

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**(ДГТУ)**

Факультет «Информатика и вычислительная техника»

наименование факультета

Кафедра «Математика и информатика»

наименование кафедры

**Отчет**

по «Лабораторная работа №1»

Выполнил студент

группы МСК21

Леонавичус Даниил

Ростов-на-Дону

2024

**Цель работы**

Освоение основных яркостных и геометрических характеристик изображений и их использование для анализа изображений.

**Теоретическое обоснование**

**Пиксель цифрового изображения**

Пиксель цифрового изображения характеризуется тремя параметрами: (x,y,I), где пара целочисленных значений (*x*,*y*) описывает геометрическое положение пикселя в плоскости изображения, а значение I*I* характеризует его яркость (интенсивность) в точке плоскости. Таким образом, в изображении можно выделить яркостную и геометрическую составляющие. В общем случае эти составляющие не связаны между собой (например, изменение в освещенности сцены не изменит геометрических параметров объектов на сцене). Из-за этого проще исследовать отдельно яркостные свойства изображения и отдельно — геометрические. Такой подход понижает порядок исследуемого изображения в случае геометрических свойств с *n*=3 (x, y, I) до *n*=2 (x, y), а в случае яркостных свойств — до *n* =1.

**Гистограмма изображения**

Яркостная составляющая изображения характеризуется одномерным массивом гистограммы, из которого можно вычислить контраст. Гистограмма — это распределение частоты встречаемости пикселей одинаковой яркости на изображении. Яркость — это среднее значение интенсивности сигнала. Контраст — это интервал значений между минимальной и максимальной яркостями изображения.

Для 8-битного полутонового изображения гистограмма яркости представляет собой одномерный целочисленный массив Hist из 256 элементов [0, 255]. Элементом гистограммы Hist[i] является сумма пикселей изображения с яркостью i. По визуальному отображению гистограммы можно оценить необходимость изменения яркости и контрастности изображения, оценить площадь, занимаемую светлыми и темными элементами, определить местоположение на плоскости изображения отдельных объектов, соответствующих некоторым диапазонам яркости. Для цветного RGB-изображения необходимо построить три гистограммы по каждому цвету.

**Профили и проекции изображения**

Для сведения геометрических составляющих изображения к одномерному массиву данных *n*=1 используются такие характеристики, как «профили» и «проекции» изображения.

Профиль вдоль линии: это функция интенсивности изображения, распределенного вдоль данной линии (прорезки). Простейшим случаем профиля изображения является профиль строки.

Проекция на ось: это сумма интенсивностей пикселей изображения, взятая в направлении перпендикулярном данной оси. Простейшим случаем проекции двумерного изображения являются вертикальная проекция на ось *Ox*, представляющая собой сумму интенсивностей пикселей по столбцам изображения и горизонтальная проекция на ось *Oy*, представляющая собой сумму интенсивностей пикселей по строкам изображения.

Анализ массива проекции позволяет выделять характерные точки функции проекции, которые соответствуют контурам объектов на изображении. Например, если на изображении имеются контрастные объекты, то в проекции будут видны перепады или экстремумы функции, соответствующие положению каждого из объектов.

**Арифметические операции**

Простейшими способами выравнивания гистограммы являются арифметические операции с изображениями. Например, если большинство значений гистограммы находятся слева, то изображение является темным. Для увеличения детализации темных областей можно сдвинуть гистограмму правее, в более светлую область, например, на 50 градаций для каждого цвета. Однако, этот подход обладает недостатком: повышение интенсивностей темных областей приводит к сдвигу светлых к максимуму, что может привести к потере информации в светлых областях.

**Растяжение динамического диапазона**

Если интенсивности пикселей областей интереса находятся в узком динамическом диапазоне, то можно растянуть этот диапазон.

**Таблица поиска (LUT)**

Таблица поиска или LUT (Lookup table) — удобный инструмент для преобразования интенсивностей точек всего изображения. Поскольку большинство изображений определяется с использованием ограниченного набора возможных дискретных значений цвета (чаще всего цвет задается с использованием целых чисел, находящихся в диапазоне [0, 255]), то можно заранее вычислить преобразование цвета для каждого потенциально-возможного значения цвета, сохранить их в таблице поиска, а затем преобразовать цвет исходного изображения.

