

# JPEG и JPEG 2000

---

Дмитрий Сергеевич Ватолин  
*Video Group*  
*CS MSU Graphics&Media Lab*

# Алгоритм JPEG

# Алгоритм JPEG

- Алгоритм разработан в 1991 году группой экспертов в области фотографии (JPEG — Joint Photographic Expert Group — подразделение в рамках ISO) специально для сжатия 24-битных изображений
- Алгоритм основан на дискретном косинусном преобразовании (в дальнейшем ДКП), применяемом к матрице изображения для получения некоторой новой матрицы коэффициентов

# Этапы работы алгоритма JPEG



Конвейер операций, используемый в JPEG

# Алгоритм JPEG: Смена цветового пространства и дискретизация

# Этапы работы алгоритма JPEG



Конвейер операций, используемый в JPEG

# RGB в YUV

- Переход из пространства RGB в YCbCr  
 Y – компонента яркости, Cb и Cr – цветовые компоненты



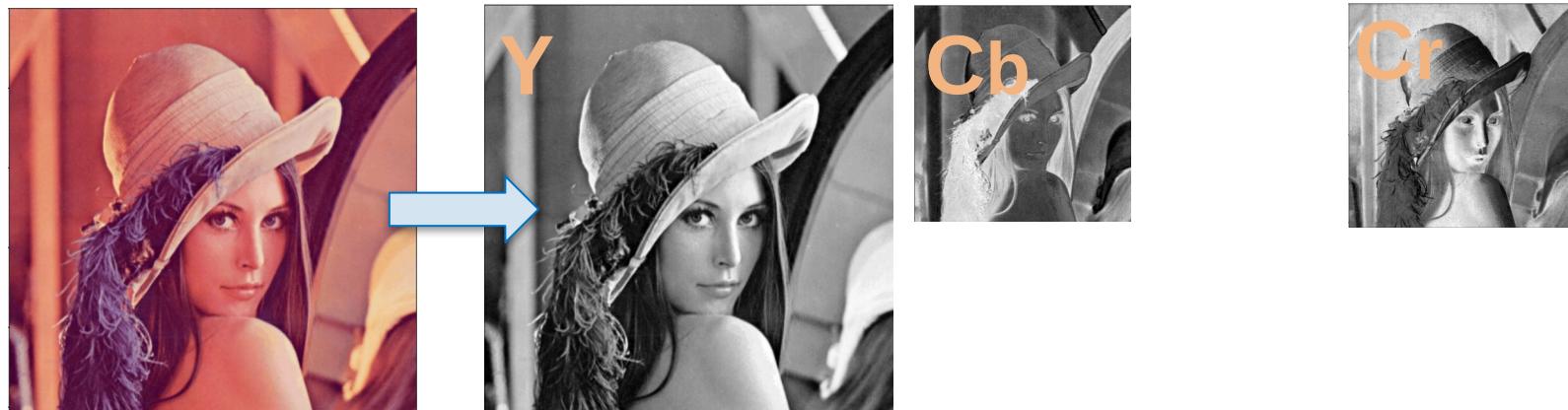
$$Y = 0.299*R + 0.587*G + 0.114*B$$

$$Cb = -0.172*R - 0.339*G + 0.511*B + 128$$

$$Cr = 0.511*R - 0.428*G - 0.083*B + 128$$

# Дискретизация YUV

- Переход из пространства RGB в YCbCr  
Y – компонента яркости, Cb и Cr – цветовые компоненты
- Уменьшение цветовых компонент в 2 раза по каждой стороне



$$Y = 0.299 * R + 0.587 * G + 0.114 * B$$

$$Cb = -0.172 * R - 0.339 * G + 0.511 * B + 128$$

$$Cr = 0.511 * R - 0.428 * G - 0.083 * B + 128$$

# Дискретизация YUV (1)



Red



Green



Blue



Luminance



Blue Chrominance

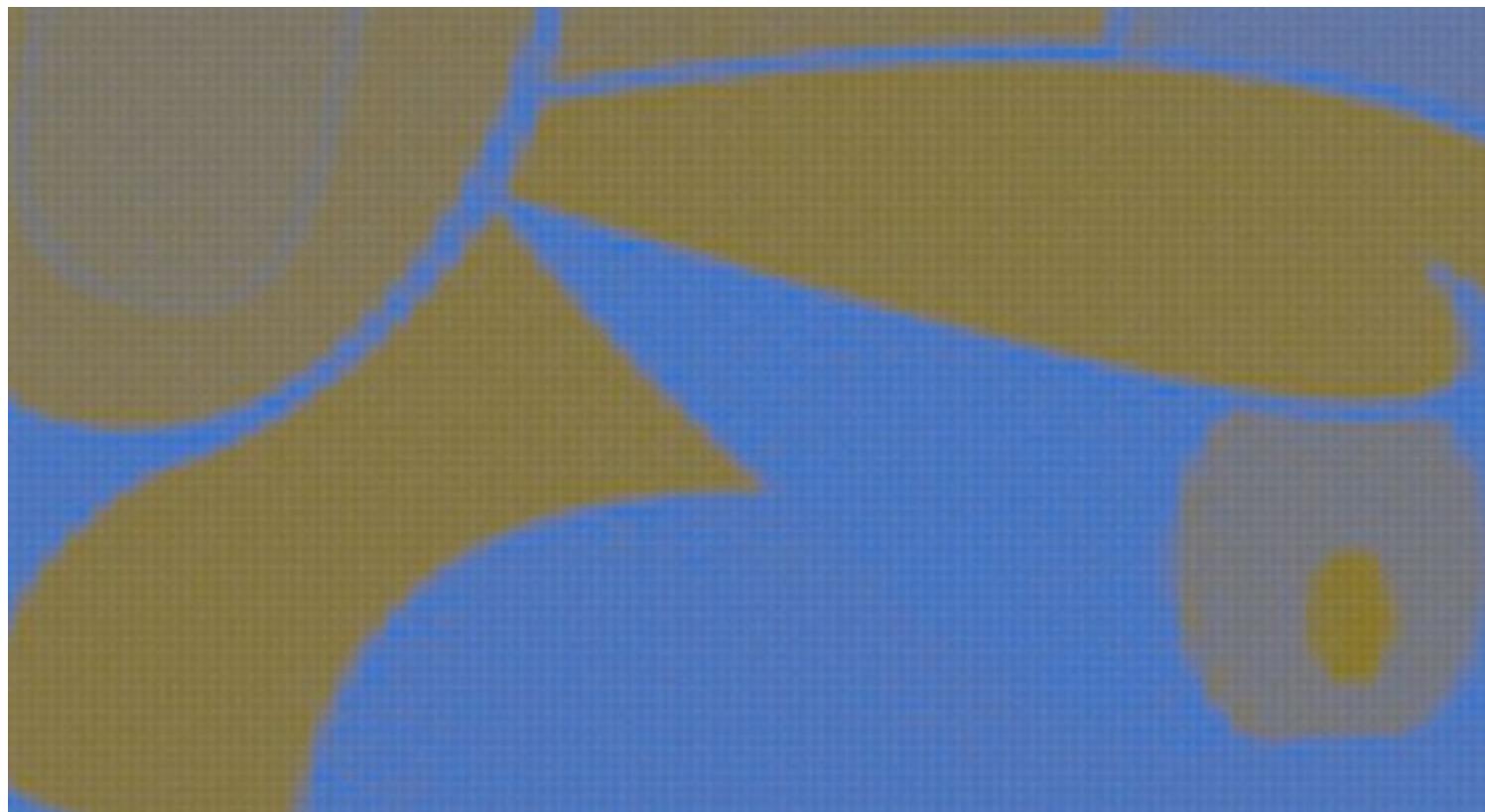


Red Chrominance

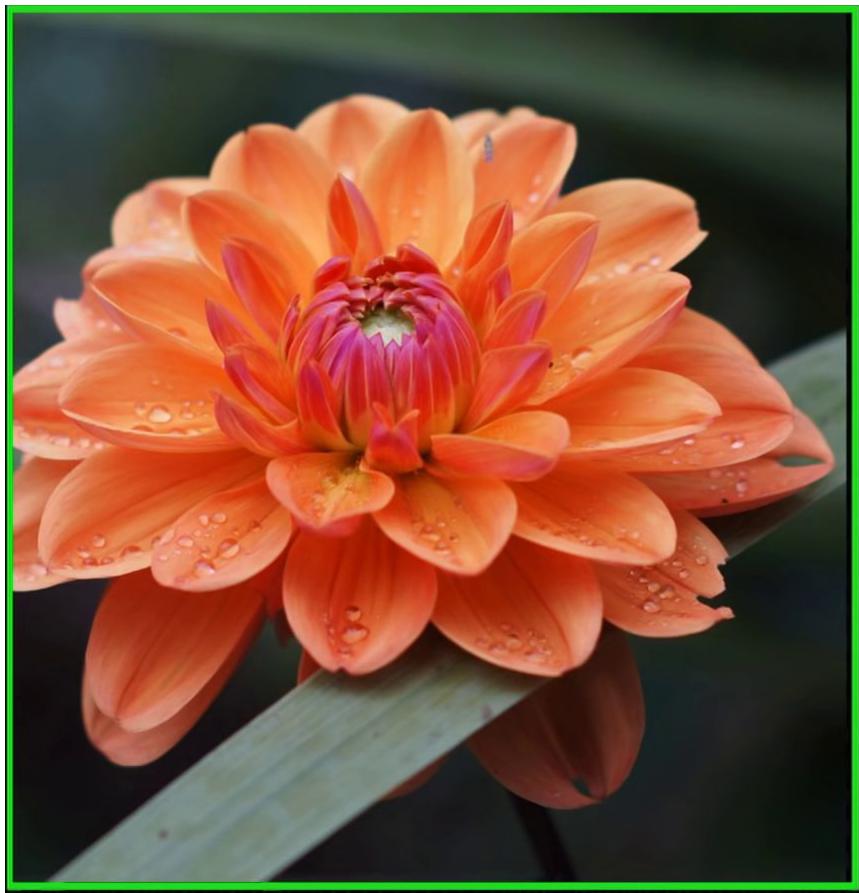
Subsample percent: 0%

# Дискретизация YUV (2)

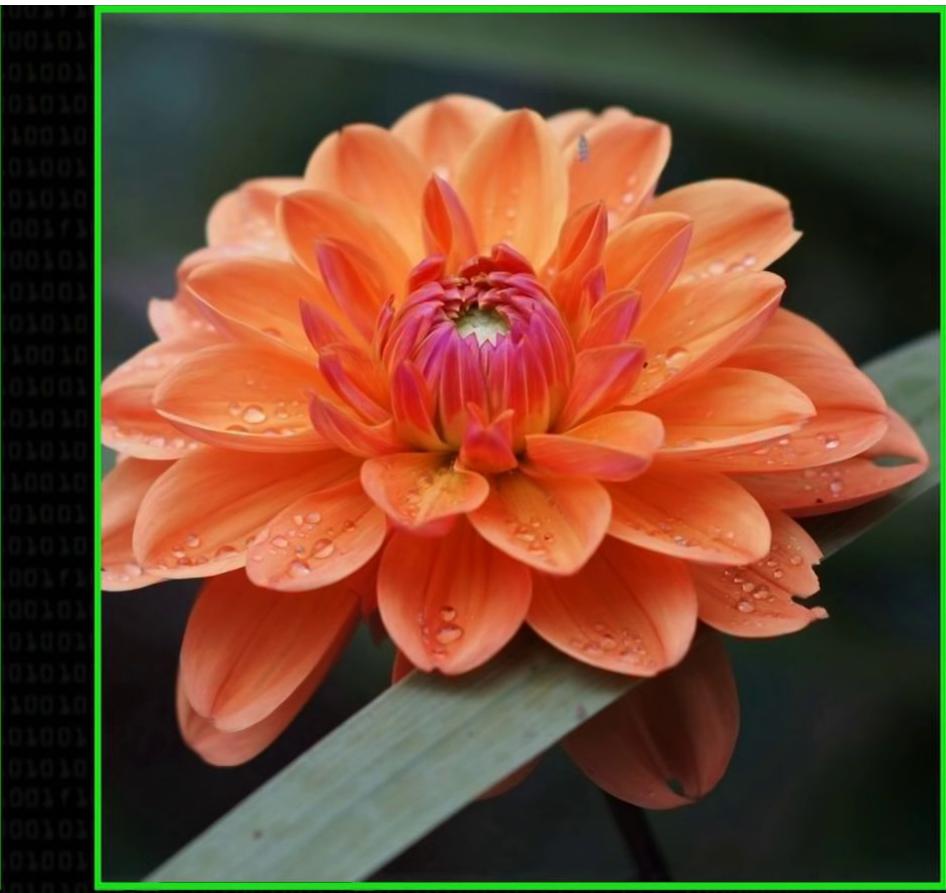
Значения компонент U и V прореживаются путем их усреднения для окна пикселей  $2 \times 2$



# Дискретизация YUV (3)

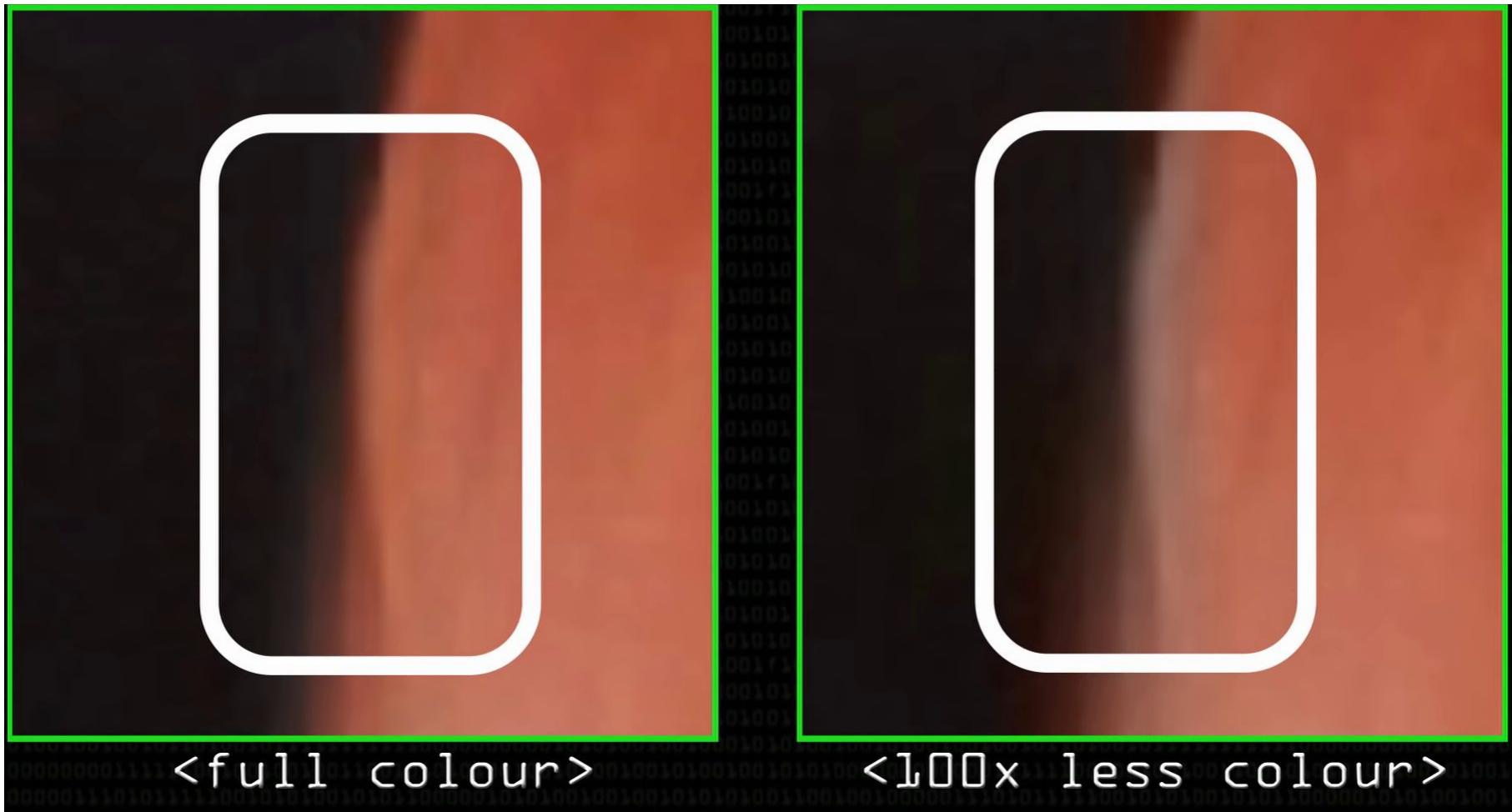


<full colour>



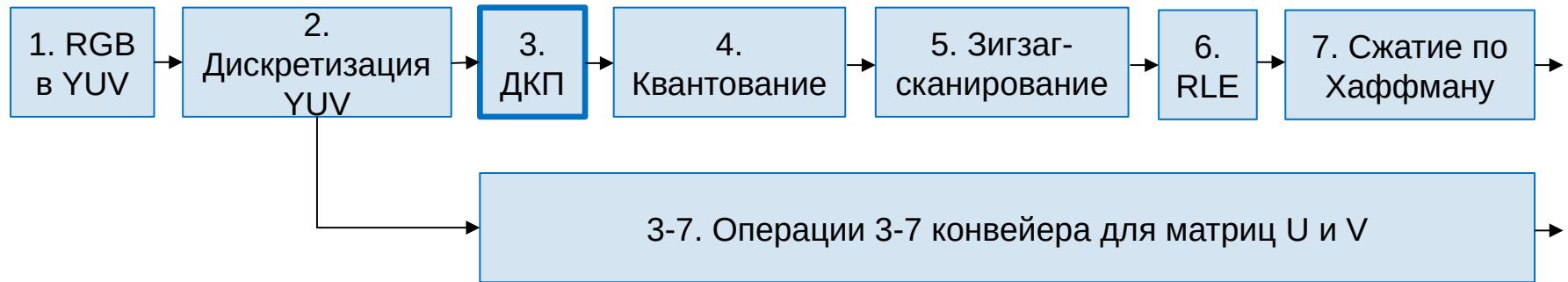
<100x less colour>

# Дискретизация YUV (4)



# Дискретное косинусное преобразование

# Этапы работы алгоритма JPEG



Конвейер операций, используемый в JPEG

# Разбиение на блоки

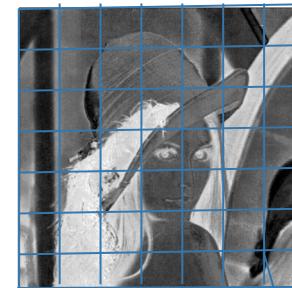
Изображение разбивается на блоки  $8 \times 8$  пикселей



