МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ:
Проректор по учебной работе
_____ О.Г. Локтионова
«___» _____ 2018 г.

ЦИФРОВАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ В СРЕДЕ MATLAB.

Методические указания по выполнению лабораторных работ
по дисциплине «Компьютерное зрение»
для студентов направления подготовки магистров
09.04.04 «Программная инженерия»

Курск 2018

УДК 004.932

Составители: Р.А. Томакова, В.В. Апальков

Рецензент
Кандидат технических наук, доцент *Е.И. Аникина*

Цифровая обработка изображений в среде MatLab: методические указания по выполнению лабораторных работ / Юго-Зап. гос. ун-т; сост.: Р.А. Томакова, В.В. Апальков. – Курск: ЮЗГУ, 2018. – 54 с.

Излагаются цели лабораторных работ, в теоретической части рассматриваются методы цифровой обработки изображений. В практической части приводятся примеры выполнения заданий на лабораторные работы и вопросы для самопроверки.

Методические указания соответствуют требованиям рабочей программы дисциплины «Компьютерное зрение».

Предназначены для студентов всех форм обучения направления подготовки магистров 09.04.04 «Программная инженерия».

Текст печатается в авторской редакции

Подписано в печать 1.02.18 Формат 60×84 1/16.
Усл. печ. л. 3,3 Уч.- изд. л. 3,1 . Тираж 25 экз. Заказ 765 Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Содержание

1 Фильтрация изображений

Цель работы: изучить возможности программного комплекса MatLab для моделирования и исследования методов обработки изображений, приобрести практические навыки его использования.

Теоретическая часть

Изображения, полученные на выходе оптико-электронных устройств, искажены помехами. Это затрудняет визуальный анализ изображений, а также автоматическую компьютерную обработку. При обработке изображений помехами являются и некоторые области самого изображения. Например, при анализе объектов на сложном фоне, фон тоже представляет собой помеху.

При цифровой обработке изображений необходимо устранять геометрические искажения изображений, подавлять шумы различной природы, производить апертурную коррекцию. Ослабление действия помех достигается *фильтрацией*.

Методы фильтрации изображений делятся на две большие категории: методы обработки в пространственной области и методы обработки в частотной области.

При пространственной фильтрации преобразование выполняется непосредственно над значениями отсчетов изображения. Результатом фильтрации является оценка полезного сигнала изображения.

В некоторых случаях признаком полезного сигнала являются резкие перепады яркости. Однако, как правило, частота этих перепадов относительно невелика. В этом случае свойства сигнала проявляются не только в отдельной точке, но и при анализе ее окрестности.

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 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 заключается в прохождении маски по изображению.

Свертка делается также, но маску надо повернуть на 180° перед прохождением по изображению.

Если сдвигаемая маска является симметричной, то корреляция и свертка дают одинаковые результаты.

Методы обработки в частотной области основаны на модификации сигнала, формируемого путем применения к изображению преобразования Фурье.

В программном комплексе MatLab в пакете расширения Image Processing Toolbox имеется большое количество функций фильтрации изображений.

Рассмотрим более подробно некоторые из них.

Функция conv2

При обработке изображений методом свертки предназначена функция conv2. Синтаксис вызова функции следующий:

D = conv2(S, h, 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 за вычетом размера маски.

Функция filter2

Рассмотрим функцию двумерной линейной фильтрации filter2. Обращение к функции имеет вид:

D = filter2(h, S, 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 значение параметра shape есть '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.

Функция medfilt2

При обработке изображений методом медианной фильтрации предназначена функция medfilt2. Синтаксис вызова функции следующий:

ID = medfilt2(IS, [m,n], padopt).

Функция выполняет нелинейную фильтрацию, механизм которой аналогичен линейной фильтрации. 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. Пикселю результирующего изображения ID (r, c), присваивается значение медианы последовательности, где r и c – координаты текущего положения центрального элемента маски.

Параметр padopt определяет три возможные опции расширения границ изображения:

1) опция по умолчанию 'zeros' с нулевым расширением;

2) 'symmetric', при которой изображение IS расширяется путем его зеркального отражения через границы;

3) 'indexed', при которой IS расширяется значением 1, если IS имеет класс double, и значением 0 в противном случае.

Функция fspecial

Функция fspecial позволяет получить маску предопределенного типа. Обращение к функции имеет вид:

h=fspecial (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. Могут быть заданы:

- 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 низких частот Гаусса. По умолчанию размер маски – 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 маску h;

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

4) h = fspecial ('1ap1acian', a) возвращает маску h фильтра Лапласа высоких частот. Размер маски 3 × 3. С помощью фильтра Лапласа можно выполнять улучшение изображения, используя формулу $g(x,y) = f(x,y) + c\nabla^2 f(x,y)$, 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;

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 устанавливаются: размер маски – 5×5 и 0.5 соответственно;

6) h = fspecial ('average', n) возвращает маску h усредняющего низкочастотного 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.

Функция freqz2

Функция freqz2 используется для расчета амплитудно-частотной характеристики (АЧХ) фильтра. Синтаксис вызова функции следующий:

[H,f1,f2] = freqz2(h , n1, n2).

Функция формирует матрицу H размера n1×n2, которая является АЧХ на частотах, содержащихся в векторах f1 и f2 линейного двумерного КИХ-фильтра с маской h.

В векторы f1 и f2 функция помещает равномерно распределенные нормализованные значения частот в диапазоне [–1,1], где 1 соответствует половине частоты дискретизации (частоте Найквиста). Вектор f1 содержит n1 значений частоты, а вектор f2 содержит n2 значений частоты. Если при вызове функции параметры n1 и n2 опущены, то они берутся равными 64.

Функция fsamp2

Функция fsamp2 формирует маску линейного фильтра по желаемой АЧХ. Обращение к функции имеет вид:

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.

Функция ftrans2

Функция ftrans2 используется для формирования маски линейного фильтра методом преобразования частот. Синтаксис вызова функции следующий:

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:

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

В функции ftrans2(b) можно указать другую матрицу трансформации.

Функция nlfilter

Функция nlfilter предназначена для общей нелинейной фильтрации. Обращение к функции имеет вид:

D = nlfilter(S,[m n],fun,P1,P2, …).