**Профиль изображения**

Профилем изображения вдоль некоторой линии называется функция интенсивности изображения, распределенного вдоль данной линии (прорезки). Простейшим случаем профиля изображения является профиль строки.

В общем случае профиль можно рассматривать вдоль любой прямой, ломаной или кривой линии, пересекающей изображение. После формирования массива профиля изображения проводится его анализ стандартными средствами. Анализ позволяет автоматически выделять особые точки функции профиля, соответствующие контурам изображения, пересекаемым данной линией.

**Проекция изображения**

Проекцией изображения на некоторую ось называется сумма интенсивностей пикселей изображения в направлении, перпендикулярном данной оси. Простейшим случаем проекции двумерного изображения являются вертикальная проекция на ось *Ox*, представляющая собой сумму интенсивностей пикселей по столбцам изображения и горизонтальная проекция на ось *Oy*, представляющая собой сумму интенсивностей пикселей по строкам изображения.

Анализ массива проекции позволяет выделять характерные точки функции проекции, которые соответствуют контурам объектов на изображении. Например, если на изображении имеются контрастные объекты, то в проекции будут видны перепады или экстремумы функции, соответствующие положению каждого из объектов.

**Ход выполнения работы**

**1. Гистограммы**

**Исходное изображение**



**Листинг программной реализаций**

/\*

1. Гистограммы. Выбрать произвольное слабоконтрастное изображение.

Выполнить выравнивание гистограммы и растяжение контраста,

использовать рассмотренные преобразования.

Сравнить полученные результаты.

\*/

#include <opencv2/opencv.hpp>

#include <iostream>

#include <cmath>

using namespace cv;

using namespace std;

// Функция для экспоненциального преобразования

Mat exponentialTransform(const Mat& src, double gamma) {

    Mat dst;

    src.convertTo(dst, CV\_32F, 1.0 / 255.0);  // Нормализация изображения к диапазону [0, 1]

    pow(dst, gamma, dst);                     // Применение экспоненциального преобразования

    dst.convertTo(dst, CV\_8U, 255);           // Возвращение к диапазону [0, 255] и 8-битному формату

    return dst;

}

// Функция для гиперболического преобразования

Mat hyperbolicTransform(const Mat& src, double alpha) {

    Mat dst;

    src.convertTo(dst, CV\_32F); // Конвертация в 32-битный формат

    dst = alpha \* dst;          // Масштабирование с коэффициентом alpha

    Mat expPos, expNeg;

    exp(dst, expPos);           // Вычисление e^(alpha \* dst)

    exp(-dst, expNeg);          // Вычисление e^(-alpha \* dst)

    dst = (expPos - expNeg) / (expPos + expNeg); // Гиперболическое преобразование

    return dst;

}

// Функция для преобразования по закону степени (2/3)

Mat powerTransform(const Mat& src, double power) {

    Mat dst;

    src.convertTo(dst, CV\_32F, 1.0 / 255.0); // Нормализация к диапазону [0, 1]

    pow(dst, power, dst);                    // Преобразование по степени

    dst.convertTo(dst, CV\_8U, 255);          // Обратная конвертация к диапазону [0, 255]

    return dst;

}

int main() {

    // Загрузка изображения

    Mat I = imread("background.png", IMREAD\_COLOR);

    if (I.empty()) {

        cerr << "Ошибка: не удалось загрузить изображение!" << endl;

        return -1;

    }

    // Преобразование в оттенки серого

    Mat gray;

    cvtColor(I, gray, COLOR\_BGR2GRAY);

    // 1. Выравнивание гистограммы

    Mat equalized;

    equalizeHist(gray, equalized);

    // 2. Растяжение динамического диапазона

    Mat stretched;

    normalize(gray, stretched, 0, 255, NORM\_MINMAX);

    // 3. Экспоненциальное преобразование

    Mat exponential = exponentialTransform(gray, 2.0);

