ภาคผนวก ข
ส่วนของโปรแกรม C#

โปรแกรม C#

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using ZedGraph;
using OtherLibs;
using CxCore;
using Cv;
namespace asdRobot3
{
  public partial class Form1 : Form
 {
 #region define val

VidFormat vidFmt = new VidFormat(500, 350, 30);
 int camnull = 0;
private IplImage image;
private IplImage image3;
private IplImage gray;
private IplImage gray2;
private IplImage image2;
private IplImage img2scl;
private IplImage image5;
private IplImage img5scl;
//private CvPoint pt;
```

```
private CvPoint pt1, pt2;   //plot Up line
private CvPoint pt7, pt8;   //plot Up line
private CvPoint pt3, pt4;   //plot Down line
private CvPoint pt9, pt10;   //plot Down line
private CvCapture capture;
private CvCapture capture2;
 private CvScalar s1, s2, s3;
private double PI = 3.1415926535;
private int x1, x2, x3, x4;
private int u1, u2;
 private int d1, d2;
 private int t1, t2, t3, t4;
private double angle = 90, prev_angle = 90, angleCam2;
private int line1 = 0, line1_x = 0, line1_y = 0;
private int line2 = 0, line2_x = 0, line2_y = 0;
private int line3 = 0, line3_x = 0, line3_y = 0
 private int line4 = 0, line4_x = 0, line4_y = 0;
int OK = 0;
string Check1;
string Check2;
 string Torque;
private string str, str1, str2, str3;
private byte[] bCMD = new byte[3];
#endregion
public Form1()
{
InitializeComponent();
 }
int TickStart1, TickStart2, TickStart3;
int intMode = 0;
```

```
 private void Form1_Load(object sender, EventArgs e)
{
#region Comport Baudrate index
comboBox1.Items.Add("COM1");
comboBox1.Items.Add("COM2");
comboBox1.Items.Add("COM3");
 comboBox1.Items.Add("COM4");
 comboBox1.Items.Add("COM5");
 comboBox1.Items.Add("COM6");
 comboBox1.Items.Add("COM7");
 comboBox1.Items.Add("COM8");
 comboBox1.Items.Add("COM9");
 comboBox1.Items.Add("COM10");
 comboBox1.SelectedIndex = 2;
comboBox2.Items.Add("2400");
comboBox2.Items.Add("4800");
comboBox2.Items.Add("9600");
comboBox2.Items.Add("14400");
comboBox2.Items.Add("19200");
comboBox2.SelectedIndex = 2;
#endregion
 #region ZedGraph
GraphPane myPane1 = zedGraphControl1.GraphPane;
myPane1.Title.Text = "Image1";
myPane1.XAxis.Title.Text = "Time";
myPane1.YAxis.Title.Text = "Error";
 GraphPane myPane2 = zedGraphControl2.GraphPane;
myPane2.Title.Text = "Image2";
myPane2.XAxis.Title.Text = "Time";
 myPane2.YAxis.Title.Text = "Error";
```

```
GraphPane myPane3 = zedGraphControl3.GraphPane;
myPane3.Title.Text = "Torque";
 myPane3.XAxis.Title.Text = "Time";
myPane3.YAxis.Title.Text = "Error";
 RollingPointPairList list1 = new RollingPointPairList(60000);
RollingPointPairList list2 = new RollingPointPairList(60000);
RollingPointPairList list3 = new RollingPointPairList(60000);
RollingPointPairList list4 = new RollingPointPairList(60000);
RollingPointPairList list5 = new RollingPointPairList(60000);
RollingPointPairList list6 = new RollingPointPairList(60000);
LineItem curve1 = myPane1.AddCurve("SetPoint", list1, Color.Black, SymbolType.None)
LineItem cueve2 = myPane1.AddCurve("FeedBack", list2, Color.Red, SymbolType.None);

