中国电子教育学会高教分会推荐·现代通信技术系列教材
高等学校新工科应用型人才培养“十三五”规划教材

数字图像处理算法典型实例与工程案例

苏军　任小玲　胡文学
陈　宁　师红宇　编著

西安电子科技大学出版社
http://www.xduph.com

中国电子教育学会高教分会推荐·现代通信技术系列教材

高等学校新工科应用型人才培养“十三五”规划教材

数字图像处理算法典型实例与工程案例

苏　军　任小玲　胡文学

陈　宁　师红宇　　编著

西安电子科技大学出版社

内 容 简 介

本书是数字图像处理实验教材，内容包含图像增强、图像还原和图像分割等图像处理技术的基本原理、典型算法以及实验程序代码。除了基础知识外，书中还加入了实际的工程案例章节，提供了相关的原理分析、算法描述和程序代码实现。

本书共 9 章，分别是调试软件的使用、图像增强、图像还原、图像分割、图像压缩、图像采集、静态视频监视下运动目标的检测、手写数字图像的识别和织物疵点检测。每章都包含相关知识介绍、典型实验及程序编码，以及对程序编码的必要说明，使读者能够掌握基于 OpenCV 的图像处理编程技术和方法，这也是本书的特色。

本书可作为高等学校计算机、通信和自动化等相关专业本科生、研究生的教材，或者作为工作在图像处理、识别领域一线的广大技术人员的参考资料，也可作为数字图像处理课程设计的素材。

图书在版编目(CIP)数据

数字图像处理算法典型实例与工程案例 / 苏军等编著. —西安：西安电子科技大学出版社，2019.12

ISBN 978-7-5606-5505-5

Ⅰ. ① 数… Ⅱ. ① 苏… Ⅲ. ① 数字图像处理—教材 Ⅳ. ① TN911.73

中国版本图书馆 CIP 数据核字(2019)第 226579 号

策划编辑 戚文艳
责任编辑 姚智颖 雷鸿俊
出版发行 西安电子科技大学出版社(西安市太白南路 2 号)
电 话 (029)88242885 88201467 邮 编 710071
网 址 www.xduph.com 电子邮箱 xdupfxb001@163.com
经 销 新华书店
印刷单位 陕西天意印务有限责任公司
版 次 2019 年 12 月第 1 版 2019 年 12 月第 1 次印刷
开 本 787 毫米×1092 毫米 1/16 印 张 8.5
字 数 196 千字
印 数 1～3000 册
定 价 22.00 元
ISBN 978-7-5606-5505-5 / TN
XDUP 5807001-1

前　言

为了全面深入地掌握数字图像处理学科的相关知识，上机实验是数字图像处理课程中非常重要的实践环节。编者在教学中发现大部分数字图像处理教材采用 MATLAB 作为教学语言，利用 MATLAB 中的相关函数实现对数字图像的处理。

随着 OpenCV 发行，一个开源、跨平台的计算机视觉库为我们提供了图像处理和计算机视觉方面的算法。本书作为数字图像处理课程的实验教材，采用了 Visual Studio 2010 与 OpenCV2.4.9 共同搭建的实验环境，给出了图像处理相关算法描述，并提供了对应的程序代码。

本书共 9 章，第 1~5 章介绍典型数字图像处理算法，第 6~9 章提供了一些工程案例。书中各章包括实验目的、相关基础知识、实验内容、实验报告要求等内容，工程案例部分还附有相关函数和程序阅读。实验目的部分概括了每章实验需要重点掌握或一般了解的知识点；相关基础知识简要地介绍了数字图像处理的相关理论知识；实验内容给出实验需要完成的内容及程序代码；实验报告要求给出实验报告需要包含的内容；相关函数和程序阅读对工程案例实验中用到的关键程序代码进行了详细的注释和解读，供学生阅读以加深对实验内容的理解。

本书是一本数字图像处理课程的实验教材，基本覆盖了数字图像处理的主要内容。希望读者能够体会到实验目的、相关基础知识、实验内容所蕴含的数字图像处理概念，学会一定的 OpenCV 编程技术。书中所有程序代码都通过上机测试，需要的读者可发邮件到 junsus@163.com 免费索取。

