Работа 1. Исследование гамма-коррекции

Автор: Парчиев Р.Б. дата: 2022-02-16T00:14:11

url: https://github.com/J0hnArren/parchiev_r_b/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)

pow: 5 ms

прямое обращение: 8 ms

Текст программы

```
#include <iostream>
#include <chrono>
#include <opencv2/opencv.hpp>
#define GAMMA 2.4

using namespace std;
using namespace cv;


int main() {
    auto t0 = std::chrono::steady_clock::now();
    Mat image(Mat::zeros(60, 768, CV_8UC1));
    for (int i=0; i < image.cols; ++i) {
        cv::line(image,
```

```
            Point(i, 255),
            Point(i, 0),
            Scalar(i / 3, i / 3, i / 3),
            3, // gradation
            LINE_8); // type of the line
    }

    auto t1 = std::chrono::steady_clock::now();

    // Gamma correction by pow functuion
    Mat pow_res(image);
    pow_res.convertTo(pow_res, CV_64F, 1 / 255.0);
    pow(pow_res, GAMMA, pow_res);
    pow_res.convertTo(pow_res, CV_8UC1, 255.0);

    auto t2 = std::chrono::steady_clock::now();

    // Gamma correction by direct reference to pixels
    Mat pixel_res;
    image.convertTo(pixel_res, CV_64F);
    for (int i = 0; i < pixel_res.rows; ++i)
        for (int j = 0; j < pixel_res.cols; ++j) {
            pixel_res.at<double>(i, j) = (pow(pixel_res.at<double>(i, j) /
                255.0, GAMMA) * 255.0);
        }
    pixel_res.convertTo(pixel_res, CV_8UC1);

    auto t3 = std::chrono::steady_clock::now();

    int rows = image.rows;
    int columns = image.cols;
    Mat picture = Mat::zeros(rows * 3, columns, CV_8UC1);
    image.copyTo(picture(Rect(0, rows * 0, columns, rows)));
    pow_res.copyTo(picture(Rect(0, rows * 1, columns, rows)));
    pixel_res.copyTo(picture(Rect(0, rows * 2, columns, rows)));
    imshow("Press any key ...", picture);
    auto time1 = std::chrono::duration_cast<std::chrono::milliseconds>(t1 - t0);
    auto time2 = std::chrono::duration_cast<std::chrono::milliseconds>(t2 - t1);
    auto time3 = std::chrono::duration_cast<std::chrono::milliseconds>(t3 - t2);
    cout << "\nGenerating image in: "
         << time1.count() << " ms\n"
         << "Gamma correction by pow functuion in : "
         << time2.count() << " ms\n"
         << "Gamma correction by direct reference to pixels in: "
         << time3.count() << " ms " << endl;

    // Saving image
    imwrite("lab01.png", picture);
    waitKey(0);

    return 0;
}
```