

СПОСОБЫ СЖАТИЯ ГРАФИЧЕСКИХ ФАЙЛОВ




1. Основные понятия

Файл - это ограниченная область памяти с определенным именем и атрибутами.

Форматом файла называют структуру данных, записанных в этом файле.





Алгоритм сжатия — это набор инструкций, который в конечное число шагов приводит к преобразованию исходного кода цифрового изображения в код меньшего объёма за счёт устранения избыточности.

Метод — это более общее понятие, чем алгоритм. Например, метод сжатия RLE может быть реализован различными алгоритмами.

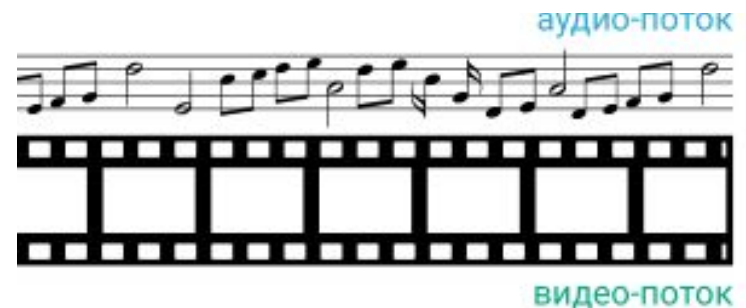
ФОРМАТ ≠ АЛГОРИТМ

Например, информация в файле PDF может записываться с использованием сразу нескольких алгоритмов сжатия для разных видов содержащихся в нем данных.

Кодек – это программа, реализующая алгоритм кодирования, в результате которого формируются потоки аудио и видео информации.

Контейнер определяет структуру файла, возможные кодеки и их параметры, состав потоков и дополнительных данных, например, субтитры.

кодировщик
↓
КО ДЕК
↑
декодировщик



Форматом видео считают комбинацию:

- контейнера,
- кодеков, которыми сжаты потоки, содержащиеся в этом контейнере,
- ключевые параметры сжатия (разрешение и битрейт).

2. Сжатие информации

Сжатие изображений — применение алгоритмов сжатия данных к изображениям, хранящимся в цифровом виде.

Алгоритмы сжатия используют такие свойства графических данных:

- Избыточность – группы одинаковых символов,
- Предсказуемость – часто повторяющиеся одинаковые символы,
- Необязательность – данные мало влияющие на человеческое восприятие (цвет).

Алгоритмы сжатия

Без потерь

(кодирование информации
меньшим числом битов без ее
искажения)

С потерями

(потеря той информации,
которая не существенна для
представления данных)

Оценки методов сжатия

- Степень сжатия
- Возможность масштабирования
- Устойчивость к ошибкам
- Учет специфики изображения
- Стоимость аппаратной реализации или/и эффективность программной реализации

2. Сжатие без потерь

RLE - кодирование длин серий

RLE (run length encoding)

Используется:

- в форматах РСХ — в качестве основного метода;
- в форматах BMP, TGA, TIFF в качестве одного из доступных.

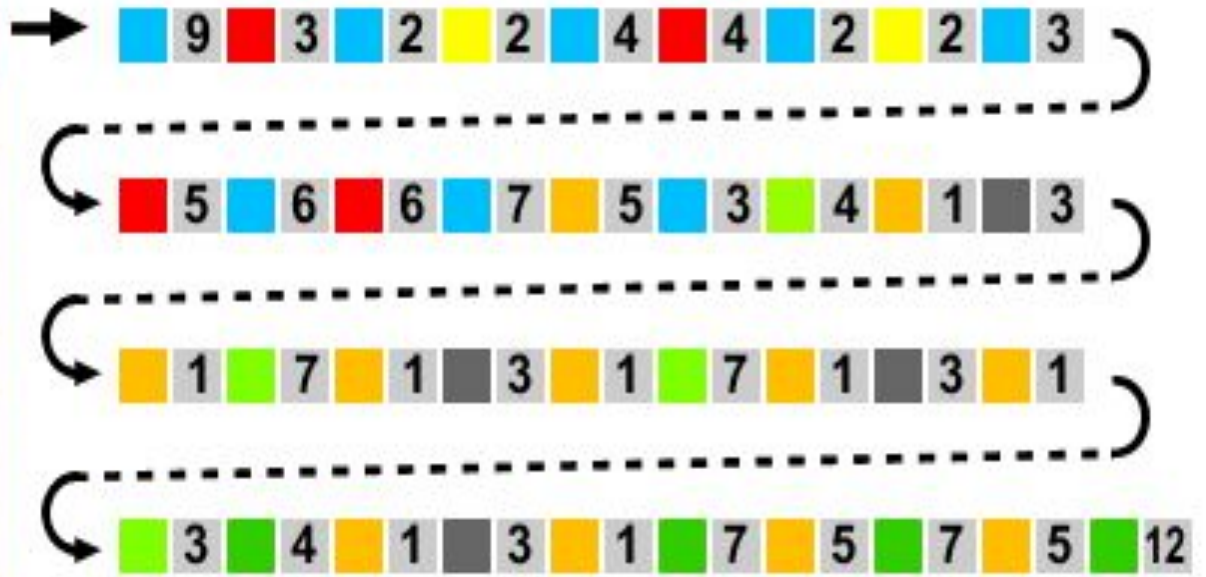
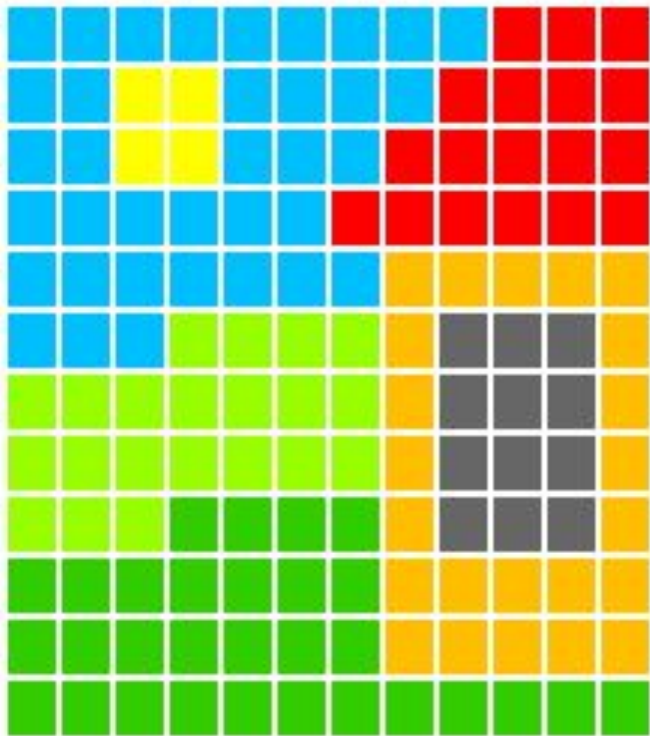
Алгоритм группового кодирования RLE

- Изображение вытягивается в цепочку байт по строкам раstra.
- Серия повторяющихся величин (значений пиксела) заменяется одной величиной и количеством ее повторений.

