**Цель работы**

Реализовать программно в среде MATLAB: алгоритмы медианной и пороговой фильтрации, а также градиентной фильтрации, используя операторы Робертса и Собеля. Научиться работать с изображениями в среде MATLAB.

**Краткие теоретические сведения**

Под изображением будем понимать многомерный сигнал, предназначенный для зрительного восприятия.

Обработка изображений в широком смысле слова означает выполнение различных операций над многомерными сигналами. Преследуемые цели могут быть отнесены к улучшению изображений, эффективной кодировке, распознаванию, машинной графике.

Предварительная обработка направлена на облегчение дальнейших этапов технического зрения и проводится в целях коррекции различных искажений, удаления шумов и т.д.

**Методы обработки в пространственной области.**

**Пороговый метод**

С помощью него ликвидируются пики освещённости, т.е. изображение сглаживается. Суть в том, что последовательно проверяется цифровое представление яркости всех элементов q(i,j), и если яркость центрального элемента анализируемой группы из элементов NxN (где N – чётное: 3, 5,7) превышает яркость группы на заданное пороговое значение η, то его яркость заменяется на среднюю яркость группы Gср, что, в конечном итоге, приводит к ликвидации пиков освещённости, т.е. изображение сглаживается.

Расчёт среднего значения яркости группы окна можно провести по формуле:

.

**Алгоритм медианной фильтрации**

Данный алгоритм может эффективно применяться при устранении импульсного шума.

Апертура своим центром посещает каждый пиксел изображения. Элемент изображения, попавший в центр окна заменяется медианным значением, вычисленным из массива, образованного всеми элементами изображения, попавшими в окно NxN (где N – чётное: 3, 5,7).

Медианой *упорядоченной* последовательности A(i), i = 1,2,…N является значение:

A[(N+1)/2], если N – нечётное;

A[A(N/2)+A((N+2)/2)]/2, если N – чётное.

**Градиентная фильтрация**

Для нахождения контура объекта применяется градиентная фильтрация.

Градиент изображения g(x,y) – это двумерный вектор, указывающий направление максимального изменения функции яркости g(x,y) в точке (x,y).

Для получения контурного рисунка, требуется оценка величины модуля градиента функции. Эту величину можно вычислить, если известны производные этой функции по каким либо двум ортогональным направлениям и способ приближённого вычисления производной (одномерной), чтобы иметь все необходимые составляющие алгоритма. Операция взятия градиента преобразует исходное изображение в новое с подчёркнутыми границами. Заключительной операцией выделения элементов является сравнение с пороговым значением r\*.

**Оператор Робертса**

Наиболее известным методом оценки градиента является этот метод, основанный на анализе в окне 2х2 элемента. При этом модуль градиента в точке (i,j) аппроксимируется:

Здесь g(i,j) – интенсивность элемента с координатами (i,j).

**Оператор Собеля**

Использование окна 3х3 элемента даёт более точные результаты.

Определим для каждого элемента изображения в точке (i,j) две составляющие оценки локального градиента яркости по горизонтальной – x и вертикальной – y координатам.

Тогда градиент в точке (i,j) можно определить как

**Интерфейс программы**

В среде MATLAB напишем программу, которая будет представлять интерфейс для загрузки, обработки и сохранения изображения. Она будет иметь функции наложения шумов различных типов, фильтрации шумов, взятия градиента.

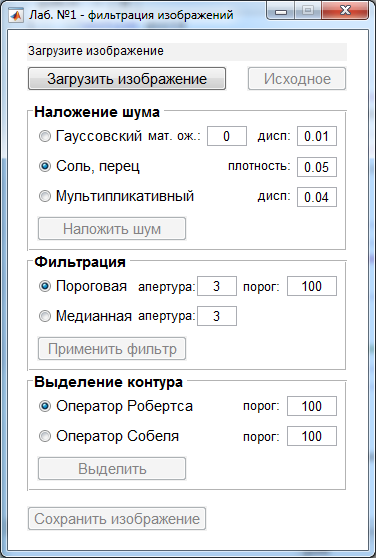


Рисунок 1 — Интерфейс программы.

