Введение в Компьютерное Зрение
Лекция №2, осень 2020

Обработка сигналов

ЛИТПМ

МФТИ

Кафедра технологий проектирования сложных технических систем

Мотивация к обработке изображений

De-noising

Super-resolution

In-painting

Мотивация к обработке изображений

Бинаризация

Выделение компонент связности

Выделение краев

План лекции

- Представление изображения в частотной области. Преобразование Фурье

- Системы и фильтры

- Свертки

Изображение как дискретная функция

- Изображения обычно цифровые (дискретные):
  - Пример 2D пространства на регулярной сетке

- Представлено в виде матрицы целочисленных значений

Pix

m

n

62	79	23	119	120	105	4	0
10	10	9	62	12	78	34	0
10	58	197	46	46	0	0	48
176	135	5	188	191	68	0	49
2	1	1	29	26	37	0	77
0	89	144	147	187	102	62	208
255	252	0	166	123	62	0	31
166	63	127	17	1	0	99	30

Изображение как дискретная функция

Декартовые координаты

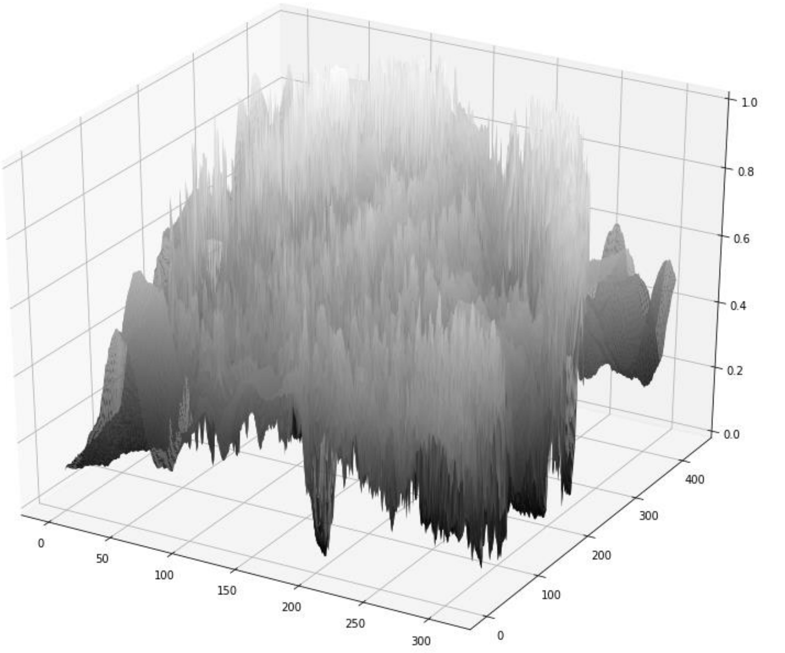
$$f[n,m] = \begin{bmatrix} \ddots & & \vdots & & \\ & f[-1,1] & f[0,1] & f[1,1] & \\ \dots & f[-1,0] & \underline{f[0,0]} & f[1,0] & \dots \\ & f[-1,-1] & f[0,-1] & f[1,-1] & \\ & & \vdots & & \ddots \end{bmatrix}$$

Изображение как дискретная функция

Изображение как функция *f* от R^2 до R^M:

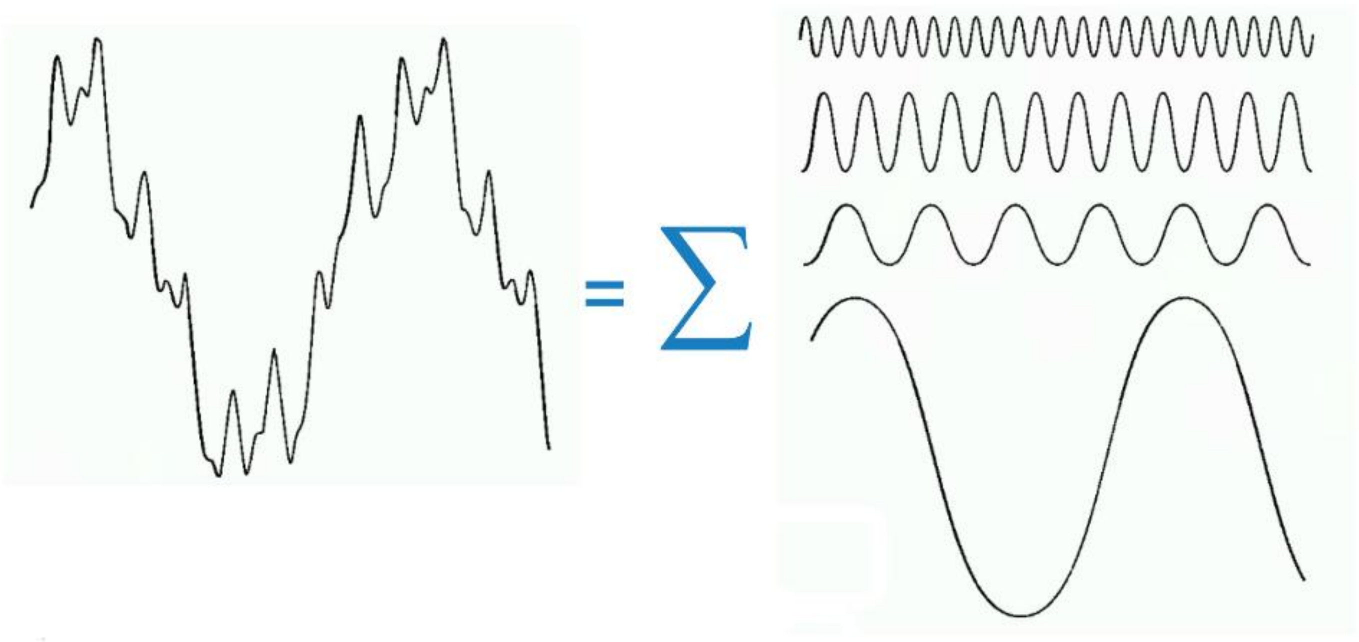
- *f*(*x, y*) дает интенсивность в позиции (*x, y*)
- Определяется через прямоугольник, с конечным диапазоном:

  $f: [a,b] \times [c,d] \rightarrow [0,255]$



Ряд Фурье

Периодический сигнал может быть представлен в виде суммы



Ряд и преобразование Фурье

Ряд Фурье́ — представление функции f с периодом τ в виде ряда

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{+\infty} A_k \cos\left(k\frac{2\pi}{\tau}x + \theta_k\right)$$

Этот ряд может быть также записан в виде

$$f(x) = \sum_{k=-\infty}^{+\infty} \hat{f}_k e^{ik\frac{2\pi}{\tau}x},$$

где

A_k — амплитуда k-го гармонического колебания,

$k\frac{2\pi}{\tau} = k\omega$ — круговая частота гармонического колебания,

θ_k — начальная фаза k-го колебания,

$\hat{f}_k$ — k-я комплексная амплитуда

Преобразование Фурье

Прямое $$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ix\omega}\, dx.$$

Обратное $$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{ix\omega}\, d\omega$$

Преобразование Фурье для двумерного случая

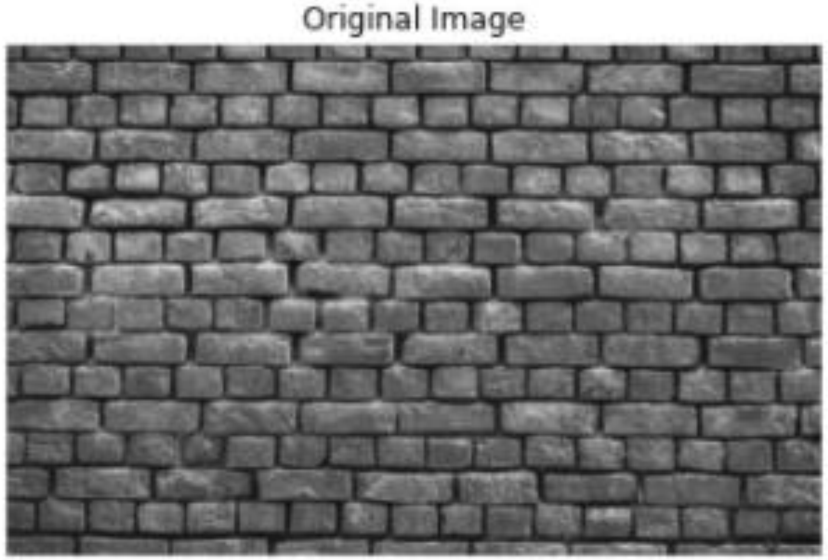
Прямое преобразование

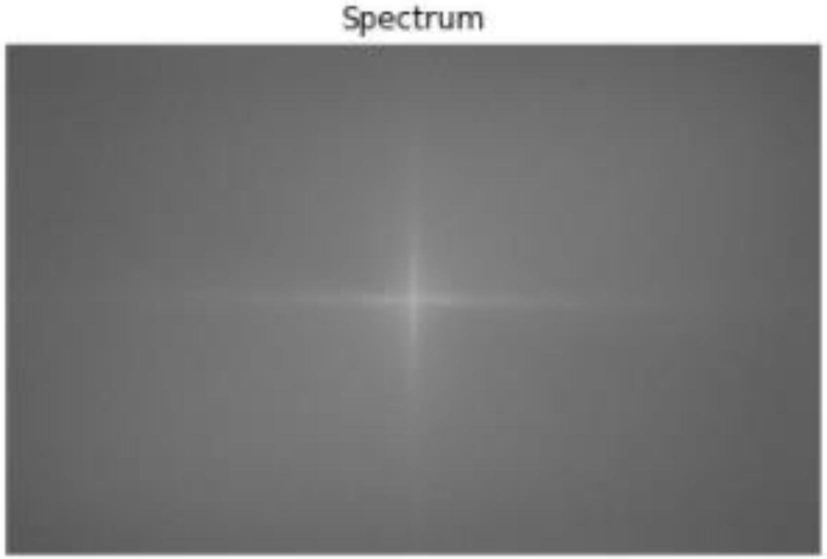
$$F(k,l) = \sum_{p=0}^{N-1}\sum_{q=0}^{N-1} f(p,q)e^{-2i\pi\left(\frac{kp}{N}+\frac{lq}{N}\right)}$$

Обратное преобразование

$$f(p,q) = \frac{1}{N^2}\sum_{k=0}^{N-1}\sum_{l=0}^{N-1} F(k,l)e^{-2i\pi\left(\frac{kp}{N}+\frac{lq}{N}\right)}$$