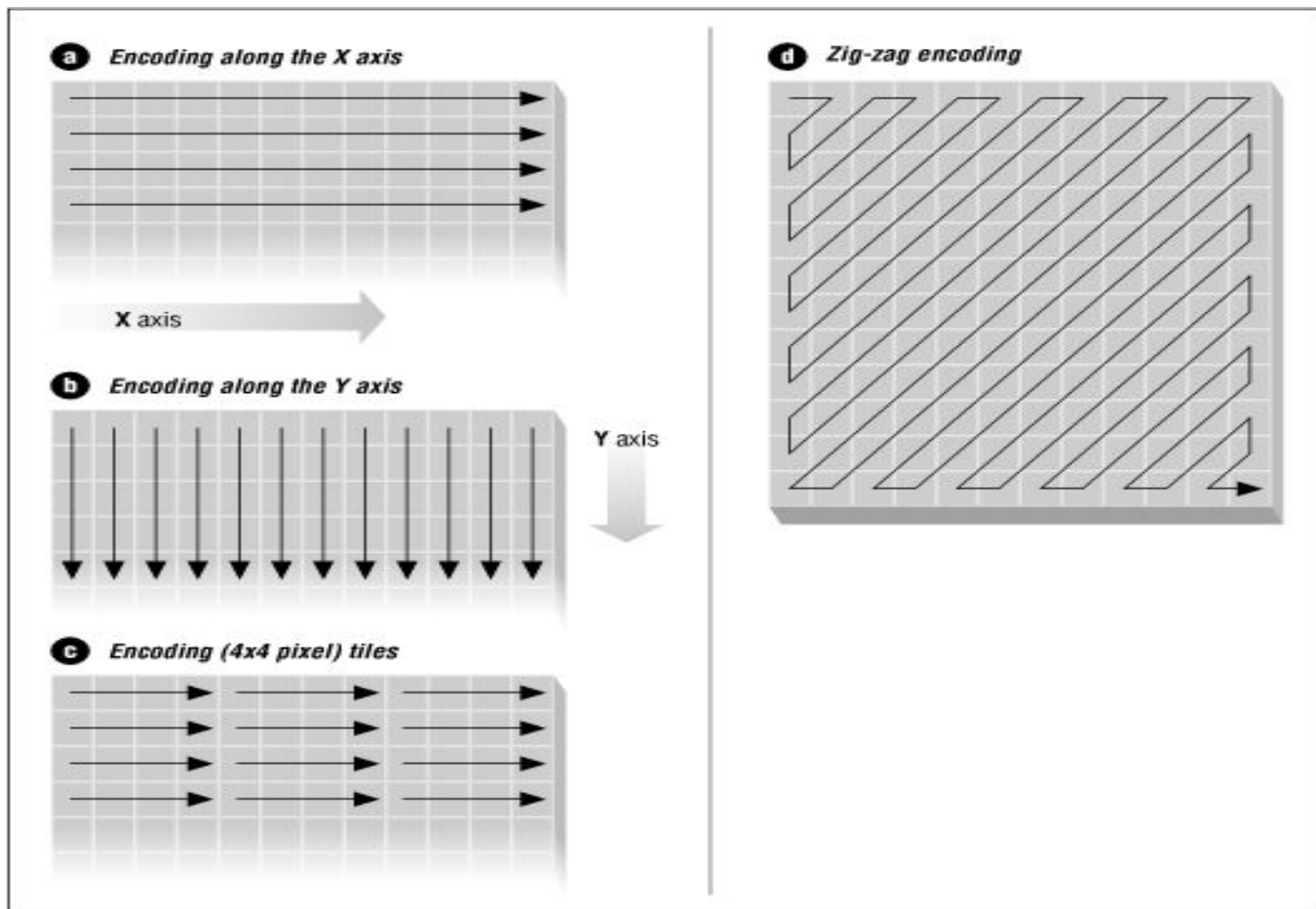
Abbbbbbbccdddeeee – 1a7b2c3d4e

- Этот подход хорошо работает с длинными сериями повторяющихся величин, т.е. с изображениями с большими областями постоянной яркости (или цвета).
- Возможные проблемы могут быть связаны с порядком записи величины и количества повторений:

1a7b2c3d4e или a1b7c2d3e4.



Способы обхода изображения в RLE-алгоритме



LZW, Lempel-Ziv-Welch

LZW реализован в форматах GIF и TIFF.



Абрахам Лемпель



Якоб Зив

LZW, Lempel-Ziv-Welch

Сжатие осуществляется за счет одинаковых цепочек байт (шаблонов).

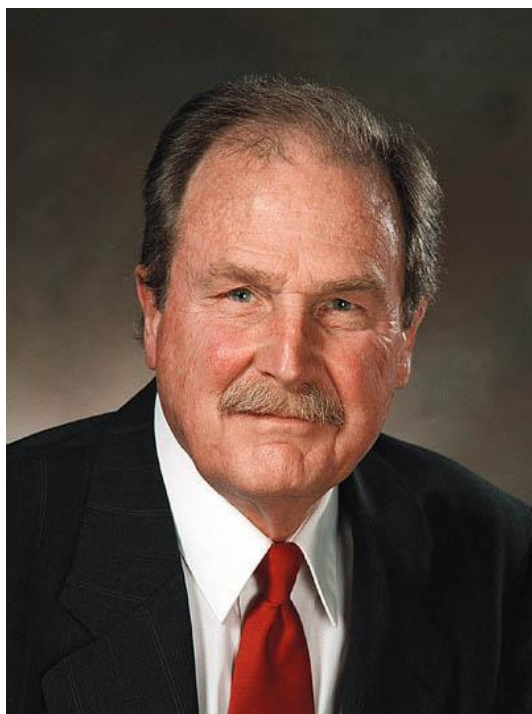
Алгоритм создает таблицу кодов, представляющих повторяющиеся пиксельные «узоры», начиная с простой таблицы и формирует более эффективную таблицу по мере своего продвижения – этот алгоритм является адаптивным.



0 1 2 0 1 2 0 1 2 0 1 2 0 1 2 3

LZW

Алгоритм Хаффмана



Дэвид Хаффман

Алгоритм

Хаффмана — адаптивный жадный алгоритм оптимального префиксного кодирования алфавита с минимальной избыточностью.

Был разработан в 1952 году аспирантом Массачусетского технологического института Дэвидом Хаффманом при написании им курсовой работы.

В настоящее время используется во многих программах сжатия

Предположим, что нам надо заархивировать следующую символьную последовательность:

AAABCCD

Эта строка занимает 7 байт.

