ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

УПРАВЛЕНИЕ ЦИФРОВЫХ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

Кафедра «Приборостроение и биомедицинская инженерия»

Учебно-методическое пособие

по дисциплине

«Проектирование оптико-электронных приборов и систем»

Авторы

Цыбрий И. К.,

Пахомов И. В.

Ростов-на-Дону, 2019

Аннотация

Учебно-методическое пособие предназначено для студентов очной формы обучения направления 12.04.01 Приборостроение.

Авторы

к.т.н, доцент кафедры
«Приборостроение и
биомедицинская инженерия»
Цыбрий И.К.
к.т.н., ассистент кафедры
«Приборостроение и
биомедицинская инженерия»
Пахомов И.В.

Оглавление

ЗАДАНИЕ 4
1 РАЗРАБОТКА ПРОГРАММЫ....4
1.1 Особенности тепловизионных изображений....4
1.2 Описание функций и кодов программы 5
1.2.1 Меню «Файл» 6
1.2.2 Меню «Изменить класс»....10
1.2.3 Меню «Функции» 12
1.2.4 Меню «Наложение шумов»....26
1.2.5 Меню «Фильтрация» 27
1.2.6 Меню «Арифметические операции» 33
1.2.7 Меню «Статистика» 37
1.2.8 Меню «Справка» 39
ТЕКСТ ПРОГРАММЫ ОБРАБОТКИ ТЕПЛОВИЗИОННЫХ ИЗОБРАЖЕНИЙ 41
Список литературы 73

ЗАДАНИЕ

на практическую работу «Обработка тепловизионных изображений в программе MATLAB»

1. Открыть файл исходного тепловизионного изображения;
2. Преобразовать файл в массив для обработки в MATLAB;
3. Осуществить следующие преобразования исходного изображения с помощью встроенных функций: увеличение, поворот, сегментирование, зеркальное отображение, пределение края, построить контурного графика, инверсию цветов, гамма-коррекцию, пороговое преобразование;
4. Наложить белоый шум на исходное изображение;
5. Произвести линейную пространственную фильтрацию;
6. Произвести арифметические операции с матрицей изображения;
7. Составить статистику по обрабатываемому изображению;
8. Вывести информацию об обрабатываемом изображении.

1 РАЗРАБОТКА ПРОГРАММЫ

1.1 ОСОБЕННОСТИ ТЕПЛОВИЗИОННЫХ ИЗОБРАЖЕНИЙ

В ходе выполнения практической работы будет разработана программа обработки тепловизионных изображений в программном пакете MATLAB [1-6].

Тепловизионные изображения в большинстве случаев представлены в монохромном виде так, как в этом виде наиболее различим тепловой контраст «объект-фон», поэтому большая часть реализованных функций направлена на обработку монохромных восьми битных изображений. Для этого в разрабатываемой программе реализована функция перевода цветного изображения в монохромное, что позволяет расширить функциональные возможности программы. Данная программа поддерживает наиболее распространенные графические форматы, что делает её универсальной.

1.2 ОПИСАНИЕ ФУНКЦИЙ И КОДОВ ПРОГРАММЫ

Графическое окно программы имеет вид, показанный на рис. 1.1.

Меню «Файл» состоит из следующих элементов: «Открыть», «Открыть DICOM-изображение», «Открыть в новом окне», «Сохранить как…», «Исходное», «Изменить размер», «Преобразовать в монохромное», «Печать», «Выход».

Меню «Изменить класс» содержит: «uint8», «uint16», «double», «logical».

Меню «Функции» содержит: «Увеличить», «Повернуть», «Сегмент», «Зеркало», «Определить края», «Контурный график», «Инверсия», «Гамма-коррекция», «Выравнивание гистограммы», «Растяжение контрастности», «Логарифмическое преобразование», «Пороговое преобразование». «Отобразить выше пороговой яркости», «Изменить палитру».

Меню «Наложение шума» состоит из: «Белый гауссовский шум», «Пуассоновский шум», «Шум типа «Соль и перец».

Меню «Фильтрация» состоит из: «Линейная фильтрация», «Ранговая фильтрация», «Медианная фильтрация», «Адаптивная фильтрация Винера».

Меню «Арифметические операции» содержит: «Возвести в квадрат», «Сложить изображения», «Перемножить изображения», «Вычесть из изображения X изображение У», «Из исходного вычесть отфильтрованное», «Вычесть из изображения константу», «Разделить изображение X на изображение У», «Разделить изображение на константу».

Меню «Статистика» состоит из: «Гистограмма», «Среднее значение яркости», «Стандартное отклонение», «Двумерный коэффициент корреляции».

Меню «Справка» содержит элементы «Информация об изображении» и «О программе».

Текст программы приведен в приложении А.

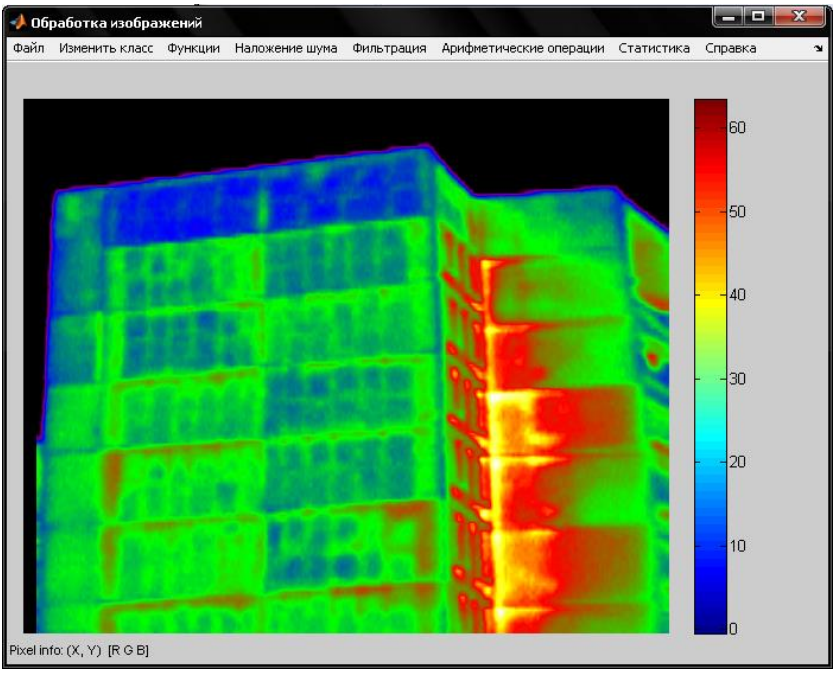

Рисунок – 1.1 – Общий вид окна программы обработки тепловизионных изображений

1.2.1 МЕНЮ «ФАЙЛ»

Чтобы открыть файл изображения, имеющий расширение tif, tiff, jpg, jpeg, gif, bmp, png или xwd, необходимо выбрать пункт «Открыть». При этом вызывается Callback-функция «openfile», которая имеет следующий синтаксис:

```
[file_name,file_path]=uigetfile('*.*','Выберите файл')
save imaging;
I=imread([file_path,file_name]);
set(gca,'Position',[0.02 0.03 0.91 0.93]);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

Команда uigetfile вызывает стандартное диалоговое окно выбора имени файла для чтения. После выбора необходимого файла, его

имя и расширение записываются в переменные file_name и file_path соответственно, а также сохраняются во временный файл «imaging.mat» для быстрого изъятия при выполнении некоторых предстоящих операций. Команда imread считывает указанный файл и сохраняет его в виде матрицы под переменной «I». Команда imshow отображает выбранное изображение в рабочей области графического окна. Для отображения панели цветов справа от изображения используется команда colorbar. Команда impixelinfo отображает в левом нижнем углу графического окна информацию о пикселе (координаты и значение яркости), на который наведен указатель. Для удобства при выполнении дальнейших операций над данным изображением, оно сохраняется во временные файлы «picture.bmp», «picturegam.bmp», «picturesave.bmp».

В данной программе реализована поддержка изображений стандарта DICOM. Для этого необходимо выбрать пункт «Открыть DICOM-изображение». Эта операция реализуется с помощью Callback-функции «opendicom»:

```
[file_name,file_path]=uigetfile('*.*','Выберите файл (*.dcm, *.dic)')
I=dicomread([file_path,file_name]);
set(gca,'Position',[0.02 0.03 0.91 0.93]);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

При этом необходимо выбрать файл с разрешением dcm или dic.

При выборе пункта «Открыть в новом окне» текущее изображение открывается в отдельном окне, в котором отсутствуют элементы управления. Эта функция полезна при визуальном сравнении изображений. Для удобства, изображение можно увеличивать с помощью манипулятора типа «мышь». При выборе данного пункта выполняется Callback-функция «opennfg»:

```
figure('NumberTitle','on','WindowStyle','normal', 'MenuBar', 'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
I=imread('picturesave.bmp');
imshow(I, []);
colorbar('vert');
impixelinfo;
zoom;
```

Команда figure создает новое графическое окно с указанными в

скобках параметрами. Команда zoom позволяет увеличить данное изображение в 2 раза при каждом нажатии левой кнопки манипулятора, либо при варащении колесика.
Пункт «Сохранить как…» позволяет сохранить текущее изображение в указанную директорию. При этом необходимо ввести имя файла с расширением tif, tiff, jpg, jpeg, gif, bmp, png или xwd. Вызывается Callback-функция «save»:

```
[file_name,file_path]=uiputfile('*.*','Сохранить изображение как...')
S=imread('picturesave.bmp');
imwrite(S,[file_path,file_name]);
```

Команда uiputfile вызывает стандартное диалоговое окно выбора имени файла для записи. Введенные имя файла и расширение сохраняются под переменными file_name и file_path соответственно. После этого изображение, хранящееся во временном файле «picturesave.bmp», записывается в указанный файл командой imwrite.
Пункт «Исходное» загружает изображение, которое было открыто с помощью меню «Файл» «Открать». Эта функция возвращает текущее изображение к первоначальному виду.
Callback-функция «source» имеет следующий синтаксис:

```
load imaging
I=imread([file_path,file_name]);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

Пункт «Изменить размер» позволяет увеличить или уменьшить разрешение изображения в заданное число раз. Для этого необходимо ввести с клавиатуры фактор (желательно в пределах от 0.1 до 10) на который необходимо умножить высоту и ширину изображения. При этом вызывается Callback-функция «resizeimage»:

```
I=imread('picture.bmp');
D=inputdlg('Введите множитель(от 0.1 до 10)', 'Введите множитель');
A=cell2mat(D);
B=str2double(A);
J=imresize(I, B, 'bicubic');
imshow(J, []);
colorbar('vert');
```

```
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Команда inputdlg вызывает стандартное диалоговое окно для ввода строк. Введенное значение сохраняется под переменной D и имеет тип «cell». Перед дальнейшим использованием переменную D необходимо сначала преобразовать в тип «mat» командой cell2mat, а затем в тип «double» командой str2double. Собственно изменение размера изображения осуществляется командой imresize.