Функция используется для обработки бинарных и полутоновых изображений. Функция формирует новое изображение D, пиксели которого являются результатом обработки функцией fun пикселей исходного изображения S, соответствующих положению маски фильтра размера m×n. Операция применяется нерекурсивно для всех положений маски. При проведении вычислений исходное изображение временно дополняется n/2 столбцами нулевых пикселей справа и слева, и m/2 строками нулевых пикселей сверху и снизу. P1,P2 – дополнительные параметры, передаваемые в функцию fun.

Практическая часть

Примеры заданий

Задание 1. Осуществить фильтрацию палитрового изображения, используя маску:

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

```
>> [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)
```

Задание 2. 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,[]);
```

Задание 3. 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 4. Отфильтровать изображение фильтром Собела.

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)
```

Задание 5. Формирование АЧХ 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')
```

Задание 6. 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 фильтра, используя так называемую метрику городских кварталов:

```
>> dist = abs(f1)+abs(f2);
>> H = dist/max(dist(:));
```

3. Вывести на экран АЧХ фильтра:

```
>> 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);
```

Задание 7. Выполнить размытие изображения с целью подавления муара с помощью линейного двумерного фильтра низких частот (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)
```

Задание 8. Создать функцию фильтрации импульсного шума для использования в качестве параметра fun в функции обобщенного

нелинейного фильтра. В функции фильтрация импульсного шума осуществляется операцией усреднения с порогом – центральному пикселю в пределах маски присваивается среднее значение яркости всех пикселей 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, наложить на изображение импульсный шум 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)
```

Задания для самостоятельного выполнения

1. Прочитать палитровое изображение из файла Earth.bmp, вывести на экран, преобразовать в полутоновое. Преобразовать изображение, наложив на него гауссов, импульсный и мультипликативный шум. Вывести на экран зашумленные изображения. Отфильтровать зашумленное изображение медианной фильтрацией и обобщенным нелинейным фильтром. Зашумленное и отфильтрованные изображения вывести в одном окне для сравнения.

2. Осуществить фильтрацию изображения из файла Bigbird.bmp:

а) Отфильтровать изображение, используя маски кругового градиента:

север	северо-восток	восток	юго-восток
$\begin{matrix}1 & 1 & 1\\ 1 & -2 & 1\\ -1 & -1 & -1\end{matrix}$	$\begin{matrix}1 & 1 & 1\\ -1 & -2 & 1\\ -1 & -1 & 1\end{matrix}$	$\begin{matrix}-1 & 1 & 1\\ -1 & -2 & 1\\ -1 & 1 & 1\end{matrix}$	$\begin{matrix}-1 & 1 & 1\\ -1 & -2 & 1\\ 1 & 1 & 1\end{matrix}$
юг	**юго-запад**	**запад**	**северо-запад**
$\begin{matrix}-1 & -1 & -1\\ 1 & -2 & 1\\ 1 & 1 & 1\end{matrix}$	$\begin{matrix}1 & -1 & -1\\ 1 & -2 & 1\\ 1 & 1 & 1\end{matrix}$	$\begin{matrix}1 & 1 & -1\\ 1 & -2 & -1\\ 1 & 1 & -1\end{matrix}$	$\begin{matrix}1 & 1 & -1\\ 1 & -2 & -1\\ 1 & 1 & -1\end{matrix}$

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 фильтров Лапласа:

$$\begin{matrix}0 & 1 & 0\\ 1 & -4 & 1\\ 0 & 1 & 0\end{matrix} \qquad \begin{matrix}-1 & -1 & -1\\ -1 & 8 & -1\\ -1 & -1 & -1\end{matrix} \qquad \begin{matrix}1 & -2 & 1\\ -2 & 4 & -2\\ 1 & -2 & 1\end{matrix}$$

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.

Вопросы для самопроверки

1. Типы фильтров, для которых функция fspecial позволяет получить маску?

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. 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?

Литература

1. Апальков, В.В. Основы моделирования цифровой обработки сигналов в среде MATLAB [Текст]: учебное пособие / В.В. Апальков, Р.А. Томакова, Н.Н. Епишев. – Курск, Юго-Зап. гос. ун-т. – 2015. –137с.

2. Гонсалес, Р. Цифровая обработка изображений в среде Matlab [Текст] / Р. Гонсалес, Р. Вудс, С. Эддинс. – М.: Техносфера, 2006. – 616 с.

3. Дашковская, Т.В. Цифровая обработка сигналов в среде MATLAB [Текст]: учебное пособие / Т.В. Дашковская. – Новосибирск, СГАА, 2010. – 83с.

4.Журавель, И.М. Краткий курс теории обработки изображений [Электронный ресурс] / И.М. Журавель. – Режим доступа: http://matlab. exponenta.ru/imageprocess/book2/index.php.

5. Томакова, Р.А. Метод обработки и анализа сложноструктурируемых изображений на основе встроенных функций среды MATLAB / Р.А. Томакова, С.А. Филист // Вестник Читинского государственного университета. – 2012. – № 1 (80). – С. 3–9.

6. Томакова, Р.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2012. – 222 с.

7. Томакова, Р.А. Теоретические основы и методы обработки и анализа микроскопических изображений биоматериалов [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2011. – 214с.

2 Геометрические преобразования изображений

Цель работы: изучить возможности программного комплекса MatLab для геометрических преобразований изображений, приобрести практические навыки его использования.

Теоретическая часть

Геометрические преобразования изменяют пространственное местоположение элементов в изображении. Задача геометрических преобразований сводится к тому, чтобы по заданному изображению определить значение сигнала в каждой точке изображения в новых координатах.

В программном комплексе MatLab в пакете расширения Image Processing Toolbox имеется большое количество функций геометрических преобразований изображений.

Рассмотрим более подробно некоторые из них.

Функция imcrop

Для вырезания фрагмента из изображения предназначена функция imcrop. Синтаксис вызова функции следующий:

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

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

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

Xd = imcrop (Xs,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:

[Xmin Ymin w,h], где Xmin и Ymin – координаты верхнего левого угла прямоугольника, w – его ширина, h – высота.

Функция imresize

Для изменения размеров изображения используется функция imresize. Обращение к функции имеет вид:

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 применяется один из методов интерполяции, который задается во входном параметре method в виде одной из следующих строк:

'nearest' – использует значение ближайшего пикселя (установлено по умолчанию);

'bilinear' – употребляет интерполяцию по билинейной поверхности;

'bicubic' – применяет интерполяцию по бикубической поверхности.

Функция imrotate

Для поворота изображения предназначена функция imrotate. Синтаксис вызова функции следующий:

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.

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

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

Такой формулой можно задать *сжатие, поворот, перенос* или *сдвиг*, соответствующим образом определяя элементы матрицы Т.

В таблице 1 показано, как выбирать эти величины для совершения различных преобразований.

Таблица 1 – Типы аффинных преобразований

Тип	Аффинная матрица Т	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	$\begin{vmatrix} \cos\alpha & \sin\lambda & 0 \\ -\sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{vmatrix}$	$x = w\cos\alpha - z\sin\alpha$ $y = w\sin\alpha + z\cos\alpha$
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 Image Processing Toolbox пространственное преобразование задается в виде так называемой tform-структуры.

Для задания структуры можно использовать функцию **maketform**. Обращение к функции имеет вид:

tform = maketform(transftype, T),

где transftype – тип преобразования (строковая константа),

T – матрица задания аффинного преобразования.

В таблице 2 приведены значения типов преобразования transftype.

Таблица 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 вызова функции следующий:

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 находящихся вне исходного изображения:

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

Для демонстрации преобразований часто применяется изображение шахматной доски, которое создается функцией **checkerboard**.

Обращение к функции имеет вид:

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 не указаны, то создается квадратная доска размерностью 8×8.

Изменение размеров изображения можно осуществить, непосредственно задавая необходимую индексацию массива изображения:

- вырезание фрагмента – Ic = I (Y:Ym, X:Xn,);
- зеркальное отражение изображения по вертикали – Iy = I(end: –1:1, :);
- зеркальное отражение изображения по горизонтали – Iy = I(:,end: –1:1);
- "прореживание" изображения – Id = I(1:2:end, l:2:end).

Практическая часть

Примеры заданий

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)
```