С архивацией по методу **RLE** она бы выглядела бы так:

3,"A",1,"B",2,"C",1,"D"

то есть возросла бы до 8-ми байтов.

Алгоритм Хаффмана может сократить ее почти до **двух** байтов!.

Алгоритм Хафмана

использует частоту появления исходных байт в изображении - более короткие коды используются для часто повторяющихся величин, и наоборот. Присвоения хранятся в таблице перекодировки.

Для кодировки используются алгоритмы построения бинарных деревьев.

Требует двух проходов по изображению:

- в первый проход создается статистическая модель, т. е. каждой величине ставится в соответствие число ее повторений;
- во второй проход – кодируются данные.

Алгоритм Хаффмана

Входной алфавит:

A	B	C	D
00	01	10	11

и его двухбитовое
исходное представление

Упорядочение по
вероятности:

A	B	D	C
0.5	0.24	0.15	0.11

сканирование по
всему входному
потoku символов

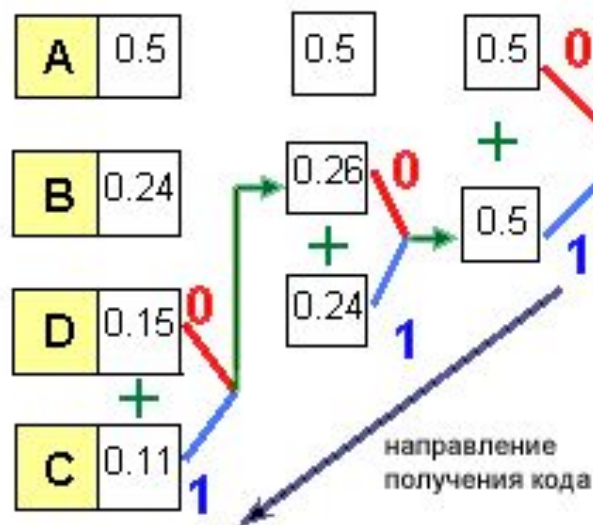
Входной поток символов:

A	B	A	A	A	B	D	A
---	---	---	---	---	---	---	---

входной поток битов (16):

0	0	0	1	0	0	0	0	0	0	1	0	1	1	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---

Построение
бинарного дерева:



Коды по Хаффману:

чем чаще встречается
код, тем он более
короткий

A	B	C	D
0	11	101	100

выходной поток битов (12):

0	1	1	0	0	0	1	1	1	0	0	0
---	---	---	---	---	---	---	---	---	---	---	---

Первая проблема состоит в том, что на разных файлах метод может генерировать различные двоичные деревья.

1 решение: в самом начале архивированного файла надо записывать само дерево с помощью которого проходила архивация

2 решение: иметь несколько "стандартных" деревьев и перед сжатым файлом давать один байт, в котором записан номер стандартной таблицы ([PkZip](#), [Lha](#), [Zoo](#), [Arj](#), ...).

В описанном алгоритме очень большая проблема связана с определением конца файла.

Как декомпрессирующая программа поймет, что файл закончен? какой маркер поставить в финале?

Первый вариант решения связан с созданием специального хедера, который записывается перед файлом и в котором указано количество бит в файле.

Второй вариант - в качестве маркера используется символ, которого не может быть в исходном файле.

Алгоритм Хаффмана



Практически не применяется непосредственно к изображениям, обычно используется как один из этапов компрессии по более сложной схеме, например, в JPEG

3. Сжатие с потерями

Алгоритм JPEG

Разработан в 1991 г. группой экспертов в области фотографии (Joint Photographic Expert Group) специально для сжатия 24-битных изображений.

Основу алгоритма составляет дискретное косинусное преобразование Фурье (DCT)

Оперирует блоками 8x8, внутри которых яркость и цвет меняются сравнительно плавно.

1. Перевод RGB в YCbCr

На первом шаге яркость записывается отдельно от цветности.

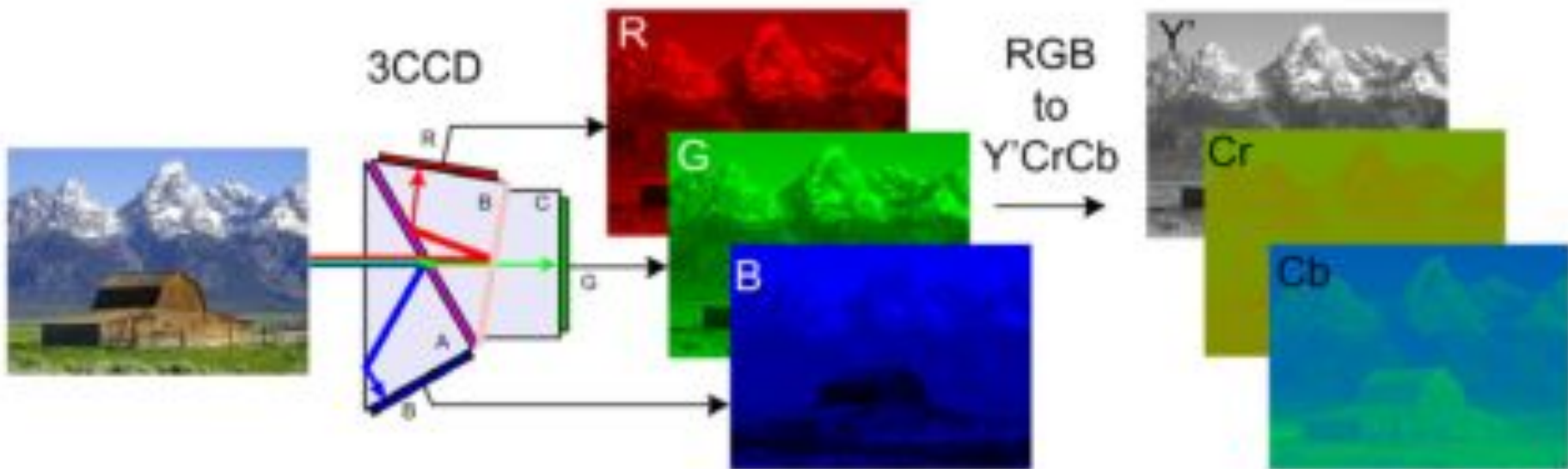
Преобразование цветовой модели RGB в модель YCbCr осуществляется с помощью следующих соотношений:

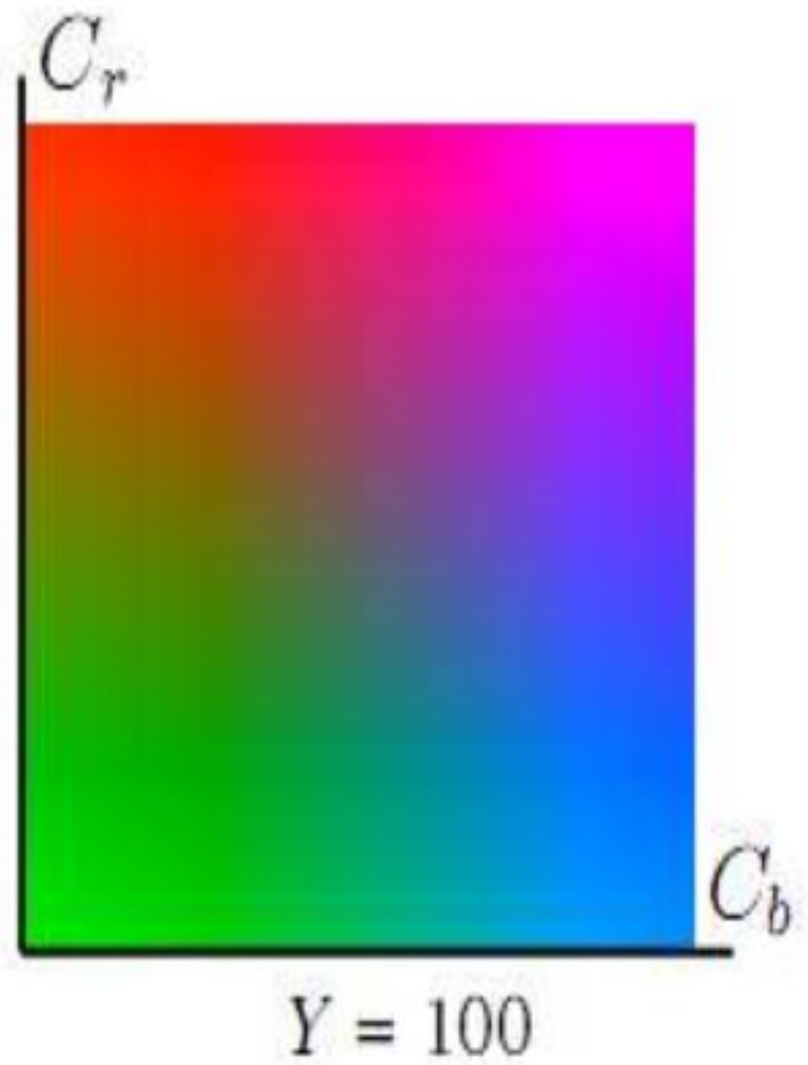
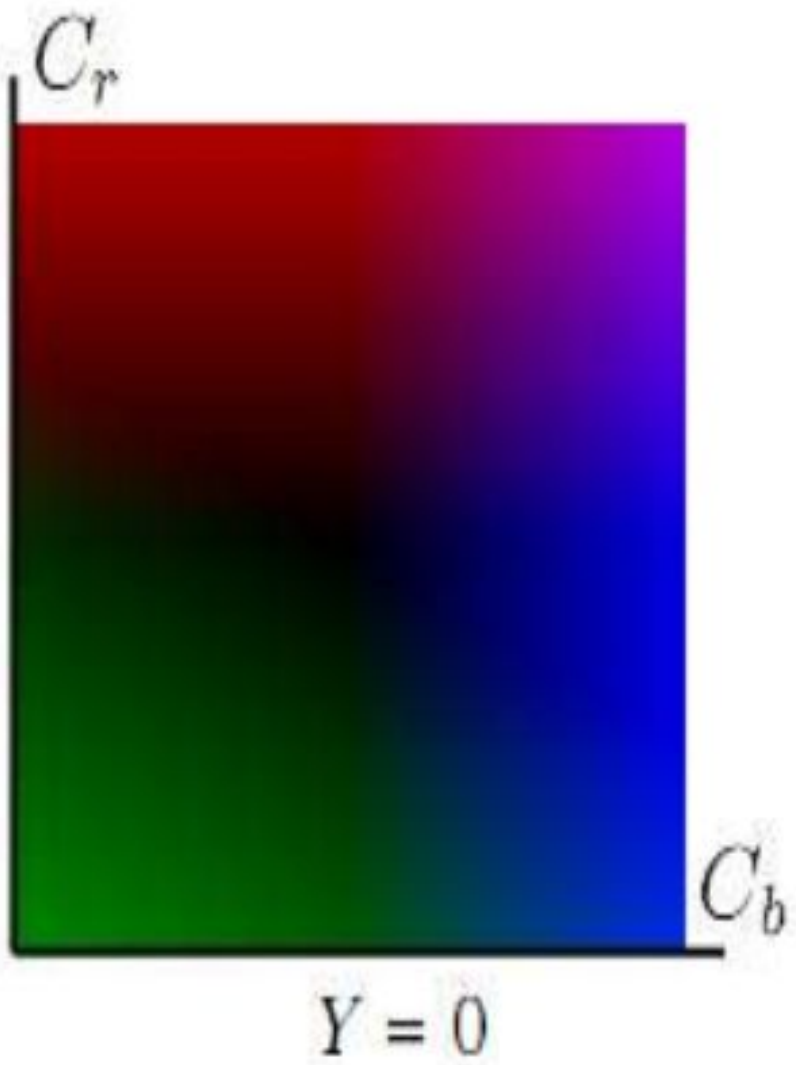
$$Y = 0,299R + 0,587G + 0,114B$$

$$Cb = -0,1687R - 0,3313G + 0,5B + 128$$

$$Cr = 0,5R - 0,4187G - 0,0813B + 128$$

При этом цветовые каналы станут значительно менее контрастными, а значит, обнаружить однородные области будет



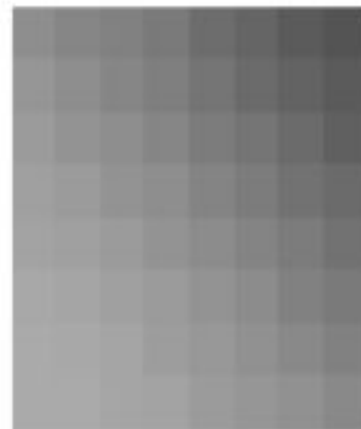


Второй шаг алгоритма JPEG

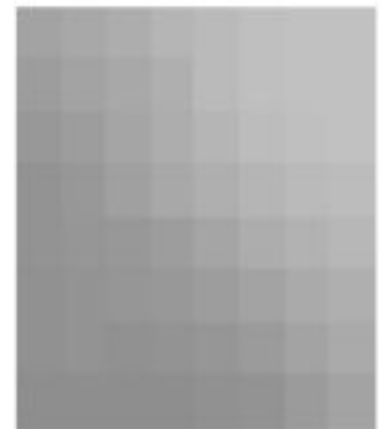
Делим изображение на блоки 8x8 пикселей и получаем матрицы 8x8 для Y , C_b и C_r



Y



C_b



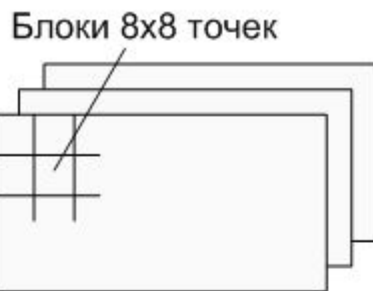
C_r

- Здесь происходит потеря, которая называется **субдискретизацией**.