Большинство алгоритмов обработки изображений, используемых в данном приложении, применимы к монохромным изображениям классов «uint8» и «double». Поэтому, в случае необходимости, можно преобразовать исходное цветное изображение в монохромное, выбрав пункт «Преобразовать в монохромное». При этом вызывается Callback-функция «convertinmonohrom»:

```
load imaging;
I=imread([file_path,file_name]);
J=rgb2gray(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Команда load загружает из файла «imaging.mat» ранее сохраненные в нем имя и расширение открытого изображения. Команда rgb2gray преобразует матрицу цветного изображения в матрицу монохромного класса «uint8» (т. е. 256 оттенков серого цвета).

Для вывода изображения на печать необходимо выбрать пункт «Печать». При этом командой printdlg вызовется стандартное диалоговое окно настройки параметров печати.

Для выхода из приложения необходимо выбрать пункт «Выход». При этом появится стандартное диалоговое окно с запросом подтверждения, которое вызывается командой questdlg. В случае нажатия кнопки «Yes» приложение и все открытые в нем окна закроются, а так же будут удалены все временные файлы (picture.bmp, picturegam.bmp, picturesave.bmp, picturefilt.bmp, imaging.mat). Callback-функция «close» имеет следующий вид:

```
a = questdlg('Несохраненные данные будут потеряны. Продолжить ?','Выход','Yes','No','Yes');
```

```
if a=='Yes'
    delete picture.bmp;
    delete picturegam.bmp;
    delete picturesave.bmp;
    delete picturefilt.bmp;
    delete imaging.mat;
    delete(gcf);
    close all;
else
    return;
end
```

Команда delete удаляет указанные файлы.
Такой же эффект будет при нажатии кнопки с крестиком в верхнем правом углу графического окна. При этом вызовется Callback-функция «figure1_CloseRequestFcn» с аналогичным синтаксисом.

1.2.2 МЕНЮ «ИЗМЕНИТЬ КЛАСС»

В операциях с изображениями в Matlab используются четыре основных класса данных:
- класс «double» - вещественные числа с плавающей запятой двойной точности в диапазоне, примерно, от —10-308 до 10308 (8 байт на число);
- класс «uint8» - целые числа без знака в интервале от 0 до 255 (1 байт на число);
- класс «uint16» - целые числа без знака в интервале от 0 до 65535 (2 байта на число);
- класс «logical» - значения 0 или 1 (1 байт на элемент)

Для преобразования данного массива изображения в масив двойной точности необходимо выбрать пункт «double». При этом вызывается Callback-функция «classdoub»:

```
J=imread('picturesave.bmp');
I=im2double(J);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

Преобразование выполняет команда im2double.
Для преобразования данного массива изображения в восьми бит-

ный целый массив без знака необходимо выбрать пункт «uint8». При этом вызывается Callback-функция «class»:

```
J=imread('picturesave.bmp');
I=im2uint8(J);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

Преобразование выполняет команда im2uint8.

Для преобразования данного массива изображения в шестнадцати битный целый массив без знака необходимо выбрать пункт «uint16». При этом вызывается Callback-функция «class16»:

```
J=imread('picturesave.bmp');
I=im2uint16(J);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

Преобразование выполняет команда im2uint16.

Для преобразования данного изображения в двоичное с помощью пороговой обработки необходимо выбрать пункт «logical». При этом все пиксели, яркость которых меньше пороговой примут значение «0» (черный цвет), а все остальные – «1» (белый цвет). Уровень пороговой яркости определяется с помощью наведения указателя на область изображения, яркость которой считается равной пороговой, и нажатием правой кнопки манипулятора (либо двойное нажатие левой кнопки). Callback-функция «classlogoc» имеет следующий синтаксис:

```
J=imread('picturesave.bmp');
I=im2uint16(J);
P=impixel(I);
n=max(P);
m=n/65535;
B=im2bw(I, m);
imshow(B, []);
colorbar('vert');
impixelinfo;
imwrite(B,'picture.bmp');
```

```
imwrite(B,'picturegam.bmp');
imwrite(B,'picturesave.bmp');
```

Преобразование выполняет команда im2bw, с указанием порога m. Команда impixel возвращает переменной P уровень яркости выбранного с помощью указателя пикселя. Так как Matlab при преобразовании оперирует значениями пороговой яркости, представленными в классе double, то полученное значение яркости необходимо разделить на 65535.

1.2.3 МЕНЮ «ФУНКЦИИ»

При выборе пункта «Увеличить» активируется режим увеличения изображения. Увеличение осуществляется путем наведения указателя на изображение и нажатием левой кнопки манипулятора типа «мышь», либо вращением колесика. Кратность увеличения равняется двум, при этом при использовании левой кнопки манипулятора, точка, на которую был наведен указатель, станет центральной точкой увеличенного изображения, если же для увеличения использовалось колесико, то центр увеличенного изображения будет совпадать с центром исходного. В случае необходимости увеличения конкретной области изображения на все рабочее поле графического окна, нужно нажав и удерживая левую кнопку манипулятора выделить эту область перетаскиванием курсора. Для возвращения изображения в исходное состояние необходимо нажать правую кнопку манипулятора и в появившемся контекстном меню выбрать пункт «Reset to original view», либо прокрутить колесико в обратном направлении.

Увеличение выполняет команда zoom в соответствии с рисунком 1.2.

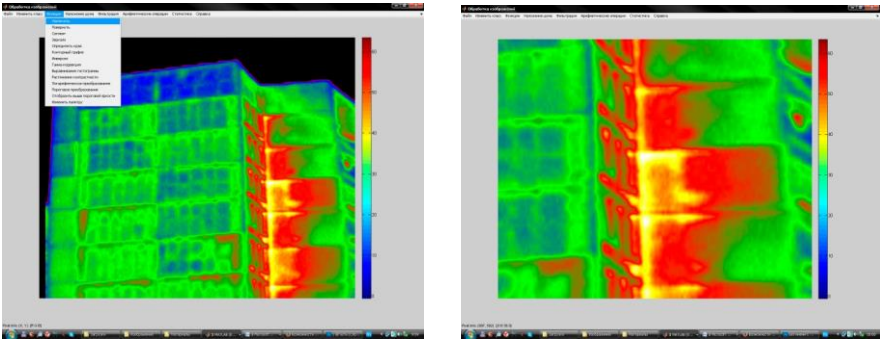

Рисунок – 1.2 – Исходное изображение (слева) и увеличенное изображение (справа)

Для поворота изображения на заданный угол необходимо выбрать пункт «Повернуть». При этом в появившемся стандартном диалоговом окне для ввода строк необходимо ввести угол, задаваемый в градусах. По умолчанию вводится угол равный 90°. После подтверждения команда imrotate повернет изображение на заданный угол против часовой стрелки. Callback-функция «rotate» имеет следующий синтаксис:

```
I=imread('picture.bmp');
lines = 1;
def = {'90'};
G=inputdlg('Введите угол поворота (от 0 до 360)', 'Введите угол', lines, def)
A=cell2mat(G);
B=str2double(A);
J=imrotate(I, B, 'bicubic');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Для выделения сегмента изображения необходимо выбрать пункт «Сегмент», при этом появляется специальный курсор в виде прямоугольника. Изменяя ширину и высоту прямоугольника правой кнопкой манипулятора, мы тем самым выделяем границы нового изображения. После выделения необходимо дважды нажать левой кнопкой манипулятора по выбранной области в соответствии с рисунком 4. Callback-функция «segment» имеет следующий синтаксис:

```
I=imread('picture.bmp');
J=imcrop(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Для выделения сегмента используется команда imcrop.

Пункт «Зеркало» позволяет зеркально отобразить изображение вокруг вертикальной оси, т. е. перевернуть матрицу изображения слева направо. Отображение осуществляется командой fliplr. При этом вызывается Callback-функция «zerkal»:

```
I=imread('picture.bmp');
J=fliplr(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Для обнаружения перепадов яркости пикселей необходимо выбрать пункт «Определить края», при этом в новом окне будет построено изображение, отображающее эти перепады белом цвете на черном фоне. При поиске таких перепадов используются производные первого и второго порядка. Основная идея обнаружения перепадов базируется на поиске мест изображения, где яркость меняется быстро, с помощью следующих двух общих критериев:

- найти места, где первая производная яркости превосходит по модулю некоторый заранее заданный порог;
- найти места, где вторые производные яркости имеют пересечения нулевого уровня

Таким образом, участки изображения, на которых скорость изменения яркости превысит некоторую нормированную величину, будут отображаться белым цветом, а остальные образуют черный фон.

Обнаружение перепадов яркости реализуется командой edge, при этом предварительно необходимо задать метод. В открывшемся стандартном диалоговом окне выбора из списка, которое вызывается командой listdlg, необходимо выбрать один из пяти предлагаемых методов (sobel, prewitt, roberts, zerocross или canny):

- метод «sobel» для обнаружения краев использует детектор Собела. Обнаруживает края с помощью приближений Собела первых производных;
- метод «prewitt» для обнаружения краев использует детектор Превитта. Обнаруживает края с помощью приближений Превитта первых производных;
- метод «roberts» для обнаружения краев использует детектор Робертса. Обнаруживает края с помощью приближений Робертса первых производных;
- метод «canny» для обнаружения краев использует детектор Канни. Обнаруживает края, выполняя поиск локальных максимумов градиента f(x,y). Градиент вычисляется от гауссиана. Метод использует два порога для нахождения сильных и слабых краев.

Слабые края включаются в выход, если они связаны с сильными. Следовательно, этот метод с большей вероятностью обнаруживает настоящие слабые края;
- метод «zerocross» для обнаружения краев использует детектор пересечения нулевого уровня. Обнаруживает края, выполняя поиск пересечений нулевого уровня после фильтрации f(x, y) фильтром, заданным пользователем
Callback-функция имеет следующий вид:

```
I=imread('picture.bmp');
S={'sobel','prewitt','roberts','zerocross','canny'};
[ind,ok]=listdlg('ListSize', [180 160], 'Name', 'Select method', 'ListString', S, 'PromptString','Выбор метода', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
   M='sobel';
case 2
   M='prewitt';
case 3
   M='roberts';
case 4
   M='zerocross';
case 5
   M='canny';
end
J=edge(I,M);
figure('Name','Края изображения','NumberTitle','off','WindowStyle','normal', 'MenuBar', 'none');
imshow(J, []);
imwrite(J,'picturesave.bmp');
zoom;
```

В денной программе реализована функция построения контурных графиков данных изображения. Для этого необходимо выбрать пункт «Контурный график». При этом необходимо будет ввести количество плоскостей для построения контурных линий. Вводить рекомендуется целые числа от 1 до 50. Чем больше плоскостей будет использоваться для построения графика, тем более информативным он будет являться. Следует отметить, что данная операция занимает много времени. За построение контурного графика отвечает команда imcontour. Она рассматривает полутоновое

изображение как трехмерную поверхность (x, y - пространственной системы координат и третья координата – яркость), строит для поверхности линии уровня и выводит их на экран. Линии уровня образуются в результате пересечения плоскостей, перпендикулярных оси яркости и рассматриваемой поверхности. Линии уровня, соответствующие различным яркостям (лежащим в различных плоскостях), обозначаются разными цветами.
Callback-функция «contourgraf» имеет следующий синтаксис:

```
I=imread('picture.bmp');
D=inputdlg('Введите количество цветов (целое число от 1 до 50)',
'Количество цветов');
A=cell2mat(D);
B=str2double(A);
figure('Name','Контурный
график','NumberTitle','off','WindowStyle','normal', 'MenuBar', 'none');
imcontour(I, B);
zoom;
```

При выборе пункта «инверсия», строится негативное (дополнительное) изображение. Эта операция реализуется командой imcomplement в соответствии с рисунком 10. Callback-функция «neg» имеет следующий синтаксис:

```
I=imread('picture.bmp');
J=imcomplement(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Нередко изображения оказываются слишком светлыми или слишком темными. Эти дефекты обус¬ловлены тем или иным видом передаточной характеристики по ярко¬сти — зависимости яркости пиксела от его значения. Для устранения этого неприятного эффекта необходимо выбрать пункт «Гамма-коррекция». При этом вызовется стандартное диалоговое окно для ввода строк, где необходимо ввести с клавиатуры коэффициент гамма (желательно в диапазоне от 0.1 до 10).
Гамма-коррекция — коррекция функции яркости пикселей. Повышение показателя гамма-коррекции позволяет повысить контрастность, разборчивость тёмных участков изображения, не делая при этом чрезмерно контрастными или яркими светлые детали снимка. Гамма-коррекция осуществляется командой imadjust.

