Санкт-Петербургский политехнический университет Петра Великого

Институт машиностроения, материала и транспорта

Высшая школа автоматизации и робототехники

Курсовая работа

Дисциплина: Объектно-ориентированное программирование

Тема: Алгоритм поиска краёв объектов на изображении (Edge Detection)

Выполнил студент гр. 3331506/20102 Артамонов Е.А.

Преподаватель Ананьевский М. С.

Санкт-Петербург

2025

# Содержание

[Введение 2](#_Toc197384450)

[Алгоритм поиска краёв 3](#_Toc197384451)

[Теоретические сведения 3](#_Toc197384452)

[Описание алгоритма 3](#_Toc197384453)

[Преобразование в градации серого 3](#_Toc197384454)

[Обнаружение краёв с помощью алгоритма Кэнни 4](#_Toc197384455)

[Вычисление ориентации краёв 4](#_Toc197384456)

[Классификация краёв 5](#_Toc197384457)

[Математическая структура алгоритма 5](#_Toc197384458)

[Анализ производительности 5](#_Toc197384459)

[Особенности реализации 6](#_Toc197384460)

[Заключение 6](#_Toc197384461)

[Литература 6](#_Toc197384462)

[Приложение 7](#_Toc197384463)

# Введение

Данная курсовая работа посвящена изучению и практической реализации алгоритма поиска краёв объектов на изображении (Edge Detection), который является одной из ключевых технологий в области компьютерного зрения и обработки изображений. Поиск краёв позволяет выделять границы объектов на изображении, что является основой для множества приложений, таких как распознавание объектов, сегментация изображений, анализ текстур и 3D-реконструкция.

**Постановка задачи:** Основной задачей является разработка и анализ алгоритма, способного эффективно определять края объектов на изображении с использованием современных методов обработки изображений. Алгоритм должен быть реализован в рамках объектно-ориентированного подхода, обеспечивать высокую точность обнаружения краёв и быть устойчивым к шумам и вариациям освещения.

**Актуальность и значимость:** Алгоритмы поиска краёв широко применяются в различных областях, включая робототехнику (например, для навигации автономных систем), медицинскую визуализацию (анализ медицинских снимков), графические редакторы и системы видеонаблюдения. Разработка эффективных и быстрых алгоритмов обработки изображений позволяет улучшить качество анализа данных и повысить производительность систем компьютерного зрения.

**Область применения:**

* Компьютерное зрение: распознавание и классификация объектов.
* Робототехника: определение препятствий и навигация.
* Медицина: анализ рентгеновских снимков и МРТ.
* Промышленная автоматизация: контроль качества и дефектоскопия.
* Графические приложения: создание стилизованных изображений, таких как ASCII-арт.

Целью данной работы является изучение теоретических основ алгоритмов поиска краёв, реализация одного из таких алгоритмов в языке Python с использованием объектно-ориентированного подхода и анализ его производительности.

# Алгоритм поиска краёв

Алгоритм поиска краёв (Edge Detection) заключается в обнаружении резких изменений интенсивности пикселей на изображении, которые обычно соответствуют границам объектов. В данной работе рассматривается гибридный подход, который сочетает в себе алгоритм Кэнни (Canny Edge Detection) [1], операторы Собеля и дополнительные методы обработки для повышения качества результата.

## Теоретические сведения

Алгоритм поиска краёв в изображении основывается на следующих принципах:

* **Градиент изображения:** Края определяются как области с высокой величиной градиента, что указывает на резкие изменения интенсивности пикселей.
* **Подавление шума:** Перед обнаружением краёв изображение сглаживается для устранения шумов, которые могут быть ошибочно приняты за края.
* **Ориентация краёв:** Определение направления градиента позволяет классифицировать края по их ориентации (например, горизонтальные, вертикальные или диагональные).
* **Пороговая обработка:** Использование пороговых значений для выделения значимых краёв и подавления слабых.

В данной реализации используется алгоритм Кэнни [1], дополненный операторами Собеля для определения ориентации краёв и последующей их классификации для применения текстур (например, в задачах преобразования изображения в ASCII-арт).