- Если для канала яркости считываются все пиксели, то в цветовых каналах пиксели выбираются через одну строку и через один ряд.
- Таким образом, мы теряем $\frac{3}{4}$ полезной информации о цветовых составляющих.
- Но для всего файла получили **сжатие в два раза**



RGB -> YCrCb

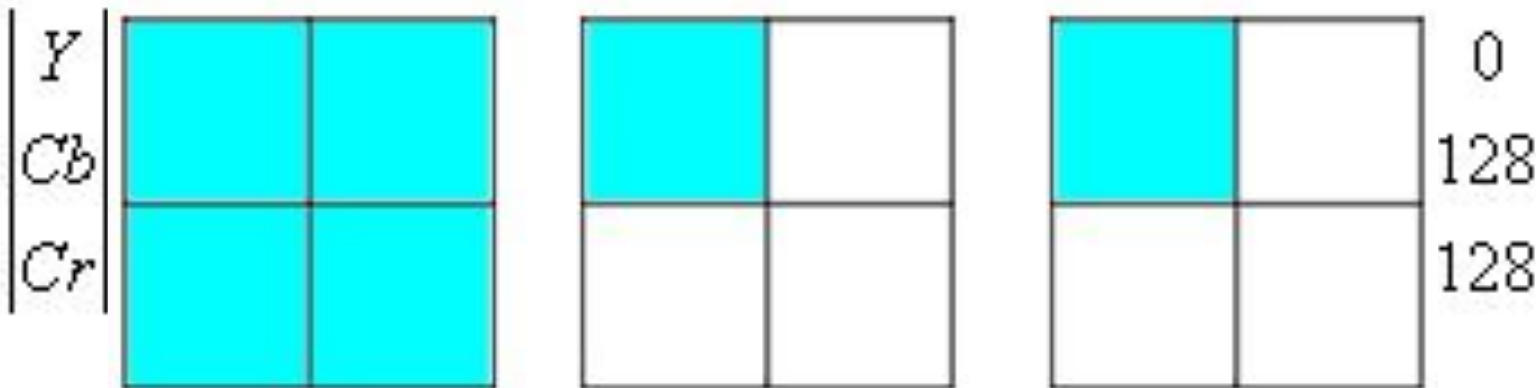


Исходное изображение

Y

U

V



3. Дискретно-косинусное преобразование (ДКП)

DCT (discrete cosine transform)

- ДКП превращает исходные данные в частоты, содержащие информацию о том, насколько быстро меняются яркость и цвет пикселей.
- ДКП отдельно применяется к цвету и яркости для прямоугольников 8×8 с нумерацией от $(0,0)$ в левом верхнем углу (низкие частоты) до $(7,7)$ в правом нижнем (высокие частоты).

Дискретное двумерное быстрое
cos преобразование Фурье,
которое отображает элементы
каждой матрицы –

числа Y_{ij} в числа F_{mn}

Прямое преобразование для Y компоненты




$$F_{u,v} = C_{u,v} \sum_{x=0}^7 \sum_{y=0}^7 Y_{x,y} \cos \left[\frac{(2x+1)u\pi}{16} \right] \\ \times \cos \left[\frac{(2y+1)v\pi}{16} \right]$$

Обратное преобразование

$$Y_{x,y} = \sum_{u=0}^7 \sum_{v=0}^7 C_{u,v} F_{u,v} \cos \left[\frac{(2x+1)u\pi}{16} \right] \\ \times \cos \left[\frac{(2y+1)v\pi}{16} \right].$$

- Понятие частоты здесь следует из рассмотрения изображения как двумерного сигнала.
- Плавные изменения значений соответствуют низкой частоте, а резкие скачки – высокой.
- Основная часть изображения содержится в низкочастотной области, а различные шумы – в высокочастотной.
- Таким образом, отделяется значимая часть изображения от мелочей.

	1	2	3	4	5	6	7	8
1	1603	203	11	45	-30	-14	-14	-7
2	108	-93	10	49	27	6	8	2
3	-42	-20	-6	16	17	9	3	2
4	56	69	7	-25	-10	-3	-2	-2
5	-33	-21	17	8	3	-4	-5	-3
6	-16	-14	8	2	-4	-2	1	1
7	0	-5	-6	-1	2	3	0	1
8	9	5	-6	-9	0	3	3	1

	- НЧ компоненты;
	- СЧ компоненты;
	- ВЧ компоненты

Четвертый шаг алгоритма JPEG

“Квантование” - т.е переход от коэффициентов Фурье к нормированным коэффициентам

$$F_{mn} \longrightarrow \frac{F_{mn}}{\alpha Q_{mn}}$$

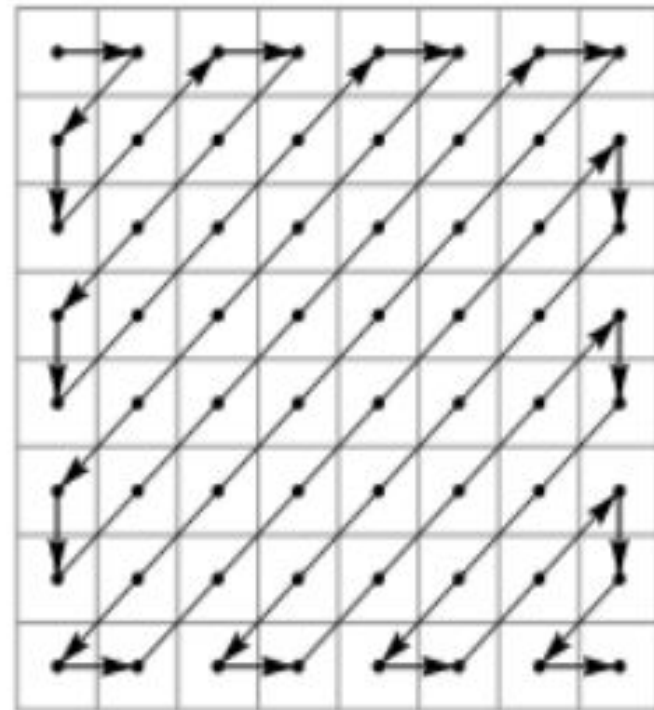
α – коэффициент сжатия, чем он больше, тем меньше и размер файла и качество изображения.

Матрица Q выбрана эмпирическим путем для Y, Cb и Cr компонент так, чтобы иметь много нулей после округления. Например, для Y

$$Q = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

5. Свертывание

- На этом шаге происходит считывание матрицы для представления значений в строку.
- При считывании таким зигзагом получаем строку из 64 значений, из которых в начале некоторое количество ненулевых, а дальше – нули.