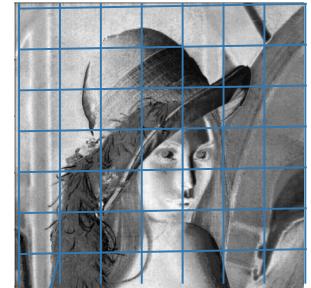
Исходное изображение



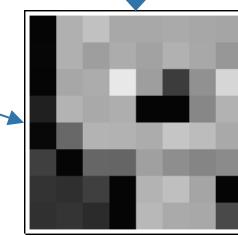
Y-компоненты разбитая  
на блоки  $8 \times 8$  пикселей



U-компонента



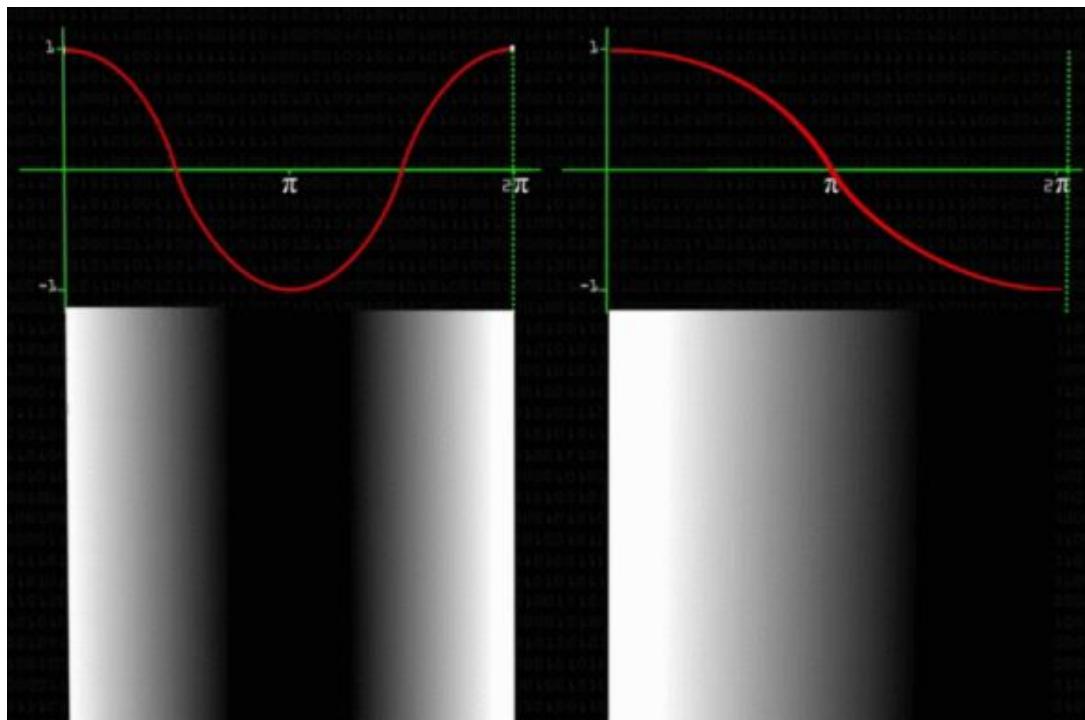
V-компонента



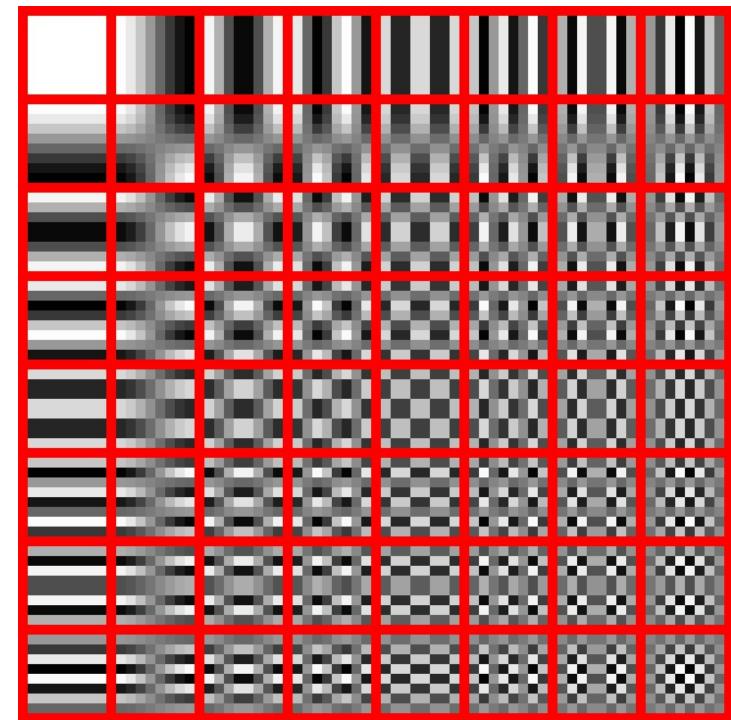
Блок  $8 \times 8$   
пикселей

# Идея разложения по базису (1)

Полученные блоки можно разложить по базису функций двумерных косинусов



Визуализация сложения двумерных косинусов

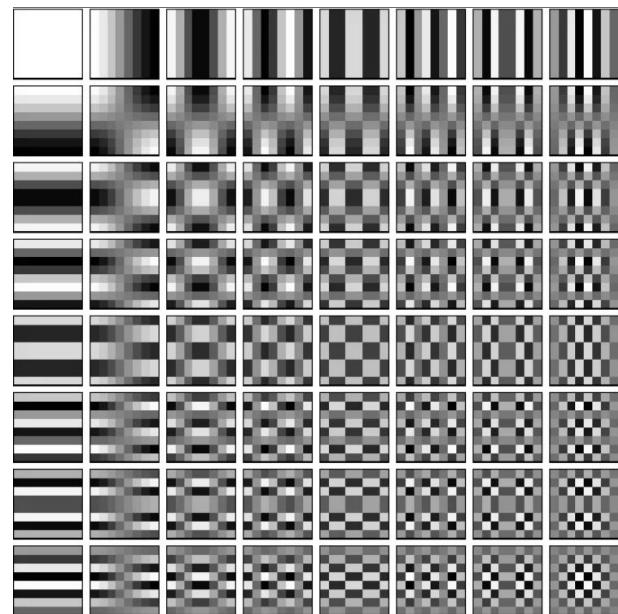


Базис из 64 двумерных косинусов (блоков 8x8)

# Идея разложения по базису (2)

+ 6,192 \*

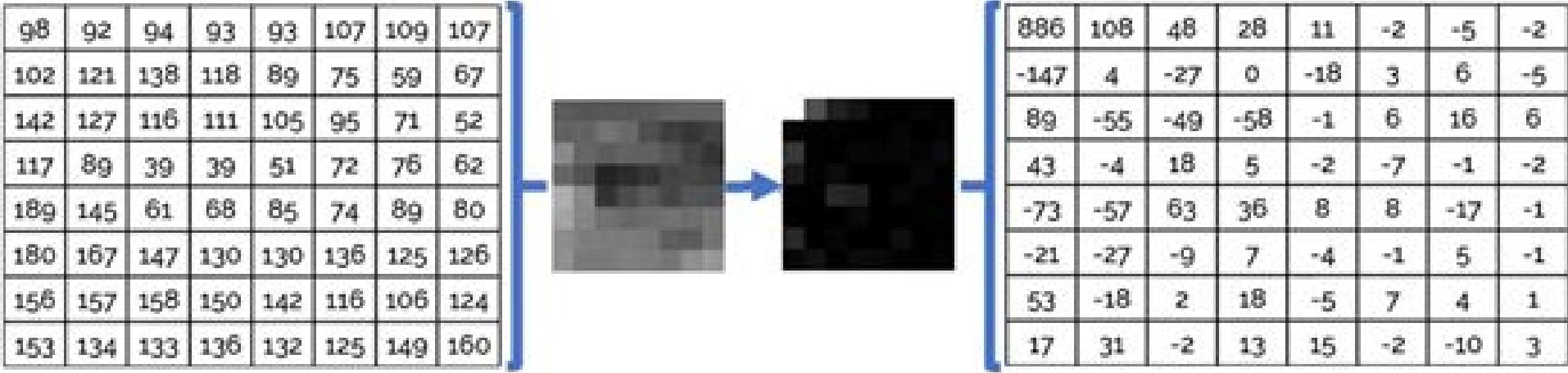
Комбинирование  
двумерных косинусов в  
букву «А»



Базис из 64 двумерных  
косинусов (блоков 8×8)

# Идея разложения по базису (3)

- Из каждого пикселя вычитаем 128
- Полученный блок  $8 \times 8$  раскладываем по базису из 64 двумерных косинусов с помощью дискретного косинусного преобразования



Блок изображения

Коэффициенты ДКП

# Формула ДКП

$$DCT : y[i, j] \rightarrow Y[u, v]$$

$$Y[u, v] = \frac{1}{4} \sum_{i=0}^{n-1} \sum_{j=0}^{n-1} C(i, u) \times C(j, v) \times y[i, j],$$

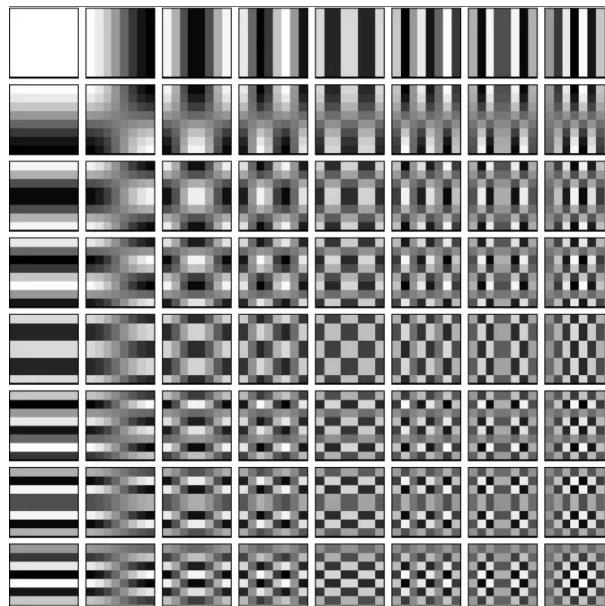
$$C(i, u) = A(u) \times \cos \frac{(2i + 1)u\pi}{2n},$$

$$A(u) = \begin{cases} \frac{1}{\sqrt{2}}, & \text{for } u = 0 \\ 1, & \text{for } u \neq 0 \end{cases}$$

# Дискретное косинусное преобразование. Примеры

# Матрица коэффициентов ДКП

После ДКП в матрице записаны коэффициенты, с которыми нужно сложить базисные блоки, чтобы получить исходный блок



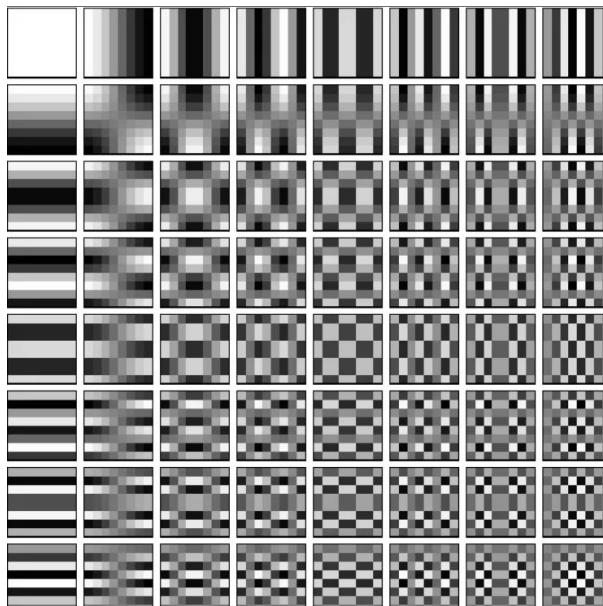
Базис из блоков 8x8

235.7	-0.9	-12.1	-5.4	2.0	-1.6	-2.5	1.5
-22.8	-17.6	-6.1	-2.9	-2.7	-0.1	0.2	-1.4
-10.8	-9.1	-1.7	1.3	0.0	-0.9	-0.4	0.1
-7.2	-2.0	0.3	1.7	1.0	-0.1	-0.2	0.2
-0.5	-0.7	1.4	1.4	-0.3	-0.6	0.8	1.4
1.7	-0.3	1.7	-0.2	-0.7	1.4	0.9	-1.1
-1.2	-0.3	-0.4	-1.6	-0.6	1.8	1.2	-0.7
-2.6	1.5	-3.7	-1.8	1.9	1.2	-0.6	-0.5

Блок 8x8 после ДКП

# Матрица коэффициентов ДКП

- Большую часть информации несут низкочастотные базисы
- JPEG сжимает высокие частоты**



Базис из блоков 8x8

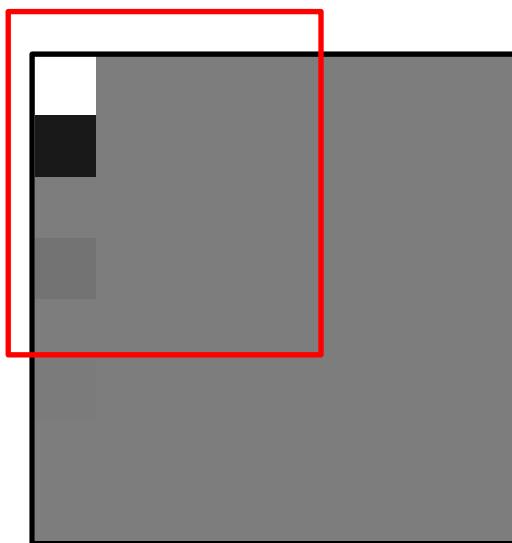
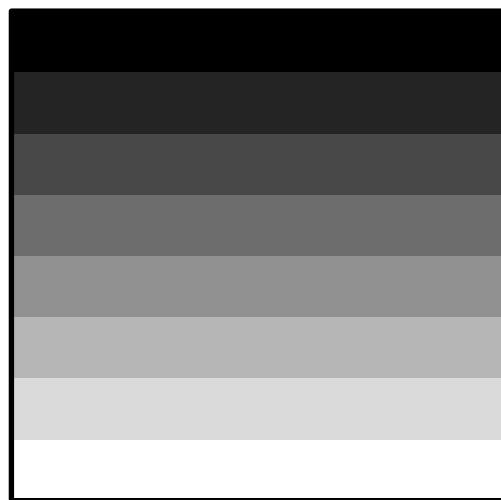
235.7	-0.9	-12.							
-22.8	-17.6	-6.1							
-10.8	-9.1	-1.7							
-7.2	-2.0	0.3							
-0.5	-0.7	1.4							
1.7	-0.3	1.7	0.2	0.7	1.1	0.5	1.1		
-1.2	-0.3	-0.4	-1.6	-0.6	1.8	1.2	-0.7		
-2.6	1.5	-3.7	-1.8	1.9	1.2	-0.6	-0.5		

