实用技术

图形图象文件格式解码实用程序(续 7)

张明敏

7 24 位 PCX 文件

PC Paintbrush 文件格式是为 Zsoft 公司的商品化着色程序——PC Paintbrush 软件包而开发的。PC Paintbrush 文件格式是一种支持彩色的格式,在 PC Paintbrush 的早期版本中,图形/图象文件最多只有 256 种彩色。经过不断改进,Zsoft 公司的 PC Paintbrush 已经成为 PC 机上受到用户普遍欢迎的绘图软件。随着 Windows 软件的用户大大增加,Zsoft 公司推出 Windows 版的 PC Paintbrush 软件,使得用户在 Windows 3.0/3.1 环境下可运行 PC Paintbrush 软件。除了用户界面有了较大改进外,Windows 版的 PC Paintbrush 软件支持 24 位 PCX 文件(或 BMP 文件),用户可以在 16 色的显示器上编辑 24 位图形文件。

Windows 版的 PC Paintbrush 支持多种图象文件格式,除了 PCX 外,还有 DIB、BMP 和 MSP 等。当然 PCX 仍然是一种重要的文件格式。由于广大用户使用 PC Paintbrush 绘图软件,使得 PCX 几乎成为 PC 机上应用程序的一种标准图象文件格式,该文件被很多台式出版软件作为输入文件而使用,也可用作几种绘图软件的模板。另外很多其它应用程序也能产生 PCX 文件输出。

这里主要讨论 PC Paintbrush 中使用的新 24 位真彩色 PCX 文件标准,并对显示 PCX 文件的技术进行详细介绍,同时给出 C 语言源代码。

PCX 标准已经存在很长一段时间,但它一直在发展。最近又推出 24 位 PCX 文件标准,它对原有的 8 位 PCX 文件有很大的改进。对图象数据进行压缩处理,而不管是否能减少存储图象数据的空间。因为行程编码方法并不是总能够减小 24 位复杂图象的大小,所以如果读者发现对 24 位 PCX 文件进行解码得到的结果文件比原来的文件还要小,那么也是正常的事。事实上,用于保存 PCX 的行程编码过程更适合于压缩绘制的图形,而对扫描得到的图象就不是很合适。它比较擅于处理大块连续不变的彩色,而不是颜色混杂的像素区域。

因为 PC Paintbrush 是一种绘图程序,所以它的文件格式适合于保存绘制的图形,而不适合于保存扫描的图象。因而 PC Paintbrush 只适合于作为一绘图软件包,而不适合作为一扫描仪驱动程序。使用 PCX 压缩相对于其它行程压缩设计方法(如 Targa 行程编码)的一个显著优点是:PCX 编码方法保存的图象的解码速度快。24 位 PCX 版本有自己的特色,在下面讨论它们的组织方式以及读写方法。

7.1 PCX 文件结构

如前面所述,PCX 文件由 128 个字节的头和很多压缩图象数据组成,在图象数据的最后是 768 字节长的调色板和一个用于定位的标志。因为调色板对 24 位的 PCX 文件已不起作用,所以下面介绍的代码中将不对调色板进行处理。

不管 PCX 文件支持多少彩色位,它的头标总是相同的,如下面的 C 语言结构定义:

```
typedef struct {
        char manufacturer;
        char version;
        char encoding;
        char bits_per_pixel;
        int xmin,ymin;
        int xmax,ymax;
        int hres;
        int vres;
```

* 注:需源程序者请与浙江大学 CAD&CG 国家重点实验室 张明敏联系
邮编:310027,电话:(0571)7951045,7990451

```
char palette[48]
char reserved;
char color_planes;
int bytes_per_line;
int palette_type;
char filler[58];
} PCXHEAD;
```

上面这个 PCXHEAD 结构还是比较容易理解的，每个域的含义可以从它们的域名中推断出。

(1) manufacturer 域的值总是 10，PCX 文件解码程序就是通过这个域来识别 PCX 文件。

(2) version 域的值根据它自身的文件结构以及产生它的软件的年代不同而有不同的值。假定讨论的 PCX 文件是由 PC Paintbrush 生成的，那么 version 就可取以下值：

