**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра CАУ**

отчет

**по практике**

**Тема: Алгоритмы поиска и распознавания объектов на двумерных изображениях**

|  |  |  |
| --- | --- | --- |
| Студент гр. 8491 |  | Каримжонов Х.Т. |
| Руководитель |  | Никоза А.В. |

Санкт-Петербург

2022

**ЗАДАНИЕ**

**на практику**

|  |  |  |
| --- | --- | --- |
| Студент Каримжонов Х.Т. | | |
| Группа 8491 | | |
| Тема практики: Алгоритмы поиска и распознавания объектов на двумерных изображениях | | |
| Задание на практику:  Составить план для ВКР и формулировать решаемых задач. | | |
| Сроки прохождения практики: 13.01.2022 – 13.02.2022 | | |
| Дата сдачи отчета: | | |
| Дата защиты отчета: | | |
|  | | |
| Студент |  | Каримжонов Х.Т. |
| Руководитель |  | Никоза А.В |

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 4 |
| 1. | Обзор алгоритмов распознания объектов на двумерных изображениях | 5 |
| 2. | Алгоритм поиска максимума кросскорреляционной функции двух изображений | 6 |
| 2.1. | Модификация алгоритма поиска максимума дискретной кросскорреляционной функции от двух изображений | 7 |
| 2.2. | Модификации алгоритма поиска максимума кросскорреляционной функции. | 8 |
| 3. | Обнаружение устойчивых признаков изображения: метод SURF | 9 |
| 3.1. | Градиент в точке вычисляется с помощью фильтров Хаара. | 10 |
| 3.2. | Интегральное представление | 11 |
| 3.3. | Вычисление матрицы8 Гессе | 12 |
| 3.4. | Шкалы | 14 |
| 3.5. | Нахождение локального максимума гессиана | 16 |
| 3.6. | Нахождение ориентации особой точки | 18 |
| 3.7. | Вычисление дескриптора особой точки | 20 |
|  | Заключение | 23 |
|  | Список использованных источников | 24 |

**введение**

В настоящее время с развитием вычислительных систем и информационных технологий растет популярность систем автоматизации и роботизации, как в промышленности и науке, так и в повседневной жизни. Как следствие растет потребность в эффективных методах обработки поступающей информации, в частности видео данных. Эффективная обработка и использование входной информации позволяет значительно повысить производительность и расширить круг применения данных систем.

Подтверждением вышесказанному является повсеместное развитие и использование робототехники, систем распознания текстов, охранных систем основанных на распознании изображений, и т.д. В последнее время все большее внимание уделяется системам, использующим машинное зрение в качестве основного источника информации. Что привело к возникновению потребности в новых алгоритмах обработки и распознания изображений

Не смотря на это, задача распознавания образов до сих пор не решена в полном объеме. Однако, в рамках существенных ограничений, есть методы, позволяющие приблизится к ее решению.

**Компьютерное зрение** — это технология создания машин, которые могут производить обнаружение, отслеживание и классификацию объектов. Для обработки данных в ней применяются статистические методы, а также модели, построенные при помощи геометрии, физики и теории обучения. Начало разработок, связанных с данным направлением, относится к 1950-м годам. Первый реальный успех в этой области был достигнут в Корнеллской лаборатории аэронавтики в 1958-1960 годах в связи с реализацией на ЭВМ IBM-740 аппаратного варианта системы распознавания простейших зрительных образов — Mark I Perceptron (автор разработки - Фрэнк Розенблатт).

1. **Обзор алгоритмов распознания объектов на двумерных**

**изображениях**

Основным разделом компьютерного зрения является извлечение информации из изображения или последовательности изображений. Чаще всего это применяется для определения и распознавания объекта. Распознавание объекта разделено на два этапа: фильтрация изображения и анализ результатов фильтрации.

**Фильтрация**. На этапе фильтрации используются различные методы, выделяющие на изображениях области интереса. Во время фильтрации не производится анализ объекта, но точки, которые проходят фильтрацию, можно рассматривать как области с особыми характеристиками. Среди методов можно выделить наиболее часто применяемые группы методов: определение краев, бинаризация, вейвлет-анализ и фильтрация функций.