Коэффициенты низкочастотных базисов больше по модулю, чем коэффициенты для высокочастотных

Блок 8x8 после ДКП

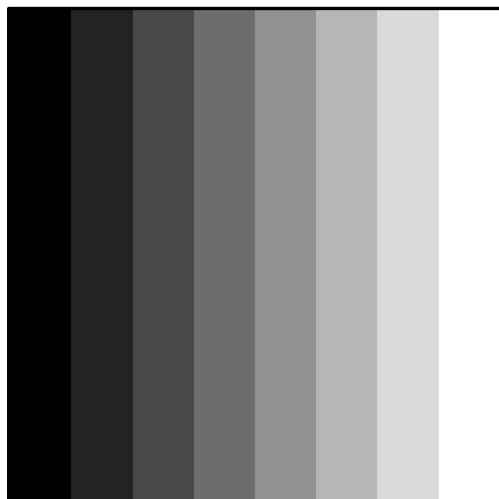
# Примеры ДКП (1)

Низкие частоты



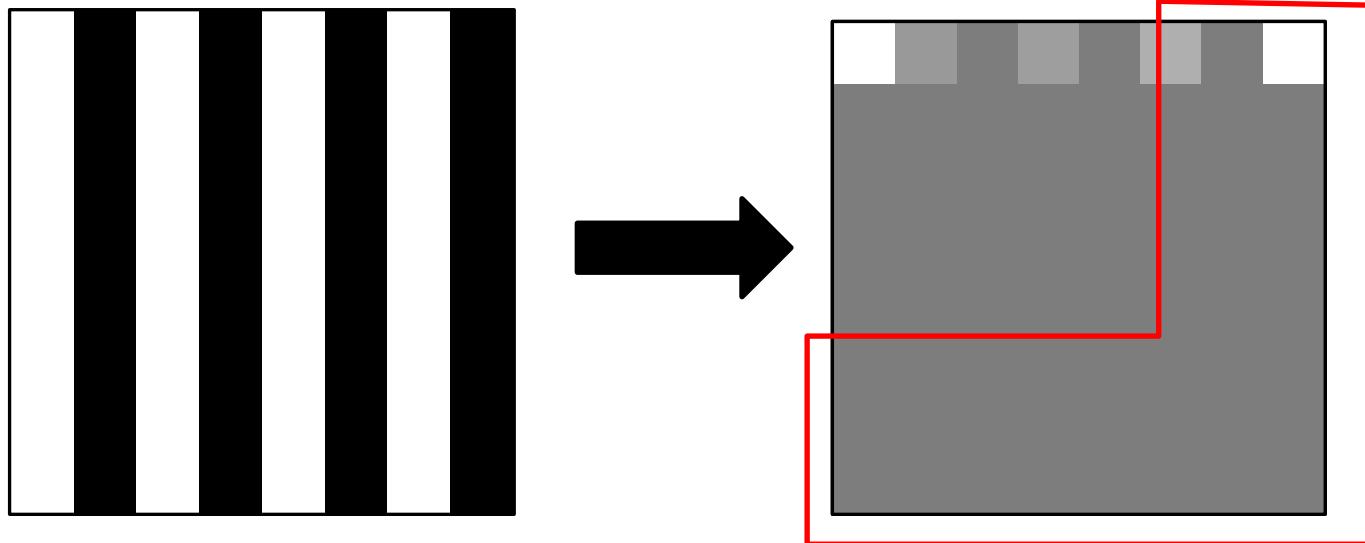
# Примеры ДКП (2)

Низкие частоты



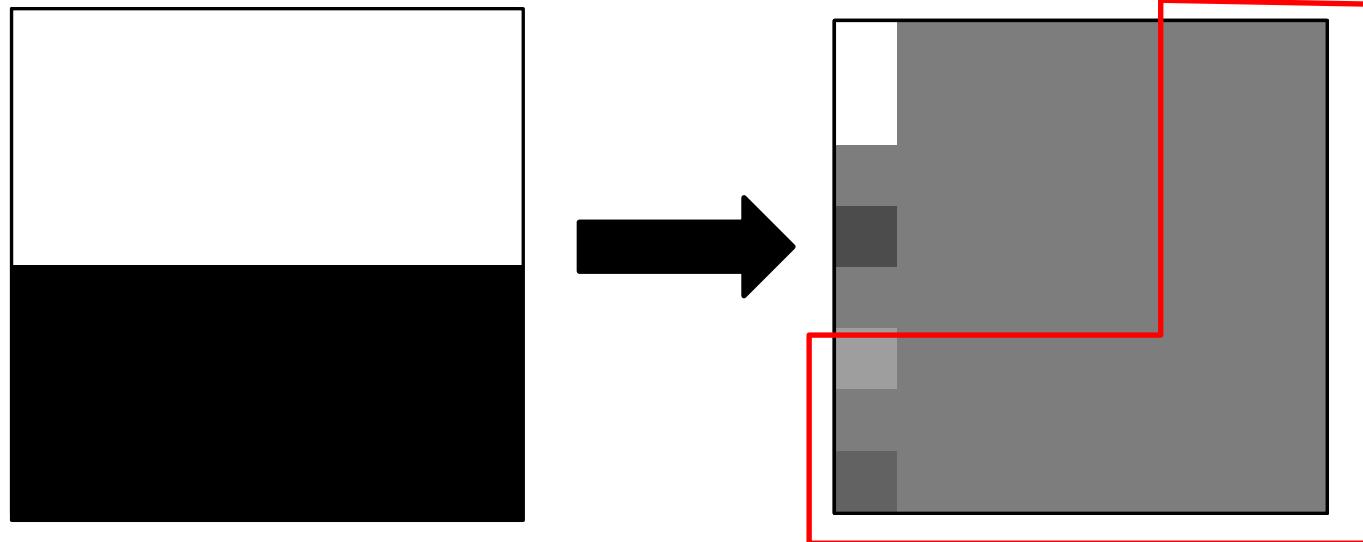
# Примеры ДКП (3)

Высокие частоты



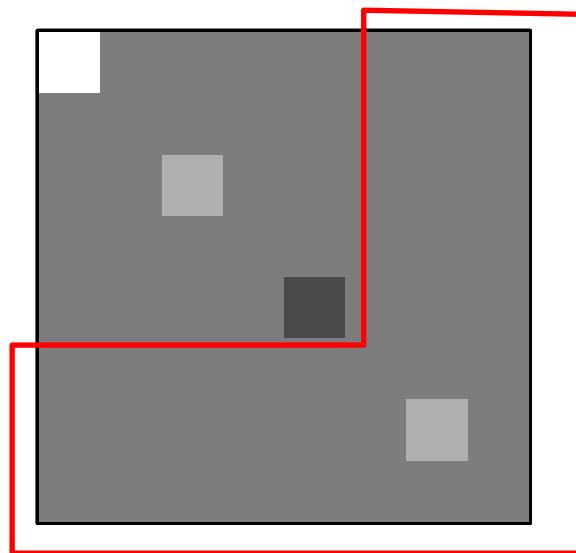
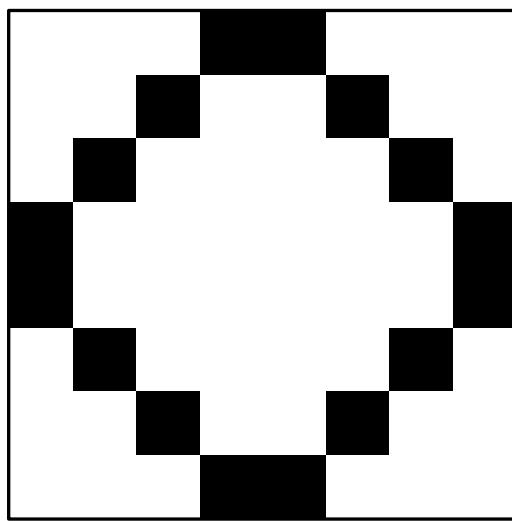
# Примеры ДКП (4)

Высокие частоты



# Примеры ДКП (5)

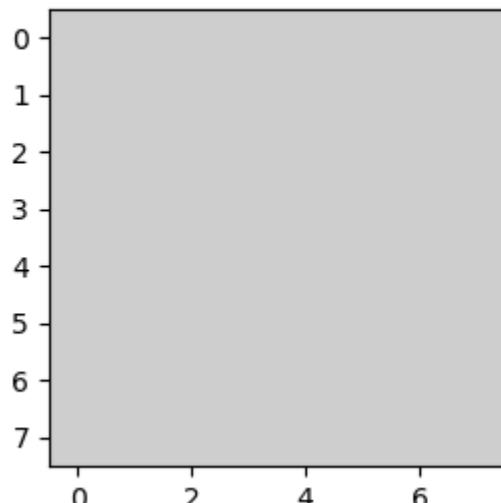
Высокие частоты



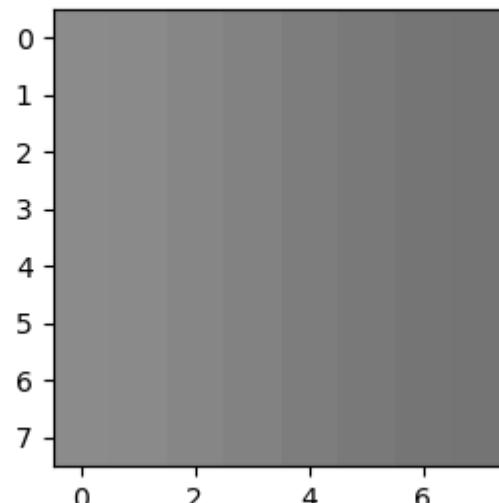
# Пример комбинирования двумерных косинусов в букву “A”



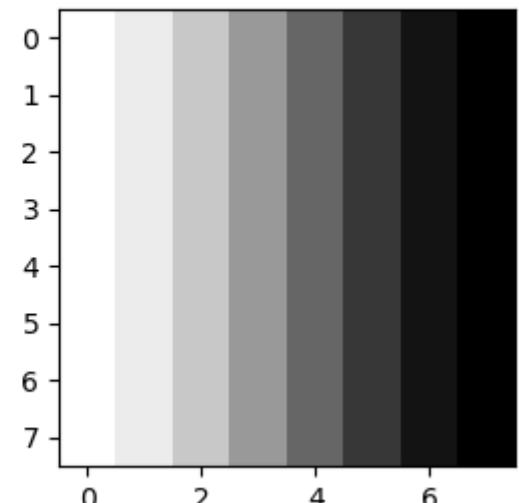
Reconstruction at 1 components



Coeff.: 0.267



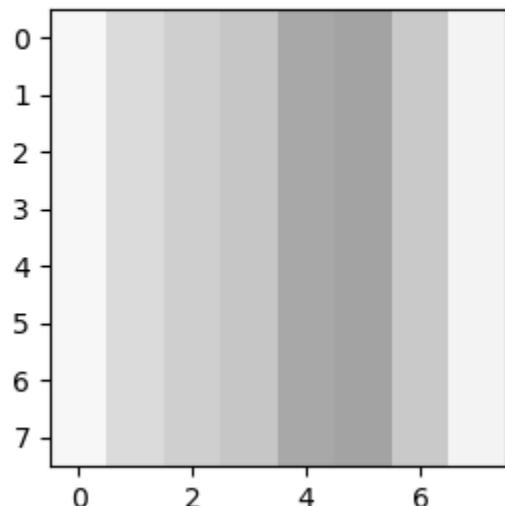
Base block at 0, 1



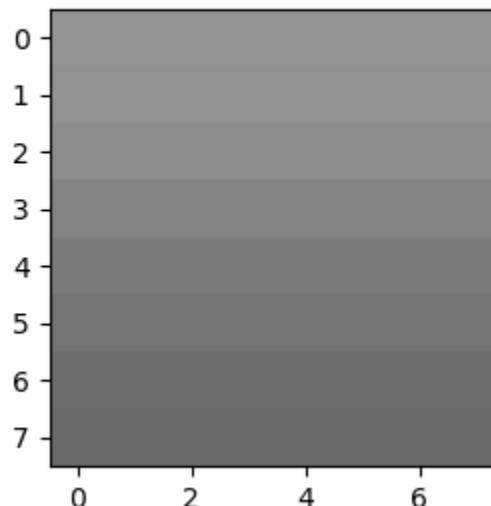
# Пример комбинирования двумерных косинусов в букву “A”



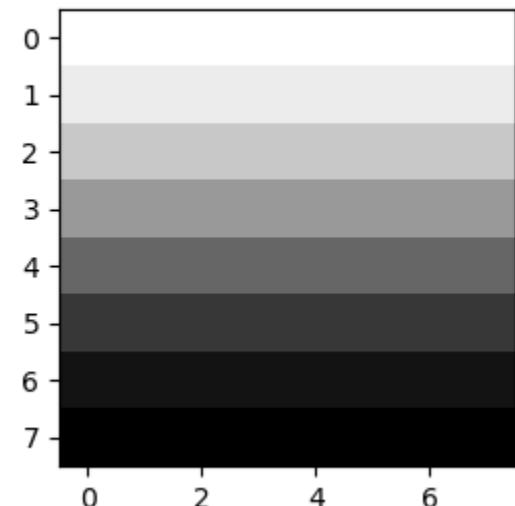
Reconstruction at 8 components



Coeff.: 0.490

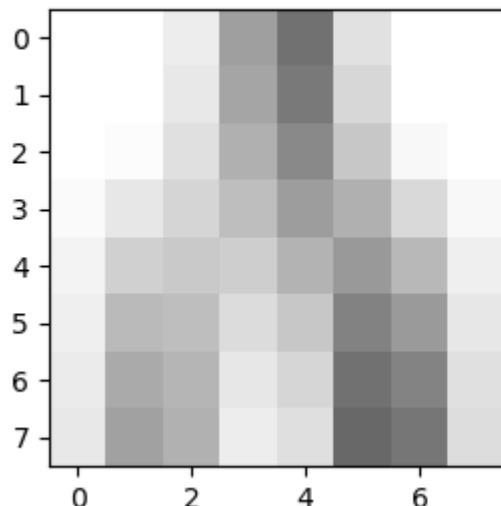


Base block at 1, 0

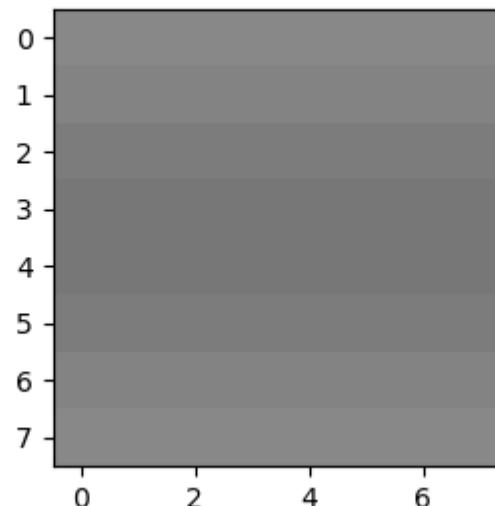


# Пример комбинирования двумерных косинусов в букву “A”

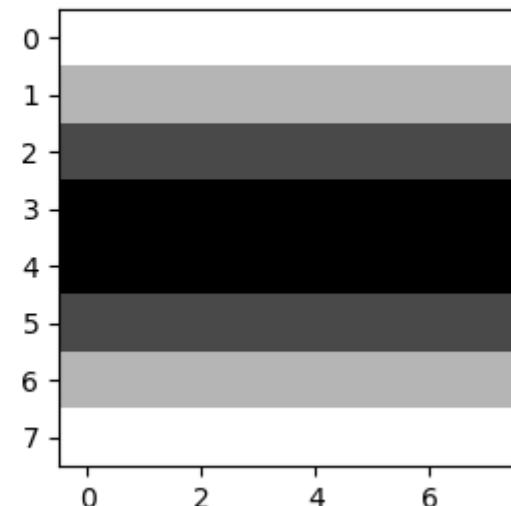
Reconstruction at 16 components



Coeff.: 0.201

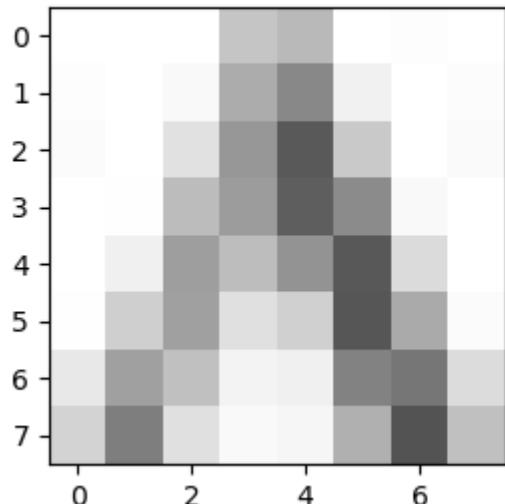


Base block at 2, 0

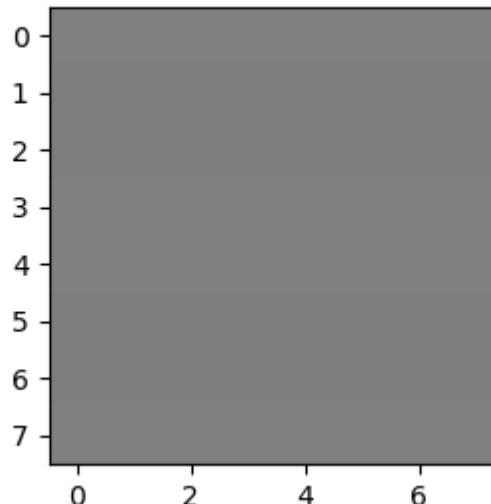


# Пример комбинирования двумерных косинусов в букву “A”

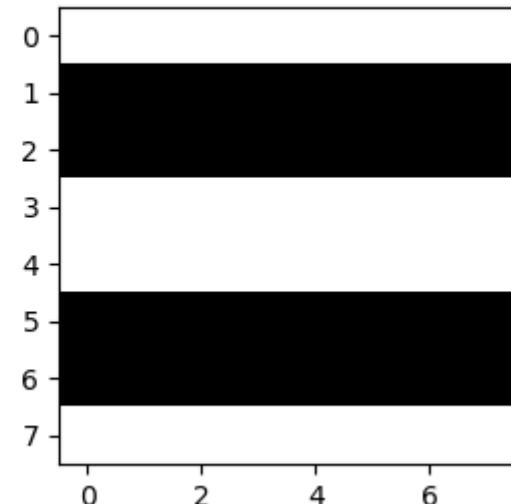
Reconstruction at 32 components



Coeff.: 0.055



Base block at 4, 0



# Квантование

# Этапы работы алгоритма JPEG



Конвейер операций, используемый в JPEG

# Матрица квантования

- Полученный блок (матрицу) поделим поэлементно на матрицу квантования и округлим
- В итоге получим блок, содержащий много нулей

