Факультет компьютерных технологий и прикладной математики

Кафедра вычислительных технологий

02.03.02

Алгоритмы цифровой обработки мультимедиа

Лабораторная работа № 4

Методы выделения границ. Алгоритм Канни.

Работа будет осуществляется средствами языка Python 3.10 и IDE PyCharm2022.1.2 с учебной лицензией. Для работы необходимо установить библиотеку opencv.

В рамках данной лабораторной работы будет рассматриваться решение задачи выявления границ объектов на изображении. Данная задача является необходимой частью более сложных и полных задач – выявление объекта на изображении, идентификация объекта, распознавание текста на фрагментах изображения, детектор движения в видеопотоке, обнаружение объекта в видеопотоке, трекинг объекта и многих других.

В данной работе контур рассматривается как совокупность пикселей, в окрестности которых наблюдается скачкообразное изменение функции яркости. Точки контура представляют собой границу объекта, отделяющую его от фона. В дальнейшем в данной работе также для обозначения контура будет использоваться понятие граница, подразумевающее границу яркости.

В данной работе под цифровым изображением подразумевается двумерное изображение, представленное в цифровом виде. Рассматриваются растровые изображения, которые представляются как прямоугольный двумерный массив чисел, при этом каждое число соответствует одному элементу изображения или пикселю.

В данной работе под изображением понимается функция двух вещественных переменных $I(i,j)$, где I – это интенсивность (яркость) в точке с координатами (i,j).

Методы выделения границ делятся на следующие основные группы:

- градиентные методы;
- метод активных контуров;
- методы на основе вейвлет-преобразования;
- методы на основе нечёткой логики;
- методы на основе нейронной сети;
- методы на основе генетических алгоритмов;
- методы на основе математической морфологии;
- методы на основе статистических характеристик;
- методы на основе фазовой конгруэнтности;
- методы на основе кластеризации;
- комбинированные методы.

В данной лабораторной работе будут рассмотрены градиентные методы, в частности комбинированный градиентный метод, являющийся наиболее часто применяемым для решения поставленной задачи – алгоритм Канни (Кэнни, Кенни).

Градиентные методы основаны на достаточно простой идее, заключающейся в том, что на *границе объекта происходит сильный скачок яркости изображения.* Получаем, что если в данном пикселе *СИЛЬНО МЕНЯЕТСЯ ЗНАЧЕНИЕ ЯРКОСТИ, ДАННЫЙ ПИКСЕЛЬ ЯВЛЯЕТСЯ ГРАНИЦЕЙ*. Далее необходимо понять, как же определять *КОЛИЧЕСТВЕННО* , насколько сильно меняется значение яркости. В данном случае необходимо вспомнить, что изображение рассматривается, как функция двух переменных. И яркость пикселя – это функция. Тогда – скорость изменения яркости – это скорость изменения функции яркости, то есть частные производные функции яркости (яркость – функция двух переменных, для нее нет понятия производная, а есть понятия частные производные),

точнее градиент данной функции. И выставляя некоторое пороговое значение для величины градиента мы можем определять, является ли данный пиксель границей или нет. Градиент – это вектор, состоящий из двух значений, значит нам необходимо дополнительно задать правило вычисления величины вектора. В градиентных методах используется простейший случай – длина вектора, вычисляемая согласно классическому определению.

И тогда получается достаточно простая идея – *если длина градиента яркости пикселя больше длины градиента соседей и больше некоторой порогово величины, то данный пиксель считается границей*.

В данном подходе существует множество частных случаев и аспектов, требующих уточнения, и градиентные методы отличаются друг от друга способами решения возникающих вопросов.

Теперь перейдем к разбору градиентного комбинированного метода, который называется детектор границ Канни. Данный метод был разработан в 1986 году Джоном Кэнни и использует многоступенчатый алгоритм для обнаружения границ на изображениях. Алгоритм состоит из следующих шагов:

1) построение черно-белого изображения;
2) применение фильтрации для подавления шумов;
3) вычисление градиентов функции яркости;
4) подавление немаксимумов (если значение градиента пикселя больше соседних, то пиксель определяется как граничный, иначе значение пикселя подавляется);
5) двойная пороговая фильтрация(сравнение величины градиента с двумя пороговыми значениями).

Далее рассмотрим подробно каждый из этих этапов алгоритма.

1) Построение черно белого изображения было разобрано на первых двух лабораторных работах.

2) В качестве фильтра подавления шумов в алгоритме Канни используется размытие по Гауссу, которое было разобрано в предыдущей лабораторной работе.

