测量识别

Halcon：

Model：

```
*一、创建形状模板
*画出待识别工件
dev_get_window (WindowHandle)
dev_disp_text ('请画出要识别的工件', 'window', 'top', 'left', 'black', [], [])
draw_rectangle1 (WindowHandle, Row11, Column11, Row21, Column21)
gen_rectangle1 (ROI_0,Row11, Column11, Row21, Column21)
reduce_domain (GrayImage, ROI_0, ImageReduced)
*取中心点坐标
threshold (ImageReduced, ShapeModelRegion, 0, 256)
area_center (ShapeModelRegion, Area, ShapeModelRow, ShapeModelColumn)
orientation_region (ShapeModelRegion, ShapeModelPhi)
*创建形状模板
create_shape_model (ImageReduced, 6, 0, rad(360), 'auto', 'auto', 'use_polarity', 'auto', 'auto',
get_shape_model_contours (ModelContours, ShapeModelID, 1)
*保存形状模板
FileShapeModel := './Model/ShapeModel.sbm'
write_shape_model (ShapeModelID, FileShapeModel)

*二、创建计量模型
create_metrology_model (MetrologyHandle)
create_metrology_model (MetrologyHandle1)
set_metrology_model_image_size (MetrologyHandle, Width, Height)
read_image (Image, './Image/mode.png')
*添加线模型
dev_disp_text ('请画出要识别的直线', 'window', 'top', 'left', 'black', [], [])
draw_line (WindowHandle, Row1, Column1, Row2, Column2)
add_metrology_object_line_measure (MetrologyHandle1, Row1, Column1, Row2, Column2, 20, 5, 1, 30, [
apply_metrology_model (GrayImage, MetrologyHandle1)
get_metrology_object_result_contour (Contour, MetrologyHandle1, 0, 'all', 1.5)
get_metrology_object_result (MetrologyHandle1, 0, 'all', 'result_type', 'all_param', LineModel)
add_metrology_object_line_measure (MetrologyHandle, LineModel[0], LineModel[1], LineModel[2], Line
*设置模型的坐标原点
set_metrology_model_param (MetrologyHandle, 'reference_system', [ShapeModelRow, ShapeModelColumn,0
get_metrology_object_model_contour (Contour1, MetrologyHandle, 0, 1.5)
*保存计量模型
MeasureModel := './Model/Measure.mtr'
write_metrology_model (MetrologyHandle, MeasureModel)
```

1.创建形状模板

2.创建卡尺测量模型，找寻边，将边设置为新的测量模型（设置模型的坐标原点，即将测量卡尺绑定到形状模板上），

保证以后检测到的模型都是相对于此原点进行测量，如有缩放（旋转）也可缩放（旋转）

Find_Model：

```
get_image_size (Image, Width, Height)
dev_open_window (0, 0, Width/2, Height/2, 'black', WindowHandle)
a:=Index+1
fwrite_string (FileHandle, '第'+a+'张图'+'    ')
rgb1_to_gray (Image, GrayImage)
*模板匹配
find_shape_model (GrayImage, ShapeModelModelID, 0, rad(360), 0.5, 3, 0, 'least_squares', 5, 0.
*取得匹配项的坐标
dev_display_shape_matching_results (ShapeModelModelID, 'red', FindRow, FindColumn, FindAngle,
*求匹配项个数
FindNumber:=|FindRow|
fwrite_string (FileHandle, '共匹配'+FindNumber+'个目标')
fnew_line (FileHandle)
dev_display (Image)
*循环处理每个匹配项
for i := 0 to FindNumber-1 by 1
    b:=i+1
    fwrite_string (FileHandle, '第'+b+'个目标')
    fnew_line (FileHandle)
    *测量
    align_metrology_model (MetrologyHandle, FindRow[i], FindColumn[i], FindAngle[i])
    get_metrology_object_model_contour (Contour2, MetrologyHandle, 'all', 1.5)
    apply_metrology_model (GrayImage, MetrologyHandle)
    get_metrology_object_result_contour (Contour3, MetrologyHandle, 'all', 'all', 1.5)
    dev_set_color ('red')
    dev_set_line_width (2)
    dev_display (Contour3)

    *取得第一条线结果
    get_metrology_object_result (MetrologyHandle, 0, 'all', 'result_type', 'all_param', Line1)
    distance_pp (Line1[0], Line1[1], Line1[2], Line1[3], Distance1)
    dev_disp_text ('Line1='+Distance1, 'image', (Line1[0]+Line1[2])/2, (Line1[1]+Line1[3])/2,
    fwrite_string (FileHandle, 'Line1='+Distance1)
    fnew_line (FileHandle)
    fnew_line (FileHandle)
endfor
dump_window_image (Image1, WindowHandle)
write_image (Image1, 'png', 0, '第'+a+'张图')
```