DCT Coefficients

150	80	40	14	4	2	1	0
92	75	36	10	6	1	0	0
52	38	26	8	7	4	0	0
12	8	6	4	2	1	0	0
4	3	2	0	0	0	0	0
2	2	1	1	0	0	0	0
1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Quantization table

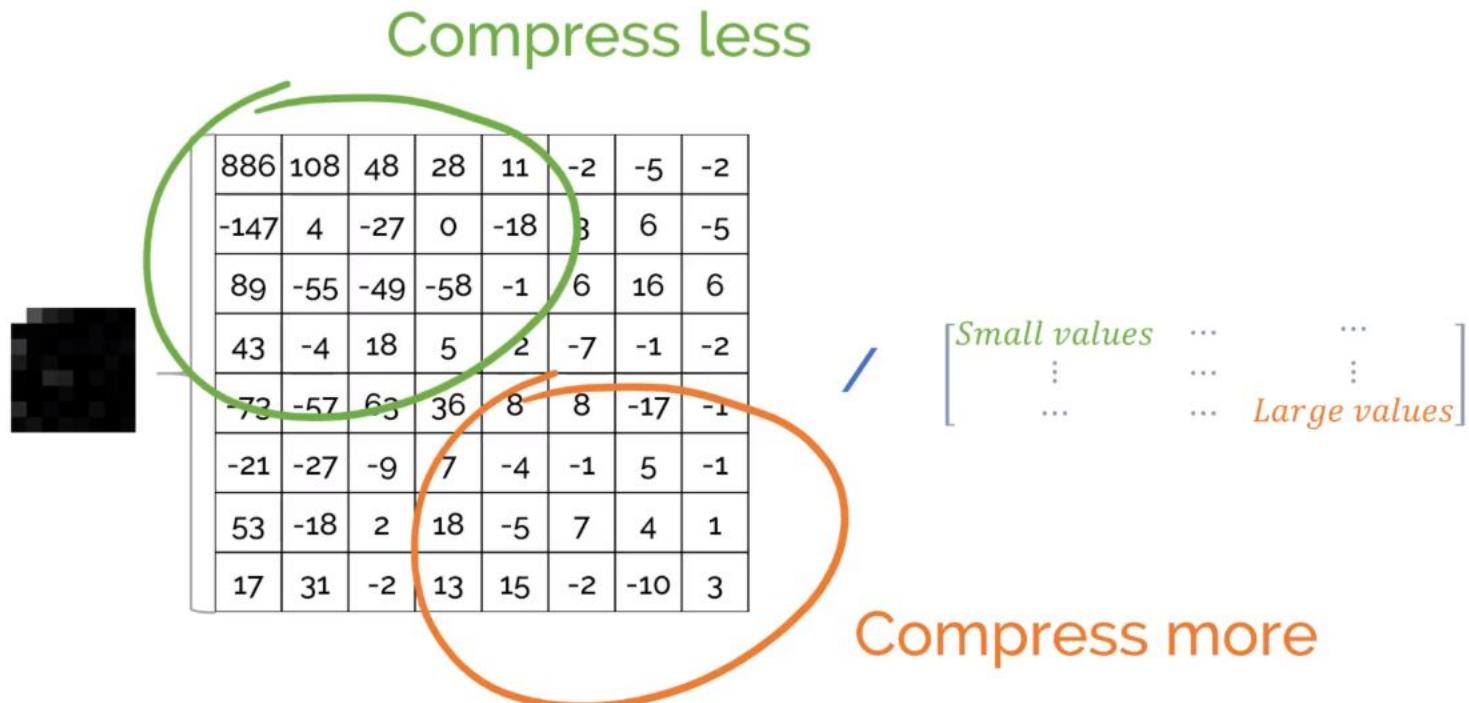
1	1	2	4	8	16	32	64
1	1	2	4	8	16	32	64
2	2	2	4	8	16	32	64
4	4	4	4	8	16	32	64
8	8	8	8	8	16	32	64
16	16	16	16	16	16	32	64
32	32	32	32	32	32	32	64
64	64	64	64	64	64	64	64

Quantized coefficients

150	80	20	4	1	0	0	0
92	75	18	3	1	0	0	0
26	19	13	2	1	0	0	0
3	2	2	1	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

# Квантование

Матрица коэффициентов поэлементно делится на матрицу квантования и округляется до целого



# Результат квантования (1)



Квантование с помощью  
однородной матрицы



Квантование с помощью  
подогнанной матрицы

# Результат квантования (2)



Квантование с помощью однородной матрицы



Квантование с помощью подогнанной матрицы

# Зигзаг-сканирование

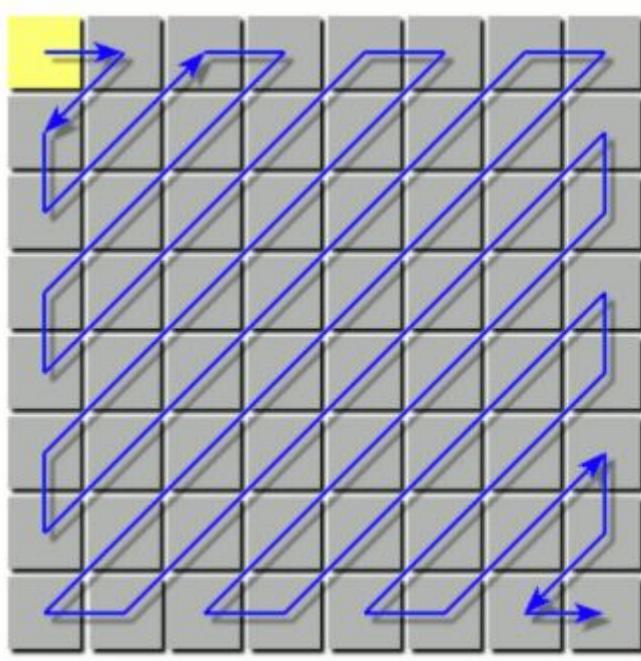
# Этапы работы алгоритма JPEG



Конвейер операций, используемый в JPEG

# Зигзаг-сканирование

Зигзаг-сканирование помещает высокочастотные косинусы подряд, благодаря чему получаются длинные серии нулей



# RLE и сжатие по Хаффману

# Этапы работы алгоритма JPEG

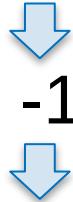


Конвейер операций, используемый в JPEG

# RLE и сжатие по Хаффману

Полученная после зигзаг-сканирования строка сжимается с помощью RLE, затем применяется сжатие по Хаффману

15, **0**, -2, -1, -1, **0**, **0**, -1, **0**, ..., **0**



15, **0**, **1**, -2, -1, -1, **0**, **2**, -1, **0**, **56**



# JPEG: Преимущества и недостатки

# Текстовое изображение “Lena”



Качество: 100

# Текстовое изображение “Lena”



Качество: 50

# Текстовое изображение “Lena”

Симптомы JPEG  
налицо...



Качество: 30

# Текстовое изображение “Lena”



Симптомы JPEG  
налицо...

Качество: 15

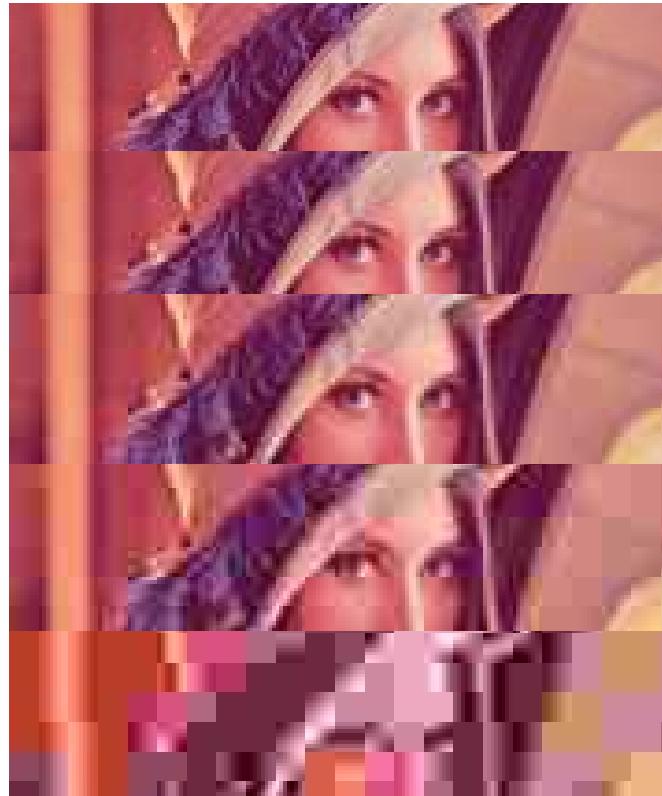
# Текстовое изображение “Lena”



Симптомы JPEG  
налицо...

Качество: 1

# Текстовое изображение “Lena”



Сравнение уровней качества

# JPEG не подходит для сжатия текста



При сжатии возникает “ореол” вокруг резких горизонтальных и вертикальных границ в изображении (эффект Гиббса)

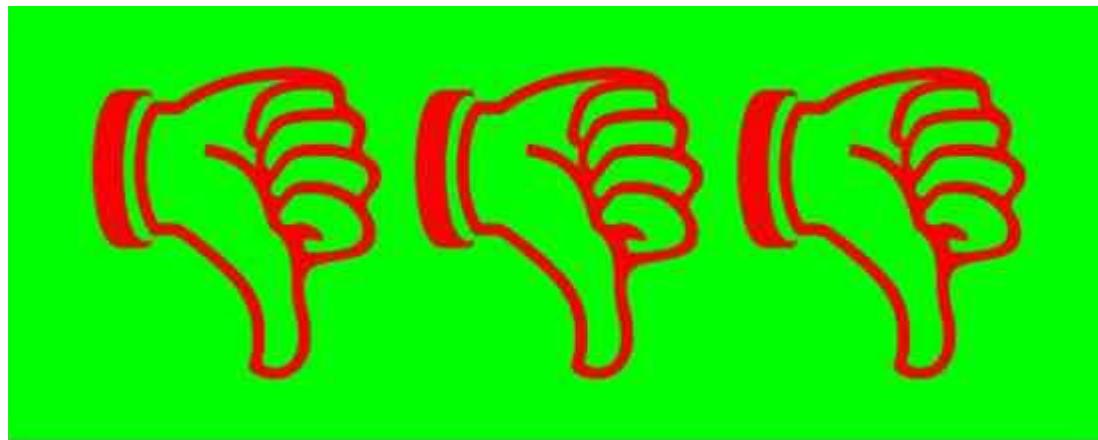
**JPEG не подходит  
для сжатия текста**

Качество: 1

# JPEG не подходит для сжатия графики



При сжатии возникает “ореол” вокруг резких горизонтальных и вертикальных границ в изображении (эффект Гиббса)



Качество: 30

# Характеристики JPEG

- **Коэффициенты компрессии:**  
2-100 (Задается пользователем)
- **Класс изображений:**  
Полноцветные 24-битные изображения или изображения в градациях серого без резких переходов цветов (фотографии)
- **Характерные особенности:**  
В некоторых случаях алгоритм создает “ореол” вокруг резких горизонтальных и вертикальных границ в изображении (эффект Гиббса). Кроме того, при высокой степени сжатия изображение распадается на блоки 8×8 пикселов



# JPEG 2000



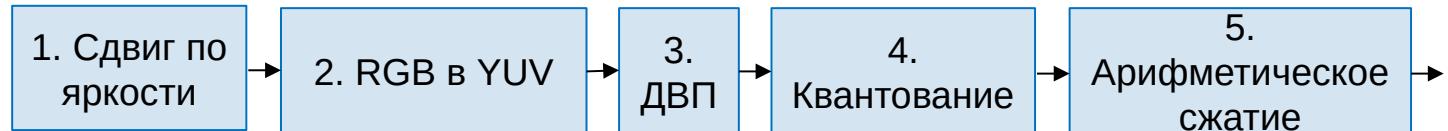
# JPEG 2000

- Алгоритм JPEG 2000 разработан той же группой экспертов в области фотографии, что и JPEG. Формирование JPEG как международного стандарта было закончено в 1992 году
- В 1997 стало ясно, что необходим новый, более гибкий и мощный стандарт, который и был доработан к зиме 2000 года

# Отличия в этапах работы JPEG и JPEG 2000



Конвейер операций, используемый в JPEG



Конвейер операций, используемый в JPEG 2000

# Отличия реализации от JPEG (1)

Базовая схема JPEG 2000 очень похожа на базовую схему JPEG. Отличия заключаются в следующем:

- Вместо дискретного косинусного преобразования (DCT) используется дискретное вейвлетное преобразование (DWT)
- Вместо кодирования по Хаффману используется арифметическое сжатие
- В алгоритм изначально заложено управление качеством областей изображения
- Не используется уменьшение разрешения цветоразностных компонент  $U$  и  $V$

# Отличия реализации от JPEG (2)

Базовая схема JPEG 2000 очень похожа на базовую схему JPEG. Отличия заключаются в следующем:

- Кодирование с явным заданием требуемого размера на ряду с традиционным методом кодирования по качеству
- Поддержка сжатия без потерь. Поддержка сжатия однобитных (2-цветных) изображений
- На уровне формата поддерживается прозрачность

# Этапы работы алгоритма JPEG 2000



## Конвейер операций, используемый в JPEG 2000



# Этапы работы алгоритма JPEG 2000



## Конвейер операций, используемый в JPEG 2000



# RGB в YUV

Этот шаг аналогичен JPEG (см. матрицы преобразования в описании JPEG), за тем исключением, что кроме преобразования с потерями предусмотрено также и преобразование без потерь

$$\begin{vmatrix} Y \\ U \\ V \end{vmatrix} = \begin{pmatrix} \left\lfloor \frac{R + 2G + B}{4} \right\rfloor \\ R - G \\ B - G \end{pmatrix} \quad \begin{vmatrix} R \\ G \\ B \end{vmatrix} = \begin{vmatrix} U + G \\ Y - \frac{U + V}{4} \\ V + G \end{vmatrix}$$

