**МИНОБРНАУКИ РОССИИ**

**Санкт-Петербургский государственный**

**электротехнический университет**

**«ЛЭТИ» им. В.И. Ульянова (Ленина)**

**Кафедра Телевидения и Видеотехники**

**курсовой проект**

**по дисциплине «Информационные технологии»**

**Тема: Разработка редактора изображений**

|  |  |  |
| --- | --- | --- |
| Студент гр. 2106 |  | Стрельников И.Д. |
| Преподаватель |  | Чиркунова А.А. |

Санкт-Петербург

2023

**ЗАДАНИЕ**

**на курсовой проект**

|  |  |  |
| --- | --- | --- |
| Студент Стрельников И.Д. | | |
| Группа 2106 | | |
| Тема проекта: Разработка редактора изображений | | |
| Исходные данные:  Изображение формата .bmp; 1.3(вырез участка изображения); 2.1(Увеличение масштаба по вертикали); 3.9(Соляризация); 4.1(линейная растяжка гистограммы); 5.1(пороговая сегментация оранжевый HSV). | | |
| Содержание пояснительной записки:  Содержание, Введение, 1. Теоретическая часть,2. Программные документы, Заключение, Список использованных источников | | |
| Предполагаемый объем пояснительной записки:  Не менее 15страниц. | | |
| Дата выдачи задания:24.03.2023 | | |
| Дата сдачи реферата: | | |
| Дата защиты реферата: | | |
| Студент |  | Стрельников И.Д. |
| Преподаватель |  | Чиркунова А.А. |

**Аннотация**

Моя курсовая работа основывается на обработке изображений. Она заключается в загрузке изображении, использовании фильтров и применении разных методов работы с ними. Курсовое задние заключается в создании или загрузки изображения и изменении его визуальных свойств. Используются такие фильтры как:

- Вырез участка изображения

- Увеличение масштаба по вертикали

- СОЛЯРИЗАЦИЯ

- Линейная растяжка гистограммы

- Пороговая сегментация

**содержание**

|  |  |  |
| --- | --- | --- |
|  | Введение | 6 |
| 1. | Теоретическая часть | 7 |
| 1.1. | Фильтр «Вырез участка изображения» | 7 |
| 1.2. | Фильтр «Увеличение масштаба по вертикали» | 7 |
| 1.3. | Фильтр «СОЛЯРИЗАЦИЯ» | 7 |
| 1.4. | Фильтр «Линейная растяжка гистограммы» | 7 |
| 1.5. | Фильтр «Пороговая сегментация» | 7 |
| 2. | Программные документы | 8 |
| 2.1. | Спецификация | 8 |
| 2.2. | Описание программы | 8 |
| 2.2.1. | Общие сведения | 0 |
| 2.2.2. | Функциональное назначение | 0 |
| 2.2.3. | Описание логической структуры | 0 |
| 2.2.4. | Используемые технические средства | 0 |
| 2.2.5. | Вызов и загрузка | 0 |
| 2.2.6. | Выходные данные | 0 |
| 2.3. | Текст программы | 0 |
| 2.4. | Эксплуатационные документы | 0 |
| 2.4.1. | Ведомость эксплуатационных документов | 0 |
| 2.4.2. | Описание применения | 0 |
| 2.4.2.1. | Назначение программы | 0 |
| 2.4.2.2. | Условия применения | 0 |
| 2.4.2.3. | Описание задачи | 0 |
| 2.4.2.4 | Входные и выходные данные | 0 |
| 2.4.3. | **Руководство оператора** | 0 |
| 2.4.3.1. | Назначение программы | 0 |
| 2.4.3.2. | Условия выполнения программы | 0 |
| 2.4.3.3. | Выполнение программы | 0 |
| 2.4.3.4. | Выполнение программы | 0 |
| 2.4.3.5. | Сообщения оператору | 0 |
| 2.4.4. | **Описание языка** | 0 |
| 2.4.4.1. | Общие сведения | 0 |
| 2.4.4.2. | Элементы языка | 0 |
| 2.4.4.3. | Средства обмена данными | 0 |
| 2.4.4.4. | Встроенные элементы | 0 |
| 2.4.4.5. | Средства отладки программы | 0 |
|  | Заключение | 0 |
|  | Список использованных источников | 0 |
|  | Приложение А. Ссылочные нормативные документы | 18 |

**введение**

Цель работы: разработка программного обеспечения для редактирования изображений, отладка программного обеспечения.