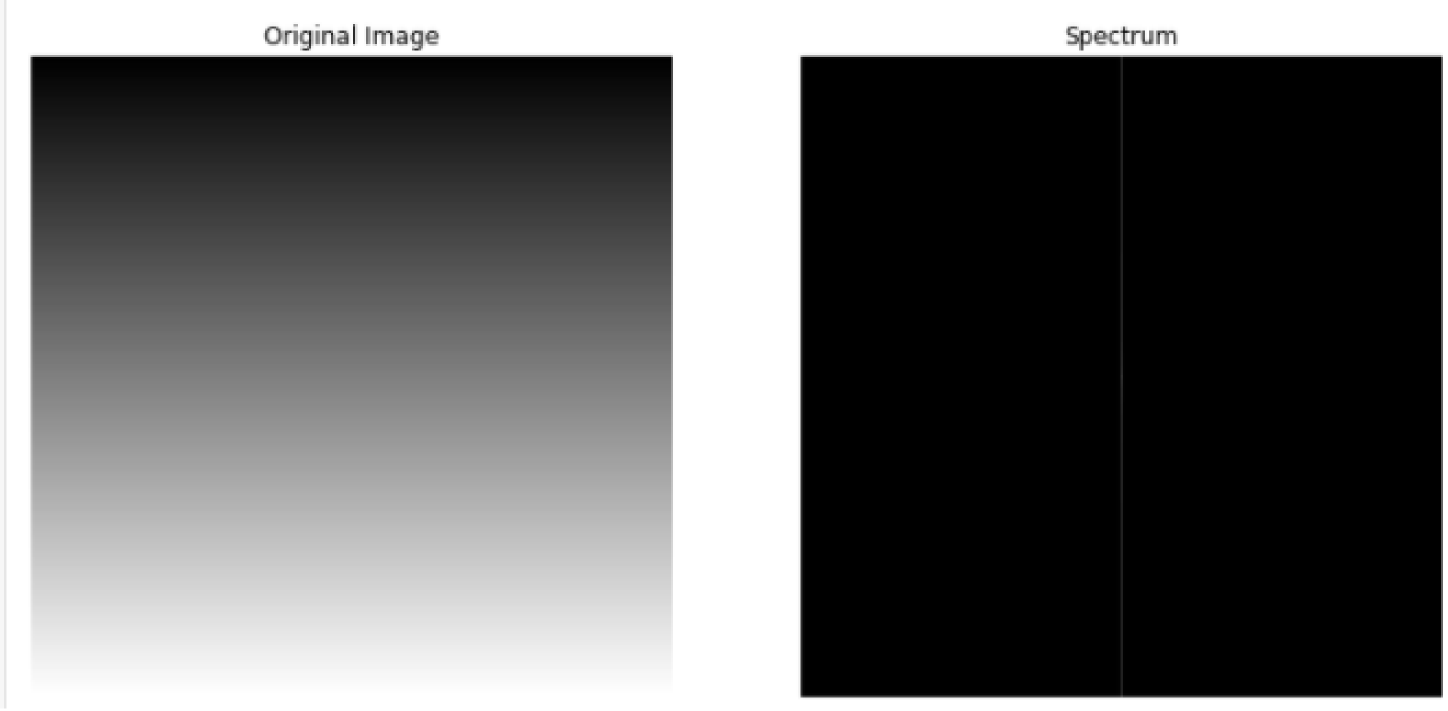
Преобразования Фурье для изображений





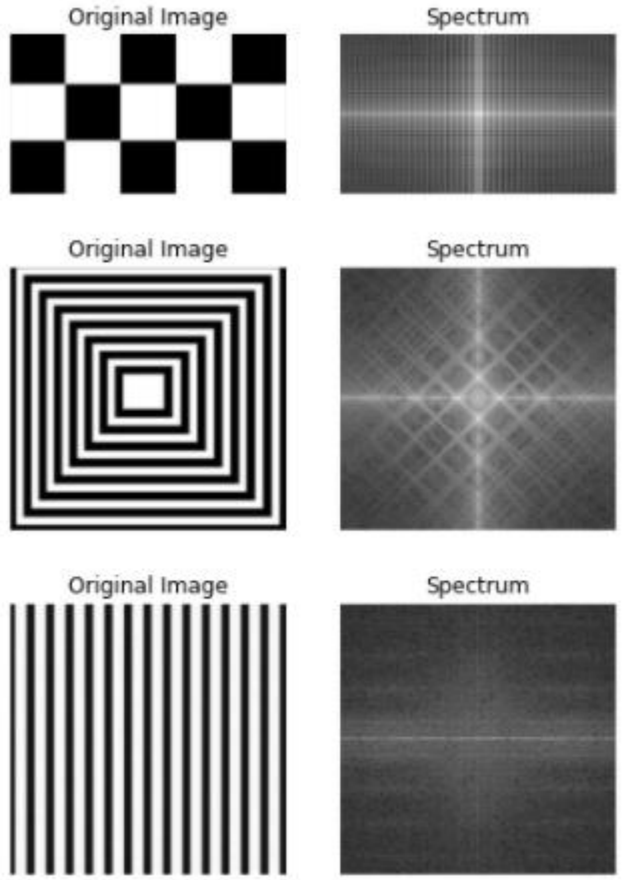
«Высокие» частоты: область <u>с сильными и частыми</u> перепадами значений пикселей

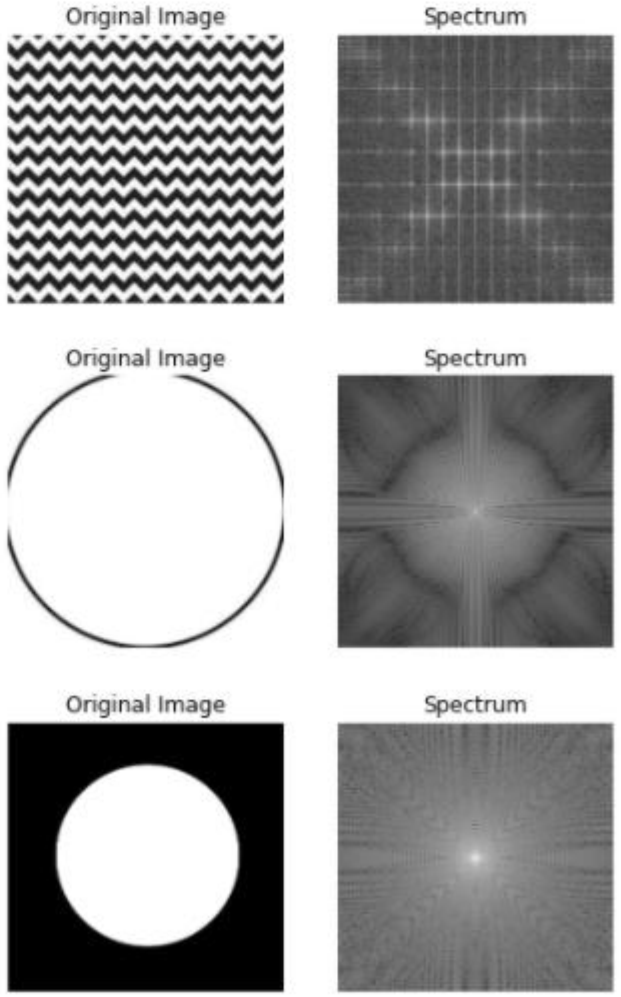
Преобразования Фурье для изображений



«Низкие» частоты: области <u>с слабыми и редкими</u> перепадами значений пикселей

Интерпретация спектра изображения





План лекции

- Представление изображения в частотной области. Преобразование Фурье
- **Системы и фильтры**
- Свертки

Системы и фильтры

Фильтрация – формирование нового изображения, значения пикселей которого трансформируются из исходных значений пикселей.

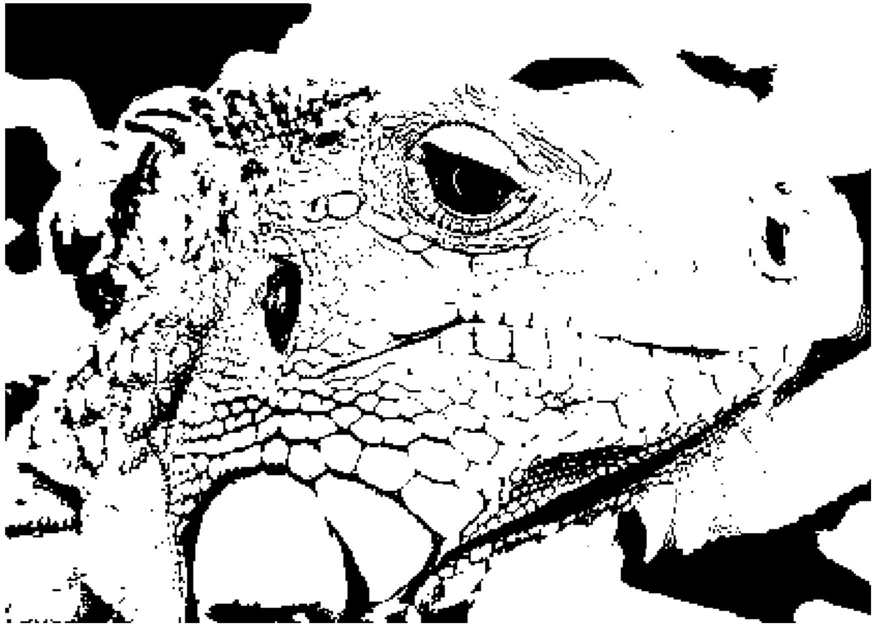
Мотивация:

- Выделить полезную информацию
- Изменить или улучшить свойства полезных признаков на изображении

Интуитивное понимание систем

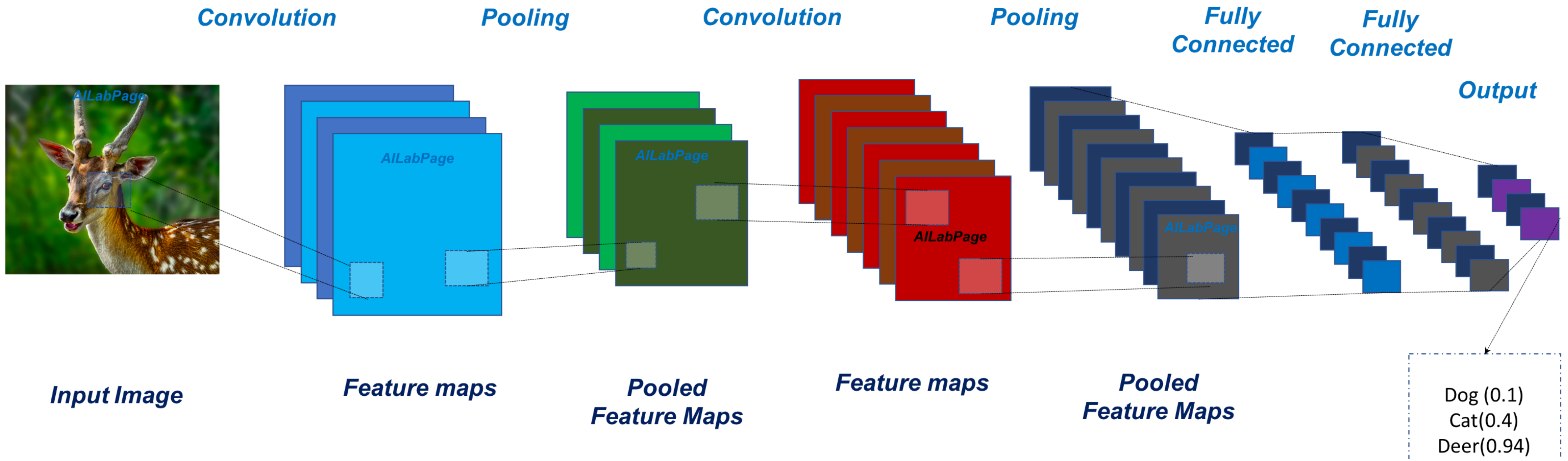
Мы рассмотрим линейные системы как вид функции, которая применяется к изображениям, как двумерным функциям.

Преобразование изображения или его умножение на константу оставляет семантическое содержание нетронутым – можно выделить некоторые закономерности.



Кстати говоря…

Нейронные сети и, в частности, сверточные нейронные сети – это тип системы или нелинейная система, содержащая несколько отдельных линейных подсистем.



(подробнее об этом в другом курсе)

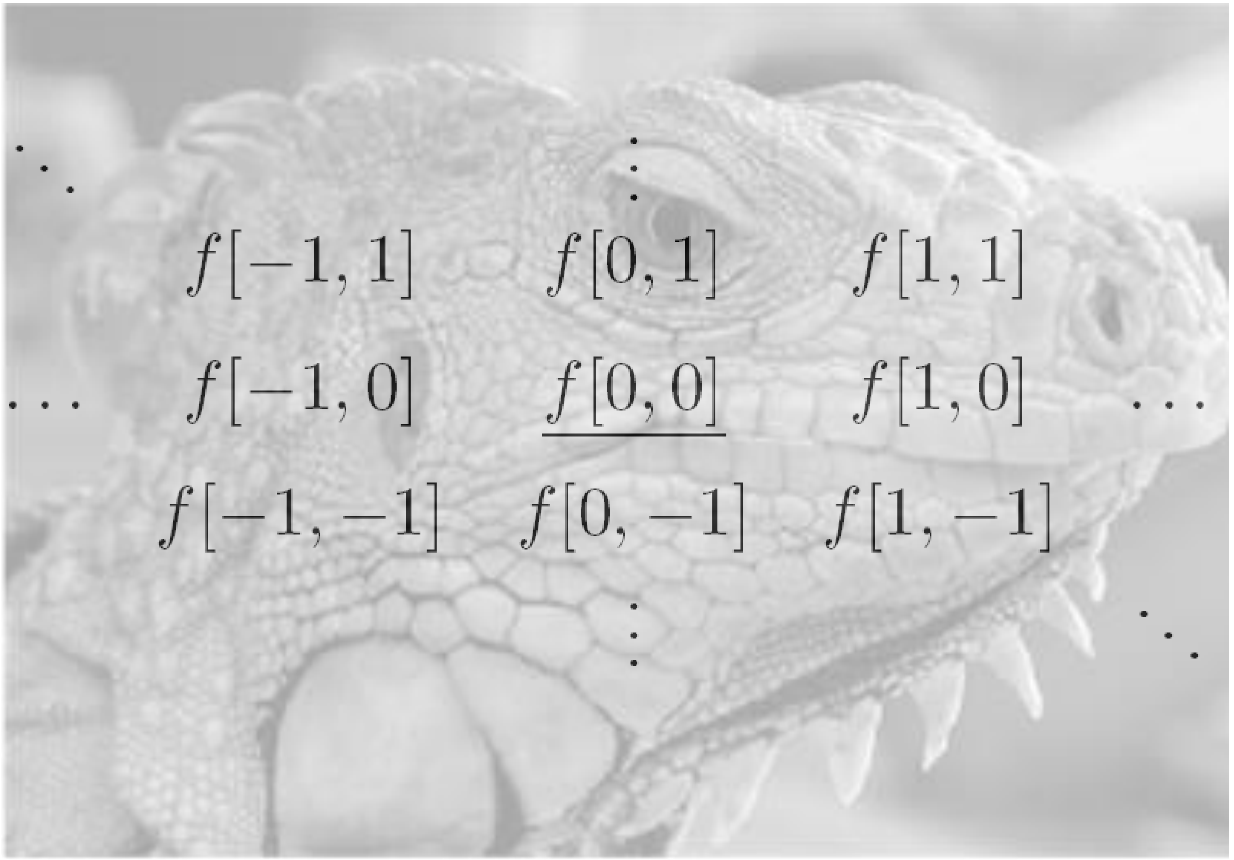
Системы и фильтры

Определим **систему** как единицу, которая преобразует входную функцию $f[n,m]$ в выходную (или ответную) функцию $g[n,m]$, где (n,m) являются независимыми переменными.