- В результате образуются пары типа (**пропустить, число**), где **пропустить** – счетчик пропускаемых нулей, **число** – значение, которое необходимо поставить в следующую ячейку.

Например: вектор

(4 2 3 0 0 0 -2 0 0 0 0 1) ...

будет свернут в пары

(0,4) (0,3) (3,-2) (4,1) ...

- В результате образуются пары типа (**пропустить, число**), где **пропустить** – счетчик пропускаемых нулей, **число** – значение, которое необходимо поставить в следующую ячейку.

Например: вектор

(4 2 3 0 0 0 -2 0 0 0 0 1) ...

будет свернут в пары

(0,4) (0,3) (3,-2) (4,1) ...

6. Сжатие



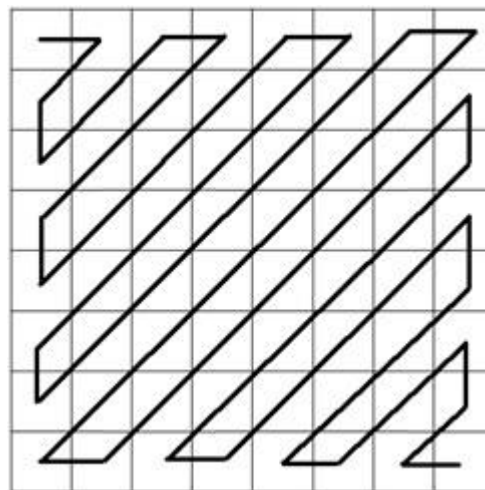
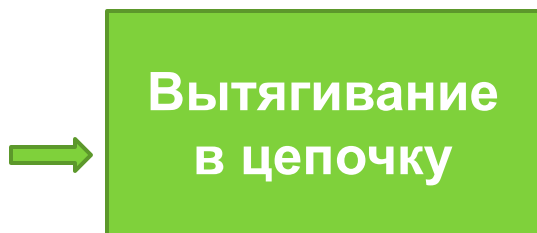
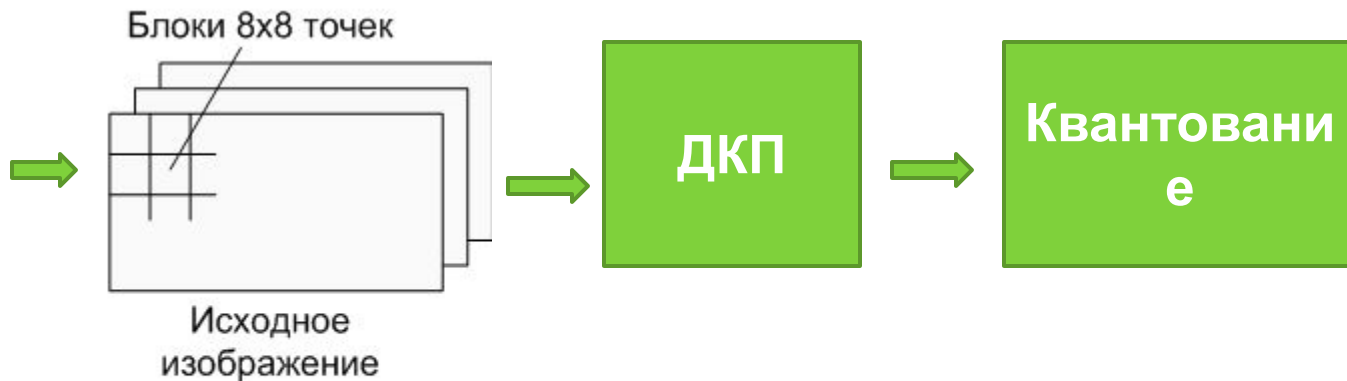
К полной последовательности чисел применяется алгоритм сжатия:

- RLE - алгоритм сжатия без потерь,
- Алгоритм Хаффмана

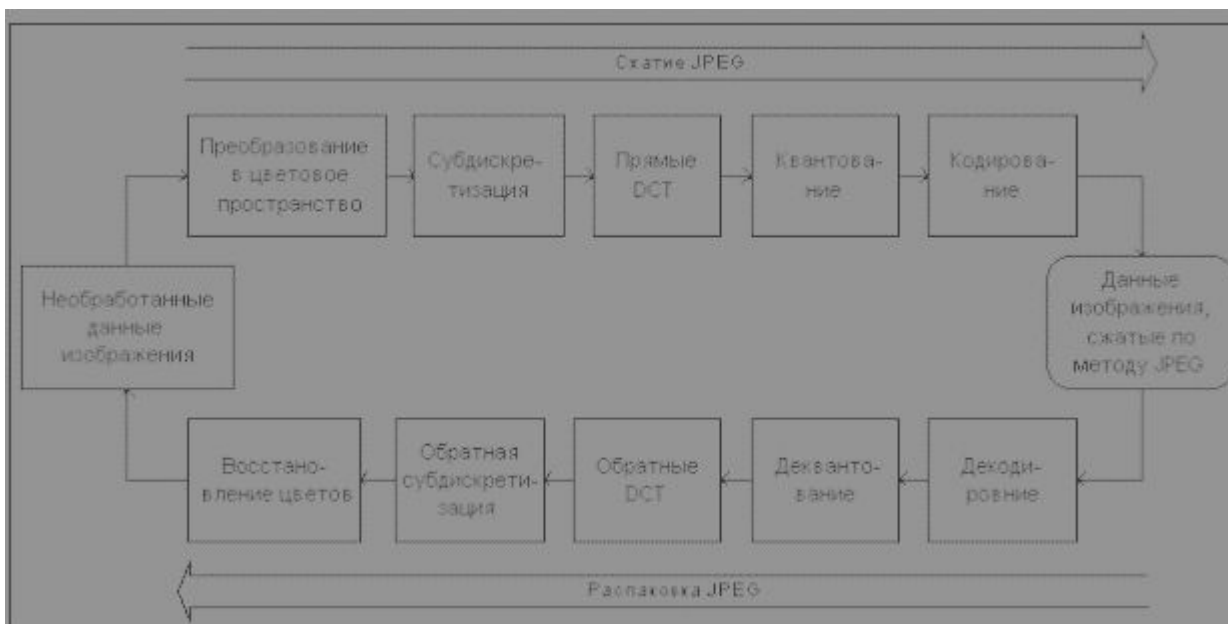
Алгоритм JPG



RGB -> YCrCb



Восстановление картинки по коду



Достоинства JPEG

- Наилучшее сжатие для фотоизображений ввиду отсутствия там резких линий (букв) и больших областей однотонной окраски.
- Стандарт де-факто для хранения, обработки и передачи фотоизображений.
- Регулируемая степень сжатия.

