科技排版软件 TEX 中文接口

CCT DOS 版 参 考 手 册

中国科学院计算数学与科学工程计算研究所张林波编

一九九七年十一月

目 录

第一章　CCT 系统的总体介绍

§1.1　CCT 系统的运行环境要求

CCT MSDOS 版可在各种具有 512K 以上内存并配置有硬盘的 IBM–PC/XT, AT 及兼容机上运行。CCT 系统支持下列图形方式:

CGA (640×200), EGA (640×350), VGA (640×480), CGE (Color 400, 640×400) 及一些增强 VGA 方式及 VESA 方式 (800×600, 1024×768, 等等)。

CCT 支持的输出设备有: 各种 24 针打印机; HP, 北佳 (PECAN) 及与之兼容的激光打印机; HP, CANON 喷墨打印机; 长春 JZJ、IPX 激光照排机等。

CCT MSDOS 版运行所需要的环境为: MSDOS 3.0 以上版本, 任何基于 GB2312 编码的中文编辑系统 (用于汉字的输入和编辑), 以及 2.0 以上版本的 TEX 系统。

CCT 系统的最新 DOS 版本可从网址 `ftp://ftp.cc.ac.cn/pub/cct/msdos/` 获得。

§1.2　CCT 系统的构成

CCT 系统由下列文件组成:

初始化程序:	`CCTINIT.EXE`
预处理程序:	`CCT.EXE`
屏幕显示驱动程序:	`DVISCR.EXE`
24 针打印机驱动程序:	`DVI24P.EXE` (也用于驱动 Canon 喷墨打印机)
HP 激光打印机驱动程序:	`DVILJP.EXE` (适用于 HP LaserJet+ 激光打印机)
北佳激光打印机驱动程序:	`DVIPECAN.EXE` (适用于北佳视频卡)
喷墨打印机驱动程序:	`DVIDJ500.EXE` (适用于 HP DeskJet 系列喷墨打印机)
字号定义文件:	`CCT.DAT`
字库定义文件:	`CCFONTS.DEF`
绘图程序:	`PICTEX.EXE, PDFTOBMF.EXE`
图象图形接口程序:	`HPGL2CCT.EXE, IMG2CCT.EXE, BMF2TIF.EXE, BMF2PCX.EXE`
用户拼字程序:	`PZ.EXE`及有关文件

此外, CCT 系统还提供了一些非常有用的辅助程序, 包括: `MAKEPK.EXE`, `PATCHDVI.EXE`, `TEXT2DVI.EXE`, `UNCCT.EXE`, `BMFTEXT.EXE`, `FORALL.EXE`. 这些程序中均带有简单说明, 用户从文件名及程序输出的信息中很容易知道它们的功能与用法, 我们将不在本手册中对它们进行介绍。

§1.3　CCT 系统的基本使用流程

CCT 系统采用批处理方式。用户在使用 CCT 系统排版前, 必须首先准备好自己的 CCT 源文件 (CCT 源文件的扩展名通常应取为 "`.CTX`")。用户可以用自己熟悉的汉字编辑系统来准

[1] 由于北佳激光打印机驱动程序的用法与其它驱动程序类似, 因此我们在后面将不对它做专门介绍。

备 CCT 源文件。CCT 源文件除含有汉字外, 基本格式与 TEX 源文件一样 (有关 CCT 源文件的详细说明请参看 §2.1 (第 7 页))。由于 TEX 软件不能直接处理含有汉字的源文件, CCT 系统提供一个预处理程序 CCT.EXE 将 CCT 源文件转换为 TEX 源文件, 然后用户用 TEX 软件进行排版, 以产生一个 DVI 文件 (排版结果文件)。由于这种 DVI 文件中含有汉字信息, 因此不能直接用 TEX 软件所配置的驱动程序来显示或打印排版结果, 而必须使用 CCT 系统提供的驱动程序来输出排版的结果, 或者用PATCHDVI.EXE程序对 DVI 文件进行转换后再调用标准的 DVI 驱动程序 (如DVIPS.EXE) 来进行输出。

归纳起来用 CCT 系统进行排版的过程主要包括以下几步:

1. 准备 CCT 源文件。源文件中包含排版内容及排版命令。
2. 用 CCT 的预处理程序 CCT.EXE 将 CCT 源文件转换为 TEX 源文件。
3. 用 TEX 软件对得到的 TEX 源文件进行排版处理。
4. 用 CCT 的驱动程序显示或打印排版结果。

为更清楚起见, 我们看一个例子。假如用户的 CCT 源文件采用 LATEX 排版并且文件名为 "test.ctx", 则典型的排版过程包括以下几步:

- 将 test.ctx 转换为 test.tex:

```
C>cct test
```

- 用 LATEX 系统编译 test.tex:

```
C>latex test
```

- 显示排版结果:

```
C>dviscr test
```

- 打印排版结果:

```
C>dvi24p test
```

§1.4 CCT 初始化程序 CCTINIT.EXE

这个程序用来初始化 CCT 系统。它的作用是根据字号定义文件 CCT.DAT 中所定义的各个字号的大小生成 TEX 排版时需要用到的TFM 文件 (TEX Font Metric file) 以及宏定义文件 CCHEAD.STY。**在首次使用 CCT 系统前或每次修改了 CCT.DAT 中的参数后必须运行一次此程序以保证整个系统的正常运行。**

CCTINIT.EXE 的用法如下:

```
C>cctinit  [可选项]
```

其中的方括号代表可以省略的部分 (方括号本身不用给出), 称为命令行可选项, 用来改变或设置程序运行时的一些参数。每个可选项的第一个字符必须是 "-" (减号), 后面紧跟一个字符 (字母不区分大小写) 给出可选项的名称。有些可选项的后面还带有参数。"-" 与可选项名及参数之间不能有空格。不同的可选项之间必须用空格隔开。对于没用可选项给出的参数, 程序通常自动选取一个缺省值。

例如命令:

```
C>cctinit -t\tex\texinput -x
```

中包含两个可选项 "-t" 和 "-x", 可选项 "-t" 带有参数 "\tex\texinput"。

CCTINIT.EXE 有下面一些可选项:

-T 后面必须跟随一个字符串参数给出放置程序产生的TFM文件的路径名。缺省值为 CCT 系统所在子目录 +"\FONTTFMS"

-H 后面必须跟随一个字符串参数给出放置程序产生的 CCHEAD.STY 文件的路径名。缺省值为 CCT 系统所在子目录 +"\INPUTS"

-X 没有参数。给出此可选项时 CCT 利用 TEX 的 \special 指令来处理汉字的字体, 而无此可选项时 CCT 利用一个虚拟字库 "CCDUMMY.TFM" 来处理汉字的字体。两种形式对用户是等效的。

例如指令:

```
C>cctinit -hc:\tex\macros -tc:\tex\fonts
```

指示 CCTINIT.EXE 将生成的 CCHEAD.STY 文件放在 C: 盘的子目录 "\TEX\MACROS" 中, 而将生成的TFM 文件放在 C: 盘的子目录 "\TEX\FONTS" 中 (这些子目录最好事先建好)。

正常情况下用户不必给出可选项, 直接使用 CCTINIT.EXE 设定的缺省值即可。

"-H" 和 "-T" 可选项中的参数还可分别通过环境变量 TEXINPUTS 和 FONTTFMS 来进行设置。

用户也可以通过初始化文件 CCTINIT.INI 来设置参数值, 其用法与驱动程序类似, 参看 §3.1 (第 17 页) 最后的说明。

§1.5 CCT 预处理程序 CCT.EXE

CCT.EXE 将 CCT 源文件转换为 TEX 源文件。用户在准备好 CCT 源文件后必须首先将其转换为 TEX 源文件, 然后才能进行排版处理。

调用 CCT.EXE 的格式如下:

```
C>CCT  [输入文件名  [输出文件名]]
```

其中输入文件名为用户的 CCT 源文件名, 输出文件名为转换得到的 TEX 源文件名。如果省略输入文件名中的扩展名, 程序会自动加上 ".CTX" 作为扩展名; 如果省略输出文件名中的扩展名, 程序会自动加上 ".TEX" 作为扩展名; 如果省略输出文件名, 程序会取输入文件名加上扩展 ".TEX" 作为输出文件名。例如命令:

```
C>cct test
```

等价于

```
C>cct test.ctx test.tex
```

CCT.EXE 支持下述命令行选项:

-M 产生一个与 TEX 源文件同名, 但扩展名为 ".MAP" 的文件。这个文件中给出 CCT 源文件与 TEX 源文件中的行号对应关系, 供用户在 CCT 源文件中查错时用。文件中的每行具有如下格式:

CCT 文件中的行号 ==> TEX 文件中的行号

例如, 假如 ".MAP" 文件中包含下面一行:

```
5 ==> 6 7 8
```

则表示 CCT 源文件中的第 5 行对应着 TEX 文件中的 6, 7, 8 三行。

-S CCT.EXE 处理时将每个汉字码转换成对应的两个 ASCII 字符。如汉字"亮"被转换成两个 "A"。如果转换后的码正好是 TEX 的保留字符, 如 "\", "$" 等, 则CCT.EXE将它们转换

成 “\char###” 的形式, 其中 “###” 代表字符的 ASCII 码, 从而保证 TEX 能正确地处理这些字符。缺省情况下被转换成 “\char###” 形式的字符有下面一些:

```
# $ % & @ \ ^ _ { } ~
```

它们可满足大部分情况的要求。但有时用户希望CCT.EXE将上述字符之外的某些字符也转换成 “\char###” 的形式, “-S” 选项便是为此而设, 它后面需跟随一个 ASCII 码 (十进制数), 表示将该字符也转换成 “\char###” 的形式。用户可以使用多个 “-S” 选项。例如, 当使用 “\index” 命令及 “MAKEIDX.EXE” 程序生成索引时, 字符 “!” 被用作二级索引命令, 此时可使用 “-S33” 选项 (33 是 “!” 的 ASCII 码) 来使CCT.EXE将由汉字所产生的 “!” 写成 “\char33” 的形式, 以避免 “MAKEIDX.EXE” 将其当做二级索引命令 (参看 §2.4.3)。

-H 显示一个简单的使用说明。

用户也可以在初始化文件 CCT.INI 中使用上述选项。请参考 §3.1 (第 17 页) 最后的说明。

§1.6 字号定义文件 CCT.DAT

CCT 的程序运行时从 CCT.DAT 文件中得到有关汉字大小、间距等信息。用户可以通过修改此文件来设置每种字号的尺寸。如果修改了这个文件, 则需重新运行一次 CCTINIT.EXE 程序来生成相应的 “.TFM” 文件及 “CCHEAD.STY” 文件。CCT.DAT必须与CCTINIT.EXE位于同一子目录中。

“CCT.DAT” 是一个普通正文文件。它的开头是一个正整数, 给出可以同时使用的不同字号的个数, 最大不能超过 26 (因此用户最多只能同时使用 26 种不同大小的字号)。接着顺序给出定义每个字号的数据。每个字号由 5 个数定义, 其中前 4 个数是实数, 而第 5 个数为整数。这些数的含义如下:

第一个数: 字宽 (以 pt 为单位, 1 英寸 =72.27pt=2.54cm)

第二个数: 字高 (以 pt 为单位)

第三个数: 字距与字宽之比

第四个数: 行距与字高之比

第五个数: 给出 \zihao 指令中应该使用的参数 (即字号)

不同数据之间需用空格或换行隔开。

下面给出的是 CCT 系统配置的原始 CCT.DAT 文件中的数据, 它定义了十种不同大小的字号:

```
10
34.0    34.0    0.06        0.0         0
26.0    26.0    0.06        0.0         1
20.0    20.0    0.06        0.0         2
15.0    15.0    0.06        0.0         3
13.0    13.0    0.06        0.0         4
11.32   11.32   0.06        0.0        -4
9.9     9.9     0.06        0.0         5
8.5     8.5     0.06        0.0        -5
```

```
7.5     7.5     0.06            0.0             6
5.0     5.0     0.06            0.0             7
```

例如第四行数据说明 \zihao{0} (初号字) 的大小为: 字宽 $= 34$pt, 字距 $= 34 \times 0.06 = 2.04$pt, 字高 $= 34$pt, 行距 $= 34 \times 0.0 = 0$pt. 由于其中定义的字号属于标准字号, 因此用户最好不要修改它们的大小, 若想使用其它大小的字号可以加在它们后面。

值得注意的是, CCT.DAT 文件中所定义的大小不是绝对的。如果用户在 CCT 源文件中或输出结果时使用了 "\magnification" 参数, 则所有的汉字也会相应地被放大或缩小。

§1.7 字库定义文件 CCFONTS.DEF

这是一个只被 CCT 的设备驱动程序用到的正文文件。它定义汉字字库的文件名。

CCT 允许用户最多同时使用 26 种字体。每种字体被分为两级, 分别存贮在两个字库文件中。第一级字库包括区位表中第 16 区到第 55 区中的汉字, 第二级字库包括第 56 区到第 87 区中的汉字。另外还有一个文件包含区位表中第 1 区到第 15 区中的符号和一个用户字库文件。CCFONTS.DEF 文件中按下面的顺序给出这些字库文件的文件名:

用户字库
标点符号库 (1–15 区)
第一种字体 (\ziti{A}) 的一级字库名
第一种字体 (\ziti{A}) 的二级字库名
第二种字体 (\ziti{B}) 的一级字库名
第二种字体 (\ziti{B}) 的二级字库名
....

前五种字体通常保留为宋体、黑体、楷书体、仿宋体和标宋。

每个字库文件名后面还必须跟随一个比例因子用来调整该字库的相对大小。字库文件名与比例因子之间必须用空格或换行隔开。由于驱动程序忽略这些数据之后的内容, 用户可在最后一个字库文件名和比例因子之后加入自己的注解或说明。

下面是一个字库定义文件的例子, 其中定义了宋体、黑体、楷书、仿宋和标宋五种标准字体的字库名。

```
USERFONT.PZ   1.0
CLIB256E.PPS  1.0
CL256S0.PPS   1.0
CL256S1.PPS   1.0
CL256H0.PPS   1.0
CL256H1.PPS   1.0
CL256K0.PPS   1.1
CL256K1.PPS   1.1
CL256F0.PPS   1.1
CL256F1.PPS   1.1
CL96BS0.PPS   1.0
CL96BS1.PPS   1.0
```

它说明 1–15 区的符号在文件 "CLIB256E.PPS" 中, 一级宋体字库在文件 "CL256S0.PPS"

中, 二级宋体字库在文件 “CL256S1.PPS” 中, 用户字库文件为 “USERFONT.PZ”, 等等。其中仿宋体和楷体输出时将被放大 1.1 倍。

如果需要使用五种基本字体之外的字体, 可将相应的字库文件名加在上述五种基本字库文件之后, 并在排版源文件中利用 \ziti 命令来选用相应的字体 (参看 §2.1 (第 7 页))。

比例因子限制在 0.5 到 2 之间。如果所给出的比例因子小于等于 0.5 或大于等于 2 则它将被忽略。

CCT 5.13 版的驱动程序在调入用户字库时, 首先在 DVI 文件所在的目录中寻找与 DVI 文件同名, 扩展名为 “.PZ” 的文件, 如果该文件存在, 则将该文件做为用户字库调入, 当该文件不存在时才使用CCFONTS.DEF 中定义的用户字库。这样做可以方便用户在不同的 DVI 文件中使用不同的用户字库。

第二章　CCT 的排版命令

§2.1　CCT 源文件的格式

标准的 TEX 源文件由一个或多个 ASCII 文件组成。文件中包含正文和指令两部分内容。正文只需直接键入, 而各种指令则用来控制字体的选择及排版的方式等等。通常, TEX的指令以字符 "\" 开头, 后面跟一个指令名, 指令名必须由字母构成, 如 `\rm`, `\vskip` 等等。一般情况下, TEX 源文件应以 ".TEX" 为其扩展名。

CCT 源文件与TEX 源文件的格式一样。正文 (包括中文、英文等等) 可以直接输入而指令仍然由英文字母构成。用户可以用一个自己熟悉的中文编辑程序来准备 CCT 源文件。CCT 源文件的扩展名最好是 ".CTX"。用户在准备好 CCT 源文件后, 只需用预处理程序 CCT.EXE 将其转换为TEX源文件便可用TEX 对它进行排版。

除了标准的 TEX 命令外, CCT 系统定义了下面一些与汉字有关的命令:

```
\ziti       \zihao      \ziju
\songti     \heiti      \kaishu     \fangsong  \biaosong
\ccwd       \ccht       \ccdp
\pushziti   \popziti
```

由于这些指令定义在文件 "CCHEAD.STY" 中, 因此用户在 CCT 源文件开头必须加入下面的指令:

```
\input cchead.sty
```

或者 (LATEX 2_ε)

```
\usepackage{cchead}
```

来调入这些命令。下面分别说明这些命令的含义及用法。

§2.1.1　`\zihao` 命令

用来选择汉字的大小 (即字号), 后面必须用一个括在大括号中的整数给出选用的字号。如指令 `\zihao{5}` 将当前汉字的大小设为五号字。各个字号的大小由 "CCT.DAT" 文件定义 (参看 §1.6 (第 4 页))。CCT 系统配置的标准的 "CCT.DAT" 文件中允许用户使用十种不同大小的字号, 它们是:

CCT 指令	对应的字号	CCT 指令	对应的字号
`\zihao{0}`	初号	`\zihao{-4}`	小四号
`\zihao{1}`	一号	`\zihao{5}`	五号
`\zihao{2}`	二号	`\zihao{-5}`	小五号
`\zihao{3}`	三号	`\zihao{6}`	六号
`\zihao{4}`	四号	`\zihao{7}`	七号

例: 下述指令:

```
\zihao{5} 这是五号字; \zihao{-4} 这是小四号字
```

的输出为:

这是五号字; 这是小四号字

§2.1.2 \ziti, \songti, \kaishu, \heiti, \fangsong 及 \biaosong 命令

\ziti 命令用来设置汉字的字体, 后面跟随一个括在花括号中的字母 (不区分大小写)。通常 \ziti{A}, \ziti{B}, \ziti{C}, \ziti{D} 和 \ziti{E} 分别保留为宋体、黑体、楷书体、仿宋体和标宋, 它们可分别简写为 \songti, \kaishu, \heiti, \fangsong 及 \biaosong。如果用户想要使用这五种标准字体之外的字体, 则可用命令 \ziti{F}, \ziti{G} 等等来选择, 并在文件 CCFONTS.DEF 中给出相应的字库文件名 (参看 §1.7 (第 5 页))。用户最多可同时使用 26 种不同字体。

例: 下述指令

```
\zihao{5}\songti 这是宋体五号, \kaishu\zihao{-5} 这是楷书体小五号,
\songti 这是宋体小五号
```

的输出为:

这是宋体五号, 这是楷书体小五号, 这是宋体小五号

§2.1.3 \ccwd, \ccht, \ccdp 命令

它们分别给出当前字号的字宽 + 字距 (width+inter word space)、字高 (height) 和字深 (depth)。其中字高指的是一个字符在其所在行的基线以上部分的高度, 而字深指的是基线以下部分的高度。它们可做为长度单位使用。如: \hsize=40\ccwd 将版心宽度置为 40 个字宽, \parindent=2\ccwd 将段头空设为两个当前字号的字宽。