LineItem curve3 = myPane2.AddCurve("SetPoint", list3, Color.Black,
SymbolType.None);LineItem cueve4 = myPane2.AddCurve("FeedBack", list4, Color.Red,
SymbolType.None);LineItem curve5 = myPane3.AddCurve("SetPoint", list5, Color.Black,
SymbolType.None); LineItem cueve6 = myPane3.AddCurve("FeedBack", list6, Color.Red,
SymbolType.None);
 myPane1.XAxis.Scale.Min = 0;
 myPane1.XAxis.Scale.Max = 30;
 myPane1.YAxis.Scale.Min = 0;
 myPane1.YAxis.Scale.Max = 120;
 myPane1.XAxis.Scale.MinorStep = 1;
 myPane1.XAxis.Scale.MajorStep = 5;

 myPane2.XAxis.Scale.Min = 0;
 myPane2.XAxis.Scale.Max = 30;
 myPane2.YAxis.Scale.Min = 0;
 myPane2.YAxis.Scale.Max = 120;
 myPane2.XAxis.Scale.MinorStep = 1;
```

```
 myPane2.XAxis.Scale.MajorStep = 5;

 myPane3.XAxis.Scale.Min = 0;
 myPane3.XAxis.Scale.Max = 30;
 myPane3.XAxis.Scale.MinorStep = 1;
 myPane3.XAxis.Scale.MajorStep = 5;

zedGraphControl1.AxisChange();
zedGraphControl2.AxisChange();
zedGraphControl3.AxisChange();
TickStart1 = Environment.TickCount;
TickStart2 = Environment.TickCount;
TickStart3 = Environment.TickCount;
#endregion
#region Initial Value
textBox15.Text = "20000";
textBox17.Text = "80";
textBox18.Text = "100";
textBox19.Text = "100";
textBox30.Text = "10";
textBox31.Text = "10";
#endregion
timer3.Stop();
timer5.Stop();
}
private void Draw1(string SetPoint1, string Current1)
{
Current1 = textBox10.Text;
double intSetPoint1;
double intCurrent1;
```

```
double.TryParse(SetPoint1, out intSetPoint1);
double.TryParse(Current1, out intCurrent1);
if (zedGraphControl1.GraphPane.CurveList.Count <= 0)
return;
LineItem curve1 = zedGraphControl1.GraphPane.CurveList[0] as LineItem;
LineItem curve2 = zedGraphControl1.GraphPane.CurveList[1] as LineItem;
if (curve1 == null)
return;
if (curve2 == null)
return;
IPointListEdit list1 = curve1.Points as IPointListEdit;
IPointListEdit list2 = curve2.Points as IPointListEdit;
if (list1 == null)
return;
if (list2 == null)
return;
double time = (Environment.TickCount - TickStart1) / 1000.0;
list1.Add(time, intSetPoint1);
list2.Add(time, intCurrent1);
Scale xScale = zedGraphControl1.GraphPane.XAxis.Scale;
if (intMode == 1)
{
xScale.Max = time + xScale.MajorStep;
xScale.Min = 0;
}
else
{
xScale.Max = time + xScale.MajorStep;
xScale.Min = xScale.Max - 30;
}
zedGraphControl1.AxisChange();
```

```
zedGraphControl1.Invalidate();
}
private void Draw2(string SetPoint2, string Current2)
{
Current2 = textBox9.Text;
double intSetPoint2;
double intCurrent2;
double.TryParse(SetPoint2, out intSetPoint2);
double.TryParse(Current2, out intCurrent2);
if (zedGraphControl2.GraphPane.CurveList.Count <= 0)
return;

LineItem curve3 = zedGraphControl2.GraphPane.CurveList[0] as LineItem;
LineItem curve4 = zedGraphControl2.GraphPane.CurveList[1] as LineItem
if (curve3 == null)
return;
if (curve4 == null)
return
IPointListEdit list3 = curve3.Points as IPointListEdit;
IPointListEdit list4 = curve4.Points as IPointListEdit;
if (list3 == null)
return;

if (list4 == null)
return;

double time = (Environment.TickCount - TickStart2) / 1000.0;
list3.Add(time, intSetPoint2);
list4.Add(time, intCurrent2);
Scale xScale = zedGraphControl2.GraphPane.XAxis.Scale;
```

