[19]中华人民共和国国家知识产权局

[51]Int. Cl[7]
G09G 5/24

[12]发明专利说明书

[21] ZL 专利号 94112935.7

[45]授权公告日 2003 年 1 月 8 日

[11]授权公告号 CN 1098515C

[22]申请日 1994.12.9 [21]申请号 94112935.7

[30]优先权

[32]1993.12.9 [33]JP [31]309555/93

[32]1993.12.9 [33]JP [31]309556/93

[73]专利权人 佳能株式会社

地址 日本东京

[72]发明人 吉田政幸

审查员 张静海

[74]专利代理机构 中国国际贸易促进委员会专利商标事务所

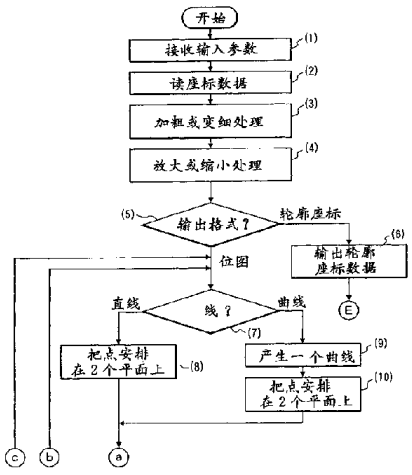
代理人 范本国

权利要求书 3 页 说明书 39 页 附图 34 页

[54]发明名称 字符发生装置及其实现方法

[57]摘要

字符发生装置及其实现方法。该字符发生装置包括:存放包括座标数据的字符数据的存储装置,根据存在该存储装置中的座标数据生成字模的发生装置,确定使该字模粗细的参数的确定装置,以及根据确定装置确定的参数转换座标数据的转换装置;并且根据转换装置所转换的座标数据,利用发生装置生成粗的或细的字模。



ISSN1008-4274

1.一种字符处理设备，包括:

存储装置，用于存储表示多个字体数据的一个表，该多个字体数据具有相应的字体风格和权重，这些字体风格和权重被得到存储以进行使用;

判定装置，用于根据存储在所述存储装置中的所述表来判定具有一个所请求的字体风格和一个所请求的权重的字体数据是否被存储;

生成装置，用于当所述判定装置判定具有所请求的字体风格和所请求的权重的字体数据被存储时获得该字体数据并从所获得的字体数据生成具有所请求的字体风格和所请求的权重的一个字符，并用于当所述判定装置判定具有所请求的字体风格和所请求的权重的字体数据未被存储时选择具有所请求的字体风格和与所请求的权重接近的一个权重的字体数据、对所选定的字体数据进行一种加粗/变细处理、并从进行了该加粗/变细处理的字体数据生成具有所请求的字体风格和所请求的权重的一个字符。

2.根据权利要求1的设备，其中当所述判定装置判定未存储具有所请求的字体风格和所请求的权重的字体数据时，所述生成装置优先选择具有比所请求的权重小的一个权重的字体数据。

3.根据权利要求1的设备，其中当所述判定装置判定未存储具有所请求的字体风格和所请求的权重的字体数据时，所述生成装置造成一种判定 - 该判定用于判断是否存储了具有所请求的字体风格和比所请求的权重小的一个权重的字体数据，且其中（a）如果该判定表明存储了具有所请求的字体风格和该较小的权重的至少一个字体数据，所述生成装置在该至少一个字体数据中选择具有所请求的字体风格和最接近所请求的权重的一个权重的字体数据、对所选定的字体数据进行加粗处理、并从进行了加粗处理的该字体数据生成具有所请求的字体风格和所请求的权重的一个字符，以及，（b）如果该判定表明未存储具有所请求的字体风格和该较小的权重的字体数据，所述生成装置从具有所请求的风格和比所请求的权重大的一个权重的至少一个字体数据中选择具有

所请求的字体风格和最接近所请求的权重的一个权重的字体数据、对所选定的该字体数据进行变细处理、并从进行了变细处理的该字体数据类属具有所请求的字体风格和所请求的权重的一个字符。

4.根据权利要求1的设备，进一步包括用于打印所述生成装置所生成的字符的打印装置。

5.根据权利要求1的设备，进一步包括用于显示所述生成装置所生成的字符的显示装置。

6.一种字符处理方法，包括:

一个判定步骤，用于根据一个表- 该表表示被存储的具有各自的字体风格和权重的多个字体数据-来判定具有一个所请求的字体风格和一个所请求的权重的字体数据是否被存储; 以及

生成步骤，用于当所述判定步骤判定具有所请求的字体风格和所请求的权重的字体数据被存储时获得该字体数据并从所获得的字体数据生成具有所请求的字体风格和所请求的权重的一个字符，并用于当所述判定步骤判定具有所请求的字体风格和所请求的权重的字体数据未被存储时选择具有所请求的字体风格和与所请求的权重接近的一个权重的字体数据、对所选定的字体数据进行一种加粗/变细处理、并从进行了该加粗/变细处理的字体数据生成具有所请求的字体风格和所请求的权重的一个字符。

7.根据权利要求6的方法，其中当所述判定步骤判定未存储具有所请求的字体风格和所请求的权重的字体数据时，所述生成步骤优先选择具有比所请求的权重小的一个权重的字体数据。

8.根据权利要求6的方法，其中当所述判定步骤判定未存储具有所请求的字体风格和所请求的权重的字体数据时，所述生成步骤造成一种判定 - 该判定用于判断是否存储了具有所请求的字体风格和比所请求的权重小的一个权重的字体数据，且其中（a）如果该判定表明存储了具有所请求的字体风格和该较小的权重的至少一个字体数据，所述生成步骤在该至少一个字体数据中选择具有所请求的字体风格和最接近所请求的权重的一个权重的字体数据、对所选定的字体数据进行加粗处理、并从进行了加粗处理的该字体数据生成具有所请求的字体风格和所请求的权重的一个字符，以及，（b ）如果该判定表明未存储具有所请求的字体风格

和该较小的权重的字体数据，所述生成步骤从具有所请求的风格和比所请求的权重大的一个权重的至少一个字体数据中选择具有所请求的字体风格和最接近所请求的权重的一个权重的字体数据、对所选定的该字体数据进行变细处理、并从进行了变细处理的该字体数据类属具有所请求的字体风格和所请求的权重的一个字符。

字符发生装置及其实现方法

本发明涉及把按矢量形式编码的字符等转换为按点形式编码的字符的一种字符发生装置以及用于实现所述转换的一种方法。

在把以矢量形式存储的数据变成位图和输出字符的常规装置中，被存在 *ROM* 或硬盘中的座标数据被读出，并通过按一定比例放大或缩小，使之被转换为所需的尺寸。然后，结果数据被转换为点形式的数据，从而提供了字符数据。

然而，在这种情况下，若同一字体形式具有不同的权值，则对应每个权值必须有一组座标数据。对于日语的字体形式来说，每种字体形式约有 8000 个字符，每种字体所需座标数据的存储量必须是 *1M* 字节到 *3M* 字节，如果对每种权值都提供座标数据，则存储量将是十分庞大的。

为了克服上述的缺点，本发明的一个目的是从字符数据中为同一字体形式产生不同的权值数据，包括至少对一种字体形式的若干轮廓数据，因此，用很小的存储量就能产生对单个字体形式的各种权值字符数据。

现在解释本发明的特征。

根据本发明的字符发生装置包括：存放由座标数据组成的字符数据的存储装置，根据存放在存储装置中的座标数据产生字模的发生装置，决定字模粗细的参数的确定装置，以及根据由确定装置决定的参数转换座标数据的转换装置；并且根据由转换装置转换的座标数据，用发生装置产生粗的或细的字模。

根据本发明的字符发生装置的转换装置通过参考邻近于某个目标的座标的两点的座标数据来决定座标数据。

根据本发明的字符发生装置的座标数据包括外部轮廓数据和内部轮廓数据。

根据本发明的字符发生装置中的确定装置决定一个参数，该参数指示大的字符数值或小的字符数值，分别对应外部轮廓数据和内部轮廓数据。

根据本发明的字符发生装置中的确定装置分别从外部轮廓数据和内部轮廓数据中、在 x 方向和 y 方向上决定大字符权值或小字符权值。

根据本发明的字符发生装置中的发生装置根据被转换的轮廓数据输出位图形式、轮廓座标数据形式或灰度形式。

从下面的说明和附图中，本发明的其他目的和特征将是清楚的。

图 1 是一个简图，说明根据本发明的字符发生装置的第一个应用系统；

图 2 是一个简图，说明根据本发明的字符发生装置的第二个应用系统；

图 3 包括图 3*A* 和图 3*B*，给出了根据本发明的字符发生装置的第一种字符发生方法的流程图；

图 4 是一个具体的图形，表示在被根据本发明的字符发生装置转换之前，一个轮廓字体的布局；

图 5 是一个具体的图形，表示被根据本发明的字符发生装置转换后，一个轮廓字体的布局；

图 6 表示根据本发明的字符发生装置的被转换轮廓字体的座标输出数据；

图 7 是一个具体的图形，表示在根据本发明的字符发生装置中被掩盖状态的点；

图 8 是一个具体的图形，表示在根据本发明的字符发生装置中的轮廓 *OR*(或)点；

图 9 表示在根据本发明的字符发生装置中转换成为第三 *Besier* 曲线的短矢量集合的过程；

图 10 是一个具体的图形，表示在根据本发明的字符发生装置中的一种被掩盖状态；

图 11 表示在根据本发明的字符发生装置中一个位图字体的产生过程；

图 12 是一个流程图，表示根据本发明的字符发生装置的一个

加粗/变细的过程例子；

图 13 表示在根据本发明的字符发生装置中决定外部轮廓或内部轮廓的预处理过程；

图 14 表示在根据本发明的字符发生装置中明体加粗处理的结果；

图 15 表示 在根据本发明的字符发生装置中哥特体加粗处理的结果；

图 16 表示在根据本发明的字符发生装置中用于决定加粗参数的第一 个表格数据；

图 17 是一个具体的图形，表示在根据本发明的字符发生装置中的外部轮廓加粗处理过程；

图 18 表示在根据本发明的字符发生装置中随着加粗或变细处理过程对框架尺寸的纠正；

图 19 是一个流程图，表示为根据本发明的字符发生装置决定外部/内部轮廓的过程示例；

图 20 是一个原理图，说明根据本发明的字符发生装置决定内部轮廓特性的处理过程；

图 21 是一个原理图，说明根据本发明的字符发生装置决定外部轮廓特性的处理过程；

图 22 是一个流程图，表示根据本发明的字符发生装置选择加粗参数的过程示例；

图 23*A* 和图 23*B* 表示在根据本发明的字符发生装置中用于决定加粗参数的第二个表格数据；

图 24 给出了根据本发明的字符发生装置中被用来准备灰度字体的一个位图；

图 25 包括图 25*A* 和图 25*B*，表示根据本发明的字符发生装置第二种字符产生方法的流程图；

图 26*A*、26*B* 和 26*C* 中的每一个都给出了根据本发明的字符发生装置中灰度字体转换表的例子；

图 27 表示根据本发明的字符发生装置中灰度字体的转换过程；

图 28 表示根据本发明的字符发生装置的被转换状态的灰度；

图 29 包括图 29*A* 和图 29*B*，表示根据本发明的字符发生装置的第三种字符产生方法的流程图；

图 30 是一个具体的图形，表示在被根据本发明的字符发生装置转换之前，一个轮廓字体的布局；

图 31 是一个具体的图形，表示在被根据本发明的字符发生装置转换之后一个轮廓字体的布局；

图 32 表示根据本发明的字符发生装置的被转换轮廓字体的座标输出数据；

图 33 是一个具体的图形，表示利用根据本发明的字符发生装置中的座标数据准备绘画表；

图 34 表示根据本发明的字符发生装置的位图字体生成过程；

图 35 是一个流程图，表示根据本发明的字符发生装置的加粗/变细过程例子；

图 36 表示根据本发明的字符发生装置决定外部轮廓或内部轮廓的预处理；

图 37 表示经过根据本发明的字符发生装置的哥特体加粗处理所得到的结果；

图 38 是根据本发明的字符发生装置用于决定加粗参数的第三个表格数据；

图 39*A* 和图 39*B* 表示根据本发明的字符发生装置用于决定加粗参数的第四个表格数据；

图 40 说明在根据本发明的字符发生装置中用来准备灰度字体的位图；

图 41 由图 41*A* 和图 41*B* 组成，表示根据本发明的字符发生装置第二种字符产生方法的流程图；

图 42*A*、42*B* 和 42*C* 中的每一个都给出了根据本发明的字符发生装置的灰度字体转换表的例子；

图 43 表示根据本发明的字符发生装置的灰度字体转换过程；以及

图 44 表示根据本发明的字符发生装置的灰度被转换状态。

第一实施例

现在将描述本发明的第一个实施例。应该注意到：本发明可适用于具有多个设备的系统或包含单个设备的装置。本发明也可以通过为这样一个系统或装置提供一个程序来实现。

图 1 是一个简图，说明利用根据本发明的字符发生装置的第一个系统的控制结构。该系统可以是一个日语的字处理机、工作站或计算机系统。

在图 1 中，*CPU*1 是中央处理器，控制整个系统并执行算术操作等等。*ROM*2 是一个只读存储器，其中存放着系统启动程序和流程图有关的若干程序（后面将被描述）以及字模数据。*RAM*3 是一个随机存取存储器，这是一个没有限制用途的数据存储区，对于每个处理过程，可装入并执行各种程序和数据。*KBC*4 是一个键盘控制 部分，从 *KB*5（键盘）中接收键输入数据，并把数据送到 *CPU*1。*ROM*2 中存放着图 16、图 26、图 38 和图 42 中所示的这类数据，将在后面被描述。

*CRTC*6 是一个显示控制器，它把将要被显示的数据送到 *CRT*7（一个显示器）。

外部存储设备 9，例如 *FD*（软盘驱动器）或 *HD*（硬盘设备）用来存放程序和数据，当在执行期间需要时，这些程序和数据就被引用或装入到 *RAM*3。*DKC*8 是控制数据传输的磁盘控制器。*PRTC*10 是打印机控制器，*PRT*11 为打印机设备，而系统总线则是上述各种组件的数据通路。

图 2 是利用本发明的字符发生装置的第二个系统的控制结构简图。该系统可以是激光打印机、喷墨打印机或感热式的印刷机。

在图 2 中，*CPU*21 是控制整个装置并执行算术运算的中央处理器。*ROM*22 是存放系统启动程序、字模数据等的只读存储器。

*RAM*23 是一个随机存取存储器，这是一个没有限制用途的数据存储区，在每个处理过程中装入并执行各种程序和数据。

*PRTC*10 是把数据传送到 *PRT*11(打印机)的打印机控制器，而打印机再打印这些数据。

在这样一个字符发生装置中，*CPU*1 根据指示某个被决定的字符权值的参数执行被存放在 *ROM*2 中的转换程序，并且把轮廓数据的座标值转换为其他的数据(后面将说明)。接着 *CPU*1 根据其座标已经被转换的轮廓数据产生一个或粗或细的字模，并且用少数几个字符数据来产生具有不同权值的粗字符或细字符。