3) Теперь разберем подробно вычисление градиента функции яркости.

Напомним, что изображение представляет из себя функцию яркости $I(x, y)$, где x, y – координаты пикселя изображения. Градиентом функции двух переменных в точке x_0, y_0 называется вектор

$$\nabla I(x_0, y_0) = \left(\frac{\partial I}{\partial x}(x_0, y_0), \frac{\partial I}{\partial y}(x_0, y_0)\right), \quad (1)$$

полученный из значений частных производных в точке x_0, y_0. Будем далее для увеличения наглядности преобразований обозначать градиент следующим образом: $\nabla I(x_0, y_0) = (G_x, G_y)$.

Величиной градиента будем считать его длину, вычисляемую согласно определению длины вектора $|\nabla I| = \sqrt{{G_x}^2 + {G_y}^2}$

Здесь возникает интересная особенность: понятие частной производной определено для непрерывной на области определения функции, в то время как рассматриваемая функция изображения $I(x, y)$ определена на конечном множестве натуральных точек. Для разрешения данной ситуации необходимо рассмотреть численные методы вычисления частных производных. В алгоритме Канни рассматривается оператор Собеля для вычисления частных производных.

$$\frac{\partial I}{\partial x}(x_0, y_0) = I(x+1, y+1) - I(x-1, y-1) + I(x+1, y-1) - I(x-1, y+1) + 2 * \big(I(x+1, y) - I(x-1, y)\big)$$

$$\frac{\partial I}{\partial x}(x_0, y_0) = I(x+1, y+1) - I(x-1, y-1) + I(x-1, y+1) - I(x+1, y-1) + 2 * \big(I(x, y+1) - I(x, y-1)\big)$$

Для повышения наглядности вычислений вспомним определенную нами в прошлой лабораторной работе операцию свертки изображения с помощью ядра свертки.

Операция свертки заключается в преобразовании исходной матрицы B размерности $n \times n$ в числовое значение с помощью специальной матрицы ker размерности $n \times n$, называемой ядром свертки:

$$val = \sum_{k=1}^{n} \sum_{l=1}^{n} B[k,l] * \ker[k,l] \quad (2)$$

Тогда оператор Собеля удобно представить в виде применения операции свертки с двумя матрицами свертки вида:

$$G_x = \begin{pmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{pmatrix}, G_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{pmatrix}$$

В градиентных методах для вычисления градиента возможно использование не только оператора Собеля, но и в качестве альтернативы: оператор Робертса, Оператор Прюитта, оператор Кирша.

После вычисления частных производных для все внутренних пикселей изображения (все, кроме первых и последних строки и столбца) необходимо вычислить длину градиента для каждого пикселя. Алгоритм Канни строится на том предположении, что границей будут являтся такие пиксели, градиент которых будет локальным максимумом. Для реализации этого алгоритма необходимо сравнить величину градиента пикселя с соседями. И если длина градиента больше, то мы имеем дело с границей.

Но здесь возможен следующий интересный аспект. Рассмотрим изображение 1.

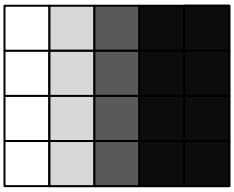


Рисунок 1 – Смена цвета объекта

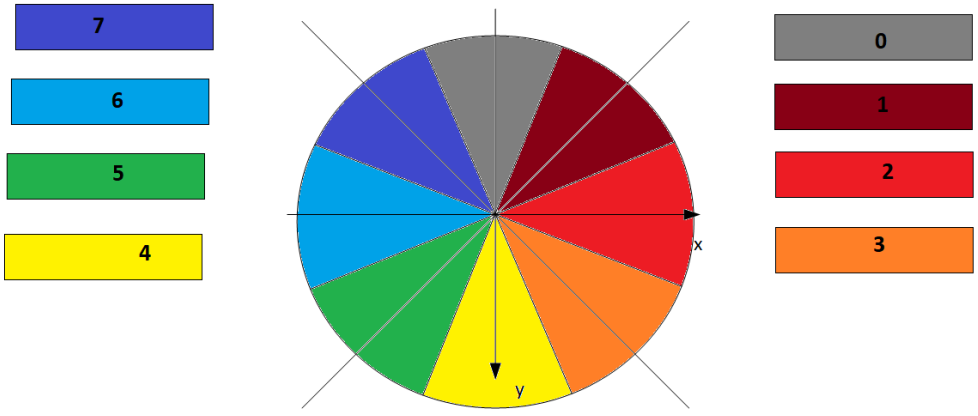
В данном изображении граница объекта будет идти снизу вверх. При этом градиент всех пикселей третьего столбца будет одинаковым, при чем явно видно, что именно третий столбец будет граничным. В этом случае