```
if (intMode == 1)
{
xScale.Max = time + xScale.MajorStep;
xScale.Min = 0;
}
else
{
xScale.Max = time + xScale.MajorStep;
xScale.Min = xScale.Max - 30;
}
zedGraphControl2.AxisChange();
zedGraphControl2.Invalidate();
private void Draw3(string SetPoint3, string Current3
{
Current3 = textBox16.Text;
double intSetPoint3;
double intCurrent3;
double.TryParse(SetPoint3, out intSetPoint3);
double.TryParse(Current3, out intCurrent3);
if (zedGraphControl1.GraphPane.CurveList.Count <= 0)
return;
LineItem curve5 = zedGraphControl3.GraphPane.CurveList[0] as LineItem;
LineItem curve6 = zedGraphControl3.GraphPane.CurveList[1] as LineItem;
if (curve5 == null)
return;
if (curve6 == null)
return;
IPointListEdit list5 = curve5.Points as IPointListEdit;
IPointListEdit list6 = curve6.Points as IPointListEdit;
```

```
if (list5 == null)
return;
if (list6 == null)
return;

double time = (Environment.TickCount - TickStart3) / 1000.0;

list5.Add(time, intSetPoint3);
list6.Add(time, intCurrent3);

Scale xScale = zedGraphControl3.GraphPane.XAxis.Scale;
if (intMode == 1)
{
 xScale.Max = time + xScale.MajorStep;
 xScale.Min = 0;
}
else
{
 xScale.Max = time + xScale.MajorStep;
 xScale.Min = xScale.Max - 30;
}

zedGraphControl3.AxisChange();
zedGraphControl3.Invalidate();

            }

            private void Form1_FormClosed(object sender, FormClosedEventArgs e)
            {
                // release unmanaged resources
```

```
            if (image.ptr != IntPtr.Zero) cxcore.CvReleaseImage(ref image);
            if (gray.ptr != IntPtr.Zero) cxcore.CvReleaseImage(ref gray);
            if (capture.ptr != IntPtr.Zero) highgui.CvReleaseCapture(ref capture);
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            IplImage frame;
            IplImage frame2;
            int i;
            CvSize size;
            size.height = 250;
            size.width = 300;

            // capture a frame
            frame = highgui.CvQueryFrame(ref capture);
            frame2 = highgui.CvQueryFrame(ref capture2);

            // check if the frame valid
            if (frame.ptr == IntPtr.Zero)
            {
 // optional stop capturing
timer1.Stop();
MessageBox.Show("Invalid Frame. Capturing has been stopped.");
return;
}
if (frame2.ptr == IntPtr.Zero)

{
 // optional stop capturing
```

```
 timer1.Stop();
 MessageBox.Show("Invalid Frame2. Capturing has been stopped.");
 return;
 }
// check if this is the first run
if (image.ptr == IntPtr.Zero)
{
// allocate all the buffers
image = cxcore.CvCreateImage(cxcore.CvGetSize(ref frame), 8, 3);
image.origin = frame.origin;
image3 = cxcore.CvCreateImage(cxcore.CvGetSize(ref frame2), 8, 3);
 image.origin = frame.origin;
gray = cxcore.CvCreateImage(cxcore.CvGetSize(ref frame), 8, 1);
 gray2 = cxcore.CvCreateImage(cxcore.CvGetSize(ref frame2), 8, 1);
image2 = cxcore.CvCreateImage(cxcore.CvGetSize(ref frame), 8, 3);
img2scl = cxcore.CvCreateImage(size, 8, 3)
image5 = cxcore.CvCreateImage(cxcore.CvGetSize(ref frame2), 8, 3);
img5scl = cxcore.CvCreateImage(size, 8, 3);
}

t1 = trackBar1.Value;
t2 = trackBar2.Value;
t3 = trackBar8.Value
t4 = trackBar7.Value;
textBox2.Text = (trackBar1.Value.ToString());
textBox6.Text = (trackBar2.Value.ToString());
textBox7.Text = (trackBar3.Value.ToString());
textBox8.Text = (trackBar4.Value.ToString());
textBox11.Text = (trackBar5.Value.ToString());
textBox13.Text = (trackBar6.Value.ToString());
```

