**《基于子母机协同的高效铁轨检修机操作平台(V1.0)》源代码**



**武汉理工大学**

2019年3月

源程序代码：

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

using System.IO.Ports;

using System.Threading;

using System.Threading.Tasks;

using System.Data.SqlClient;

using System.Globalization;

using AForge.Video.DirectShow;

using MathNet.Numerics;

using MathNet.Numerics.LinearAlgebra.Double;

using MathNet.Numerics.IntegralTransforms;

using Emgu.CV;

using Emgu.CV.Structure;

using Emgu.CV.Util;

using System.Numerics;

namespace HostComputerForRail

{

public partial class Form1 : Form

{

private DateTime TimeStart = DateTime.Now;

bool bool\_start = false;

public Form1()

{

InitializeComponent();

}

private void Form1\_Load(object sender, EventArgs e)

{

try

{

//图像识别部分

VideoCapture\_ImageRecognize\_Load();

//系统信息部分

timer\_System.Start();

//实时监控部分

MonitorCamera\_Load();

//倾角仪传输部分

comboBox\_Inclinometer\_Load();

SerialPort\_Inclinometer1.DataReceived += new System.IO.Ports.SerialDataReceivedEventHandler(this.SerialPort\_DataReceived1);

SerialPort\_Inclinometer2.DataReceived += new System.IO.Ports.SerialDataReceivedEventHandler(this.SerialPort\_DataReceived2);

//底栏状态调整

if (bool\_haveCamera)

{

toolStripStatusLabel\_Camera.Text = "摄像头已连接";

}

else

{

toolStripStatusLabel\_Camera.Text = "摄像头未连接";

}

}

catch (Exception ex)

{

toolStripStatusLabel\_State.Text = "错误：" + ex.Message;

}

}

private void pictureBox\_Start\_Click(object sender, EventArgs e)

{

try

{

bool\_start = true;

timer\_Main.Start();

timer\_FFT.Start();

}

catch (Exception ex)

{

toolStripStatusLabel\_State.Text = "错误：" + ex.Message;

}

}

private void pictureBox\_End\_Click(object sender, EventArgs e)

{

try

{

bool\_start = false;

timer\_Main.Stop();

timer\_FFT.Stop();

Thread.Sleep(100);

}

catch (Exception ex)

{

toolStripStatusLabel\_State.Text = "错误：" + ex.Message;

}

}

private void timer\_Main\_Tick(object sender, EventArgs e)

{

try

{

if (bool\_start)

{

a\_AfterTransform\_Before = a\_AfterTransform;

if (bool\_startIntegrate)

{

DateTime\_Before = DateTime.Now;

}

else

{

DateTime\_Before = DateTime\_Now;

}

velocity\_Before = velocity;

DateTime\_Now = DateTime.Now;

Transform();

RemoveInit();

Integration();

//TODO: 刷新数据

label\_Inclinometer1\_Ax.Text = a\_AfterTransform[0, 0].ToString("F6");

label\_Inclinometer1\_Ay.Text = a\_AfterTransform[0, 1].ToString("F6");

label\_Inclinometer1\_Az.Text = a\_AfterTransform[0, 2].ToString("F6");

label\_Inclinometer1\_THETAx.Text = Angle[0, 0].ToString("F6");

label\_Inclinometer1\_THETAy.Text = Angle[0, 1].ToString("F6");

label\_Inclinometer1\_THETAz.Text = Angle[0, 2].ToString("F6");

label\_Inclinometer2\_Ax.Text = a\_AfterTransform[1, 0].ToString("F6");

label\_Inclinometer2\_Ay.Text = a\_AfterTransform[1, 1].ToString("F6");

label\_Inclinometer2\_Az.Text = a\_AfterTransform[1, 2].ToString("F6");

label\_Inclinometer2\_THETAx.Text = Angle[1, 0].ToString("F6");

label\_Inclinometer2\_THETAy.Text = Angle[1, 1].ToString("F6");

label\_Inclinometer2\_THETAz.Text = Angle[1, 2].ToString("F6");

//TODO: 更改 label\_IncrementalTime 的计算逻辑

label\_IncrementalTime.Text = (DateTime.Now - TimeStart).TotalMilliseconds / 1000 + "";

if (!((a[0, 0] == 0 && a[0, 1] == 0 && a[0, 2] == 0 && Angle[0, 0] == 0 && Angle[0, 1] == 0 && Angle[0, 2] == 0) ||

(a[1, 0] == 0 && a[1, 1] == 0 && a[1, 2] == 0 && Angle[1, 0] == 0 && Angle[1, 1] == 0 && Angle[1, 2] == 0)))

{

chart1\_Run();

statusStrip\_Bottom.BackColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(20)))));

SQLconnect();

//TODO: 改变toolStripStatusLabel\_State.Image

}

}

}

catch (Exception ex)