形状匹配，矫正卡尺（将测量工具与平移、旋转绑定），执行测量（测量和拟合测量模型的所有测量对象的几何形状），使用点到点的测量。

halcon输出.txt文件和图片

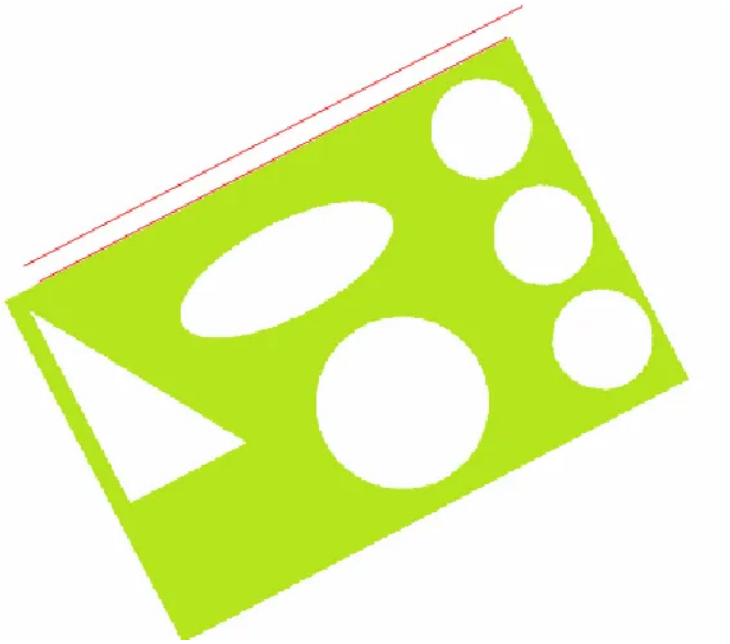
```
*准备记录数据
open_file ('note.TXT', 'append', FileHandle)
get_system_time (MSecond, Second, Minute, Hour, Day, YDay, Month, Year)
fwrite_string (FileHandle, Day+'日'+Hour+'时'+Minute+'分'+Second+'秒')
fnew_line (FileHandle)
fnew_line (FileHandle)
fwrite_string (FileHandle, '共匹配'+FindNumber+'个目标')
fwrite_string (FileHandle, '第'+a+'张图'+'   ')
fwrite_string (FileHandle, 'Line1='+Distance1)
*将窗口内容写入一个图像对象（更新图片）
dump_window_image (Image1, WindowHandle)
*保存图片
write_image (Image1, 'png', 0, '第'+a+'张图')
```