**Определение краев** — термин, обозначающий набор математических методов, направленных на выявление точек в цифровом изображении, в которых яркость изображения резко меняется. Эти точки обычно организованы в виде набора кривых линий. Выделенные края могут быть двух типов: независящие и зависящие от точки зрения. Независящие границы отображают такие свойства, как цвет и форма поверхности. Зависящие могут меняться в разных точках обзора и отображают геометрию сцены.

**Бинаризация**. Одним из наиболее простых и естественных способов бинаризации изображения является пороговое преобразование по яркости. Бинаризация — преобразование изображения в монохромное. Обычно участки изображения, прошедшие фильтрацию, окрашиваются в белый, а все остальное изображение в черный. Один из наиболее быстрых и эффективных методов поиска порогового значения — метод, придуманный японским ученым Нобуюки Оцу, и основанный на обработке гистограммы яркости изображения. В другом популярном способе бинаризации используется поиск объекта по цвету. Данному методу мешает влияние множества факторов. Например, освещенность. Видимый цвет — это результат взаимодействия спектра излучаемого света и поверхности. Поэтому если белый лист освещать красным светом, то и лист будет казаться красным. Проще всего использовать для детектора границ цветовую модель HSV. Для определения границ цветового пятна все три слоя цветового пространства отделяются друг от друга. Далее для каждого слоя устанавливаются пороговые значения. После сложения итогового изображения белыми пикселями на нем будут отображены те точки, в которых значения выше порогового во всех трех координатах цветовой модели.

**Вейвлет-анализ**. Традиционно сложилось, что во многих командах вейвлет-анализом называется поиск произвольного паттерна на изображении при помощи свёртки с моделью этого паттерна. Задача сводится к поиску объекта по шаблону с некоторыми допущениями. Хорошим примером использования расширенной трактовки вейвлетов является задачка поиска блика в глазу, для которой вейвлетом является сам блик.

**Анализ результатов фильтрации**. Среди множества способов обработки результатов фильтрации выделяют несколько: алгоритмы контурного анализа и алгоритмы на нейронных сетях. Контурный анализ позволяет описывать и производить объекты, которые представлены в виде своих внешних очертаний — контуров. Контур содержит всю необходимую информацию о форме объекта. Но внутренние точки объекта не учитываются. Это ограничивает образ применения контурного анализа, но позволяет перейти от двумерного пространства изображений к пространству контуров, что позволяет очень сильно снизить алгоритмическую и временную сложность программы. Обычно выделяют два алгоритма: метод «жука» и цепной код Фримана.

**Алгоритмы на нейронных сетях**. Одним из таких алгоритмов является каскад Хаара. Алгоритм, основанный на каскадах примитивов, на вейвлете Хаара и классификаторе AdaBoost. Примитивы Хаара — признаки цифрового изображения, используемые в распознавании образов и основанные на вейвлете Хаара. Эти признаки представляют собой разделенные на две, три или четыре области прямоугольники, которые позиционируются на изображении. Интенсивность пикселей в каждой области суммируется, после чего вычисляется разность между суммами. Эта разность является значением данного признака Хаара. Другим часто используемым методом является SURF. Метод ищет ключевые точки при помощи матрицы Гессе. Этот метод позволяет найти стабильные ключевые точки, которые будут определяться даже при повороте объекта. Для каждой ключевой точки при помощи фильтров Хаара считается градиент, а из матрицы Гессе берется масштаб. Для каждой точки формируется дескриптор, состоящий из 64 или 128 чисел, каждое из которых обозначает флуктуации градиента вокруг точки. В системе контроля на производстве гораздо легче использовать цветные маркеры и распознавание цвета, чем создавать и обучать нейронную сеть для классификации объекта. Так же, как и при детекции движения проще следить за изменением границ областей с различной яркостью. Но в системе автоопределения автомобильных номеров или системе распознавания лиц, наоборот, вряд ли подойдет какой-либо метод, в котором не используются элементы искусственного интеллекта. Таким образом, все вышеперечисленные методы имеют свои преимущества.