{

toolStripStatusLabel\_State.Text = "错误：" + ex.Message;

}

}

/\*

\* —————————————————————————————————————————图像识别程序段——————————————————————————————————————————

\*/

private VideoCapture VideoCapture\_ImageRecognize;

private Mat frame;

private int index\_ImageRecognize = 2;

private void VideoCapture\_ImageRecognize\_Load()

{

try

{

VideoCapture\_ImageRecognize = new VideoCapture(index\_ImageRecognize);

VideoCapture\_ImageRecognize.ImageGrabbed += ProcessFrame;

frame = new Mat();

VideoCapture\_ImageRecognize.Start();

}

catch (Exception ex)

{

toolStripStatusLabel\_Camera.Text = "错误：" + ex.Message;

}

}

private void ProcessFrame(object sender, EventArgs e)

{

try

{

if (VideoCapture\_ImageRecognize != null && VideoCapture\_ImageRecognize.Ptr != IntPtr.Zero)

{

VideoCapture\_ImageRecognize.Retrieve(frame, 0);

Image<Bgr, Byte> img = frame.ToImage<Bgr, Byte>();

Image<Gray, Byte> grayImage = img.Convert<Gray, Byte>();

Image<Gray, Byte> addImage = grayImage.ThresholdBinary(new Gray(65), new Gray(255));

Image<Gray, Byte> cannyGray = grayImage.Canny(90, 150);

cannyGray = cannyGray.Not();

Image<Bgr, Byte> finalImage = img.Add(img,cannyGray);

imageBox1.Image = finalImage;

//imageBox1.Image = cannyGray;

//imageBox1.Image = frame;

//TODO: 如果 cannyGray 出现白色，将时间记录在数据库中

}

}

catch (Exception ex)

{

toolStripStatusLabel\_Camera.Text = "错误：" + ex.Message;

}

}

/\*

\* —————————————————————————————————————————不平顺数据处理——————————————————————————————————————————

\*/

private double[,] a\_AfterTransform = new double[2, 3];

private double[,] a\_AfterTransform\_Before = new double[2, 3];

private double[,] velocity = new double[2, 3];

private double[,] velocity\_Before = new double[2, 3];

private double[,] displacement = new double[2, 3];

private bool bool\_startIntegrate = false;

private DateTime DateTime\_Before;

private DateTime DateTime\_Now;

private int zeroPointForVelocity;

private double[,] error = new double[2, 3]; //零点漂移误差阈值

private double[,] a\_AfterTransform\_Init = new double[2, 3]; //初始误差

private List<double>[,] FixInit\_Data = { { new List<double>(), new List<double>(), new List<double>() }, { new List<double>(), new List<double>(), new List<double>() } };

private void Transform()

{

try

{

for (int number = 0; number < 2; number++)

{

// 建立数组

MathNet.Numerics.LinearAlgebra.Matrix<double> a\_Matrix = DenseMatrix.OfArray(new double[,] { { a[number, 0], a[number, 1], a[number, 2], 1 } });

MathNet.Numerics.LinearAlgebra.Matrix<double> Trx1 = DenseMatrix.OfArray(new double[,]

{

{1, 0, 0, 0},

{0, Math.Cos(Angle[number,0]\*Math.PI/180), Math.Sin(Angle[number,0]\*Math.PI/180), 0},

{0, -Math.Sin(Angle[number,0]\*Math.PI/180), Math.Cos(Angle[number,0]\*Math.PI/180), 0},

{0, 0, 0, 1},

});

MathNet.Numerics.LinearAlgebra.Matrix<double> Try1 = DenseMatrix.OfArray(new double[,]

{

{Math.Cos(Angle[number,1]\*Math.PI/180), 0, -Math.Sin(Angle[number,1]\*Math.PI/180), 0},

{0, 1, 0, 0},

{Math.Sin(Angle[number,1]\*Math.PI/180), 0, Math.Cos(Angle[number,1]\*Math.PI/180), 0},

{0, 0, 0, 1},

});

MathNet.Numerics.LinearAlgebra.Matrix<double> Trz1 = DenseMatrix.OfArray(new double[,]

{

{Math.Cos(Angle[number,2]\*Math.PI/180), Math.Sin(Angle[number,2]\*Math.PI/180), 0, 0},

{-Math.Sin(Angle[number,2]\*Math.PI/180), Math.Cos(Angle[number,2]\*Math.PI/180), 0, 0},

{0, 0, 1, 0},

{0, 0, 0, 1},

});

MathNet.Numerics.LinearAlgebra.Matrix<double> Trx3 = DenseMatrix.OfArray(new double[,]

{

{1, 0, 0, 0},

{0, Math.Cos(-Angle[number,0]\*Math.PI/180), Math.Sin(-Angle[number,0]\*Math.PI/180), 0},