    // 4. Преобразование по закону степени (2/3)

    Mat power = powerTransform(gray, 2.0 / 3.0);

    // 5. Гиперболическое преобразование

    Mat hyperbolic = hyperbolicTransform(gray, 5.0);

    // Отображение всех изображений

    namedWindow("Оригинальное изображение", WINDOW\_NORMAL); // Создание окна для оригинального изображения

    resizeWindow("Оригинальное изображение", 800, 600);    // Изменение размера окна

    imshow("Оригинальное изображение", gray);               // Показ изображения

    namedWindow("Выравнивание гистограммы", WINDOW\_NORMAL);

    resizeWindow("Выравнивание гистограммы", 800, 600);

    imshow("Выравнивание гистограммы", equalized);          // Показ изображения после выравнивания гистограммы

    namedWindow("Растяжение динамического диапазона", WINDOW\_NORMAL);

    resizeWindow("Растяжение динамического диапазона", 800, 600);

    imshow("Растяжение динамического диапазона", stretched); // Показ изображения после растяжения диапазона

    namedWindow("Экспоненциальное преобразование", WINDOW\_NORMAL);

    resizeWindow("Экспоненциальное преобразование", 800, 600);

    imshow("Экспоненциальное преобразование", exponential);  // Показ изображения после экспоненциального преобразования

    namedWindow("Преобразование по закону степени 2/3", WINDOW\_NORMAL);

    resizeWindow("Преобразование по закону степени 2/3", 800, 600);

    imshow("Преобразование по закону степени 2/3", power);   // Показ изображения после преобразования по степени 2/3

    namedWindow("Гиперболическое преобразование", WINDOW\_NORMAL);

    resizeWindow("Гиперболическое преобразование", 800, 600);

    imshow("Гиперболическое преобразование", hyperbolic);    // Показ изображения после гиперболического преобразования

    // Ожидание нажатия клавиши для закрытия окон

    while (true) {

        int key = waitKey(0);

        if (key == 'q' || key == 'Q') {

            break;

        }

    }

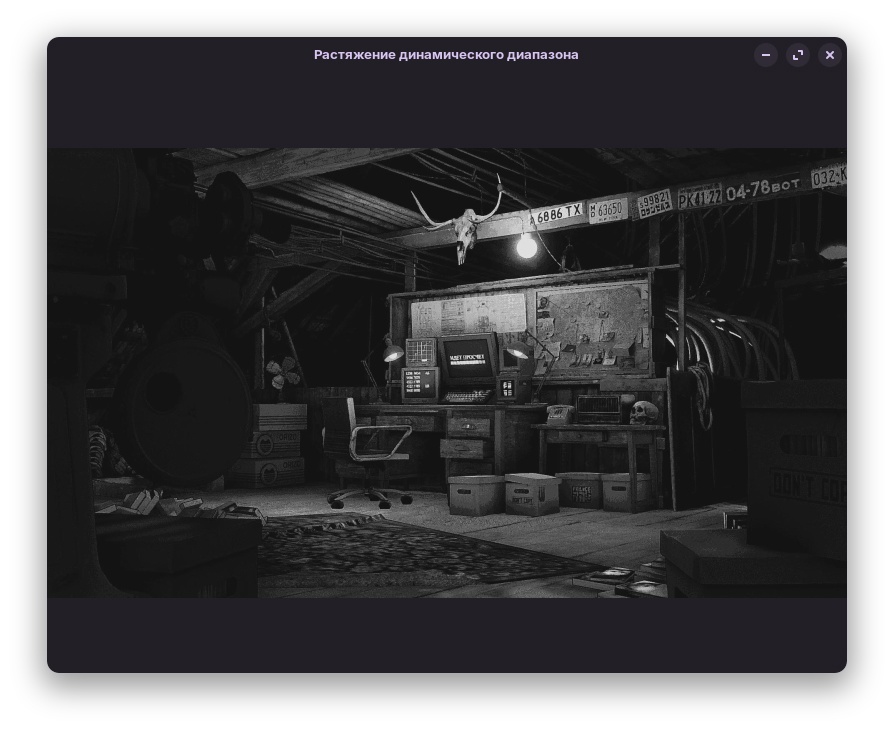
    return 0;

