**Отчет по лабораторной работе №6 по предмету «Системы технического зрения в автоматизированных системах»**

**Тема:** Техническое зрение в задачах позиционирования и навигации

**Цель работы:** Разработать программное обеспечение для контроля положения мобильной роботизированной системы

**Задание:**

* Разработать приложение получения изображения с видеокамеры, видеофайла или файла-изображения с использованием библиотеки OpenCV (OpenCVSharp);
* Разработать функцию установки и применения цветового ключа для

бинаризации;

* Реализовать функцию ввода параметров и положения камеры относительно центра опоры робота и функцию загрузки положения маркеров (сетчатых шаблонов);
* Реализовать функции анализа контура объектов и выделения кандидатов на навигационные маркеры (сетчатые шаблоны);
* Реализовать функцию детектирования маркеров и вычисления их положения относительно камеры;
* Реализовать функцию поиска размещения робота методом ангуляции (по центрам маркеров) по каждой паре видимых маркеров;
* Реализовать функцию поиска размещения роботов методом реконструкции положения и ориентации маркера (Coplanar POSIT);
* Разработать систему визуализации всех вычисленных положений и ориентаций робота.

**Код программы:**

using System;

using System.Collections.Generic;

using System.Collections;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.IO;

using OpenCvSharp;

using OpenCvSharp.Extensions;

using System.Threading;

using System.Runtime.InteropServices;

namespace Lab\_6

{

public partial class Form1 : Form

{

int tempNum = 0;

int tempCount = 0;

List<Mat> tempArray = new List<Mat>();

List<bool[]> codeArray = new List<bool[]>();

List<string> paths = new List<string>();

Mat frame\_in;

Mat frame\_out;

Bitmap show;

bool cameraOpened = false;

VideoCapture capture;

private Thread camera;

public Form1()

{

InitializeComponent();

}

private void Load\_Click(object sender, EventArgs e)

{

//try

//{

System.Windows.Forms.OpenFileDialog dlg = new System.Windows.Forms.OpenFileDialog();

dlg.FileName = "Document";

dlg.DefaultExt = ".png";

dlg.Filter = "Text documents (.png)|\*.png";

DialogResult result = dlg.ShowDialog();

paths.Add(dlg.FileName);

tempArray.Add(new Mat(dlg.FileName));

codeArray.Add(new bool[49]);

int xR = 2;

int yR = 2;

int codeNum = -1;

Bitmap displayBitmap = new Bitmap(dlg.FileName);

for (int m = 0; m < 7; m++)

{

yR = 2;

xR += 15;

for (int n = 0; n < 7; n++)

{

codeNum++;

yR += 15;

int white = 0;

int black = 0;

for (int q = xR; q < xR + 11; q++)

{

for (int r = yR; r < yR + 11; r++)

{

if((q < 136)&&(r < 136))

if (displayBitmap.GetPixel(q, r).R == 255)

white++;

else

black++;

}

}

if (white > black)

{

codeArray[codeArray.Count - 1][codeNum] = true;

Cv2.PutText(tempArray[tempArray.Count() - 1], "W", new OpenCvSharp.Point(xR + 2, yR + 11), HersheyFonts.Italic, 0.3, new Scalar(0, 0, 0));

Cv2.Rectangle(tempArray[tempArray.Count() - 1], new OpenCvSharp.Point(xR, yR), new OpenCvSharp.Point(xR + 11, yR + 11), new Scalar(0, 0, 0));

}

else

{

codeArray[codeArray.Count - 1][codeNum] = false;

Cv2.PutText(tempArray[tempArray.Count() - 1], "B", new OpenCvSharp.Point(xR + 2, yR + 11), HersheyFonts.Italic, 0.3, new Scalar(255, 255, 255));

Cv2.Rectangle(tempArray[tempArray.Count() - 1], new OpenCvSharp.Point(xR, yR), new OpenCvSharp.Point(xR + 11, yR + 11), new Scalar(255, 255, 255));

}

}

}

imgTemp.Image = BitmapConverter.ToBitmap(tempArray[tempArray.Count() - 1]);

list.Items.Add("Шаблон №" + paths.Count());

//}

//catch { }

}

private void SetSails(object sender, EventArgs e)

{

try

{

if (ss.Text.Equals("Поднять паруса!"))