```
textBox14.Text = (trackBar8.Value.ToString());
textBox12.Text = (trackBar7.Value.ToString()
cxcore.CvCopy(ref frame, ref image);
cxcore.CvCopy(ref frame2, ref image5);
// create gray level image from source
cv.CvCvtColor(ref image, ref gray, cvtypes.CV_BGR2GRAY)
cv.CvCvtColor(ref image5, ref gray2, cvtypes.CV_BGR2GRAY);
cv.CvSmooth(ref gray,ref gray,cvtypes.CV_GAUSSIAN_5x5 ,9,9)
cv.CvSmooth(ref gray2, ref gray2, cvtypes.CV_GAUSSIAN_5x5, 9, 9);
cv.CvThreshold(ref gray, ref gray, trackBar2.Value, 255, cv.CV_THRESH_BINARY);
cv.CvThreshold(ref gray2, ref gray2, trackBar7.Value, 255, cv.CV_THRESH_BINARY);
// apply the operator
cv.CvCanny(ref gray, ref gray, trackBar1.Value, trackBar1.Value * 2, 5);
cv.CvCanny(ref gray2, ref gray2, trackBar8.Value, trackBar8.Value * 2, 5);
cv.CvCvtColor(ref gray, ref image2, cvtypes.CV_GRAY2BGR);
cv.CvCvtColor(ref gray2, ref image5, cvtypes.CV_GRAY2BGR);
// Down   Line Algorithm
for (i = 0; i < image2.width-1; i++)
x1 = image2.width - 1 - i;
d1 = trackBar3.Value;
if ((x1 < image2.width) && (x1 > 0) && (d1 < image2.height) && (d1 >= 0))
{
s1 = cxcore.CvGet2D(ref image2, d1, x1);
textBox3.Text = s1.val1.ToString();
textBox4.Text = s1.val2.ToString();
textBox5.Text = s1.val3.ToString();
if (s1.val1 == 255)
{
line1 = 1;
line1_x = x1;
```

```
line1_y = d1;
i = 640;
break;
}
else
{
line1 = 0;
line1_x = 0;
line1_y = 0;
}
}
}
// Down   Line Algorithm
for (i = 0; i < image5.width - 1; i++)
{
x3 = image5.width - 1 - i;
d2 = trackBar6.Value;
if ((x3 < image5.width) && (x3 > 0) && (d2 < image5.height) && (d2 >= 0))
{
s2 = cxcore.CvGet2D(ref image5, d2, x3);
textBox3.Text = s2.val1.ToString();
textBox4.Text = s2.val2.ToString();
textBox5.Text = s2.val3.ToString();
if (s2.val1 == 255)
{
line3 = 1;
line3_x = x3;
line3_y = d2;
i = 640;
break;
```

```
}
else
{
line3 = 0;
line3_x = 0;
line3_y = 0;
}
}
}
// Up   Line Algorithm
for (i = 0; i < image2.width - 1; i+
{
x2 = image2.width - 1 - i;
u1 = trackBar4.Value;
if ((x2 < image2.width) && (x2 > 0) && (u1 < image2.height) && (u1 >= 0))
{
s1 = cxcore.CvGet2D(ref image2, u1, x2);

if (s1.val1 == 255)
{
line2 = 1;
line2_x = x2;
line2_y = u1;
i = 640;
break;
}
else
{
line2 = 0;
line2_x = 0;
```

```
line2_y = 0;
}
}
}
 // Up   Line Algorithm
for (i = 0; i < image5.width - 1; i++)
{
x4 = image5.width - 1 - i;
u2 = trackBar5.Value;
if ((x4 < image5.width) && (x4 > 0) && (u2 < image5.height) && (u2 >= 0))
s2 = cxcore.CvGet2D(ref image5, u2, x4);
if (s2.val1 == 255)
{
line4 = 1;
line4_x = x4;
line4_y = u2;
i = 640;
break;
}
Else
{
line4 = 0;
line4_x = 0;
line4_y = 0;
}
}
}

//Color Blue
s3.val1 = 255;
```

```
s3.val2 = 0;
s3.val3 = 0;

//Color Green
s2.val1 = 0;
s2.val2 = 255;
s2.val3 = 0;

//Color Red
s1.val1 = 0;
s1.val2 = 0;
s1.val3 = 255;

pt1.x = line1_x;
pt1.y = trackBar3.Value;

pt2.x = 639;
pt2.y = trackBar3.Value;

pt7.x = line3_x;
pt7.y = trackBar6.Value;
 pt8.x = 639;
 pt8.y = trackBar6.Value;
 if (line1 == 1)
 textBox7.BackColor = Color.Lime;
  else
 textBox7.BackColor = Color.White;
 if (line2 == 1)
 textBox8.BackColor = Color.Lime;
 else
```

```
textBox8.BackColor = Color.White;
pt3.x = line2_x;
pt3.y = trackBar4.Value;
pt4.x = 639;
pt4.y = trackBar4.Value
pt9.x = line4_x;
pt9.y = trackBar5.Value;
pt10.x = 639;
pt10.y = trackBar5.Value;
angle = (Math.Atan2(Convert.ToDouble(line2_y - line1_y), Convert.ToDouble(line2_x -
line1_x))) * 180 / PI;
angleCam2 = (Math.Atan2(Convert.ToDouble(line4_y - line3_y), Convert.ToDouble(line4_x -
line3_x))) * 180 / PI;

 int index1 = 0;
 str = (angle).ToString();
 index1 = str.IndexOf(".");
 if (str.Length > index1 + 3)
 str1 = str.Substring(0, index1 + 3);



 int index2 = 0;
 str2 = (angleCam2).ToString();
 index2 = str2.IndexOf(".");
 if (str2.Length > index2 + 3)
 str3 = str2.Substring(0, index2 + 3);

 textBox9.Text = str3;
 textBox10.Text = str1;
```

```
 if (Check1 == "1")
{
if (Connect_btn.Text.CompareTo("Disable") == 0 && Check2 == "1")
{
timer1.Stop();
serialPort1.Write("Aix[" + str + "]" + "Aiy[" + str2 + "]");
OK = 1
timer1.Start();
}
prev_angle = angle;
if (((angle > 88) && (angle < 92)) && ((angleCam2 > 88) && (angleCam2 < 92)) && (OK ==
1))
{
timer1.Stop();
serialPort1.Write("Az[" + textBox30.Text + "][" + textBox31.Text + "]");
timer1.Start();
timer5.Start();
Check2 = "0";
}
}
if (checkBox1.Checked)
{
cxcore.CvLine(ref image2, pt1, pt2, s1, 1, 8);
cxcore.CvLine(ref image2, pt3, pt4, s2, 1, 8);
cxcore.CvLine(ref image5, pt7, pt8, s1, 1, 8);
cxcore.CvLine(ref image5, pt9, pt10, s2, 1, 8);
cxcore.CvFlip(ref image2, ref image2, 0);
cxcore.CvFlip(ref image5, ref image5, 0);
cv.CvResize(ref image2, ref img2scl,cv.CV_INTER_LINEAR);
```

```
cv.CvResize(ref image5, ref img5scl, cv.CV_INTER_LINEAR);
//this.pictureBox1.Image = (Bitmap)image2;
//this.pictureBox2.Image = (Bitmap)image5;
this.pictureBox3.Image = (Bitmap)img2scl;
this.pictureBox4.Image = (Bitmap)img5scl;
}
else
{
 cxcore.CvLine(ref image, pt1, pt2, s1, 1, 8);
 cxcore.CvLine(ref image, pt3, pt4, s2, 1, 8);
 cxcore.CvLine(ref image3, pt7, pt8, s1, 1, 8);
 cxcore.CvLine(ref image3, pt9, pt10, s2, 1, 8);
 cxcore.CvFlip(ref image, ref image, 0);
 cxcore.CvFlip(ref image5, ref image5, 0);
 cv.CvResize(ref image, ref img2scl, cv.CV_INTER_LINEAR);
 cv.CvResize(ref image5, ref img5scl, cv.CV_INTER_LINEAR);
 //this.pictureBox1.Image = (Bitmap)image;
//this.pictureBox2.Image = (Bitmap)image5;
this.pictureBox3.Image = (Bitmap)img2scl;
this.pictureBox4.Image = (Bitmap)img5scl;
}
}
private void sendByte(byte[] bDat)
{
timer1.Stop();
serialPort1.Write(bDat, 0, bDat.Length);
timer1.Start();
private void button1_Click(object sender, EventArgs e)
{
int ncams = cvcam.CvcamGetCamerasCount();
```

