

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ГОУВПО «СИБИРСКАЯ ГОСУДАРСТВЕННАЯ ГЕОДЕЗИЧЕСКАЯ АКАДЕМИЯ»

Т.В. Дашковская

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ В СРЕДЕ MATLAB

Часть 2

Утверждено редакционно-издательским советом академии
в качестве лабораторного практикума для студентов, обучающихся
по направлению 200200 «Оптотехника», специальности 200203
«Оптико-электронные приборы и системы»

Новосибирск
СГГА
2010

УДК 528.7:004
Д217

Рецензенты: кандидат технических наук, доцент, НГТУ Г.А. Сырецкий
доктор технических наук, профессор, СГГА В.В. Малинин

Дашковская Т.В.

Д217 Цифровая обработка сигналов в среде matlab [Текст]: лабораторный практикум в 2 ч. Ч. 2 / Т.В. Дашковская. – Новосибирск: СГГА, 2010. – 83 с.

ISBN 978-5-87693-400-0 (ч. 2)
ISBN 978-5-87693-398-0

Данный лабораторный практикум по дисциплинам «Теория изображений в оптико-электронных системах», «Цифровая обработка изображений» (4 курс, бакалавры), «Компьютерное моделирование и проектирование» (3 курс), «Специальные главы ЦОИ» (6 курс, магистры) предназначен для студентов, обучающихся по направлению 200200 «Оптехника», специальности 200203 «Оптико-электронные приборы и системы».

В практикуме приведены лабораторные работы по цифровой обработке изображений с использованием пакета системы Image Processing ToolBoxes.

Ответственный редактор: кандидат технических наук, доцент, СГГА Е.В. Грицкевич

Печатается по решению редакционно-издательского совета СГГА

УДК 528.7:004

ISBN 978-5-87693-400-0 (ч. 2)
ISBN 978-5-87693-398-0

© ГОУ ВПО «Сибирская государственная геодезическая академия»
(СГГА), 2010

СОДЕРЖАНИЕ

Лабораторная работа № 1. Типы изображений и работа с файлами изображений	4
Лабораторная работа № 2. Дискретизация и квантование изображения.....	12
Лабораторная работа № 3. Геометрические преобразования изображения	19
Лабораторная работа № 4. Амплитудные преобразования изображения....	27
Лабораторная работа № 5. Функции, используемые для анализа изображений	35
Лабораторная работа № 6. Функции фильтрации изображения	41
Лабораторная работа № 7. Восстановление изображения	50
Лабораторная работа № 8. Бинаризация изображений	55
Лабораторная работа № 9. Морфологические операции над бинарными изображениями	60
Лабораторная работа № 10. Сегментация изображений	70
Лабораторная работа № 11. Функции поиска объектов и вычисления их признаков	79
Лабораторная работа № 12. Функции спектрального анализа изображения	85
Список рекомендуемой литературы для углубленного изучения материала	90

ЛАБОРАТОРНАЯ РАБОТА № 1. ТИПЫ ИЗОБРАЖЕНИЙ И РАБОТА С ФАЙЛАМИ ИЗОБРАЖЕНИЙ

Цель работы – изучение типов изображений, функций (средств) системы MatLab, применяемых для работы с файлами изображений и приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Типы изображений

По способу сохранения описание изображения может быть:

- Векторным – изображение описывается как набор графических примитивов, из которых и формируется изображение;
- Растровым – изображение описывается двумерным массивом, каждый элемент которого представляет собой некоторое описание цвета.

Элемент растрового изображения называют пикселом (от pixel, picture element – элемент изображения), или точкой.

Объем памяти в байтах, необходимый для хранения растрового изображения, можно вычислить по формуле:

$$V = (c \cdot r \cdot d) / 8,$$

где c – количество столбцов; r – количество строк; d – глубина цвета (бит/пиксел).

Существуют следующие типы изображений:

- Бинарные – (black and white) пикселы могут принимать только два значения: 0 и 1 (черный и белый цвет);
- Полутоновые (серые или изображениями в градациях серого intensity, grayscale) – пикселы могут принимать одно из значений интенсивности какого-либо одного цвета в диапазоне от минимальной до максимальной интенсивности;
- Палитровые (indexed) – значения пикселей являются ссылками на ячейки карты цветов (colormap), которые и содержат описания цвета пиксела в некоторой цветовой системе (палитре);
- Полноцветные или просто цветные (truecolor, rgb) – изображения, пикселы которого непосредственно хранят информацию об интенсивностях цветовых составляющих.

Пикселы изображений, представленных массивами в формате double и uint8, должны удовлетворять требованиям, представленным в табл. 1.1.

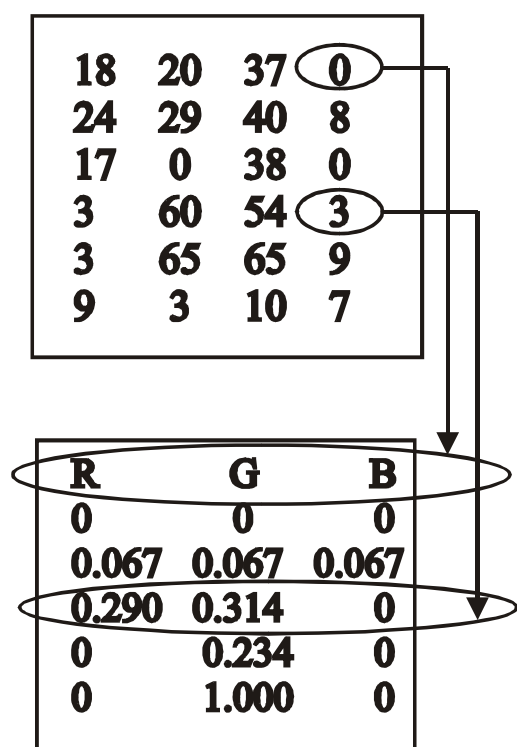
Таблица 1.1 Диапазоны представления пикселей изображения

Тип изображения	double	uint8
Бинарное	Значения 0 и 1	Значения 0 и 1
Полутоновое	Диапазон значений [0, 1]	Диапазон значений [0, 255]
Палитровое	Диапазон значений [1, размер палитры], значение	Диапазон значений [0, 255], значение 0 указывает

	1 указывает на первую строку палитры	на первую строку палитры
Полноцветное	Диапазон значений [0, 1]	Диапазон значений [0, 255]

Полутонные и бинарные изображения хранятся в виде двумерных массивов. Для доступа к значению (в данном случае яркости) пиксела изображения I надо указать строку r и столбец c : $I(r, c)$.

Полноцветные изображения хранятся в виде трехмерных массивов, где



третье измерение – значения интенсивности R , G , B . Для доступа к значениям интенсивности цветовых составляющих пиксела I надо указать строку r , столбец c и номер составляющей: 1 – для R , 2 – для G , 3 – для B , например, $I(r, c, 1)$ позволяет получить значение красной составляющей.

Палитровые изображения хранятся в виде двумерных массивов индексов. Для каждого палитрового изображения существует двумерный массив палитры. Массив палитры всегда имеет тип `double`, и в трех его столбцах расположены интенсивности R , G , B . Пример палитрового изображения, использующего формат представления данных `uint8`, приведен на рис. 1.1.

Рис. 1.1

Принятые обозначения

В описании функций и в примерах применяются следующие обозначения для различных типов изображений:

- I – полутонные; X – палитровые; RGB – полноцветные; BW – бинарные;
- S – для исходного изображения любого типа; D – результирующее изображение.

Символ «многоточие» (...) в описании функции означает, что может быть использован любой упомянутый ранее в описании набор входных или выходных параметров.

В системе MatLab функции по обработке изображения находятся в пакете Image Processing Toolboxes (IPT).

Работа с файлами изображений

Функция чтения из файла информации об изображении `imfinfo`

Синтаксис

`info = imfinfo(<имя файла>)`

В структуре `info` возвращается информация об изображении и способе его хранения из файла с именем `<имя файла>`; имя включает в себя путь к файлу, его имя и расширение, например, `'c:\mmm\Earth.bmp'`.

Графические форматы файлов, с которыми система MatLab поддерживает работу, приведены в табл. 1.2.

Таблица 1.2 Форматы файлов изображений

Формат файла	Название формата
'bmp'	Windows Bitmap (BMP)
'tif' или 'tiff'	Tag Image File Format (TIFF)
'jpg' или 'jpeg'	Joint Photographic Experts Group (JPEG)

Информация об изображении и способе его хранения в данном файле возвращается в структуре `info`. Структуры для разных форматов файлов отличаются друг от друга. Общие для всех форматов первые 9 полей структуры, по которым можно определить формат файла, тип и размеры изображения, приведены в табл. 1.3.

Таблица 1.3 Описание первых девяти полей структуры `info`

Имя поля	Тип данных	Описание
Filename	Строка	Имя файла, если файл находится в текущей директории, или полный путь к файлу
FileModDate	Строка	Дата и время последней модификации файла
FileSize	Число	Размер файла в байтах
Format	Строка	Формат файла
FormatVersion	Строка или число	Версия формата
Width	Число	Ширина изображения в пикселах
Height	Число	Высота изображения в пикселах
BitDepth	Число	Глубина цвета изображения в битах на пиксел
ColorType	Строка	Тип изображения: 'truecolor' или 'RGB' – для полноцветных изображений; 'grayscale' – для полутоновых изображений; 'indexed' – для палитровых изображений

В файлах форматов TIFF и HDF может храниться несколько изображений. В этом случае `info` является массивом структур.

Задание 1. Получить информацию об изображении, хранящемся в файле на диске с:

```
>> f = imfinfo('c:\Image\Athena.bmp').
```

Функция чтения изображения из файла `imread`

Синтаксис

D = imread(<имя файла>) – читает из файла <имя файла> непалитровое изображение и помещает его в массив D, (<имя файла>) включает в себя путь к файлу, его имя и расширение, например, 'c:\mmm\Athena.bmp'.

[X, map] = imread(<имя файла>) – читает из файла с именем <имя файла> палитровое изображение X с палитрой map.

Прочитанное из файла изображение имеет формат представления данных uint8.

Задание 2. Прочитать палитровое изображение из файла Handshak.bmp.

```
>> [X,map] = imread('c:\Image\Handshak.bmp');
```

Функция записи изображения в файл imwrite

Синтаксис

imwrite(S, <имя файла>) – записывает в файл с именем <имя файла> бинарное, полутоновое или полноцветное изображение S.

imwrite(X, map, <имя файла>) – записывает в файл с именем <имя файла> палитровое изображение X с палитрой map.

Вывод изображения на экран

Функция вывода изображения на экран imshow

Синтаксис

imshow(S) – вывод непалитрового изображения.

imshow(I,n) – вывод полутонового изображения I, по умолчанию n = 256.

imshow(I,[low high]) – вывод полутонового изображения с контрастированием.

imshow(X,map) – вывод палитрового изображения X, map – палитра.

imshow (<имя файла>) – вывод изображения непосредственно с диска.

Функция imshow(I,[low high]) выводит на экран полутоновое изображение I, дополнительно контрастируя выводимое изображение. Пикселы, яркость которых меньше/равна low, отображаются черным цветом, больше/равна high – белым. Все уровни серого равномерно распределены от low до high. Если вторым параметром указать пустой массив [], то low = min(I(:)), high = max(I(:)).

Для того чтобы вывести на экран прочитанное палитровое изображение из предыдущего примера, надо использовать команду:

```
>> imshow(X,map).
```

Вывод нескольких изображений в одном окне

В сочетании с функцией subplot для вывода нескольких изображений в одном окне используется функция subimage:

Синтаксис

subimage(S) – вывод в графическое окно бинарного, полутонового или полноцветного изображения.

subimage(X,map) – вывод в графическое окно палитрового изображения X с палитрой map.

Преобразования классов данных и типов изображений

Классы данных представления изображений

Двумя основными классами данных (форматами) представления элементов массива изображений являются:

- `double` – в виде действительных чисел двойной точности; каждый элемент формата `double` занимает 8 байт памяти;
- `uint8` – значение пиксела есть беззнаковое целое однобайтовое число, диапазон возможных значений которого $[0, 255]$.

Для представления изображения в формате `double` используется функция `im2double`.

Синтаксис

`D = im2double(S)` – преобразует бинарное, полутоновое или полноцветное изображение `S` в формат `double` и осуществляет приведение значений пикселей к диапазону $[0,1]$.

`XD = im2double(XS, 'indexed')` – преобразует палитровое изображение `XS` в формат `double`.

Функция `mat2gray` позволяет преобразовывать произвольный массив `S` формата `double` в перенормированный `D`. Изображение `D` имеет значения пикселей в интервале от 0 (белый) до 1 (черный).

Синтаксис

`D = mat2gray(S)`

Для представления изображения в формате `uint8` используется функция `im2uint8`.

Синтаксис

`D = im2uint8 (S)` – преобразует бинарное, полутоновое или полноцветное изображение `S` в формат `uint8`; представляет все пиксели в виде целых неотрицательных чисел в диапазоне $[0, 255]$.

`XD = im2uint8 (XS, 'indexed')` – преобразует палитровое изображение `XS` в формат `uint8`.

Преобразования типов изображений

Существуют ряд функций для преобразований изображений из одного типа в другой:

`I = rgb2gray(RGB)` – преобразование полноцветного изображения в полутоновое;

`I = ind2gray(X,map)` – преобразование палитрового изображения в полутоновое;

`[X map] = gray2ind(I,n)` – преобразование полутонового изображения в палитровое, `n` по умолчанию равно 64;

`RGB = ind2rgb(X,map)` – преобразование палитрового изображения в полноцветное;

`[X map] = rgb2ind(RGB)` – преобразование полноцветного изображения в палитровое.

Задание 3. Преобразовать палитровое изображение, прочитанное из файла 'Handshak.bmp' каталога Image, в полутоновое и вывести изображения в одно окно по горизонтали.


```
>> [X,map] = imread('c:\Image\Handshak.bmp');  
>> Xd = im2double(X, 'indexed');  
>> I = ind2gray(X,map);  
>> figure,subplot(1,2,1),subimage(X,map);  
>> subplot(1,2,2),subimage(I).
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Получить информацию об изображении из файла 'butterfly.bmp'.
2. Прочитать изображение из файла 'butterfly.bmp', преобразовать его в изображение: а) полутоновое б) палитровое. Полутоновое изображение вывести на экран, дополнительно отконтрастировав таким образом, чтобы пиксеты, яркость которых меньше или равна 0.2, отображалась черным цветом, а пиксеты, яркость которых больше или равна 0.8 – белым.
3. Преобразовать пиксеты полноцветного и полутонового изображений в формат double.
4. Вывести полноцветное и полутоновое изображения на экран в одно окно, разместив их по вертикали.

3. ВОПРОСЫ

1. Какие классы данных (форматы) представления пикселей изображения существуют?
2. Какие типы растровых изображений используются в пакете IPT?
3. С помощью какой функции можно получить информацию о размере, типе изображения?
4. С какими форматами графических файлов можно работать в системе MatLab?
5. С помощью каких функций можно прочитать изображение из файла на диске и записать изображение на диск?
6. Какие аргументы функции `imshow` изменяют контраст полутонового изображения при его выводе на экран?
7. Какие вы знаете функции преобразования типов изображений?

ЛАБОРАТОРНАЯ РАБОТА № 2. ДИСКРЕТИЗАЦИЯ И КВАНТОВАНИЕ ИЗОБРАЖЕНИЯ

Цель работы – изучение функций, используемых для моделирования процессов квантования и дискретизации изображения, а также знакомство с М-файлами и приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Процессы дискретизации и квантования изображения

Непрерывное изображение $f(x, y)$ является функцией двух пространственных переменных x и y на ограниченной прямоугольной области (рис. 2.1).

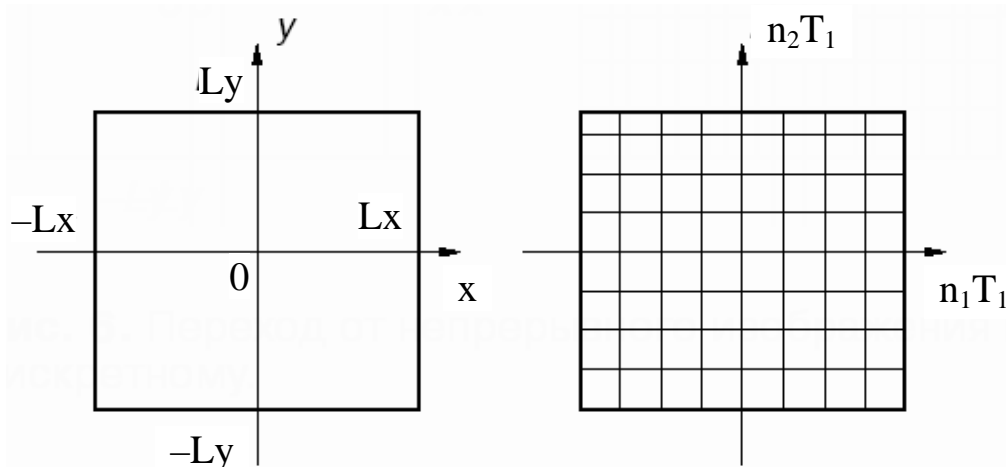


Рис. 2.1

По пространственной переменной x задается шаг дискретизации T_1 , по переменной y – T_2 . В точках, удаленных друг от друга на расстояние T_1 по оси x , расположены точечные видеодатчики. Если такие видеодатчики установить по всей прямоугольной области, то изображение окажется заданным на двумерной решетке:

$$f(n_1T_1, n_2T_2) = f(x, y) \mid x = n_1T_1, y = n_2T_2;$$

$$f(n_1T_1, n_2T_2) \equiv f(n_1, n_2).$$

Таким образом, дискретизация изображения по пространственным переменным переводит его в таблицу выборочных значений. Размерность таблицы (число строк и столбцов) определяется геометрическими размерами исходной прямоугольной области и выбором шага дискретизации по формуле:

$$M_x = [2L_x / T_1], M_y = [2L_y / T_2].$$

Элемент таблицы, полученный путем дискретизации, называется пиксел (picture element). Шаги дискретизации T_1 , T_2 должны выбираться достаточно малыми для того, чтобы погрешность дискретизации была незначительна и цифровое представление сохраняло основную информацию. С физической точки зрения выбор шага дискретизации диктуется шириной пространственного спектра изображения. Чем больше ширина спектра w , тем меньше шаг

дискретизации Т. Практически при дискретизации стремятся удовлетворить соотношению:

$$T \leq 2\pi / w.$$

Память компьютера способна хранить только дискретные числа. Поэтому для записи в памяти величина $f(x, y)$ должна быть подвергнута аналого-цифровому преобразованию с шагом df . Операцию дискретизации непрерывной величины по уровням часто называют квантованием. Число уровней квантования:

$$K = [A / df],$$

где A определяет диапазон значений яркостей функции $f(x, y)$.

Значения яркостей исходного изображения сравниваются с заданными уровнями квантования (шкалой) и принимают новое значение ближайшего уровня.

Функции, используемые для моделирования процессов дискретизации и квантования изображения

Функция обработки блоков изображения **blkproc**

Синтаксис

$$D = \text{blkproc}(S, [m \ n], \text{fun}, P1, P2)$$

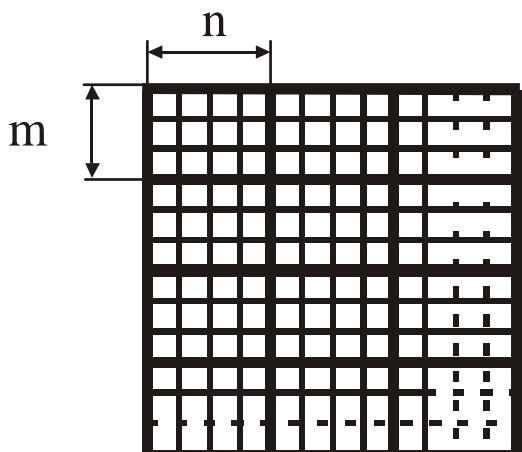


Рис. 2.2

Функция $D = \text{blkproc}(S, [m \ n], \text{fun}, P1, P2)$ формирует новое изображение D , пиксеты которого являются результатом обработки функцией fun каждого неперекрывающегося блока размера $m \times n$ исходного изображения S . Исходное изображение временно дополняется столбцами нулей справа и строками нулей снизу так, чтобы размеры изображения были кратны соответствующим размерам блока (рис. 2.2).

Функция используется для обработки бинарных и полутоновых изображений.

Варианты задания параметра fun приведены в табл. 2.1. Идентификатор x является условным обозначением обрабатываемого блока изображения.

Таблица 2.1 Варианты задания параметра fun функции **blkproc**

Параметр fun	Пример	Комментарий
Строка – имя функции	$D = \text{blkproc}(S, [m \ n], 'myfun');$	Предварительно должна быть создана функция myfun , которая получает блок x и возвращает результат $y = \text{myfun}(x)$
Строка – выражение языка	$D = \text{blkproc}(S, [m \ n], 'mean2(x)*ones(size(x))');$	В строку помещается выражение, допустимое синтаксисом MatLab. Это выражение интерпретируется при выполнении каждого блока

P1, P2 и т. д. – передаваемые в функцию fun дополнительные параметры.

При проведении вычислений исходное изображение временно дополняется либо единицами (при формате представления данных X_S – double), либо нулями (при формате представления данных X_S – uint8).

Задание 1. Создать функцию по имитации дискретизации изображения:

1. Открыть новый М-файл – File/New/M-file;
2. Ввести команды функции дискретизации