而且，通过参考邻近某目标座标的两点的座标值，*CPU*1 决定被转换的座标值，并产生一个被很好平衡的粗的或细的字模。

此外，*CPU*1 从轮廓数据中提取外部轮廓数据和内部轮廓数据，并产生令人满意的一个粗的或细的字模。

而且，利用被抽取的外部轮廓数据和内部轮廓数据，*CPU*1 分别决定粗字符权值和细字符权值，并产生能令人满意的粗的或细的字模。

接着，利用被抽取的外部或内部轮廓数据，*CPU*1 决定 x 方向

和 y 方向独立的字符权值，并产生能有效地反映每个字体形式特性的或粗或细的字模。

此外，根据被转换的轮廓数据，*CPU*1 输出位图化的字体、轮廓座标数据或灰度字体，并且以某种适当的数据形式把或粗或细的字符数据提供给各种输出装置。

而且，根据指示被决定的字符权值的参数，从存放在 *ROM*2 中的一个表格中决定 *CPU*1 将转换的字符数据，并且转换对应被决定字符数据的轮廓数据的座标值。根据其座标被转换的轮廓数据，产生或粗或细的字模，因此，字符数据源被用来产生最佳的或粗或细的字符数据。

另外，通过转换其字符权值接近被决定的字符权值的字符数据，*CPU*1 就能产生更加精确的或粗或细的字符数据。

现在参考图 3*A* 和图 3*B* 中的流程图说明本发明的这个实施例的详细处理过程。

图 3*A* 和图 3*B* 中的流程图给出根据本发明的字符发生装置的第一字符产生方法。数字(1)到(17)指示该过程的步骤。

给出的说明包括使用一种字体形式的数据(系统中存在的并且具有某一特定的权值)来产具有不同权值字符的数据。

步(1)接收输入参数。

输入参数可能是(例如)一个字符代码、一种字体形式、权值数据以及将被输出的字符输出尺寸和形式。

字符代码由字符代码系统所决定，例如，*JIS*（日本工业标准）代码、移位 *JIS* 代码、*EUS* 代码或 *UNI* 代码，由此可预先标识目标系统。

字体形式从系统预先加入的数据〔如明体、哥特体字 或圆哥特体〕或被加入做为可供选择的数据中选择出来的。权值表示字体形式的线条粗细程度， 数据“ 很细”、“细”、“标准”、“粗”以及“很粗”被提供。

输出尺寸是表示将被输出的字体数据的实际大小的数据。

输出形式是所需字体的输出数据形式，以及发出输出请求，例如轮廓座标数据输出或位图化输出。

步（ 2）读出目标字符的座标数据。这些数据被预先存放在 *ROM*、*RAM*、硬盘或软盘上。检查在步(1)中得到的数据中的字体形式数据和字符代码数据，并且读出所需的座标数据量。

将被读出的座标数据是通过提取某个字符的轮廓的特征点得到的数据，如图 4 所示。这些数据包括每个座标点的属性信息，例如直线数据/曲线数据确标识以及轮廓起点/终点标识。

虽然在 该实施例中所用曲线数据的插值表达式可以是第二或第三 *B* 样条曲线或第二或第三 *Besier* 曲线，但所用的插值表达式是预先决定的。

指示字符框架的座标最小值是“0”．最大值是“800”。

步（ 3）根据被包含在输入参数中的权值数据对座标数据进行

加粗或变细的处理。

这一处理过程将在后面参考图 12 中的流程图加以描述。做为这一处理的结果，座标数据被变换为粗化或细化轮廓的数据，如图 5 所示。

执行加粗或变细的处理后，座标点具有一对一的对应关系，每个点的属性标识符并没有变化。

步(4)根据输入参数中的输出尺寸，对在步(3)中得到的粗化或细化的座标数据执行扩大或缩小处理。

当所需的输出尺寸为(Ax, Ay)时，步(3)中得到的座标值为(x,y)，执行放大或缩小处理后的座标值为(X,Y)，而被存储的字符框架尺寸为(Mx,My)，则有：

$(X,Y)=(x\times Ax/Mx, y\times Ay/My)$。

对一个字符的所有座标都用上面的表达式计算。在步(3)中得到的每个座标点的属性标识符都不改变。

当在步(5)中需要位图化输出时，程序控制进入步(7)。

步(7)到步(13)是从座标数据中实际准备位图化数据的过程。步(7)执行一次检测，制定目标座标数据是否为直线或曲线数据。在步(7)上，当座标数据为直线数据时，座标点被定义为一个直线的起点，而其后的一个座标点被定义为该直线的终点。然后把程序控制移到步(8)。

在步(7)上，当目标数据为曲线数据时，该座标点和曲线终止标

识符所在的点之间所有点的座标数据被假定为曲线数据。程序控制然后移到步(9)。

步(8)执行产生一条直线的处理过程。在这种情况下，用 *DDA* 方法来产生一条直线。点被放在两个平面中，一个平面用于描绘，而且如图 7 所示，每个点只放在一个 x 座标上，对应一个 y 座标，因为显示时线被从左到右扫描，而且在奇数位置 上的"1"到偶数位置上的"1"之间的区间都用"1"表示。若点被放在对应一个 Y 座标的多个 X 座标上，则不能很好地执行显示功能。

另一个平面为轮廓 *OR* 平面，如图 8 所示，在该平面中，对单个 Y 座标的所有 X 座标都被置"1"，因为该轮廓 *OR* 平面被用来补偿在描绘平面中被跳过的那些位。

步(9)执行把曲线数据转换为一组短直线(短矢量)的处理过程。

图 9 说明把一个第三 *Besier* 曲线转换为一组短矢量。

在图 9 中，点 A、B、C 和 D 是在步(3)中经过座标转换得到的曲线数据(第三 *Besier* 曲线点)。这些点被用来计算点 a、b 和 c。点 a 位于点 A 和点 B 之间的中点，点 b 位于点 B 和点 C 之间的中点，而点 c 则位于点 C 和点 D 之间的中点。

接着，计算点 x、y 和 z。点 x 为点 a 和点 b 之间的中点，点 z 为点 b 和 c 之间的中点，而点 y 为点 x 和 z 之间的中点。于是，点行 *Aaxy* 为新的第三 *Besier* 曲线点，而点行 *yzcD* 为另一个第三

Besier 曲线点。

当这些 *Besier* 曲线点按照相同的方式但更详细地划分并满足某种具体的制定参考时，该划分过程就停止了。因此得到的第三 *Besier* 曲线点行就是短矢量的集合。

步(10)根据步(9)得到的短矢量集合把点放在两个平面中。把点放在两个平面中的方法与步(8)中执行的方法相同。对所有的短矢量重复执行这一过程。

步(11)执行一次检测，判断对一个轮廓的所有座标数据的处理是否已经完成。若完成则程序控制移到步(13)，否则移到步(12)。

步(12)更新当前座标数据的指针以便处理下一个数据。若下一个数据是直线数据，则指针被更新为指向下一个座标数据。若下一个数据为曲线数据，则指针被更新为指向曲线的终止座标点。然后程序控制 返回步(7)，执行一次检测，判定是否为直线的数据或曲线的数据，并根据这些数据安排点。

步(13)判定一个字符的所有轮廓数据是否都已经被处理过。若所有的轮廓数据已经被处理，程序控制进入步(15)。否则转到步(14)。

因为一个轮廓的数据已经被处理，在步(14)中指针移到下一个轮廓的头，程序控制返回步(7)。

在步(15)中，因为对所有座标数据把点安排在两个平面上的处理已经完成，如图 10 所示，用于描绘的平面沿着每个被扫描的线

从左边开始扫描，并且在奇数标号“1”到偶数标号“1”之间的区间都用“1”显示。

对所有的被扫描行执行显示处理。然后，在步(16)，如图 11 所示，把从步(15)中得到的描绘平面的数据和在步(8)和(10)中得到的轮廓 *OR* 平面的数据进行逻辑或运算，从而得到一个字符的位图化数据。

最后，步(17)把步(16)得到的一个字符的数据 返回到请求一方指定的某个区域中，并终止处理过程。

现在参考图 12 中的流程图，详细描述图 3 中步(3)的加粗/变细示例过程。

图 12 是一个流程图，表示根据本发明的字符发生装置的一个加粗/变细过程示例。数字(1)到(12)表示该过程的步骤。

在该实施例中的加粗/变细处理中，加粗/变细参数根据外部轮廓 *a* 和内部轮廓 *b* 和 *c* 而变化，如图 4 所示。由于轮廓座标点行并没有区分为外部轮廓和内部轮廓，因此，必须对座标进行检测以判断是外部轮廓或是对内部轮廓的。为了改变轮廓座标，外部轮廓座标使用了一个外部轮廓加粗/变细参数，而内部轮廓座标也用了一个内部轮廓加粗/变细参数。

应该注意到：当形成一个轮廓的外部轮廓点行为逆时针时，则内部轮廓点行被安排为顺时针方向的。而后外部轮廓点行是顺时针的，则内部轮廓点行就安排为逆时针的。

在图 12 的步(1)到步(3)中，执行预处理以判断将在步(4)中被处理的目标轮廓是否为外部或内部轮廓。

在该处理过程中，抽取一个起点和两个和该起点相邻的点。当起点为 *AS* 时，在起点之前的点(一个轮廓的最终点)为 *A* 而跟在起点后的点为 *B*；在通过划分矢量 *AS* 和 *SB* 形成的角度的方向上提供用于预处理的一个点。

当外部轮廓点行为逆时针方向时，起点由某个具体的值向右移动(当面向该矢量的移动方向时)。当外部轮廓点行为顺时针方向时，起点由某个特定的值向左移动(当面向矢量移动的方向时)。

程序控制移到步(4)，执行一次检测，判断目标轮廓是外部轮廓或内部轮廓，使用了在步(3)中得到的用于预处理的点。这一步的处理将结合图 19 中的流程图详细说明。

接着，在步(5)上，根据权值决定外部和内部轮廓厚度的参数。决一轮廓厚 度的参数在 x 方向和 y 方向上对外部和内部轮廓有独立的值，并且分别负责水平线和垂直线的加粗处理。

因为例如为了加粗一个明体字符(如图 14 所示)其水平线不需要加粗太多，而其垂直线就必须加粗很多，因此，x 方向和 y 方向要求不同的值。

然而，对于圆哥特体字体形式的字符来说，水平方向上的线加粗的程度却和垂直方向上的线一样，如图 15 所示。

因此，单个字体形式的加粗必须被修改。如图 16 所示，表示从

外部轮廓和内部轮廓到字体形式和权值的水平和垂直线的中心的参考值被预先放在一个表格中（通过从 *ROM*2 或 22、或到 *RAM*3 或 23 的另一存储介质中写数据准备的）。标准字体权值和所需字体权值之间的差被用来决定在 x 方向和 y 方向上外部轮廓和内部轮廓的加粗。对于为变细而把座标数据用作标准数据的明体形式来说，为了产生一个粗的明体字符，通过确定外部轮廓的 x 部分的 15 和 9 之间的差以及其 y 部分的 25 和 15 之间的差来决定加粗的值（参数）。

当加粗的值为正时，执行加粗处理，而当加粗的值为负时，执行变细处理。在步(6)到步(10)的过程期间，对构成一个轮廓的所有座标点行执行加粗/变细处理。

首先，在步(6)中得到将被处理的一个目标点。接着，在步(7)中，求出与该目标点相邻的点。最后，在步(8)中实际执行加粗处理过程。

如图 17 所示，假定目标点为 *B*，前面的点为 *A*，后面的点为 *C*。当外部轮廓为逆时针方向时，该目标点相对于平分曲矢量 *AB* 和 *BC* 形成的角的线向右移动。当外部轮廓为顺时针方向时，目标点向左移动。该点移动的范围等于和在步(5)中得到的 x 加粗值和 y 加粗值形成的三角形的斜边。

在步(9)中，当一个轮廓的所有座标点都已经被处理时，程序控制转到步(11)。当有一些轮廓点要被处理时，程序控制转到步

(10),在这里更新指针,使之指向下一个轮廓点,并再次执行加粗处理过程。

在步(11)中执行一次检测,判断是否对一个 字符的所有轮廓已经被处理。当处理完成时,程序控制移到步(12)。当还有剩下的轮廓要处理时,更新指针使之指向下一轮廓点,并再次执行加粗处理。

当对所有的轮廓座标点的加粗处理都完成时,程序控制转到步(12)。如图 18 所示,当加粗处理被执行时,结果字符框架变大,而当变细处理被执行时,结果框架变小。

因此,在步(12)中,需要调整结果框架的尺寸,使之符合原始字符框架的尺寸。放大/缩小率是在步(5)中得到的对外部轮廓加粗值的两倍(对变细处理为页的加粗值)。

若原始字符框架的 X 宽度为 Bx,外部轮廓的水平加粗值为 Fx, Y 宽度为 By,外部轮廓的垂直加粗值为 Fy,执行加粗处理的座标为(x,y),则调整字符框架尺寸后的座标为(X,Y),

$$(X,Y)=((x+Fx)\times Bx/(Bx+Fx\times 2),(y+Fy)\times By/(By+Fy\times 2))。$$

对一个字符的所有座标点行执行这一计算,并且终止如图 10 所示的加粗处理过程。

现在结合图 19 中的流程图详细介绍图 12 中的步(4)的外部/内部轮廓确定过程。

图 19 中的 流程图是根据本发明的字符发生装置确定外部/内部轮廓的一个示例过程。数字(1)到(8)表示该过程的步骤。

如图 20 所示，通过在某个特定的点上连接两个轮廓座标点 *ABCD* 形成的逆时针方向角被定义为正值；并且当这些点的角度之和为 2&*P* 时，这些点被确定为是对内部轮廓的。

当用图 20 中所执行的同样方式计算，而角度之和为“0”时，这些点被确定为是对外部轮廓的。因为在图 12 中的步(1)到(3)已经计算出具体的点，因此，执行该特定点形成的角度之和的计算过程，如图 19 中的流程图所示。

步(1)执行初始化，置角度之和为“0”。步(2)抽取两个相邻的点。步(3)通过确定矢量以及矢量之间角度的大小把该特定的点和两个被提取的点连接起来。假定该特定点为 *X*，而两个提取点为 *A* 和 *B*，可由外积表达式求出角度。其结果在步(4)中被加入角度之和。

步(5)执行一次检测，判断两点的一个轮廓提取是否已完成。当对所有的点执行这种计算后，程序控制转到步(6)。当还有剩下的数据要计算时，程序控制返回到步(2)以便重复这一计算。步(6)执行一次检测，判断角度之和是否为“0”或“2π”，若其和为“0”，程序控制转到步(7)，置外部轮廓标识符并终止该处理过程。若其和为“2π”，程序控制转到步(8)，置内部轮廓标识符并终止处理过程。