}

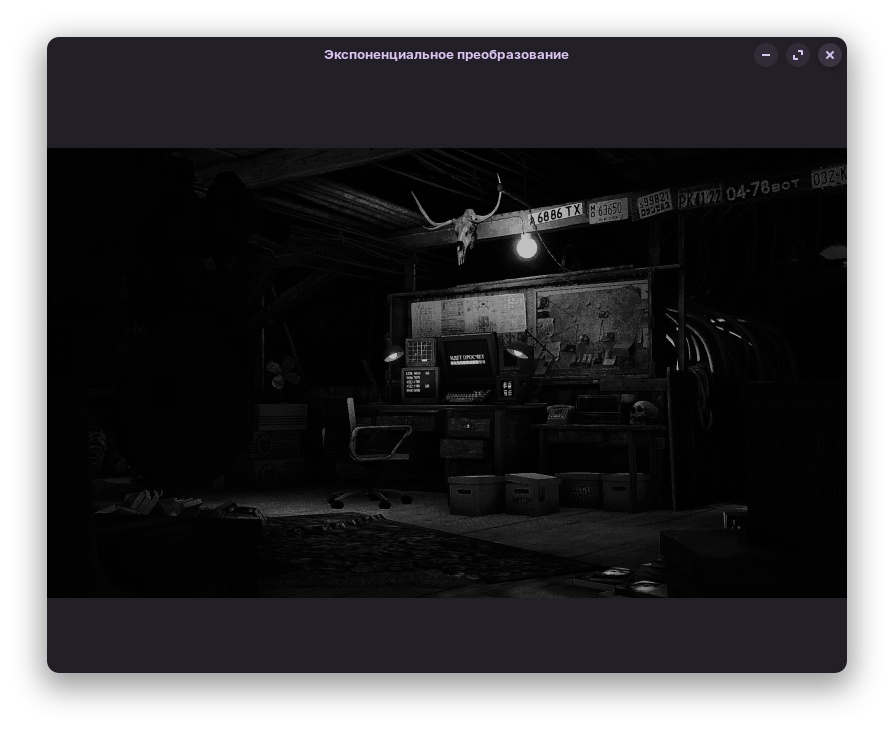
**Комментарии**

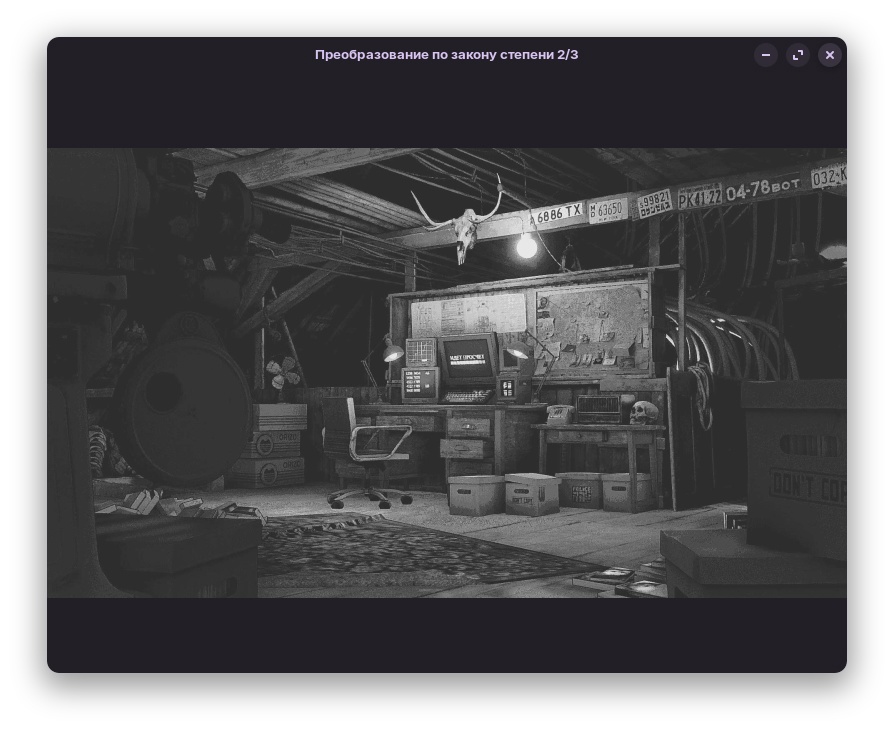
* Выравнивание гистограммы: улучшает контрастность изображения, делая его более четким и детализированным.
* Растяжение динамического диапазона: увеличивает контраст, делая изображение более ярким и насыщенным.
* Экспоненциальное преобразование: усиливает яркие области и ослабляет темные, что может быть полезно для выделения определенных деталей.
* Преобразование по закону степени (2/3): сглаживает изображение, уменьшая контраст между яркими и темными областями.
* Гиперболическое преобразование: придает изображению более контрастный вид, выделяя детали.

