Лабораторная работа №2
Бинаризация и математическая морфология

Под *бинаризацией* понимают построение разметки изображения, при котором точки фона получают разметку «0», а точки объектов – разметку «1». Таким образом, разметка отвечает на вопрос: принадлежит ли данная точка объекту (1, true) или не принадлежит (0, false).

Бинаризованное изображение не следует путать с изображением монохромным: разметка монохромного изображения даёт информацию о яркости точки, но ничего не говорит о её принадлежности объекту или фону.

Процесс бинаризации часто также называют *сегментацией* изображения.

Использование бинаризации позволяет существенно ускорить и упростить обработку изображений, поскольку появляется возможность разбить изображение на отдельные объекты и работать с ними независимо друг от друга.

В простейшем случае бинаризация выполняется следующим образом: выбирается некоторая величина – порог бинаризации. Если объект светлее фона, то все точки с яркостью выше порога помечаются единицами, а ниже – нулями (если объект темнее фона, то, соответственно, наоборот).

Обратите внимание, что в любом случае точки фона должны быть помечены нулями, а точки объекта – единицами.

Посмотрим, как работает бинаризация.

Создайте новый скрипт, напишите команду для загрузки изображения *cameraman.tif,* и отобразите загруженное изображение на экране, например:

A=imread('cameraman.tif');

figure,imshow(A),title('исходное изображение');

Постройте гистограмму яркости:

figure,plot(imhist(A));

Гистограмма имеет чётко выраженную моду тёмного объекта и моду светлого фона. Порог бинаризации выберем посередине между этими модами, например – на величине 75.

Для бинаризации изображения можно воспользоваться функцией *im2bw().* Она принимает два аргумента – исходное серое изображение и порог бинаризации. Порог должен быть нормирован в диапазон [0..1].

B=im2bw(A, 75 / 255);

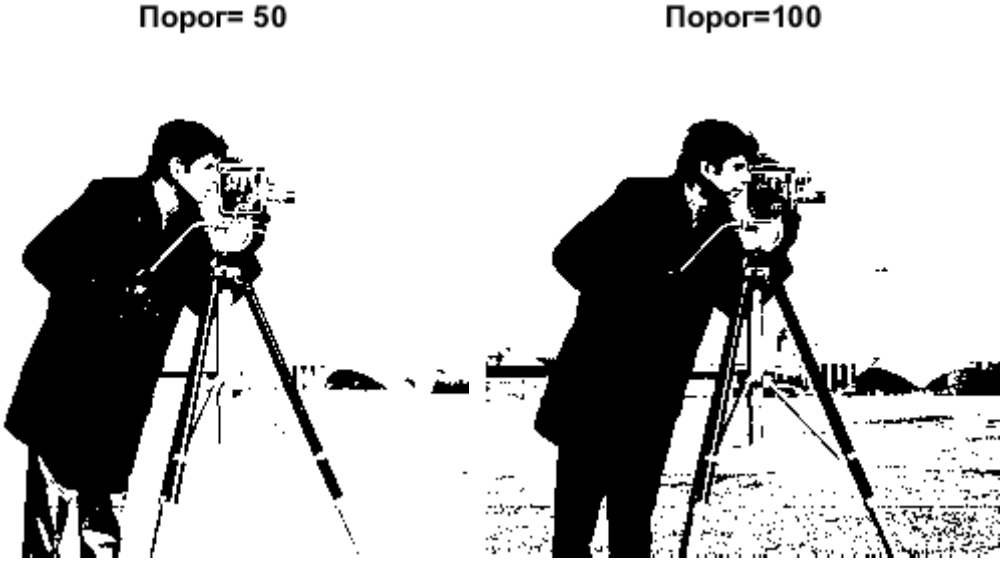
figure,imshow(B),title('результат бинаризации');

Обратите внимание, что объект (фотограф) темнее объекта и его точки получили разметку 0 вместо 1. Чтобы исправить это, картинку необходимо инвертировать, например – функцией *imcomplement():*

B=imcomplement(B);imshow(B), title('результат бинаризации');

Обратите внимание: картинка с белым фотографом воспринимается хуже, чем исходная: мы привыкли видеть объекты более темными на светлом фоне, а не наоборот. Поэтому для автоматической машинной обработки мы будем использовать белые объекты на черном фоне, а для оператора (на экране, в отчете) – наоборот, черные на белом.

На полученном изображении видно, что не все точки получили правильную разметку. Возможно, их удастся исправить более точным выбором порога бинаризации? Выполните бинаризацию с порогами 50 и 100.



Видно, что даже небольшое изменение порога приводит к значительным изменениям разметки. Поэтому точный выбор порога бинаризации играет очень важное значение.

Наиболее широко используемый метод расчёта порога – *метод Otcy.* Метод ищет такой порог, который при разбиении на фон и объект даёт минимальную сумму дисперсий

яркостей отдельно точек фона и точек объекта. Вычислить такой порог можно, используя функцию *graythresh():*

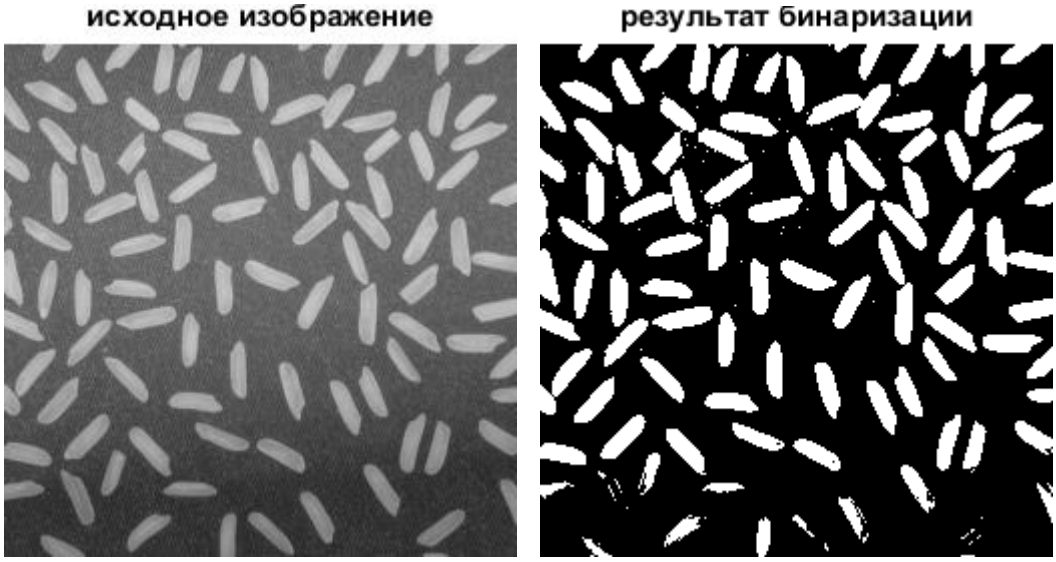
L=graythresh(A);

Функция вернёт значение L=0.345, лежащее в диапазоне [0…1]. Если пересчитать его в диапазон [0…255], то получим значение 88, которое оказывается очень близким к интуитивно выбранному нами по гистограмме значения 75.