При этом значения яркости в диапазоне от «low» до «high» преобразуются в значения яркости в диапазоне от «bottom» до «top». Значения яркости, меньшие «low», принимают значение «bottom», а значения яркости, большие «high», принимают значение «top». Параметр гамма (gamma) определяет форму кривой характеристики передачи уровней яркости. Если гамма меньше 1, то характеристика передачи уровней будет выпуклой и результирующее изображение будет светлее, чем исходное. Если гамма больше 1, то характеристика передачи уровней будет вогнутой и результирующее изображение будет темнее, чем исходное в соответствии с рисунком 1.3, 1.4.

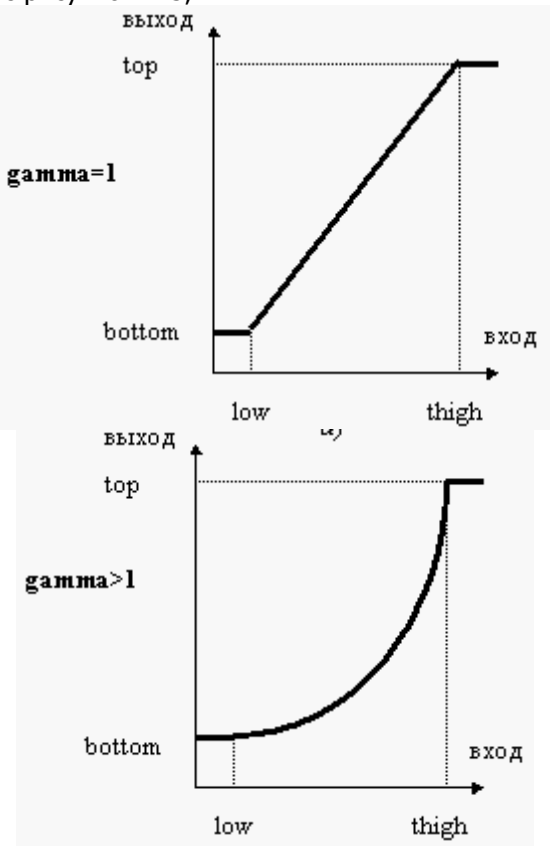

Рисунок – 1.3 – Передаточная характеристика яркости пикселя при значении гамма равном единицы (слева) и при значениях гамма больше единицы (справа)

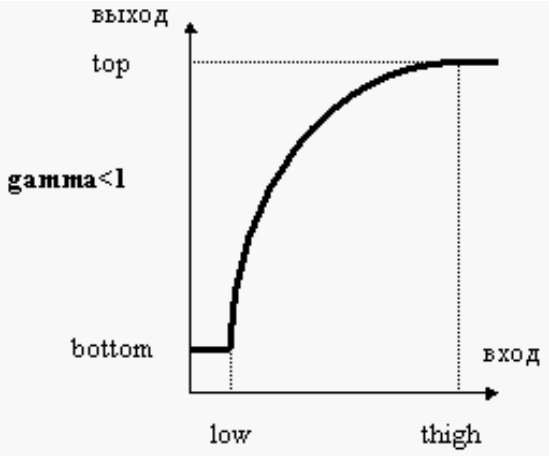

Рисунок – 1.4 – Передаточная характеристика яркости пикселя при значениях гамма меньше единицы

Callback-функция «gcor» имеет следующий синтаксис:

```
I=imread('picturegam.bmp');
D=inputdlg('Введите коэффициент гамма (от 0.1 до 10)', 'Введите коэффициент гамма');
A=cell2mat(D);
B=str2double(A);
J=imadjust(I, [], [], B);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturesave.bmp');
```

Для выравнивания (эквализации) гистограммы данного изображения необходимо выбрать пункт «Выравнивание гистограммы». Целью выравнивания гистограммы является такое преобразование, при котором, в идеале, все уровни яркости приобрели бы одинаковую частоту, а гистограмма яркостей отвечала бы равномерному закону распределения. Эту процедуру выполняет команда histeq в соответствии с рисунком 16 - 19. Расстояние Δf между дискретными уровнями яркости от fi до fi+1 в гистограмме исходного изображения одинаковое, но на каждый уровень выпадает различное число пикселей. При эквализации гистограммы расстояние Δgi между уровнями gi gi+1 различно, но число пикселей на каждом уровне, в среднем, одинаковое. Другими словами, преоб-

разование порождает изображение, уровни яркости которого являются равновероятными и покрывают весь интервал от 0 до 255. Результат этого процесса эквализации изображения состоит в увеличении динамического диапазона уровней яркости, что обычно означает большую контрастность выходного изображения.

Callback-функция «corgist» имеет следующий синтаксис:

```
I=imread('picture.bmp');
J=histeq(I, 256);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

При выборе пункта «Растяжение контрастности», выполняется функция преобразования растяжения контрастности. Она имеет следующий вид:

$$S = \frac{1}{1+\left(\frac{m}{r}\right)^{E}} \qquad (1.\ 1)$$

где s — соответствующая яркость выходного изображения;

m – среде арифметическое значений яркости пикселей исходного изображения;

r — это яркость входного изображения;

E – параметр, контролирующий наклон функции

Для определения m используется команда mean2. Степень E вводится с клавиатуры. Данная функция сжимает входные величины, меньшие чем m, в более узкий поддиапазон темных уровней на выходном изображении, и, соответственно, величины, большие m — в более узкую полосу ярких уровней в соответствии с рисунком 21 - 23. В результате получается изображение с большей контрастностью. В предельном случае, когда степень E больше 64, преобразование стремится к пороговому в соответствии с рисунком 1.5, то есть все уровни яркости меньшие m отображаются черным цветом, а уровни яркости больше m – белым цветом.

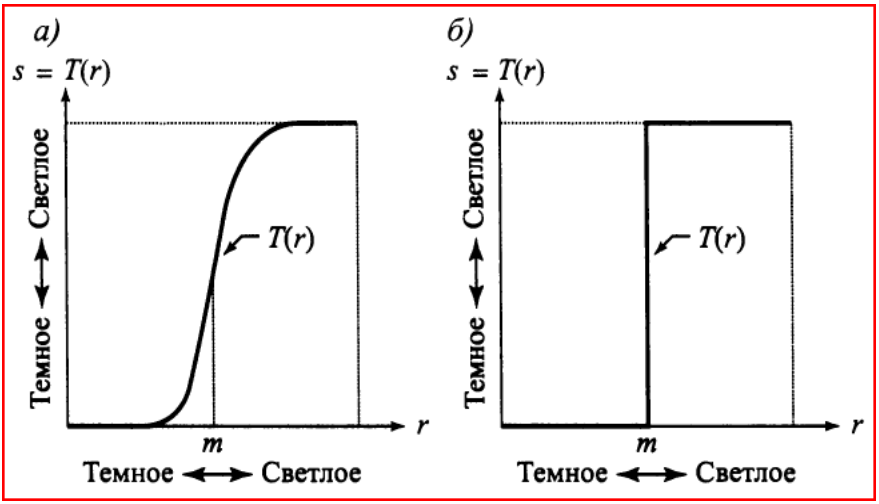

Рисунок – 1.5- а) Преобразование растяжения контрастности;
б) Пороговое преобразование

Callback-функция «contrastsprain» имеет следующий синтаксис:

```
I=imread('picture.bmp');
classin = class(I);
if strcmp(class(I), 'double') & max(I(:)) > 1 & ...
    ~strcmp(varargin{l}, 'log')
    I = mat2gray(I);
else
    I = im2double(I);
end
D=inputdlg('Введите степень E (при значениях более 24 преобразование приближается к пороговому)', 'Введите степень');
A=cell2mat(D);
B=str2double(A);
m = mean2(I);
E = B;
g = 1./(1 + (m./(double(I) + eps)).^E);
gs = im2uint8(mat2gray(g));
imshow(gs, []);
colorbar('vert');
impixelinfo;
imwrite(gs,'picture.bmp');
imwrite(gs,'picturegam.bmp');
imwrite(gs,'picturesave.bmp');
```

При выборе пункта «Логарифмическое преобразование», реали-

зуется функция логарифмического преобразования, которая имеет следующий вид:

$$g = c * \log(1 + r) \quad (1.\ 2)$$

где g — соответствующая яркость выходного изображения;
c – константа (в данной программе равна 1);
r — яркость входного изображения
Основное применение логарифмического преобразования состоит в сжатии динамического диапазона. Callback-функция «logtransformation» имеет следующий синтаксис:

```
I=imread('picture.bmp');
classin = class(I);
if strcmp(class(I), 'double') & max(I(:)) > 1 & ...
   ~strcmp(varargin{l}, 'log')
   I = mat2gray(I);
else
   I = im2double(I);
end
J=1*(log(1+ double(I)));
Js = im2uint8(mat2gray(J));
imshow(Js, []);
colorbar('vert');
impixelinfo;
imwrite(Js,'picture.bmp');
imwrite(Js,'picturegam.bmp');
imwrite(Js,'picturesave.bmp');
```

Для реализации порогового преобразования необходимо выбрать пункт «Пороговое преобразование». Преобразование осуществляется командой im2bw, при этом все пиксели, яркость которых меньше пороговой примут значение «0» (черный цвет), а все остальные – «1» (белый цвет). Уровень пороговой яркости определяется с помощью наведения указателя на область изображения, яркость которой считается равной пороговой, и нажатием правой кнопки манипулятора (либо двойное нажатие левой кнопки).
Callback-функция «porogtransformation» имеет следующий синтаксис:

```
I=imread('picture.bmp');
P=impixel(I);
n=max(P);
m=n/255;
```

```
J=im2bw(I, m);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

При выборе пункта «Отобразить выше пороговой яркости», сначала выполняется пороговое преобразование, а за тем из преобразованного изображения вычитается негатив исходного изображения, при этом порог яркости можно выбрать курсором, щелкнув правой кнопкой манипулятора по изображению, а можно взять как среднее арифметическое или стандартное отклонение яркости пикселей. После такой операции пиксели, имеющие яркость ниже пороговой будут отображаться черным цветом, а остальные пиксели будут иметь яркость как в исходном изображении. Эта функция полезна, если необходимо выделить участки с определенной яркостью.

Callback-функция «displayabovethreshold» имеет следующий синтаксис:

```
I=imread('picture.bmp');
S={'Определить с помощью указателя','Взять как среднее арифметическое','Взять как стандартное отклонение'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Select manner', 'ListString', S, 'PromptString','Способ определения порогового значения', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
    P=impixel(I);
    n=max(P);
    m=n/255;
case 2
    n=mean2(I);
    m=n/255;
case 3
    n=std2(I);
    m=n/255;
end
J=im2bw(I, m);
J=im2uint8(J);
N=imcomplement(I);
```

```
N=im2uint8(N);
A=imsubtract(J,N);
imshow(A, []);
colorbar('vert');
impixelinfo;
imwrite(A,'picture.bmp');
imwrite(A,'picturegam.bmp');
imwrite(A,'picturesave.bmp');
```

Команда imsubtract вычитает матрицу негативного изображения из матрицы изображения, полученного с помощью команды im2bw.

При выборе пункта «Изменить палитру», выбранное ранее изображение отображается с новой палитрой цветов в новом графическом окне. Пользователю предлагается выбрать одну из четырнадцати доступных цветовых палитр. Палитра цветов M – это матрица размера m на 3 действительных чисел из диапазона от 0,0 до 1,0. Строка k палитры сформирована из трех чисел, которые указывают интенсивность красного, зеленого и синего цветов, то есть M(k:) = [r(k) g(k) b(k)].

Команда colormap(M) устанавливает палитру согласно матрице M. На рисунке 1.6 представлены шкалы цветов использующихся палитр.