本书第 1、2 章由胡文学编写，第 3、5 章由师红宇编写，第 4、8 章由任小玲编写，第 6、7 章由苏军编写，第 9 章由陈宁编写。苏军设计了全书的结

构，并做了统稿工作。本书在编写和出版过程中得到了西安工程大学领导、西安电子科技大学出版社戚文艳编辑的大力支持和帮助，在此表示感谢。

由于编者水平有限，书中难免有疏漏和不足之处，恳请广大读者和同行专家批评指正。

编　者

2019 年 7 月

目　录

第 1 章　调试软件的使用

1.1　实 验 目 的

- 熟悉 Visual Studio 2010(可简写为 VS2010)软件的集成开发环境，并掌握其编译、调试、运行等操作。
- 掌握 OpenCV 的环境配置及程序开发步骤。
- 能编写简单的图像处理程序，能验证 Visual Studio 2010 下 OpenCV 环境配置的正确性。
- 掌握程序在编译、链接、运行中常见错误的处理方法。

1.2　关于 OpenCV

OpenCV(Open Source Computer Vision，开源计算机视觉库)最初由 Intel 开发，是一个免费的跨平台实时图像处理库。OpenCV 可以进行计算机视觉有关的事务处理，它已经成为一个标准库工具，为图像处理、模式识别等提供多种标准库函数。读者如果需要 OpenCV 库，可登录 http://opencv.org 免费下载，该网站提供最新发布的版本以及各种老版本。本书采用的版本是 OpenCV 2.4.9。

OpenCV 为模块化结构，代码包中包含了每个模块的一个静态库或动态库(DLL)，代码包中的主模块有：

- core 模块：提供 OpenCV 的一些基本数据结构(包括密集的多维数组 Mat)和所有其他模块使用的基本函数。
- imgproc 模块：提供一些图像处理函数，包括线性和非线性图像滤波函数、几何图像转换(调整大小、仿射和透视变形、基于通用表的重映射)函数、颜色空间转换函数、直方图函数等。
- video 模块：提供视频分析功能，包括运动估计、目标跟踪和前景提取。
- features2d 模块：用于特征检测，包含特征检测、特征描述和特征匹配等函数。
- calib3d 模块：包含相机标定、双视角几何估计以及立体函数。
- objdetect 模块：用于目标检测，汇集了目标检测函数，如面部和人体探测器等。

另外，OpenCV 库还包含了一些其他的实用模块，如机器学习函数(ml 模块)、计算几何算法(flann 模块)、共享代码(contrib 模块)、过时的代码(legacy 模块)以及 GPU 加速代码(gpu 模块)等。

1.3　OpenCV 在 VS2010 中的环境配置及程序开发步骤

首先，安装 Visual Studio 2010 开发工具和 OpenCV 库(这里的版本是 2.4.9)，将 OpenCV 库解压安装在 F 盘根目录下。下面说明如何在 Visual Studio 2010 上配置 OpenCV 库。