1. **Алгоритм поиска максимума кросскорреляционной функции двух изображений**

Дискретной кросс-корреляций функций f(t) и g(t), определенных на множестве целых чисел Z, называется следующая операция:

Кросс-корреляция чаще всего применяется в обработке сигналов, при этом f считается образцом, а g – сигналом, содержащим образец. Результат – это вектор чисел, показывающих, насколько сильно образец выражен в сигнале. В нашем случае мы можем представить наши изображения в виде двух дискретных функций яркости и применить к ним операцию кросскорреляции. Максимум этой функции будет находиться в той области где изображения совпадают больше всего. По сути операция кросскорреляции является сверткой двух функций.

*2.1 Модификация алгоритма поиска максимума дискретной кросскорреляционной функции от двух изображений*

Как упоминалось ранее операция кросскорреляции по сути является сверткой двух функций.

**Свёртка фу́нкций** — операция в функциональном анализе, показывающая «схожесть» одной функции с отражённой и сдвинутой копией другой.

Свойства

1) Коммутативность:

.

2) Ассоциативность:

1. Дистрибутивнсть:
2. Ассоциативность умножения на скаляр:
3. Правило дифференцирования:

где Df обозначает производную функции f.

1. Свойство Фурье-образа:

где F(f) обозначает преобразование Фурье функции f.

Как следствие мы можем воспользоваться свойством 6 свертки.

**Преобразование Фурье** – операция, сопоставляющая функции вещественной переменной другую функцию вещественной переменной. Эта новая функция описывает коэффициенты («амплитуды») при разложении исходной функции на элементарные составляющие — гармонические колебания с разными частотами.

Преобразование Фурье функции f вещественной переменной является интегральным преобразованием и задается следующей формулой:

По сути говоря мы заменяем операцию вычисления свертки операцией умножения Фурье-образов функций яркости изображений. Что позволяет значительно повыситьпроизводительность.

Применительно к нашей области мы воспользуемся дискретным преобразованием Фурье.

Прямое преобразование:

Обратное преобразование:

Применя это преобразование к функциям яркости наших изображений мы получаем вектора комплексных чисел. Перемножая поэлементно их мы получим вектор комплексных чисел. Беря обратное преобразование Фурье получаем значения функции свертки. Находя максимум среди этих значений мы находим точку в которой наши два изображения наиболее похожи.

Достоинства метода:

1. Простота реализации
2. Быстрота работы
3. Качественное детектирование объектов на изображении
4. Устойчивость к масштабированию

Недостатки метода:

1. Т. к. метод ищет глобальный максимум, то он может найти только 1 совпадение
2. Не устойчив к поворотам
3. Относительно плохо устойчив к изменению общего уровня яркости изображений

*2.2 Модификации алгоритма поиска максимума кросскорреляционной функции.*

Как упоминалось ранее, данный алгоритм ищет глобальный максимум следовательно он может найти лишь 1 совпадение. Данная модификация призвана решить эту проблему. Решение заключается в том, что помимо глобального максимума необходимо найти и другие. Сделать это можно несколькими способами

1. Найти все точки с координатами i, j такие что

где maxi и maxj координаты глобального максимума, а e- некоторая константа. е можно выбирать разными способами.

1. Найти все точки с координатами i, j такие что

Где w-ширина изображения h – высота изображения N=w\*h.

Результатом данной модификации является избавление от проблемы детектирования лишь 1 объекта.

**3. Обнаружение устойчивых признаков изображения: метод SURF**

Принцип работы алоритма заключается в следующем. Для изображения сцены и изображения эталона с помощью метода SURF находятся особые точки и уникальные дескрипторы для них. Сравнивая эти наборы дескрипторов, мы можем выделить эталонный объект на сцене. А теперь более подробно.

SURF решает две задачи – поиск особых точек изображения и создание их дескрипторов, инвариантных к масштабу и вращению.

Метод ищет особые точки с помощью матрицы Гессе. Детерминант матрицы Гессе (т.н. гессиан) достигает экстремума в точках максимального изменения градиента яркости. Он хорошо детектирует пятна, углы и края линий.

Гессиан инвариантен относительно вращения. Но не инвариантен масштабу. Поэтому SURF использует разномасштабные фильтры для нахождения гессианов.

Для каждой ключевой точки считается направление максимального изменения яркости (градиент) и масштаб, взятый из масштабного коэффициента матрицы Гессе.

