Работа 3. Яркостные преобразования

автор: Котелевскйи А.А. дата: 2022-03-14T19:49:24

url: https://github.com/artemiskot/Methods-and-means-of-image-processing/tree/main/prj.labs/lab03

Задание

1. В качестве тестового использовать изображение data/cross_0256x0256.png
2. Сгенерировать нетривиальную новую (не стоит использовать) функцию преобразования яркости.
3. Сгенерировать визуализацию функцию преобразования яркости в виде изображения размером 512x512, черные точки а 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

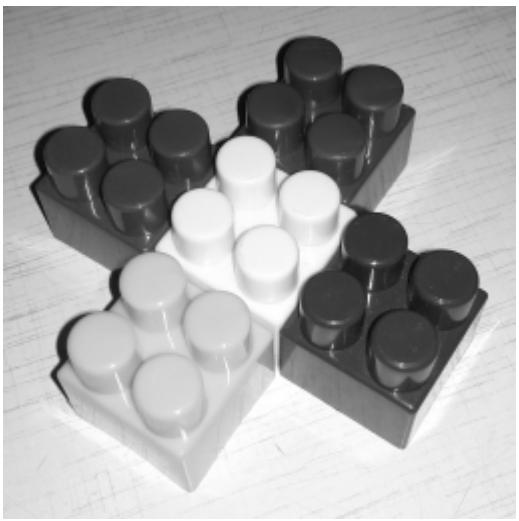
Рис. 2. Тестовое изображение greyscale

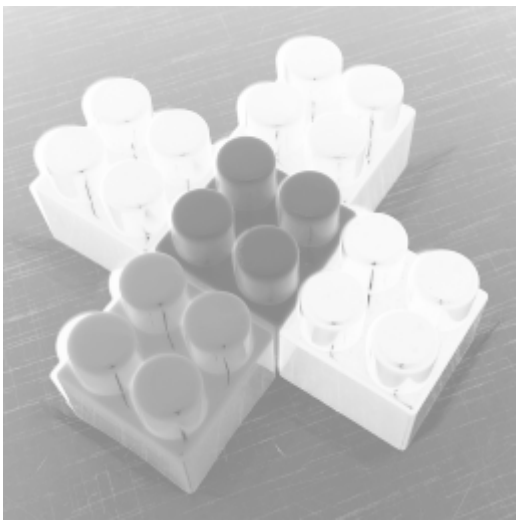
Рис. 3. Результат применения функции преобразования яркости для greyscale


