人机交互信息汉化的高效方法

石油大学(华东)(257062)　李村合　姚　军

《中文信息》1995年第4期登载了王向阳、杨红颖的《人机交互信息的汉化方法》一文，介绍了对西文软件中的人机交互信息进行汉化的三种方法。利用文中介绍的方法虽可完成汉化，但在实际使用时存在以下不足：

(1) 效率低。文中介绍的方法都需要将西文软件调入内存，查找出人机交互信息后将其汉化，再用覆盖的方法代替原来的信息，需要用户自己找到原西文串的结束符，并数出其长度，既不方便，又浪费时间；

(2) 可靠性差。因需要直接对原西文软件进行修改，若不小心修改了人机交互信息以外的内容(例如译文超长等)，会引起副作用，使汉化后的软件无法运行；

(3) 汉化可能不彻底。由于各种原因，有的人机交互信息可能未找到，从而不能对其进行汉化。

针对这些不足，本文介绍一种对西文软件的人机交互信息进行汉化的通用方法，可以高效地、可靠地、彻底地完成汉化工作。

1. 实现思想

本文介绍的汉化方法的基本思想是：

(1) 用软件的方法将西文软件中所有的人机交互信息及其开始位置、长度找出来，并保存到一个文本文件中；

(2) 用WPS、CCED等文字编辑软件修改存放人机交互信息的文本文件，将所有的信息汉化，即用相应的汉字串代替原西文串；

(3) 根据记录下来的所有人机交互信息的开始位置，用软件的方法将已汉化的信息从文本文件中写回到西文软件的原来位置，并进行译文长度检查。

2. 程序说明

笔者用C语言编写了两个程序，借助它们可以顺利完成汉化。现对其说明如下：

2.1 GETINF 程序

功能：获取西文软件中的可显示字符串及其开始位置、长度，并存入文本文件。

用法：GETINF file text minlen

说明：GETINF有三个命令行参数(各参数之间用空格隔开)。其中file是欲汉化的西文软件文件名，text是欲存放人机交互信息的文本文件名，minlen是欲输出的可显示字符串的最小长度。该程序将file中长度不小于minlen的可显示字符串及其开始位置、长度输出到文本文件text中，得到一个可显示字符串清单。text文件中每行信息的格式为：

开始位置:长度:字符串

另外，某些特殊字符，虽然不可显示，但也是人机交互信息中经常出现的字符，如空字符(0)、响铃(7)、退格(8)、水平制表(9)、垂直制表(11)、走纸(12)、回车(13)、换行(10)等。我们把这些特殊字符当作可显示字符对待，程序在输出时进行了转换，转换规则类似于C语言的输出格式转义序列，如下表所示：

意义	符号	ASCII码	转义序列
空字符	NUL	0	\0
响铃	BEL	7	\a
退格	BS	8	\b
水平制表	HT	9	\t
换行	LF(NL)	10	\n
垂直制表	VT	11	\v
走纸	FF	12	\f
回车	CR	13	\r
反斜线	\	92	\\

2.2 PUTINF 程序

功能:根据通过 GETINF 程序获取的、已修改和汉化的人机交互信息文件 text 的内容,将已汉化的所有人机交互信息写回到西文软件的原来位置,并进行译文长度检查,输出超长译文清单。

用法:PUTINF file text

说明:PUTINF 有两个命令行参数(各参数之间用空格隔开)。其中 file 是欲汉化的西文软件名(同 GETINF 中的 file),text 是存放提示信息的文本文件名。text 是对 GETINF 中输出的 text 进行修改和翻译后得到的,已删去了其中不明其义和不需翻译的行。得到人机交互信息清单后,每行数据的"位置"和"长度"不变,"字符串(人机交互信息)"已由英文翻译成中文。该程序按行读取 text 文件,计算每行译文的实际长度,并与原长度进行比较,若译文超长,则将本行输出到屏幕,否则根据本行人机交互信息的位置将汉化信息写回西文软件 file 中,直到所有的行写完,从而完成对西文软件的汉化。为了便于检查超长译文的实际长度并进行压缩,输出到屏幕的超长译文的格式与 text 中略有不同,如下:

开始位置:汉化前长度:汉化后长度:字符串

3. 使用方法及注意问题

3.1 使用方法

首先对文章中所附的 C 语言源程序在 Turbo C 2.0 或 Borland C++ 3.1 环境下进行编译,生成可执行文件 GETINF.EXE 和 PUTINF.EXE,然后按以下步骤对某个西文软件 file 进行汉化:

(1) C>GETINF file text minlen

(2) 用 WPS、CCED 等文字编辑软件修改 text,去掉不明其义及不需翻译的行,得到"人机交互信息清单",并将它们从西文翻译成中文;