Локальная бинаризация

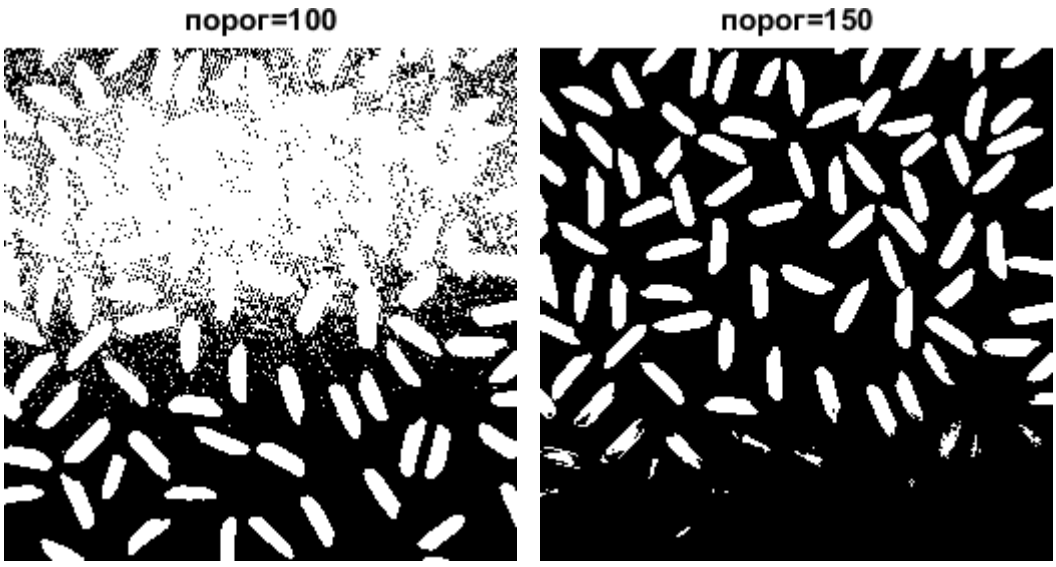
Создайте новый скрипт и загрузите изображение ‘rice.png’. Как нетрудно догадаться, оно содержит изображения зёрен риса на тёмной поверхности. Бинаризуйте изображение и отобразите результат на экране:

A=imread(‘rice.png’*);*
figure,imshow(A),title(‘исходное изображение’);
L=graythresh(A);
B=im2bw(A, L);
figure,imshow(B),title(‘результат бинаризации’);



Обратите внимание, что часть точек фона в центре ошибочно помечено единицами, а часть точек риса внизу – пропало: порог бинаризации выбран неверно.

Попробуйте выполнить бинаризацию с другими порогами. Получилось? Почему?

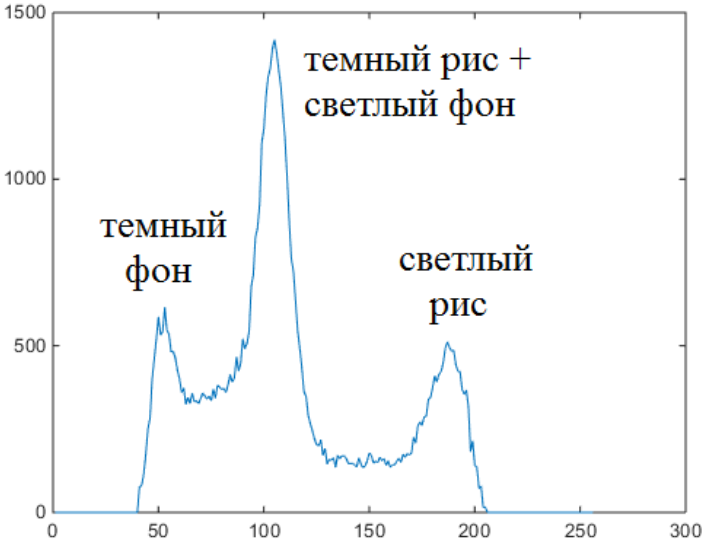


Результаты бинаризации с разными порогами

Выполнить бинаризацию без ошибок не удаётся: освещение верхней и нижней части сильно отличается, и эти части требуют разных порогов бинаризации.

Отобразите гистограмму яркости риса:

figure,plot(imhist(A));



Видно, что гистограмма имеет не две, а три моды: светлый рис, тёмный рис+светлый фон, тёмный фон. При этом выбрать общий порог, который отделил бы все точки риса ото всех точек фона невозможно.

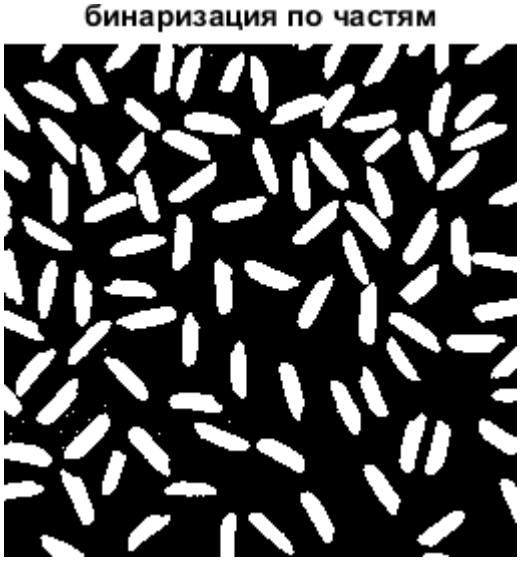
Разобьём картинку на верхнюю и нижнюю части, и бинаризуем их по отдельности, выбрав для каждой части свой порог:

```
A1=A(1:180,:);      %верхняя часть
A2=A(181:end,:);    %нижняя часть
L1=graythresh(A1);  %порог для верхней части
L2=graythresh(A2);  %порог для нижней части
B1=im2bw(A1,L1);    %бинаризация верхней части
B2=im2bw(A2,L2);    %бинаризация нижней части
```

Склеим результат:

```
B=zeros(size(A));   %создаём матрицу результата, куда будем копировать части
B(1:180,:)=B1;      %копируем верхнюю часть
```

B(181:end,:)=B2; %копируем нижнюю часть
figure,imshow(B),title('бинаризация по частям');



Видно, что теперь обе части изображения бинаризованы корректно (за исключением небольшого количества шумовых точек).