Рис. 4. Результат применения функции преобразования яркости для каналов

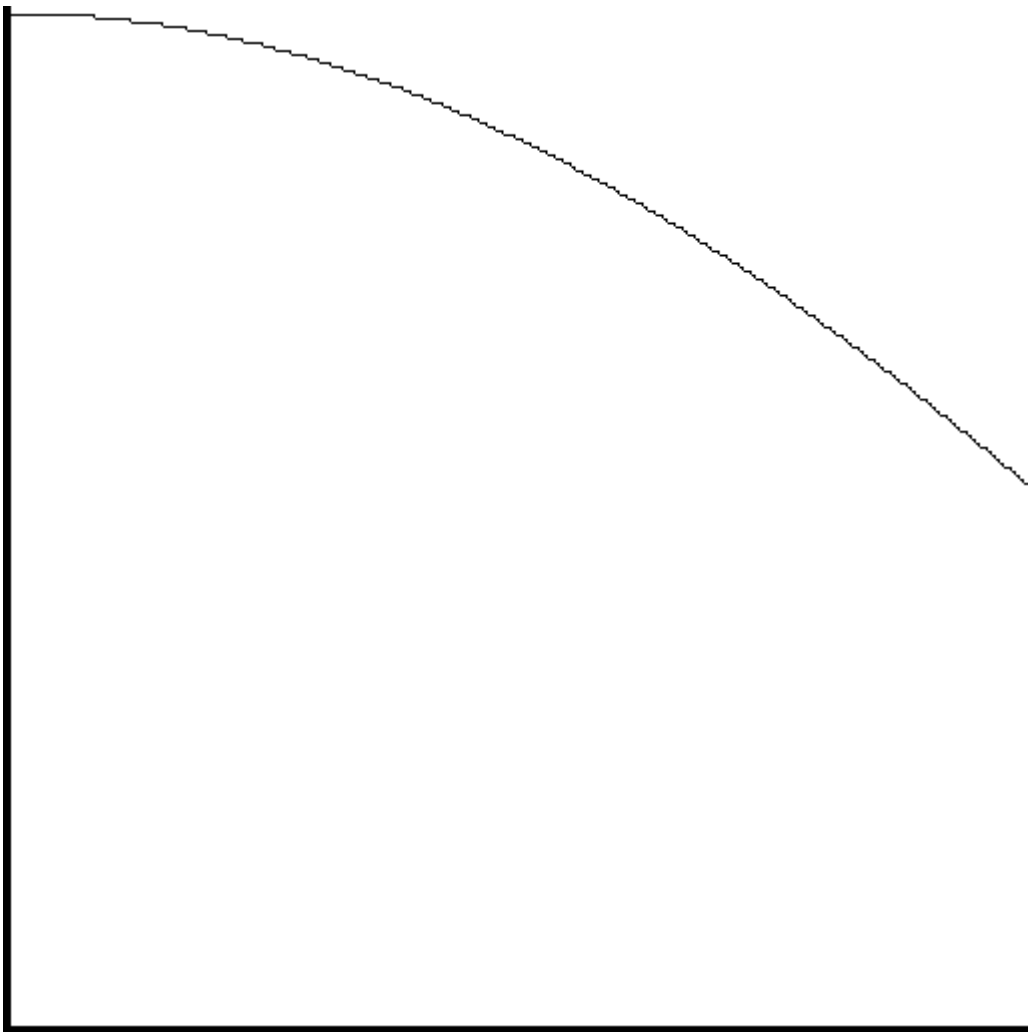

Рис. 5. Визуализация функции яркостного преобразования

Текст программы

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

void countBrightness(int* pixel_array) {
    for (int i = 0; i < 256; i++) {
        pixel_array[i] = cos(i / 255.0) * 255.0;
    }
}
void changeBrightnessGrayscale(cv::Mat grayscale_image, int* pixel_array) {

    for (int i = 0; i < grayscale_image.rows; i++) {
        for (int j = 0; j < grayscale_image.cols; j++) {
            grayscale_image.at<uchar>(i, j) = pixel_array[grayscale_image.at<uchar>(i,
j)];
        }
    }
}
cv::Mat changeBrightness(cv::Mat image, int* pixel_array) {
    cv::Mat channels_split[3];
    cv::split(image, channels_split);
    changeBrightnessGrayscale(channels_split[0], pixel_array);
    changeBrightnessGrayscale(channels_split[1], pixel_array);
    changeBrightnessGrayscale(channels_split[2], pixel_array);

    cv::Mat changed_image;
    cv::Mat channels_merge[3] = { channels_split[0], channels_split[1],
channels_split[2] };
    cv::merge(channels_merge, 3, changed_image);
    return changed_image;
}

int main() {
    cv::Mat image_png = cv::imread("../../../data/cross_0256x0256.png");
    cv::imwrite("lab03_rgb.png", image_png);

    cv::Mat grayscale_image;
    cv::cvtColor(image_png, grayscale_image, cv::COLOR_BGR2GRAY);
    cv::imwrite("lab03_gre.png", grayscale_image);

    int pixels_count[256];
    countBrightness(pixels_count);

    changeBrightnessGrayscale(grayscale_image, pixels_count);
    cv::imwrite("lab03_gre_res.png", grayscale_image);

    cv::imwrite("lab03_rgb_res.png", changeBrightness(image_png, pixels_count));

    cv::Mat graphic(512, 512, CV_8UC3, cv::Scalar(255, 255, 255));

    for (int i = 0; i < 511; i++)
    {
```

```
        cv::line(graphic, cv::Point(i, 512 - pixels_count[(i / 2)] * 2),
            cv::Point(i + 1, 512 - pixels_count[(i + 1) / 2] * 2),
            cv::Scalar(0, 0, 0), 1, 8, 0);
    }
    cv::line(graphic, cv::Point((0), 512),
        cv::Point(512, 512),
        cv::Scalar(0, 0, 0), 5, 8, 0);
    cv::line(graphic, cv::Point((0), 512),
        cv::Point(0, 0),
        cv::Scalar(0, 0, 0), 5, 8, 0);
    cv::imwrite("lab03_viz_func.png", graphic);

}
```