第二实施例

现在讨论另一个实施例。

下面 将讨论这样一种情况：当需要一个具有特定权值的字体形式时，存在两个以上的权值可被用作同一字体形式的标准。在这种情况下，从同一字体形式的多个字体权值中选出将被用来处理所需权值的字体权值是很重要的。

当具有标准权值的字体形式被决定时，按照对第 一实施例相同的方式执行下面的权值转换过程。因此，只说明选择标准权值的方法。

图 22 是一个流程图，给出了根据本发明的字符发生装置选择加粗参数的示例过程。数字(1)到(6)表示该过程的步骤。

首先，步(1)需要检测哪种字体形式的哪个权值存在存储设备中。存在该字体形式的头区中的头数据被引用到一个表格中，指示这类数据存在的数据也被输入到该表格中。

在图 23*A* 和 2*B* 中的示例表格中，权值 3 和 7 对明体形式，权值 5 对圆哥特体形式 ，权值 6 对斜哥特体形式，而权值 4 和 7 则对应楷体 形式。

在步(2)中，执行一次检测，判断所需权值数据是否已经被存在例如 120*M* 或硬盘这样的存储设备中。对步(1)中准备的表格检索，通过利用所需的字体形式和权值数据，就能判定数据是否被存在存储设备中。当所需的权值数据被存在某个存储设备中时，程序控制到步(3)，否则转到步(4)。

当所需的权值数据已经被存在存储设备时执行步(3)的处理。因不要求加粗/变细处理，从存储设备中读出座标数据。通过引用这些数据，根据输出的尺寸执行放大/缩小处理，并准备好位图化的字体，然后终止处理过程。

录所需的权值数据没被存在某个存储器中时，执行步(4)的处理。执行加粗/变细处理以输出具有所需权值的字符。

因此，需要选择源数据来执行加粗/变细处理。通常，加粗处理得到的结果变坏的程度不如变细处理的结果。因此，利用步(1)中准备好的表格，执行一次检测以判断权值的数据是否小于被存在存储设备中的所需的权值。

当权值 的 数据小于存在存储设备中所需数值时，程序控制转到步(5)。当较小权值的数据没被存在存储设备中时，程序控制转到步(6)。

例如，为了输出具有权值 5 的明体形式字体，因为具有权值 3 的明体形式字体已经存在存储设备中，程序控制转到步(5)。而为了输出具有权值 3 的斜哥特体形式的字体，因为不存在具有较小权值的斜哥特 体形式的字体，因此，程序控制转到步(6)。

在步(5)中，具有一个将被用作加粗处理标准的权值的字体形式被选择。x 方向和 y 方向上的参数被置为将要执行加粗处理的外部和内部轮廓。

为了输出具有权值 5 的明体形式字体，因为存在着具有权值 3

的明体形式字体，因此，每个外部和内部轮廓的 x 方向和 y 方向的权值 3 和权值 5 之间的差被置为加粗参数。在步(6)中，选择一种具有用作变细处理标准的权值的字体形式。接着，为每个外部和内部轮廓的 x 方向和 y 方向设置参数。

为了输出具有权值 3 的斜哥特体形式的字体，因为具有权值 6 的斜哥特体形式的字体已存在，轮廓的 x 方向和 y 方向的权值 3 和 6 之间的差被置为参数。

在上述的步(5)或(6)中设置 3 加粗或变细参数后，执行在第一实施例中所说明的处理过程，并输出一个具有所需权值的字符。虽然只介绍了判断是否存在具有比目标字符的权值小的权值的字体数据，但也可以执行检测来判断是否存在比目标字符的权值大的权值的字体数据。

此外，可以选择具有和目标字符的权值最接近的权值的数据，用来执行权值变换。

第三实施例

现在介绍另一个实施例。

下面将说明在第一和第二实施例中执行的加粗过程中使用灰度字体的情况。

位图化的字体是一种二进制字体，其点值为“0”或“1”，而灰度字体则是一种多值字体，其中的每个点都可能有多个值，如 0 到 3、0 到 5、或 0 到 255。

做为产生灰度字体的一种方法，通常，当具有 $n2$ 灰度级的灰度字体将被输出时，如图 24 所示，所需的输出尺寸在图 25*A* 和 25*B* 中的步(4)在垂直和水平方向上被放大 n 倍，并且利用得到的输出尺寸准备好位图化的字体。

一个位图被垂直和水平地划分形成 n 个位，如图 24 所示。被包含在每个 $n \times n$ 的正方形域中的位1的数目决定了灰度每点的多值。

当第一实 施例被应用到灰度字体时，所执行的处理过程如图 25*A* 和 25*B* 中的流程图所示。

图 25*A* 和 25*B* 是根据本发明的字符发生装置第二字符产生方法的流程图。数字(1)到(19)表示该过程的步骤。

由于这 个处理 过程的流程图同第一实施例中对位图化字体的生成或轮廓座标输出的处理几乎是一样的，并且其步骤几乎对应图 3 中的那些步骤，因此，这里只讨论由于灰度字体的生成而不同的过程步骤。

不同于图 3*A* 和 3*B* 的步骤有步(1)和(4)以及步(17)和(18)，这些步骤将被加入。

步(1)取得输入参数，并且把灰度数据也加到输入参数中。例如，这些参数中有字符代码、字体形式、权值数据、字符输出尺寸、灰度级数据、输出形式以及输出设备的特性。

字符代码由字符代码系统所决定，例如 *JIS* 代码、移位 *JIS* 代

码、*EUS* 代码或 *UNI* 代码，由此预先标识一个目标系统。

字体形式从预先加入系统的数据中选择，如明体、哥特式或圆哥特体，或者被作为可选项加入的数据。

数据的内容和生成位图化字体时所用的数据完全相同。用于产生灰度字体的具体数据没有被存储。

权值数据是用于决定 字体形式的行粗细程度的数据，被提供的这类数据有“很细”、“细”、“标准”、“粗”和“很粗”。输出尺寸是表示将被输出的字体数据实际大小的数据。x 方向上和 y 方向上的大小都被要求。灰度级数据指出灰度字体将被准备的灰度级。

根据输出设备的特性，灰度级被置为 4、16 或 256 级。输出格式是对所需字体的输出数据形式，包括轮廓座标数据输出、位图化字体输出、灰度字体输出和一点(1—*dot*)表达式形式。例如，灰度字体的 1—*dot* 表达式形式用来表示一字节是否表达一个点，或者一字节表达 2 个点或 4 个点。输出设备的特性是表示如何为显示器准备一种最佳灰度字体而决定一种灰度级的数据。

在步(4)中，通过引用灰度字体的输出规格和灰度级，对步(2)中读出的座标数据执行放大/缩小处理。当所需输出尺寸为(Ax, Ay)时，灰度级为 n、在步(3)中得到的座标为(x,y)，经过放大/缩小 处理后得到的座标为(X,Y)，而被存储数据的字符框架尺寸为(Mx,My)，用于数据放大或缩小的表达式为：

在步(17)和(18)中,利用步(16)中准备的位图化字体得到一个灰度字体。首先,在步(17)中,根据输出设备特性(要求作为一个输入参数)选择灰度转换表格。采用16灰度级的一个4×4的掩码,代表输出设备特性的值被预先存在灰度字体转换表中,如图26所示。

图26*A*给出了一个例子,其中,输出设备的亮度特性是一致的;图26*B*也给出了一个例子,但其中,点中心亮度高,而其外圈部分亮度却较低;图26*C*的例子中,点的外圈部分亮度高而其中心亮度却较低。

从中选择最适合输出设备高度特性的表格,然后,在步(18)中利用步(17)选择的表格准备好一个灰度字体。现在结合图27讨论这一处理过程。

图27给出了在步(16)中得到的位图化字体。垂直和水平的长度已经由灰度级n乘$\sqrt{n}$得到。

因此,垂直和水平长度被除以$\sqrt{n}$并且提取$\sqrt{n}\times\sqrt{n}$的网格正方形。每个网格正方形的位值被进入表中其对应网格正方形的值所乘(表格是在步(17)中得到的)。

对这些乘积求和就得到一个目标点的灰度值。

图27给出一个例子,其中,具有16灰度级的灰度字体被输出。为4×4的网格正方形选择图26*B*中的表格。

对所有的网格正方形执行上述的处理过程,因此产生如图28

所示的灰度字体。接着，在步(19)中，根据输出格式存储该灰度字体并且把数据返回到请求一方。

如果所需的输出格式是一字节对一点，则网格正方形中的值被压缩为一字节并且被存储起来。

如果所需的输出格式是把邻近的点压缩到一字节，则一个点被压缩到4位并被存储，并且数据被返回到请求一方。接着终止该处理过程。

第四实施例

因为该实施例的字符发生装置的分块电路已经结合图1和图2被说明过，因此，这里将不作解释。

在图1和图2所示的字符发生装置中，根据指示某个被确定的字符权值的参数，*CPU*1执行存在120*M*2中的一个转换程序，并把轮廓数据的座标值变换为其他的数据(将在后面描述)。接着，*CPU*1根据其座标已经被变换的轮廓数据生成一个粗的或细的字模，并利用少数几个字符数据产生具有不同权值的一个粗字符或细字符。

而且，通过参考邻近某个目标的座标的两个点的座标值，*CPU*1确定一个被变换的座标值，并产生一个被很好平衡的粗的或细的字模。

此外，*CPU*1利用轮廓数据，确定x方向和y方向的独立的字符权值，并生成能有效地表示每个字体形式特性的一个或粗或细的

字模。

而且，根据被转换的轮廓数据，*CPU*1 输出位图化字体，轮廓座标数据或灰度字体，并以某种适当的数据格式把或粗或细的字符数据提供给各种输出装置。接着，根据指示某个被确定的字符权值的参数，从某个被存在 120*M*2 中的表格中的确定 *CPU*1 将转换的字符数据，并且变换对应这些被确定字符数据的轮廓数据的座标值。根据其座标被变换的轮廓数据，产生一个或粗或细的字模，使得字符数据源能被用来产生最佳的粗的或细的字符数据。

另外，通过转换那些其权值接某个被确定字符权值的字符数据，*CPU*1 生成更加精确的粗的 细的字符数据。

现在结合图 29*A* 和 29*B* 中的流程图讨论本发明的该实施例的详细处理过程。

图 29*A* 和 29*B* 的流程图给出了根据本发明的字符发生装置的第三种字符发生方法。数字(1)到(16)表示该过程的步骤。

将给出的解释包括利用一种字体形式的数据(存在一个系统中并具有特定的权值)来产 具有不同权值的字符的数据。

步(1) 接收输入参数。例如，输入参数可以是字符代码、字体形式、权值数据、以及将要输出的字符尺寸和格式。字符代码由某个字符代码系统确定，例如 *JIS* 代码、移位 *JIS* 代码、*EUS* 代码或 *UNI* 代码，由此预先标识某个目标系统。字体形式从该系统预先加入的数据中选择，例如明体、哥特体、圆哥 特体或作为所选项加

入的数据。权值是关于字体形式的线粗细程度的数据1被提供的这类数据有“很细”、“细”、“标准”、“粗”和“很粗”。输出尺寸(规格)是表示将被输出的字体数据的 实际大小的数据。输出格式是所需字体的输出数据形式，并且发出如轮廓座标数据输出或位图化输出这样的一个输出请求。

步(2)读出某个目标字符的座标数据。这些数据被预先存在*ROM*、*RAM*，硬盘或软盘中。检测步(1)得到的字体形式数据和字体代码数据，并读出所需数量的座标数据。将被读出的座标数据是通过提取字符轮廓的特征点所得到的数据，如图4所示。些数据包括每个座标点的属性信息，例如直线数据/；曲线数据确定标识符和轮廓起点/终点标识符。虽然用于该实施例的曲线数据的插值表达式可以是第二或第三*B*样条曲线，或第二或第三*Besier*曲线，但所用的插值表达式是预先确定的。

用来描述字符框架的座标最小值为“0”，而最大值为“800”。也包括在每个笔划框架中从字符的原始点到标准点的偏移信息。在步(3)中，根据被包括在输入参数中的权值数据，对座标数据执行加粗或变细处理。该处理过程的细节将在后面结合图37中的流程图描述。做为这一处理过程的结果，座标数据被变换为被粗化或细化轮廓的数据，如图31所示。

执行加粗或变细处理之后，座标点具有一对一的对应关系，而且每个点的属性标识符不变。步(4)根据被包含做为输入参数的输

出尺寸，对步(3)中得到的被加粗或变细的座标数据执行放大或缩小处理。

当所需的输出尺寸为(Ax,Ay)时，在步(3)中得到的座标值为(x,y)，经过放大或缩小处理后的座标值为(X, Y)，而被存放的字符框架尺寸为(Mx,My)，则有：

(X,Y)=($x\times Ax/Mx$,$y\times Ay/My$)。

用上面的表达式计算来自单个字符的所有座标。在步(3)中得到的每个座标点的属性标识符不变。当步(5)要求一个位图化输出时，程序控制转到步(7)。步(7)到(13)是从座标数据中实际准备位图化数据的过程。步(7)执行一次检测，判断目标座标数据是否为直线数据或曲线数据。

在步(7)中，当座标数据为直线数据时，该座标点被定义为直线的起点，而且随后的座标点被定义为直线的终点。接着程序控制转到步(8)。在步(7)中，当目标数据为曲线数据时，在该座标点和放有曲线终止标识的点之间的所有点的座标数据被假定为曲线数据。程序控制然后移到步(9)。步(8)执行产生一条直线的处理过程。在这种情况下，利用 *DDA* 方法生成一条直线。由 *DDA* 得到的座标数据被输入到一个用于显示(描绘)的座标表格中，如图 33 所示。

在如图 33 所示的用于描绘的座标表中，对应输出区中的每个 y 座标输入 x 座标的一个起点座标和一个终点座标。做为 *DDA* 方法的结果，当对应单个 y 座标存在多个 x 座标时，这样设置 x 座标，

使得最后的一个 x 座标与一个笔划的轮廓相关。在步(9)中，曲线数据被转换为一组短直线(短矢量)。

图 9 说明把第三 *Besier* 曲线转换为一组短矢量。因为该处理过程以前已经描述过，这里将不给出该处理过程的解释。

在步(10)中，根据步(9)中得到的短矢量集合，把座标数据输入到用于描绘的座标表格中。把座标数据输入该表的方法和步(8)中执行的方法一样。对所有的短矢量重复执行这一过程

步(11)执行一次检测，判断一个轮廓的所有座标数据的处理是否已经完成。若处理已经完成，程序控制转到步(13)。还则，程序控制转到步(12)。在步(12)更新当前座标数据的指针以便处理下一个数据。若下一个数据为直线数据，更新指针使之指向下一个座标数据。若下一个数据为曲线数据，更新指针使之指向曲线的终止座标点。

