

MINISTRY OF SCIENCE AND HIGHER EDUCATION OF THE
RUSSIAN FEDERATION

FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION OF HIGHER
EDUCATION
"NORTH CAUCASUS FEDERAL UNIVERSITY"

SUGGESTIONS FOR TEACHING
on the implementation of practical classes in the
discipline

Digital Image Processing

For students

Areas of training

11.03.02 Infocommunication Technologies and
Communication Systems

Profile

Intelligent Data Processing in Infocommunication
Systems and Networks

CONTENTS

INTRODUCTION

PRACTICAL LESSON №1. HISTOGRAMS, PROFILES AND PROJECTIONS

PRACTICAL LESSON №2. GEOMETRIC TRANSFORMATIONS OF
IMAGES

PRACTICAL LESSON №3. FILTRATION AND CONTOUR SELECTION

PRACTICAL LESSON №4. IMAGE SEGMENTATION

PRACTICAL LESSON №5. HOUGH'S TRANSFORMATION

PRACTICAL LESSON №6. MORPHOLOGICAL IMAGE ANALYSIS

INTRODUCTION

The purpose of the discipline is to form students' professional competencies based on the development of modern computer vision systems, as well as the basics of digital image processing that meet the requirements of professional standards and employers.

Objectives of mastering the discipline:

Provide students with basic information in an easy-to-understand way about:

- sampling and quantization of continuous images;
- Improve the visual quality of images by element-by-element transformation.
- image filtering;
- image restoration;
- geometric transformations and image reference;
- photogrammetry and stereovision;
- management of image processing and analysis processes.

Place of the discipline (module) in the structure of the educational program

Discipline "Digital Processing images" Applies K Parts
disciplines, formed participants Educational Disciplines
(modules) of choice 3 (DV.3).

List of planned learning outcomes in the discipline (module) correlated with the planned results of the development of the educational program

Code, Statement of competence	Indicator code, wording	Planned learning outcomes in the discipline (module), Characterizing the stages of formation Competencies, Indicators
PC-1. Capable Develop Requirements and Design Software Collateral	ID-1PK-1 understands Principles Constructions Architecture program Provision and Types of software architecture support; understands the methods and Funds design program of support, Methods and Funds design bases data; understands Methods and Funds design program interfaces; understands methods and	Knows and understands Principles Constructions Architecture program Provision and Species Architecture program support; knows Main Methods and Funds design program

Code, Statement of competence	Indicator code, wording	Planned learning outcomes in the discipline (module), Characterizing the stages of formation Competencies, Indicators
	methods of formalizing tasks; understands Capabilities of modern and future means Development program products, technical means.	support; understands the methods and means of designing computer vision systems; understands the methods and techniques of formalizing tasks Digital Processing images; understands the possibilities of modern and advanced development tools software products, technical funds.
	ID-2PK-1 conducts performance analysis requirements; develops Implementation options requirements; evaluates and substantiates the recommended decisions; selects the means of implementing software requirements; uses existing standard solutions and software design patterns.	Carries out analysis of the subject area, develops development options computer vision systems and conducts Assessment and substantiation of the developed decisions. Makes a choice Technical and program tools, uses existing standard solutions and Design patterns In progress system development technical vision.
	ID-3PK-1 carries out Time Estimation and labor intensity	Produces Time Estimation and

Code, Statement of competence	Indicator code, wording	Planned learning outcomes in the discipline (module), Characterizing the stages of formation Competencies, Indicators
	implementation of requirements for software; applies methods and means of designing software, data structures, databases, program interfaces.	labor intensity of the implementation of requirements for the developed software security. Applies Methods and means design Systems technical vision.
PC-3. Able to carry out conceptual, functional and logical design of medium and large systems scale and difficulties	ID-1PK32 implements methods for planning project work; applies methods of classical system analysis; uses the theory of business process management, methods of conceptual design, methods for assessing the quality of software systems; solves optimization problems when developing business requirements for the system	Applies methods of planning project works, basics Development and management business processes. Understands and Applies Methods of conceptual design.
	ID-2PK-3 develops a feasibility study; develops a Terms of reference on system; develops requirements for the subsystems of the system and carries out their quality control; organizes conformity assessment the requirements of existing systems and their analogues; performs support of acceptance tests and commissioning of operation systems;	Develops Component Models technical systems vision. Develops computer vision systems and the necessary requirements for them.

Code, Statement of competence	Indicator code, wording	Planned learning outcomes in the discipline (module), Characterizing the stages of formation Competencies, Indicators
	Processes Inquiries to change the requirements for the system.	
	ID-3PK-3 draws up schedules of tests measures; develops business requirements for the system; is able to determine the key properties and limitations of the system; identifies the subsystems of the system	Identifies the business requirements for the system in the process design, defines the corresponding constraints of the system.
	ID-4PC-3 Analyzes requests for changes in the properties and requirements of the developed system and analyzes their impact	On your own and actively analyzes requests for changes in properties and requirements for the development of the system and analysis of their impact
	ID-6PK-3 Applies methods and algorithms Executable code optimizations	Competently applies optimization methods and algorithms executable code
	ID-8PK-3 Develops implementation options Software	Determines Implementation options Software

Practical Lesson No1 Histograms, Profiles and Projections

Purpose of the work

Mastering the basic brightness and geometric characteristics of images and using them for image analysis.

Methodological recommendations

Before starting work, students should familiarize themselves with the main functions of the MATLAB environment for working with histograms, profiles and image projections.

Theoretical information

A pixel in a digital image is characterized by three parameters: (x, y, I) , where a pair of integer values (x, y) describes the geometric position of the pixel in the image plane, and the value I characterizes its brightness (intensity) at a point in the plane. (for example, a change in the illumination of the scene will not change the geometric parameters of the objects in the scene). Because of this, it is easier to study the brightness properties of the image separately and the geometric properties separately. This approach lowers the order of the studied image in the case of geometric properties from $n = 3$ (x, y, I) to $n = 2$ (x, y) , and in the case of luminance properties, to $n = 1$ (I) . The luminance component of the image is characterized by a one-dimensional histogram array, from which it is possible to calculate contrast.

A histogram is the distribution of the frequency of occurrence of pixels of the same brightness in an image.

Luminance is the average value of the signal intensity.

Contrast is the range between the minimum and maximum brightness of an image.

To reduce the geometric components of an image to a single-numbered $n = 1$ data set, such characteristics as "*profiles*" and "*projections*" of the image are used.

A *profile* along a line is a function of the intensity of an image distributed along a given line (*slot*).

The projection on the axis is the sum of the intensities of the pixels of the image, taken in the direction perpendicular to this axis.

Image histogram

For an 8-bit grayscale image, the luminance histogram is a one-dimensional integer Hist array of 256 elements [0 . . . 255] The element of the Hist[*i*] histogram is the sum of the pixels of the image with the brightness *i*. From the visual display of the histogram, it is possible to assess the need to change the brightness and contrast of the image, to estimate the area occupied by light and dark elements, to determine the location on the image plane of individual objects corresponding to certain brightness ranges. For a color RGB image, it is necessary to build three histograms for each color.

Listing 1.1. Drawing image histograms.

```
1 [ num Rows num Cols Layers] = size ( I);  
2 imhist( I(:, :, 1)); %red  
3 imhist( I(:, :, 2)); %green  
4 imhist( I(:, :, 3)); %blue
```

If the histogram is uneven, it can be aligned to improve the image, and the histogram can be aligned in different ways depending on the problem being solved.

Arithmetic operations

For example, if most of the histogram values are on the left, then the image is dark. To increase the detail of the dark areas, you can move the histogram to the right, to a lighter area, for example, by 50 gradations for each color.

$$I_{new}(i,j) = I(i,j) + 50 / 255;$$

The histogram of the original image is shifted to the middle part of the range, which is more acceptable from the point of view of visual perception. This approach has the following disadvantage: increasing the intensity of the dark areas leads to a shift of the light areas to the maximum, which can lead to the loss of information in the light areas.

Dynamic range stretching

If the pixel intensities of the regions of interest are in a narrow dynamic range, you can stretch this range. Such transformations are performed according to the following statement:

$$I_{new} = \left(\frac{I - I_{min}}{I_{max} - I_{min}} \right)^a, \quad (1.1)$$

where I and I_{new} are arrays of the intensity values of the original and new images, respectively; I_{min} and I_{max} are the minimum and maximum values of the intensities of the original image, respectively; a is the nonlinearity coefficient.

This expression is nonlinear because of the coefficient a . If $a = 1$, applying formula (1.1) to the original image will not give the desired effect, since the histograms of the color components of the image occupy the entire possible diapason. Nonlinear transformations are carried out for each color component.

Listing 1.2. Nonlinear extension of the dynamic range at $a = 0.5$.

```

1 [ num Rows num Cols Layers] = size ( I);
2 for k =1:1: Layers;
3     Imin = min ( min ( I(:, :, k)));
4     Imax = max( max( I(:, :, k)));
5     for i=1:1: num Rows;
6         for j=1:1: num Cols;
7             Inew (i,j, k) = ...
8                 ((( I(i,j, k) - Imin ) / ...
9                     ( Imax - Imin ))) ^ 0.5 ;

```

```

10             if Inew (i,j, k) > 1;
11                 Inew (i,j, k) = 1;
12             end
13             if Inew (i,j, k) < 0;
14                 Inew (i,j, k) = 0;
15             end
16         end
17     end
18 end

```

Uniform Conversion

It is carried out according to the following formula:

$$I_{new} = (I_{max} - I_{min}) \cdot P(I) + I_{min} \quad (1.2)$$

where I_{min} , I_{max} are the minimum and maximum values of the intensities of the original image I ; $P(I)$ is a probability distribution function of the original image, which is approximated by a cumulative histogram:

$$P(I) \approx \sum_{m=0}^I \text{Hist}(m). \quad (1.3)$$

The cumulative histogram of the source image in the MATLAB environment can be calculated using the `cumsum()` function:

```
CH = cumsum(H) ./ (numRows * numCols);
```

where H is the histogram of the original image, `numRows` and `numCols` are the number of rows and columns of the original image, respectively.

According to formula (1.2), it is possible to calculate the values of the pixel intensities of the resulting image. In the MATLAB environment, the software implementation is as follows:

```
Inew(i,j) = CH(ceil(255 * I(i,j) + eps));
```

The `eps` parameter is necessary to protect the program from assigning zero values to the indices of the cumulative histogram.

Exponential Transformation

It is carried out according to the following formula:

$$I_{new} = I_{min} - a \frac{1}{\ln(1 - P(\lambda))}, \quad (1.4)$$

where a — constant, characterizing the steepness of the transformation. According to the formula (1.4) It is possible to calculate the pixel intensity values of the resulting image. In the MATLAB environment, the software implementation is as follows:

$$I_{new}(i,j) = I_{min} - (1 / \text{alfa}) * \dots \\ \log_{10}(1 - \text{CH}(\text{ceil}(255 * I(i,j) + \text{eps}))));$$

Transformation according to Rayleigh's Law

It is carried out according to the following formula:

$$I_{new} = \frac{1}{2a^2} \left(\frac{1}{1 - P(\lambda)} \right)^{1/2}, \quad (1.5)$$

where a is a constant that characterizes the histogram of the distribution of intensities of the elements of the resulting image. In the MATLAB environment, the software implementation looks like this:

$$I_{new}(i,j) = I_{min} + \text{sqrt}(2 * \text{alfa}^2 * \dots \\ \log_{10}(1 / (1 - \text{CH}(\text{ceil}(255 * I(i,j) + \text{eps})))));$$

Transformation by Law of Degree 2/3

It is carried out according to the following formula:

$$I_{new} = (P(\lambda))^{2/3}. \quad (1.6)$$

In the MATLAB environment, the software implementation is as follows:

$$I_{new}(i,j) = (\text{CH}(\text{ceil}(255 * I(i,j) + \text{eps}))))^{(2/3)};$$

Hyperbolic transformation

It is carried out according to the following formula:

$$I_{new} = a^{(P(\lambda))}, \quad (1.7)$$

where σ is the constant relative to which the transformation is carried out and, as a rule, is equal to the minimum value of the intensity of the elements of the original image $\sigma = I_{min}$.

In the MATLAB environment, the software implementation looks like this:

$$I_{new}(i,j) = I_{min}^{(CH(ceil(255 * I(i,j) + eps)))};$$

The histogram conversion techniques discussed above can be used to eliminate distortions in the transmission of the levels of quantification that the images have been subjected to during the generation, transmission or processing of the data. In addition, these methods can be applied not only to the entire image, but can be used locally in *a sliding window*, which will increase the detail of individual areas.

MATLAB implements several functions that automatically align grayscale histograms:

1. `imadjust()` — increases the contrast of the image by changing the range of intensities of the original image;
2. `histeq()` — equalizes (aligns) the histogram of the distribution of intensity values of the elements of the original image;
3. `adapthisteq()` — performs contrast-constrained adaptive alignment of the histogram by analyzing and equalizing histograms of the local surroundings of the image.

For color image processing, these functions can be applied to each color in turn.

Image Profile

An image profile along a line is a function of the intensity of an image distributed along a given line (*cut*). The simplest case of an image profile is a row profile:

$$\text{Profile } I(x) = I(x, i), \quad (1.8)$$

where i is the line number of the image I .

Image Column Profile:

$$\text{Profile } \mathcal{A}(j) = \mathcal{A}_j(j), \quad (1.9)$$

where j is the number of the image column I .

In general, the profile can be viewed along any straight, broken or curved line that intersects the image. After the image profile array is formed, it is analyzed using standard tools. The analysis allows you to automatically select special points of the profile function that correspond to the contours of the image intersected by this line. Fig. 1.1 shows the image profile of a barcode taken along the Ox axis. This profile contains all the necessary information for reading the barcode, since it allows you to determine the sequence of alternating "thick" and "thin" hrews and spaces of different widths. In the MATLAB environment, the image profile can be found using the `improfile()` function.

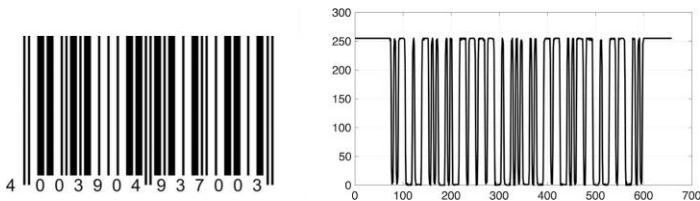


Fig. 1.1 — On the left is the barcode, on the right — its profile along the Ox axis.

Listing 1.3. Search for a barcode profile along the Ox axis.

```
1 I = imread (' code . jpg ');
2 [ numRows , numCols , Layers] = size ( I);
3 x = [1 num Cols ];
4 y = [ ceil( num Rows /2) ceil( num Rows /2)];
5 figure
6 improfile (I,x, y), grid on;
```

To interactively specify the line (polyline) along which the profile should be drawn, you must use the parameterless `improfile` function.

Listing 1.4. Interactive definition of the profile line.

```
1 I = imread ( ' code . jpg ');
2 imshow ( I);
3 improfile
```

Image projection

The projection of an image on a certain axis is the sum of the intensities of the pixels of the image in the direction perpendicular to this axis. The simplest case of projection of a two-dimensional image is a vertical projection on the Ox axis, which is the sum of the intensities of pixels along the columns of the image:

$$\text{Proj } A(y) = \sum_{x=0}^{\text{dir } Y-1} A(x, y), \quad (1.10)$$

and the horizontal projection on the Oy axis, which is the sum of the pixel intensities *along the image lines*:

$$\text{Proj } A(x) = \sum_{y=0}^{\text{dir } X-1} A(x, y). \quad (1.11)$$

Let's write an expression for projection on an arbitrary axis. Let's assume that the direction of the axis is given by a unit vector with coordinates (e_x, e_y) . Then the projection of the image on the axis Oe is defined by the following expression:

$$\text{Proj } A(t) = \sum_{xe_x + ye_y = t} A(x, y). \quad (1.12)$$

Analysis of the projection array makes it possible to identify characteristic points of the projection function that correspond to the contours of objects

Build image projections on the vertical and horizontal axes. Define the boundaries of object areas.

3. *Projections*. Select an arbitrary image containing a barcode. Build an image profile along the barcode.

Contents of the report

1. Objective.
2. Theoretical substantiation of the applied methods and functions of geometric transformations.
3. Progress of work:
 - (a) Original images;
 - (b) Listings of software implementations;
 - (c) Comments;
 - (d) Resulting images.
4. Conclusions on the work done.

Questions to the defense

1. What is image contrast and how can I change it?
2. What are the benefits of using image profiles and projections?
3. How can you find an object on a uniform background?

Practical Lesson No2 Geometric Transformations of Images

Purpose of the work

Mastering the main types of mappings and using geometric transformations to solve problems of spatial correction of images.

Methodological recommendations

Before starting work, students should familiarize themselves with the basic functions of the MATLAB environment for working with geometric image transformations.

Theoretical information

Geometric image transformations involve spatially changing the location of a set of pixels with integer coordinates (X, Y) into another set with coordinates (X', Y') , while the intensity of the pixels is preserved. In two-dimensional plane geometric transformations, as a rule, Euclidean space **P2** is used with an orthonormalized Cartesian coordinate system. In this case, the pixel of the image corresponds to a pair of Cartesian coordinates, which are interpreted as a two-dimensional vector represented by a segment from point $(0, 0)$ to point $X_i = (x_i, y_i)$. Two-dimensional transformations on a plane can be represented as the movement of points corresponding to a set of pixels.

For generality with further transformations, we will use *homogeneous coordinates*, which have the property that the object they define does not change when all the coordinates are multiplied by the same non-zero number. Because of this property, the required number of coordinates to represent points is always one

For example, to represent a point $X = (x, y)$ on a plane in a two-dimensional space $P2$, three coordinates $X = (x, y, w)$ are needed.