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

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,'bicuic'),map);
```

Задание 6. Выполнить аффинные преобразования, в качестве тестового изображения выбрав шахматную доску. Для этого необходимо выполнить следующее:

1. Создать М-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)
```

Задание 7. Выполнить зеркальное отражение изображения по горизонтали 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)
```

Примеры выполнения заданий

1. Растянуть изображение по горизонтали и по вертикали в (Sx, Sy) раз.

```
function affintr(I,T,type)
tform=maketform('affine',T);
I1 = imtransform(I, tform);
figure, imshow(I1)
title(type)
>> T=[4 0 0;0 2 0;0 0 1];
type='resize';
affintr(A,T,type);
```

Результат представлен на рисунке 1.

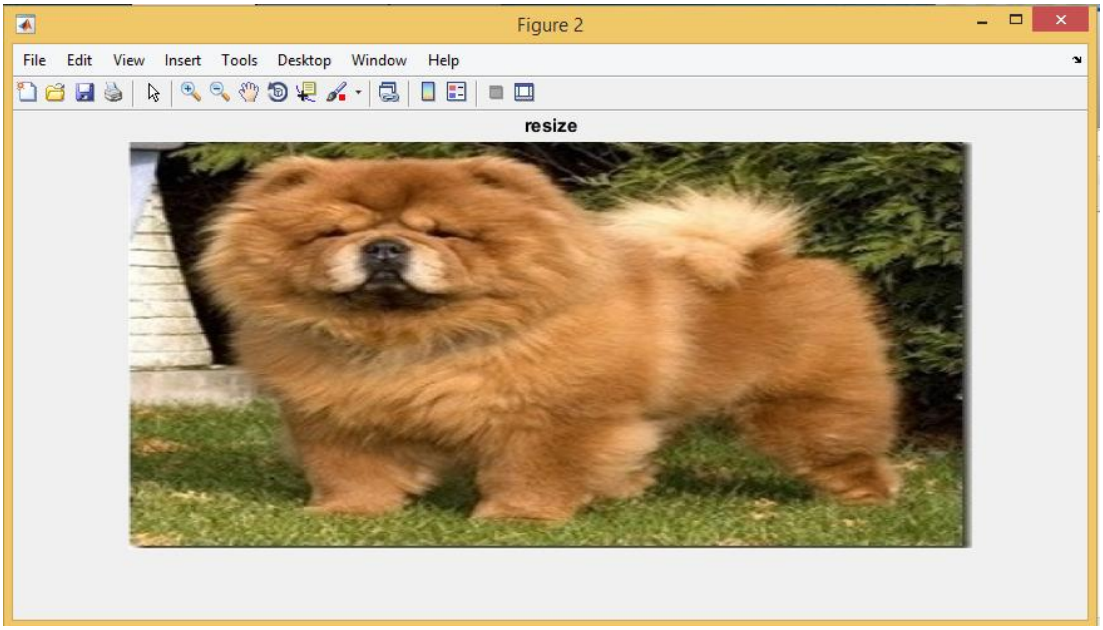

Рисунок 1 – Растянутое изображение

2. Скомбинировать сдвиг изображения по горизонтали и по вертикали на (a, b).

