Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Сибирский государственный университет телекоммуникаций и
информатики»

(СибГУТИ)

Кафедра прикладной математики и кибернетики

Лабораторная работа №1
по курсу программирование графических процессов

Выполнили:
студенты группы ИП-014

Гулая А.С.
Обухов А.И.
Малышев В.А.

Работу проверил:
доцент каф. ПМиК
Перцев И.В.

Новосибирск 2024 г.

Задание:

Преобразовать цветной BMP файл с глубиной цвета 8 бит в BMP файл в оттенках серого (найти в файле палитру, преобразовать ее, усреднив по тройкам RGB цветов и записать получившийся файл под новым именем) Вывести основные характеристики BMP изображения (Работа с заголовком и палитрой).

Листинг программы:

```
#include <stdint.h>
#include <stdio.h>
#include <stdlib.h>

#pragma pack(1)
typedef struct {
  uint16_t signature;
  uint32_t filesize;
  uint32_t reserved;
  uint32_t offset;
  uint32_t header_size;
  uint32_t width;
  uint32_t height;
  uint16_t planes;
  uint16_t bpp;
  uint32_t compression;
  uint32_t image_size;
  uint32_t x_pixels_per_m;
  uint32_t y_pixels_per_m;
  uint32_t colors_used;
  uint32_t colors_important;
} Head;


typedef struct {
  Head head;
  uint8_t** rastr;
  uint8_t *palette;
} Image;


void read_head(Head *head, FILE *file) {
  fread(head, sizeof(Head), 1, file);
}


Image read_image(FILE *file) {
  Image image = { 0 };

  read_head(&image.head, file);

  if (image.head.bpp > 8) {
    fprintf(stderr, "Unsupported image depth");
    exit(-1);
  }
```

```
  fseek(file, sizeof(image.head), SEEK_SET);

  int colors_number = 1 << image.head.bpp;
  image.palette = malloc(colors_number * 4 * sizeof(uint8_t));
  fread(image.palette, sizeof(uint8_t), colors_number * 4, file);

  image.rastr = malloc(image.head.height * sizeof(uint8_t *));

  for (int i = 0; i <= image.head.height; i++) {
    image.rastr[i] = malloc(image.head.width * sizeof(uint8_t));
    for (int j = 0; j <= image.head.width; j++) {
      fread(&image.rastr[i][j], sizeof(uint8_t), 1, file);
    }
  }

  return image;

}

void print_head(Head head) {

  printf("signature: %s\n", (char *)&head.signature);
  printf("filesize: %d\n", head.filesize);
  printf("reserved: %d\n", head.reserved);
  printf("offset: %d\n", head.offset);
  printf("header_size: %d\n", head.header_size);
  printf("width: %d\n", head.width);
  printf("height: %d\n", head.height);
  printf("planes: %d\n", head.planes);
  printf("bpp: %d\n", head.bpp);
  printf("compression: %d\n", head.compression);
  printf("image_size: %d\n", head.image_size);
  printf("x_pixels_per_m: %d\n", head.x_pixels_per_m);
  printf("y_pixels_per_m: %d\n", head.y_pixels_per_m);
  printf("colors_used: %d\n", head.colors_used);
  printf("colors_important: %d\n", head.colors_important);
}


void write_image(FILE *file, Image image) {

  fwrite(&image.head, sizeof(Head), 1, file);

  if (NULL != image.palette) {
    fwrite(image.palette, sizeof(uint8_t), (1 << image.head.bpp) * 4,
file);
  }


  for (int i = 0; i < image.head.height; i++) {
    for (int j = 0; j <= image.head.width; j++) {
      fwrite(&image.rastr[i][j], sizeof(uint8_t), 1, file);
    }
  }
}


void make_black_and_white(Image *image) {
```

```
  for (int iterator = 0; iterator < (1 << image->head.bpp) * 4;
iterator += 4) {
    int average = (image->palette[iterator] + image->palette[iterator +
1] + image->palette[iterator + 2]) / 3;
    image->palette[iterator] = average;
    image->palette[iterator + 1] = average;
    image->palette[iterator + 2] = average;
    image->palette[iterator + 3] = 0;
  }
}


int main(int argc, char *argv[]) {
  if (argc <= 2) {
    fprintf(stderr, "Usage: %s <input> <output>\n", argv[0]);
    return -1;
  }

  char *filename = argv[1];

  FILE *file = fopen(filename, "rb");
  if (NULL == file) {
    perror(filename);
    return -1;
  }

  Image image = read_image(file);

  print_head(image.head);

  make_black_and_white(&image);

  write_image(fopen(argv[2], "wb"), image);

  fclose(file);
  return 0;
}
```

Результат работы:

Рис. 1 цветной BMP файл с глубиной цвета 8 бит

Рис. 2 BMP файл в оттенках серого