{0, -Math.Sin(-Angle[number,0]\*Math.PI/180), Math.Cos(-Angle[number,0]\*Math.PI/180), 0},

{0, 0, 0, 1},

});

MathNet.Numerics.LinearAlgebra.Matrix<double> Try3 = DenseMatrix.OfArray(new double[,]

{

{Math.Cos(-Angle[number,1]\*Math.PI/180), 0, -Math.Sin(-Angle[number,1]\*Math.PI/180), 0},

{0, 1, 0, 0},

{Math.Sin(-Angle[number,1]\*Math.PI/180), 0, Math.Cos(-Angle[number,1]\*Math.PI/180), 0},

{0, 0, 0, 1},

});

MathNet.Numerics.LinearAlgebra.Matrix<double> Trz3 = DenseMatrix.OfArray(new double[,]

{

{Math.Cos(-Angle[number,2]\*Math.PI/180), Math.Sin(-Angle[number,2]\*Math.PI/180), 0, 0},

{-Math.Sin(-Angle[number,2]\*Math.PI/180), Math.Cos(-Angle[number,2]\*Math.PI/180), 0, 0},

{0, 0, 1, 0},

{0, 0, 0, 1},

});

// 三角变换

MathNet.Numerics.LinearAlgebra.Matrix<double> T1 = Trx1 \* Try1 \* Trz1;

MathNet.Numerics.LinearAlgebra.Matrix<double> T3 = Trz3 \* Try3 \* Trx3;

MathNet.Numerics.LinearAlgebra.Matrix<double> T2 = DenseMatrix.OfArray(new double[,]

{

{1, 0, 0, 0},

{0, 1, 0, 0},

{0, 0, 1, 0},

{0, 0, -1, 1}

});

MathNet.Numerics.LinearAlgebra.Matrix<double> T = T1 \* T2 \* T3;

a\_Matrix = a\_Matrix \* T;

for (int i = 0; i < 3; i++)

{

a\_AfterTransform[number, i] = a\_Matrix[0, i];

}

}

}

catch (Exception ex)

{

toolStripStatusLabel\_Inclinometer.Text = "错误：" + ex.Message;

}

}

private void Integration()

{

try

{

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 3; j++)

{

// 积分求解位移和速度

velocity[i, j] = velocity\_Before[i, j]

+ 0.5 \* (a\_AfterTransform\_Before[i, j] + a\_AfterTransform[i, j]) \* (DateTime\_Now - DateTime\_Before).TotalMilliseconds / 1000;

displacement[i, j] = displacement[i, j]

+ 0.5 \* (a\_AfterTransform\_Before[i, j] + a\_AfterTransform[i, j]) \* (DateTime\_Now - DateTime\_Before).TotalMilliseconds / 1000;

// 去除速度的零点漂移

if (a\_AfterTransform[i, j] == 0)

{

zeroPointForVelocity++;

}

else

{

zeroPointForVelocity = 0;

}

if (zeroPointForVelocity >= 30)

{

velocity\_Before[i, j] = 0;

}

}

}

}

catch (Exception ex)

{

toolStripStatusLabel\_Inclinometer.Text = "错误：" + ex.Message;

}

}

private void RemoveInit()

{

try

{

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 3; j++)

{

//获得误差

a\_AfterTransform[i, j] = a\_AfterTransform[i, j] - a\_AfterTransform\_Init[i, j];

//去除零点漂移

if (Math.Abs(a\_AfterTransform[i, j]) < error[i, j])

{

a\_AfterTransform[i, j] = 0;

}

}

}

}

catch (Exception ex)

{

toolStripStatusLabel\_Inclinometer.Text = "错误：" + ex.Message;

}

}

//频域分析

private static int Samples\_num = 250;

readonly Complex[] sample\_Ay1 = new Complex[Samples\_num];

private Complex[] sample\_Ay2 = new Complex[Samples\_num];

private Complex[] sample\_Az1 = new Complex[Samples\_num];

private Complex[] sample\_Az2 = new Complex[Samples\_num];

private Complex[] sample\_θx1 = new Complex[Samples\_num];

private Complex[] sample\_θx2 = new Complex[Samples\_num];

private int add\_num = 0;

private double[,] sample\_data = new double[6, Samples\_num];

Thread thread\_sample;

private delegate void delegate\_FFT();

private void FFT(/\*object state\*/)

{

Fourier.Forward(sample\_Ay1);

}

private void PlotFftAnalys()

{

BeginInvoke(new delegate\_FFT(label3\_Start));

timer\_FFT.Stop();

for (int i = 0; i < Samples\_num; i++)

{

sample\_Ay1[i] = new Complex(sample\_data[0, i], 0);

}

BeginInvoke(new delegate\_FFT(label3\_Doing));

try

{

FFT();

}

catch (Exception ex)

{

MessageBox.Show("" + ex.Message);

