Работа 2. Исследование каналов и JPEG-сжатия

автор: Плохотнюк А. Д. дата: 2022-02-22T11:16:20

url: https://github.com/Gorgeousanya/ImageProcessing

Задание

1. В качестве тестового использовать изображение data/cross_0256x0256.png
2. Сохранить тестовое изображение в формате JPEG с качеством 25%.
3. Используя cv::merge и cv::split сделать "мозаику" с визуализацией каналов для исходного тестового изображения и JPEG-версии тестового изображения

- левый верхний - трехканальное изображение
- левый нижний - монохромная (черно-зеленая) визуализация канала G
- правый верхний - монохромная (черно-красная) визуализация канала R
- правый нижний - монохромная (черно-синяя) визуализация канала B

4. Результы сохранить для вставки в отчет
5. Сделать мозаику из визуализации гистограммы для исходного тестового изображения и JPEG-версии тестового изображения, сохранить для вставки в отчет.

Результаты

Рис. 1. Тестовое изображение после сохранения в формате JPEG с качеством 25%

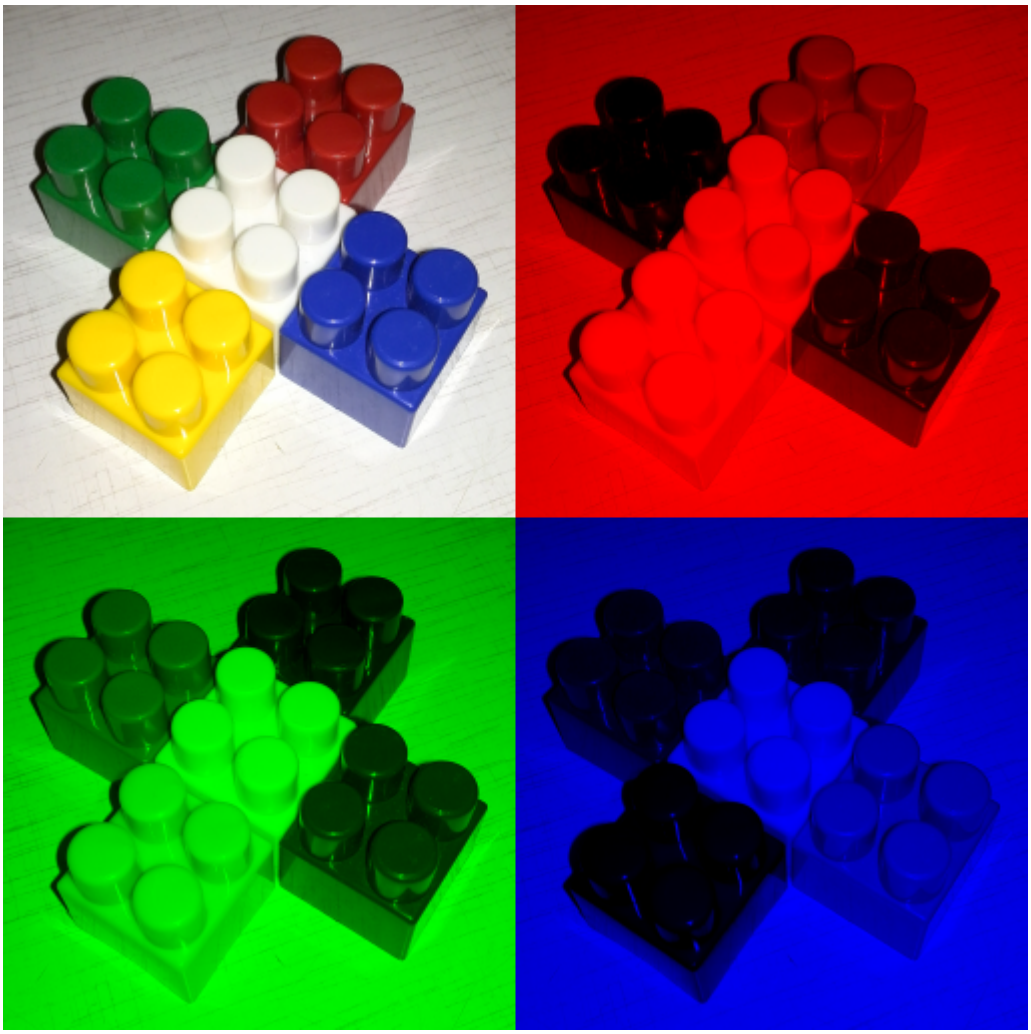

Рис. 2. Визуализация каналов исходного тестового изображения

Рис. 3. Визуализация каналов JPEG-версии тестового изображения

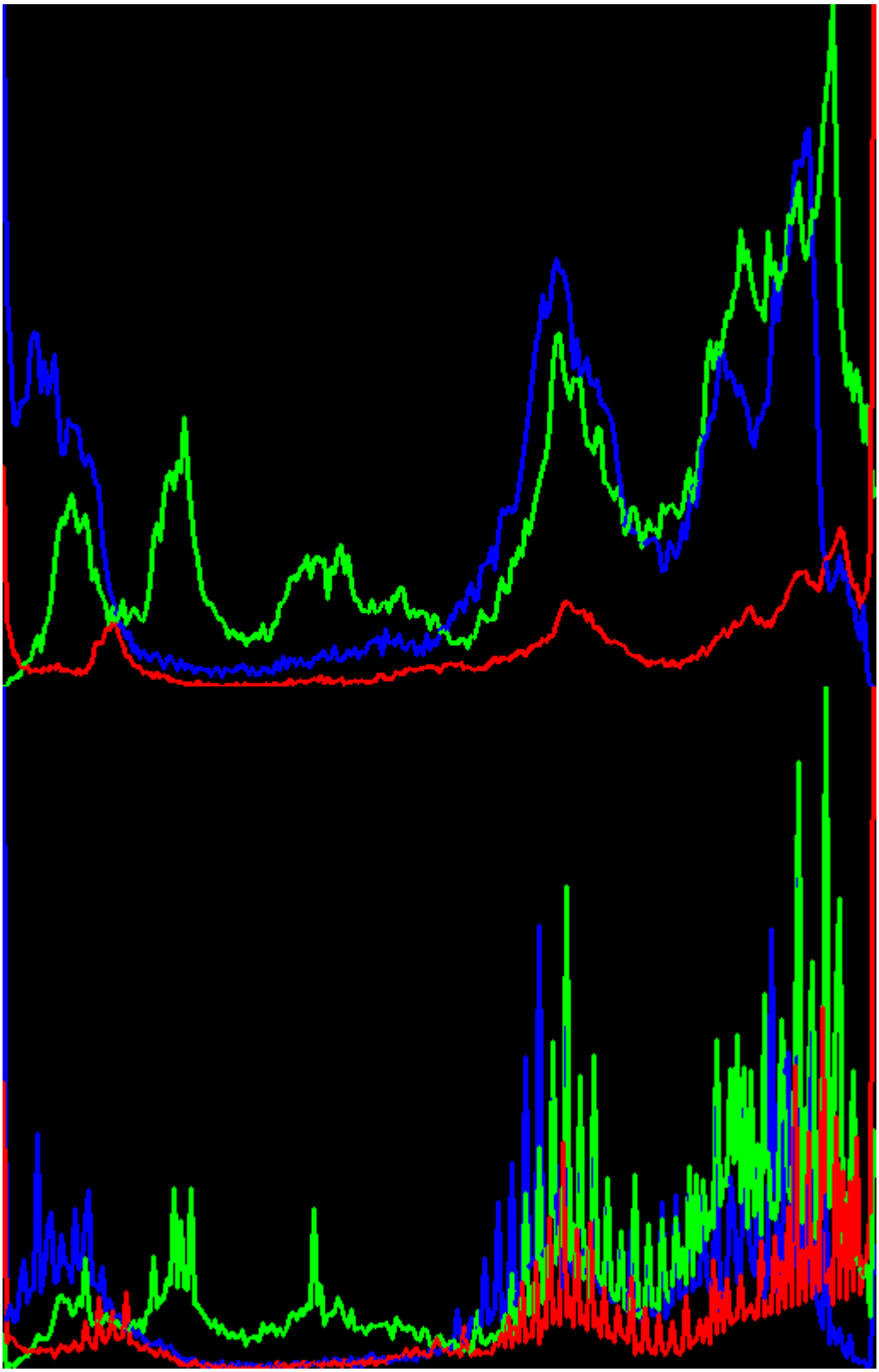

Рис. 3. Визуализация гистограм исходного и JPEG-версии тестового изображения

Текст программы

```
#include <opencv2/opencv.hpp>

cv::Mat mono(cv::Mat image, std::vector<cv::Mat> rgbChannels){
    cv::Mat g, red_img, img1, img2, img;
    g = cv::Mat::zeros(cv::Size(image.cols, image.rows), CV_8UC1);
```

```
    std::vector<cv::Mat> channels_r;
    channels_r.push_back(g);
    channels_r.push_back(g);
