◎产品、研发、测试◎

跨平台 BMP 图像处理程序的实现

潘永之[1],李玉和[1],刘东岳[2]
PAN Yong-zhi[1],LI Yu-he[1],LIU Dong-yue[2]

1.清华大学 精密仪器与机械学系,北京 100084
2.北京市 1303 信箱 14 分箱,北京 100073
1.Department of Precision Instruments and Mechanology,Tsinghua University,Beijing 100084,China
2.No.14,Mailbox 1303,Beijing 100073,China
E-mail:panyongzhi@gmail.com

PAN Yong-zhi,LI Yu-he,LIU Dong-yue.Implementation of cross-platform BMP image processing program.Computer Engineering and Applications,2007,43(14):84-86.

Abstract: In this article the concept and characteristics of cross-platform are introduced.Based on the analysis of the structure of BMP images,several key points of cross-platform image processing programming are brought forward.Following these points,a number of cross-platform BMP image processing programs are written in the C++ language,and are built successfully under several platforms such as Linux/x86,FreeBSD,SunOS/sparc and Windows/x86-64.The experimental results show that cross-platform programming can boost the universality and compatibility of programs,broadening the way of programming techniques.
Key words: cross-platform;BMP;image processing;EasyBMP;portable

摘 要:介绍了跨平台的概念和特点,以及跨平台编程的思想,在对 BMP 图像结构分析的基础上,提出了编写跨平台图像处理程序的几个关键问题。并遵循这些要点,利用 C++语言编写了若干跨平台 BMP 图像处理的程序,这些程序在 Linux/x86,FreeBSD,SunOS/sparc,Windows/x86-64 等多个平台下成功编译和运行。实验结果表明,跨平台编程可以大大提高程序的通用性和兼容性,为编程技术拓宽了思路。
关键词:跨平台;BMP;图像处理;EasyBMP;可移植
文章编号:1002-8331(2007)14-0084-03 **文献标识码**:A **中图分类号**:TP391

1 引言

BMP(Bitmap)图像是科学研究中常用的图像格式,但是目前所见对 BMP 图像的处理程序大多依赖于特定的平台,使得程序的跨平台性很差。

本文介绍了跨平台概念及其在程序开发中的重要性,并给出了跨平台 BMP 图像处理程序的实现方法,以及若干用 C++编写的跨平台 BMP 图像处理程序,这些程序无需任何修改就可以在不同的硬件平台和不同的操作系统下编译和运行。然而跨平台编程并不局限于图像处理,它可以应用在各种程序的开发上。

2 BMP 图像格式

BMP 图像是 Windows 操作系统保存图像的一种通用位图文件格式,在其他平台下也很常见。BMP 图像内部结构可分为 4 部分:位图文件头(bmfh)、位图信息头(bmih)、调色板[1](aColors)和位图数据(aBitmapBits)[2],如图 1 所示。

位图文件头是识别信息,典型的应用程序会首先读取这部分数据以确保是位图文件并且没有损坏;位图信息头告诉应用程序图像的详细信息,在屏幕上显示图像将会使用这些信息;调色板定义了图像中使用的颜色;位图数据则逐个像素地描述了图像。

BITMAPFILEHEADER bmfh
BITMAPINFOHEADER bmfh
RGBQUAD aColors[ ]
BYTE aBitmapBits[ ]

图 1 BMP 图像存储结构

3 跨平台程序开发

跨平台(Cross-platform)指编程语言、应用软件或硬件设备,可以在多种计算平台下工作[3]。跨平台的应用程序可以在多种平台下不加改动或者稍加改动地编译(对某些语言不必编译)并且运行。这里的计算平台包括硬件平台和软件平台,硬件平台指中央处理器(CPU)架构,例如 x86 和 x86-64 就是当今个人电脑最常见的 CPU 架构。软件平台指操作系统或者编程环境,或者两者的结合,典型的软件平台有 Windows,Linux/