接着程序控制返回步(7) ，执行检测以判断该数据是直线的或曲线的，并根据该数据安排点。步(13)执行检测以判断是否已经处理了一个字符的所有轮廓数据。当所有的轮廓数据已经被处理时，程序控制转到步(15)。否则转到步(14)。因为一个轮廓的数据已经被处理，在步(14)中移动指针，使之指向下一个轮廓的头，程序控制返回到步(7)。因为在步(15)中，所有座标数据的点已经被放在两个平面上，因此，采用一种非零的卷绕方法，对对应 y 座标并在步(8) 和(10)中被存在描绘座标表中的 x 座标执行描绘过程，如

图 34 所示。根据这种方法，沿着每个被扫描的线从左边开始对平面扫描。在起点的标识符的值被增 1，虽然标识符的值被减 1。若该标识符的值非“0”，则在这些标识符的点之间的区间被描绘为“1”。

最后，在步(16)中，步(15)得到的对一个字符的数据被返回到由请求一方所指定的区域上，并终止该处理过程。

现在结合图 35 中的流程图详细描述图 29 中步(3)的加粗/变细示例过程。

在该实施例的加粗/变细处理过程中，改变加粗/变细一个笔划的参数以便修改每个轮廓点的座标。

图 35 是一个流程图，给出根据本发明的字符发生装置的加粗/变细过程的示例。数字(1)到(9)表示该过程的步骤。

步(1)根据某个权值决定一个轮廓粗细程度的参 数。确定轮廓粗细度的参数在对该轮廓的 x 方向和 y 方向上有独立的值，并分别处理水平线和垂直线的加粗过程。

这是因为，例如为了加粗一个明体字符，如图 36 所示，其水平线并不需要被加粗太多，而其垂直线必须被加粗许多，因此，x 方向和 y 方向要求不同的值。然而，对于圆哥特体字体形式的字符来说，水平方向的线被加粗的程度如同垂直方向上的线一样，如图 37 所示。因此，对单个字体形式的加粗值必须被修改。如图 38 所示，表示从轮廓到字体形式的水平和垂直线中心的参考值和权值的数据被预先送入一 个表中。标准字体权值和所需字体权值之间的差

被用来确定 x 方向和 y 方向上轮廓的加粗处理过程。当该加粗值为正数时，执行加粗处理过程，而当加粗值为负数时，执行变细处理过程。对于用加粗的座标数据做为标准数据的圆哥特本形式，为了产生一个很细的圆哥特体字符，通过确定 x 方向的 10 和 25 之间的差和 y 方向的 10 和 25 之间的差来决定一个加粗值(参数)。

步(2)根据加粗参数修改每个笔划的偏移信息。为了获取被修改的偏移信息，加粗参数的 x 值和 y 值被从该偏移的 x 座标和 y 座标中减去。

在步(3)到步(7)中，对组成一个笔划的所有座标点行执行加粗/变细处理。首先，步(3)取出将被处理的目标点。接着，步(4)取出与该目标点相邻的点。

步(5)实际执行该加粗处理过程。因为该处理过程已经结合图17详细解释过，因此，这里将不再解释。点被移动的范围等于由步(1)得到的 x 加粗值和 y 加粗值形成的三角形的斜边。因这在这一处理过程中获得的座标值就是单个笔划框架的座标值，因此，它们被加到步(2)中求出的偏移座标值上，以便从字符原点中提供座标。

在步(6)中，当一个轮廓的所有座标点都被 处理时，程序控制转到步(8)。当还有一些轮廓点要处理时，程序控制移到步(7)，修改指针使之指向下一条轮廓点，并且两次执行该加粗处理过程。步(8)执行检测以确定一个字符的所有轮廓是否已经都被处理。当这

个处理过程完成时，程序控制移到步(9)。否则更新指针使之指向下一个轮廓点，并再次执行加粗处理。

当对所有的轮廓座标点的加粗处理都已经被执行时，程序控制转到步(9)。如图15所示，当执行加粗处理时，所得到的字符框架变大，而当执行变细处理时，其结果的框架变小。

因此需要调整结果框架的尺寸，使之符合原始字符框架的尺寸。放大/缩小率是步(1)中得到的轮廓加粗值(变细时加粗值是负数)的两倍。

若原始字符框架的 x 宽度为 Bx，外部轮廓的水平加粗值为 Fx，y 宽度为 By，外部轮廓的垂直加粗值为 Fy，执行加粗处理的座标为(x,y)，则在字符框架尺寸被调节后的座标为(X,Y)

$$(X,Y)=((x+Fx)\times Bx/(Bx+Fx\times 2),(y+Fy)\times By/(By+Fy\times 2))。$$

对一个字符的所有座标点行执行这一计算，然后终止图35所示的加粗处理过程。

第五个实施例

现在讨论第五个实施例。给出的说明将对于这样的一种情况：当所需的字体形式具有某个特定的权值时，存在两个以上的权值可用作这个相同字体形式的标准。

在这种情况下，从同一字体形式的多个字体权值中，选择将被用来处理所需权值的字体权值是很重要的。因此，当确定具有某个

标准权值的字体形式时，采用和第一实施例中相同的方式执行下面的权值转换处理过程。这里只说明选择一个权值作为标准的方法。

图 22 是一个流程图，给出根据本发明的字符发生装置选择一个加粗参数的示例过程。数字(1)到(6)表示该过程的步骤。

首先，步(1)需要检查哪个字体形式的哪个权值被存在存储设备中。存放在字体形式头部中的头数据被引用到一个表格中，而指示这类数据存在的数据也被送入该表中。在图 39*A* 和 39*B* 的示例表格中，权值 3 和 7 对明体形式，权值 5 对应圆哥特体形式，权值 6 对应斜哥特体形式，而权值 4 和 7 对应 *Block* 形式。

步(2)执行检测，判断所需的权值数据是否已经被存在某个存储设备中，例如 *ROM* 或硬盘。检索步(1)所准备好的表格，利用所需的字体形式和权值数据，就能确定数据是否被存在存储设备中。当所需的数值数据已经被存在存储设备中，程序控制转到步(3)。否则转到步(4)。当所需的权值数据被存在存储设备中时执行步(3)的处理。因为不要求加粗/变细处理，因此，从存储设备中读出座标数据。通过引用这些座标数据，根据输出尺寸执行放大/缩小处理过程，并准备好位图化字体。然后终止该处理过程 。

当所需权值数据没被存在存储设备中时，执行步(4)的处理。执行加粗/变细处理以输出具有所需权值的字符。因此需要选择源数据来执行该加粗/变细处理过程。一般来说，加粗处理的结果比变细

处理的结果要好一些。因此，利用步(1)准备好的表格，执行一次检测，判断小于所需权值的权值数据是否被存在存储设备中。

当小于所需权值的权值数据被存在某个存储设备中时，程序控制移到步(5)。否则转到步(6)。

例如，为了输出具有权值为5的明体形式字体，因为存储设备中存在具有权值为3的明形式字体，因此程序控制转到步(5)。为了输出具有权值为3的斜哥特体形式的字体，因为不存在具有较小权值的斜哥特体形式的体，因此程序控制转到步(6)。

步(5)选择一个具有将用作加粗处理标准的权值的字体形式。设置 x 方向和 y 方向上的参数，以便执行加粗处理。为了输出具有权值5的 *Ming* 形式字体，因为存在权值为3的 *Ming* 形式字体，因此，轮廓的 x 方向和 y 方向上的权值3和权值5之间的差被置为加粗参数。步(6)选择具有用作变细处理标准的权值的字体形式。接着设置轮廓的 x 方向和 y 方向的参数。

为了输出权值为3的斜哥特体形式的字体，因为存在权值为6的斜哥特体形式的字体，因此，轮廓的 x 方向和 y 方向的权值3和6之间的差被置为参数。在上述的步(5)和(6)中设置3加粗或变细的参数之后，执行第四实施例中所介绍的处理过程，并输出具有所需权值的一个字符。

虽然给出的说明只用于判断权值小于某个目标字符的字体数据是否存在，但也可以检测判断权值大于某个目标字符的字体数

据是否存在。此外，可以选择权值最接近目标的数据，并用来执行权值变换。

第六个实施例

现在将介绍第六个实施例。该实施例给出的说明是对于在第四和第五实施例中执行的加粗处理的灰度字体。

位图化字体是一种二进制字体，其点的值或为“0”或为“1”，而灰度字体则是一种多值字体，其中的每个点可以有多个值，例如 0 到 3、0 到 5 或 0 到 255。通常，作为产生灰度字体的一种方法，当具有 $n2$ 灰度级的灰度字体将被输出时，如图 40 所示，在图 41*A* 和 41*B* 的步(4)中，在垂直和水平方向上把所需的输出尺寸放大 n 倍，并利用该结果输出尺寸准备好一个位图化的字体。

如图 40 所示，一个位图被垂直和水平地划分，形成几个位(位组)。被包含 在每个 $n\times n$ 正方形域部分中的位 1 的数目确定了灰度的一个点的多个值，所执行的处理过程如图 41 中的流程图所示。

图 41*A* 和 4 1*B* 是根据本发明的字符发生装置的第四种字符生成方法的流程图。数字(1)到(18)表示该过程的步骤。

因为该处理过程的流程图几乎和第四实施例中的位图化字体生成或轮廓座标输出的处理一样，并且其处理步骤几乎对应图 29 中的那些步骤，因此，这里只讨论由于生成灰度字体而不同的过程步骤。

这些不同的步骤是步(1)和(4)，以及步(16)和(17)，这些步骤

都被加上。

步(1)获取输入参数,并把灰度数据加到输入参数中。例如,参数为字符代码、字体形式、权值数据以及字符输出尺寸、灰度级数据、输出格 式和输出设备的特性。字符代码由某个字符代码系统所确定,例如*JIS*代码、移动*JIS*代码、*EUS*代码或*UNI*代码,由此预先标识一个目标系统。字体形式从预先被 加 入 该系统的数据中选择,例如明体、哥特体、圆哥特体或作为可选项加上的数据。

数据的内容同用来生成位图化字体的数据完全一样。不保存生成灰度字体 的特定数据。权值数据是关于字体形式的线粗细程度的数据,所提供的数据有“ 很细”、“细”、“标准”、“粗”以及“很粗”。

输出尺寸是表示将被输出的字体数据的实际尺寸的数据。x方向和y方向上的尺寸都被要求。灰度级数据指示将被准备的灰度字体的灰度级。根据输出设备的特性,灰度级被置为4、16或256级。

输出格式是所需字体的输出数据形式,包括轮廓座标数据输出,位图化字体输出,灰度级字体输出和1—点表达式形式。

例如,灰度字体的1—点表达式形式用来表示是否1字节表达1个点,或是否1字节表达2个或4个点。

输出设备的特性数据表示如何为显示器准备一种最佳灰度字体确定一个灰度级。

在步(4)中,通过引用灰度字体的输出尺寸和灰度级,对在步(2)中 读出的座标数据执行放大/缩小处理。若所需的输出尺寸为

(Ax, Ay) ，灰度级为 n，步(3)中得到的座标为(x,y)，经过放大/缩小处理后得到的座标为(X,Y)，所存数据的字符框架尺寸为(Mx,My)，则数据放大或缩小的表达式为：

$$(X,Y)=(x\times\sqrt{n}\times Ax/Mx,\ y\times\sqrt{n}\times Ay/My)$$

在步(16)和(17)中，利用步(15)准备好的位图化字体来准备灰度字体。

首先，在步(16)中，根据被用来作为一个输入参数的输出设备特性选择灰度转换表。对 16 个灰度级采用 4×4 的掩码，表示输出设备特性的值被预先放在该灰度字体转换表中，如图 42A 到 42C 所示。

图 42A 给出了输出设备的亮度特性一致的例子；图 42B 给出在点的中心亮度高而在其外围部分中亮度却较低的例子；图 42C 给出的则是在点的外围部分亮度高而在其中心却亮度较低的例子。从中选择最适合于输出设备亮度特性的表格。

接着，步(17)利用步(16)中选择的表格准备好一个灰度字体。现在结合图 43 说明这一处理过程。

图 43 给出步(15)得到的位图化字体。通过用灰度级 n 乘以已经准备好垂直和水平的长度。因此，垂直和水平长度被$\sqrt{n}$除，并提取$\sqrt{n}\times\sqrt{n}$ 的网格正方形。每个网格正方形的位值乘以被送入步(16)得到的表格中的对应网格正方形中的一个值。

这些乘积之和为目标点提供一个灰度值。图 42 给出了一个输出具有 16 个灰度级的灰度字体的例子。对 4×4 网格正方形选择图 42*B* 中的表。对所有的网格正方形执行上述的处理过程，因此产生如图 44 所示的灰度字体。

接着，步(18)根据输出格式存放灰度字体并把数据返回到请求一方。如果被请求的输出格式为 1 字节对 1 点，则网格正方形的值被压缩为 1 字节并被保存起来。若被请求的输出格式把相邻的点压缩为 1 字节，则 1 个点被压缩为 4 位并被存储，接着，数据被返回到请求一方。然后终止该处理过程。

如上所述，根据本发明的一个实施例，转换装置根据被确定的字符权值变换轮廓数据的座标值，然后，发生装置根据其座标已经被变换的轮廓数据产生一个或粗或细的字模，因此利用少数几个字符数据就能产生具有不同权值的一个粗字符或细字符。

此外，根据本发明的另一个实施例，通过参考和某个目标的座标相邻的两个点的座标值，就能确定被变换的座标值，因此可以产生一个被很好平衡的粗的或细的字模。

另外，根据本发明的另一个实施例，可以从轮廓数据中提取外部轮廓数据和内部轮廓数据，并且能产生一个令人满意的粗的或细的字模。

而且，根据本发明的另一个实施例，利用被提取的外部轮廓数据和内部轮廓数据，就能确定输入粗的字符权值或输入细的字符权

值，并能产生令人满意的粗的或细的字模。

此外，根据本发明的另一个实施例，利用被提取的外部和内部轮廓数据，就能确定 x 方向和 y 方向的独立字符权值，并能产生能有效地表示每种字体形式的特性的粗的或细的字模。

另外，根据本发明的另一个实施例，根据被转换的轮廓数据就能输出位图化字体，轮廓座标数据或灰度字体，并且能按某种适当的数据形式把粗的或细的字符数据提供给各种输出装置。

而且，根据本发明的另一个实施例，通过利用将被产生的字符的权值以及被存储的字符数据的权值之间的差确定一个参数，转换装置利用该被确定的参数来转换轮廓数据的座标值，而生成装置则利用得出的轮廓数据来产生一个或粗或细的字模，因此，字符数据源被用来产生最佳的粗的或细的字符数据。