    channels_r.push_back(rgbChannels[2]);

    cv::merge(channels_r,  red_img);

    cv::Mat green_img;
    std::vector<cv::Mat> channels_g;
    channels_g.push_back(g);
    channels_g.push_back(rgbChannels[1]);
    channels_g.push_back(g);

    cv::merge(channels_g,  green_img);

    cv::Mat blue_img;
    std::vector<cv::Mat> channels_b;
    channels_b.push_back(rgbChannels[0]);
    channels_b.push_back(g);
    channels_b.push_back(g);

    cv::merge(channels_b,  blue_img);

    cv::hconcat(image, red_img, img1);
    cv::hconcat(green_img, blue_img, img2);
    cv::vconcat(img1, img2, img);
    return img;
}

cv::Mat histogram(cv::Mat image, std::vector<cv::Mat> rgbChannels){
    int histSize = 256;
    float range[] = { 0, 256 };
    const float* histRange[] = { range };
    bool uniform = true, accumulate = false;
    cv::Mat b_hist, g_hist, r_hist;
    calcHist( &rgbChannels[0], 1, 0, cv::Mat(), b_hist, 1, &histSize,
histRange, uniform, accumulate );
    calcHist( &rgbChannels[1], 1, 0, cv::Mat(), g_hist, 1, &histSize,
histRange, uniform, accumulate );
    calcHist( &rgbChannels[2], 1, 0, cv::Mat(), r_hist, 1, &histSize,
histRange, uniform, accumulate );
    int hist_w = 512, hist_h = 400;
    int bin_w = cvRound( (double) hist_w/histSize );
    cv::Mat histImage( hist_h, hist_w, CV_8UC3, cv::Scalar( 0,0,0) );
    normalize(b_hist, b_hist, 0, histImage.rows, cv::NORM_MINMAX, -1,
cv::Mat() );
    normalize(g_hist, g_hist, 0, histImage.rows, cv::NORM_MINMAX, -1,
cv::Mat() );
    normalize(r_hist, r_hist, 0, histImage.rows, cv::NORM_MINMAX, -1,
cv::Mat() );
    for( int i = 1; i < histSize; i++ )
    {