1. 新建项目

(1) 启动 Visual Studio 2010，进入开发窗口，如图 1-1 所示。

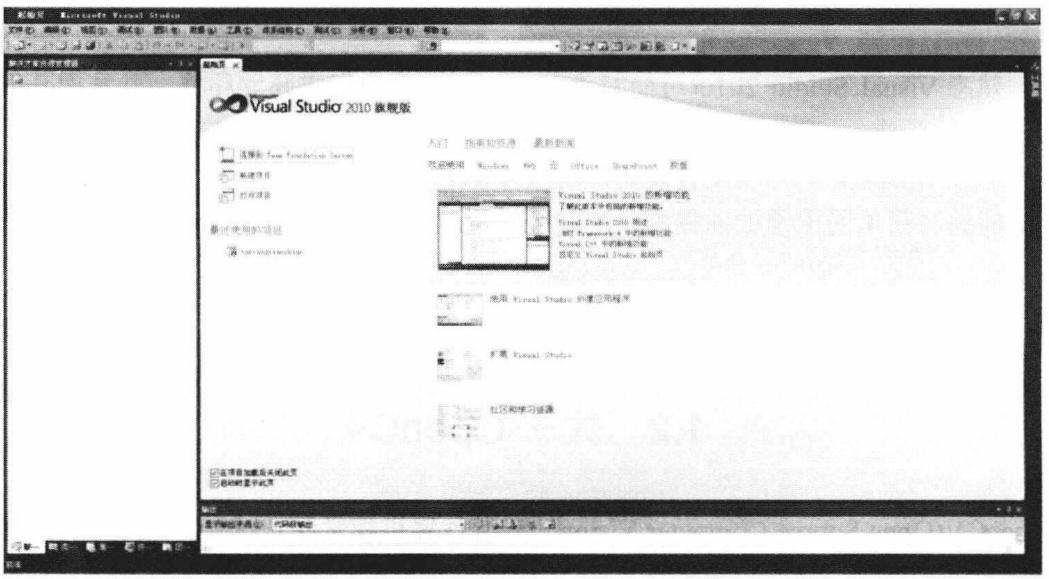

图 1-1　Visual Studio 2010 开发窗口

(2) 选择“文件→新建→项目”，弹出对话框如图 1-2 所示。在“已安装的模板”中选择“Visual C++”，在项目文件类型选择框中选中“Win32 控制台应用程序”，输入项目名称“showing”，选择保存项目文件的位置为 F:\范例\范例 1-1，单击“确定”按钮。

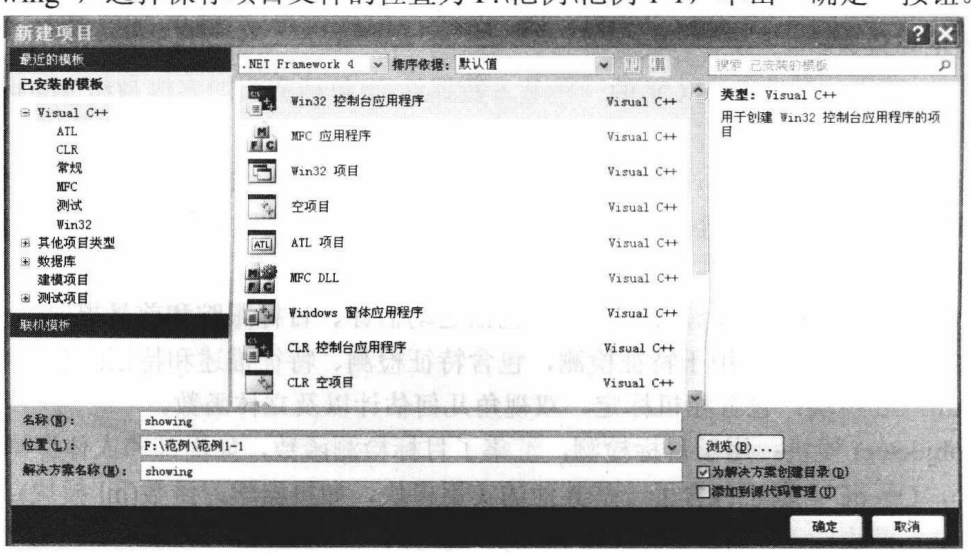

图 1-2　新建项目

(3) 弹出对话框如图 1-3 所示，单击“下一步”按钮。

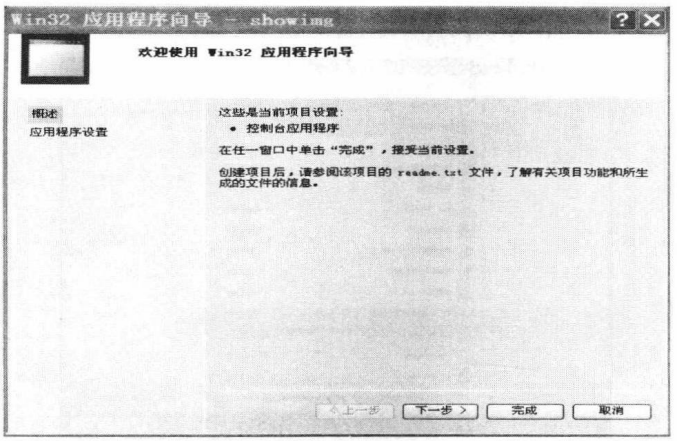

图 1-3　Win32 控制台应用程序向导

(4) 弹出对话框如图 1-4 所示，在“应用程序类型”栏目中选择“控制台应用程序”单选按钮，“附加选项”栏目中选中“空项目”复选框，单击“完成”按钮，完成对项目文件的创建。