**Результирующие изображения**











**2. Проекции**

**Исходное изображение**



**Листинг программной реализаций**

/\*

2. Проекции. Выбрать произвольное изображение, содержащее монотонные области и выделяющиеся объекты.

Произвести построение проекций изображения на вертикальную и горизонтальную оси.

Определить границы областей объектов.

\*/

#include <opencv2/opencv.hpp>

#include <iostream>

using namespace cv;

using namespace std;

int main() {

    Mat I = imread("ava2.jpg", IMREAD\_GRAYSCALE);

    // Проверка успешности загрузки изображения

    if (I.empty()) {

        cerr << "Ошибка загрузки изображения!" << endl;

        return -1;

    }

    // Вертикальная проекция (на Oy)

    Mat proj\_oy = Mat::zeros(I.rows, 1, CV\_32F);  // Создаем матрицу для вертикальной проекции

    for (int i = 0; i < I.rows; i++) {

        float sum = 0;

        for (int j = 0; j < I.cols; j++) {

            sum += I.at<uchar>(i, j);  // Суммируем интенсивности всех пикселей в строке i

        }

        proj\_oy.at<float>(i) = sum;  // Сохраняем сумму интенсивностей в матрице вертикальной проекции

    }

    // Горизонтальная проекция (на Ox)

    Mat proj\_ox = Mat::zeros(1, I.cols, CV\_32F);  // Создаем матрицу для горизонтальной проекции

    for (int j = 0; j < I.cols; j++) {

        float sum = 0;

        for (int i = 0; i < I.rows; i++) {

            sum += I.at<uchar>(i, j);  // Суммируем интенсивности всех пикселей в столбце j

        }

        proj\_ox.at<float>(0, j) = sum;  // Сохраняем сумму интенсивностей в матрице горизонтальной проекции

    }

    // Нормализация проекций для отображения

    double proj\_oy\_max, proj\_ox\_max;

    minMaxLoc(proj\_oy, nullptr, &proj\_oy\_max);  // Находим максимальное значение в вертикальной проекции

    minMaxLoc(proj\_ox, nullptr, &proj\_ox\_max);  // Находим максимальное значение в горизонтальной проекции

    // Создаем изображения для отображения проекций

    Mat proj\_oy\_img = Mat::ones(256, I.cols, CV\_8UC3) \* 255;  // Белое изображение для вертикальной проекции

    Mat proj\_ox\_img = Mat::ones(I.rows, 256, CV\_8UC3) \* 255;  // Белое изображение для горизонтальной проекции

    // Нормализуем значения проекций для графического отображения

    for (int i = 0; i < proj\_oy.rows; i++) {

        int normalized\_value = cvRound(proj\_oy.at<float>(i) \* proj\_oy\_img.cols / proj\_oy\_max);  // Нормализуем значение к ширине изображения

        line(proj\_oy\_img, Point(0, i), Point(normalized\_value, i), Scalar(0, 0, 0), 1);  // Рисуем линию, отображающую вертикальную проекцию

