ЛР1: Создание BMP-изображения

Алгоритмы цифровой обработки изображений

Изображения

- Получение: регистрация некоторого потока частиц
- Аналог => Цифра
  - Дискретизация
  - Квантование
- Растровые / Векторные
- Цветовая модель / кодирование (+сжатие)
- Фильтрация, улучшение
- Анализ и распознавание

Цель работы

- Создать программу, которая зачитывает двумерный массив чисел и сохраняет изображение в формате BMP, пиксели которого соответствуют этим числам.

Формат BMP

- Растровый формат без потерь, Microsoft
- http://msdn.microsoft.com/en-us/library/cc250370.aspx
- *Device-Independent* Bitmap (DIB) – внешний по отношению к конкретному дисплею
- Несколько версий
- Без сжатия либо простейшее RLE: сжатие последовательности одинаковых значений, грубо говоря, 2 2 2 2 2 2 2 => 7 2

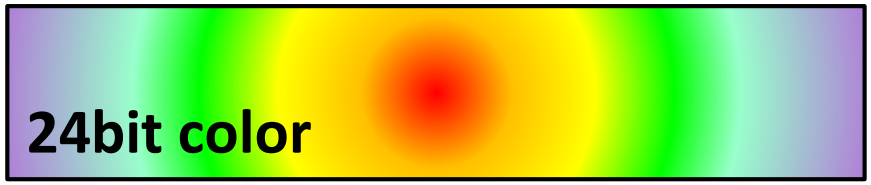
Формат входного текстового файла (*N* – ширина, *M* – высота)

Grayscale

N M 0

v_{11} v_{12} … v_{1N}
v_{21} v_{22} … v_{2N}
……
v_{M1} v_{M2} … v_{MN}

- M строк по N значений, каждое от 0 до 255
- Яркость:
  0 – черный,
  255 - белый



N M 2

r_{11} g_{11} b_{11} … r_{1N} g_{1N} b_{1N}
r_{21} g_{21} b_{21} … r_{2N} g_{2N} b_{2N}
……
r_{M1} g_{M1} b_{M1} … r_{MN} g_{MN} b_{MN}

- M строк по 3N значений, каждое от 0 до 255
- r_{ij} g_{ij} b_{ij} – компоненты цвета пикселя в i-й строке j-м столбце

Структура BMP-файла

- Заголовок BMP (14)
- Заголовок DIB (??, Разные варианты в зависимости от версий)
- Таблица цветов (опционально)
- Данные изображения

Все целые little-endian (Intel)

Структура BMP-файлов для нас

Grayscale 256

BITMAPFILEHEADER	14
BITMAPINFOHEADER	40
COLORTABLE	256x4
PIXELDATA	…

24bit Truecolor

BITMAPFILEHEADER	14
BITMAPINFOHEADER	40
PIXELDATA	…

Заголовок BITMAPFILEHEADER

- 14 байт
  - 42h 4Dh ("BM") (2)
  - Размер в байтах(4)
  - РезервА **=0** (2)
  - РезервБ **=0** (2)
  - Начало данных (4) – сдвиг массива пикселей относит начала файла

Заголовок DIB в варианте BITMAPINFOHEADER (для нас)

- 40 байт, BMPv3
  - размер заголовка **=40** (4)
  - ширина, пикс (4)
  - высота, пикс (со знаком!) (4)
  - число плоскостей цвета, **=1** (2)
  - глуб цвета, бит/пикс 1, 4, **8**, 16, **24** или 32(2)
  - сжатие **=0** для файла без сжатия (4)
  - размер данных изобр, **=0** для несж (4)
  - гориз разреш, пикс/м (4)
  - верт разреш, пикс/м (4)
  - число цветов, **=0** для 2^n , где n – глуб цв (4)
  - число важных цветов, **=0** (4)
- (другие для др версий Windows и для OS/2)