*3.1. Градиент в точке вычисляется с помощью фильтров Хаара.*

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

Флуктуации градиента окрестностей ключевой точки считаются относительно направления градиента вокруг точки в целом (по всей окрестности ключевой точки). Таким образом, достигается инвариантность дескриптора относительно вращения. Размер же области, на которой считается дескриптор, определяется масштабом матрицы Гессе, что обеспечивает инвариантность относительно масштаба.

Флуктуации градиента также считаются с помощью фильтра Хаара.

*3.2. Интегральное представление*

Для эффективного вычисления фильтров Гессе и Хаара – используется интегральное представление изображений.

Если кратко, то интегральное представление является матрицей, размерность которой совпадает с размерностью исходного изображения, а элементы считаются по формуле:

Где I(i,j) – яркость пикселов исходного изображения.

Имея интегральную матрицу можно очень быстро вычислять сумму яркостей пикселов произвольных прямоугольных областей изображения, по формуле:

SumOfRect(ABCD) = II(A) + II(С) — II(B) — II(D)

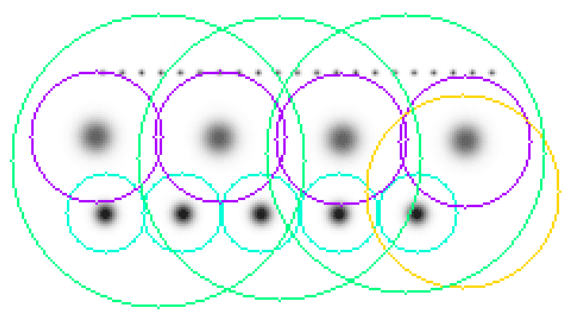
Где ABCD – интересующий нас прямоугольник.

*3.3 Вычисление матрицы Гессе*

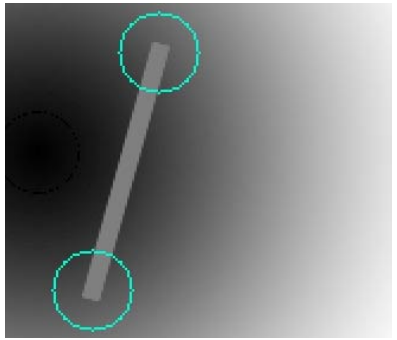
Обнаружение особых точек в SURF основано на вычислении детерминанта матрицы Гессе (гессиана).

Матрица Гессе для двумерной функции и ее детерминант определяется следующим образом:

Значение гессиана используется для нахождения локального минимума или максимума яркости изображения. В этих точках значение гессиана достигает экстремума.

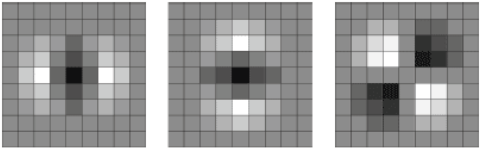


На картинке видно, что особые точки (очерченные цветными кругами) представляют собой локальные экстремумы яркости изображения. Мелкие точки не распознаны как особые, из-за порогового отсечения по величине гессиана.



На рисунке показаны концы отрезка, распознанные как ключевые точки, с помощью матрицы Гессе.

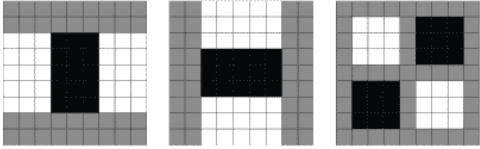
Теоретически, вычисление матрицы Гессе сводится к нахождению Лапласиана Гауссиан.По сути, элементы матрицы Гессе вычисляются как свертка (сумма произведений) пикселов изображения на фильтры, изображенные на рисунке:



На рисунке изображены дискретизированные фильтры для нахождения четырех элементов матрицы Гессе (четвертый – совпадает с третьим, поскольку матрица Гессе симметрична). Фильтры имеют пространственный масштаб 9x9 пикселов. Темные участки соответствуют отрицательным значениям фильтра, светлые – положительным.

Однако, SURF не использует лапласиан гауссианы в том виде, который изображен на рисунке. Во-первых, по утверждению авторов, дискретизированный лапласиан гауссианы имеет довольно большой разброс значения детерминанта, при вращении образца (напомним, что в идеале гессиан должен быть инвариантен к вращению). Особенно детерминант «проседает» в районе поворота на 45 граудсов. А во-вторых, и это главное, фильтр для лапласиана гауссианы имеет непрерывный характер. Почти все пикселы фильтра имеют разные величины яркости. А это не позволяет эффективно использовать такой мощный механизм расчёта, как интегральную матрицу изображения.