Требования: рисование фигуры, вырез участка изображения; масштабирование изображения(увеличение по вертикали); линейное контрастирование изображения(Соляризация); линейная растяжка гистограммы; пороговая сегментация цветных изображений(HSV оранжевый).

**1. тЕОРЕТИЧЕСКАЯ ЧАСТЬ**

**1.1. Фильтр** «Рисование фигуры»

**1.2. Фильтр** «Масштабирование изображения»

**1.3. Фильтр** «Соляризация»

**1.4. Фильтр** «Гистограмма»

**1.5. Фильтр** «Пороговая сегментация цветных изображений»

**2. пРограммная документация**

**2.1. Спецификация**

- Компоненты – windows 10, visual studio 22, c++(opencv)

|  |  |  |
| --- | --- | --- |
| Обозначение | Наименование | Примечание |
|  |  |  |
|  |  |  |
|  |  |  |

**2.2. Описание программы**

2.2.1. Общие сведения: windows 10, visual studio 22, c++(opencv)

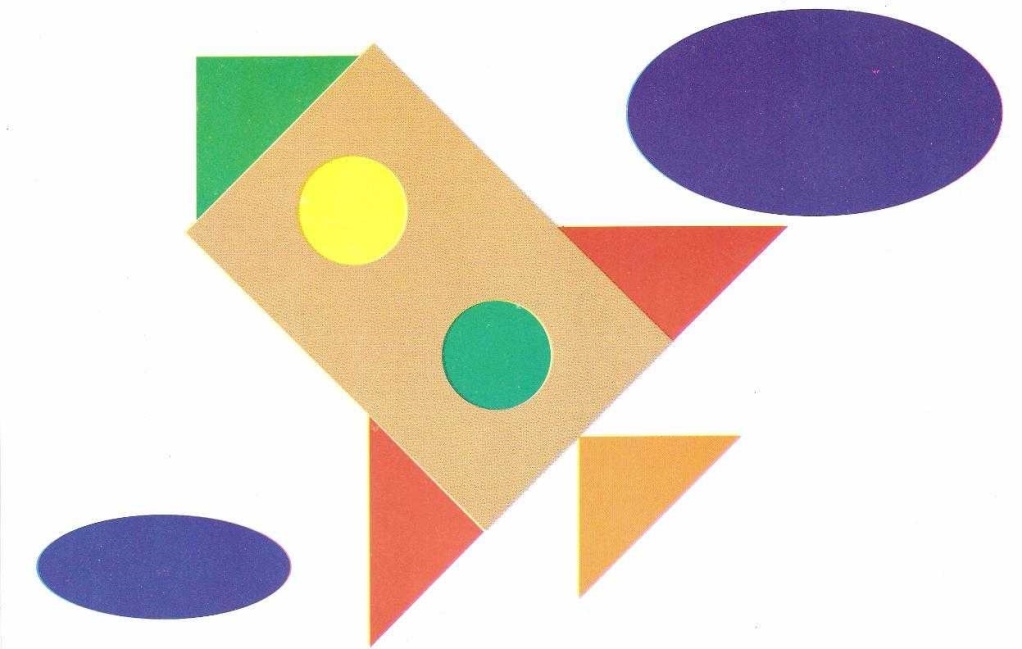
2.2.2. Функциональное назначение: обработка изображений, ограничение: обработка текстовых файлов.

