Преобразования изображений и фильтрация

Линейные операции, свёртка и пространственные методы обработки

Линейные операторы над изображениями

Оператор $\mathcal{L}$ называется линейным, если выполняется:

$$\mathcal{L}(aI_1 + bI_2) = a\mathcal{L}(I_1) + b\mathcal{L}(I_2)$$

где I_1, I_2 — изображения, a, b — скаляры

Почему это важно?

- Анализ через базисные функции
- Переход в частотную область
- Основа для CNN

Свёртка как базовая операция

Дискретная двумерная свёртка определяется как:

$$(I * K)(x, y) = \sum_{u=-m}^{m} \sum_{v=-n}^{n} I(x-u, y-v) K(u, v)$$

Локальное усреднение

Ядро «скользит» по изображению, комбинируя значения пикселей в окрестности

Размер ядра

Чем больше окно, тем сильнее сглаживание, но и размытие деталей

Свёртка vs корреляция

Свёртка

$$(I * K)(x, y) = \sum I(x - u, y - v)K(u, v)$$

Ядро переворачивается относительно центра

Корреляция

$$(I \star K)(x, y) = \sum I(x + u, y + v)K(u, v)$$

Ядро применяется без переворота

OpenCV и большинство библиотек реализуют корреляцию, называя её свёрткой

Свойства линейности и инвариантности

01

Линейность

$$(I_1 + I_2) * K = I_1 * K + I_2 * K$$

02

Инвариантность к сдвигу

$$T_{\Delta x, \Delta y}(I * K) = (T_{\Delta x, \Delta y} I) * K$$

03

LSI-система

Поведение фильтра не зависит от абсолютного положения структуры

Усредняющий фильтр (box-filter)

Простейшее ядро для квадратного окна $(2k+1) \times (2k+1)$:

$$K(u,v) = \frac{1}{(2k+1)^2}, \quad -k \le u, v \le k$$

Преимущества

Простота реализации, подавление высокочастотного шума

Недостатки

Размывает границы, тонкие линии и мелкие текстуры

```
blur_5 = cv2.blur(img, (5, 5))
k = np.ones((5,5), np.float32) / 25.0
blur_manual = cv2.filter2D(img, -1, k)
```

Гауссов фильтр

Оптимальный сглаживающий оператор с ядром:

$$K(u,v) = \frac{1}{2\pi\sigma^2}\exp\left(-\frac{u^2+v^2}{2\sigma^2}\right)$$

Параметр σ

Управляет степенью размытия. Чем больше σ, тем сильнее сглаживание

Сепарабельность

Двумерная свёртка = две одномерные: сложность снижается с $O(k^2)$ до $O(2k)$

```
gauss_1 = cv2.GaussianBlur(img, (5,5), sigmaX=1.0)
gauss_3 = cv2.GaussianBlur(img, (11,11), sigmaX=3.0)
```

Лапласиан и вторые производные

Дискретный Лапласиан аппроксимирует вторую производную:

$$\nabla^2 I(x,y) \approx I(x+1,y) + I(x-1,y) + I(x,y+1) + I(x,y-1) - 4I(x,y)$$

Ядро 4-связное

$$K_4 = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Ядро 8-связное

$$K_8 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -8 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Усиливает области быстрого изменения яркости, но очень чувствителен к шуму

Реализация свёртки

```
kernel = np.array([[1, 1, 1],
 [1, -8, 1],
 [1, 1, 1]], np.float32)
edges_cv2 = cv2.filter2D(img, cv2.CV_32F, kernel)

# Наивная реализация
def conv2d(image, kernel):
 h, w = image.shape
 kh, kw = kernel.shape
 pad_h, pad_w = kh // 2, kw // 2
 padded = np.pad(image, ((pad_h, pad_h),
 (pad_w, pad_w)), mode="reflect")
 out = np.zeros((h, w), dtype=np.float32)
 for i in range(h):
 for j in range(w):
 region = padded[i:i+kh, j:j+kw]
 out[i, j] = np.sum(region * kernel)
 return out
```