§2.1.4 \ziju 命令

用来改变缺省的汉字字距。

有时用户希望在同一篇文章中使用不同的字距和行距。通过在 CCT 源文件中适当设置 TEX 的 \baselineskip, \lineskip 和 \lineskiplimit 等变量, 用户可以得到任意的行距。为使用户能够在 CCT 源文件中修改字距, "CCHEAD.STY" 中定义了一条命令 "\ziju" 用来设置字距, 它包含一个参数给出字距与字宽之比。例如, "\ziju{0.25}" 将字距设为字宽的 0.25 倍。本段开头包含命令 "\ziju{0.24}", 因此段落中的字距是其它段落的 4 倍 (本说明书使用的正常字距为 0.06)。

§2.1.5 \pushziti, \popziti 命令

这两条命令均没有参数。\pushziti 命令在输出时使得驱动程序将当前字体压入一个栈中, 而 \popziti 命令使得驱动程序将当前字体恢复成上一次 \pushziti 命令所保存的字体 (栈的最大深度为 16)。

用 TEX 系统排版时有些内容 (例如页眉, 页脚, 脚注以及 LATEX 中的 table 和 figure 环境等) 输出时不一定出现在它们在源文件中所处的位置上, 这些内容通常称为浮动体 (因为排版处理后它们可能"浮动"到其它位置上)。由于受 CCT 目前的处理方式的影响, 如果一个浮动体中的汉字字体与它在输出时所处位置上的其它部分的字体不同, 则紧跟在浮动体后面的汉字的字体可能会被改变。这个问题最典型的例子是一页开始的几个汉字的字体变成了与页眉中的字体一样而它们本来应该是不同的。上述两条命令即是为解决此问题而设置: 用户只需

在一个浮动体的开始加入一条 \pushziti 命令, 而在最后加入一条 \popziti 命令即可避免它改变所处位置上的汉字字体。

做为一个例子, 当使用 LaTeX 时可采用下述方式定义书眉:

```
\markboth{\pushziti奇数页书眉\popziti}{\pushziti偶数页书眉\popziti}
```

来避免当书眉与正文字体不同时正文字体受书眉影响而发生变化。

说明:

1. 如果用户想要在数学状态下加入汉字, 则所有汉字都必须包含在 \hbox 或 \vbox 中, 例如指令:

   ```
   \songti\zihao{5}$\sin(x)\quad\hbox{正弦函数}$
   ```

   的输出为:

   $\sin(x)$ 正弦函数
2. 所有有关汉字的指令与一般的TEX指令的用法完全一样, 它们的作用区域符合TEX 的分组规则 (即 "group"), 这可由下例说明

   ```
   \songti\zihao{5}这是宋体五号{\zihao{6}这是宋体六号
   \kaishu这是楷书六号}这是宋体五号
   ```

   输出为:

   这是宋体五号这是宋体六号这是楷书六号这是宋体五号

   这种性质使得 CCT 较好地保持了与西文TEX 的兼容性, 从而能够处理各种带浮动的情况。
3. 上述指令不改变当前英文的字体和大小。

§2.2 用户造字功能

CCT 提供了一个拼字程序 PZ.EXE 用来拼造区位表中没有的字, 用户可利用它来建造用户字库。为将用户字库中的字加入排版中, 只须在 CCT 的源文件中使用形如的 "#[n]" 命令, 其中n代表非负整数, 它给出所要插入的字在用户字库文件中的序号 (第一个字序号为 0, 第二个字序号为 1, 依次类推)。例如, "#[15]" 代表用户字库中的第 16 个字。所给序号必须在 0 到 3759 之间 (因此一共可使用 3760 个用户字)。命令之间不能有空格及其它字符。任何不符合上述格式的命令将被 CCT.EXE 当做普通 ASCII 字符串拷贝到输出文件中。

用户若想要在源文件中输入形如 "#[2]" 的字符串而不想让 CCT 将其误认为用户字库中的字, 只需在其中插入一些不起作用的其它字符, 例如, 可将它写成 "#{}[2]"。

输出使用了用户字库的 DVI 文件时, 用户可以在文件 "CCFONTS.DEF" 中给出用户字库文件名 (参看 §1.7 (第 5 页)) 并且将用户字库文件与其它标准字库放在同一子目录中, 也可以将用户字库文件放在 DVI 文件所在的目录并且将用户字库文件名取为 DVI 文件的文件名加上扩展名 ".PZ"。

有关拼字程序 PZ.EXE 的使用将在第五章中进行专门介绍。

§2.3 emTEX 的画线命令

emTEX 是MSDOS 系统上最为流行的一个免费 TEX 版本。emTEX 中定义了一套画任意斜率的直线的命令, 扩展了 LATEX 的`picture` 环境中的绘图功能。目前有一些软件 (如 TEXCAD) 中也用到了 emTEX 的画线命令。CCT 的驱动程序支持 emTEX 的画线命令, 这一方面可方便用户直接绘制一些示意图, 另一方面也使得用户能够在 CCT 中使用基于 emTEX 的画线命令的其它软件。

CCT 所支持的 emTEX 命令有以下五条(描述命令格式时方括号 `[...]` 代表可选项, 即可以省略的部分, 而竖线 `|` 用来描述一个参数的几种允许值):

1. `\special{em:point` n`}`
   将当前页面位置定义为点 n, 其中 n 是介于 1 到 32767 之间的整数给出该点的点号。点的定义只对当前页有效。
2. `\special{em:line` a`[h|v|p],`b`[h|v|p][,`w`]}`
   画一条从点 a 到点 b 的直线, 直线线宽为 w, 其中 a, b 为点的编号 (由命令 `em:point` 所定义), w 中可采用下述长度单位:

`cm`	centimeter (厘米,)
`mm`	milimeter (毫米, 10 mm = 1 cm)
`in`	inch (英寸, 1 in = 2.54 cm)
`pt`	point (磅, 1 in = 72.27 pt)
`sp`	scaled point (1 pt = 65536 sp)
`bp`	big point (1 in = 72 bp)
`pc`	pica (1 pc = 12 pt)
`dd`	didot point (1157 dd = 1238 pt)
`cc`	cicero (1 cc = 12 dd)
`px`	pixel (象素)

   其中前 9 个为绝对单位, 最后一个与输出设备的分辨率有关, 省缺单位时以 cm 为单位。如果命令中省略 w, 则线宽取为由 `\special{em:linewidth` w`}` 命令所定义的宽度。点号 a 和 b 后面还可跟随字母 `h`, `v` 或 `p` 来描述直线端点处的切口形状, `h` 表示横向切口, `v` 表示纵向切口, `p` 表示切口与直线垂直, 缺省值为 `p`。如:

   ```
   \special{em:line 1h,2v,0.8pt}
   ```

   表示画一条点 1 到点 2 宽为 0.8pt 的直线, 直线在点 1 处为横向切口, 而在点 2 处为纵向切口。
3. `\special{em:linewidth` w`}`
   将缺省线宽设为 w。缺省线宽的初始值为 0.4pt。
4. `\special{em:moveto}`
   将当前页面位置定义为画线的当前点。

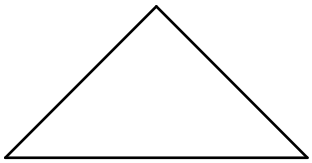

图 2.1: emTEX 绘图实例

5. \special{em:lineto}
   从画线的当前点画一条至当前页面位置的直线, 线宽由 linewidth 定义, 并将画线的当前点移至当前页面位置。
   用该命令画线时先用 moveto 命令将游标移至起点位置, 然后可用多个 lineto 命令来画出一条连续曲线。

例如, 下述命令:

```
\special{em:linewidth 1pt}
\begin{center}
\unitlength=1mm
\begin{picture}(40,20)
  \put(0,0){\special{em:point 1}}
  \put(20,20){\special{em:point 2}}
  \put(40,0){\special{em:point 3}}
  \special{em:line 1,2} \special{em:line 2,3}
  \special{em:line 3,1}
\end{picture}
\end{center}
```

将用 1pt 宽的线画出一个如图 2.1 所示的三角形。 图 2.1 也可用下面的命令来画出:

```
\special{em:linewidth 1pt}
\begin{center}
\unitlength=1mm
\begin{picture}(40,20)
  \put(0,0){\special{em:moveto}}
  \put(20,20){\special{em:lineto}}
  \put(40,0){\special{em:lineto}}
  \put(0,0){\special{em:lineto}}
\end{picture}
\end{center}
```

注: 激光打印机驱动程序 DVILJP.EXE 只能输出利用上述命令画的简单图形, 对于复杂的图形最好使用 DVIDJ500.EXE 来输出 (当用 DVILJP.EXE 输出含有 emTEX 的画线命令的文件时程序会给出一个警告信息)。

§2.4 关于 CCT 的一些问题及使用时的注意事项

本节中我们列出一些 CCT 已知的问题及使用时如何避免它们。我们将根据我们使用 CCT

过程中的经验及用户提供的信息不断丰富本节内容, 以帮助用户改善排版的质量。

§2.4.1 汉字与西文之间的空

假设 CCT 的源文件中含有下面的内容:

```
CCCCCCCCeeeeCCCCCCCC
```

其中 “C” 代表汉字, 而 “e” 代表其它 (非汉字) 字符, 则 CCT 的预处理程序将其转换成:

```
{\CC ........} eeee {\CC ........}
```

其中 “...” 代表汉字转换后产生的字符。这样的处理使得在汉字与西文字符间有相当于一个西文空格的空。用户排版源文件中中西文之间留不留空格都不影响排版输出的结果。但有时用户不希望有多余的空格。例如下面的一段排版命令:

```
啊啊啊 \index{abc} 啊啊
```

它的排版输出为:

啊啊啊 啊啊

其中出现了一个多余的空格。目前 CCT 的处理方式尚无法自动避免这个问题。做为一个解决方案, 我们建议用户定义下面两条命令:

```
\def\CS{{\CC\ }}
\def\ccnospace#1{\leavevmode\unskip #1\ignorespaces }
```

其中命令 “\CS” 留出相当于两个汉字之间字间距的空, 而 “\ccnospace” 则用来取消多余的空。将上例中的排版命令改成:

```
啊啊啊 \ccnospace{\CS\index{abc}} 啊啊
```

则输出变为:

啊啊啊啊啊

即保证了汉字字间距的均匀性。注意上例中使用了 “\CS” 命令以留出汉字间的正常字间距。

对于一些需要大量在汉字中间使用而又不希望引入多余的空格的命令, 可引进适当的宏定义以简化排版。例如, 如果需要大量使用 \index 命令, 则可引进定义:

```
\def\cindex#1{\ccnospace{\CS\index{#1}}}
```

在两个汉字之间使用 \cindex 命令, 而在其它情况 (前或后为西文时) 仍使用 \index 命令, 便可避免在插入 \index 命令的地方出现多余的空格。

如果插入的内容由汉字构成, 当用 “\ccnospace” 命令去除多余空格时, 还需注意在前后留出汉字的正常字间距。请看下例:

```
\ziju{0.5} 宋体宋体 \ccnospace{{\heiti 黑体 }} 宋体宋体
```

这里为了看得更清楚, 我们有意使用了大的字间距。上述排版命令的输出如下:

宋 体 宋 体 **黑 体** 宋 体 宋 体

注意“黑体”前后的空被 “\ccnospace” 命令吃掉了。上述排版命令应该写成如下形式:

```
\ziju{0.5} 宋体宋体 \ccnospace{\CS{\heiti 黑体 }\CS} 宋体宋体
```

才能得到正确的输出结果:

宋 体 宋 体 **黑 体** 宋 体 宋 体

还有一点要注意的是与西文排版一样, TEX 命令后面的空格不起作用, 因此必要时需要注意命令的写法。如应该将

```
中文 \TeX 排版系统
```

写成

```
中文 \TeX\排版系统
```

具体排版过程中用户可根据情况灵活使用上述技巧来保证排版质量。最好结合宏定义来使用上面介绍的命令, 使得排版源文件更加简洁。

§2.4.2 中文排版的断行处理

中文排版与西文排版一样, 对于在何处断行有所限制, 排版术语中称为“行禁则”。中文排版的行禁则主要与标点符号有关, 如 “,”, “;” 等之前不应断行, “(” 之后不应断行, 等等。无论是中文还是西文, 用户排版时都应该注意不要在不应断行的地方输入多余的空格。只要做到这一点, CCT 基本上可以保证排版输出符合行禁则。必要时也可使用 LaTeX 的“\nolinebreak” 命令 (当在汉字中间时应该写成 “\ccnospace{\CS\nolinebreak}” 的形式来避免引进多余的空格) 或 “\mbox” 命令来禁止在某处断行。

§2.4.3 西文标点符号的转换

一些用户在输入标点符号时习惯使用西文标点符号 (半角符号), 而有的则喜欢使用中文标点符号 (全角符号)。为了保持排版输出的一致性, CCT 的预处理程序自动将一些紧跟在汉字后面的半角符号转换成全角符号。这样的符号有: “!”, “,”, “.”, “:”, “;”, “?”. 通常这样处理不会产生问题, 事实上许多用户从未注意到这种转换。但在一些特殊情况也须加以注意。例如, \index 命令中用字符 “!” 来表示次级索引, 因此使用 \index 命令生成索引时可能会有下面形式的排版命令:

```
\index{ 动力系统 ! 耗散型动力系统 }
```

其中的字符 “!” 用来指明“耗散型动力系统”是一个二级索引。但由于 “!” 跟在汉字的后面, 因此 CCT 的预处理程序会将它转换成全角字符。此时用户需在 “!” 的前面插入一些不影响排版的字符将它与汉字隔开来避免发生问题。例如, 可将上面的排版命令写成:

```
\index{ 动力系统 {}! 耗散型动力系统 }
```

或

```
\index{ 动力系统 \relax! 耗散型动力系统 }
```

此外在用 “CCT.EXE” 处理时还应该使用 “-s33 -s34 -s124” 选项, 避免索引中的汉字码转换成字符 “!”, “"” 和 “|” (参看 §1.5).

§2.4.4 标点符号与汉字的对齐

CCT 排版时会将一些标点符号前面或后面的空减少一些, 使它们的宽度比一个汉字小。但有些时候用户希望标点符号与汉字的宽度一样以便对齐, 为此只需使用下面形式的排版命令:

```
{
    \def\CCA{ }\def\CCAS{ }
    \def\CCB{ }\def\CCBS{ }

    排版内容 ... ...
}
```

§2.4.5 用字数控制距离

有时用户希望用汉字的字数为单位来进行度量。例如为了空出两个汉字的距离, 有的用户可能会使用 "\hskip2\ccwd" 命令, 但由于 §2.4.1 节中所指出的原因, 这样留出的空实际上多出一个西文的空格。该问题可使用 §2.4.1 节所介绍的 "\ccnospace" 命令来解决, 即使用形如 \ccnospace{\CS\hskip2\ccwd} 的命令。更为简单的办法是输入两个区位码为 0101 的字符, 因为该字符不产生排版输出, 而它的宽度正好为一个汉字的宽度。

§2.4.6 关于字符 "%" 的处理

字符 "%" 通常用来在源文件中引入说明。为了避免出现花括号不配对、汉字换行时产生多余的空格等麻烦, CCT 预处理程序在转换时不处理 "%" 后面的汉字, 而简单地将它们拷贝到输出文件中去。但如果 "%" 前面的字符是 "\", 则 CCT 认为它不是说明字符而对跟随在它后面的汉字照常处理。

用户需要注意的是, 当排版正文中用到 "%" 时要注意写法以免引起错误。例如, 当使用 LaTeX 命令"\verb+%+" 时, 排版源文件中同一行上字符 "%" 的后面不能有汉字, 否则将会出错。当然, 用户也可将命令写成其它形式, 如 "\texttt{\%}" 或 "\texttt{\char37}".

§2.5 CCT 的其它实用程序

§2.5.1 PATCHDVI.EXE

有时用户希望使用标准的 DVI 驱动程序输出用 CCT 排版的含有汉字的 DVI 文件。例如, CCT 系统没有提供 PostScript 驱动程序, 用户无法直接将排版结果转换成 PostScript 的格式。此外 CCT 的设备驱动程序目前不支持一些在西文 TeX 中广为使用的 \special 命令, 包括 PostScript 图形的插入、彩色打印以及 LaTeX 2ε 的graphics 包中提供的缩放、旋转等功能。为此, CCT 系统中提供了一个转换程序 PATCHDVI.EXE, 它可以将含有汉字的 DVI 文件变成标准的 DVI 文件, 同时将其中的汉字转换成临时的 PK 字库, 转换后的 DVI 文件可以用任何标准的 DVI 驱动程序进行显示、打印。这样, 用户可以根据自己的排版需要选用任何 DVI 驱动程序输出排版的结果。目前使用最广、功能最全的 DVI 驱动程序是 dvips, 国内用户可从 FTP 站点:

ftp://ftp.pku.edu.cn/pub/emtex/emtex-dos/dvips

处下载 dvips 的 DOS 版本。

我们不打算介绍 PATCHDVI.EXE 的详细用法。用户只需在 DOS 命令行上键入 PATCHDVI 命令, 即可得到一个简要的使用说明。

§2.5.2 MAKEPK.EXE

有时打印或显示 DVI 文件时, 会产生形如 "cannot load "???.PK"" 或 "Cannot find subdirectory" 之类的错误信息, 这是因为驱动程序找不到 DVI 文件所需要的一些 PK 字库。这些 PK 字库通常由 MF 字库生成。为了方便用户生成 PK 字库, CCT 提供了一个程序 MAKEPK.EXE, 其使用格式如下:

C>makepk [选项] DVI文件名 [> temp.bat]

上面的命令生成指定的 DVI 文件所用到的 PK 字库。MAKEPK.EXE 支持下列选项:

-R 给出 PK 字库的分辨率, 如 -R300, -R240:216 等。合法的分辨率定义在文件 MAKEPK.MOD 中 (该文件必须与 MAKEPK.EXE 位于同一目录并且其中的定义必须与

%EMTEXDIR%\mfinput\etc\local.mf

文件一致), 这里 %EMTEXDIR% 指 emTEX 所在的目录。不给出该选项时使用 MAKEPK.MOD 中定义的第一种打印机的分辨率。

-M 给出 magnification. 缺省值为 1000.