Поэтому SURF использует бинаризированную аппроксимацию лапласиана гауссиан (авторы назвали его Fast-Hessian):



На рисунке изображены фильтры, используемые для нахождения матрицы Гессе в SURF. Белые области соответствуют значению +1, черные -2 (на третьем фильтре -1), серые – нулевые. Пространственный масштаб – 9x9 пикселов.

Этот фильтр более устойчив к вращению, и его можно эффективно вычислить с помощью интегральной матрицы.

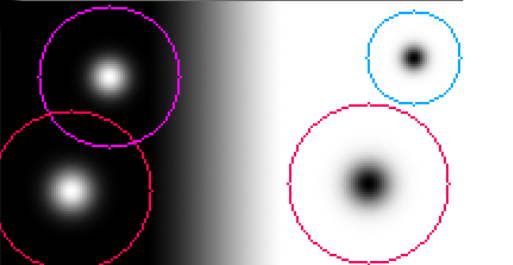
Таким образом, в SURF, гессиан вычисляется так:

Где Dxx, Dyy, Dxy – свертки по фильтрам, изображенным на рисунке вверху. Коэффициент 0.9 имеет теоретическое обоснование, и корректирует приближенный характер вычислений.

Итак, для нахождения особых точек, SURF пробегается по пикселам изображения и ищет максимум гессиана. Способ нахождения локального максимума гессиана мы рассмотрим позднее. В методе задается пороговое значение гессиана. Если вычисленное значение для пиксела выше порога – пиксел рассматривается как кандидат на ключевую точку.

Тут еще полезно заметить следующее. Поскольку гессиан является производной, и зависит только от перепада яркости, но не от абсолютного ее уровня, то он инвариантен по отношению к сдвигу яркости изображения. Таким образом, изменение уровня освещения образца не влияет на обнаружение ключевых точек.

Кроме того, свойства гессиана таковы, что он достигает максимума, как в точке белого пятна на черном фоне, так и черного пятна на белом фоне. Таким образом, метод обнаруживает и темные, и светлые особенности изображения.



Метод распознает как светлые точки на темном фоне, так и темные точки на светлом фоне.

*3.4 Шкалы*

Как уже отмечалось, гессиан не инвариантен относительно масштаба. Это значит, что для одного и того же пиксела, гессиан может меняться при изменении масштаба фильтра. Решение этой проблемы только одно – перебирать различные масштабы фильтров и поочередно их применять к данному пикселу.

Из соображений симметрии и дискретизации, размер фильтра Fast-Hessian не может принимать произвольные значения. Допустимые размеры этого фильтра таковы (начиная с минимального): 9, 15, 21, 27 и так далее, с шагом 6. Однако, на практике, постепенно увеличивать размер фильтра на 6 — не выгодно, потому что для крупных масштабов шаг 6 оказывается слишком мелким, а фильтры — избыточными. Поэтому (и по некоторым другим причинам), SURF разбивает все множество масштабов на так называемые октавы. Каждая октава покрывает определенный интервал масштабов, и имеет свой характерный размер фильтра.

При этом если бы на октаву приходился только один фильтр, это было бы слишком грубым приближением. Кроме того, мы бы не могли найти локальный максимум гессиана, среди разных масштабов, в разных октавах. Ведь одна и та же точка может иметь несколько локальных максимумов гессиана, в разных масштабах. Это хорошо видно на картинке:

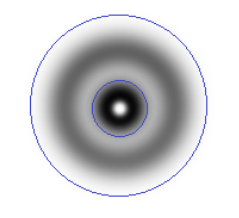
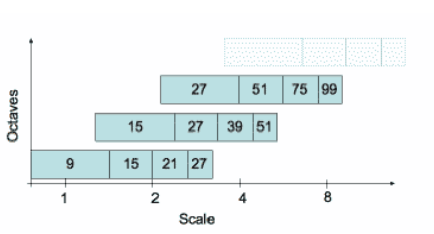


Рисунок показывает две ключевые точки разного масштаба в одной точке изображения.