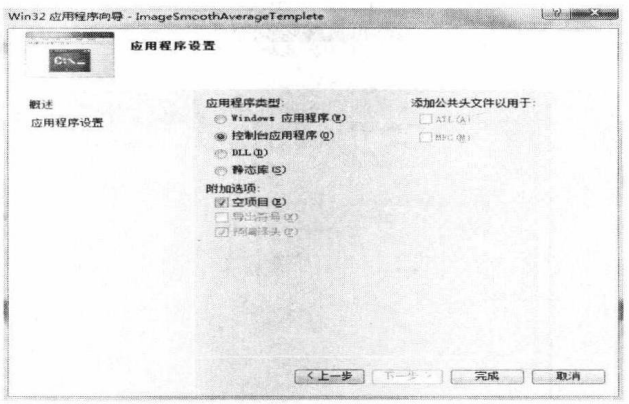

图 1-4　创建控制台应用程序

(5) 项目文件创建完成后，接着需要创建项目文件的源文件。在图 1-5 所示的界面中，选中“源文件”列表，单击鼠标右键，在弹出的快捷菜单中选择“添加→新建项”添加源文件。

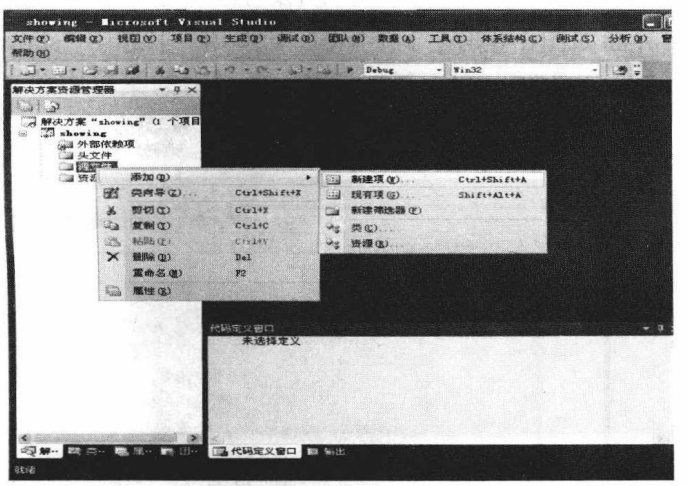

图 1-5　添加源文件

(6) 弹出对话框如图 1-6 所示，选择源文件的类型为“C++ 文件(.cpp)”选项，输入源文件名称“showimg.cpp”，完成源文件的创建。

图 1-6　创建 showimg.cpp 源文件

2. 完成 OpenCV 的配置

(1) 如图 1-7 所示为项目开发窗口。在完成新建项目后，需要配置 OpenCV 库(包含目录和库目录)，在项目开发窗口中单击“视图→属性管理器”，弹出对话框如图 1-8 所示。

图 1-7　项目开发窗口

(2) 属性管理器界面如图 1-8 所示。在“属性管理器”中选中“showimg→Debug|Win32→Microsoft.Cpp.Win32.user”项，单击鼠标右键，在弹出的菜单中单击“属性”。