Недостатки JPEG

- Резкие линии после обработки по алгоритму JPEG выглядят слегка размытыми, а в однотонной окраске появляются переливы (муар).
- При больших степенях сжатия возникает эффект блочности.
- Довольно медленная программная обработка.
- Существование несовместимых реализаций из-за необязательных дополнений в стандарте.

JPEG2000

- Вместо преобразования Фурье используется **вейвлет-преобразование** (волновое, рекурсивное).
- Идея - в файл сохраняется разница – число между средними значениями соседних блоков изображения.

JPEG2000

Для всего изображения создаётся четыре матрицы половинного размера по вертикали и горизонтали:

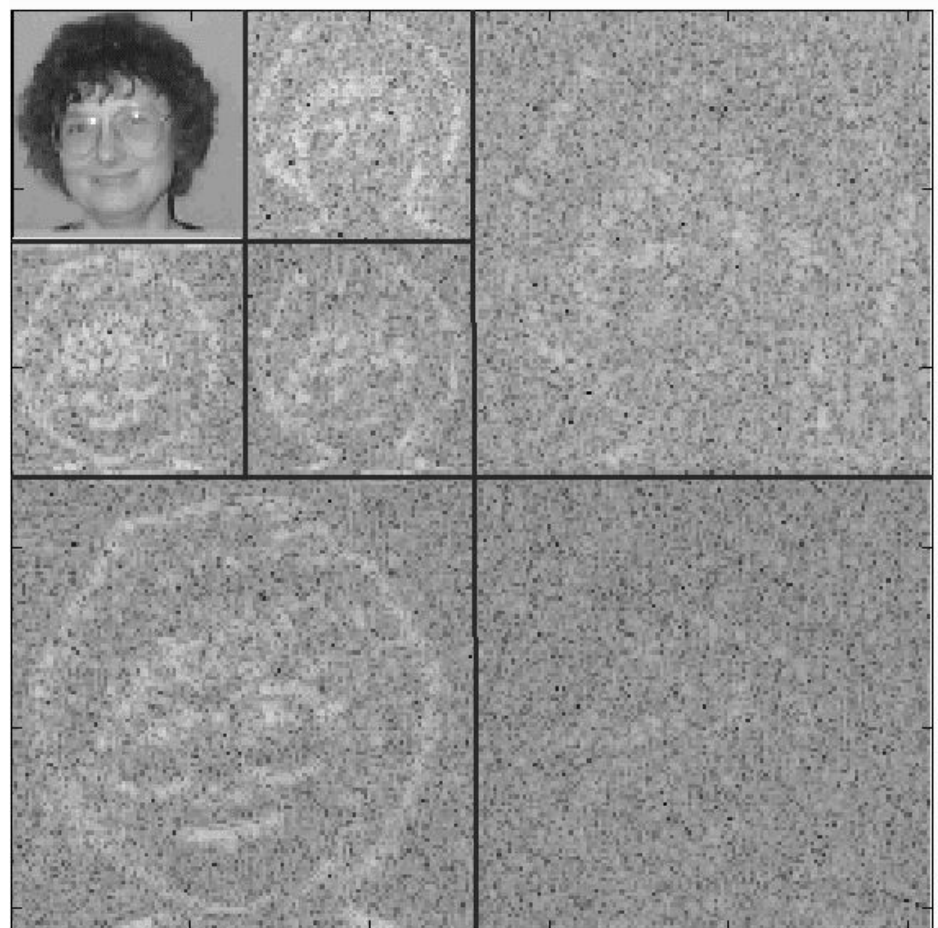
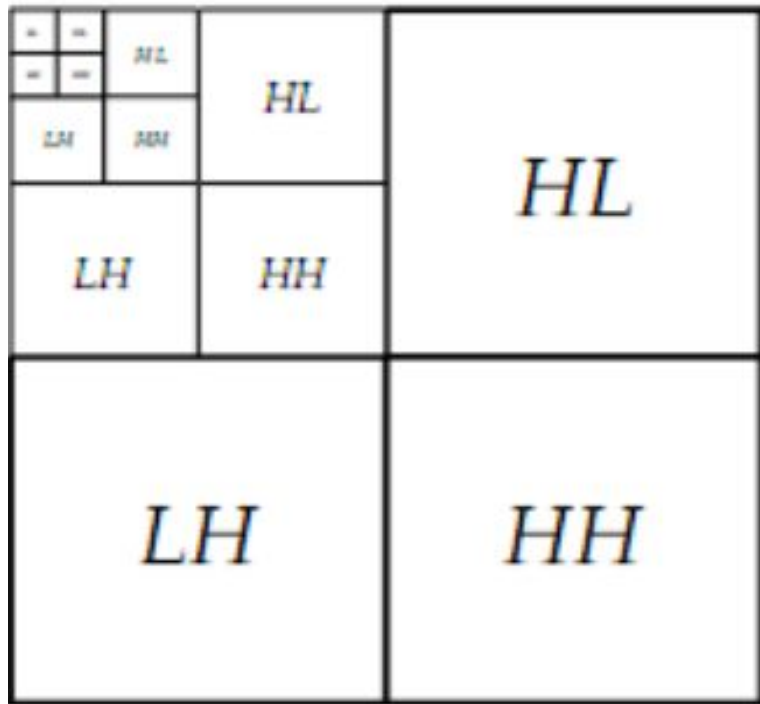
- первая хранит уменьшенное изображение,
- вторая и третья – значения разности пар пикселей по вертикали и горизонтали,
- четвёртая – усреднённую разницу значений пикселей по диагонали.

Далее операция повторяется для первой матрицы несколько

раз, в итоге мы имеем миниатюру исходного изображения и

множество матриц, описывающих разницу.

JPEG2000



Достоинства JPEG2000

- Регулируемая степень сжатия (от двух до двухсот раз)Нет дробления на блоки
- Вместо кодирования по методу Хаффмана используется более эффективное арифметическое кодирование
- Поддержка сжатия без потерь
- Поддержка сжатия 1-битных изображений
- Поддержка прозрачности при помощи отдельного канала
- Постепенное декодирование (удобно передавать по сети)
- Наличие миниатюры изображения

JPEG и JPEG2000 при сжатии в 30 раз



JPEG и JPEG2000 при сжатии в 130 раз



JPEG: сохранено больше деталей



JPEG-2000: отсутствие блочных артефактов



Альтернативы JPEG

Формат VP8

был создан Фабрисом Белларом.

Возможности:

- Высокий уровень сжатия.
- Меньший размер по сравнению с файлами JPEG аналогичного качества.
- Поддержка в большинстве браузеров, благодаря наличию декодировщика на языке JavaScript.
- Методы кодирования основаны на подмножестве стандарта сжатия видео HEVC/H.265.
- Нет потерь при преобразовании из JPEG.