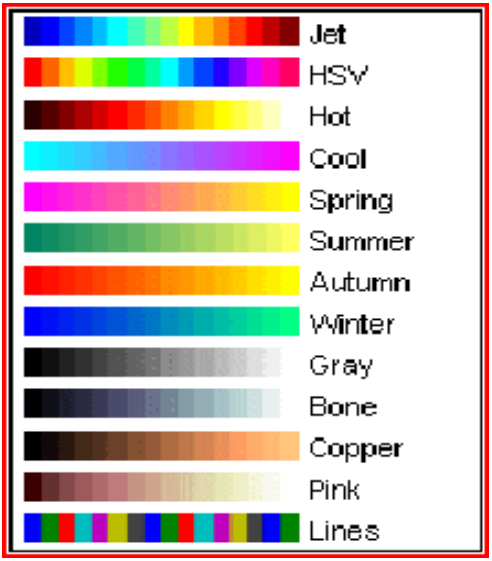

Рисунок – 1.6 – Шкалы цветов различных палитр

В таблице 1. 1 представлено описание некоторых палитр.
Таблица 1. 1 – Характеристики палитр

Название	Расшифровка	Описание
cool	Shades of cyan and magenta color map	Палитра с оттенками голубого и фиолетого
copper	Linear copper-tone color map	Линейная палитра в оттенках меди
gray	Linear grey-scale color map	Линейная палитра в оттенках серого
hot	Black-red-yellow-white color map	Палитра с чередованием черного, красного, желтого и белого
hsv	Hue-saturation-value color map	Палитра радуги
jet	Variant of hsv	Разновидность hsv-палитры
pink	Pastel shades of pink color map	Розовая палитра с оттенками пастели

Продолжение таблицы 1.1 – Характеристики палитр

Название	Расшифровка	Описание
rism	Prism (red-orange-yellow-green -blue-violet) color map	Палитра с чередованием красного, оранжевого, желтого, зеленого, синего и фиолетового
bone	Grey-scale with tinge of blue color map	Серая палитра с оттенком синего

Callback-функция «changemap» имеет следующий синтаксис:

```
I=imread('picturesave.bmp');
S={'autumn','colorcube','cool','copper','gray','hot','hsv','jet','lines','pink'
,'prism','spring','summer','winter'};
[ind,ok]=listdlg('ListSize', [180 160], 'Name', 'Select palette',
'ListString', S, 'PromptString','Выберите тип палитры', 'Selec-
tionMode', 'single');
Num=ind;
switch Num
case 1
   M='autumn';
```

```
case 2
    M='colorcube';
case 3
    M='cool';
case 4
    M='copper';
case 5
    M='gray';
case 6
    M='hot';
case 7
    M='hsv';
case 8
    M='jet';
case 9
    M='lines';
case 10
    M='pink';
case 11
    M='prism';
case 12
    M='spring';
case 13
    M='summer';
case 14
    M='winter';
end
figure('NumberTitle','on','WindowStyle','normal', 'MenuBar', 'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(I, []);
colormap(M)
colorbar('vert');
impixelinfo;
zoom;
```

1.2.4 МЕНЮ «НАЛОЖЕНИЕ ШУМОВ»

При моделировании задач фильтрации в MATLAB возникает необходимость генерирования моделей различных по характеру шумов и зашумления изображений.

Для гауссового белого шума используется Callback-функция «white»

Callback-функция «white»имеет следующий синтаксис:

```
I=imread('picture.bmp');
J=imnoise(I,'gaussian');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Для шума Пуассона используется Callback-функция «puasson»

Callback-функция «puasson»имеет следующий синтаксис:

```
I=imread('picture.bmp');
J=imnoise(I,'poisson');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

Для шума в виде черных и белых пикселов (шум называется - «соль и перец») используется Callback-функция «solt»

Callback-функция «solt»имеет следующий синтаксис:

```
I=imread('picture.bmp');
J=imnoise(I,'salt & pepper',0.02);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');
```

1.2.5 МЕНЮ «ФИЛЬТРАЦИЯ»

Для удаления шумов определенного типа, можно воспользоваться линейной пространственной фильтрацией, реализуемой командой imfilter. Для этого необходимо выбрать пункт «Линейная фильтрация» и выбрать соответствующий тип фильтра. Механизм линейной пространственной фильтрации заключается в перемещении центра фильтрующей маски w (ядра фильтра) от точки к точке изображения f. В каждой точке (x, y) откликом фильтра является сумма произведений коэффициентов фильтра и соответствующих пикселей окрестности, которые накрываются фильтрующей маской. В данной программе реализовано 10 типов линейных фильтров: «Прямоугольный усредняющий фильтр», «Круговой усредняющий фильтр», «Низкочастотный гауссов фильтр», «Фильтр Лапласа 4», «Фильтр Лапласа 8», «Лаплас от гауссова фильтра (LoG)», «Motion», «Фильтр Превитта», «Фильтр Собела», «Нечеткий фильтр». Разница между этими типами фильтров состоит в том, каким образом учитываются значения соседних пикселей. Описание линейных фильтров приведено в таблице 1. 2.

Таблица 1. 2 – Типы линейных фильтров

Тип	Описание
average	Прямоугольный усредняющий фильтр размера 3x3
disk	Круговой усредняющий фильтр (внутри квадрата со стороной 2r+1) радиуса r равным 3.
gaussian	Низкочастотный гауссов фильтр размера 3x3 со стандартным (положительным) отклонением 0.5.
laplacian	Фильтр Лапласа 3x3. Фильтр Лапласа 4 имеет форму ядра [0 1 0; 1 -4 1; 0 1 0], а фильтр Лапласа 8 - [1 1 1; 1 -8 1; 1 1 1]
log	Лаплас от гауссова фильтра (LoG) размера 5x5 со стандартным (положительным) отклонением 0.5
motion	Выдает фильтр, который, будучи свернутым с изображением, приближает линейное перемещение (видеокамеры по отношению к изображению) на 6 пикселей. Направление перемещения задается углом theta равным 0, который измеряется в градусах от горизонтали против часовой стрелки

prewitt	Выдает 3x3 маску Превитта wv, которая аппроксимирует вертикальный градиент
sobel	Выдает 3x3 маску Собела sv, которая аппроксимирует вертикальный градиент
unsharp	Выдает маску 3x3 нечеткого фильтра

Фильтры Превитта, Собела, а так же фильтры «Лапласа 4» и «Лапласа 8» подчеркивают перепады уровней яркостей на изображении, при этом изображение сильно искажается. Для устранения этого негативного эффекта необходимо изображение, отфильтрованное одним из этих фильтров, вычесть из исходного изображения (меню «Арифметические операции» «Из исходного вычесть отфильтрованное»). Эта процедура позволяет повысить четкость изображения.

Callback-функция «linfilt» имеет следующий синтаксис:

```
I=imread('picture.bmp');
F1=fspecial('average', [3 3]);
F2=fspecial('disk', 3);
F3=fspecial('gaussian', [3 3], 0.5);
F4=fspecial('laplacian', 0.5);
F5=[1 1 1; 1 -8 1; 1 1 1];
F6=fspecial('log', [5 5], 0.5);
F7=fspecial('motion', 6, 0);
F8=fspecial('prewitt');
F9=fspecial('sobel');
F10=fspecial('unsharp', 0.2);
S={'Прямоугольный усредняющий фильтр','Круговой усредняющий фильтр','Низкочастотный гауссов фильтр','Фильтр Лапласа 4','Фильтр Лапласа 8','Лаплас от гауссова фильтра (LoG) ','Motion','Фильтр Превитта','Фильтр Собела','Нечеткий фильтр'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Type filtr', 'ListString', S, 'PromptString','Выберите тип фильтра', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
   F=F1;
case 2
   F=F2;
case 3
   F=F3;
case 4
   F=F4;
```

```
case 5
    F=F5;
case 6
    F=F6;
case 7
    F=F7;
case 8
    F=F8;
case 9
    F=F9;
case 10
    F=F10;
end
G=imfilter(I, F, 'replicate');
imshow(G, []);
colorbar('vert');
impixelinfo;
imwrite(G,'picture.bmp');
imwrite(G,'picturegam.bmp');
imwrite(G,'picturesave.bmp');
imwrite(G,'picturefilt.bmp');
```

К нелинейным фильтрам, использующимся в данном приложении, относятся ранговый фильтр (фильтр порядковых статистик), медианный фильтр и адаптивный фильтр Винера.

Что бы осуществить ранговую фильтрацию, необходимо выбрать пункт «Ранговая фильтрация». Отклик этого нелинейного пространственного фильтра основан на предварительном упорядочении (ранжировании) пикселей изображения из текущей окрестности, после чего центральному пикселю присваивается значение, определенное в результате данного упорядочения. Ранговую фильтрацию выполняет команда ordfilt2 (I, order, domain), где I – матрица входного изображения, order – порядковый номер пикселя в матрице domin, domain — это матрица m на n, состоящая из нулей и единиц, которые обозначают позиции пикселей, участвующие в вычислениях. В этом смысле матрица domain действует как маска. Пиксели окрестности, которым соответствуют нули в domain, не участвуют в вычислениях. Пиксели исходного изображения, соответствующие ненулевым элементам маски фильтра domain, сортируются в порядке возрастания. Пикселю изображения I, соответствующему центральному элементу маски, присваивается значение с порядковым номером order. Это делается нерекурсивно, чтобы размеры входного и выходного изображения бы-

ли одинаковы.

Самым интересным свойством этого алгоритма является возможность создания эффектов расфокусировки (эрозии) и фокусировки (уточнения, наращивания фона и т. д.) изображения. В первом случае центральный пиксель маски надо выбрать как минимальный по порядку в маске (так называемый фильтр минимума), а во втором случае – как пиксель максимальный (фильтр максимума). Используя терминологию статистик, фильтр минимума (первый элемент упорядоченной по возрастанию последовательности) — это нулевой процентиль. Аналогично, сотый процентиль — это последний элемент упорядоченной последовательности, имеющий номер m*n, он соответствует фильтру максимума. Самым известным фильтром порядковых статистик в цифровой обработке изображений является медианный фильтр, который соответствует пятидесятому процентилю. Таким образом, изменяя положение центрального пикселя маски, можно менять в широких пределах свойства ранговой фильтрации.

Callback-функция «rangfilt» имеет следующий синтаксис:

```
I=imread('picture.bmp');
S={'[2 2]','[4 4]','[8 8]'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Enter domain', 'ListString', S, 'PromptString','Задайте параметр domain', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
    prompt={'Введите параметр order (целое число от 1 до 4)'};
    dlg_title='Параметр order';
    num_lines = 1;
    def={'2'};
    D=inputdlg(prompt, dlg_title, num_lines, def);
    A=cell2mat(D);
    B=str2double(A);
    F=ordfilt2 (I, B, ones(2,2));
case 2
    prompt={'Введите параметр order (целое число от 1 до 16)'};
    dlg_title='Параметр order';
    num_lines = 1;
    def={'4'};
    D=inputdlg(prompt, dlg_title, num_lines, def);
    A=cell2mat(D);
    B=str2double(A);
```

```
    F=ordfilt2 (I, B, ones(4,4));
case 3
    prompt={'Введите параметр order (целое число от 1 до 64)'};
    dlg_title='Параметр order';
    num_lines = 1;
    def={'32'};
    D=inputdlg(prompt, dlg_title, num_lines, def);
    A=cell2mat(D);
    B=str2double(A);
    F=ordfilt2 (I, B, ones(8,8));
end
imshow (F);
colorbar('vert');
impixelinfo;
imwrite(F,'picture.bmp');
imwrite(F,'picturegam.bmp');
imwrite(F,'picturesave.bmp');
```

Эффективным средством фильтрации шума типа «соль и перец» («salt and pepper») является медианная фильтрация. Для осуществления этой процедуры необходимо выбрать пункт «Медианная фильтрация», после чего будет предложено выбрать параметр domain (размер маски фильтра). Размер маски фильтра следует выбирать исходя из степени зашумленности исходного изображения. Медианная фильтрация является частным случаем ранговой фильтрации. Команда medfilt2(I, [m n]) фильтрует матрицу исходного изображения, используя маску фильтра размера m на n. Центральный пиксель маски получают усреднением всех ее пикселей. Маска применяется не рекурсивно ко всему изображению.