}

BeginInvoke(new delegate\_FFT(label3\_Finish));

BeginInvoke(new delegate\_FFT(UIchange\_FFT));

}

private void label3\_Start()

{

label3.Text = "初始化中";

}

private void label3\_Doing()

{

label3.Text = "FFT计算中";

}

private void label3\_Finish()

{

label3.Visible = false;

}

private void UIchange\_FFT()

{

crtFft.Series["Frequency"].Points.Clear();

for (int i = 0; i < sample\_Ay1.Length / 4; i++)

{

double mag = (2.0 / Samples\_num) \* (Math.Abs(Math.Sqrt(Math.Pow(sample\_Ay1[i].Real, 2) +

Math.Pow(sample\_Ay1[i].Imaginary, 2))));

double hzPerSample = 20 / Samples\_num;

crtFft.Series["Frequency"].Points.AddXY(hzPerSample \* i, mag);

}

}

private void timer\_FFT\_Tick(object sender, EventArgs e)

{

if (add\_num < Samples\_num)

{

sample\_data[0, add\_num] = a[0, 1];

sample\_data[1, add\_num] = a[1, 1];

sample\_data[2, add\_num] = a[0, 2];

sample\_data[3, add\_num] = a[1, 2];

sample\_data[4, add\_num] = Angle[0, 0];

sample\_data[5, add\_num] = Angle[1, 0];

}

else

{

for (int i = 0; i < Samples\_num - 1; i++)

{

for (int j = 0; j < 6; j++)

{

sample\_data[j, i] = sample\_data[j, i + 1];

}

sample\_data[0, Samples\_num - 1] = a[0, 1];

sample\_data[1, Samples\_num - 1] = a[1, 1];

sample\_data[2, Samples\_num - 1] = a[0, 2];

sample\_data[3, Samples\_num - 1] = a[1, 2];

sample\_data[4, Samples\_num - 1] = Angle[0, 0];

sample\_data[5, Samples\_num - 1] = Angle[1, 0];

}

add\_num = 0;

label3.Text = "正在处理";

thread\_sample = new Thread(PlotFftAnalys);

thread\_sample.IsBackground = true;

thread\_sample.Start();

}

add\_num += 1;

}

private void pictureBox\_fixSensor\_Click(object sender, EventArgs e)

{

try

{

timer\_Sensor.Start();

timer\_FixInit.Start();

label\_Inclinometer1\_Ax.Text = "倾角仪正在校准";

label\_Inclinometer1\_Ay.Text = "倾角仪正在校准";

label\_Inclinometer1\_Az.Text = "倾角仪正在校准";

label\_Inclinometer1\_THETAx.Text = "倾角仪正在校准";

label\_Inclinometer1\_THETAy.Text = "倾角仪正在校准";

label\_Inclinometer1\_THETAz.Text = "倾角仪正在校准";

label\_Inclinometer2\_Ax.Text = "倾角仪正在校准";

label\_Inclinometer2\_Ay.Text = "倾角仪正在校准";

label\_Inclinometer2\_Az.Text = "倾角仪正在校准";

label\_Inclinometer2\_THETAx.Text = "倾角仪正在校准";

label\_Inclinometer2\_THETAy.Text = "倾角仪正在校准";

label\_Inclinometer2\_THETAz.Text = "倾角仪正在校准";

}

catch (Exception ex)

{

toolStripStatusLabel\_Inclinometer.Text = "错误：" + ex.Message;

}

}

private void timer\_FixInit\_Tick(object sender, EventArgs e)

{

try

{

timer\_Sensor.Stop();

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 3; j++)

{

// 获取样本的无偏估计, 及其标准偏差

double sum = 0;

double sigma = 0;

int k = 0;

a\_AfterTransform\_Init[i, j] = FixInit\_Data[i, j].Average();

foreach (double x in FixInit\_Data[i, j])

{

sum = sum + x;

sigma = sigma + Math.Pow((x - a\_AfterTransform\_Init[i, j]), 2);

k++;

}

error[i, j] = 3 \* Math.Sqrt(sigma / ((k - 1) \* k)) \* 12;

FixInit\_Data[i, j].Clear();

}

}

timer\_FixInit.Stop();

label\_Inclinometer1\_Ax.Text = a\_AfterTransform\_Init[0, 0] + "";

label\_Inclinometer1\_Ay.Text = a\_AfterTransform\_Init[0, 1] + "";

label\_Inclinometer1\_Az.Text = a\_AfterTransform\_Init[0, 2] + "";

label\_Inclinometer1\_THETAx.Text = "校准完成";

label\_Inclinometer1\_THETAy.Text = "校准完成";

label\_Inclinometer1\_THETAz.Text = "校准完成";

label\_Inclinometer2\_Ax.Text = a\_AfterTransform\_Init[1, 0] + "";

label\_Inclinometer2\_Ay.Text = a\_AfterTransform\_Init[1, 1] + "";

label\_Inclinometer2\_Az.Text = a\_AfterTransform\_Init[1, 2] + "";

label\_Inclinometer2\_THETAx.Text = "校准完成";

label\_Inclinometer2\_THETAy.Text = "校准完成";

label\_Inclinometer2\_THETAz.Text = "校准完成";