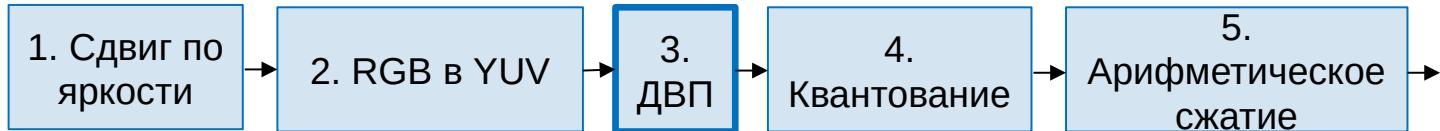
Матрицы преобразования без потерь

# Дискретное вейвлетное преобразование

# Этапы работы алгоритма JPEG 2000

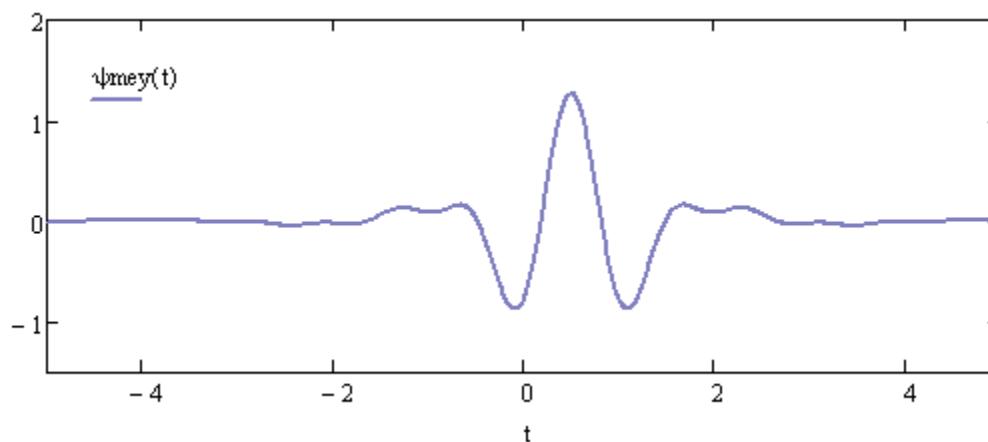


## Конвейер операций, используемый в JPEG 2000



# Вейвлетное преобразование

Вейвлетные преобразования используются в обработке сигналов, нередко заменяя обычное преобразование Фурье во многих областях физики. Вместо разложения по синусам и косинусам раскладывается по вейвлету (всплеску)



Пример вейвлета

# Дискретное вейвлетное преобразование (1)

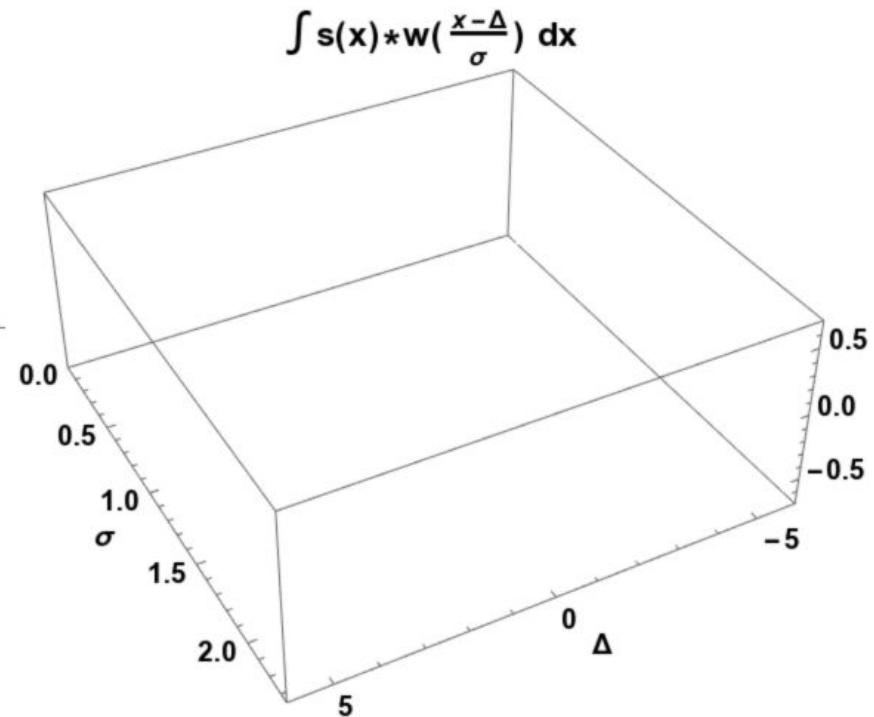


- В одномерном случае применение DWT – это «обычная фильтрация»
- Из строки мы получаем строку по приведенным формулам
- В двумерном случае мы сначала применяем эти формулы по всем строкам изображения, а потом по всем столбцам

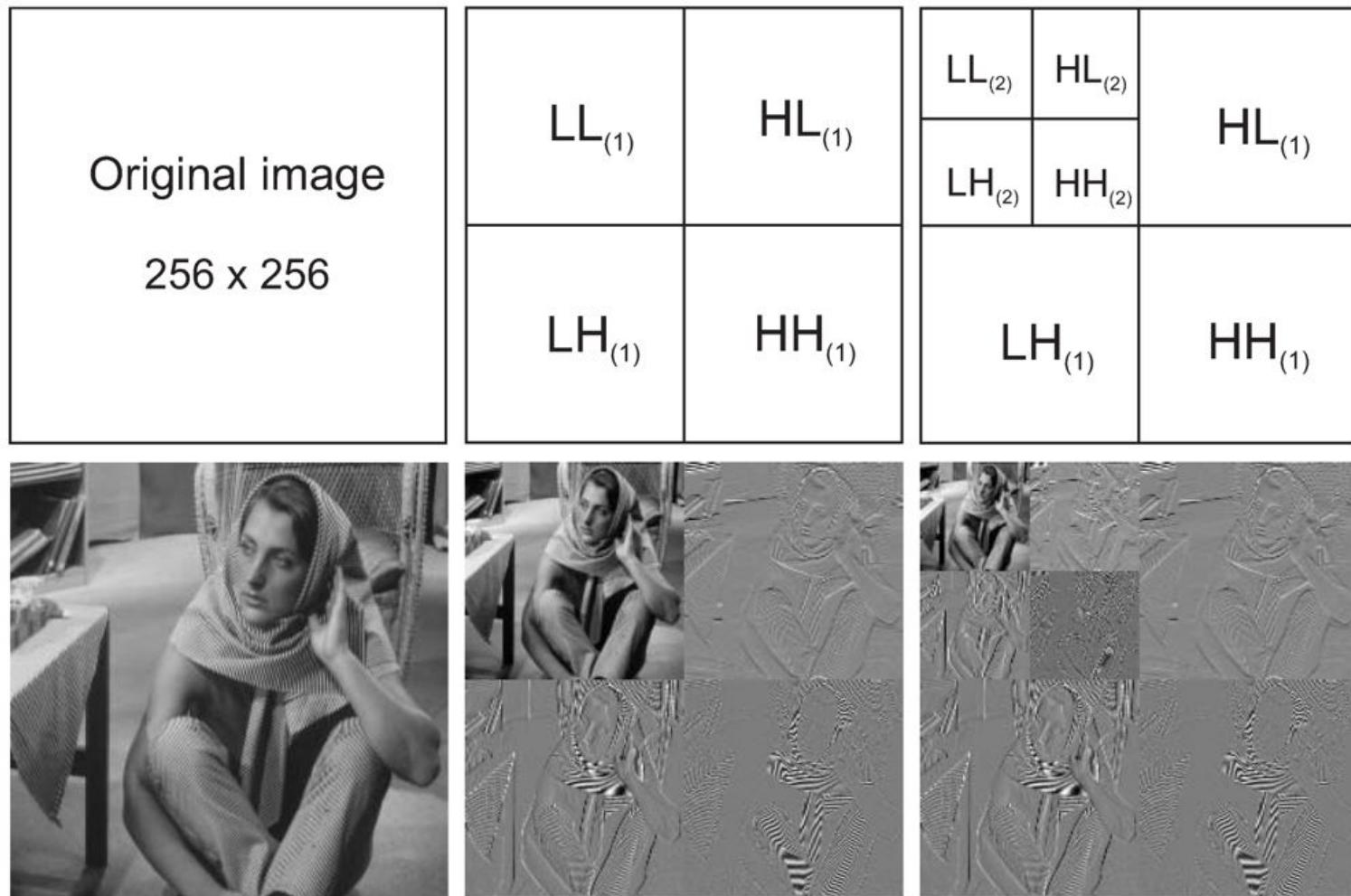
$$y_{out}(2n) = \sum_{j=0}^{N-1} x_{in}(j)h_H(j - 2n),$$

$$y_{out}(2n + 1) = \sum_{j=0}^{N-1} x_{in}(j)h_L(j - 2n - 1)$$

# Дискретное вейвлетное преобразование (2)



# Дискретное вейвлетное преобразование (3)



# Коэффициенты ДВП

Коэффициенты при упаковке

$i$	НЧ коэфф. $h_L(i)$	ВЧ коэфф. $h_H(i)$
0	<b>1.115087052456994</b>	<b>0.6029490182363579</b>
$\pm 1$	<b>0.5912717631142470</b>	<b>-0.2668641184428723</b>
$\pm 2$	<b>-0.05754352622849957</b>	<b>-0.07822326652898785</b>
$\pm 3$	<b>-0.09127176311424948</b>	<b>0.01686411844287495</b>
$\pm 4$	<b>0</b>	<b>0.02674875741080976</b>
Другие $i$	<b>0</b>	<b>0</b>

коэффициенты '9/7' DWT при сжатии с потерями

# Коэффициенты ДВП без потерь

	При упаковке		При распаковке	
$i$	НЧ коэфф. $h_L(i)$	ВЧ коэфф. $h_H(i)$	НЧ коэфф. $g_L(i)$	ВЧ коэфф. $g_H(i)$
0	6/8	1	1	6/8
$\pm 1$	2/8	-1/2	1/2	-2/8
$\pm 2$	-1/8	0	0	-1/8

Коэффициенты '5/3' DWT при сжатии без потерь

# Упрощение ДВП без потерь (1)

Поскольку большинство  $h_L(i)$ , кроме окрестности  $i=0$ , равны 0, то можно переписать приведенные формулы короче:

$$y_{out}(2n) = \frac{-x_{in}(2n-1) + 2 \cdot x_{in}(2n) + 6 \cdot x_{in}(2n+1) + 2 \cdot x_{in}(2n+2) - x_{in}(2n+3)}{8}$$

$$y_{out}(2n+1) = -\frac{x_{in}(2n)}{2} + x_{in}(2n+1) - \frac{x_{in}(2n+2)}{2}$$

# Упрощение ДВП без потерь (2)

После упрощения прямое преобразование задается как:

$$y_{out}(2n+1) = x_{in}(2n+1) - \left\lfloor \frac{x_{in}(2n) + x_{in}(2n+2)}{2} \right\rfloor$$

$$y_{out}(2n) = x_{in}(2n) + \left\lfloor \frac{y_{out}(2n-1) + y_{out}(2n+1) + 2}{4} \right\rfloor$$

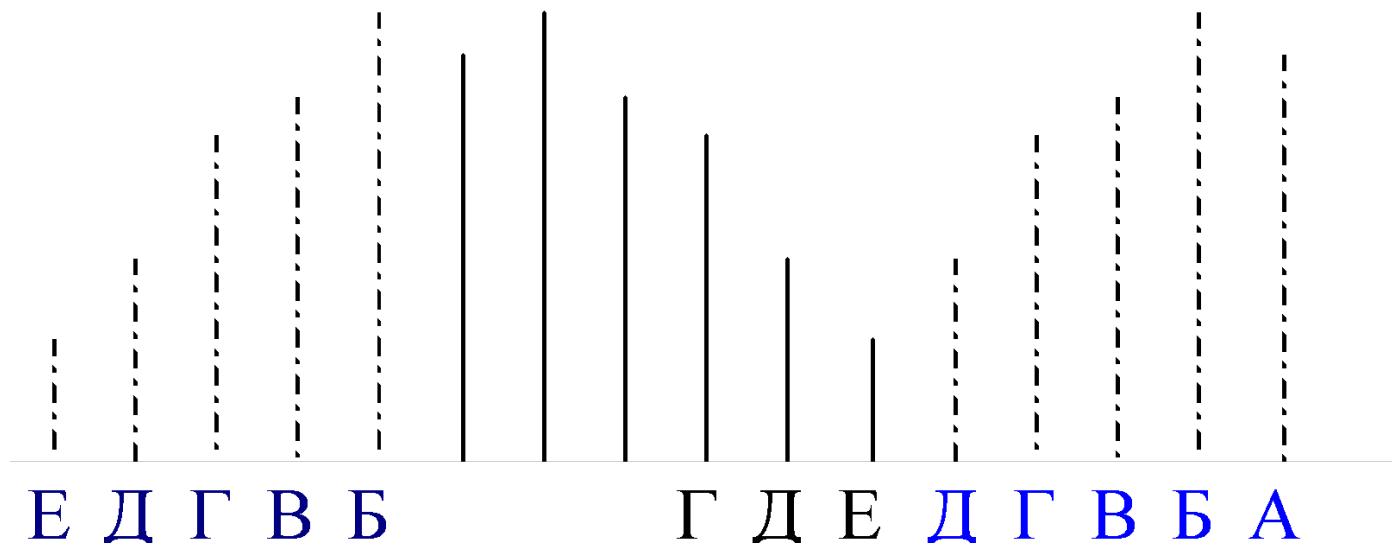
Обратное преобразование задается как:

$$x_{out}(2n) = y_{out}(2n) - \left\lfloor \frac{y_{out}(2n-1) + y_{out}(2n+1) + 2}{4} \right\rfloor$$

$$x_{out}(2n+1) = y_{out}(2n+1) + \left\lfloor \frac{x_{out}(2n) + x_{out}(2n+2)}{2} \right\rfloor$$

# Поведение ДВП на краях сигнала

Применение DWT на краях изображения:



Симметричное расширение изображения  
(яркости АБ...Е) по строке вправо и влево

# Пример ДВП

Пусть мы преобразуем строку из 10 пикселов.  
 Расширим ее значения вправо и влево и применим DWT преобразование:

n	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11
$x_{in}$	3	2	1	2	3	7	10	15	12	9	10	5	10	9
$y_{out}$		0	1	0	3	1	11	4	13	-2	8	-5		

Получившаяся строка 1, 0, 3, 1, 11, 4, 13, -2, 8, -5 полностью и однозначно задает исходные данные.  
 Обратное преобразование осуществляется по:

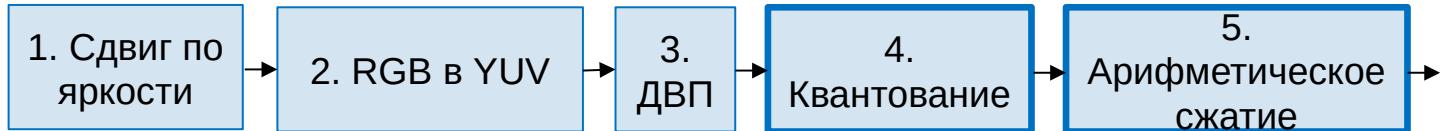
n	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11
$y_{out}$		0	1	0	3	1	11	4	13	-2	8	-5	8	-2
$x_{out}$			1	2	3	7	10	15	12	9	10	5	10	

# Квантование и арифметическое сжатие

# Этапы работы алгоритма JPEG 2000



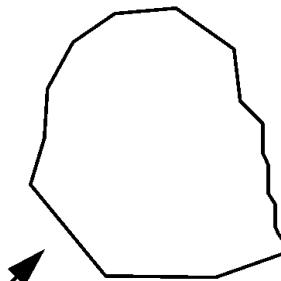
## Конвейер операций, используемый в JPEG 2000



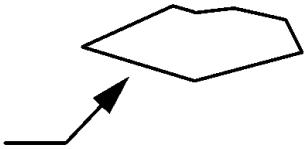
# Изменение качества областей

Когда практически достигнут предел сжатия изображения в целом и различные методы дают очень небольшой выигрыш, мы можем существенно (в разы) увеличить степень сжатия за счет изменения качества разных участков изображения.