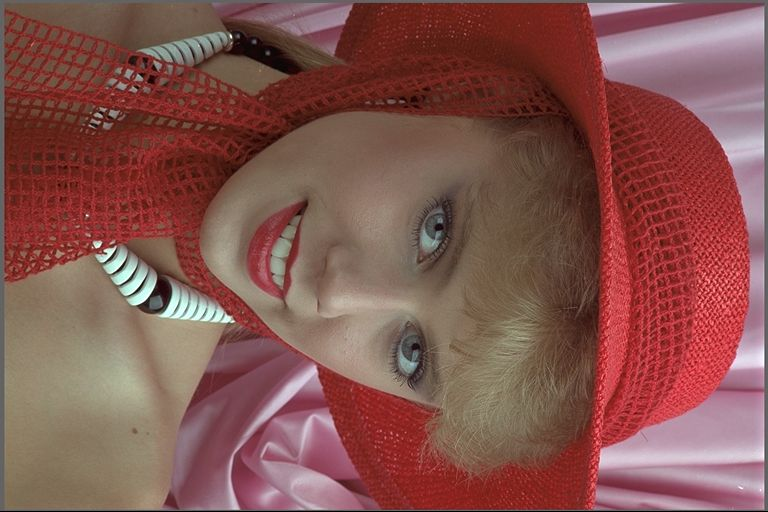
2.2.3. Описание логической структуры блок-схема алгоритма программы,

2.2.4. Используемые технические средства: компьютер на windows 10 с 8 гб ОЗУ и четырехъядерным процессором.

2.2.5. Вызов и загрузка способ вызова программы: запуск программы нажатием клавиши F5, объем оперативной памяти 8 гб, объем программы: 414 строчки кода(32.1 озу)

Входные данные:







2.2.7. Выходные данные

Изображение выглядит как текст, снимок экрана, Шрифт

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, снимок экрана, Человеческое лицо, Модный аксессуар

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как Человеческое лицо, человек, Модный аксессуар, улыбка

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, снимок экрана, мультфильм

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, Человеческое лицо, черно-белый, искусство

Контент, сгенерированный ИИ, может содержать ошибки.

Изображение выглядит как текст, снимок экрана, программное обеспечение

Контент, сгенерированный ИИ, может содержать ошибки.

- **2.3. Текст программы**

#include <iostream>

#include <opencv2/core.hpp>

#include <opencv2/imgproc.hpp>

#include <opencv2/highgui.hpp>

#include<vector>

#include<algorithm>

#include<string>

using namespace std;

using namespace cv;

Mat img, tmpImg;

int k = 0;

int s = 0;

vector <int> b;

vector <int> b1;

Point selection\_start, selection\_end;

bool is\_dragging = false;

void mousecallback(int event, int x, int y, int flags, void\* userdata)

{

Mat\* image = static\_cast<Mat\*>(userdata);

if (event == EVENT\_LBUTTONDOWN) //запоминаем начальные координаты прямоугольной области

{

selection\_start = Point(x + s, y + s);

is\_dragging = true;

}

else if (event == EVENT\_LBUTTONUP) //завершаем выбор прямоугольной области

{

selection\_end = Point(x - s, y - s);

is\_dragging = false;

Rect roi\_rect(selection\_start, selection\_end);//вырезаем выбранную область изображения

Mat roi = (\*image)(roi\_rect);

namedWindow("roi");//создаем новое окно и выводим вырезанный участок

imshow("roi", roi);

rectangle(tmpImg, Point(x - s, y - s), Point(x + s, y + s), Scalar(255, 255, 255), FILLED,0);

imshow("Paint\_img", tmpImg);

}

}

void on\_mouse(int EVENT, int x, int y, int flags, void\* ustc)

{

switch (EVENT)

{

case EVENT\_LBUTTONDOWN:

{

rectangle(tmpImg, Point(x-s,y-s), Point(x + s,y + s), Scalar(255, 255, 255), FILLED, 0);

imshow("Paint\_img", tmpImg);