В случае изображений (n,m) представляет пространственное положение на изображении.

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

Изображение как дискретная функция



$$f[n,m] = \begin{bmatrix} \ddots & & \vdots & & \\ & f[-1,1] & f[0,1] & f[1,1] & \\ \dots & f[-1,0] & \underline{f[0,0]} & f[1,0] & \dots \\ & f[-1,-1] & f[0,-1] & f[1,-1] & \\ & & \vdots & & \ddots \end{bmatrix}$$

Двумерные дискретные системы (фильтры)

S – системный оператор, определяемый как отображение или назначение члена набора возможных выходов $g[n,m]$ каждому члену набора возможных входов $f[n,m]$.

S – системный оператор, определяемый как отображение или назначение члена набора возможных выходов $g[n,m]$ каждому члену набора возможных входов $f[n,m]$.

$$g = \mathcal{S}[f], \quad g[n,m] = \mathcal{S}\{f[n,m]\}$$

$$f[n,m] \xrightarrow{\mathcal{S}} g[n,m]$$

Пример фильтра №1: Размытие

Пример фильтра №1: Размытие

2D DS moving average over a 3 × 3 window of neighborhood

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} f[n-k, m-l]$$

$$h = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Пример фильтра №1: Размытие

$\mathrm{f}[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\mathrm{g}[n, m]$

	0								

Пример фильтра №1: Размытие

$f[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n, m]$

	0	10							

Пример фильтра №1: Размытие

$\mathrm{f}[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\mathrm{g}[n, m]$

	0	10	20						

Пример фильтра №1: Размытие

$\mathrm{f}[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\mathrm{g}[n, m]$

	0	10	20	30					

Пример фильтра №1: Размытие

$\text{f}[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\text{g}[n, m]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Пример фильтра №1: Размытие

Подводя итог:

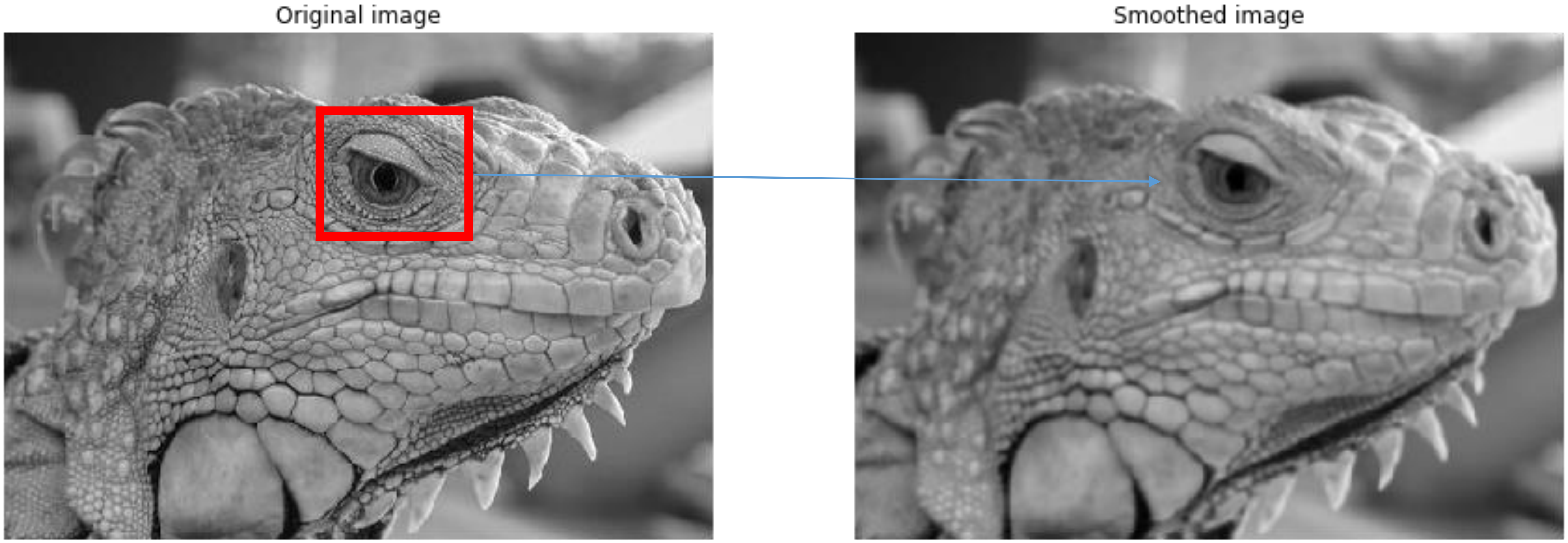
- Данный фильтр "Заменяет" каждый пиксель средним значением по окрестностям.

- Достигается эффект сглаживания (осреднение резких переходов значений пикселей).

h

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

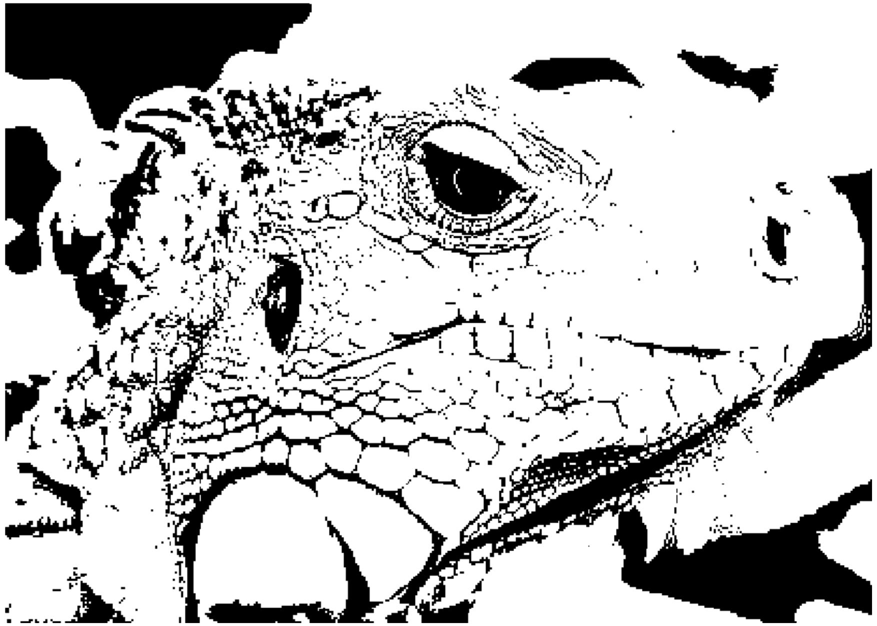
Пример фильтра №1: Размытие



ПЕРЕДЕЛАТЬ !!!

Пример фильтра №2: Пороговое правило

$$g[n,m] = \begin{cases} 1, & f[n,m] > 100 \\ 0, & \text{otherwise.} \end{cases}$$



Свойства системы

Амплитудные свойства:

- Аддитивность $S[f_i[n,m] + f_j[n,m]] = S[f_i[n,m]] + S[f_j[n,m]]$
- Однородность $S[\alpha f_i[n,m]] = \alpha S[f_i[n,m]]]$
- Суперпозиция $S[\alpha f_i[n,m] + \beta f_j[n,m]] = \alpha S[f_i[n,m]] + \beta S[f_j[n,m]]$
- Стабильность $|f[n,m]| \leq k \implies |g[n,m]| \leq ck$
- Инвертируемость $S^{-1}[S[f_i[n,m]]] = f[n,m]$

Свойства системы

Пространственные свойства:

- Вырожденность $$\text{for } n < n_0, m < m_0, \text{ if } f[n,m] = 0 \implies g[n,m] = 0$$
- Инвариантность сдвига $$f[n-n_0, m-m_0] \xrightarrow{\mathcal{S}} g[n-n_0, m-m_0]$$

Размытие инвариантно к сдвигу?

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

f$[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

g$[n, m]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Размытие инвариантно к сдвигу?

$$f[n,m] \xrightarrow{\mathcal{S}} g[n,m]$$

$$g[n,m] = \frac{1}{9}\sum_{k=n-1}^{n+1}\sum_{l=m-1}^{m+1} f[k,l] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-k,m-l]$$

$$\mathrm{g}[n-n_0,m-m_0] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-n_0-k,m-m_0-l]$$

Пусть $f[n-n_0,m-m_0]$ будет смещенным входом от $f[n,m]$

Теперрь подставим это в систему $f[n-n_0,m-m_0]$:

$$f[n-n_0,m-m_0] \xrightarrow{S} \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1} f[n-n_0-k,m-m_0-l]$$

Да!

Размытие простое?

$$g[n,m] = \frac{1}{9} \sum_{k=n-1}^{n+1} \sum_{l=m-1}^{m+1} f[k,l]$$

$\mathrm{f}[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\mathrm{g}[n, m]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

$$\text{for } n < n_0, m < m_0, \text{ if } f[n,m] = 0 \implies g[n,m] = 0$$

Линейные системы (фильтры)

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

Линейная фильтрация:

- Сформировать новое изображение, пиксели которого представляют собой взвешенную сумму исходных значений пикселей.
- Используйте один и тот же набор весов в каждой точке.

- **S** – это <u>линейная система </u>(функция), если она удовлетворяет условию:

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\,f_i[n,m]] + \beta\, S[\,f_j[k,l]\,]$$

Линейные системы (фильтры)

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

- Размытие это линейная система?

- Пороговое правило это линейная система?
  - f1[n,m] + f2[n,m] > T
  - f1[n,m] < T
  - f2[n,m] < T

Нет!

Линейные инвариантные системы

Удовлетворяют следующим свойствам:

- Суперпозиция

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\, f_i[n,m]] + \beta\, S[\, f_j[k,l]\,]$$

- Инвариантность к сдвигу:

$$f[n-n_0, m-m_0] \xrightarrow{\mathcal{S}} g[n-n_0, m-m_0]$$

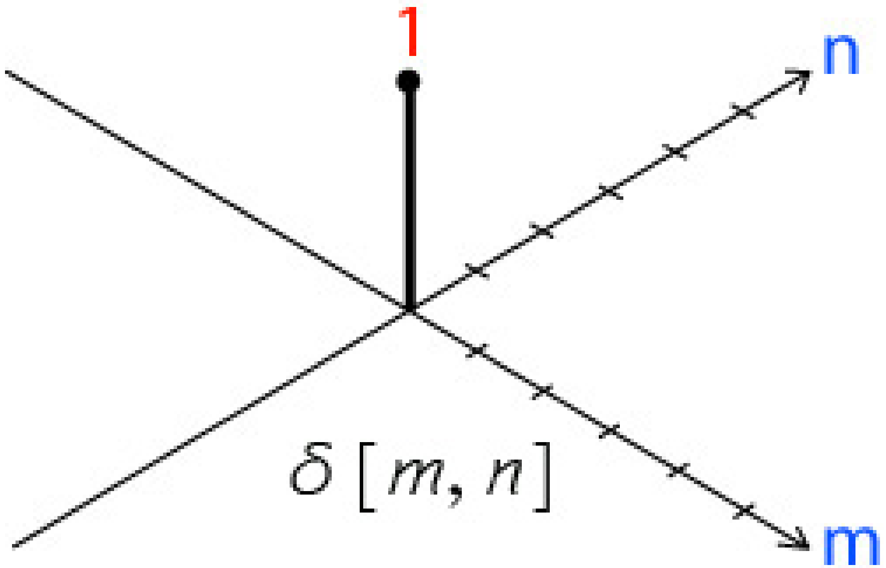
План лекции

- Представление изображения в частотной области. Преобразование Фурье
- Системы и фильтры
- **Свертки**