```
function disk(I,s)
I1 = blkproc(I, [s(1) s(2)], 'mean2(x)*ones(size(x))');
figure, imshow(I1)
str = mat2str(s);
str = strcat('diskret image', str);
title(str);
```

3. Сохранить на диске в файл с именем созданной функции.

Функция квантования сигнала quantiz

В системе MatLab функции по квантованию сигналов имеются в пакете Signal Processing Toolboxes. Описание одной из функций квантования приведено ниже.

Функция quantiz используется для неравномерного квантования сигналов.

Синтаксис

```
[indx, kv, ds] = quantiz(sg, prt, cdb),
```

где sg – отсчеты сигнала; prt – вектор границ зон квантования; cdb – вектор уровней квантования.

Для indx результат определяется следующим образом:

- Если $sg(k) \leq prt(1)$, то $indx(k) = 0$;
- Если $prt(1) < sg(k) \leq prt(m+1)$, то $indx(k) = m$;
- Если $sg(k) > prt(end)$, то $indx(k) = length(prt)$;

Результат kv содержит квантованные значения входного сигнала и рассчитывается как $cdb(indx + 1)$. Таким образом, величина $cdb(1)$ задает уровень квантования для всех значений сигнала, меньших, чем $prt(1)$; величина $cdb(m)$ соответствует зоне квантования $prt(m-1) \dots prt(m)$; последнее значение $cdb(end)$ используется в качестве уровня квантования для всех значений сигнала, превышающих $prt(end)$.

Параметр ds содержит значения среднего квадрата ошибок квантования, их можно опустить.

Задание 2. Использовать функцию quantiz для квантования изображения файла Clouds.bmp каталога Image.

```
>> [S, map] = imread('c:\Image\Clouds.bmp');
>> I = im2double(ind2gray(S, map));
>> [M N] = size(I);
```

```
>> p = 0:0.1:1;
>> [INDX] = quantiz(I(:, p);
>> figure,imshow(I)
>> I1 = reshape(INDX,M,N);
>> figure,imshow(I1,[])
```

Задание 3. Создать функцию по квантованию изображения:

- 1) Открыть новый М-файл – File/New/M-file;
- 2) Ввести команды функции дискретизации:

```
function kvant (I,p);
[M N] = size(I);
INDX = quantiz(I(:, p);
I1 = reshape(INDX,M,N);
I2 = mat2gray(I1);
figure,imshow(I2);
title('kvant image');
```

- 3) Сохранить на диске файл с именем созданной функции.

Задание 4. Создать файл-программу с меню для выполнения моделирования процессов квантования и дискретизации:

- 1) Открыть новый М-файл – File/New/M-file;
- 2) Ввести команды файл-программы:

```
y = 1;
while y == 1
St = input('ВВЕДИ ПУТЬ И ИМЯ ФАЙЛА С ИЗОБРАЖЕНИЕМ ');
[X,map] = imread(St);
I = im2double (ind2gray(X,map));
figure,imshow(I)
title('gray image');
```

```
y1 = 1;
while y1 == 1
M = menu('МОДЕЛИРОВАНИЕ ПРОЦЕССОВ', ...
'КВАНТОВАНИЕ' 'ИЗОБРАЖЕНИЯ','ДИСКРЕТИЗАЦИЯ
```

ИЗОБРАЖЕНИЯ');

```
if M == 1
p = input('ВВЕДИ ВЕКТОР ГРАНИЦ ЗОН КВАНТОВАНИЯ ([ma:d:mi]) ');
kvant(I,p);
end
```

```
if M == 2
s = input('Введи размер блока дискретизации ([m n]) ');
diskr(I,s);
end
```

```
y1 = input('Продолжить работу с этим изображением? (1–да) – ');
end
```

```
y = input('Продолжить работу (1–да) – ');
```

end

3) Сохранить на диске – File/Save as в файл. В качестве имени файла можно взять инициалы имени и фамилии в латинском регистре;

4) Для вызова файл-программы в окне команд (Command Window) вводим имя М-файла.

Результаты работы приведены на рис. 2.3.

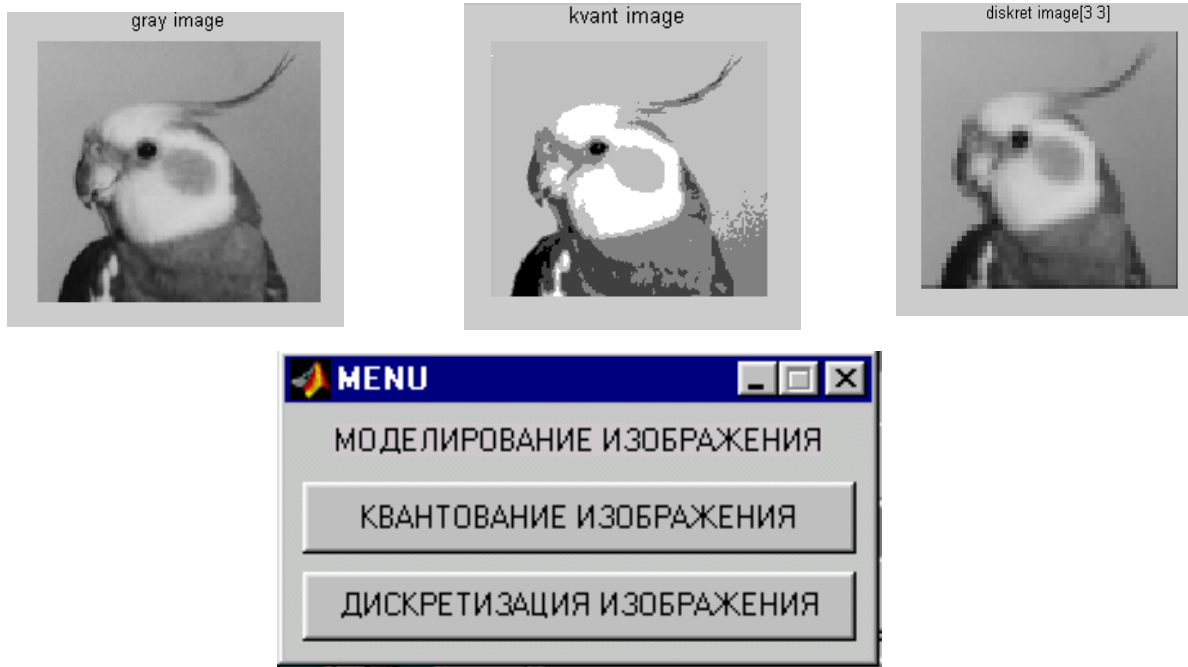


Рис. 2.3

2. ЗАДАНИЕ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Откорректировать функции квантования и дискретизации и файл-программу, приведенные в лабораторной работе, так, чтобы все результаты выдавались в одном окне.

3. ВОПРОСЫ

1. Каким образом осуществляется дискретизация сигнала?
2. Как выбирается величина шага дискретизации?
3. Каким образом осуществляется квантование сигнала?
4. Для чего используется функция а) `quantiz`; б) `blkproc`?

ЛАБОРАТОРНАЯ РАБОТА № 3. ГЕОМЕТРИЧЕСКИЕ ПРЕОБРАЗОВАНИЯ ИЗОБРАЖЕНИЯ

Цель работы – изучение функций, используемых для геометрических преобразований, и приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Функции геометрических преобразований изображения

Геометрические преобразования изменяют пространственное местоположение элементов в изображении.

Функция вырезания фрагмента из изображения `imcrop`

Синтаксис

[D,rect] = imcrop (S) – для непалитровых изображений, фрагмент задается мышкой.

[X_d,rect] = imcrop (X_s,map) – для палитровых изображений, фрагмент задается мышкой.

D = imcrop (S, rect) – **фрагмент задается в векторе rect.**

X_d = imcrop (X_s,map, rect) – **фрагмент задается в векторе rect.**

D = imcrop – функция оперирует с изображением в текущем графическом окне.

imcrop(S) – результат вырезания фрагмента из изображения S отображается в новом графическом окне.

Для выделения квадратного фрагмента следует при перемещении курсора мыши держать нажатой клавишу Shift.

Вектор `rect`, задающий фрагмент, содержит четыре элемента: $[X_{\min} \ Y_{\min} \ w, h]$, где X_{\min} и Y_{\min} – координаты верхнего левого угла прямоугольника; w – его ширина; h – высота.

Задание 1. Вырезать фрагмент с помощью мышки.

```
>> [S,map] = imread('c:\Image\Athena.bmp');  
>> figure,imshow(S,map);  
>> [A,rect] = imcrop(S,map); % кадрирование мышкой  
>> figure,imshow(A,map);  
>> rect
```

Задание 2. Вырезать фрагмент, заданный с помощью вектора `rect`

```
>> rect = [0,0,112.5,68.5]; % задание информации о фрагменте  
>> P = imcrop(S,map,rect);  
>> figure,imshow(P,map)
```

Функция изменения размеров изображения `imresize`

Синтаксис

D = imresize (S, m, method)

Функция создает изображение D меньше S, если m принадлежит диапазону от 0 до 1. Если m больше 1, то D больше S. Для изменения размеров используется один из методов интерполяции, который задается во входном параметре `method` в виде одной из следующих строк:

‘nearest’ – использовать значение ближайшего пиксела (установлено по умолчанию);

‘bilinear’ – использовать интерполяцию по билинейной поверхности;

‘bicubic’ – использовать интерполяцию по бикубической поверхности.

Задание 3. Увеличить полутоновое изображение

```
>> [S,map] = imread('c:\Image\Athena.bmp');  
>> I = ind2gray(S,map); % перевод в полутоновое изображение  
>> imshow(I)  
>> figure,imshow(imresize(I,2)), title('nearist')  
>> figure,imshow(imresize(I,2,'bilinear')), title('bilinear')  
>> figure,imshow(imresize(I,2,'bicubic')), title('bicubic')
```

Задание 4. Уменьшить полутоновое изображение

```
>> X1 = imresize(I,0.5);  
>> figure,imshow(X1,[]); title('nearist')  
>> X2 = imresize(I,0.5,'bilinear');  
>> figure,imshow(X2,[]); title('bilinear')  
>> X3 = imresize(I,0.5,'bicubic');  
>> figure,imshow(X3,[]); title('bicubic')
```

Функция поворота изображения imrotate

Синтаксис

D = imrotate(S, angle, method)

Функция создает изображение D, соответствующее повернутому исходному изображению S, используя один из предопределенных методов интерполяции (см. функцию imresize). Угол поворота angle задается в градусах. Положительные значения данного параметра соответствуют повороту против часовой стрелки, а отрицательные – по часовой.

Задание 5. Повернуть палитровое изображение на 45 градусов по часовой стрелке.

```
>> [D,map] = imread('c:\image\Chip.bmp');  
>> figure,subplot(1,3,1),subimage(imrotate(D,45),map)  
>> subplot(1,3,2),subimage(imrotate(D,45,'bilinear'),map);  
>> subplot(1,3,3),subimage(imrotate(D,45,'bicubic'),map);
```

Аффинные преобразования

Поворот, изменение размеров изображения относятся к геометрическим преобразованиям, представители которого называются аффинными преобразованиями. Аффинное преобразование можно записать в матричной форме:

$$\begin{bmatrix} x & y & 1 \end{bmatrix} = \begin{bmatrix} w & z & 1 \end{bmatrix} \mathbf{T} \begin{vmatrix} t_{11} & t_{12} & 0 \\ t_{21} & t_{22} & 0 \\ t_{31} & t_{32} & 0 \end{vmatrix}.$$

Такой формулой можно задать сжатие, поворот, перенос или сдвиг, соответствующим образом определяя элементы матрицы \mathbf{T} . В табл. 3.1 показано, как выбирать эти величины для совершения различных преобразований.

Таблица 3.1 Типы аффинных преобразований

Тип	Аффинная матрица \mathbf{T}	Координатное уравнения
Растяжение	$\begin{vmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{vmatrix}$	$X = S_x W$ $Y = S_y Z$
Поворот	$\begin{vmatrix} \cos(a) & \sin(a) & 0 \\ -\sin(a) & \cos(a) & 0 \\ 0 & 0 & 1 \end{vmatrix}$	$x = w \cos(a) - z \sin(a)$ $y = w \sin(a) + z \cos(a)$
Сдвиг (горизонтальный)	$\begin{vmatrix} 1 & 0 & 0 \\ a & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$	$x = w + az$ $y = z$
Сдвиг (вертикальный)	$\begin{vmatrix} 1 & b & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{vmatrix}$	$x = w$ $y = bw + z$
Перенос	$\begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ S_x & S_y & 1 \end{vmatrix}$	$X = W + S_x$ $Y = Z + S_y$

В пакете IPT пространственное преобразование задается в виде так называемой tform-структуры. Для задания структуры можно использовать функцию maketform.

Синтаксис

tform = maketform(transftype, T),

где transftype – тип преобразования (строковая константа); \mathbf{T} – матрица задания аффинного преобразования.

В табл.3.2 приведены значения типов преобразования transftype.

Таблица 3.2 Типы преобразований функции maketform

Тип преобразования	Описание
Affine	Комбинация растяжения/сжатия, поворота, сдвига и переноса. Прямые линии остаются прямыми, параллельные линии остаются параллельными
Box	Независимое растяжение/сжатие и перенос по любой размерности; подмножество аффинных преобразований

Composite	Семейство пространственных преобразований, которые применяются последовательно
Custom	Пространственное преобразование, заданное пользователем, который определяет функции для вычисления T и T-1
Projective	Как и при аффинных преобразованиях, прямые линии остаются прямыми, однако параллельные переходят в непараллельные с удаленной точкой пересечения

Для выполнения аффинных преобразований над изображением используется функция `imtransform`.

Синтаксис

`D = imtransform(I, tform, type, P)`,

где `I` – исходное изображение; `tform` – `tform`-структура пространственного преобразования; `type` – строковая константа, определяющая метод интерполяции ближайших пикселей для вычисления значения выходного пикселя. Может принимать одно из следующих значений: 'nearest', 'bilinear' и 'bicubic'. По умолчанию используется 'bilinear'. `P` – дополнительные параметры, например, параметр `P = 'FillValue'` контролирует цвет, который функция использует для пикселей, которые находятся вне исходного изображения:

```
>> D = imtransform(I, tform, 'FillValue', 0.5);
```

Для демонстрации преобразований часто используется изображение шахматной доски, которое создается функцией `checkerboard`.

Синтаксис

`I=checkerboard(N,P,Q)`,

где параметр `N` – число пикселей, определяющее размер клетки доски; параметр `P` определяет количество клеток по вертикали (`2P`); параметр `Q` определяет количество клеток по горизонтали (`2Q`). Если параметры `P` и `Q` опущены, создается квадратная доска с восемью клетками по горизонтали и вертикали.

Задание 6. Выполнить аффинные преобразования; в качестве тестового изображения взять шахматную доску. Для этого необходимо выполнить следующее.

1. Создать M-функцию `affintr`:

```
function affintr(I,T,type)
tform=maketform('affine',T);
I1 = imtransform(I, tform) ;
figure, imshow(I1)
title(type)
```

2. Создать тестовое изображение:

```
>> I=checkerboard(40);
>> figure, imshow(I)
```

3. Выполнить преобразование ‘Растяжение’:

```
>> T=[3 0 0;0 2 0;0 0 1]; type='resize';
>> affintr(I,T,type);
```

4. Выполнить преобразование ‘Сдвиг’:

```
>> T=[1 0 0;0 .2 0;0 0 1]; type='Sdvig';
```

```
>> affintr(I,T,type);
```

5. Выполнить преобразование ‘Поворот’:

```
>> T = [cos(pi/4) sin(pi/4) 0;-sin(pi/4) cos(pi/4) 0; 0 0 1]; type='Rotate';
```

```
>> affintr(I,T,type);
```

6. Выполнить преобразование комбинацией растяжения, поворота и сдвига:

```
>> Tscale = [1.5 0 0; 0 2 0; 0 0 1]; % растяжение
```

```
>> Trot = [cos(pi) sin(pi) 0;-sin(pi) cos(pi) 0; 0 0 1]; % поворот
```

```
>> Tshear = [1 0 0; .2 1 0; 0 0 1]; % сдвиг
```

```
>> T1 = Tscale*Trot*Tshear;
```

```
>> tform=maketform('affine',T1); type='All';
```

```
>> affintr(I,T1,type);
```

7. Выполнить преобразование ‘Перенос’:

```
>> T = [1 0 0; 0 1 0; 50 50 1];
```

```
>> tform=maketform('affine',T);
```

```
>> I1 = imtransform(I, tform, 'XData', [1 320], 'YData', [1 320], 'FillValue', 0.5);
```

```
>> figure, imshow(I1)
```

Операции над изображениями на основе индексирования массивов

Изменить размеры изображения можно, непосредственно задавая необходимую индексацию массива изображения:

- Вырезание фрагмента – $I_c = I(Y:Y_m, X:X_n)$;
- Зеркальное отражение изображения по вертикали – $I_y = I(\text{end}:-1:1, :)$;
- Зеркальное отражение изображения по горизонтали – $I_y = I(:, \text{end}:-1:1)$;
- “прореживание” изображения – $I_d = I(1:2:\text{end}, 1:2:\text{end})$;

Задание 7. Выполнить зеркальное отражение изображения по горизонтали изображения файла Bigbird.bmp и сделать “прореживание” отраженного изображения по горизонтали.

```
>> [x,map]=imread('c:\image\bigbird.bmp');
```

```
>> I=im2double(ind2gray(x,map));
```

```
>> figure, imshow(I)
```

```
>> Iy = I(:,end:-1:1);
```

```
>> figure, imshow(Iy)
```

```
>> Id = Iy(:, 1:2:end);
```

```
>> figure, imshow(Id)
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Выполнить преобразования заданного изображения из табл. 3.3. Варианты преобразований приведены в табл. 3.4. Варианты заданий – в табл. 3.5 (для преобразования изображения в последующем шаге берется изображение из предыдущего шага). В табл. 3.5 в круглых скобках рядом с основным номером варианта представлена дополнительная информация по заданию табл. 3.4 (номер изображения из табл. 3.3, значение угла поворота и т. п.).

Таблица 3.3 Файлы изображений

Номер изображения	Наименование файла изображения
1	bike.bmp
2	Blaise.bmp
3	Clouds.bmp
4	Handshak.bmp
5	Technlgy.bmp
6	Saturn.bmp
7	Construc.bmp
8	Bigbird.bmp

Таблица 3.4 Варианты преобразований

Номер варианта	Преобразования изображений
1	Получить информацию из файла об изображении (номер изображения)
2	Прочитать изображение из файла (номер изображения)
3	Вырезать квадратный фрагмент D с помощью мыши
4	Вырезать фрагмент, задавая в команде положение координаты его верхнего левого угла как целую часть от 1/3 ширины и высоты изображения соответственно, значения ширины и высоты фрагмента определяются также
5	Вырезать фрагмент с помощью мыши из текущего окна без задания для него переменной и из полученного окна вырезать квадратный фрагмент с помощью мыши в переменную S
6	Увеличить фрагмент в (N) раз, используя метод: a – 'nearest'; b – 'bilinear'; d – 'bicubic'
7	Увеличить изображение в (N) раз, используя метод: a – 'nearest'; b – 'bilinear'; d – 'bicubic'
8	Повернуть изображение на заданный угол по часовой стрелке (угол), используя метод: a – 'nearest'; b – 'bilinear'; d – 'bicubic'
9	Повернуть изображение на заданный угол против часовой стрелки (угол), используя метод: a – 'nearest'; b – 'bilinear'; d – 'bicubic'
10	Уменьшить изображение в (N) раз, используя метод: a – 'nearest', b – 'bilinear', d – 'bicubic'

Номер варианта	Преобразования изображений
11	Растянуть изображение по горизонтали и по вертикали в (S_x, S_y) раз
12	Скомбинировать сдвиг изображения по горизонтали и по вертикали на (a, b)
13	Перенести изображение по горизонтали и по вертикали на (S_x, S_y)
14	Вывести на экран исходное изображение
15	Вывести на экран результаты преобразования в разные окна
16	Вывести на экран результаты преобразования в одно окно

Таблица 3.5 Варианты самостоятельных заданий

Номер задания	Номера вариантов из табл. 3.4
1	1(4); 2(4); 4, 6(3)a; 8(45)a; 14; 15
2	1(1); 2(1); 7(4)a; 5, 6(4)b; 9(90)a; 14; 16
3	1(3); 2(3); 10(2)a; 8(90)b; 3; 6(3)d; 14; 15
4	2(7); 11(2,4); 14; 16
5	1(2); 2(2); 9(45)a; 9(45)b; 9(45)d; 14; 15
6	1(6); 2(6); 6(3)b; 4; 6(3)b; 9(180); 14; 15
7	1(5); 2(5); 5; 8(60)a; 8(60)b; 8(60)d; 14; 16
8	2(7); 13(20,40); 14; 16
9	2(8); 12(0.6, -0.6); 14, 16

3. ВОПРОСЫ

1. Какими способами можно задать вырезание фрагмента для функции `imcrop`?
2. Что является входными аргументами для функции изменения размеров?
3. Что является входными аргументами для функции поворота?
4. Как задать аффинную матрицу:
5. а) для растяжения; б) для сдвига; в) для поворота; г) для переноса?
6. Как можно выполнить зеркальное отображение изображения?
7. Как можно выполнить «прореживание» изображения?

ЛАБОРАТОРНАЯ РАБОТА № 4. АМПЛИТУДНЫЕ ПРЕОБРАЗОВАНИЯ ИЗОБРАЖЕНИЯ

Цель работы – изучение функций амплитудных преобразований и приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Амплитудные преобразования изображений

Амплитудные преобразования относят к точечным процессам. Точечные процессы – это алгоритмы, которые изменяют значения элементов в изображении. Алгоритмы точечных процессов сканируют изображение элемент за элементом, осуществляя преобразование элементов изображения.

Контраст изображения

Важной характеристикой изображения является интенсивность. Существует верхняя граница интенсивности J_{\max} , определяющая величину допустимой интенсивности реального изображения и нижняя граница интенсивности, которая может быть больше нуля $J_{\min} > 0$. В этом случае разность $D_j = J_{\max} - J_{\min}$ определяет диапазон значений интенсивности. Для характеристики относительного изменения полутонов в соседних точках используют величину контраста. Пусть интенсивность изображения в точке (X_1, Y_1) составляет $J_1 = J(X_1, Y_1)$, а в точке (X_2, Y_2) равна $J_2 = J(X_2, Y_2)$. Найдем $J'_{\max} = \max(J_1, J_2)$ и $J'_{\min} = \min(J_1, J_2)$. Тогда за контраст принимается следующая величина:

$$K = \frac{J'_{\max} - J'_{\min}}{J'_{\max}}.$$

Повышение контраста производит отображение входного диапазона яркостей изображения в выходной диапазон яркостей. Наиболее простым и распространенным способом является отображение входного диапазона яркостей в максимально допустимый диапазон яркостей с помощью линейной функции (рис. 4.1).

Такая операция дает большую разность в яркостях соседних пикселей, что облегчает выделение перепадов яркостей и, как правило, более комфортное зрительное восприятие.

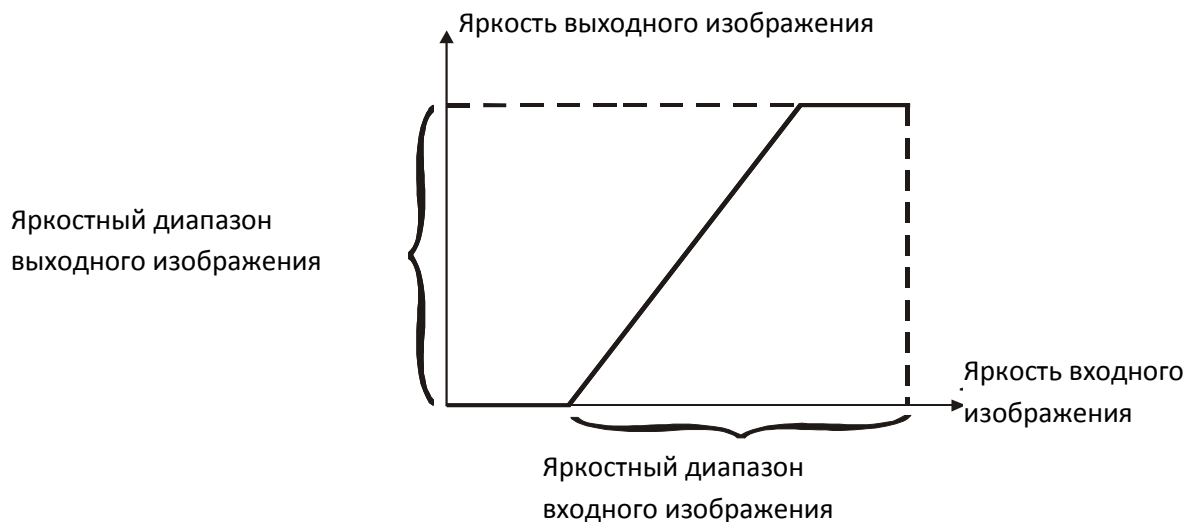


Рис. 4.1

Для получения негатива полутонового изображения I необходимо найти значение максимального уровня серого MaxI и получить негатив NI по формуле: $NI = \text{MaxI} - I$.

Задание 1. Получить негатив полутонового изображения файла Construc.bmp

```
>> [D,map] = imread('c:\ Image\ Construc.bmp');
>> I = ind2gray(D,map);
>> figure,imshow(I);
>> MaxI = max(I(:));
>> NI = MaxI-I;
>> figure,subplot(1,2,1),subimage(I);
>> subplot(1,2,2),subimage(NI);
```

Функции, используемые для изменения контраста изображения

Функция контрастирования изображения с гамма-коррекцией
imadjust

Синтаксис

$I_D = \text{imadjust}(I_S, [\text{low}, \text{high}], [\text{bottom top}], \text{gamma})$

Функция $I_D = \text{imajust}(I_S, [\text{low}, \text{high}], [\text{bottom top}], \text{gamma})$ создает полутоновое изображение I_D путем контрастирования исходного полутонового изображения I_S . Значения яркости в диапазоне $[\text{low}, \text{high}]$ преобразуются в значения яркости в диапазоне $[\text{bottom top}]$. Значения яркости, меньшие low , принимают значения bottom , а значения яркости, большие high , принимают значения top . Значения top , bottom , low , high должны лежать в диапазоне $[0,1]$. Если в качестве второго или третьего параметров передать пустой вектор $[]$, то по умолчанию будет использоваться вектор $[0,1]$.

Параметр gamma определяет форму кривой характеристики передачи уровней яркости. Если gamma меньше 1, то характеристика передачи уровней будет выпуклой и результирующее изображение будет светлее, чем исходное. Если gamma больше 1, то характеристика передачи уровней будет вогнутой и результирующее изображение будет темнее, чем исходное. По умолчанию gamma равен 1, что соответствует линейной характеристике передачи уровней и

отсутствию гамма-коррекции. Характеристика передачи уровней для различных значений γ приведены на рис. 4.2.

Задание 2. Выполнить контрастирование изображения с разными коэффициентами гамма-коррекции.

```
>> [D,map] = imread('c:\Image\Technlgy.bmp');  
>> I = ind2gray(D,map);  
>> figure,imshow(I);  
>> I1 = imadjust(I,[0 0.9],[],1);  
>> I2 = imadjust(I,[0 0.9],[],0.5);  
>> I3 = imadjust(I,[0 0.9],[],2);  
>> figure,subplot(1,3,1),subimage(I1);  
>> subplot(1,3,2),subimage(I2);  
>> subplot(1,3,3),subimage(I3);
```

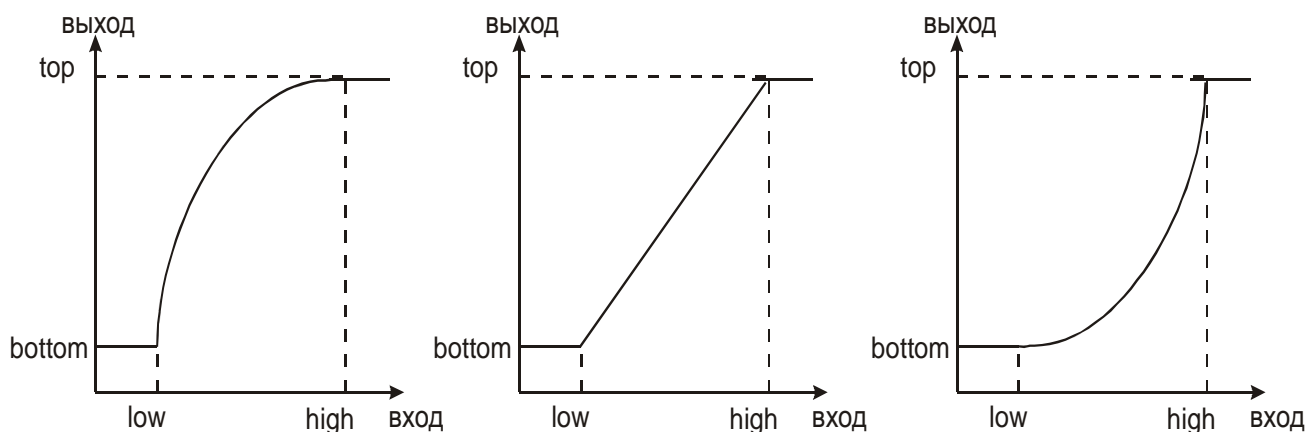


Рис. 4.2

Функция управления яркостью палитры **brighten**

Синтаксис

brighten(beta) – заменяет текущую палитру палитрового изображения;

newmap = brighten(beta) – возвращает палитру, полученную преобразованием текущей палитры изображения, без изменения текущей палитры на экране;

newmap = brighten(map,beta) – возвращает палитру, полученную преобразованием палитры **map**.

Во всех случаях палитра меняется на более яркую, если β принадлежит диапазону $[0, 1]$, и более темную, если β принадлежит диапазону $[-1, 0]$.

Задание 3. Изменить палитру полутонового изображения: а) на более светлую; б) на более темную и вывести изображение с исходной и полученными палитрами на экран в одно окно.

```
>> [D,map] = imread('c:\Image\earth.bmp');  
>> map1 = brighten(map,0.5);  
>> map2 = brighten(map,-0.8);
```

```
>> figure,subplot(1,3,1),subimage(D,map);
>> title('map')
>> subplot(1,3,2),subimage(D,map1);
>> title('map1')
>> subplot(1,3,3),subimage(D,map2);
>> title('map2')
```

Функция уменьшения количества цветов палитрового изображения **imapprox**

Синтаксис

```
[XD, newmap] = imapprox(XS, map, tol,differ_option)
```

```
[XD, newmap] = imapprox(XS, map, n,differ_option)
```

Функция `imapprox` создает новое палитровое изображение X_D из исходного X_S , уменьшая количество используемых цветов.

При использовании параметра `tol` устанавливается равномерная палитра `newmap` из цветов, которые в диапазоне $[0, 1]$ берутся с шагом `tol` по каждой из цветовых составляющих R, G, B; значение `tol` должно быть в диапазоне $[0, 1]$.

При использовании параметра `n` (`n` – число, большее 1), создается палитровое изображение X_D с палитрой `newmap` из `n` цветов. Действительное число цветов в палитре может оказаться меньше `n`, так как из палитры удаляются все цвета, которые отсутствуют в изображении.

Параметр `differ_option` позволяет применять или отказываться от диффузионного псевдосмещения цветов. Он может быть равен одной из следующих строковых констант:

'dither' – использовать диффузионное псевдосмещение цветов;

'nodither' – не использовать диффузионное псевдосмещение цветов.

По умолчанию применяется диффузионное псевдосмещение цветов. Оно создает впечатление, что на изображении присутствует большее количество различных цветов, чем это есть на самом деле. Этот визуальный эффект достигается за счет группирования в локальной области пикселов с цветами, представленными в палитре, смещение которых дало бы близкий к необходимому цвет.