```
function DoublSdv(A,T1,T2)
tform=maketform('affine',T1);
A1 = imtransform(A, tform) ;
tform=maketform('affine',T2);
A2=imtransform(A1, tform);
figure, imshow(A2)
title('DoubleSdvig');
```

```
>> DoublSdv(A,[1 0 0; 0.3 1 0; 0 0 1],[1 0.5 0; 0 1 0; 0 0 1]);
```

Результат представлен на рисунке 2.

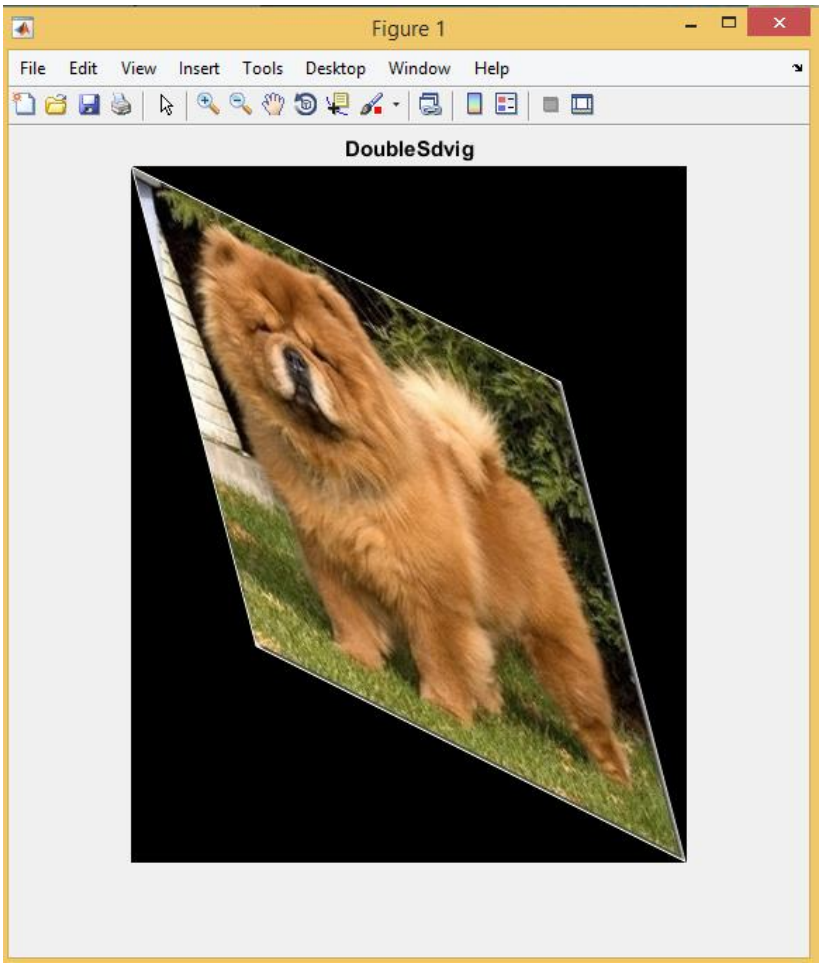

Рисунок 2 – Сдвиг изображения по горизонтали и по вертикали

3. Вывести на экран результаты преобразования в одно окно.

```
function AllTrans(A)
tform=maketform('affine',[4 0 0;0 2 0;0 0 1]);
A1 = imtransform(A, tform);
figure ,subplot (3,1,1), subimage(A1)
title('Stretch');
tform=maketform('affine',[1 0 0; 0.3 1 0; 0 0 1]);
A11 = imtransform(A1, tform);
tform=maketform('affine',[1 0.5 0; 0 1 0; 0 0 1]);
A2=imtransform(A11, tform);
subplot (3,1,2), subimage(A2)
title('DoubleSdvig');
```