-Q 取消 MAKEPK.EXE 运行时的信息。

-A 强制生成 DVI 文件需要的所有 PK 字库。缺省时只生成所缺的 PK 字库。

-T 将产生的 TFM 文件拷贝到目录 %EMTEXDIR%\tfm 下。

-E 如果不使用 "-E" 可选项, MAKEPK.EXE 并不直接生成 PK 字库, 而是将生成 PK 字库的命令写到标准输出。用户可将这些输出定向到一个批命令文件中 (如 ">temp.bat"), 然后执行该批命令文件来产生所需的 PK 字库。如果使用了 "-E" 可选项, 则 MAKEPK.EXE 将直接执行生 PK 字库的命令, 而不是将这些命令显示出来。

使用 MAKEPK.EXE 时必须安装 emTEX 的MF386.EXE 程序及相关文件。

MAKEPK.EXE 使用与它处于同一子目录的文件MAKEPK.MOD 中第一个与用户所指定的分辨率相匹配的 MetaFont 模式名 (mode_def)。如果省略 "-r" 选项, 则MAKEPK.EXE 使用文件MAKEPK.MOD 中定义的第一个模式名。用户可以编辑文件MAKEPK.MOD, 将对应于自己所使用的打印机的模式定义移到文件开头, 这样在调用MAKEPK.EXE 时可以省略 "-r" 选项。

此外, 如果用户在文件 %EMTEXDIR%\MFINPUT\ETC\LOCAL.MF 中增加了新的模式并生成了新的CM.BAS 和PLAIN.BAS 文件 (这两个文件应该位于目录 %EMTEXDIR%\BMFBASES 中), 则需将所增加的模式定义加到MAKEPK.MOD 文件中去, 请参考该文件中其它模式的定义形式。

§2.5.3 UNCCT.EXE

UNCCT.EXE 程序将经过 CCT 预处理产生的文件"复原"成转换前的源文件。用户如果丢失了 CCT 源文件但有 CCT 预处理后产生的 TEX 文件, 可以用这个程序来恢复 CCT 源文件。我们曾在排印一本书时利用该程序对 LATEX 2_ε 产生的包含汉字的 .idx 文件进行处理, 然后进行排序, 最终生成书中的索引。

§2.5.4 PSCVT.EXE

目前使用最广泛的 DVI 驱动程序是 dvips, 它将 DVI 文件转换成 PostScript 文件。dvips 支持几种形式的\special 命令, 它们被用来在排版中插入 PostScript 格式的图形文件或产生一些特殊的排版效果 (对字符进行旋转、缩放等)。许多绘图软件与 TEX 的接口中都使用了dvips 的这一功能。用户也经常用epsf.sty 中所定义的命令在排版中插入 PostScript 图形文件。由于 CCT 的驱动程序不支持 dvips 的\special 命令, 因此无法显示或打印出插在排版中的 PostScript 图形。为了部分地解决这个问题, CCT 系统提供了一个转换程序PSCVT.EXE, 它将一个包含有dvips 的\special 命令的 DVI 文件转换成一个新的 DVI 文件, 其中dvips 的\special 命令被转换成 CCT 支持的 "\special{BMF=...}" 的形式, 并自动生成相应的 BMF 文件, 这样用户可以用 CCT 的设备驱动程序处理新的 DVI 文件从而得到正确的插图。

PSCVT.EXE 程序运行时需要调用 GhostScript 的 DOS 版程序GS386.EXE, 以及 CCT 的图形转换程序IMG2CCT.EXE (参看 4.4), 用户可以通过修改文件PSCVT.INI 中的 "-g" 可选项来设

定 GhostScript 的路径。关于PSCVT.EXE 的命令行格式可参看程序运行时的说明。

目前版本的PSCVT.EXE 程序只处理 “\special{psfile=...}” 形式的命令 (epsf.sty) 以及 “\special{" ...}” 形式的命令。当遇到其它形式的dvips \special 命令时 (它们包括 “\special{header=...}”, “\special{! ...}”, “\special{ps:...}”), 程序会给出警告。

第三章　CCT 的设备驱动程序

TEX 软件采用批处理方式, 它将排版结果存在一个 DVI 文件中 (device independent file 的简称, 以 ".DVI" 为扩展名)。为了显示或打印排版结果, 需要使用适当的设备驱动程序 (device driver)。使用 CCT 系统时, 由于产生的 DVI 文件中包含有有关汉字的特殊信息, 因此不能直接用 TEX 软件原来配置的设备驱动程序来进行处理。为此, CCT 提供了一系列适用于不同类型显示器、打印机的设备驱动程序。这些驱动程序在处理只含西文的 DVI 文件时与 TEX 软件的设备驱动程序功能一样, 同时它们还可以处理由 CCT 系统产生的含有中文的 DVI 文件。另外, 这些设备驱动程序还允许用户通过 BMF 文件将事先准备好的图形、图象直接插入排版结果中 (参看第四章), 实现图文同时输出。

如果用户希望使用 TEX 系统中的标准驱动程序(例如用DVIPS.EXE输出 PostScript 文件), 则必须先用程序PATCHDVI.EXE 对 DVI 文件进行转换。

CCT 系统配置了以下几个基本的设备驱动程序:

DVISCR.EXE:　用于在屏幕上显示排版结果

DVI24P.EXE:　用于各种 24 针打印机

DVILJP.EXE:　用于 HP LaserJet+ 或与之兼容的激光打印机

DVIDJ500.EXE: 用于 HP DeskJet 系列喷墨打印机及 LaserJet 系列激光打印机

DVIPECAN.EXE: 用于北佳系列激光打印机 (配置北佳视频卡)

我们将在 §3.1 中介绍这些驱动程序的共同特点, 在 §3.2 至 §3.4 中分别对每个驱动程序加以补充性的说明。另外还有用于驱动一些特殊设备 (如激光照排机) 的驱动程序, 它们的使用同上述程序类似, 我们不在这里进行介绍。

§3.1　驱动程序的参数及命令行可选项

所有的设备驱动程序都用到一些参数, 用户通过设置这些参数来控制输出结果。对大多数参数而言, 用户只需使用程序设定的缺省值而无需改变它们。

用户可通过多种方式改变参数的值: 第一种方式是在启动驱动程序的命令行上用可选项的形式给出, 主要用在批命令文件中, 以设置一些与驱动程序内部设定的缺省值不同的值作为每次使用的缺省值; 第二种方式是在驱动程序的菜单中直接修改参数值, 可用来在每次运行时临时改变一些参数; 第三种方式是通过 DOS 的环境变量进行设置; 第四种方式是通过一个与驱动程序同名、扩展名为 ".INI" 的初始化文件来设置参数的缺省值 (参看本节最后的说明)。

下面介绍 CCT 的驱动程序的用法及各个参数的作用。不同驱动程序的具体使用将在后面几节中分别加以介绍。

启动 CCT 的驱动程序的格式如下:

C>*驱动程序名*　[可选项]　[DVI文件名] [可选项]

其中 DVI 文件名给出排版结果文件名 (当省略扩展名时驱动程序会自动加上 ".DVI")。例如:

```
C>dviscr -gvga test -r300
```

上例中调用驱动程序 "DVISCR.EXE", "-g" 和 "-r" 为可选项, DVI 文件为 "TEST.DVI"。所有可选项都必须以字符 "-" (减号) 开始, 以便同 DVI 文件名区分开来 (用户应该避免将 DVI 文件名的第一个字符取为 "-"), 紧跟着 "-" 的字符即是可选项的名称 (如果是字母的话不区分大小写)。大部分可选项后面还需跟随一些参数 (如上例中的 "vga" 和 "300")。可选项可放在 DVI 文件名的前面或后面, 它们出现的顺序不限, 但可选项与可选项之间、可选项与 DVI 文件名之间必须用至少一个空格隔开。如果同一个可选项出现两次以上, 则以最后一次的值为准。我们将结合驱动程序的菜单来解释每个可选项的含义。

启动驱动程序后, 用户便进入一个菜单。菜单中的每一项显示出一个参数的当前值。这些值或是驱动程序内部设定的值, 或是在命令行中用可选项给出的值, 也有些值可能是通过环境变量或 ".INI" 文件设置的。每一项的名称后面还有一个括在方括弧中的字母, 它给出的是对应的命令行可选项的名称。用户可利用上下箭头键在不同的项目之间移动, 也可按方括号中的字符键来直接跳至某一参数。若想修改游标处的参数值, 只需按一下左箭头键或右箭头键 (如果游标在别处也可直接按方括弧中的字母), 这时有下面两种可能的反应:

1. 游标所在的项被清除并且用不同的底色显示。这表明已经进入编辑状态。此时用户可以直接输入新的参数值。输入过程中用户可以用左右箭头键来移动游标, 用 <Ctrl> 键加左右箭头键来快速移动游标, 用 <Ins> 键来切换"插入"和"改写"状态, 用 <Del> 或 <Backspace> 键来删去一个字符。此外, 还可用 <F3> 键回忆出原来的内容, 用 Ctrl-Y 删去当前的内容。若想放弃修改过的内容而保留原来的参数值, 只需按 <Esc> 键。修改完毕后, 按回车键即可退出编辑状态。
2. 有些参数项只允许用户在几个规定的值中选取一个。如果用户在这些项上, 则按一次右箭头键使参数取下一个可能值, 而按一次左箭头键使参数取前一个可能值。

当用户修改完毕菜单中的参数后, 可按回车键来开始程序的运行, 也可按 <Esc> 键来放弃程序的运行而返回 DOS。

驱动程序运行时, 首先调用程序 MAKEPK.EXE(参看 14 页 §2.5.2) 生成缺少的 PK 字库, 然后扫描一遍需要处理的页, 接着将用到的中西文字符调入内存, 然后顺序输出每一页。

在调入一个西文字库时驱动程序在屏幕上显示出正在调入的字库名, 例如当调入放大至 12pt 的字库 cmr10 时程序会显示:

```
Loading font cmr10 at 12pt                          Buffer=???K+???K
```

其中 "buffer=???" 显示的是用于装载字库数据的剩余内存空间。剩余内存空间由两部分给出, 第一部分给出常规内存中的剩余空间, 第二部分给出扩展、扩充内存中的剩余空间。

在调入汉字字库时驱动程序显示出正在调入的汉字字库名、已从该字库读入的字数和正在处理的汉字的区位码。例如, 假如屏幕上显示出:

```
Loading font CL256S0.PPS - xxxx (code=yyyy)         Buffer=???K+???K
```

则 "CL256S0.PPS" 为正在调入的字库名, "xxxx" 为已从该字库调入的字数, "yyyy" 为正在调入的汉字的区位码 (如果在调入汉字字库时出错, 则往往是由于用户的源文件中含有乱码, 此时可通过这里给出的区位码找到源文件中出错的位置)。

下面结合驱动程序中的菜单来介绍它们的参数。由于每个参数对应着一个命令行可选项, 我们在介绍参数的同时给出相应的命令行可选项的用法 (菜单中每项参数名后面的方括号中给出了对应的命令行可选项名)。

1. **DVI file name (适用于所有驱动程序)**

指待处理的 DVI 文件名。它对应着命令行上给出的文件名。

2. **Reduction factor (适用于所有驱动程序)**

   指显示或打印时的缩小因子。如 2 表示缩小一半, 0.5 表示放大一倍等等。如果该参数为 0, 则驱动程序将自动选取适当的缩小因子 (通常用于 `DVISCR.EXE`)。驱动程序通过插值运算将整个输出结果按该参数进行放大或缩小。主要用途有以下几方面:

   (a) 在屏幕上显示时, 可以选择一个适当的大于 1 的值以便观察排版结果的总体效果。

   (b) 通常提供给 24 针打印机使用的字库分辨率为 180 DPI, 如果打印机的分辨率不等于 180 DPI, 打印的结果将同标准的大小不一致。这时可以通过适当地调整本参数使得打印结果的大小标准化 (但打印质量会差一些)。如 Brother M1724 的分辨率为 160 DPI, 只需将本参数设为 $\frac{180}{160} = 1.125$ 便可获得标准大小的输出 (当然最好是直接使用 160 DPI 的字库)。

   (c) TEX 软件对不同分辨率的输出设备配置了不同密度的字库。常用的字库有 180 DPI (24 针打印机用) 和 300 DPI (激光打印机用)。有时用户为节省硬盘空间, 希望将同一密度的字库用于不同分辨率的输出设备, 此时需通过调整本参数来获得标准大小的输出。例如, 如果想要将 300 DPI 的激光打印机字库用于 180 DPI 的 24 针打印机, 则只需将程序 `DVI24P.EXE` 中的 "Font resolution" 参数设为 300, 同时将本参数值取为 $\frac{300}{180} \approx 1.6666667$。用户还可给出不同的横、纵向缩小因子, 两个数之间用冒号隔开。如 1.5:2 表示横向缩小 1.5 倍, 纵向缩小 2 倍。当只给出一个数时表示横、纵向取相同的缩小因子。

   缺省值为 1。对应的命令行可选项为 "`-Z`", 如:

   ```
   C>dviscr test -z1.125
   ```

3. **Font resolution (适用于所有驱动程序)**

   给出使用的 TEX 字库的分辨率 (以 DPI 即每英寸的点数为单位)。驱动程序也将其作为输出设备的分辨率。

   屏幕显示和 24 针打印机驱动程序设定的缺省值为 180 DPI, 而激光打印机设定的缺省值为 300 DPI。对应的命令行可选项为 "`-R`", 如:

   ```
   C>dvi24p test -R180
   ```

   与缩小因子类似, 用户可在横纵向上给出不同的分辨率, 两个分辨率之间用冒号隔开。

4. **Magnification (适用于所有驱动程序)**

   这项参数用于替代 TEX 或 CCT 源文件中由 `\magnification` 指令所给出的整体放大系数。例如将本参数取为 1200, 则相当于在源文件中给出了 `\magnification=1200`, 而不管源文件中的 `\magnification` 值为多少。如果本参数等于 0, 则表示保留源文件中设定的值。

   缺省值为 0。对应的命令行可选项为 "`-M`", 如:

   ```
   C>dviscr -m1440
   ```

5. **X margin:Y margin (适用于所有驱动程序)**

   给出应该留出的边限 ("X margin" 给出左右边限值, "Y margin" 给出上下边限值, 可使用 §2.3 (第 10 页) 中所介绍的所有合法单位, 两个边限值中间用冒号隔开, 它们必须是非负值)。驱动程序将整个 DVI 文件中的最大的 TEX 页面的左右分别加上 "X margin" 给

出的宽度, 上下分别加上 “Y margin” 给出的宽度作为有效的页面范围, 超出此范围的内容将丢失。当用户选择将几页内容打印在一页纸上 (参看 “Output format” 参数) 时还用它们来确定页间的距离。

对应的命令行可选项为 “-B”, 参数中的两个长度值用 “:” 隔开, 如:

```
C>dviscr -B1.5:2.54 sample
```

6. **X offset:Y offset (适用于所有打印程序)**

“X offset” 给出水平定位值 (页面右移的距离); “Y offset” 给出垂直定位值 (页面下移的距离), 中间用冒号隔开 (可使用 §2.3 (第 10 页) 中所介绍的所有合法单位, 并且可以取负值)。与 “X margin” 和 “Y margin” 参数不同的是, 它们总是相对于绝对纸张的方向, 而不管输出页面的方向如何。用户还可在 “X offset” 中使用关键字”left” (顶左), ”center” (居中) 和”right” (顶右) 以及在 “Y offset” 中使用”top” (顶上), ”center” (居中) 和”bottom” (顶下) 来告诉驱动程序自动计算水平或垂直定位值。

对应的命令行可选项为 “-W”, 如:

```
C>dviljp -wcenter:1cm
```

指示驱动程序将输出的页面在横向居中, 而在纵向下移 1cm。

7. **PK font path (适用于所有驱动程序)**

给出 PK 字库的路径名。所给出的路径中必须包含有字符串 “$d”, 当驱动程序构造一个 PK 字库的文件名时, $d 将被一个取决于字库分辨率及放大倍数的整数所替换。

缺省值为 CCT 所在目录 +\PIXEL\DPI$d。对应的命令行可选项为 “-P”, 如:

```
C>dviscr -pc:\tex\pixels\$ddpi
```

注: CCT 目前只支持 PK 类型, ID 为 89 的 TEX 字库, 当驱动程序寻找一个 PK 字库文件时, 它将“PK font path”中的 “$d” 用一个整数替换来做为字库文件所在的目录名, 假设这个整数为 n, 则 n 由下式给出:

$$n = (\text{Font resolution}) \times \frac{\text{Scale size}}{\text{Design size}} \times \frac{\text{Magnification}}{1000}$$

(舍入取整)。例如, 假定:

$$\text{PK font path} = \texttt{c:\TEX\PIXEL\DPI\$d}$$

$$\text{Font resolution} = \texttt{180 DPI}$$

$$\text{Magnification} = 1000$$

并假定用户使用了由下面 TEX 命令定义的字库:

```
\font\bigrm=cmr12 at 14.4pt
```

则 Design size=12pt, Scale size=14.4pt, 因此:

$$n = 180 * \frac{14.4}{12} * \frac{1000}{1000} = 216$$

即字库文件所在的目录为

```
\tex\pixel\dpi216
```

而字库文件的全名为

```
\tex\pixel\dpi216\cmr12.pk
```

注意: 说明 PK 字库路径的字符串中的 "$d" 必须使用小写字母 'd'。

8. **Chinese font path (适用于所有驱动程序)**

给出汉字字库及 CCFONTS.DEF 文件的路径名。驱动程序总是认为 CCFONTS.DEF 文件同它所定义的字库文件在同一个子目录中。用户可以同时将几套不同的汉字字库分别放在不同的子目录中, 并对每套字库配置一个 CCFONTS.DEF 文件, 而利用本参数来选用其中的一套字库 (例如在屏幕显示或 24 针打印机打印时用较低点阵的字库以提高处理速度, 而用激光打印机打印时用较高点阵的字库以保证打印质量)。

缺省值为 CCT 所在目录 +"\HZFONTS"。对应的命令行可选项为 "-I", 如:

```
C>dviscr -ic:\hzfnt24
```

9. **Range of page numbers (适用于所有驱动程序)**

选择输出的页号范围。每个页号范围或是由单独一个页号给出, 表示一页; 或是由用 ":" (冒号) 隔开的两个页号给出, 表示从第一个页号到第二个页号之间的所有页。用户可以给出一连串的页号范围, 每两个页号范围之间用 ";" 隔开。只有当一页处于某个给出的页号范围中时驱动程序才输出它 (当不给出任何页号范围时驱动程序将输出所有的页)。用户在一个页号范围中还可以省略冒号前面的页号 (表示从最小的页号开始) 或冒号后面的页号 (表示直到最大页号)。例如:

:3;5:6;8;10:

表示要打印第 3 页前的所有页 (包括第三页), 第 5, 6, 8 页和从第 10 页开始的所有页。这个参数主要在以下几种情况使用:

(a) 屏幕显示时用本参数让驱动程序只处理想要看的页, 可以提高处理速度 (可使驱动程序避免扫描不用看的页和从字库中调入只被这些页用到的字符)。

(b) 打印时可以选择只打印某些页。

(c) 由于激光打印机驱动程序 DVILJP.EXE 采用下载字库的方式 (down load), 而 HP LaserJet+ 激光打印机最多允许同时调入 32 个字库 (每个字库可含 191 个不同字符), 并且每页中最多只能用到 16 个不同的字库, 因此当一篇文章中用到太多的不同字符时 (超过 6112) 将无法一次输出。这时可分成几次打印, 使得每次打印时用到的不同字符数不超过 6112。

使用的页号可以是 TEX 的页号 (即 TEX 中的 "\count0"), 也可以是 DVI 文件中所有页的顺序编号 (即第一页页号为 1, 第二页页号为 2 等等, 而不管这些页的 TEX 页号等于多少)。用户可通过参数 "Page numbering" 来选择使用哪种页编号 (参看后面的说明)。

对应的命令行可选项为 "-N", 如:

```
C>dviscr -n1;3;5:8
```

10. **Parity of page numbers (适用于所有驱动程序)**

可用来选择只输出页号为奇数的页或页号为偶数的页。有下列三个选择:

Process all page numbers	输出时不管页号的奇偶。
Process odd page numbers	只输出页号为奇数的页。
Process even page numbers	只输出页号为偶数的页。