Свёртка и частотная область

Теорема свёртки: свёртка в пространстве = умножение в частотной области

$$\mathcal{F}\{I * K\}(u, v) = F_I(u, v) \cdot F_K(u, v)$$

1

Низкие частоты

Сглаживание: высокие значения H(u,v) вблизи нуля

2

Высокие частоты

Усиление деталей: подавление низких частот

Гауссов фильтр: Фурье-преобразование гауссиана — тоже гауссиан

Свёртка как прототип CNN

Свёрточный слой вычисляет:

$$y_{i,j}^{(k)} = \sigma\left(\sum_{c}\sum_{u,v} w_{u,v}^{(k,c)} x_{i+u,j+v}^{(c)} + b^{(k)}\right)$$

Обучаемые веса

Ядро не задаётся вручную, а обучается градиентным спуском

Нелинейность

Функция активации σ (ReLU) расширяет выразительность

Классификация пространственных фильтров

Сглаживающие (low-pass)

Уменьшают высокочастотные компоненты, подавляют шум

Усиливающие (high-pass)

Подавляют низкие частоты, подчёркивают границы

Комбинированные

Объединяют свойства разных типов (unsharp masking)

Градиентные

Аппроксимируют первую производную, основа детекторов границ

Медианный фильтр

Нелинейный оператор, заменяющий пиксель медианой окрестности:

$$I'(x, y) = \text{median}\{I(x + u, y + v)\}$$

Преимущества

- Эффективен против импульсного шума («соль и перец»)
- Лучше сохраняет края
- Не создаёт серые артефакты

Недостатки

- Вычислительно дороже линейных фильтров
- Может сглаживать мелкие детали
- Требует сортировки значений

```
median = cv2.medianBlur(noisy, 5)
```

Билатеральный фильтр

Учитывает как положение, так и близость яркости пикселей:

$$I'(x,y) = \frac{1}{W} \sum_{(u,v)} I(u,v) \exp\left(-\frac{(x-u)^2 + (y-v)^2}{2\sigma_s^2} - \frac{(I(x,y) - I(u,v))^2}{2\sigma_r^2} \right)$$

σ_s — пространственный масштаб

Определяет размер окна сглаживания

σ_r — яркостное различие

Уменьшает вклад пикселей с сильно отличающейся яркостью

```
bil = cv2.bilateralFilter(img, d=9,
 sigmaColor=50,
 sigmaSpace=7)
```

Повышение резкости (Unsharp Masking)

1 2 3

Шаг 1

Сглаживание

$$I_{\text{blur}} = G_\sigma * I$$

Шаг 2

Высокие частоты

$$H = I - I_{\text{blur}}$$

Шаг 3

Усиление

$$I_{\text{sharp}} = I + \lambda H$$

```
blur = cv2.GaussianBlur(img, (5,5), 1.0)
sharp = cv2.addWeighted(img, 1.0 + 1.5,
 blur, -1.5, 0)
```

Операторы Собеля и Превитта

Аппроксимируют градиент яркости с встроенным сглаживанием

Собель

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

$$G_y = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Превитт

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$P_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$$

```
gx = cv2.Sobel(img, cv2.CV_32F, 1, 0, ksize=3)
gy = cv2.Sobel(img, cv2.CV_32F, 0, 1, ksize=3)
mag = cv2.magnitude(gx, gy)
```

Фильтр Лапласа–Гаусса (LoG)

Комбинация сглаживания и выделения границ:

$$\nabla^2(G_\sigma * I) = (\nabla^2 G_\sigma) * I$$

Устойчивость к шуму

Предварительное гауссово сглаживание подавляет высокочастотный шум

Выделение структур

Лапласиан усиливает области быстрого изменения яркости

```
blur = cv2.GaussianBlur(img, (5,5), 1.0)
lap = cv2.Laplacian(blur, cv2.CV_32F)
```

Сравнение фильтров

Box-filter

Сильное усреднение, размывает детали, простая реализация

Гауссов

Мягкое сглаживание, сохраняет структуру, оптимален для шума

Медианный

Устраняет импульсный шум, сохраняет края, нелинейный

Билатеральный

Сглаживает однородные области, сохраняет границы, медленный

Фильтрация как предобработка

01

Подавление шума

Сглаживающие фильтры устраняют высокочастотные искажения

02

Выделение структур

Градиентные операторы подчёркивают границы и контуры

03

Подготовка к анализу

Улучшение качества для сегментации, детекции и распознавания

Переход к частотной области

Пространственная фильтрация показывает, как преобразования изменяют изображение в координатном пространстве