2日16时37分43秒

第1张图
共匹配1个目标
第1个目标
Line1=209.576

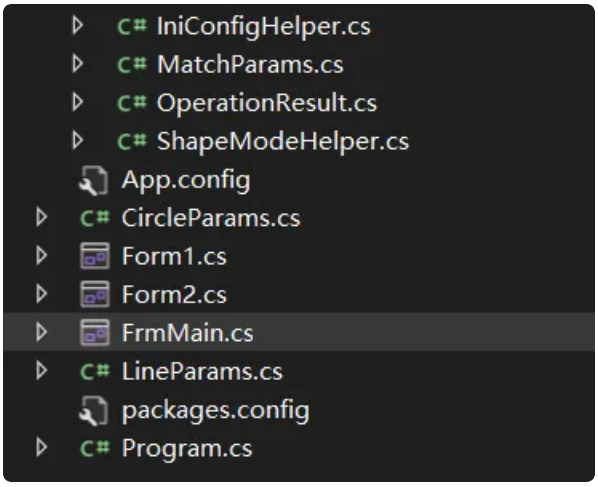
第2张图
共匹配1个目标
第1个目标
Line1=208.892

第3张图
共匹配1个目标
第1个目标
Line1=209.308



c#

项目列表



iniconfighelper.cs	INI类，INI（Initialization）配置文件是一种常见的文本文件格式，用于存储和管理程序配置信息。它通常由一系列节（section）和键值对（key-value pairs）组成。
matchparams.cs	模板匹配参数类，将在shapemodehelper.cs中存储模板、加载模板中实例化和方便在几何定位模块中修改参数，还有通过查找模板名来判断会不会报错
operationresult.cs	操作结果类，实例化成检测和检测线，圆，后续有附上讲解
shapemodehelper.cs	形状模板匹配类，halcon导出直接使用，创建模板，查找模板，清除模板，存储模板，获取所有的模板，加载模板
circleparams.cs	圆类，关于圆、卡尺的一些参数
form1.cs	登陆界面窗体，实现登录和退出
form2.cs	表格类，实现导出PDF和Excel
frmmain.cs	主窗体
lineparams.cs	线类，关于线、卡尺的一些参数
program.cs	主入口

界面展示：

登录界面：


用户名：
密码：
退出
登录

C#

```
public partial class Form1 : Form
{
    public Form1()
    {
        InitializeComponent();
        textBox1.Text = "aa";
        textBox2.Text = "33";
    }
    private void button1_Click(object sender, EventArgs e)
    {
        //构造连接数据库的字符串
        SQLiteConnectionStringBuilder connectionString = new SQLiteConnect
ionStringBuilder();
        connectionString.DataSource = "D://sjk//lwmb.db";
        //连接数据库
        SQLiteConnection conn = new SQLiteConnection(connectionString.ToSt
ring());
        conn.Open();
        //查询语句select
        string sql = string.Format("select* from User");
        SQLiteDataAdapter da = new SQLiteDataAdapter(sql, conn);
        //获取数据
        DataSet ds = new DataSet();
        da.Fill(ds);
        DataTable re = ds.Tables[0];
        //及时释放资源
        da.Dispose();
        conn.Close();
        int wzdflwjsygfww = 0;
        int wzdflwjsygfw = 0;
        foreach (DataRow v in re.Rows)  //循环表中数据
        {
            string dlname = v["name"].ToString();
            string dlmark = v["password"].ToString();
            if (textBox1.Text == dlname)
            {
                if (textBox2.Text == dlmark)
                {
                    wzdflwjsygfw = 1;
                }
                wzdflwjsygfww = 1;
            }
        }
        if (wzdflwjsygfww == 0)
```

```
43          {
44              MessageBox.Show("账号不存在");
45          }
46          else
47          {
48              if (wzdflwjsygfw == 0)
49              {
50                  MessageBox.Show("密码不正确");
51              }
52              else
53              {
54                  MessageBox.Show("登陆成功");
55                  this.DialogResult = DialogResult.OK;
56              }
57          }
58      }
59  }
```

有俩种实现方式，他这种打开表格，逐个查找

我使用的是mysql匹配（可用SqliteHelper类，实现增删改查）

```C#
private void button1_Click(object sender, EventArgs e)
{
    SQLiteHelpers.Open();
    if (this.textname.Text == "" || this.textname.Text == "")
    {
        MessageBox.Show("请你输入你的用户名或密码！", "提示", MessageBoxButtons.OK, MessageBoxIcon.Warning);
    }
    else
    {
        SQLiteParameter[] parameter = new SQLiteParameter[]
        {
            new SQLiteParameter("Name", this.textname.Text.ToString().Trim()),
            new SQLiteParameter("Pwd", this.textpwd.Text.ToString().Trim()),
            new SQLiteParameter("ID", 0.ToString())
        };

        string sql = "SELECT * FROM table1 WHERE Pwd = @Pwd AND Name = @Name AND ID=@ID";
        //DataSet dataSet = SQLiteHelpers.ExecuteDataSet(sql, parameter);
        SQLiteDataReader dr = SQLiteHelpers.ExecuteReader(sql, parameter);
        if (dr.Read())
        {
            Form1 main = new Form1();
            main.Show();
            this.Hide();
        }
        else
        {
            MessageBox.Show("你输入的密码错误，请重新输入！", "提示", MessageBoxButtons.OK, MessageBoxIcon.Information);
        }
    }
}
```