0——PC Paintbrush 版本 2.5

2——PC Paintbrush 版本 2.8，包括了调色板

3——PC Paintbrush 版本 2.8，使用默认调色板

5——PC Paintbrush 版本 3.0 以上(包括 3.0 版)

这里讨论的文件的 version 值为 5。

(3) encoding 域总为 1，表明使用行程编码。在现有情况下，行程编码是 PCX 支持的唯一编码类型。

(4) bit_per_pixel 域表明所处理的图象中每个像素所用的彩色位数。注意对 24 位 PCX 文件而言，该值取 8。

(5) xmin 和 ymin 域指图象的左上角相对于用户屏幕的左上角的坐标位置，通常情况下这两个值为 0。

(6) xmax 和 ymax 域定义左上角与右下角之间的距离。因此，如果 PCX 是一个类型为 PCXHEAD 的结构，并且用一个 PCX 文件头标加载它，那么就可以用下面的语句来计算 PCX 图象的实际尺寸：

```
int width,depth;
width=PCX·xmax_PCX·xmin+1;
depth=PCX·ymax_PCX·ymin+1;
```

(7) hres 和 yres 域分别用于指定水平和垂直分辨率，这些值通常用不到。

(8) palette 域保存图象文件的彩色调色板(如果有的情况下)。这里也假定图象支持的彩色不超过 16 种。如果彩色多于 16 种，那么在 256 种彩色情况下，调色板信息保存在图象数据的后面。

(9) color_planes 域指定图象中的彩色平面数，在 16 位图象中，该值为 4，在 24 位图象中，该值为 3。从上面的赋值情况可以看出 PCX 在处理 24 位图象数据时有独到之处。对 3 字节的 RGB 像素进行行程编码是一个值得研究的问题，完成这一功能可以有几种方法，每种方法都有自己的优缺点。在压缩 24 位 Targa 文件的数据时，使用了 3 字节行程编码，这是一种显而易见的方法。但是在对压缩图象行进行解码时速度比较慢，这是因为程序代码要对 3 的整数倍字节进行操作。我们知道计算机比较擅长处理 2 的整数次幂的数。把某一个值乘以 3 需要真正的乘法操作，相比之下，把某一个值乘以 2 则只需要进行简单的移位操作，后者速度明显快于前者。PCX 文件把单个的 RGB 行作为三个位平面数据进行压缩，第一个位平面由所有红色像素组成；第二个位平面由所有绿色像素组成；第三个位平面则由所有蓝色像素组成。在这种方法中，每个图象行只需对不超过 1 个字节长的对象进行操作。因而，可以快速对 24 位 PCX 文件进行解码。

(10) bytes_per_line 域代表图象中一个平面上的字节值。单个 RGB 平面中的字节数值可能与一行上像素数目不一样，这完全依赖于产生相应图象的应用程序。在某些情形下，要对字节数进行特殊处理；以保证字节数是偶数值(把一哑字节附加到图象行的末尾)。因此，在解码 PCX 文件时，应该使用 types_per_line 域中的值，而不应该根据图象宽度得出每行的字节数。

(11) palette_type 域仅适用于 256 色的 PCX 文件。它表明用彩色还是用灰度来显示当前的图象：如果用彩色显示，那么该值为 2；如果用灰度显示，那么该值为 1。

在 PCX 文件头标后的下一个字节是第一个压缩红色平面中的第一个字节。在 PCX 中使用的行程编码压缩方法很容易理解。为了解码一行，必须读取一字节并检查该字节中的高两位是否为 1，如果为 1，那么把高位和 '00' 相“与”(AND)，并把剩下的值称为是长度说明符(rs)，读取下一个字节并重复写该字节 rs 次。如果两个高位未置为 1，那么就认为该字节是未编码字节，直接完成写动作。在读入要解码行的一个域后，再重复这一过程，直到一完整图象行被读入为止。一完整行将是一未压缩的字节串，字节串的长度等于 pcx.types_per_line 的值。