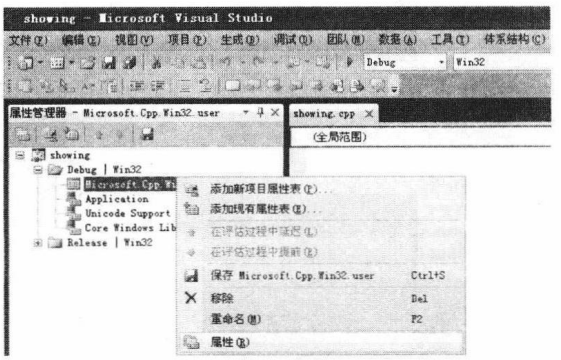

图 1-8　属性管理器界面

(3) 弹出属性配置界面如图 1-9 所示，选择“通用属性→VC++ 目录→包含目录”，单击右边的下拉按钮，选中“编辑”项。

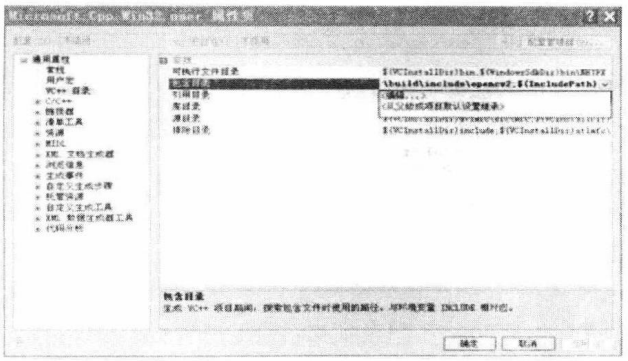

图 1-9　属性配置界面

(4) 弹出的对话框如图 1-10 所示，在文本框中输入以下内容：

F:\opencv\opencv\build\include

F:\opencv\opencv\build\include\opencv

F:\opencv\opencv\build\include\opencv2

注意，以上路径均为 OpenCV 的安装路径。单击“确定”按钮，至此，包含目录的配置就完成了。

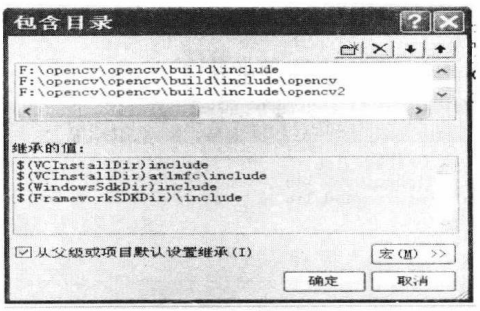

图 1-10　配置包含目录的对话框

(5) 在如图 1-9 所示的界面中单击“通用属性→VC++目录→库目录”，单击右边的下拉按钮，选择“编辑”，弹出对话框如图 1-11 所示，在文本框中输入以下内容：