另外，根据本发明的另一个实施例，根据其权值接近被确定的字符权值的字符数据确定一个参数，就能产生更精确的粗的或细的字符数据。

因此，根据本发明，只需一个较小的存储量，就能产生具有各种数权值的字体形式的字符数据。

图 1

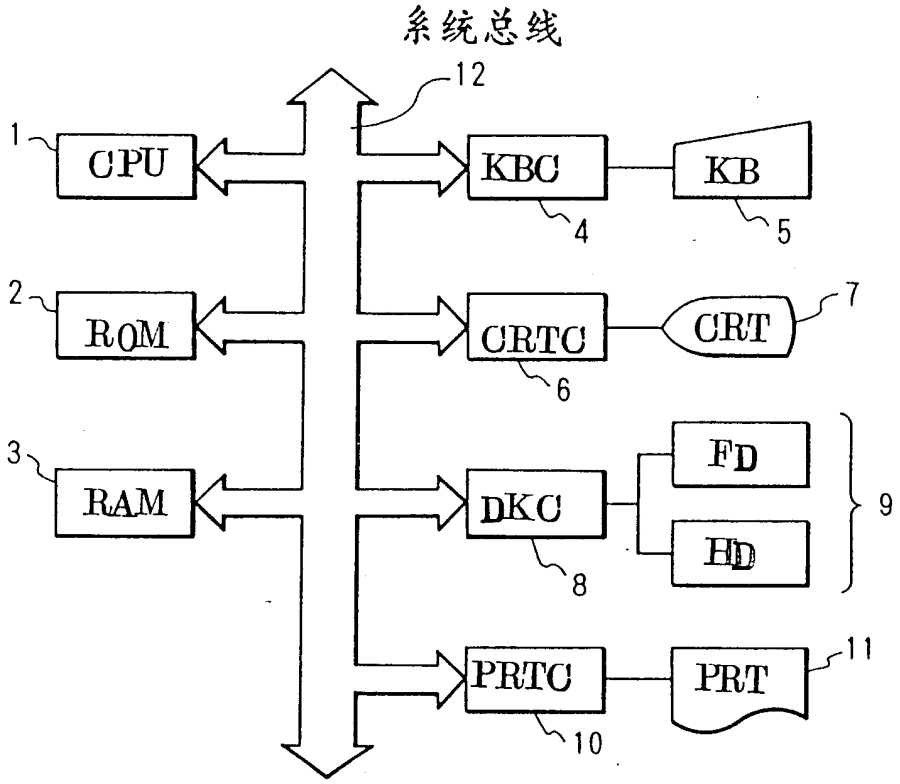

系统总线
12
1
CPU
KBC
KB
4
5
2
ROM
CRTC
CRT
7
6
3
RAM
DKC
FD
HD
9
8
PRTC
PRT
11
10

图 2

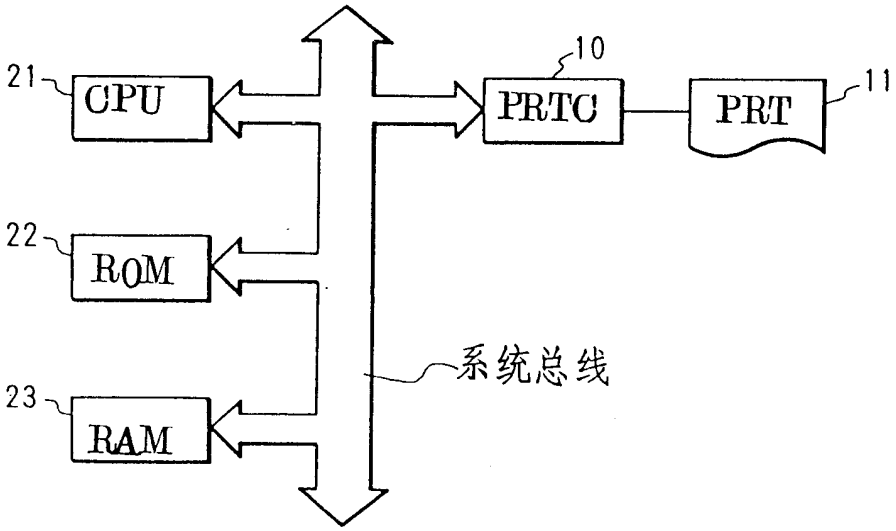

10
21
CPU
PRTC
PRT
11
22
ROM
系统总线
23
RAM

图 3

图 3A

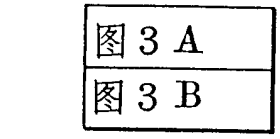


图3A
图3B

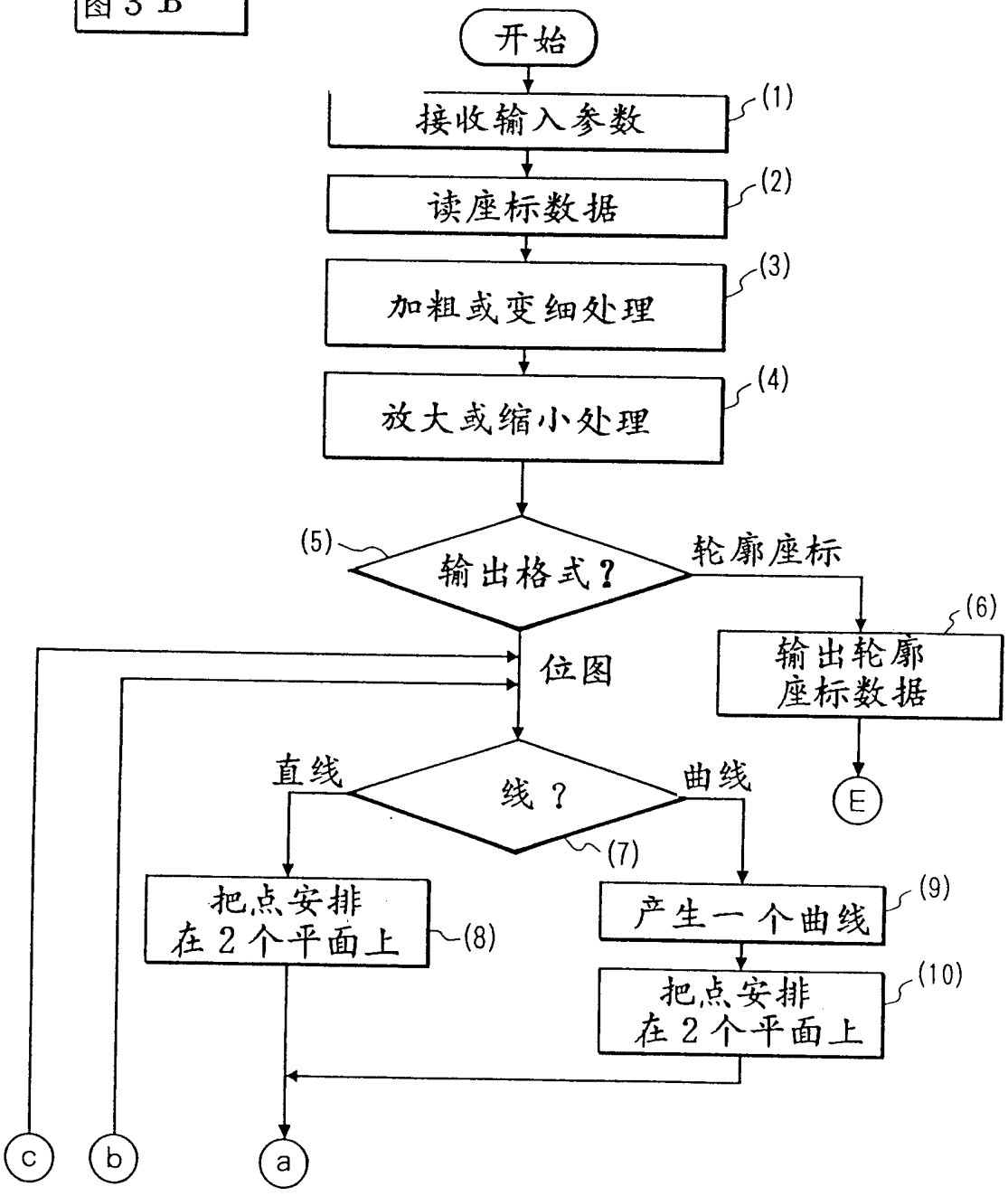

开始
接收输入参数
(1)
读座标数据
(2)
加粗或变细处理
(3)
放大或缩小处理
(4)
(5)
输出格式？
轮廓座标
(6)
输出轮廓
座标数据
位图
直线
线？
曲线
(7)
E
把点安排
在2个平面上
(8)
产生一个曲线
(9)
把点安排
在2个平面上
(10)
c
b
a

图 3B

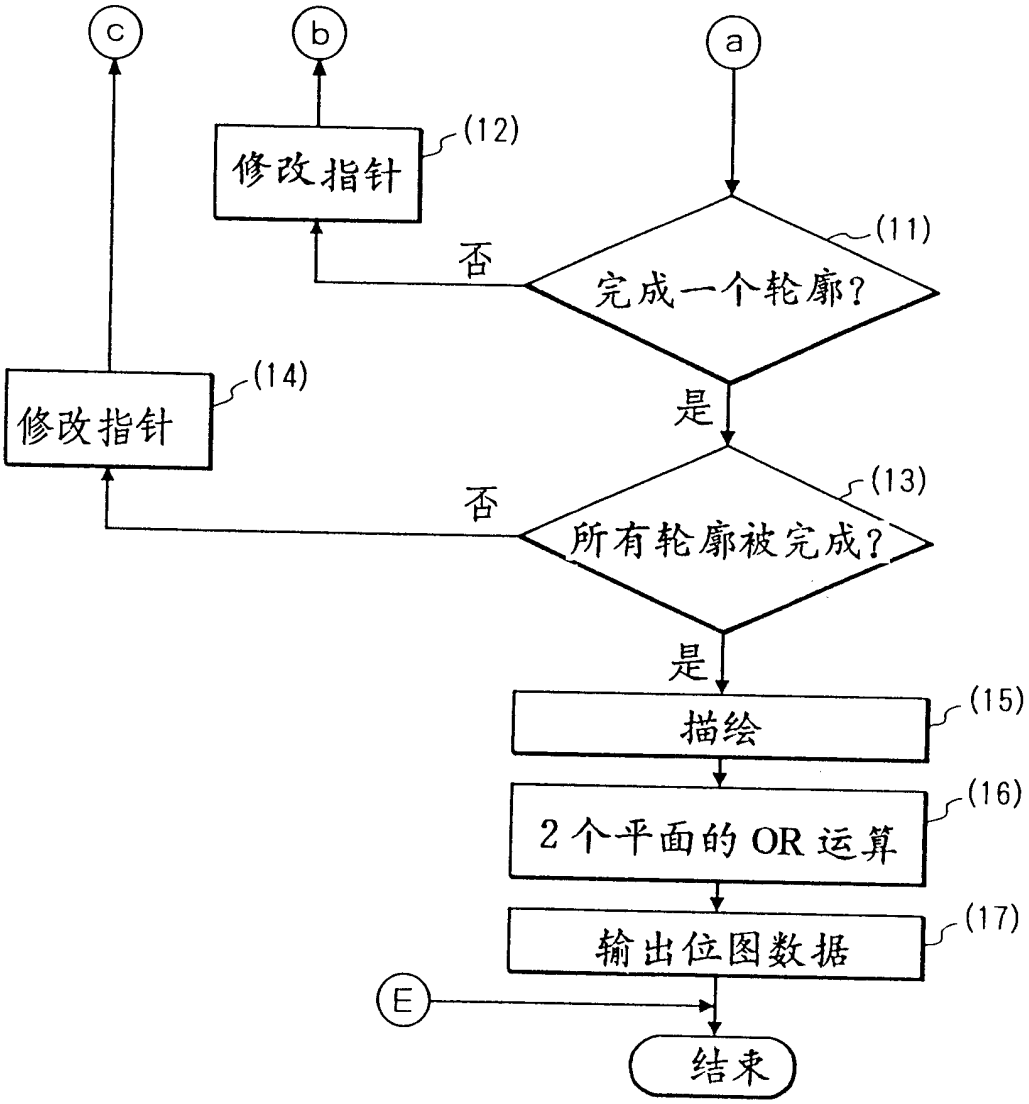
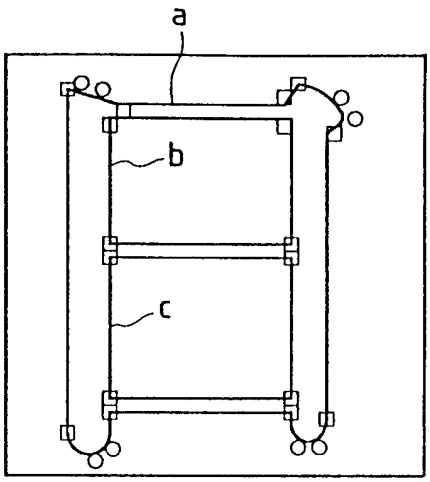

c
b
a
修改指针
(12)
否
完成一个轮廓?
(11)
修改指针
(14)
是
否
所有轮廓被完成?
(13)
是
描绘
(15)
2个平面的OR运算
(16)
输出位图数据
(17)
E
结束

图 4



□ 直线的终点

○ 曲线的中点

图 5

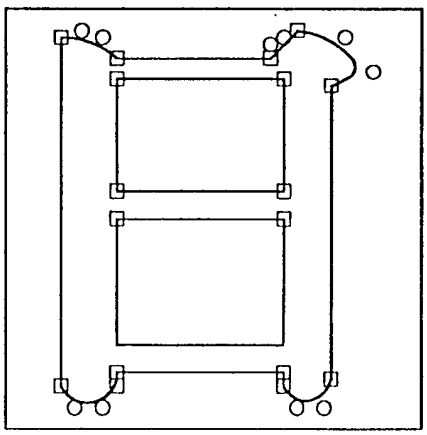


图 6

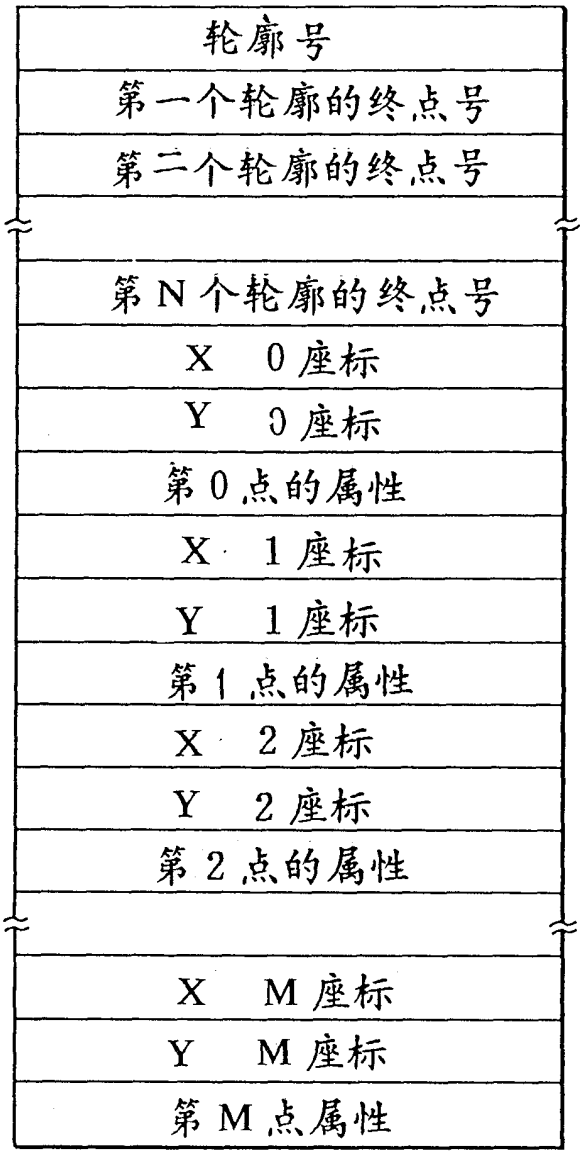

轮廓号
第一个轮廓的终点号
第二个轮廓的终点号
第N个轮廓的终点号
X 0座标
Y 0座标
第0点的属性
X 1座标
Y 1座标
第1点的属性
X 2座标
Y 2座标
第2点的属性
X M座标
Y M座标
第M点属性