下面是读入 PCX 文件中一行的函数：

```
readpcxline (p,fp,bytes)
```

```
        char *p;
        FILE *fp;
        int bytes;
    {int n=0,c,i;
        do {
        c=fgetc(fp)&0xff;
          if((C&0xc0)==0xc0) {
            i=c&0x3f;
            c=fgetc(fp);
            while(i--)p[n++]=c;
          }
          else p[n++]=c;
        } while(n<bytes);
        return(n);
    }
```

很明显,PCX 压缩的一个特殊优点就是它的解码方法简单。

值得注意的是,所有 PCX 文件使用相同类型的行程压缩方法,而不依赖于它们有多少种彩色位。

7.2 解码 PCX 文件的程序

程序 readpcx.c 仅处理 24 位的 PCX 文件,并不能处理彩色位数少于 24 位的 PCX 文件。事实上处理彩色数少的 PCX 文件要相对困难一些,特别是 16 色的 PCX 文件。

```
/*************************
        24 位 PCX 解码程序
*************************/
main(argc,argv)
  int argc;
  char *argv[];
{FILE *fp;
  static char results[5][16] = {  "Ok","Bad file",
    "Bad read","Memory error","Too few colours"};
  char path[80];
  int r;
  if(argc > 1) {
    strmfe(path,argv[1],"PCX");
    strupr(path);
    if((fp = fopen(path,"rb")) != NULL) {
      fi.setup=dosetup;
      fi.closedown=doclosedown;
      r=unpackpcx(fp,&fi);
      printf("\%s",results[r]);
      fclose(fp);
    } else printf("Error opening %s",path);
  } else puts("Argument:     path to a PCX file");
}
/* unpack an PCX file */
unpackpcx(fp,fi)
  FILE *fp;
  FILEINFO *fi;
{PCXHEAD pcx;
  char *p,*pr;
  int i,j,bytes;
  for(i=0;i<256;++i)
    memset(fi->palette+(i*RGB_SIZE),
    i,RGB_SIZE);
  if(fread((char *)&pcx,1,sizeof(PCXHEAD),fp) ==
  sizeof(PCXHEAD) && pcx.manufacturer==10) {
    if(pcx.bits_per_pixel==8 &&
    pcx.colour_planes==3) {
      bytes=pcx.bytes_per_line;
      fi->width=pcx.xmax-pcx.xmin+1;
      fi->depth=pcx.ymax-pcx.ymin+1;
      fi->bytes=bytes*RGB_SIZE;
      if((p=malloc(fi->bytes)) == NULL)
      return(MEMORY_ERROR);
      if((pr=malloc(fi->width)) == NULL) {
        free(p);
        return(MEMORY_ERROR);
      }
      if((fi->setup)(fi) != GOOD_READ) {
        free(pr);

        free(p);
        return(MEMORY_ERROR);
      }
      for(i=0;i<fi->depth;++i) {
        for(j=0;j<RGB_SIZE;++j) {
          if(readpcxline(p+(j*bytes),fp,bytes) !=
          bytes) {
            freebuffer();
            free(pr);
            free(p);
            return(BAD_READ);
          }
        }
      for(j=0;j<fi->width;++j)
        pr[j]=greyvalue(p[j],
              p[RGB_GREEN*bytes+j],
              p[RGB_BLUE*bytes+j]);
        putline(pr,i);
      }
      (fi->closedown)(fi);
```

```
        freebuffer();
        free(pr);
        free(p);
    } else return(WRONG_BITS);
    } else return(BAD_READ);
    return(GOOD_READ);
}
/* read and decode a PCX line into p */
readpcxline(p,fp,bytes)
    char *p;
    FILE *fp;
    int bytes;
{int n=0,c,i;
    do {
    c=fgetc(fp) & 0xff;
