полученных сопоставлений, потому что иногда сопоставляемые точки могут быть соотнесены некорректно, а RANSAC отсеивает «выпадающие» (outliers) из статистики точки.

ЗАКЛЮЧЕНИЕ

По итогу проделанной работы была достигнута цель - проведён анализ методов применяемых при построении объёмного изображения по стереопаре.

Также были решены все поставленные задачи, а именно:

- описаны термины предметной области и обозначена проблема;
- проведён обзор существующих программных решений в области стерео зрения;
- выбраны критерии для их оценки и проведено сравнение;
- выбран наиболее предпочтительный метод.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Scharstein D., Szeliski R. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms // Int. Journal of Computer Vision 47. April-June 2002. С. 7–42.

2. Bobick A. F., Intille S. S. Large occlusion stereo // Int. Journal of Computer Vision. 33(3). 1999. С. 181–200.

3. Ohta Y., Kanade T. Stereo by intra- and inter- scanline search using dynamic programming // IEEE TPAMI. 7(2). 1985. С. 139–154.

4. Veksler O. Stereo correspondence by dynamic programming on a tree // Proc. CVPR. Vol. 2. 2005. С. 384–390.

5. Naveed I. R., Huijun Di, GuangYou Xu Refine stereo correspondence using bayesian network and dynamic programming on a color based minimal span tree // ACIVS. 2006. С. 610–619.

6. Kolmogorov V., Zabih R. Computing visual correspondence with occlusions using graph cuts // ICCV. Vol. 2. 2001. С. 508–515.

7. Boykov Y., Veksler O., Zabih R. Fast approximate energy minimization via graph cuts // IEEE TPAMI 23(11). 2001. С. 1222–1239.

8. Sun J., Shum H., Zheng .N Stereo matching using belief propagation // In ECCV. 2002. С. 510–524.

9. Fischler M. A. and Bolles R. C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. // Commun. ACM 24. 1981. С. 381-395

10. Lowe D. G. Object recognition from local scale invariant features // Proceedings of the Seventh IEEE International Conference on Computer Vision 2. 1999. С. 1150-1157

11. Lowe D. G. Distinctive Image Features from Scale-Invariant Keypoints. // International Journal of Computer Vision. 2004. С. 91-110

12. Bay H., Ess A., Tuytelaars T., Gool L.V. Speeded-Up Robust Features (SURF). //Computer Vision and Image Understanding Volume 110. 2008. 346-359.

13. Гаврилов Д.А., Павлов А.В. Поточная аппаратная реали-зация алгоритма SURF // Известия высших учебных заведений Электроника. 2018. № 5. С. 502–511.

14. Тупицын И. В. Реконструкция трехмерной модели объекта на основе стереопары при решении задач 3D-моделирования. // Сибирский аэрокосмический журнал. 2011. С. 88-92.

15. Карпов, Д.П. Сшивка изображений, полученных в результате аэрофотосъёмки [Электронный ресурс] / Д.П. Карпов. - СПб.: НИУ ИТМиО, 2012. - Режим доступа: http://is.ifmo.ru/projects/2012/karpov/description.pdf, свободный. - Загл. с экрана.

16. Bay, H. SURF: Speeded up robust features / H. Bay, T. Tuytealaars, L.V. Gool // Proc. of European Conference on Computer Vision, ECCV'2006. - 2006. - P. 404-417.

17. Khan, N. SIFT and SURF Performance Evaluation Against Various Image Deformations on Benchmark Dataset / N. Khan, B. McCane, G. Wyvill // Proc. of Digital Image Computing Techniques and Applications (DICTA). - 2011. - P. 501-506.