становится очевидным, что величину градиента нужно сравнивать не со всеми соседними пикселями, а с теми пикселями, по направлению которых значение функции отличается сильнее всего. Для этого необходимо вычислить направление, в котором функция меняется наиболее быстро. Для этого найдем величину угла градиента

$$\varphi = arctg\frac{G_y}{G_x}$$

Далее, по сути нам нет необходимости точно знать, какова величина угла 10 или 15 градусов, например. По факту, нам необходимо для направления выбрать лишь один из соседних 8 пикселей. То есть необходимо округлить величину угла до 45 градусов.

С учетом того, что система координат здесь видоизменена относительно классической математической удобно воспользоваться следующей схемой для округления угла, изображенной на рисунке 2.



0 – x>0, y<0, tg < -2.414 || x<0, y<0, tg > 2.414
1 – x>0, y<0, tg <-0,414
2 – x>0, y<0, tg>-0,414 || x>0, y>0, tg<0.414
3 – x>0, y>0, tg<2.414
4 – x>0, y>0, tg>2.414 || x<0, y>0, tg < -2.414
5 – x<0, y>0, tg<-0,414
6 – x<0, y>0, tg>-0,414 || x<0, y<0, tg < 0.414
7 – x<0, y < 0, tg < 2.414

Рисунок 2 – Схема округления угла до 45 градусов

Здесь принято в качестве угла использовать одно из значений 0 – 7, характеризующееся указанными на схеме границами значений частных производных по x и y и тангенсом угла градиента.

Теперь рассмотрим реализацию данного алгоритма.

1) Зададим матрицы оператора Собеля, создаем матрицы для значений частных производных, длины градиента и угла градиента

2) Для каждого внутреннего пикселя:

a. применяем операторы свертки для вычисления (G_x, G_y);

b. находим длину вектора градиента $|\nabla I| = \sqrt{{G_x}^2 + {G_y}^2}$;

c. находим $tg\varphi = \frac{G_y}{G_x}$;

d. согласно рисунку 2 находим округленное значение угла 0..7.

3) Находим максимальное значение длины градиента для всего изображения

Пример реализации приведен в яндекс диске и на гит-репозитории преподавателя, будет доступен после проведения занятия.

4) Теперь перейдем к 4 пункту алгоритма.

Подавление немаксимумов означает следующую идею. Вернемся к примеру, показанному на рисунке 1. Рассмотрим центральный пиксель. Предположим, что значение длины градиента вычислено и равно, например 400. По рисунку видно, что наибольший рост значения яркости достигает при движении по горизонтали, причем при движении вдоль оси OX значение яркости уменьшается. Для нас это означает, что направление наибольшего роста функции будет выявлено в 6 сегменте по рисунку 2. При этом заметим, что все строки данного фрагмента изображения одинаковы, значит в центральном столбце величина градиента будет одинаковой. Но вот слева и справа от центрального столбца значение градиента будет явно меньше. При этом именно это направление и задано углом градиента.

Исходя из выше сказанного будет работать следующее правило: *ГРАНИЦЕЙ БУДЕТ СЧИТАТЬСЯ ПИКСЕЛЬ, ГРАДИЕНТ КОТОРОГО МАКСИМАЛЕН В СРАВНЕНИИ С ПИКСЕЛЯМИ ПО НАПРАВЛЕНИЮ НАИБОЛЬШЕГО РОСТА ФУНКЦИИ*. В нашем примере, направление задано числом 6, то есть направление вдоль оси X, рассматриваем значение градиента

в пикселях слева и справа от заданного. *ЕСЛИ ЗНАЧЕНИЕ ГРАДИЕНТА ВЫШЕ, ЧЕМ У ПИКСЕЛЕЙ СЛЕВА И СПРАВА, ТО ДАННЫЙ ПИКСЕЛЬ – ЭТО ГРАНИЦА, ИНАЧЕ – НЕ ГРАНИЦА*.

В граничные пиксели выставляем значение яркости 0 (черный), в остальные 255 (белый) или наоборот.

Пример реализации приведен в яндекс диске и на гит-репозитории преподавателя, будет доступен после проведения занятия.

5) Теперь рассмотрим принцип двойной пороговой фильтрации.