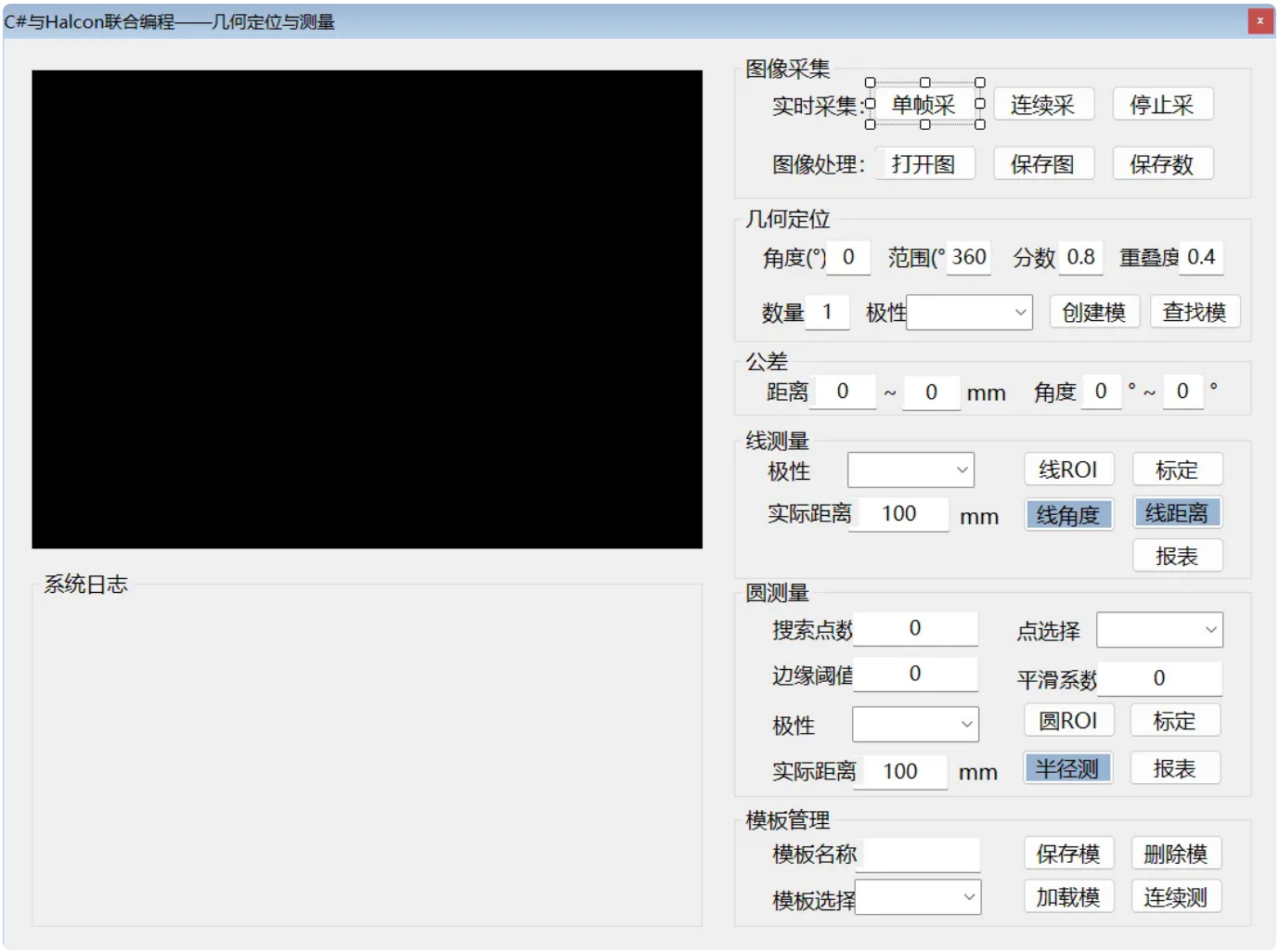
个人觉得加个填写验证码的过程

C#

```
using System;
using System.Windows.Forms;

namespace CaptchaExample
{
    public partial class MainForm : Form
    {
        private string captchaCode; // 随机生成的验证码

        public MainForm()
        {
            InitializeComponent();
        }

        private void MainForm_Load(object sender, EventArgs e)
        {
            // 初始化界面
            GenerateCaptcha();
            UpdateButtonState();
        }

        private void GenerateCaptcha()
        {
            // 生成随机的验证码
            Random random = new Random();
            int captchaNumber = random.Next(1000, 9999); // 生成4位随机数作为验证码
            captchaCode = captchaNumber.ToString();
        }

        private void UpdateButtonState()
        {
            // 根据验证码输入是否匹配来启用或禁用按钮
            if (textBoxCaptcha.Text == captchaCode)
            {
                buttonSubmit.Enabled = true;
            }
            else
            {
                buttonSubmit.Enabled = false;
            }
        }


```

```
        private void textBoxCaptcha_TextChanged(object sender, EventArgs
e)
        {
            // 当验证码文本框内容发生变化时更新按钮状态
            UpdateButtonState();
        }

        private void buttonSubmit_Click(object sender, EventArgs e)
        {
            // 提交按钮点击事件处理逻辑
            MessageBox.Show("验证码匹配成功，提交按钮被点击！");
            GenerateCaptcha(); // 重新生成验证码
            UpdateButtonState(); // 更新按钮状态
            textBoxCaptcha.Text = ""; // 清空文本框
        }

        private void pictureBoxCaptcha_Click(object sender, EventArgs e)
        {
            // 验证码图片点击事件处理逻辑
            GenerateCaptcha(); // 重新生成验证码
        }
    }
```

主界面：



图像采集

用groupbox装，又时间和能力方便实现多线程

单采

```C#
private void btn_OneShot_Click(object sender, EventArgs e)
{
    var result = image.GrabImage(ref hWindowHandle, ref hImage);

    if (result.IsSuccess)
    {
        AddLog(0, "图像采集成功");
    }
    else
    {