}

catch (Exception ex)

{

toolStripStatusLabel\_Inclinometer.Text = "错误：" + ex.Message;

}

}

private void timer\_Sensor\_Tick(object sender, EventArgs e)

{

try

{

Transform();

for (int i = 0; i < 2; i++)

{

for (int j = 0; j < 3; j++)

{

FixInit\_Data[i, j].Add(a\_AfterTransform[i, j]);

}

}

}

catch (Exception ex)

{

toolStripStatusLabel\_Inclinometer.Text = "错误：" + ex.Message;

}

}

private void pictureBox1\_Click\_1(object sender, EventArgs e)

{

label3.Text = "正在处理";

thread\_sample = new Thread(PlotFftAnalys);

thread\_sample.IsBackground = true;

thread\_sample.Start();

//PlotFftAnalys();

}

/\*

\* —————————————————————————————————————————数据库传输部分——————————————————————————————————————————

\*/

private string sqlDate;

private void SQLconnect()

{

// SQL server

string connsql = "server=FU-QINGCHEN\\SQLEXPRESS;integrated security=SSPI;database=Test";

try

{

using (SqlConnection mySQL = new SqlConnection())

{

mySQL.ConnectionString = connsql;

// 打开数据库连接

mySQL.Open();

// 向数据库中插入数据

var format = "yyyy-MM-dd HH:mm:ss:fffffff";

var stringDate = DateTime.Now.ToString(format);

var convertedBack = DateTime.ParseExact(stringDate, format, CultureInfo.InvariantCulture);

sqlDate = "insert Inclination\_OriginDate(DateTimes,"

+ "Accelerate1\_X,Accelerate1\_Y,Accelerate1\_Z," + "Inclination1\_X,Inclination1\_Y,Inclination1\_Z,"

+ "Accelerate2\_X,Accelerate2\_Y,Accelerate2\_Z," + "Inclination2\_X,Inclination2\_Y,Inclination2\_Z"

+ ")values(SYSDATETIME(),"

+ a\_AfterTransform[0, 0] + "," + a\_AfterTransform[0, 1] + "," + a\_AfterTransform[0, 2] + "," + Angle[0, 0] + "," + Angle[0, 1] + "," + Angle[0, 2] + ","

+ a\_AfterTransform[1, 0] + "," + a\_AfterTransform[1, 1] + "," + a\_AfterTransform[1, 2] + "," + Angle[1, 0] + "," + Angle[1, 1] + "," + Angle[1, 2]

+ ")";

// 建立一个命令

SqlCommand sqlCommand = new SqlCommand(sqlDate, mySQL);

// 执行命令

sqlCommand.ExecuteNonQuery();

}

}

catch (Exception ex)

{

toolStripStatusLabel\_SQL.Text = "错误：" + ex.Message;

}

finally

{

toolStripStatusLabel\_SQL.Text = "数据库已连接";

}

// Azure SQL

SqlConnectionStringBuilder sqlConnectionStringBuilder\_Azure = new SqlConnectionStringBuilder();

sqlConnectionStringBuilder\_Azure.DataSource = "mysampleserver.database.chinacloudapi.cn";

sqlConnectionStringBuilder\_Azure.UserID = "WHUT";

sqlConnectionStringBuilder\_Azure.Password = "0121618380615Fqc";

sqlConnectionStringBuilder\_Azure.InitialCatalog = "RailOriginDate";

try

{

using(SqlConnection sqlConnection = new SqlConnection(sqlConnectionStringBuilder\_Azure.ConnectionString))

{

sqlConnection.Open();

var format = "yyyy-MM-dd HH:mm:ss:fffffff";

var stringDate = DateTime.Now.ToString(format);

var convertedBack = DateTime.ParseExact(stringDate, format, CultureInfo.InvariantCulture);

sqlDate = "insert Inclination\_OriginDate(DateTimes,"

+ "Accelerate1\_X,Accelerate1\_Y,Accelerate1\_Z," + "Inclination1\_X,Inclination1\_Y,Inclination1\_Z,"

+ "Accelerate2\_X,Accelerate2\_Y,Accelerate2\_Z," + "Inclination2\_X,Inclination2\_Y,Inclination2\_Z"

+ ")values(SYSDATETIME(),"

+ a\_AfterTransform[0, 0] + "," + a\_AfterTransform[0, 1] + "," + a\_AfterTransform[0, 2] + "," + Angle[0, 0] + "," + Angle[0, 1] + "," + Angle[0, 2] + ","

+ a\_AfterTransform[1, 0] + "," + a\_AfterTransform[1, 1] + "," + a\_AfterTransform[1, 2] + "," + Angle[1, 0] + "," + Angle[1, 1] + "," + Angle[1, 2]

+ ")";

SqlCommand sqlCommand = new SqlCommand(sqlDate, sqlConnection);

sqlCommand.ExecuteNonQuery();

}

}

catch(Exception ex)

{

toolStripStatusLabel\_SQL.Text = "错误：" + ex.Message;

}

finally

{

toolStripStatusLabel\_SQL.Text = "数据库已连接";

}

}

/\*

\* —————————————————————————————————————————倾角仪传输部分——————————————————————————————————————————

\*/

private bool bool\_Inclinometer1 = false, bool\_Inclinometer2 = false;

string[] serialPortName;

SerialPort SerialPort\_Inclinometer1 = new SerialPort();

SerialPort SerialPort\_Inclinometer2 = new SerialPort();

private double[,] a = new double[2, 3], Angle = new double[2, 3];

private int serialPortNumber;

//查询串口并加载

private void comboBox\_Inclinometer\_Load()

{

serialPortName = SerialPort.GetPortNames();

if (serialPortName == null)

{

toolStripStatusLabel\_Inclinometer.Text = "无串口连接";

}

else

{

foreach (string name in serialPortName)

{

comboBox\_Inclinometer1.Items.Add(name);

comboBox\_Inclinometer1.SelectedIndex = -1;

comboBox\_Inclinometer2.Items.Add(name);

comboBox\_Inclinometer2.SelectedIndex = -1;

}

toolStripStatusLabel\_Inclinometer.Text = "请选择倾角仪串口";

}

SerialPort\_Inclinometer1.BaudRate = 115200;

SerialPort\_Inclinometer2.BaudRate = 115200;

}

//关闭串口，释放资源

private void serialPort\_Close(SerialPort serialPort)

{

if (serialPort.IsOpen == true)

{

serialPort.Dispose();

serialPort.Close();

}

}

private void comboBox\_Inclinometer1\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboBox\_Inclinometer1.SelectedIndex >= 0)

{

SerialPort\_Inclinometer1.PortName = serialPortName[comboBox\_Inclinometer1.SelectedIndex];

}

bool\_Inclinometer1 = true;

if (bool\_Inclinometer2 == true)

{

toolStripStatusLabel\_Inclinometer.Text = "倾角仪已连接";

}

serialPort\_Close(SerialPort\_Inclinometer1);

SerialPort\_Inclinometer1.Open();

//timer\_Main.Start();

}

private void comboBox\_Inclinometer2\_SelectedIndexChanged(object sender, EventArgs e)

{

if (comboBox\_Inclinometer2.SelectedIndex >= 0)

{

SerialPort\_Inclinometer2.PortName = serialPortName[comboBox\_Inclinometer2.SelectedIndex];

}

bool\_Inclinometer2 = true;

if (bool\_Inclinometer1 == true)

{

toolStripStatusLabel\_Inclinometer.Text = "倾角仪已连接";

}

serialPort\_Close(SerialPort\_Inclinometer2);

SerialPort\_Inclinometer2.Open();

//timer\_Main.Start();

}

//以下获取串口数据部分, 改编于传感器厂商开源代码

delegate void UpdateData1(byte[] byteData);//声明一个委托

delegate void UpdateData2(byte[] byteData);//声明一个委托

byte[] RxBuffer1 = new byte[1000];

byte[] RxBuffer2 = new byte[1000];

UInt16 usRxLength1 = 0;

UInt16 usRxLength2 = 0;

private double[] LastTime1 = new double[10];

private double[] LastTime2 = new double[10];

//接收数据

private void SerialPort\_DataReceived1(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)

{

byte[] byteTemp = new byte[1000];

UInt16 usLength = 0;

usLength = (UInt16)SerialPort\_Inclinometer1.Read(RxBuffer1, usRxLength1, 700);

usRxLength1 += usLength;

while (usRxLength1 >= 11)

{

UpdateData1 Update = new UpdateData1(DecodeData1);

RxBuffer1.CopyTo(byteTemp, 0);

if (!((byteTemp[0] == 0x55) & ((byteTemp[1] & 0x50) == 0x50)))

{

for (int i = 1; i < usRxLength1; i++) RxBuffer1[i - 1] = RxBuffer1[i];

usRxLength1--;

continue;

}

if (((byteTemp[0] + byteTemp[1] + byteTemp[2] + byteTemp[3] + byteTemp[4] + byteTemp[5] + byteTemp[6] + byteTemp[7] + byteTemp[8] + byteTemp[9]) & 0xff) == byteTemp[10])

this.Invoke(Update, byteTemp);

for (int i = 11; i < usRxLength1; i++) RxBuffer1[i - 11] = RxBuffer1[i];

usRxLength1 -= 11;