这里所使用的页号与前一项参数 (即 "Range of page numbers") 一样。

一个页号只有同时满足本参数与前一参数的要求才被输出。

缺省值为 "Process all page numbers"。对应的命令行可选项为 "`-U`", 后面跟随数字 0, 1 或 2, 0 表示 "Process all page numbers", 1 表示 "Process odd page numbers", 2 表示 "Process even page numbers"。如:

```
C>dviljp -u1
```

11. **Page numbering (适用于所有驱动程序)**

给出前两项参数中所使用的页号类型。有以下两个选择:

TEX page numbers	使用 TEX 页号, 即 TEX 中 `\count0` 的值。
Sequential numbering	使用每页在 DVI 文件中的顺序编号 (从 1 开始)。

缺省值为使用 TEX 的页号。对应的命令行可选项为 "`-V`", 后面跟随 0 (TEX page numbers) 或 1 (Sequential numbering)。如:

```
C>dvi24p -v1
```

12. **Scratch file name (适用于除 `DVILJP.EXE` 之外的所有驱动程序)**

驱动程序在输出之前需从字库文件中调入所有用到的字符。如果留给字库的内存空间容纳不下需调进的字库时, 驱动程序会自动在硬盘上建立一个临时文件来存放部分字库。本参数给出的是临时文件的文件名 (包括路径名)。如果用户拥有一个足够大的虚盘, 可选择将临时文件建在虚盘上来提高处理速度 (当然更好的办法是不建虚盘, 而将扩展或扩充内存直接留给驱动程序使用)。

缺省值为 `\$DVITEMP.$$$`。对应的命令行可选项为 "`-T`", 如:

```
C>dviscr -te:\tmp\$dvitemp.$$$
```

13. **Output device name (适用于所有打印程序)**

给出输出设备名。"`LPT1`", "`LPT2`" 及 "`LPT3`" 分别代表三个并行口, 其它情况则当做文件名处理, 即将打印输出写入一个文件中。如果选择了将输出存入文件, 则用户可在程序处理完毕后用下面的指令:

```
C>copy/b 打印输出文件名  prn:
```

来将结果送往打印机 (如此可将打印输出文件拷贝到软盘上拿到其它没有安装 CCT 系统的计算机上进行打印输出)。

缺省值为 "`LPT1`"。对应的命令行可选项为 "`-L`"。如:

```
C>dviljp test -ltest.hp
```

表示将打印输出写入文件 "`test.hp`" 中。

14. **Output format (适用于所有打印程序)**

这项参数给出打印在同一张纸上的页数及格式。参数格式为 "nxm", 其中 n 为横向页数, m 为纵向页数 (打印在一张纸上的总页数为 $n \times m$), n 和 m 之间用字母 "x" 隔开 (大小写均可)。各页间的间距可通过参数 "X margin" 和 "Y margin" 来控制 (参看下面的说明)。

缺省值为 "1x1"。对应的命令行可选项为 "`-F`", 后面跟随的参数和菜单中的格式一样, 如下例中在每张纸上横向打印两页的内容:

```
C>dviljp -f1x2 -o1
```

注: 当选择将多个 TEX 页打印在一页纸上时 (即 "Output format" 参数不等于 "`1x1`") 整个页面由多个有效页面排成一个方阵而构成 (中间不留空), 驱动程序在每个有效页面中

打印一个 TEX 页的内容。横向两页间的距离等于两倍的 "X margin", 纵向两页间的距离等于两倍的 "Y margin"。用户可通过增大或减小 "X margin" 或 "Y margin" 的值来改变页面间的距离。

15. **Start page no. (适用于 DVISCR.EXE)**

给出第一个显示的页号。如果页号不存在或不在显示的页号范围内则程序首先显示显示范围中的第一个页 (这里用的总是 TEX 页号)。

缺省值为显示第一个要显示的页。对应的命令行可选项为 "-#", 如:

```
C>dviscr -#5
```

16. **Reverse video (适用于 DVISCR.EXE)**

选择是否采用反色显示。"No" 代表不反色 (即黑底白字), "Yes" 代表反色 (白底黑字)。该选项仅用于设定初始状态, 显示过程中用户可用 <Ins> 键随时进行切换。

缺省值 "Yes"。对应的命令行可选项为 "-A" ("-A0" 选择 "No", "-A1" 选择 "Yes"), 如:

```
C>dviscr -A1
```

17. **Graphics mode (适用于 DVISCR.EXE)**

用来选择一种图形方式。DVISCR.EXE 所支持的图形方式及命令行可选项中应该给出的参数值由下表给出:

图形方式	分辨率	命令行参数值
CGA	640 × 200	CGA
CGE (Color 400)	640 × 400	CGE
单色 EGA	640 × 350	EGAM
彩色 EGA	640 × 350	EGA
PS/2 VGA	640 × 480	VGAM
VGA	640 × 480	VGA
Hercules	720×348	HERC
扩展 VGA 或 VESA 方式	***	***

缺省情况下程序会自动选择一种合适的图形方式。对应的命令行可选项为 "-G", 后面跟随一个由上表第三列给出的可选项值, 如:

```
C>dviscr -gvga
```

表示选择 VGA 方式。

上表中的最后一行是一种特殊的情况, 主要用来支持一些增强型的 VGA 卡 (如 VESA, TVGA 等)。它允许使用以下两种参数形式:

(a) -G 图形模式号: 横向分辨率: 纵向分辨率: 视屏缓冲区段址

用于支持一些扩展的 VGA 方式。用户在参数中给出图形模式号、显示分辨率及视屏缓冲区的段址。驱动程序根据这些参数来对屏幕进行控制。图形方式号和视屏缓冲区段址用十六进制数, 而横向分辨率和纵向分辨率用十进制数。例如, 为选择模

式号为 5BH 的 800×600 的显示可采用下列参数:

```
-g5b:800:600:a000
```

用户若想使用增强的 EGA 或 VGA 方式, 应该从图形卡的使用手册中查出图形模式号、分辨率及视屏缓冲区段址。参数最后一项 (视屏缓冲区段址) 等于 `0A000H` 时可被省略, 因此 "`-g5b:800:600:a000`" 与 "`-g5b:800:600`" 是等效的。通过该方法也可以选择标准的 EGA 或 VGA 模式。例如, 标准的彩色 EGA 模式的模式号为 10H, 分辨率为 640×350, 因此, 可选项 "`-gega`" 同 "`-g10:640:350`" 是等效的。

(b) `-GVESA`: *横向分辨率*: *纵向分辨率*

指示 `DVISCR.EXE` 使用 VESA 标准 BIOS 调用来设置图形方式, 程序根据用户所要求的分辨率自动选择适当的显示模式。如果所要求的显示分辨率无法满足或显示卡不支持 VESA BIOS 调用 (`DVISCR.EXE` 只支持单色和 16 色的显示模式), 则程序给出一个错误信息。常用的分辨率有 1024:768, 800:600, 640:480 和 1280:1024。使用高分辨率 (1280×1024, 1024×768) 时需注意显示器是否能达到 (使用一种显示器达不到的高分辨率有可能对显示器造成损害)。

18. **Gamma:Foreground:Background (适用于 `DVISCR.EXE`)**

本参数只对 EGA, VGA 及 VESA 模式有效。在 VGA 或 VESA 模式下, `DVISCR.EXE` 使用不同级别的辉度来改善显示质量。该参数实际上包含三个参数: "Gamma" 为浮点数, 用于控制显示时辉度变化曲线, 如果将 "Gamma" 设为 0, 则驱动程序将不使用辉度 (这在一些较慢的机器上可加快显示速度); "Foreground" 和 "Background" 为整数, 用于改变显示的前景色和底色, 它们通常用形如 "`0xRRGGBB`" 的十六进制数给出, 如`0x112233` 表示 R=`0x11`, G=`0x22`, B=`0x33` (色彩值必须在 0 到 63 之间)。给出该参数时, 也可只给 "Gamma" 的值如 "`-^0`"。

缺省值为 2.0:0x3f3f3f:0x000000。对应的命令行可选项为 "`-^`".

19. **Allow changeable zoom (适用于 `DVISCR.EXE`)**

该参数用于控制是否允许用户在显示过程中改变显示的比例因子。当其值为 "Yes" 时用户可在显示过程中随时用一些特殊键来进行放大或缩小 (参看 §3.2 (第 27 页))。在速度较慢的机器上建议将该参数设为 "No"。

缺省值为 "Yes"。命令行可选项为 "`-&`", 后面跟随 0 (代表 "No") 或 1 (代表 "Yes")。

20. **Number of copies (适用于 `DVILJP.EXE` 和 `DVIDJ500.EXE`)**

给出打印时的拷贝份数 (1–99)。

缺省值为 1。对应的命令行可选项为 "`-C`", 如:

```
C>dviljp -c3
```

需要指出的是该可选项对 HP DeskJet 系列的喷墨打印机不起作用。

21. **Output orientation (适用于 `DVILJP.EXE` 和 `DVIDJ500.EXE`)**

用以控制页面相对于纸张的方向。有下面两个选择:

Portrait	纵向输出
Landscape	横向输出

缺省值为 "Portrait"。命令行可选项为 "`-O`", 后跟 0 或 1 分别表示 "Portrait" 和 "Landscape", 如:

```
C>dviljp -o1
```

22. **Printing area (适用于所有打印程序)**

给出打印机的有效打印范围, 当用户选择了自动页面定位时 (参看 "X offset" 及 "Y offset" 参数) 驱动程序利用它计算页面位移量, 在某些情况下驱动程序也用它判断是否有打印内容丢失。这项参数中必须给出四个长度值, 中间用 ":" 隔开, 分别表示打印机的有效打印范围的最小、最大横坐标及最小、最大纵坐标 (可使用 §2.3 (第 10 页) 中所介绍的所有长度单位)。假设这四个长度分别为 $x_0 : x_1 : y_0 : y_1$, 则有效打印范围为矩形区域 $(x_0, x_1) \times (y_0, y_1)$。

缺省值取决于不同驱动程序。

对应的命令行可选项为 "-D", 如:

```
C>dviljp -d0:21cm:0:29.5cm
```

表示打印机的有效打印范围为 (0cm,21cm)×(0cm,29.5cm)。

23. **Stop between pages (适用于 DVI24P.EXE)**

当它为 "Yes" 时, 驱动程序在输出每一页之前会停下来等待用户敲入任意一键再开始输出。

命令行可选项为 "-S", 后跟数字 "0" 或 "1"。"0" 表示No, "1" 表示 Yes。缺省值为 No (即打印时不停)。

24. **Display Chinese characters (适用于 DVISCR.EXE)**

当它为 "No" 时, DVISCR.EXE将只显示西文、公式及标点符号, 而用黑块来代表汉字。主要为检查版面和数学公式而设, 可节省调入汉字库的时间。

命令行可选项为 "-H", 后跟数字 "0" 或 "1"。"0" 表示No, "1" 表示 Yes。缺省值为 Yes (即显示汉字)。

25. **Reflection mode (适用于 DVILJP.EXE 及 DVIDJ500.EXE)**

DVILJP.EXE 有 "On" 和 "Off" 两个选择, DVIDJ500.EXE 有 "Off", "Horizontal" 和 "Vertical" 三个选择。当设为 "On" 或 "Horizontal" 或 "Vertical" 时程序将翻转输出结果 (即镜象输出)。某些情况下用这样输出的结果制版印刷可以改善印刷质量。

命令行可选项为 "-Q"。"-Q0" 表示 "Off", "-Q1" 表示 "On" (DVIDJ500.EXE 中 "-Q1" 表示 "Horizontal", "-Q2" 表示 "Vertical"), 缺省值为 "Off"。

26. **PCL compression mode (适用于 DVIDJ500.EXE)**

DVIDJ500.EXE 采用全页面缓存的工作方式, 送往打印机的数据量往往比较大。用户通过此参数选择是否采用压缩数据格式来减少数据量。其参数为一个非负整数, 含义如下:

0: 不压缩

1: 压缩方式 1

2: 压缩方式 2

4: 压缩方式 3

以上选择还可进行迭加, 如 $6(= 2 + 4)$ 表示同时使用压缩方式 2 和 3。

命令行可选项为 "-J"。如 "-J2" 表示只使用压缩方式 2。缺省值为 -J7, 即同时使用压缩方式 1, 2, 3。

27. **Output buffer size (适用于 DVIDJ500.EXE)**

    定义输出页面缓冲区的大小, 用 64KB 的倍数给出, 最小值为 1 (64KB), 最大值为 4 (256KB)。通常该参数选得越大, 程序运行速度越快, 但若选得过大则有可能使得剩余的内存不够调入特别复杂的页面。

    命令行可选项为 "-K"。如 "-K2" 表示使用 128KB 的输出缓冲区。缺省值为 -K4。

28. **Output quality (适用于 DVIDJ500.EXE)**

    选择打印输出的质量。"final" 表示选择最终输出质量, "draft" 表示选择草稿输出质量 (只打印出一半点, 节省硒鼓或墨水)。

    命令行可选项为 "-Y"。"-y0" 选择最终输出, "-y1" 选择草稿输出。缺省值为-y0, 即选择最终输出质量。

此外, 还有几个命令行可选项不在菜单中出现, 它们是:

-@ 指示当对点阵字库进行放大时是否做光滑处理 (包括所有西文字库和扩展为 ".PK" 的中文字库)。"-@0" 表示不做光滑处理, "-@1" 表示一级光滑, "-@2" 表示二级光滑 (缺省值)。

-! 用于控制字符缓冲区大小, 后跟一个数给出缓冲区的长度 (字节数)。用户通常不用关心这个参数, 必要时程序会指示用户使用适当的参数值。

-E 后跟数字 "0" 或 "1", 缺省为 "-E0"。"-E1" 表示 "Batch mode", 即程序不在菜单中停留而直接进行输出。当用批命令方式同时处理多个 DVI 文件时可使用 "-E1" 可选项。

-X 后跟数字 "0" 或 "1", 用以控制扩展及扩充内存的使用 (扩展内存指地址在 1MB 以上的内存, 而扩充内存则指符合 EMS 规范的扩充内存)。-X0 表示不使用扩展及扩充内存, -X1 表示使用它们。-X1 后面还可用 ":xxxxxx" 的形式给出可用的扩展内存的起始地址 (必须大于或等于 100000H, 缺省值为 110000H), 其中 xxxxxx 为 16 进制数, 如 "-X1:200000" 表示只使用地址在 200000H 以后的扩展内存。缺省值为 "-X1".

参数 7 (PK font path), 8 (Chinese font path) 和 12 (Scratch file) 还可分别通过 DOS 环境变量 "ccpkpath", "cchzpath" 和 "ccwkpath" 来进行设置。如:

```
C>SET CCPKPATH=D:\TEX\PIXELS\PXL$d
```

告诉驱动程序将 "D:\TEX\PIXELS\PXL$d" 作为 PK 字库路径名的缺省值。

CCT 对驱动程序增加了初始化文件的设置。驱动程序开始运行时在自己所在的子目录中寻找一个与其同名, 以 ".INI" 为扩展名的 ASCII 文件 (如 DVISCR.INI, DVIDJ500.INI 等等, 我们称这些文件为初始化文件), 如果该文件存在, 则驱动程序将其中所定义的参数读入。初始化文件中包含一些合法的可选项, 可选项之间用空格、制表符或回车隔开, 文件长度不限。如果初始化文件中的一行以字符 "%" 开始, 则驱动程序将忽略整行内容, 因此用户还可以在初始化文件中加入自己的说明。下面是一个初始化文件的例子:

```
-r300 -w1cm:2cm
```

它将分辨率设为 300 DPI, 将水平定位值和垂直定位值 (即 "X offset" 和 "Y offset") 分别设为 1cm 和 2cm。

此外, 用户还可通过一个形为 "xxxxOPTS" 的环境变量来设置驱动程序的参数, 其中 "xxxx" 代表驱动程序的文件名 (如 DVISCROPTS, DVIDJ500OPTS 等)。用户可在该环境变量中给出任意数目的命令行可选项, 如:

```
C>SET DVISCROPTS=-gvesa:1280:1024 -r600
```

驱动程序按以下顺序对参数进行赋值:

1. 程序内部定义的初始值
2. 环境变量
3. 初始化文件 (如果该文件存在的话)
4. 命令行可选项
5. 菜单

如果一个参数被多次赋值, 则以最后一次的值为准。

§3.2 屏幕显示驱动程序 DVISCR.EXE

在使用本程序显示排版结果时, 必须选择一种用户的计算机所支持的图形方式才能在屏幕上显示出排版结果。

程序首先将所有字库调入内存, 然后将第一页的内容显示出来。显示过程中用户可使用下列控制键:

1. 上下左右四个箭头键用来在当前页中移动显示窗口以便能看到页面上各部分的内容。每按一次箭头键窗口向相应的方向移动半个屏幕的距离。
2. <Home> 键将显示窗口移至页面的左上角, <End> 键将显示窗口移至页面的右下角。
3. <PgUp> 用来调入前一页, <PgDn> 用来调入下一页。用户若想直接跳至页号在 0–255 之间的某一页 (指 TEX 页号), 可先按住 <Alt> 键, 接着在数字键盘上敲入页号, 然后再松开 <Alt> 键 (若给出的页号不存在或不在显示的页号范围内, 则计算机将停留在当前页上)。
4. <Ins> 键用来反转显示颜色。
5. 如果 “Allow changeable zoom” 为 “Yes”, 用户还可用 '0'–'9', '+'/'-', '<'/'>' 等键来改变显示的比例因子。
6. <Esc> 键用来终止程序的运行而返回 DOS。

§3.3 24 针打印机驱动程序 DVI24P.EXE

DVI24P.EXE 是一个通用 24 针打印机驱动程序。它在运行时自动调入一个扩展名为 .DRV 的驱动文件以驱动一种给定的打印机。CCT 提供下述驱动文件:

驱动文件名	适用的打印机
EPSONLQK.DRV	EPSON LQ–K 系列 (内含汉字库), 180×180 DPI
EPSONLQ.DRV	EPSON LQ 系列 (不含汉字库), 180×180 DPI
LQHIRES.DRV	EPSON LQ 系列, 360×360 DPI
BJ10E.DRV	Canon BJ-10e/300/330 (mode 2), 360×360 DPI
BROTHER.DRV	BROTHER M1724/2024, 160×160 DPI
M1570.DRV	Model 1570 S/SC, 180×180 DPI
OKI8320.DRV	OKI 8320C, 180×180 DPI
STAR2463.DRV	STAR AR–2463, 180×180 DPI
TOS1350.DRV	Toshiba P1350/1351, 180×180 DPI
TOS3070.DRV	Toshiba 3070, 180×180 DPI

用户应该将所需驱动文件放在与 DVI24P.EXE 同一子目录中, 通过修改文件 DVI24P.PAR 来选择打印机类型。DVI24P.PAR 是一个正文文件, 它也与 DVI24P.EXE 处于同一子目录。用户只须在其中给出相应于所用打印机的驱动文件名 (不含扩展, 如 "EPSONLQ", "BROTHER" 等等) 即可, 原始的 DVI24P.PAR 文件中给出的是 "EPSONLQK", 表示使用 EPSON LQ–K 系列打印机 (如 LQ 1600K 等)。