```
tform=maketform('affine',[1 0 0; 0 1 0; 50 50 1]);
A3 = imtransform(A, tform, 'XData', [1 320], 'YData', [1 320], 'FillValue', 0.5);
subplot (3,1,3), subimage(A3)
title('Replace');
>> AllTrans(A);
```

Результат представлен на рисунке 3.

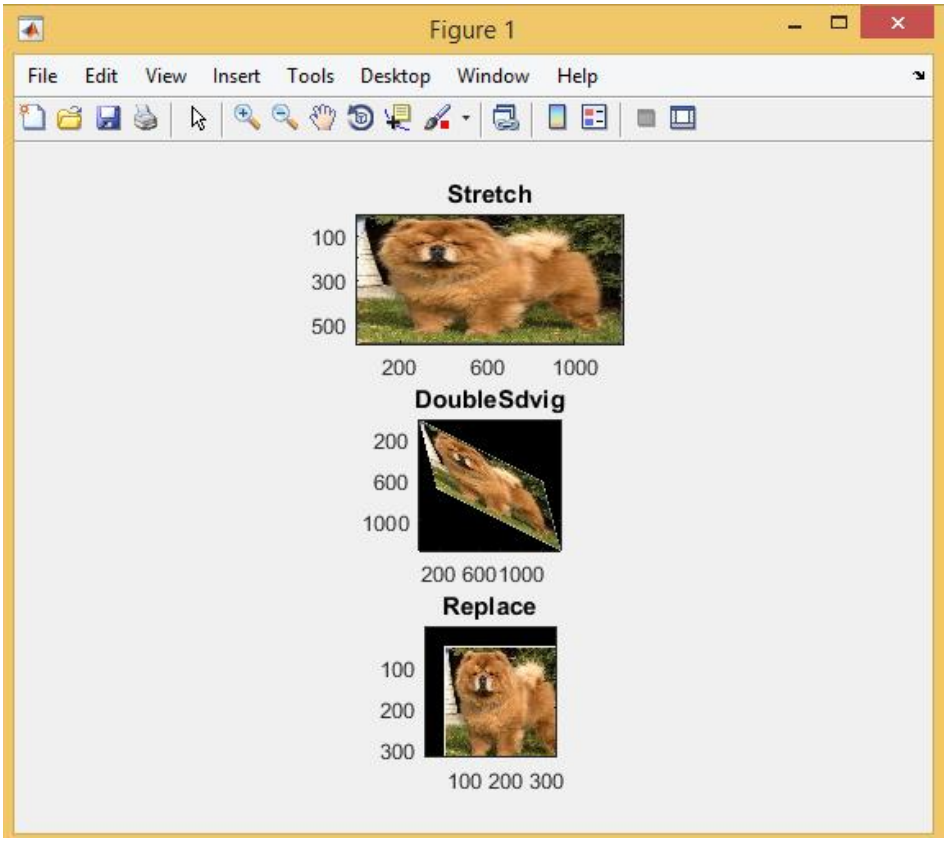

Рисунок 3 – Результаты преобразования

Задания для самостоятельного выполнения

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 методы:

a) 'nearest'; b) 'bilinear'; c) 'bicubic'.

7. Увеличить изображение в (N) раз, используя методы:

a) 'nearest'; b) 'bilinear'; c) 'bicubic'.

8. Повернуть изображение на заданный угол по часовой стрелке (угол), используя методы:

a) 'nearest'; b) 'bilinear'; c) 'bicubic'.

9. Повернуть изображение на заданный угол против часовой стрелки (угол), используя методы:

a) 'nearest'; b) 'bilinear'; c) 'bicubic'.

10. Уменьшить изображение в (N) раз, используя методы:

a) 'nearest'; b) 'bilinear'; c) 'bicubic'.

11. Растянуть изображение по горизонтали и по вертикали в (Sx, Sy) раз.

12. Скомбинировать сдвиг изображения по горизонтали и по вертикали на (a, b) .

13. Перенести изображение по горизонтали и по вертикали на (Sx, Sy)

14. Вывести на экран исходное изображение.

15. Вывести на экран результаты преобразования в разные окна.

16. Вывести на экран результаты преобразования в одно окно.

Вопросы для самопроверки

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 осуществляется операция «прореживания» изображения?

8. Как можно выполнить комбинированный сдвиг изображения по горизонтали и по вертикали на (a, b)?

9. Как осуществляется операция растяжения изображения по горизонтали и по вертикали в (Sx, Sy) раз.

10. Как выполняется изменение размеров изображения, непосредственно задавая необходимую индексацию массива?

Литература

1. Гонсалес, Р. Цифровая обработка изображений в среде Matlab [Текст] / Р. Гонсалес, Р. Вудс, С. Эддинс. – М.: Техносфера, 2006. – 616 с.

2. Дашковская, Т.В. Цифровая обработка сигналов в среде MATLAB [Текст]: учебное пособие / Т.В. Дашковская. – Новосибирск, СГАА, 2010. – 83с.

3. Журавель, И.М. Краткий курс теории обработки изображений [Электронный ресурс] / И.М. Журавель. – Режим доступа: http://matlab. exponenta.ru/imageprocess/book2/index.php.

4. Томакова, Р.А. Метод обработки и анализа сложноструктурируемых изображений на основе встроенных функций среды MATLAB / Р.А. Томакова, С.А. Филист // Вестник Читинского государственного университета. – 2012. – № 1 (80). – С. 3–9.

5. Томакова, Р.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2012. – 222 с.

6. Томакова, Р.А. Теоретические основы и методы обработки и анализа микроскопических изображений биоматериалов [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2011. – 214с.

6. Томакова Р.А., Емельянов С.Г., Филист С.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений/ Юго-Зап. гос. ун-т. – Курск, 2012. – 222 с.

10. Фисенко, В.Т. Компьютерная обработка и распознавание изображений / В.Т. Фисенко, Т.Ю. Фисенко. – СПб.: СПбГУ ИТМО, 2008. – 192 с.

3 Морфологические операции над бинарными изображениями

Цель работы: изучить возможности программного комплекса MatLab для бинаризации изображений и реализации морфологических операций, приобрести практические навыки его использования.

Теоретическая часть

Бинаризация изображений

Анализ бинарных изображений включает операции, применяемые в бинарном машинном зрении, например, при анализе частиц, при обработке документов, и при анализе полутоновых изображений. Он предполагает обработку бинарных изображений, яркости элементов которых имеют значения 0 или 1, при этом 0 представляет фон, 1 представляет объект.

По признаку цветности из изображения выделяются объекты заданного цвета. В результате сегментации формируется бинарное изображение, на котором наряду с полезной информацией остаются области шума. В результате селекции связных областей каждой связной области назначается определенный уровень яркости (номер области). После выполнения селекции и выделения областей в некотором заданном диапазоне площадей удается отфильтровать шум и сосчитать клетки заданного цвета, пронумеровав их.

Первая операция бинарного анализа состоит в выделении объектов на фоне из множества других объектов. Только после этого выполняется объединение элементов изображения каждого объекта. После получения бинарного изображения объекта возможно производить вычисление признаков объекта.

К основным операциям бинарного анализа относится операция «*выделение признаков»*, по которым можно выделить объект на фоне. В связи с этим перед бинарными операциями выполняются полутоновые преобразования, позволяющие усилить признаки объекта.

Следующая задача связана с определением порогов для выполнения бинаризации.

Пороговая бинаризация полутоновых изображений в соответствии с амплитудными характеристиками, позволяет получить бинарное изображение объекта.

Пусть f_{ij} – полутоновое изображение, T – порог и b_0 и b_1 – два бинарных значения. Результатом порогового разделения является бинарное изображение, полученное следующим образом:

$$f_{ij} = \begin{cases} b_0, & \text{åñëè } f_{ij} \le T, \\ b_1, & \text{åñëè } f_{ij} > T. \end{cases}$$

Основной задачей является выбор порогового 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 применяться одно пороговое значение для всего изображения. Использование единственного порогового значения для всех пикселей называется *глобальным пороговым значением*.

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 > threshold), получается *позитив*;

«меньше выбранного уровня» (I < threshold) – *негатив*.

Существует много способов выбора порогового значения, например, разделом двух основных пиков на гистограмме яркости, усреднением функции яркости и др.

Для автоматического выбора порога предлагается следующий алгоритм:

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: G_1 и G_2. Область G_1 состоит из пикселей, яркость которых больше или равна T, а область G_2 – из пикселей, яркость которых меньше T.

3. Вычислить среднюю яркость пикселей S_1 и S_2 по областям G_1 и G_2.

4. Вычислить новое значение порога $T = \frac{1}{2(S_1 + S_2)}$.

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

Для вычисления порога также можно использовать функцию **graythresh** пакета расширения Image Processing Toolbox программного комплекса MatLab, которая вычисляет порог по *методу Отса*. Синтаксис вызова функции следующий:

T = graythresh(S),

где S – исходное полутоновое изображение,

T – глобальный порог, значения которого принадлежат интервалу [0 1].

Бинаризация изображения отсечением по порогу яркости выполняется с использованием функции **im2bw**. Обращение к функции имеет вид:

BW = im2bw(I, threshold),
BW = im2bw(X, map, threshold),
BW = im2bw(RGB, threshold),

где I – полутоновое изображение,
threshold – порог яркости,
X – палитровое изображение,
map – палитра,
BW – бинарное изображение.

Функция im2bw создает бинарное изображение, используя отсечение по порогу яркости threshold. Для этой цели функция конвертирует полноцветные и палитровые изображения в полутоновые. Пороговое значение threshold должно задаваться из диапазона [0,1]. По умолчанию threshold = 0,5. Для вычисления порога можно воспользоваться функцией graythresh.

Для многих изображений глобальное пороговое значение не может применяться из-за неоднородностей внутри областей фона и объекта. Для таких изображений требуются различные пороговые значения для различных частей изображения. Использование различ-

ных пороговых значений для различных частей изображения называется *аддитивным* или *локальным пороговым разделением*.

Методика локального порогового разделения основана на разбиении 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.

Основные морфологические операции

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.

Пиксель p с координатами (x, y) имеет четыре горизонтальных и вертикальных соседних пикселя 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 $N_4(p)$.

Данные четыре пикселя 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 пикселя 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 пикселя 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.

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

Объект называется *восьмисвязным***,** если для каждого пикселя 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 ("дыры").

Морфологические операции (их также называют морфологическими фильтрами) представляют собой нелинейный способ обработки изображений. Их применяют с целью изменения формы объектов.

В математической морфологии используется язык теории множеств. Изображения представляются множествами, элементами которых являются пиксели с атрибутами координат и яркости или координат и R, G, B – кодов.

Над бинарными изображениями можно совершать операции как над множествами с помощью следующих логических операций системы MatLab:

OR ($|$) – логическое сложение, аналог – объединение множеств $A \cup B$;

AND (&) – логическое умножение, аналог – пересечение множеств $A \cup B$;

NOT (~) – отрицание, аналог – дополнение множества A^C;

DIFFERENCE ($A \& \widetilde{B}$), аналог – разность множеств $A \setminus B$.

Логические операции MatLab, применяемые к бинарным изображениям, приведены в таблице 1.

Таблица 1 – Логические операции Matlab для бинарных изображений

Операции	Выражение Matlab для бинарных выражений	Имя
$A \cup B$	A | B	OR
$A \cap B$	A&B	AND

Продолжение таблицы 1

A^C	$\widetilde{A}$	NOT
$A \setminus B$	$A \,\&\, \widetilde{B}$	DIFFERENCE

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 strel

Функция strel строит структурообразующие элементы различных форм и размеров. Синтаксис вызова функции следующий:

se = strel(shape, parameters),

где shape – строка, задающая форму структурообразующего элемента,

parameters – дополнительные параметры, которые уточняют информацию о его форме,

se – структурообразующий элемент.

В таблице 2 приведены некоторые варианты задания формы структурообразующего элемента.

Таблица 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

Продолжение таблицы 2

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.** Обращение к функции имеет вид:

seq=getsequence(se),

где параметр se является массивом структурных элементов, функция возвращает еще один массив структурных элементов seq, содержащий отдельные структурные элементы, которые являются разложением se.

Операции *дилатации* и *эрозии* имеют основополагающее значение при морфологической обработке бинарных изображений. Способ и степень этих преобразований контролируется формой структурного элемента.

Для обеих этих операций задается структурный элемент, который называют маской морфологического фильтра. Ненулевые значе-

ния в маске определяют, какие из соседних пикселей следует учитывать при осуществлении операции.

Дилатация (морфологическое расширение) – свертка изображения или выделенной области изображения с маской. При этом в маске выделяется единственная ведущая позиция, которая совмещается с текущим пикселем при вычислении свертки. Операция дилатации сводится к проходу шаблоном по всему изображению и применению оператора поиска локального максимума к интенсивностям пикселей изображения, которые накрываются шаблоном. Такая операция вызывает рост светлых областей на изображении.

Эрозия (морфологическое сужение) – обратная операция. Действие эрозии подобно дилатации, разница лишь в том, что используется оператор поиска локального минимума.

Для выполнения операций дилатации и эрозии используются функции **imdilate** и **imerode** соответственно. Синтаксис вызова функций следующий:

D = imdilate (S, se),

D = imerode(S, se),

где S – двоичное или полутоновое изображение,

se – матрица из нулей и единиц, которая определяет структурообразующий элемент.

Эрозия и дилатация – операции, предназначенные в первую очередь для выявления различных морфологических особенностей изображения с использованием различных структурных элементов. Так, эрозия посредством круга радиуса 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.

Операции *замыкание* и *размыкание* являются комбинированием дилатации и эрозии.

Результатом морфологического размыкания является дилатация, которая применяется к эрозии исходного изображения. Операция эрозии позволяет удалить все мелкие объекты и шум на изобра-

жении, но ее применение приводит к значительному уменьшению размеров оставшихся объектов. Чтобы увеличить размер объектов, выделенных с помощью эрозии, достаточно применить дилатацию.

Морфологическое замыкание – операция обратная размыканию. Операция замыкания позволяет удалить небольшие внутренние "дыры" и убрать зернистость по краям области. "Дыры" удаляются за счет начального применения дилатации, но дилатация приводит к росту границы. Последующее применение эрозии обеспечивает обратное уменьшение границы.

Для выполнения операций замыкания и размыкания используются функции **imopen** и **imclose** соответственно. Обращение к функциям имеет вид:

D = imopen(S, se),

D = imclose(S, se),

где S – двоичное или полутоновое изображение,

se – матрица из нулей и единиц, которая определяет структурообразующий элемент.

Практическая часть

Примеры заданий

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)
```

