Работа 2. Исследование каналов и JPEG-сжатия

автор: Панин Г. И. дата: 2022-02-22T22:43:24

url: https://github.com/ssimpletonn/panin-g-i/tree/main/prj.labs/lab01

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

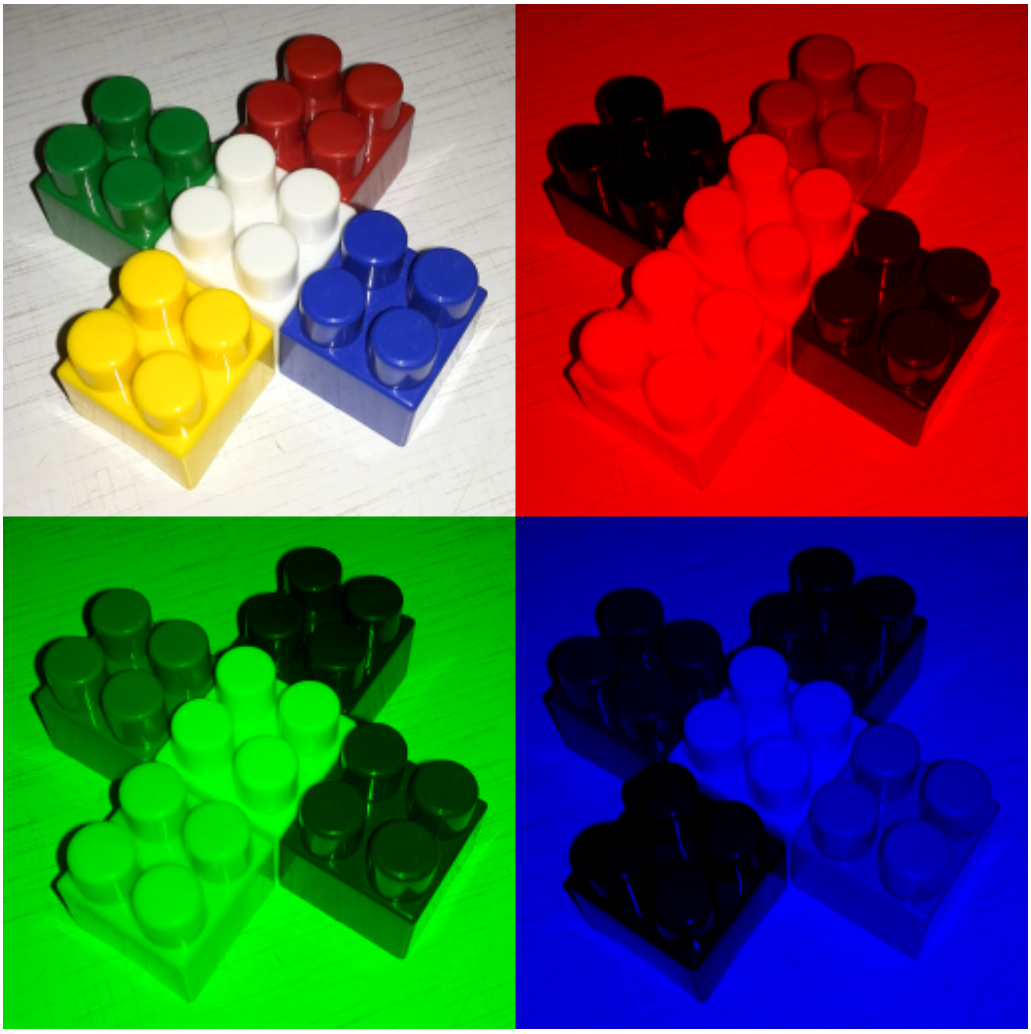
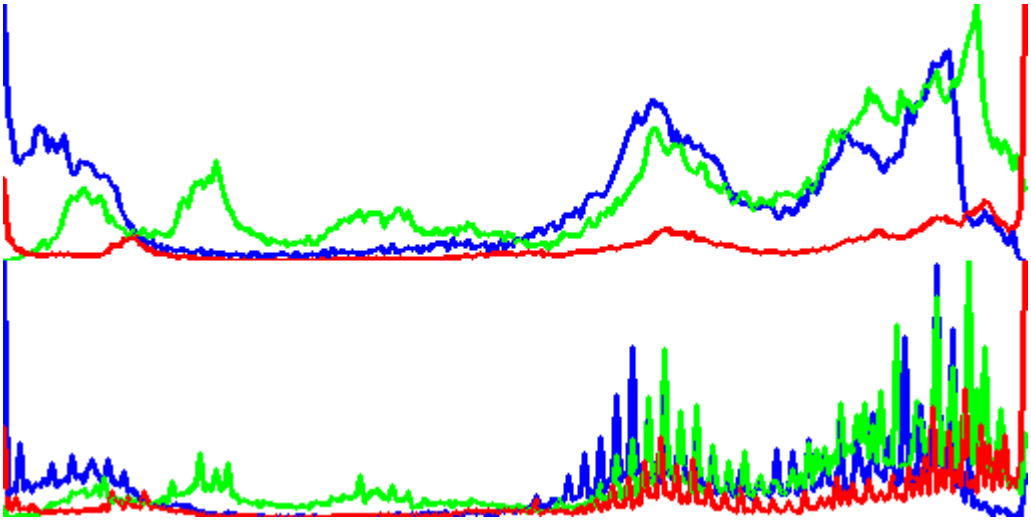