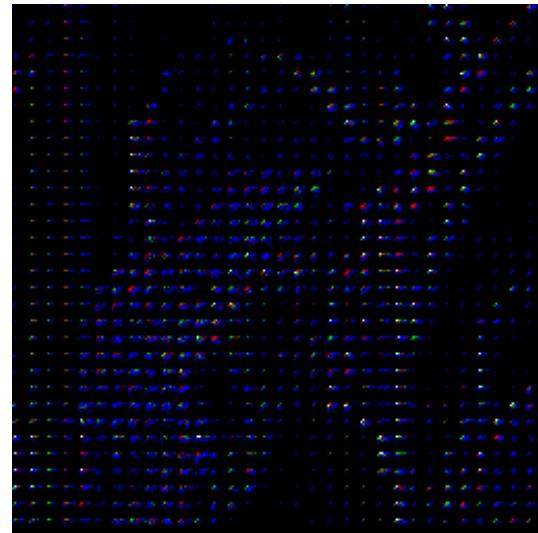
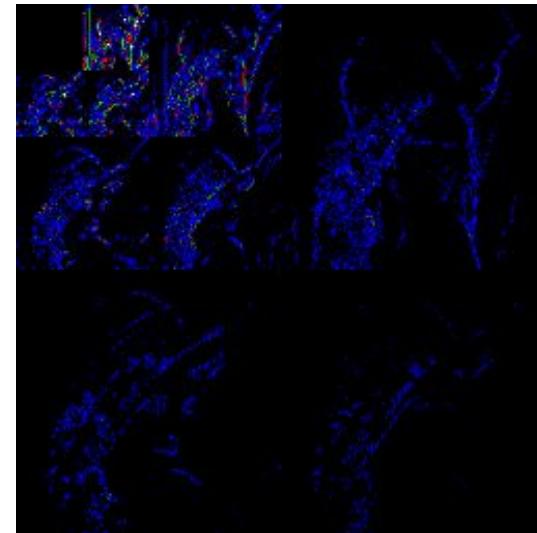
Изображение,  
сжатое с  
большими  
потерями



Области  
повышенного  
качества



# Сравнение этапа сжатия без потерь JPEG и JPEG-2000

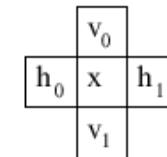
	JPEG	JPEG-2000
Структура разбиения	пространственная	частотная
Проход по коэффициентам	частотный	пространственный
Кодирование коэффициентов	групповое кодирование не нулевых	проходы по соответствующим битам коэффициентов
		

# Кодирование битовых плоскостей

- Разбиение DWT-пространства на одинаковые блоки, по умолчанию размером 64x64
  - Каждый блок кодируется независимо от других
  - В отличие от EZW и SPIHT (*set partitioning in hierarchical trees*) межуровневые зависимости не учитываются
- Кодирование одной битовой плоскости одного блока осуществляется в три этапа:
  - Кодирование старших бит
  - Уточняющий проход
  - Очищающий проход

# Кодирование старших бит

- Кодирование предсказанных и при подтверждении гипотезы, кодирование знака
- Контекст при кодирования значимости:
  - значимость соседних 8 связанных коэффициентов
  - Тип бэнда: LL,LH,HL,HH
- Контекст при кодирования знака:
  - Значимость и знаки 4-х связанных коэффициентов

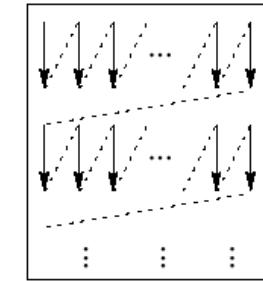


$d_0$	$v_0$	$d_1$
$h_0$	x	$h_1$
$d_2$	$v_1$	$d_3$

4-х и 8-ми  
связность

# Уточняющий и очищающий проходы

- Уточняющий проход:
  - Кодирование существенных битов, расположенных ниже первого
  - Контекст для бита:
    - «Это второй по важности бит?»
    - Значимость 8 связанных коэффициентов
- Очищающий проход:
  - Кодирование не предсказанных, но существенных битов



Порядок обхода

# Внешний цикл кодирования

Цель: записать в поток результаты кодирования битовых плоскостей.

Единица потока – пакет.

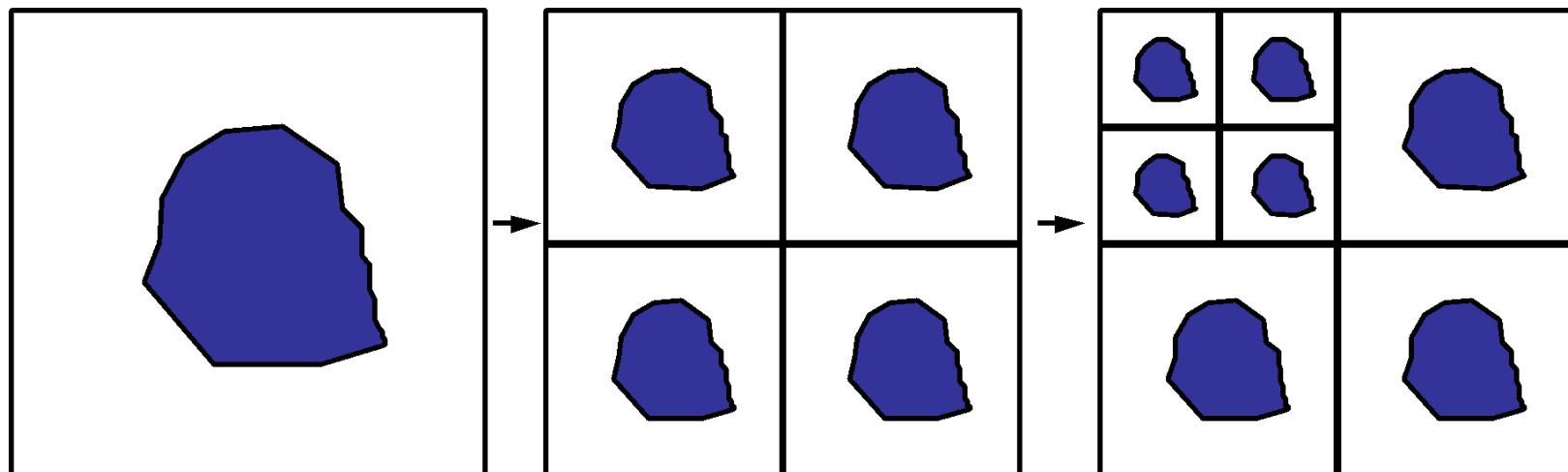
Пакет – компрессированный проход одной битовой плоскости одного блока.

Сортировка пакетов в соответствии с выбранной стратегией:

- Слой-разрешение-компонент-позиция:  
возможность прогрессивной визуализации
- Разрешение-слой-компонент-позиция:  
прогрессивная восстановление по разрешению
- Другие три сценария

# Изменение качества областей

В JPEG-2000 используется неявное представление бинарной маски, внутри которой точность квантования коэффициентов другая, нежели вне её. Метод представления и компрессии маски будет описан позже.



# Алгоритм изменения качества областей



- При кодировании:
  - Разделение битовой маски на выделенные и принадлежащие фону
  - Достаточный сдвиг (умножение на степень двойки) выделенных коэффициентов на , чтобы биты выделенного изображения и фона не пересекались
- При декодировании:
  - После распаковки, все коэффициенты, большие , сдвигаются направо на

Плюс такого подхода:

- Нет необходимости явного хранения бинарной маски

# Пресеты квантования



Адаптированный пресет,  
лучше качество

Стандартный пресет,  
больше PSNR

# Заключение

# Текстовое изображение “Lena” сжатое JPEG 2000



Качество: 100

# Текстовое изображение “Lena” сжатое JPEG 2000



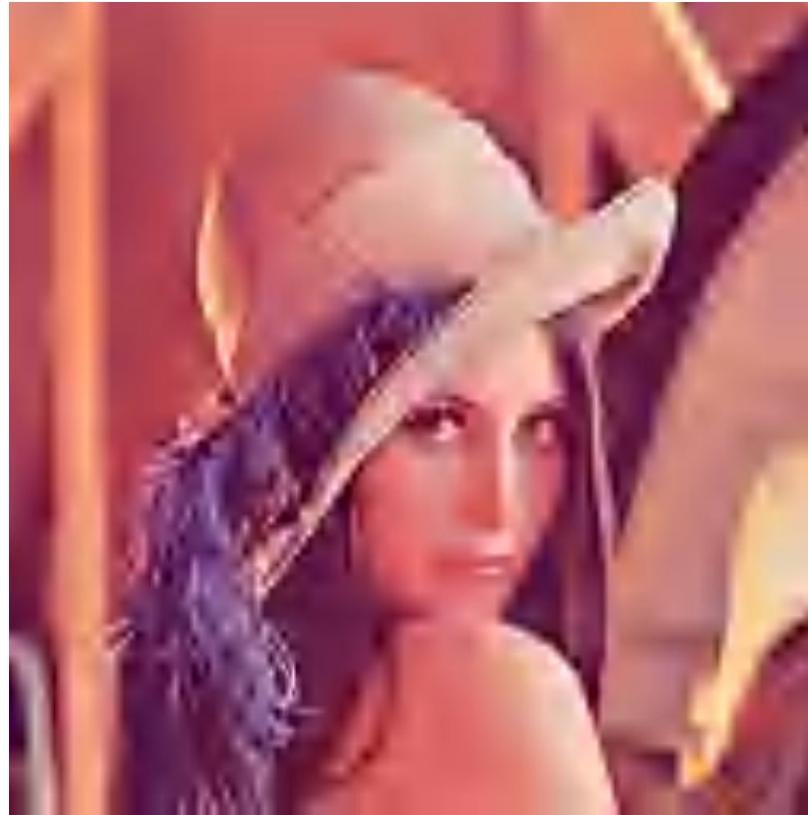
Качество: 50

# Текстовое изображение “Lena” сжатое JPEG 2000



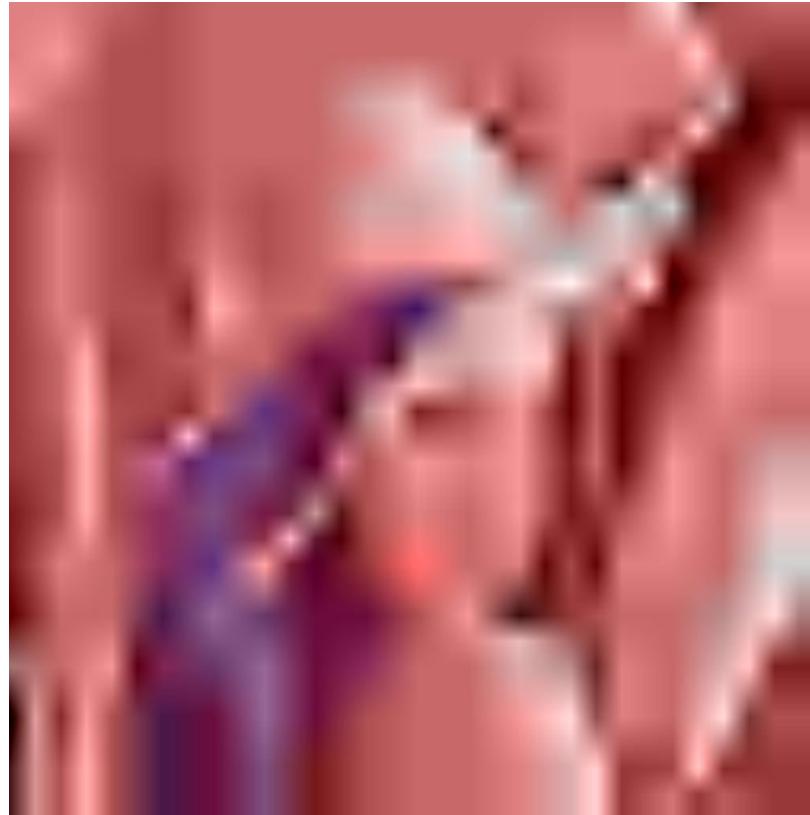
Качество: 30

# Текстовое изображение “Lena” сжатое JPEG 2000



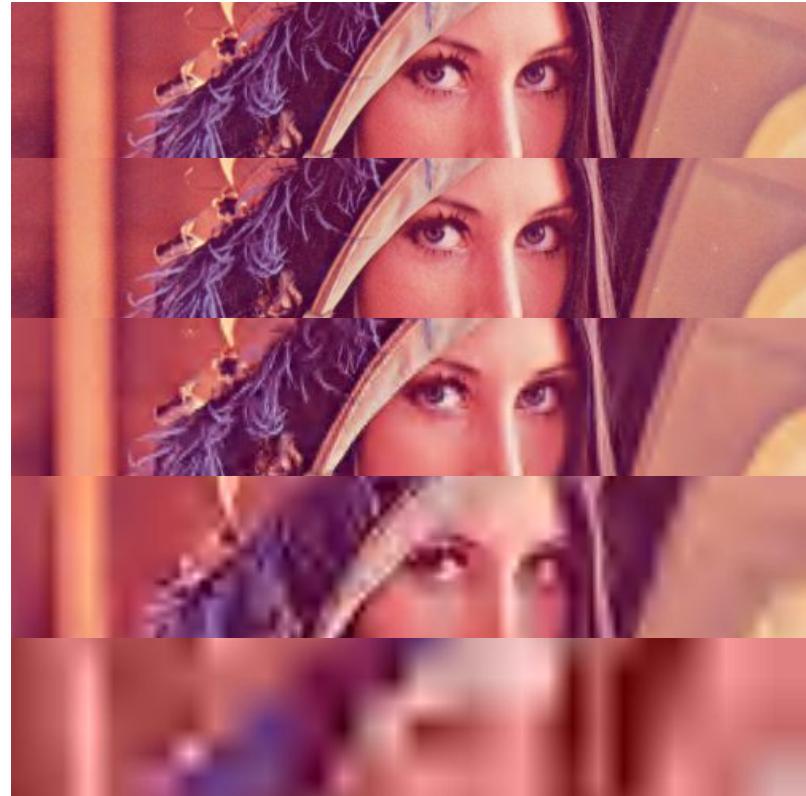
Качество: 25

# Текстовое изображение “Lena” сжатое JPEG 2000



Качество: 20

# Текстовое изображение “Lena” сжатое JPEG 2000



Сравнение уровня качества

# JPEG vs JPEG 2000. Сжатие в 30 раз



JPEG



JPEG 2000

# JPEG vs JPEG 2000. Сжатие в 130 раз



JPEG: сохранено  
больше деталей



JPEG-2000: отсутствие  
блочных артефактов

# Отличия JPEG 2000 от JPEG (1)

- Лучшее качество изображения при сильной степени сжатия
- Поддержка кодирования отдельных областей с лучшим качеством
- ДКП заменено на ДВП (плавное проявление изображения теперь изначально заложено в стандарт).  
Для повышения степени сжатия в алгоритме используется арифметическое сжатие
- Кодирование с явным заданием требуемого размера наряду с традиционным метод кодирования по качеству

# Отличия JPEG 2000 от JPEG (2)

- Поддержка сжатия без потерь (становится возможным использование JPEG для сжатия медицинских изображений, в полиграфии, при сохранении текста под распознавание OCR)
- Поддержка сжатия однобитных (2-цветных) изображений
- На уровне формата поддерживается прозрачность
- На уровне формата поддерживаются включение в изображение информации о копирайте
- Поддержка устойчивости к битовым ошибкам при передаче и широковещании

# Характеристики JPEG 2000

- **Коэффициенты компрессии:**  
2-200 (Задается пользователем), возможно сжатие без потерь
- **Класс изображений:**  
Полноцветные 24-битные изображения, изображения в градациях серого, 1-битные изображения (JPEG-2000 - наиболее универсален)
- **Симметричность:** 1
- **Характерные особенности:**  
Можно задавать качество участков изображений

