

МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Факультет Информационных технологий
Кафедра Информатики и информационных технологий

направление подготовки

09.03.02 «Информационные системы и технологии»

ЛАБОРАТОРНАЯ РАБОТА № 4

Дисциплина: «Распознавание образов в информационных и автоматизированных системах копия 1»

Тема: «Поиск контуров и примитивов на изображении»

Выполнил: студент группы 211-723

Сергеев Станислав Олегович

Дата, подпись _____
(Дата) (Подпись)

Проверил: _____
(Фамилия И.О., степень, звание) **(Оценка)**

Дата, подпись _____
(Дата) (Подпись)

Замечания: _____

Москва

2022

Поиск контуров и примитивов на изображении.

Цель:

Целью данной работы является изучение алгоритмов поиска контуров и примитивов на изображениях.

Постановка задачи:

Необходимо разработать приложение Windows Forms, способное осуществлять:

1. Обнаружение контуров.
2. Обнаружение примитивов (окружность, треугольник, четырёхугольник).

Листинг программы

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Emgu.CV;
using Emgu.CV.CvEnum;
using Emgu.CV.Structure;
using Emgu.CV.Util;
using Emgu.CV.ML.MlEnum;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace _4
{
    public partial class Form1 : Form
    {
        Image<Bgr, byte> sourceImage;
        public Form1()
        {
            InitializeComponent();
        }
        private void button1_Click(object sender, EventArgs e)
        {
            try
            {
                OpenFileDialog ofd = new OpenFileDialog();
                ofd.Filter = "Изображения (*.jpg, *.jpeg, *.png) | *.jpg; *.jpeg; *.png";
                if (ofd.ShowDialog() == DialogResult.OK)
                {
                    sourceImage = new Image<Bgr, byte>(ofd.FileName);
                }
                imageBox1.Image = sourceImage.Resize(400, 400, Inter.Linear);
            }
            catch (Exception err)
            {
                MessageBox.Show(err.ToString());
            }
        }
        private void button2_Click(object sender, EventArgs e)
```