打印时, 驱动程序在传送数据给打印机的过程中如果发现出错 (通常是打印机没联上, 或是打印机没开, 或是打印机没在线 (On Line)), 程序会显示下列信息:

```
Printer error (status=???)! R-retry, A-abort ?
```

其中 "???" 代表中断 17H 返回的打印机状态码 (十进制数)。此时用户需检查打印机是否就绪, 然后按 "R" 键来继续打印, 或是按 "A" 键来终止打印。

下面我们介绍一下驱动文件的作用及结构，以便用户能自行开发新的驱动文件以适应一些特殊打印机。CCT 系统盘中包含有一些驱动文件的源程序 (在文件 DRVSRC.ZIP 中), 感兴趣的用户可阅读这些程序。

DVI24P.EXE 将所用到的打印机的控制命令分成 8 类。每当需要获得打印机的控制序列时，DVI24P.EXE 便调用驱动文件中的相应子程序，并提供给它所需控制命令的类别和必要的参数，而驱动文件中的子程序将打印机控制码返回给 DVI24P.EXE。驱动文件中包含一个数据区，一个主程序和 8 个子程序。数据区提供给 DVI24P.EXE 一些有关打印机的信息 (如分辨率、打印机名等等), 主程序根据 DVI24P.EXE 的要求调用相应的子程序, 而每个子程序则负责处理一类控制命令。我们将每个子程序所处理的内容称为一个功能 (function)。驱动文件中的主程序实际上是下面的 C 语言函数：

```
void far driver(int func_no,int arg1,int arg2,int arg3)
```

其中 func_no (功能号) 指驱动文件中的程序需要完成的功能编号，而 arg1, arg2 和 arg3 为调用参数。用户在设计自己的驱动文件时必须保留寄存器 DS, BP, SI 和 DI 的值。

驱动文件必须从一个段的位移 0 处开始，文件最开头必须是一条短跳转指令转移至主程序开始, 随后是数据区。以下我们简要介绍一下数据区的格式。若想了解更多的细节请参阅驱动文件的源程序。

数据区开头必须是字符串 "$DVI24P$" 用以标识驱动文件，随后用 18 个字节分别给出：

打印机横向分辨率，四字节整数，以 $2^{-16} \times$ DPI 为单位

打印机纵向分辨率，四字节整数，以 $2^{-16} \times$ DPI 为单位

缺省的横向缩小因子乘以 2^{16}，四字节整数

缺省的纵向缩小因子乘以 2^{16}，四字节整数

打印机一个空格的宽度 (点数), 双字节整数

以上的整数均按低位在前，高位在后的顺序，前四个参数被 DVI24P.EXE 做为菜单中相应参数的缺省值；而第 5 个参数使得 DVI24P.EXE 将每行打印数据中开头的空白转换成相应数目的空格字符，这样可以减少送往打印机的数据量。用户可将其设成 0 来使 DVI24P.EXE 不做这样的转换。

数据区最后是一个零结尾的字符串给出打印机的名称 (用来显示在 DVI24P.EXE 的菜单的标题中)。

为方便用户设置打印机, CCT 提供了一个程序 SETPRT.EXE, 它自动搜寻驱动文件并提示用户选择打印机的类型。

§3.4 激光打印机驱动程序 DVILJP.EXE

程序 DVILJP.EXE 只能用于驱动 HP LaserJet+ 或与之兼容的激光打印机 (包括 HP LaserJet II 以上的所有型号), 但不能驱动 HP 的喷墨打印机及一些老型号的激光打印机 (如 HP LaserJet), 对这些打印机需使用 DVIDJ500.EXE 进行驱动。DVILJP.EXE 在输出速度上要快于 DVIDJ500.EXE。另外, DVILJP.EXE 只能用于 300 DPI 以下的打印输出。

§3.5 喷墨打印机驱动程序 DVIDJ500.EXE

DVIDJ500.EXE 主要用于驱动 HP DeskJet 500 喷墨打印机。它也可用来驱动 HP 其它类型的喷墨及激光打印机, 但需通过 -J 可选项选择适当的压缩方式。当用户选择同时使用几种压缩方式时, 程序将分别对每行点阵数据选用压缩比最高的一种压缩方式。下表中给出了常见的 HP 喷墨及激光打印机所支持的数据压缩方式, 供用户参考:

打印机型号	所支持的压缩方式
LaserJet	无
LaserJet+	无
LaserJet II+	方式 2
DeskJet	方式 2
LaserJet III	方式 2 和 3
DeskJet 500	方式 2 和 3

注: 上表中列出的只有方式 2 和 3, 实际上为了节省处理时间, `DVIDJ500.EXE` 从不使用压缩方式 1, 因为它对主要由文字构成的页面的压缩效果远不如方式 2 和 3。用户可以试用不同的压缩方式并将结果存入一个文件中来比较各压缩方式之间的差别。

当使用 `-J0` 可选项时, `DVIDJ500.EXE` 可驱动所有 HP LaserJet 及 DeskJet 系列的打印机。通常对于一个特定的打印机, 用户可根据上表选择尽可能高的压缩方式以减少送往打印机的数据量。但若用户的主机速度较慢, 则有可能花费在数据压缩处理上的时间大于传输未压缩的数据所多需要的时间, 此时则应降低或减少所使用的压缩方式。

使用 `DVIDJ500.EXE` 还可以在 HP LaserJet IV 型打印机上获得 600 DPI 的打印输出。为此用户必须使用可选项 `-R600`。

§3.6 打印输出页面位置的控制

用户在使用 CCT 的打印驱动程序输出时, 经常遇到的问题之一是怎样控制输出页面的位置。这里解释一下影响输出页面位置的几个参数的作用, 以帮助用户利用它们进行页面定位。

我们称一个 DVI 文件中所有页中最宽页的宽度和最高页的高度所构成的矩形区域为 TEX 页面 (PLAIN TEX 中影响 TEX 页面大小的主要参数有`\hsize`, `\vsize`, `\hoffset`, `\voffset` 及页眉、页脚的高度, 而 LATEX 中影响 TEX 页面大小的主要参数有 `\textwidth`, `\textheight`, `\oddsidemargin`, `\evensidemargin` (如果使用了 `twoside` 选项), `\footheight`, `\footskip`, `\headheight`, `\headsep` 等)。驱动程序在 TEX 页面的左右分别加上由 "X margin" 参数给出的宽度, 上下分别加上由 "Y margin" 参数给出的宽度, 将所得到的区域作为输出页面的大小, 所有位于输出页面之外的内容将丢失并带来一个警告信息 (`DVILJP.EXE` 除外)。

从逻辑上说, 驱动程序在计算一页中的每个元素 (字符、图形等) 的位置时, 将 DVI 文件中所定义的横竖坐标值分别加上 "X margin" 和 "Y margin" 作为其在输出页面中的位置, 如果用户选择了横向输出 (即 Landscape), 则将该元素在输出页面中的位置再进行旋转 90^{o} 的变换。在生成整个输出页面后, 再将输出页面输出到打印纸上, 并使得输出页面的左上角位于打印纸的 (X offset,Y offset) 位置上。由此可知, 通过调整 "X margin" 和 "Y margin" 参数或 "X offset" 和 "Y offset" 参数都可以改变输出的页面位置, 但 "X margin" 和 "Y margin" 是相对于 TEX 页面而言, 因此在纵向输出时 "X margin" 影响页面的左右位置, "Y margin" 影响页面的上下位置, 在横向输出时 "Y margin" 影响页面的左右位置, "X margin" 影响页面的上下位置, 而 "X offset" 和 "Y offset" 是相对于打印纸而言, 因此 "X offset" 总是影响页面的左右位

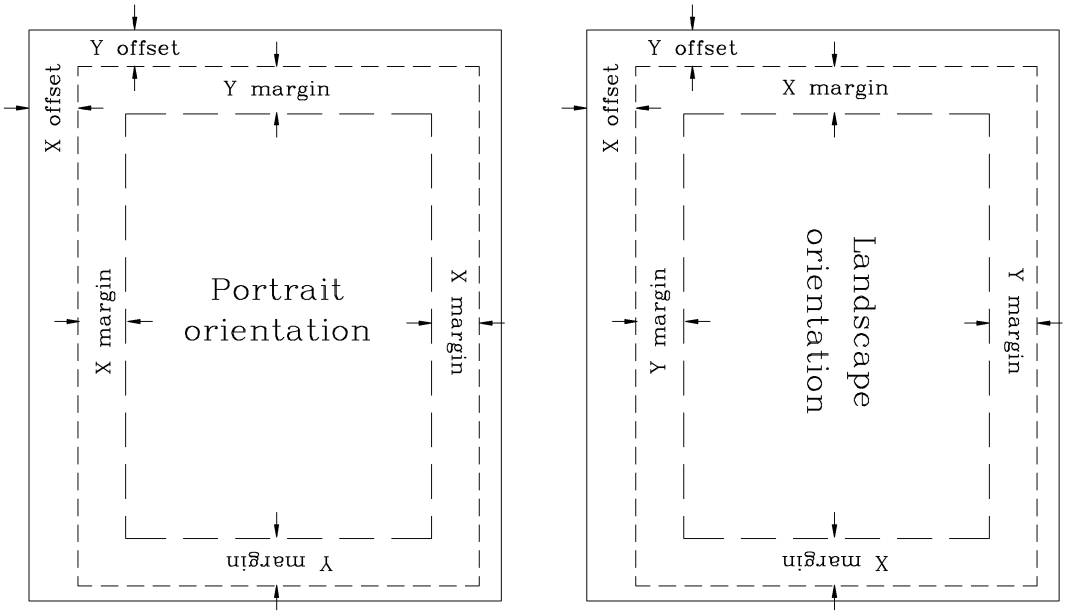

图 3.1: 输出页面的位置参数的作用

置, “Y offset” 总是影响页面的上下位置, 而不管是横向输出还是纵向输出。

“X offset” 和 “Y offset” 参数除可直接给出输出页面在打印纸上的位移量外, 还可通过一些关键字来让驱动程序自动进行定位。用户可以使用下面一些关键字 (大小写等效):

X offset:	left	页面顶左
	center	页面在左右方向居中
	right	页面顶右
Y offset:	top	页面顶上
	center	页面在上下方向居中
	bottom	页面顶下

当用户选择自动定位时, 驱动程序利用 “Printing area” 参数中的值来计算 “X offset” 和 “Y offset” 的值, 并且在开始打印前显示出实际算出的位移量。“Printing area” 参数应该对应于打印机的实际打印范围才能保证自动定位的准确性。

缺省情况下, 驱动程序将 “Printing area” 参数设为 A4 纸的大小 (210mm×297mm), 而将 “X offset” 和 “Y offset” 参数设为自动居中 (center)。

图 3.1 中描述了输出格式为 `1x1` 时各位置参数的作用。其中实线框代表打印机的物理打印范围, 大的虚线框代表输出页面, 而小的虚线框代表 TEX 页面(输出页面由 TEX 页面上下和左右分别加上宽度为“X margin” 和 “Y margin” 的边限得到)。由图中可看出 “X offset” 和 “Y offset” 总是相对于打印纸的方向而言, 它们所起的作用与 “Output orientation” 参数的设置无关, 而 “X margin” 和 “Y margin” 参数是相对于 TEX 页面方向而言, 它们所起的作用与 “Output orientation” 参数的设置有关。

第四章　CCT 的图形、图象接口

CCT 系统允许用户直接将图形、图象插入排版的文本中, 与文字部分一起输出, 做到图文并茂, 以提高出版物的质量, 简化排版过程。CCT 系统的图形图象处理功能主要基于设备驱动程序所提供的图象接口, 这个接口允许用户将符合一定格式的图象文件 (称为 BMF 文件) 插入到版面的任意位置上。用户只需设法将自己的图形转换为 BMF 文件即可实现图文的同时输出。以此接口为基础, CCT 提供下列图形处理及转换程序:

1. 绘图程序 PICTEX。提供给用户一套简单的绘图指令用来绘制平面曲线图。
2. 将 HP 绘图仪的绘图指令 (也叫做 GL 语言) 转换为 PICTEX 的图形文件 (PDF 文件) 的程序 `HPGL2CCT.EXE`。用来实现与一些常用绘图软件的接口 (如 AutoCAD, GS 等等)。
3. 将 TIFF, PCX, PCL 等文件格式转换为 BMF 文件的程序 `IMG2CCT.EXE`。可用来处理通过数字化扫描仪输入的图象。
4. 将 BMF 文件转换为 TIFF 格式的程序 `BMF2TIF.EXE`。
5. 将 BMF 文件转换为 PCX 格式的程序 `BMF2PCX.EXE`。

我们在 §4.1 中介绍驱动程序的图形接口的用法及 BMF 文件的结构。在 §4.2 中介绍 PICTEX 的使用。在 §4.3 中介绍如何从其它绘图软件的绘图仪或激光打印输出中获取图形数据。在 §4.4 中介绍与数字化扫描仪的接口。

§4.1　CCT 驱动程序的图形接口

TEX 软件提供了一条特殊指令 `\special`, 它以一个字符串作为参数, 其使用格式如下:

```
\special{字符串}
```

这条指令对排版的版面不会产生任何影响, 而只将相应的指令码和参数中的字符串拷贝到 DVI 文件的相应位置上。CCT 系统利用这条指令来告诉驱动程序应该在当前位置上插入一幅图形。指令格式如下:

```
\special{BMF=文件名}
```

其中的文件名给出一个符合规定格式的黑白图象文件的文件名, 我们称这种图象文件为“BMF 文件”。驱动程序在 DVI 文件中遇到这条指令时会自动将由相应的 BMF 文件所定义的图象加在版面的当前位置上。例如, 假如用户 BMF 文件名为 “`MYPIC.BMF`”, 则只需在排版的源文件中加入指令:

```
\special{BMF=MYPIC.BMF}
```

即可将图象加入到排版中去。

由于 TEX 在排版时对 `\special` 指令不做任何处理, 因此用户在利用它插入图象时应该根据图象的大小在周围留出足够放下图象的空间。驱动程序在处理 BMF 文件时以图象的左下角为原点, 即图象的左下角正好处在 `\special` 指令的位置上。例如, 假设用户想给前面的例子中的图象留出 6cm 宽, 8cm 高的空间, 则可采用下面的指令:

```
\vbox{\hsize 6 true cm\vskip 8 true cm\relax
  \hbox to\hsize{\hbox to0pt{\special{BMF=mypic.bmf}}\hfill}
}
```

当驱动程序在 DVI 文件中遇到指令 `\special{BMF=...}` 时, 如果找不到相应的 BMF 文件, 这条指令将被忽略并产生一个警告信息。

BMF 文件通常可由 CCT 配置的图形处理程序自动生成。这里给出 BMF 文件的结构以便用户能够自己将一些图形转换为 BMF 文件。

BMF 文件中给出的是黑白图象的点阵数据。文件开始是 16 个字节的文件头, 它们顺序给出:

1. 横向分辨率 (4 字节整数, 高位在前, 低位在后, 以 $2^{-16}\times$DPI 为单位)
2. 纵向分辨率 (4 字节整数, 高位在前, 低位在后, 以 $2^{-16}\times$DPI 为单位)
3. 图象宽度 (双字节整数, 高位在前, 低位在后, 以点数为单位)
4. 图象高度 (双字节整数, 高位在前, 低位在后, 以点数为单位)
5. 横向位移量 (双字节整数, 高位在前, 低位在后, 以点数为单位)
6. 纵向位移量 (双字节整数, 高位在前, 低位在后, 以点数为单位)

其中最后两个参数为带符号的整数, 取值范围为 −32768—32767。横向位移量表示应该将图象向左移动的点数, 纵向位移量表示应该将图象向上移动的点数。紧跟在文件头后面即是定义图象的点阵数据。图象中每行点从左到右每 8 个点分成一组 (最后不足 8 个点也作为一组), 每组点用一个字节表示, 字节中的每个位元代表一个点 (0= 白点, 1= 黑点)。8 个点中最左边的点对应字节中的最高位位元。将代表每行点的字节按从左到右的顺序排列, 并且按从上至下的顺序排出所有的行中的字节即得到构成图象的点阵数据。这种排列方式与通常的图象数据的排列是一致的。假如图象的宽度为 n 个点, 高度为 m 个点, 则每行数据的长度为 $(n+7)/8$ (取整), 整个数据长度为 $((n+7)/8)*m$, 加上文件开头的 16 个字节, BMF 文件的总长度应该等于 $((n+7)/8)*m+16$。

4.1 以后版本的 CCT 软件支持压缩格式的 BMF 文件, 可以在不影响处理效率的前提下大大节省用户的磁盘空间。`BMFPACK.EXE` 程序完成对老的 BMF 文件的压缩, 用法如下:

`C>BMFPACK` *文件名*

其中文件名中可以使用通配符 "`*`" 和 "`?`"。如果省略扩展名, `BMFPACK.EXE` 将自动加上扩展 "`.BMF`"。缺省情况是将 BMF 压缩。如果使用可选项 "`-U`", 则把压缩的 BMF 文件还原成非压缩格式。

§4.2 绘图软件 PICTEX

PICTEX 是一个利用 CCT 的图象接口及 TEX 的图形处理功能进行绘图的软件。它是一个多年前开发的功能比较简单的绘图程序, 名称上正好与目前一个较为流行的也叫 "PICTEX" 的 TEX 绘图宏库一样, 请用户不要混淆。本节中简单地介绍一下该程序的功能及使用。由于我们早已不再对该程序进行维护及更新, 并且将来的 CCT 版本中可能不再包含它, 因此我们建议用户尽量使用其它功能更强的绘图程序而不要依赖于它。

PICTEX 采用批处理的方式, 即用户通过一些特定的绘图指令来绘制图形。用户将绘图指令存在一个文本文件中 (称为 PICTEX 的数据文件), 然后利用 PICTEX 来产生一个 TEX 源文件 (某些情况下可能还需产生一个 BMF 文件)。用户只需将产生的 TEX 源文件插入到排版的源文件中即可在排版结果中得到所绘图形。

PICTEX 包括可执行文件`PICTEX.EXE` 和`PDFTOBMF.EXE`。

PICTEX 的使用包括以下几个步骤:

1. 准备一个数据文件。这个文件中包含 PICTEX 的绘图指令。
2. 运行 PICTEX 程序来产生一个 TEX 源文件, 此文件中包含产生图形的 TEX 指令。
3. 将产生的 TEX 文件插入到排版的文件中去 (也可单独编译产生的 TEX 文件并输出所绘图形)。
4. 如果图形中含有无法用 LATEX 的绘图指令画出的内容 (曲线) 时, PICTEX 还会生成一个叫做 PDF 的文件 (与产生的 TEX 源文件同名, 但扩展名为 ".PDF"), 并在所产生的 TEX 源文件的适当位置上插入 \special{BMF=···} 指令。用户在输出结果前需用程序 PDFTOBMF.EXE 将 PDF 文件转换为 BMF 文件才能得到这部分图形。