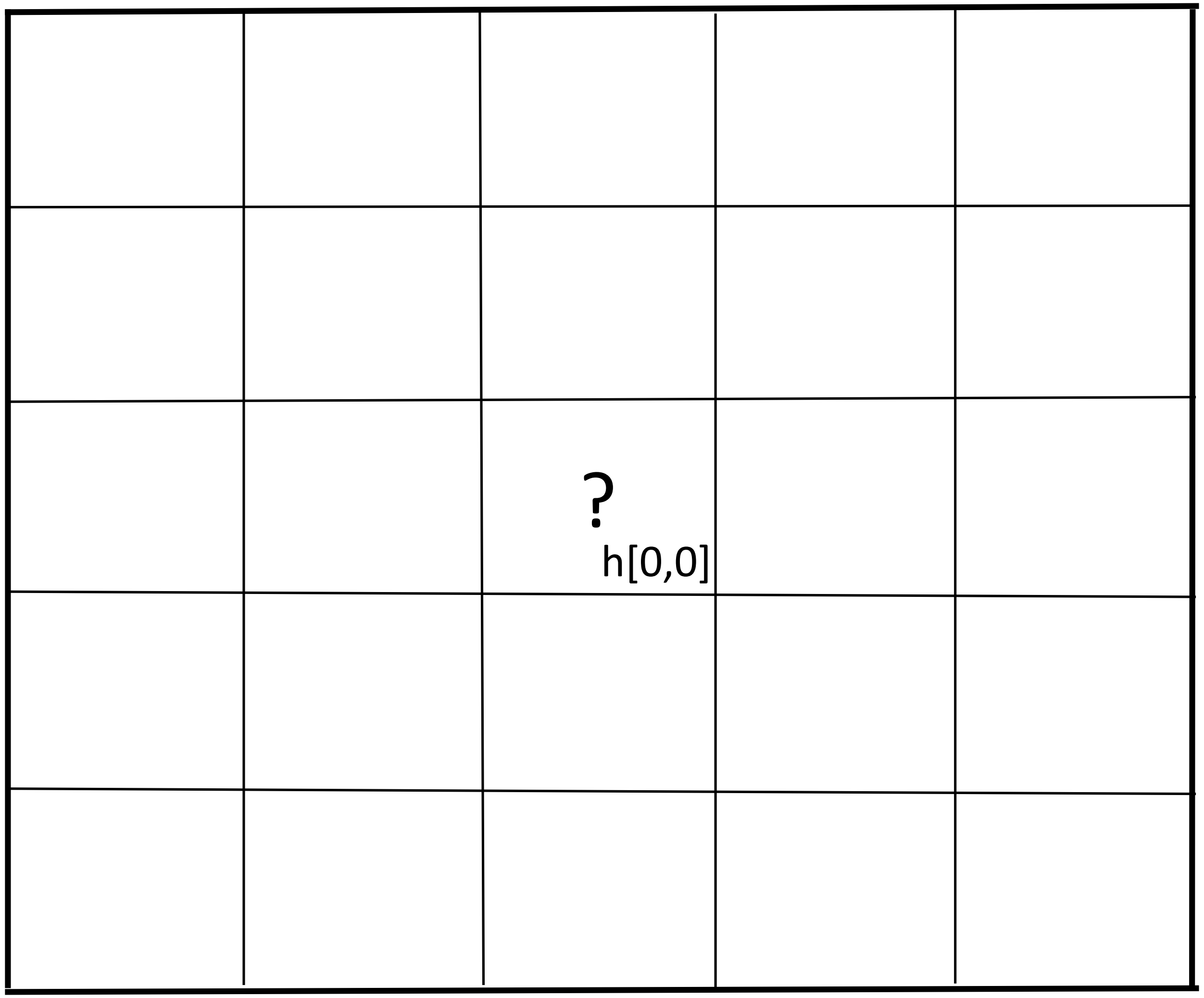
Импульсная функция

Рассмотрим специальную функцию:

- равна 1, в точке [0,0].
- равна 0, во всех остальных точках



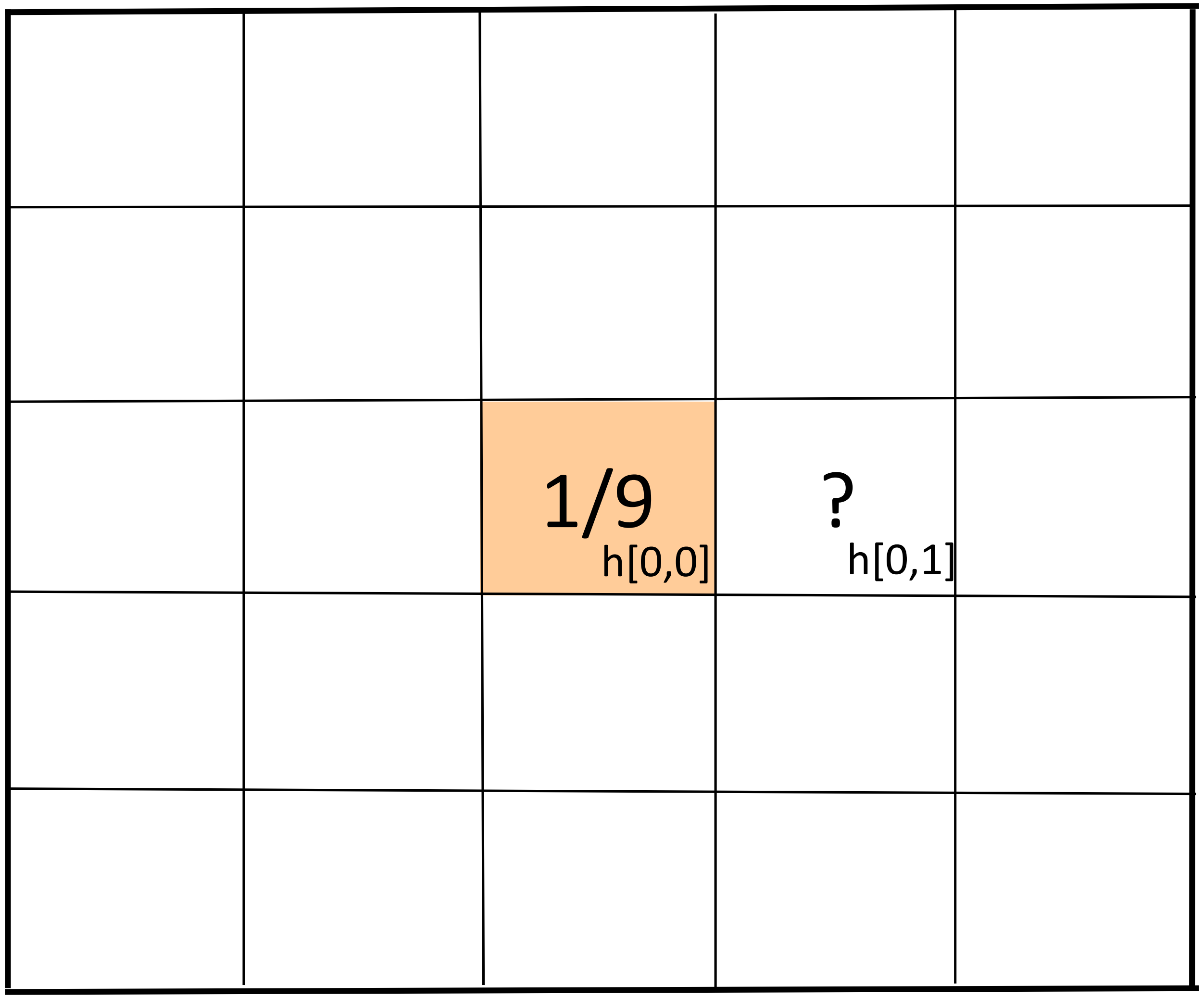
Импульсный отклик от фильтра размытия



$$\delta_2 \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

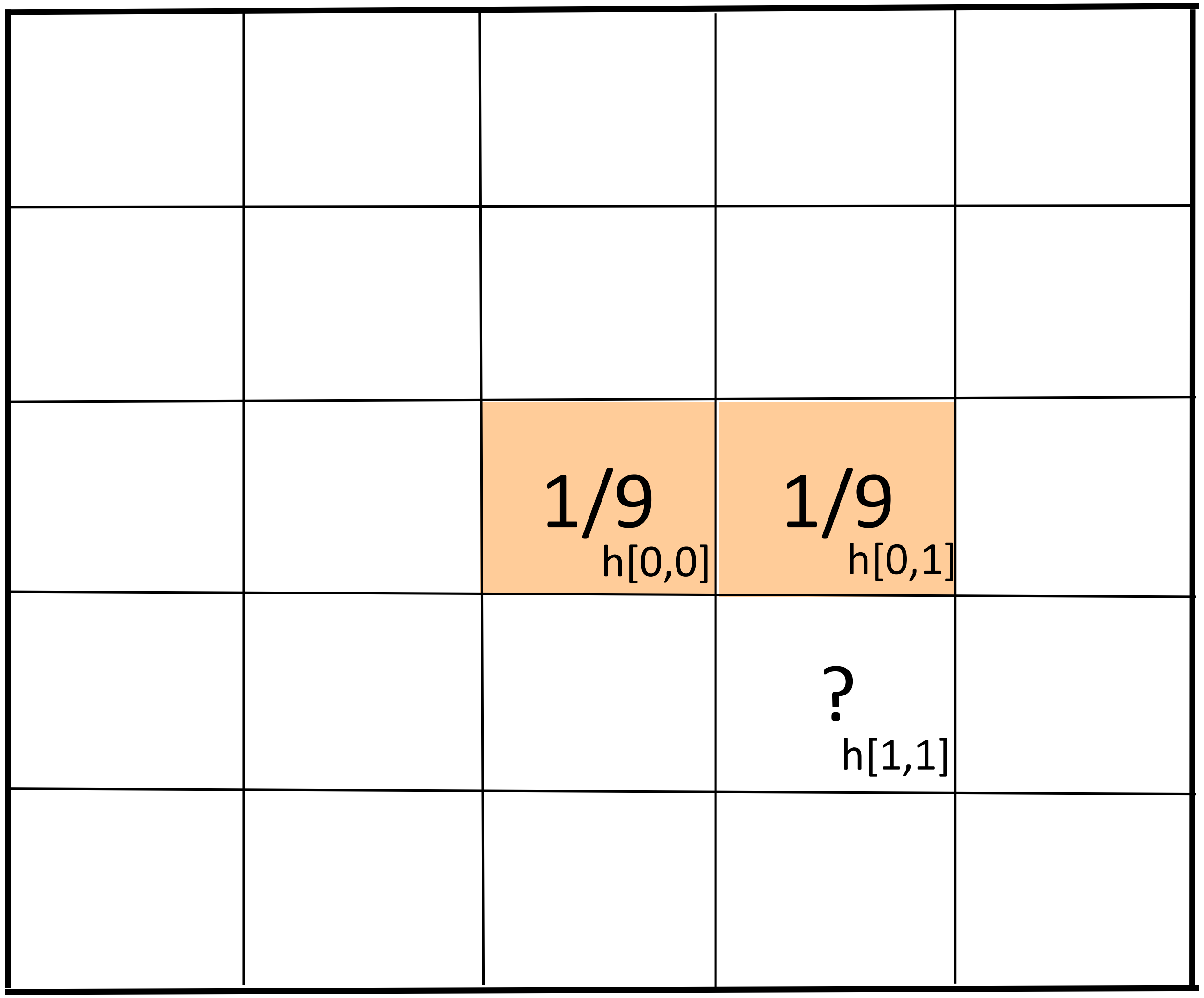
Импульсный отклик от фильтра размытия



$$\delta_2 \xrightarrow{S} h[n, m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