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = w \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Leftrightarrow \begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} w, \quad (2.1)$$

where w — scalar arbitrary factor,
 w

$$x = \frac{x'}{w}, \quad y = \frac{y'}{w}$$

With the help of triples of homogeneous coordinates and third-order matrices, any linear transformation of a plane can be described. Thus, geometric transformations are matrix transformations, and the set coordinates of the pixels of the transformed image and the original image are connected by the following matrix relation, either in the inline form $X' = XT$ or in the columnar form $X' = T^T X$. Let's describe these matrix relationships:

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} x \\ y \\ w \end{bmatrix} \cdot \begin{bmatrix} A & D & G \\ B & E & H \\ C & F & I \end{bmatrix} \quad (2.2)$$

$$\Leftrightarrow \begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ w \end{bmatrix}. \quad (2.3)$$

A point with Cartesian coordinates (x, y) in homogeneous coordinates will be written as $(x, y, 1)$:

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \cdot \begin{bmatrix} A & D & 0 \\ B & E & 0 \\ C & F & 1 \end{bmatrix} \quad (2.4)$$

$$\Leftrightarrow \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} A & B & C \\ D & E & F \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}. \quad (2.5)$$

Formula (2.4) can be represented as the following system of equations:

$$\begin{cases} x' = Ax + By + C, \\ y' = Dx + Ey + F. \end{cases} \quad (2.6)$$

Linear transformations

A conformal mapping is a mapping that preserves the shape of infinitesimal shapes and the angles between curves at their intersections. The basic linear conformal transformations are Euclidean transformations. These include shift, reflection, uniform scaling, and rotation. Conformal transformations are a subset of affine transformations.

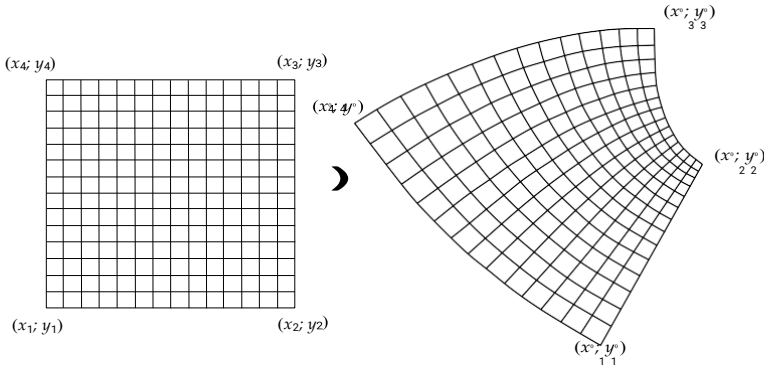


Fig. 2.1 — Conformal mapping: the perpendicularity of lines at their intersections is preserved.

Image Shift

System of equations (2.6) and coordinate transformation matrix T in the case of *image shift* $A = E = 1, B = D = 0$ will take the form:

$$\begin{cases} x' = x + C, \\ y' = y + D, \end{cases} \quad (2.7)$$

$$\Leftrightarrow \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} T \Rightarrow T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ C & F & 1 \end{bmatrix}, \quad (2.8)$$

where C and F are the offset along the axes Ox and Oy , respectively.

Listing 2.1. Shifting the image by 50 and 100 pixels along the axes Ox and Oy respectively. The size of the original drawing roadSign.jpg 300×300 pixels.

```
1 I = imread ( ' road Sign . jpg ' );
2 T = [1 0 0; 0 1 0; 50 100 1];
3 tform = affine2d ( T );
4 I_shift = imwarp ( I, tform , ...
5     ' Interp ', ' nearest ', ...
6     ' OutputView ', imref2d ( size ( I ), ...
7     [1 size ( I, 2 )], [1 size ( I, 1 )]) ...
8 );
```

The `affine2d()` function creates a transformation matrix that is applied to image I using the `imwarp()` function. The additional parameters specify the same coordinates for the original and transformed images, as well as the interpretation method for undefined pixels.

Image reflection

System of equations (2.6) and coordinate transformation matrix

T in case of *reflection* of the image along the *axis* Ox at $A = 1$,

$E = -1, B = C = D = F = 0$ will take the form:

$$\begin{cases} x' = x \\ y' = -y \end{cases} \Rightarrow T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.9)$$

Listing 2.2. Reflection relative to the Ox axis.

```
1 T = [1 0 0; 0 -1 0; 0 0 1];
2 tform = affine2d ( T );
3 I_reflect = imwarp ( I, tform );
```

Uniform image scaling

System of equations (2.6) and coordinate transformation matrix

T in the case of *image scaling* $A = a, E = \beta, B = C =$

$D = F = 0$ will take the form:

$$\begin{cases} x' = ax, a > 0 \\ y' = \beta y, \beta > 0 \end{cases} \Rightarrow T = \begin{bmatrix} a & 0 & 0 \\ 0 & \beta & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (2.10)$$

If $a < 1$ and $\beta < 1$, then the image is reduced if $a > 1$ and $\beta > 1$

If $a = \beta$, then the proportions will not be the same in width and height. In general, this mapping will be affine, not conformal.

Listing 2.3. Double the size of the original image.

```
1 T = [2 0 0; 0 2 0; 0 0 1];
2 tform = affine2d(T);
3 I_scale = imwarp(I, tform);
```

Image rotation

System of equations (2.6) and coordinate transformation matrix

T in case of *clockwise rotation* of the image $A = \cos \varphi,$

$B = -\sin \varphi, D = \sin \varphi, E = \cos \varphi, C = F = 0$ will take the form:

$$\begin{cases} x' = x \cos \varphi - y \sin \varphi \\ y' = x \sin \varphi + y \cos \varphi \end{cases} \Rightarrow T = \begin{bmatrix} \cos \varphi & \sin \varphi & 0 \\ -\sin \varphi & \cos \varphi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.11)$$

Listing 2.4. Rotating the image by $\varphi = 17^\circ$.

```
1 phi = 17 * pi / 180;
2 T = [ cos(phi) sin(phi) 0;
3      -sin(phi) cos(phi) 0;
4        0 0 1];
5 tform = affine2d(T);
6 I_rotate = imwarp(I, tform);
```



Fig. 2.2 — Conformal transformations, top row from left to right: original image, shift, reflection, rotation; bottom row: homogeneous scaling.

If $\varphi = 90^\circ$, then $\cos \varphi = 0$ and $\sin \varphi = 1$ and the expression (2.11) will become:

$$\begin{cases} x' = -y \\ y' = x \end{cases} \Rightarrow T = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad (2.12)$$

Affine mapping is a mapping in which parallel lines pass into parallel lines, intersecting into intersecting lines, intersecting into intersecting lines; the relations of the lengths of segments lying on the same line (or on parallel lines) and the relations of the areas of figures are preserved. The basic transformations are conformal transformations, bevels, and inhomogeneous scaling. An arbitrary affine transformation can be obtained by the sequential product of the matrices of basic transformations. In continuous geometry, any affine transformation has an inverse affine transformation, and the product of the direct and inverse gives a single transformation that leaves all points in place.

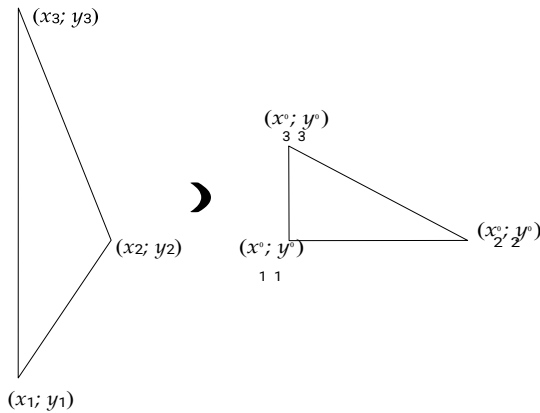


Fig. 2.3 — Affine mapping.

Image bevel

System of equations (2.6) and coordinate transformation matrix T in the case of *image bevel* $A = E = 1, B = S, C = D = F = 0$ will take the form:

$$\begin{cases} x' = x + sy \\ y' = y \end{cases} \Rightarrow T = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (2.13)$$

Listing 2.5. Image bevel, $S = 0.3$.

```
1 T = [1 0 0; 0.3 1 0; 0 0 1];
2 tform = affine2d(T);
3 I_bevel = imwarp(I, tform);
```

Piecewise linear transformations

A **piecewise linear mapping** is a mapping in which an image is divided into parts and then different linear transformations are applied to each of these parts.

Listing 2.6. An example of a piecewise linear mapping is where the left half of the image remains unchanged and the right half is stretched twice along the Ox axis.

```
1 imid = round(size(I,2) / 2);
```

```

2 I_left = I(:, 1: imid , :);
3 stretch = 2;
4 I_right = I(:, ( imid + 1: end ) , :);
5 T = [ stretch 0 0; 0 1 0; 0 0 1];
6 tform = affine2 d ( T);
7 I_scale = imwarp ( I_right , tform );
8 I_piecelinear = [ I_left I_scale ];

```



Fig. 2.4 — Left — bevel, right — piecewise linear extension.

Nonlinear transformations

When considering geometric transformations, it is assumed that the images were obtained using an ideal model of the cage. In fact, the formation of images is accompanied by various nonlinear distortions, see Fig. 2.5. Various nonlinear functions are used to correct them.

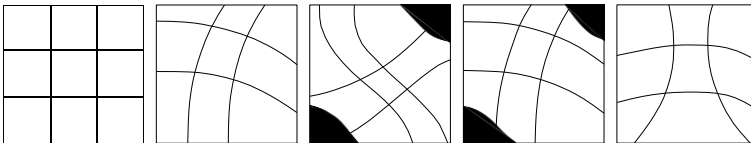


Fig. 2.5 — Examples of nonlinear distortions.

Projection Transform

A projection mapping is a mapping in which straight lines remain straight lines, but the geometry of the figure can be broken because the mapping does not generally preserve the parallelism of the lines. A property that is preserved in the projective transformation is the *collinearity of* points: three points lying on the same straight line (collinear), after

The transformations remain on one line. Projection reflection can be both *parallel* (the scale changes) and *projective* (the geometry of the figure changes).

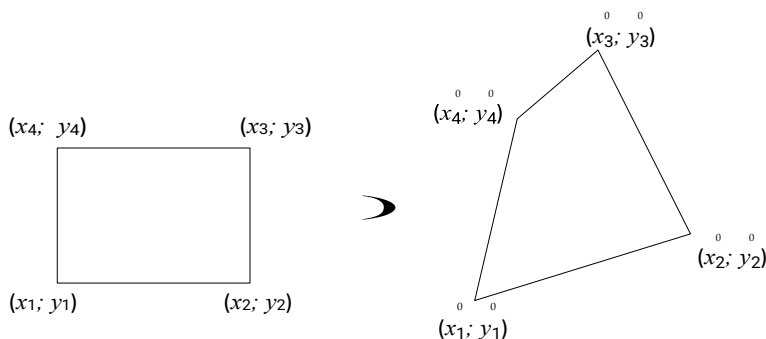


Fig. 2.6 — Projective mapping: lines remain straight, but parallel ones are mapped to intersecting ones.

In the case of projective mapping, the points of the three-dimensional scene P^3 are projected onto the two-dimensional plane P^2 of the image. This transformation $P^3 \rightarrow P^2$ represents the Euclidean point of the scene

$P = (X, Y, Z)$ (in homogeneous coordinates (X', Y', Z', W')) to the point of representation $X = (x, y)$ (in homogeneous coordinates (X', Y', W')). To find Cartesian coordinates of points from homogeneous coordinates.

use the following relations: $P = \begin{pmatrix} X' \\ Y' \\ Z' \\ W' \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \begin{pmatrix} W' \\ W' \\ W' \end{pmatrix}$

coordinates of the vector $\mathbf{P} \rightarrow X' = \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} X' \\ Y' \end{pmatrix} \frac{1}{W'}$ — vector coordinates $\mathbf{X}$.
and

Substituting in (2.2) $W = 1$ for vector $\mathbf{X}$, we get a system of equations:

$$\begin{cases} Ax + By + C \\ Gx + Hy + I \end{cases} = \begin{bmatrix} A & B & C \\ G & H & I \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow T = \begin{bmatrix} E & F \\ G & H & 1 \end{bmatrix} \quad (2.14)$$

Because of the normalization of the coordinates at w' , the projection mapping is generally nonlinear.

Listing 2.7. Example of projective mapping when $A = 1.1$, $B = 0.35$, $C = F = 0$, $D = 0.2$, $E = 1.1$, $G = 0.00075$, $H = 0.0005$, $I = 1$.

```
1 T = [1.1 0.35 0; 0.2 1.1 0; 0.00075 0.0005 1];
2 tform = projective2d ( T);
3 I_projective = imwarp (I, tform );
```

Polynomial transformation

A polynomial mapping is a representation of the original image using polynomials. In this case, the matrix of the transformation of the coordinates *of the coordinates* T will contain the coefficients of polynomials of the corresponding orders for *the* x and y coordinates.

$$\begin{cases} x' = a_1 + a_2x + a_3y + a_4x^2 + a_5xy + a_6y^2, \\ y' = b_1 + b_2x + b_3y + b_4x^2 + b_5xy + b_6y^2, \end{cases} \quad (2.15)$$

where x, y are the coordinates of points in one coordinate system; x', y' are the coordinates of these points in another coordinate system; $a_1 \dots a_6, b_1 \dots b_6$

– conversion coefficients.

Listing 2.8. An example of a polynomial mapping with a given transformation matrix T and that does not use MATLAB predefined functions. The resulting transformed image will not contain interpolated pixels.

```
1 [ numRows , numCols , Layers] = size ( I);
2 T = [0 0; 1 0; 0 1; 0.00001 0;
3      0.002 0; 0.001 0];
4 for k =1:1: Layers
5     for y =1:1: num Cols
6         for x =1:1: num Rows
7             xnew = round ( T (1,1)+ T (2,1)* x +...
8                           T (3,1)* y+T (4,1)* x ^ 2 +...
9                           T (5,1)* x* y+T (6,1)* y ^2);
10            ynew = round ( T (1,2)+ T (2,2)* x +...
```

```

11          T (3,2)* y+T (4,2)* x ^ 2 +...
12          T (5,2)* x*y+T (6,2)* y ^2);
13      I_polynomial( xnew , ynew , k) = ...
14          I(x,y, k);
15      end
16  end
17 end

```

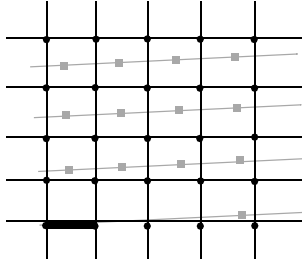


Fig. 2.7 — Offset of pixel coordinates in the transformed image.

As a rule, with nonlinear distortion, the coefficients $a_1 \dots a_6, b_1 \dots b_6$ and the coordinate transformation matrix T are unknown. In the case where the original image is available in addition to the distorted image, it is possible to calculate the conversion factors by finding pairs of corresponding points on the original and transformed images. The number of minimum required pairs of points is calculated using the formula:

$$t_{min} = \frac{(n+1)(n+2)}{2}, \quad (2.16)$$

where n is the conversion order.

If there is a second-order polynomial distortion, then according to (2.16) it is necessary to know the coordinates of at least *six pairs* of corresponding points before and after the transformation:

$(x_1, y_1) \dots (x_6, y_6)$ and $(x', y') \dots (x'_6, y'_6)$. According to (2.15) we will write down

Equations for calculating coefficients in matrix form:

$$\begin{bmatrix} 1 \\ x_1 \\ y_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1 y_1 & y_1^2 \\ 1 & x_6 & y_6 & x_6^2 & x_6 y_6 & y_6^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}^{-1} \begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \end{bmatrix}, \quad (2.17)$$

$$\begin{bmatrix} 1 \\ x_1 \\ y_1 \\ \vdots \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 & x_1^2 & x_1 y_1 & y_1^2 \\ 1 & x_6 & y_6 & x_6^2 & x_6 y_6 & y_6^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}^{-1} \begin{bmatrix} x'_1 \\ y'_1 \\ \vdots \end{bmatrix}. \quad (2.18)$$

To interactively specify the same points in the original and converted images, you can use the MATLAB `cpselect('source.jpg', 'transformed.jpg')` function, which returns two arrays `movingPoints` and `fixedPoints` with the coordinates of the converted and source images, respectively.

Sinusoidal distortion

Harmonic image distortion can be considered as another example of nonlinear transformation.

Listing 2.9. Example of sinusoidal image distortion.

```
1 [xi, yi] = meshgrid (1: ncols ,1: nrows );
2 imid = round ( size (I,2 )/ 2 );
3 u = xi + 20* sin (2* pi* yi /90);
4 v = yi;
5 tmap_B = cat(3 ,u, v);
6 resamp = makesampler(' linear ', ' fill ');
7 I_sinusoid = tformarray (I,[], resamp ,...
8     [2 1],[1 2],[], tmap_B ,.3);
```

Correction of distortion

When an image is formed by an optical system, distortion can occur on it. *Distortion* is an optical distortion expressed in the curvature of straight lines. Light rays



Fig. 2.8 — Polynomial distortion on the left, projective distortion in the center, and sinusoidal distortion on the right.

For example, the image of a square whose center is crossed by the optical axis has the form of a pillow (*pillow distortion*) in the case of positive distortion and the appearance of a barrel (*barrel distortion*) in the case of negative distortion (Fig. 1). 2.9).