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

```
>> rgb = imread('c:\Image\bike.jpg');
>> 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 = I < T;
>> figure,imshow(BW)
```

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)
```

Задание 5. 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)
```

Задание 6. 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 проводов.

```
>> [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)
```

Задание 7. 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.jpg');
>> 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')
```

Задание 8. 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)
```

Задания для самостоятельного выполнения

1. Преобразовать палитровое изображение в бинарное с использованием автоматического выбора порогового значения.

2. Преобразовать цветное изображение в бинарное, используя функцию graythresh.

3. Выполнить бинаризацию палитрового изображения с выбором порога по умолчанию и вычислить порог с помощью функции graythresh.

4. Выполнить дилатацию объекта на изображении, изменяя тип структурообразующего элемента, руководствуясь данными из таблицы 2.

5. Выполнить эрозию объекта на изображения, изменяя тип структурообразующего элемента, руководствуясь данными из таблицы 2.

6. Выделить объекты на изображении файла, используя операции замыкания и размыкания.

Вопросы для самопроверки

1. Как можно получить бинарное изображение?

2. Чем отличаются глобальное и локальное пороговое разделение?

3. Какие функции используются для бинаризации изображения в системе MatLab?

4. В чем заключается алгоритм автоматического выбора порога?

5. Как определяются координаты 4-х соседних пикселей изображения?

6. Как определяются 4-х связный объект?

7. Какие морфологические операции вы знаете?

8. Дайте определение понятия дилатации.

9. Дайте определение понятия эрозии.

10. В чем заключается действие операции размыкание?

11. В чем заключается действие операции замыкание?

Литература

1. Гонсалес, Р. Цифровая обработка изображений в среде Matlab [Текст] / Р. Гонсалес, Р. Вудс, С. Эддинс. – М.: Техносфера, 2006. – 616 с.

2. Дашковская, Т.В. Цифровая обработка сигналов в среде MATLAB [Текст]: учебное пособие / Т.В. Дашковская. – Новосибирск, СГАА, 2010. – 83с.

3. Журавель, И.М. Краткий курс теории обработки изображений [Электронный ресурс] / И.М. Журавель. – Режим доступа: http://matlab. exponenta.ru/imageprocess/book2/index.php.

4. Томакова, Р.А. Метод обработки и анализа сложноструктурируемых изображений на основе встроенных функций среды MATLAB / Р.А. Томакова, С.А. Филист // Вестник Читинского государственного университета. – 2012. – № 1 (80). – С. 3–9.

5. Томакова, Р.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2012. – 222 с.

6. Томакова, Р.А. Теоретические основы и методы обработки и анализа микроскопических изображений биоматериалов [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2011. – 214с.

6. Томакова Р.А., Емельянов С.Г., Филист С.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений/ Юго-Зап. гос. ун-т. – Курск, 2012. – 222 с.

10. Фисенко, В.Т. Компьютерная обработка и распознавание изображений / В.Т. Фисенко, Т.Ю. Фисенко. – СПб.: СПбГУ ИТМО, 2008. – 192 с.

1 Методы сегментации изображений

Цель работы: изучить возможности программного комплекса MatLab для моделирования и исследования методов сегментации изображений, приобрести практические навыки его использования.

Теоретическая часть

Сегментация изображения представляет собой разделение изображения на несколько областей (сегментов, т.е. формируется множество пикселей, которые также называют суперпикселями) по сходству свойств (признаков) в их точках.

Цель сегментации заключается в упрощении и/или изменении изображения, так чтобы его было проще и легче анализировать. Сегментация изображений обычно используется для того, чтобы *выделить объекты и границы* (линии, кривые, и т. д.) на изображениях. Более точно, сегментация изображений – это процесс присвоения меток каждому пикселю изображения, так что пиксели с одинаковыми метками имеют общие визуальные характеристики.

Если R – область изображения, сегментация – это процесс разбиения этой области на R_1, R_2, K, R_n областей, удовлетворяющих условиям:

1) $\bigcup_{i=1}^{n} R_i = R$;

2) $\forall R_i, i = \overline{1,n}$ – связные области;

3) $R_i \cap R_j = \varnothing, \ \forall i, j, \quad i \neq j$;

Признаки подразделяются на *естественные* и *искусственны*е. 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*.

Сегментация – весьма трудный и до конца не алгоритмизированный процесс для произвольных изображений. Наиболее распространены методы сегментации, основанные на определении *однородных яркостей* (цветов) или *однородностей типа текстур.*

Методы сегментации можно разделить на два класса: *автоматические* – не требующие взаимодействия с пользователем и *интерактивные* – использующие пользовательский ввод непосредственно в процессе работы. Для анализа биомедицинских материалов более интересны автоматические методы сегментации.

Задачи автоматической сегментации делятся на два класса:

1) выделение областей изображения с известными свойствами;

2) разбиение изображения на однородные области.

В первом случае задача сегментации состоит в поиске определенных областей, о которых имеется априорная информация (например, мы знаем цвет, форму областей, или интересующие нас области представляют собой изображения известного объекта). Методы этой группы узко специализированы для каждой конкретной задачи. Сегментация в такой постановке используется в основном в задачах машинного зрения (например, анализ сцен, поиск объектов на изображении).

Во втором случае априорная информация о свойствах областей не используется, но на само разбиение изображения накладываются некоторые условия (например, все области должны быть однородны по цвету и текстуре). Так как при такой постановке задачи сегментации не используется априорная информация об изображенных объектах, то методы этой группы универсальны и применимы к любым изображениям. В основном сегментация в такой постановке применяется на начальном этапе решения задачи, для того чтобы получить представление изображения в более удобном виде для дальнейшей работы.

На результаты сегментации очень сильно влияет как выбор критерия однородности, так и способ построения однородных регионов. Далеко не всегда для изображения есть единственно «правильная» сегментация, и далеко не всегда задача сегментации имеет единственное решение. По той же причине нет и объективного критерия оценки качества разбиения изображения.

Для грубой оценки качества метода в конкретной задаче обычно фиксируют несколько свойств, которыми должна обладать хорошая сегментация. Качество работы метода оценивается в зависимости от того, насколько полученная сегментация обладает этими свойствами. Наиболее часто используются следующие свойства:

1. Однородность регионов (однородность цвета или текстуры).
2. Непохожесть соседних регионов.
3. Гладкость границы региона.
4. Маленькое количество мелких «дырок» внутри региона и т.д.

Для сегментации изображения применяется *метод выращивания областей*, основная идея которого заключается в группировании пикселей изображения в более крупные области по заранее заданным критериям роста. Определяются «*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).