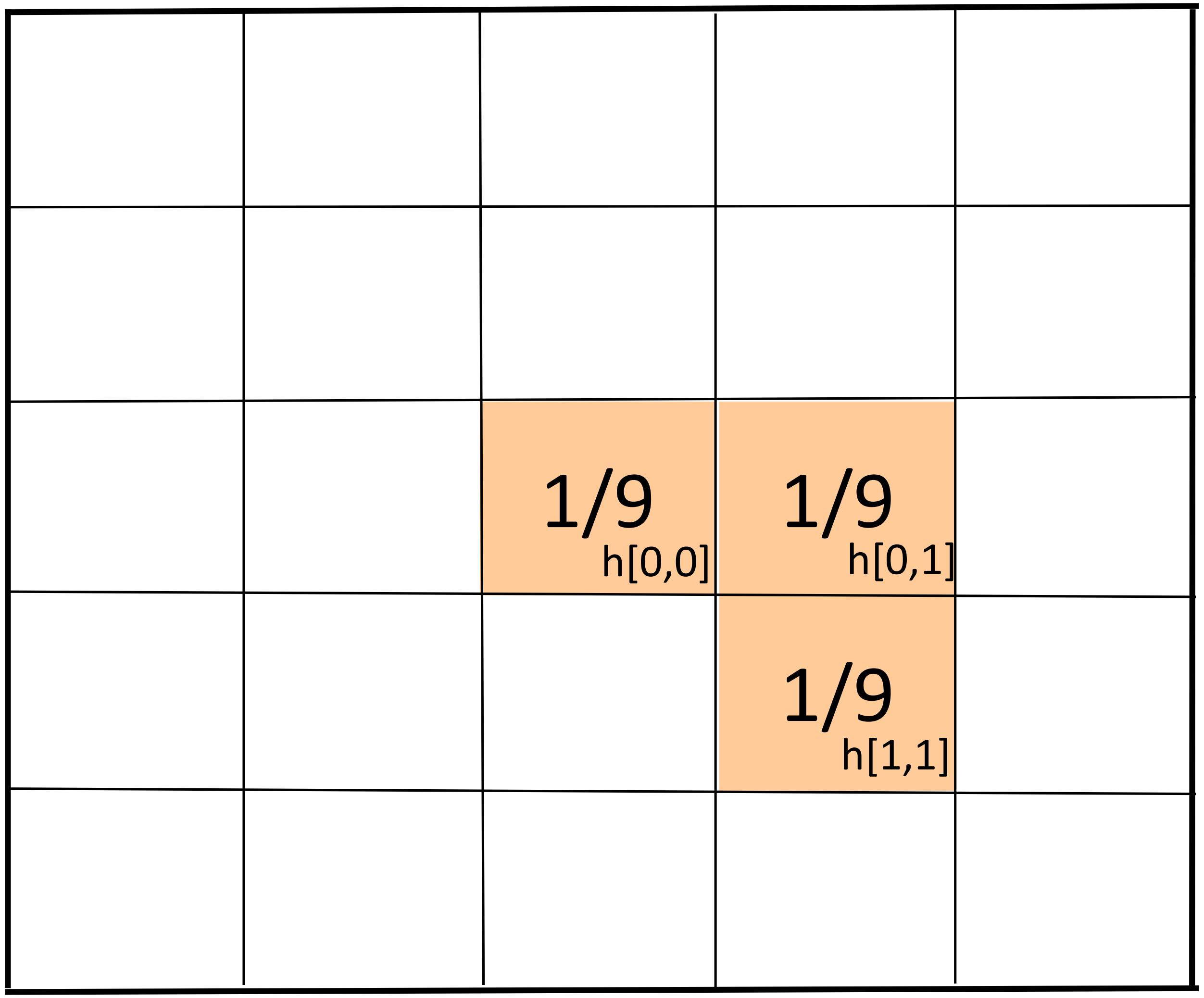
Импульсный отклик от фильтра размытия



$$\delta_2 \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1}\delta_2[n-k,m-l]$$

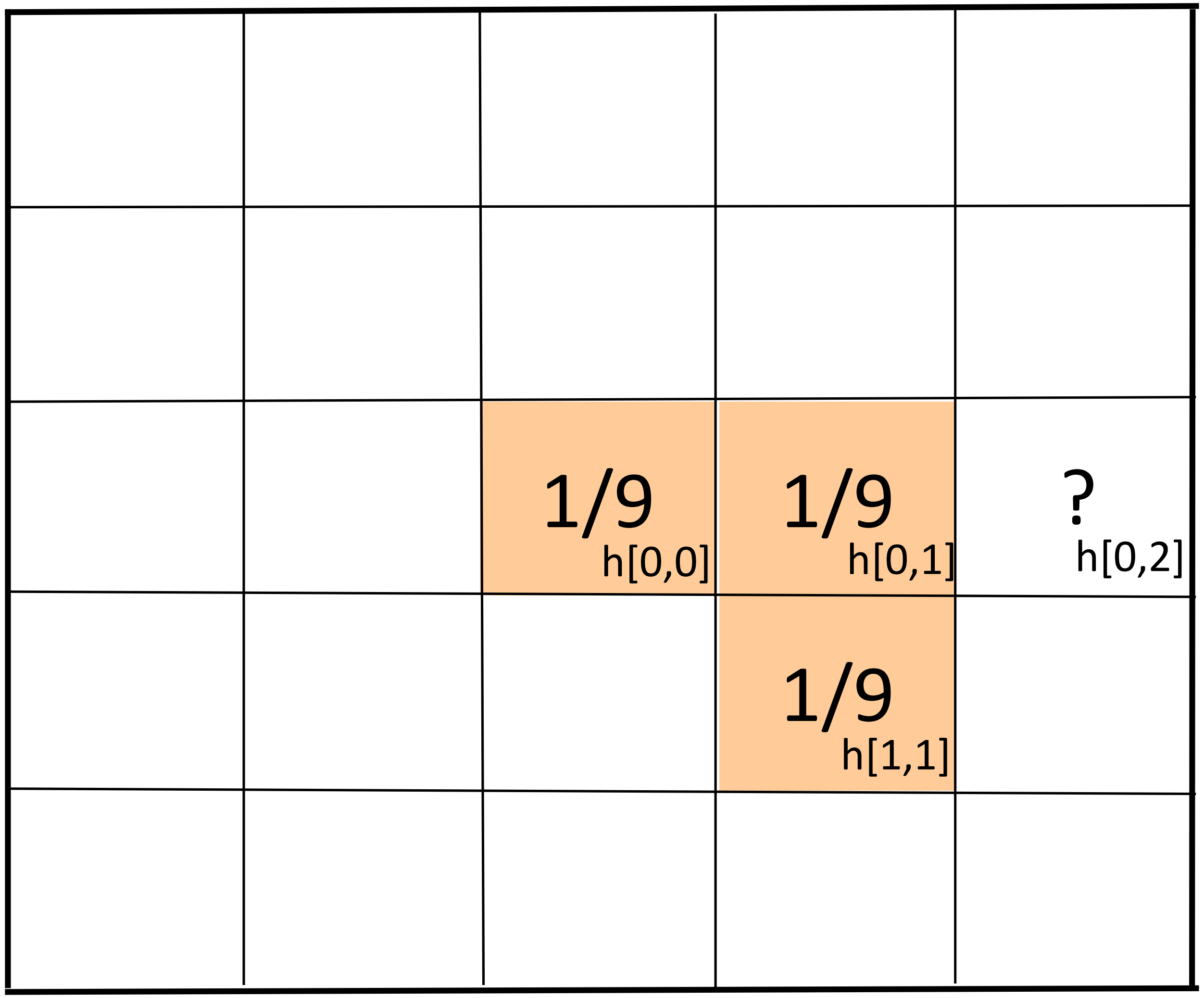
Импульсный отклик от фильтра размытия



$$\delta_2 \xrightarrow{S} h[n, m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

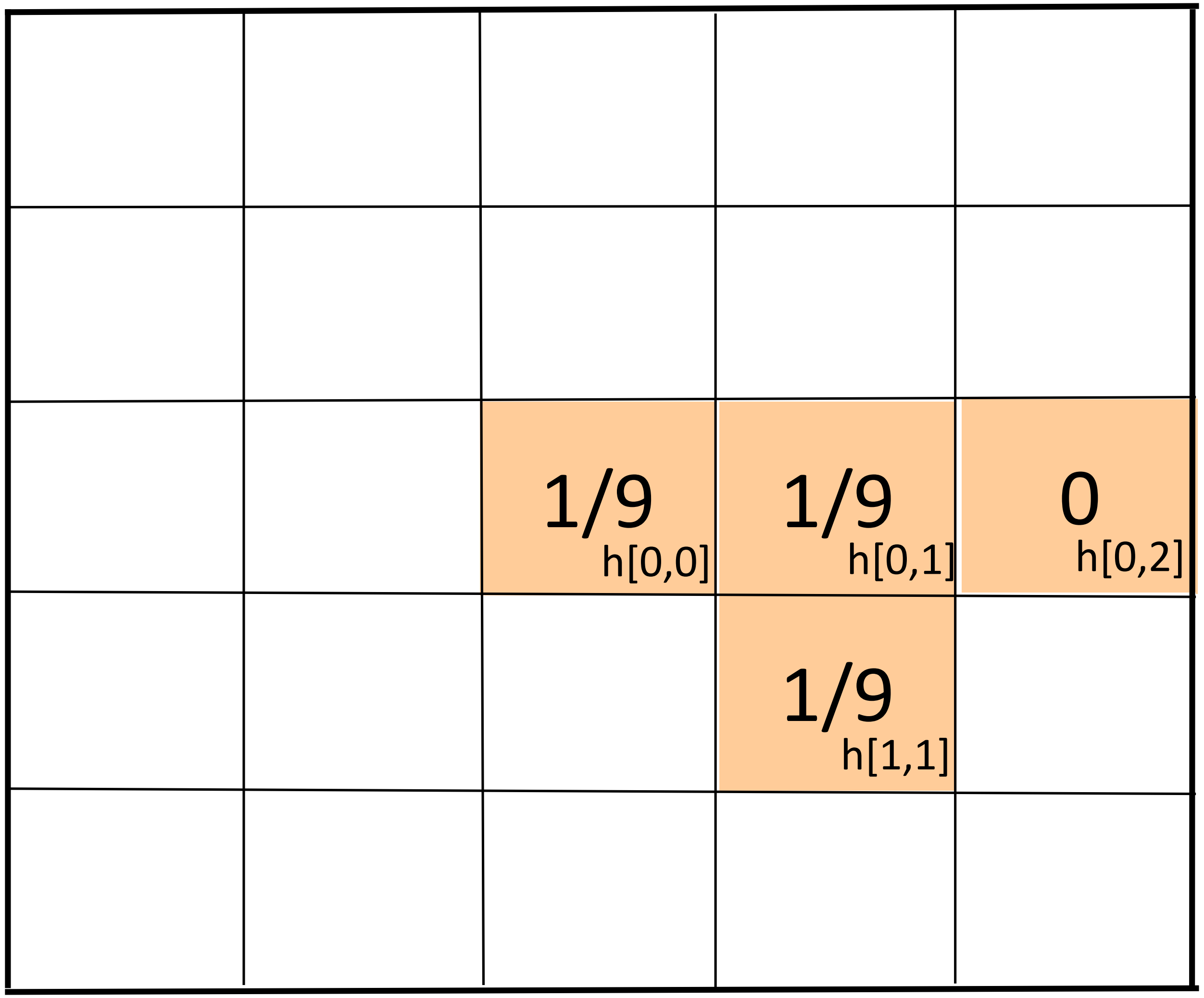
Импульсный отклик от фильтра размытия



$$\delta_2 \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1}\delta_2[n-k,m-l]$$

Импульсный отклик от фильтра размытия



$$\delta_2 \xrightarrow{S} h[n,m]$$

$$= \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1}\delta_2[n-k,m-l]$$

Импульсный отклик от фильтра размытия

0	0	0	0	0
0	1/9 h[-1,-1]	1/9	1/9	0
0	1/9	1/9 h[0,0]	1/9 h[0,1]	0 h[0,2]
0	1/9	1/9	1/9 h[1,1]	0
0	0	0	0	0

$$\delta_2 \overset{S}{\rightarrow} g[n,m]$$

$$= \frac{1}{9} \sum_{k=-1}^{1} \sum_{l=-1}^{1} \delta_2[n-k, m-l]$$

Фильтр размытия через импульсные функции

$$h[n,m] = \frac{1}{9}\sum_{k=-1}^{1}\sum_{l=-1}^{1}\delta_2[n-k, m-l]$$

$$= \begin{bmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & \underline{1/9} & 1/9 \\ 1/9 & 1/9 & 1/9 \end{bmatrix}$$

h

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

Линейные инвариантные системы

- Передавая импульсный отклик в линейную систему, мы получаем его <u>импульсный отклик</u>.
- Итак, если мы не знаем, что делает линейная система, мы можем передать в нее импульс, чтобы получить фильтр $h[n,m]$, который скажет нам, что система на самом деле делает.

$$\delta_2[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow h[n,m]$$

- Но как мы используем $h[n,m]$ для вычисления g$[n,m]$ из f$[n,m]$?

$$f[n,m] \rightarrow \boxed{\text{System } \mathcal{S}} \rightarrow g[n,m]$$

Вспомните фильтр Размытие и то, как мы уже использовали его импульсный отклик

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0								

Вспомните фильтр Размытие и то, как мы уже использовали его импульсный отклик

$f[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n, m]$

	0	10							

Вспомните фильтр Размытие и то, как мы уже использовали его импульсный отклик

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10	20						

Вспомните фильтр Размытие и то, как мы уже использовали его импульсный отклик

$f[n,m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$g[n,m]$

	0	10	20	30					

Вспомните фильтр Размытие и то, как мы уже использовали его импульсный отклик

$\mathrm{f}[n, m]$

0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	90	0	90	90	90	0	0
0	0	0	90	90	90	90	90	0	0
0	0	0	0	0	0	0	0	0	0
0	0	90	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

