ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ

"ДОНЕЦКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ"

Факультет КНТ

Кафедра ПИ

Лабораторная работа №3

по курсу: «Введение в .Net и Java»

по теме: «Изучение средств организации многопоточности в С#»

Выполнил:

ст. гр. ПИ-19а

Саевский О.В.

Проверил:

Артёменко О.Г.

Ногтев Е.А.

ДОНЕЦК – 2021

Задание

Используя одно и то же изображение, но с разным разрешением (dpi), построить график зависимости времени работы алгоритма от размера изображения.

Async Task (Async-Await)

Код программы:

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Diagnostics;

using System.Drawing;

using System.Drawing.Imaging;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace \_2

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

ManualResetEvent suspend = new ManualResetEvent(false);

List<Image> images = new List<Image>();

private void button1\_Click(object sender, EventArgs e)

{

openFileDialog1.Multiselect = true;

openFileDialog1.Filter = "Image files (\*.jpg, \*.jpeg, \*.png) | \*.jpg; \*.jpeg; \*.png";

var state = openFileDialog1.ShowDialog();

if (state == DialogResult.Cancel)

return;

if (state == DialogResult.OK)

{

for (int i = 0; i < openFileDialog1.FileNames.Length; i++)

{

Image image = Image.FromFile(openFileDialog1.FileNames[i]);

images.Add(image);

ds.Tables[0].Rows.Add(openFileDialog1.FileNames[i], image.Height.ToString(), image.Width.ToString());

}

}

button2.Enabled = false;

}

static int count\_ready = 0;

private void button2\_Click(object sender, EventArgs e)

{

saveFileDialog1.Filter = "\*.jpeg | \*.jpeg";

var fbd = new FolderBrowserDialog();

string name = "processing";

DialogResult result = fbd.ShowDialog();

if (result != DialogResult.OK && string.IsNullOrWhiteSpace(fbd.SelectedPath))

return;

int y = 0;

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

{

string[] FoundFiles;

do

{

y += 1;

FoundFiles = Directory.GetFiles(fbd.SelectedPath, name + y.ToString() + ".jpeg",

SearchOption.TopDirectoryOnly);

} while (FoundFiles.Length != 0);

images[i].Save(fbd.SelectedPath + name + y.ToString() + ".jpeg", ImageFormat.Jpeg);

}

}

public struct YCbCrColor

{

public byte Y { set; get; }

public byte Cb { set; get; }

public byte Cr { set; get; }

public YCbCrColor(byte y, byte cb, byte cr)

: this()

{

Y = y;

Cb = cb;

Cr = cr;

}

public Color ToRgbColor()

{

int r = Convert.ToInt32((double)this.Y +

1.402 \* (double)(this.Cr - 128));

int g = Convert.ToInt32((double)this.Y -

0.34414 \* (double)(this.Cb - 128) -

0.71414 \* (double)(this.Cr - 128));

int b = Convert.ToInt32((double)this.Y +

1.772 \* (double)(this.Cb - 128));

if (r < 0)

r = 0;

if (g < 0)

g = 0;

if (b < 0)

b = 0;

if (r > 255)

r = 255;

if (g > 255)

g = 255;

if (b > 255)

b = 255;

return Color.FromArgb(r, g, b);

}

public static YCbCrColor FromRgbColor(Color color)

{

byte y = Convert.ToByte(0.299 \* (double)color.R +0.587 \* (double)color.G +0.114 \* (double)color.B);

byte cb = Convert.ToByte(128 - 0.168736 \* (double)color.R - 0.331264 \* (double)color.G + 0.5 \* (double)color.B);

byte cr = Convert.ToByte(128 + 0.5 \* (double)color.R - 0.418688 \* (double)color.G - 0.081312 \* (double)color.B);

return new YCbCrColor(y, cb, cr);

}

}

private void button3\_Click(object sender, EventArgs e)

{

button1.Enabled = false;

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

{

Image img1 = images[i];

process(img1, i);

}

suspend.Set();