Сегментация изображений в среде MatLab позволяет осуществить различные методы обработки изображений. Например, применение функции **regiongrow** позволяет выполнять выращивание областей. Синтаксис вызова функции следующий:

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

где f – сегментируемое изображение,
S – массив (с размерами как у f) или скаляр,
T – 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.

Если T – массив, то его элементы являются локальными пороговыми значениями для f. Скаляр T определяет глобальный порог.

Если S=a и T=b при сравнении яркости считаем, что пиксель похож на пиксель со значением a, если модуль разности их значений не больше b.

Выходом g является сегментированное изображение, причем элементы каждой области помечены одним и тем же целым числом.

Параметр NR равен числу выделенных областей.

SI – изображение, на котором отмечены центры кристаллизации.

TI – изображение, содержащее пиксели, связной области.

SI, TI – изображения с размерами как у f.

Метод разделения используется для сегментации изображений различных видов.

В основу метода положена идея разбиения изображения (или выделенного фрагмента) на непересекающиеся блоки, которые с помощью введенного критерия проверяются на однородность. Функция **qtdecomp** осуществляет сегментацию полутоновых изображений *методом разделения*. Обращение к функции имеет вид:

A = qtdecomp(I, threshold, mindim),

где I – полутоновое изображение,
threshold – скаляр,
mindim – минимальный размер блока,
A – разреженный массив.

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 числа 2.

Функция qtdecomp осуществляет сегментацию полутонового изображения I методом разделения и помещает результат в разреженный массив A (тип данных sparse системы MatLab). Элементам матрицы A, соответствующим координатам левых верхних углов блоков на исходном изображении I, присваиваются значения, определяющие размеры каждого блока.

В качестве критерия однородности выбирается критерий, основанный на утверждении: *блок считается однородным, если разница между максимальным и минимальным значением пикселей блока меньше параметра threshold*.

Функция **qtgetblk** возвращает в массив vals все блоки размером dim, получившиеся в результате сегментации полутонового изображения I с помощью функции qtdecomp. Синтаксис вызова функции следующий:

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

где A – разреженный массив, описывающий квадро-дерево с результатами сегментации,

idx – вектор координат левых верхних углов блоков, помещенных в массив vals.

Если нет ни одного блока размером dim, то всем возвращаемым параметрам присваиваются значения пустых vals матриц.

Функция **qtsetblk** создает новое полутоновое изображение ID, заменяя в исходном полутоновом изображении IS все блоки размера dim, полученные в результате сегментации с помощью функции qtdecomp, на блоки из массива vals. Массив vals должен содержать столько же блоков размера dim, сколько их хранится в разреженном массиве A, описывающем квадро-дерево с результатами сегментации. Обращение к функции имеет вид:

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

Функция **roicolor** позволяет выбрать интересующую область изображения по цвету. Синтаксис вызова функции следующий:

BW = roicolor(S, low, high),

BW = roicolor(S, v).

Для любого варианта вызова функции roicolor бинарное изображение формируется по следующему алгоритму:

пикселю бинарного изображения BW(r, c) присваивается значение 1, если яркость пикселя S(r, c) исходного полутонового изображения или индекс S(r, c) палитрового изображения принадлежат диапазону [low, high] или любому из значений вектора v. В противном случае BW(r, c) присваивается значение 0.

Функция **impixel** возвращает значения красной (R), зеленой (G) и синей (B) составляющих цвета для определенных пикселей изображения. Обращение к функции имеет вид:

P = impixel(S, c, r),

где S – полноцветное изображение,

c и r – векторы значений строк и столбцов требуемых пикселей.

При этом в i-той строке матрицы P будут находиться R-, G-, B-составляющие пикселя из c(i) столбца и r(i) строки.

Используя функцию impixel для выполнения яркостного среза, можно выделить те участки изображения, где яркость соответствует выделенному интервалу.

Практическая часть

Примеры заданий

Задание 1. Создать функцию **regiongrow**.

```
function [g, NR, SI, TI] = regiongrow(f, S, T)
f=double (f);
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 методом наращивания 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)
```

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, значение 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 квадро-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 пикселов 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)
```

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')
```

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 = impixel(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)
```

Задания для самостоятельного выполнения

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.

Вопросы для самопроверки

1. В чем заключается сегментация изображения?
2. Какие признаки используются для сегментации?
3. В чем заключается алгоритм метода выращивания областей, использующийся для сегментации изображения?
4. В чем заключается алгоритм метода разделения, использующийся для сегментации изображения?
5. Какие параметры являются входными параметрами функции сегментации методом разделения?
6. В чем заключается преобразование яркостного среза?

Литература

1. Гонсалес, Р. Цифровая обработка изображений в среде Matlab [Текст] / Р. Гонсалес, Р. Вудс, С. Эддинс. – М.: Техносфера, 2006. – 616 с.

2. Дашковская, Т.В. Цифровая обработка сигналов в среде MATLAB [Текст]: учебное пособие / Т.В. Дашковская. – Новосибирск, СГАА, 2010. – 83с.

3. Журавель, И.М. Краткий курс теории обработки изображений [Электронный ресурс] / И.М. Журавель. – Режим доступа: http://matlab. exponenta.ru/imageprocess/book2/index.php.

4. Томакова, Р.А. Метод обработки и анализа сложноструктурируемых изображений на основе встроенных функций среды MATLAB / Р.А. Томакова, С.А. Филист // Вестник Читинского государственного университета. – 2012. – № 1 (80). – С. 3–9.

5. Томакова, Р.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений [Текст]: монография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2012. – 222 с.

6. Томакова, Р.А. Теоретические основы и методы обработки и анализа микроскопических изображений биоматериалов [Текст]: мо-

нография / Р.А. Томакова, С.Г. Емельянов, С.А. Филист. – Курск, Юго-Зап. гос. ун-т. – 2011. – 214с.

6. Томакова Р.А., Емельянов С.Г., Филист С.А. Интеллектуальные технологии сегментации и классификации биомедицинских изображений/ Юго-Зап. гос. ун-т. – Курск, 2012. – 222 с.

10. Фисенко, В.Т. Компьютерная обработка и распознавание изображений / В.Т. Фисенко, Т.Ю. Фисенко. – СПб.: СПбГУ ИТМО, 2008. – 192 с.