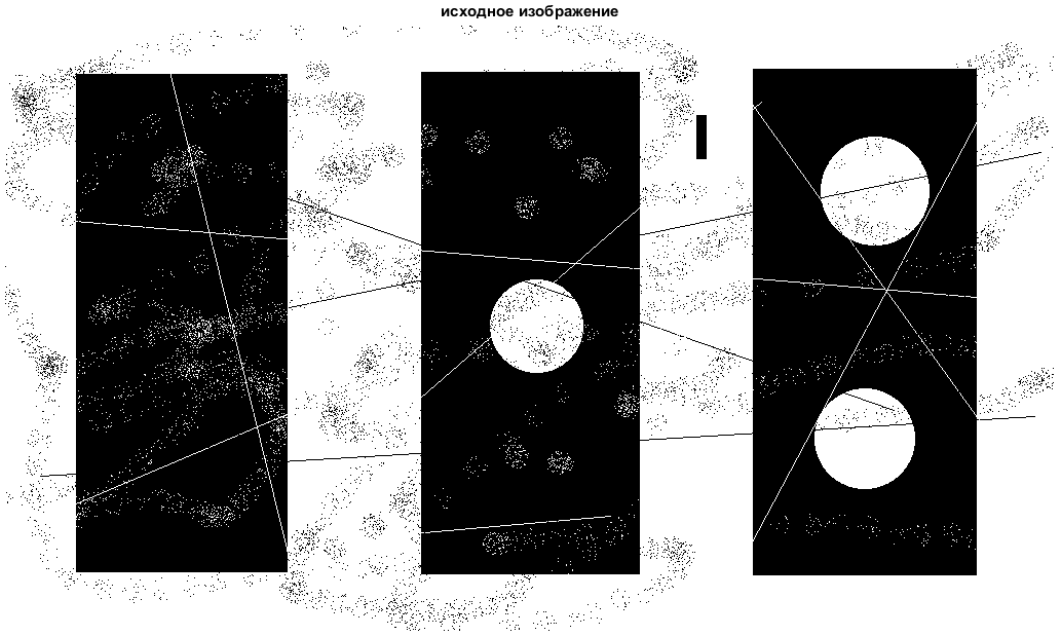
Бинаризация, порог которой отличается для разных областей изображения, называется *локальной*. Локальная бинаризация позволяет нивелировать изменения освещённости в разных областях одного изображения.

Сохраните скрипт с локальной бинаризацией риса. Он нам потребуется дальше, когда будем работать с разметкой изображения.

Математическая морфология

Бинаризованное изображение интересно прежде всего тем, что его можно обрабатывать методами математической морфологии. Эти методы предоставляют широкий класс инструментов обработки, которые способны работать в реальном масштабе времени и хорошо распараллеливаются на современных вычислительных средствах.

Откройте графический редактор mspaint (Win+R → Выполнить → mspaint). Создайте изображение трех тёмных прямоугольных деталей на светлом фоне. Одну деталь сделайте с отверстием по центру, еще одну – с двумя. Добавьте шум – отдельные чёрные и белые точки, линии, кривые, в том числе – соединяющие детали в одну.



Сохраните изображение в папку с вашими скриптами. Создайте новый скрипт, загрузите созданное изображение и отобразите его на экране.

Обратите внимание – редактор mspaint по умолчанию сохраняет изображение как цветное, даже если оно содержит только чёрные и белые точки. В результате загруженная матрица, представляющая изображение, будет трёхмерной. Третье измерение – это номер цветовой плоскости (1–красная, 2–зелёная, 3–синяя). Для дальнейшей работы нужно перейти к серому изображению, с двухмерной матрицей. Наиболее простой способ – взять одну из цветовых плоскостей, если они – одинаковы:

A=A(:,:,1);

Второй вариант – использовать функцию преобразования *rgb2gray().*

Попробуем очистить изображение от шума.

Из предыдущей лабораторной работы мы знаем, что шум легко подавить медианным фильтром (функция *medfilt2()*). Однако он удалит лишь изолированные точки. Шумовые линии и небольшие пятна медианный фильтр убрать не сможет, и нам необходимы другие инструменты.

Бинаризуем изображение. Обратите внимание, что деталь – тёмная на светлом фоне, поэтому разметку нужно инвертировать.

```
A=imread('тут указать имя вашего файла');
A=rgb2gray(A);
A=medfilt2(A);
B=im2bw(A,0.5);
B=imcomplement(B);
figure,imshow(B),title('бинаризованные детали');
```

Эрозия и дилатация

Применим морфологические операции эрозии и дилатации.

Эрозия представляет собой удаление из объектов точек, лежащих на их границах.

Дилатация представляет собой добавление к объекту соседних точек фона.

Для использования эрозии и дилатации необходимо задать матрицу структурного элемента. В маске единицами должны быть помечены те элементы, которые мы считаем соседними для центрального. Так, если соседями считаются только верхний, нижний, левый и правый пиксель (4-соседство), матрица будет иметь форму креста. Мы будем

использовать 8-соседство (диагональные элементы – тоже соседи). Тогда матрица структурного элемента будет квадратной матрицей 3x3 из единичек:

se=ones(3,3);

Для выполнения эрозии служит функция *imerode():*

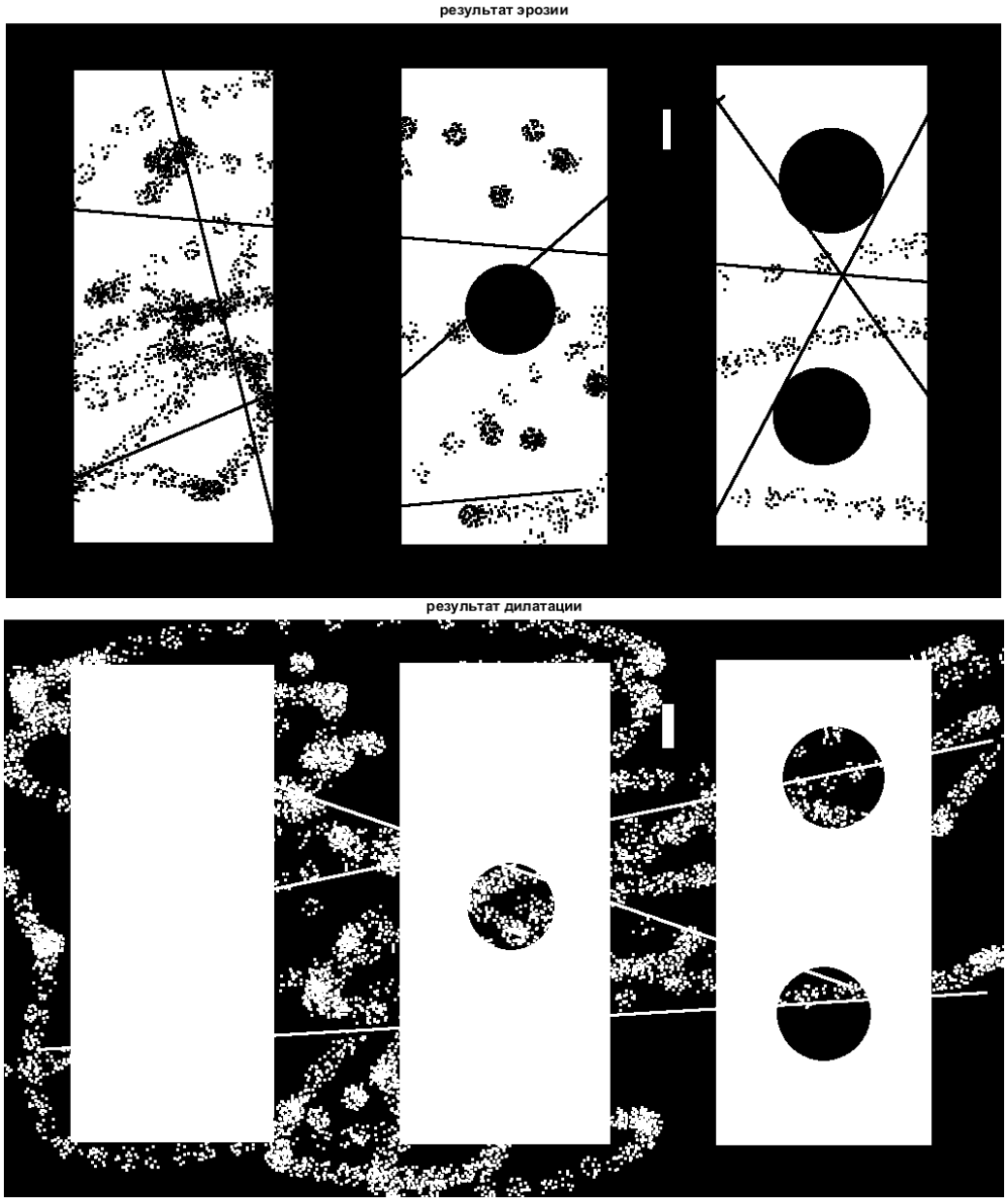
```
C=imerode(B,se);figure,imshow(C),title('результат эрозии');
```