      if((c & 0xc0) == 0xc0) {
          i=c & 0x3f;
          c=fgetc(fp);
          while(i--) p[n++]=c;
      }
      else p[n++]=c;
    } while(n < bytes);
    return(n);
}
/* this function is called after the BMHD and CMAP
chunks have been read but before the BODY is unpacked
*/
dosetup(fi)
    FILEINFO *fi;
{union REGS r;
    if(! getbuffer((long)fi->width * (long)fi->depth,
    fi->width,fi->depth))
      return(MEMORY_ERROR);
    r.x.ax=0x0013;
    int86(0x10,&r,&r);
    setvgapalette(fi->palette,256,0);
    return(GOOD_READ);
}
/* This function a called after an image has been un-
packed. It must display the image and deallocate memory.
*/
doclosedown(fi)
    FILEINFO *fi;
{union REGS r;
    int c,i,n,x=0,y=0;
    if(fi->width > SCREENWIDE) n=SCREENWIDE;
    else n=fi->width;
    do {for(i=0;i<SCREENDEEP;++i) {
      c=y+i;
      if(c>=fi->depth) break;
      memcpy(MK_FP(0xa000,SCREENWIDE*i),
      getline(c)+x,n);
    }
    c=GetKey();
    switch(c) {
        case CURSOR_LEFT:
            if((x-STEP) > 0) x-=STEP;
            else x=0;
            break;
        case CURSOR_RIGHT:
            if((x+STEP+SCREENWIDE) <
            fi->width) x+=STEP;
            else if(fi->width > SCREENWIDE)
              x=fi->width-SCREENWIDE;
            else x=0;
            break;
        case CURSOR_UP:
            if((y-STEP) > 0) y-=STEP;
            else y=0;
            break;
        case CURSOR_DOWN:
            if((y+STEP+SCREENDEEP) <
            fi->depth) y+=STEP;
            else if(fi->depth > SCREENDEEP)
              y=fi->depth-SCREENDEEP;
            else y=0;
            break;
        case HOME:
            x=y=0;
            break;
        case END:
            if(fi->width > SCREENWIDE)
              x=fi->width-SCREENWIDE;
            else x=0;
            if(fi->depth > SCREENDEEP)
              y=fi->depth-SCREENDEEP;
            else y=0;
            break;
        }
    } while(c ! = 27);
    freebuffer();
    r.x.ax=0x0003;
    int86(0x10,&r,&r);
    return(GOOD_READ);
}
/* set the VGA palette and background */
```

```
setvgapalette(p,n,b)
  char *p;
  int n,b;
  {union REGS r;
  int i;
  outp(0x3c6,0xff);
  for(i=0;i<n;++i) {
    outp(0x3c8,i);
    outp(0x3c9,(*p++) >> 2);
    outp(0x3c9,(*p++) >> 2);
    outp(0x3c9,(*p++) >> 2);
  }
  r.x.ax=0x1001;
  r.h.bh=b;
  int86(0x10,&r,&r);
}
```

所有针对 PCX 特定格式的程序代码都在 unpackpcx 函数中，该函数完成绝大部分解码工作。它首先读取文件的最前面 128 个字节并把它们传送给 PCXHEAD 结构，然后抽取必要的信息。随后的代码用于读取图象平面，把一个 PCX 24 位行的 3 个平面转换为 8 位灰度行的过程需要较多的代码控制。虽然这里给出的代码易于理解，但是速度比较慢。

上面给出的代码非常简单，原因在于它只处理一种类型的 PCX 文件，即 24 位 PCX 文件(24 位 PCX 文件只有一种标准结构，没有变种)。因此，在上面的程序中没有 case 语句和复杂的 if else 树型结构。如果要使上面的程序支持所有类型的 PCX 文件，那么就需要使用很多 C 语言中的分支语句。