        AddLog(1, "图像采集失败: " + result.ErrorMsg);
    }

}
```

连采

C#

```
#region 连续采集单帧图像
private void btn_Grab_Click(object sender, EventArgs e)
{
    if (image.isCamOK == false)
    {
        var result = image.OpenCam(camName);
        if (result.IsSuccess)
        {
            AddLog(0, "相机打开成功");
        }
        else
        {
            AddLog(1, "相机打开失败：" + result.ErrorMsg);
            return;
        }
    }
    cts = new CancellationTokenSource();
    Task.Run(() =>
                    {
                        GrabImageThread();
                    }, cts.Token);
    AddLog(0, "开始连续彩图");
}

private void GrabImageThread()
{
    while (!cts.IsCancellationRequested)
    {
        if (image.isCamOK)
        {
            var result = image.GrabImage(ref hWindowHandle, ref hImage);

            if (result.IsSuccess)
            {
                //处理
            }

            else
            {
                AddLog(1, "连续彩图失败");
                break;
            }
        }
        else
```

```
45            {
46                AddLog(1, "连续彩图失败");
47                break;
48            }
49
50        }
51    }
```

停采

```C#
//停止采集图像
private void btn_StopGrab_Click(object sender, EventArgs e)
{
    AddLog(0, "停止连续彩图");

    cts.Cancel();
}
```

显示日志信息

C#

```
#region MyRegion 显示信息

private string CurrentTime
{
    get { return DateTime.Now.ToString("HH:mm:ss"); }
}

private void AddLog(int index, string log)
{
    if (this.lst_Info.InvokeRequired)
    {
        this.lst_Info.Invoke(new Action(() =>
        {
            ListViewItem listViewItem = new ListViewItem(" " + CurrentTime, index);
            listViewItem.SubItems.Add(log);
            this.lst_Info.Items.Insert(0, listViewItem);
        }));
    }
    else
    {
        ListViewItem listViewItem = new ListViewItem(" " + CurrentTime, index);
        listViewItem.SubItems.Add(log);
        this.lst_Info.Items.Insert(0, listViewItem);
    }
}
```