Рис. 3. Визуализация каналов JPEG-версии тестового изображения

Рис. 4. Визуализация гистограм исходного и JPEG-версии тестового изображения



Текст программы

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

cv::Mat mosaic(cv::Mat& img) {
    cv::Mat mosaic(512, 512, CV_8UC3);
```

```
    cv::Mat temp;
    mosaic = 0;
    cv::Rect2d rec = {0, 0, 256, 256};

    std::vector<cv::Mat> ch(3);
    cv::split(img, ch);
    cv::Mat zeros = cv::Mat::zeros(img.rows, img.cols, CV_8UC1);

    std::vector<cv::Mat> B = {ch[0], zeros, zeros};
    std::vector<cv::Mat> G = {zeros, ch[1], zeros};
    std::vector<cv::Mat> R = {zeros, zeros, ch[2]};

    img.copyTo(mosaic(rec));
    rec.y += 256;
    cv::merge(G, temp);
    temp.copyTo(mosaic(rec));
    rec.x += 256;
    cv::merge(B, temp);
    temp.copyTo(mosaic(rec));
    rec.y -= 256;
    cv::merge(R, temp);
    temp.copyTo(mosaic(rec));

    return mosaic;
}

cv::Mat histogram(cv::Mat& img) {
    std::vector<cv::Mat> ch;
    cv::split(img, ch);

    int histSize = 256;
    float range[] = {0, 256};
    const float* histRange[] = { range };
    bool uniform = true, accumulate = false;

    cv::Mat b_hist, g_hist, r_hist;
    cv::calcHist(&ch[0], 1, 0, cv::Mat(), b_hist, 1, &histSize, histRange,
uniform, accumulate);
    cv::calcHist(&ch[1], 1, 0, cv::Mat(), g_hist, 1, &histSize, histRange,
uniform, accumulate);
    cv::calcHist(&ch[2], 1, 0, cv::Mat(), r_hist, 1, &histSize, histRange,
uniform, accumulate);

    int hist_w = 512, hist_h = 128;
    int bin_w = cvRound( (double) hist_w/histSize);
    cv::Mat histImage( hist_h, hist_w, CV_8UC3, cv::Scalar( 255,255,255) );
    cv::normalize(b_hist, b_hist, 0, histImage.rows, cv::NORM_MINMAX, -1,
cv::Mat());
    cv::normalize(g_hist, g_hist, 0, histImage.rows, cv::NORM_MINMAX, -1,
cv::Mat());
    cv::normalize(r_hist, r_hist, 0, histImage.rows, cv::NORM_MINMAX, -1,
cv::Mat());

    for(int i = 1; i < histSize; i++)
```

```
    {
        cv::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);
        cv::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);
        cv::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("../data/cross_0256x0256.png");

    //JPEG качество 25%
    std::vector<int> p = {cv::IMWRITE_JPEG_QUALITY, 25};
    cv::imwrite("cross_0256x0256_025.jpeg", img, p);
    cv::Mat img1 = cv::imread("cross_0256x0256_025.jpeg");

    //Мозаика png и jpeg
    cv::Mat mosaic1 = mosaic(img);
    cv::Mat mosaic2 = mosaic(img1);

    cv::imshow("m1", mosaic1);
    cv::imshow("m2", mosaic2);

    cv::imwrite("cross_0256x0256_png_channels.png", mosaic1);
    cv::imwrite("cross_0256x0256_jpg_channels.png", mosaic2);


    //Гистограммы
    cv::Mat histograms;
    histograms = 0;

    cv::Mat hist1 = histogram(img);
    cv::Mat hist2 = histogram(img1);

    cv::vconcat(hist1, hist2, histograms);
    cv::imshow("histograms", histograms);
    cv::imwrite("cross_0256x0256_hists.png", histograms);

    cv::waitKey(0);
}
```