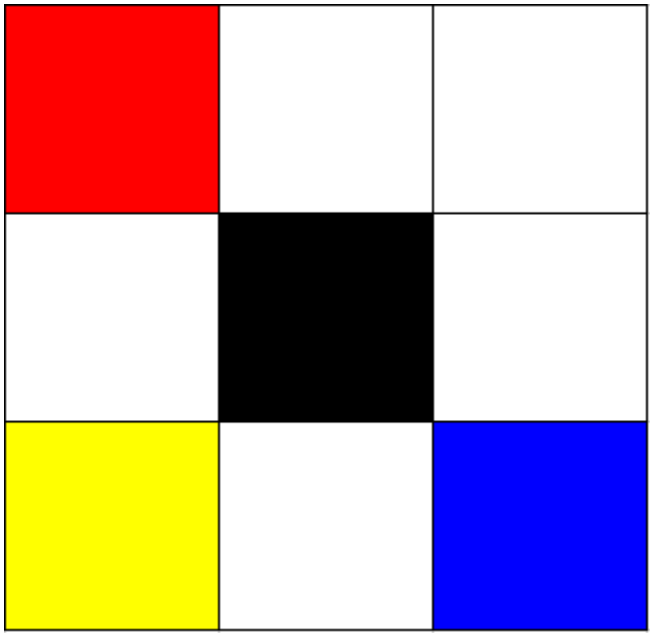
Таблица цветов

- Обязательна для <16 бит на пиксель
  - Для нас: в случае 256 градаций серого (grayscale)
- Хранит по 4 байта (b g r a) на каждый цвет для 2^n цветов, где n – глубина цвета
- Альфа-канал для нас нулевой

Данные изображения

- По строкам, каждая строка дополнена (обычно нулями) до границы в 4 байт
- Строки идут снизу вверх (или сверху вниз, если в качестве высоты в BITMAPINFOHEADER указано отрицательное число)
- В варианте 24-битного полноцветного изображения без сжатия - идет сразу за заголовками, цвета пикселей в порядке b, g, r
- Таким образом размер строки – наименьшее кратное четырем, большее или равное 3*N*

Пример: очень маленькое изображение, 3×3



90-байтный полноцветный (24-bi true color) BMP-файл

Пример

- Мы изучим байты, которыми записан этот файл
- Байты представляется записью в 16-ричном виде
- Каждый байт записывается двумя 16-ричными цифрами, т.к. байт = 8 бит = 2^8 вариантов = 16^2 вариантов
- Например, байт со значением 90 записывается как 5A, т.к. $5A_{16} = 5 \times 16 + 10$

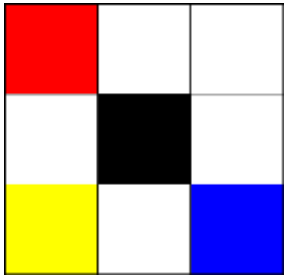
BITMAPFILEHEADER

BITMAPFILEHEADER

Размер файла 5A=90_{10}=54 + 3×12

Пикс данные начинаются в 36=54_{10}

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	0123456789ABCDEF
000	42	4D	5A	00	00	00	00	00	00	00	36	00	00	00	28	00	BMZ.....6...(.
010	00	00	03	00	00	00	03	00	00	00	01	00	18	00	00	00	..♥...♥...☺.↑..
020	00	00	24	00	00	00	00	00	00	00	00	00	00	00	00	00	..$.....ÿÿÿÿÿÿ
030	00	00	00	00	00	00	00	FF	FF				FF	00	00	00	.....ÿÿÿÿÿÿ...
040	00	00				00	00	00				00	00	00	00	00	..ÿÿÿ...ÿÿÿ
050	FF							00	00	00							ÿÿÿÿÿÿÿÿ...