```

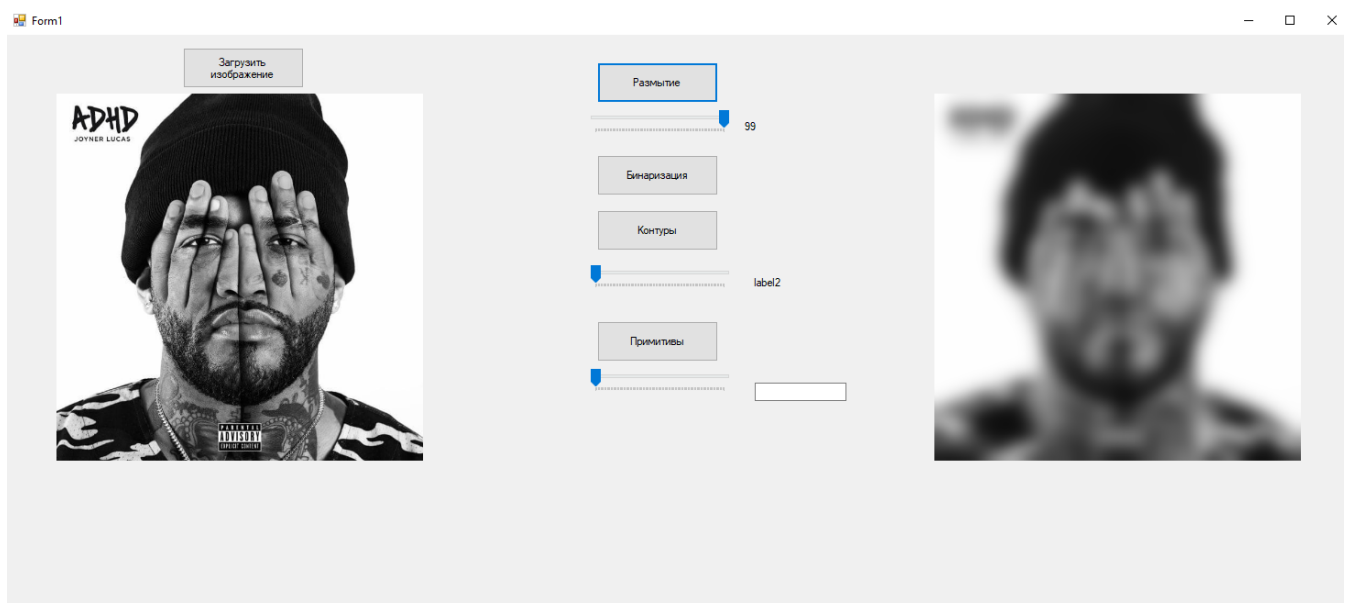
{
    var blurImage = blur(sourceImage.Clone(), trackBar1.Value);
    imageBox2.Image = blurImage.Resize(400, 400, Inter.Linear);
}
private void button3_Click(object sender, EventArgs e)
{
    var blurImage = blur(sourceImage.Clone(), trackBar1.Value);
    var binarizedImage = bimarize(blurImage, trackBar2.Value);
    imageBox2.Image = binarizedImage.Resize(400, 400, Inter.Linear);
}
private void button4_Click(object sender, EventArgs e)
{
    var blurImage = blur(sourceImage.Clone(), trackBar2.Value);
    var binarizedImage = bimarize(blurImage, trackBar1.Value);
    var contours = findCounters(binarizedImage);
    var contoursImageBlank = drawCounters(sourceImage.CopyBlank(), contours);
    imageBox2.Image = contoursImageBlank.Resize(400, 400, Inter.Linear);
}
private void button5_Click(object sender, EventArgs e)
{
    var blurImage = blur(sourceImage.Clone(), trackBar2.Value);
    var binarizedImage = bimarize(blurImage, trackBar1.Value);
    var contours = findCounters(binarizedImage);
    var contoursImageBlank = drawCounters(sourceImage.CopyBlank(), contours);
    var approxedBorders = primitivesFind(contoursImageBlank.CopyBlank(), contours,
trackBar3.Value);
    var numberOfPrimitives = primitives(contoursImageBlank.CopyBlank(), contours,
trackBar3.Value);
    imageBox2.Image = approxedBorders.Resize(400, 400, Inter.Linear);
    textBox1.Text = "" + numberOfPrimitives;
}
private void trackBar1_Scroll(object sender, EventArgs e)
{
    label1.Text = String.Format("{0}", trackBar1.Value);
}
private void trackBar2_Scroll(object sender, EventArgs e)
{
    label2.Text = String.Format("{0}", trackBar2.Value);
}
public Image<Gray, byte> blur(Image<Bgr, byte> _sourceImage, int kernelSize)
{
    var grayImage = _sourceImage.Convert<Gray, byte>();
    var blurredImage = grayImage.SmoothGaussian(kernelSize);
    return blurredImage;
}
public Image<Gray, byte> bimarize(Image<Gray, byte> _sourceImage, int
binarizationThreshold)
{
    var threshold = new Gray(80);
    var color = new Gray(255);
    var binarizedImage = _sourceImage.ThresholdBinary(threshold, color);
    return binarizedImage;
}
public VectorOfVectorOfPoint findCounters(Image<Gray, byte> binImg)
{
    var contours = new VectorOfVectorOfPoint();
    CvInvoke.FindContours(binImg, contours, null, RetrType.List,
ChainApproxMethod.ChainApproxSimple);
    return contours;
}
public Image<Bgr, byte> drawCounters(Image<Bgr, byte> dstImage, VectorOfVectorOfPoint
borders)
{
    for (int i = 0; i < borders.Size; i++)
    {
        var points = borders[i].ToArray();
        dstImage.Draw(points, new Bgr(Color.GreenYellow), 2);
    }
    return dstImage;
}

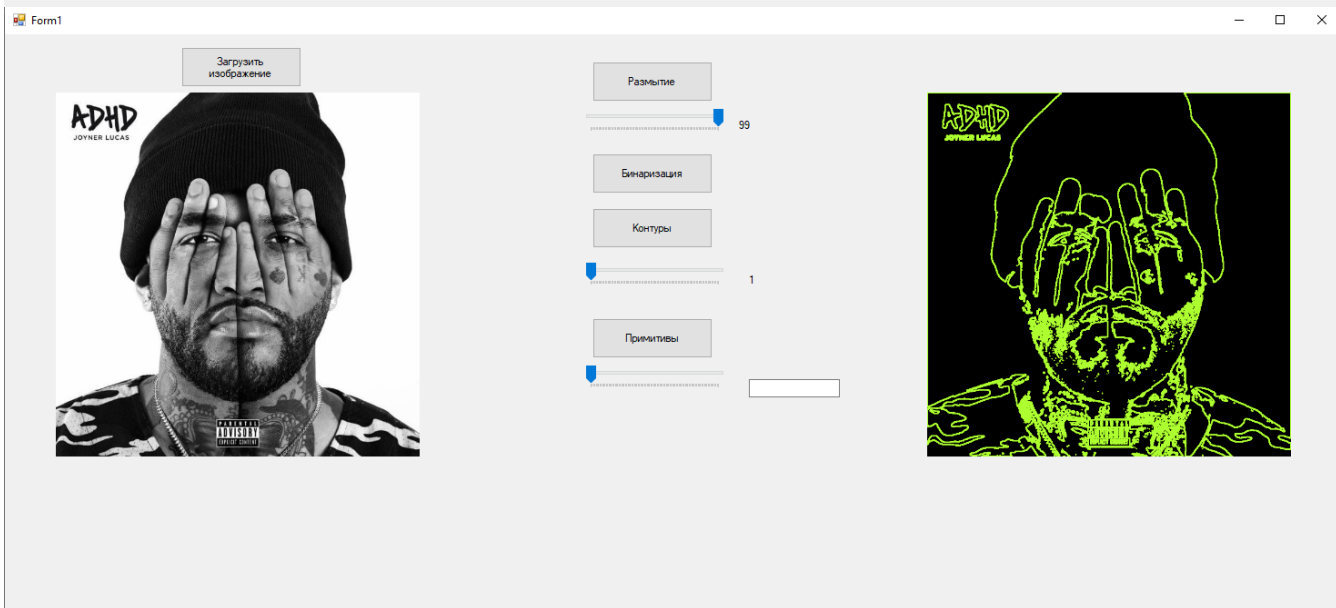
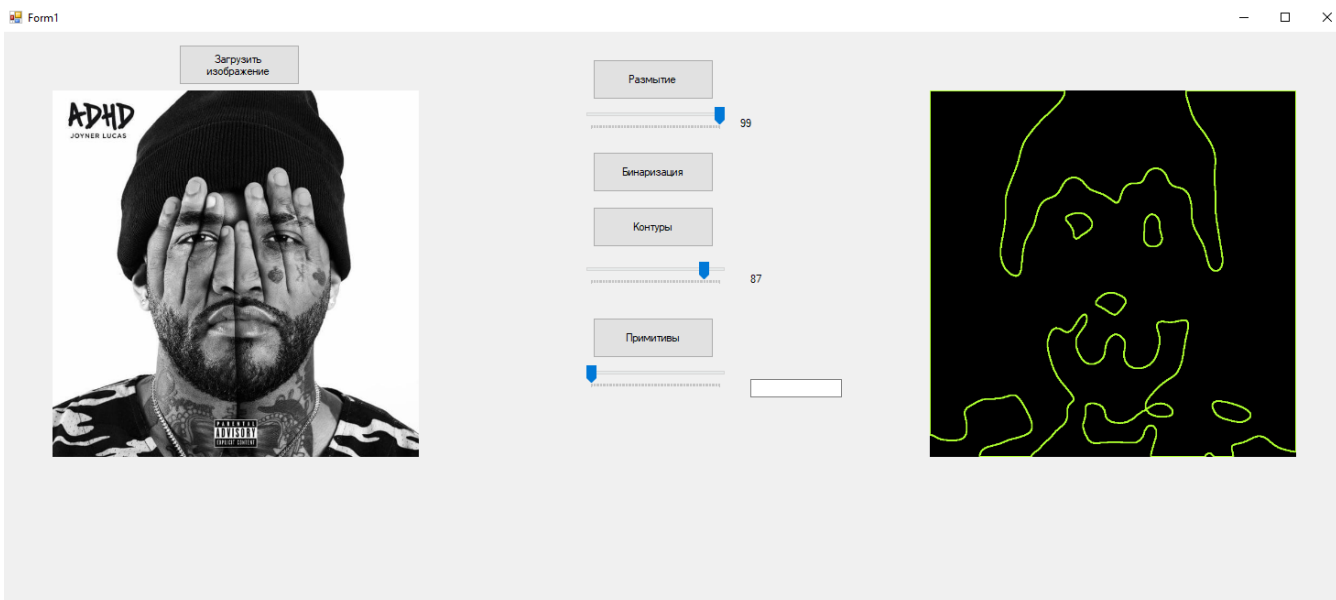
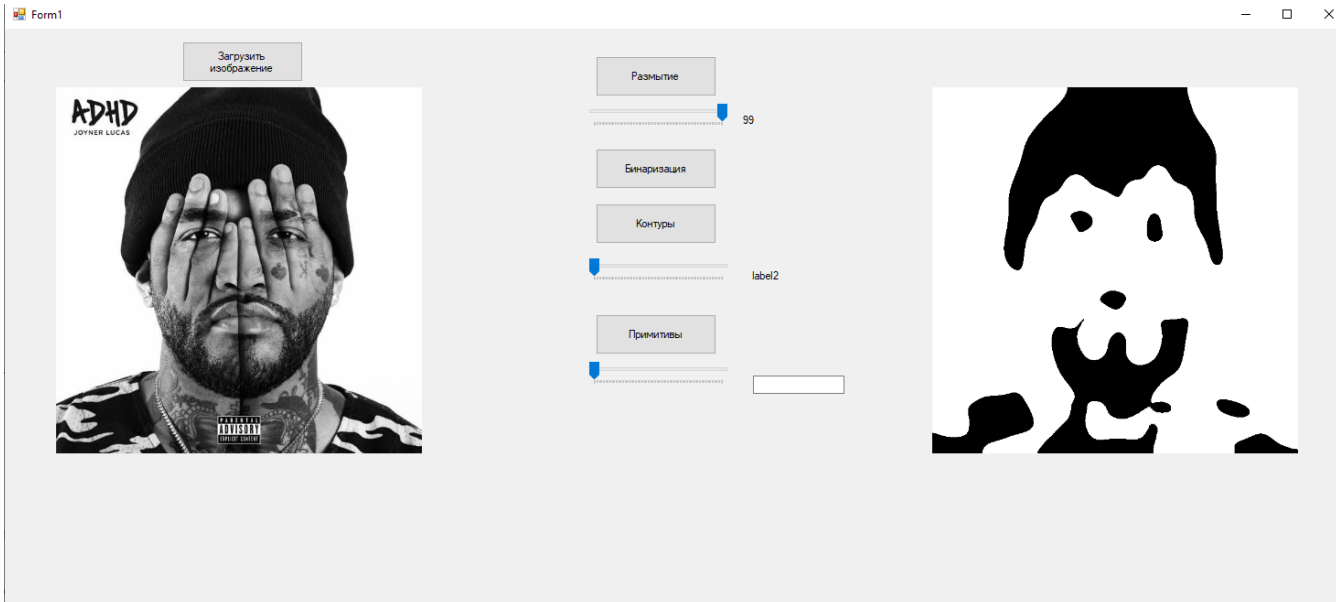
```

```


    }
    public Image<Bgr, byte> primitivesFind(Image<Bgr, byte> contImg, VectorOfVectorOfPoint
borders, double minArea)
    {
        for (int i = 0; i < borders.Size; i++)
        {
            var approxContour = new VectorOfPoint();
            CvInvoke.ApproxPolyDP(borders[i], approxContour, CvInvoke.ArcLength(borders[i],
true) * 0.05, true);
            if (CvInvoke.ContourArea(approxContour, false) > minArea)
            {
                if (approxContour.Size == 3)
                {
                    var points = approxContour.ToArray();
                    contImg.Draw(new Triangle2DF(points[0], points[1], points[2]), new
Bgr(Color.GreenYellow), 2);
                }
            }
        }
        return contImg;
    }
    public int primitives(Image<Bgr, byte> contImg, VectorOfVectorOfPoint borders, double
min)
    {
        int count = 0;
        for (int i = 0; i < borders.Size; i++)
        {
            var approxContour = new VectorOfPoint();
            CvInvoke.ApproxPolyDP(borders[i], approxContour, CvInvoke.ArcLength(borders[i],
true) * 0.05, true);
            if (CvInvoke.ContourArea(approxContour, false) > min)
            {
                if (approxContour.Size == 3)
                {
                    var points = approxContour.ToArray();
                    contImg.Draw(new Triangle2DF(points[0], points[1], points[2]), new
Bgr(Color.GreenYellow), 2);
                    count++;
                }
            }
        }
        return count;
    }
}
}
}

```





Загрузить изображение



Размытие

99

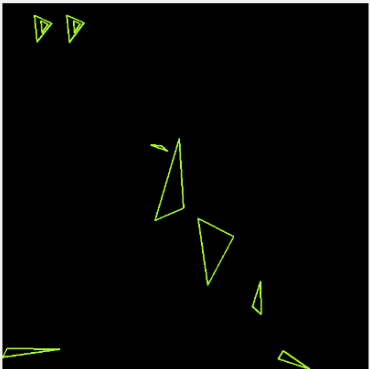
Бинаризация

Контур


1

Примитивы

10



Загрузить изображение



Размытие

99

Бинаризация

Контур

23

Примитивы

5