Видно, что эрозия удаляет шумовые точки и линии на фоне. Однако шумовые области на самом объекте не только не уменьшаются, а наоборот, разрастаются. Сам объект уменьшается на единицу с каждой стороны.

Для выполнения дилатации служит функция *imdilate():*

```
C=imdilate(B,se);figure,imshow(C),title('результат дилатации');
```

Видно, что дилатация, наоборот, удаляет шумовые точки и линии на объекте. Шумовые области на фоне разрастаются. Объект увеличивается на единицу с каждой стороны.



Размыкание и замыкание

Дилатация и эрозия очень похожи друг на друга: если инвертировать изображение, то дилатация превратится в эрозию и наоборот. Однако легко заметить, что они не являются взаимно обратными. Если применить сначала эрозию, а потом – дилатацию, исходное изображение мы уже не получим. Если эрозия удалила изолированную точку или точку линии, то дилатация её уже восстановить не сможет. При этом точки крупных объектов останутся без изменений. Это свойство можно использовать для фильтрации мелких шумовых элементов на фоне. При этом такая фильтрация не будет затрагивать сам объект.

Последовательность операций эрозии и дилатации называется операцией *размыкания*. Она размыкает перемычки, связывающие объекты, а также удаляет шумовые точки фона. Точки объекта при этом не меняются.

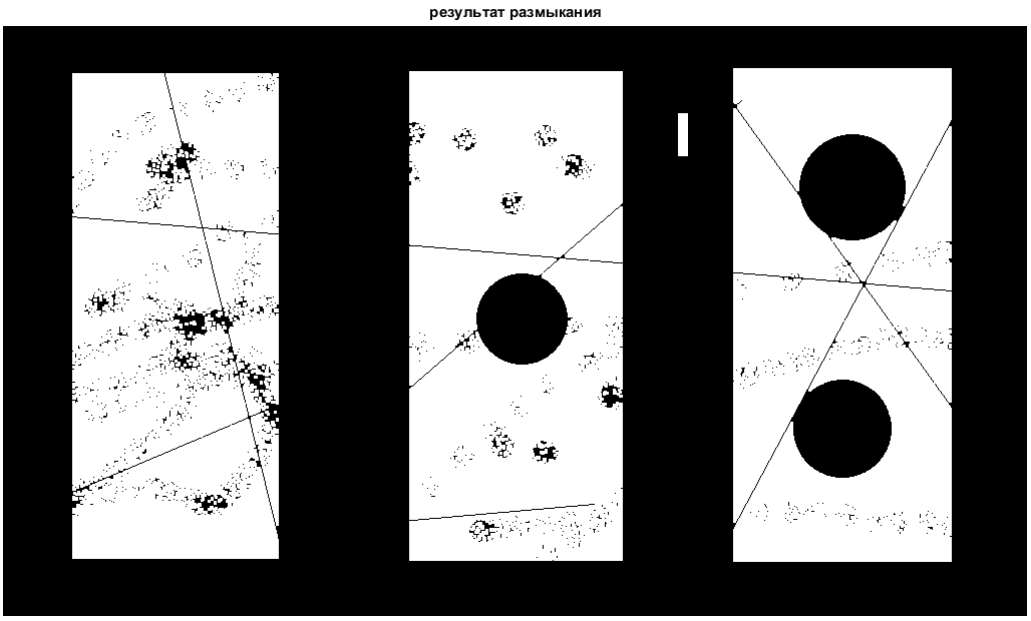
Для операции размыкания можно использовать либо последовательный вызов функций *imerode()* и *imdilate()*, либо отдельную функцию *imopen()*.

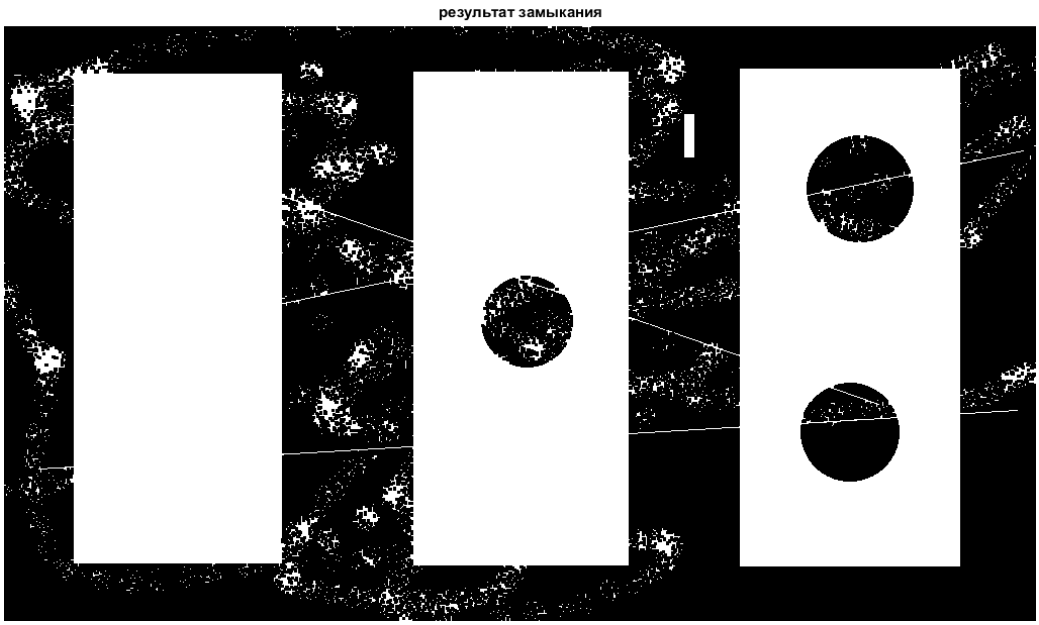
C=imopen(B,se);

Последовательность операций дилатации и эрозии производят другой эффект: она удаляет шумовые области внутри объекта (ложные отверстия), не затрагивая фон. Такая операция называется операцией *замыкания*. Для её реализации служит функция *imclose()*.

C=imclose(B,se);

Используя операции размыкания и замыкания, можно очень качественно очистить изображение от шумовых элементов, которое невозможно сделать, используя низкоуровневые фильтры.