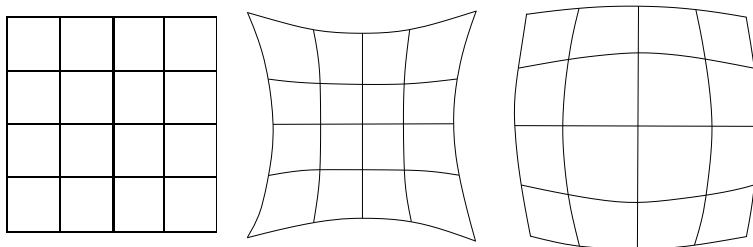


Fig. 2.9 — Examples of distortions. On the left is the original image, in the center is a pillow-shaped distortion, on the right is barrel-shaped.

Let $\mathbf{r} = (x, y)$ be a vector that defines two coordinates in a plane perpendicular to the optical axis. For an ideal image, all the rays that left this point and passed through

through the optical system, will reach the image point with **R coordinates**, which are determined by the formula:

$$\mathbf{R} = b_0 \mathbf{r}, \quad (2.19)$$

where b_0 is the coefficient of linear increase.

If there is a distortion of a higher order (for axisymmetric optical systems, distortion can only be of odd orders: third, fifth, seventh, etc.), then the corresponding terms should be added to expression (2.19):

$$\mathbf{R} = b_0 \mathbf{r} + F_3 r^3 \mathbf{r} + F_5 r^5 \mathbf{r} + \dots, \quad (2.20)$$

where r is the length of the vector $\mathbf{r}$; F_i , $i = 3, 5, \dots, n$ are the *n*-order *distortion coefficients* that contribute the most to the distortion of the image shape. In third-order distortion, if the coefficient F_3 has the same sign as b_0 : $\text{sign}(F_3) = \text{sign}(b_0)$, a pillow-like distortion occurs, otherwise it is a barrel distortion. In the first place, it is very common for people to be able

To correct distortion, the approach described above for projective mapping is used. An image of a regular grid and its distorted image are used, pairs of points in these images are found, and corrective transform coefficients are calculated.

Barrel distortion

Listing 2.10. Example of applying fifth-order barrel distortion to the original image.

```

1 [ xi , yi] = meshgrid (1: ncols ,1: nrows );
2 imid = round ( size (I ,2 )/ 2 );
3 xt = xi(:) - imid ;
4 yt = yi(:) - imid ;
5 [ theta , r] = cart2 pol( xt , yt);
6 F3 = .000001 ;
7 F5 = .0000012 ;
8 R = r + F3 * r.^ 2 + F5 * r.^ 4;
9 [ ut , vt] = pol2 cart( theta , R);
10 u = reshape ( ut , size ( xi)) + imid ;
11 v = reshape ( vt , size ( yi)) + imid ;

```

```

12 tmap_B = cat(3 , u, v);
13 resamp = makesampler(' linear ', ' fill ');
14 I_barrel = tformarray (I,[], resamp ,...
15     [2 1],[1 2],[], tmap_B ,.3);

```

The `meshgrid()` function is then used to form the image mesh. Then, on lines 2-4, a centered set of coordinates of all pixels is obtained. After that, all coordinates are converted by the `cart2pol()` function from the Cartesian coordinate system to the polar coordinate system, for more convenient use of the shape (2.20) through the radius vectors of each point. Then the pixel coordinates are converted using the `pol2cart()` functions and `reshape()` back to Cartesian coordinates. From the resulting mesh, a three-dimensional `tmap_B` transformation matrix is formed for the `tformarray` function, which performs the transformation of the original image according to the given mesh.

Pillow distortion

Listing 2.11. Example of applying a third-order pillow distortion to the original image.

```

1 F3 = -0.003;
2 R = r + F3 * r. ^ 2;

```



Fig. 2.10 — On the left — the original image, in the center — barrel distortion, on the right — pillow-shaped distortion.

"Stitching" images

Geometric transformations can be used, for example, to build a mosaic of several images. Mosaics ("stitching", "gluing") is the combination of two or more images into a single whole, and the coordinate systems of the glued

images may differ due to different camera angles, camera position changes, or the movement of the subject itself. However, both images must have overlapping areas, i.e. the same objects are present.

The main task of processing such images is to convert them into a common coordinate system. The system of the first image can be used as a common coordinate system, in order to find the transformation of the coordinates of all pixels in the second image (x, y) into a common coordinate system (x', y') . If polynomial distortion occurs, the system of equations (2.15) can be used to recalculate the coordinates. In the case of an affine mapping, system (2.15) will become:

$$\begin{cases} x' = a_1 + a_2x + a_3y, \\ y' = b_1 + b_2x + b_3y. \end{cases} \quad (2.21)$$

Therefore, it is necessary to find only 3 transformation factors for each coordinate:

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}^{-1} \begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix}, \quad (2.22)$$

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & x_1 & y_1 \\ 1 & x_2 & y_2 \\ 1 & x_3 & y_3 \end{bmatrix}^{-1} \begin{bmatrix} y'_1 \\ y'_2 \\ y'_3 \end{bmatrix}. \quad (2.23)$$

To do this, select the corresponding pairs of points in both images (three pairs in the case of affine distortion and at least six pairs in the case of polynomial distortion). You can perform this operation interactively using the MATLAB `cpselect()` function, or you can use special algorithmic rhythms. For example, you can determine these points based on the correlation coefficient, which is calculated in MATLAB using the `corr2()` function.

Let's consider a simple case of "gluing" two undistorted images of the same width. It is necessary to glue them vertically, i.e. add the second to the first from the bottom. However,

The gluing boundary is unknown. This problem can be implemented using the correlation approach.

Listing 2.12. An example of "merging" images. Reading the top and bottom images, respectively:

```
1 top Part = imread ( ' itmo Top . jpg ');
2 botPart = imread ( ' itmo Bot. jpg ');
```

For ease of determining the correlation coefficient, we convert color images into grayscale images:

```
3 top PartHT = im 2 double ( rgb 2 gray ( top Part ));
4 botPartHT = im 2 double ( rgb 2 gray ( botPart ));
```

Let's determine the size of the halftone images and initialize the intermediate arrays for calculations:

```
5 [ numRows , numCols , Layers] = size ( top PartHT );
6 [ num RowsBot , num ColsBot] = size ( botPartHT );
7 botPartCorrHT = zeros( intersecPart , num Cols );
8 top PartCorrHT = zeros( intersecPart , num Cols );
9 correlation Array = [];
```

Let's define the top line of the bottom image to calculate the correlation ratio with the lines of the upper one:

```
10 intersecPart = 5;
11 for j = 1:1: num Cols
12     for i = 1:1: intersecPart
13         botPartCorrHT (i, j) = botPartHT (i, j);
14     end
15 end
```

Let's compare the resulting string with the rows of the upper image and calculate the correlation coefficient:

```
16 for j = 0:1: numRows - intersecPart
17     for i = 1:1: intersecPart
18         top PartCorrHT (i,:) = top PartHT ( i+j ,:);
19     end
20     correlation Coefficient = ...
21         corr2 ( top PartCorrHT , botPartCorrHT );
22     correlation Array = ...
```

```

23         [ correlation Array
24         correlation Coefficient ];
25     correlation Coefficient = 0;
26 end
27 [M, I] = max( correlation Array );

```

Let's build a color "glued" image of the result_img with the gluing line based on the resulting index corresponding to the line number with the maximum correlation coefficient:

```

28 num RowsBotCorr = num RowsBot + I - 1;
29 for k = 1:1: Layers
30     for j = 1:1: num Cols
31         for i = 1: I-1
32             result_img (i,j, k) = ..
33                 top Part(i,j, k);
34         end
35         for i = I:1: num RowsBotCorr
36             result_img (i,j, k) = ...
37                 botPart(i- I+1 ,j, k);
38         end
39     end
40 end

```



Fig. 2.11 — Left — itmoTop.jpg, center — itmoBot.jpg, right
— result_img.

Procedure for performing work

1. *The simplest geometric transformations.* Choose an image of your choice. Perform linear and non-linear transformations (conformal, affine and projective mappings) on it.

2. *Correction of distortion.* Select a free image with either pillow-shaped or barrel-shaped distortion. Perform image correction.
3. *"Fusion" of images.* Select two images (camera images, fragments of a scanned image, etc.) on which there is an area of intersection. Correct the second image to convert it into the coordinate system of the first; then perform an automatic "fusion" of two images into one.

Contents of the report

1. Objective.
2. Theoretical substantiation of the applied methods and functions of geometric transformations.
3. Progress of work:
 - (a) Original images;
 - (b) Listings of software implementations;
 - (c) Comments;
 - (d) Resulting images.
4. Conclusions on the work done.

Questions to protect. How can I rotate an image without using a rotation matrix?

1. What is the minimum number of matching point pairs that need to be set in the original and distorted images if the conversion order is $n = 4$?
2. After the image is geometrically transformed, pixels with indeterminate intensity values may appear. What is the reason for this and how does it work?

Practical Lesson No3 Filtering and Selecting Paths

Purpose of the work

Mastering the basic methods of filtering images from noise and highlighting contours.

Methodological recommendations

Prior to starting work, students should be familiar with the basic features of the MATLAB image filtering environment. Have an understanding of low-frequency and high-pass filtering.

Theoretical information

Types of noise

Digital images obtained by various optical devices can contain a variety of distortions due to various types of interference, which are commonly referred to as *noise*. Noise in an image makes it difficult to process it by automatic means, and since noise can be of different nature, it is necessary to determine an adequate mathematical model for its successful suppression. Let's consider the most common noise models. In the MATLAB environment, noise can be superimposed on the image using the `imnoise()` function.

Impulsive Noise

In the case of impulsive noise, the signal is distorted by outliers with very large negative or positive values of short duration, and can occur, for example, due to decoding errors. Such noise leads to the appearance of white ("salt") or black ("pepper") dots in the image, which is why it is often called *point* noise. To describe it, it should be taken into account that the appearance of a noise emission in each pixel $I(x, y)$ does not depend on the quality of the original image, nor on the presence of noise at other points, and is likely to occur



Fig. 3.1 — Original halftone image.

p , and the intensity value of the pixel $I(x, y)$ will be changed to *the value* of $d \in [0.255]$:

$$I(x, y) = \begin{cases} d, & \text{with a probability of } p, \\ S_{x,y}, & \text{with probability } (1 - p) \end{cases} \quad (3.1)$$

where $S_{x,y}$ is the intensity of the pixel of the original image, I is the noisy image, if $d = 0$ is the "pepper" noise, if $d = 255$ is the "salt" noise. In the MATLAB environment, it is given by the 'salt & pepper' parameter of the `imnoise()` function: `imnoise(I, 'salt & pepper')`.

Additive noise

Additive noise is described by the following expression:

$$I_{new}(x, y) = I(x, y) + \eta(x, y), \quad (3.2)$$

where I_{new} — noisy image, I — original image, η is a signal-independent additive noise with a Gaussian or any other distribution of a probability density function.

Multiplicative noise

Multiplicative noise is described as follows:

e:

$$I_{new}(x, y) = I(x, y) \cdot \eta(x, y), \quad (3.3)$$

where I_{new} is the noisy image, I is the original image, η — signal-independent multiplicative noise, multiplying

Examples include *film grain, ultrasonic images, etc.* A special case of *multiplicative noise* is speckle noise, which appears in images taken by coherent imaging devices, such as medical scanners or radars. In the MATLAB environment, speckle noise is superimposed on the image by the `imnoise(I,'speckle')` function.

Gaussian (normal) noise

Gaussian noise in an image can occur as a result of insufficient illumination of the scene, high temperature, etc. The noise model is widely used in low-pass image filtering problems. Probability density distribution function

$p(z)$ of the random variable z is described by the following expression:

$$p(z) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}}, \quad (3.4)$$

where z — The intensity of the image (for example, for grayscale Images $z \in [0, 255]$), μ — average (mathematical expectation) A random variable z , σ — RMS deviation dispersion σ^2 determines the power of the introduced noise. Example- but 67% of the random variable value z is concentrated in the range zone $[(\mu - \sigma), (\mu + \sigma)]$ and about 96% in the range $[(\mu - 2\sigma), (\mu + 2\sigma)]$. In a MATLAB environment, noise can be set using the `imnoise(I,'gaussian')` or `imnoise(I,'localvar')` In the case of a left mathematical expectation.

Quantization noise

Quantization noise can lead, for example, to false contours around objects or to remove low-contrast details in the image. Quantization noise is not eliminated. Quantization noise can be approximated by the Poisson distribution and given by the `imnoise(I,'poisson')`.

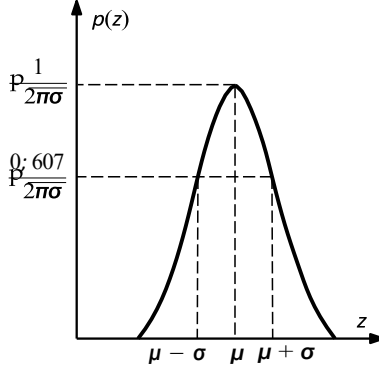


Fig. 3.2 — Probability density distribution function $p(z)$.

Image filtering

Let's consider the basic methods of filtering images. If the values of neighboring pixels in a certain neighborhood are taken into account to calculate the intensity value of each pixel, then this transformation is called *local*, and the neighborhood itself is a *window* that is a certain matrix called a *mask*, *filter*, *filter core*, and the values of the matrix elements themselves are called *coefficients*. The center of the mask is aligned with the analyzed pixel, and the mask coefficients multiply

As a rule, the mask has a square shape of size 3×3 , 5×5 , etc. Filtering an image I with dimensions $M \times N$ using a mask of size $m \times n$ is described by the formula:

$$I_{new}(x, y) = \sum_s \sum_t w(s, t) I(x + s, y + t), \quad (3.5)$$

where s and t are the coordinates of the mask elements relative to its center (in the center, $s = t = 0$).

$I_{new}(x, y)$ window w , in which the filter mask is described, is shifted and the intensity of the next pixel is calculated, so this transformation is called *sliding window filtering*.

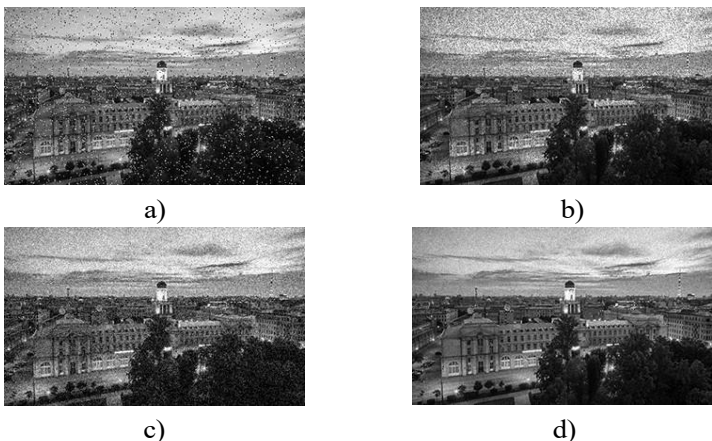


Fig. 3.3 — The result of the superposition: a) noise of the "salt" and "pepper" types, b) speckle noise, c) normal noise, d) Poisson noise.

In the MATLAB environment, image filtering can be done using the `filter2(mask,I)` function, where the mask matrix specifies the filter mask. The mask can be specified either manually or using the `fspecial()` function.

Low-Pass Filtering

Low-pass spatial filters attenuate high-frequency components (areas with a strong change in intensity) and leave the low-frequency components of the image unchanged. They are used to reduce the noise level and the distance of high-frequency components, which allows you to increase the accuracy of the study of the content of low-frequency components. As a result of the use of low-frequency filters, we get a smoothed or blurred image. The main distinguishing features are:

1. non-negative mask coefficients;
2. The sum of all coefficients is equal to one.

Examples of low-pass filter cores:

$$w = \frac{1}{9} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \quad w = \frac{1}{10} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 20 & 1 \\ 1 & 1 & 1 \end{bmatrix}. \quad (3.6)$$

Let's consider the main types of low-frequency smoothing filters.

Arithmetic Averaging Filter

This filter averages the pixel intensity value over the neighborhood using a mask with the same coefficients, for example, for a mask size 3 3 the coefficients are 1/9, for 5 5 - 1/25. The arithmetic averaging is achieved by using the following formula:

$$I_{new}(x, y) = \frac{1}{m \cdot n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} I(i, j), \quad (3.7)$$

where $I_{new}(x, y)$ is the pixel intensity of the filtered image, $I(i, j)$ is the pixel intensity of the original image in the mask, *and* m and n are the width and height of the filter mask, respectively. with a size 3 mask 3 can be done using the filter2(fspecial('average',3),I) function.

Geometric Averaging Filter

Geometric averaging is calculated using the form:

$$I_{new}(x, y) = \sqrt[m \cdot n]{\prod_{i=0}^{m-1} \prod_{j=0}^{n-1} I(i, j)}. \quad (3.8)$$

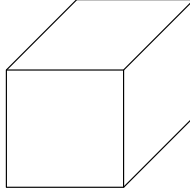


Fig. 3.4 — Graphical representation of *the box filter*.

The effect of this filter is similar to the previous method, but the individual objects in the original image are less distorted. The filter can be used to suppress high-frequency additive noise with better statistical characteristics than an arithmetic averaging filter.

Harmonic Averaging Filter

The filter is based on the expression:

$$I_{new}(x, y) = \frac{\sum_{i=0}^m \sum_{j=0}^n \frac{1}{I_{ij}}}{m \cdot n} \quad (3.9)$$

The filter suppresses salt-type noises well and does not work with pepper-type noises.