    }

    for (int j = 0; j < proj\_ox.cols; j++) {

        int normalized\_value = cvRound(proj\_ox.at<float>(0, j) \* proj\_ox\_img.rows / proj\_ox\_max);  // Нормализуем значение к высоте изображения

        line(proj\_ox\_img, Point(j, 0), Point(j, normalized\_value), Scalar(0, 0, 0), 1);  // Рисуем линию, отображающую горизонтальную проекцию

    }

    // Отображение всех изображений

    namedWindow("Оригинальное изображение", WINDOW\_NORMAL);

    resizeWindow("Оригинальное изображение", 800, 600);

    namedWindow("Вертикальная проекция", WINDOW\_NORMAL);

    resizeWindow("Вертикальная проекция", 800, 600);

    namedWindow("Горизонтальная проекция", WINDOW\_NORMAL);

    resizeWindow("Горизонтальная проекция", 800, 600);

    imshow("Оригинальное изображение", I);           // Показ оригинального изображения

    imshow("Вертикальная проекция", proj\_oy\_img);    // Показ вертикальной проекции

    imshow("Горизонтальная проекция", proj\_ox\_img);  // Показ горизонтальной проекции

    // Ожидание нажатия клавиши для закрытия окон

    while (true) {

        int key = waitKey(0);

        if (key == 'q' || key == 'Q') {

            break;

        }

    }

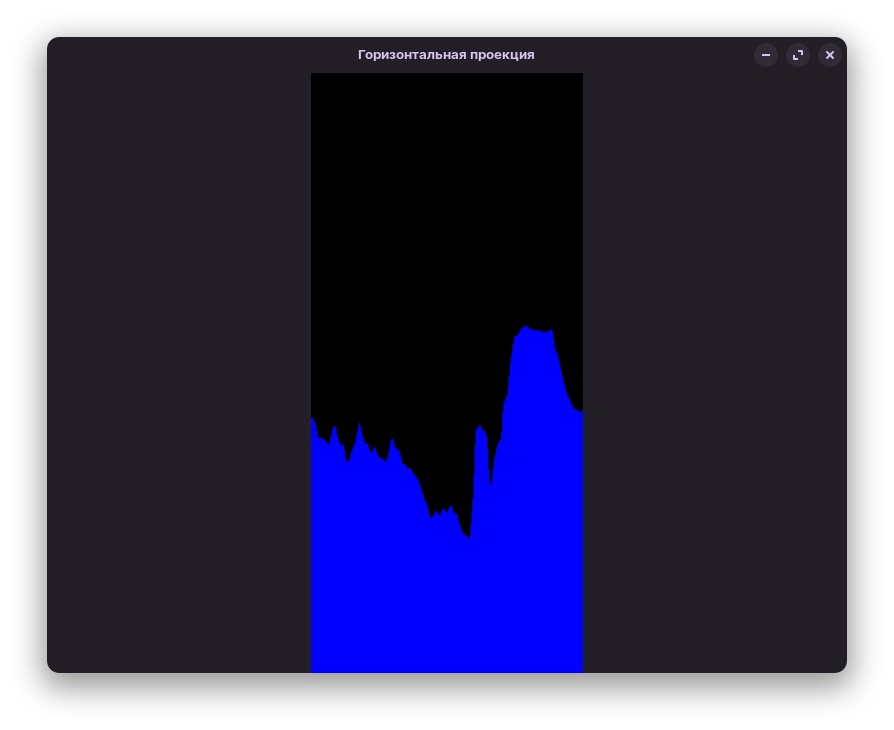
    return 0;