Частотная фильтрация раскрывает действие операций в спектральной области, где эффекты сглаживания и усиления становятся особенно наглядными

Двумерное ДПФ для изображений

Дискретное преобразование Фурье переводит изображение в частотную область:

$$F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} I(x,y) e^{-j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

Обратное преобразование восстанавливает изображение:

$$I(x,y) = \frac{1}{MN} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{j2\pi\left(\frac{ux}{M} + \frac{vy}{N}\right)}$$

Низкие частоты

Вблизи центра спектра, плавные изменения яркости

Высокие частоты

Ближе к краям, резкие изменения и детали

Амплитудный и фазовый спектр

Амплитуда

$$A(u,v) = |F(u,v)|$$

Отражает «силу» частоты — насколько выражены структуры с данной частотой

Фаза

$$\phi(u,v) = \arg F(u,v)$$

Кодирует пространственное расположение деталей, форму объектов и контуры

Структура изображения в значительной степени закодирована в фазе, а не в амплитуде

Низкочастотные фильтры

Идеальный частотный low-pass фильтр:

$$H(u,v) = \begin{cases} 1, & D(u,v) \le D_0 \\ 0, & D(u,v) > D_0 \end{cases}$$

Преимущество

Полностью удаляет высокие частоты, оставляя крупные формы

Недостаток

Резкое обрезание вызывает эффект Гиббса — осцилляции вокруг границ

Гауссов и Баттервортов фильтры

Гауссов low-pass

$$H(u,v) = \exp\left(-\frac{D(u,v)^2}{2D_0^2}\right)$$

Плавный переход, нет эффекта Гиббса

Баттервортов фильтр

$$H(u,v) = \frac{1}{1 + \left(\frac{D(u,v)}{D_0}\right)^{2n}}$$

Параметр n регулирует крутизну среза

Высокочастотные фильтры

High-pass фильтр строится как дополнение к low-pass:

$$H_{\text{high}}(u, v) = 1 - H_{\text{low}}(u, v)$$

Эффект

Ослабляет низкие частоты, усиливает высокие — границы, детали, текстуры

Применение

Повышение резкости, выделение контуров, усиление локального контраста

Риск

Усиливает не только детали, но и шум — требуется осторожность

Реализация частотной фильтрации

```
F = np.fft.fft2(img)
F_shift = np.fft.fftshift(F)

rows, cols = img.shape
crow, ccol = rows//2, cols//2

# Маска низких частот радиуса D0
D0 = 40
mask = np.zeros_like(img)
for u in range(rows):
 for v in range(cols):
 if (u-crow)**2 + (v-ccol)**2 <= D0**2:
 mask[u,v] = 1.0

F_filtered = F_shift * mask
F_ishift = np.fft.ifftshift(F_filtered)
img_back = np.fft.ifft2(F_ishift)
img_back = np.abs(img_back)
```

Эффект Гиббса

Резкое обрезание частот приводит к осцилляциям в пространственной области

- Идеальные фильтры редко применяются на практике — предпочтительны плавные переходы (гауссовы, баттервортовы)

Гомоморфная фильтрация

Изображение как произведение освещённости и отражения:

$$I(x,y) = L(x,y)R(x,y)$$

1

2

3

Логарифм

$$\log I = \log L + \log R$$

High-pass фильтр

Подавляет низкие частоты (освещённость)

Экспонента

$$I' = \exp(\mathrm{HPF}(\log I))$$

Устраняет неравномерность освещённости, усиливает детали

Связь пространственной и частотной фильтрации

Гауссово сглаживание	Лапласиан	Собель
Гауссов фильтр в пространстве ↔️ гауссова маска в частотной области	Оператор второй производной ↔️ передаточная функция растёт с частотой	Выделяет горизонтальные частоты ↔️ диагональные максимумы в спектре