}

Thread.Sleep(10);

}

private void SerialPort\_DataReceived2(object sender, System.IO.Ports.SerialDataReceivedEventArgs e)

{

byte[] byteTemp = new byte[1000];

UInt16 usLength = 0;

usLength = (UInt16)SerialPort\_Inclinometer2.Read(RxBuffer2, usRxLength2, 700);

usRxLength2 += usLength;

while (usRxLength2 >= 11)

{

UpdateData2 Update2 = new UpdateData2(DecodeData2);

RxBuffer2.CopyTo(byteTemp, 0);

if (!((byteTemp[0] == 0x55) & ((byteTemp[1] & 0x50) == 0x50)))

{

for (int i = 1; i < usRxLength2; i++) RxBuffer2[i - 1] = RxBuffer2[i];

usRxLength2--;

continue;

}

if (((byteTemp[0] + byteTemp[1] + byteTemp[2] + byteTemp[3] + byteTemp[4] + byteTemp[5] + byteTemp[6] + byteTemp[7] + byteTemp[8] + byteTemp[9]) & 0xff) == byteTemp[10])

this.Invoke(Update2, byteTemp);

for (int i = 11; i < usRxLength2; i++) RxBuffer2[i - 11] = RxBuffer2[i];

usRxLength2 -= 11;

}

Thread.Sleep(10);

}

//解码数据

private void DecodeData1(byte[] byteTemp)

{

serialPortNumber = 0;

double[] Data = new double[4];

double TimeElapse = (DateTime.Now - TimeStart).TotalMilliseconds / 1000;

Data[0] = BitConverter.ToInt16(byteTemp, 2);

Data[1] = BitConverter.ToInt16(byteTemp, 4);

Data[2] = BitConverter.ToInt16(byteTemp, 6);

Data[3] = BitConverter.ToInt16(byteTemp, 8);

switch (byteTemp[1])

{

case 0x51: //加速度输出

Data[0] = Data[0] / 32768.0 \* 16;

Data[1] = Data[1] / 32768.0 \* 16;

Data[2] = Data[2] / 32768.0 \* 16;

a[serialPortNumber, 0] = Data[0];

a[serialPortNumber, 1] = Data[1];

a[serialPortNumber, 2] = Data[2];

if ((TimeElapse - LastTime1[1]) < 0.1) return;

LastTime1[1] = TimeElapse;

break;

case 0x53: //角度输出

Data[0] = Data[0] / 32768.0 \* 180;

Data[1] = Data[1] / 32768.0 \* 180;

Data[2] = Data[2] / 32768.0 \* 180;

Angle[serialPortNumber, 0] = Data[0];

Angle[serialPortNumber, 1] = Data[1];

Angle[serialPortNumber, 2] = Data[2];

if ((TimeElapse - LastTime1[3]) < 0.1) return;

LastTime1[3] = TimeElapse;

break;

default:

break;

}

}

private void DecodeData2(byte[] byteTemp)

{

serialPortNumber = 1;

double[] Data = new double[4];

double TimeElapse = (DateTime.Now - TimeStart).TotalMilliseconds / 1000;

Data[0] = BitConverter.ToInt16(byteTemp, 2);

Data[1] = BitConverter.ToInt16(byteTemp, 4);

Data[2] = BitConverter.ToInt16(byteTemp, 6);

Data[3] = BitConverter.ToInt16(byteTemp, 8);

switch (byteTemp[1])

{

case 0x51: //加速度输出

Data[0] = Data[0] / 32768.0 \* 16;

Data[1] = Data[1] / 32768.0 \* 16;

Data[2] = Data[2] / 32768.0 \* 16;

a[serialPortNumber, 0] = Data[0];

a[serialPortNumber, 1] = Data[1];

a[serialPortNumber, 2] = Data[2];

if ((TimeElapse - LastTime2[1]) < 0.1) return;

LastTime2[1] = TimeElapse;

break;

case 0x53: //角度输出

Data[0] = Data[0] / 32768.0 \* 180;

Data[1] = Data[1] / 32768.0 \* 180;

Data[2] = Data[2] / 32768.0 \* 180;

Angle[serialPortNumber, 0] = Data[0];

Angle[serialPortNumber, 1] = Data[1];

Angle[serialPortNumber, 2] = Data[2];

if ((TimeElapse - LastTime2[3]) < 0.1) return;

LastTime2[3] = TimeElapse;

break;

default:

break;

}

}

/\*

\* ————————————————————————————————————————————实时监控部分————————————————————————————————————————————