图 7

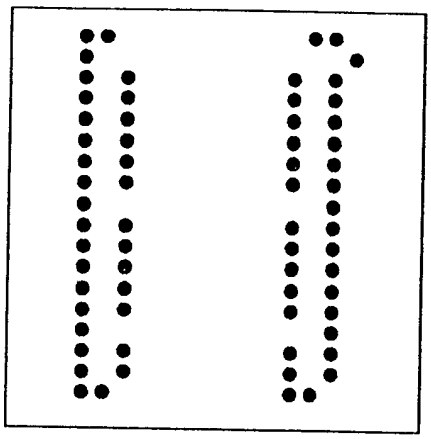


图 8

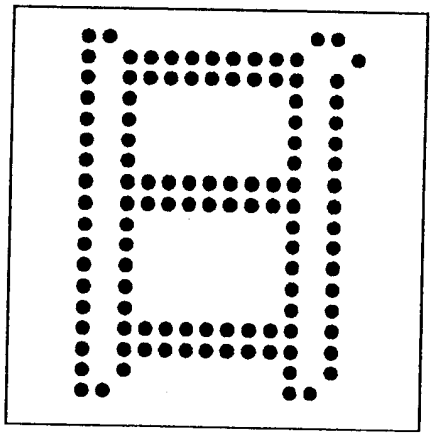


图 9

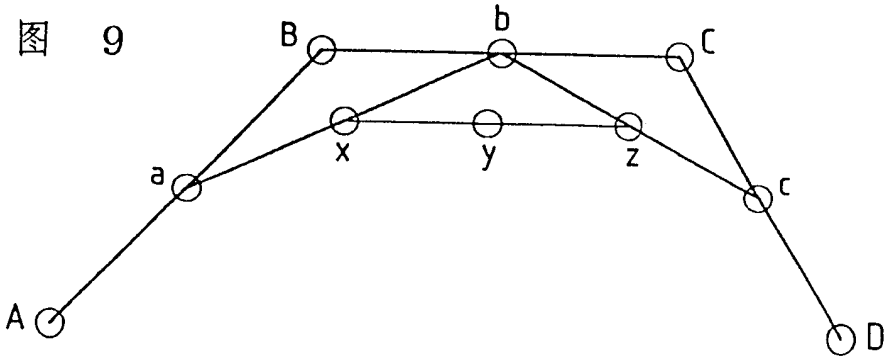


图 10

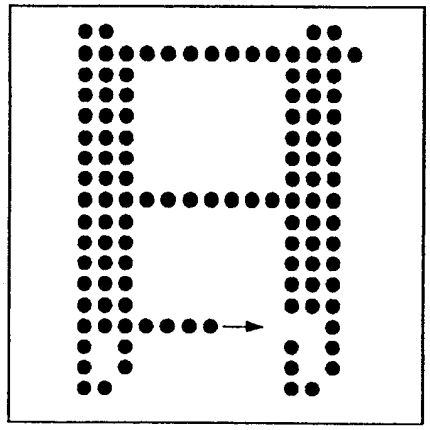


图 11

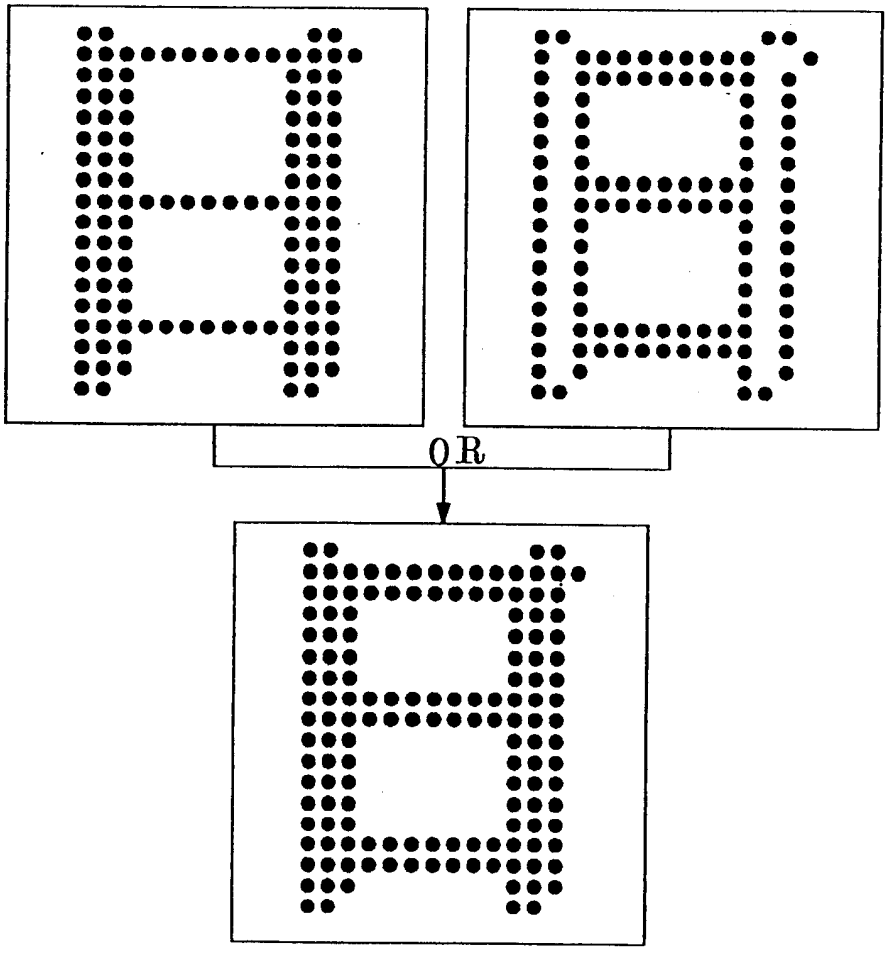


图 12

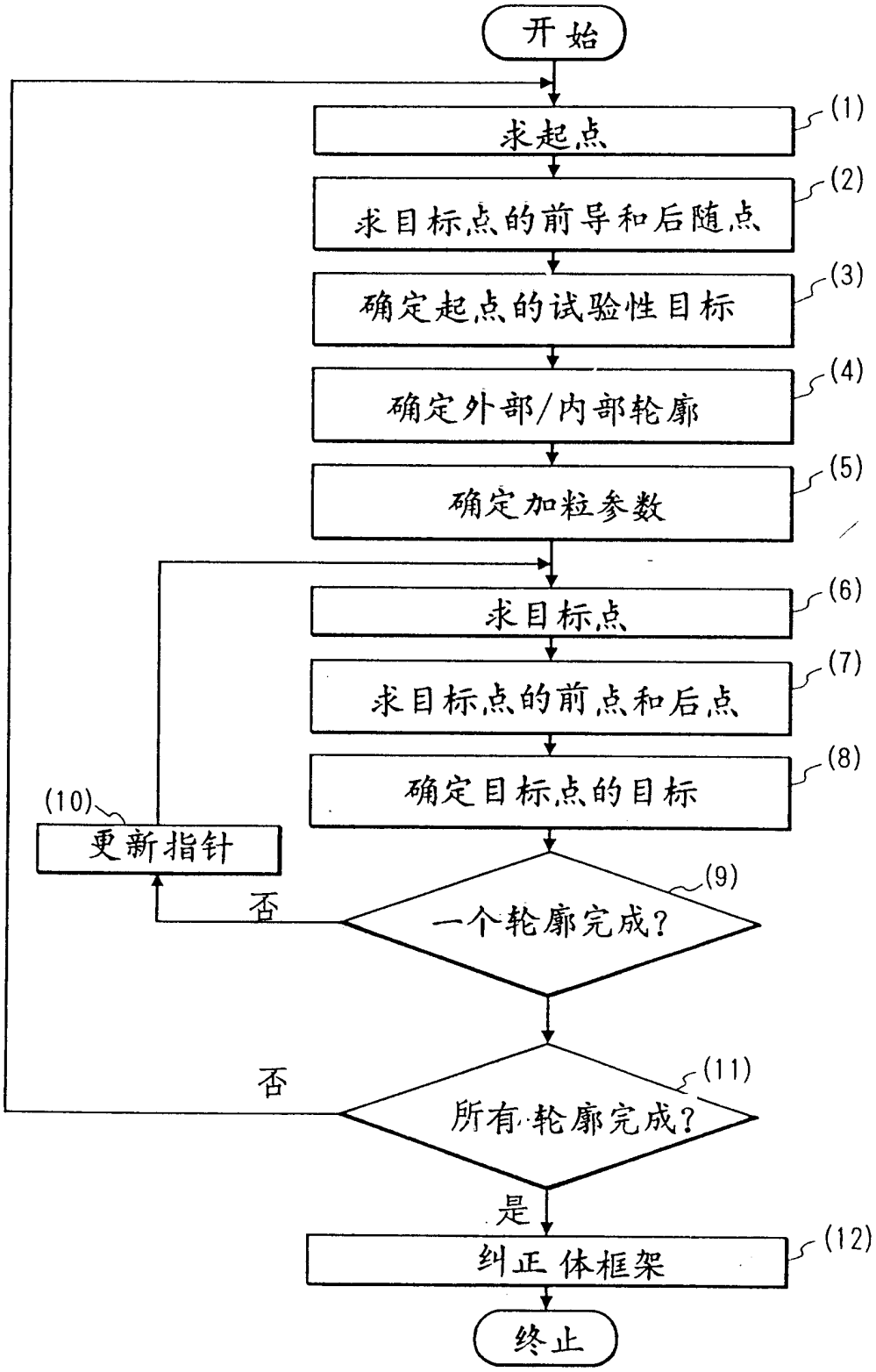

开始
求起点
(1)
求目标点的前导和后随点
(2)
确定起点的试验性目标
(3)
确定外部/内部轮廓
(4)
确定加粒参数
(5)
求目标点
(6)
求目标点的前点和后点
(7)
确定目标点的目标
(8)
(10)
更新指针
否
一个轮廓完成?
(9)
否
所有轮廓完成?
(11)
是
纠正体框架
(12)
终止