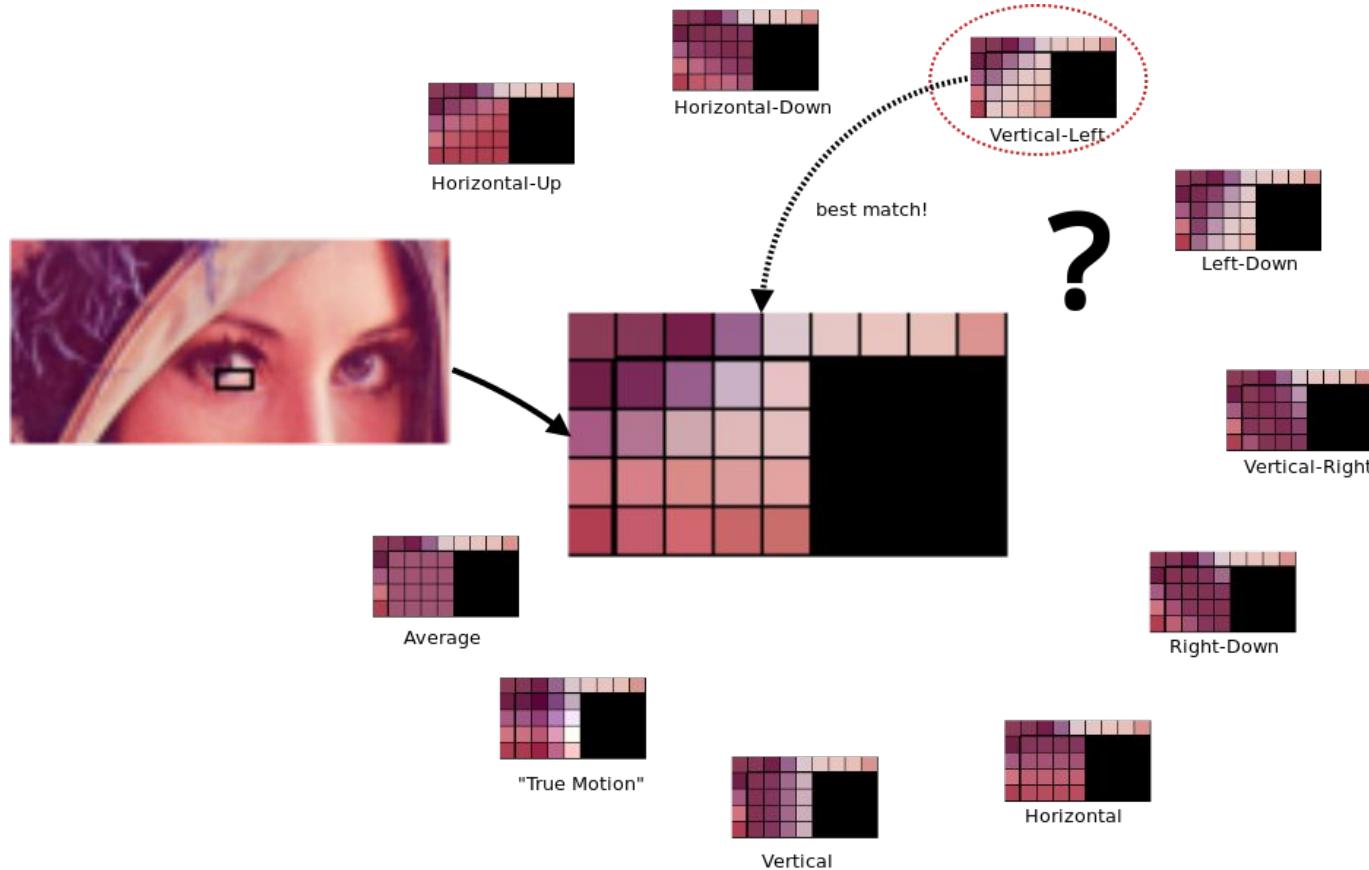
# WebP, HEIC и AVIF

---

# WebP

- WebP — формат сжатия изображений как с потерями, так и без, разработанный в 2010 году на основе алгоритма сжатия ключевых кадров видеокодека VP8
- Работает с блоками от  $16 \times 16$  до  $4 \times 4$
- Использует предсказание блоков на основе уже декодированных блоков (три блока над текущим и один блок слева от него)
- Для сжатия ошибок предсказания использует ДКП, адаптивное квантование и арифметическое кодирование
- Обладает хорошей поддержкой в браузерах

# Предсказание блоков в WebP



Разные предсказания 4×4 блоков

# High Efficiency Image File (HEIF)

- HEIF — формат файлов (контейнер) для хранения отдельных изображений или их последовательностей
- Разработка спецификации была завершена в 2015 году группой экспертов в области движущихся изображений (MPEG — Moving Picture Experts Group)
- Поддерживает изменение ориентации и обрезку изображений без перекодирования данных
- Позволяет упаковывать в один файл серии снимков
- Поддерживает HDR
- Проблемы с нативной поддержкой в браузерах

# HEIC и AVIF

## HEIC — HEVC in HEIF

- Основан на алгоритме сжатия ключевых кадров видеокодека HEVC
- Работает с блоками от  $64 \times 64$  до  $4 \times 4$
- Расширение файла .heic, .heics

## AVIF — AV1 in HEIF

- Основан на алгоритме сжатия ключевых кадров видеокодека AV1
- Работает с блоками от  $128 \times 128$  до  $4 \times 4$
- Расширение файла .avif, .avifs

# Текстовое изображение “Lena”



HEIC, Качество: 80  
230.0 kB



AVIF, Качество: 80  
74.0 kB

# Текстовое изображение “Lena”



HEIC, Качество: 60  
72.2 kB

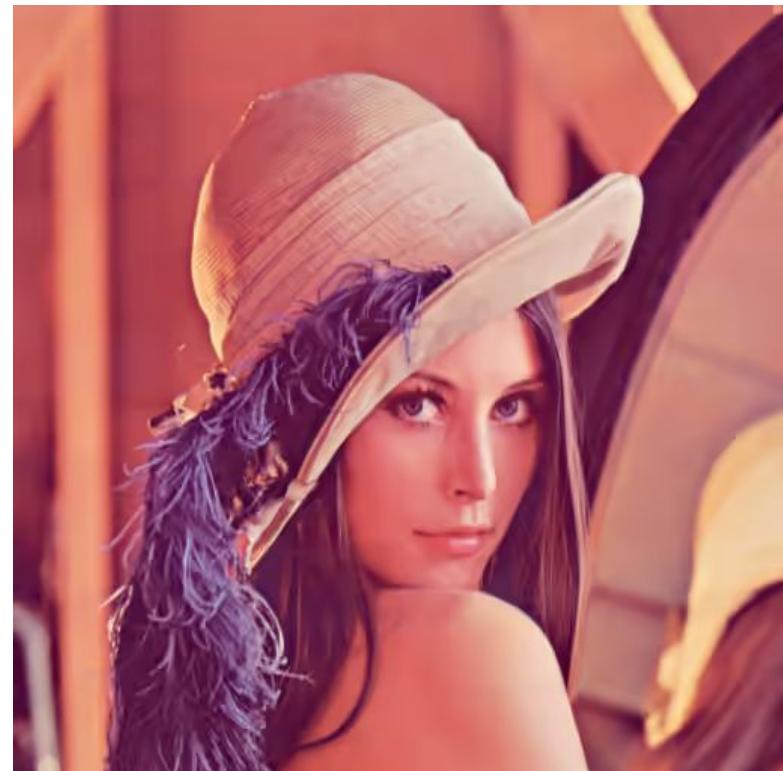


AVIF, Качество: 60  
32.2 kB

# Текстовое изображение “Lena”

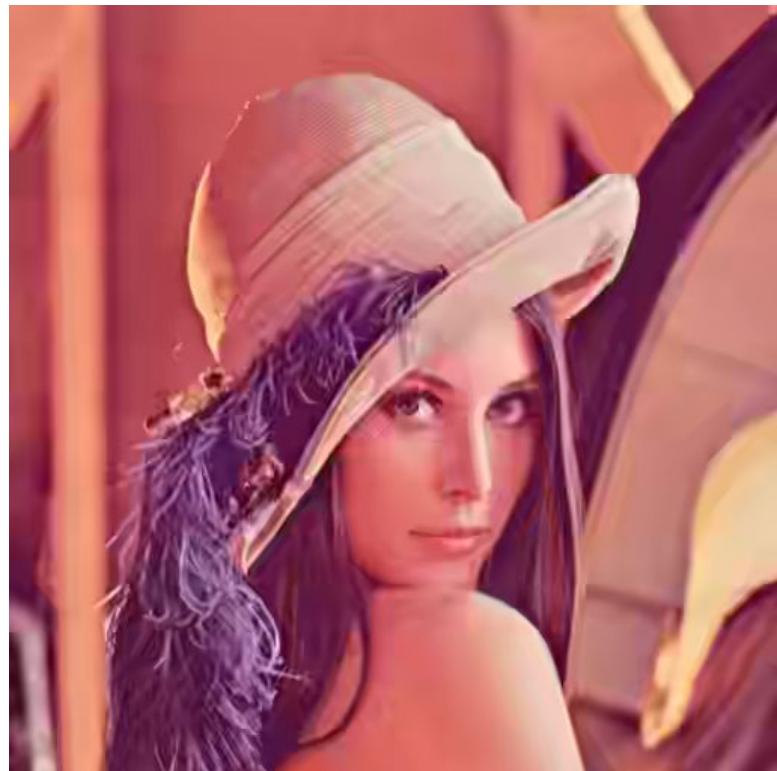


HEIC, Качество: 40  
22.5 kB



AVIF, Качество: 40  
17.9 kB

# Текстовое изображение “Lena”

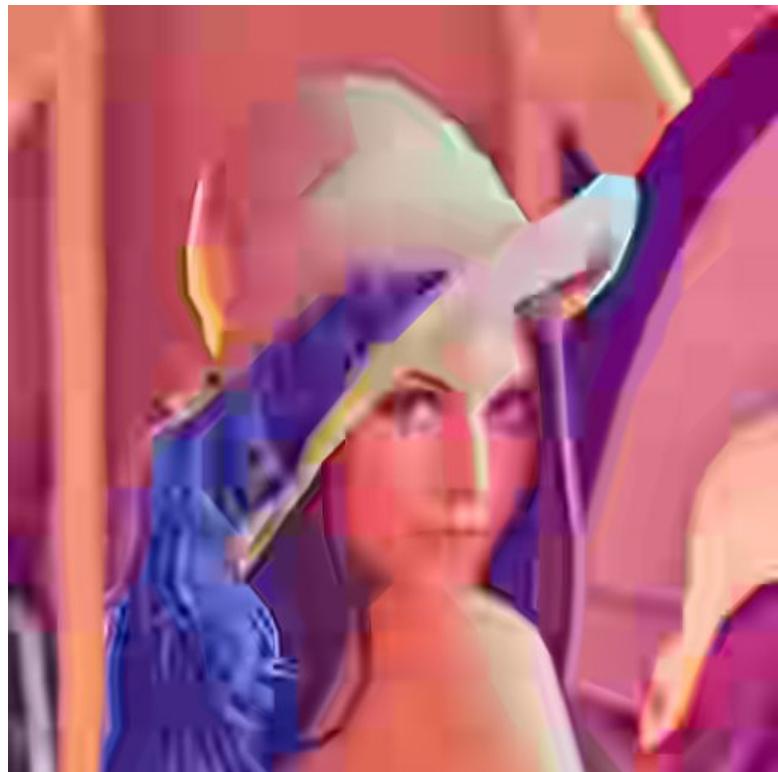


HEIC, Качество: 20  
13.2 kB

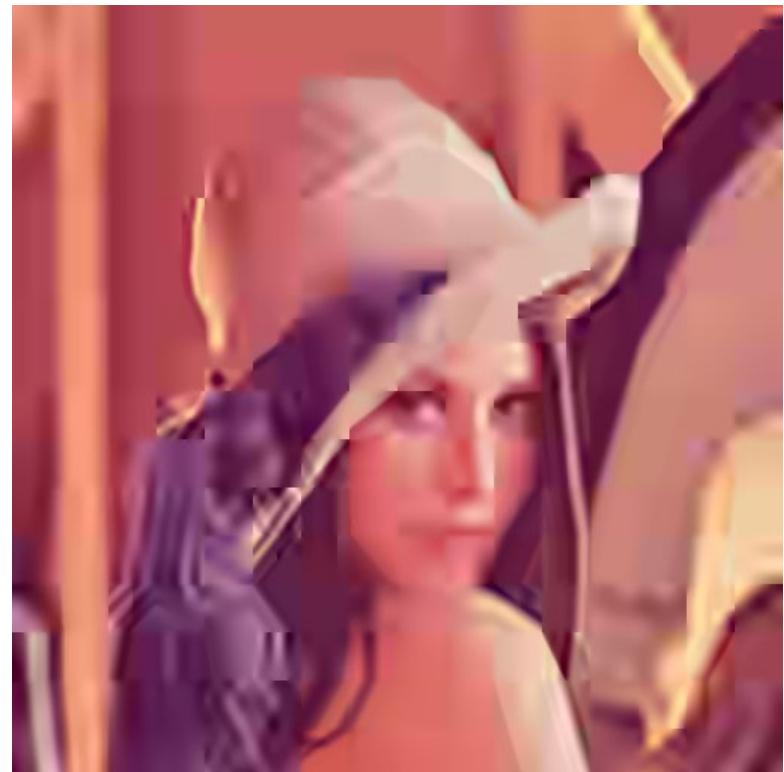


AVIF, Качество: 20  
13.0 kB

# Текстовое изображение “Lena”

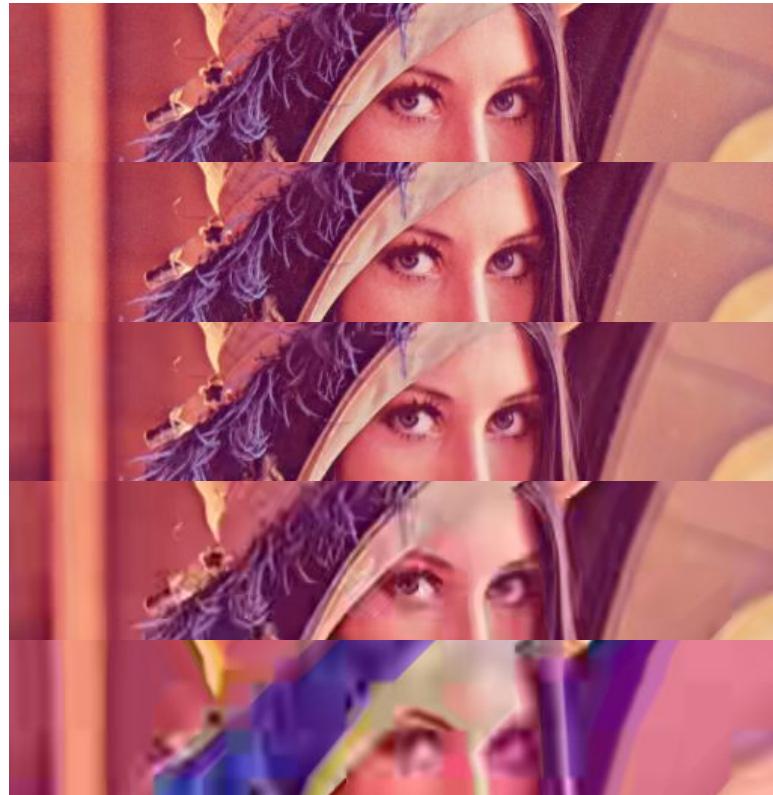


HEIC, Качество: 0  
10.5 kB

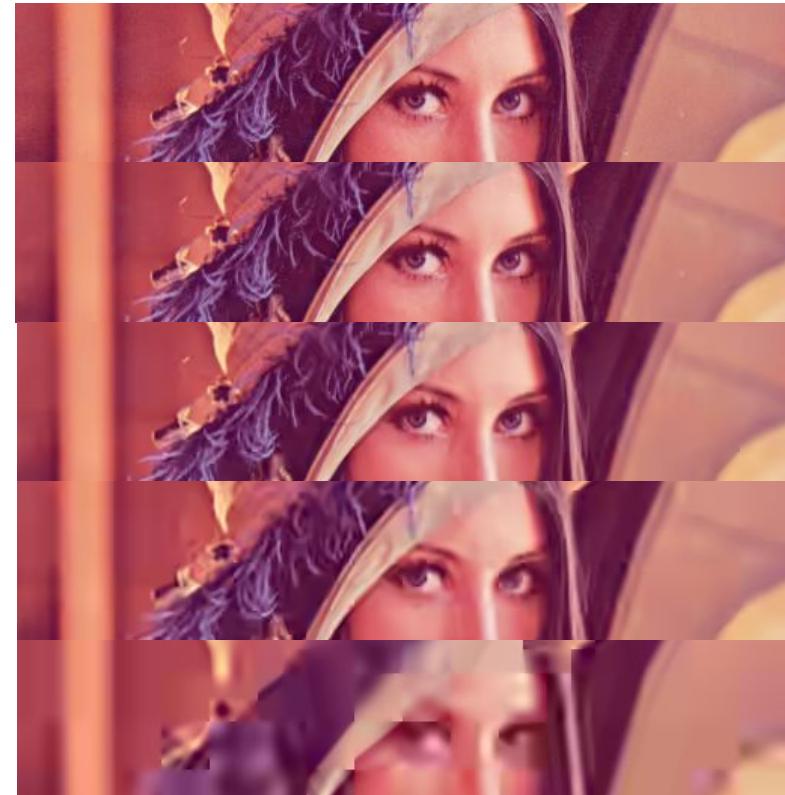


AVIF, Качество: 0  
10.2 kB

# Текстовое изображение “Lena”



HEIC, Сравнение  
уровней качества



AVIF, Сравнение  
уровней качества

# Текстовое изображение “Lena”



JPEG, Качество: 20  
12.3 kB



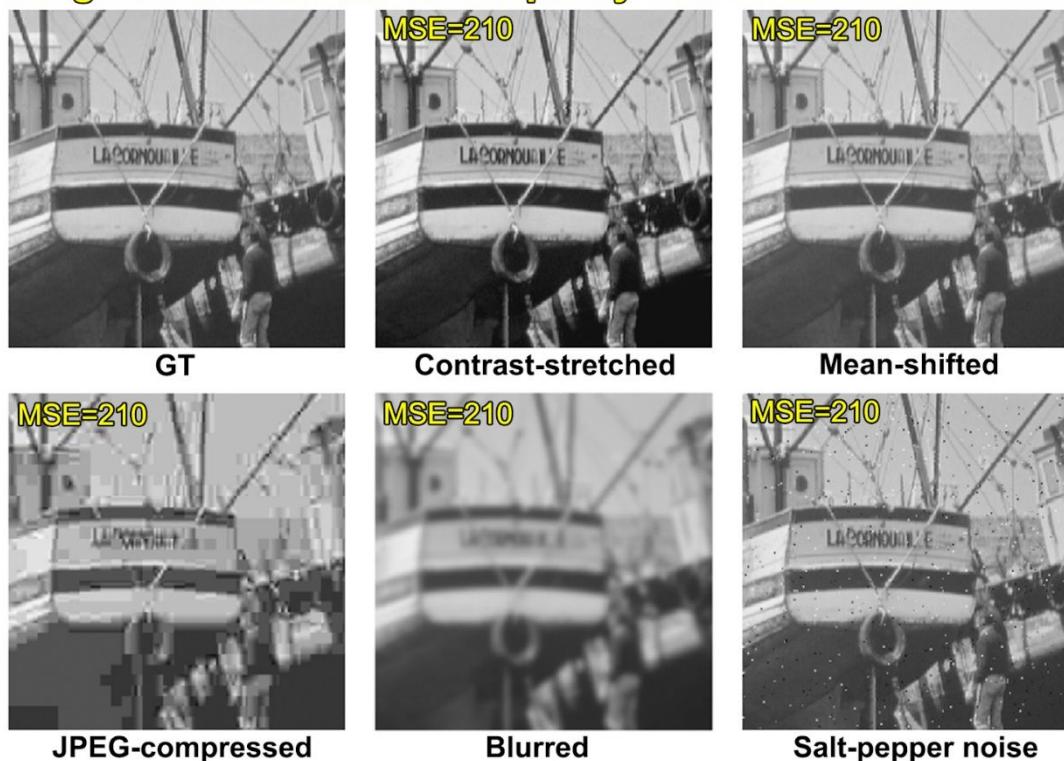
AVIF, Качество: 20  
13.0 kB

# Способы оценки качества

# Качество изображений

Не существует метода оценки качества изображения, полностью адекватного человеческому восприятию