§4.2.1 数据文件的格式及绘图指令

数据文件由一串绘图指令构成。绘图指令之间用空格、逗号或换行隔开。绘图指令包括指令名和参数。指令名和参数之间用等号 ("=")、空格、逗号或换行隔开, 同一指令的不同参数之间用空格、逗号或换行隔开。指令名中可包含字母及字符 "_"、"*", 其中的字母用大小写均可。如: pic_size, Window 等等。指令的参数有三种类型: 数字、字符串和关键字。数字参数就是通常的实数或整数, 如1, 2.345, 5e-7; 字符串参数是指括在单引号中的一串字符, 如: 'abcde'; 如果字符串本身含有单引号, 则每个单引号在字符串中必须用连续两个单引号来表示, 如字符串: 'Let''s go' 代表 "Let's go"; 关键字参数指一些规定的关键字, 如: ON, OFF, REAL 等等。

PICTEX 的绘图指令有两大类。第一类用于定义绘图参数, 如图形的大小, 线条的粗细等等, 这类参数如果用户不给出, PICTEX 会使用为它们的设定的缺省值; 第二类指令用来进行实际绘图。下面给出所有绘图指令的格式及功能。

PIC_SIZE, PIC_SIZE*: 指令形式为: PIC_SIZE=w,h 或PIC_SIZE*=w,h

其中 w,h 是两个实数, 它们分别给出图形的宽和高 (以毫米为单位)。在缺省情况下绘图区域的宽为 110mm, 高为 70mm。PIC_SIZE和PIC_SIZE*的区别在于如果用的是PIC_SIZE, 实际输出的图形要略大一些, 以便留出一定的空间来打印坐标轴、坐标值等。实际输出的图形的高度等于 $h+30$, 宽度等于 $w+30$ (即在横竖方向各多出 3cm)。如果想让实际输出的图形的大小恰好等于绘图区域, 则需使用PIC_SIZE*指令。

WINDOW: 指令形式为: WINDOW=$x_{\min},x_{\max},y_{\min},y_{\max}$

这条指令用来定义用户窗口: $(x_{\min},y_{\min})$ 给出图形左下角的用户坐标, $(x_{\max},y_{\max})$ 给出图形右上角的用户坐标。缺省情况下 $x_{\min}$, $x_{\max}$ 分别等于 DATA 指令中所有横坐标的最小和最大值, $y_{\min}$, $y_{\max}$ 分别等于 DATA 指令中所有纵坐标的最小和最大值。其它指令中的坐标均以此为参照系。

ORIGIN: 指令形式为: ORIGIN=a,b

这条指令用来定义坐标轴的位置: a 给出 y 轴位置的横坐标, b 给出 x 轴位置的纵坐标。缺省情况下 $a=x_{\min}$, $b=y_{\min}$。

LINE, VECTOR: 指令LINE=x_1,y_1,x_2,y_2在图中画出一条从 (x_1,y_1) 到 (x_2,y_2) 的直线段。由于 TEX 只能画出有限种斜率的直线段, PICTEX 将选择其中最接近的一条。类似地, 指令VECTOR=x_1,y_1,x_2,y_2在图中画出一条从 (x_1,y_1) 到 (x_2,y_2) 的带箭头的线段。

STRING: 指令形式为: STRING=s

其中 s 为一个字符串。这条指令将 s 所代表的字符串拷贝到输出文件中。如果人们希望将字符串中对应一个坐标值或长度的某个数由用户坐标转换为 TEX 中的坐标再写入到输出文件, 则可在这个数前面加上 "%??", 其中 "??" 代表下面由两个字母构成的四个指令之一:

CX, CY, LX, LY

其中字母X,Y,C,L 用大小写都可以。C用来说明后面紧跟着的数对应一个坐标, L用来说明后面紧跟着的数对应一个长度, X用来说明对应的是横向 (即按横向的比例进行转换), 而Y用来说明对应的是纵向。"%??" 不会被写进输出文件。例如下面的指令:

```
STRING='\put(%cx0.5,%cy0.5){\circle{%lx0.1}}'
```

将画出一个以 (0.5,0.5) 为圆心, 以 0.1 为半径 (按 x 方向的长度单位) 的圆, 而:

```
STRING='\put(%cx0.5,%cy0.5){$\Omega$}'
```

将在坐标为 (0.5,0.5) 的位置上打印一个 "Ω"。

"%??" 和它们后面的数之间不能留有空格或其它任何字符。除了上述情形外字符串中不能含有%。

GRID: 指令形式为: GRID=n, m

它指示程序在图中画出一个 $(n+1)\times(m+1)$ 的网格。

GRAD: 指令形式为: GRAD=n, m

当 $n>0$ 时指示程序在 x 轴上打印 $n+1$ 个坐标值; 当 $n<0$ 时指示程序自动控制 x 轴上坐标值的打印; 而当 $n=0$ 时指示程序不打印 x 轴上的坐标值。类似地, 当 $m>0$ 时指示程序在 y 轴上打印 $m+1$ 个坐标值; 当 $m<0$ 时指示程序自动控制 y 轴上坐标值的打印; 而当 $m=0$ 时指示程序不打印 y 轴上的坐标值。

注意: 若选择了在 x 轴上打印整数值, 则 $x_{\max}-x_{\min}$ 应能被 n 整除。类似地, 若选择了在 y 轴上打印整数值, 则 $y_{\max}-y_{\min}$ 应能被 m 整除。否则打印的值可能不对。

AXIS_STATUS: 指令形式为: AXIS_STATUS=k_1, k_2

其中 k_1 和 k_2 必须是下列关键字之一:

OFF, REAL, INTEGER

k_1 对应于 x 轴, k_2 对应于 y 轴。OFF表示不画坐标轴, REAL表示在坐标轴上打印浮点数刻度, 而INTEGER表示在坐标轴上打印整数刻度。如: AXIS_STATUS=INTEGER, OFF表示不打印 y 轴, 在 x 轴上打印整数值。缺省值为:

AXIS_STATUS=REAL, REAL

AXIS_TYPE: 指令形式为: AXIS_TYPE=k_1, k_2

其中 k_1 和 k_2 必须是下列关键字之一:

NORMAL, LOG

k_1 对应于 x 轴, k_2 对应于 y 轴。NORMAL表示坐标轴为线性坐标 (普通坐标), LOG表示坐标轴为LOG坐标 (对数坐标), 即在打印坐标值时在坐标值为 a 的地方打印出 10^a 的值。缺省值为:

AXIS_TYPE=NORMAL,NORMAL

AXIS_NAME: 指令形式为: AXIS_NAME=s_1,s_2

其中 s_1 和 s_2 是两个字符串, 它们分别给出 x 轴和 y 轴的名称。如:

```
AXIS_NAME='$x$','$y$'
```

缺省值为AXIS_NAME=’’,’’。

DATA: 指令形式为: DATA=$n, x_1, y_1, \ldots, x_n, y_n$。

每个DATA指令给出一条曲线的数据。它后面的参数给出曲线上一串点的坐标值。n 是所给出的点数, 这些点的坐标由 $(x_1, y_1), \ldots, (x_n, y_n)$ 给出。绘制曲线所需要的其它一些数据由指令CURVE_TYPE, INTERPOLATION, MARK_SYMBOL 及 DASHED_CURVE 给出, 请参看后面的有关说明。

END: 这条指令没有参数。它表示数据文件到此结束。后面的内容将被忽略。

以下 4 条指令用来设定绘制曲线的一些参数。这些参数一但被设定, 则其值对跟随其后的所有DATA指令都有效, 直到它们被重新赋给新的值。

CURVE_TYPE: 指令形式为: CURVE_TYPE=k

其中 k 必须是下列关键字之一:

THIN, THICK, MARK

它用来选择曲线的粗细。当k =THICK时为粗线, 而当k =THIN时为细线。初始值 (即缺省值) 为:

CURVE_TYPE=THIN

若k =MARK, 则DATA指令不画出曲线, 而是在每个给出的点上画一个由MARK_SYMBOL指令给出的符号。

INTERPOLATION: 指令形式为: INTERPOLATION=k

其中 k 必须是下列关键字之一:

LINEAR, SPLINE

它用来选择绘制曲线时所用的插值类型。LINEAR=线性插值, SPLINE=三次样条插值。初始状态下 (即缺省情况下) 为:

INTERPOLATION=LINEAR

MARK_SYMBOL: 指令形式为: MARK_SYMBOL=s

其中 s 为一个字符串。PICTEX 在曲线上每隔一定的距离标上一个 s 所代表的 TEX 符号, 用以标识曲线。初始状态下 (缺省情况下) MARK_SYMBOL为一个空字符串。

DASHED_CURVE: 指令形式为: DASHED_CURVE=$n, l_1, \ldots, l_n$

其中 $n \geq 0$ 为整数 (当 $n = 0$ 时指令变成DASHED_CURVE=0), $l_1, \ldots, l_n$ 是 n 个长度值 (以毫米为单位), 它们可被用来得到各种不同的虚、实线。PICTEX 绘制的曲线由线段和空白交替构成, $l_1, l_3, \ldots$ 依次给出线段的长度, $l_2, l_4, \ldots$ 依次给出空白的长度。当 $l_1 + \ldots + l_n$ 小于整个曲线长度时则重复使用 $l_1, \ldots, l_n$。n 通常应该是偶数, 因为当 n 为奇数时DASHED_CURVE=$n, l_1, \ldots, l_n$ 等价于 DASHED_CURVE=$n-1, l_1+l_n, \ldots, l_{n-1}$。如果 $n = 0$ 或没有空白 ($n = 1$) 或所有空白的长度为 0 则得到一条实线。初始状态 (即缺省情况下) 为:

DASHED_CURVE=0

即曲线为实线。

若数据文件中的一行的第一个字符是 “%”, 则这一行的内容将被忽略。它可用来在数据文件里插入一些说明, 或是临时取消某些绘图指令。

PICTEX 还在产生的 TEX 源文件中定义了一条 TEX 命令 \centerbox。这条命令使得其参数的正中心为其参考点。在STRING指令中可用它帮助定位。例如:

```
STRING = '\put(%cx0.5,%cy0.5){\centerbox{$\Omega$}}'
```

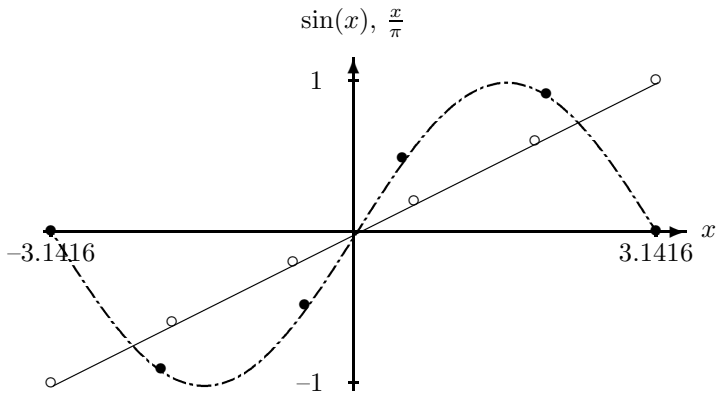

图 4.1: 用 PICTEX 绘制的正弦曲线

将使字符 “Ω” 的中心恰好位于 (0.5,0.5)。

下面给出的是一个数据文件的例子, 其结果见图 4.1。

```
pic_size=80,40
origin=0,0
axis_status=real,integer
axis_name='$x$','$\sin(x)$, $\frac{x}{\pi}$'
grad=1,1
%----- First curve ----------------------
curve_type=thick
interpolation=spline
mark_symbol='$\bullet$'
dashed_curve=4,2,0.5,0.5,0.5
data=21
  3.1415926536E+00,  0.0000000000E+00
  2.8274333882E+00,  3.0901699437E-01
  2.5132741229E+00,  5.8778525229E-01
  2.1991148575E+00,  8.0901699437E-01
  1.8849555922E+00,  9.5105651629E-01
  1.5707963268E+00,  9.9999999999E-01
  1.2566370614E+00,  9.5105651630E-01
  9.4247779608E-01,  8.0901699438E-01
  6.2831853072E-01,  5.8778525230E-01
  3.1415926536E-01,  3.0901699438E-01
  0.0000000000E+00,  0.0000000000E+00
 -3.1415926535E-01, -3.0901699436E-01
 -6.2831853071E-01, -5.8778525228E-01
 -9.4247779607E-01, -8.0901699437E-01
 -1.2566370614E+00, -9.5105651629E-01
```

```
 -1.5707963268E+00, -9.9999999999E-01
 -1.8849555921E+00, -9.5105651630E-01
 -2.1991148575E+00, -8.0901699438E-01
 -2.5132741229E+00, -5.8778525230E-01
 -2.8274333882E+00, -3.0901699438E-01
 -3.1415926536E+00,  3.6379788071E-12
%----- Second curve -----------------------
curve_type=thin
interpolation=linear
mark_symbol='$\circ$'
dashed_curve=0
data=2,-3.1416,-1.0,3.1416,1.0
end
```

§4.2.2 用 PICTEX 处理数据文件

准备好绘图的数据文件后, 首先应该用程序 PICTEX.EXE 对其进行处理, 将它转换为 TEX 源文件。PICTEX.EXE 的调用格式如下:

```
C>pictex 文件名
```

或:

```
C>pictex
```

在后一种情况下, 程序会要求用户键入一个文件名。这里的文件名即是待处理的数据文件名。如果省略文件名中的扩展名, 程序会加上 ".DAT" 作为扩展名。用户可在文件名中使用全局性字符 (即 "?" 和 "*"), 程序会自动寻找与之匹配的所有文件并逐个进行处理。运行完毕后对每个数据文件产生两个输出文件, 这两个文件的文件名与数据文件相同, 扩展名分别为 ".TEX" 和 ".PDF"。

第一个文件是 TEX 源文件, 它包含除用DATA 指令给出的曲线之外的所有其它图形元素 (包括坐标轴、坐标值等), 这部分图形元素均利用 LATEX 的 picture 环境中的指令直接画出, 因此将它们直接放入产生的 TEX 源文件中。用户只需将产生的 TEX 源文件插入到自己的 CCT 或 TEX 源文件中即可得到相应图形。

第二个文件叫做 PDF 文件, 其中包含数据文件中用 DATA 指令给出的曲线的数据。由于无法用 TEX 直接画出任意曲线, PICTEX.EXE 处理时将这些曲线的数据按一定格式存在 PDF 文件中, 并在产生的 TEX 源文件中加进指令 "\special{BMF=BMF 文件名 }", 其中的 BMF 文件名由数据文件的文件名加上扩展名 ".BMF" 构成。这样用户只需在显示或打印结果之前利用程序 PDFTOBMF.EXE 将 PDF 文件转换为 BMF 文件, 便可利用 CCT 的设备驱动程序的图形接口将这部分图形迭加到前一部分图形上去。

如果数据文件中不包含 DATA 指令, 则不产生 PDF 文件。

按上述方式产生的 TEX 源文件只包含绘图指令, 因此不能直接用 TEX 进行编译, 而只能被插入到其它的 TEX 源文件中一起处理。用户有时希望能够单独编译一个图形以便于修改加工, 为此 PICTEX.EXE 提供了一个命令行可选项 "-H", 当给出此可选项时程序会在所产生的 TEX 源文件的开头和结尾加入一些必要的指令使其构成一个完整的可直接用 TEX 编译的源文件 (但这样得到的源文件不能被直接插入其它 TEX 源文件)。

§4.2.3 将 PDF 文件转换为 BMF 文件

前面已经提到, 必须用程序 PDFTOBMF.EXE 将 PDF 文件转换成 BMF 文件, 才能在显示或打印时得到图形中的曲线部分。PDFTOBMF.EXE 的用法与 PICTEX.EXE 类似, 调用格式为:

```
C>pdftobmf 文件名
```

或:

```
C>pdftobmf
```

在后一种情况下, 程序会要求用户键入一个文件名。这里的文件名即是待转换的 PDF 文件名, 如果省略文件名中的扩展名, 则程序会加上 "`.PDF`" 作为扩展名。用户可在文件名中使用全局性字符 (即 "`?`" 和 "`*`"), 程序会自动寻找与之匹配的所有文件并逐个进行处理。转换产生的 BMF 文件名由 PDF 文件名加上扩展 "`.BMF`" 构成。

另外, 用户还可用可选项 "`-R`" 来给出所产生的图形的分辨率，以每英寸的点数为单位 (DPI)。给出的分辨率最好等于输出设备的分辨率, 否则驱动程序在输出时将进行插值运算从而降低图形的质量和处理速度。如果用户没有使用此可选项, 则程序开始运行前会询问:

Resolution in number of dots per inch (180) ?

即让用户键入分辨率。当分辨率等于 180 DPI 时可以直接敲入回车键 (程序设定的缺省值), 否则需先输入分辨率再敲入回车键。

当使用不同分辨率的输出设备时, 最好按输出设备的分辨率重新生成 BMF 文件, 以保证图形的质量。

使用 "`-R`" 可选项或回答询问时还可用 "`xxx:yyy`" 的形式给出横纵向上不同的分辨率。如 "`-R240:216`" 表示横向分辨率为 240 DPI, 纵向分辨率为 216 DPI。

§4.3 HP 绘图仪格式转换程序 HPGL2CCT.EXE

HPGL2CCT.EXE 将包含 HP 绘图仪命令的文件转换成 PDF 文件和 TEX 文件 (关于 PDF 文件请参看 §4.2 (第 33 页))。由于大部分绘图软件均支持 HP 绘图仪, 此程序可做为一个通用的图形接口程序。用户只需将绘图软件 (如 AutoCAD) 所产生的 HP 绘图仪命令存入一个文件即可插入 CCT 的排版中。其命令格式如下:

```
C>HPGL2CCT [ 可选项 ]  输入文件名 [ 可选项 ]
```

其中输入文件即为包含 HP 绘图命令的文件。程序运行时首先扫描输入文件, 然后便进入图 4.2 所示的菜单以便用户可以修改一些参数或在屏幕上显示图形。

为在菜单中修改某项参数或执行某条命令, 用户只需用箭头键将游标移到相应的位置上然后键入回车即可。用户还可按每项中用不同颜色显示的字符键来直接跳至某一项。下面解释一下菜单中每项的含义。

1. Current picture no.

   此参数当输入文件中包含数幅图形时用来选择处理第几幅图。对 HPGL2CCT.EXE 而言此参数不起作用 (因为一个输入文件中只能有一幅图形)。它主要为其它一些接口程序而设 (如用于处理 NCAR 中间绘图文件的程序 NCAR2CCT.EXE)。

2. Output file name

   给出输出文件名。其中不能含有扩展名。HPGL2CCT.EXE 将其加上 "`.PDF`" 做为 PDF 文件名, 而加上 "`.TEX`" 做为 TEX 文件名。缺省值为输入文件名去掉扩展。