}

break; }

}

Mat img3(245, 268, CV\_8UC3, Scalar(255, 255, 255));

void on\_mouse1(int EVENT, int x, int y, int flags, void\* ustc)

{

switch (EVENT)

{

case EVENT\_LBUTTONDOWN:

{

b.push\_back(x);

b.push\_back(y);

break;

}

case EVENT\_LBUTTONUP:

{

b1.push\_back(x);

b1.push\_back(y);

vector<Mat> channels{};

split(tmpImg, channels);

vector<Mat> channels1{};

split(img3, channels1);

for (int i = b[b.size() - 2], i1 = 0; i < x; i++, i1++) {

for (int j = b[b.size() - 1], j1 = 0; j < y; j++, j1++) {

channels1.at(0).at<uchar>(j1, i1) = channels.at(0).at<uchar>(j, i);

channels1.at(1).at<uchar>(j1, i1) = channels.at(1).at<uchar>(j, i);

channels1.at(2).at<uchar>(j1, i1) = channels.at(2).at<uchar>(j, i);

channels.at(0).at<uchar>(j, i) = 255;

channels.at(1).at<uchar>(j, i) = 255;

channels.at(2).at<uchar>(j, i) = 255;

}

}

merge(channels, tmpImg);

imshow("Paint\_img", tmpImg);

merge(channels1, img3);

break;

}

case EVENT\_RBUTTONDOWN: {

vector<Mat> channels{};

split(tmpImg, channels);

vector<Mat> channels1{};

split(img3, channels1);

for (int i = x, i1 = 0; i < (x + b1[b1.size() - 2] - b[b.size() - 2]); i++, i1++) {

for (int j = y, j1 = 0; j < (y + b1[b1.size() - 1] - b[b.size() - 1]); j++, j1++) {

channels.at(0).at<uchar>(j, i) = channels1.at(0).at<uchar>(j1, i1);

channels.at(1).at<uchar>(j, i) = channels1.at(1).at<uchar>(j1, i1);

channels.at(2).at<uchar>(j, i) = channels1.at(2).at<uchar>(j1, i1);

}

}

merge(channels, tmpImg);

imshow("Paint\_img", tmpImg);

break;

}

}

}

int main()

{

setlocale(LC\_ALL, "rus");

int n;

cout << "Выберите фильтр: " << endl;

cout << "0. Перемещение прямоугольной области" << endl;//

cout << "1. Вырез участка изображения" << endl;//

cout << "2. Увеличение масштаба по вертикали" << endl;//

cout << "3. СОЛЯРИЗАЦИЯ" << endl;

cout << "31. СОЛЯРИЗАЦИЯ полутонового изображения" << endl;

cout << "4. Линейная растяжка гистограммы" << endl;//

cout << "5. Пороговая сегментация" << endl;

cout << "Введите номер фильтра" << endl;

cin >> n;

switch (n)

{

case 0:

{

img = imread("18.jpg");

tmpImg = img.clone();

imshow("Paint\_img", tmpImg);

while (true)

{

setMouseCallback("Paint\_img", on\_mouse1, (void\*)(&tmpImg)); // Вызов функции обратного вызова

waitKey(0);

}

destroyWindow("Paint\_img");

break;

}

case 1:

{

img = imread("18.bmp");

tmpImg = img.clone();

imshow("Paint\_img", tmpImg);

createTrackbar("Размер ластика", "Paint\_img", &s, 100);

while (true)

{

setMouseCallback("Paint\_img", mousecallback, (void\*)(&tmpImg)); // Вызов функции обратного вызова

waitKey(0);

}

destroyWindow("Paint\_img");

break;

}

case 2:

{

img = imread("18.bmp");

tmpImg = img.clone();

Mat img1;

int k = 1;

namedWindow("Увеличение", WINDOW\_AUTOSIZE);

createTrackbar("Кратность", "Увеличение", &k, 10);

while (true)