Callback-функция «medfiltr» имеет следующий синтаксис:

```
I=imread('picture.bmp');
S={'[2 2]','[4 4]','[6 6]','[8 8]','[4 2]','[8 4]'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Enter domain', 'ListString', S, 'PromptString','Задайте параметр domain', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
    I1=medfilt2 (I, [2 2]);
case 2
    I1=medfilt2 (I, [4 4]);
case 3
```

```
   I1=medfilt2 (I, [6 6]);
case 4
   I1=medfilt2 (I, [8 8]);
case 5
   I1=medfilt2 (I, [4 2]);
case 6
   I1=medfilt2 (I, [8 4]);
end
imshow(I1, []);
colorbar('vert');
impixelinfo;
imwrite(I1,'picture.bmp');
imwrite(I1,'picturegam.bmp');
imwrite(I1,'picturesave.bmp');
```

При выборе пункта «Адаптивная фильтрация Винера», реализуется нелинейный адаптивный фильтр Винера. Данный метод фильтрации обычно применяется для подавления гауссовского белого шума на исходном изображении. Он реализуется командой wiener2(I, [m n]). Вектор [m n] задает размеры скользящего окна фильтра. При осуществлении фильтрации учитываются статистические особенности изображения в пределах окна, в частности среднее значение яркости и ее среднеквадратическое отклонение.

Callback-функция «filtrviner» имеет следующий синтаксис:

```
I=imread('picture.bmp');
S={'[2 2]','[4 4]','[6 6]','[8 8]','[4 2]','[8 4]'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Enter domain', 'ListString', S, 'PromptString','Задайте параметр domain', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
   I2=wiener2(I, [2 2]);
case 2
   I2=wiener2(I, [4 4]);
case 3
   I2=wiener2(I, [6 6]);
case 4
   I2=wiener2(I, [8 8]);
case 5
   I2=wiener2(I, [4 2]);
case 6
```

```
    I2=wiener2(I, [8 4]);
end
imshow (I2, []);
colorbar('vert');
impixelinfo;
imwrite(I2,'picture.bmp');
imwrite(I2,'picturegam.bmp');
imwrite(I2,'picturesave.bmp');
```

1.2.6 МЕНЮ «АРИФМЕТИЧЕСКИЕ ОПЕРАЦИИ»

При выборе пункта «Возвести в квадрат», все элементы матрицы исходного изображения поэлементно возводятся во вторую степень. Эта операция бывает полезна для коррекции темных оттенков изображения, при которой они станут более насыщенными, а изображение в целом – более темное. Возведенное в квадрат изображение открывается в новом графическом окне и не подлежит сохранению и дальнейшей обработке [1,6].
Callback-функция «elevateindegree2» имеет следующий синтаксис:

```
I=imread('picturesave.bmp');
I16=uint16(I);
R=immultiply(I16,I16);
figure('Name','Квадрат
изображения','NumberTitle','off','WindowStyle','normal',     'MenuBar',
'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(R, []);
colorbar('vert');
impixelinfo;
zoom;
```

При выборе пункта «Сложить изображения», вызываются стандартные диалоговые окна выбора имени файла для чтения, где необходимо указать графические файлы, изображения которых требуется сложить. Результат суммирования отобразится в новом окне без возможности сохранения и дальнейшей обработки. Сложение матриц изображений выполняет команда imadd.
Callback-функция «sumim» имеет следующий синтаксис:

```
[file_name,file_path]=uigetfile('*.*','Выберите файл X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите файл Y')
J=imread([file_path,file_name]);
```

```
R=imadd(I,J,'uint16');
figure('Name','Сумма
изображений','NumberTitle','off','WindowStyle','normal',  'MenuBar',
'none');
set(gca,'Position',[0.02 0.05 0.98 0.94]);
imshow(R, []);
colorbar('vert');
impixelinfo;
zoom;
```

При выборе пункта «Перемножить изображения», вызываются стандартные диалоговые окна выбора имени файла для чтения, где необходимо указать графические файлы, изображения которых требуется перемножить. Произведение изображений отобразится в новом окне без возможности сохранения и дальнейшей обработки. Перемножение матриц изображений выполняет команда immultiply.

Callback-функция «multipim» имеет следующий синтаксис:

```
[file_name,file_path]=uigetfile('*.*','Выберите файл X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите файл Y')
J=imread([file_path,file_name]);
I16=uint16(I);
J16=uint16(J);
R=immultiply(I16,J16);
figure('Name','Произведение
изображений','NumberTitle','off','WindowStyle','normal',  'MenuBar',
'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(R, []);
colorbar('vert');
impixelinfo;
zoom;
```

При выборе пункта «Вычесть из изображения X изображение Y», вызываются стандартные диалоговые окна выбора имени файла для чтения, где необходимо указать графические файлы, для которых необходимо получить разность. Матрица изображения «У» вычитается из матрицы изображения «Х». Разность изображений отображается в главном графическом окне с возможностью сохранения и дальнейшей обработки.

Callback-функция «substractim» имеет следующий синтаксис:

```
[file_name,file_path]=uigetfile('*.*','Выберите изображение X')
I=imread([file_path,file_name]);
```

```
[file_name,file_path]=uigetfile('*.*','Выберите изображение Y')
set(gca,'Position',[0.02 0.02 0.91 0.93]);
J=imread([file_path,file_name]);
R=imsubtract(I,J);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');
```

Команда msubtract из матрицы I вычитает матрицу J.

При выборе пункта «Из исходного вычесть отфильтрованное», берется разность между исходным изображением и отфильтрованным с помощью одного из вышеописанных линейных фильтров. При использовании фильтра «Лапласа 4», фильтра «Лапласа 8», Превитта или фильтра Собела, достигается эффект выделения контуров изображения и, как следствие, повышение четкости.

Callback-функция «substrfilt» имеет следующий синтаксис:

```
load imaging
I=imread([file_path,file_name]);
F=imread('picturefilt.bmp');
R=imsubtract(I, F);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');
```

При выборе пункта «Вычесть из изображения константу», из каждого элемента матрицы данного изображения вычитается константа, которая вводится с клавиатуры (целое число от 0 до 255). Значение константы не должно превышать максимальной яркости данного изображения.

Callback-функция «substractconst» имеет следующий синтаксис:

```
I=imread('picture.bmp');
D=inputdlg('Введите константу', 'Константа');
A=cell2mat(D);
B=str2double(A);
R=imsubtract(I, B);
imshow(R, []);
colorbar('vert');
```

```
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');
```

При выборе пункта «Разделить изображение X на изображение У», происходит поэлементное деление матрицы изображения «X» на матрицу изображения «У». Изображения выбираются с помощью стандартных диалоговых окон ввода имени файла для чтения. Деление осуществляется командой imdivide.

Callback-функция «divim» имеет следующий синтаксис:

```
[file_name,file_path]=uigetfile('*.*','Выберите изображение X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите изображение Y')
J=imread([file_path,file_name]);
set(gca,'Position',[0.02 0.02 0.91 0.93]);
R=imdivide(I,J);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');
```

При выборе пункта «Разделить изображение на константу», каждый элемент матрицы данного изображения будет разделен на константу, введенную с клавиатуры. Результат данной операции будет отображен в главном графическом окне приложения с возможностью дальнейшего сохранения и преобразования.

Callback-функция «divconst» имеет следующий синтаксис:

```
I=imread('picture.bmp');
D=inputdlg('Введите константу', 'Константа');
A=cell2mat(D);
B=str2double(A);
R=imdivide(I, B);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');
```

1.2.7 МЕНЮ «СТАТИСТИКА»

Чтобы построить гистограмму яркости пикселей данного изображения, необходимо выбрать пункт «Гистограмма». Гистограмма будет отображена в отдельном окне. По горизонтальной оси откладываются значения яркости пикселей (для удобства числовая шкала совмещена со шкалой цветности пикселей), а по вертикальной оси – частота обнаружения этих пикселей в изображении. Для построения гистограммы данных изображения используется команда imhist [1,6].

Callback-функция «gistim» имеет следующий синтаксис:

```
I=imread('picturesave.bmp');
figure('Name','Гистограмма','NumberTitle','off','WindowStyle','normal',
'MenuBar', 'none');
imhist(I, 256);
axis([0 255 0 15000]);
set(gca, 'xtick', 0:50:255);
set(gca, 'ytick', 0:1000:15000);
```

Для определения среднеарифметической яркости данного изображения необходимо выбрать пункт «Среднее значение яркости». При этом, рассчитанное с помощью команды mean2, среднее значение яркости отобразится в стандартном диалоговом окне со справочной информацией, вызванном командой helpdlg. Для корректного отображения результат вычисления среднего арифметического переводится в символьный формат командой char.

Callback-функция «meanpix» имеет следующий синтаксис:

```
I=imread('picturesave.bmp');
M=mean2(I);
M1=num2str(M);
M2=char(M1);
helpdlg(M2,'Среднее значение элементов матрицы изображения');
```

Для определения стандартного (среднеквадратического) отклонения элементов матрицы данного изображения необходимо выбрать пункт «Стандартное отклонение». Рассчитанное командой std2 значение отобразится в стандартном диалоговом окне со справочной информацией.

Callback-функция «meanpix» имеет следующий синтаксис:

```
I=imread('picturesave.bmp');
M=std2(I);
M1=num2str(M);
M2=char(M1);
```

helpdlg(M2,'Стандартное отклонение элементов матрицы изображения');

Для определения двумерного коэффициента корреляции между матрицами двух изображений в соответствии с рисунком 57, необходимо выбрать пункт «Двумерный коэффициент корреляции», при этом матрицы должны иметь одинаковый размер. В появившихся стандартных диалоговых окнах выбора имени файла для чтения, необходимо выбрать графические файлы, матрицы которых (A и B) будут участвовать в расчете коэффициента корреляции.

Коэффициент корреляции между матрицами A и B одинакового размера вычисляется с помощью команды corr2, по следующей формуле:

$$r = \frac{\sum_m \sum_n \left(A_{mn} - \overline{A}\right) \cdot \left(B_{mn} - \overline{B}\right)}{\sqrt{\left(\sum_m \sum_n \left(A_{mn} - \overline{A}\right)^2\right) \cdot \left(\sum_m \sum_n \left(B_{mn} - \overline{B}\right)^2\right)}} \qquad (1.\ 3)$$

где $\overline{A}$ и $\overline{B}$ - среднее арифметическое элементов матриц изображений A и B соответственно [1,6].