F:\opencv\opencv\build\x86\vc10\lib

注意，以上路径为 OpenCV 的安装路径。单击“确定”按钮，至此，库目录的配置就完成了。

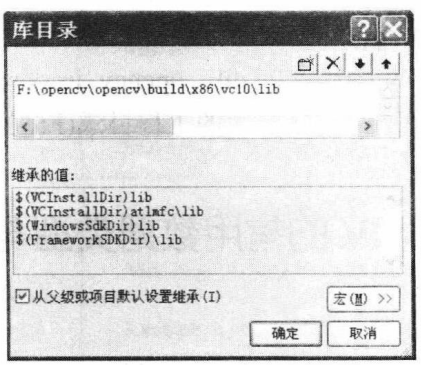

图 1-11　配置库目录的对话框

(6) 完成附加依赖项的配置。在如图 1-12 所示的界面中单击“通用属性→链接器→输入→附加依赖项”，单击右边的下拉按钮，选择“编辑”。

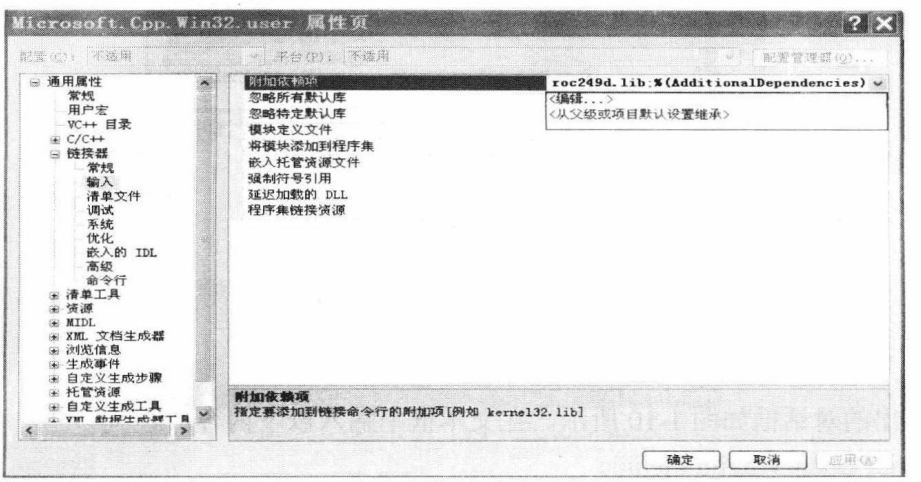

图 1-12　打开“附加依赖项”对话框

(7) 弹出对话框如图 1-13 所示，在对话框的文本框中依次添加：

opencv_core249d.lib

opencv_highgui249d.lib

opencv_imgproc249d.lib

单击“确定”按钮，至此，附加依赖项的配置就完成了。

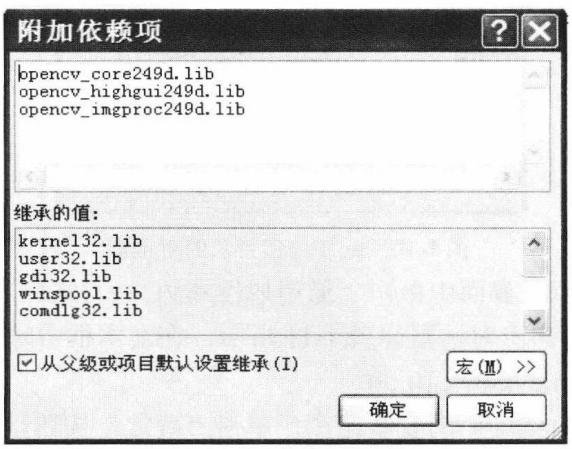

图 1-13　配置附加依赖项

(8) 上述步骤已完成对 OpenCV 的配置。要保证用户程序能正常运行，还需将 opencv_core249d.dll、opencv_highgui249d.dll、opencv_imgproc249d.dll 这三个文件从 F:\opencv\build\x86\vc10\bin 路径下拷贝到对应的工程目录(F:\范例\范例 1-1)下。

1.4　OpenCV 的常用数据类型与常用操作

1. 常用数据类型

OpenCV 的常用数据类型可参见表 1-1。

表 1-1　OpenCV 的常用数据类型

类　型	类型关键字	示　例
(Small) Vec (向量类)	VecAB(A 可以是 2、3、4、5 或 6；B 可以是 b、s、i、f 或 d)	Vec3b rgb; rgb[0]=255;
(UP to 4) scalars (表示颜色，如 RGB 的颜色值)	Scalar	Scalar (a, b, c); //RGB颜色值中的红色分量为 a，绿色分量为 b，蓝色分量为 c

Point
(表示点)
	PointAB(A 可以是 2 或 3；B 可以是 i、f 或 d)	Point3d p;
		p.x=0;
		p.y=0;
		p.z=0;

Size
(表示尺寸)
	Size	Size s;
		s.width=30;
		s.height=40;

Rectangle
(表示图像的部分区域)
	Rect	Rect r;
		r.x=r.y=0;
		r.weight=r.height=100;

2. 常用操作

OpenCV 的常用操作可参见表 1-2。

表 1-2　OpenCV 的常用操作

操　作	代 码 示 例
设置矩阵的值	img.setTo(0);　　// 1 个通道的图像 img.setTo(Scalar(B, G, R));　// 3 个通道的图像
Mat 矩阵初始化	Mat m1 = Mat::eye(100, 100, CV_64F); Mat m2 = Mat::zeros(100, 100, CV_8UC1); Mat m3 = Mat::ones(100, 100, CV_8UC1)*255;
随机初始化	Mat m1 = Mat(100, 100, CV_8UC1); randu(m1, 0, 255);
创建矩阵的一个副本	Mat img1 = img.clone();
创建一个(具有掩码)矩阵的副本	img.copy(img1, mask);
引用一个子矩阵(不复制数据)	Mat img1 = img(Range(r1, r2), Range(c1, c2));
图像裁剪	Rectroi(r1, c2, width, height); Mat img1 = img(roi).clone();　//数据拷贝

续表

操　作	代 码 示 例
调整图像大小	resize(img, img1, Size(), 0.5, 0.5); //将图像变为原来的 1/2

翻转图像	flip(imgsrc, imgdst, code);
	//code=0 => 垂直翻转
	//code>0 => 水平翻转
	//code<0 => 垂直和水平翻转