{

if (k >= 1) {

resize(tmpImg, img1, Size(), 1, k);

imshow("Увеличение", img1);

waitKey(1);

}

else break;

}

break;

}

case 3:

{

Mat img1;

img = imread("18.bmp");

namedWindow("Trackbars");

int q11 = 10, q1m = 55, q12 = 10, q2m = 55, q13 = 10, q3m = 55;

createTrackbar("q11", "Trackbars", &q11, 255);

createTrackbar("q1m", "Trackbars", &q1m, 255);

createTrackbar("q12", "Trackbars", &q12, 255);

createTrackbar("qm2", "Trackbars", &q2m, 255);

createTrackbar("q13", "Trackbars", &q13, 255);

createTrackbar("qm3", "Trackbars", &q3m, 255);

while (true)

{

int m1 = 0, m2 = 0, m3 = 0;

img1 = img.clone();

vector<Mat> channels{};

split(img1, channels);

int k1 = q1m / q11;

int k2 = q2m / q12;

int k3 = q3m / q13;

int k11 = 0, k22 = 0, k33 = 0;

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

if (channels.at(0).at<uchar>(i, j) > m1) m1 = channels.at(0).at<uchar>(i, j);

if (channels.at(0).at<uchar>(i, j) > m2) m1 = channels.at(1).at<uchar>(i, j);

if (channels.at(0).at<uchar>(i, j) > m3) m1 = channels.at(2).at<uchar>(i, j);

}

}

if (m1 - q11 > 0 && m2 - q12 > 0 && m3 - q13 > 0)

{

k11 = (q1m - (q1m \* q11 / m1) - m1 + q11) / (m1 - q11);

k22 = (q2m - (q2m \* q12 / m2) - m2 + q12) / (m2 - q12);

k33 = (q1m - (q3m \* q13 / m3) - m3 + q13) / (m3 - q13);

}

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

if (channels.at(0).at<uchar>(i, j) < q11)

{

channels.at(0).at<uchar>(i, j) = channels.at(0).at<uchar>(i, j) \* k1;

}

if (channels.at(1).at<uchar>(i, j) < q12)

{

channels.at(1).at<uchar>(i, j) = channels.at(1).at<uchar>(i, j) \* k2;

}

if (channels.at(2).at<uchar>(i, j) < q13)

{

channels.at(2).at<uchar>(i, j) = channels.at(2).at<uchar>(i, j) \* k3;

}

if (channels.at(0).at<uchar>(i, j) >= q11)

{

channels.at(0).at<uchar>(i, j) = q1m - channels.at(0).at<uchar>(i, j) \* k11;

}

if (channels.at(1).at<uchar>(i, j) >= q12)

{

channels.at(1).at<uchar>(i, j) = q2m - channels.at(1).at<uchar>(i, j) \* k22;

}

if (channels.at(2).at<uchar>(i, j) >= q13)

{

channels.at(2).at<uchar>(i, j) = q3m - channels.at(2).at<uchar>(i, j) \* k33;

}

}

}

merge(channels, img1);

imshow("Trackbars", img1);

waitKey(1);

}

}

case 31:

{

Mat img1;

img = imread("38.bmp");

namedWindow("Trackbars");

int q11 = 10, q1m = 55;

createTrackbar("q11", "Trackbars", &q11, 255);

createTrackbar("q1m", "Trackbars", &q1m, 255);

while (true)

{

int m1 = 0;

img1 = img.clone();

cvtColor(img1, img1, COLOR\_BGR2GRAY);

int k1 =q1m / q11;

int k11 = 0;

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

if (img1.at<uchar>(i, j) > m1) m1 = img1.at<uchar>(i, j);

}

}

if (m1 - q11 > 0)

{

k11 = (q1m - (q1m \* q11 / m1) - m1 + q11) / (m1 - q11);

}

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

if (img1.at<uchar>(i, j) < q11)

{

img1.at<uchar>(i, j) = img1.at<uchar>(i, j) \* k1;