Если мы будем искать максимум среди всех гессианов, по всем масштабам, то мы бы нашли только один из максимумов, в то время как их может быть несколько. Один – в одном масштабе, другой – в другом.

Исходя из перечисленного, октава содержит не один фильтр, а четыре фильтра, которые хорошо покрывают характерный масштаб октавы:



На рисунке показаны первые три октавы метода SURF. Цифры в прямоугольниках показывают размер фильтра Fast-Hessian. Логарифмическая шкала снизу – показывает масштабы, покрываемые октавами.

Шаг размера фильтра в первой октаве – составляет 6, во второй – 12, в третьей – 24 и так далее.

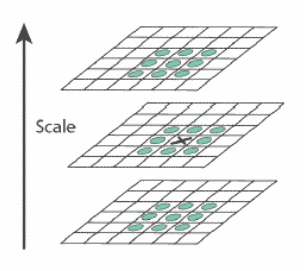
Как видим, октавы значительно перекрываются друг другом. Это увеличивает надежность нахождения локальных максимумов. Почему в октаве именно четыре фильтра станет ясно из следующей главы.

Возникает вопрос, а сколько собственно октав достаточно для покрытия множества особых точек разных масштабов? Теоретически, масштабы бесконечны, однако в реальных изображениях, они вполне конечны, и основная масса сосредоточена в интервале от 1 до 10 (по данным авторов метода). Для покрытия этого диапазона достаточно четырех октав. Плюс добавляется одна или две октавы для покрытия больших масштабов. Итого, используется 5-6 октав. Теоретически, этого вполне достаточно для покрытия всевозможных масштабов на изображении 1024x768 пикселов.

*3.5. Нахождение локального максимума гессиана*

Для нахождения локального максимума гессиана, используется так называемый метод соседних точек 3x3x3.

Его смысл понятен из рисунка ниже:



Пиксел, помеченный крестиком считается локальным максимумом, если его гессиан больше чем у любого его соседа в его масштабе, а также больше любого из соседей масштабом меньше и масштабом больше (всего 26 соседей).

Исходя из такого определения локального максимума, понятно, что октава должна содержать не менее трех фильтров, иначе мы не сможем определить факт нахождения локального максимума гессиана внутри октавы.

Отметим еще такой момент. Фильтры октавы считаются не для всех пикселов подряд. Первая октава считается для каждого второго пиксела изображения. Вторая – для каждого четвертого, третья – для каждого восьмого и так далее. Смысл понятен – две точки с расстоянием 2 не могут содержать более одного максимума масштаба 2, 3 или более высоких масштабов. Поэтому нет смысла перебирать все точки изображения, для нахождения максимума масштаба 3, например.

Удвоение шага пикселов для октав позволяет экономить при расчёте фильтров. Как вы наверно уже заметили, размеры фильтров в октавах повторяются. Так, например, фильтр размером 27 присутствует в трех октавах. Так вот, при вычислениях, этот фильтр будет считаться только для первой октавы. Вторая и третья – просто используют расчёты первой октавы. А удвоение шага пикселов гарантирует, что точки в которых нужно считать гессиан, уже были просчитаны предыдущей октавой.

Поэтому, несмотря на то, что октава содержит четыре фильтра, на самом деле каждая октава(кроме первой) считает только два характерных для нее размера, два других – всегда можно взять из предыдущих октав. Первая же октава вынуждена считать все четыре своих фильтра.

Итак, после нахождения максимального гессиана методом соседних точек 3x3x3, мы нашли пиксел в котором этот максимум достигается. Однако, поскольку, октава перебирает не все точки изображения, то истинный максимум может не совпадать с найденным пикселом, а лежать где-то рядом, в соседних пикселах.

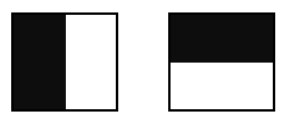
Для нахождения точки истинного максимума, используется интерполирование найденных гессианов куба 3x3x3 квадратичной функцией. Далее, вычисляется производная (методом конечных разностей соседних точек). Если она близка к нулю – мы в точке истинного максимума. Если производная велика – сдвигаемся в сторону ее уменьшения, и повторяем итерацию, до тех пор пока производная не станет меньше заданного порога. Если в процессе итераций мы отходим от начальной точки слишком далеко, то это считается ложным максимумом, и точка больше не считается особой.