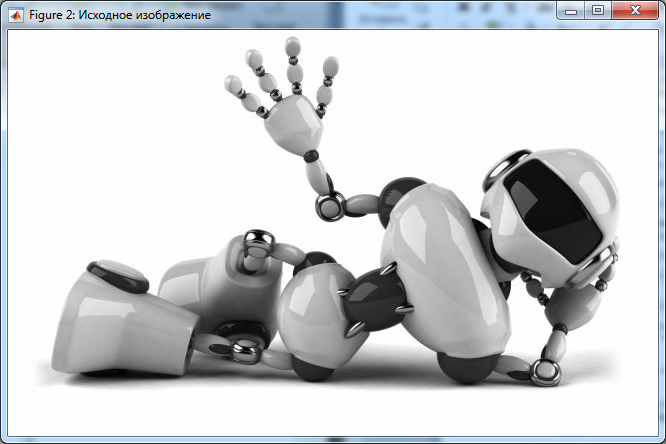


Рисунок 2 — Исходное изображение.

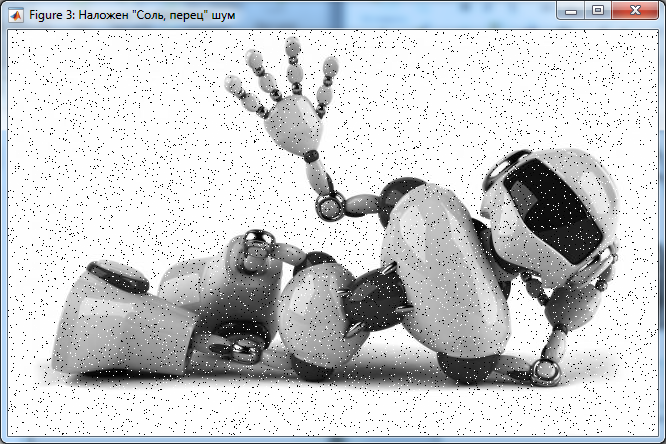


Рисунок 3 — Изображение с наложенным шумом типа «соль, перец», с параметрами: плотность 0.05.

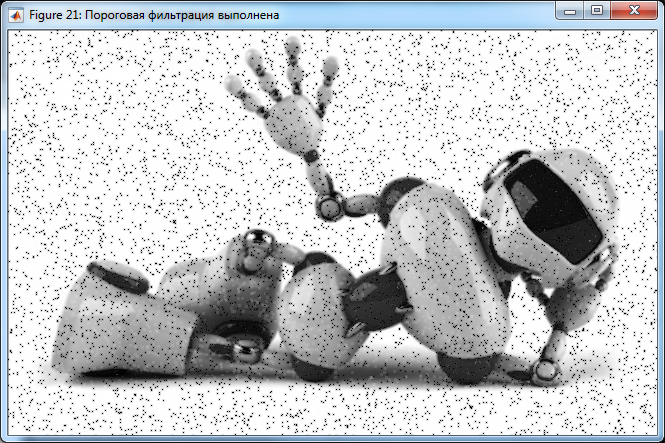


Рисунок 4 — Результат работы медианного фильтра с апертурой 3х3, порогом 100.

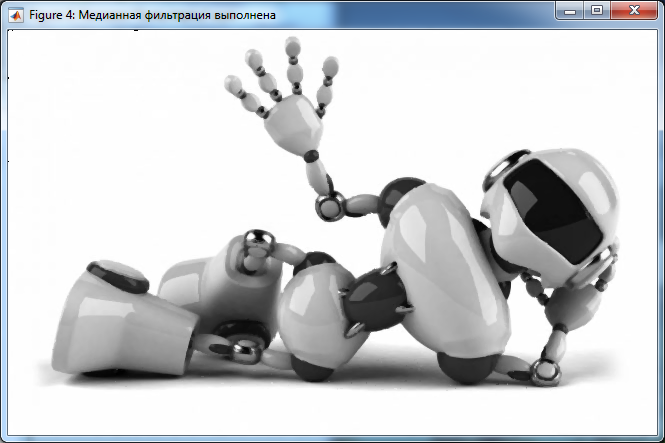


Рисунок 5 — Результат работы порогового фильтра с апертурой 3х3.