Задание 4. Уменьшить количество цветов палитрового изображения до $n = 5$, вывести результат, преобразовать в полутоновые исходное и полученное изображения, вывести их на экран в одно окно.

```
>> [D,map] = imread('c:\Image\Blaise.bmp');
>> [D1,newmap] = imapprox(D,map,5);
>> figure,subplot(1,2,1),subimage(D,map)
>> subplot(1,2,2),subimage(D1,newmap)
>> I = ind2gray(D,map);
>> I1 = ind2gray(D1,newmap);
>> figure,subplot(1,2,1),subimage(I)
>> subplot(1,2,2),subimage(I1)
```

Задание 5. Уменьшить количество цветов палитрового изображения с использованием параметра $tol = 0.5$ и диффузионного псевдосмещения цветов и без него.

```
>> [D,map] = imread('c:\image\Factory.bmp');  
>> figure,imshow(D,map)  
>> [D1,nm] = imapprox(D, map, 0.5);  
>> figure,imshow(D1,nm)  
>> [D11,nm1] = imapprox(D, map, 0.5,'nodither');  
>> figure,imshow(D11,nm1)
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Выполнить преобразования заданного изображения из табл. 4.1. Варианты преобразований приведены в табл. 4.2. Варианты заданий приведены в табл. 4.3 (для преобразования изображения в последующем шаге берется изображение из предыдущего шага).

Таблица 4.1 Файлы изображений

Номер изображения	Наименование файла изображения
1	butterfly.bmp
2	Saturn.bmp
3	Clouds.bmp
4	Handshak.bmp
5	Bigbird.bmp
6	Technlgy.bmp

Таблица 4.2 Варианты преобразований

Номер варианта	Преобразования
1	Получить информацию из файла об изображении (номер изображения)
2	Прочитать изображение из файла (номер изображения)
3	Выполнить контрастирование, задав исходный [low high] и выходной [bottom top] интервал и коэффициент gamma
4	Преобразовать изображение(я) в полутоновое
5	Преобразовать изображение в палитровое
6	Получить негатив
7	Уменьшить количество цветов палитрового изображения, задав значение (tol): а) с диффузионным псевдосмещением цветов; б) без диффузионного псевдосмещения цветов
8	Уменьшить количество цветов палитрового изображения, задав значение количество цветов (n): а) с диффузионным псевдосмещением цветов; б) без диффузионного псевдосмещения цветов
9	Вывести в цикле в одно окно изображения с измененной палитрой, beta меняется в диапазоне [min max] с шагом (k)
10	Вывести на экран исходное изображение
11	Вывести на экран результаты преобразования в разные окна
12	Вывести на экран результаты преобразования в одно окно, при необходимости предварительно преобразовав их в формат double
13	Вывести на экран исходное изображение и результаты преобразования в одно окно, при необходимости предварительно преобразовав их в формат double

Таблица 4.3 Варианты самостоятельных заданий

Номер задания	Номера вариантов из табл. 4.2
1	1(1); 2(1); 10; 4; выполнить в цикле задание 3[0.3 0.7],[], когда гамма меняется от 0 до 2 с шагом 1; 12
2	2(2); 11; выполнить в цикле задание 8(от 2 до 6 с шагом 2)a,b; 12
3	1(3); 2(3); 9[−0.5 0.5] (0.5); 13
4	1(4); 2(4); выполнить в цикле задание 7(от 0.4 до 0.8 с шагом 0.2)a,b; 13
5	1(5), 2(5), 4; выполнить в цикле задание 3[0.2 0.7],[], когда гамма меняется от 0.5 до 1.5 с шагом 0.5; 6; 12
6	1(6); 2(6); 9[−1 1] (1); 6; 12

3. ВОПРОСЫ

1. В чем особенность амплитудных преобразований?
2. Что такое контраст изображения?
3. Какие входные аргументы передаются в функции по изменению контраста?
4. Какие вы знаете способы вызова функции `brighten`?
5. С помощью какой функции можно уменьшить количество цветов палитрового изображения? Какие входные аргументы используются для этого?

ЛАБОРАТОРНАЯ РАБОТА № 5. ФУНКЦИИ, ИСПОЛЬЗУЕМЫЕ ДЛЯ АНАЛИЗА ИЗОБРАЖЕНИЙ

Цель работы – изучение функций анализа изображений и приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Функция построения гистограммы распределения яркости изображения

Гистограмма распределения яркости изображения

Гистограмма цифрового изображения – это дискретная функция, описывающая частоту появления (вероятность) уровня серого в изображении, представленная в виде графика.

По оси абсцисс откладываются номера градаций уровней серого по возрастанию (значения интенсивности), а по оси ординат – количество пикселей, имеющих данный уровень серого (частоту появления данной интенсивности).

Гистограмма может свидетельствовать об общей яркости и контрасте изображений, поэтому являются ценным методом как количественной, так и качественной обработки изображения.

Функция построения гистограммы `imhist`

Синтаксис

`imhist(I,n)` – выводит в текущее окно гистограмму яркостей пикселей полутонового изображения. Гистограмма состоит из n столбцов. По умолчанию $n = 256$ для полутонового изображения и $n = 2$ для бинарного изображения. Под рисунком гистограммы выводится шкала яркостей;

`imhist(X,map)` – в текущем окне строит гистограмму индексов пикселей палитрового изображения X . Под рисунком гистограммы выводится палитра map ;

`[h, cx] = imhist(I,n)`

`[h,cx] = imhist(X,map)` – функции `[h, cx] = imhist(...)` вычисляют вектор гистограммы h и вектор положения центров столбцов гистограммы cx на оси яркостей (для полутоновых и бинарных изображений) или на оси индексов (для палитровых изображений), что позволяет производить дальнейшую обработку гистограммы h .

Задание 1. Построить гистограмму палитрового изображения.

```
>> [S,map] = imread('c:\Image\chip.bmp');  
>> figure,imshow(S,map);  
>> imhist(S, map)
```

Задание 2. Построить гистограмму полутонового изображения двумя способами.

```
>> [S,map] = imread('c:\Image\chip.bmp');  
>> I = ind2gray (S,map);  
>> figure,imshow(I);
```

```
>> figure, imhist(I);
>> [h,cx] = imhist(I);
>> figure,stem(cx,h)
```

Функция по выравниванию (эквализации) изображения histeq

Гистограмма распределения яркости типичного изображения имеет ярко выраженный перекосяк в сторону малых уровней. Яркость большинства элементов ниже среднего, на темных участках детали часто оказываются неразличимыми. Одним из методов улучшения визуального восприятия изображения является изменение гистограммы. Оно направлено на выравнивание гистограммы яркости и называется эквализацией или выравниванием гистограммы.

Синтаксис

ID = histeq(Is,n)

ID= histeq(Is, hgram)

Функция ID = histeq(Is,n) преобразует исходное полутоновое изображение Is так, чтобы результирующее полутоновое изображение ID имело гистограмму яркости пикселей, близкую к равномерной. Чем меньше n по сравнению с количеством градаций яркости в изображении Is, тем более равномерной получается гистограмма яркостей пикселей результирующего изображения ID. По умолчанию значение n равно 64, и данный параметр можно не указывать при вызове функции.

Функция ID = histeq(Is,hgram) улучшает контраст изображения с помощью преобразования значения пикселей исходного изображения так, чтобы гистограмма яркостей пикселей результирующего изображения приблизительно соответствовала некоторой заданной гистограмме hgram.

Метод построения обработанного изображения с заданной гистограммой называется гистограммной подгонкой, или гистограммной спецификацией.

Задание 3. Смоделировать бимодальную гистограмму с помощью нормированных мультимодальных гауссовых функций с помощью приведенной ниже M-функции twomgauss и вывести на экран:

```
function p= twomgauss(ml, sigl, m2, sig2, A1, A2, k)
% входные параметры: ml, sigl, m2, sig2, – мат. ожидания и среднекв. отклоне-
% ния для двухмодальной гистограммы; A1, A2 – амплитуды, k – коэффициент
% рекомендуемые значения входных параметров
% ( ml, sigl, m2, sig2, A1, A2, k):
% (0.15, 0.05, 0.75, 0.05, 1, 0.07, 0.002)
c1=A1 * (1 /(((2 * pi) ^0.5) * sigl));
k1 = 2 * (sigl ^2);
c2 = A2 * (1/(((2*pi)^0.5) * sig2));
k2 = 2 * (sig2 ^2);
z = linspace(0, 1, 256);
p = k + c1 * exp(-((z-ml) .^2) ./ k1) + c2 * exp(-((z-m2) .^2) ./ k2);
p = p ./ sum(p(:));
>> p= twomgauss(0.15, 0.05, 0.75, 0.05, 1, 0.07, 0.002);
```

```
>> plot(p)
```

Задание 4. Повысить контраст изображения с помощью выравнивания гистограммы.

```
>> [D,map] = imread('c:\Image\mona.bmp');  
>> newmap = histeq(D,map)  
>> figure,imshow(D,map)  
>> figure,imshow(D,newmap)
```

Функции, используемые при корреляционном анализе

Корреляционный анализ позволяет получить на практике представление о некоторых свойствах изображения, например, о скорости изменения интенсивности по координатам, о протяженности однородных участков без разложения их на гармонические составляющие. Смысл корреляционного анализа состоит в количественном измерении степени сходства различных сигналов. Для этого служат корреляционные функции, рассмотренные ниже.

Функция вычисления среднего значения элементов матрицы mean2

Синтаксис

`m = mean2(S)`

Функция `m = mean2(S)` вычисляет среднее значение элементов матрицы `S`. Данная функция эквивалентна функции `mean(S(:))`.

Функция вычисления среднеквадратического отклонения элементов матрицы std2

Функция `std2` вычисляет среднеквадратическое отклонение.

Синтаксис

`d = std2(S)` – вычисляет среднеквадратическое отклонения элементов матрицы `S`. Данная функция эквивалентна функции `std(S(:))`.

Функция вычисления коэффициента корреляции между двумя матрицами corr2

Синтаксис

`k = corr2(A,B)` – вычисляет коэффициент корреляции `k` между двумя матрицами `A,B`. Размеры матриц `A` и `B` должны совпадать.

Коэффициент корреляции вычисляется с помощью следующего соотношения:

$$k = \frac{\sum_c \sum_r (A(c,r) - A_m) \cdot (B(c,r) - B_m)}{\sqrt{(\sum_c \sum_r (A(c,r) - A_m)^2 \cdot (B(c,r) - B_m)^2)}},$$

где $A_m = \text{mean2}(A)$; $B_m = \text{mean2}(B)$ – среднее значение матрицы.

Функция вычисления двумерной взаимной корреляционной функции xcorr2

Синтаксис

`c = xcorr2(a,b)` – вычисляет двумерную взаимную корреляционную функцию между изображениями `a` и `b`.

`c = xcorr2(a)` – вычисляет автокорреляционную функцию и эквивалентна функции `c = xcorr2(a,a)`.

Задание 5. Получить автокорреляционную функцию для изображения файла bike.bmp и двумерную взаимную корреляционную функцию между исходным изображением и его негативом. Графики полученных функций вывести на экран.

```
D=imread('c:\Image\bike.bmp');  
I=im2double(rgb2gray(D));  
c1=xcorr2(I,I);  
MaI=max(I(:));  
NI=MaI-I;  
c2=xcorr2(I,NI);  
figure,mesh(c1)  
title('autocorr. function')  
figure,mesh(c2)  
title('corr. function')
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Выполнить следующие действия:

1. Получить полутоновое изображения из полноцветного, которое хранится в файле `wall.bmp`. Эквиализировать его. Вывести полутоновое и эквализированное изображения и их гистограммы в одно окно.
2. Получить из полноцветного изображения из файла `butterfly.bmp` палитровое. Уменьшить количество цветов палитрового изображения до 256. Эквиализировать его. Вывести палитровое и эквализированное изображения и их гистограммы в одно окно.
3. Найти двумерную взаимную корреляционную функцию между исходным и повернутым на 180 градусов изображением файла `bigbird.bmp`.

3. ВОПРОСЫ

1. Что такое гистограмма?
2. Какая функция используется для получения гистограммы?
3. В чем отличие гистограммы полутонового изображения от гистограммы палитрового изображения?
4. Что такое эквализация изображения? Какая функция выполняет эквализацию? Ее способы вызова.
5. В чем смысл корреляционного анализа сигналов?
6. Какие функции корреляционного анализа вы знаете?

ЛАБОРАТОРНАЯ РАБОТА № 6. ФУНКЦИИ ФИЛЬТРАЦИИ ИЗОБРАЖЕНИЯ

Цель работы – изучение функций зашумления и фильтрации изображения, приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Функция добавления шума `imnoise`

Синтаксис

`ID = imnoise(Is, type, params)`

Функция `imnoise` создает новое изображение `ID` путем добавления шума к исходному полутоновому изображению `Is`. Функция может добавлять шум трех типов, которые задаются строковой константой `type`. Количество и смысл параметров `params` определяются выбранным типом шума. Если параметры не указаны, то будут использованы значения по умолчанию:

1) `ID = imnoise(Is, 'gaussian', m,v)` добавляет к изображению `Is` гауссовый белый шум с математическим ожиданием `m` и дисперсией `v`. По умолчанию `m = 0`, `v = 0.01`;

2) `ID = imnoise(Is, 'salt & pepper', d)` добавляет к изображению `Is` импульсный (точечный) шум. Параметр `d` определяет плотность шума и равен доле искаженных пикселей. По умолчанию `d = 0.05`;

3) `ID = imnoise(Is, 'speckle',v)` добавляет к изображению `Is` мультипликативный шум $ID = Is + n * Is$, где `n` – равномерно распределенный случайный шум с математическим ожиданием ноль и дисперсией `v`. По умолчанию `v = 0.04`.

Задание 1. Зашумить изображение гауссовым, импульсным и мультипликативным шумом.

```
>> [S,map] = imread('c:\Image\Mona.bmp');  
>> I = ind2gray(S,map);  
>> S1 = imnoise(I,'gaussian',0,0.1);  
>> S2 = imnoise(I,'salt & pepper');  
>> S3 = imnoise(I,'speckle');  
>> figure,subplot(1,3,1),subimage(S1);  
>> subplot(1,3,2),subimage(S2);  
>> subplot(1,3,3),subimage(S3)
```

Фильтрация изображения

Для выполнения линейной пространственной фильтрации используют метод двумерной пространственной свертки локальной окрестности обрабатываемого элемента с линейным оператором, которая называется маской или матрицей коэффициентов фильтра.

Алгоритм свертки заключается в том, что маска сканирует исходное изображение. На каждом шаге находится сумма произведений элементов маски и соответствующих элементов исходного изображения, и найденное значение присваивается одному элементу результирующего изображения. Достигнув

таким образом конца строки, маска смещается на одну строку вниз, в начало строки, и процесс повторяется.

Имеется две тесно связанные концепции, которые необходимо понимать при совершении линейной пространственной фильтрации. Первая – это корреляция, а вторая – свертка. Корреляция состоит в прохождении маски по изображению. С точки зрения механики процесса, свертка делается так же, но маску надо повернуть на 180° перед прохождением по изображению. Если сдвигаемая маска является симметричной, то корреляция и свертка дают одинаковые результаты.

Ниже приводятся функции, используемые для фильтрации изображения.

Функция вычисления двумерной свертки conv2

Синтаксис

$D = \text{conv2}(S, h, \text{shape})$

Функция выполняет двумерную пространственную свертку изображения S с маской h . Параметр shape , определяющий размер результирующего изображения D , может принимать следующие значения:

- 1) 'full' – полноразмерная свертка (по умолчанию);
- 2) 'same' – центральная часть размера изображения S ;
- 3) 'valid' – центральная часть размера изображения S с вычетом размера маски.

Функция вычисления двумерной линейной фильтрации filter2

Синтаксис

$D = \text{filter2}(h, S, \text{shape})$

Результат вычисляется как корреляция массива S двумерным фильтром, коэффициенты которого сведены в матрицу h . Как правило, S и D являются полутоновыми изображениями. Значения параметра shape такие же, как у функции conv2 . По умолчанию $\text{shape} = \text{'same'}$. С точки зрения выполнения процесса, свертка делается так же, но маску h надо повернуть на 180° перед прохождением по изображению S . Для этого можно использовать функцию rot90 .

Задание 2. Выполнить фильтрацию палитрового изображения, используя маску

$$h = \begin{bmatrix} 1 & 1 & 1 \\ 1 & -2 & 1 \\ -1 & -1 & -1 \end{bmatrix}.$$

```
>> [x,map] = imread('c:\image\mona.bmp');
>> I = ind2gray(x,map);
>> I = im2double(I);
>> h = [1 1 1; 1 -2 1; -1 -1 -1];
>> h1 = rot90(h,2);
>> I1 = filter2(h1,I);
>> figure,imshow(I)
>> figure,imshow(I1)
```

Функция вычисления медианной фильтрации **medfilt2**

Синтаксис

$I_D = \text{medfilt2}(I_S, [m, n], \text{padopt})$

Функция выполняет нелинейную фильтрацию, механизм выполнения которой аналогичен линейной фильтрации, но на каждом шаге сканирования маски размера $m \times n$ (по умолчанию 3×3) выполняется следующая нелинейная операция: пиксели изображения, находящиеся под маской, сортируются и составляют упорядоченную последовательность A . Пикселу результирующего изображения $I_D(r, c)$, где r и c – координаты текущего положения центрального элемента маски, присваивается значение медианы последовательности A .

Параметр **padopt** определяет три возможные опции расширения границ изображения: опция по умолчанию 'zeros' с нулевым расширением, 'symmetric', при которой изображение I_S расширяется путем его зеркального отражения через границы, и 'indexed', при которой I_S расширяется значением 1, если I_S имеет класс double, и значением 0 в противном случае.

Задание 3. Отфильтровать изображение файла *Athena.bmp*, зашумленного импульсным шумом.

```
>> [S,map] = imread('c:\Image\Mona.bmp');  
>> I = ind2gray(S,map);  
>> S = imnoise(I,'salt & pepper');  
>> figure,imshow(S,[]);  
>> D = medfilt2(S);  
>> figure,imshow(D,[]);
```

Функция вычисления двумерной АЧХ **freqz2**

Синтаксис

freqz2(h,n1,n2) – выводит на экран двумерную АЧХ.

