Работа 3. Яркостные преобразования

автор: Егубова П.А.
дата: 2021-03-04T14:00:19

url: https://mysvn.ru/egubova_p_a/egubova_p_a/prj.labs/lab03/

Задание

1. В качестве тестового использовать изображение data/cross_0256x0256.png
2. Сгенерировать нетривиальную новую функцию преобразования яркости (не стоит использовать линейную функцию, гамму, случайная).
3. Сгенерировать визуализацию функции преобразования яркости в виде изображения размером 512x512, черные точки на белом фоне.
4. Преобразовать пиксели grayscale версии тестового изображения при помощи LUT для сгенерированной функции преобразования.
5. Преобразовать пиксели каждого канала тестового изображения при помощи LUT для сгенерированной функции преобразования.
6. Результы сохранить для вставки в отчет.

Результаты

Рис. 1. Исходное тестовое изображение

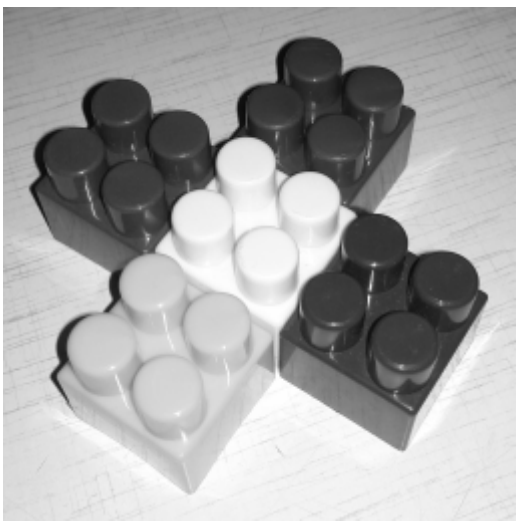

Рис. 2. Тестовое изображение greyscale

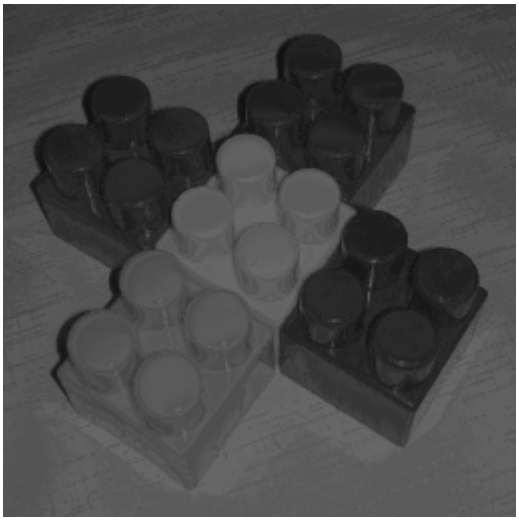

Рис. 3. Результат применения функции преобразования яркости для greyscale

Рис. 4. Результат применения функции преобразования яркости для каналов

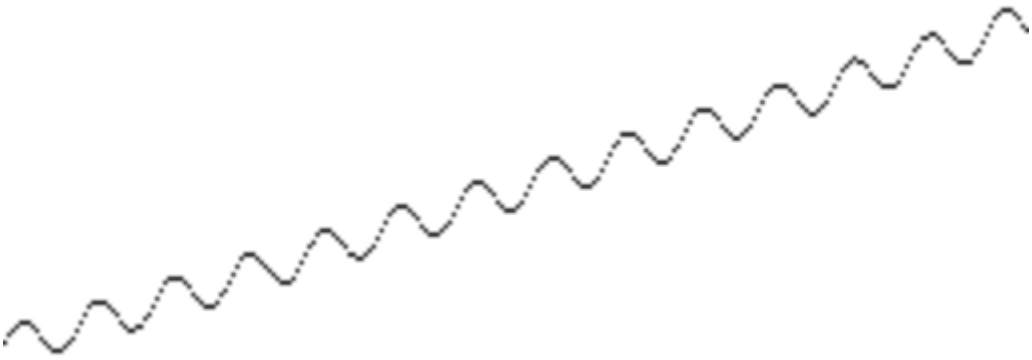

Рис. 5. Визуализация функции яркостного преобразования

Текст программы

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

using namespace std;
using namespace cv;

double bright_func(double brightness) {
    return 5 * cos(30 + brightness / 3) + brightness / 3 + 1000 / (50 +
sqrt(brightness));
}

int main() {
    Mat img_rgb = imread("cross_0256x0256.png");
    imwrite("lab03_rgb.png", img_rgb);
    Mat img_gre = Mat(img_rgb.cols, img_rgb.rows, CV_8U);
    cvtColor(img_rgb, img_gre, COLOR_BGR2GRAY);
    imwrite("lab03_gre.png", img_gre);

    Mat viz_func_256 = Mat::zeros(256, 256, CV_8U);
    viz_func_256.setTo(Scalar(255, 255, 255));
    for (int i = 0; i < viz_func_256.cols; ++i) {
        viz_func_256.at<uchar>(static_cast<int>(viz_func_256.rows -
bright_func(i)), i) = 0;
    }
    Mat viz_func_512;
```

```
    resize(viz_func_256, viz_func_512, Size(), 2, 2);
    imwrite("lab03_viz_func.png", viz_func_512);

    Mat lut(1, 256, CV_8U);
    for (int i = 0; i < 256; ++i) {
        lut.at<uchar>(0, i) = static_cast<int>(bright_func(i));
    }

    Mat img_rgb_res = img_rgb.clone();
    LUT(img_rgb, lut, img_rgb_res);
    imwrite("lab03_rgb_res.png", img_rgb_res);

    Mat img_gre_res = img_gre.clone();
    LUT(img_gre, lut, img_gre_res);
    imwrite("lab03_gre_res.png", img_gre_res);

    return 0;
}
```