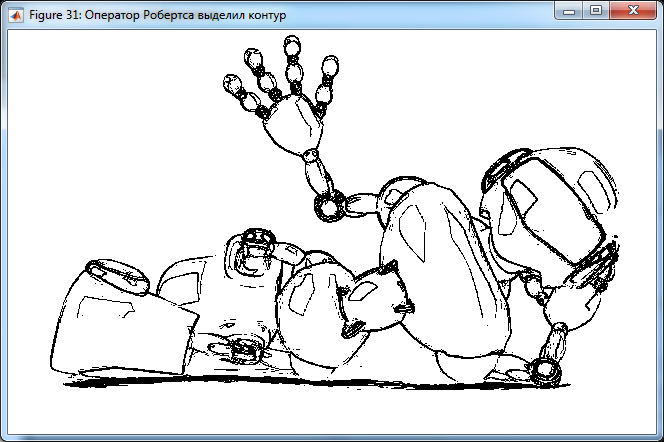


Рисунок 6 — Выделение контура с помощью оператора Робертса c порогом 90

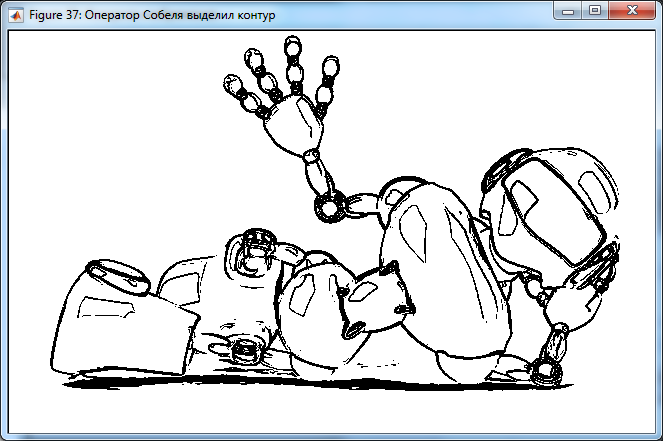


Рисунок 7 — Выделение контура с помощью оператора Собеля с порогом 90.

**ПРИЛОЖЕНИЕ**

**Листинг MATLAB**

**main.m**

function guitk

clc; clear; close all;

% построение интерфейса

mainfig = figure('Name','Лаб. №1 - фильтрация изображений',...

'Color','w','MenuBar','none',...

'Position',[10 60 360 520],...

'Resize','off',...

'NumberTitle','off',...

'Tag','mainfig');

% кнопка загрузки изображения

uicontrol('Style','pushbutton',...

'Position',[20 460 200 25],...

'String','Загрузить изображение',...

'FontSize',11,...

'Callback',@loadBtnCallback,...

'Tag','loadBtn');

% кнопка отмены всех преобразования

uicontrol('Style','pushbutton',...

'Position',[240 460 100 25],...

'String','Исходное',...

'FontSize',11,...

'Enable','off',...

'Callback',@loadDefIBtnCallback,...

'Tag','loadDefIBtn');

% статус бар

uicontrol('Style','text',...

'Position',[20 490 320 18],...

'FontSize',9,...

'BackgroundColor',[0.95 0.95 0.95 0.95],...

'HorizontalAlignment','left',...

'String','Загрузите изображение',...

'Tag','statusBar');

% блок шума

bgNoise = uibuttongroup('Units','pixels',...

'Title','Наложение шума',...

'FontSize',11,...

'FontWeight','bold',...

'BackgroundColor','w',...

'Position',[20 300 320 150],...

'Tag','bgNoise');

uicontrol(bgNoise,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Гауссовский',...

'FontSize',11,...

'Position',[10 100 200 30],...

'HandleVisibility','off',...

'Tag','gaussBgNoise');

defSet = uicontrol(bgNoise,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Соль, перец',...

'FontSize',11,...

'Position',[10 70 200 30],...

'HandleVisibility','off',...

'Tag','salpepBgNoise');

uicontrol(bgNoise,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Мультипликативный',...

'FontSize',11,...

'Position',[10 40 200 30],...

'HandleVisibility','off',...

'Tag','speckBgNoise');

bgNoise.SelectedObject = defSet;

% кнопка наложения шума

uicontrol('Style','pushbutton',...

'Position',[30 310 150 25],...

'String','Наложить шум',...

'FontSize',11,...

'Callback',@noiseBtnCallback,...

'Tag','noiseBtn',...

'Enable','off');

uicontrol('Style','text',...

'String','мат. ож.:',...

'Position',[140 399 80 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

% мат. ожидание

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[200 405 40 20],...

'Tag','meanGauss',...

'String',0);

uicontrol('Style','text',...

'String','дисп:',...

'Position',[250 399 40 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

% дисперсия

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[290 405 40 20],...

'Tag','varGauss',...

'String',0.01);

uicontrol('Style','text',...

'String','плотность:',...

'Position',[220 369 90 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

% плотность

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[290 374 40 20],...

'Tag','densitySaltPep',...