7.3 使用 ATI VGA 显示 24 位 PCX 文件

ATI VGA 卡是一种引人注目的 VGA 卡，它是提供真彩色模式的廉价显示设备之一。READPCX2 中没有什么前面程序中未曾出现过的内容，PCX 文件象在前面讨论的一样被处理，但它们不是以位片行转换成灰度级调色板行，而是转换成每像素 3 字节的 24 位彩

```
/* * * * * * * * * * * * * * * * * * * * * * * *
   用 ATI VGA 卡显示 24 位 PCX 文件的程序
* * * * * * * * * * * * * * * * * * * * * * * * */
main(argc,argv)
  int argc;
  char *argv[];
{FILE *fp;
  static char results[6][16] = { "Ok","Bad file",
    "Bad read","Memory error","Too few colours",
    "No driver mode"};
  char path[80];
  int r;
  if(argc > 2) {
    strmfe(path,argv[2],"DRV");
    if((gd=loadDriver(path))== NULL) {
      printf("Error loading driver %s\n",path);
      exit(1);
    }
    strmfe(path,argv[1],"PCX");
    strupr(path);
    if((fp = fopen(path,"rb")) ! = NULL) {
      fi.setup=dosetup;
      fi.closedown=doclosedown;
      r=unpackpcx(fp,&fi);
      printf("\%s",results[r]);
      fclose(fp);
    } else printf("Error opening %s",path);
  } else puts("Argument:    path to a PCX file");
}
/* unpack an PCX file */unpackpcx(fp,fi)
  FILE *fp;
  FILEINFO *fi;
{PCXHEAD pcx;
  char *p,*pr;
  int i,j,bytes;
  if(fread((char *)&pcx,1,sizeof(PCXHEAD),fp) ==
  sizeof(PCXHEAD) && pcx.manufacturer==10) {
    if(pcx.bits_per_pixel==8 &&
    pcx.colour_planes==3) {
      bytes=pcx.bytes_per_line;
      fi->width=pcx.xmax-pcx.xmin+1;
      fi->depth=pcx.ymax-pcx.ymin+1;
      fi->bytes=bytes*RGB_SIZE;
      if((p=dosalloc(fi->bytes)) == NULL)
      return(MEMORY_ERROR);
      if((pr=dosalloc(fi->bytes)) == NULL) {
        dosfree(p);
        return(MEMORY_ERROR);
      }
      if((fi->setup)(fi) ! = GOOD_READ) {
        dosfree(pr);
        dosfree(p);
        return(MEMORY_ERROR);
      }
      for(i=0;i<fi->depth;++i) {
```

```
        for(j=0;j<RGB_SIZE;++j) {
          if(readpcxline(p+(j * bytes),fp,bytes)! =
          bytes) {
            freebuffer();
            dosfree(pr);
            dosfree(p);
            return(BAD_READ);
            }
          }
        for(j=0;j<fi->width;++j) {
          pr[j * RGB_SIZE+RGB_RED]=p[j];
            pr[j * RGB_SIZE + RGB_GREEN] = p
            [RGB_GREEN * bytes+j];
          pr[j * RGB_SIZE+RGB_BLUE]=
          p[RGB_BLUE * bytes+j];
        }
        putline(pr,i);
      }
      (fi->closedown)(fi);
      freebuffer();
      dosfree(pr);
      dosfree(p);
    }else return(WRONG_BITS);
  }else return(BAD_READ);
  return(GOOD_READ);
}
/* read and decode a PCX line into p */
readpcxline(p,fp,bytes)
  char *p;
  FILE *fp;
  int bytes;
{
  int n=0,c,i;
  do {c=fgetc(fp) & 0xff;
    if((c & 0xc0) == 0xc0) {
      i=c & 0x3f;
      c=fgetc(fp);
      while(i--) p[n++]=c;
    }
    else p[n++]=c;
  } while(n < bytes);
  return(n);
}
GRAFDRIVER *loadDriver(s)
/* load an external graphics driver */
  char *s;
{GRAFDRIVER *gd;
  char *grafDriver=NULL;
  FILE *fp;
  long l;
  char b[8];
  unsigned int seg;
  if((fp=fopen(s,"rb")) ! = NULL) {
    fseek(fp,0L,SEEK_END);
    l=ftell(fp);
    rewind(fp);
    fread(b,1,8,fp);
    if(! memcmp(b,"ALCHDRV2",8)) {
      l--=8L;
      if(l < 0xffffL &&
       (grafDriver = dosalloc((unsigned int)l)) ! =
       NULL){
        if(fread(grafDriver,1,(unsigned int)l,fp)==
          (unsigned int)l) {
            gd=(GRAFDRIVER *)grafDriver;
            seg=FP_SEG(grafDriver);
            if(gd->vga_on ! = NULL) {
                grafDriver[2]=seg;
                grafDriver[3]=(seg >> 8);
                grafDriver[6]=seg;
                grafDriver[7]=(seg >> 8);
                grafDriver[10]=seg;
                grafDriver[11]=(seg >> 8);
                grafDriver[14]=seg;
                grafDriver[15]=(seg >> 8);
                grafDriver[18]=seg;
              grafDriver[19]=(seg >> 8);
            }
            if(gd->ega_on ! = NULL) {
                grafDriver[22]=seg;
                grafDriver[23]=(seg >> 8);
                grafDriver[26]=seg;
                grafDriver[27]=(seg >> 8);
                grafDriver[30]=seg;
                grafDriver[31]=(seg >> 8);
                grafDriver[34]=seg;
                grafDriver[35]=(seg >> 8);
            }
            if(gd->mono_on ! = NULL) {
                grafDriver[38]=seg;
                grafDriver[39]=(seg >> 8);
                grafDriver[42]=seg;
                grafDriver[43]=(seg >> 8);
                grafDriver[46]=seg;
                grafDriver[47]=(seg >> 8);
                grafDriver[50]=seg;
```

```
            grafDriver[51]=(seg >> 8);
        }
        if(gd->rgb_on != NULL) {
            grafDriver[54]=seg;
            grafDriver[55]=(seg >> 8);
            grafDriver[58]=seg;
            grafDriver[59]=(seg >> 8);
            grafDriver[62]=seg;
            grafDriver[63]=(seg >> 8);
        }
    }
                else {
                    dosfree(grafDriver);
                    grafDriver=NULL;
                }
            }
        }
        fclose(fp);
    }
    return((GRAFDRIVER *)grafDriver);
}
```