}

private async void process(Image image, int y)

{

await Task.Run(() =>

{

YCbCrColor ybc;

Bitmap bmp = new Bitmap(image);

float process = 0;

Stopwatch stwatch2 = new Stopwatch();

stwatch2.Start();

for (int i = 0; i < image.Size.Width; i++)

{

process = i / image.Size.Width \* 100;

Invoke((Action)(() =>

{

dataGridView1[3, y].Value = i;//process

}));

for (int j = 0; j < image.Size.Height; j++)

{

Color clr = bmp.GetPixel(i, j); // Получить цвет пикселя в точке (5, 5)

ybc = YCbCrColor.FromRgbColor(clr);

bmp.SetPixel(i, j, ybc.ToRgbColor());

}

}

stwatch2.Stop();

images[y] = bmp;

Invoke((Action)(() =>

{

dataGridView1[4, y].Value = Convert.ToInt32(stwatch2.Elapsed.TotalSeconds);

count\_ready += 1;

if (count\_ready == images.Count())

{

button2.Enabled = true;

button1.Enabled = true;

count\_ready = 0;

}

}));

});

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

DataSet ds = new DataSet();

DataTable dt = new DataTable();

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

{

ds.Tables.Add("Изображения");

ds.Tables[0].Columns.Add("Имя файла изображения");

ds.Tables[0].Columns.Add("Высота");

ds.Tables[0].Columns.Add("Ширина");

ds.Tables[0].Columns.Add("Процесс обработки");

ds.Tables[0].Columns.Add("Время обработки");

dataGridView1.DataSource = ds.Tables[0];

dataGridView1.ReadOnly = true;

}

Thread thread;

Stopwatch swatch = new Stopwatch();

private void RenderChart()

{

Invoke((Action)(() =>

{

openFileDialog1.Filter = "Image files (\*.jpg, \*.jpeg, \*.png) | \*.jpg; \*.jpeg; \*.png";

var state = openFileDialog1.ShowDialog();

if (state == DialogResult.Cancel)

return;

chart1.Series.Clear();

var original = chart1.Series.Add("Время");

var modified = chart1.Series.Add("Разрешение");

}

));

Image img1 = Image.FromFile(openFileDialog1.FileName);

Bitmap bitmap = new Bitmap(img1);

Size size = new Size(bitmap.Width / 4, bitmap.Height / 4);

Bitmap newBitmap = new Bitmap(bitmap, size);

Image img2 = newBitmap;

YCbCrColor ybc;

Image[] image = new Image[2] { img1, img2 };

int y;

for (y = 0; y < 2; y++)

{

swatch.Start();

Bitmap temp = new Bitmap(image[y], image[y].Size);

for (int i = 0; i < temp.Width; i++)

{

for (int j = 0; j < temp.Height; j++)

{

Color clr = temp.GetPixel(i, j); // Получить цвет пикселя в точке (5, 5)

ybc = YCbCrColor.FromRgbColor(clr);

temp.SetPixel(i, j, ybc.ToRgbColor());

}

}

swatch.Stop();

Invoke((Action)(() =>

{

chart1.ChartAreas[0].AxisY.MajorGrid.Enabled = false;

chart1.ChartAreas[0].AxisY.MinorGrid.Enabled = false;

chart1.ChartAreas[0].AxisX.MajorGrid.Enabled = false;

chart1.ChartAreas[0].AxisX.MinorGrid.Enabled = false;

chart1.Series[0].Points.AddXY(image[y].Width.ToString() + "x" + image[y].Height.ToString(), swatch.Elapsed.TotalSeconds);

}));

swatch.Reset();

}

}

private void button4\_Click(object sender, EventArgs e)

{

if (thread == null || thread.IsAlive == false)

{

thread = new Thread(RenderChart);

thread.Start();

}

suspend.Set();

}

bool close = false;

private void Form1\_FormClosing(object sender, FormClosingEventArgs e)

{

if(thread != null)

thread.Abort();

}

}

}

Пример работы программы:



