Работа 1. Исследование гамма-коррекции

автор: Попов Д.В.
дата: 2022-02-21T22:48:22

https://github.com/Popivzanin/opencv/tree/master/popov_d_v/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

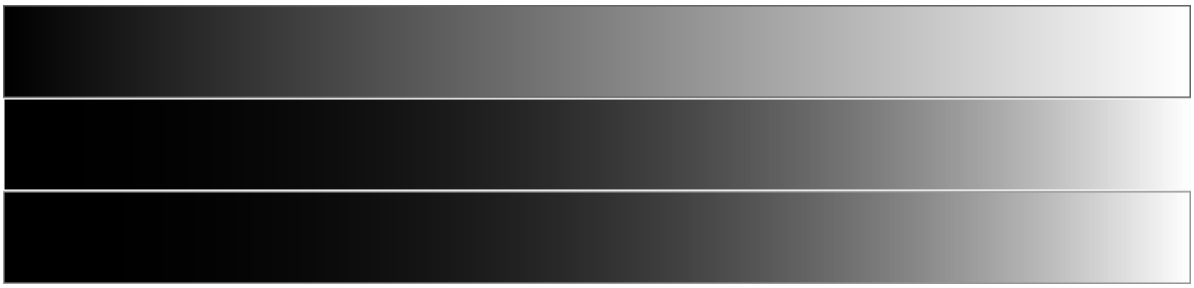

Рис. 1. Результаты работы программы (сверху вниз I_1, G_1, G_2)

time1 = 0 ms
time2 = 1 ms

Текст программы

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

int main() {
  cv::Mat img(180,768,CV_8UC1);
  // draw dummy image
  img = 0;
  cv::Rect2d rc = {0, 0, 768, 60 };
  for (int y = 0; y < 180; y++) {
      for (int x = 0; x < 768; x++) {
          img.at<uchar>(y, x) = x / 3;
      }
  }
  cv::rectangle(img, rc, { 100 }, 1);
  rc.y += rc.height;
  cv::Mat img1(img);
  auto start = std::chrono::high_resolution_clock::now();
```

```
  img1.convertTo(img1, CV_32F,1.0/255.0);
  cv::pow(img1, 2.3F, img1);
  img1.convertTo(img1, CV_8UC1,255.0);
  auto stop = std::chrono::high_resolution_clock::now();
  auto duration = std::chrono::duration_cast<std::chrono::milliseconds>(stop -
start);
  std::cout << "time1 = " << duration.count() << " ms" << std::endl;
  img1(rc).copyTo(img(rc));
    cv::rectangle(img, rc, { 250 }, 1);
  rc.y += rc.height;
  start = std::chrono::high_resolution_clock::now();
  for (int y = rc.y; y < 180; y++) {
      for (int x = 0; x < 768; x++) {
          img.at<uchar>(y, x) = pow(( (img.at<uchar>(y,x) / 255.0)), 2.4)*255.0;
      }
  }
  stop = std::chrono::high_resolution_clock::now();
  duration = std::chrono::duration_cast<std::chrono::milliseconds>(stop -
start);
  std::cout << "time2 = " << duration.count() << " ms" << std::endl;

  cv::rectangle(img, rc, { 150 }, 1);
  // save result
  cv::imwrite("lab01.png", img);
}
```