Callback-функция «corrim» имеет следующий синтаксис:

```
[file_name,file_path]=uigetfile('*.*','Выберите изображение A')
A=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите изображение B')
B=imread([file_path,file_name]);
r=corr2(A,B);
r1=num2str(r);
r2=char(r1);
helpdlg(r2,'Коэффициент корреляции');
```

1.2.8 МЕНЮ «СПРАВКА»

При выборе пункта «Информация об изображении», откроется стандартное диалоговое окно выбора из списка, в котором необходимо выбрать один из параметров текущего изображения, о котором необходимо получить информацию. Предлагаются следующие параметры: «Размер файла» (FileSize) - размер файла в килобайтах, «Формат» (Format) - формат временного файла «bmp», «Ширина» (Width) – ширина изображения в пикселях, «Высота» (Height) – высота изображения в пикселях, «Глубина цвета» (BitDepth) – разрядность изображения в битах, «Разрешение» (BitmapSize) – произведение высоты и ширины изображения в пикселях в соответствии с рисунком 58, «Тип сжатия» (CompressionType), «Тип изображения» (ColorType), «Количество цветовых оттенков» (NumColormapEntries), «MaxSampleValue» - максимальное значение яркости пикселя, «MinSampleValue» - минимальное значение яркости пикселя.
Callback-функция «iminf» имеет следующий синтаксис:

```
I=imread('picturesave.bmp');
K=imfinfo ('picturesave.bmp')
S={'Размер файла', 'Формат', 'Ширина', 'Высота', 'Глубина цвета', 'Разрешение', 'Тип сжатия', 'Тип изображения', 'Количество цветовых оттенков', 'MaxSampleValue', 'MinSampleValue'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Select parameter', 'ListString', S, 'PromptString','Выберите параметр изображения', 'SelectionMode', 'single');
Num=ind;
switch Num
 case 1
    Fs=K.FileSize/1024;
    Fs1=num2str(Fs);
    Fs2=char(Fs1);
    helpdlg(Fs2,'Размер файла (КБ)');
case 2
    Frt=K.Format;
    Frt=char(Frt);
    helpdlg(Frt,'Формат');
case 3
    W=K.Width;
    W1=num2str(W);
    W2=char(W1);
```

```
    helpdlg(W2,'Ширина (pix)');
case 4
    H=K.Height;
    H1=num2str(H);
    H2=char(H1);
    helpdlg(H2,'Высота (pix)');
case 5
    BD=K.BitDepth;
    BD1=num2str(BD);
    BD2=char(BD1);
    helpdlg(BD2,'Глубина цвета (бит)');
case 6
    BS=K.BitmapSize;
    BS1=num2str(BS);
    BS2=char(BS1);
    helpdlg(BS2,'Разрешение (pix)');
case 7
    Cmp=K.CompressionType;
    Cmp=char(Cmp);
    helpdlg(Cmp,'Тип сжатия');
case 8
    CT=K.ColorType;
    CT=char(CT);
    helpdlg(CT,'Тип изображения');
case 9
    NCU=K.NumColorsUsed;
    NCU1=num2str(NCU);
    NCU2=char(NCU1);
    helpdlg(NCU2,'Количество цветовых оттенков');
case 10
    MaSV=K.MaxSampleValue;
    MaSV1=num2str(MaSV);
    MaSV2=char(MaSV1);
    helpdlg(MaSV2,'MaxSampleValue');
case 11
    MiSV=K.MinSampleValue;
    MiSV1=num2str(MiSV);
    MiSV2=char(MiSV1);
    helpdlg(MiSV2,'MinSampleValue');
end
```

При выборе пункта «О программе», в новом графическом окне модального типа отображается название программы, краткое

описание программы, а также информация о разработчике.
Callback-функция «aboutprog» имеет следующий синтаксис:

```
X = imread('ATT00042_11.bmp');
[m n k]=size(X);
figure('Name','O
программе...','NumberTitle','off','Resize','off','Units','pixels','Position',[1
00 100 n m],'WindowStyle','modal');
set(gca,'Position',[0 0 1 1]);
imshow (X);
```

ТЕКСТ ПРОГРАММЫ ОБРАБОТКИ ТЕПЛОВИЗИОННЫХ ИЗОБРАЖЕНИЙ

```
function gui08
figure ('MenuBar', 'None', ...
    'Name', 'Обработка изображений', ...
    'NumberTitle', 'Off')

m = uimenu ('Label', 'Файл');
uimenu ('Label', 'Открыть', ...
    'Parent', m, ...
    'CallBack', @openfile)
uimenu ('Label', 'Открыть DICOM-изображение', ...
     'Parent', m, ...
    'CallBack', @opendicom)
uimenu ('Label', 'Открыть  в новом окне', ...
        'Parent', m, ...
    'CallBack', @opennfg)
uimenu ('Label', 'Сохранить как…', ...
    'Parent', m, ...
    'CallBack', @save2)
uimenu ('Label', 'Исходное', ...
    'Parent', m, ...
    'CallBack', @source)
uimenu ('Label', 'Изменить размер', ...
        'Parent', m, ...
    'CallBack', @resizeimage)
uimenu ('Label', 'Преобразовать в монохромное', ...
        'Parent', m, ...
    'CallBack', @convertinmonohrom)
uimenu ('Label', 'Печать', ...
        'Parent', m, ...
```

```
    'CallBack', @Print)
uimenu ('Label', 'Выход', ...
        'Parent', m, ...
    'CallBack', @close)

m = uimenu ('Label', 'Изменить класс');
uimenu ('Label', 'uint8', ...
    'Parent', m, ...
    'CallBack', @class8)
uimenu ('Label', 'uint16', ...
     'Parent', m, ...
    'CallBack', @class16)
uimenu ('Label', 'double', ...
        'Parent', m, ...
    'CallBack', @classdoub)
uimenu ('Label', 'Logical', ...
    'Parent', m, ...
    'CallBack', @classlogoc)

m = uimenu ('Label', 'Функции');
uimenu ('Label', 'Увеличить', ...
    'Parent', m, ...
    'CallBack', @Zoomin)
uimenu ('Label', 'Повернуть', ...
     'Parent', m, ...
    'CallBack', @rotate)
uimenu ('Label', 'Сегмент', ...
        'Parent', m, ...
    'CallBack', @segment)
uimenu ('Label', 'Зеркало', ...
    'Parent', m, ...
    'CallBack', @zerkal)
uimenu ('Label', 'Опредилить края', ...
     'Parent', m, ...
    'CallBack', @Untitled_5)
uimenu ('Label', 'Контурный график', ...
        'Parent', m, ...
    'CallBack', @contourgraf)
uimenu ('Label', 'Инверсия', ...
    'Parent', m, ...
    'CallBack', @neg)
uimenu ('Label', 'Гамма-коррекция', ...
```

```
    'Parent', m, ...
   'CallBack', @gcor)
uimenu ('Label', 'Выравнивание гистограммы', ...
       'Parent', m, ...
   'CallBack', @corgist)
uimenu ('Label', 'Растяжение контрастности', ...
   'Parent', m, ...
   'CallBack', @contrastsprain)
uimenu ('Label', 'Логарифмическое преобразование', ...
    'Parent', m, ...
   'CallBack', @logtransformation)
uimenu ('Label', 'Пороговое преобразование', ...
       'Parent', m, ...
   'CallBack', @porogtransformation)
uimenu ('Label', 'Отобразить выше пороговой яркости', ...
   'Parent', m, ...
   'CallBack', @displayabovethreshold)
uimenu ('Label', 'Изменить палитру', ...
   'Parent', m, ...
   'CallBack', @changemap)

m = uimenu ('Label', 'Наложение шума');
uimenu ('Label', 'Белый гауссовский шум', ...
   'Parent', m, ...
   'CallBack', @white)
uimenu ('Label', 'Пуассоновский шум', ...
    'Parent', m, ...
   'CallBack', @puasson)
uimenu ('Label', 'Шум типа "Соль и перец"', ...
       'Parent', m, ...
   'CallBack', @solt)


m = uimenu ('Label', 'Фильтрация');
uimenu ('Label', 'Линейная фильтрация', ...
   'Parent', m, ...
   'CallBack', @linfilt)
uimenu ('Label', 'Ранговая фильтрация', ...
    'Parent', m, ...
   'CallBack', @rangfilt)
uimenu ('Label', 'Медианная фильтрация', ...
       'Parent', m, ...
```

```
    'CallBack', @medfiltr)
uimenu ('Label', 'Адаптивная фильтрация Винера', ...
    'Parent', m, ...
    'CallBack', @filtrviner)

m = uimenu ('Label', 'Арифметические операции');
uimenu ('Label', 'Возвести в квадрат', ...
    'Parent', m, ...
    'CallBack', @elevateindegree2)
uimenu ('Label', 'Сложить изображения', ...
     'Parent', m, ...
    'CallBack', @sumim)
uimenu ('Label', 'Перемножить изображения', ...
        'Parent', m, ...
    'CallBack', @multipim)
uimenu ('Label', 'Вычесть из изображения X изображение У', ...
    'Parent', m, ...
    'CallBack', @substractim)
uimenu ('Label', 'Из исходного вычесть отфильтрованное', ...
     'Parent', m, ...
    'CallBack', @substrfilt)
uimenu ('Label', 'Вычесть из изображения константу', ...
        'Parent', m, ...
    'CallBack', @substractconst)
uimenu ('Label', 'Разделить изображение X на изображение У', ...
    'Parent', m, ...
    'CallBack', @divim)
uimenu ('Label', 'Разделить изображение на константу', ...
    'Parent', m, ...
    'CallBack', @divconst)

m = uimenu ('Label', 'Статистика');
uimenu ('Label', 'Гистограмма', ...
    'Parent', m, ...
    'CallBack', @gistim)
uimenu ('Label', 'Среднее значение яркости', ...
     'Parent', m, ...
    'CallBack', @meanpix)
uimenu ('Label', 'Стандартное отклонение', ...
        'Parent', m, ...
    'CallBack', @standdev)
uimenu ('Label', 'Двумерный коэффициент корреляции', ...
```

```
    'Parent', m, ...
    'CallBack', @corrim)

m = uimenu ('Label', 'Справка');
uimenu ('Label', 'Информация об изображении', ...
    'Parent', m, ...
    'CallBack', @iminf)
uimenu ('Label', 'О программе', ...
     'Parent', m, ...
    'CallBack', @aboutprog)


% ------------------------------------------------------------------

function varargout = teplovizor(varargin)
% TEPLOVIZOR M-file for teplovizor.fig
%      TEPLOVIZOR, by itself, creates a new TEPLOVIZOR or raises
the existing
%      singleton*.
%
%      H = TEPLOVIZOR returns the handle to a new TEPLOVIZOR or
the handle to
%      the existing singleton*.
%
%      TEPLOVIZOR('CALLBACK',hObject,eventData,handles,...) calls
the local
%      function named CALLBACK in TEPLOVIZOR.M with the given
input arguments.
%
%      TEPLOVIZOR('Property','Value',...) creates a new TEPLOVIZOR
or raises the
%      existing singleton*.  Starting from the left, property value pairs
are
%      applied to the GUI before teplovizor_OpeningFunction gets
called.  An
%      unrecognized property name or invalid value makes property
application
%      stop.  All inputs are passed to teplovizor_OpeningFcn via
varargin.
%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI allows
only one
```

```
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES
% Edit the above text to modify the response to help teplovizor
% Last Modified by GUIDE v2.5 09-Feb-2014 15:11:25
% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @teplovizor_OpeningFcn, ...
                   'gui_OutputFcn',  @teplovizor_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT
% --- Executes just before teplovizor is made visible.
function teplovizor_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to teplovizor (see VARARGIN)
% Choose default command line output for teplovizor
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
% UIWAIT makes teplovizor wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = teplovizor_OutputFcn(hObject, eventdata, han-
dles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --------------------------------------------------------------------
function openfile(hObject, eventdata, handles)
% hObject    handle to openfile (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите файл')
save imaging;
I=imread([file_path,file_name]);
set(gca,'Position',[0.02 0.03 0.91 0.93]);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
zoom;

% --------------------------------------------------------------------
function Zoomin(hObject, eventdata, handles)
% hObject    handle to Zoomin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
zoom;

% --------------------------------------------------------------------
function gcor(hObject, eventdata, handles)
% hObject    handle to gcor (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturegam.bmp');
D=inputdlg('Введите коэффициент гамма (от 0.1 до 10)', 'Введите
коэффициент гамма');
A=cell2mat(D);
B=str2double(A);
J=imadjust(I, [], [], B);
imshow(J, []);
```

```
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturesave.bmp');

% --- Executes on slider movement.
function slider10_Callback(hObject, eventdata, handles)
g=get(hObject, 'Value');
    I=imread('picturegam.bmp');
    J=imadjust(I, [ ], [ ], g);
    imshow(J, []);
    colorbar('vert');
    impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturesave.bmp');
% hObject    handle to slider10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% Hints: get(hObject,'Value') returns position of slider
%        get(hObject,'Min') and get(hObject,'Max') to determine range
of slider

% --- Executes during object creation, after setting all properties.
function slider10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to slider10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: slider controls usually have a light gray background.
if isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor',[.9 .9 .9]);
end

% --------------------------------------------------------------------
function corgist(hObject, eventdata, handles)
% hObject    handle to corgist (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=histeq(I, 256);
```

```
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function close(hObject, eventdata, handles)
% hObject    handle to close (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
a = questdlg('Несохраненные данные будут потеряны. Продолжить
?','Выход','Yes','No','Yes');
if a=='Yes'
    delete picture.bmp;
    delete picturegam.bmp;
    delete picturesave.bmp;
    delete picturefilt.bmp;
    delete imaging.mat;
    delete(gcf);
    close all;
else
    return;
end

% --------------------------------------------------------------------
function medfiltr(hObject, eventdata, handles)
% hObject    handle to medfiltr (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
S={'[2 2]','[4 4]','[6 6]','[8 8]','[4 2]','[8 4]'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Enter domain',
'ListString', S, 'PromptString','Задайте параметр domain', 'Selec-
tionMode', 'single');
Num=ind;
switch Num
case 1
    I1=medfilt2 (I, [2 2]);
case 2
    I1=medfilt2 (I, [4 4]);
```

```
case 3
   I1=medfilt2 (I, [6 6]);
case 4
   I1=medfilt2 (I, [8 8]);
case 5
   I1=medfilt2 (I, [4 2]);
case 6
   I1=medfilt2 (I, [8 4]);
end
imshow(I1, []);
colorbar('vert');
impixelinfo;
imwrite(I1,'picture.bmp');
imwrite(I1,'picturegam.bmp');
imwrite(I1,'picturesave.bmp');

% --------------------------------------------------------------------
function filtrviner(hObject, eventdata, handles)
% hObject    handle to filtrviner (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
S={'[2 2]','[4 4]','[6 6]','[8 8]','[4 2]','[8 4]'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Enter domain',
'ListString', S, 'PromptString','Задайте параметр domain', 'Selec-
tionMode', 'single');
Num=ind;
switch Num
case 1
   I2=wiener2(I, [2 2]);
case 2
   I2=wiener2(I, [4 4]);
case 3
   I2=wiener2(I, [6 6]);
case 4
   I2=wiener2(I, [8 8]);
case 5
   I2=wiener2(I, [4 2]);
case 6
   I2=wiener2(I, [8 4]);
end
imshow (I2, []);
```

```
colorbar('vert');
impixelinfo;
imwrite(I2,'picture.bmp');
imwrite(I2,'picturegam.bmp');
imwrite(I2,'picturesave.bmp');

% --------------------------------------------------------------------
function save2(hObject, eventdata, handles)
% hObject    handle to save (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uiputfile('*.*','Сохранить изображение как...')
S=imread('picturesave.bmp');
imwrite(S,[file_path,file_name]);

% --------------------------------------------------------------------
function rangfilt(hObject, eventdata, handles)
% hObject    handle to rangfilt (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
S={'[2 2]','[4 4]','[8 8]'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Enter domain',
'ListString', S, 'PromptString','Задайте параметр domain', 'Selec-
tionMode', 'single');
Num=ind;
switch Num
case 1
    prompt={'Введите параметр order (целое число от 1 до 4)'};
    dlg_title='Параметр order';
    num_lines = 1;
    def={'2'};
    D=inputdlg(prompt, dlg_title, num_lines, def);
    A=cell2mat(D);
    B=str2double(A);
    F=ordfilt2 (I, B, ones(2,2));
case 2
    prompt={'Введите параметр order (целое число от 1 до 16)'};
    dlg_title='Параметр order';
    num_lines = 1;
    def={'4'};
    D=inputdlg(prompt, dlg_title, num_lines, def);
```

```
    A=cell2mat(D);
    B=str2double(A);
    F=ordfilt2 (I, B, ones(4,4));
case 3
    prompt={'Введите параметр order (целое число от 1 до 64)'};
    dlg_title='Параметр order';
    num_lines = 1;
    def={'32'};
    D=inputdlg(prompt, dlg_title, num_lines, def);
    A=cell2mat(D);
    B=str2double(A);
    F=ordfilt2 (I, B, ones(8,8));
end
imshow (F);
colorbar('vert');
impixelinfo;
imwrite(F,'picture.bmp');
imwrite(F,'picturegam.bmp');
imwrite(F,'picturesave.bmp');

% --------------------------------------------------------------------
function aboutprog(hObject, eventdata, handles)
% hObject    handle to aboutprog (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
X = imread('ATT00042_11.bmp');
[m n k]=size(X);
figure('Name','O
программе...','NumberTitle','off','Resize','off','Units','pixels','Position',[1
00 100 n m],'WindowStyle','modal');
set(gca,'Position',[0 0 1 1]);
imshow (X);

% --------------------------------------------------------------------
function Print(hObject, eventdata, handles)
% hObject    handle to Print (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
printdlg;

% --- Executes when user attempts to close figure1.
function figure1_CloseRequestFcn(hObject, eventdata, handles)
```

```
a = questdlg('Несохраненные данные будут потеряны. Продолжить
?','Выход','Yes','No','Yes');
if a=='Yes'
    delete picture.bmp;
    delete picturegam.bmp;
    delete picturesave.bmp;
    delete picturefilt.bmp;
    delete imaging.mat;
    delete(gcf);
    close all;
else
    return;
end

% --------------------------------------------------------------------
function neg(hObject, eventdata, handles)
% hObject    handle to neg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imcomplement(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

function white(hObject, eventdata, handles)
% hObject    handle to neg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imnoise(I,'gaussian');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

function puasson(hObject, eventdata, handles)
```

```
% hObject    handle to neg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imnoise(I,'poisson');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

function solt(hObject, eventdata, handles)
% hObject    handle to neg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imnoise(I,'salt & pepper',0.02);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --- Executes on button press in invers.
function invers(hObject, eventdata, handles)
% hObject    handle to invers (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imcomplement(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function iminf(hObject, eventdata, handles)
% hObject    handle to iminf (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
K=imfinfo ('picturesave.bmp')
S={'Размер файла','Формат','Ширина','Высота','Глубина цве-
та','Разрешение','Тип сжатия','Тип изображения','Количество цве-
товых оттенков','MaxSampleValue','MinSampleValue'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Select parameter',
'ListString', S, 'PromptString','Выберите параметр изображения', 'Se-
lectionMode', 'single');
Num=ind;
switch Num
case 1
    Fs=K.FileSize/1024;
    Fs1=num2str(Fs);
    Fs2=char(Fs1);
    helpdlg(Fs2,'Размер файла (КБ)');
case 2
    Frt=K.Format;
    Frt=char(Frt);
    helpdlg(Frt,'Формат');
case 3
    W=K.Width;
    W1=num2str(W);
    W2=char(W1);
    helpdlg(W2,'Ширина (pix)');
case 4
    H=K.Height;
    H1=num2str(H);
    H2=char(H1);
    helpdlg(H2,'Высота (pix)');
case 5
    BD=K.BitDepth;
    BD1=num2str(BD);
    BD2=char(BD1);
    helpdlg(BD2,'Глубина цвета (бит)');
case 6
    BS=K.BitmapSize;
    BS1=num2str(BS);
    BS2=char(BS1);
    helpdlg(BS2,'Разрешение (pix)');
case 7
```

```
    Cmp=K.CompressionType;
    Cmp=char(Cmp);
    helpdlg(Cmp,'Тип сжатия');
case 8
    CT=K.ColorType;
    CT=char(CT);
    helpdlg(CT,'Тип изображения');
case 9
    NCU=K.NumColorsUsed;
    NCU1=num2str(NCU);
    NCU2=char(NCU1);
    helpdlg(NCU2,'Количество цветовых оттенков');
case 10
    MaSV=K.MaxSampleValue;
    MaSV1=num2str(MaSV);
    MaSV2=char(MaSV1);
    helpdlg(MaSV2,'MaxSampleValue');
case 11
    MiSV=K.MinSampleValue;
    MiSV1=num2str(MiSV);
    MiSV2=char(MiSV1);
    helpdlg(MiSV2,'MinSampleValue');
end

