

EmguCV 图像处理库在 WPF 中的应用

任 超 冯勇建

(厦门大学物理与机电工程学院 福建 厦门 361005)

摘 要 以一款游戏软件的设计为背景,重点介绍基于 C#语言的 EmguCV 图像处理库在 WPF 平台的应用实例。软件需要完成对图像的滤波、特征提取、相似度计算、图形匹配等任务,最终实现对输入图形图像的识别功能。图像处理单元使用 EmguCV 图形图像处理库,它允许人们在 C#语言或者 .NET 框架中使用 OpenCV 图像处理库的功能。

关键词 WPF EmguCV OpenCV 图像处理 边缘匹配

中图分类号 TP311 文献标识码 A DOI: 10.3969/j.issn.1000-386x.2013.05.066

APPLYING EMGUCV IMAGE PROCESSING LIBRARY IN WPF

Ren Chao Feng Yongjian

(School of Physics and Mechanical and Electrical Engineering, Xiamen University, Xiamen 361005, Fujian, China)

Abstract A game software design is taken as the background, in the article we introduce the application example of C# language-based EmguCV image processing library in WPF platform. The software has to complete some tasks, such as filtering, image feature extraction, similarity computation and shape matching, etc. to eventually realise the recognition function on inputted graphics and images. The use of EmguCV graphics and image processing library in image processing unit allows us to use OpenCV image processing library in C# language or the .NET framework.

Keywords Windows Presentation Foundation (WPF) EmguCV OpenCV Image processing Edge matching

0 引言

科技的发展与腾飞需要人们不停地适应快节奏的变化,新的技术与工具不断出现在我们的生活中,在提高工作效率的同时难免带来一些其他的困扰,例如以前用习惯的老技术现在不适用,不兼容了。本文所介绍的内容便是要解决一件“老工具”不兼容的问题:让 OpenCV 这个用于 C/C++ 平台上的图像处理库能够将其强大的功能发挥在基于 C#语言的和 .NET 平台的 WPF 上。

1 WPF 简介

WPF 是由微软推出,被称为下一代显示系统的窗口显示设计技术,现已发展到第四版(WPF Version 4)。由于其拥有强大的界面显示功能,因此能够生成不同以往令人惊讶的 Windows 客户端/网页应用程序。WPF 的图形设计是基于一个矢量的显示引擎核心,这样的话就与显示本身的分辨率无关,能够充分利用现代图形硬件的优势。它有着强大的显示功能和广泛的适用性,可以创建独立的客户端应用程序或者基于浏览器的应用程序。

WPF 大多隶属于 System.Windows 命名空间下。可以选择使用 C# 或 Visual Basic 作为后台逻辑处理语言,进行实例化类、设置属性、调用方法以及处理事件等操作,前台使用 XAML 标记语言来描述控件的位置或大小等外观信息,并使用托管来实

现其相应的行为,这种界面与事件分离的编程方式使得开发者之间的合作更容易形成,而产品也会更加地美观和稳定。

2 EmguCV 在 WPF 中的应用

由以上的介绍可知,WPF 的兴起让 Windows 应用程序的开发变得更加简洁、快速、美观。然而一些基于 C/C++ 语言的工具,如 OpenCV 则不能应用在这个平台上。因此我们需要寻求一个可替代方案,既能享受到 WPF 带给我们高效的界面显示编程,又能根据我们自己的需求,加入强大的图形处理能力。EmguCV 便能满足我们的需求。

2.1 OpenCV 图像处理库简介

在介绍 EmguCV 之前,我们有必要了解 OpenCV 的功能及其特点。OpenCV 是一个开源的计算机视觉处理库,它的设计目标是执行速度尽量快,主要关注实时应用。它采用优化的 C 语言代码编写,能够充分利用 CPU 的优势。OpenCV 旨在构建一个简单易用的计算机视觉框架,以帮助开发人员更加便捷地设计更复杂的计算机视觉相关应用程序。主要优点有以下几个方面:

(1) 高性能 OpenCV 中所有的算法的实现都是基于封装在具有很高灵活性的 IPL 动态数据结构中,并且其中大多数的函数在运算及编译汇编时会被 Intel 公司针对其所生产的处理

收稿日期:2012-05-02。任超,硕士生,主研领域:计算机图形学,视频压缩与显示,WPF 程序设计。冯勇建,教授。

器所优化,得到性能的提升。

(2) 独立性 OpenCV 是一个包括了超过 300 个 C 语言函数的应用程序编程接口,它并不依赖于外部程序库,既可以独立的运行,也能够运行的时候使用其它的外部程序库。