Counterharmonic Averaging Filter

The filter is based on the expression:

$$I_{new}(x, y) = \frac{\sum_{i=0}^m \sum_{j=0}^n \frac{(I_{ij})^{Q+1}}{(I_{ij})^{Q'}}}{m \cdot n \cdot \sum_{i=0}^m \sum_{j=0}^n \frac{(I_{ij})^{Q+1}}{(I_{ij})^{Q'}}} \quad (3.10)$$

where Q is the order of the filter. A counterharmonic filter is a generalization of averaging filters and at $Q > 0$ it *suppresses pepper noise*, and at $Q < 0$ it suppresses salt noise, but it is impossible to remove white and black dots at the same time.

Gaussian filter

The pixels in the sliding window that are closer to the pixel being analyzed should have a greater impact on the filtering result than the outermost ones. Therefore, the weighting coefficients of the mask can be described by the Gaussian bell function (3.4). A two-dimensional Gaussian filter is used to filter images:

$$G_{\sigma} = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \cdot \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{y^2}{2\sigma^2}}. \quad (3.11)$$

The larger the parameter σ , the more blurred the image. Typically, the filter radius $r = 3\sigma$. In this case, the size of the mask $2r + 1$ $2r + 1$ and matrix size $6\sigma + 1$ $6\sigma + 1$. Outside of this neighborhood, the Gaussian function values will be negligible. In MATLAB, image filtering with a Gaussian filter can be implemented using the `imgaussfilt(I)`.

Nonlinear filtering

Low-pass filters are linear and optimal when there is a normal distribution of interference in the digital image. Linear filters locally average the impulse interference by smoothing out the images.

Median Filtering

In the classic median filter, a mask with single coefficients is used. An arbitrary window shape can be specified using zero coefficients. The pixel intensity values in the window are represented as a column vector and sorted in ascending order. The filtered pixel is assigned the median (average) value of the intensity in the series. The number of the median element after sorting can be calculated using the formula $n = \frac{N+1}{2}$, where N is the number of pixels involved in the sorting. In MATLAB it can be implemented using the `medfilt2(I)` function.

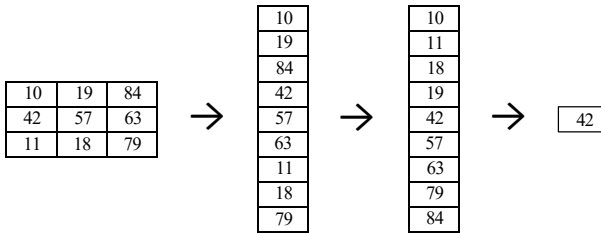


Fig. 3.5 — Illustration of the median filtering in window 3×3 .

Weighted median filtering

In this modification of median filtering, weights (numbers 2, 3, etc.) are used in the mask to reflect the greater influence on the filtering result of the pixels located closer to the filtered element. Median filtering removes impulsive noise as well, and does not introduce new intensity values on grayscale images. As the window size increases, the noise reduction capacity of the filter increases, but the outlines of objects begin to distort

n]], where the second argument [m n] of the medfilt2() function specifies a filter mask of size $m \times n$.

Adaptive median filtering

In this modification of the filter, the sliding window of size s is adaptively enlarged depending on the result of filtering. Let's denote the minimum, maximum, and median values of intensities in the window using z_{min} , z_{max} , and z_{med} are the intensity values of the pixel with coordinates (i, j) , and s_{max} is the maximum allowable window size. The adaptive median filtering algorithm consists of next steps:

1. Calculating the values of z_{min} , z_{max} , z_{med} , $A_1 = z_{med} - z_{min}$, $A_2 = z_{max} - z_{med}$ pixel (i, j) in a given window.
 - (a) If $A_1 > 0$ and $A_2 < 0$, go to step 2. Otherwise, increase the size of the window.

- (b) If the current window size *is* $s \leq s_{max}$, repeat step 1. Otherwise, the filter result is $z_{i,j}$.
2. Calculating *the values of* $B_1 = z_{i,j} - z_{min}$, $B_2 = z_{i,j} - z_{max}$.
 - (a) If $B_1 > 0$ and $B_2 < 0$, the filter result is $z_{i,j}$. Otherwise, z_{med} .
3. Change the coordinates (i,j) .
 - (a) If the image limit is not reached, go to the 1. Otherwise, the filtration is over.

The basic idea is to increase the window size until the algorithm finds a median value that is not the pulse noise or until it reaches the maximum window size. In the latter case, the algorithm will return the value $z_{i,j}$.

Rank filtering

A generalization of median filtering is a *rank filter* of order r , which selects a pixel with the number r [1, N] from the resulting vector-column of mask elements, which will be the result of filtering.

1. If the number of pixels in window N is odd and $r = \frac{N+1}{2}$, then the rank filter is *median*. In the case of ok- on 3 3, you can use the `ordfilt2(I,5,ones(3,3))` function in MATLAB.
2. If $r = 1$, the filter chooses the lowest intensity value and is called the *min filter*. In MATLAB, it is specified by the `ordfilt2(I,1,ones(3,3))` function.
3. If $r = N$, the filter selects the maximum intensity value and is called the *max filter*. In MATLAB, it is specified by the `ordfilt2(I,9,ones(3,3))` function.

Rank is sometimes written as a percentage, for example, for a *min filter*, the rank is 0%, the median is 50%, and the *max filter* is 100%.

Wiener filtration

Uses the pixel-adaptive Wiener method, based on statistical data estimated from the local neighborhood of each pixel.

$$\mu = \frac{1}{n \cdot m} = \sum_{i=0}^m \sum_{j=0}^n I(i, j), \quad (3.12)$$

$$\sigma^2 = \frac{1}{n \cdot m} \sum_{i=0}^m \sum_{j=0}^n I^2(i, j) - \mu^2, \quad (3.13)$$

$$I_{new}(x, y) = \mu + \frac{\sigma^2 - v^2}{\sigma^2} (I(x, y) - \mu), \quad (3.14)$$

where μ is the mean in the neighborhood, σ^2 is the dispersion, *and* v^2 is the dispersion of noise. In MATLAB, it is implemented using the `wiener2(l, [m n])`.

High-Pass Filtering

High-frequency spatial filters enhance high-frequency components (areas with a strong change in intensity) and attenuate low-frequency components of an image. They are used to highlight differences in intensity and define boundaries (contours) in images. For this purpose, the `edge()` function can be used in MATLAB.

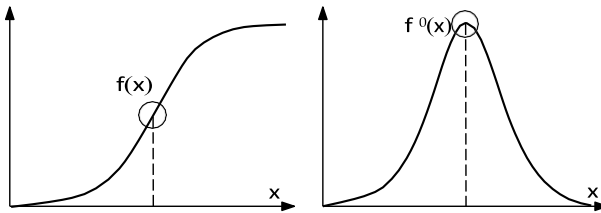


Fig. 3.6 — Intensity function and its first derivative, the maximum derivative corresponds to the edge.

High-pass filters approximate the computation of the producer by direction, with the increment of the argument Δx assumed to be 1 or 2.

1. filter mask coefficients can take negative values;
2. The sum of all coefficients is zero.

Roberts filter

The Roberts filter works with the minimum permissible derivative mask of dimension 2 for calculation, so it is fast and quite efficient. Possible mask options for finding the gradient along the Ox and Oy axes:

$$G_x = \begin{bmatrix} 1 & -1 \\ 0 & 0 \end{bmatrix}, G_y = \begin{bmatrix} 1 & 0 \\ -1 & 0 \end{bmatrix}, \quad (3.15)$$

or

$$G_x = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, G_y = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}. \quad (3.16)$$

As a result of the application of the Roberts differential operator, we obtain an estimate of the gradient in the directions G_x and G_y . The gradient modulus of all boundary detectors can be calculated from

formula $G = \sqrt{G_x^2 + G_y^2} = \sqrt{|G_x|^2 + |G_y|^2}$, but the direction of the gradient — according to the formula $\arctan \frac{G_y}{G_x}$. In MATLAB, using a differentiated

of the Roberts boundary can be distinguished using the `edge(I, 'Roberts')` function.

Prewitt filter

In this approach, two orthogonal masks of size 3 3 are used to calculate the derivatives of the Ox and Oy axes more accurately :

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}, G_y = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}. \quad (3.17)$$

In MATLAB, the Prewitt filter can select boundaries using the `edge(I, 'Prewitt')` function.

Sobel filter

This approach is similar to the Roberts filter, but different mask weights are used. A typical example of a Roberts filter is:

$$G_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}, \quad G_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}. \quad (3.18)$$

In MATLAB, the boundaries of the Sobel filter can be selected using the `edge(I, 'Sobel')` function.

Laplace filter

The Laplace filter uses an approximation of the second derivatives along the Ox and Oy axes, in contrast to the previous approaches that use the first derivative:

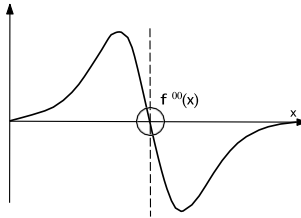


Fig. 3.7 — The second derivative of the brightness function changes sign (passes through zero at the point corresponding to the edge).

$$\Delta(x, y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \quad (3.19)$$

Formula 3.19 can be approximated by the following mask:

$$W = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}. \quad (3.20)$$

Canny's algorithm

One of the most common and effective algorithms for selecting contours in an image is the *Canny algorithm*. This algorithm allows you to determine not only edge pixels, but also connected boundary lines. The algorithm consists of the following steps:

1. Remove fine detail by smoothing the original image using a Gaussian filter.
2. Use of the Sobel differential operator to determine the gradient modulus values of all pixels in the image, with the result rounded in 45° increments.
3. Analysis of the values of the moduli of the gradient of the pixels located orthogonally to the examined pixel. If the value of the modulus of the gradient of the examined pixel is greater than that of the orthogonal neighboring pixels, then it is the *marginal value*, otherwise it is a *non-maximum*.
4. Perform a double threshold filtering of the marginal pixels selected in the previous step:
 - (a) If the value of the gradient modulus is above the threshold t_2 , then the presence of an edge in the pixel is valid.
 - (b) If the value of the gradient modulus is below the threshold t_1 , then the pixel is definitely not a boundary value.
 - (c) If the value of the gradient modulus lies within the range $[t_1, t_2]$, then such a pixel is considered *ambiguous*.
5. Suppression of all ambiguous pixels that are not associated with readily available pixels by 8-connectivity.

In MATLAB, the Canny algorithm can be used to define boundaries using the `edge(I, 'Canny')` function.

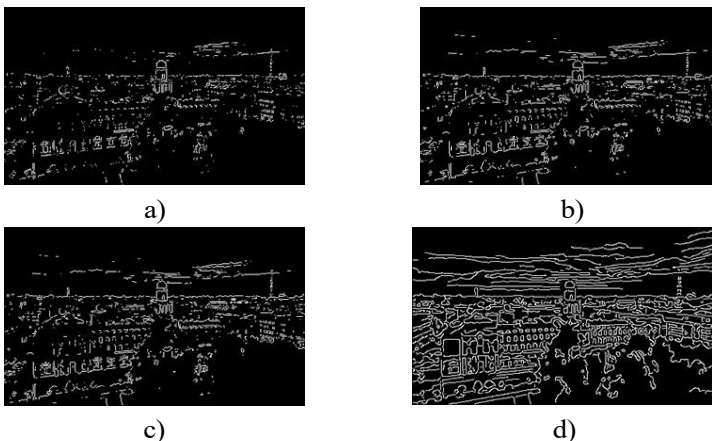


Fig. 3.8 — Result of boundary extraction: a) Roberts filter, b) Prewitt filter, c) Sobel filter, d) Canny's algorithm.

Procedure for performing work

1. *Types of noise.* Select a custom image. Get images distorted by various noises using the `imnoise()` function with parameters other than the default values.
2. *Low-pass filtering.* Process the distorted images obtained in the previous step with a Gaussian filter and a counterharmonic averaging filter with different values of the Q parameter.
3. *Nonlinear filtering.* Process the distorted images obtained in the first step with median, weighted median, rank and Wiener filtering at different mask sizes and coefficients. Implement adaptive median filtering.
4. *High-pass filtering.* Select the original image. Highlight the boundaries with Roberts, Prewitt, Sobel, Laplace, Canny filters.

Contents of the report

1. Objective.
2. Theoretical substantiation of the applied methods and functions of image filtering.
3. Progress of work:
 - (a) Original images;
 - (b) Listings of software implementations;
 - (c) Comments;
 - (d) Resulting images.
4. Conclusions on the work done.

Questions for the defense of the work

1. What are the main disadvantages of adaptive image filtering methods?
2. At what values of the Q parameter will a counterharmonic filter work as an arithmetic filter, and at what values will it work as a harmonic filter?
3. What operators can be used to distinguish the boundaries in the image?
4. Why is low-pass filtering usually performed in the first step of contour extraction?

Practical Lesson No4 Image Segmentation

Purpose of the work

Mastering the basic methods of segmenting images into semantic areas.

Methodological recommendations

Prior to starting work, students should be familiar with the basic features of the MATLAB framework for converting image color spaces and how to define thresholds.

Theoretical information

Image Binarization

The simplest way to segment an image into two classes (background pixels and object pixels) is *binarization*. Binarization can be performed by threshold or by double threshold. In the first case:

$$I_{new}(x,y) = \begin{cases} 0, & I(x,y) \leq t \\ 1, & I(x,y) > t \end{cases} \quad (4.1)$$

where I is the original image, I_{new} is the binarized image, and t is the binarization threshold. Binarization by this method in the MATLAB medium can be performed using the functions `im2bw()` (obsolete) or `imbinarize()`.

Listing 4.1. Binarization.

```
1 I = imread (' pic. jpg ');
2 L = 255;
3 t = 127 / L; %norm to 0...1
4 Inew = im2bw (I, t);
```

Dual threshold binarization (band binarization):

$$I_{new}(x,y) = \begin{cases} 0, & I(x,y) \leq t_1 \\ 1, & t_1 < I(x,y) \leq t_2 \\ 0, & I(x,y) > t_2 \end{cases} \quad (4.2)$$

where I is the original image, I_{new} is the binarized image, t_1 and t_2 are the upper and lower thresholds of binarization. Binarization by this method in the MATLAB environment can be performed using the `roicolor()` function. `rgb2gray()`.

Listing 4.2. Dual threshold binarization.

```
1 I = imread ( ' pic. jpg ');
2 t1 = 110;
3 t2 = 200;
4 Igray = rgb2gray ( I);
5 Inew = roicolor( Igray , t1 , t2 );
```

The binarization thresholds t , $t1$ and $t2$ can either be set manually or calculated using special algorithms. In the case of automatic threshold calculation, the following algorithms can be used.

1. Find the maximum I_{max} and minimum $I_{min \text{ intensity}}$ of the original grayscale image and find their arithmetic mean. The arithmetic mean will be the global threshold of binarization t :

$$t = \frac{I_{max} - I_{min}}{2}, \quad (4.3)$$

2. Find the optimal threshold t based on the brightness gradient modulus of each pixel. To do this, first calculate the gradient modulus at each point (x,y) :

$$G(x,y) = \max\{ |I(x+1,y) - I(x-1,y)|, |I(x,y+1) - I(x,y-1)| \}, \quad (4.4)$$

The optimal threshold t is then calculated:

$$t = \frac{\sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} A(x,y) G(x,y)}{\sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} G(x,y)}. \quad (4.5)$$

3. Calculation of the optimal *threshold* t by the statistical method of Otsu, dividing all pixels into two classes 1 and 2, minimizing the variance within each class $\sigma^2(t)$ and $\sigma^2(t)$

and maximizing variance between classes.

Algorithm for calculating the threshold using the Otsu method:

1. Calculation of the histogram of image intensities and

Strength $p_i = \frac{n_i}{N}$ for each intensity level, where

n_i is the number of pixels with intensity level i , N is the number of squeaks in the image.

2. Specifies the initial threshold $t = 0$ and the threshold k ($0, L$), which divides all pixels into two classes, where L is the maximum value of the image intensity. In the loop, for each threshold value from $k = 1$ to $k = L - 1$:

- (a) Calculation of the probabilities of the two classes ω_j (0) and the arithmetic mean μ_j (0), where $j = 1, 2$:

$$\omega_1(k) = \sum_{s=0}^k p_s \quad (4.6)$$

$$\omega_2(k) = \sum_{s=k+1}^L p_s = 1 - \omega_1(k), \quad (4.7)$$

$$\mu_1(k) = \sum_{s=0}^k s \cdot \frac{p_s}{\omega_1}, \quad (4.8)$$

$$\mu_2(k) = \sum_{s=k+1}^L s \cdot \frac{p_s}{\omega_2}, \quad (4.9)$$

- (b) Calculation of interclass variance $\sigma^2(k)$:

$$\sigma^2(\mathcal{K}) = \omega_1(\mathcal{K})\omega_2(\mathcal{K})(\mu_1(\mathcal{K}) - \mu_2(\mathcal{K}))^2. \quad (4.10)$$

- (c) If the calculated value of $\sigma^2(k)$ is greater than the current value of t , then assign the threshold to the value of interclass variance $t = \sigma^2(k)$.
3. The optimal threshold t corresponds to the maximum $\sigma^2(k)$.

In the MATLAB environment, the *threshold* t by the Otsu method can be computed using the `graythresh()` function:

Listing 4.3. Otsu binarization.

```
1 I = imread ( ' pic. jpg ');
2 t = graythresh ( I);
3 Inew = im2bw (I, t);
```

or using the `otsuthresh()` function based on the image histogram:

Listing 4.4. Histogram-based Otsu binarization.

```
1 I = imread ( ' pic. jpg ');
2 Igray = rgb 2 gray ( I);
3 [ counts , x] = imhist( Igray );
4 t = otsuthresh ( counts );
5 Inew = imbinarize ( Igray , t);
```

4. Adaptive methods that do not work with the entire image, but only with its fragments. Such approaches are often used when working with images that depict heterogeneously lit objects. In the MATLAB environment, the threshold t of the adaptive method can be calculated using the `adaptthresh()` function:

Code Listing 4.5. Adaptive binarization.

```
1 I = imread ( ' pic. jpg ');
2 Igray = rgb 2 gray ( I);
3 t = adaptthresh ( Igray );
4 Inew = imbinarize ( Igray , t);
```

In addition to the methods considered, there are many others, for example, the methods of Bernsen, Aykwell, Niblack, Janowitz and Bruckstein, etc.