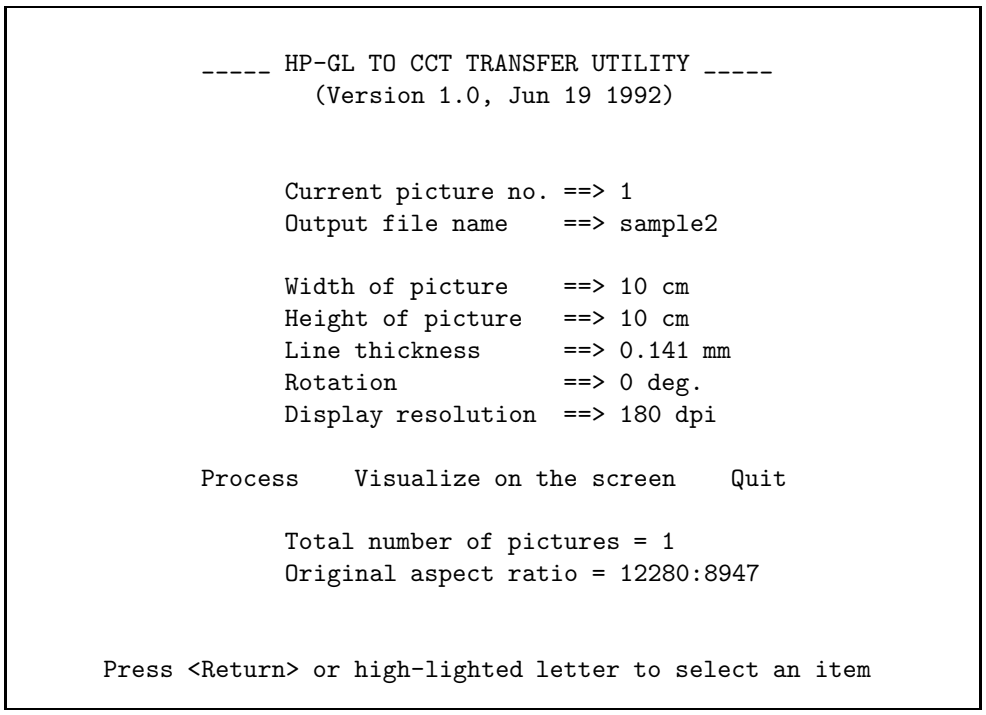

图 4.2: `HPGL2CCT.EXE` 的菜单

3. Width of Picture

   给出输出图形的宽度, 缺省值为 10cm。

4. Height of picture

   给出输出图形的高度, 缺省值为 0cm (此时程序将根据图形的宽度及高宽比来计算图形的高度)。

5. Line thickness

   给出输出图形中线条的宽度 (粗细), 以毫米为单位, 缺省值为 0.141mm。

6. Rotation

   选择将图形旋转 0°, 90°, 180° 或 270°。当用命令行可选项给出此参数时在 "`-R`" 后面跟随 0, 1, 2 或 3, "`-R`n" 表示旋转 $n \times 90°$。缺省值为 0°。

7. Display resolution

   给出屏幕显示时 (即 "Visualize on the screen") 所假定的分辨率。通过修改它可在不改变图形宽度和高度的情况下改变屏幕上所显示图形的大小。此参数不影响输出的 PDF 和 TEX 文件。缺省值为 180 DPI。用户也可用 "`xxx:yyy`" 的形式在横竖方向给出不同的分辨率, 如 "`180:160`" 表示横向分辨率为 180 DPI, 纵向分辨率为 160 DPI。

8. Process

   选择好各项参数后即可执行此命令指示 `HPGL2CCT.EXE` 开始进行转换并输出 PDF 和 TEX 文件。

9. Visualize on the screen

   在屏幕上显示所处理的图形以帮助用户选择参数。

10. Quit

退出程序返回 DOS。

菜单中最下面三行告诉用户一些有用的信息。“Total number of pictures” 给出输入文件中图形的个数, 在这里总是 1。“Original aspect ratio” 给出输入图形的宽度与高度之比。用户若想使输出图形保持原有的宽高比例则应选择 “Width of picture” 与 “Height of picture” 使其符合此比例。如果将”Width of Picture” 和”Height of Picture” 两个参数中的一个选成 0, 则程序会自动根据输入图形的高宽比及另一个参数值来计算它的值。例如假设输入图形的宽高比为 4:3, Width=0cm, Height=6cm, 则程序将计算出 Width=$6 \times 4/3$cm = 8cm。

上述参数值均可用相应的命令行可选项来设定初值。可选项的名字就是菜单中用不同颜色显示的字符, 后面跟随参数值即可。如 “-W15.3” 将 “Width of picture” 置成 15.3cm, “-R2” 将 “Rotation” 置成 180°。除菜单中的参数外还有下列可选项:

-G 设置显示图形时采用的屏幕图形方式, 与 DVISCR.EXE 中的 “-G” 的参数相同。

-B 后面不跟参数。用来选择批处理方式 (batch mode)。使用此可选项后程序将不再进入菜单而直接开始运行。主要用于批命令中。用此可选项时用户应该同时用其它可选项来设置所有的参数 (除非使用缺省值的参数)。我们建议用户在调整好一幅图形的参数后马上建立一个批命令文件并在其中用命令行可选项给出所有参数, 这样下次再需转换这幅图时只须运行批命令文件即可而不必记住所用的参数。

关于如何将所产生的 PDF 和 TEX 文件插入 CCT 的排版中请参看 §4.2 (第 33 页)。

最后需要说明的是 HPGL2CCT.EXE 并不处理 HP 的所有绘图仪命令而只支持其中一些最基本也是最常用的命令。

§4.4 图象文件接口程序 IMG2CCT.EXE

在排版过程中, 有时需要将一些图象、照片插在正文中。为此, CCT 提供了一个接口程序 IMG2CCT.EXE, 它可以把 TIFF, PCX 等格式的图象文件转换成 BMF 文件 (参看 §4.1 (第 32 页)), 这样便可利用 CCT 的设备驱动程序的图象插入功能将扫描图象等直接插入到正文中。

接口程序 IMG2CCT.EXE 尚处于不断变动中, 这里所介绍的用法可能会随时间而改变 (当用户不给出任何参数时, 程序将自动显示所支持的图象格式及参数形式)。该程序目前的用法如下:

```
C>IMG2CCT [可选项] 输入文件名 [可选项]
```

如果不使用 “-T” 选项指明输入文件类型, 则输入文件名中必须包含 “.TIF”, “.IMG”, “.PCX”, “.BMP” 或 “.PCL” 中的一个做为扩展名, 程序将根据扩展名来选择输入文件的类型。当使用了 “-T” 选项指明输入文件的类型时, 输入文件的扩展名可以省略, 此时程序根据输入文件的类型自动加上 “.TIF” (如果输入文件为 TIFF 格式), “.PCX” (如果输入文件为 PCX 格式), “.IMG” (如果输入文件为 Star 400 手持扫描仪的 “B” 格式文件), “.BMP” (如果输入文件为 BMP 格式) 或 “.PCL” (如果输入文件为 HP PCL5 光栅图象) 作为输入文件的扩展名。程序运行完毕后产生一个与输入文件同名, 但扩展名为 “.BMF” 的输出文件。同时, 为了便于用户将图象插入正文, IMG2CCT.EXE 程序还自动生成一个与输入文件同名但扩展名为 “.TEX” 的文件, 其中包含了插入图象的 TEX 指令 (程序会自动计算正文中应该留出的空间)。这样用户只需将生成的 TEX 文件插入到用户源文件中的适当位置上, 即可将图象插入正文。

IMG2CCT.EXE 允许用户使用下列可选项:

`-N` 反转原图象中的黑白颜色 (即将原来图象中的白点变为黑点, 黑点变为白点)。

`-C` 用来选择是否对输入图象进行剪裁 (即删去图象四周的空白)。“`-C0`” 表示不剪裁, “`-C1`” 表示剪裁。缺省值为 “`-C1`”。该选项当输入文件类型为 PCL5 时不起作用。

`-R` 后面跟随一个正数给出 BMF 文件的 DPI 值 (即图象的分辨率), 缺省时程序将 TIFF 文件中的 DPI 值直接拷入 BMF 文件 (如果用户给出一个不同于 TIFF 文件中的 DPI 值, 程序并不对图象进行插值转换, 即图象点阵的大小保持不变, 这意味着输出图象的尺寸将会改变)。与 `PDFTOBMF.EXE` 一样, 用户还可用 “`-Rxxx:yyy`” 的形式在横向和纵向给出不同的分辨率。

`-T` 用来指示输入文件的格式。“`-T0`” 表示输入文件为 TIFF 格式, “`-T1`” 表示输入文件为 STAR–400 手持扫描仪所产生的 “B” 格式 (此时缺省分辨率设为 300 DPI), “`-T2`” 表示输入文件 PCX 格式, “`-T3`” 表示 BMP 文件, “`-T4`” 表示 PCL5 格式。

`-X` 只生成 BMF 文件而不产生 TEX 文件。

这些可选项可放在文件名前, 也可放在文件名后, 但必须用空格与文件名隔开。

目前 `IMG2CCT.EXE` 只能处理包含单个非压缩黑白图象的文件。当输入文件中包含彩色图象、灰度图象或采用了压缩格式时需先用其它软件将所需图象转换成黑白图象并采用非压缩格式存贮。

HP PCL5 文件指 HP 激光或喷墨打印机的打印文件。`IMG2CCT.EXE` 只对其中包含的光栅图象进行转换, 可处理采用方式 0, 1, 2 和 3 压缩的光栅图象。当输入文件中包含多页时只对第一页的内容进行转换。当页面中包含多个光栅图象时, `IMG2CCT.EXE` 会根据它们的位置将其合并成一个图象 (当页面中包含有字符、矢量等非光栅图象命令时, 各个光栅图象间的相对位置可能不对)。

CCT 系统中还提供程序 `BMF2TIF.EXE` 和 `BMF2PCX.EXE`, 可将 BMF 文件转换为 TIFF 或 PCX 文件, 用法为:

`C>BMF2TIF` BMF *文件名*

`C>BMF2PCX` BMF *文件名*

其中 BMF 文件名中可以省略扩展名, 程序自动将输入文件名加上扩展 “`.TIF`” 或 “`.PCX`” 做为输出文件名。

第五章　CCT 的用户造字程序

CCT 配置的用户拼字程序名为 PZ.EXE。它允许用户利用标准字库中现有的字的偏旁、笔划等来拼造出新的字，并可对所造的字进行各种修改 (包括精修、缩放、移动、旋转、翻转等等)。用户利用它来建造用户字库，以在排版中使用国标一、二级字库中所没有的字。有关如何在 CCT 中使用用户字库请参看 §2.2 (第 9 页)。这里介绍如何用 PZ.EXE 来建立用户字库。

CCT 的用户造字系统包括下述文件:

PZ.EXE	拼字程序
PZFONT16.DAT	16 点阵字库用于 PZ.EXE 中的莱单显示
PZPYBIAO.DAT	包含一级汉字的拼音表

这些文件必须位于同一子目录中。PZ.EXE 运行时会自动将另外两个文件调入内存。

PZ.EXE 所造汉字为矢量结构。每个字由一些封闭的曲线围成。矢量字库的优点在于操作简单，拼字方便，放大后不会有毛刺。用 PZ.EXE 造的字可同时满足屏幕显示、针打、激光打印和激光照排等不同分辨率的要求，其字形质量完全能达到正规出版物的标准，并且字体及风格与标准字库中完全一致。另外, PZ.EXE 采用三次 Bézier 曲线描述字型, 具有字形光滑、笔划丰满、修改方便等特点, 并且结构上与 PostScript 的 Type 1 字库完全一致。

注: CCT 4.2 以前的版本配置的造字程序为 ZZ.EXE, 其字库格式与 PZ.EXE 不兼容, 为此我们提供了一个程序 ZZ2PZ.EXE 用于将 ZZ.EXE 所造字库转化为 PZ.EXE 的格式, 用法如下:

C>ZZ2PZ 输入字库文件名　输出字库文件名

其中输入字库为 ZZ.EXE 格式的字库, 而输出字库为 PZ.EXE 格式的字库。

PZ.EXE 还识别环境变量 CCHZPATH 并将它所给出的路径做为 CCT 汉字库的缺省路径 (选取参考字时用)。

§5.1　Bézier 曲线及曲线矢量字库简介

PZ.EXE 采用折线及三次 Bézier 曲线来描述围线。采用三次 Bézier 曲线的理由是它们具有很好的几何直观性, 容易控制, 处理简单, 并且与 PostScript 语言中所采用的线型兼容。由于采用了曲线描述, 所造字型能够适用于输出特别大的字符或用于特别高分辨率的输出设备上而不失真。

Bézier 曲线最初由法国雷诺汽车公司工程师 Bézier 发明并用于汽车外型设计上, 它在几何造型、计算机图形学、计算机辅助设计等领域有着广范应用。以下我们对 Bézier 曲线的定义及基本性质作一简单介绍, 感兴趣的读者可参考有关的专著 (如孙家昶所著的《样条函数与计算几何》, 科学出版社 1982 年出版)。

一条平面三次 Bézier 曲线由平面上的四个点 (称为 Bézier 曲线的控制顶点) 所确定, 这四个点按顺序构成的四边形称为它的特征多边形。图 5.1 中给出了两条三次 Bézier 曲线, 它们的四个控制顶点分别为 $\vec{p}_0$, $\vec{p}_1$, $\vec{p}_2$ 和 $\vec{p}_3$。

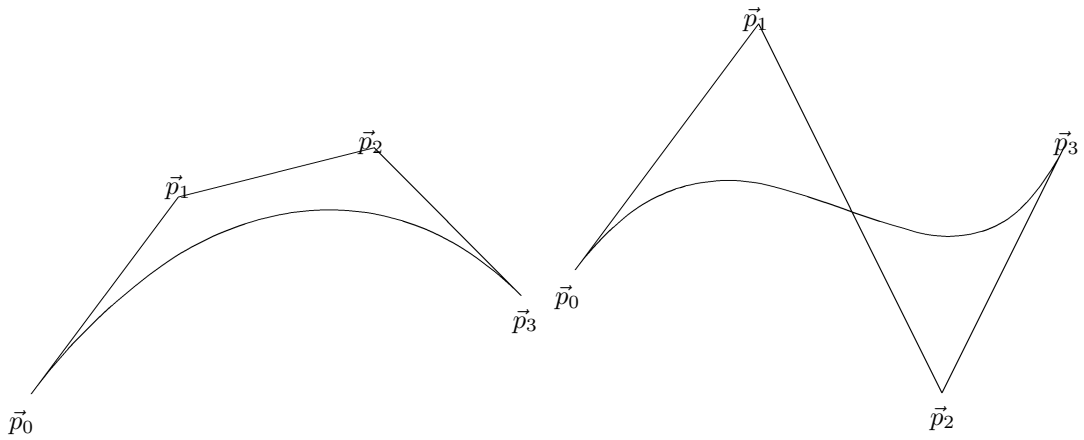

图 5.1: Bézier 曲线

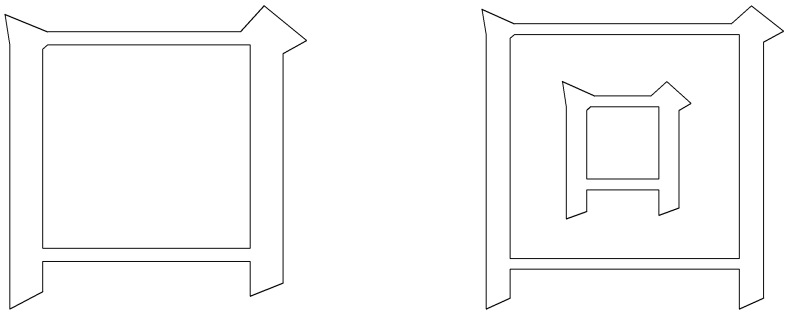

图 5.2: 围线矢量汉字

假设 Bézier 曲线的四个控制顶点为 $\vec{p}_0, \vec{p}_1, \vec{p}_2$ 和 $\vec{p}_3$, 则它的参数方程为:

$$\vec{B}(t) = (x(t), y(t)) = \sum_{k=0}^{3} C_3^k t^k (1-t)^{3-k} \vec{p}_k, \quad 0 \le t \le 1 \tag{1}$$

其中 $C_n^k = \dfrac{n!}{(n-k)! \cdot k!}$。容易看出 $\vec{B}(0) = \vec{p}_0$, $\vec{B}(1) = \vec{p}_3$, 当 t 从 0 连续变化到 1 时, $\vec{B}(t)$ 所画出的轨迹即是对应的 Bézier 曲线。因此, Bézier 曲线起始于它的第一个控制点, 终止于最后一个控制点, 但不经过中间的控制点。

下面几条有关 Bézier 曲线的基本性质对于熟练地掌握拼字程序的使用有一定帮助:

1. 一条三次 Bézier 曲线由它的四个控制顶点所唯一确定, 因此要想修改它只需修改它的控制顶点。
2. Bézier 曲线总是包含在它的特征多边形所限定的区域内。
3. 三次 Bézier 曲线在起点处与 $\vec{p}_0$ 到 $\vec{p}_1$ 的连线相切, 在终点处与 $\vec{p}_2$ 到 $\vec{p}_3$ 的连线相切。这条性质使得我们可以容易地将多条 Bézier 曲线光滑地连接起来。

`PZ.EXE` 采用围线矢量结构, 每个汉字用它的边界线 (称为围线) 来描述。图 5.2 中给出了两个字例 (它们也叫做“空心字”)。 其中“口”字由两条围线构成,“回”字由四条围线构成。为描述一个汉字, 只要给出它的所有围线。当复原成点阵字时，程序会自动将笔划内部填充。

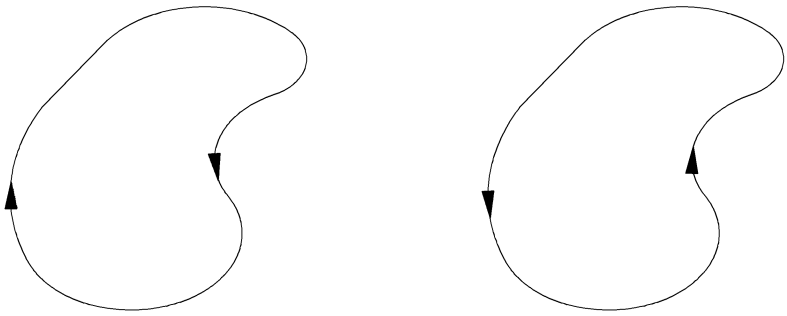

图 5.3: 围线的方向

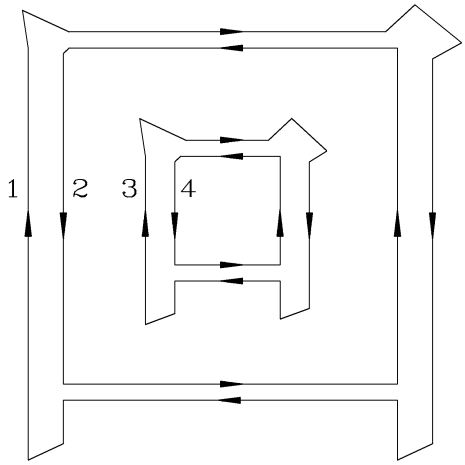

图 5.4: 围线的方向

一条围线用一串线段来描述, 其中每个线段或者是一条直线, 或者是一条三次 Bézier 曲线, 它们首尾相接, 并且最后一条线段的终点与第一条线段的起点相连, 形成一条封闭曲线。我们要求每条围线自身不相交，并且任意两条围线也互不相交。

每条围线将平面分割成两部分, 一个称为围线的内部 (有界部分), 另一个称为围线的外部 (无界部分)。我们称一条围线为顺时针方向如果沿围线行走时围线的内部总在右侧，反之则称其为逆时针方向 (图 5.3)。

一个汉字通常由数条围线共同围成，这些围线又可被分成内线和外线。当沿围线的顺时针方向绕一条围线行走时，如果汉字笔划的内部总在右侧，则称这条围线为外线，反之则称其为内线。如图 5.4 中的围线 1, 3 为外线，而围线 2, 4 则为内线。