'String',0.05);

uicontrol('Style','text',...

'String','дисп:',...

'Position',[250 339 40 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

% дисперсия

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[290 344 40 20],...

'Tag','varSpeck',...

'String',0.04);

% блок фильтрации

bgFilt = uibuttongroup('Units','pixels',...

'Title','Фильтрация',...

'FontSize',11,...

'FontWeight','bold',...

'BackgroundColor','w',...

'Position',[20 180 320 120],...

'Tag','bgFilt');

uicontrol(bgFilt,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Пороговая',...

'FontSize',11,...

'Position',[10 70 200 30],...

'HandleVisibility','off',...

'Tag','threshBgFilt');

uicontrol(bgFilt,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Медианная',...

'FontSize',11,...

'Position',[10 40 200 30],...

'HandleVisibility','off',...

'Tag','medBgFilt');

% кнопка применения фильтра

uicontrol('Style','pushbutton',...

'Position',[30 190 150 25],...

'String','Применить фильтр',...

'FontSize',11,...

'Callback',@filterBtnCallback,...

'Tag','filterBtn',...

'Enable','off');

uicontrol('Style','text',...

'String','апертура:',...

'Position',[130 248 60 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[190 255 40 20],...

'Tag','apertThresh',...

'String',3);

uicontrol('Style','text',...

'String','порог:',...

'Position',[235 248 40 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

uicontrol('Style','edit',...

'FontSize', 10,...

'Position', [280 255 50 20],...

'Tag','threshThresh',...

'String',100);

uicontrol('Style','text',...

'String','апертура:',...

'Position',[130 219 60 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[190 225 40 20],...

'Tag','apertMed',...

'String',3);

% блок выделения контура

bgCont = uibuttongroup('Units','pixels',...

'Title','Выделение контура',...

'FontSize',11,...

'FontWeight','bold',...

'BackgroundColor','w',...

'Position',[20 60 320 120],...

'Tag','bgCont');

uicontrol(bgCont,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Оператор Робертса',...

'FontSize',11,...

'Position',[10 70 200 30],...

'HandleVisibility','off',...

'Tag','robBgCont');

uicontrol(bgCont,...

'Style','radiobutton',...

'BackgroundColor','w',...

'String','Оператор Собеля',...

'FontSize',11,...

'Position',[10 40 200 30],...

'HandleVisibility','off',...

'Tag','sobBgCont');

% кнопка выделения контура

uicontrol('Style','pushbutton',...

'Position',[30 70 150 25],...

'String','Выделить',...

'FontSize',11,...

'Callback',@contBtnCallback,...

'Tag','contBtn',...

'Enable','off');

uicontrol('Style','text',...

'String','порог:',...

'Position',[235 129 40 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[280 135 50 20],...

'Tag','threshRob',...

'String',100);

uicontrol('Style','text',...

'String','порог:',...

'Position',[235 98 40 25],...

'FontSize',10,...

'BackgroundColor','w',...

'HorizontalAlignment','left');

uicontrol('Style','edit',...

'FontSize',10,...

'Position',[280 105 50 20],...

'Tag','threshSob',...

'String',100);

% кнопка сохранения изображения

uicontrol('Style','pushbutton', ...

'Position',[20 20 180 25],...

'String','Сохранить изображение',...

'FontSize',11,...

'Callback',@saveBtnCallback,...

'Tag','saveBtn',...

'Enable','off');

handles = guihandles(mainfig);

handles.ud = struct(); % пустая структура для пользовательских данных

guidata(mainfig,handles);

end

% загрузка изображения (в handles.ud.I)

function loadBtnCallback(src,evt)

handles = guidata(src);

[fileName,pathName] = uigetfile({'\*.jpg; \*.jpeg; \*.png; \*.bmp; \*.gif; \*.tif; \*.tiff'},...

'Выбор изображения');

if ((fileName ~= 0) | (pathName ~= 0)) % если задан путь

if exist([pathName fileName], 'file') == 2 % если существует файл

handles.statusBar.String = 'Изображение загружается...';

I = imread([pathName fileName]);

dimOfI = size(size(I));

if dimOfI(2) ~= 2 % если не чб

I = rgb2gray(I);

end

I = im2double(I);

if ~isfield(handles.ud,'I') % если загружается в первый раз

handles.filterBtn.Enable = 'on';

handles.noiseBtn.Enable = 'on';

handles.contBtn.Enable = 'on';

handles.saveBtn.Enable = 'on';

handles.loadDefIBtn.Enable = 'on';

end

figure('Name','Исходное изображение',...

'Color','w','MenuBar','none');

imshow(I,'Border','tight');

handles.ud.I = I;

handles.ud.defI = I;

handles.ud.pathName = pathName;

handles.ud.filename = fileName;

handles.statusBar.String = [pathName fileName];

handles.statusBar.TooltipString = [pathName fileName];

else

errordlg('Файл не существует','Ошибка');

end

end

guidata(src,handles);

end

% отмена всех преобразований (Исходное)

function loadDefIBtnCallback(src,evt)

handles = guidata(src);

handles.ud.I = handles.ud.defI;

figure('Name','Исходное изображение',...

'Color','w','MenuBar','none');

imshow(handles.ud.I,'Border','tight');

handles.statusBar.String = 'Возвращено исходное изображение';

guidata(src,handles);

end

% зашумление

function noiseBtnCallback(src,evt)

handles = guidata(src);

% тип шума

switch handles.bgNoise.SelectedObject.Tag

case 'gaussBgNoise'

[handles,gaussMean,err1] = getAndCheckString(handles,'meanGauss');

[handles,gaussVar,err2] = getAndCheckString(handles,'varGauss');

if (err1 == 1) | (err2 == 1); guidata(src,handles); return; end

handles.ud.I = imnoise(handles.ud.I,'gaussian',gaussMean,gaussVar);

case 'salpepBgNoise'

[handles,salpepDens,err] = getAndCheckString(handles,'densitySaltPep');

if err == 1; guidata(src,handles); return; end

handles.ud.I = imnoise(handles.ud.I,'salt & pepper',salpepDens);

case 'speckBgNoise'

[handles,speckVar,err] = getAndCheckString(handles,'varSpeck');

if err == 1; guidata(src,handles); return; end

handles.ud.I = imnoise(handles.ud.I,'speckle',speckVar);

end

handles.statusBar.String = ['Наложен "' handles.bgNoise.SelectedObject.String '" шум'];

figure('Name',['Наложен "' handles.bgNoise.SelectedObject.String '" шум'],...

'Color','w','MenuBar','none');

imshow(handles.ud.I,'Border','tight');

guidata(src,handles);

end

% фильтрация

function filterBtnCallback(src,evt)

handles = guidata(src);

switch handles.bgFilt.SelectedObject.Tag

case 'threshBgFilt'

[handles,A,err1] = getAndCheckString(handles,'apertThresh');

[handles,thresh,err2] = getAndCheckString(handles,'threshThresh');

if (err1 == 1) | (err2 == 1); guidata(src,handles); return; end

[handles,A] = checkApert(handles,A,'apertThresh');

[handles,thresh] = checkThresh(handles,thresh,'threshThresh');

handles.statusBar.String = 'Выполняется пороговая фильтрация...';

drawnow;

% пороговый фильтр

handles.ud.I = myThreshFilt(handles.ud.I,A,thresh);

case 'medBgFilt'

[handles,A,err] = getAndCheckString(handles,'apertMed');

if err == 1; guidata(src,handles); return; end

[handles,A] = checkApert(handles,A,'apertMed');

handles.statusBar.String = 'Выполняется медианная фильтрация...';

drawnow;

% медианный фильтр

handles.ud.I = myMedFilt(handles.ud.I,A);

end

handles.statusBar.String = [handles.bgFilt.SelectedObject.String ' фильтрация выполнена'];

figure('Name',[handles.bgFilt.SelectedObject.String ' фильтрация выполнена'],...

'Color','w','MenuBar','none');

imshow(handles.ud.I,'Border','tight');

guidata(src,handles);

end

% выделение контура

function contBtnCallback(src,evt)

handles = guidata(src);

% тип выделения

switch handles.bgCont.SelectedObject.Tag

case 'robBgCont'

[handles,thresh,err] = getAndCheckString(handles,'threshRob');

if err == 1; guidata(src,handles); return; end

[handles,thresh] = checkThresh(handles,thresh,'threshRob');

handles.statusBar.String = 'Выполняется выделение контура (Робертс)...';

drawnow;

% робертса

handles.ud.I = myRobCont(handles.ud.I,thresh);

case 'sobBgCont'

[handles,thresh,err] = getAndCheckString(handles,'threshSob');

if err == 1; guidata(src,handles); return; end

[handles,thresh] = checkThresh(handles,thresh,'threshSob');

handles.statusBar.String = 'Выполняется выделение контура (Собель)...';

drawnow;

% собеля

handles.ud.I = mySobCont(handles.ud.I,thresh);

end

handles.statusBar.String = [handles.bgCont.SelectedObject.String ' выделил контур'];

figure('Name',[handles.bgCont.SelectedObject.String ' выделил контур'],...