Image segmentation

Let's consider several basic methods of image segmentation.

Based on the Weber principle

The algorithm is designed to segment halftone images. *The Weber principle* implies that the human eye does not perceive the difference in gray levels between $I(n)$ and $I(n) +$

$W(I(n))$, where $W(I(n))$ is Weber's function, n is the class number, I is a piecewise nonlinear function of grayscales. The Weber function can be calculated using the formula:

$$W(I) = \begin{cases} 20 - \frac{12I}{88}, & 0 \leq I \leq 88, \\ \frac{0,002(I - 88)^2}{7(I - 138)} + 13, & 88 < I \leq 255. \end{cases} \quad (4.11)$$

You can combine gray levels from the range $[I(n), I(n) + W(I(n))]$ by replacing them with a single intensity value.

The segmentation algorithm consists of the following steps:

1. Initializing Initial Conditions: First Class Number
 $n = 1$, gray level $I(n) = 0$.
2. Calculate the value of $W(I(n))$ according to Weber's formula and assign the value $I(n)$ to all pixels whose intensity is in the range $[I(n), I(n) + W(I(n))]$.
3. Find pixels with an intensity higher than $G = I(n) + W(I(n)) +$
1. If found, then increase the class number $n = n + 1$, assign $I(n) = G$ and move on to the second step. Otherwise, finish the work. The image will be segmented into n classes with an intensity of $W(I(n))$.

Segmentation of RGB images by skin color

A common principle of this approach is to determine the criterion for the proximity of pixel intensity to skin tone. It is quite difficult to describe skin tone analytically, since

its description is based on human perception of color, changes when lighting changes, differs from one ethnicity to another, and so on.

There are several analytical descriptions for images in the RGB color space that allow you to classify a pixel as skin if the following conditions are met:

$$\begin{cases}
 R > 95, \\
 G > 40, \\
 B < 20, \\
 \max R, G, B - \min R, G, B > 15, \\
 |R - G| > 15, \\
 R > G, \\
 R > B,
 \end{cases} \quad (4.12)$$

or

$$\begin{cases}
 R > 220, \\
 G > 210, \\
 B > 170, \\
 |R - G| \leq 15, \\
 G > B, \\
 R > B,
 \end{cases} \quad (4.13)$$

or

$$\begin{cases}
 r = \frac{R}{R+G+B}, \\
 g = \frac{G}{R+G+B}, \\
 b = \frac{B}{R+G+B}, \\
 \frac{r}{g} > 1.185,
 \end{cases} \quad (4.14)$$

$$\begin{cases}
 \frac{r \cdot b}{(r+g+b)^2} > 0.107, \\
 \frac{r \cdot g}{(r+g+b)^2} > 0.112.
 \end{cases}$$

Based on the CIE Lab color space

In the Lab color space, the value of lightness is separated from the value of the chromatic component of the color (tone, saturation).

The lightness is given by the L coordinate, which can range from 0 (dark) to 100 (light). The chromatic component of the color is given by two Cartesian coordinates: a (denoting the position of the color in the range from *green* (128) to *red* (127)) and b (denoting the position of the color in the range from *blue* (128) to *yellow* (127)). A binary image is obtained at zero values of the coordinates a and b . The idea of the algorithm is to divide a color image into segments of dominant colors.

As a starting point, we will choose the following color image:



Fig. 4.1 — Original color image.

First of all, to reduce the influence of light on the segmentation result, we convert the full-color image from the RGB color space to the **Lab** space.

Code Listing 4.6. Segmentation based on color space **Lab**.

```
1 I = imread (' pic2 . jpg ');
2 Ilab = rgb 2 lab ( I);
3 L = Ilab (: ,: ,1);
4 a = Ilab (: ,: ,2);
5 b = Ilab (: ,: ,3);
```

In the next step, you need to determine the number of colors into which the image will be segmented, and specify areas containing pixels of approximately the same color. Regions can be specified for each color interactively as polygons using the `roipoly()` function:

```
6 num Colors = 3;
7 sample Areas = false ([ size (I, 1)
```

```

8      size (I, 2) num Colors ]]);
9  for i=1:1: num Colors
10     [ BW , xi , yi] = roipoly ( I);
11     sample Areas (:,:, i) = roipoly (I, xi , yi);
12 end

```

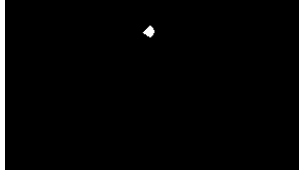


Fig. 4.2 — An example of a selected red area of four dots.

Next, you need to define the color labels for each of the segments by calculating the average chromaticity value in each selected area. The average values can be calculated using the `mean2()` function:

```

13 colorMarks = zeros ([ num Colors , 2]);
14 for i=1:1: num Colors
15     colorMarks(i,1) = ...
16         mean2 ( a( sample Areas (:,:, i)));
17     colorMarks(i,2) = ...
18         mean2 ( b( sample Areas (:,:, i)));
19 end

```

Next, we use the nearest neighborhood principle to classify the pixels by calculating Euclidean metrics between pixels and labels: the smaller the distance to the mark, the better the pixel corresponds to the segment. The Euclidean metric from the two color coordinates is calculated using the formula:

$$\sqrt{(a_{x,y} - a_{\text{mark}})^2 + (b_{x,y} - b_{\text{mark}})^2}. \text{ To find the minimum}$$

We'll use the `min()` function. Here's a listing to find label label labels for each pixel:

```

20 distance = zeros ([ size ( a), num Colors ]]);
21 for i=1:1: num Colors
22     distance (:,:, i) = ...

```

```

23      (( a- colorMarks(i,1)).^ 2 + ...
24      (b- colorMarks(i ,2)).^ 2 ).^ 0.5 ;
25      colorLabels = i;
26 end
27 [~, label] = min ( distance , [], 3);
28 label = colorLabels( label );

```

Thus, a label dimensionality matrix equal to the original image will contain class IDs for each pixel. To segment the image into segmentedFrames, use the following listing:

```

29 rgb Label = repmat( label , [1 1 3]);
30 segmented Frames = ...
31     zeros ([ size ( I), num Colors], ' uint8 ');
32 for i=1:1: num Colors
33     color = I;
34     color( rgb Label ~= colorLabels( i)) = 0;
35     segmented Frames (:,:, i) = color;
36 end

```



a)



b)

Fig. 4.3 — Segmented areas: a) red, b) yellow.

Note the distribution of segmented pixels on the coordinate plane (a,b) :

```

37 figure
38 for i=1:1: num Colors
39     plot( a( label == i), b( label ==i), ...
40         '.', ' MarkerEdgeColor ', ...
41         plotColors{ i}, ...
42         ' MarkerFace Color ', plotColors{ i});

```

```

43     hold on
44 end

```

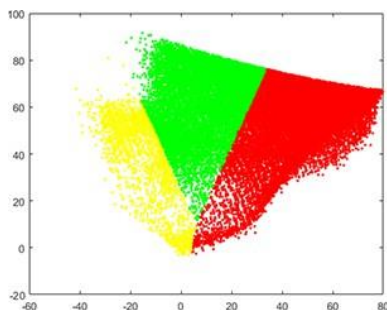


Fig. 4.4 — Distribution of segmented pixels on the coordinate plane (a, b) .

Based on k-means clustering

The idea of the method is to determine the centers *of k -clusters* *and assign to each cluster the pixels that are most closely related* to these centers. All pixels are considered as vectors x_i , $i = 1, p$. The segmentation algorithm consists of the following steps:

1. Random determination of k vectors m_j , $j = 1, k$, which are declared by the initial centers of clusters.
2. Update the values of the mean vectors m_j by calculating the distances from each *vector* x_i to each m_j and classifying them according to the criterion of the minimum distance from the vector to the cluster, recalculating the average values *of* m_j for all clusters.
3. Repeat steps 2 and 3 until the cluster centers stop changing.

The implementation of the method is very similar to the previous approach and contains a number of similar actions (we use the original image of Fig. 4.1). We're going to work in the Lab color space, so the first step is to move from **the RGB space** to the **Lab**:

Code Listing 4.7. Clustering-based segmentation

k-means.

```

1 I = imread ( ' pic2 . jpg ');
2 Ilab = rgb 2 lab ( I);
3 L = Ilab (: ,: ,1);
4 a = Ilab (: ,: ,2);
5 b = Ilab (: ,: ,3);

```

Consider the coordinate plane (a,b) . To do this, form a three-dimensional array ab , and then use the `reshape()` function to transform it into a two-dimensional vector containing all the pixels in the image:

```

6 ab (: ,: ,1) = a;
7 ab (: ,: ,2) = b;
8 nrows = size (I, 1);
9 ncols = size (I, 2);
10 ab = reshape ( ab , nrows * ncols , 2);

```

K-means clustering in the MATLAB environment is performed by the `kmeans()` function. Similar to the previous method, divide the image into three areas of the corresponding colors. Use the Euclidean metric (the 'distance' parameter with the value 'sqEuclidean' and repeat the clustering procedure three times to improve accuracy (the 'Replicates' parameter with the value 3)):

```

11 k = 3;
12 [ ids , centers] = kmeans( ab , k, ' distance ',...
13     ' sq Euclidean ', ' Replicates ', 3);
14 label = reshape ( ids , nrows , ncols );

```

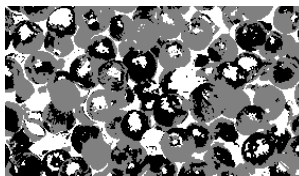


Fig. 4.5 — Class labels.

A label matrix of a size equal to the original image will contain the class IDs for each pixel. To segment the image into `segmentedFrames`, we use the following listing:



Fig. 4.6 — Segmented areas: a) red, b) yellow.

```

15 segmented Frames = cell(1 , 3);
16 rgb Label = repmat( label , [1 1 3]);
17 for i = 1:1: k
18     color = I;
19     color( rgb Label ~= i) = 0;
20     segmented Frames { i} = color;
21     figure , imshow ( segmented Frames { i});
22 end

```

The L array contains a value about the brightness of the image. Using this data, you can, for example, divide segmented red areas into light red and dark red segments.

Texture segmentation

In textural segmentation, three main methods are used to describe a texture: statistical, structural, and speckling. In the laboratory, we will consider a statistical approach that describes the texture of a segment as smooth, coarse, or grainy. 4.1. Let us consider an example of the image shown in Fig. 4.7, in which there are two types of textures. In general, they cannot be separated using simple binarization alone.

Let us consider image intensity I as a random variable z , which corresponds to the probability of the distribution $p(z)$ calculated from the image histogram. *Central*



Fig. 4.7 — Original halftone image.

The *angular moment* of the order n of the random variable Z is called the parameter $\mu_n(Z)$, calculated by the formula:

$$\mu_n(Z) = \sum_{i=0}^{L-1} (z_i - m)^n p(z_i), \quad (4.15)$$

where L is the number of intensity levels, m is the average value of the occasional value z :

$$m = \sum_{i=0}^{L-1} z_i p(z_i). \quad (4.16)$$

From 4.15 it follows that $\mu_0 = 1$ and $\mu_1 = 0$. For the description of the text, the dispersion of a random variable equal to the second moment is important

$\sigma^2(Z) = \mu_2(Z)$ and is a measure of luminance contrast that can be used to calculate *signs of smoothness*. Let's introduce the measure of relative smoothness R :

$$R = 1 - \frac{1}{1 + \sigma^2(Z)}, \quad (4.17)$$

which is zero for regions of constant intensity (zero dispersion) and approaches one for large dispersion values $\sigma^2(Z)$. For halftone images with an intensity interval $[0, 255]$, it is necessary to normalize the variance to the interval $[0, 1]$, since the dispersion values will be too large for the initial range.

Table 4.1 — Values of texture parameters.

Texture	m	s	$R \in [0,1]$
Smooth	82,64	11,79	0,0020
Rough	143,56	74,63	0,0079
Periodic	99,72	33,73	0,0170

Texture	$\mu_B(z)$	U	E
Smooth	-0,105	0,026	5,434
Rough	-0,151	0,005	7,783
Periodic	0,750	0,013	6,674

dispersion $\sigma^2(z)$ on $(L - 1)^2$. Texture characteristics are also often used *Standard Deviation*:

$$s = \sigma(z). \quad (4.18)$$

The third point is *the characteristic of the symmetry of the histogram*. To assess textural features, the *entropy function* E is used, which determines the spread of intensities of neighboring pixels:

$$E = - \sum_{i=0}^{L-1} \mu(z) \log_2 \mu(z). \quad (4.19)$$

Another important characteristic describing texture is *the measure of uniformity* U , which evaluates the uniformity of the histogram:

$$U = \sum_{i=0}^{L-1} \mu^2(z). \quad (4.20)$$

After calculating a feature or a set of features, it is necessary to build a binary mask, on the basis of which the image will be segmented. For example, you can use the entropy E in the vicinity of each pixel (x, y) . For this purpose, the MATLAB environment uses the `entropyfilt()` function, which defaults to a neighborhood of size 9×9 .

In the range from 0 to 1, we use the `mat2gray()` function, and to construct the mask, we binarize the resulting normalized array `Eim` using the Otsu method.

Listing 4.8. Texture segmentation.

```
1 I = imread ( ' pic3 . jpg ' );
2 E = entropyfilt( I);
3 Eim = mat2 gray ( E);
4 BW1 = imbinarize ( Eim , graythresh ( Eim ));
```



Fig. 4.8 — a) Entropy of the original image, b) Binarized image.

After that, we use morphological filters (to be discussed in more detail in Lab No6), first to remove connected areas containing less than a specified number of pixels (the `bwareaopen()` function), and then to remove internal *shape defects* or "holes" (the `imclose()` function with a structural element of size 9). Fill in the remaining large "holes" using the `imfill()`. Thus, we get a mask:

```
5 BWao = bwareaopen ( BW1 ,2000 );
6 nhood = true (9);
7 close BWao = imclose ( BWao , nhood );
8 Mask1 = imfill( closeBWao , ' holes ');
```

Applying the resulting mask to the original image, we highlight the segments of water and land.

Calculate the boundary between the textures using the `bwperim()` perimeter detection function:



Fig. 4.9 — a) The result of the execution of the `bwareaopen()` function; b) the result of the execution of the `imclose()` function.



Fig. 4.10 – a) The texture of the land, b) the texture of the water.

```

9 boundary = bwperim ( Mask1 );
10 segmentResults = I;
11 segmentResults ( boundary ) = 255;

```

A similar approach can be used to build a mask relative to sushi:

```

12 I2 = I;
13 I2 ( Mask1 ) = 0;
14 E2 = entropyfilt( I2 );
15 E2im = mat2gray ( E2 );
16 BW2 = imbinarize ( E2im , graythresh ( E2im ));
17 Mask2 = bwareaopen ( BW2 ,2000 );
18 boundary = bwperim ( Mask2 );
19 segmentResults = I;
20 segmentResults ( boundary ) = 255;

```

Let's find the textures of land and water:

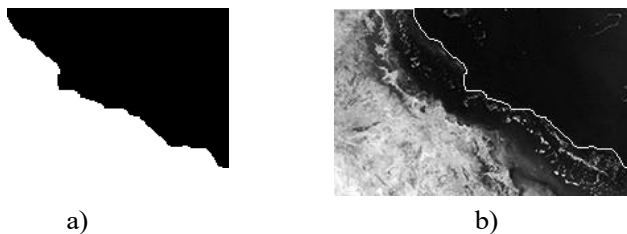


Fig. 4.11 — a) The result of the execution of the `imfill()` function, b) the selected border by the `bwperim()` function.

```

21 texture1 = I;
22 texture1 (~ Mask2 ) = 0;
23 texture2 = I;
24 texture2 ( Mask2 ) = 0;

```

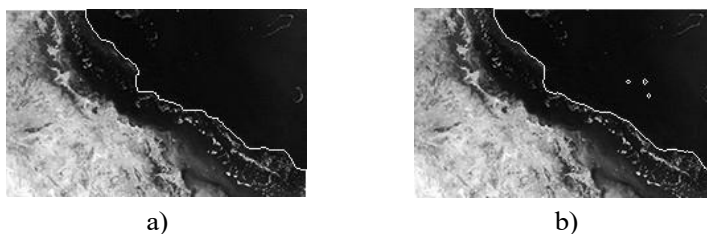


Fig. 4.12 — a) Result of segmentation relative to water, b) result of segmentation relative to land.

Procedure for performing work

1. *Binarization*. Select an arbitrary image. Binarize the image using the methods considered. Depending on the image, use binarization according to the upper or lower threshold.
2. *Segmentation 1*. Select a custom image containing the person(s). Segment the image either according to the Weber principle or based on skin color (optional).

3. *Segmentation 2. Select* an arbitrary image containing a limited number of colored objects. Segment the image in **CIE Lab** space using either the nearest neighbor method or the *k-means method (optional)*.
4. *Segmentation 3. Select* an arbitrary image containing two dissimilar textures. Perform texture segmentation of the image, evaluate at least three parameters of the selected textures, and determine which class the textures belong to.

Contents of the report

1. Objective.
2. Theoretical substantiation of the applied methods and functions of image segmentation.
3. Progress of work:
 - (a) Original images;
 - (b) Listings of software implementations;
 - (c) Comments;
 - (d) Resulting images.
4. Conclusions on the work done.