用户造字时，应该使内线的走向为逆时针方向而外线的走向为顺时针方向。这样沿围线行进时笔划的内部总在右侧, 如图 5.4 所示。CCT 的矢量字库复原程序对围线的方向没有特殊要求，但用户在造字、修字过程中可能会需要将两条围线合并成一条围线，此时如果这两条围线的走向不符合上述要求，则连起来后的围线可能出现扭曲的情形 (即围线自身相交)。CCT 的矢量字库中的所有围线均符合上述走向要求，因此用户造字时最好也与其保持一致。

§5.2 命令行格式及屏幕布局

PZ.EXE 的命令行格式为：

C>PZ [可选项] 字库名 [可选项]

其中字库名为用户字库文件名。如果省略字库名中的扩展名，PZ.EXE 自动加上 ".PZ" 做为扩展名。允许的选项主要有 "-g", 用于选择图形方式, 具体参数及用法请参看程序中的说明。

如果用户给出的字库文件不存在，则 PZ.EXE 会询问是否建立一个新库, 用户需回答 "Y" 来确认。接着 PZ.EXE 还会询问所建字库的记录长度, 字符宽度和高度。用户所建造的字库文件中采用定长的记录来存贮每个汉字，记录长度即给出每个记录的长度，也就是每个所造字的数据的最大长度。如果选得过小，则当所造字的数据长度超出时无法将其存盘，选得过大则造成不必要的浪费。PZ.EXE 所建议的缺省值为 2048，即每个字在字库文件中占用 2K 字节 (通常一个字的数据量不会超过 1000 个字节)。字符的高度和宽度指所造汉字的分辨率，用户最好取 CCT 配置的标准字库的宽度和高度的倍数做为用户字库的高度和宽度。

启动程序 PZ.EXE 后屏幕显示分成以下几部分:

- 屏幕左下角显示的是当前正编辑的字在库中的编号 ("字号 = ?") 及字库中的字号范围 "(?--?)";
- 屏幕左侧显示的是字符编辑命令;
- 屏幕上部显示的是主菜单, 也用于显示提示信息 (用户在使用过程中需随时注意该处所显示的信息);
- 屏幕右侧为编辑窗口。

每个字由数条围线构成。有一个闪烁的十字游标标明当前所在的围线及节点。用户可用两种方式来选择执行编辑命令或主菜单中的命令。第一种是直接按下屏幕上所显示的命令键 (如 <F1> 用来移动当前点，<F3> 用来删除当前点，"F" 选择"文件"命令, 等等); 第二种方式是按回车键来进入字符编辑命令的窗口或按 <Esc> 键来进入主菜单窗口, 然后利用箭头键移至要执行的命令之上再按回车键来执行。在绝大部分情形下用户可用 <Esc> 键来放弃正在执行的命令, 或是从编辑命令窗口或主菜单中返回编辑窗口等等。

§5.3 字符编辑命令

对围线进行编辑时有两种形式的游标移动。第一种是从一条围线上跳到另一条围线上，或从围线上的一个节点移至另一个节点，用户在绝大部分情况下都是处于这种方式。第二种方式是连续移动游标，如当调整围线上一个节点的位置或移动一组围线时。在第一种方式下，用户用右箭头或下箭头键来将游标移至下一节点，用左箭头或上箭头键来将游标移至围线上的前一个节点，用 <PgUp> 键跳至前一条围线，用 <PgDn> 键跳至下一条围线。在第二种方式下，用户用四个箭头键来设定游标的位置，每次移动一个点，或用 <Shift> 键加数字键盘上的四个箭头键来快速移动游标，也可按 <Ins> 键来进入快速游标移动 (再按一次 <Ins> 键又将恢复正常移动速度)。

PZ.EXE 包含一些下拉式菜单, 进入一个下拉菜单后用户可用上下箭头键来选择其中的一项再打回车键来执行相应的操作, 也可直接键入菜单中每项左侧标明的字母键来选择执行某一项, 或者按 <Esc> 键来放弃操作。

下面解释一下字符编辑命令。

§5.3.1 移动点 (<F1>)

用于调整当前节点的位置，调好位置后键入回车即可。

§5.3.2 加入点 (<F2>)

在当前线段上 (即当前围线上用不同颜色所显示的线段) 插入一点。在调整好新插入点的位置后键入回车来认可。

§5.3.3 删除点 (<F3>)

将当前节点从围线上删掉。用户需回答 "Y" 来确认。需要注意的是当节点的一端为 Bézier 曲线时不能直接将其删除, 可利用"分段拟合"功能 (<F10>, 见 10) 来进行删除。

§5.3.4 移动围线 (<F4>)

用于移动一组围线。用户利用 <PgUp>、<PgDn> 及回车键来选择想移动的围线 (被选中的围线将改变颜色), 选择完毕后按 "<Esc>" 键进入一个菜单, 其中包含"执行"和"放弃"两个选择, "执行"指执行移动操作, "放弃"指放弃操作。移动时将选中的围线移到新的位置上再键入回车键即可。

§5.3.5 画新围线 (<F5>)

画一条新的围线。这条命令很少需要用到。程序先进入一个菜单, 要求用户选择是画圆或椭圆还是画一般曲线, 用户根据需要选择其中之一。

1. 画圆或椭圆

   用户需先给出椭圆的中心位置, 然后再移动游标来确定椭圆的大小, 此时程序会随光标的移动随时画出相应的椭圆及其长短轴, 没有画出长短轴时表明长短轴正好相等, 此时得到的将是一个标准圆。用户在确定椭圆的大小后打入回车键来确认要画的圆或椭圆。

2. 画一般曲线

   程序先要求用户给出曲线的起始点, 用户利用箭头键将光标移至起点位置再键入回车, 此时程序将在编辑窗口的边框上标上该点的位置并进入一个选择线型的菜单, 其中包括下述选择:

   - **加入直线:** 选择该项后用户可将一串直线段加入到所画围线中去, 每打入一次回车将加入一条从围线上最后一个点到光标所在点的线段。
   - **加入三次 Bézier 曲线:** 程序将围线上最后一个点做为 Bézier 曲线的第一个控制顶点, 并要求用户顺序输入第二、第三和第四个控制顶点。在用户输入了所有控制顶点后程序画出相应的 Bézier 曲线, 然后等待用户对所画的 Bézier 曲线进行调整, 此时, 为调整某个控制顶点的位置, 只需将光标移到其上并打入回车, 将其移到新的位置后再打入回车。当调整完所画的 Bézier 曲线后按 <Esc> 键进入一个菜单, 其中包括"继续"和"放弃"两个选择, "继续"指将所画 Bézier 曲线加入到围线上并继续画下一条 Bézier 曲线, "放弃"指放弃所画的 Bézier 曲线。用户在输入和调整控制顶点时程序会用虚线标出围线已画出部分的首尾处的切线方向以便于用户控制所画的 Bézier 曲线是否要与前面的线段光滑相连 (即相切)。
   - **加入相切圆弧:** 在围线上加入圆弧, 它们光滑地延长已画的围线 (即相切)。用户只需移动游标来确定圆弧的终点并键入回车即可画出一条圆弧。如果围线上还没有线段

的话, 程序会先要求用户给出圆弧的起始切线方向 (用户将游标移至某一位置并键入回车, 程序将围线起始点到该位置的连线做为圆弧的起始切线方向)。

- **加入任意圆弧:** 与加入相切圆弧时类似, 只是每次所画的圆弧不一定与前面的线段相切。用户在画每段圆弧时先给出它的起始切线方向, 然后再给出它的终止点。

用户在画围线过程中可随时按 <Esc> 回到选择线型的菜单以便改变所用的线型。当围线的终点回到起点时表示围线已经画完, 此时程序会打开一个菜单要求用户选择所画的围线是外线还是内线, 程序将根据用户的回答自动调整围线的走向 (请参看 §5.1 (第 43 页) 中关于内外线的说明)。

这里需指出的是当用户画圆弧、圆或椭圆时, 程序自动将其用 Bézier 曲线来近似, 某些情况下可能会带来一些小的偏差。

§5.3.6 删除围线 (<F6>)

删除当前围线。用户需回答 "Y" 来确认。

§5.3.7 拷贝围线 (<F7>)

拷贝一组围线。与移动围线时类似，用户先选择要拷贝的围线然后按 <Esc> 键并选择执行拷贝或放弃拷贝。选择了执行拷贝后将拷贝产生的围线移至新的位置上再键入回车。

§5.3.8 切断/连接 (<F8>)

用来将一条围线从某处截断成两条围线或是将两条围线连接成一条围线。执行此命令时程序将当前节点做为一个断点并要求用户给出第二个断点。用户将游标移至第二个断点然后键入回车。如果两个断点位于同一条围线上则表示将这条围线从断点处切断成两条围线。如果两个断点位于不同的围线上则表示将这两条围线从断点处连接起来 (此时需先确保这两条围线的走向符合 §5.1 (第 43 页) 中所述要求，否则用户将得到一条扭曲的围线)。

§5.3.9 缩放 (<F9>)

用于对一组围线进行放大、缩小。与移动围线类似，用户先选择要缩放的围线然后按 <Esc> 键并选择执行缩放或放弃, 选择了执行缩放后程序将画出一个矩形正好框住要缩放的围线并要求用户定义另一方框 (目标窗口) 来标明缩放后的大小及位置。用户先将游标移到目标窗口的一个角的位置并键入回车，然后再将游标移至其对角位置再键入回车。

§5.3.10 分段拟合 (<F10>)

用一条三次 Bézier 曲线或直线来拟合围线上的一段。程序将当前节点作为拟合区间的起点并要求用户给出拟合区间的终点。将游标移至拟合区间的终点并键入回车, 再选择用直线进行拟合还是用三次 Bézier 曲线进行拟合。如果选择了用 Bézier 曲线拟合的话, 程序将计算出一条拟合曲线然后进入"曲线微调"方式, 以便用户对拟合产生的曲线进行微调, 请参看 §5.3.11 (第 48 页) 中的**曲线微调** 命令。该命令执行完后围线上相应的区间将被拟合产生的 Bézier 曲线或直线所代替。

§5.3.11 曲线微调 (<Ctrl>+<Enter>)

用于对围线上的一条 Bézier 曲线进行微调。程序将当前的 Bézier 曲线放大显示出来, 并

且显示出它的两个中间控制顶点, 用户可通过调整这两个控制顶点来改变曲线的形状。为方便用户控制是否让该曲线与相邻的线段相切, 程序还将两个相邻的切线方向用虚线标出 (用户可将控制顶点移到相邻的切线方向上来产生光滑的连接)。为调整一个控制顶点的位置, 将光标移到该点上并键入回车, 再用箭头键来移动该点, 然后再键入回车。为退出曲线微调状态, 按 <Esc> 键进入一个菜单, 选择其中的“完毕”项来接受调整后的曲线, 或“放弃”项来放弃所做的调整。

§5.3.12 围线反向 (<Tab>)

改变围线的走向。

§5.3.13 取偏旁 (<Ctrl>+<F1>)

这是造字程序的主要拼字命令，用于从参考字中任意选取一些围线加入到正在编辑的字中来 (关于如何选择、定义和编辑参考字请参看 §5.5 (第 51 页) 中有关参考字的说明)。如果用户只想要参考字的某条围线的一部分则只需先编辑参考字并将所需部分截离出来 (利用 <F8> 命令)，或将整条围线拷贝过来后再进行截断并删除多余的围线。执行此命令时程序在屏幕上打开另一编辑窗口并将参考字显示在其中。用户利用 <PgUp>、<PgDn> 和回车键来选择所要的围线 (类似于其它命令中选择围线的做法), 选择完后按 <Esc> 键便进入一个菜单, 并选择其中的“执行”项来执行命令, “放弃”项来放弃操作。

在对参考字进行编辑时这条命令具有不同的含义。当参考字为点阵字时它告诉程序自动抽取参考字的轮廓线。当参考字是向量字时此命令不起作用。

§5.3.14 旋转/反射 (<Ctrl>+<F2>)

用于对一组围线做旋转、反射 (翻转) 变换。用户先用类似于移动、拷贝等命令中的方法选择需进行变换的围线, 然后再对选中的围线进行操作。请参考屏幕上的提示信息。

§5.3.15 重新显示 (<Ctrl>+<F3>)

重新显示编辑窗口中的内容以恢复被一系列操作打乱了的屏幕。

§5.3.16 上条围线 (<PgUp>)

跳至前一条围线。

§5.3.17 下条围线 (<PgDn>)

跳至下一条围线。

§5.4 主菜单命令

§5.4.1 文件

主要用于字库文件管理, 用来选择编辑字库中的某个字，加新字到库中，临时返回操作系统等等。包含下述子命令：

1. 跳至库中给定字号

   用来选择编辑字库中的某个字。用户也可利用此命令重新从库中调入正在编辑的字。

2. 跳至库中下一个字
   编辑库中的下一个字。
3. 跳至库中前一个字
   编辑库中的前一个字。
4. 加新字到库中
   编辑一个新字并将其加入库中。
5. 清除当前字
   将当前正在编辑的字清除。这里的清除是在内存中进行而不改变字库中的字。
6. 产生用户字库列表
   指示程序生成一个 CCT 源文件，其中包含用户字库中所有字的列表。用户可用 CCT 对其进行排版处理 (在显示或打印结果之前需先在文件 CCFONTS.DEF 中给出用户字库名，参看 §1.7 (第 5 页))。
7. 临时退回 DOS
   临时返回操作系统以执行一些 DOS 命令。为返回 PZ.EXE 只需使用 DOS 命令 "EXIT"。

§5.4.2 存盘

将当前字存到库中。PZ.EXE 没有自动存盘功能。用户在造字、改字过程中若想将当前内容存盘必须使用此命令。如果一个字已被改动而尚未存盘，当其数据有可能丢失时 (如用户选择退出 PZ.EXE 或编辑一个新的字) 程序将给出警告。

§5.4.3 显示

这条命令将当前字填充成点阵或放大后的轮廓显示在屏幕上以便用户观察所造字的效果。

§5.4.4 参考字

包含对参考字的各种操作。请参看 §5.5 (第 51 页)。

§5.4.5 设置

用于设置 PZ.EXE 中的一些参数。PZ.EXE 在它开始执行时会自动在它所位于的子目录中寻找文件 PZCONFIG.DAT。如果找到则从中读进参数值, 否则则使用参数的缺省值。此命令包含下述子命令：

1. CCT 字库路径
   给出 CCT 的标准字库及文件 CCFONTS.DEF 所在的路径。以供调入参考字时用。
2. 颜色
   设置屏幕颜色 (目前不能用)。
3. 使用鼠标器
   选择是否使用鼠标器。当无法用箭头键移动游标时往往是由于某些不兼容的鼠标驱动程序所致, 可通过选择不使用鼠标器来避免此类问题。
4. 调入参数文件
   将以前存在盘上的参数读进来。
5. 将参数存盘
   将修改后的参数存盘。

§5.4.6 退出

结束造字返回 DOS。

§5.5 利用参考字进行拼字

PZ.EXE 的主要功能之一是让用户利用已有的字来拼出新的字, 这是通过参考字的使用来实现的。用户可以任意选择一个字做为参考字并将其调入内存，对参考字进行编辑、加工，然后将参考字的某些部分加进正在造的字中去。下面解释主菜单中“参考字”项中的子命令。

主菜单中的参考字命令一共包含下列三条子命令。

§5.5.1 编辑参考字

对参考字进行编辑、加工。此时程序打开另外一个编辑窗口并将参考字显示在其中以便用户对其进行编辑。除了 ≐¡Ctrl¿+¡F1¿= 命令具有不同含义外, 所有字符编辑命令均可使用。当参考字为点阵字时用户可用 ≐¡Ctrl¿+¡F1¿= 命令来让程序自动生成轮廓线。为退出参考字的编辑窗口只需按 <Esc> 键或某条主菜单命令键。

§5.5.2 选取参考字

选择一个字做为参考字并将其调入内存。包括下列子命令：

1. 参考字体
   选择参考字的字体。程序打开一个新的窗口以便用户选择一种字体, 除 CCT 的标准字体外, 还有“用户字库”和“TIFF 文件”两个选择。其中“用户字库”表示从当前所造的字库中取参考字, 而“TIFF 文件”表示调入一个 TIFF 文件中包含的点阵图象做为参考字 (只支持非压缩的黑白 TIFF 文件格式)。由于通常扫描仪产生的图象均可存为 TIFF 格式, 利用该功能可以进行扫描造字。
2. 拼音或区位码
   当用标准字库中的字做参考字时用户可以用汉语拼音或区位码来挑字 (使用汉语拼音时需采用全拼并只能选取一级字库中的字)。如果用用户字库中的字做参考字则只能给出用户字库中的字号。
3. 调入定义的参考字
   将按上述拼音或区位码及字体所定义的参考字调入内存。当使用拼音时程序将打开一个窗口并在其中显示出所有同音字, 用户在其中选择所要的字, 然后键入回车。
4. 列出参考字供选择
   当使用拼音时此命令与“调入定义的参考字”等效。程序打开一个窗口并将从用户给出的字号开始的字显示在其中 (使用拼音时显示的是同音字)。用户可用四个箭头键在窗口中移动，用 <PgUp> 及 <PgDn> 键来翻页。当找到所要的字后打入回车来将其调入。

§5.5.3 清除参考字

将当前的参考字清除。

最后需要强调说明的是如果参考字库为点阵字库, 则调入的参考字亦为点阵字, 为使用其轮廓线用户需先对它进行编辑 (用参考字命令的子命令 2.)。编辑时用户既可将参考字做底样利用 <F5> 命令来进行人工围边，也可用 ≐¡Ctrl¿+¡F1¿= 命令来使程序自动生成轮廓线。

§5.6 使用鼠标器

`PZ.EXE` 支持 Microsoft Mouse。使用鼠标器时用户只需记住下面几条规则：

1. 移动鼠标器相当于使用四个箭头键。
2. 鼠标器的左边按键相当于回车键。
3. 鼠标器的右边按键相当于 `<Esc>` 键。
4. 如果有三个按键则中间按键相当于 `<PgDn>` 键 (用以跳至下一条围线)。

如果碰到游标不能正常移动时有可能是由于鼠标器带来的问题, 此时可在主菜单的参数项中选择不使用鼠标器。实际上 =PZ.EXE= 更适合于直接用键盘操作。

参考文献

[1] Donald E. Knouth, “The TEXbook”, Addison-Wesley, 1984

[2] 丁卫星、赖天树,《LATEX 实用教程》, 中国科技大学出版社, 1993 年

[3] 郭力、张林波等,《CCT 中外文科技激光照排系统》, 海洋出版社, 1993 年

[4] A. Borde, “Mathematical TEX by Examples”, Academic Press, Inc., 1993

[5] Leslie Lamport, “LATEX: A Document Preparation System”, Addison-Wesley, 1994

[6] M. Goossens, F. Millelbach, and A. Samarin, “The LATEX Companion”, Addison–Wesley, 1994

[7] H. Kopka and P. W. Daly, “A Guide to LATEX 2_ε : Document Preparation for Beginners and Advanced Users”, Addison-Wesley, 1995

[8] G. Grätzer, “Math into LATEX”, BIRK HÄUSER, 1996