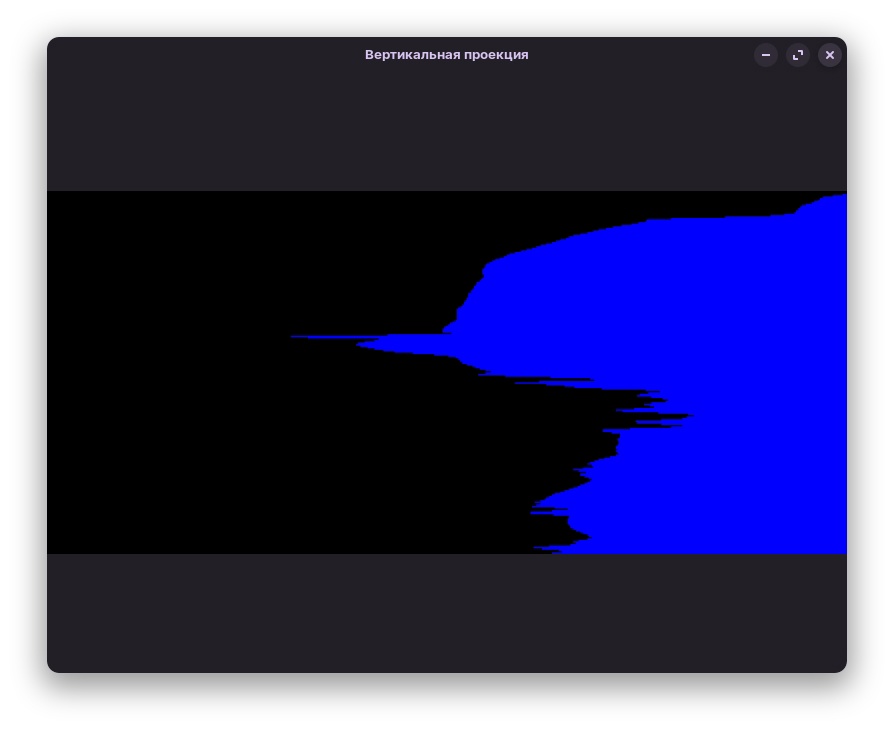
}

**Комментарии**

* Вертикальная проекция: позволяет определить границы объектов по вертикали, выделяя области с высокой плотностью пикселей.
* Горизонтальная проекция: позволяет определить границы объектов по горизонтали, выделяя области с высокой плотностью пикселей.

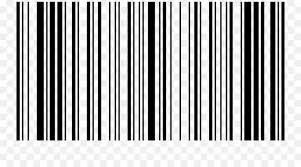
**Результирующие изображения**





**3. Профили**

**Исходное изображение**



**Листинг программной реализаций**

/\*

3. Профили. Выбрать произвольное изображение, содержащие штрих-код.

Выполнить построение профиля изображения вдоль штрих-кода.

\*/

#include <opencv2/opencv.hpp>

#include <iostream>

using namespace cv;

using namespace std;

int main() {

    // Загрузка изображения с штрих-кодом в оттенках серого

    Mat I = imread("images.jpeg", IMREAD\_GRAYSCALE);

    // Проверка успешности загрузки изображения

    if (I.empty()) {

        cerr << "Ошибка загрузки изображения!" << endl;

        return -1;

    }

    // Бинаризация изображения с применением автоматического порогового значения Otsu

    Mat binaryImage;

    threshold(I, binaryImage, 0, 255, THRESH\_BINARY | THRESH\_OTSU);

    // Определение центральной линии для профиля штрих-кода

    int profileLine = I.rows / 2; // Высота линии профиля - середина изображения

    Mat profile = Mat::zeros(1, I.cols, CV\_32F); // Массив для хранения значений профиля

    // Суммируем интенсивности пикселей вдоль горизонтальной области вокруг центральной линии

    int profileBand = 10; // Полу-ширина зоны, используемой для усреднения

    for (int j = 0; j < I.cols; j++) {

        float sum = 0;

        for (int i = profileLine - profileBand; i <= profileLine + profileBand; i++) {

            sum += binaryImage.at<uchar>(i, j); // Суммируем интенсивности пикселей

        }

        profile.at<float>(0, j) = sum / (2 \* profileBand + 1); // Считаем среднюю интенсивность

    }

    // Нормализация профиля для отображения

    normalize(profile, profile, 0, 255, NORM\_MINMAX);

    // Создаем изображение для графического отображения профиля

    Mat profileImg = Mat::zeros(100, I.cols, CV\_8UC3); // Пустое изображение для профиля

    for (int j = 0; j < I.cols; j++) {

        int intensity = cvRound(profile.at<float>(0, j)); // Преобразуем значение профиля в целое число

        line(profileImg, Point(j, profileImg.rows), Point(j, profileImg.rows - intensity), Scalar(255, 255, 255), 1); // Рисуем линию интенсивности