BITMAPINFOHEADER

BITMAPINFOHEADER

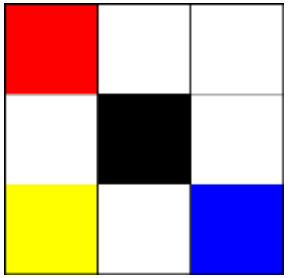
Ширина, 3 пикс

Угадайте что

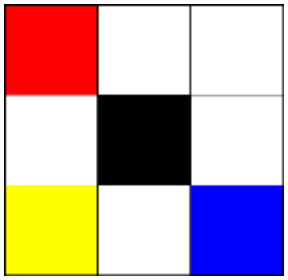
глубина цвета: $18=24_{10}$ бит на пиксель

```
    00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   0123456789ABCDEF
000 42 4D 5A 00 00 00 00 00 00 00 36 00 00 00 28 00   BMZ.....6...(.
010 00 00 03 00 00 00 03 00 00 00 01 00 18 00 00 00   ..♥...♥...☺.↑...
020 00 00 24 00 00 00 00 00 00 00 00 00 00 00 00 00   ..$.............
030 00 00 00 00 00 00 00 FF FF FF FF FF 00 00 00 00   .......ÿÿÿÿÿÿ...
040 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ..ÿÿÿ...ÿÿÿ
050 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF   ÿÿÿÿÿÿÿÿ...
```

Размер пикс данных,
3 строки по 12 байт = $36_{10} = 24_{16}$



PIXEL DATA



```
   00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   0123456789ABCDEF
000 42 4D 5A 00 00 00 00 00 00 00 36 00 00 00 28 00   BMZ.....6...(.
010 00 00 03 00 00 00 03 00 00 00 01 00 18 00 00 00   ..♥...♥..☺.↑..
020 00 00 24 00 00 00 00 00 00 00 00 00 00 00 00 00   ..$.............
030 00 00 00 00 00 00 00 FF FF FF FF FF FF 00 00 00   .......ÿÿÿÿÿÿ...
040 00 00 FF FF FF 00 00 00 FF FF FF 00 00 00 00 00   ..ÿÿÿ...ÿÿÿ
050 FF FF FF FF FF FF FF 00 00 00                     ÿÿÿÿÿÿÿ...
```

Начало пикс данных, строки снизу вверх, комп цвета в порядке B, G, R

Строка, выравненная тремя нулевыми байтами до миним кратного 4, т.е. до 12 байт

Код программы

1. Зачитать входные данные, сформировать один (для grayscale) или три (для цветного) двумерных массива
2. Сформировать заголовок BITMAPFILEHEADER
   Он выглядит как 42 4D s s s s 00 00 p p p p, где:
   - ssss – байты целого числа, равного размеру файла
   - pppp – байты целого числа, равного сдвигу данных (54 для цветного, 54+256*4 для градаций серого)
3. Сформировать заголовок BITMAPINFOHEADER
   - подсчитать размер одной строки в байтах, умножить на M, заполнить поле размер пиксельных данных

Код программы

4. Сформировать, если надо (случай grayscale) таблицу из 256 цветов вида (0 0 0 0), (1 1 1 0), (2 2 2 0) … (255 255 255 0)
5. Сформировать массив пиксельных данных. Помните, что:
   - строки идут снизу вверх
   - длина выравнивается до границы 4 байта
   - в случае true color порядок цветов B,G,R

Hints

- На Си можно пользоваться структурами из <windows.h>: `BITMAPFILEHEADER` и `BITMAPINFOHEADER`
- На Питоне можно использовать пакет `struct`. Например, чтобы сформировать массив байтов из little-endian интов 512 и 400, можно сделать:

```
import struct
a = struct.pack('<ii', 512, 400)
```

- Болванки и примеры текстовых файлов в репозитории git на https://git.io/vdFYM
- **НИКАКИМИ БИБЛИОТЕКАМИ ДЛЯ РАБОТЫ С ИЗОБРАЖЕНИЯМИ ПОЛЬЗОВАТЬСЯ НЕЛЬЗЯ!**