Questions for the defense of practical work

1. When is it advisable to use Weber's segmentation?
2. What are the values *of the a and b color coordinates of the* CIE Lab **color space** in a half-width image?
3. Why Segment in a Color Space **CIE Lab** and not in the original **RGB**?
4. What is *a color space* and *color gamut*?

Practical lesson No5

Hough Transform

Purpose of the work

Mastering the transformation to find geometric primitives.

Methodological recommendations

Prior to starting work, students should be familiar with the features of the Hough transform MATLAB environment. Be aware of the point "voting" approach.

Theoretical information

For example, this approach is used when constructing a triangle on three given sides, when first one side of the triangle is plotted, after which the ends of the segment are considered as the centers of circles with radii equal to the lengths of the second and third segments. The place where the two circles intersect is the common GMT, from where segments are drawn to the ends of the first segment. In other words, we can say that *two points were voted in* favor of the probable location of the third vertex of the triangle.

Let's generalize this idea for working with real data, when there are a large number of special characteristic points in the image that participate in voting. Suppose you need to find a circle of a known radius R in a binary point set, and in this set there may also be false points that do not lie on the desired circle. The set of centers of possible circles of the desired radius around each characteristic point forms a circle of radius R , see Fig. 5.2.

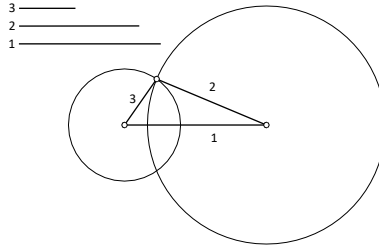


Fig. 5.1 — Drawing a triangle on three given sides.

Thus, the point corresponding to the maximum intersection of the number of circles will be the center of the circle of the radius.

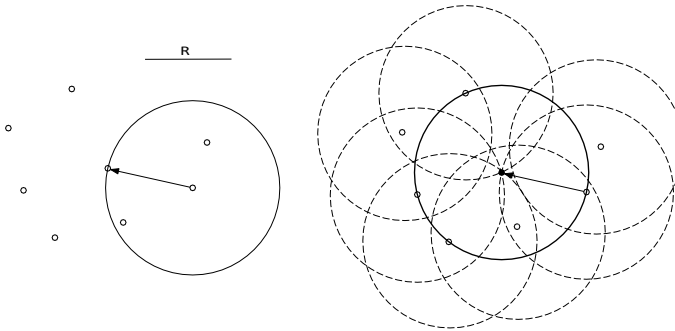


Fig. 5.2 — Detection of a circle of known radius in a point set.

The classical Hough transform, based on the considered idea of point voting, was originally intended to distinguish lines in binary images. In the Hough transformation, the parameter space is used to find geometric primitives. The most common parametric equation of lines is:

$$y = kx + b, \quad (5.1)$$

$$x \cos \Theta + y \sin \Theta = \rho, \quad (5.2)$$

where ρ is the radius vector drawn from the origin to the line; Θ is the angle of inclination of the radius vector.

In the Cartesian coordinate system, let the line be given by equation (5.1), from which it is easy to calculate the radius-vector ρ and the angle Θ (5.2). Then, in the space of Hough's parameters, the line will be represented by a point with coordinates (ρ_0, Θ_0) , see Fig. 5.3.

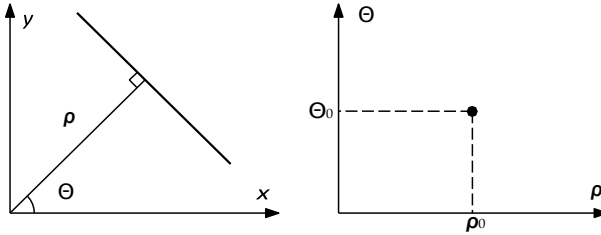


Fig. 5.3 — Representation of a line in Hough space.

The approach of the Hough transform is that for each point of the parameter space, the number of votes cast for it is summed up, so in a discrete form, the Hough space is called *an accumulator* and is a kind of matrix $A(\rho, \Theta)$ that stores information about voting. An infinite number of lines can be drawn through each point in the Cartesian coordinate system, the aggregate of which will generate a sinusoidal response function in the parameter space. Thus, any two sinusoidal response functions in the space of parameters will intersect at the point (ρ, Θ) only if the points generating them in the original space lie on a straight line, see Fig. 1. 5.4. Based on this, we can conclude that in order to find the lines in the source space, it is necessary to find all the local maxima of the battery.

The considered algorithm for finding lines can be used in the same way to find any other curve described in space by a certain function with a certain number of parameters $F = (a_1, a_2, \dots, a_n, x, y)$, which will affect only the dimension of the parameter space.

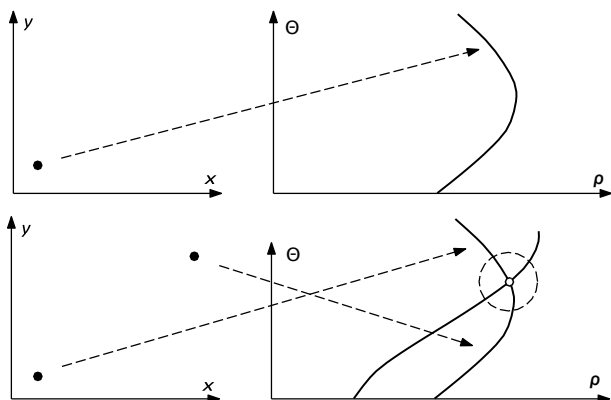


Fig. 5.4 — Voting procedure.

It is known that a circle on a plane is described by the formula $(x - x_0)^2 + (y - y_0)^2 = R^2$. The set of centers of all possible circles of radius R passing through a characteristic point forms a circle of radius R around that point, so the click function in the Hough transform to find circles represents a circle of the same size centered on the voting point. Then, similarly to the previous case, it is necessary to find the local maxima of the accumulator function $A(x, y)$ in the space of parameters (x, y) , which will be the centers of the circles sought.

The Hough transform is invariant to shift, scaling, and rotation. Given that in projective transformations of three-dimensional space, straight lines always turn into straight lines (in the degenerate case, into points), the Hough transform makes it possible to detect lines invariantly not only to affine transformations of the plane, but also to a group of projective transformations in space.

Let's give some image. Let's select the contours with Kenny's algorithm and perform the Hough transformation with the `hough()` function.

Listing 5.1. Searching for lines by the Hough transform.

```

1 I = imread ( ' pic. png ');
2 Iedge = edge (I, ' Canny ');
3 [H, Theta , rho ] = hough ( Iedge );
4 figure , imshow ( imadjust( mat2 gray ( H )) ,[] ,...
5     ' YData ', rho , ' XData ', Theta ,...
6     ' InitialMagnification ', ' fit ');
7 xlabel ( '\ rho '), ylabel ( '\ Theta ');
8 axis on , axis normal , hold on

```

Let's calculate the peaks with the `houghpeaks()` function in the Hough space and plot them on the resulting image of the response functions:

```

9 peaks = houghpeaks(H ,100 , ' threshold ',...
10     ceil (0.5* max( H (:)))));
11 x = Theta ( peaks (: ,2));
12 y = rho ( peaks (: ,1));
13 plot(x,y,'s',' color ', ' white ');

```

Let's define the lines of the `houghlines()` function based on the peaks and plot them on the original image:

```

14 lines = houghlines( Iedge , Theta , rho , peaks ,...
15     ' FillGap ',5 , ' Min Length ',10);
16 figure , imshow ( I), hold on
17 for k = 1: length ( lines)
18     xy = [ lines( k). point1 ; lines( k). point2 ];
19     plot( xy (:,1), xy (:,2), ' LineWidth ',2 ,...
20         ' Color ', ' green ');
21 end

```

To find circles using the Hough transform, you can use the `imfindcircles(I,R)` function.

Procedure for performing work

1. *Search for lines.* Select three arbitrary images containing lines. Search for lines using the Hough transform for both the original image and the image obtained using a differential operator. Reflect the found lines on the original image. Mark the points on the

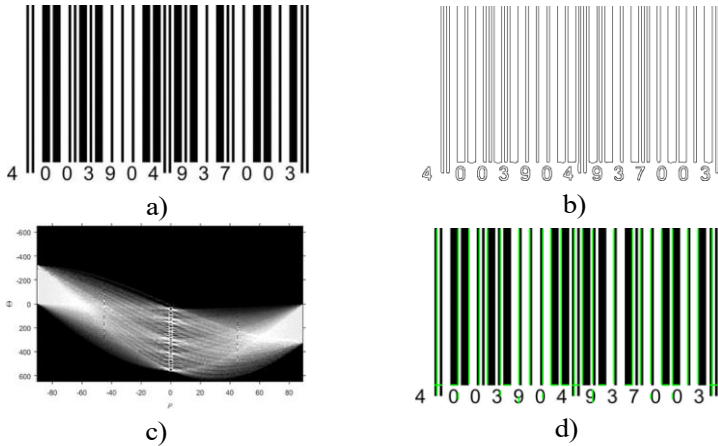


Fig. 5.5 — a) Original image, b) processed by Canny's algorithm, c) parameter space, d) selected lines.

and the end of the lines. Determine the lengths of the shortest and longest lines, calculate the number of lines found.

2. *Search for circles.* Select three arbitrary images containing circles. Search for circles of both a defined radius and a specified range using the Hough transform for both the original image and the image obtained using a differential operator. Display the circles found in the original image.

Contents of the report

1. Objective.
2. Theoretical substantiation of the applied transformation for the search for geometric primitives.
3. Progress of work:

- (a) Original images;
 - (b) Listings of software implementations;
 - (c) Comments;
 - (d) Resulting images.
4. Conclusions on the work done.

Questions for the defense of practical work

1. What is the idea behind Hough's transformation?
2. Can the Hough transform be used to find arbitrary paths that cannot be described analytically?
3. What are *the recurrent* and *generalized* Hough transformations?
4. What are the methods of parameterization in the Hough transformation?

Practical Lesson No6 Morphological Analysis of Images

Purpose of the work

Mastering the principles of mathematical morphology in the field of image processing and analysis.

Methodological recommendations

Before starting work, students should familiarize themselves with the functions of the MATLAB environment for working with binary morphology. Know the basic operations and provisions of binary morphology.

Theoretical information

Morphological analysis is used in many fields of knowledge, including image processing. Morphological analysis is a relatively universal approach, so it stands apart from all the methods considered earlier. Using mathematical morphology in image processing, it is possible to filter noise, segment objects, highlight contours, search for a given object in the image, calculate the "skeleton" of the image, etc.

1. Dilation (extension, build-up): $A \oplus B$, in MATLAB performed by the function `imdilate(A,B)`, extends the binary image A with structural element B ;
2. Erosion (compression, narrowing): $A \ominus B$, in MATLAB performed by the `imerode(A,B)` function, narrows the binary image A with the structural element B ;

3. Opening (unlocking, opening, opening): $(A \oslash B) \oslash B$, in MATLAB is performed by the `imopen(A,B)`, removes external defects in the binary image A structural element B ;
4. Closure (closure): $(A \oslash B) \oslash B$, in MATLAB is performed by the `imclose(A,B)`, removes internal defects in the binary image A structural element B ;

where $\oplus$ and $\ominus$ is Minkowski addition and subtraction, respectively. MATLAB has the following useful functions, often used in conjunction with morphological operations:

1. `strel()` — specifies a structural element;
2. `bwmorph(A,operation,n)` - Applies morphological operation filtering to an A image n times;
3. `bwhitmiss(A,B,C)` performs a "hit & miss" operation that identifies pixels whose neighborhoods are the same as structural element B and whose neighbors are not the same as structural element C.
4. `bwareaopen(A,dim)` — removes objects in image A that contain less than dim pixels;
5. `bwselect()` — selects certain objects in the image;
6. `bwarea()` — calculates the area of objects;
7. `bwlabel()` — marks connected objects of a binary image;
8. `bweuler()` — calculates the Euler number of the binary image.

Examples of the use of morphological operations

Selecting object boundaries

Object boundaries can be identified using the following approach:

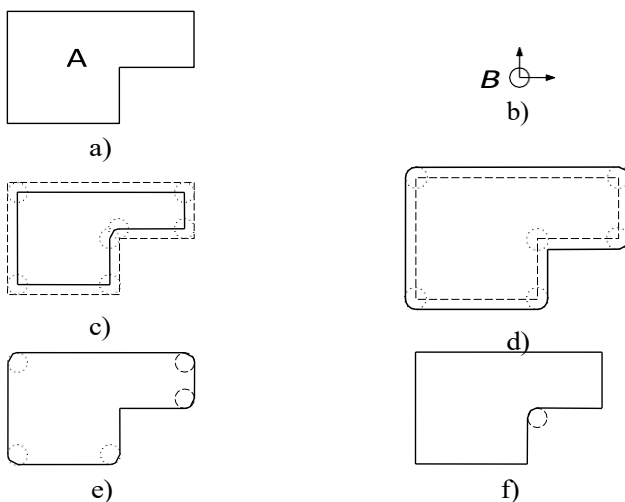


Fig. 6.1 — The result of performing binary morphology operations: a) binary image A , b) disk structural element B , c) dilation operation, d) erosion operation, e) opening operation, f) closing operation.

1. $C = A - (A \ominus B)$ is the formation of the internal contour;
2. $C = (A \oplus B) - A$ is the formation of the outer contour.

Separation of "glued" objects

One example of using morphological operations on an image is the problem of separating objects glued together in an image. The problem can be solved with a sufficient degree of accuracy by sequentially executing a compression filter several times, and then expanding the result as much as possible. The intersection of the original image with the processed image will allow you to separate the glued objects.

Listing 6.1. Separation of "glued" objects.

```
1 I = imread ( ' pic. jpg ');
2 t = graythresh ( I);
```

```

3 Inew = im2bw (I, t);
4 Inew = ~ Inew ;
5 BW2 = bwmorph ( Inew , ' erode ', 7);
6 BW2 = bwmorph ( BW2 , ' thicken ', Inf);
7 Inew = ~( Inew & BW2 );

```

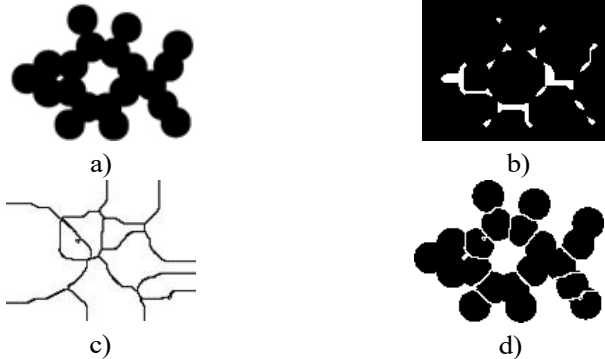


Fig. 6.2 — Separation of "glued" objects: a) original image, b) erosion of the binary image, c) expansion of objects, d) result of separation.

Segmentation using the managed divide method

In the watershed segmentation approach, an image is viewed as a map of heights, in which pixel intensities describe heights relative to a certain level. Rain "pours" on such "high-altitude terrain", forming many *catchment basins*. Gradually, water from overflowing basins overflows, and the basins merge into larger ones, see Fig. 6.3. 6.3. The junctions of basins are marked as watershed lines. If the rain is stopped, then the image will be segmented into small areas, and if it is too late, into large areas. In this approach, all pixels are divided into three types:

1. *local lows*;
2. located *on a slope* (from which water rolls down to the same local minimum);
3. *local maxima* (from which water rolls down to more than one minimum).

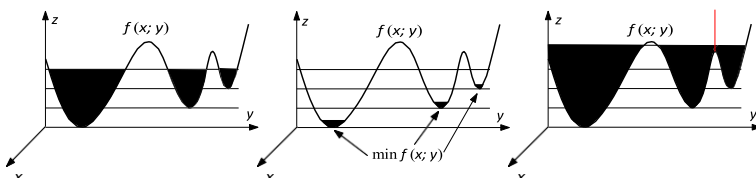


Fig. 6.3 — Watershed of regions.

In implementing this method, it is necessary to define watersheds and watershed lines by treating local areas and calculating their characteristics. The segmentation algorithm consists of the following steps:

1. Calculation of the segmentation function. Typically, a gradient representation of the image is used for this.
2. Calculate foreground markers based on the pixel connectivity of each object.
3. Compute background markers that represent pixels that are not objects.
4. Modification of the segmentation function to take into account the relative position of the foreground and background markers.

As a result of the algorithm's work, a mask will be obtained, where the peaks of identical segments will be marked with the same labels and will form a connected area. Let's assume that the image of Fig. 6.4.

Let's perform morphological filtering of the image using basic morphological operations and morphological reconstruction `imreconstruct()`. The `imcomplement()` function



Fig. 6.4 — Original image.

calculates the image complementary to the argument image and is an inverted image.

Listing 6.2. Watershed segmentation.

```
1 rgb = imread ( ' pic. jpg ' );
2 A = rgb 2 gray ( rgb );
3 B = strel( ' disk ',6);
4 C = imerode (A, B);
5 Cr = imreconstruct(C, A);
6 Crd = imdilate ( Cr , B);
7 Crdr = imreconstruct( imcomplement( Crd ), ...
8     imcomplement( Cr));
9 Crdr = imcomplement( Crdr );
```

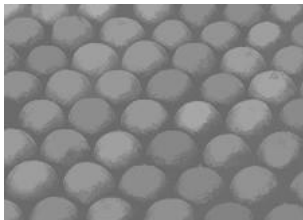


Fig. 6.5 — Filtered image.