(3) C>PUTINF file text

(4) 若屏幕显示出有超长译文,则转(2),根据信息未汉化前的原长度和汉化后的新长度继续修改 text,压缩超长译文;否则汉化完毕。

3.2 注意问题

(1) GETINF 生成的文本文件 text 是一个"可显示字符串清单(包括特殊字符)",其中可能有一些无意义的字符串,因此还不能称作是"人机交互信息清单"。使用此命令时要适当地调整参数 minlen,其值设置小了会出现大量的无意义字符串,设置大了又会丢掉部分较短的人机交互信息。根据笔者的使用经验,minlen 一般取 4~12 比较适宜;

(2) 编辑 text 时,一定不要修改每行的"位置"和"长度"值,并注意"长度"和"字符串"之间有一空格。由于 text 中可能有无意义的字符串,因此应先删除其中不明其义和不需翻译的行,得到一个"人机交互信息清单",然后再进行翻译修改。

(3) 空字符(NUL)和字符' $'通常用作字符串的结束标志,在将西文翻译成中文时应注意对它们的处理。一般说来它可随译文的长度改变,总放在字符串的末尾。对回车、换行符需灵活处理,可随译文的长度而改变,放在字符串末尾;也可放在原来位置不变。

(4) 若人机交互信息根据某种算法进行了变换,则可参照王向阳介绍的"数据解释汉化法"进行处理。在步骤(2)中,先将密文还原成明文,汉化后再将明文转换成密文。

本文介绍的汉化方法不直接触及可执行代码,只需编辑修改 GETINF 输出的文本文件 text 即可实现汉化,且能自动检查超长译文,具有使用方便、简单可靠、效率高、汉化彻底等特点。该方法对用户熟悉一个新软件也是非常有用的,可以利用 GETINF 得到新软件的所有人机交互信息并进行研究、熟悉。文中所附程序在 386、486 微机上用 Turbo C 2.0 或 Borland C++ 3.1 均调试通过,作者已用此方法对多个西文软件的人机交互信息进行了汉化,效果令人满意。

4. 源程序清单及汉化实例

4.1 GETINF.C 源程序清单

```
#include〈fcnt1.h〉
#include〈io.h〉
#include〈stdio.h〉
#include〈stdlib.h〉
#include〈string.h〉
#define MAXLEN 60 /*交互信息的最大长度*/
void main(int ac,char *av[])
{int fb,len,minlen,i,strend;
  FILE *ft;     /*交互信息文件指针*/
  long pos;     /*当前字符位置*/
char ch,str[100];
if (ac〈4)printf("格式:GETINF file text minlen\n");
else if (fb=open(av[1],o.RDONLY|0.BINARY))
〈0)
printf("文件%s 无法打开! \n",av[1]);
else
{ft=fopen(av[2],"wt"; /*打开交互信息文件*/
 minlen=atoi(av[3]); /*可显示串最小长度*/
 pos=0L;i=len=0;
 while(read(fb,&ch,1)〉0) /*读一字符*/
 {post++; /*改变字符位置*/
  strend=1; /*可显示串结束标志初值置为真*/
  if(ch〉=32&&ch〈=126) /*若是可显示字符*/
  {str [i++]=ch; /*加到当前可显示串中*/
      len++; /*可显示串长度加1*/
      strend=0; /*可显示串结束标志置为假*/
      if(ch=='\\')str[i++]=ch;/*反斜线*/
 }else if (ch==0||ch)=7&&ch〈=13) /*特殊字
 符转换*/
 {str[i++]='\\';len++;/*字符加入串中*/
 switch(ch)
{ case 0:str[i++]='0' : break;/*NUL*/
  case 7:str[i++]='a';break;/*响铃*/
  case 8:str[i++]='b';break;/*退格*/
  case 9:str[i++]='t';break;/*水平制表*/
  case 10:str[i++]='n';break;/*换行*/
  case 11:str[i++]='v';break;/*垂直制表*/
  case 12:str[i++]='f';break;/*走纸*/
  case 13:str[i++]='r';break;/*回车*/
  }
if 束! =标strend=标/*志为假NUL,置结束标志
为假*/
}
if(ch=='¥'||len==MAXLEN) strend=1;
/*为真束标志 * 为真束标志已为真束标志 *为真
束标志为真*/
if(strend) /*结束显示串结束*/
  {*f/lee〉=m*eleen m 于串长不小于 minlen*/
    {str[i]='\0'; /*符串字符串*/
fprintf(ft,"%61d:%3d:%s\n",pos-len,len,str);
/*输出格式为:始位置:串长度:可显示串*/
      }
i=len=0;str[0]='0\'; /*重新初始化*/
    }
  }
close (fb);fclose(ft); /*关闭文件*/
  }
}
```