$\mathrm{g}[n, m]$

	0	10	20	30	30	30	20	10	
	0	20	40	60	60	60	40	20	
	0	30	60	90	90	90	60	30	
	0	30	50	80	80	90	60	30	
	0	30	50	80	80	90	60	30	
	0	20	30	50	50	60	40	20	
	10	20	30	30	30	30	20	10	
	10	10	10	0	0	0	0	0	

Общая линейная система с инвариантностью сдвига

Допустим, наш входной f – это изображение 3x3:

f[0,0]	f[0,1]	f[1,1]
f[1,0]	f[1,1]	f[1,2]
f[2,0]	f[2,1]	f[2,2]

Мы можем переписать $f[n,m]$ как сумму дельта функций:

$$
\begin{aligned}
f[n,m] = {} & f[0,0] \times \delta_2[n-0, m-0] \\
& + f[0,1] \times \delta_2[n-0, m-1] \\
& + \ldots \\
& + f[n,m] \times \delta_2[0,0] \\
& + \ldots
\end{aligned}
$$

Или записать так: $$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times \delta_2[n-k, m-l]$$

Свойства системы

Теперь мы знаем, что происходит, когда мы посылаем функцию дельты через систему:

$$\delta_2[n,m] \;\longrightarrow\; \boxed{\text{System } \mathcal{S}} \;\longrightarrow\; h[n,m]$$

Мы также знаем, что система сдвигает выход при смещении входа :

$$\delta_2[n-k,m-l] \;\longrightarrow\; \boxed{\text{System } \mathcal{S}} \;\longrightarrow\; h[n-k,m-l]$$

Наконец, принцип суперпозиции:

$$S[\,\alpha f_i[n,m] + \beta f_j[k,l]\,] \quad = \quad \alpha\, S[\, f_i[n,m]] + \beta\, S[\, f_j[k,l]\,]$$

Свойства системы

Мы можем обобщить этот принцип суперпозиции …

$$S[\ \alpha f_i[n,m] + \ \beta f_j[k,l]\ ] \quad = \quad \alpha\, S[\ f_i[n,m]]\ +\ \beta\, S[\ f_j[k,l]\ ]$$

… теперь через дельта функции…

$$f[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times\ \delta_2[n-k, m-l]$$

… получаем:

$$S\left[\sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times\ \delta_2[n-k, m-l]\right]$$
$$= \sum_{k=-\infty}^{\infty} \sum_{l=\infty}^{\infty} f[k,l] \times\ S[\delta_2[n-k, m-l]]$$

Свойства системы

$$S\left[\sum_{k=-\infty}^{\infty}\sum_{l=\infty}^{\infty} f[k,l]\times\ \delta_2[n-k,m-l]\right]$$

$$=\sum_{k=-\infty}^{\infty}\sum_{l=\infty}^{\infty} f[k,l]\times\ S[\delta_2[n-k,m-l]]$$

$$=\sum_{k=-\infty}^{\infty}\sum_{l=\infty}^{\infty} f[k,l]\times h[n-k,m-l]$$

Linear Shift Invariant systems

Система полностью определяется ее импульсным откликом

$$f[n,m] \rightarrow \boxed{\mathcal{S}\ \mathrm{LSI}} \rightarrow \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

$$o_2[n,m] \rightarrow \boxed{\mathcal{S}} \rightarrow h[n,m]$$

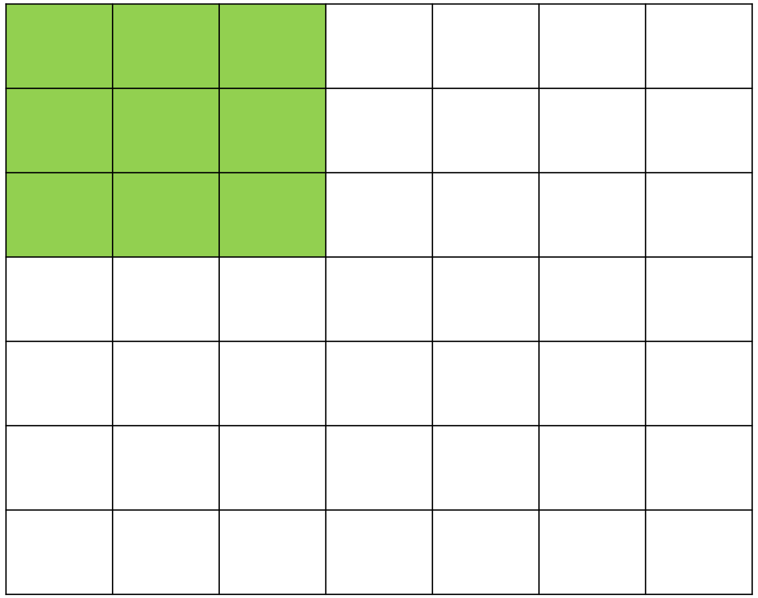
Дискретная свертка

$$S[f] = f[n,m] * h[n,m]$$

Двумерная свертка

2D свёртка очень похожа на 1D.
Основное отличие состоит в том, что теперь нам приходится проводить итерации по 2 осям вместо 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

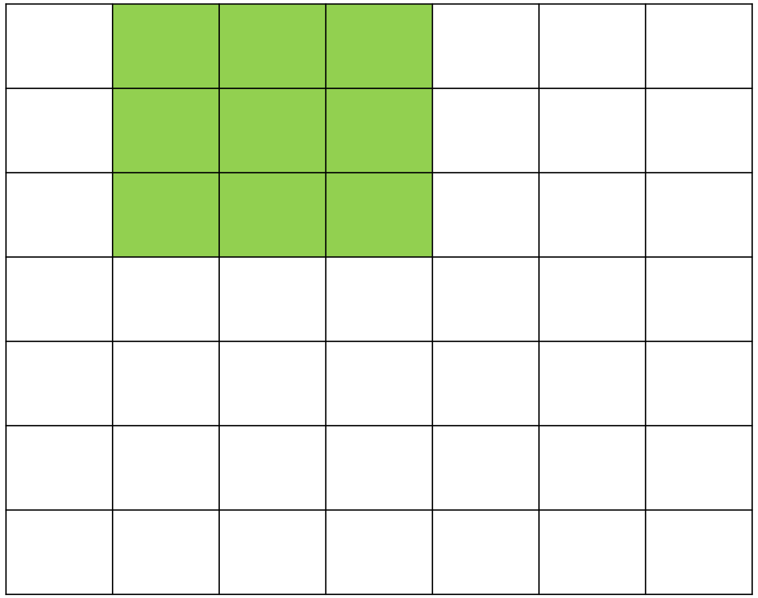


Предположим, что у нас есть фильтр(h[,]) размером 3x3 и изображение (f[,]) размером 7x7.

Двумерная свертка

2D свёртка очень похожа на 1D.
Основное отличие состоит в том, что теперь нам приходится проводить итерации по 2 осям вместо 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

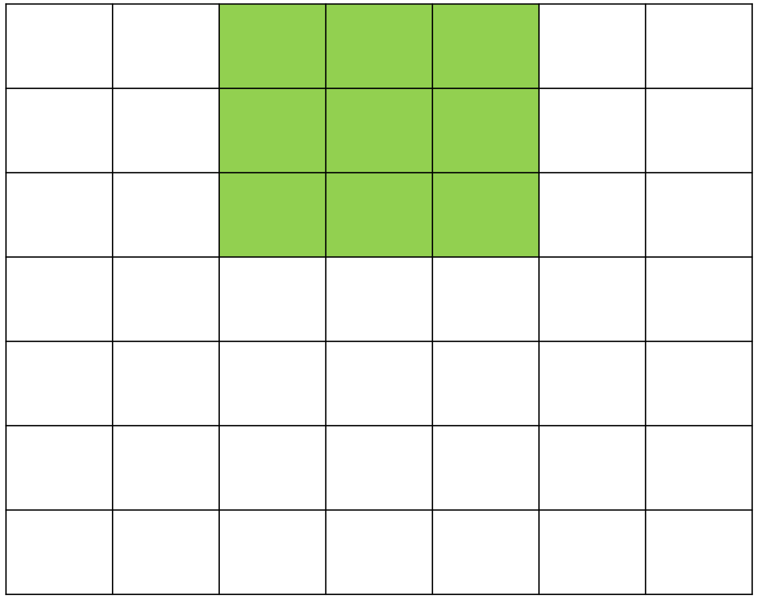


Предположим, что у нас есть фильтр(h[,]) размером 3x3 и изображение (f[,]) размером 7x7.

Двумерная свертка

2D свёртка очень похожа на 1D.
Основное отличие состоит в том, что теперь нам приходится проводить итерации по 2 осям вместо 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

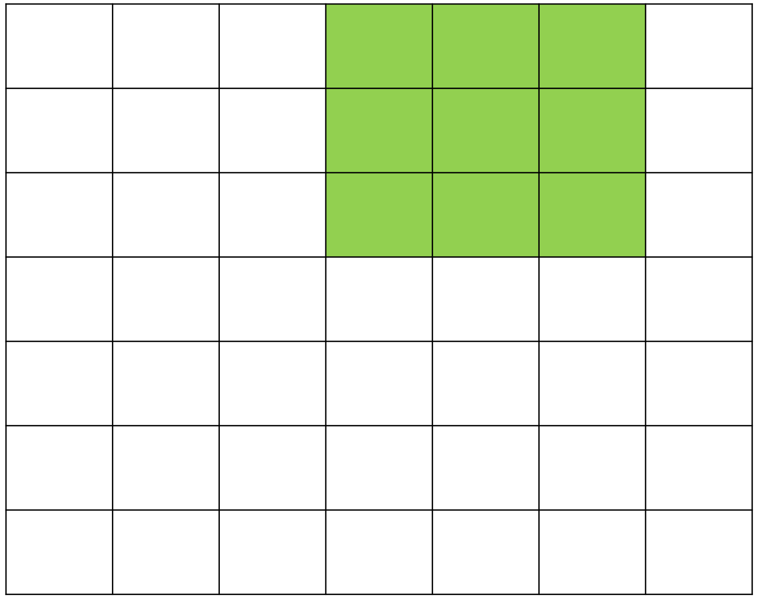


Предположим, что у нас есть фильтр(h[,]) размером 3x3 и изображение (f[,]) размером 7x7.

Двумерная свертка

2D свёртка очень похожа на 1D.
Основное отличие состоит в том, что теперь нам приходится проводить итерации по 2 осям вместо 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

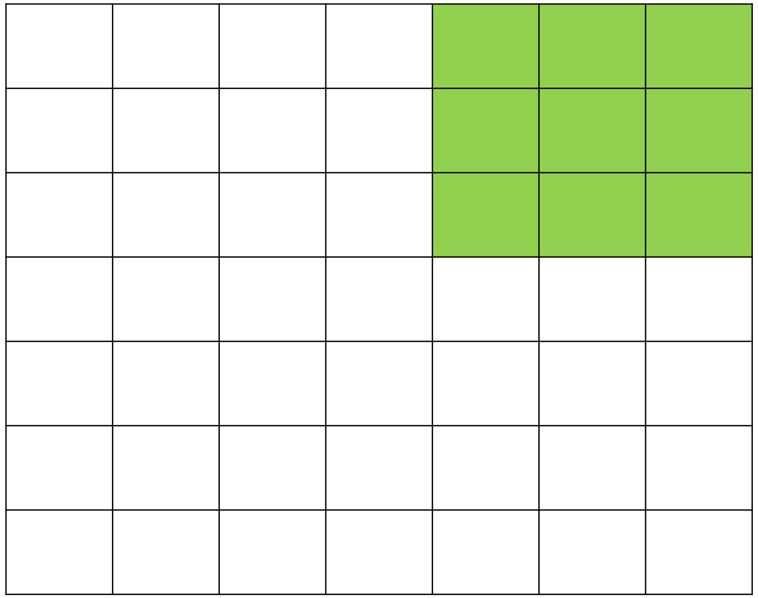


Предположим, что у нас есть фильтр(h[,]) размером 3x3 и изображение (f[,]) размером 7x7.

Двумерная свертка

2D свёртка очень похожа на 1D.
Основное отличие состоит в том, что теперь нам приходится проводить итерации по 2 осям вместо 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$

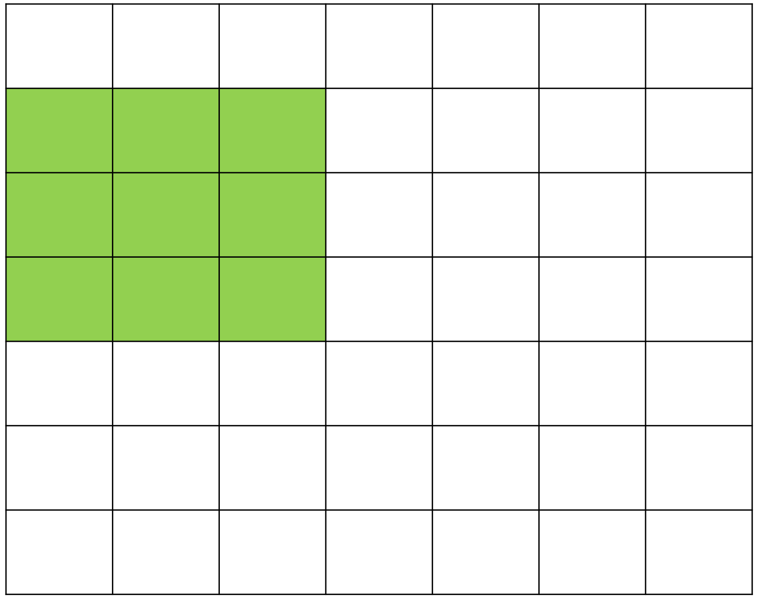


Предположим, что у нас есть фильтр(h[,]) размером 3x3 и изображение (f[,]) размером 7x7.