(3) 兼容性 提供了一些与诸如 EiC、Ch、MATLAB 等其它的语言或应用环境的接口,这些接口在其安装完之后位于安装目录 opener/ interfaces 下,方便使用者移植或调用其应用程序。

(4) 开放性 不管是对于商业的还是非商业的用途,OpenCV 的使用都是完全免费的,它的源代码也是完全开放的,应用者可以对源代码进行修改,可以将自己设计的新的函数类型添加到库中,只要设计符合 OpenCV 的规范,自己编写的代码也可以被其他使用者广泛使用。

2.2 什么是 EmguCV

以上对 OpenCV 的介绍是概括性的,EmguCV 同样具有 OpenCV 的这些特点和功能。EmguCV 是 OpenCV 图像处理库一个跨平台的 .NET 封装,允许在 .NET 兼容语言里调用 OpenCV 函数,如 C#,VB 或 IronPython 等。

EmguCV 全部用 C#编写,它可以在 Mono 环境里编译,在任何 Mono 支持的平台(如 Linux, Solaris, Mac OS X)上运行。EmguCV 的其他优势还有,支持通用颜色和深度的图像类,自动垃圾收集,可 XML 序列化的图像,XML 文档和 Intel license 支持,自由选择图像类或 OpenCV 的 direct invoke 函数,图像像素上的通用操作等。也就是说,EmguCV 不仅继承了 OpenCV 的所有功能和特点,而且在跨平台支持方面也有更大的突破。

2.3 配置 EmguCV 并完成图像识别功能

通过以上的介绍我们了解到,在基于 .NET 平台的 WPF 中,我们能够使用 EmguCV 高效的处理图形图像,并且加入了 C#面向对象编程的一些优势,让代码更清晰简洁,效率更高。下面具体介绍在 Windows 系统下,在 VS2010 中配置 EmguCV 2.1 以及使用其完成图像识别任务的应用。

2.3.1 配置 EmguCV

安装下载好的 Emgu.CV.x86.msi 到相应的文件夹,将 emgu 文件夹中所有的动态链接库(.dll 文件)都拷贝到系统路径中,或将当前路径加入到系统环境变量中,之后便可以开始工作了。在 VS2010 中新建 WPF 工程,并添加相应的.dll 文件到工程的引用目录中,这样就能够通过声明库中相应的命名空间来调用其中的方法去处理图像。

2.3.2 图像识别任务简述

图像识别的任务是基于一个小型游戏软件,画像的输入是通过触摸显示屏的人机交互界面来实现的,软件的使用者将在主体页面完成一副彩色动物画像的绘制,通过对动物画像的一些技术处理,使其能够与绘制模版进行比对,来判断所画图像与模版间的相似程度,根据这个相似度来进行画像的识别,最后输出所画的动物图像。

2.3.3 图像识别的实现

画像的输入是通过触摸显示屏的人机交互界面来实现的。参与者可以选择想要绘制的动物,点选进入图形编辑模式,之后,屏幕中出现白色透明画布,下面显示出所选动物的大体轮廓,供参与者画图时参考。虽然参与者画出的图形色彩不一,形状各异,但同种动物的轮廓较为相近,因此,我们可以去掉图像中没用的彩色通道,仅保留图形的灰度值信息,并通过二值化来强调图像的边界信息,通过 canny 算子提取其边界轮廓,再与标

准模板匹配,识别出所画动物的轮廓。

如图 1 所示,图像识别过程可简要概括为如下几个步骤:图形输入,图像预处理(滤波),特征提取,计算相似度,相似度排序,图形识别,画像输出。

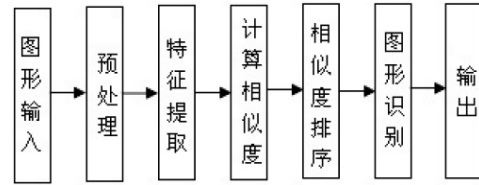


图 1 图像识别方案流程

1) 图形输入 由于采用了触摸屏显示器,图像的输入将以触摸的形式绘制于计算机屏幕上,而 Win7 系统拥有内建的对触摸屏的支持,因此我们选择在 Win7 系统上建立 WPF 程序。WPF 对于触摸事件的处理如同鼠标点击事件一样简单,在程序建立事件委托,便可处理触摸响应。绘图板可以选择 WPF 中的 InkCanvas 控件,它可以通过人机交互来接收和显示墨迹的输入,绘制的笔画被创建为 Stroke 对象,可以对其进行修改,删除等操作。输入的图像通过 RenderTargetBitmap 类对 InkCanvas 中绘制的内容转换为 Bitmap 图形数据,再通过 BitmapEncoder 类将数据编码为 bmp 图形文件,实现图形的计算机输入及保存功能,部分实现代码如下:

```
FileStream fs = new FileStream( path , FileMode. Create );
RenderTargetBitmap bmp = new RenderTargetBitmap ( width , height ,
0 , 0 , PixelFormats. Default );
bmp. Render( this. inkCanvas1 );
BitmapEncoder bmpencoder = new PngBitmapEncoder( );
bmpencoder. Frames. Add( BitmapFrame. Create( bmp ) );
bmpencoder. Save( fs );
```