% --------------------------------------------------------------------
function rotate(hObject, eventdata, handles)
% hObject    handle to rotate2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
lines = 1;
def = {'90'};
G=inputdlg('Введите угол поворота (от 0 до 360)', 'Введите угол',
lines, def)
A=cell2mat(G);
B=str2double(A);
J=imrotate(I, B, 'bicubic');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
```

```
imwrite(J,'picturesave.bmp');

% --- Executes on button press in rotate2.
function rotateim(hObject, eventdata, handles)
% hObject    handle to rotate2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imrotate(I, 90, 'bicubic');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function zerkal(hObject, eventdata, handles)
% hObject    handle to zerkal (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=fliplr(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function segment(hObject, eventdata, handles)
% hObject    handle to segment (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
J=imcrop(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
```

```
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function Untitled_5(hObject, eventdata, handles)
% hObject    handle to Untitled_5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
S={'sobel','prewitt','roberts','zerocross','canny'};
[ind,ok]=listdlg('ListSize', [180 160], 'Name', 'Select method',
'ListString', S, 'PromptString','Выбор метода', 'SelectionMode', 'sin-
gle');
Num=ind;
switch Num
case 1
   M='sobel';
case 2
   M='prewitt';
case 3
   M='roberts';
case 4
   M='zerocross';
case 5
   M='canny';
end
J=edge(I,M);
figure('Name','Края
изображения','NumberTitle','off','WindowStyle','normal', 'MenuBar',
'none');
imshow(J, []);
imwrite(J,'picturesave.bmp');
zoom;

% --------------------------------------------------------------------
function contrastsprain(hObject, eventdata, handles)
% hObject    handle to contrastsprain (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
classin = class(I);
if strcmp(class(I), 'double') & max(I(:)) > 1 & ...
   ~strcmp(varargin{l}, 'log')
```

```
    I = mat2gray(I);
else
    I = im2double(I);
end
D=inputdlg('Введите степень E (при значениях более 24 преобра-
зование приближается к пороговому)', 'Введите степень');
A=cell2mat(D);
B=str2double(A);
m = mean2(I);
E = B;
g = 1./(1 + (m./(I + eps)).^E);
gs = im2uint8(mat2gray(g));
imshow(gs, []);
colorbar('vert');
impixelinfo;
imwrite(gs,'picture.bmp');
imwrite(gs,'picturegam.bmp');
imwrite(gs,'picturesave.bmp');

% --------------------------------------------------------------------
function contourgraf(hObject, eventdata, handles)
% hObject    handle to contourgraf (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
D=inputdlg('Введите количество цветов (целое число от 1 до 50)',
'Количество цветов');
A=cell2mat(D);
B=str2double(A);
figure('Name','Контурный
график','NumberTitle','off','WindowStyle','normal', 'MenuBar', 'none');
imcontour(I, B);
zoom;

% --------------------------------------------------------------------
function logtransformation(hObject, eventdata, handles)
% hObject    handle to logtransformation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
classin = class(I);
if strcmp(class(I), 'double') & max(I(:)) > 1 & ...
```

```
    ~strcmp(varargin{l}, 'log')
    I = mat2gray(I);
else
    I = im2double(I);
end
J=1*(log(1+ double(I)));
Js = im2uint8(mat2gray(J));
imshow(Js, []);
colorbar('vert');
impixelinfo;
imwrite(Js,'picture.bmp');
imwrite(Js,'picturegam.bmp');
imwrite(Js,'picturesave.bmp');

% --------------------------------------------------------------------
function porogtransformation(hObject, eventdata, handles)
% hObject    handle to porogtransformation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
P=impixel(I);
n=max(P);
m=n/255;
J=im2bw(I, m);
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function convertinmonohrom(hObject, eventdata, handles)
% hObject    handle to convertinmonohrom (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load imaging
I=imread([file_path,file_name]);
J=rgb2gray(I);
imshow(J, []);
colorbar('vert');
impixelinfo;
```

```
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function displayabovethreshold(hObject, eventdata, handles)
% hObject    handle to displayabovethreshold (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
S={'Определить с помощью указателя','Взять как среднее ариф-
метическое','Взять как стандартное отклонение'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Select manner',
'ListString', S, 'PromptString','Способ определения порогового зна-
чения', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
    P=impixel(I);
    n=max(P);
    m=n/255;
case 2
    n=mean2(I);
    m=n/255;
case 3
    n=std2(I);
    m=n/255;
end
J=im2bw(I, m);
J=im2uint8(J);
N=imcomplement(I);
N=im2uint8(N);
A=imsubtract(J,N);
imshow(A, []);
colorbar('vert');
impixelinfo;
imwrite(A,'picture.bmp');
imwrite(A,'picturegam.bmp');
imwrite(A,'picturesave.bmp');

% --- Executes on button press in source.
function source(hObject, eventdata, handles)
```

```
% hObject    handle to source (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load imaging
I=imread([file_path,file_name]);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');

% --------------------------------------------------------------------
function class8(hObject, eventdata, handles)
% hObject    handle to class (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
J=imread('picturesave.bmp');
I=im2uint8(J);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');

% --------------------------------------------------------------------
function class16(hObject, eventdata, handles)
% hObject    handle to class16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
J=imread('picturesave.bmp');
I=im2uint16(J);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');

% --------------------------------------------------------------------
function classdoub(hObject, eventdata, handles)
```

```
% hObject    handle to classdoub (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
J=imread('picturesave.bmp');
I=im2double(J);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');

% --------------------------------------------------------------------
function classlogoc(hObject, eventdata, handles)
% hObject    handle to classlogoc (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
J=imread('picturesave.bmp');
I=im2uint16(J);
P=impixel(I);
n=max(P);
m=n/65535;
B=im2bw(I, m);
imshow(B, []);
colorbar('vert');
impixelinfo;
imwrite(B,'picture.bmp');
imwrite(B,'picturegam.bmp');
imwrite(B,'picturesave.bmp');


% --------------------------------------------------------------------
function sumim(hObject, eventdata, handles)
% hObject    handle to sumim (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите файл X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите файл Y')
J=imread([file_path,file_name]);
R=imadd(I,J,'uint16');
figure('Name','Сумма
```

```
изображений','NumberTitle','off','WindowStyle','normal', 'MenuBar',
'none');
set(gca,'Position',[0.02 0.05 0.98 0.94]);
imshow(R, []);
colorbar('vert');
impixelinfo;
zoom;

% --------------------------------------------------------------------
function substractim(hObject, eventdata, handles)
% hObject    handle to substractim (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите изображение X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите изображение Y')
set(gca,'Position',[0.02 0.02 0.91 0.93]);
J=imread([file_path,file_name]);
R=imsubtract(I,J);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');

% --------------------------------------------------------------------
function divconst(hObject, eventdata, handles)
% hObject    handle to divconst (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
D=inputdlg('Введите константу', 'Константа');
A=cell2mat(D);
B=str2double(A);
R=imdivide(I, B);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');
```

```
% --------------------------------------------------------------------
function multipim(hObject, eventdata, handles)
% hObject    handle to multipim (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите файл X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите файл Y')
J=imread([file_path,file_name]);
I16=uint16(I);
J16=uint16(J);
R=immultiply(I16,J16);
figure('Name','Произведение
изображений','NumberTitle','off','WindowStyle','normal', 'MenuBar',
'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(R, []);
colorbar('vert');
impixelinfo;
zoom;

% --------------------------------------------------------------------
function resizeimage(hObject, eventdata, handles)
% hObject    handle to resizeimage (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
D=inputdlg('Введите множитель(от 0.1 до 10)', 'Введите множи-
тель');
A=cell2mat(D);
B=str2double(A);
J=imresize(I, B, 'bicubic');
imshow(J, []);
colorbar('vert');
impixelinfo;
imwrite(J,'picture.bmp');
imwrite(J,'picturegam.bmp');
imwrite(J,'picturesave.bmp');

% --------------------------------------------------------------------
function substractconst(hObject, eventdata, handles)
```

```
% hObject    handle to substractconst (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
D=inputdlg('Введите константу', 'Константа');
A=cell2mat(D);
B=str2double(A);
R=imsubtract(I, B);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');

% --------------------------------------------------------------------
function divim(hObject, eventdata, handles)
% hObject    handle to divim (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите изображение X')
I=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите изображение Y')
J=imread([file_path,file_name]);
set(gca,'Position',[0.02 0.02 0.91 0.93]);
R=imdivide(I,J);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');

% --------------------------------------------------------------------
function opennfg(hObject, eventdata, handles)
% hObject    handle to opennfg (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
figure('NumberTitle','on','WindowStyle','normal', 'MenuBar', 'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
I=imread('picturesave.bmp');
imshow(I, []);
```

```
colorbar('vert');
impixelinfo;
zoom;

% --------------------------------------------------------------------
function elevateindegree2(hObject, eventdata, handles)
% hObject    handle to elevateindegree2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
I16=uint16(I);
R=immultiply(I16,I16);
figure('Name','Квадрат
изображения','NumberTitle','off','WindowStyle','normal', 'MenuBar',
'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(R, []);
colorbar('vert');
impixelinfo;
zoom;

% --------------------------------------------------------------------
function changemap(hObject, eventdata, handles)
% hObject    handle to changemap (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
S={'autumn','colorcube','cool','copper','gray','hot','hsv','jet','lines','pink'
,'prism','spring','summer','winter'};
[ind,ok]=listdlg('ListSize', [180 160], 'Name', 'Select palette',
'ListString', S, 'PromptString','Выберите тип палитры', 'Selec-
tionMode', 'single');
Num=ind;
switch Num
case 1
   M='autumn';
case 2
   M='colorcube';
case 3
   M='cool';
case 4
   M='copper';
```

```
case 5
    M='gray';
case 6
    M='hot';
case 7
    M='hsv';
case 8
    M='jet';
case 9
    M='lines';
case 10
    M='pink';
case 11
    M='prism';
case 12
    M='spring';
case 13
    M='summer';
case 14
    M='winter';
end
figure('NumberTitle','on','WindowStyle','normal', 'MenuBar', 'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(I, []);
colormap(M)
colorbar('vert');
impixelinfo;
zoom;

% --- Executes on button press in newwin.
function newwin(hObject, eventdata, handles)
% hObject    handle to newwin (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
figure('NumberTitle','on','WindowStyle','normal', 'MenuBar', 'none');
set(gca,'Position',[0.02 0.05 0.98 0.92]);
imshow(I, []);
colorbar('vert');
impixelinfo;
zoom;
```

```
% --------------------------------------------------------------------
function linfilt(hObject, eventdata, handles)
% hObject    handle to linfilt (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picture.bmp');
F1=fspecial('average', [3 3]);
F2=fspecial('disk', 3);
F3=fspecial('gaussian', [3 3], 0.5);
F4=fspecial('laplacian', 0.5);
F5=[1 1 1; 1 -8 1; 1 1 1];
F6=fspecial('log', [5 5], 0.5);
F7=fspecial('motion', 6, 0);
F8=fspecial('prewitt');
F9=fspecial('sobel');
F10=fspecial('unsharp', 0.2);
S={'Прямоугольный усредняющий фильтр','Круговой усредняю-
щий фильтр','Низкочастотный гауссов фильтр','Фильтр Лапласа
4','Фильтр Лапласа 8','Лаплас от гауссова фильтра (LoG)
','Motion','Фильтр Превитта','Фильтр Собела','Нечеткий фильтр'};
[ind,ok]=listdlg('ListSize', [240 140], 'Name', 'Type filtr', 'ListString', S,
'PromptString','Выберите тип фильтра', 'SelectionMode', 'single');
Num=ind;
switch Num
case 1
   F=F1;
case 2
   F=F2;
case 3
   F=F3;
case 4
   F=F4;
case 5
   F=F5;
case 6
   F=F6;
case 7
   F=F7;
case 8
   F=F8;
case 9
   F=F9;
```

```
case 10
    F=F10;
end
G=imfilter(I, F, 'replicate');
imshow(G, []);
colorbar('vert');
impixelinfo;
imwrite(G,'picture.bmp');
imwrite(G,'picturegam.bmp');
imwrite(G,'picturesave.bmp');
imwrite(G,'picturefilt.bmp');

% --------------------------------------------------------------------
function substrfilt(hObject, eventdata, handles)
% hObject    handle to substrfilt (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
load imaging
I=imread([file_path,file_name]);
F=imread('picturefilt.bmp');
R=imsubtract(I, F);
imshow(R, []);
colorbar('vert');
impixelinfo;
imwrite(R,'picture.bmp');
imwrite(R,'picturegam.bmp');
imwrite(R,'picturesave.bmp');

% --------------------------------------------------------------------
function meanpix(hObject, eventdata, handles)
% hObject    handle to meanpix (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
M=mean2(I);
M1=num2str(M);
M2=char(M1);
helpdlg(M2,'Среднее значение элементов матрицы изображения');

% --------------------------------------------------------------------
function standdev(hObject, eventdata, handles)
% hObject    handle to standdev (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
M=std2(I);
M1=num2str(M);
M2=char(M1);
helpdlg(M2,'Стандартное отклонение элементов матрицы изобра-
жения');

% --------------------------------------------------------------------
function gistim(hObject, eventdata, handles)
% hObject    handle to gistim (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
I=imread('picturesave.bmp');
figure('Name','Гистограмма','NumberTitle','off','WindowStyle','normal',
'MenuBar', 'none');
imhist(I, 256);
axis([0 255 0 15000]);
set(gca, 'xtick', 0:50:255);
set(gca, 'ytick', 0:1000:15000);

% --------------------------------------------------------------------
function corrim(hObject, eventdata, handles)
% hObject    handle to corrim (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите изображение A')
A=imread([file_path,file_name]);
[file_name,file_path]=uigetfile('*.*','Выберите изображение B')
B=imread([file_path,file_name]);
r=corr2(A,B);
r1=num2str(r);
r2=char(r1);
helpdlg(r2,'Коэффициент корреляции');

% --------------------------------------------------------------------
function opendicom(hObject, eventdata, handles)
% hObject    handle to opendicom (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
[file_name,file_path]=uigetfile('*.*','Выберите файл (*.dcm, *.dic)')
```

```
I=dicomread([file_path,file_name]);
set(gca,'Position',[0.02 0.03 0.91 0.93]);
imshow(I, []);
colorbar('vert');
impixelinfo;
imwrite(I,'picture.bmp');
imwrite(I,'picturegam.bmp');
imwrite(I,'picturesave.bmp');
```

СПИСОК ЛИТЕРАТУРЫ

1. Гультяев, А. П. Визуальное моделирование в среде MATLAB: учеб. Курс / А.П. Гультяев.– СПб.: Питер, 2000.– 432 с.
2. Дьяконов, В.П. MATLAB: учеб. курс / В.П. Дьяконов. – СПб.: Питер, 2001.– 560 с.
3. Дьяконов, В.П. Simulink 4. Специальный справочник / В.П. Дьяконов.– СПб.: Питер, 2002.– 528 с.
4. Кетков, Ю.Л. MATLAB 6.x . Программирование численных методов / Ю.Л. Кетков, А.Ю Кетков, М.М. Шульц.– СПб.: БХВ-Петербург, 2004.–672 с.
5. Конев, В.Ю. Основные функции пакета MATLAB: учеб.пособие / В.Ю. Конев, Л.А. Мироновский.– 2-ое издание.– СПб.: ГААПСПб., 1994.–76 л.
6. Потемкин, В. Г. Система MATLAB: справ. Пособие / В.Г. Потемкин.– М.: ДИАЛОГ-МИФИ, 1997.– 350 с.