Частотный анализ фильтров

Спектральный анализ — универсальный язык описания фильтров

Сглаживание

Подавление высоких частот

Резкость

Усиление высоких частот

Границы

Выделение переходов яркости

Освещённость

Подавление низких частот

Морфологические операции

Нелинейные преобразования, основанные на анализе геометрической структуры объектов

Основа

Операции минимума и максимума в пределах структурирующего элемента

Структурирующий элемент

Матрица, описывающая форму и размеры локальной области анализа

Применение

Очистка масок, выделение границ, анализ топологии объектов

Эрозия

Уменьшение объектов изображения через операцию минимума:

$$(I \ominus B)(x, y) = \min_{(u,v) \in B} I(x + u, y + v)$$

Эффект

- «Съедает» яркие участки
- Границы отступают внутрь
- Удаляет тонкие детали

Применение

- Подавление мелких фрагментов
- Удаление шумовых точек
- Разделение объектов

Дилатация

Расширение объектов через операцию максимума:

$$(I \oplus B)(x, y) = \max_{(u,v) \in B} I(x - u, y - v)$$

Эффект

Увеличивает белые области, заполняет узкие разрывы, соединяет близкие объекты

Применение

Устранение разрывов в масках, увеличение области объекта, создание защитного слоя

Открытие и закрытие

Открытие

$$I \circ B = (I \ominus B) \oplus B$$

Эрозия → дилатация

- Удаляет мелкие элементы
- Сглаживает контуры
- Устраняет шумовые включения

Закрытие

$$I \bullet B = (I \oplus B) \ominus B$$

Дилатация → эрозия

- Заполняет отверстия
- Устраняет разрывы
- Объединяет фрагменты

Бинарные и полутоновые изображения

Бинарные маски

Морфология работает с принадлежностью к объекту или фону. Операции имеют естественный геометрический смысл: очистка, объединение, восстановление структуры

Полутоновые изображения

Операции над уровнями яркости. Эрозия уменьшает яркие участки, дилатация усиливает их. Полезно для локального подавления особенностей

Реализация морфологии в OpenCV

```
kernel = cv2.getStructuringElement(
    cv2.MORPH_ELLIPSE, (5,5))

eroded = cv2.erode(mask, kernel, iterations=1)
dilated = cv2.dilate(mask, kernel, iterations=1)

opened = cv2.morphologyEx(mask,
                cv2.MORPH_OPEN, kernel)
closed = cv2.morphologyEx(mask,
                cv2.MORPH_CLOSE, kernel)
```

1

Эллиптический

Сохраняет округлые объекты

2

Квадратный

Удобен для технических структур

3

Крестовидный

Для тонких линий

Морфологический градиент

Разница между дилатацией и эрозией выделяет границы:

$$\mathrm{grad}(I) = (I \oplus B) - (I \ominus B)$$

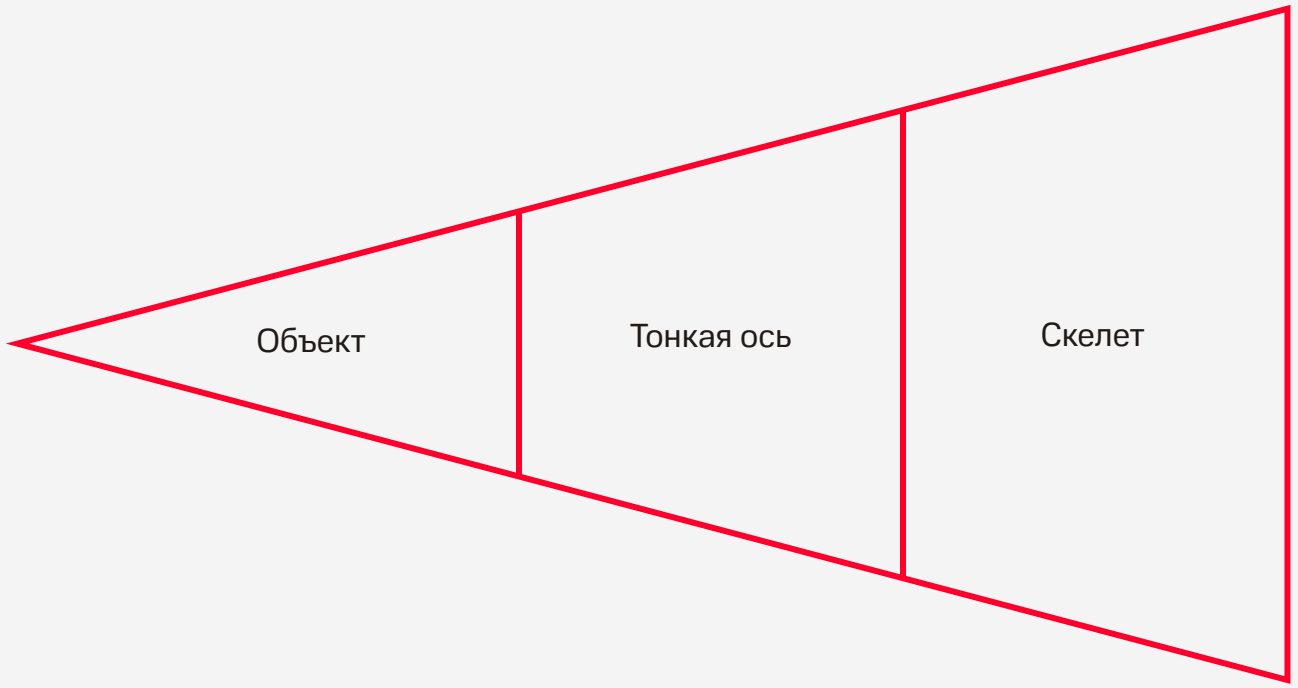
Преимущество

Менее чувствителен к шуму, чем линейные градиентные фильтры

Особенность

Точнее следует форме объекта, зависит от геометрии структурирующего элемента

Скелетизация и топология



Скелетизация сводит объект к линии минимальной толщины, сохраняя топологию

01

Анализ символов

Рукописные буквы, цифры

02

Медицинские снимки

Форма сосудов, структура тканей

03

Дорожная разметка

Линии, контуры дорог

Морфология vs линейная фильтрация

Морфология

- Нелинейные операции
- Работает с формой объектов
- Точное воздействие на бинарные структуры
- Не создаёт промежуточных значений

Линейная фильтрация

- Взвешенные суммы значений
- Работает с яркостью
- Размывает границы на бинарных масках
- Требует порогования для восстановления

На практике методы дополняют друг друга: сглаживание → сегментация → морфология

Морфология как предобработка

1

После сегментации

Очистка масок, заполнение разрывов, объединение компонент

2

Перед анализом контуров

Обеспечение целостности и корректности формы объектов

3

Ранние этапы

Адаптивная пороговая обработка, выделение текстур

Понятие границы и роль контуров

Граница — область резкого изменения интенсивности, где локальная структура меняется быстрее окружающих регионов

Для человека

Основной носитель формы: распознавание объектов, оценка положения и глубины

Для алгоритмов

Структурные элементы для сегментации, анализа формы, сопоставления

Математически

Максимум модуля градиента яркости с учётом направления и контекста

Градиент как мера изменения

Двумерный вектор, указывающий направление наибыстрейшего изменения яркости:

$$\nabla I = \left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right]$$

Модуль градиента:

$$|\nabla I| = \sqrt{I_x^2 + I_y^2}$$

Модуль

Интенсивность границы — насколько быстро меняется яркость

Направление

Ориентация границы — куда направлена нормаль к контуру

Операторы Собеля и Прюитта

Аппроксимируют производные с встроенным сглаживанием

Собель (более устойчив)

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Прюитт

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

```
gx = cv2.Sobel(img, cv2.CV_32F, 1, 0, ksize=3)
gy = cv2.Sobel(img, cv2.CV_32F, 0, 1, ksize=3)
mag = cv2.magnitude(gx, gy)
```