Как видно из предыдущего пункта алгоритма, в качестве границы выбрано достаточно большое количество пикселей, не являющихся границей объекта, а лишь показывающих изменение цвета внутри объекта(например, за счет освещения). Для того, чтобы выделить непосредственно границы, применим такой этап, как двойная пороговая фильтрация. Для этого необходимо выбрать два пороговых значения для градиента, например

```
low_level = max_grad // 25
high_level = max_grad // 10
```

Как видно в примере, в качестве пороговых значение удобно брать не конкретные значения, а исходя из максимального градиента по изображению.

И далее производим следующую фильтрацию для пикселей, которые УЖЕ ВЫБРАНЫ КАК ГРАНИЦА на предыдущем шаге: если значение градиента меньше нижней границы, то пиксель не граница, если значение градиента выше- верхней границы, то пиксель точно граница.

После такого фильтра останутся пиксели, значение градиента которых заключено между границами. Для них воспользуемся следующим предположением: если пиксель – это граница, то он не может быть отдельной границей, рядом должен быть еще пиксель с границей.

Исходя из этого предположения добавим проверку на то, что рядом с границей есть другая граница, для чего необходимо проверить 8 пикселей вокруг заданного.

Пример реализации приведен в яндекс диске и на гит-репозитории преподавателя, будет доступен после проведения занятия.

Задание 1. Реализовать метод, который принимает в качестве строки полный адрес файла изображения, читает изображение, переводит его в черно белый цвет и выводит его на экран применяет размытие по Гауссу и выводит полученное изображение на экран.

Задание 2. Модифицировать построенный метод так, чтобы в результате вычислялось и выводилось на экран две матрицы – матрица значений длин и матрица значений углов градиентов всех пикселей изображения.

Задание 3. Модифицировать метод так, чтобы он выполнял подавление немаксимумов и выводил полученное изображение на экран. Рассмотреть изображение, сделать выводы.

Задание 4. Модифицировать метод так, чтобы он выполнял двойную пороговую фильтрацию и выводил полученное изображение на экран.

Задание 5 (самостоятельно). Провести опыты для различных параметров размытия и различных пороговых значений градиента, определить наилучшие параметры для Вашего изображения. Показать преподавателю значения параметров и результат работы на следующем занятии.

Задание 6 (самостоятельно). Реализовать алгоритм Канни на другом языке программирования.

Контрольные вопросы

1. Опишите, в чем заключается задача выявления контуров, и области применения этой задачи.
2. На чем основываются градиентные методы выявления контуров?
3. Опишите основные этапы алгоритма Канни.
4. Что такое градиент пикселя изображения и какие могут возникнуть проблемы с его вычислением? Объясните почему они возникают?
5. Опишите принцип работы оператора Собеля и особенности его использования в алгоритме Канни.
6. Какие операторы возможно использовать вместо оператора Собеля, найдите самостоятельно и опишите, в чем их отличие от оператора Собеля.

7. Каким образом и для чего осуществляется округление угла градиента? Опишите на примере матрицы изображения, зачем хранить угол и для чего его округлять. Поясните на чертеже, как происходит округление.

8. Опишите, в чем суть этапа подавление немаксимумов, покажите роль угла градиента в данном этапе.

9. Опишите, в чем принцип двойной пороговой фильтрации.

Формат оценивания выполнения заданий на лабораторной работе:

- оценка «+» ставится на лабораторной работе, если студент выполняет задания 1, 2;

- оценка «удовлетворительно» ставится на лабораторной работе, если студент выполняет задания 1-3;

- оценка «хорошо» ставится на лабораторной работе, если студент выполняет задачи 1 - 5;

- оценка «отлично» ставится на лабораторной работе, если студент отвечает правильно на все теоретические вопросы.

Если студент сдаёт работу позже, то применяется формат оценивания, указанный в документе «Структура лаб АЦОМ», то есть необходим отчет, гит и полноценная защита лабораторной работы.

Основные источники

1. A Computational Approach to Edge Detection JOHN CANNY, MEMBER, IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, VOL. PAMI-8, NO. 6, NOVEMBER 1986, доступно по ссылке https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.420.3300&rep=rep1&type=pdf, файл доступен в папке литература

2. https://habr.com/ru/post/114589/ - толковая статья с алгоритмом реализации

3. Костюхина Г.В. Модель, метод и комплекс программ выделения контуров на изображениях с использованием энергетических признаков, Диссертация на соискание ученой степени кандидата технических наук, Казань, 2020 – доступна в папке литература

4. Остальная вспомогательная литература доступна в папке литература и в файле «Некоторые интересные онлайн источники» в разделе ЛР3 Методы выделения границ. Алгоритм Канни.