*3.6. Нахождение ориентации особой точки*

Для инвариантности вычисления дескрипторов особой точки, которые будут рассмотрены ниже, требуется определить преобладающую ориентацию перепадов яркости в особой точке. Это понятие близко к понятию градиента, но SURF использует немного другой алгоритм нахождения вектора ориентации.

Сначала, вычисляются точечные градиенты в пикселах, соседних с особой точкой. Для рассмотрения берутся пикселы в окружности радиуса 6s вокруг особой точки. Где s – масштаб особой точки. Для первой октавы берутся точки из окрестности радиусом 12.

Для вычисления градиента, используется фильтр Хаара. Размер фильтра берется равным 4s, где s – масштаб особой точки. Вид фильтров Хаара показан на картинке:

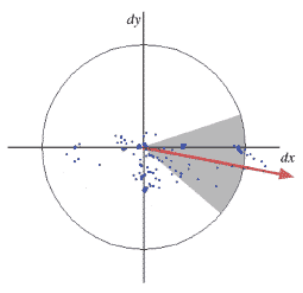


Фильтры Хаара. Черные области имеют значения -1, белые +1.

Фильтры Хаара дают точечное значение перепада яркости по оси X и Y соответственно.Поскольку фильтры Хаара имеют прямоугольную форму, их значения легко считаются с помощью интегральной матрицы. Для расчёта одного фильтра произвольного размератребуется всего 6 операций.

Значения вейвлета Хаара dX и dY для каждой точки умножаются на вес и запоминаются в массиве. Вес определяется как значение гауссианы с центром в особой точке и сигмой равной 2s. Взвешивание на гауссиане необходимо для отсечения случайных помех на далеких от особой точки расстояниях.

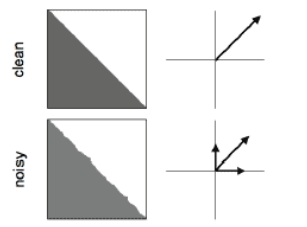
Далее, все найденные значения dX и dY, условно наносятся в виде точек на плоскость, как показано на рисунке:



На рисунке показаны все найденные градиенты в виде точек в пространстве dXdY.

Далее, берется угловое окно (показано серым на рисунке) размером π/3, и вращается вокруг центра координат. Выбирается такое положение окна, при котором длина суммарного вектора для попавших в окно точек – максимальна. Вычисленный таким образом вектор нормируется и принимается как приоритетное направление в области особой точки.

Манипуляции с окном нужны для уменьшения влияния шумовых точек. Ниже на рисунке приведен пример градиента при идеальном крае, и при крае с шумом:



Как видим, шум дает дополнительные градиенты в направлениях, не совпадающих с направлением основного градиента. Использование окна позволяет отсечь такие шумовые точки, и более точно вычислить истинный градиент.

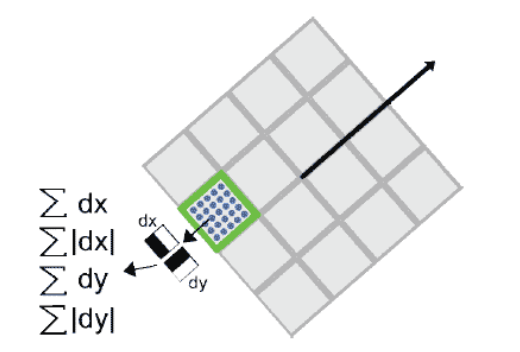
Отметим, что не всегда требуется инвариантность дескрипторов относительно вращения. Метод SURF имеет модификацию, в которой ориентация особых точек не рассчитывается. Такая модификация позволяет надежно идентифицировать точки, повернутые не более чем на ±15 градусов.

*3.7. Вычисление дескриптора особой точки*

Дескриптор представляют собой массив из 64 (в расширенной версии 128) чисел, позволяющих идентифицировать особую точку. Дескрипторы одной и той же особой точки на образце и на сцене должны примерно совпадать. Метод расчета дескриптора таков, что он не зависит от вращения и масштаба.