图像操作

C#

```
//打开图像
private void btn_OpenImage_Click(object sender, EventArgs e)
{
    using (OpenFileDialog openFileDialog = new OpenFileDialog())
    {
        openFileDialog.Filter = ".bmp|*.bmp|.png|*.png|.jpg|*.jpg|.tif|*.tif|.jpeg|*.jpeg";
        openFileDialog.Multiselect = false;

        if (openFileDialog.ShowDialog() == DialogResult.OK)
        {
            string path = openFileDialog.FileName;

            var result = image.ReadImage(path, ref hWindowHandle, ref hImage);
            if (result.IsSuccess)

            {
                AddLog(0, "打开图像成功");
            }

            else
            {

                AddLog(1, "打开图像失败：" + result.ErrorMsg);
            }
        }
    }
}

//保存图像
private void btn_SaveImage_Click(object sender, EventArgs e)
{
    using (SaveFileDialog saveFileDialog = new SaveFileDialog())
    {
        saveFileDialog.Filter = ".bmp|*.bmp|.png|*.png|.jpg|*.jpg|.jpeg|*.jpeg";
        saveFileDialog.DefaultExt = ".bmp";
        if (saveFileDialog.ShowDialog() == DialogResult.OK)
        {
            var result = image.SaveImage(saveFileDialog.FileName, saveFileDialog.FileName.Substring(saveFileDialog.FileName.LastIndexOf('.') + 1), hImage);
            if (result.IsSuccess)
```

```
40                {
41
42                    AddLog(0, "图像保存成功");
43                }
44                else
45                {
46
47                    AddLog(1, "图像保存失败：" + result.ErrorMsg);
48                }
49            }
50        }
51    }
```

这个是形状模板的一些参数设置

形状匹配及其附加功能

后续几个模块

就不一一介绍，可能用不上

讲一下识别过程

三种识别过程都类似，这里用圆来讲述

自动测量元件的边缘，有线条长度，角度和圆半径，

创建判断类

实例检测的判断类，里面包装寻找此原件的形状模板匹配，和线、圆匹配拟合，从中拿到线、圆的一些坐标，长度，半径等参数，和将参数传进List内

后续需要使用可以直接将List的值显示出来

导出为Excel和PDF的原理相差无几

C#

```
public void ExportDataToExcel(DataGridView myDGV)
{
    string path = "";
    SaveFileDialog saveDialog = new SaveFileDialog();
    //aveDialog.DefaultExt = ".pdf";
    //saveDialog.Filter = "Text documents (.pdf)|*.pdf";
    saveDialog.Title = "请选择要导出的位置";
    saveDialog.Filter = "Excel文件| *.xlsx;*.xls;*.XLSX|PDF文件|*.pdf|Word文件|*.word";
    saveDialog.ShowDialog();
    path = saveDialog.FileName;
    if (path.IndexOf(":") < 0) return; //判断是否点击取消
    try
    {
        Thread.Sleep(1000);
        StreamWriter sw = new StreamWriter(path, false, Encoding.GetEncoding("gb2312"));
        StringBuilder sb = new StringBuilder();
        //写入标题
        for (int k = 0; k < myDGV.Columns.Count; k++)
        {
            if (myDGV.Columns[k].Visible)//导出可见的标题
            {
                //"\t"就等于键盘上的Tab,加个"\t"的意思是: 填充完后进入下一个单元格.
                sb.Append(myDGV.Columns[k].HeaderText.ToString().Trim() + "\t");
            }
        }
        sb.Append(Environment.NewLine);//换行
                                       //写入每行数值
        for (int i = 0; i < myDGV.Rows.Count - 1; i++)
        {
            System.Windows.Forms.Application.DoEvents();
            for (int j = 0; j < myDGV.Columns.Count; j++)
            {
                if (myDGV.Columns[j].Visible)//导出可见的单元格
                {
                    //注意单元格有一定的字节数量限制,如果超出,就会出现两个单元格的内容是一模一样的.
                    //具体限制是多少字节,没有作深入研究.
                    sb.Append(myDGV.Rows[i].Cells[j].Value.ToString().Trim() + "\t");
                }
```

```
39                 }
40                 sb.Append(Environment.NewLine); //换行
41             }
42             sw.Write(sb.ToString());
43             sw.Flush();
44             sw.Close();
45             MessageBox.Show(path + "，导出成功", "系统提示", MessageBoxButtons.O
    K);
46         }
47         catch (Exception ex)
48         {
49             MessageBox.Show(ex.Message);
50         }
51     }
```