2) 图像预处理(滤波) 预处理目的是将彩色图像进行灰度化处理,并且消除因为输入条件的限制或输入习惯的差异而导致绘图笔划中存在的毛刺,空隙等噪声,即滤波。在解决方案中加入 EmguCV 的引用: Emgu.CV.dll, Emgu.CV.UI.dll, Emgu.CV.UI.resources.dll, Emgu.Util.dll 等,并在代码中使用命名空间:

```
using Emgu.CV;
using Emgu.CV.UI;
using Emgu.CV.Structure;
using Emgu.Util;
```

通过调用 CvInvoke.cvLoadImage() 方法来读取刚被保存的 bmp 图像文件,读取方式选择为 Emgu.CV.CvEnum.LOAD_IMAGE_TYPE.CV_LOAD_IMAGE_GRAYSCALE,即滤去彩色信息,以灰度值的方式读取图像。之后,对图像进行简单的处理:调用 CvInvoke.cvSmooth() 方法对图像进行模糊处理,参数为 Emgu.CV.CvEnum.SMOOTH_TYPE.CV_GAUSSIAN,即高斯模糊,消除微小毛刺干扰;调用 CvInvoke.cvThreshold() 方法对图像进行二值化运算,仅保留显著画笔信息的区域,然后对图像进行一定形态学运算处理,调用 CvInvoke.cvMorphologyEx() 方法对图像进行开运算处理,先对图像进行腐蚀,消除边界点,再对其膨胀,使图像平滑,从而消除线条上的毛刺,杂质等。经过这样一系列处理,得到的图像基本完成了除噪的任务,可以进行下一步处理。

3) 图形特征提取 通过以上的预处理步骤,已经将图像二值化,并将噪声和干扰剔除,现在的图像仅在白色画布上保留了黑色的具有光滑边界的形状,适合进行边缘提取。计算机进行

边缘提取的方法是计算图像的一阶或二阶导数,寻找灰度值突变的区域。OpenCV(EmguCV)支持多种求导方式,有 Sobel 算子、Laplacian 算子和 Canny 算子等。前两种微分算子实现起来相对简单,适合一般情况下的边缘检测,分别具有主体与背景不能分离和对阶跃背景无法检测等缺点。Canny 算子的实现较为复杂,但其效果最为理想,本文选用 EmguCV 封装好的 `cvCanny()` 方法来对图像进行边缘提取:

```
CvInvoke.cvCanny( img ,img ,100 ,100 ,5);
```

之后对其进行膨胀操作 滤去噪点:

```
CvInvoke.cvDilate( img ,img ,element ,1);
```

4) 计算相似度 相似度是图形匹配的结果表示,代表了图像与既定模板之间的相似关系,OpenCv(EmguCV)为图形轮廓间的匹配提供了几种不同的方法:轮廓树匹配、成对几何直方图匹配和 Hu 矩匹配等,前两种方法顾名思义,是用树的形式和直方图对比来进行轮廓的匹配;而第三种方法是通过计算轮廓的 Hu 矩来计算轮廓间的相似度,Hu 矩对包括缩放、旋转和镜像映射在内的变化具有不变性,我们选用这种方法来计算图形与模板间的相似度。这里调用 `CvInvoke.cvMatchShapes()` 方法对输入图像和模板进行形状匹配,返回值是一个 `double` 类型的数,代表了两幅图的相似程度,返回值越接近 0 则说明两幅图的相似度越高,通过如下计算公式定义相应的相似度百分比,来表征图形与模板的相似度:

```
similaraty = Math. Ceiling( 100 * Math. Pow( 0. 00067 , mat_re-  
sult)); //mat_result 为匹配返回值
```

5) 图形识别 通过所画图形与各个模板的匹配,得到一组相似度的数值,这些数值表示了图形与模板间的相似程度,对这组数值进行简单的排序,理论上将相似度最高的模板定义为最终匹配结果。