[H,f1,f2] = freqz2(h,n1,n2) – формирует матрицу H размера $n1 \times n2$, которая является АЧХ на частотах, содержащихся в векторах $f1$ и $f2$, по двумерному фильтру, коэффициенты которого сведены в матрицу h .

Задание 4. Вывод АЧХ высокочастотного фильтра двумя способами.

```
>> h = [-1,-1,-1;-1,9,-1;-1,-1,-1];  
>> figure,freqz2(h);  
>> title('1 variant')  
>> [H,f1,f2] = freqz2(h);  
>> figure,mesh(f1,f2,abs(H))  
>> title('2 variant')
```

Функция задания маски предопределенного фильтра **fspecial**

Синтаксис

$h = \text{fspecial}(\text{type}, P1, P2)$

Функция возвращает маску h предопределенного двумерного линейного фильтра, задаваемого строкой **type**. В зависимости от типа фильтра, для него

могут быть определены один или два дополнительных параметра P1, P2. Может быть задан размер маски n (если n – вектор, то размер маски n(1) × n(2), если n – скаляр, то размер маски – n × n), sigma – среднеквадратичное отклонение распределения Гаусса, которое используется при формировании маски h; параметр a, управляющий соотношением между центральным и граничными элементами маски, устанавливается в диапазоне [0, 1], по умолчанию равен 0.2.

Возможные варианты функции fspecial:

1) h = fspecial ('gaussian', n,sigma) возвращает маску h фильтра нижних частот Гаусса. По умолчанию n равно 3 × 3, а sigma равно 0.5;

2) h = fspecial ('sobel') возвращает маску фильтра Собела для выделения горизонтальных границ, для выделения вертикальных границ достаточно транспонировать данную маску h;

3) h = fspecial ('prewitt') возвращает маску фильтра Превитта для выделения горизонтальных границ, для выделения вертикальных границ достаточно транспонировать данную маску h;

4) h = fspecial ('laplacian', a) возвращает маску h ВЧ фильтр Лапласа. Размер маски 3 × 3. С помощью фильтра Лапласа можно выполнять улучшение изображения, используя формулу $g(x, y) = f(x, y) + c\Delta^2 f(x, y)$, где $f(x, y)$ – это исходное изображение; $g(x, y)$ – улучшенное изображение, а параметр c равен 1, если центральный коэффициент маски положителен, и c = -1 в противном случае; $\Delta^2 f(x, y) = d^2 f(x, y) / dx^2 + d^2 f(x, y) / dy^2$ – отфильтрованное изображение фильтром Лапласа;

5) h = fspecial ('log',n,sigma) возвращает маску h фильтра, аналогичного последовательному применению фильтров Гаусса и Лапласа, так называемого лапласиана – гауссиана, n и sigma по умолчанию устанавливается равным 5 × 5 и 0.5 соответственно;

6) h = fspecial ('average', n) возвращает маску h усредняющего НЧ фильтра. По умолчанию размер маски n устанавливается равным 3 × 3;

7) h = fspecial ('unsharp', a) возвращает маску h фильтра, повышающего резкость изображения. Размер маски: 3 × 3.

Задание 5. Улучшить изображение, используя фильтр высоких частот Лапласа.

1. Создать маску h фильтра высоких частот Лапласа:

```
>> h = fspecial('laplacian',0);
```

2. Выполнить высокочастотную фильтрацию изображения:

```
>> [S,map] = imread('c:\Image\Mona.bmp');
```

```
>> I = im2double(ind2gray(S,map));
```

```
>> figure,imshow(I)
```

```
>> I1 = filter2(h,I);
```

```
>> figure,imshow(I1)
```

```
>> I2 = I - I1;
```

```
>> figure,imshow(I2)
```

Задание 6. Отфильтровать изображение фильтром Собеля.

1. Создать маску фильтра Собеля для выделения горизонтальных границ
>> h = fspecial('sobel');

2. Создать маску фильтра Собеля для выделения вертикальных границ
>> h1 = h';

3. Отфильтровать изображение earth.bmp фильтрами Собеля.

```
>> [X,map] = imread('c:\Image\earth.bmp');
```

```
>> I = ind2gray(X,map);
```

```
>> figure,imshow(I)
```

```
>> F1 = filter2(h,I);
```

```
>> figure,imshow(F1)
```

```
>> F2 = filter2(h1,I);
```

```
>> figure,imshow(F2)
```

```
>> F3=F1+F2;
```

```
>> figure,imshow(F3)
```

Функция формирования маски линейного фильтра по желаемой АЧХ
fsamp2

Синтаксис

```
h = fsamp2(f1,f2,H)
```

Функция формирует маску h линейного двумерного фильтра, основываясь на желаемой АЧХ двумерного фильтра H для частот, передаваемых в векторах f1 и f2.

Задание 7. Повысить резкость изображения с помощью линейного фильтра, сформированного по желаемой АЧХ.

1. Получить нормализованные значения частот:

```
>>[f1,f2] = freqspace(15,'meshgrid');
```

2. Вычислить желаемую АЧХ - H, используя так называемую метрику городских кварталов:

```
>> dist = abs(f1)+abs(f2);
```

```
>> H = dist/max(dist(:));
```

3. Вывести на экран желаемую АЧХ:

```
>> figure,mesh(f1,f2,H),colormap(cool(32));
```

4. Сформировать маску фильтра по желаемой АЧХ:

```
>> h = fsamp2(f1,f2,H,[5 5]);
```

5. Вывести на экран получившуюся АЧХ:

```
>> figure,colormap(cool(32)),freqz2(h);
```

6. Прочитать изображение и вывести его на экран:

```
>> [i,map] = imread('c:\Image\athena.bmp');
```

```
>> figure,imshow(i,map);
```

7. Отфильтровать изображение:

```
>> i1 = mat2gray(filter2(h,im2double(i)));
```

```
>> figure,imshow(i1);
```

8. Контрастирование результата фильтрации:

```
>> i2 = imadjust(i1,[0 0.5],[]);
```

```
>> figure,imshow(i2);
```

2.7. Функция формирования маски линейного фильтра методом преобразования частот ftrans2

Синтаксис

```
h = ftrans2(b)
```

Функция формирует маску h линейного двумерного фильтра, используя метод преобразования частот для трансформации одномерного фильтра с коэффициентами b . Для преобразования частот используется специальная матрица трансформации t . По умолчанию применяется матрица Мак–Клеллана:

$$t = \frac{1}{8} \cdot \begin{bmatrix} 1 & 2 & 1 \\ 2 & -4 & 2 \\ 1 & 2 & 1 \end{bmatrix}.$$

В функции $h = \text{ftrans2}(b,t)$ можно указать иную матрицу трансформации.

Задание 8. Выполнить размытие изображения с целью подавления муара с помощью линейного двумерного фильтра низкой частоты (ФНЧ), сформированного из одномерного ФНЧ методом преобразования частот:

1. Создать одномерный ФНЧ 14-го порядка с частотой среза 0.2:

```
>> b = fir1(14,0.2); % проектирование фильтров методом взвешивания
```

```
>> freqz(b,1,256)
```

2. Сформировать двумерный фильтр из одномерного:

```
>> h = ftrans2(b);
```

```
>> figure,freqz2(h)
```

3. Прочитать исходное изображение и отфильтровать полученным фильтром:

```
>> [S,map] = imread('c:\Image\Athena.bmp');
```

```
>> I = ind2gray(S,map);
```

```
>> I = im2double(I);
```

```
>> figure,imshow(I)
```

```
>> I1 = filter2(h,I);
```

```
>> figure,imshow(I1)
```

```
>> I2 = mat2gray(I1);
```

```
>> figure,imshow(I2)
```

Функция вычисления общей нелинейной фильтрации nlfiter

Синтаксис

```
D = nlfiter(S,[m n],fun,P1,P2, ...)
```

Функция $D = \text{nlfiter}(S,[m\ n],\text{fun},P1,P2, \dots)$ используется для обработки бинарных и полутоновых изображений. Формирует новое изображение,

используя для фильтрации маску размером $[m \ n]$ и функцию, осуществляющую фильтрацию, fun. P1,P2 – возможные параметры, передаваемые в fun.

Задание 9. Создать функцию по фильтрации импульсного шума для использования в качестве аргумента fun в функции обобщенного нелинейного фильтра, который в этом случае будет использоваться для удаления импульсного шума в изображении. В функции фильтрация импульсного шума осуществляется операцией усреднения с порогом, которая состоит в том, что центральному пикселу в пределах маски присваивается среднее значение яркости всех пикселей в пределах маски в том случае, если разница между исходным значением центрального пикселя и средним больше заданного порогового значения. Для создания этой функции с именем 'AverageWithTh' в редакторе m-файлов необходимо выполнить следующие действия.

1. Создать новый m-файл командой главного меню:

File/New/m-file

2. Ввести команды функции:

```
function R = AverageWithTh(x,Th); % заголовочная строка
```

```
[r c] = size(x);
```

```
n = r*c;
```

```
r = floor((r+1)/2); c = floor((c+1)/2);
```

```
s = sum(x(:))/n;
```

```
if (abs(x(r,c)-s))>Th
```

```
    R = s;
```

```
else
```

```
    R = x(r,c);
```

```
end;
```

```
end;
```

3. Создать новый подкаталог.

4. Сохранить в новый подкаталог созданную функцию в файл с одноименным названием.

5. Прочитать изображение, зашумить импульсным шумом и отфильтровать его, используя созданную функцию:

```
>> [X,map] = imread('c:\Image\Mona.bmp');
```

```
>> I = im2double(ind2gray(X,map));
```

```
>> figure,imshow(I)
```

```
>> I1 = imnoise(I,'salt & pepper');
```

```
>> figure,imshow(I1)
```

```
>> I2 = nlfilter(I1,[3 3],'AverageWithTh',0.2);
```

```
>> figure,imshow(I2)
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

Выполнить следующие действия.

1. Прочитать палитровое изображение из файла Earth.bmp, вывести на экран, преобразовать в полутоновое, добавить импульсный шум, отфильтровать зашумленное изображение медианной фильтрацией и обобщенным нелинейным фильтром. Зашумленное и отфильтрованные изображения вывести в одном окне для сравнения.

2. Выполнить фильтрацию изображения из файла Bigbird.bmp:

а) Отфильтровать изображение с помощью масок кругового градиента:

север			северо-восток			восток		юго-восток			
1	1	1	1	1	1	-1	1	1	-1	-1	1
1	-2	1	-1	-2	1	-1	-2	1	-1	-2	1
-1	-1	-1	-1	-1	1	-1	1	1	1	1	1

юг			юго-запад			запад		северо-запад			
-1	-1	-1	1	-1	-1	1	1	-1	1	1	1
1	-2	1	1	-2	-1	1	-2	-1	1	-2	-1
1	1	1	1	1	1	1	1	-1	1	-1	-1

Ввод масок двумерных линейных фильтров, построение их АЧХ, фильтрацию изображения с их помощью и вывод результата выполнить в цикле;

б) Улучшить изображение с помощью масок лапласиановских фильтров:

0	1	0	-1	-1	-1	1	-2	1
1	-4	1	-1	8	-1	-2	4	-2
0	1	0	-1	-1	-1	1	-2	1

Ввод масок и вывод результатов выполнить в цикле.

3. Прочитать изображение из файла Technlgy.bmp, вывести на экран, преобразовать в полутоновое. Получить маску фильтра Превитт. Выполнить фильтрацию исходного полутонового фрагмента маской фильтра Превитт отдельно по горизонтали и по вертикали и вместе на одном изображении.

4. Прочитать цветное изображение из файла 'Bike.bmp', вывести на экран, преобразовать в полутоновое. Взять в качестве АЧХ фильтра функцию расстояния от начала координат, сформировать по АЧХ маску фильтра и отфильтровать полутоновое изображение. Исходное полутоновое изображение и результаты его обработки вывести на экран.

5. Вывести АЧХ всех фильтров, создаваемых функцией по заданию масок предопределенного фильтра, в одном окне с заголовками для каждого фильтра.

6. Зашумить в цикле полутоновое изображение (исходное взять из файла 'butterfly.bmp' гауссовым шумом с разными дисперсиями: а) по умолчанию (0.01); б) $\sigma = 0.5$ для математического ожидания, равного $m = 0.5$).

3. ВОПРОСЫ

1. Какие типы фильтров создает функция по формированию масок фильтров `fspecial`?
2. В чем заключается алгоритм двумерной свертки?
3. В каких функциях присутствует алгоритм двумерной свертки?
4. В чем отличие алгоритма медианной фильтрации от алгоритма фильтрации с помощью операции усреднения с порогом?
5. Какие типы шумов формирует функция по зашумлению изображений `imnoise`?
6. Для каких целей можно использовать функцию `freqz2`?
7. Каким образом можно сформировать маску линейного фильтра по желаемой АЧХ?
8. Какая функция позволяет сформировать двумерный фильтр из одномерного?

ЛАБОРАТОРНАЯ РАБОТА № 7. ВОССТАНОВЛЕНИЕ ИЗОБРАЖЕНИЯ

Цель работы – моделирование искажений, вносимых линейной системой при формировании изображения и восстановление изображения.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Одной из задач обработки изображений является задача устранения искажений, возникающих в процессе их формирования. В процессе записи изображения искажаются также шумами, присутствующими в любом реальном физическом устройстве. На рис. 1 приведена линейная модель формирования изображения.

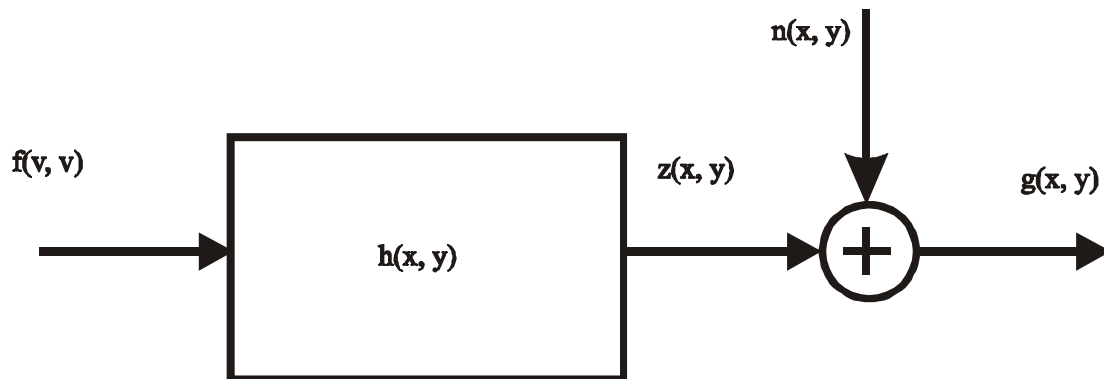


Рис. 7.1

Значение функции яркости $f(v,v)$ исходного изображения в точке с координатами (v,v) «размазывается» в соответствии с импульсным откликом $h(x,y)$: $z(x,y) = f(v,v) \otimes h(x,y)$ и искажается аддитивным шумом $n(x,y)$, таким образом, выходное изображение $g(x, y)$ представляет собой свертку входного изображения с импульсным откликом системы, искаженную аддитивным шумом $n(x, y)$.

Фильтр Винера

Задача построения восстанавливающей системы состоит в поиске импульсного отклика $r(x, y)$, минимизирующего средний квадрат ошибки J :

$$J = [f(x, y) - g(x, y) \cdot r(x, y)]^2.$$

Исходя из минимизации среднего квадрата ошибки J и рассматривая исходное изображение $f(x, y)$ и результаты его преобразований как действительные стационарные случайные сигналы, импульсный отклик восстанавливающей системы в частотной области определяется из следующего выражения:

$$W_g(u, v)R(u, v) = W_{gf}(u, v),$$

где $W_g(u, v)$ – спектр мощности случайного процесса; $R(u, v)$ – частотная функция восстанавливающей системы; $W_{gf}(u, v)$ – кросс-спектр мощности процессов g и f .

Тогда частотная функция восстанавливающей системы определяется как

$$R(u, v) = \frac{W_{gf}(u, v)}{W_g(u, v)}.$$

После учета шумов частотная функция принимает вид:

$$R(u, v) = \frac{H^*(u, v)}{|H(u, v)|^2 + \frac{W_n(u, v)}{W_f(u, v)}},$$

где $H(u, v)$ – частотная функция линейной искажающей системы; $(\dots)^*$ – символ комплексного сопряжения; $W_f(u, v)$, $W_n(u, v)$ – спектры мощности исходного изображения и шума соответственно.

Система с такой частотной функцией известна как фильтр Винера. Такая же система с соотношением спектров мощности исходного изображения и шума (шум/сигнал), равным нулю, называется инверсным фильтром.

При съемке космических объектов через атмосферу Земли происходит “размытие” изображения, обусловленное турбулентностью атмосферы. Этот эффект приближенно можно представить действием линейной системы с импульсным откликом вида

$$h(x, y) = \exp\left(-\frac{x^2 + y^2}{2\rho^2}\right),$$

которому соответствует частотная характеристика

$$H(u, v) = \exp[-2\rho^2(u^2 + v^2)],$$

где ρ – коэффициент нерезкости.

Задание 1. Смоделировать «размытие» изображения, обусловленное турбулентностью атмосферы, и восстановить его фильтром Винера:

```
[X,map] = imread('c:\Image\Athena.bmp');
I = im2double(ind2gray(X,map));
figure,imshow(I)
title('original image')
[f11,f22] = freqspace([15 15],'meshgrid');
a = 1;
H = exp(-a.^2.*(f11.^2+f22.^2));
h = fsamp2(f11,f22,H,[5,5]);
Id = conv2(I,h,'same');
figure,imshow(Id)
title('defect image')
k = 1e-4;
[HT,f1,f2] = freqz2(h,[5 5]);
HV = conj(HT)./(abs(HT).^2+k);
hv = fsamp2(HV);
Ir = conv2(Id,hv,'same');
figure,imshow(Ir)
```

title('restore image')

В пакете IPT функция, выполняющая винеровскую фильтрацию, называется `deconvwnr`.

Синтаксис

`Ir = deconvwnr(Id, h, k),`

где `Id` – искаженное изображение; `h` – импульсный отклик линейной искажающей системы; параметр `k` определяет отношение спектра мощности шума и исходного изображения. По умолчанию соотношение шум/сигнал `k = 0` (т. е. форма винеровского фильтра совпадает с инверсным фильтром).

Для выполнения восстановления изображения файла `Athena.bmp` из предыдущего задания форма вызова функции совпадает с приведенным синтаксисом:

`>> Ir = deconvwnr(Id, h, k);`

Гомоморфный фильтр

Для восстановления изображения также можно использовать гомоморфный фильтр.

При проектировании фильтра отыскивается такая линейная оценка

$$\hat{f}(x, y) = L[g(x, y)]$$

(где `L` – линейный оператор), чтобы энергетический спектр оценки равнялся энергетическому спектру исходного изображения. Пространственно-частотная характеристика гомоморфного фильтра определяется следующим выражением:

$$H_p(u, v) = \left[\frac{1}{|H(u, v)|^2 + \frac{W_n(u, v)}{W_f(u, v)}} \right]^{1/2}.$$

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Восстановить «размытое» изображение файла construc.bmp, обусловленное турбулентностью атмосферы, гомоморфным фильтром.
2. Создать М-файл для демонстрации восстановления искажений изображения, обусловленных турбулентностью атмосферы и восстановить его фильтром Винера, инверсным или гомоморфным фильтром с возможностью задания файла изображения, коэффициента нерезкости и отношения сигнал/шум с клавиатуры.

3. ВОПРОСЫ

1. Что обуславливает искажения изображения при его формировании?
2. В чем заключается задача построения восстанавливающей системы?
3. Какие функции использовались для моделирования «размытия» изображения и его восстановления в задании лабораторной работы?
4. Какие принципы лежат в основе построения фильтров Винера, гомоморфного фильтра?

ЛАБОРАТОРНАЯ РАБОТА № 8. БИНАРИЗАЦИЯ ИЗОБРАЖЕНИЙ

Цель работы – изучение средств по бинаризации изображения, приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Бинаризация изображений

Пусть $\{a_{i,j}\}$ – полутоновое изображение, t – порог и b_0 и b_1 – два бинарных значения. Результатом порогового разделения является бинарное изображение, полученное следующим образом:

$$\{a_{i,j}\} = \begin{cases} b_0, & a_{i,j} \leq t \\ b_1, & a_{i,j} > t \end{cases}$$

Основной задачей является выбор значения t с помощью некоторого критерия. Это значение может выбираться как одинаковым для всего изображения, так и различным для различных его частей. Если значения объекта и фона достаточно однородны по всему изображению, то может использоваться одно пороговое значение для всего изображения. Использование единственного порога для всех пикселей называется глобальным пороговым разделением. Бинаризацию полутонового изображения можно выполнить операцией сравнения значений уровней серого с заданным порогом (бинаризация по порогу). Если в качестве операции сравнения выбирается операция «больше выбранного уровня» ($I > \text{threshold}$), получается позитив, если «меньше» ($I < \text{threshold}$) – негатив.

Выбор порогового значения

Существует много способов выбора порогового значения, например, разделом двух основных пиков на гистограмме яркости, усреднением функции яркости и др. Для автоматического выбора порога предлагается следующая процедура.

1. Выбрать некоторую начальную оценку для значения порога T (предлагаемая величина равна среднему значению между минимальной и максимальной яркостью изображения).

2. Выполнить сегментацию с помощью порога T . В результате образуется две группы пикселей: $G1$ и $G2$. Область $G1$ состоит из пикселей, яркость которых больше или равна T , а область $G2$ – из пикселей, яркость которых меньше T .

3. Вычислить среднюю яркость пикселей $S1$ и $S2$ по областям $G1$ и $G2$.

4. Вычислить новое значение порога $T = 1 / 2 (S1 + S2)$.

Повторить шаги со 2-го по 4-й, пока разность порогов T для соседних итераций не станет меньше наперед заданного значения T .

Задание 1. Преобразовать палитровое изображение из файла *Athena.bmp* в бинарное с использованием автоматического выбора порога.

```
>>[X,map]=imread('c:\image\Athena.bmp');
```

```

>> I=im2double( ind2gray(X,map));
>> figure,imshow(I)
>> T = 0.5*(min(I(:)) + max(I(:)));
>> done = false;
>> while ~done
g=I>= T;
Tnext = 0.5*(double(min(I(g))) + double(max(I(~g))));
done = abs(T - Tnext) < 0.5;
T = Tnext;
end
>> bw = I> T;
>> figure, imshow(bw)

```

Для вычисления порога также можно использовать функцию `graythresh` пакета `IPT`, которая вычисляет порог по методу Отса.

Синтаксис

T = graythresh(S),

где *S* – исходное полутоновое изображение; *T* – глобальный порог в интервале [0 1].

Задание 2. Преобразовать цветное изображение из файла `bike.bmp` в бинарное с использованием функции `graythresh`.

```

>> rgb = imread('c:\Image\bike.bmp');
>> I= im2double(rgb2gray(rgb));
>> figure,imshow(I)
>> T = graythresh(I);
>> Bw=I > T;
>> figure, imshow(Bw)

```

Задание 3. Получить негатив с помощью бинаризации по порогу палитрового изображения, хранящегося в файле `Technlgy.bmp`.

```

>> [X,map] = imread('C:\Image\Earth.bmp');
>> I = ind2gray(X,map);
>> figure,imshow(I)
>> T=graythresh(I);
>> BW = A < T;
>> figure,imshow(BW)

```

В системе `MatLab` бинаризация изображения отсечением по порогу яркости выполняется функцией `im2bw`.

Синтаксис

BW = im2bw(I, threshold)

BW = im2bw(X, map, threshold)

BW = im2bw(RGB, threshold)

Функция `im2bw` создает бинарное изображение, используя отсечение по порогу яркости `threshold`. Для этой цели функция конвертирует полноцветные и палитровые изображения в полутоновые. Порог `threshold` должен задаваться в

диапазоне $[0,1]$. По умолчанию $\text{threshold} = 0.5$. Для вычисления порога можно воспользоваться функцией `graythresh`.

Задание 4. Выполнить бинаризацию палитрового изображения файла `Technlgy.bmp`:

а) выбрать порог по умолчанию:

```
>> [X,map] = imread('c:\Image\Technlgy.bmp');  
>> figure,imshow(X,map)  
>> BW = im2bw(X,map);  
>> figure,imshow(BW)
```

б) вычислить порог с помощью функции `graythresh`:

```
>> [X,map] = imread('c:\Image\Technlgy.bmp');  
>> I = im2double(ind2gray(X,map));  
>> figure,imshow(I)  
>> T = graythresh(I);  
>> BW = im2bw(I,T);  
>> figure,imshow(BW)
```

Для многих изображений глобальное пороговое значение не может использоваться из-за неоднородностей внутри областей фона и объекта. Для таких изображений требуются различные пороговые значения для различных частей изображения. Использование различных пороговых значений для различных частей изображения называется аддитивным или локальным пороговым разделением. Методика локального порогового разделения основана на разделении первоначального изображения на меньшие части и определении порога для каждой части изображения. В результате получается бинарное изображение с разрывами серого уровня на границах фрагментов. Для устранения неоднородностей применяются сглаживающие методики.

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Выполнить бинаризацию изображения файла Construc.bmp с использованием автоматического выбора порога.
2. Выполнить бинаризацию изображения файла Construc.bmp.
3. Получить негатив изображения файла Athena.bmp

3. ВОПРОСЫ

1. Как можно получить бинарное изображение?
2. Чем отличаются глобальное и локальное пороговое разделение?
3. Какие функции используются для бинаризации изображения в системе MatLab?

ЛАБОРАТОРНАЯ РАБОТА № 9. МОРФОЛОГИЧЕСКИЕ ОПЕРАЦИИ НАД БИНАРНЫМИ ИЗОБРАЖЕНИЯМИ

Цель работы – изучение морфологических операций, функций по реализации морфологических операций, приобретение практических навыков использования этих функций.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Рассмотрим несколько простых, но важных взаимосвязей между пикселями в цифровом изображении. Рассматриваемые понятия широко применяются в обработке бинарных изображений и морфологических операциях, при выделении объектов и вычислении их признаков. Для определенности в системе MatLab полагают, что пиксели со значениями, равными 1, относятся к объектам, а со значениями, равными 0, – к фону.

Соседние пиксели

Пиксел p с координатами (x, y) имеет четыре горизонтальных и вертикальных соседних пиксела с координатами: $(x + 1, y)$, $(x - 1, y)$, $(x, y + 1)$, $(x, y - 1)$. Эта группа пикселей x_1, x_3, x_5, x_7 , называемая «четыре соседа p » или «квартетом соседей», обозначается через $N_4(p)$:

x_4	x_3	x_2
x_5	p	x_1
x_6	x_7	x_8

Данные четыре пиксела находятся на одном расстоянии от (x, y) , а также некоторые из соседних пикселей p могут быть за пределами цифрового изображения, если (x, y) находится на границе изображения. Четыре диагональных соседних пиксела p имеют координаты

$(x + 1, y + 1)$, $(x + 1, y - 1)$, $(x - 1, y + 1)$, $(x - 1, y - 1)$

и обозначаются через $N_D(p)$. Эти точки вместе с четырьмя указанными выше называются восьми-соседями, или октетом соседей пиксела p и обозначаются через $N_8(p)$. Некоторые из точек $N_D(p)$ и $N_8(p)$ также могут выходить за пределы изображения, если (x, y) находится на границе изображения.

Связи

Объект называется четырехсвязный, если для каждого пиксела объекта среди квартета соседних пикселей существует хотя бы один, равный 1 и принадлежащий этому объекту.

Объект называется восьмисвязный, если для каждого пиксела объекта среди октета соседних пикселей существует хотя бы один, равный 1 и принадлежащий этому объекту.

Аналогичные определения связности можно ввести для фона. Четырехсвязность фона автоматически означает восьмисвязность объектов и наоборот. Среди связных областей объектов могут встречаться связные области из пикселей фона. Они называются дырами.

Морфологические операции

Теория морфологии (морфологии – наука о форме) рассматривает бинарное изображение в виде множеств его пикселей переднего плана (со значениями 1),

которое лежит в пространстве Z^2 (двухмерное пространство). Над бинарными изображениями можно совершать операции как над множествами с помощью следующих логических операций MatLab:

OR (\cup) – логическое сложение – объединение множеств
 $Z = \{z : z \in X \text{ or } z \in Y\} = X \cup Y;$

AND ($\&$) – логическое умножение – пересечение множеств
 $Z = \{z : z \in X, z \in Y\} = X \cap Y;$

NOT (\sim) – отрицание – дополнение множества $Z = Z^c = \{z : z \notin X\};$

DIFFERENCE ($X \& \sim Y$) – разность множеств X и Y
 $Z = \{z : z \in X, z \notin Y\} = X \setminus Y.$

Логические операции MatLab, применяемые к бинарным изображениям, приведены в табл. 9.1.

Таблица 9.1 Логические операции MatLab для бинарных изображений

Операции	Выражения MatLab для бинарных изображений	Имя
$X \cap Y$	$X \& Y$	AND
$X \cup Y$	$X Y$	OR
X^c	$\sim X$	NOT
$X \setminus Y$	$X \& \sim Y$	DIFFERENCE

Центральным отражением множества Z называется множество, определяемое по формуле

$$\hat{Z} = \{z : z = -X, z \in Z\}.$$

Параллельный перенос (или сдвиг) множества Z в точку $x = (x_1, x_2)$ обозначается $(Z)_x$ и задается формулой

$$(Z)_x = \{z : z = z + x, z \in Z\}.$$

Одним из основных понятий математической морфологии является понятие структурообразующего, или структурного элемента. Структурный элемент B – это множество, состоящее из двух непересекающихся подмножеств B_1 и B_2 , для которых определено общее начало.

Функция создания структурообразующего элемента strel

Функция strel строит структурообразующие элементы различных форм и размеров.

Синтаксис

$se = \text{strel}(\text{shape}, \text{parameters}),$

где se – структурообразующий элемент; shape – строка, задающая форму структурообразующего элемента; parameters – дополнительные параметры, которые уточняют информацию о его форме.

В табл. 9.2 приведены некоторые варианты задания формы структурообразующего элемента.

Таблица 9.2 Способы определения структурообразующего элемента

Вызов функции strel	Форма структурообразующего элемента	Дополнительные параметры
se = strel('diamond', R)	ромб	R – расстояние от центра структурообразующего элемента до крайней точки ромба
se = strel('disk', R)	круг	R – радиус структурообразующего элемента
se = strel('line', LEN, DEG)	линейный элемент	LEN – длина, DEG – угол наклона линии против часовой стрелки от горизонтальной оси (в градусах)
se = strel('pair', OFFSET)	две точки: первая точка находится в центре, а вторая смещена от первой на вектор OFFSET	OFFSET – двумерный вектор с неотрицательными целыми компонентами
se = strel('rectangle', MN)	прямоугольник	MN – двумерный вектор с неотрицательными целыми компонентами. MN(1) – число строк, MN(2) – число столбцов
se = strel(NHOOD)	элемент произвольной формы	NH00D – матрица из нулей и единиц, задающая его форму

Результатом выполнения функции strel является так называемый strel-объект. Для его разложения используется функция getsequence.

Синтаксис

decomp = getsequence(se)

Структурообразующие элементы разложения можно получить, индексируя переменную decomp.

Морфологические операции дилатации и эрозии имеют основополагающее значение при морфологической обработке изображений. Операция эрозии «ужимает» или «утончает» объекты двоичных изображений. Операция дилатации «наращивает» или «утолщает» на объекты двоичных изображениях. Способ и степень этих преобразований контролируется формой структурного элемента.

Эрозия

Эрозией множества X по B называется множество

$$Y = X \ominus B = \{x : (B)_x \cap X^C = \emptyset\},$$

где \emptyset – пустое множество.

Эрозия X по B состоит из пикселей с такими координатами, для которых сдвиг множества в эту точку не пересекается с фоном изображения X. Эрозия выполняется функцией imerode пакета IPT.

Синтаксис

D = imerode(S, se),

где S – это двоичное или полутоновое изображение; se – матрица из нулей и единиц, которая определяет структурообразующий элемент.

Задание 1. Выполнить эрозию изображения из файла erode.bmp. Удалить тонкие провода с изображения с сохранением всех остальных структур. Необходимо выбрать достаточно малый структурный элемент, чтобы он помещался внутри центрального квадрата, а также внутри толстых полосок у границ, и достаточно большим, чтобы он не помещался внутри удаляемых проводков.

```
>> [x,map] = imread('c:\image\erode.bmp');
>> I=im2double(ind2gray(x,map));
>> figure,imshow(I)
>> T=graythresh(I);
>> BW = I>T;
>> figure,imshow(I)
>> se = ones(15);
>> se(15,15) = 0;
>> se(15,1) = 0;
>> se(1,15) = 0;
>> se(1,1) = 0;
>> BW1 = imerode(BW,se);
>> figure,imshow(BW1)
>> se = ones(18);
>> se(18,18) = 0;
>> se(18,1) = 0;
>> se(1,18) = 0;
>> se(1,1) = 0;
>> BW1 = imerode(BW,se);
>> figure,imshow(BW1)
>> se = ones(60);
>> se(60,60) = 0;
>> se(60,1) = 0;
>> se(1,60) = 0;
>> se(1,1) = 0;
>> BW1 = imerode(BW,se);
>> figure,imshow(BW1)
```

Дилатация

Операцией, двойственной к эрозии, является дилатация (dilatation), которая определяется следующим образом:

$$Y = X \oplus B = \{x : (B)_x \cap X \neq \emptyset\}.$$

Дилатация выполняется функцией imdilate пакета IPT.

Синтаксис

$D = \text{imdilate}(S, se),$

где S – это двоичное или полутоновое изображение, se – матрица из нулей и единиц, которая определяет структурообразующий элемент.

Задание 2. Выполнить дилатацию изображения из файла TextRoman.bmp.

```
>> I = imread('c:\Image\TextRoman.bmp');
>> bw = I>150;
>> figure,imshow(bw)
>> se = [0 1 0;1 1 1;0 1 0];
>> bw1 = imdilate(bw,se);
>> figure,imshow(bw1)
```

Эрозия и дилатация – операции, предназначенные в первую очередь для выявления различных морфологических особенностей изображения с использованием различных структурных элементов. Так, эрозия посредством круга с радиусом r позволяет найти в изображении объекты, минимальный поперечный размер которых превышает $2r$. Если в качестве структурного элемента взять две точки, смещение между которыми определяется вектором h , эрозия позволит выделить объекты, имеющие соседей в направлении и на расстоянии, заданных этим вектором.

Задание 3. Найти объекты на изображении файла cgc.bmp, поперечный размер которых превышает 30 пикселей с помощью эрозии посредством круга с радиусом $r = 15$.

```
>> f1 = imread('c:\Image\cgc.bmp');
>> I=im2double(rgb2gray(f1));
>> T=graythresh(I);
>> bw=I<T;
>> figure,imshow(bw)
>> title('original')
>> r=15;
>> se = strel('disk', r)
>> bw1 = imerode(bw,se);
>> figure,imshow(bw1)
>> title('rezult')
```

Операции замыкание и размыкание являются комбинированием дилатации и эрозии.

Замыкание и размыкание

Морфологическое размыкание X по B обозначается $X \circ B$ и определяется как эрозия X по B , после которой выполняется дилатация результата по B :

$$Y = X \circ B = (X \ominus B) \oplus B.$$

Морфологическое замыкание множества X по B обозначается $X \bullet B$. Эта операция представляет собой эрозию, примененную к результату дилатации:

$$Y = X \bullet B = (X \oplus B) \ominus B.$$

Преобразование размыкания и замыкания реализовано функциями `imopen` и `imclose` соответственно.

Синтаксис

`D = imopen(S, se)`

`D = imclose(S, se),`

где S – это двоичное или полутоновое изображение; se – матрица из нулей и единиц, которая определяет структурообразующий элемент.

Задание 4. Выделить объекты на изображении файла `Ex4.bmp`, используя операции замыкания и размыкания.