4.2 PUTINF.C 源程序清单

```
#include〈fent1.h〉
#include〈io.h〉
#include〈stdio.h〉
#include〈stdlib.h〉
#include〈string.h〉
void main(int ac,char *av[])
{ int fb,len,i,j,first=1;
  FIle *ft; /*交互信息文件指针*/
  long,p; /*当前行信息在软件中的始位置*/
  char ch,str[100],buf[140];
  if(ac〈3) printf("使用格式:PUTINF file text /n");
  else if ((fb = open (av[1], 0_ WRONLY | O.
  BINARY))〈0)
    printf("文件%s 无法打开! \n",av[1]);
  else if ((ft=fopen(av[2],"rt"))==NULL)
    printf("文件 %s 无法打开! \n",av[2]);
  else
{ while (fscanf (ft,"%61d%* c%3d%* c", &p,
&len)! =EOF /*对所有的行*/
  { fgets(buf,140,ft); /*读译文到buf*/
    i=j=0;
    while((ch=buf[i++])! = '\n') /*对本行所
    有的字符*/
    {if(ch== '\\') /*若是特殊字符*/
     {switch(ch=buf[i++]) /*转换下一字符*/
```

```
{ case '0' :str[j++]=0;break;/*NUL*/
  case 'a' :str[j++]=7;break;/*响铃*/
  case 'b' :str[j++]=8;break;/*退格*/
  case 't' :str[j++]=9;break;/*水平制表*/
  case 'n' :str[j++]=10;break;/*换行*/
  case 'v' :str[j++]=11;break;/*垂直制表*/
  case 'f' :str[j++]=12;break;/*走纸*/
  case 'r' :str[j++]=13;break;/*回车*/
  case '\\' :str[j++]=ch;break;/*反斜线*/
  }
 }else str[j++]=ch;/*若不是特殊字符,则复制*/
}
str[j]='\0';/*字符串加结束标志*/
if(j<=len)/*若译文未超长(j是译文实际长度)*/
{while(j<len)str[j++]=' ';/*后面补空格*/
str[len]='\0';/*字符串加结束标志*/
lseek(fb,p,SEEK_SET);/*找到其位置*/
—write(fb,str,len);/*写入软件中,实现汉化*/
}else /*若译文超长*/
 {if(first)/*若是第一次超长,给出提示*/
  {printf("有如下超长译文,请压缩:\n\n");
printf("始位置 原长 现长 译文\n");
first=0;/*标志变为不是第一次*/
  }
printf("%61d:%3d:%3d %s\n",p,len,j,str);/*输出超长译文*/
 }
}
close(fb);fclose(ft);/*关闭文件*/
}
```

[作者简介]李村合,男,1966年2月出生,1990年4月在北方交通大学获计算机及应用专业工学硕士学位,现为石油大学(华东)计算机系软件教研室讲师。主要从事算法设计与分析、管理信息系统、语言翻译程序及各种应用软件的研究和教学工作。 地址:257062 山东省东营市石油大学计算机系

用图象处理生成精密曲线轮廓字模

空军指挥学院 瞿 洋 苏恩泽

为了得到边缘光滑、字型美观的大点阵汉字,同时减少字库存储容量,人们开始寻找点阵存放方式以外的其它途径。其中,轮廓矢量化方法以其方法简便、压缩比高尤其引人注意。轮廓矢量化就是通过跟踪字符边缘,得到字符的边界点,然后,以一定精度用直线对边界点进行拟合,用拟合直线的端点代替边界点存于字库中,形成所谓矢量字库。最后再通过填充算法近似地恢复原点阵。矢量字库具有字型较为美观、存储容量小等优点。但是,放大成高于字库量化模板大小的点阵时,会出现较为明显的多边形轮廓特征。为了消除这一影响,用光滑的曲线代替直线拟合字符轮廓的思想油然而生。

其中一种直接的想法是用同线方程逐条拟合汉字的边缘轮廓,通过储存有限个方程的系数产生曲线字库。然而,由于轮廓边缘的复杂多样性,使得曲线方程的形式难以确定,给工程上实现这种想法带来了困难,并且工作量较大。比较实际的方法是利用汉字边缘轮廓上的急剧转折点(称为关键点)以及曲线形状控制点(称为插值点),进行分段曲线插值,从而恢复光滑的汉字轮廓形貌。如果关键点和插值点选择适当,这种方法不但可以快速地生成各种美观的汉字点阵,具有放大不变形的特点,同时,还可以进一步减少字库的存储容量。本文介绍的方法就是基于上述考虑。算法分成两个部分:精密汉字轮廓的提取和轮廓的曲线量