遍历方式不同

```C#
private void ExportExcels(string fileName, DataGridView myDGV)
{
    string saveFileName = "";
    SaveFileDialog saveDialog = new SaveFileDialog();
    saveDialog.DefaultExt = "xls";
    saveDialog.Filter = "Excel文件|*.xls";
    saveDialog.FileName = fileName;
    saveDialog.ShowDialog();
    saveFileName = saveDialog.FileName;
    if (saveFileName.IndexOf(":") < 0) return; //被点了取消
    Microsoft.Office.Interop.Excel.Application xlApp = new Microsoft.Office.Interop.Excel.Application();
    if (xlApp == null)
    {
        MessageBox.Show("无法创建Excel对象，可能您的机子未安装Excel");
        return;
    }
    Microsoft.Office.Interop.Excel.Workbooks workbooks = xlApp.Workbooks;
    Microsoft.Office.Interop.Excel.Workbook workbook = workbooks.Add(Microsoft.Office.Interop.Excel.XlWBATemplate.xlWBATWorksheet);
    Microsoft.Office.Interop.Excel.Worksheet worksheet = (Microsoft.Office.Interop.Excel.Worksheet)workbook.Worksheets[1];//取得sheet1

                                                        //写入标题
    for (int i = 0; i < myDGV.ColumnCount; i++)
    {
        worksheet.Cells[1, i + 1] = myDGV.Columns[i].HeaderText;
    }
    //写入数值
    for (int r = 0; r < myDGV.Rows.Count; r++)
    {
        for (int i = 0; i < myDGV.ColumnCount; i++)
        {
            worksheet.Cells[r + 2, i + 1] = myDGV.Rows[r].Cells[i].Value;
        }
        System.Windows.Forms.Application.DoEvents();
    }
    worksheet.Columns.EntireColumn.AutoFit();//列宽自适应
    if (saveFileName != "")
    {
        try
        {
            workbook.Saved = true;
            workbook.SaveCopyAs(saveFileName);
```

```
41          }
42          catch (Exception ex)
43          {
44              MessageBox.Show("导出文件时出错,文件可能正被打开！\n" + ex.Messag
    e);
45          }
46      }
47      xlApp.Quit();
48      GC.Collect();//强行销毁
49      MessageBox.Show("文件：  " + fileName + ".xls 保存成功", "信息提示", Messa
    geBoxButtons.OK, MessageBoxIcon.Information);
50  }
```

.txt文件

C#

```
private void ExportToTxt(DataGridView dataGridView, string filePath)
{
    if (check_signal == false)
    {
        MessageBox.Show("未能成功定位图片位置");
    }
    else
    {
        using (StreamWriter sw = new StreamWriter(filePath))
        {
            // 写入列标题
            for (int i = 0; i < dataGridView.Columns.Count; i++)
            {
                sw.Write(dataGridView.Columns[i].HeaderText);
                if (i < dataGridView.Columns.Count - 1)
                {
                    sw.Write("\t");
                }
            }
            sw.WriteLine();

            // 写入数据行
            for (int row = 0; row < dataGridView.Rows.Count; row++)
            {
                for (int col = 0; col < dataGridView.Columns.Count; col++)
                {
                    sw.Write(dataGridView.Rows[row].Cells[col].Value);
                    if (col < dataGridView.Columns.Count - 1)
                    {
                        sw.Write("\t");
                    }
                }
                sw.WriteLine();
            }
        }
    }

}
```