```
>> [x,map]= imread('C:\Image\Ex4.bmp');
>> figure,imshow(x,map)
>> I=im2double(ind2gray(x,map));
>> T=graythresh(I);
>> bw=im2bw(I,T);
>> figure,imshow(bw), title('bw')
>> R=18, se = strel('disk', R);
>> bwo=imopen(bw,se);
>> figure,imshow(bwo)
>> r=int2str(R);
>> s=cat(2,'bwo-',r)
>> title(s)
>> R=5;
>> se = strel('disk', R);
>> bwcl=imclose(bwo,se);
>> figure,imshow(bwcl)
>> r=int2str(R);
>> s=cat(2,'bwcl-',r)
>> title(s)
```

Морфологическая реконструкция

Для поиска нужных объектов изображения можно использовать морфологическую реконструкцию. Одно изображение, которое называется маркером, является исходной точкой преобразования. Другое изображение, маска, накладывает определенные ограничения (связи) на отображение.

Если g – это маска ($mask$), а f – маркер ($marker$), то реконструкция g по f определяется следующей итеративной процедурой.

1. Присвоить h_i маркерное изображение f .
2. Построить структурообразующий элемент $B = ones(3)$.
3. Повторять: $h_{i+k} = (h_k \oplus B) \cap g$ до тех пор, пока не станет $h_{i+k} = h_k$.

Маркер f должен быть подмножеством g , т. е. $f \subseteq g$.

Функция `imreconstruct` из пакета `IPТ` выполняет реконструкцию.

Синтаксис

`out = imreconstruct(marker, mask),`

где $marker$ – это изображения-маркер; $mask$ – это изображения-маска.

Задание 5. Найти в тексте из файла TextRoman.bmp буквы, у которых имеются длинные вертикальные черточки.

```
>> I = imread('c:\Image\TextRoman.bmp');
>> I=im2double(I);
>> T=graythresh(I);
>> bw=I>T;
>> figure,imshow(bw)
>> title('original')
>> bwr = imopen(bw, ones(3, 1));
>> figure,imshow(bwr)
>> title('razmikanie')
>> bwR = imreconstruct(bwr, bw);
>> figure,imshow(bwR)
>> title('Rezult')
```

Функция bwmorph

Морфологические операции над бинарным изображением также можно выполнить, используя функцию bwmorph.

Синтаксис

$BW_D = \text{bwmorph}(BW_S, \text{operation}, n)$

Функцией bwmorph создается бинарное изображение BW_D при обработке морфологическим фильтром исходного бинарного изображения BW_S n раз (по умолчанию $n = 1$). Описание некоторых морфологических операций приведено в табл. 9.3.

Алгоритм.

В структурообразующем элементе, который также называют маской морфологического фильтра, ненулевые значения определяют, какие из соседних пикселей следует учитывать при осуществлении операции. При эрозии бинарного изображения пиксел исходного изображения сбрасываются в ноль, если хотя бы один из пикселей окрестности, соответствующий ненулевому элементу маски, равен 0. При наращивании бинарного изображения пиксел исходного изображения устанавливается в 1, если хотя бы один из пикселей окрестности, соответствующий ненулевому элементу маски, равен 1.

В функции bwmorph в операциях эрозии и наращивания используется структурообразующий элемент 3×3 вида:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}.$$

Данные правила применяются ко всем пикселям изображения.

Таблица 9.3 Морфологические операции функции bwmorph

Тип операции	Описание
‘erode’	Эрозия объекта. Приводит к замене значений граничных пикселей объекта на ноль
‘dilate’	Наращение объекта. Приводит к замене пикселей фона, граничащих с объектом на единицу
‘open’	Открытие. Представляет собой последовательное применение эрозии и наращения. Приводит к соединению областей фона, ранее разъединенных узкими участками пикселей объекта
‘close’	Закрытие. Представляет собой последовательное применение наращения и эрозии. Приводит к удалению небольших по площади фрагментов фона внутри объектов, например «дыр»
‘tophat’	«Верх шляпы». Соответствует вычитанию из исходного изображения результата его открытия
‘bothat’	«Низ шляпы». Соответствует вычитанию исходного изображения из результатов его закрытия
‘skel’	Построение остова объекта. Результат операции – связная линия толщиной в один пиксел, проходящая посередине
‘thin’	Утончает объекты без дыр до минимальной средней линии. Утончает объекты с дырами до колец
‘shrink’	Сжимает объекты без внутренних дыр в точки. Сжимает объекты с дырами в кольца
‘thicken’	Утолщает объекты без соединения несвязных частей

Задание 6. Разделить слишком слипшиеся объекты с помощью морфологических операций. Для этого изображение подвергается эрозии до тех пор, пока не исчезнет слипание объектов. В данном случае необходимо 10 итераций эрозии. Результат эрозии помещается в изображение BW2. Затем для BW2 «утолщаются» объекты (строится остов фона). Результат логического И изображений BW1 & BW2 дает изображение с пикселями фона в местах слипания объектов.

```
>> R = imread('c:\Image\cgc.bmp');
>> I = im2double(rgb2gray(R));
>> T=graythresh(I);
>> BW1 = I<T;
>> figure,imshow(BW1)
>> BW2 = bwmorph(BW1,'erode',10);
>> figure,imshow(BW2)
>> title('erode')
>> BW3 = bwmorph(BW2,'thicken',inf);
>> figure,imshow(BW3)
>> title('thicken')
>> BW4 = BW1&BW3;
>> figure,imshow(BW4)
>> title('Rezult')
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Выделить объекты в изображении, имеющие соседей в направлении и на расстоянии, заданные вектором h . Для этого в качестве структурного элемента взять две точки, смещение между которыми определяется вектором: а) $h = [0 \ 3]$; б) $h = [3 \ 0]$. Для получения структурного элемента использовать функцию `strel` с параметром 'pair'.

2. Найти объекты изображения из файла `Cgs.bmp`, размер которых больше или равен 100.

3. ВОПРОСЫ

1. Какие логические операции над бинарными изображениями вы знаете?
2. В чем назначение структурообразующего элемента в морфологических операциях?
3. Для чего используются морфологические операции?
4. Какие морфологические операции обработки изображения относятся к базовым?
5. Какие операции являются комбинированием эрозии и дилатации?
6. Какие функции пакета IRT выполняют операции эрозии и дилатации, замыкания, размыкания?

ЛАБОРАТОРНАЯ РАБОТА № 10. СЕГМЕНТАЦИЯ ИЗОБРАЖЕНИЙ

Цель работы – изучение операций по сегментации изображений, функций, реализующих операции по сегментации изображения, и приобретение практических навыков использования этих функций.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Сегментация изображения представляет собой разделение изображения на области по сходству свойств (признаков) в их точках. Признаки подразделяются на естественные и искусственные. Естественные признаки устанавливаются простым (визуальным) анализом изображения, а искусственные – в результате специальной обработки различных измерений. Примерами естественных признаков являются структура, текстура, яркость объекта. Примеры искусственных признаков: гистограммы распределения яркости, спектр и др.

К основным видам сегментации изображений относится сегментация по яркости, цветовым координатам, контурам, форме.

Методы сегментации

Сегментация методом выращивания областей

Для сегментации изображения можно использовать метод выращивания областей – группирование пикселей или подобластей в более крупные области по заранее заданным критериям роста. Берутся «центры кристаллизации», а затем на них наращиваются области путем добавления к каждому центру тех соседних пикселей, которые по своим свойствам близки к центру кристаллизации (например, имеют яркость или цвет в определенном диапазоне). Ниже приведена функция `regiongrow`, которая выполняет выращивание областей.

Синтаксис

`[g, NR, SI, TI] = regiongrow(f, S, T),`

где f – это сегментируемое изображение, а параметр S – массив (с размерами как у f) или скаляр. Если S – массив, то он содержит 1 в тех позициях, где расположены центры кристаллизации и 0 во всех остальных местах. Если S является скаляром, то он задает значение яркости пикселей, которые становятся центрами кристаллизации. Аналогично, T может быть массивом (с размерами, как у f) или скаляром. Если T – массив, то его элементы являются локальными пороговыми значениями для f . Скаляр T определяет глобальный порог.

Задание 1. Создать функцию `regiongrow`.

```
function [g, NR, SI, TI] = regiongrow(f, S, T)
if numel(S) == 1
    SI = f == S;
    S1 = S
else
    SI = bwmorph(S, 'shrink', Inf);
    J = find(SI);
    S1 = f(J);
```

```

end
TI = false(size(f));
for K = 1:length(S1)
seedvalue = S1(K);
S = abs(f - seedvalue) <= T;
TI = TI | S;
end
[g, NR] = bwlabel(imreconstruct(SI, TI));

```

Задание 2. Выполнить сегментацию наращиванием областей для изображения, хранящегося в файле Finance.bmp, используя функцию regiongrow.

```

[x,map] = imread('c:\image\Finance.bmp');
I = im2double(ind2gray(x,map));
figure,imshow(I)
S = 0.9783; T = 0.0651; % эти значения находятся экспериментально (взяты
из изображения)
[g, NR, SI, TI] = regiongrow(I, S, T);
figure,imshow(TI)

```

Также для сегментации используется метод разделения. Функция MatLab, реализующая подобный алгоритм, приведена ниже.

Сегментация методом разделения

Изображение разбивается на непересекающиеся блоки, которые с помощью некоторого критерия проверяются на однородность.

Функция сегментации полутоновых изображений методом разделения qtdecomp

Синтаксис

```
A = qtdecomp(I,threshold,mindim)
```

Функция qtdecomp осуществляет сегментацию полутоновых изображений методом разделения. В функции qtdecomp каждый блок разбивается на 4 неперекрывающихся блока одинакового размера. На первом шаге алгоритма блоком считается все изображение. Мельчайшим по размерам является блок, который нельзя разделить на 4 блока одинакового размера, т. е. такой блок, у которого число строк или число столбцов нечетное. Таким образом, в функции qtdecomp рекомендуется использовать изображения с размерами, равными степеням двух. Функция $A = qtdecomp(I,threshold,mindim)$ осуществляет сегментацию полутонового изображения I методом разделения и помещает результат в разреженный массив A (тип данных sparse MatLab). Элементам матрицы $A(r,c)$, соответствующим координатам левых верхних углов блоков на исходном изображении I , присваиваются значения, определяющие размеры каждого блока. Блок считается однородным, если разница между максимальным и минимальным значением пикселей блока меньше параметра threshold. Параметр mindim определяет минимальный размер блока.

Функция получения блоков из квадрато-дерева результатов сегментации qtgetblk

Синтаксис

`[vals, idx] = qtgetblk(I, A, dim)`

Функция возвращает в массив `vals` все блоки размером `dim`, получившиеся в результате сегментации полутонового изображения `I` с помощью функции `qtdecomp`. В параметре `A` передается разреженный массив, описывающий quadro-дерево с результатами сегментации. Координаты левых верхних углов блоков, помещенных в массив `vals`, находятся в векторе `idx`. Если нет ни одного блока размером `dim`, то всем возвращаемым параметрам присваиваются значения пустых `vals` матриц.

Функция замены блоков – результатов сегментирования `qtsetblk`

Синтаксис

`ID = qtsetblk(IS, A, dim, vals)`

Функция создает новое полутоновое изображение `ID`, заменяя в исходном полутоновом изображении все блоки размера `dim`, полученные в результате сегментации с помощью функции `qtdecomp`, на блоки из массива `vals`. В параметре `A` передается разреженный массив, описывающий quadro-дерево с результатами сегментации. Данная функция используется для преобразования изображения в соответствии с результатами сегментации методом разделения.

Рассмотрим работу функции `qtdecomp` совместно с функциями `qtgetblk` и `qtsetblk` для полутонового изображения размера 8×8 пикселей. Формат представления данных – `uint8`. Будем считать, что блок изображения является однородным, если величина разброса яркостей пикселей в блоке не превышает 10 градаций яркости. Установим минимально возможный размер блока. В нашем случае он равен двум.

Будем считать, что к объекту относятся блоки, средняя яркость которых не превышает 50. Требуется изменить исходное изображение так, чтобы пикселям блоков, относящихся к объекту, было присвоено значение 1, а пикселям блоков, не относящихся к объекту, 0.

Задание 3. Выполнить сегментацию небольшого текстового изображения методом разделения.

Исходное изображение:

```
>> I = [ 10 11 10 15 20 25 47 51  
11 14 17 13 27 29 52 55  
12 13 11 10 24 47 56 60  
13 14 11 13 49 54 74 77  
15 16 43 48 79 82 87 86  
17 18 45 50 85 80 80 84  
29 51 50 59 80 83 83 85  
59 61 58 61 81 85 86 88 ];
```

Сегментация методом разделения: размер минимального блока 2×2 ; блок считается однородным, если в его пределах яркость изменяется меньше, чем на 10 градаций.

```
>> A = qtdecomp(I,10,2);
```


Для удобства визуального анализа предварительно преобразуем разреженную матрицу A в обычную матрицу M с помощью функции full.

```
>> M = full(A)
M =
4 0 0 0 2 0 2 0
0 0 0 0 0 0 0 0
0 0 0 0 2 0 2 0
0 0 0 0 0 0 0 0
2 0 2 0 4 0 0 0
0 0 0 0 0 0 0 0
2 0 2 0 0 0 0 0
0 0 0 0 0 0 0 0
```

В результате сегментации получили 2 блока размером 4×4 (левая верхняя и правая нижняя части изображения) и 8 блоков размером 2×2 .

Переберем в цикле все возможные размеры блоков: 8, 4, 2.

```
>> dim = 8;
>> while dim >= 2
% получить в переменной blocks все блоки размера dim.
[blocks,idx] = qtgetblk(I,A,dim);
[x y n] = size(blocks);
% если блоки такого размера есть в quadro-дереве,
if n>0
% то перебираем все блоки размера dim
for j = 1:n
% если среднее значение яркости пикселей в пределах блока меньше 50
if (mean2(blocks(:,j))<50)
% то заменяем значения всех пикселей блока на 1,
blocks(:,j) = ones(dim,dim);
else
% иначе заменяем значения всех пикселей на 0.
blocks(:,j) = zeros(dim,dim);
end;
end % end for
% устанавливаем новые значения всех пикселей размера dim
I = qtsetblk(I,A,dim,blocks);
end; % end if
dim = dim/2;
end % end while % ссылка на 1 из списка литературы
% получившееся изображение I
>> I
I =
1 1 1 1 1 1 0 0
1 1 1 1 1 1 0 0
1 1 1 1 1 1 0 0
```

```

1 1 1 1 1 1 0 0
1 1 1 1 0 0 0 0
1 1 1 1 0 0 0 0
0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0

```

Задание 4. Выполнить сегментацию реального изображения из файла cotton3.bmp

```

rgb = imread('c:\Image\cotton3.bmp');
I = im2double(rgb2gray(rgb));
figure,imshow(I)
T=graythresh(I);
A = qtdecomp(I,0.1,2);
dim = 8;
while dim >= 2
[blocks,idx] = qtgetblk(I,A,dim);
[x y n] = size(blocks);
if n>0
for j = 1:n
if (mean2(blocks(:,j))<T)
blocks(:,j) = ones(dim,dim);
else
blocks(:,j) = zeros(dim,dim);
end
end
I1 = qtsetblk(I,A,dim,blocks);
end
dim = dim/2;
end
figure,imshow(I1)

```

Функция выбора интересующей области по цвету roicolor

Синтаксис

BW = roicolor(S,low,high)

BW = roicolor(S,v)

Для любого варианта вызова функции roicolor бинарное изображение формируется по следующему алгоритму: пикселу бинарного изображения BW(r, c) присваивается значение 1, если яркость пиксела S(r, c) исходного полутонового изображения или индекс S(r, c) палитрового изображения принадлежит диапазону [low, high] или любому из значений вектора v. В противном случае BW(r, c) присваивается значение 0.

Задание 5. Выбрать цветовые области из изображения файла chip.bmp, задавая индексы с помощью гистограммы.

```
>> [x,map]=imread('C:\Image\chip.bmp');
```

```
>> figure,imhist(x,map),title('histogramma')
>> figure,imshow(x,map),
>> bw=roicolor(x,9,12);
>> figure,imshow(bw),title('9 - 12')
>> x1=immultiply(bw,x);
>> figure,imshow(x1,map),title('9 - 12')
>> bw=roicolor(x,3,8);
>> figure,imshow(bw),title('3 - 8')
>> x1=immultiply(bw,x);
>> figure,imshow(x1,map),title('3 - 8')
```

Яркостный срез

Этот метод помогает выделить определенный диапазон яркости:

$$\forall A_{i,j} \quad B_{i,j} = \begin{cases} 0 & A_B < A_{i,j} < A_H \\ A_{i,j} & A_H \leq A_{i,j} \leq A_B \end{cases} \quad \text{или}$$

$$B_{i,j} = \begin{cases} 0 & A_B < A_{i,j} < A_H \\ K & A_H \leq A_{i,j} \leq A_B \end{cases},$$

где A_H – нижнее граничное значение определяемого диапазона яркости; A_B – верхнее граничное значение определяемого диапазона яркости.

Для выполнения яркостного среза можно использовать функцию `imrpxel`.

Синтаксис

$P = \text{imrpxel}(S, c, r)$

Функция `imrpxel` возвращает значения красной, зеленой и синей составляющих цвета для заданных координат – c и r – векторов значений столбцов и строк.