图 13

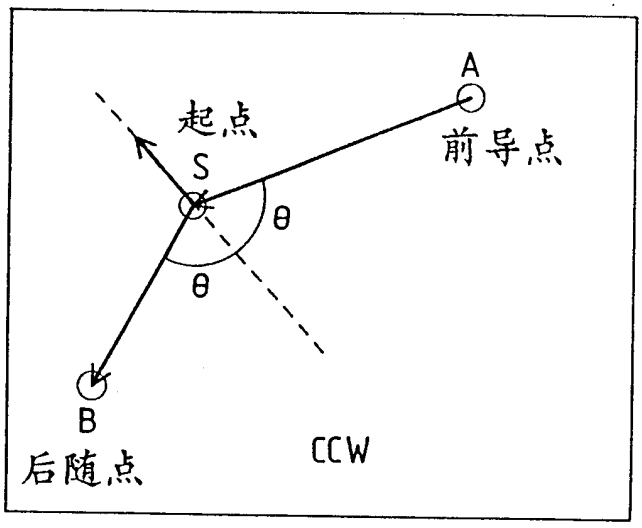


图 14

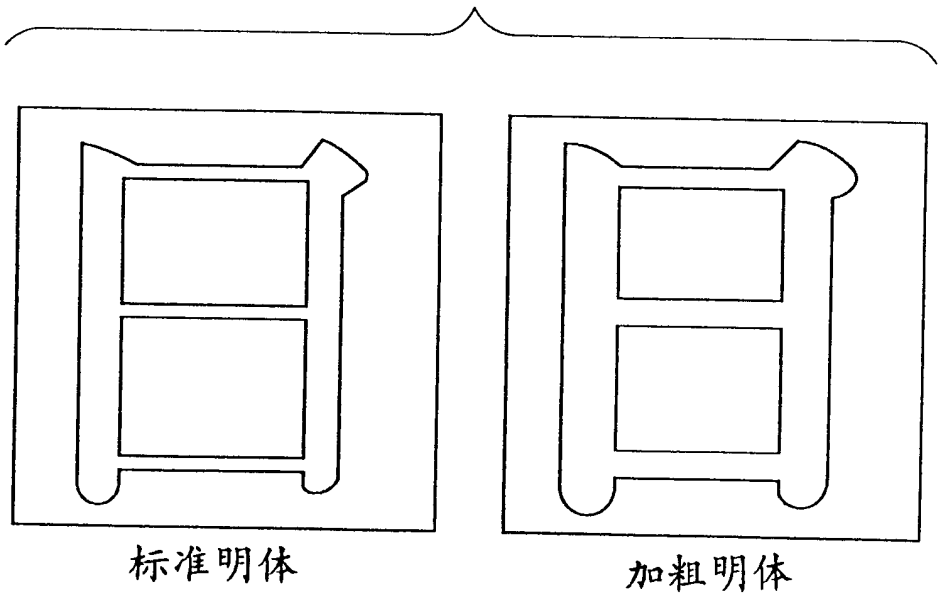


图 15

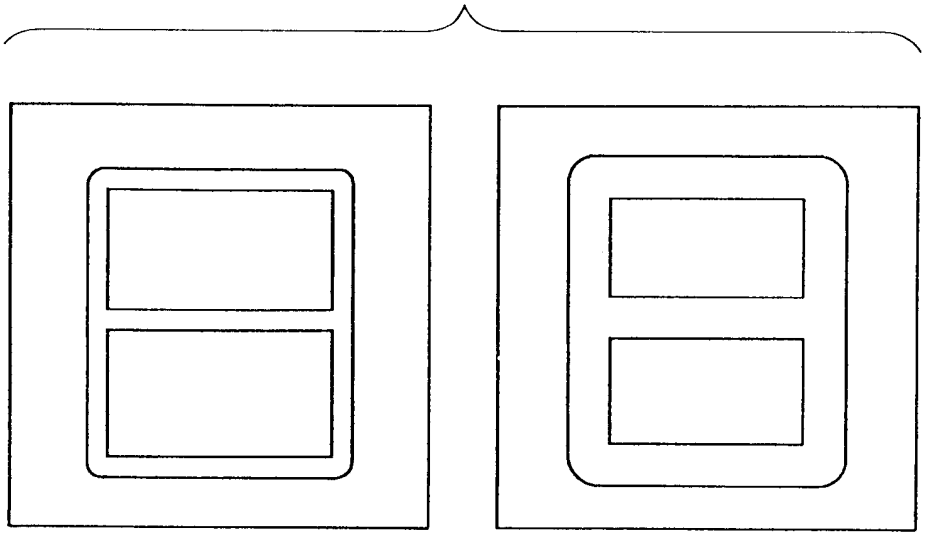


图 17

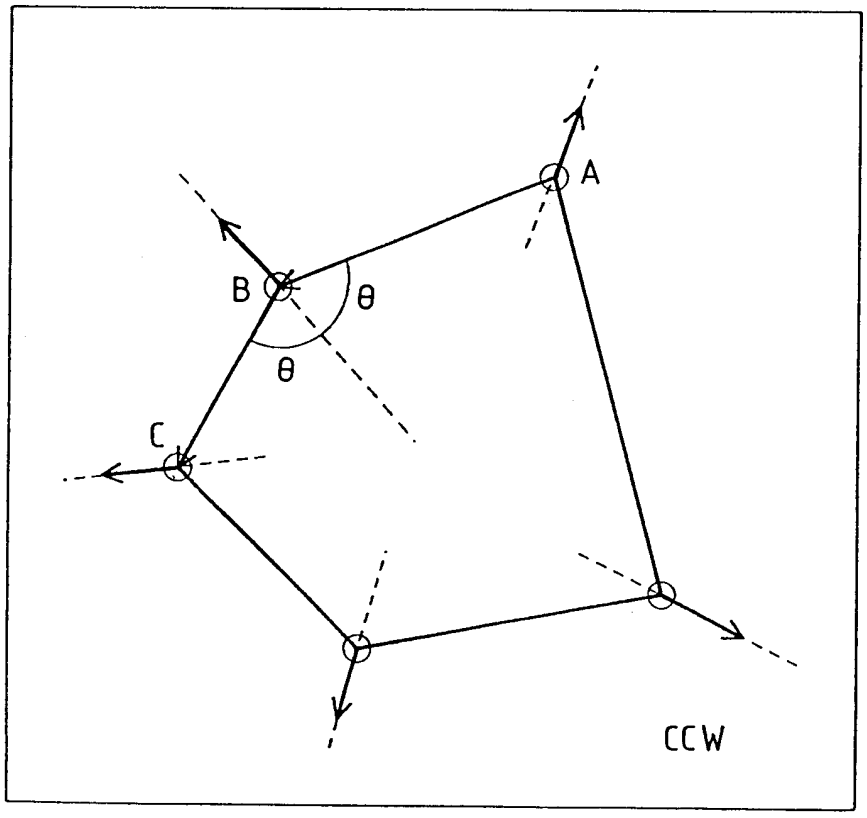


图 16

字体形式	明体				圆哥特式			
轮廓	外部		内部		外部		内部	
x/y	x	y	x	y	x	y	x	y
很细	6	10	5	8	10	10	8	8
细	9	15	7	13	15	15	13	13
标准	12	20	10	18	20	20	18	18
粗	15	25	13	23	25	25	23	23
很粗	18	30	16	28	30	30	28	28

图 18

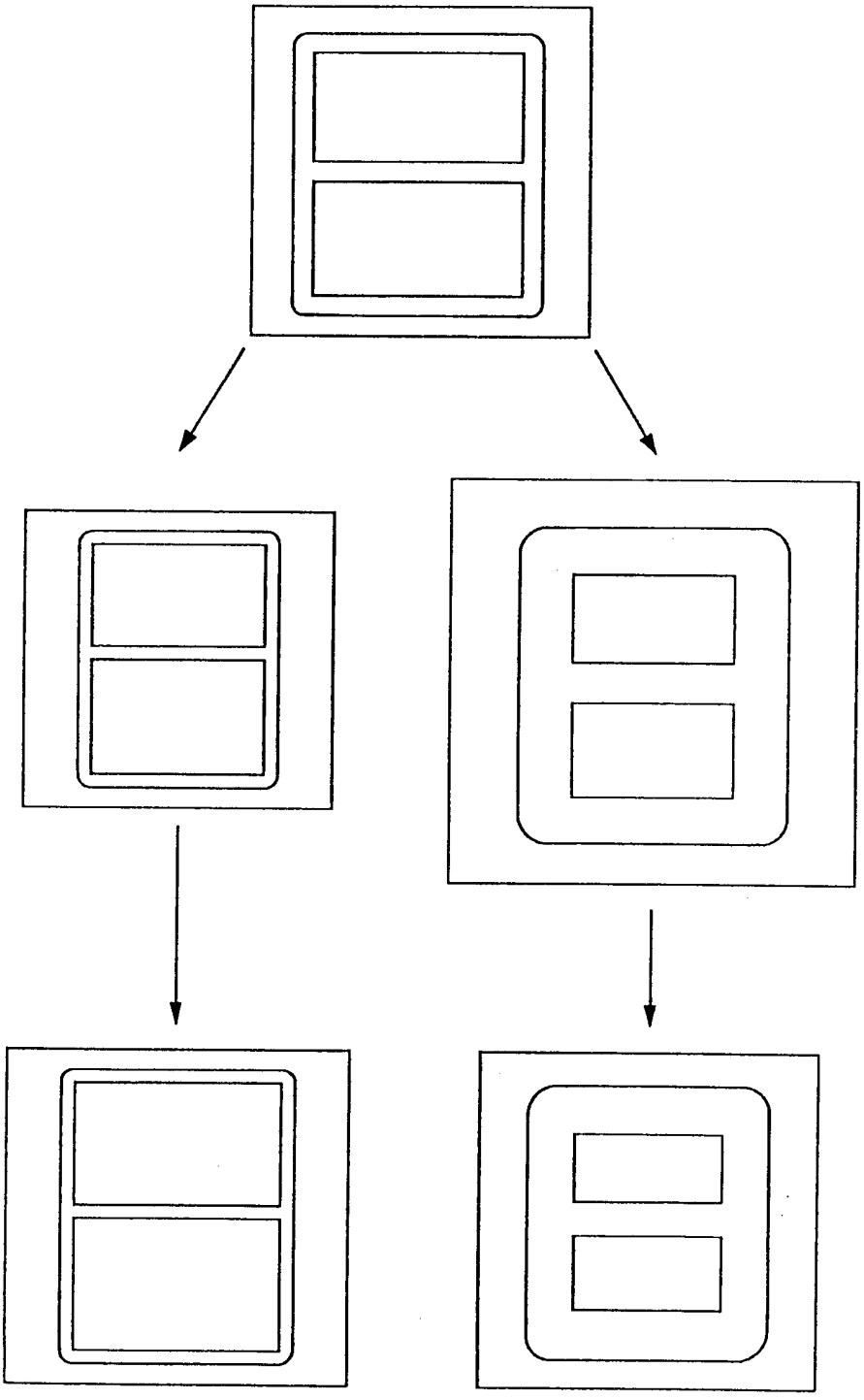


图 19

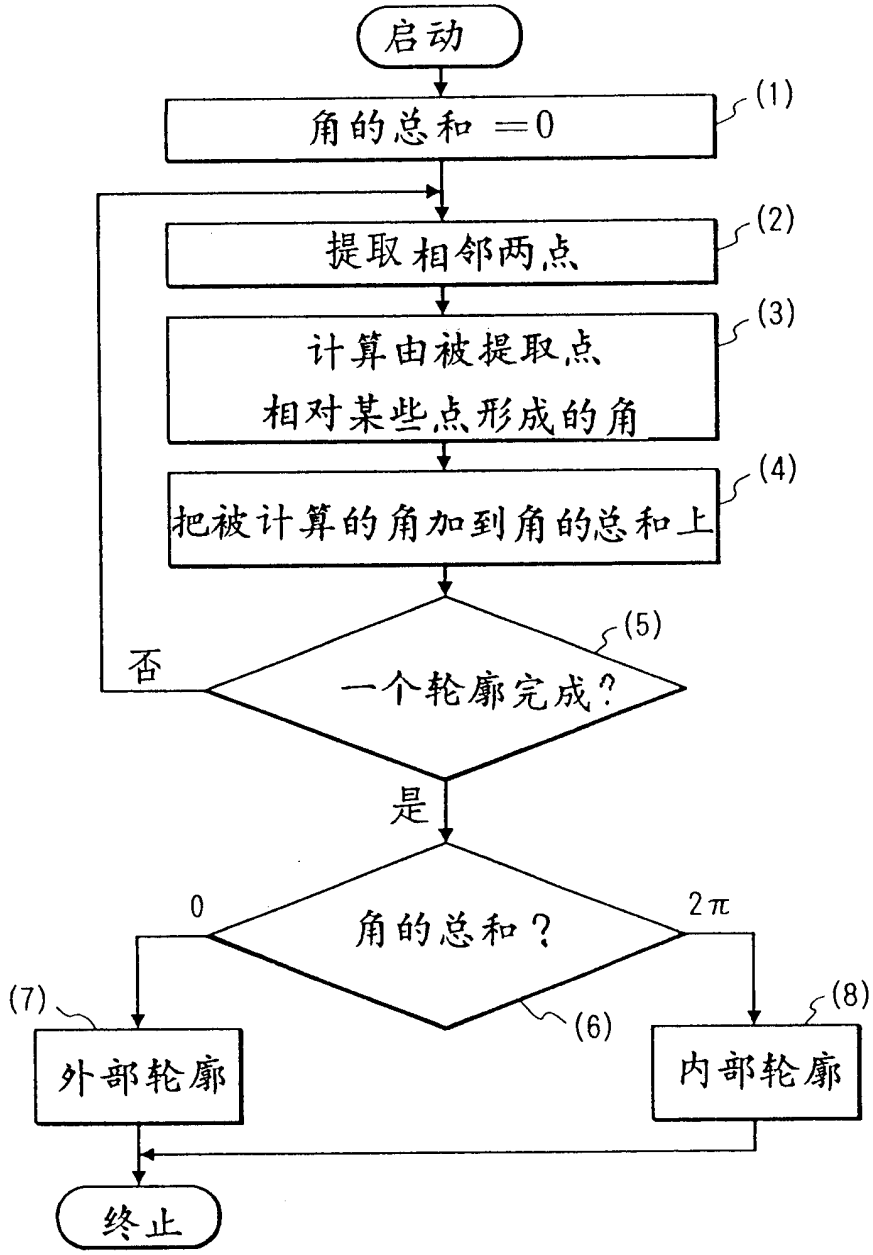

启动
角的总和 =0
(1)
提取相邻两点
(2)
计算由被提取点
相对某些点形成的角
(3)
把被计算的角加到角的总和上
(4)
一个轮廓完成?
(5)
否
是
角的总和?
0
2π
(6)
(7)
外部轮廓
(8)
内部轮廓
终止

图 20

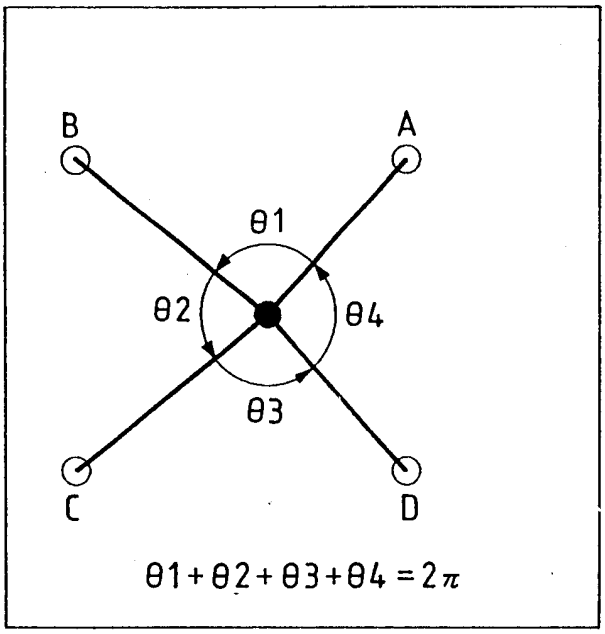


图 21

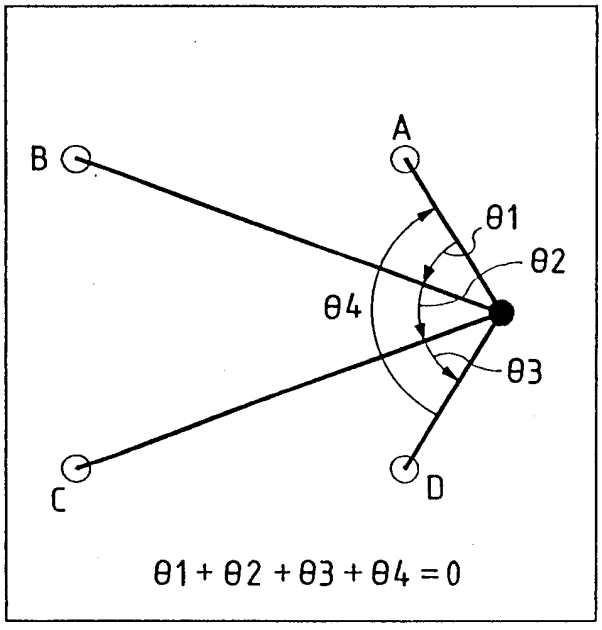


图 22

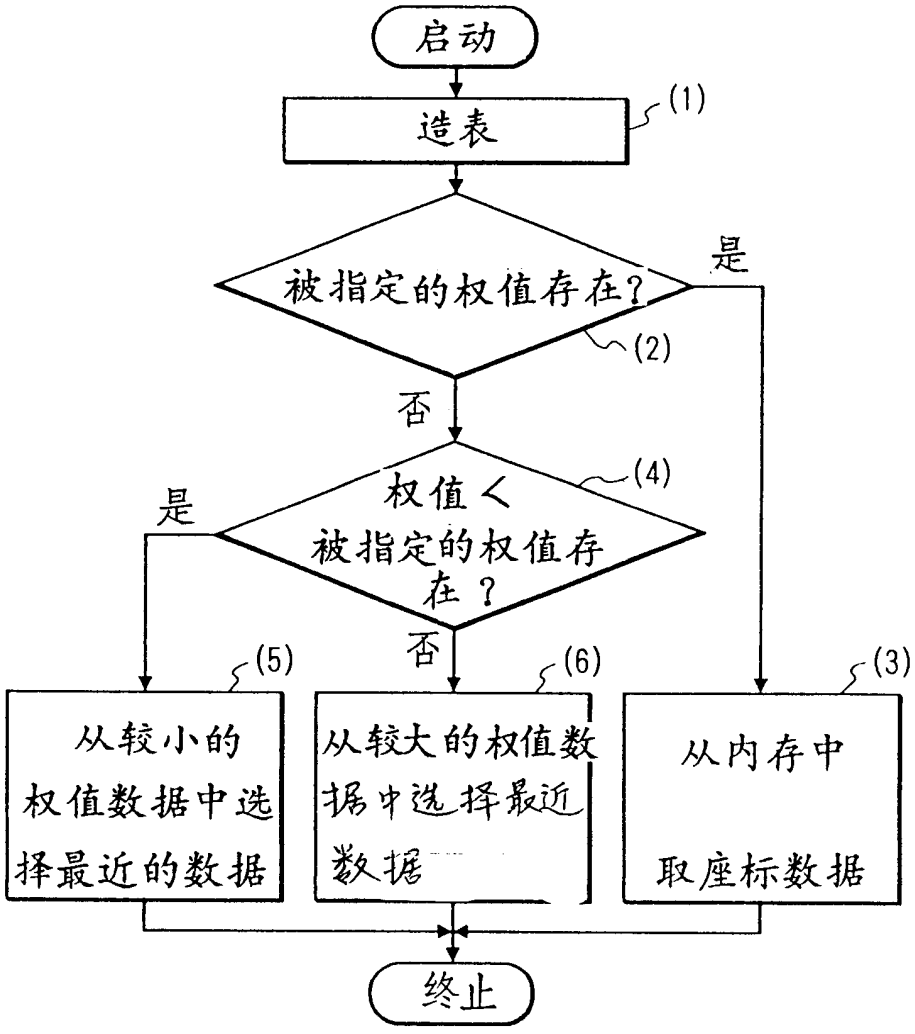

启动
造表
(1)
被指定的权值存在?
是
(2)
否
权值<被指定的权值存在?
(4)
是
否
(5)
(6)
(3)
从较小的权值数据中选择最近的数据
从较大的权值数据中选择最近数据
从内存中取座标数据
终止