分割通道	Mat chananel[3]; split(img, channel); imshow("B", channel[0]);　//显示蓝色
合并通道	merge(channel, img);
计算非零像素数	intnz = countNonZero(img);
最大值和最小值	double m, M; point mLoc, MLoc; minMaxLoc(img, &m, &M, &mLoc, &MLoc);

像素值均值	Scalar m, stdd;
	meanStdDev(img, m, stdd);
	unit mean_pxl = mean.val[0];

检查图像数据是否为空	if (img.empty()) cout<< "couldn´t load image" <<endl;

1.5　OpenCV 的常用函数

1. 窗口操作常用函数

OpenCV 的窗口操作常用函数可参见图 1-14。

namedWindow 函数

功能：namedWindow 函数用于创建一个窗口。

函数原型：void namedWindow(const string& winname, int flags=WINDOW_AUTOSIZE);

参数 1：const string&型的 winname 为需要显示的窗口标识名称；

参数 2：int 类型的 flags，为窗口的标识，可以填 WINDOW_NORMAL、WINDOW_AUTOSIZE 或 WINDOW_OPENGL。如果设置为 WINDOW_NORMAL，则用户可以改变窗口的大小(没有限制)。如果设置为 WINDOW_AUTOSIZE，则窗口大小会自动调整以适应所显示的图像，并且不能手动改变窗口大小。如果设置为 WINDOW_OPENGL，则窗口创建的时候便会支持 OpenGL。

imread 函数

功能：imread 函数用于读入图像。

函数原型：Mat imread(const string& filename, int flags=IMREAD_COLOR);

参数 1：const string&类型的 filename 为需要载入图片的路径名；

参数 2：int 类型的 flags 为载入标识，指定一个加载图像的颜色类型(默认 1 表示载入三通道的彩色图像)。其中，IMREAD_COLOR=0 表示 8 位灰度；IMREAD_COLOR>0 表示一个三通道的彩色图像。

例如：Mat image1 = imread("1. jpg", 0);　　//载入灰度图

Mat image2 = imread("1. jpg", 199);　　//载入三通道的彩色图像

imshow 函数

功能：imshow 函数在指定的窗口中显示一幅图像。

函数原型：void imshow(const string& winname, InputArray mat);

参数 1：const string& winname 为需要显示的窗口标识名称；

参数 2：InputArray 类型的 mat 为需要显示的图像。

imshow 函数用于在指定的窗口显示图像，如果窗口是用 CV_WINDOW_AUTSIZE(默认值)标志创建的，那么显示图像原始大小，否则将对图像进行缩放使之适应窗口大小。imshow 函数缩放图像的情况取决于图像的深度。

图 1-14　OpenCV 的窗口操作函数

2. 平滑滤波常用函数

OpenCV 的平滑滤波常用函数可参见图 1-15。

boxFilter 函数

函数原型：void boxFilter(InputArray src, OutputArray dst, int ddepth, Size ksize, Point anchor , bool normalize, int borderType)

参数 1：InputArray src 为输入图像 src；

参数 2：OutputArray dst 为输出图像 dst；

参数 3：int ddepth 为输出图像深度，−1 表示与输入图像使用相同的深度；

参数 4：Size ksize 定义内核(滤波器)的大小，一般用 Size(w, h)来表示内核(滤波器)的大小(其中，w 为像素宽度，h 为像素高度)。例如，Size(3, 3)就表示内核 3 × 3 的大小，Size(5, 5)就表示内核 5 × 5 的大小；

参数 5：Point anchor 表示被平滑的点 anchor(anchor pixel)定位像素的位置，其默认值(−1，−1)意味着定位像素是内核的中心；

参数 6：bool normalize 表示内核是否被归一化，默认值为 true；

参数 7：intborderType 表示边界类型。

GaussianBlur 函数

函数原型：void GaussianBlur(InputArray src, OutputArray dst, Size ksize, double sigmaX, double sigmaY = 0, int borderType = BORDER_DEFAULT)

参数 1：InputArray src 为输入图像 src；

参数 2：OutputArray dst 为输出图像 dst；