{

camera = new Thread(new ThreadStart(CaptureCameraCallback));

camera.Start();

ss.Text = "Опустить паруса!";

cameraOpened = true;

}

else

{

capture.Release();

ss.Text = "Поднять паруса!";

cameraOpened = false;

}

}

catch { }

}

void CaptureCameraCallback()

{

//frame\_in = new Mat();

//frame\_out = new Mat();

capture = new VideoCapture(1); // (0) -встроенная, (1) - первая подключенная

capture.Open(1);

if (capture.IsOpened())

{

try

{

while (cameraOpened == true) //если камера запущена

{

frame\_in = new Mat();

frame\_out = new Mat();

capture.Read(frame\_in);

frame\_in.CopyTo(frame\_out);

Recognize();

img.Image = BitmapConverter.ToBitmap(frame\_out);

frame\_in.Dispose();

frame\_out.Dispose();

//GC.Collect();

}

}

catch { }

}

}

void Recognize()

{

OpenCvSharp.Point[][] contours;

HierarchyIndex[] hierarchy;

Mat after\_2gray = new Mat();

Mat after\_canny = new Mat();

Mat after\_blur = new Mat();

Mat frame\_range = new Mat();

Mat after\_after = new Mat();

Cv2.Blur(frame\_in, after\_blur, new OpenCvSharp.Size(3, 3));

Cv2.InRange(after\_blur, new Scalar(0, 0, 0), new Scalar(70, 70, 70), frame\_range);

Cv2.FindContours(frame\_range, out contours, out hierarchy, RetrievalModes.External, ContourApproximationModes.ApproxSimple);

for (int i = 0; i < contours.Count(); i++)

{

Rect bR = Cv2.BoundingRect(contours[i]);

if ((bR.Width > 20) && (bR.Height > 20))

{

Cv2.DrawContours(frame\_out, contours, i, new Scalar(0, 0, 255), 2);

try

{

Point2d[] GetTempPoints = FindTemp(contours[i]);

ShowTemp(GetTempPoints);

}

catch { }

}

}

}

Point2d[] FindTemp(OpenCvSharp.Point[] contours)

{

//tempCount = contours.Length;

//Cv2.PutText(frame\_out, tempCount.ToString(), new OpenCvSharp.Point(20, 20),

//HersheyFonts.HersheyDuplex, 0.5, new Scalar(255, 255, 0));

OpenCvSharp.Point[] approxedRow;

OpenCvSharp.Point[] approxed = new OpenCvSharp.Point[4];

approxedRow = Cv2.ApproxPolyDP(contours, 4, false);

if (approxedRow.Length > 4)

{

int[] indexDel = new int[20];

int k = -1;

for (int i = 0; i < approxedRow.Length; i++)

{

int pl = i+1;

int mi = i-1;

if (mi == -1)

mi = approxedRow.Length - 1;

if (pl == approxedRow.Length)

pl = 0;

if ((Math.Abs(approxedRow[pl].Y - approxedRow[i].Y) < 30)

&& (Math.Abs(approxedRow[mi].Y - approxedRow[i].Y) < 30))

{

k++;

indexDel[k] = i;

}

}

int n = -1;

for (int i = 0; i < approxedRow.Length; i++)

{

if (Array.IndexOf(indexDel, i) == -1)

{

n++;

approxed[n] = approxedRow[i];

}

}

}

else

{

approxed[0] = approxedRow[0];

approxed[1] = approxedRow[1];

approxed[2] = approxedRow[2];

approxed[3] = approxedRow[3];

}

//тут мы расставляем точки по местам

Point2d[] pointsList = new Point2d[4];

IEnumerable<Point2d> points = new List<Point2d>();

Point2d[] pointsShow = new Point2d[4];

for (int p = 0; p < 4; p++)

{

int indexL = 0;

double maxL = 0;

for (int q = 0; q < 4; q++)

{

if (p != q)

{

double l = Math.Sqrt(Math.Pow(approxed[p].X - approxed[q].X, 2) + Math.Pow(approxed[p].Y - approxed[q].Y, 2));

if (maxL < l)

{

maxL = l;

indexL = q;

}

}

}

if ((approxed[indexL].X - approxed[p].X < 0) && (approxed[indexL].Y - approxed[p].Y > 0))

{

pointsList[0] = approxed[p];

pointsShow[0] = approxed[p];

}

if ((approxed[indexL].X - approxed[p].X > 0) && (approxed[indexL].Y - approxed[p].Y > 0))

{

pointsList[1] = approxed[p];

pointsShow[1] = approxed[p];

}

if ((approxed[indexL].X - approxed[p].X > 0) && (approxed[indexL].Y - approxed[p].Y < 0))

