Работа 1. Исследование гамма-коррекции

автор: Дворядкин К. А. дата: 2022-02-16T23:44:28

url: https://github.com/DiorChoppa/imageprocessing-spring-2022/tree/main/prj.labs/lab01

Задание

1. Сгенерировать серое тестовое изображение I_1 в виде прямоугольника размером 768x60 пикселя с плавным изменение пикселей от черного к белому, одна градация серого занимает 3 пикселя по горизонтали.
2. Применить к изображению I_1 гамма-коррекцию с коэффициентом из интервала 2.2-2.4 и получить изображение G_1 при помощи функци pow.
3. Применить к изображению I_1 гамма-коррекцию с коэффициентом из интервала 2.2-2.4 и получить изображение G_2 при помощи прямого обращения к пикселям.
4. Показать визуализацию результатов в виде одного изображения (сверху вниз I_1, G_1, G_2).
5. Сделать замер времени обработки изображений в п.2 и п.3, результаты отфиксировать в отчете.

Результаты

Рис. 1. Результаты работы программы (сверху вниз I_1, G_1, G_2)

Текст программы

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

int main() {

    // Drawing source image

    cv::Mat img(180, 768, CV_8UC1);
    img = 0;
    cv::Rect2d rc = {0, 0, 768, 60 };

    cv::Rect rc1 = { 0, 0, 3, 180 };
    for (int i = 0; i < 768; i += 3) {
          cv::rectangle(img, rc1, { i / 3.0 }, -1);
          rc1.x += rc1.width;
    }
    cv::rectangle(img, rc, {250}, 1);
    rc.y += rc.height;



    // Gammacorrection with pow

    unsigned int start = clock();
```

```
    cv::Mat task1{img};
    task1.convertTo(task1, CV_32FC1, 1.0f / 255.0f);
    cv::pow(task1, 2.2, task1);
    task1.convertTo(task1, CV_8UC1, 255.0f);
    task1(rc).copyTo(img(rc));

    unsigned int finish = clock();
    unsigned int total = finish - start;
    std::cout << "Time consumed for standart pow: " << total << " ms." <<
std::endl;

    cv::rectangle(img, rc, {250}, 1);
    rc.y += rc.height;


    // Gammacorrection by pixel

    start = clock();

    for (int y = rc.y; y < 180; y++){
        for (int x = 0; x < 768; x++){
            img.at<uint8_t>(y, x) = pow(img.at<uint8_t>(y, x) / 255.0f,
2.4) * 255.0f;
        }
    }

    finish = clock();
    total = finish - start;
    std::cout << "Time consumed for every pixel: " << total << " ms." <<
std::endl;

    cv::rectangle(img, rc, {250}, 1);

    cv::imwrite("lab01.png", img);
    cv::imshow("lab01", img);

    cv::waitKey(0);
    cv::destroyAllWindows();

    return 0;
}
```

Time consumed for standart pow: 1117 Time consumed for every pixel: 1630