\*/

private bool bool\_haveCamera; //判断是否有可用的摄像头

private FilterInfoCollection VideoInputDeviceCollection; //调出所有可用设备

private VideoCaptureDevice VideoCaptureDevice\_MonitorCamera; //视频源

private void MonitorCamera\_Load()

{

try

{

VideoInputDeviceCollection = new FilterInfoCollection(FilterCategory.VideoInputDevice);

VideoCaptureDevice\_MonitorCamera = new VideoCaptureDevice

(VideoInputDeviceCollection[1].MonikerString);

videoSourcePlayer\_MonitorCamera.VideoSource = VideoCaptureDevice\_MonitorCamera;

videoSourcePlayer\_MonitorCamera.Start();

toolStripStatusLabel\_Camera.Text = "摄像头已连接";

}

catch (Exception ex)

{

toolStripStatusLabel\_Camera.Text = "错误：" + ex.Message;

}

finally

{

bool\_haveCamera = true;

}

}

/\*

\* ————————————————————————————————————————————实时图表设计————————————————————————————————————————————

\*/

private bool frequency\_start = false;

private void chart1\_Run()

{

chart1.Series[0].Points.AddXY(DateTime.Now.Millisecond.ToString(), a\_AfterTransform[0, 1]);

chart1.Series[1].Points.AddXY(DateTime.Now.Millisecond.ToString(), a\_AfterTransform[1, 1]);

chart1.Series[2].Points.AddXY(DateTime.Now.Millisecond.ToString(), a\_AfterTransform[0, 2]);

chart1.Series[3].Points.AddXY(DateTime.Now.Millisecond.ToString(), a\_AfterTransform[1, 2]);

chart1.Series[4].Points.AddXY(DateTime.Now.Millisecond.ToString(), Angle[0, 0]);

chart1.Series[5].Points.AddXY(DateTime.Now.Millisecond.ToString(), Angle[1, 0]);

if (chart1.Series[0].Points.Count >= 500)

{

chart1.Series[0].Points.RemoveAt(0);

chart1.Series[1].Points.RemoveAt(0);

chart1.Series[2].Points.RemoveAt(0);

chart1.Series[3].Points.RemoveAt(0);

chart1.Series[4].Points.RemoveAt(0);

chart1.Series[5].Points.RemoveAt(0);

frequency\_start = true;

}

if (frequency\_start)

{

}

}

/\*

\* ————————————————————————————————————————————UI界面设计————————————————————————————————————————————

\*/

private void toolStripStatusLabel\_ControlCenter\_MouseEnter(object sender, EventArgs e)

{

toolStripStatusLabel\_ControlCenter.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

}

private void toolStripStatusLabel\_ControlCenter\_MouseLeave(object sender, EventArgs e)

{

toolStripStatusLabel\_ControlCenter.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel\_OriginData\_MouseEnter(object sender, EventArgs e)

{

toolStripStatusLabel\_OriginData.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

}

private void toolStripStatusLabel\_OriginData\_MouseLeave(object sender, EventArgs e)

{

toolStripStatusLabel\_OriginData.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel\_DataSolve\_MouseEnter(object sender, EventArgs e)

{

toolStripStatusLabel\_DataSolve.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

}

private void toolStripStatusLabel\_DataSolve\_MouseLeave(object sender, EventArgs e)

{

toolStripStatusLabel\_DataSolve.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel\_ControlCenter\_Click(object sender, EventArgs e)

{

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

}

private void pictureBox1\_Click(object sender, EventArgs e)

{

System.Environment.Exit(0);

}

private void pictureBox2\_Click(object sender, EventArgs e)

{

this.WindowState = FormWindowState.Minimized;

}

private void toolStripStatusLabel\_ControlCenter\_Click\_1(object sender, EventArgs e)

{

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

toolStripStatusLabel7.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

toolStripStatusLabel5.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel3\_Click(object sender, EventArgs e)

{

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

toolStripStatusLabel7.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

toolStripStatusLabel5.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel\_OriginData\_Click(object sender, EventArgs e)

{

toolStripStatusLabel7.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

toolStripStatusLabel5.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel7\_Click(object sender, EventArgs e)

{

toolStripStatusLabel7.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

toolStripStatusLabel5.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void timer\_System\_Tick(object sender, EventArgs e)

{

label\_SystemTime.Text = DateTime.Now + "";

}

private void toolStripStatusLabel\_DataSolve\_Click(object sender, EventArgs e)

{

toolStripStatusLabel5.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

toolStripStatusLabel7.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

private void toolStripStatusLabel5\_Click(object sender, EventArgs e)

{

toolStripStatusLabel5.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(0)))), ((int)(((byte)(122)))), ((int)(((byte)(204)))));

toolStripStatusLabel7.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

toolStripStatusLabel3.ForeColor =

System.Drawing.Color.FromArgb(((int)(((byte)(153)))), ((int)(((byte)(153)))), ((int)(((byte)(153)))));

}

}

}