    }

    // Отображение исходного изображения и профиля

    namedWindow("Изображение с штрих-кодом", WINDOW\_NORMAL);

    resizeWindow("Изображение с штрих-кодом", 800, 600);

    namedWindow("Профиль изображения", WINDOW\_NORMAL);

    resizeWindow("Профиль изображения", 800, 600);

    imshow("Изображение с штрих-кодом", I);         // Отображение оригинального изображения

    imshow("Профиль изображения", profileImg);      // Отображение построенного профиля

    // Ожидание нажатия клавиши для завершения программы

    while (true) {

        int key = waitKey(0);

        if (key == 'q' || key == 'Q') {

            break;

        }

    }

    return 0;

}

**Комментарии**

* Профиль изображения: позволяет анализировать структуру штрих-кода, выделяя области с высокой и низкой интенсивностью пикселей

**Результирующие изображения**



**Выводы о проделанной работе**

В ходе выполнения работы были изучены и реализованы различные методы обработки изображений с использованием библиотеки OpenCV. Построение гистограмм и их выравнивание показало, как можно улучшить визуальные характеристики изображений с низким контрастом. Проекции и профили изображения продемонстрировали возможность выделения объектов и анализа структуры изображения, что полезно при обработке изображений с монотонными областями и для автоматического выделения объектов.

**Вопросы к защите лабораторной работе**

1. **Что такое контрастность изображения и как её можно изменить?**

Контрастность изображения — это разница в яркости между самыми светлыми и самыми темными частями изображения. Высокая контрастность означает, что различия между светлыми и темными областями значительны, что делает изображение более четким и детализированным. Низкая контрастность, наоборот, делает изображение более размытым и менее детализированным.

Как изменить контрастность изображения:

* **Выравнивание гистограммы**: Этот метод перераспределяет интенсивности пикселей, чтобы гистограмма изображения стала более равномерной. Это улучшает контрастность, особенно в изображениях с низким контрастом.
* **Растяжение динамического диапазона**: Этот метод масштабирует интенсивности пикселей так, чтобы они занимали весь диапазон от 0 до 255. Это увеличивает контраст, делая изображение более ярким и насыщенным.
* **Экспоненциальное преобразование**: Применение экспоненциальной функции к интенсивностям пикселей может усилить яркие области и ослабить темные, что может быть полезно для выделения определенных деталей.
* **Преобразование по закону степени**: Этот метод изменяет интенсивности пикселей по степени, что может сгладить изображение или усилить контраст в зависимости от выбранного значения степени.
* **Гиперболическое преобразование:** Этот метод придает изображению более контрастный вид, выделяя детали.

1. **Чем эффективно использование профилей и проекций изображения?**

Профили и проекции изображения эффективны для анализа и выделения объектов на изображении. Они позволяют:

* **Выделение объектов**: Проекции и профили помогают определить границы объектов, выделяя области с высокой плотностью пикселей. Это особенно полезно для изображений с монотонными областями.
* **Анализ структуры**: Профили позволяют анализировать структуру изображения, такую как штрих-коды, выделяя области с высокой и низкой интенсивностью пикселей.
* **Автоматическое выделение объектов**: Проекции и профили могут быть использованы для автоматического выделения объектов на изображении, что полезно в задачах компьютерного зрения и обработки изображений.
* **Упрощение анализа**: Проекции и профили упрощают анализ изображения, предоставляя графическое представление интенсивностей пикселей, что облегчает понимание структуры изображения.

1. **Каким образом можно найти объект на равномерном фоне?**

Найти объект на равномерном фоне можно с помощью различных методов обработки изображений:

* **Выравнивание гистограммы**: Этот метод может улучшить контрастность изображения, делая объект более заметным на фоне.
* **Проекции и профили**: Проекции и профили помогают определить границы объектов, выделяя области с высокой плотностью пикселей. Это полезно для изображений с монотонными областями.