Двумерная свертка

2D свёртка очень похожа на 1D.
Основное отличие состоит в том, что теперь нам приходится проводить итерации по 2 осям вместо 1.

$$f[n,m] * h[n,m] = \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} f[k,l]\, h[n-k, m-l]$$



Предположим, что у нас есть фильтр(h[,]) размером 3x3 и изображение (f[,]) размером 7x7.

Convolution

$$f * h = \sum_{k}\sum_{l} f(k,l)\,h(-k,-l)$$

$f =$ Image

$h =$ Kernel

$$h = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \xrightarrow{X - flip} \begin{bmatrix} h_7 & h_8 & h_9 \\ h_4 & h_5 & h_6 \\ h_1 & h_2 & h_3 \end{bmatrix} \xrightarrow{Y - flip} \begin{bmatrix} h_9 & h_8 & h_7 \\ h_6 & h_5 & h_4 \\ h_3 & h_2 & h_1 \end{bmatrix}$$

$$f = \begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix}$$

$$\begin{bmatrix} f_1 & f_2 & f_3 \\ f_4 & f_5 & f_6 \\ f_7 & f_8 & f_9 \end{bmatrix} * \begin{bmatrix} h_9 & h_8 & h_7 \\ h_6 & h_5 & h_4 \\ h_3 & h_2 & h_1 \end{bmatrix} \longrightarrow$$

$$\begin{aligned} f * h = {} & f_1 h_9 + f_2 h_8 + f_3 h_7 \\ & + f_4 h_6 + f_5 h_5 + f_6 h_4 \\ & + f_7 h_3 + f_8 h_2 + f_9 h_1 \end{aligned}$$

Пример двумерной свертки

1	2	3
4	5	6
7	8	9

Input

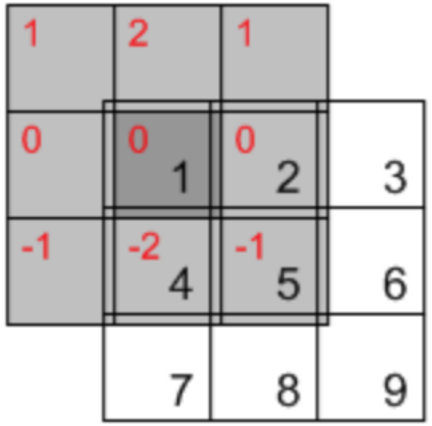
n \ m	-1	0	1
-1	-1	-2	-1
0	0	0	0
1	1	2	1

Kernel

-13	-20	-17
-18	-24	-18
13	20	17

Output

Пример двумерной свертки

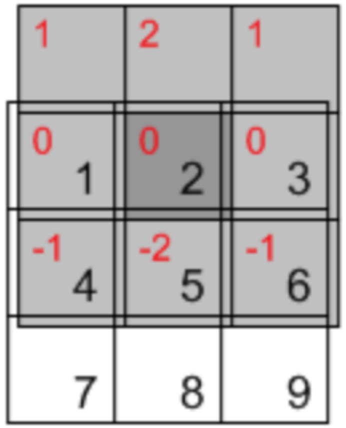


$$
\begin{aligned}
y[0,0] = {} & x[-1,-1]\cdot h[1,1] + x[0,-1]\cdot h[0,1] + x[1,-1]\cdot h[-1,1] \\
& + x[-1,0]\cdot h[1,0] + x[0,0]\cdot h[0,0] + x[1,0]\cdot h[-1,0] \\
& + x[-1,1]\cdot h[1,-1] + x[0,1]\cdot h[0,-1] + x[1,1]\cdot h[-1,-1] \\
= {} & 0\cdot 1 + 0\cdot 2 + 0\cdot 1 + 0\cdot 0 + 1\cdot 0 + 2\cdot 0 + 0\cdot(-1) + 4\cdot(-2) + 5\cdot(-1) = -13
\end{aligned}
$$

$\boxed{-13}$	-20	-17
-18	-24	-18
13	20	17

Output

Пример двумерной свертки

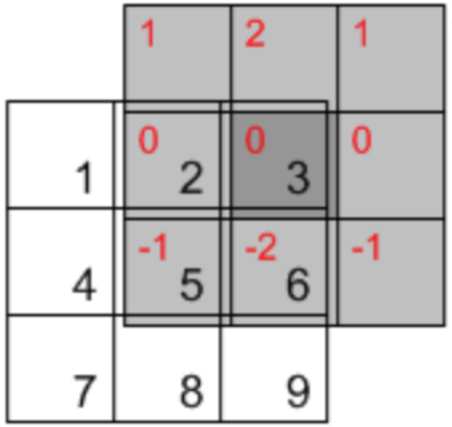


$$
\begin{aligned}
y[1,0] &= x[0,-1]\cdot h[1,1] + x[1,-1]\cdot h[0,1] + x[2,-1]\cdot h[-1,1] \\
&\quad + x[0,0]\cdot h[1,0] + x[1,0]\cdot h[0,0] + x[2,0]\cdot h[-1,0] \\
&\quad + x[0,1]\cdot h[1,-1] + x[1,1]\cdot h[0,-1] + x[2,1]\cdot h[-1,-1] \\
&= 0\cdot 1 + 0\cdot 2 + 0\cdot 1 + 1\cdot 0 + 2\cdot 0 + 3\cdot 0 + 4\cdot(-1) + 5\cdot(-2) + 6\cdot(-1) = -20
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Пример двумерной свертки

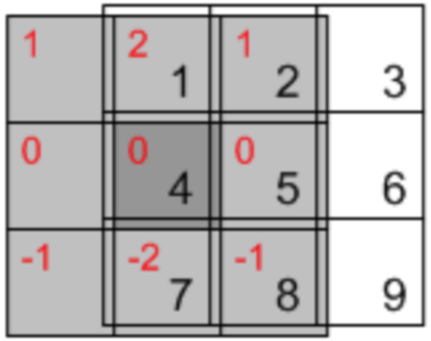


$$
\begin{aligned}
y[2,0] &= x[1,-1]\cdot h[1,1] + x[2,-1]\cdot h[0,1] + x[3,-1]\cdot h[-1,1] \\
&\quad + x[1,0]\cdot h[1,0] + x[2,0]\cdot h[0,0] + x[3,0]\cdot h[-1,0] \\
&\quad + x[1,1]\cdot h[1,-1] + x[2,1]\cdot h[0,-1] + x[3,1]\cdot h[-1,-1] \\
&= 0\cdot 1 + 0\cdot 2 + 0\cdot 1 + 2\cdot 0 + 3\cdot 0 + 0\cdot 0 + 5\cdot(-1) + 6\cdot(-2) + 0\cdot(-1) = -17
\end{aligned}
$$

-13	-20	$\boxed{-17}$
-18	-24	-18
13	20	17

Output

Пример двумерной свертки



$$
\begin{aligned}
y[0,1] &= x[-1,0]\cdot h[1,1] + x[0,0]\cdot h[0,1] + x[1,0]\cdot h[-1,1] \\
&\quad + x[-1,1]\cdot h[1,0] + x[0,1]\cdot h[0,0] + x[1,1]\cdot h[-1,0] \\
&\quad + x[-1,2]\cdot h[1,-1] + x[0,2]\cdot h[0,-1] + x[1,2]\cdot h[-1,-1] \\
&= 0\cdot 1 + 1\cdot 2 + 2\cdot 1 + 0\cdot 0 + 4\cdot 0 + 5\cdot 0 + 0\cdot(-1) + 7\cdot(-2) + 8\cdot(-1) = -18
\end{aligned}
$$

-13	-20	-17
-18	-24	-18
13	20	17

Output

Пример двумерной свертки

$$* \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} = \quad ?$$

Пример двумерной свертки

*

•0	•0	•0
•0	•1	•0
•0	•0	•0

=

Пример двумерной свертки

*

•0	•0	•0
•0	•0	•1
•0	•0	•0

= ?

Пример двумерной свертки

*

•0	•0	•0
•0	•0	•1
•0	•0	•0

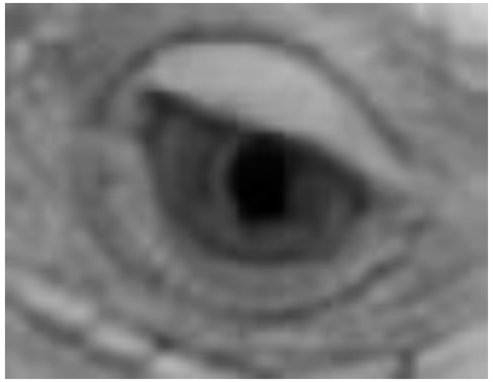
=

Пример двумерной свертки

$$* \ \frac{1}{9}\begin{bmatrix} \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \\ \bullet 1 & \bullet 1 & \bullet 1 \end{bmatrix} = \ ?$$

Пример двумерной свертки

$$* \frac{1}{9} \begin{array}{|c|c|c|} \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \end{array} =$$



Пример двумерной свертки

$$\begin{array}{|c|c|c|}\hline \bullet 0 & \bullet 0 & \bullet 0 \\ \hline \bullet 0 & \bullet 2 & \bullet 0 \\ \hline \bullet 0 & \bullet 0 & \bullet 0 \\ \hline \end{array} - \frac{1}{9}\begin{array}{|c|c|c|}\hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \end{array} = ?$$

$$\begin{array}{|c|c|c|}\hline \bullet 0 & \bullet 0 & \bullet 0 \\ \hline \bullet 0 & \bullet 1 & \bullet 0 \\ \hline \bullet 0 & \bullet 0 & \bullet 0 \\ \hline \end{array} + \underbrace{\begin{array}{|c|c|c|}\hline \bullet 0 & \bullet 0 & \bullet 0 \\ \hline \bullet 0 & \bullet 1 & \bullet 0 \\ \hline \bullet 0 & \bullet 0 & \bullet 0 \\ \hline \end{array} - \frac{1}{9}\begin{array}{|c|c|c|}\hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \bullet 1 & \bullet 1 & \bullet 1 \\ \hline \end{array}}$$

Что отнимает размытость?

Оригинальное изображение

-

Размытое

=

Детали

Оригинальное изображение

+

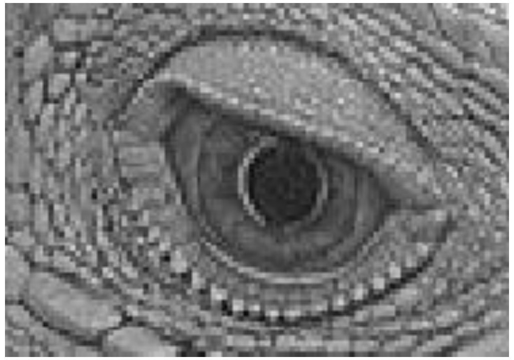
Детали

=

Повышенная резкость

Пример двумерной свертки – фильтр резкости

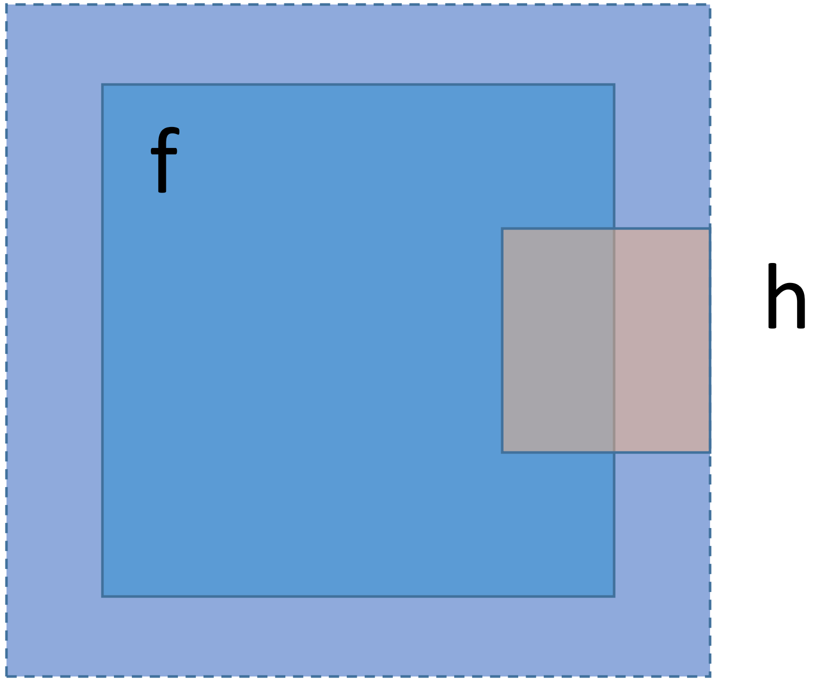
$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 0 \end{bmatrix} - \frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} =$$



Фильтр резкости: подчеркивает разность со средним местным значениями пикселей

Краевой эффект

- Компьютер будет вызывать только **конечные сигналы**.
- Что происходит на краю?