华能集团引进 PictureTel 会议电视设备

PictureTel 通过其代理商——恒德资讯科技有限公司,与中国华能集团签约,为华能集团综合卫星通信网项目提供22台集团会议电视系统 Concorde 4500。这些高性能、高质量的会议电视系统将安装于华能集团在各地的分公司和集团下属的各电厂,用来召开跨地域的集团工作会议,及时解决工作中的各种问题。

PictureTel 的集团会议系统 Concorde 4500 不仅可以提供 H.320 国际标准会议电视功能,同时还可以提供 PictureTel 专利算法 SG3、SG4,在不占用更多带宽的情况下,实现更清晰自然的图象,而 PictureTel PT724 宽带音频算法和 Virtuoso 音频系统,更创造出杰出的音频效果。无线键盘和自动语音定位技术,使系统操作更加方便自如。同时 Concorde 4500 支持 T.120 数据共享,使用户在感受身临其境的电视会议的同时,还可以进行文件、图表的传输和应用的共享。该系统充分利用各种先进的技术手段,模拟真实会议的进行,实现跨地域的协同工作。因此,华能集团在经过对会议电视各厂商产品综合比较后,选中 PictureTel 的 Concorde 4500 为其卫星网会议电视终端。

华能综合卫星通信网的实施,使华能集团办公自动化的建设又上了一个新的台阶,必将更好地提高集团的内部沟通,整体工作效率,使集团能够更好地适应瞬息万变的市场。 (汪虹)

* *