Для вычисления дескриптора, вокруг особой точки формируется прямоугольная область, имеющая размер 20s, где s – масштаб в котором была найдена особая точка. Для первой октавы, область имеет размер 40x40 пикселов. Квадрат ориентируется вдольприоритетного направления, вычисленного для особой точки.

Дескриптор считается как описание градиента для 16 квадрантов вокруг особой точки.



Далее, квадрат разбивается на 16 более мелких квадрантов, как показано на рисунке. В каждом квадранте берется регулярная сетка 5x5 и для точки сетки ищется градиент, с помощью фильтра Хаара. Размер фильтра Хаара берется равным 2s, и для первой октавы составляет 4x4.

Следует отметить, что при расчёте фильтра Хаара, изображение не поворачивается, фильтр считается в обычных координатах изображения. А вот полученные координаты градиента (dX,dY) поворачиваются на угол, соответствующий ориентации квадрата.

Итого, для вычисления дескриптора особой точки, нужно вычислить 25 фильтров Хаара, в каждом из 16 квадрантов. Итого, 400 фильтров Хаара. Учитывая, что на фильтр нужно 6 операций, выходит, что дескриптор обойдется минимум в 2400 операций.

После нахождения 25 точечных градиента квадранта, вычисляются четыре величины, которые собственно и являются компонентами дескриптора:

∑dX, ∑|dX|, ∑dY, ∑|dY|

Две из них есть просто суммарный градиент по квадранту, а две других – сумма модулей точечных градиентов.

На рисунке показано поведение этих величин для разных участков изображений.

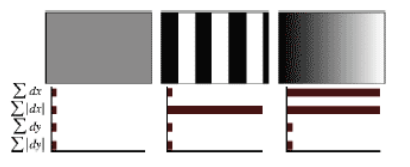


Рисунок показывает поведение дескриптора для разных изображений. Для равномерных областей – все значения близки к нулю. Для повторяющихся вертикальных полосок – все величины, кроме второй близки к нулю. При увеличении яркости в направлении оси X, две первые компоненты имеют большие значения.

Четыре компонента на каждый квадрант, и 16 квадрантов, дают 64 компонента дескриптора для всей области особой точки. При занесении в массив, значения дескрипторов взвешиваются на гауссиану, с центром в особой точке и с сигмой 3.3s. Это нужно для большей устойчивости дескриптора к шумам в удаленных от особой точки областях.

Плюс к дескриптору, для описания точки используется знак следа матрицы Гессе, то есть величина sign(Dxx+Dyy). Для светлых точек на темном фоне, след отрицателен, для темных точек на светлом фоне – положителен. Таким образом, SURF различает светлые и темные пятна.

Таким образом применив SURF к изображению мы получим набор дескрипторов который будет уникально идентифицировать эталон на сцене.

Достоинства метода:

1) Инвариантен к поворотам и масштабированию

2) Инвариантен к разнице общей яркости изображений

3) Может детектировать более 1 объекта на сцене

Недостатки метода:

1) Достаточно сложен в реализации

2) Относительно медленная работа алгоритма

**заключение**

Если делаем выводы, алгоритм поиска максимума кросскорреляционной функции - просто для реализации и выполнение такого алгаритма является быстрым, но есть недостатки, в виде не устойчивости к поворотам изображения.

А метод SURF решает эти проблемы, но скорость работы сушественно падает.

**список использованных источников**

1. Дэвид А. Форсайт, Джин Понс. Компьютерное зрение. Современный подход. — М., 2004.

2. Метод Виолы-Джонса как основа для распознавания лиц [Электронный ресурс] // Хабрахабр — URL: <http://habrahabr.ru/post/133826> (дата обращения: 15.11.2015).

3. Обнаружение объектов методом Оцу [Электронный ресурс] // Хабрахабр — URL: <http://habrahabr.ru/post/112079> (дата обращения: 15.11.2015).

4. OpenCV шаг за шагом. Поиск объекта по цвету — RGB [Электронный ресурс] // Робокрафт — URL: <http://robocraft.ru/blog/computervision/365.html> (дата обращения: 15.11.2015).

5. Детектор границ Канни [Электронный ресурс] // Хабрахабр — URL: <http://habrahabr.ru/post/114589> (дата обращения: 15.11.2015).