基金项目:国家重点基础研究发展规划(973)(the National Grand Fundamental Research 973 Program of China under Grant No.2003CB716201)。
作者简介:潘永之(1984-),男,硕士研究生,研究方向为精密技术;李玉和(1972-),男,博士、副教授,研究方向为精密测控、仪器科学与技术;刘东岳(1967-),男,高级工程师,研究方向为水面舰艇。

x86,MacOS X/PowerPC,Java 等。

图 2 显示了跨平台开发的优点。只需写出同样的代码,在不同系统下编译,便可在各个系统下得到同样的图像处理效果。如果程序没有跨平台性,要在不同平台下实现同样功能,就需要在不同平台下分别开发,这将大大增加开发周期和成本。

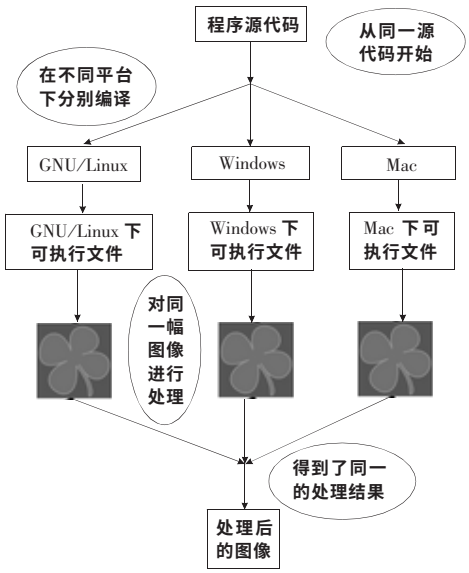

图 2 跨平台程序开发示意图

如果开发者使用了某系统下特有的功能,例如在汇编程序中使用了某 CPU 特有的指令,或者在高级语言中使用了某操作系统特有的应用程序接口(Application Program Interface),那么程序就不能跨平台,只能在特定的 CPU 或者特定的操作系统上才能运行。要使这个程序运行在别的 CPU 或者操作系统上,需要修改此程序的相应部分才可以。如果程序对特定系统依赖很大,跨平台实现几乎是不可能的。

4 跨平台 BMP 处理程序的编写

4.1 关键问题

根据跨平台开发的要求,编写跨平台 BMP 图像程序前需要注意以下几方面的关键问题。

(1)首先要使用跨平台的编程语言或环境。本文采用 C++来编写 BMP 图像处理程序。C++语言有很好的跨平台性,而且其面向对象编程(Object-Oriented Programming)特性非常适合处理图像这种具有复杂结构的数据。如果使用 Visual Basic 或者.Net 等开发语言或者环境,那么程序就通常只能在 Windows 系统下使用。

(2)不要使用特定系统的 API。虽然 C++是跨平台的,但 Windows 下用到 Win32API 的 C++程序,一般就不能在 Unix 机器上编译。应当使用标准的和跨平台的 C++库函数。BMP 图像处理有很多现成的库,但是很多库都无法满足跨平台的要求,有的 BMP 图像处理程序对最简单的功能都使用特定系统的库函数,这样的程序是无法跨平台的。这里使用了 EasyBMP 库[4],这是 Paul Macklin 用 C++写的一个优秀的跨平台 BMP 图像库,只使用了 C++的标准库函数,没有用到任何特定系统的API。

(3)要使用规范的编译器。不同编译器对语言规范的解释也有所差异,使用一个重视标准的编译器就无需修改程序使其在不同平台下通过编译。本例中使用了在多个平台下都可用的、对 C++标准支持较好的 GCC 编译器。

(4)在编写好程序后,要在多种不同类别的硬件平台(32 位、64 位、little endian、big endian 架构等)和软件平台(Windows,GNU/Linux,Unix 等)下测试,确保程序在每个平台下都能编译通过,并能得到正确的图像处理结果。

4.2 图像处理的 C++程序

程序要实现的功能是在一幅彩色图像中的目标形心上叠加十字线,这分别要对图像进行中值滤波、灰度化、二值化、形心计算和十字线叠加等操作。

由于 EasyBMP 中利用了 C++的高级数据封装特性,使得用户编写的程序简洁易懂。EasyBMP 定义了 BMP 类,可以用其声明一个 BMP 图像对象。表 1 是 EasyBMP 一些常用函数的原型,用户根据函数原型调用即可,无需了解函数的实现细节。