}

if (img1.at<uchar>(i, j) >= q11)

{

img1.at<uchar>(i, j) = q1m - img1.at<uchar>(i, j) ;

}

}

}

imshow("Trackbars", img1);

waitKey(1);

}

}

case 4:

{

Mat img1, img2, img4;

img = imread("18.bmp");

imshow("Hist0", img);

Mat hist;

img1 = img.clone();

img4 = img.clone();

img2 = img.clone();

cvtColor(img1, img1, COLOR\_BGR2GRAY);

cvtColor(img2, img2, COLOR\_BGR2GRAY);

cvtColor(img4, img4, COLOR\_BGR2GRAY);

equalizeHist(img1, img1);

int channels[] = { 0 };

int histSize[] = { 256 };

float hr1[] = { 0,255 };

const float\* ranges[] = { hr1 };

calcHist(&img1, 1, channels, Mat(), hist, 1, histSize, ranges, true, false);

Mat histimg(720, 1280, CV\_8UC3, Scalar(0, 0, 0));

for (int i = 0, j = 0; i < 256; i++, j += 3)

{

line(histimg, Point(j, 720), Point(j, 720 - (cvRound(hist.at<float>(i))) / 10), Scalar(255, 255, 255), 1, LINE\_AA);

line(histimg, Point(j + 1, 720), Point(j + 1, 720 - (cvRound(hist.at<float>(i))) / 10), Scalar(255, 255, 255), 1, LINE\_AA);

line(histimg, Point(j + 2, 720), Point(j + 2, 720 - (cvRound(hist.at<float>(i))) / 10), Scalar(255, 255, 255), 1, LINE\_AA);

}

imshow("Hist", histimg);

imshow("Hist1", img1);

int ma = 0, mi = 255;

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

if (img2.at<uchar>(i, j) > ma) ma = img2.at<uchar>(i, j);

if (img2.at<uchar>(i, j) < mi) mi = img2.at<uchar>(i, j);

}

}

double b = 255 / (ma - mi);

double a = -b \* mi;

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

img4.at<uchar>(i, j) = a + img2.at<uchar>(i, j) \* b;

}

}

imshow("Hist3", img4);

waitKey(0);

break;

}

case 5:

{

img = imread("19.bmp");

int hue\_min = 0, hue\_max = 0, saturation\_min = 0, saturation\_max = 0, value\_min = 0, value\_max = 0;

namedWindow("цвет", WINDOW\_AUTOSIZE);

createTrackbar("h\_min", "цвет", &hue\_min, 1000);

createTrackbar("h\_max", "цвет", &hue\_max, 1000);

createTrackbar("s\_min", "цвет", &saturation\_min, 1000);

createTrackbar("s\_max", "цвет", &saturation\_max, 1000);

createTrackbar("v\_min", "цвет", &value\_min, 1000);

createTrackbar("v\_max", "цвет", &value\_max, 1000);

while (true)

{

Mat img1;

img1 = img.clone();

cvtColor(img1, img1, COLOR\_BGR2HSV);

vector<Mat> channels{};

split(img1, channels);

for (int i = 0; i < img1.rows; i++)

{

for (int j = 0; j < img1.cols; j++)

{

if (channels.at(0).at<uchar>(i, j) > hue\_min && channels.at(0).at<uchar>(i, j) < hue\_max &&

channels.at(1).at<uchar>(i, j) > saturation\_min && channels.at(1).at<uchar>(i, j) < saturation\_max &&

channels.at(2).at<uchar>(i, j) > value\_min && channels.at(2).at<uchar>(i, j) < value\_max)

{

channels.at(0).at<uchar>(i, j) = 29;

channels.at(1).at<uchar>(i, j) = 61;

channels.at(2).at<uchar>(i, j) = 96;

}

else {

channels.at(0).at<uchar>(i, j) = 0;

channels.at(1).at<uchar>(i, j) = 0;

channels.at(2).at<uchar>(i, j) = 0;