Основа многих алгоритмов детекции признаков и анализа границ

Лапласиан и нули переходов

Оператор второй производной, изотропный к изменениям в любом направлении

Ядро

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

Zero-crossings

Точки, где отклик Лапласиана пересекает ноль — кандидаты на границы

Метод Марра–Хилдретта

Использует нули переходов после сглаживания для выделения границ

Алгоритм Кэнни

1

Сглаживание

Гауссов фильтр подавляет шум

2

Градиент

Вычисление модуля и направления

3

Подавление немаксимумов

Оставляем локальные максимумы вдоль градиента

4

Двустадийное порогование

Гистерезис для связности границ

```
edges = cv2.Canny(img, threshold1=50,
 threshold2=150)
```

Настройка порогов в Кэнни

Нижний порог

Определяет недостаточно сильные градиенты. Слишком низкий → ложные границы

Верхний порог

Определяет явно выраженные границы. Слишком высокий → пропуск деталей

Сглаживание

Недостаточное → шумовые максимумы. Чрезмерное → потеря тонких линий

Используют адаптивные методы: выбор порогов на основе статистики градиента или локальные пороги

Сравнение детекторов границ

Собель/Превитт

Быстрые, хорошая первичная оценка, могут давать разрывы

Лапласиан

Нули переходов, чувствителен к шуму, требует сглаживания

Морфологический

Устойчив к шуму, зависит от структурирующего элемента

Кэнни

Лучшее сочетание точности, устойчивости и непрерывности

Постобработка контуров

```
contours, hierarchy = cv2.findContours(
    edges, cv2.RETR_EXTERNAL,
    cv2.CHAIN_APPROX_SIMPLE)

# Фильтрация по площади
filtered = [c for c in contours
            if cv2.contourArea(c) > 100]

# Аппроксимация
approx = cv2.approxPolyDP(contour,
                          epsilon=0.01*perimeter,
                          closed=True)
```

01

Поиск контуров

Связные кривые, представляющие границы

02

Фильтрация

По площади, длине, форме

03

Аппроксимация

Упрощение формы контура

Направление границ

Градиент содержит информацию о силе и направлении границы

Угол направления

Показывает, куда направлена нормаль к границе, определяет ориентацию объектов

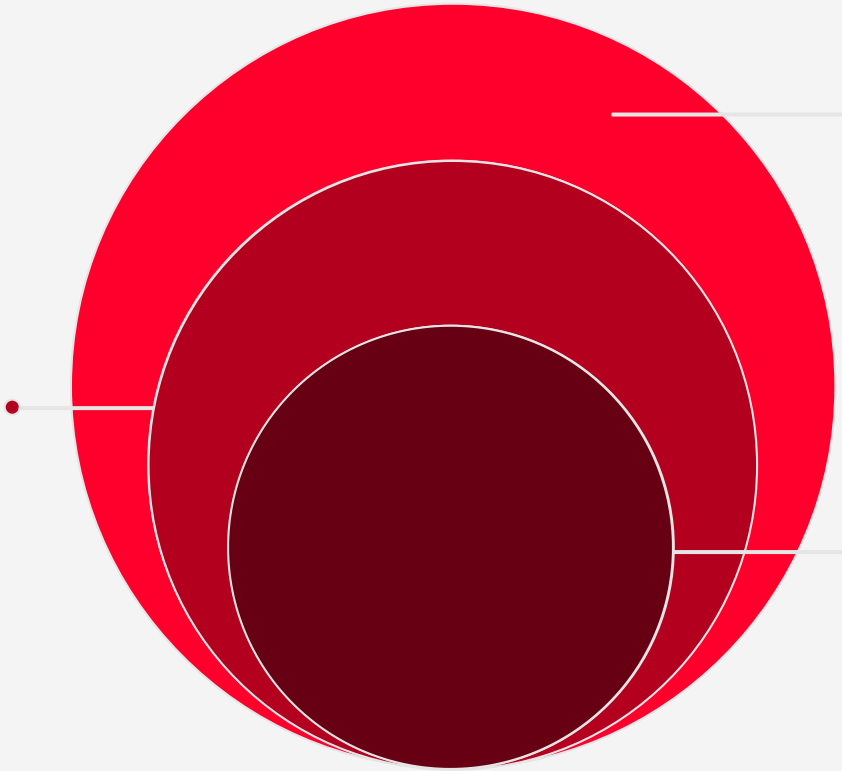
HOG дескрипторы

Гистограммы ориентированных градиентов — устойчивый признак для распознавания

Анализ текстур

Распознавание направлений линий, выделение характерных форм

Границы как фундамент признаков



Ключевые точки

Опорные дескрипторы

Выделенные границы

Фундамент для признаков

Исходное изображение

Сырые пиксели и оттенки

Границы — основа для ключевых точек, дескрипторов, распознавания объектов, восстановления структуры и анализа форм