表 1 EasyBMP 常用函数原型

实现功能	函数原型
读取 BMP 图像文件	bool BMP::ReadFromFile(const char* FileName)
写入 BMP 图像文件	bool BMP::WriteToFile(const char* FileName)
获取 BMP 图像水平像素数	int BMP::TellWidth(void)
获取 BMP 图像竖直像素数	int BMP::TellHeight(void)
获取(i,j)处的像素	RGBApixel* BMP::operator()(int i,int j)
获取 BMP 图像色深	int BMP::TellBitDepth(void)

编写用户程序时,只需在程序头部包含 EasyBMP.h,在编译时和 EasyBMP.cpp 一起编译,如下所示:

```
gcc binalize.cpp EasyBMP.cpp-o binalize
```

下面用一个完整的 C++程序展示了用 EasyBMP 进行跨平台 BMP 图像处理的方法。C++程序 binalize.cpp 对灰度 BMP 图像根据命令行的阈值进行黑白二值化,如果命令行不输入阈值,那么取默认阈值 128。完整的 C++程序源代码如下:

```
/* binalize.cpp
* 对 BMP 图像进行二值化处理
*/
#include "EasyBMP.h"
using namespace std;
int main(int argc,char* argv[]){
 if(argc ! =3 && argc ! =4){
   cout << "Usage:binalize <input_filename> "
     "<output_filename>[threshold]" << endl;
   return 1;
 }
 ebmpBYTE threshold;
 if(argc==3)
   threshold=128;// 默认阈值
 else
   threshold=(ebmpBYTE)atoi(argv[3]);
 // 声明并读取位图
 BMP Input;
 Input.ReadFromFile(argv[1]);
 // 对灰度图像二值化(不检验输入图像是否为灰度图像)
 for(int j=0;j<Input.TellHeight();j++)
   for(int i=0;i<Input.TellWidth();i++)
     Input(i,j)->Red=Input(i,j)->Green=Input(i,j)->Blue
       =(Input(i,j)->Red > threshold ? 255 :0);
 Input.WriteToFile(argv[2]);
 return 0;
}
```

中值滤波[5]、灰度化、形心计算[6]和十字线叠加等程序也用

同样的方法编写和编译。

4.3 不同平台下的实验结果

笔者在不同平台下对编写的跨平台十字线叠加程序进行了测试，它们都能够不加任何修改地在4个不同CPU和4个不同操作系统下编译和运行，如表2所示。

表2 测试过的不同平台

处理器	操作系统	编译器	运行结果
AMD Athlon	Linux 2.6.17-2-k7	GCC 4.1.2	全部通过
Intel Xeon	FreeBSD 6.0	GCC 3.4.4	全部通过
SUNW Ultra-250	SunOS 5.8	GCC 3.2	全部通过
Intel EM64T	Windows XP x64	MinGW/GCC 3.4.2	全部通过

另外，在Windows下，程序在Cygwin和Visual C++ 6.0下也都编译通过，并且得到正确的图像处理结果。

图3是原始彩色图像，图4是中值滤波后图像，图5是灰度化和二值化后的图像，图6是形心计算、并在目标中心叠加十字线的图像。

图3 原始图像

图4 中值滤波后的图像

图5 灰度化和二值化后的图像

图6 处理后叠加十字线的图像

在不同平台下的可执行程序都得到了完全相同的图像处理结果，满足了跨平台的要求。

5 结论

本文在跨平台编程思想的基础上，对跨平台的BMP图像处理方法进行了实践。在多个不同的CPU和操作系统下对笔者编写的BMP图像处理程序进行了实验，得到了很好的跨平台结果。对比分析显示，跨平台编程方法可以大大提高图像处理程序的通用性和兼容性，为图像处理编程技术拓宽了思路。（收稿日期：2007年1月）

参考文献：

[1] 李莹，张嵩，章坚武.嵌入式Linux操作系统上的BMP文件的处理[J].杭州电子工业学院学报，2004，24(4)：35-38.