## Описание алгоритма

Алгоритм, реализованный в методе detect\_edges, включает следующие этапы, каждый из которых имеет строгую математическую основу:

### Преобразование в градации серого

Входное изображение в формате RGB преобразуется в изображение в градациях серого с использованием функции cv2.cvtColor [5]. Для пикселя с компонентами интенсивность вычисляется по формуле:

где — интенсивность пикселя в позиции . Эта формула основана на стандарте ITU-R BT.601, учитывающем восприятие яркости человеческим глазом.

### Обнаружение краёв с помощью алгоритма Кэнни [1]

Функция cv2.Canny [5] выполняет обнаружение краёв через следующие шаги:

1. **Сглаживание изображения**: Изображение сглаживается гауссовым фильтром [2, 3]:

* Сглаженное изображение: .

1. **Вычисление градиента**: Градиенты по осям и вычисляются с помощью операторов Собеля:

* Величина градиента:
* Направление градиента:

1. **Подавление немаксимумов**: Пиксели, где не является локальным максимумом в направлении , подавляются.
2. **Пороговая обработка**: Используются два порога, вычисленные на основе медианы интенсивности :

* где . Пиксели классифицируются как краевые, если или и они связаны с краевым пикселем.

Результат — бинарная карта краёв , где для краевых пикселей.

### Вычисление ориентации краёв

Операторы Собеля (cv2.Sobel) [4, 5] применяются для вычисления градиентов и с использованием ядер:

Градиенты: , . Ориентация вычисляется как:

Нормализация: , чтобы .

### Классификация краёв

Изображение разбивается на блоки размером (где ). Размеры выходной карты:

Для каждого блока :

* Извлекается регион карты краёв: .
* Если , вычисляется средняя ориентация:
* Ориентация классифицируется:
  + : ‘\’ (идентификатор 3).
  + : ‘|’ (идентификатор 1).
  + : ‘/’ (идентификатор 4).
  + : ‘-‘ (идентификатор 2).
  + Остальные: ‘ ’
* Если краёв нет, присваивается идентификатор 0.

### Математическая структура алгоритма

Алгоритм можно представить как последовательность преобразований:

1. Вход: .
2. Градации серого: .
3. Сглаживание: .
4. Градиенты: , .
5. Кэнни: .
6. Ориентация: .
7. Классификация: Для блока , если , то , и присваивается идентификатор.
8. Выход: .

## Особенности реализации

* **Адаптивные пороговые значения:** Использование медианы интенсивности для настройки порогов Кэнни.
* **Поддержка текстур:** Интеграция с системой текстур для стилизации изображений.
* **Устойчивость к шуму:** Сглаживание и пороговая обработка минимизируют ложные края.

# Заключение

В ходе выполнения курсовой работы был изучен и реализован алгоритм поиска краёв объектов на изображении, основанный на комбинации алгоритма Кэнни и операторов Собеля. Реализация выполнена в рамках объектно-ориентированного подхода на языке Python с использованием библиотеки OpenCV [4, 5]. Алгоритм показал высокую эффективность и устойчивость к шумам, что делает его пригодным для использования в задачах компьютерного зрения.

Проведённые эксперименты подтвердили приемлемую производительность при обработке изображений высокого разрешения. В будущем возможно улучшение алгоритма путём оптимизации вычислений (например, с использованием GPU) и добавления методов подавления шума.

# Литература

1. Canny, J. A Computational Approach to Edge Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1986.
2. Gonzalez, R.C., Woods, R.E. Digital Image Processing. 4th Edition, 2018.
3. Szeliski, R. Computer Vision: Algorithms and Applications. 2nd Edition, 2022.
4. Bradski, G., Kaehler, A. Learning OpenCV: Computer Vision with the OpenCV Library. 2008.
5. OpenCV Documentation: https://docs.opencv.org/

# Приложение

Рисунок 1 — Алгоритм нахождения краёв на Python



Рисунок 2 — Пример работы алгоритма