Фильтры как детекторы признаков

Каждый фильтр чувствителен к своему локальному паттерну

Горизонтальный градиент

Вертикальные границы

Вертикальный градиент

Горизонтальные границы

Угловые структуры

Пересечения линий

Текстурные элементы

Повторяющиеся паттерны

Фильтры формируют первичные признаки для распознавания, сегментации и сопоставления

От фиксированных к обучаемым фильтрам

Классические методы

- Параметры задаёт инженер
- Ограниченный набор фильтров
- Не адаптируются к данным
- Требуют экспертных знаний

CNN

- Фильтры обучаются автоматически
- Градиентный спуск подбирает веса
- Адаптация под конкретную задачу
- Находят сложные паттерны

Свёрточный слой как обобщение

Свёрточный слой расширяет линейную фильтрацию:

$$y_{i,j}^{(k)} = \sigma\left(\sum_{c}\sum_{u,v} w_{u,v}^{(k,c)} x_{i+u,j+v}^{(c)} + b^{(k)}\right)$$

Многоканальность

Суммирование по входным каналам: цвет, текстуры, промежуточные признаки

Нелинейность

Функция активации σ (ReLU) расширяет выразительность

Обучаемость

Веса оптимизируются под задачу

Низкоуровневые фильтры в CNN

Первые слои CNN обучаются фильтрам, похожим на классические операторы

Края под разными углами

Напоминают фильтры Собеля и Превитта

Текстурные элементы

Полосы, точки, угловые структуры

Габоровы функции

Направленные полосы с гауссовым окном

Первые слои CNN фактически выполняют функцию преобразования в низкоуровневые признаки

Фильтры Габора

Модель отклика нейронов первичной зрительной коры:

$$g(x, y) = \exp\left(-\frac{x'^2 + \gamma^2 y'^2}{2\sigma^2}\right)\cos(2\pi f x' + \phi)$$

Чувствительность

К направлению и пространственной частоте — идеальны для текстур

Сходство с CNN

Обученные ядра часто принимают форму направленных полос

Мультимасштабный анализ

Классическое CV

Пирамиды изображений: гауссова и лапласианова. Анализ на разных масштабах через многократное сглаживание и уменьшение размера

CNN

Глубокие слои имеют большие поля восприятия. Архитектуры с пропускными соединениями (U-Net, FPN) используют несколько уровней разрешения

Регуляризация и сглаживание

Классические фильтры сглаживания — аналог регуляризации в нейросетях

Проблема

Сильные колебания весов → переобучение, чувствительность к шуму

Решение

Регуляризация «сглаживает» веса: ограничение нормы, ранняя остановка, уменьшение скорости обучения

Интерпретируемость фильтров

01

Визуализация ядер

Отображение обученных весов фильтров

02

Анализ активаций

Изучение откликов на тестовые изображения

03

Поиск паттернов

Определение локальных структур, которые выделяет фильтр

Интерпретируемость сводится к тем же принципам, что и для классических операторов

Иерархия признаков

Сложные признаки

Элементы объектов: колёса, глаза, крылья

Средний уровень

Углы, пересечения, фрагменты форм

Базовые структуры

Контуры, текстуры, линии

Каждый слой работает на результатах предыдущего, увеличивая степень абстракции

Фильтрация — фундамент CNN

Свёртка лежит в основе всех свёрточных нейронных сетей

Классические методы

Фильтры, градиенты, морфология, частотный анализ — фундамент обработки изображений

Обучаемые методы

CNN повторяют базовые операции фильтрации, делая их адаптивными и многослойными

Единство подхода

От линейной фильтрации до обучаемых признаков — один принцип локального преобразования