Next, define the local maxima with the `imregionalmax()` function to define foreground *markers* (fgm) and, for clarity, overlay the markers on the original image:

```

10 fgm = imregionalmax ( Crdr );
11 A2 = A;
12 A2 ( fgm ) = 255;

```

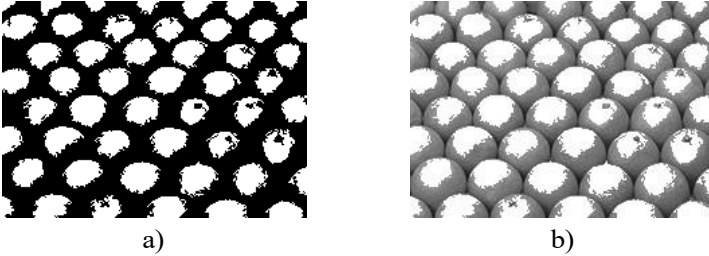


Fig. 6.6 — Foreground markers: a) computed, b) superimposed on the original image.

As you can see from the images presented, the markers are heavily cut. To smooth out the foreground markers, we will use the following sequence of morphological operations:

```

13 B2 = strel( ones (5 ,5));
14 fgm = imclose ( fgm , B2 );
15 fgm = imerode ( fgm , B2 );
16 fgm = bwareaopen ( fgm ,20);
17 A3 = A;
18 A3 ( fgm ) = 255;

```

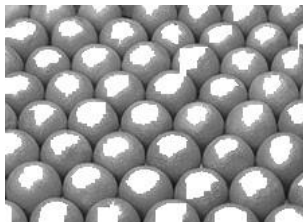


Fig. 6.7 — Filtered foreground markers.

The next step is to define *background markers* (bgm). To do this, binarize the filtering

In the Crdr image, find the Euclidean distance from each black pixel to the nearest white pixel using the `bwdist()` function, and compute the matrix `L` containing the labels of the segments obtained by the watershed method using the `watershed()` function:

```
19 bw = imbinarize ( Crdr );
20 D = bwdist( bw );
21 L = watershed ( D);
22 bgm = L == 0;
```

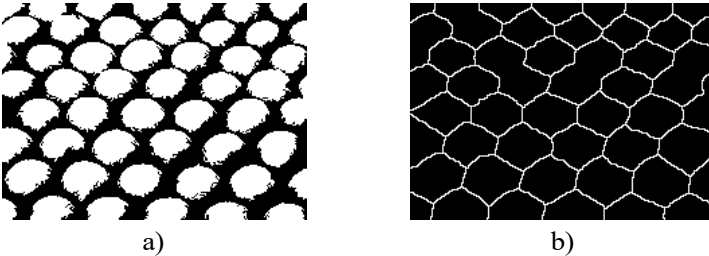


Fig. 6.8 — a) binarized image, b) background markers.

Then you need to modify the segmentation function. To do this, you can use the `imimposemin()` function, which accurately determines the local minima of the image. Thanks to this, the function corrects the values of the gradients on the image and refines the location of the foreground and background markers.

```
23 hy = fspecial('sobel ');
24 hx = hy';
25 Ay = imfilter( double ( A), hy, 'replicate ');
26 Ax = imfilter( double ( A), hx, 'replicate ');
27 grad = sqrt( Ax.^2 + Ay.^2);
28 grad = imimposemin ( grad , bgm | fgm );
```

After that, the segmentation operation of the refined gradient representation of the image based on the watershed is performed and the result of the algorithm is visualized:

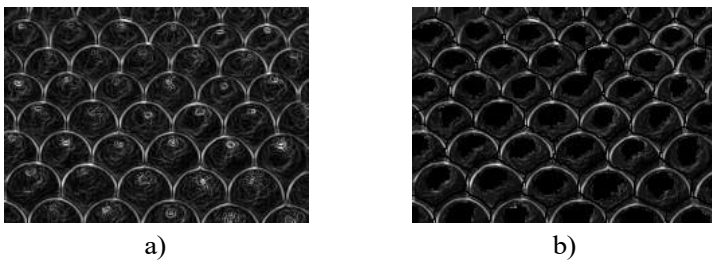


Fig. 6.9 — Gradient representation of the image: a) original, b) modified.

```

29 L = watershed ( grad );
30 A4 = A;
31 A4 ( imdilate ( L == 0 , ...
32     ones ( 3 , 3 ) ) | bgm | fg ) = 255;
33 Lrgb = label2 rgb ( L, 'jet ', 'w', ' shuffle ');

```

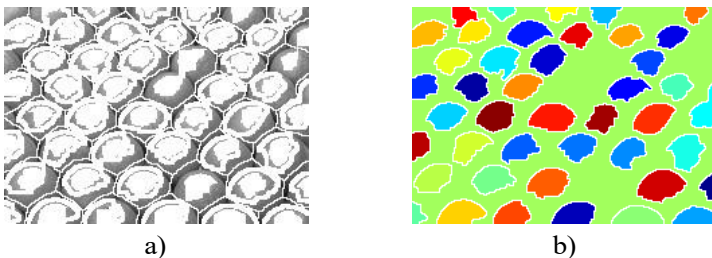


Fig. 6.10 — Markers and borders superimposed on: a) the original image, b) presented in rgb colors.

Procedure for performing work

1. *Basic morphological operations.* Select an arbitrary image containing shape defects (internal "holes" or external "protrusions") of objects. Using basic morphological operations, completely remove or minimize defects.

2. *Separation of objects.* Select an arbitrary binary image containing overlapping objects. Use binary morphology operations to separate objects.
3. *Segmentation.* Select an arbitrary image containing a small number of local minimums. Segment the image by watersheds.

Contents of the report

1. Objective.
2. Theoretical substantiation of mathematical morphology for image analysis.
3. Progress of work:
 - (a) Original images;
 - (b) Listings of software implementations;
 - (c) Comments;
 - (d) Resulting images.
4. Conclusions on the work done.

Questions for the defense of practical work

1. Does the opening result include the closing result?
2. What morphological filter should be applied to remove the protrusions from the object?
3. How can you find the contour of an object with the help of morphological operations?
4. What is *morphology*?

MINISTRY OF SCIENCE AND HIGHER EDUCATION OF THE RUSSIAN
FEDERATION

FEDERAL STATE AUTONOMOUS EDUCATIONAL INSTITUTION OF HIGHER EDUCATION
"NORTH CAUCASUS FEDERAL UNIVERSITY"

SUGGESTIONS FOR TEACHING

**on the organization and conduct of independent work of
students of the direction of training**

11.03.02 Infocommunication Technologies and Communication Systems,
Specialization "Intelligent Data Processing in Infocommunication
Systems and Networks"
in the discipline "Digital Image Processing"

Stavropol, 2026

CONTENTS

1. Independent work as the most important form of the educational process
2. Goals and main objectives of the ACS
3. Types of independent work
4. Organization of SRS
5. General recommendations for the organization of independent work
6. Independent work of a student is a necessary link in the formation of a researcher
7. Methodical recommendations for students on certain forms of independent work
8. Educational, methodological and information support of the discipline

1. Independent work as the most important form of the educational process

Independent work is the planned educational, research, research work of students, carried out in extracurricular (classroom) time on the assignment and under the methodological guidance of the teacher, but without his direct participation (with the partial direct participation of the teacher, leaving the leading role for the work of students).

It is necessary to familiarize yourself with the technological map of the student's independent work.

№	Section (topic) of the discipline and summary	Emerging competencies, indicators	Independent work, hours	Forms of current control Academic Performance
1	Modern computer systems Modeling of Technical Objects 1.1. Basic Concepts and Composition of the Computer Simulation 1.2. Levels (hierarchy) of the computer modeling system 1.3. System Engineering Modeling of Computer Vision Systems 1.4. Quality criteria of the product being developed 1.5. Generalized algorithm of the computer modeling process technical facility	PC-1 ID-1PK-1 ID-2PK-1 ID-3PK-1 PC-3 ID-1PK-3 ID-2PK3 ID-3PK-3 ID-4PK-3 ID-5PK-3 ID-6PK-3 ID-7PK-3 ID-8PK-3	6,00	Interview
2	Methods of Modeling Technical Systems 2.1. Overview of Technical Modeling devices	PC-1 ID-1PK-1 ID-2PK-1 ID-3PK-1	6,00	Protection Lab work

	2.2. Types of models of technical systems 2.3. Main types of mathematical models 2.4. Computer Simulation at the Stage of System Engineering Design of Optoelectronic Devices	PC-3 ID-1PK-3 ID-2PK3 ID-3PK-3 ID-4PK-3 ID-5PK-3 ID-6PK-3 ID-7PK-3 ID-8PK-3		
3	Basic concepts of computer simulation of computer vision systems 3.1. Trends in the development of computer vision systems 3.2. Analytical Review of Worldview Concepts of Computer Modeling of Technical Systems 3.2.1. Epistemological aspects of modeling 3.2.2. Mathematical Modeling of Technical Systems 3.2.3. Mathematical description of the object of modeling 3.2.4. Simulation modeling... 3.3. Basic principles of computer modeling of systems Computer Vision	PC-1 ID-1PK-1 ID-2PK-1 ID-3PK-1 PC-3 ID-1PK-3 ID-2PK3 ID-3PK-3 ID-4PK-3 ID-5PK-3 ID-6PK-3 ID-7PK-3 ID-8PK-3	6,00	Protection Lab work
4	Elementary Introduction to the Theoretical Foundations of Methods of Mathematical Modeling of Computer Vision Systems 4.1. Description of two-dimensional continuous images	PC-1 ID-1PK-1 ID-2PK-1 ID-3PK-1	6,00	Protection Lab work

	4.2. Fundamentals of the Theory of Linear Filtering of Image Systems 4.3. Image correlation analysis 4.4. Image noise 4.5. Digital Image Representation	PC-3 ID-1PK-3 ID-2PK3 ID-3PK-3 ID-4PK-3 ID-5PK-3 ID-6PK-3 ID-7PK-3 ID-8PK-3		
5	Detection and Object Recognition Observations in the image 5.1. Methods for detecting and recognizing objects in an image 5.2. Statistical recognition methods 5.3. Criteria for detecting and recognizing objects	PC-1 ID-1PK-1 ID-2PK-1 ID-3PK-1 PC-3 ID-1PK-3 ID-2PK3 ID-3PK-3 ID-4PK-3 ID-5PK-3 ID-6PK-3 ID-7PK-3 ID-8PK-3	6,00	Protection Lab work
6	Selecting and analyzing related areas 6.1. Selecting Related Areas in Binary Images 6.2. Methods for segmenting grayscale images 6.3. Correlation Image Processing Algorithms	PC-1 ID-1PK-1 ID-2PK-1 ID-3PK-1 PC-3 ID-1PK-3 ID-2PK3 ID-3PK-3 ID-4PK-3 ID-5PK-3 ID-6PK-3 ID-7PK-3 ID-8PK-3	6,00	Protection Lab work

	Total for the semester		24,00	
	TOTAL		24,00	

Independent work of students at the university is an important type of educational and scientific activity of the student. Independent work of students plays a significant role in the rating technology of education. Education at the university includes two parts that are almost the same in volume and mutual influence - the process of learning and the process of self-learning. Therefore, the SRS should become an effective and purposeful work of the student.

The concept of modernization of Russian education defines the main tasks of vocational education - "training a qualified worker of the appropriate level and profile, competitive in the labor market, competent, responsible, fluent in his profession and oriented in related fields of activity, capable of effective work in his specialty at the level of world standards, ready for constant professional growth, social and professional mobility".

The solution of these problems is impossible without increasing the role of students' independent work on educational material, strengthening the responsibility of teachers for the development of independent work skills, for stimulating

professional growth of students, fostering creative activity and initiative.

Society imposes a fairly wide range of requirements on a modern specialist, among which it is important that graduates have certain abilities and ability to independently obtain knowledge from various sources, systematize the information received, and assess a specific financial situation. The formation of such a skill occurs during the entire period of study through the participation of students in practical classes, the performance of control tasks and tests, the writing of term papers and graduation qualification papers. At the same time, the independent work of students plays a decisive role in the entire educational process.

The forms of independent work of students are diverse. In this discipline "Digital Image Processing" the following are provided:

The use of the material of the educational and methodological complex of disci- Plin

At the first stage, it is necessary to familiarize yourself with the mandatory minimum content of the basic educational program for the discipline studied. Next, it is necessary to study the work program of the discipline, which considers the content of the topics of the lecture course, the relationship between the topics of lectures and laboratory classes, topics and types of independent work. For each type of independent work, certain forms of reporting are provided, presented in the curriculum for the discipline "Data Bases".

Work with literature

For the successful development of the discipline, it is necessary to independently study the presented topics according to the recommended sources of information that are presented in the curriculum.

Independent work introduces students to scientific creativity, the search for and solution of urgent modern problems.

2. Goals and main objectives of the ACS

The leading goal of the organization and implementation of the SRS should coincide with the goal of the student's education - the preparation of a bachelor with higher education. In the organization of the SRS, an important and necessary condition is the formation of the ability to work independently to acquire knowledge, skills and the possibility of organizing educational and scientific activities.

The purpose of students' independent work is to master the fundamental

professional skills and abilities in the field, experience in creative and research activities. Independent work of students contributes to the development of independence, responsibility and organization, a creative approach to solving problems of the educational and professional level.

The objectives of the SRS are:

- systematization and consolidation of the theoretical knowledge and practical skills of students;
- deepening and expanding theoretical knowledge;
- formation of skills to use regulatory, legal, reference documentation and special literature;
- development of cognitive abilities and activity of students: creative initiative, independence, responsibility and organization;
- formation of independent thinking, abilities for self-development, self-improvement and self-realization;
- development of research skills;
- the use of material collected and obtained in the course of independent studies at seminars, laboratory classes, when writing course and final qualification papers, for effective preparation for final tests and exams.

3. Types of independent work

In the educational process of a higher professional educational institution, two types of independent work are distinguished: classroom, under the guidance of a teacher, and extracurricular. The close relationship between these types of work provides for the differentiation and effectiveness of the results of its implementation and depends on the organization, content, logic of the educational process (interdisciplinary connections, promising knowledge, etc.):

Classroom independent work on the discipline is carried out in classes under the direct guidance of the teacher and on his assignment.

Extracurricular independent work is carried out by the student on the instructions of the teacher, but without his direct participation.

The main types of independent work of students without the participation of a teacher in this discipline are:

- formation and assimilation of the content of lecture notes on the basis of educational literature recommended by the lecturer, including information educational resources (electronic textbooks, electronic libraries, etc.);
- preparation for laboratory work, their design.

The main types of independent work of students with the participation of teachers are:

- ongoing consultations;
- acceptance and defense of laboratory work (during the L/R).
- Implementation of the course work.

4. Organization of SRS

The methodology for organizing students' independent work depends on the structure, nature and features of the discipline "Digital Image Processing", the number of hours for its study, the type of tasks for students' independent work, the individual qualities of students and the conditions of educational activity.

The process of organizing students' independent work includes the following stages:

- preparatory (definition of goals, drawing up a program, preparation of methodological support, preparation of equipment);
- the main one (implementation of the program, the use of methods of searching for information, assimilation, processing, application, transfer of knowledge, fixing the results, self-organization of the work process);
- final (assessment of the significance and analysis of the results, their systematization, assessment of the effectiveness of the program and methods of work, conclusions about the directions of labor optimization).

Students' Activities for the Formation and Development of Skills of Educational Independent Work

In the process of independent work, the student acquires the skills of self-organization, self-control, self-management, self-reflection and becomes an active independent subject of educational activity.

Performing independent work under the supervision of a teacher, the student must:
master the minimum content submitted for independent work of students and proposed by the teacher in accordance with the educational standards of higher education;

- ☐ plan independent work in accordance with the schedule of independent work proposed by the teacher;
- ☐ the student should carry out independent work in the organizational forms provided for by the curriculum and work program of the teacher;

Perform independent work and report on its results in accordance with the schedule for submitting results, types and deadlines for reporting on students' independent work.

A student can:

in addition to the one proposed by the teacher (with justification and agreement with him) and the minimum mandatory content determined by the program for this discipline:

- independently determine the level (depth) of the study of the content of the material;

- ☐ suggest topics and questions for self-study;

- Offer a reasonable, individual timetable for independent work and reporting on the results of independent work as part of the overall timetable;

- offer their own options for organizational forms of independent work;

- ☐ to use for independent work methodological manuals, teaching aids, developments in addition to the list proposed by the teacher;

- ☐ to use not only control, but also self-control of the results of independent work in accordance with the methods of self-control proposed by the teacher or chosen independently.

Independent work of students should have an important impact on the formation of the personality of the future bachelor, it is planned by the student independently. Each student independently determines the mode of his work and the degree of work spent on mastering the educational content in each discipline. He performs extracurricular work according to a personal individual plan, depending on his training, time and other conditions.

5. General recommendations for the organization of independent work

The main form of independent work of a student is the study of lecture notes, their supplementation, recommended literature, active participation in the in laboratory classes. But for successful educational activity, its intensification, it is necessary to take into account the following subjective factors:

1. Knowledge of school program material, the presence of a solid system of gaps necessary for mastering the main university courses. This is especially important for mathematical disciplines. It is necessary to distinguish gaps in knowledge that make it difficult to assimilate new material from small abilities. By spending effort to overcome these gaps, the student will ensure normal academic performance and believe in his abilities.

2. Availability of skills, skills of mental work:

- a) the ability to take notes at lectures and when working with a book;

b) mastery of logical operations: comparison, analysis, synthesis, generalization, definition of concepts, rules of systematization and classification.

3. The specifics of cognitive mental processes: attention, memory, speech, observation, intelligence and thinking. Poor development of each of them becomes a serious obstacle in learning.

4. Good working capacity, which is ensured by a normal physical condition. After all, serious learning is a large, versatile and diverse work. The result of learning is evaluated not by the amount of information communicated, but by the quality of its assimilation, the ability to use it and the development of one's ability to further independent education.