```
textBox1.Text = ncams.ToString();
// only if we havnt already a valid capture device
if ((capture.ptr == IntPtr.Zero) && (capture2.ptr == IntPtr.Zero))
{
// create (in this case the first device)

capture = highgui.CvCreateCameraCapture(0);
capture2 = highgui.CvCreateCameraCapture(-1);

// check again
if (capture.ptr == IntPtr.Zero)
{
 MessageBox.Show("Could not intialize camera1 capture.");
 return;
}
else if (capture2.ptr == IntPtr.Zero)
{
 MessageBox.Show("Could not intialize camera2 capture.");
 return;
}
else
{
 // start the timer to capture images
timer1.Interval = 200;
timer1.Start();
}
}
}
private void button2_Click(object sender, EventArgs e)
{
```

```
OtherLibs.cvcam.CvcamStop();
OtherLibs.cvcam.CvcamExit();
this.Close();
}
private void Connect_btn_Click(object sender, EventArgs
{
 if (Connect_btn.Text.CompareTo("Enable") == 0)
{
serialPort1.PortName = comboBox1.SelectedItem.ToString();
serialPort1.BaudRate = Convert.ToInt16(comboBox2.SelectedItem);
serialPort1.Parity = System.IO.Ports.Parity.None;
serialPort1.DataBits = 8;
serialPort1.StopBits = System.IO.Ports.StopBits.One;
serialPort1.Open();
if (serialPort1.IsOpen)
{
Connect_btn.Text = "Disable";
}
}
else
{
serialPort1.Close();
Connect_btn.Text = "Enable";
}
}

private void Send_btn_Click(object sender, EventArgs e)
{
if (serialPort1.IsOpen)
{
```

```
Try
{
//serialPort1.WriteLine("At[" + textBox19.Text + "]" + "[" + textBox18.Text + "]" + "[" +
textBox17.Text + "]");
serialPort1.WriteLine("Hb");
timer4.Enabled = true;
timer2.Enabled = true;
Check2 = "1";
}
catch
{
MessageBox.Show("Error");
}
}
private void SandFreq_btn_Click(object sender, EventArgs e)
{
if (serialPort1.IsOpen)
{
try
{
serialPort1.WriteLine("Fa[" + textBox15.Text + "]");
//textBox15.Text = "";
timer3.Start();



}
catch
{
MessageBox.Show("Error");
private void timer2_Tick(object sender, EventArgs e
```

```
{
Draw1("90", "0");
Draw2("90", "0");
Draw3("100", "200")
private void button3_Click(object sender, EventArgs e
{
serialPort1.WriteLine("t");
}
private void Add_Origin_btn_Click(object sender, EventArgs
{
if (serialPort1.IsOpen
{
serialPort1.WriteLine("Ha");
timer3.Start();
Check1 = "0";
OK = 0;
}
private void Come_Origin_btn_Click(object sender, EventArgs e
{
if (serialPort1.IsOpen)
{
serialPort1.WriteLine("At[" + textBox19.Text + "]" + "[" + textBox18.Text + "]" + "[" +
textBox17.Text + "]");
//serialPort1.WriteLine("Hb")
timer3.Start();
}
}
private void timer3_Tick(object sender, EventArgs e
{
if (serialPort1.IsOpen)
```

```
{
string Feed = serialPort1.ReadExisting();
textBox20.Text = Feed;
timer3.Stop();
}
private void timer4_Tick(object sender, EventArgs e)
{
if (serialPort1.IsOpen)
{
string Feed = serialPort1.ReadExisting();
textBox20.Text = Feed;
Check1 = Feed;
if (Check1 == "1"
{
timer4.Enabled = false;
}
}
}
private void checkBox1_CheckedChanged(object sender, EventArgs e){
private void Adj_Cam1_Click(object sender, EventArgs e)
{
trackBar3.Value = 45;
trackBar4.Value = 75;
trackBar2.Value = 110;
}
private void Adj_Cam2_Click(object sender, EventArgs e)
{
trackBar5.Value = 105;
trackBar6.Value = 80;
trackBar7.Value = 85;
```

```
}
private void timer5_Tick_1(object sender, EventArgs e)
{
if (serialPort1.IsOpen)
{
Torque = serialPort1.ReadExisting();

if (Torque == "")
{
textBox20.Text = Torque;
}
Else
{
textBox16.Text = Torque;
}
}
}
}
```