}

}

}

merge(channels, img1);

imshow("цвет", img1);

waitKey(1);

}

break;

}

}

}

**2.4. Эксплуатационные документы**

**2.4.1. Ведомость эксплуатационных документов**

|  |  |  |  |
| --- | --- | --- | --- |
| Обозначение | Наименование | Кол.  Экз | Местонахождение |
|  | Документы на программу |  |  |
| А.В.ХХХХХ-ХХ 31 ХХ-ХХ | Обработка изображений  Описание применения | 1 | Каф. ТВ |
| А.В.ХХХХХ-ХХ 34 ХХ-ХХ | Обработка изображений  Руководство оператора | 1 | Каф. ТВ |
| А.В.ХХХХХ-ХХ 35 ХХ-ХХ | Обработка изображений  Описание языка | 1 | Каф. ТВ |

**2.4.2. Описание применения**

2.4.2.1. Назначение программы назначение: обработка изображений, возможности ПО: создание изображения и обработка, ограничения: создание и обработка текстовых файлов.

2.4.2.2. Условия применениязагрузка файлов в формате png, jpeg. Минимальный объем оперативной памяти ПК 4гб, четырехъядерный процессор, windows 10,11, место на жестком диске не менее 1 гб.

2.4.2.3. Описание задачи: изменить изображения применяя различные фильтры. Метод решения - использовать библиотеку opencv.

2.4.2.4. Входные и выходные данные: изображения.

**2.4.3. Руководство оператора**

2.4.3.1. Назначение программыназначение: обработка изображений, возможности ПО: создание изображения и обработка, ограничения: создание и обработка текстовых файлов.

2.4.3.2. Условия выполнения программыПК с 4 гб ОЗУ, установленный visualstudio, установленный c++ с библиотекой opencv.

2.4.3.3. Выполнение программы последовательность действий оператора , обеспечивающих загрузку, запуск, выполение и завершение программы, описание функций, формата, возможных вариантов команд и ответы программы на эти команды

2.4.3.4. Сообщения оператору: в случае загрузки изображения не того формата компилятор выдаст ошибку открытия изображения. В случае попытки открытия текстового файла компилятор выдаст ошибку на раннем этапе.

**2.4.4 Описание языка**

2.4.4.1. Общие сведения предназначен для решения различных задач,основными особенностями являются: использование условных операторов, различные шаблоны и функции, возможность пользования различными библиотеками, с++ применяется для создания и обработки изображений, для создания 3D проектов, используется для работы с большими объемами данных.

2.4.4.2. Элементы языка: в программах используется базовый синтаксис с++.

2.4.4.3. Средства обмена данными: стандартные файлы ввода и вывода(#include<iostream>).

2.4.4.4. Встроенные элементы: подключены библиотеки(opencv, iostream), библиотека opencv - открытая библиотека для работы с алгоритмами компьютерного зрения, машинным обучением и обработкой изображений, iostream - Заголовочный файл с классами, функциями и переменными для организации ввода-вывода в языке программирования C++.

2.4.4.5. Средства отладки программы:Пошаговое перемещение по коду и просмотр значений, хранящихся в переменных, задавать контрольные значения для переменных , чтобы отслеживать изменение значений, изучать путь выполнения кода и тд.

**заключение**

В ходе выполнения курсовой работы было разработано программное обеспечение для редактирования изображений, отладка программного обеспечения. Была запущена программа, написанная на языке программирования С++ с применением фильтров и использовании библиотеки opencv.

**список использованных источников**

***Ниже представлены примеры библиографического описания, В качестве названия источника в примерах приводится вариант, в котором применяется то или иное библиографическое описание.***

1. Книга двухавторов/ О.Д. Суарес, Г.Б. Гарсия. : Издательство visilab, 2012. 206 с.

2. Книга двух авторов / А.Кэлер, Г. Брэдски. : Издательство ДМК, 2017. 827 с.