        line( histImage, cv::Point( bin_w*(i-1), hist_h -
```

```
cvRound(b_hist.at<float>(i-1)) ),
              cv::Point( bin_w*(i), hist_h - cvRound(b_hist.at<float>(i))
),
              cv::Scalar( 255, 0, 0), 2, 8, 0  );
        line( histImage, cv::Point( bin_w*(i-1), hist_h -
cvRound(g_hist.at<float>(i-1)) ),
              cv::Point( bin_w*(i), hist_h - cvRound(g_hist.at<float>(i))
),
              cv::Scalar( 0, 255, 0), 2, 8, 0  );
        line( histImage, cv::Point( bin_w*(i-1), hist_h -
cvRound(r_hist.at<float>(i-1)) ),
              cv::Point( bin_w*(i), hist_h - cvRound(r_hist.at<float>(i))
),
              cv::Scalar( 0, 0, 255), 2, 8, 0  );
    }
    return  histImage ;
}

int main() {
    cv::Mat img = cv::imread(
"/Users/annaplokhotnyuk/Downloads/plokhotnyuk_a_d/data/cross_0256x0256.png
");
    cv::imwrite("cross_0256x0256_025.jpg", img, {cv::IMWRITE_JPEG_QUALITY,
25});
    cv::Mat img_25 =
cv::imread("/Users/annaplokhotnyuk/Downloads/plokhotnyuk_a_d/bin.dbg/cross
_0256x0256_025.jpg");
    std::vector<cv::Mat> rgbChannels(3), rgbChannels_25;
    split(img, rgbChannels);
    split(img_25, rgbChannels_25);
    cv::Mat mono_img= mono(img, rgbChannels);
    cv::imwrite("cross_0256x0256_png_channels.png", mono_img);
    cv::Mat mono_img_25= mono(img_25, rgbChannels_25);
    cv::imwrite("cross_0256x0256_jpg_channels.png", mono_img_25);
    cv::Mat histo_image = histogram(img, rgbChannels);
    cv::Mat histo_image_25 = histogram(img_25, rgbChannels_25);
    cv::Mat hists;
    cv::vconcat(histo_image, histo_image_25, hists);
    cv::imwrite("cross_0256x0256_hists.png", hists);

}
```