Что делать, если на изображении всё ещё остались шумовые элементы?

В этом случае можно увеличить «агрессивность» размыкания и замыкания, увеличив размер структурного элемента до 5x5, 7x7 и т.д. Структурный элемент размера NxN позволит удалять шумовые элементы с толщиной менее N. Обратите внимание, что при слишком большом размере структурного элемента могут пострадать и мелкие элементы самого объекта.

Еще один способ удалить объекты с размером меньше заданного заключается в использовании функции *bwareaopen()*:

C=bwareaopen(B,мин.размер);

Используя эти подходы, добейтесь, чтобы изображение деталей не содержало помех.

Разметка объектов

Разметка объектов представляет собой операцию, при которой точки каждого объекта получают разметку, равную номеру объекта. При этом точки фона остаются размеченными нулями. Для выполнения разметки служит функция *bwlabel():*

D=bwlabel(C);

figure,imshow(D,[]),title('разметка');

Разметка играет важную роль – она позволяет отделить объекты друг от друга и анализировать их независимо друг от друга.

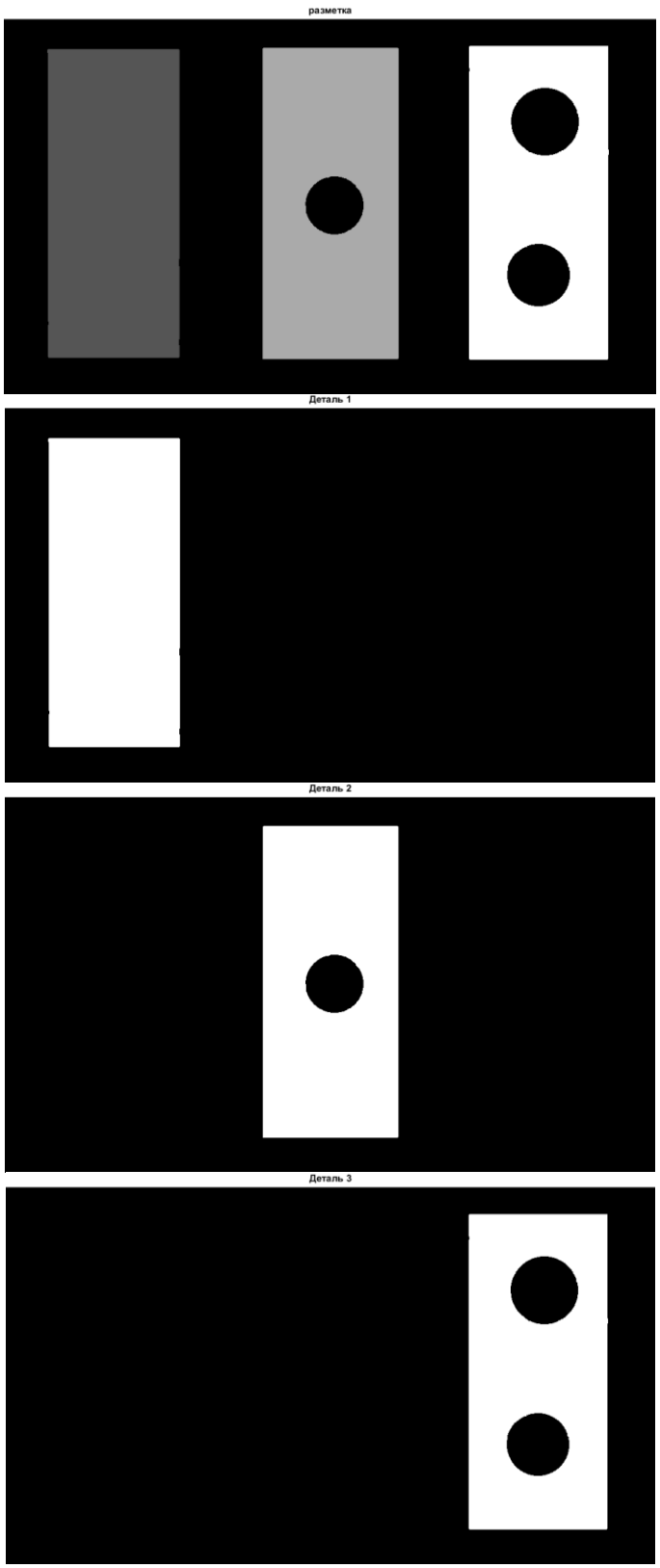
Для копирования точек с нужной разметкой удобно воспользоваться функцией *[x,y]=find(условие)*. Она возвращает список координат элементов, для которых выполняется заданное условие. Так, выполнение команд:

D2=zeros(size(D));

D2(find(D==2))=1;

приведет к копированию в матрицу D2 точек, содержащих разметку 2.

Напишите код, который скопирует детали соответственно в матрицы D1, D2 и D3.



Другие морфологические операции

Выделение границ

Для выделения границ можно использовать функцию *bwperim()*:

E=bwperim(D1);

Обратите внимание – морфологические границы всегда непрерывны и замкнуты. Даже если объект выходит за кадр – продолжением границ объекта являются границы

кадра. Границы, полученные другим путём, могут и не обладать этим свойством. В этом плане морфологические границы являются наиболее качественными.

Большинство других морфологических операций собрано в функции *bwmorph():*

результат = bwmorph(изображение, 'название операции', кол-во проходов);

Для получения списка доступных операций, в командном окне вызовите справку по этой функции:

>help bwmorph

В списке фигурируют уже известные нам операции эрозии, дилатации, размыкания и замыкания. Они тут немного менее удобны, поскольку не позволяют задать матрицу структурного элемента (она там всегда 3x3), однако их тоже можно использовать.

Помимо них, в списке присутствует ещё множество других операций. Познакомимся с некоторыми из них.

Скелетные линии

Скелетная линия объекта представляет собой геометрическое место точек, равноудалённых от границ объекта. Скелетные линии удобно использовать, например, для сравнения объектов в задаче поиска или идентификации.