[2] The.bmp file format[EB/OL].http://www.fortunecity.com/skyscraper/windows/364/bmpffrmt.html.

[3] Cross-platform[EB/OL].http://en.wikipedia.org/wiki/Cross-platform.

[4] EasyBMP Cross-Platform Windows BMP Libray[EB/OL].http://easybmp.sourceforge.net.

[5] 谷口庆治.数字图像处理[M].朱虹，译.北京：科学出版社，2002：80-85.

[6] 闫伟，金元郁.基于Visual C++的运动目标形心捕获[J].微计算机信息，2005，21(2)：180-182.

（上接59页）

[12] Zhang J，Modestino W，Langan D A.Maximum-likelihood parameter estimation for unsupervised model-based image segmentation[J].IEEE Trans Image Processing，1994，3：404-420.

[13] Marroquin J L，Vemuri B C，Botello S，et al.An accurate and efficient Bayesian method for automatic segmentation of brain MRI[J].IEEE Trans Med Imag，2002，21：934-945.

[14] 詹劲峰，戚飞虎，王海龙.基于时空马尔可夫随机场的运动目标分割技术[J].通信学报，2000，7(5)：434-439.

[15] Marroquin J L，Botello S，Calderon F，et al.MPM-MAP for image segmentation[C]//Proc ICPR 2000，Barcelona，Spain，2000，4：300-310.

[16] Zhang Yong-yue，Brady M，Smith S.Segmentation of brain MR images through a hidden Markov random field model and the expection-maximization algorithm[J].IEEE Transactions on Medical Imaging，2001，20(1)，45-57.

（上接72页）

4 结论

利用遗传算法解决约束优化问题时，出现的问题是在进行交叉、变异操作时，会出现不可行解。可以按照处理不可行解的方法对约束优化问题处理方法进行分类。完全拒绝不可行解的方法就是拒绝策略，惩罚策略对不可行解的适应度值进行处理。本文的方法是引入可行种群和不可行种群两个种群进行混合交叉、变异操作来扩大搜索范围，增加种群的多样性，产生新的解；然后两个种群分别进行选择操作以提高种群的平均适应度值。通过几个常用的测试问题的求解以及与其它几种约束处理方法的比较表明：本文提出的约束处理方法具有很好的性能，且处理方法简单。

同时，必须指出，本研究在理论分析和实践应用上都尚不充分，进一步的研究问题还包括：所提方法的有效性分析；各种约束处理方法的比较研究，特别是各种处理方法的效率的比较，遗传算法中种群大小、交叉概率、变异概率、迭代次数等参数的选取对算法的性能的影响的研究。同时，任何算法都有其适用范围，因此进一步研究本文算法对哪些问题效果好也十分重要和必要。（收稿日期：2006年9月）

参考文献：

[1] 林丹，李敏强，寇纪淞.基于遗传算法求解约束优化问题的一种算法[J].软件学报，2001，12(4)：628-632.

[2] Goldberg D E.Genetic algorithms in search，optimization & machine learning[M].[S.l.]：Addison-Wesley publishing company，1989.

[3] Michalewicz Z，Dasgupta D，Le Riche R G，et al.Evolutionary algorithms for constrained engineering problems[J].Computers & Industrial Engineering Journal，1996，30(2)：851-870.

[4] 玄光男，程润伟.遗传算法与工程设计[M].北京：科学出版社，2000.

[5] Michalewicz Z.Evolutionary algorithms for constrained optimization[C]//IWEC'2000 International Workshop on Evolutionary Computation，Wuhan，2000：1-11.

[6] Tang Jia-fu，Wang Ding-wei，Ip A，et al.A hybrid genetic algorithm for a type of non-linear programming problems[J].Computers and Mathematics with Applications，1998，36(5)：11-21.

[7] Himmelblau M.Applied nonlinear programming[M].New York：McGraw-Hill，1972.

[8] Homaifar A，Qi C，Lai S.Constrained optimization via genetic algorithms[J].Simulation，1994，62(4)：242-254.

[9] 李敏强，寇纪淞，林丹，等.遗传算法的基本理论与应用[M].北京：科学出版社，2002.