2.3.4 匹配实验与结果

图 2 所示为用户通过触摸显示屏绘制的动物“马”的原始图形,将此图形经过以上几个步骤的处理,同下面的模板进行相似度比对,模板如图 3 所示。

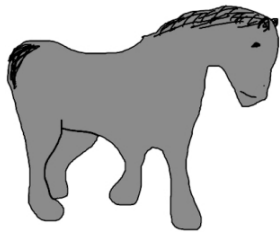


图 2 原始输入图形

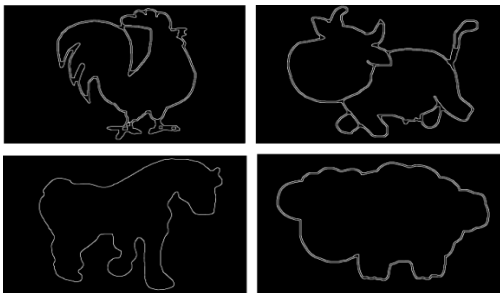


图 3 图形模板

经过比对,我们分别得到了以上图形模板的相似度值,分别为鸡:26%,牛:36%,马:91%,羊:53%,最终“马”匹配成功。

综上所述,实验结果比较理想。在软件实际使用过程中,匹配成功率也比较高,达到了 90% 以上。可见,通过在 VS2010 中

为 WPF 配置加入 EmguCV,将 OpenCV 强大的图形处理功能加入到 .net 平台中,我们并不需要将大量的时间花费在具体算法的实现和调试上,只需注重任务本身的要求,对症下药,设计合理的处理框架,选取合适的方法来调用,开发效率之高可见一斑。

3 结 语

本文简单介绍了新一代客户端设计系统 WPF 的特点与优势,并以一款游戏软件的设计为背景,说明了如何在 WPF 中配置 EmguCV 的过程,详细叙述了运用基于 C# 语言的 EmguCV 图像处理库来完成具体的图像处理任务,主要包括图像的滤波、特征提取、相似度计算等内容,最后通过实验来验证处理的效果。

本文介绍的图形处理方法在 WPF 设计框架下具有广泛的用途,而在普遍使用面向对象语言的今天,基于 C# 的 EmguCV 想必也能够延续 OpenCV 的辉煌。

参 考 文 献

- [1] 黎松,平西建,丁益洪.开放源代码的计算机视觉类库 OpenCV 的应用[J].计算机应用与软件,2005,22(8):134-136.
- [2] 黄建岗.浅谈 WPF 设计模式[J].中小企业管理与科技(上旬刊),2010(10).
- [3] 李学勇,路长厚,李国平.基于二阶梯度图的 Canny 检测边缘修补方法[J].光电子激光,2007,18(3):377-380.
- [4] 秦小文,温志芳,乔维维.基于 OpenCV 的图像处理[J].电子测试,2011(7).
- [5] 方玫.OpenCV 技术在数字图像处理中的应用[J].北京教育学院学报:自然科学版,2011,6(1):7-11.

(上接第 154 页)

- [8] 冯林,王国胤,李天瑞.连续值属性决策表中的知识获取方法[J].电子学报,2009,37(11):2432-2438.
- [9] 冯林,原永乐,苟仕蓉,等.一种实域粗糙集模型及属性约简方法[J].控制与决策,2012,27(4):562-566.
- [10] Yao Y Y. Three-way decisions with probabilistic rough sets[J]. Information Sciences, 2010, 180(3):341-353.
- [11] Yao Y Y, Zhao Y. Attribute reduction in decision-theoretic rough set models[J]. Information Sciences, 2008, 178(1):3356-3373.
- [12] Yao Y Y. Three-way decision: an interpretation of rules in rough set theory[C]//The 4th International Conference on Rough Sets and Knowledge Technology, 2009.
- [13] 曾黄麟.粗糙集理论及其应用[M].重庆:重庆大学出版社,1996.
- [14] Frank A, Asuncion A. UCI Machine Learning Repository[EB/OL]. Irvine, CA: University of California, School of Information and Computer Science. (2010). <http://archive.ics.uci.edu/ml>.
- [15] Øhrn A, Komorowski J, Rosetta: A rough Set Toolkit for Analysis of Data[C]//Third International Joint Conference on Information Sciences, Fifth International Workshop on Rough Sets and Soft Computing, Durham, NC, USA, RSSC'97(3):403-407.
- [16] Nir Friedman, Dan Geiger. Bayesian Network Classifiers [J]. Machine Learning, 1997, 29:131-163.
- [17] Chang Chih-Chung, Lin Chih-Jen. LIBSVM: a library for support vector machines [J]. ACM Transactions on Intelligent Systems and Technology, 2011, 2(3):1-27.
- [18] Nello Cristianini, John Shawe-Taylor. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods [M]. England: Cambridge University Press, 2000.