{

pointsList[2] = approxed[p];

pointsShow[2] = approxed[p];

}

if ((approxed[indexL].X - approxed[p].X < 0) && (approxed[indexL].Y - approxed[p].Y < 0))

{

pointsList[3] = approxed[p];

pointsShow[3] = approxed[p];

}

}

points = pointsList;

for (int i = 0; i < 4; i++)

{

Cv2.PutText(frame\_out, Convert.ToString(i), new OpenCvSharp.Point(Convert.ToInt32(pointsShow[i].X), Convert.ToInt32(pointsShow[i].Y)),

HersheyFonts.HersheyDuplex, 0.3, new Scalar(0, 255, 0));

}

return pointsList;

}

void ShowTemp(Point2d[] temp)

{

IEnumerable<Point2d> needhom = new List<Point2d>()

{

new Point2d(136, 0),

new Point2d(0, 0),

new Point2d(0, 136),

new Point2d(136, 136)

};

Mat h = Cv2.FindHomography(temp, needhom);

Mat display = new Mat(136, 136, MatType.CV\_8U);

Cv2.WarpPerspective(frame\_in, display, h, new OpenCvSharp.Size(136, 136));

Cv2.InRange(display, new Scalar(0, 0, 0), new Scalar(70, 70, 70), display);

//Bitmap displayBitmap = BitmapConverter.ToBitmap(display);

//imgTemp.Image = displayBitmap;

bool[] codeArrayReс = new bool[49];

int codeNum = -1;

int xR = 0;

int yR = 0;

for (int m = 0; m < 7; m++)

{

yR = 2;

xR += 15;

for (int n = 0; n < 7; n++)

{

codeNum++;

yR += 15;

int white = 0;

int black = 0;

for (int q = xR; q < xR + 11; q++)

{

for (int r = yR; r < yR + 11; r++)

{

if ((q < 136) && (r < 136))

{

var pixel = display.Get<Vec3b>(r, q);

if ((pixel.Item0 == 255)|| (pixel.Item1 == 255)|| (pixel.Item2 == 255))

white++;

else

black++;

}

}

}

if (white > black)

{

codeArrayReс[codeNum] = true;

}

else

{

codeArrayReс[codeNum] = false;

}

}

}

for (int i = 0; i < codeArray.Count; i++)

{

if (codeArray[i].SequenceEqual(codeArrayReс))

{

Cv2.PutText(frame\_out, "Temp "+(i + 1).ToString(), new OpenCvSharp.Point(temp[1].X + 10, temp[1].Y - 25),

HersheyFonts.Italic, 0.5, new Scalar(180, 250, 180));

}

}

double x1Mid = (double)(temp[0].X + temp[1].X) / 2.0;

double y1Mid = (double)(temp[0].Y + temp[1].Y) / 2.0;

double x2Mid = (double)(temp[2].X + temp[3].X) / 2.0;

double y2Mid = (double)(temp[2].Y + temp[3].Y) / 2.0;

Cv2.Line(frame\_out, new OpenCvSharp.Point(x1Mid, y1Mid), new OpenCvSharp.Point(x2Mid, y2Mid), new Scalar(0,255,255), 1);

double length = Math.Sqrt(

Math.Pow(Math.Abs(x1Mid - x2Mid) ,2)

+Math.Pow(Math.Abs(y1Mid - y2Mid), 2)

);

double distance = Math.Round(5304.6 / length, 2, MidpointRounding.AwayFromZero);

Cv2.PutText(frame\_out, "Dist. " + distance.ToString() + " cm", new OpenCvSharp.Point(temp[1].X + 10, temp[1].Y - 10),

HersheyFonts.Italic, 0.5, new Scalar(180, 250, 180));

}

private void ToLeft(object sender, EventArgs e)

{

if (tempNum != 0)

tempNum--;

}

private void ToRight(object sender, EventArgs e)

{

if (tempNum != tempCount - 1)

tempNum++;

}

private void SelectedChange(object sender, EventArgs e)

{

Bitmap bitmapShowTemp = new Bitmap(paths[list.SelectedIndex]);

imgTemp.Image = bitmapShowTemp;

/\*string stringCode = "";

for (int i = 0; i < codeArray[list.SelectedIndex].Count(); i++)

{

stringCode = stringCode + codeArray[list.SelectedIndex][i];

}

MessageBox.Show(stringCode);\*/

}

}

}

**Скриншоты работы программы:**