- нулевой паддинг
- повторение на краях
- отзеркаливание

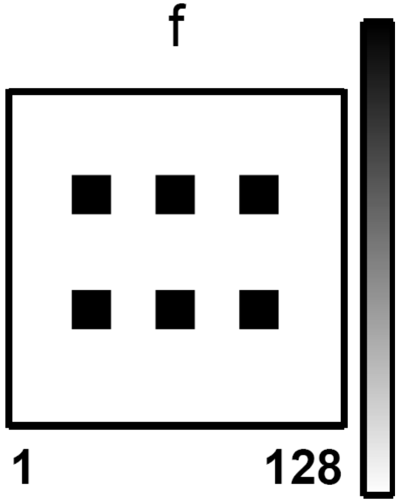
Кросс-корреляция

Кросс-корреляция двух 2D сигналов f[n,m] и h[n,m].

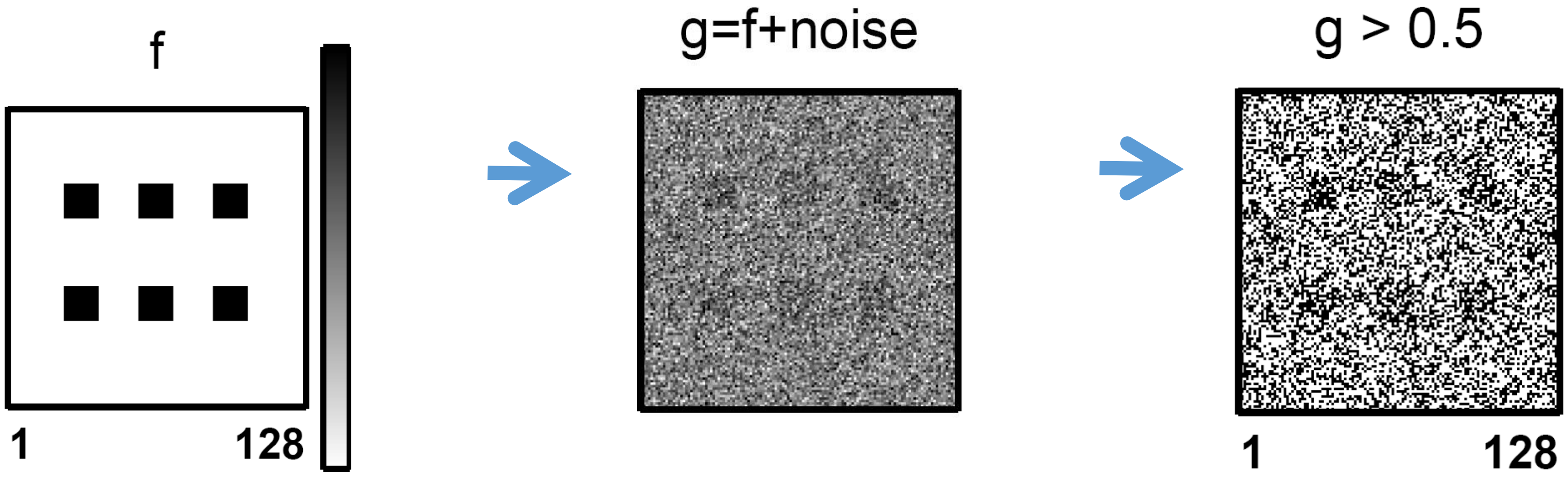
$$f[n,m] **h[n,m] = \sum_k \sum_l f[k,l]h[n-k,m-l]$$

- Эквивалент свертывания без переворачивания
- Измерения "сходства" между f и h.

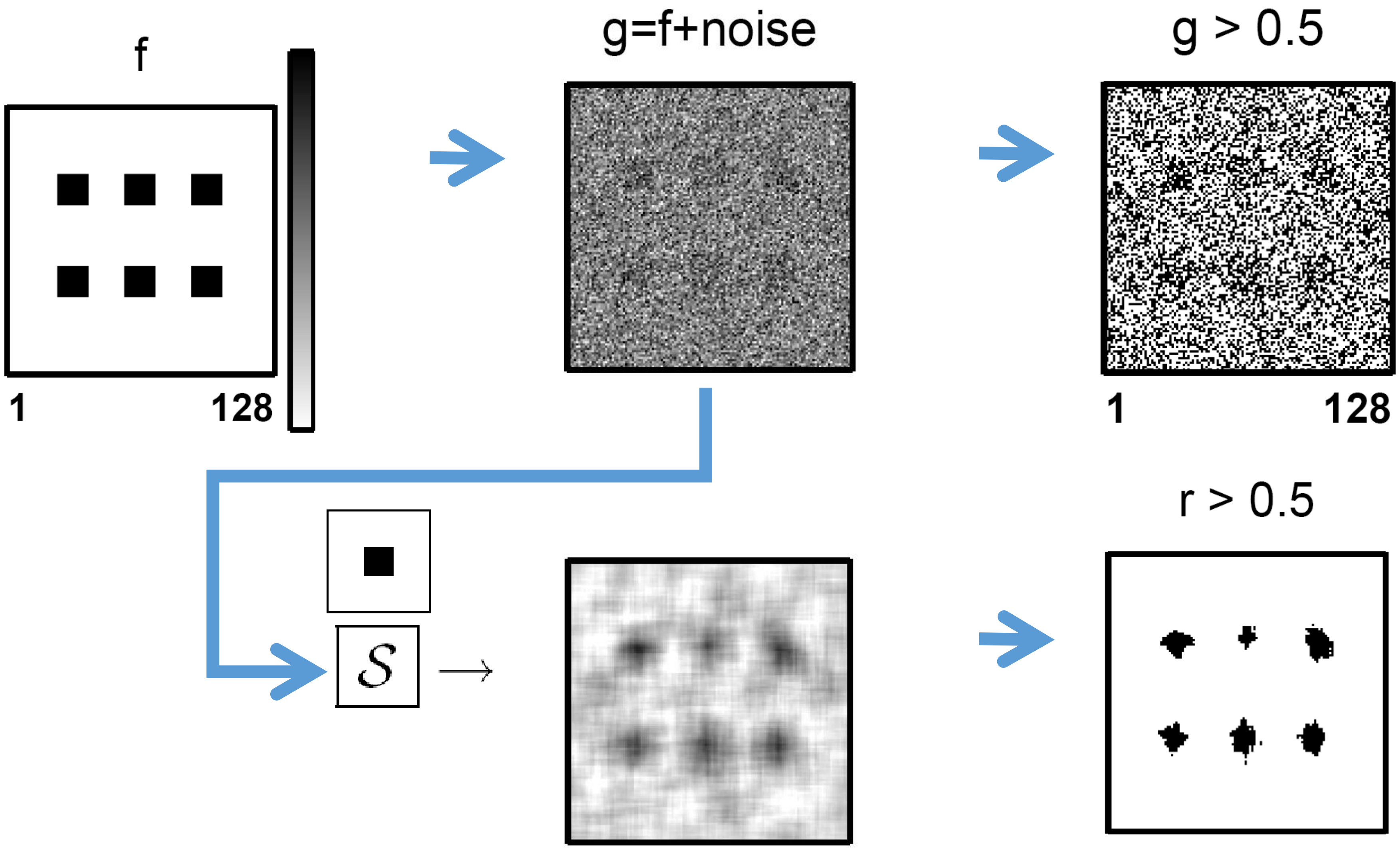
Пример кросс-корреляции



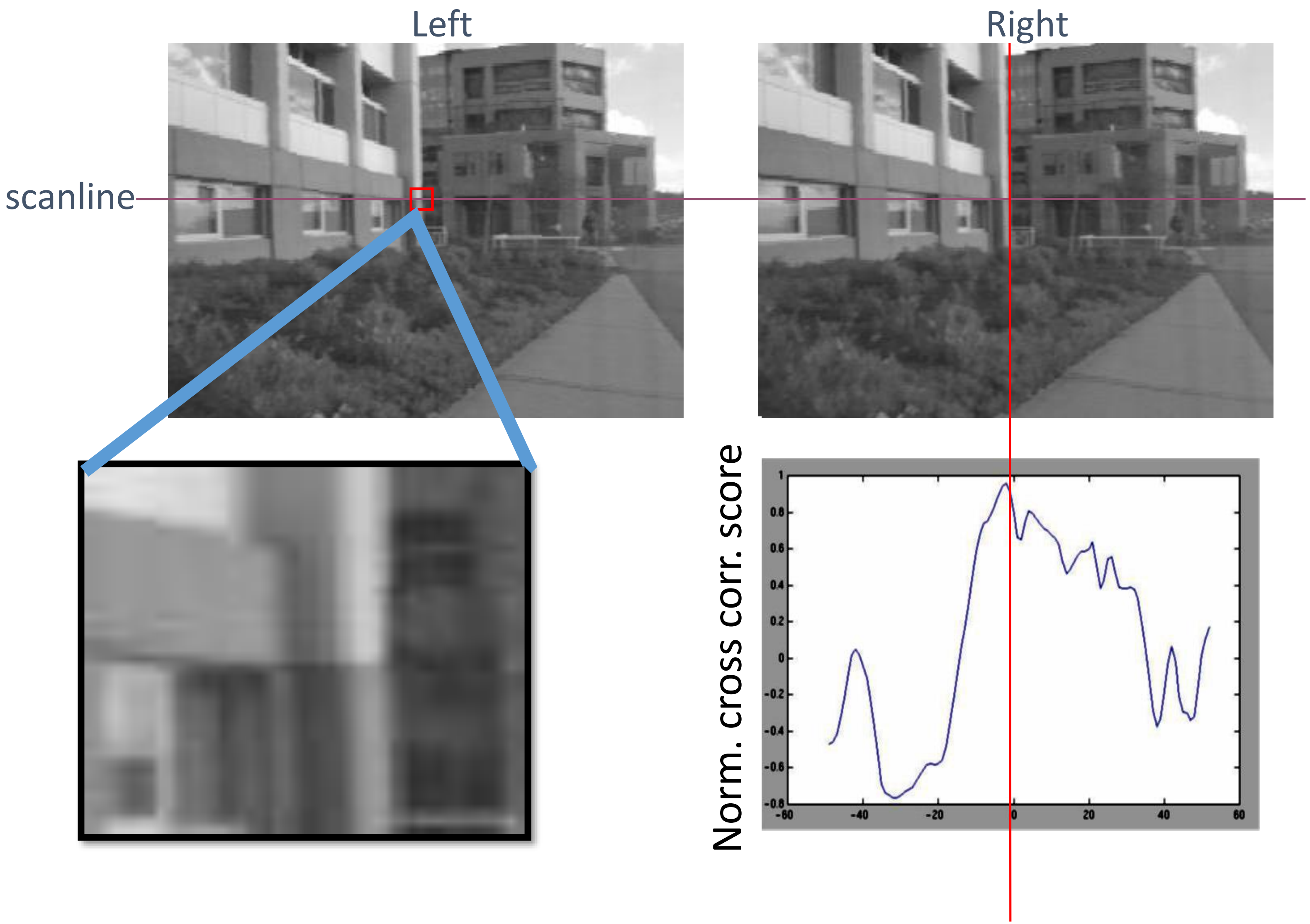
Пример кросс-корреляции



Пример кросс-корреляции



Пример кросс-корреляции



Свертка vs кросс-корреляция

- **Свертка** – это интеграл, выражающий величину перекрытия одной функции при ее смещении по другой.
  - свертка – это операция фильтрации

- **Корреляция** сравнивает **сходство** двух **наборов данных**. Корреляция рассчитывает меру сходства двух входных сигналов при их смещении друг от друга. Результат корреляции достигает максимума в тот момент, когда два сигнала совпадают наилучшим образом .
  - корреляция является мерой сходства двух сигналов.

Итоги

- Рассмотрено частотное представление изображения
- Показаны методы фильтрации в пространственной и частотной областях
- Изучено понятие свертки и кросс-корреляции