Задание 6. Выполнить яркостный срез полноцветного изображения файла `cotton3.bmp`.

```
[img] = imread('c:\Image\cotton3.bmp');
[m,n,k] = size(img)
img = im2double(img);
R = zeros(m,n,3);
z = [0.1,0.8;0.1,0.8; 0.1,0.9];
for y = 1:m
for x = 1:n
b = imrpxel(img,x,y);
if ((b(1)>= z(1,1))&(b(1)<= z(1,2)))& ((b(2)>= z(2,1))&(b(2)<= z(2,2))) ...
& ((b(3)>= z(3,1))&(b(3)<= z(3,2)))
R(x,y,1) = b(1); R(x,y,2) = b(2); R(x,y,3) = b(3);
else
R(x,y,1) = 0; R(x,y,2) = 0; R(x,y,3) = 0;
end
end
end
```

```
figure, imshow(img)
figure, imshow(R)
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Выполнить сегментацию изображений файла Clouds.bmp методом разделения.
2. Выполнить сегментацию изображений файла Construc.bmp методом выращивания областей.
3. Выбрать цветовые области из изображения файла bike.bmp, задавая индексы с помощью гистограммы.
4. Выполнить яркостный срез полноцветного изображения файла bike.bmp, задавая диапазон r от 0.2 до 0.8; g от 0.2 до 0.7; b от 0.1 до 0.7.

3. ВОПРОСЫ

1. В чем заключается сегментация изображения?
2. Какие признаки используются для сегментации?
3. В чем заключается метод выращивания областей, использующийся для сегментации изображения?
4. В чем заключается метод разделения, использующийся для сегментации изображения?
5. Что является входными параметрами функции сегментации методом разделения?
6. В чем заключается преобразование яркостного среза?
7. Какие параметры возвращает функция `imrixel`?

ЛАБОРАТОРНАЯ РАБОТА № 11. ФУНКЦИИ ПОИСКА ОБЪЕКТОВ И ВЫЧИСЛЕНИЯ ИХ ПРИЗНАКОВ

Цель работы – изучение функций, осуществляющих поиск объектов бинарных изображений, определение их характеристик и приобретение навыков использования этих функций.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

При анализе формы областей изображения применяют различные метрические, топологические и аналитические характеристики: расстояние, площадь, периметр, число Эйлера, функции кривизны и их производные. Ниже приведены функции MatLab, позволяющие выполнять выделение нужных объектов на изображении, поиск связанных областей и вычисление морфометрических признаков: площадей объектов, центров масс объектов, коэффициентов формы и других метрических и топологических характеристик.

Функция выделения объектов bwselect

Синтаксис

$BW_D = bwselect(BW_S, n)$

$BW_D = bwselect(BW_S, c, r, n)$

Функция $BW_D = bwselect(BW_S, n)$ выводит изображение BW_S на экран и предоставляет пользователю возможность интерактивно отметить затравочные пиксели (пиксел, с которого начинается выделение объекта, называется затравочным). Координаты затравочных пикселей задаются щелчком левой кнопки мыши. Предыдущий можно удалить клавишей Backspace или Delete. Последний затравочный пиксел задается двойным щелчком левой кнопки или однократным щелчком правой кнопки мыши. Нажатие клавиши Enter завершает процесс выделения затравочных пикселей. Все отмеченные объекты переносятся сразу после нажатия клавиши Enter, после этого сразу создается новое бинарное изображение BW_D .

Функция $BW_D = bwselect(BW_S, c, r, n)$ переносит с изображения BW_S на изображение BW_D все объекты с координатами затравочных пикселей из векторов c и r .

Параметр n для всех рассматриваемых функций bwselect задает критерий связности. Он может принимать значение 4 или 8 (по умолчанию $n = 8$).

Функция поиска объектов в бинарных изображениях bwlabel

Синтаксис

$[L, num] = bwlabel(BW, n)$

Функция bwlabel ищет на бинарном изображении BW связанные области пикселей объектов и создает матрицу номеров объектов L . Элементы, имеющие значение 1, относятся к первому объекту, 2 – ко второму и т. д. Элементы, имеющие значение 0, относятся к фону. В параметре num возвращается количество объектов, найденных на изображении BW . Параметр n задает критерий связности, используемый для нахождения связанных областей объектов. Он может принимать значение 4 или 8 (по умолчанию $n = 8$).

Функция вычисления признаков объектов imfeature

Синтаксис

feats = imfeature(L,measurement,n)

Функция вычисляет признаки всех объектов, занесенных в матрицу номеров объектов L функцией bwlabel. Значения признаков возвращаются в массиве структур feats.

Вычисляемые признаки

Вычисляемые функцией imfeature признаки можно условно разбить на группы: различные типы изображений объектов (табл. 11.1), габариты объекта (табл. 11.2), различные варианты определения площади объекта (табл. 11.3), коэффициенты формы (табл. 11.4) и все оставшиеся признаки.

Таблица 11.1 Типы изображений объектов

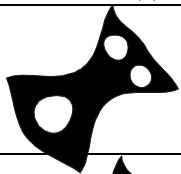
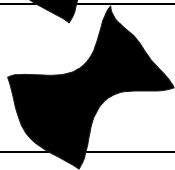
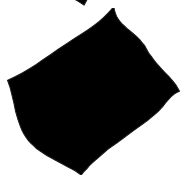
Параметр measurement	Описание	Вид
'Image'	Изображение объекта	
'FilledImage'	Изображение объекта с залитыми дырами	
'ConvexImage'	Изображение 'залитого' пикселями объекта выпуклого многоугольника, в который вписан объект	

Таблица 11.2 Габариты объекта

Параметр measurement	Описание	Вид
'BoundingBox'	Ограничивающий прямоугольник	Массив [x y width height], где (x y) – координаты левого верхнего угла прямоугольника; width – ширина; height – высота
'ConvexHull'	Выпуклый многоугольник, в который вписан объект	Матрица $M \times 2$; каждая строка матрицы содержит (x, y) координаты вершин многоугольника
'Extreme'	Экстремальные координаты объекта	Матрица 8×2 содержит экстремальные координаты объекта
'PixelList'	Список всех пикселей объекта	$N \times 2$ – матрица; каждая строка содержит (x, y) – координаты центров всех пикселей объекта

Таблица 11.3 Площадь объектов

Параметр measurement	Описание
'Area'	Площадь: количество пикселей объекта
'FilledArea'	Полная площадь – соответствует площади изображения FilledImage
'ConvexArea'	Выпуклая площадь: площадь выпуклого многоугольника, в который вписан объект. Соответствует площади изображения ConvexImage

Таблица 11.4 Коэффициенты формы объектов

Параметр measurement	Описание
'Solidity'	Коэффициент выпуклости: равен отношению $\frac{Area}{ConvexArea}$ площади к выпуклой площади объекта:
'Extent'	Коэффициент заполнения: равен отношению площади к площади ограничивающего прямоугольника: $\frac{Area}{width \times height}$
'Eccentricity'	Эксцентриситет эллипса с главным моментом инерции объекта

Оставшиеся признаки объектов

К вычисляемым признакам также относятся: 'Centroid' – координаты центра масс; 'EquivDiameter' – диаметр круга с площадью Area; 'MajorAxisLength' – длина максимальной оси инерции; 'MinorAxisLength' – длина минимальной оси инерции; 'Orientation' – угол в градусах между максимальной осью инерции и осью X; 'EulerNumber' – число Эйлера: количество объектов с одинаковым индексом в матрице L минус количество дыр в этих объектах.

Задание 1. Выполнить выделение, поиск объектов изображения файла cgc.bmp и вычислить их признаки. Для этого надо выполнить следующие операции:

– Прочитать изображение файла cgc.bmp с диска, преобразовать в полутоновое, получить позитив:

```
rgb = imread('c:\Image\cgc.bmp');
```

```
I=im2double(rgb2gray(rgb));
```

```
I=max(I(:))-I;
```

```
figure, imshow(I);
```

получить информацию о размере и записать ее в файл raz.txt на диск c:

```
info = imfinfo('c:\Image\cgc.bmp');
```

```
[F, mes] = fopen('c:\raz.txt', 'wt');
```

```
fprintf(F, '%d %d', info.Width, info.Height);
```

```
fclose(F);
```

– Выбрать три объекта интерактивно, отмечая их крестиком с помощью мыши:

```
bw = bwselect(I);
```

```
figure, imshow(bw);
```

– Получить матрицу номеров объектов L и количество объектов num, записать количество объектов в файл 'num.txt' на диск c:

```
[L, num] = bwlabel(bw);
```

```
[F, mes] = fopen('c:\num.txt', 'wt');
```

```
fprintf(F, '%d ', num);
```

```

fclose (F);
– Найти коэффициенты выпуклости объектов и их изображения:
feats = imfeature (L, 'Image','Solidity');
– Записать коэффициенты выпуклости объектов в файл 'k1.txt' на диск:
[F, mes] = fopen('c:\k1.txt', 'wt');
for i = 1:num
fprintf (F, '%5.3f ', feats (i).Solidity);
end;
fclose (F);
– Вывести изображения объектов на экран:
for i = 1:num
figure,imshow(feats(i).Image);
end;
– Записать изображения объектов на диск:
for i = 1: num
ch = int2str (i);
ch1 = 'c:\f';
ch2 = '.bmp';
str = strcat (ch1, ch, ch2);
imwrite (feats (i).Image, str);
end;

```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Выделить объекты из изображения файла Ex4.bmp, определить площадь всех объектов, результаты записать на диск, поместить каждый объект в отдельное изображение, вывести на экран.

3. ВОПРОСЫ

1. Как выделить нужные объекты изображения с помощью функции `bwselect`?
2. Что из себя представляет матрица номеров объектов?
3. Какие признаки вычисляет функция `imfeature`?
4. Какие виды изображений объекта можно получить с помощью функции `imfeature`?
5. Какие типы габаритов объекта можно определить с помощью функции `imfeature`?
6. Какие типы площадей объекта можно определить с помощью функции `imfeature`?
7. Какие коэффициенты формы объекта можно определить с помощью функции `imfeature`?
8. Какие вычисляемые функцией `imfeature` признаки относятся к оставшимся?

ЛАБОРАТОРНАЯ РАБОТА № 12. ФУНКЦИИ СПЕКТРАЛЬНОГО АНАЛИЗА ИЗОБРАЖЕНИЯ

Цель работы – изучение функций по спектральному анализу изображений и приобретение практических навыков их использования.

1. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Функции преобразования Фурье

1. Прямое преобразование Фурье

Синтаксис

$Y = \text{fft2}(X)$

$Y = \text{fft2}(X, m, n)$

Функция $Y = \text{fft2}(X)$ – двумерное быстрое преобразование Фурье (БПФ) – возвращает результат в матрице комплексных чисел Y , имеющей тот же размер, что и матрица X .

Функция $Y = \text{fft2}(X, m, n)$ вычисляет БПФ матрицы $m \times n$, при необходимости дополняя нулями или усекая исходную матрицу X . Возвращается матрица Y размером $m \times n$.

Массив Y имеет формат представления данных `double`.

Алгоритм: двумерное преобразование Фурье вычисляется выполнением двух одномерных преобразований `fft(fft(x).')`. Сначала вычисляется ДПФ каждого столбца, а затем каждой строки результата.

2. Обратное двумерное БПФ

Синтаксис

$Y = \text{ifft2}(X)$

$Y = \text{ifft2}(X, m, n)$

Функция $Y = \text{ifft2}(X)$ вычисляет обратное двумерное ДПФ, возвращая результат в матрице Y , имеющий тот же размер, что и матрица X .

Функция $Y = \text{ifft2}(X, m, n)$ вычисляет обратное двумерное преобразование Фурье матрицы $m \times n$, при необходимости дополняя нулями или усекая исходную матрицу X . Возвращается матрица Y размером $m \times n$.

Задание 1. Необходимо выполнить следующие действия:

– Сформировать двумерный сигнал, являющийся суммой двух пространственных волн, $I = \sin(2\pi x/8) + \sin(2\pi y/16)$; вывести его на экран:

```
>> [x,y] = meshgrid(1:32);  
>> I = sin(2*pi*x/8)+sin(2*pi*y/16);  
>> figure,imshow(mat2gray(I));  
>> figure,surf(x,y,I);  
>> colormap(gray)  
>> shading interp
```

– Вычислить спектр сигнала и вывести его на экран (так как спектр симметричен, четыре отчетливых пика соответствуют двум частотам волн, из которых сформировано исходное изображение).

```
>> h = fft2(I);
```

```
>> figure,surf(x,y,abs(h));
>> colormap(gray)
>> shading interp
```

– Удалить из спектра одну из частот и восстановить сигнал с помощью обратного преобразования Фурье:

```
>> h(1:32,1:2) = 0;
>> figure,surf(x,y,abs(h));
>> colormap(gray)
>> shading interp
>> I = ifft2(abs(h));
>> figure,imshow(mat2gray(I));
>> figure,surf(x,y,abs(I));
>> colormap(gray)
>> shading interp
```

Сокращение избыточности изображения

Избыточность информации может быть описана функцией корреляции между отсчетами изображения. Она проявляется во взаимной статистической прогнозируемости близлежащих отсчетов, взятых с изображения. Конечной целью операций сжатия является устранение этой статистической прогнозируемости, то есть необходимо уменьшить коррелируемость отсчетов. Фурье-преобразование изображений переводит изображение в новую систему координат, что приводит к уменьшению коррелируемости отсчетов, после чего основные принципы сжатия пространственных преобразований заключаются в избирательной передаче коэффициентов разложения, что приводит к уменьшению полосы частот.

Многие методы сокращения полосы частот могут быть проанализированы с точки зрения двумерной дискретизации. На рис. 12.1 приведена схема двумерной дискретизации.

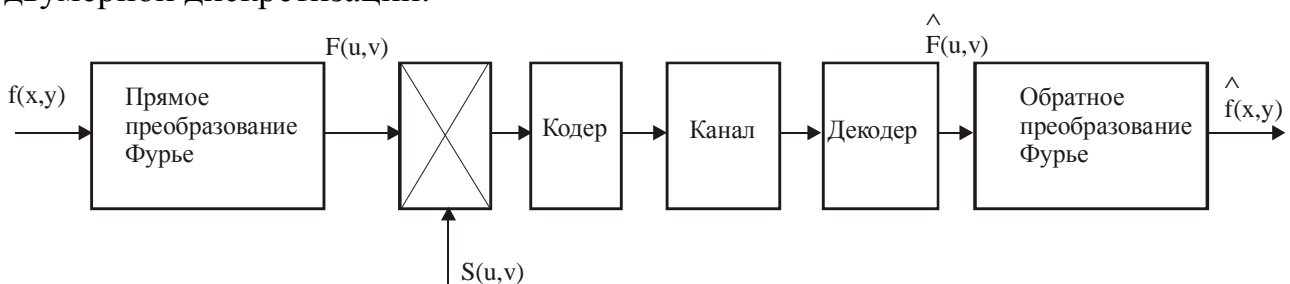


Рис. 12.1

Для функции дискретизации типа шахматного поля, которая вычисляется по формуле

$$S(u, v) = \frac{1 + (-1)^{v+u}}{2},$$

восстановленное изображение (после обратного преобразования Фурье) состоит из исходного изображения и наложенного на него такого же изображения, сдвинутого по вертикали и горизонтали:

$$\hat{f}(x, y) = \frac{1}{2} \left[f(x, y) + f\left(x + \frac{N}{2}, y + \frac{N}{2}\right) \right].$$

Задание 2. Выполнить восстановление изображения после двумерной дискретизации с использованием функции дискретизации типа шахматного поля.

Пояснение. Изображение берется из файла 'Technlgy.bmp', преобразовывается в полутоновое. Для создания изображения, сдвинутого по вертикали и горизонтали, вырезается соответствующий фрагмент из исходного изображения (этого надо предварительно определить его размеры) и дополняется нулями до размера исходного изображения. Затем восстановленное изображения получается из сложения исходного и сдвинутого изображения.

```
>>[X,map] = imread('C:\Image\Technlgy.bmp');
>> I = ind2gray(X,map);
>> figure,imshow(I)
>> [M N] = size(I);
>> rect = [floor(N/2), floor(M/2), floor(N/2)-1, floor(M/2)-1];
>> I1 = imcrop(I,rect);
>> IM = zeros(floor(M/2),floor(N/2));
>> IP = [I1,IM];
>> IN = zeros(floor(M/2),N);
>> IR = [IP;IN];
>> figure,imshow(IR)
>> D = I+IR;
>> figure,imshow(D)
```

2. ЗАДАНИЯ ДЛЯ САМОСТОЯТЕЛЬНОГО РЕШЕНИЯ

1. Выполнить прямое преобразование Фурье полутонового изображения, полученного из палитрового изображения, хранящегося в файле 'Athena.bmp', удалить из спектра низкочастотные компоненты и восстановить с помощью обратного преобразования Фурье.

2. Выполнить двумерную дискретизацию с использованием функции дискретизации типа шахматного поля над полутоновым изображением, полученного из палитрового изображения, хранящегося в файле "Athena.bmp", и получить восстановленное изображение.

3. ВОПРОСЫ

1. Какие функции используются для выполнения двумерного прямого и обратного преобразования Фурье в системе MatLab?
2. Зачем используется двумерная дискретизация? Приведите примеры функции дискретизации.

СПИСОК РЕКОМЕНДУЕМОЙ ЛИТЕРАТУРЫ ДЛЯ УГЛУБЛЕННОГО ИЗУЧЕНИЯ МАТЕРИАЛА

1. Рудаков, П.И. Обработка сигналов и изображений matlab 5x [Текст] / П.И. Рудаков, В.И. Сафонов; под общ. ред. В.Г. Потемника. – М.: ДИАЛОГ–МИФИ, 2000. – 416 с.
2. Гонсалес, Р. Цифровая обработка изображений в среде matlab [Текст] / Р. Гонсалес, В. Вудс, С. Эддинс; пер. с англ. В.В. Чепыжова. – М.: Техносфера, 2006. – 512 с.
3. Косых, В.П. Цифровая обработка изображений [Текст]: учеб. пособие / В.П. Косых. – Новосибирск: НГУ, 2006. – 95 с.
4. Лайонс, Ричард. Цифровая обработка сигналов [Текст] / Ричард Лайонс; пер. с англ., под ред. А.А. Бритова. – 2-е изд. – М.: БИНOM-Пресс, 2007. – 653 с.