Формат BPG

- Поддержка схем формирования цвета CMYK, RGB, YCgCo.
- Поддержка от 8 до 14 битов на цветовой канал;
- Наличие режима сжатия без потерь.
- Возможность интеграции в файл различных метаданных.
- Поддержка анимации небольшой длительности.



Формат WebP

Продвигается Google.

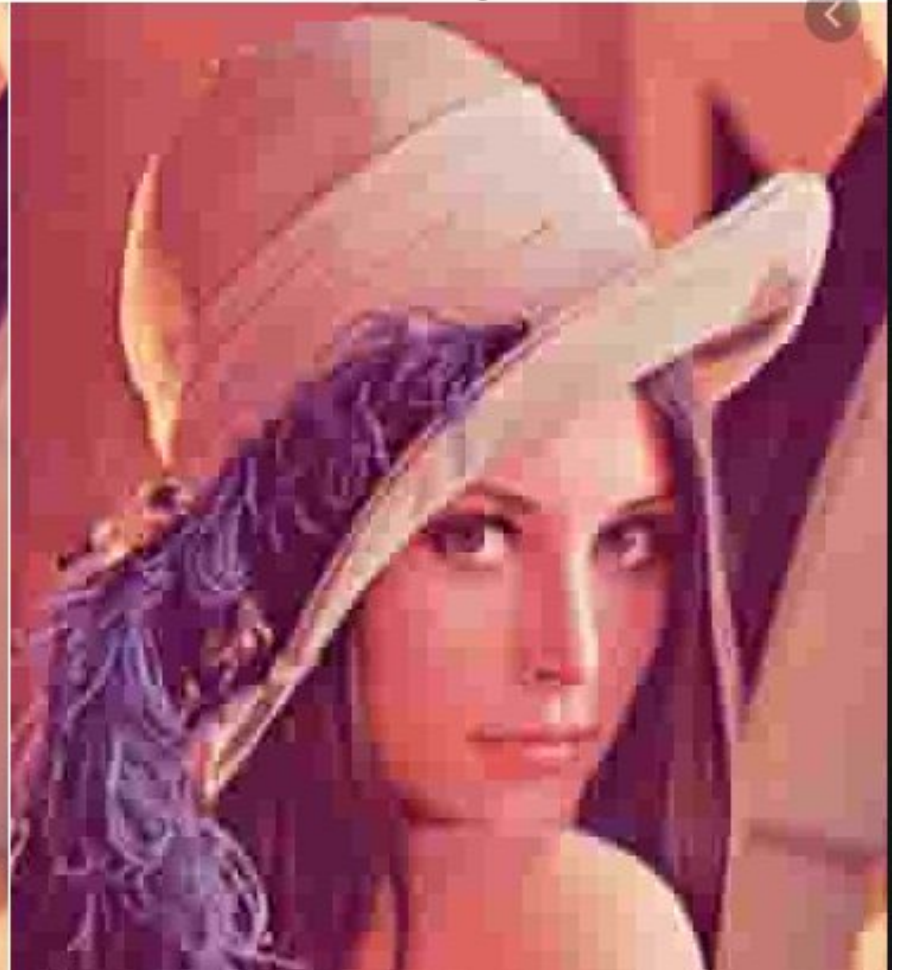
Возможности:

- Предсказание блоков, которые кодируются, исходя из уже известной информации о соседних блоках.
- Адаптивное распределение уровней квантования в соответствии со сложностью изображения.
- Сохраняется четверть цветовой информации от изначально имевшейся.
- Доступна только глубина цвета 8 бит.
- Возможно сжатие с потерями и без потерь.
- Поддержка прозрачности.
- Поддержка анимации.

BPG 5836 bytes (qp=36)



JPEG 5872 bytes



Формат GIF

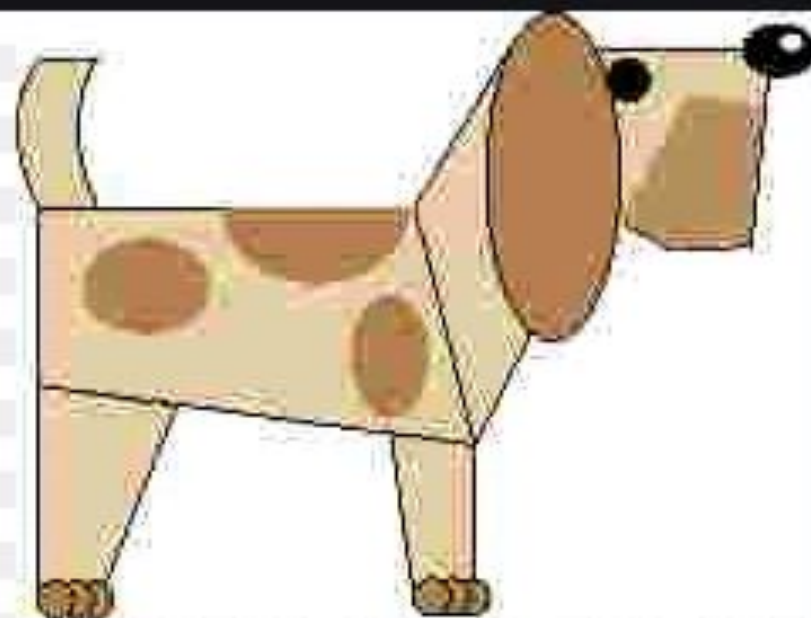
- Использует сжатие по алгоритму LZW.
- Сжатие без потерь.
- Поддержка прозрачности и анимации.
- Кодирование и декодирование производятся довольно быстро.

Недостатки:

- Построчное кодирование, что существенно ограничивает его способности находить избыточность.
- индексированная палитра 256 цветов в пределе, может быть меньше.



Формат GIF. Размер 237x180.
4 цвета (2 бита на пиксел).
Размер файла 1867 байт.



Формат JPEG. Размер 237x180.
Сжатие в 42 раза.
Размер файла 3092 байт.

Формат PNG

- Открытый, бесплатный.
- Поддержка прозрачности.
- Хранение метаданных.
- Поддержка трех режимов цветности:
 - greyscale,
 - indexed 8 bit (как в GIF),
 - полноцветный PNG-24.

Недостатки:

- Не поддерживает модель CMYK – не применим в полиграфии.
- Собственный формат анимации, плохо поддерживаемый браузерами.

JPEG



PNG



Формат PNG следует использовать

- Изображения с четкими линиями и текстом. На таких картинках пикселизация будет особенно заметна. Поэтому для них нужно использовать формат, который позволит избежать неприятного эффекта и сохранить высокий уровень сжатия файла.
- Портфолио работ. Для демонстрации высокого уровня профессионализма нужно представлять свои работы в выгодном свете. Поэтому нужно сохранить качество изображения.
- Прозрачные области изображения. В таких случаях однозначно поможет PNG формат.