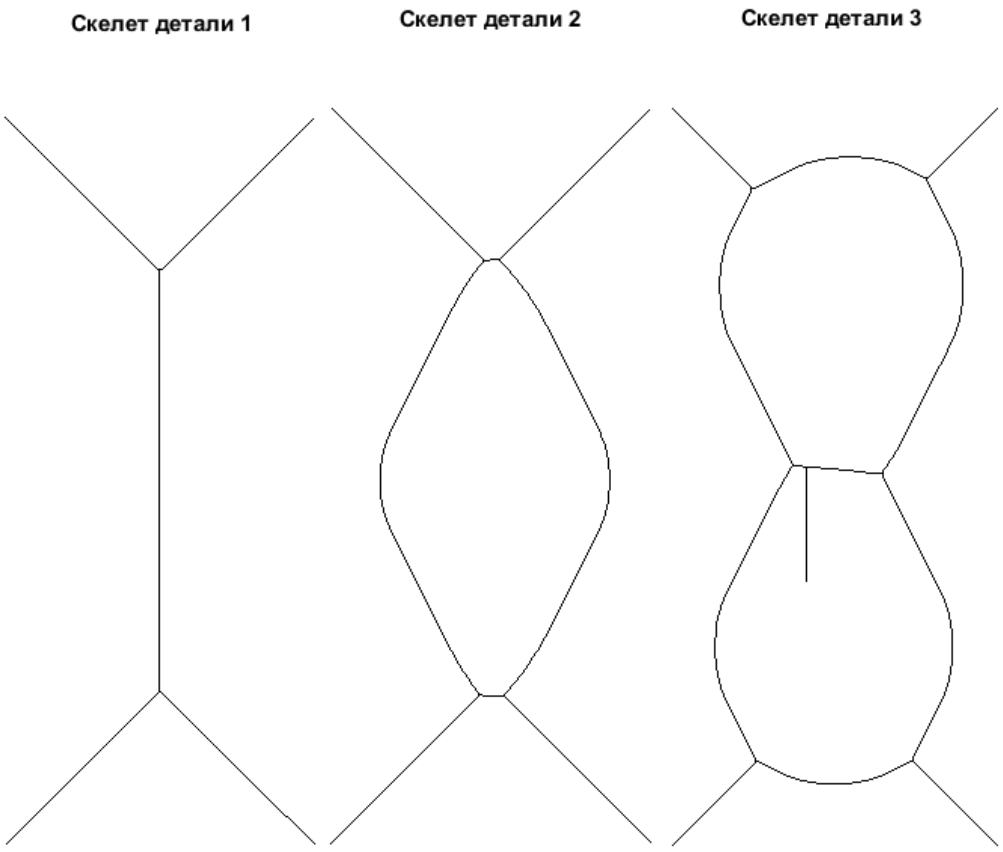
Для построения скелетной линии воспользуемся функцией bwmorph(). В качестве максимального количества проходов укажем Inf (бесконечность) – функция будет работать, пока не останутся только точки скелетных линий.

C1=bwmorph(D1,'skel',inf);

figure,imshow(C1),title('скелетная линия первой детали');

Аналогично постройте скелетную линию второй детали.

Почему скелетные линии деталей сильно отличаются?



Удаление шпор

Полученные скелетные линии имеют своеобразный дефект в виде отростков – шпор. Для их удаления можно воспользоваться операцией удаления шпор (‘spur’):

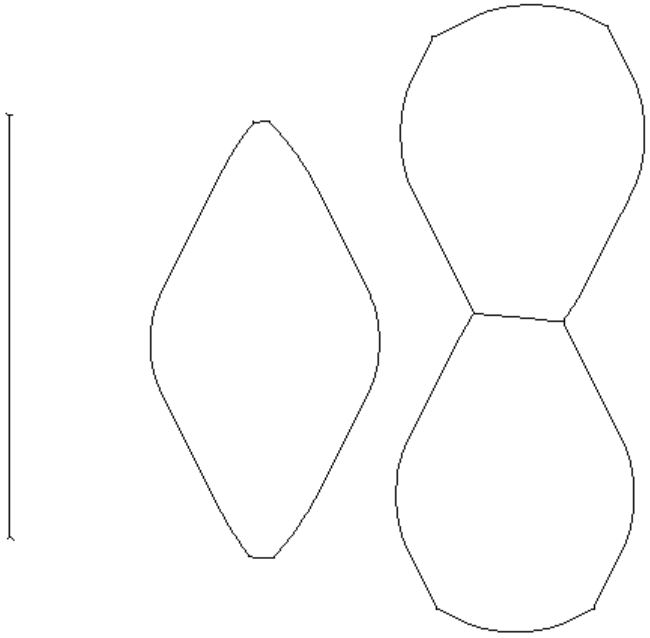
C1=bwmorph(D1,’skel’,inf);

C1=bwmorph(C1,’spur’,80);

figure,imshow(C1),title(‘удаление шпор первой детали’);

Аналогично удалите шпоры для остальных деталей.

Скелет без шпор-1 Скелет без шпор-2 Скелет без шпор-3

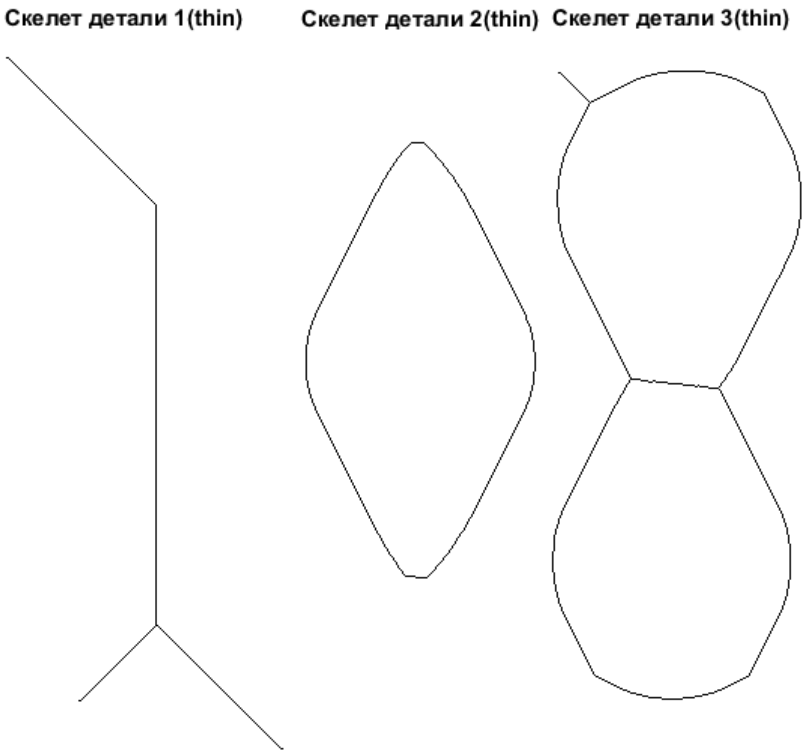


Альтернативным подходом является использовании операции уменьшения толщины ‘thin’. Она также позволяет строить скелетную линию, но при этом не образует шпор:

C1=bwmorph(D1,’thin’,inf);

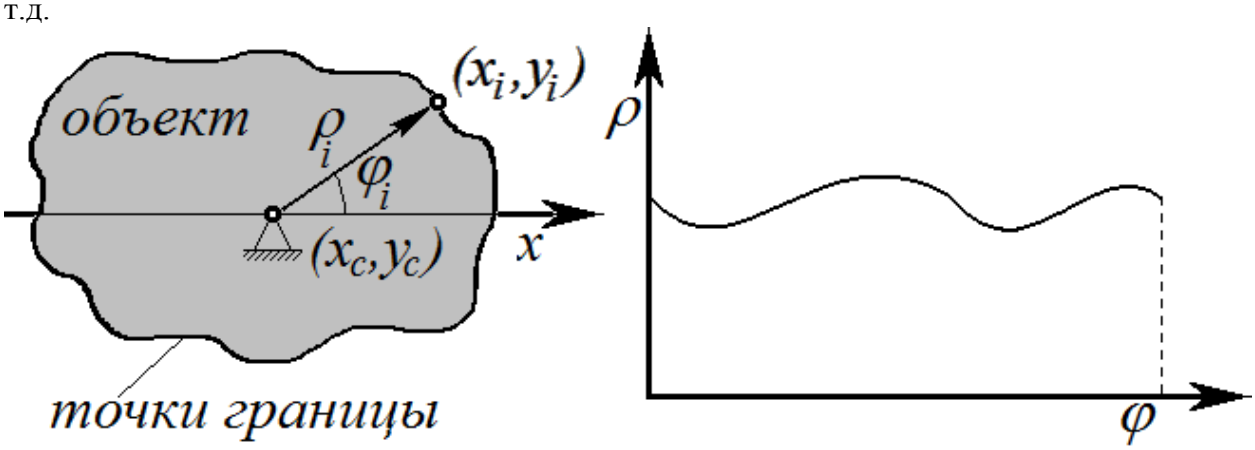
figure,imshow(C1),title(‘скелетная линия –thin – первой детали’);

Аналогично постройте скелетную линию для остальных деталей.