5. Correspondence of the chosen activity, profession to individual abilities. It is necessary to develop the ability to self-regulate one's emotional state and eliminate circumstances that disturb the business mood, interfere with the planned work.

6. Mastering the optimal style of work that ensures success in activity. Alternating work and breaks in work, rest periods, individually justified sleep duration, preference for evening or morning classes, stress resistance to exams and features of preparation for them,

7. The level of requirements for oneself, determined by the existing self-esteem. Adequate assessment of knowledge, advantages, disadvantages is an important component of the self-organization of a person, without it it is impossible to successfully manage one's behavior and activity.

One of the main features of education in higher education is that constant external control is replaced by self-control, and the active role in learning belongs not so much to the teacher as to the student.

Knowing the basic methods of scientific organization of mental work, it is possible to achieve the best results with the least expenditure of time, money and labor efforts.

The effectiveness of assimilation of incoming information depends on the ability of a person to work at a particular moment of his activity.

Working capacity is the ability of a person to work with a high degree of intensity for a certain time. There are internal and external factors of working capacity.

Internal factors of working capacity include intellectual characteristics, will, and state of health.

External:

- organization of the workplace, work and rest schedule;
- the level of labor organization - the ability to obtain a certificate and use

information;

- the amount of mental load.

The outstanding Russian physiologist N. E. Vvedensky identified the following conditions for the productivity of mental activity:

- every work must be entered into gradually;
- dimensionality and rhythm of work. Different people have a more or less different pace of work;
- the usual sequence and systematicity of activity;
- correct alternation of work and rest.

Rest does not necessarily imply complete inaction on the part of a person, it can be achieved by a simple change of business. During the day, the ability to work changes. The most fruitful time is the *morning time (from 8 to 14 hours)*, and the maximum working capacity falls on the period from 10 to 13 hours, then *the afternoon (from 16 to 19 hours)* and *the evening (from 20 to 24 hours)*. It is better to study very difficult material at the beginning of each period of time (preferably in the morning) after a good rest. After 1-1.5 hours, breaks of 10 - 15 minutes are needed, after 3 - 4 hours of work, the rest should be long - about an hour.

An integral part of the scientific organization of mental labor is the mastery of the technique of mental labor.

The time available to the student to complete the curriculum consists of two components: one of them is classroom work at the university according to the schedule of classes, the other is extracurricular independent work. Tasks and materials for independent work are given during training sessions according to the schedule, in the same classes the teacher controls independent work, and also assists students in the correct organization of work.

Self-study will require intensive mental work, which must not only be properly organized, but also stimulated. At the same time, it is very important to be able to maintain a steady attention to the material being studied. The development of attention requires significant volitional efforts. That is why, if a student notices that he is often distracted during self-study, he needs to force himself to concentrate. Such a procedure must be done constantly, as this is attention training. Sustained attention appears when a person treats the matter with interest.

It is necessary to properly organize your classes in time: 50 minutes - work, 5-10 minutes - break; after 3 hours of work, a break - 20-25 minutes. Otherwise, the growing fatigue will lead to instability of attention. A very important factor influencing the increase in mental performance is the

systematic physical education classes. The organization of active recreation provides for the alternation of mental and physical activity, which fully restores a person's working capacity.

6. Independent work of a student is a necessary link in the formation of Researcher's Work

The progress of science and technology, information technology leads to a significant increase in scientific information, which imposes higher requirements not only on the moral and ethical qualities of a person, but also, in particular, constantly increasing requirements in the field of education - renewal, modernization of general and professional knowledge, skills of a specialist.

Any education should act as a dynamic process that exists in a person and continues throughout his life. Mastering the scientific thought and language of science is a necessary component in the self-organization of a future research specialist. This is understood not so much as the accumulation of knowledge, but the mastery of scientifically based ways of acquiring it. This, generally speaking, is the main task of the university.

The specificity of the university educational process, in the organization of which more and more space is given to the independent work of the student, is that it seems to be the last and most adequate link for the implementation of this task. For during the study at the university, the development of the style and skills of educational (cognitive) activity takes place, the rational nature of which will contribute to the constant renewal of the knowledge of a highly qualified university graduate.

However, there are certain difficulties before this path, in particular, the transition of the student from the synthetic process of learning in secondary school to the analytical one in higher school. This is due to both the new content of education (expansion of general education and deepening of professional training) and new, hitherto unknown forms of learning (lectures, seminars, laboratory classes, etc.). The student not only acquires the knowledge provided by the program and teaching aids, but he must also become acquainted with the ways of acquiring knowledge in such a way as to be able to assess what we know, how we know it and how we have achieved this knowledge. All this is achieved through one's own independent work.

This is also because self-acquired knowledge is more operational, it becomes personal property, as well as a motive for behavior, develops intellectual traits, attention, observation, criticality, and the ability to evaluate. The role of the teacher is mainly to help accumulate knowledge (in relation to first-year students), and in the subsequent years of study, in the senior years, to jointly identify problems and take care of the student's independent search, as well as to control their activities. It should be noted that it is not possible to limit oneself only to the acquisition of knowledge provided for by the program of the discipline being studied, it is necessary to constantly deepen the knowledge gained, focusing it on some narrow defined area that corresponds to the interests of the student. In-depth study of all subjects provided for by the program is possible in practice, and good organization of work saves time, which creates conditions for a deep, systematic, interested study of the topic independently chosen by the student.

Of course, all the advice, examples, recommendations in this area given by the teacher, or certain publications, or other sources, do not guarantee any success without the manifestation of one's own activity in this matter, i.e. they do not give ready-made recipes, but should contribute to the analysis of one's own work, its goals, and organization in accordance with individual characteristics. Taking into account personal capabilities, existing living and working conditions, skills, on the basis of these recommendations, it is possible to develop an individually justified set of methods, ways, find one's own style or improve it, in order to study certain material, have time to assess its significance, suitability and possibilities of its application, in order to ultimately ensure the success of one's studies from future professional activity

7. Methodological recommendations for students on individual forms of independent work

The system of higher education implies a much greater independence of students in planning and organizing their activities.

Work with literature

When working with literature, it is necessary to select literature, learn how to read it correctly, and keep notes. For the selection of literature in the library, alphabetical and systematic catalogs are used.

It is important to remember that rational skills of working with a book are always a great saving of time and effort.

The correct selection of textbooks is recommended by the teacher who reads the lecture course. The necessary literature can also be indicated in the methodological developments for this course.

When studying the material from the textbook, one should move on to the next question only after the previous one has been correctly understood, describing on paper all the calculations and calculations (including those that are omitted in the textbook or given in the lecture for independent conclusion).

In the study of any discipline, independent individual work plays a large and important role.

Particular attention should be paid to the definition of the basic concepts of the course. The student should analyze in detail the examples that explain such definitions, and be able to build similar examples independently. It is necessary to achieve an accurate idea of what you are studying. It is useful to make reference notes. When studying the material from the textbook, it is useful to supplement the lecture notes in the notebook (in specially designated margins). There you should also note the questions selected by the student for consultation with the teacher.

It is recommended to highlight the conclusions obtained as a result of the study in the notes so that they are better remembered when rereading the notes.

Experience shows that many students are helped by compiling a list of reference signals containing the most important and most frequently used formulas and concepts. Such a sheet helps to memorize formulas, the main provisions of the lecture, and can also serve as a permanent reference for the student. There are two types of reading: primary and secondary. *Primary reading* is attentive, unhurried reading, in which you can dwell on difficult places. may not always be understood after the initial reading.

The task of *secondary* reading is the complete assimilation of the meaning of the whole (this reading may not be the second, but the third or fourth).

Reading a scientific text is a part of cognitive activity. Its purpose is to extract the necessary information from the text. The effectiveness of the action largely depends on how conscious the reader is of his own internal attitude when referring to the word (to find the necessary information, to assimilate the information in whole or in part, to critically analyze the material, etc.).

There are four main attitudes in reading a scientific text:

1. information retrieval (the task is to find, highlight the information sought)

2. assimilative (the reader's efforts are aimed at understanding and remembering as fully as possible both the information presented by the author and the entire logic of his reasoning)

3. analytical-critical (the reader seeks to critically comprehend the material, analyzing it, determining his attitude to it)

4. creative (creates in the reader a readiness in one form or another – as a starting point for his reasoning, as an image for action by analogy, etc. – to use the author's judgments, the course of his thoughts, the result of observation, the developed methodology, to supplement them, to subject them to a new check).

The existence of several types of reading *is associated with the presence of different attitudes to addressing a scientific text:*

1. bibliographic – viewing catalog cards, recommendation lists, summary lists of journals and articles for the year, etc.;

2. browsing – used to search for materials containing the necessary information, usually it is used immediately after working with references and catalogs, as a result of such viewing the reader establishes which of the sources will be used in further work;

3. introductory – implies a continuous, sufficiently detailed reading of the selected articles, chapters, individual pages, the purpose is to get acquainted with the nature of the information, to find out what issues are submitted by the author for consideration, to sort the material;

4. Studying – involves a thorough mastery of the material; In the course of such reading, the reader's trust in the author, readiness to accept the information presented, the attitude to the most complete understanding of the material is realized;

5. Analytical-critical and creative reading are two types of reading that are close to each other in that they participate in solving research problems. The first of them involves a directed critical analysis of both the information itself and the ways of its receipt and presentation by the author; the second is the search for those judgments, facts, on which or in connection with which the reader considers it necessary to express his own thoughts.

Of all the types of reading considered, the main one for students is the learning one – it allows them to accumulate knowledge in various fields when working with educational literature. That is why this type of reading should be mastered in the first place. In addition, when mastering this type of reading, the main techniques are formed that increase the effectiveness of working with a scientific text.

The main types of systematized recording of what has been read:

1. Annotation is an extremely brief coherent description of the book (article) viewed or read, its contents, sources, nature and purpose;
2. Planning is a brief logical organization of the text, revealing the content and structure of the material being studied;
3. Thesis is a concise reproduction of the author's main statements without involving factual material;
4. Citation is a verbatim copying from the text of excerpts, extracts that most significantly reflect a particular idea of the author;
5. Note-taking is a brief and consistent presentation of the content of what has been read.

A synopsis is a complex way of presenting the content of a book or article in a logical sequence. A synopsis accumulates the previous types of notes, allows you to comprehensively cover the content of a book or article. Therefore, the ability to make a plan, theses, make extracts and other notes determines the technology of compiling a note.

Methodical recommendations for compiling a synopsis:

1. Read the text carefully. Clarify the incomprehensible words in the reference literature. When writing down, do not forget to put the reference data in the margins of the note.
2. Highlight the main thing, make a plan.
3. Briefly formulate the main points of the text, note the author's description.
4. Take notes of the material, clearly following the points of the plan. When taking notes, try to express the idea in your own words. Notes should be kept clear and clear.
5. Write down quotes correctly. When quoting, take into account the conciseness and significance of the thought.

In the text of the synopsis, it is desirable to give not only thesis statements, but also their proofs. When drawing up a synopsis, it is necessary to strive for the capacity of each sentence. The thoughts of the author of the book should be stated briefly, taking care of the style and expressiveness of the writing. The number of additional elements of the synopsis should be logically justified, the notes should be distributed in a certain sequence that corresponds to the logical structure of the work. Margins should be left for clarification and addition.

Mastering the skills of taking notes requires dedication and daily independent work from the student.

Practical exercises

It should be emphasized that only after mastering the lecture material from a certain point of view (namely, from the point of view from which it is presented at the lectures) it will be consolidated in the laboratory classes both as a result of discussion and analysis of the lecture material, and with the help of completing the tasks of the laboratory work, the tasks set. Under these conditions, the student will not only learn the material well, but also learn how to apply it in practice, and will also receive an additional incentive (and this is very important) for active study of the lecture.

When performing tasks independently, it is necessary to justify each stage of the solution, based on the theoretical provisions of the course. If the student sees several ways to solve a problem (task), then it is necessary to compare them and choose the most rational one. It is useful to make a brief plan for solving the problem (task) before starting calculations. The solution of problems or examples should be presented in fractions, calculations should be arranged in a strict order, separating auxiliary calculations from the main ones. Solutions, if necessary, should be accompanied by comments, diagrams, drawings and drawings.

It should be remembered that the performance of each educational task should be carried out before the final logical answer required by the condition, and, if possible, with a conclusion. The performance of tasks of this type should be continued until the acquisition of solid skills in solving them.

Self-Test

After studying a certain topic from the notes in the notes and textbook, as well as solving a sufficient number of relevant tasks in laboratory classes and independently, the student is recommended, using a sheet of reference materials, to reproduce from memory definitions, conclusions of formulas, formulations of the main provisions and evidence.

If necessary, you need to carefully understand the material again.

Sometimes the insufficiency of assimilation of a particular issue becomes clear only when studying further material. In this case, it is necessary to go back and repeat the poorly mastered material. An important criterion for mastering theoretical material is the ability to solve problems or pass a test on the material covered. However, it should be remembered that the correct solution is

can be obtained as a result of the application of mechanically memorized formulas without understanding the essence of theoretical propositions.

Consulting

If in the process of independent work on the study of theoretical material or when performing tasks, the student has questions that cannot be solved independently, it is necessary to contact the teacher to obtain explanations or instructions from him. In his questions, the student should clearly express what he is experiencing difficulties, the nature of this difficulty. Advice should also be sought if there are doubts about the correctness of the answers to the self-test questions.

Preparation for the exam in the discipline "Digital Image Processing".

The study of many general professional and special disciplines ends with a test. Preparation for the test contributes to the consolidation, deepening and generalization of knowledge obtained in the process of study, as well as its application to solving practical problems. Preparing for it, the student eliminates the existing gaps in knowledge, deepens, systematizes and organizes his knowledge.

To pass the test, the student must fully complete the tasks of the laboratory work of the classes, prepare a report on each laboratory work and a report on the results of independent study of topics, according to the curriculum. If there are debts on the current certification in this discipline, the student can receive a grade "not passed".

Systematic preparation for classes during the semester will allow you to use the time of the exam session to systematize knowledge.

Current certification of students is carried out by teachers who conduct laboratory classes in the discipline in the following forms: interview during the defense of laboratory work with the provision of a written report. Admission to laboratory work occurs if students have a printed version of the report. The defense of the report takes place in the form of an interview.

8. Educational, methodological and information support of the discipline

List of basic and additional literature necessary for mastering the discipline (module)

List of basic literature:

1. Shefer, E. A.; Digital Image Processing: Textbook / E. A. Shefer. - Digital Image Processing, 2031-02-04. - Electron. dan. (1 file). - St. Petersburg : St. Petersburg State University

Industrial Technologies and Design, 2019. - 100 p. - electronic. - The book is in the premium version of EBS IPR BOOKS. - ISBN 2227-8397, unlimited copies

2. Pismennyy G. V., Mikhailov B. B., Korneev A. Yu. Moscow, Mashinostroenie Publ., 1991.

- 88 p. - (Industrial Robots and Flexible Production Systems). -

References: p. 84. - ISBN 5-217-00995-0, copies 1

3. Selyankin, V. V.; Computer vision. Analysis and processing of images Electronic resource / Selyankin V. V. : textbook. - St. Petersburg : Lan, 2019. - 152 p. - ISBN 978-5-8114-3368-1, unlimited copies

4. Selyankin, V. V.; Solving problems of computer vision: a textbook / V.V. Selyankin ; Ministry of Education and Science of the Russian Federation ; Southern Federal University ; Engineering and Technology Academy. - Taganrog : Publishing House of the Southern Federal University, 2016. - 93 p. : schemes, tabl. - <http://biblioclub.ru/>. - Bibliogr. in the book - ISBN 978-5-9275-2090-9, unlimited copies

8.1.2. List of additional literature:

1. Gonzalez, R.; Digital Image Processing: Practical Advice / R. Gonzalez, R. Woods ; transl. by P. A. Chochia ; trans. by L. I. Rubanov. - 3rd ed., ed. and add. - Moscow : Technosphere, 2012. - 1104 p. : ill., tabl., schemes. - (The World of Digital Processing). - <http://biblioclub.ru/>. - ISBN 978-5-94836-331-8, unlimited copies

2. Rafael, Gonzalez.; Digital Image Processing Electronic resource / Gonzalez Raphael, Woods Richard ; transl.: L. I. Rubanov, P. A. Chochia; ed. by P. A. Chochia. - Moscow : Technosphere, 2012. - 1104 p. - The book is in the premium version of EBS IPR BOOKS. - ISBN 978-5-94836-331-8, unlimited copies

3. Potapov, A. S.; Computer Vision Systems Elektronnyi resurs : Uchebno-metodicheskoe posobie po laboratornomu praktikumu [Computer vision systems] / A. S. Potapov, P.

O. Malashin. - Computer Vision Systems, 2022-10-01. - St. Petersburg: ITMO University, 2012. - 41 p. - The book is in the premium version of EBS IPR BOOKS. - ISBN 2227-8397, unlimited copies

www.biblioclub.ru - Electronic Library System "University Library Online"

www.eLibrary.ru - Scientific Electronic Library

www.iprbookshop.ru - IPRbooks Electronic Library System

List of information technologies used in the implementation of the educational process in the discipline (module), including a list of software and information

Reference systems

When reading lectures, computer equipment is used, demonstration of presentation multimedia materials. In laboratory classes, students perform laboratory work and demonstrate the result by means of computer technology and reports in electronic form, prepared by them during the hours of independent work.

Information reference systems:

Information and reference and information and legal systems used in the study of the discipline:

- 1 Electronic Library System "University Library Online" – www.biblioclub.ru
- 2 IPRbooks Electronic Library System – <http://iprbooks.ru>
- 3 Electronic Library System Lan – <https://e.lanbook.com/>

Software:

- 1 Viola Workstation 10
- 2 Alt Workstation K
- 3 Viola "Server"
- 4 Office software package - P7-Office