'Color','w','MenuBar','none');

imshow(handles.ud.I,'Border','tight');

guidata(src,handles);

end

% сохранение последнего изображения

function saveBtnCallback(src,evt)

handles = guidata(src);

[fileName,pathName] = uiputfile({'\*.png'},...

'Сохранить как...',...

['Обработанное ',handles.ud.filename]);

if ((fileName ~= 0) | (pathName ~= 0)) % если задан путь

imwrite(handles.ud.I,[pathName fileName]);

handles.statusBar.String = 'Последнее преобразованное изображение сохранено';

end

guidata(src,handles);

end

% проверка валидности данных в edit (если err == 1 - не соотв)

function [handles,res,err] = getAndCheckString(handles,tag)

str = handles.(tag).String;

if errStr(str) == 1 % если ошибки есть

handles.(tag).BackgroundColor = [1 0.5 0.5];

err = 1;

else

handles.(tag).BackgroundColor = [0.94 0.94 0.94];

err = 0;

end

res = str2double(str);

end

% проверка валидности строки (если err == 1 - не соотв)

function err = errStr(str)

spl = regexp(str,'^[0-9]\*[.]?[0-9]+$','Match');

err = isempty([spl{:}]);

end

% проверка валидности размера апертуры

function [handles,A] = checkApert(handles,A,tag)

minSizeA = min(size(handles.ud.I));

if A > minSizeA

A = minSizeA;

handles.(tag).String = minSizeA;

end

end

% проверка валидности уровня порога

function [handles,T] = checkThresh(handles,T,tag)

if T > 255

T = 255;

handles.(tag).String = 255;

end

end

%-Алгоритмы----------------------------------------------------------------

% пороговая фильтрация

function P = myThreshFilt(I,A,T)

modifyI = zeros(size(I)+ceil(A/2));

for x = 1:size(I,1)

for y = 1:size(I,2)

modifyI(x+floor(A/2),y+floor(A/2)) = I(x,y);

end

end

for i = 1:size(modifyI,1)-A

for j = 1:size(modifyI,2)-A

frame = zeros(A^2,1);

inc = 1;

for x = 0:A-1

for y = 0:A-1

frame(inc) = modifyI(i+x,j+y);

inc = inc+1;

end

end

mean = sum(frame)/(inc - 1);

centerA = frame(ceil((A^2)/2),1);

if centerA > T/255

P(i,j) = mean;

else

P(i,j) = centerA;

end

end

end

end

% медианная фильтрация

function P = myMedFilt(I,A)

modifyI = zeros(size(I)+ceil(A/2));

for x = 1:size(I,1)

for y = 1:size(I,2)

modifyI(x+floor(A/2),y+floor(A/2)) = I(x,y);

end

end

for i = 1:size(modifyI,1)-A

for j = 1:size(modifyI,2)-A

frame = zeros(A^2,1);

inc = 1;

for x = 0:A-1

for y = 0:A-1

frame(inc) = modifyI(i+x,j+y);

inc = inc+1;

end

end

med = sort(frame);

P(i,j) = med(ceil(A^2/2));

end

end

end

% выделение контура, оператор Робертса

function P = myRobCont(g,T)

[m,n]=size(g);

for i=1:m-2

for j=1:n-2

R = sqrt((g(i+1, j+1) - g(i,j))^2 + (g(i,j+1) - g(i+1,j))^2);

if T/255 >= R

P(i,j) = 255;

else

P(i,j) = 0;

end

end

end

end

% выделение контура, оператор Собеля

function P = mySobCont(g,T)

[m,n]=size(g);

for i=2:m-3

for j=2:n-3

Sx = (g(i-1,j+1) + 2\*g(i,j+1) + g(i+1,j+1)) - (g(i-1,j-1) + 2\*g(i,j-1) + g(i+1,j-1));

Sy = (g(i+1,j-1) + 2\*g(i+1,j) + g(i+1,j+1)) - (g(i-1,j-1) + 2\*g(i-1,j) + g(i-1,j+1));

S = sqrt(Sx^2 + Sy^2);

if T/255 >= S

P(i,j) = 255;

else

P(i,j) = 0;

end

end

end

end