Images with different visual quality but the same MSE score



# Задача текстовых наборов

Основные задачи тестовых наборов:

- Обеспечить единую базу сравнения разных алгоритмов (в статьях и т.п.)
- Обеспечить выявление разных типов артефактов в алгоритмах

# Тестовое изображение «Барбара»



Много полосок  
(высоких частот)  
в разных направлениях  
и разной толщины.

# Тестовое изображение «Boat»



Много тонких деталей  
и наклонных границ  
в разном направлении.

# PSNR – Peak Signal-to-Noise Ratio

Подходит для:

- Сравнения на фиксированном наборе изображений
- Оценки шума и изменений в контрасте и яркости

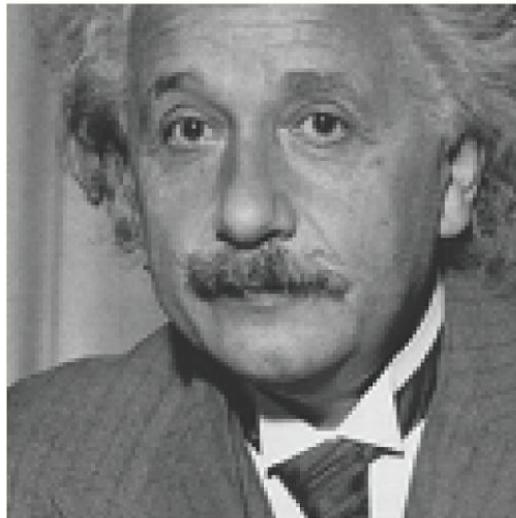
Не подходит для:

- Сравнения качества разных изображений
- Оценки размытия, поворота и смещения изображения

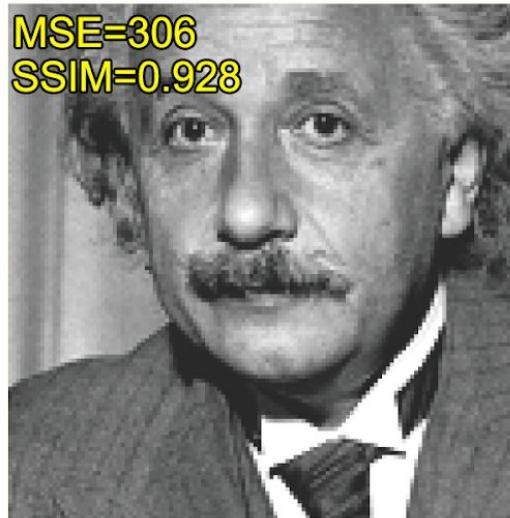
$$\text{PSNR}(x, y) = 10 \times \log_{10} \frac{255^2 \times n^2}{\sum_{i=1}^n \sum_{j=1}^n (x_{ij} - y_{ij})^2}$$

# Пример неприменимости PSNR

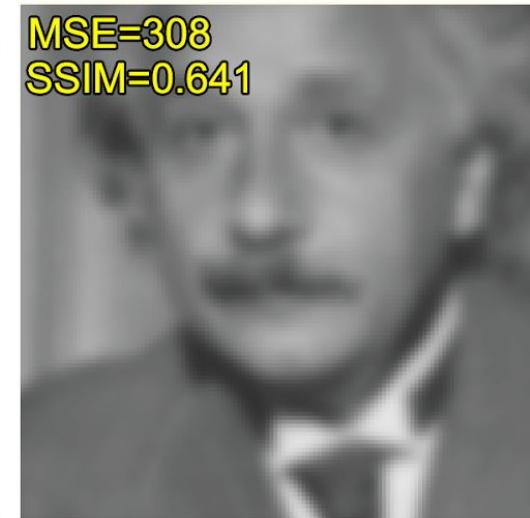
Значения MSE практически одинаковы, но визуальное качество правого изображения заметно хуже



GT



Mean-shifted



Blurred

Почти одинаковое значение MSE при размытии

# SSIM – Structure Similarity

Подходит для:

- Оценки качества сжатия
- Оценки шума и размытия

Не подходит для:

- Оценки изменений в контрасте и яркости
- Оценки поворота и смещения изображения

$$\text{SSIM}(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

Вычисляется для каждого пикселя, итоговое значение метрики — арифметическое средние по всему изображению.

# Пример уязвимости SSIM

$$\text{MSSIM} \left( \begin{array}{c|c} \text{white} & \text{white} \\ \hline 253/255 & 255/255 \end{array} \right) = 0.99997$$

$$\text{MSSIM} \left( \begin{array}{c|c} \text{gray} & \text{gray} \\ \hline 128/255 & 130/255 \end{array} \right) = 0.99988$$

$$\text{MSSIM} \left( \begin{array}{c|c} \text{black} & \text{black} \\ \hline 0 & 2/255 \end{array} \right) = 0.61914$$

$$\text{MSSIM} \left( \begin{array}{c|c} \text{light gray} & \text{white} \\ \hline 222/255 & 255/255 \end{array} \right) = 0.99047$$

$$\text{MSSIM} \left( \begin{array}{c|c} \text{black} & \text{black} \\ \hline 0/255 & 26/255 \end{array} \right) = 0.00953$$

**Малозаметные изменения в интенсивности**

$$\text{MSSIM} \left( \begin{array}{c|c} \text{white} & \text{cyan} \\ \hline \text{white} & (0.56, 1, 1) \end{array} \right) = 0.99047$$

$$\text{MSSIM} \left( \begin{array}{c|c} \text{white} & \text{pink} \\ \hline \text{white} & (1, 0.78, 1) \end{array} \right) = 0.99047$$

$$\text{MSSIM} \left( \begin{array}{c|c} \text{white} & \text{yellow} \\ \hline \text{white} & (1, 1, 0) \end{array} \right) = 0.99276$$

$$\text{SSIM} \left( \begin{array}{c|c} \text{A} & \text{B} \\ \hline \text{A} & \text{B} \end{array} \right) = 0.51$$

$$\text{SSIM} \left( \begin{array}{c|c} \text{A} & \text{B} \\ \hline \text{A} & \text{B} \end{array} \right) = -0.07$$

$$\text{SSIM} \left( \begin{array}{c|c} \text{A} & \text{B} \\ \hline \text{A} & \text{B} \end{array} \right) = -0.82$$

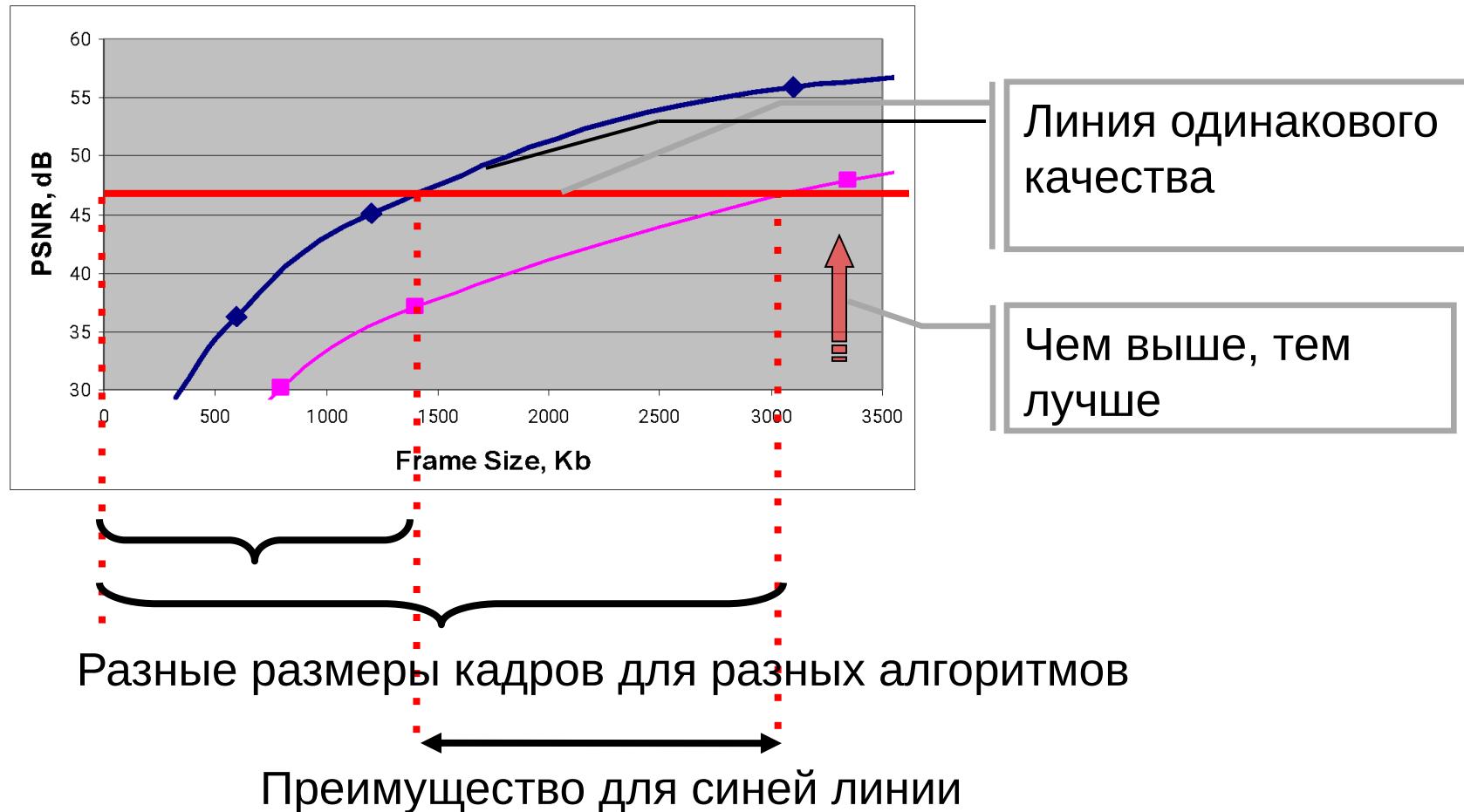
**Изменение оттенка и градиенты на разных масштабах**

256 × 256

64 × 64

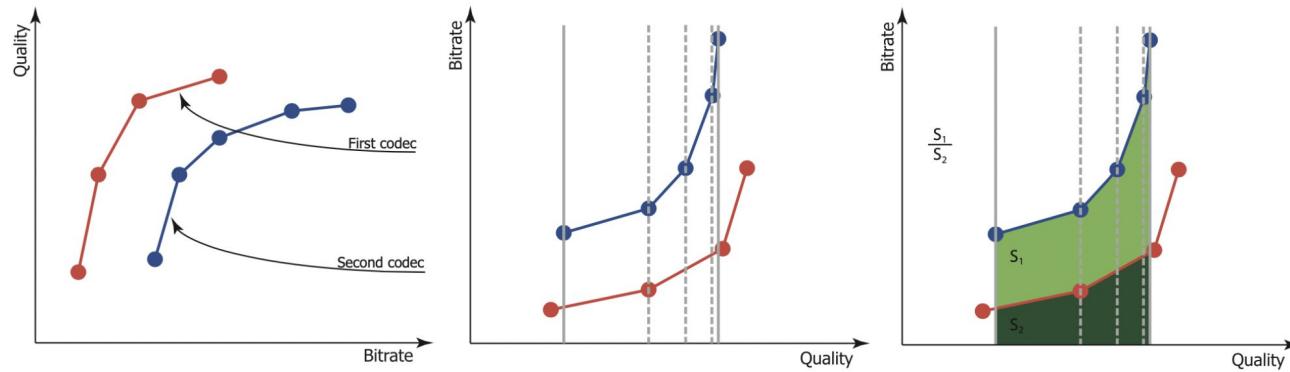
16 × 16

# Rate Distortion (RD) кривые



# Bjontegaard Delta-Rate

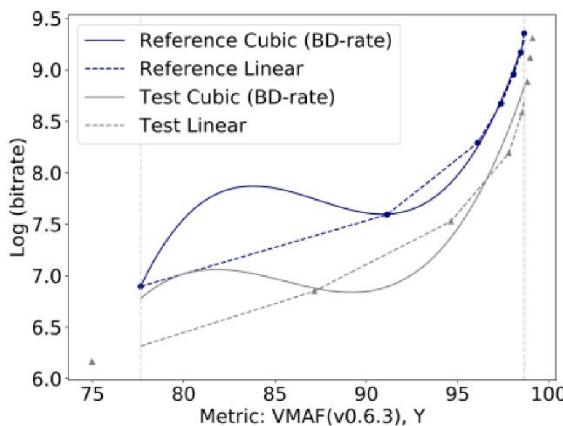
- BD-Rate позволяет измерять уменьшение битрейта при сохранении одинакового качества вычисляемого с помощью объективных метрик
- Для построение RD кривых используется кубическая интерполяция
- Вычисляется как отношение площадей под RD кривыми



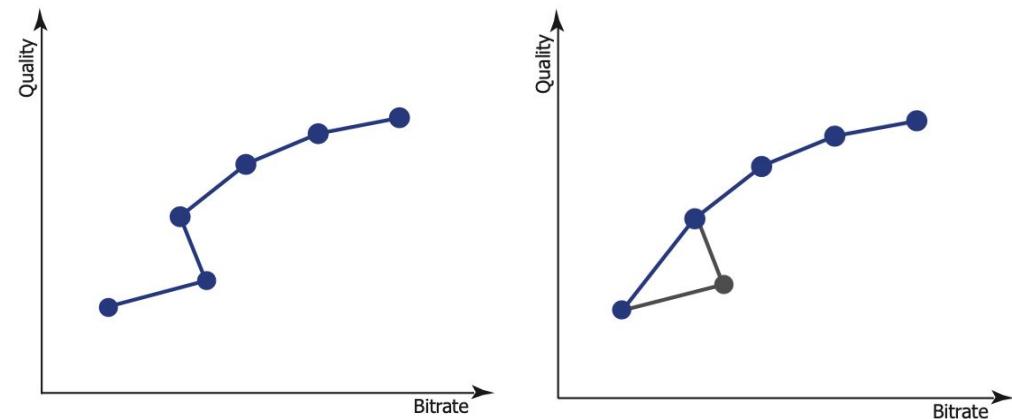
Этапы вычисления BD-rate

# Bitrate for the Same Quality

- Использует линейную интерполяцию кривых вместо кубической
- Корректно обрабатывает случай немонотонных кривых
- Позволяет проводить сравнение в условиях непересекающихся по оси качества кривых



Артефакты кубической интер.



Исправление немонотонности