Центроидальный профиль.

Центроидальным профилем называют зависимость длины радиус-вектора, пущенного из центра масс объекта к его границам, от направления этого радиус вектора. Центроидальный профиль обладает многими важными свойствами, делают его удобным инструментом для поиска и распознавания объекта, определения его формы, ориентации и т.д.



Вновь откройте редактор mspaint и нарисуйте чорный пятиугольник на белом фоне. Создайте новый скрипт, загрузите изображение, бинаризуйте его.

Найдите границы объекта, используя функцию *bwperim()*.

Найдите центр масс объекта (можно использовать прямую формулу, либо функции *find* и *mean*). Самый простой способ найти центр масс – построить список координат его точек (при помощи функции find()) и посчитать среднее (функция mean()).

Далее представим центроидальный профиль массивом R(f), где R – длина радиус-вектора, а f – его направление. Будем перебирать в двойном цикле все точки границы. Для каждой точки посчитаем длину и направление радиус-вектора:

ro=sqrt ((x–xc)^2 + (y–yc)^2));

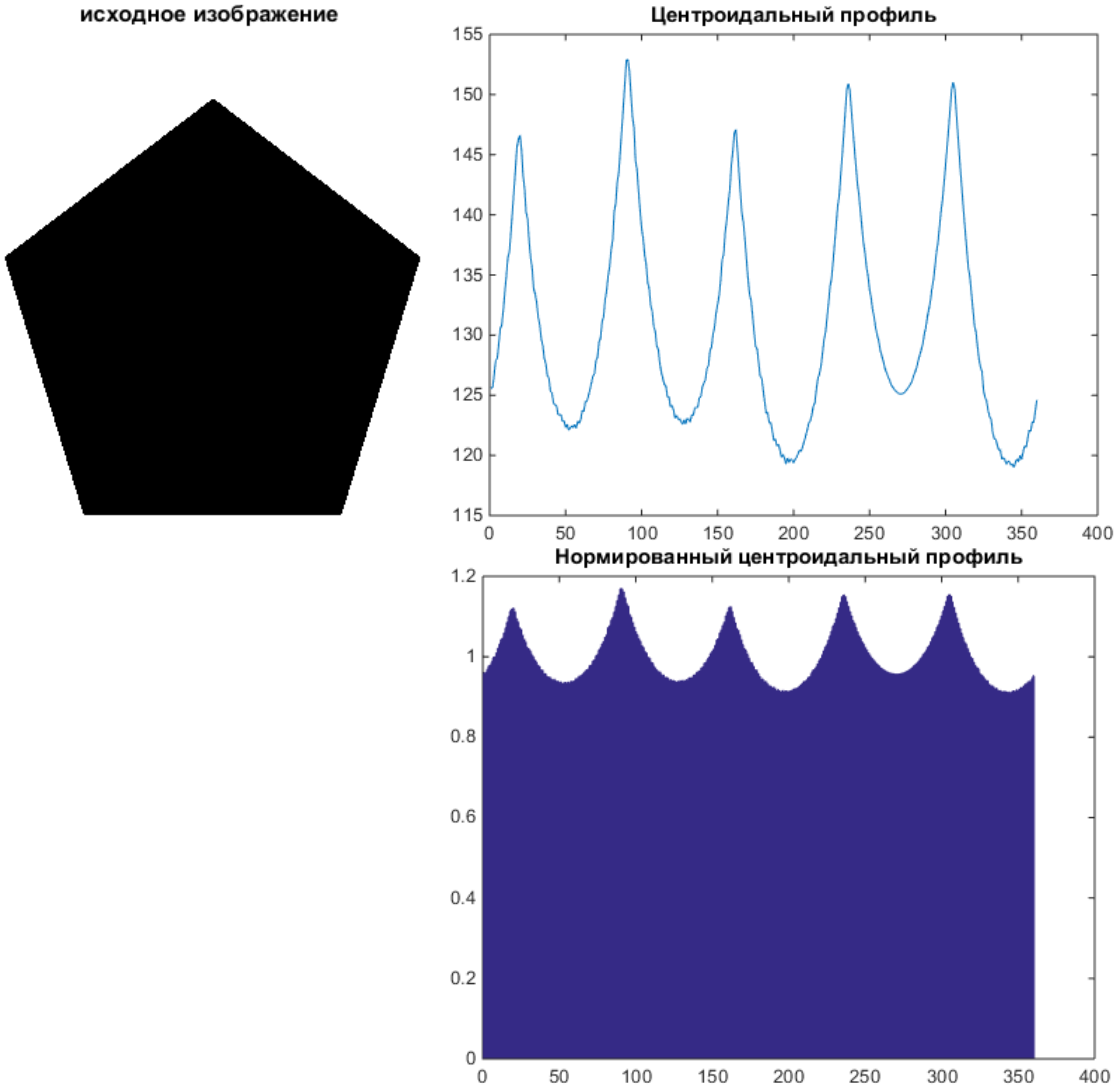
fi=atan2((y–y2), (x–x2));

После этого найденное значение ro занесём в R[fi].

После просмотра всех точек границ получим центроидальный профиль в виде вектора R[fi].

Нормируйте полученный вектор R, разделив его элементы на его среднее значение.

Выведите его на экран при помощи функции plot() или bar().



Инвариантность центроидального профиля.

Измените масштаб исходного изображения при помощи функции *imresize()*. Увеличьте масштаб на 25%, уменьшите на 25%. Как изменился центроидальный профиль?

Поверните исходное изображение на 10 градусов по часовой стрелке и против часовой при помощи функции *imrotate()* (перед поворотом изображение надо инвертировать, чтобы фон стал черным). Постройте центроидальный профиль. Как изменился центроидальный профиль?

Оформите расчет центроидального профиля в виде функции. На вход должен подаваться выделенный объект, на выходе – нормированный центроидальный профиль.

Задания для самостоятельной работы

Часть 1. Бинаризация, фильтрация, разметка.

Загрузите изображение риса «rice.png»

Бинаризуйте и отфильтруйте его, выполните разметку.

Посчитайте, сколько всего зёрен риса содержит изображение.

Выведите на экран зерно, номер которого равен вашему номеру по журналу, умноженному на пять.

Найдите изображения рисинок, которые склеились в один объект и подлежат разделению.

Результаты добавьте в отчет.

Часть 2. Характеристики бинарных объектов

Откройте редактор mspaint, нарисуйте изображение нескольких деталей.

Одна из деталей должна иметь форму правильного пятиугольника.

Одна из деталей должна содержать четыре отверстия.

Пользуясь методами, изученными в лабораторной работе, напишите скрипт, который позволяет найти и закрасить на изображении пятиугольную деталь красным цветом, а деталь с четырьмя отверстиями – зеленым.

Результаты добавьте в отчет.

Приложения

Приложение 1. Скрипт для изучения бинаризации (фотограф).

```
clear;
close all;
A=imread('cameraman.tif');
figure,imshow(A),title('Загруженное изображение');
figure,plot(imhist(A)),title('Гистограмма яркости');

%Делаем бинаризацию с порогом 75, выбранным по гистограмме
B=im2bw(A, 75 / 255 );
figure,imshow(B),title('результат бинаризации');
B=imcomplement(B);imshow(B), title('Инверсия');

%Пробуем бинаризацию с большим и меньшим порогом,
%чтобы убрать ложные объекты
B1=im2bw(A, 50/255 ); figure,imshow(B1),title('Порог= 50');
B2=im2bw(A,100/255 ); figure,imshow(B2),title('Порог=100');

%Автоматический выбор порога методот Отсу
L=graythresh(A);
fprintf('L=%f (%f)\n',L,L*255);
B=im2bw(A,L);
figure,imshow(B),title('graythresh');
```

Приложение 2. Скрипт для изучения локальной бинаризации (рис).

```
clear;
close all;
A=imread('rice.png');
figure,imshow(A),title('исходное изображение');
L=graythresh(A);
B=im2bw(A, L );
figure,imshow(B),title('результат бинаризации');

B1=im2bw(A, 100/255 );figure,imshow(B1),title('порог=100');
B2=im2bw(A, 150/255 );figure,imshow(B2),title('порог=150');
figure,plot(imhist(A));

A1=A(1:180,:);  %верхняя часть
A2=A(181:end,:);    %нижняя часть
L1=graythresh(A1);  %порог для верхней части
L2=graythresh(A2);  %порог для нижней части
B1=im2bw(A1,L1);    %бинаризация верхней части
B2=im2bw(A2,L2);    %бинаризация нижней части

% Склеим результат:
B=zeros(size(A));   %создаём матрицу результата, куда будем копировать части
B(1:180,:)=B1;  %копируем верхнюю часть
B(181:end,:)=B2;    %копируем нижнюю часть
figure,imshow(B),title('бинаризация по частям');
```

Приложение 3. Скрипт для морфологических алгоритмов (детали).

```
clear;
close all;
A=imread('detal.bmp');
figure,imshow(A),title('исходное изображение');
B=im2bw(A,0.5);
B=imcomplement(B);
figure,imshow(B),title('результат бинаризации');

%Эрозия и дилатация
se=ones(3,3);
C=imerode(B,se);figure,imshow(C),title('результат эрозии');
C=imdilate(B,se);figure,imshow(C),title('результат дилатации');

%Размыкание и замыкание
C=imopen(B,se); figure,imshow(C),title('результат размыкания');
C=imclose(B,se);figure,imshow(C),title('результат замыкания');

%фильтрация
C=medfilt2(B);

se=ones(3,3);
C=imopen(C,se); figure,imshow(C),title('результат размыкания');
C=imclose(C,se);figure,imshow(C),title('результат замыкания');
se=ones(5,5);
C=imopen(C,se); figure,imshow(C),title('результат размыкания 5x5');
C=imclose(C,se);figure,imshow(C),title('результат замыкания 5x5');
se=ones(7,7);
C=imopen(C,se); figure,imshow(C),title('результат размыкания 7x7');
C=imclose(C,se);figure,imshow(C),title('результат замыкания 7x7');
C=bwareaopen(C,500);figure,imshow(C),title('результат bwareaopen');

% Разметка и выделение объектов
D=bwlabel(C);
figure,imshow(D,[]),title('разметка');
D1=zeros(size(D)); D2=D1;D3=D1;
D1(find(D==1))=1;
D2(find(D==2))=1;
D3(find(D==3))=1;
figure,imshow(D1),title('Деталь 1');
figure,imshow(D2),title('Деталь 2');
figure,imshow(D3),title('Деталь 3');

% Скелетные линии
S1=bwmorph(D1,'skel',Inf);
S2=bwmorph(D2,'skel',Inf);
S3=bwmorph(D3,'skel',Inf);
figure,imshow(imcomplement(S1)),title('Скелет детали 1');
figure,imshow(imcomplement(S2)),title('Скелет детали 2');
figure,imshow(imcomplement(S3)),title('Скелет детали 3');

% Удаление шпор
S1=bwmorph(S1,'spur',100);
S2=bwmorph(S2,'spur',100);
S3=bwmorph(S3,'spur',100);
figure,imshow(imcomplement(S1)),title('Скелет без шпор-1');
figure,imshow(imcomplement(S2)),title('Скелет без шпор-2');
figure,imshow(imcomplement(S3)),title('Скелет без шпор-3');

% Скелетные линии - через уменьшение толщины
S1=bwmorph(D1,'thin',Inf);
S2=bwmorph(D2,'thin',Inf);
```

```
S3=bwmorph(D3,'thin',Inf);
figure,imshow(imcomplement(S1)),title('Скелет детали 1(thin)');
figure,imshow(imcomplement(S2)),title('Скелет детали 2(thin)');
figure,imshow(imcomplement(S3)),title('Скелет детали 3(thin)');
```

Приложение 4. Скрипт для изучения свойств центроидального профиля (пятиугольник).

```
clear;
close all;
A=imread('pentagon.bmp');
figure,imshow(A),title('исходное изображение');
B=im2bw(A,0.5);
B=imcomplement(B);
figure,imshow(B),title('результат бинаризации');

E=bwperim(B);

[y,x]=find(B==1);
xc=mean(x);
yc=mean(y);
B(round(yc),round(xc))=0;
figure,imshow(B),title('центр масс');

[y,x]=find(E==1);
figure,imshow(E),title('границы');
dx=x-xc;
dy=y-yc;

ro=sqrt(dx.^2+dy.^2);
fi=atan2d(dy,dx);
f=floor(fi)+180+1;

n=length(x);
R=zeros(360,1);

for i=1:n
    R(f(i))=max(R(f(i)),ro(i));
end

figure,plot(R),title('Центроидальный профиль');

R=R./mean(R);
figure,bar(R),title('Нормированный центроидальный профиль');
```

Приложение 5. Функция построения нормированного центроидального профиля.

```
function [ R ] = CentProf( B )
%CentProf строит нормированный центроидальный профиль объекта
%B-бинаризованное изображение

E=bwperim(B);

[y,x]=find(B==1);
xc=mean(x);
yc=mean(y);

[y,x]=find(E==1);
dx=x-xc;
dy=y-yc;

ro=sqrt(dx.^2+dy.^2);
fi=atan2d(dy,dx);
f=floor(fi)+180+1;

n=length(x);
R=zeros(360,1);

for i=1:n
    R(f(i))=max(R(f(i)),ro(i));
end

R=R./mean(R);

end
```