图 23A

字体形式	明体					圆哥特体					
轮廓/存在	外部		内部		存在	外部		内部		存在	
x/y	x	y	x	y		x	y	x	y		
权值很细	3	5	2	5		5	5	3	3		
细 2	5	10	4	8		10	10	8	8		
标准细 3	7	15	6	13	○	15	15	13	13		
标准 4	10	20	9	18		20	20	18	18		
标准粗 5	12	25	11	23		25	25	23	23	○	
粗 6	14	30	13	28		30	30	28	28		
很粗 7	16	35	15	33	○	35	35	33	33		

图 23B

字体形式	斜哥特式					BLOCK					
轮廓/存在	外部		内部		存在	外部		内部		存在	
x/y	x	y	x	y		x	y	x	y		
权值很细 1	5	5	3	3		3	5	3	5		
细 2	10	10	8	8		6	10	5	8		
标准细 3	15	15	13	13		9	15	7	13		
标准 4	20	20	18	18		12	20	10	18	○	
标准粗 5	25	25	23	23		15	25	12	23		
粗 6	30	30	28	28	○	18	30	14	28		
很粗 7	35	35	33	33		21	35	16	33	○	

图 24

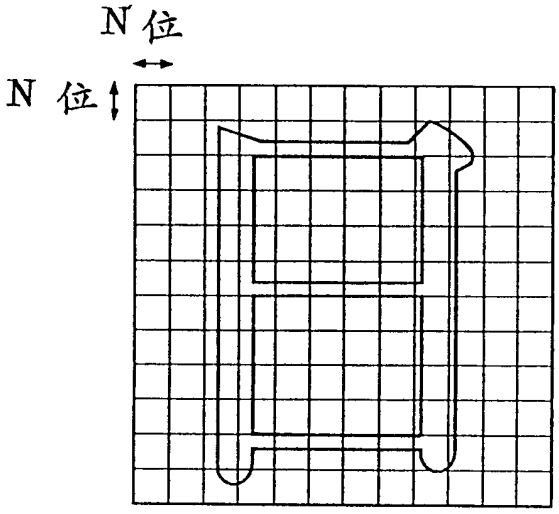


图 26A

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

图 26B

0.8	0.8	0.8	0.8
0.8	1.6	1.6	0.8
0.8	1.6	1.6	0.8
0.8	0.8	0.8	0.8

图 26C

1.1	1.1	1.1	1.1
1.1	0.7	0.7	1.1
1.1	0.7	0.7	1.1
1.1	1.1	1.1	1.1

图 25

图25A
图25B

图 25A

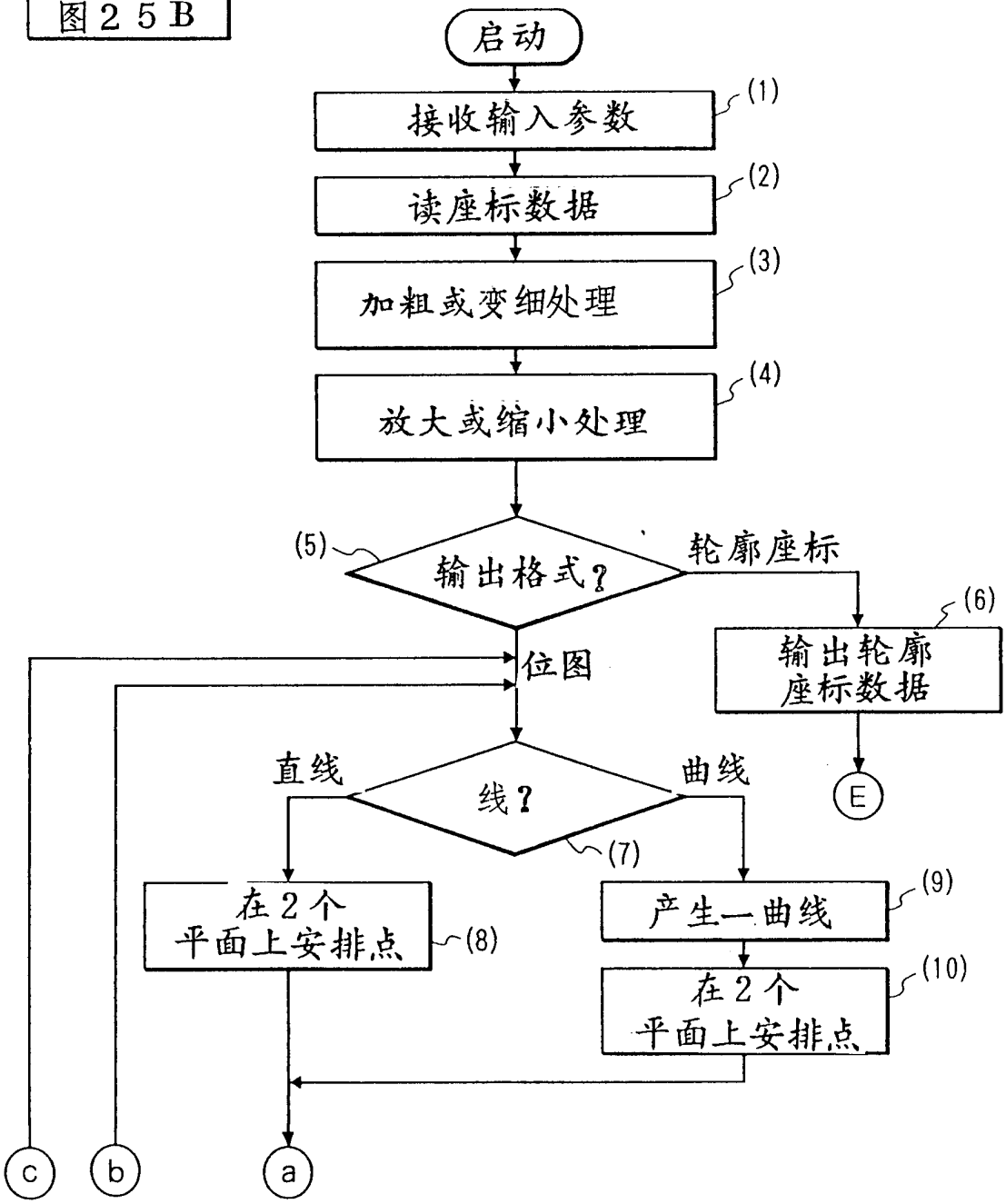


图 25B

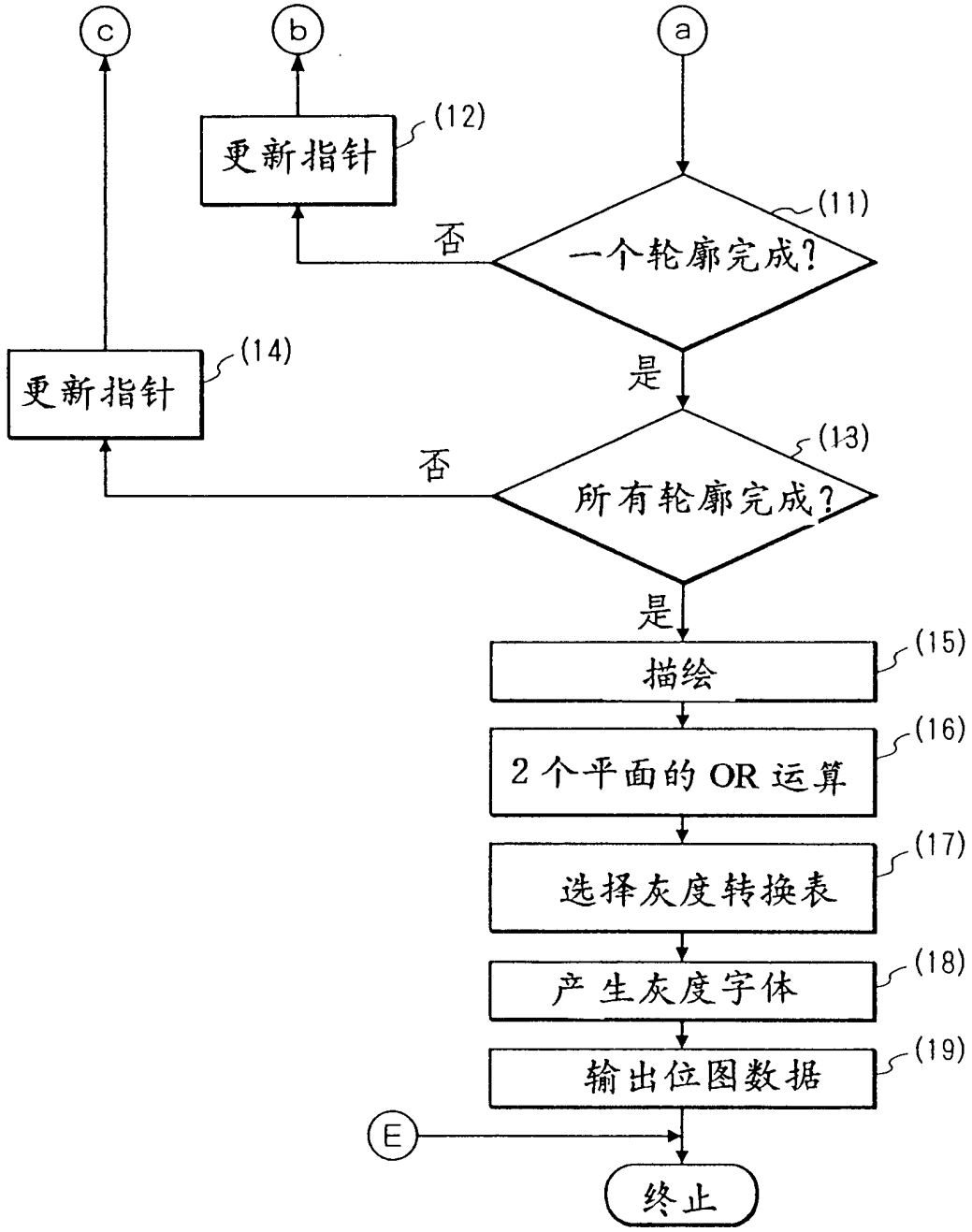

c
b
a
更新指针
(12)
否
一个轮廓完成?
(11)
更新指针
(14)
是
否
所有轮廓完成?
(13)
是
描绘
(15)
2个平面的OR运算
(16)
选择灰度转换表
(17)
产生灰度字体
(18)
输出位图数据
(19)
E
终止

图 27

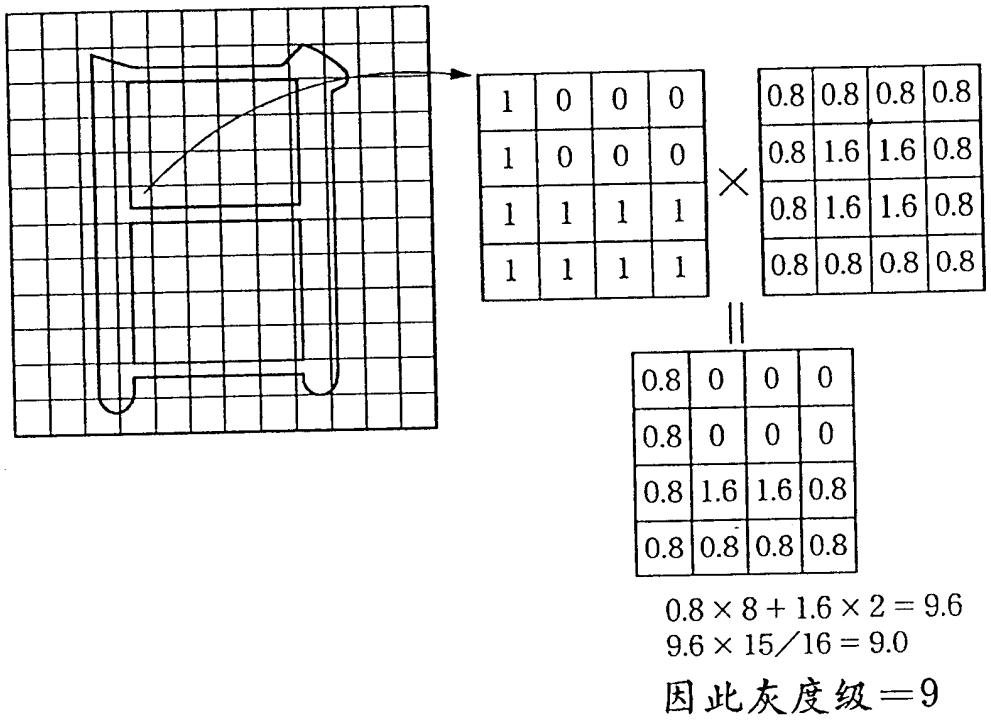


图 28

0	0	0	0	0	0	0	0	0	0	0	0
0	0	10	9	8	8	8	9	13	5	0	0
0	0	12	3	0	0	0	0	12	4	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	9	8	8	8	9	13	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	9	8	8	8	8	12	2	0	0
0	0	2	1	0	0	0	0	0	0	0	0

图２９

图２９Ａ
图２９Ｂ

图 ２９Ａ

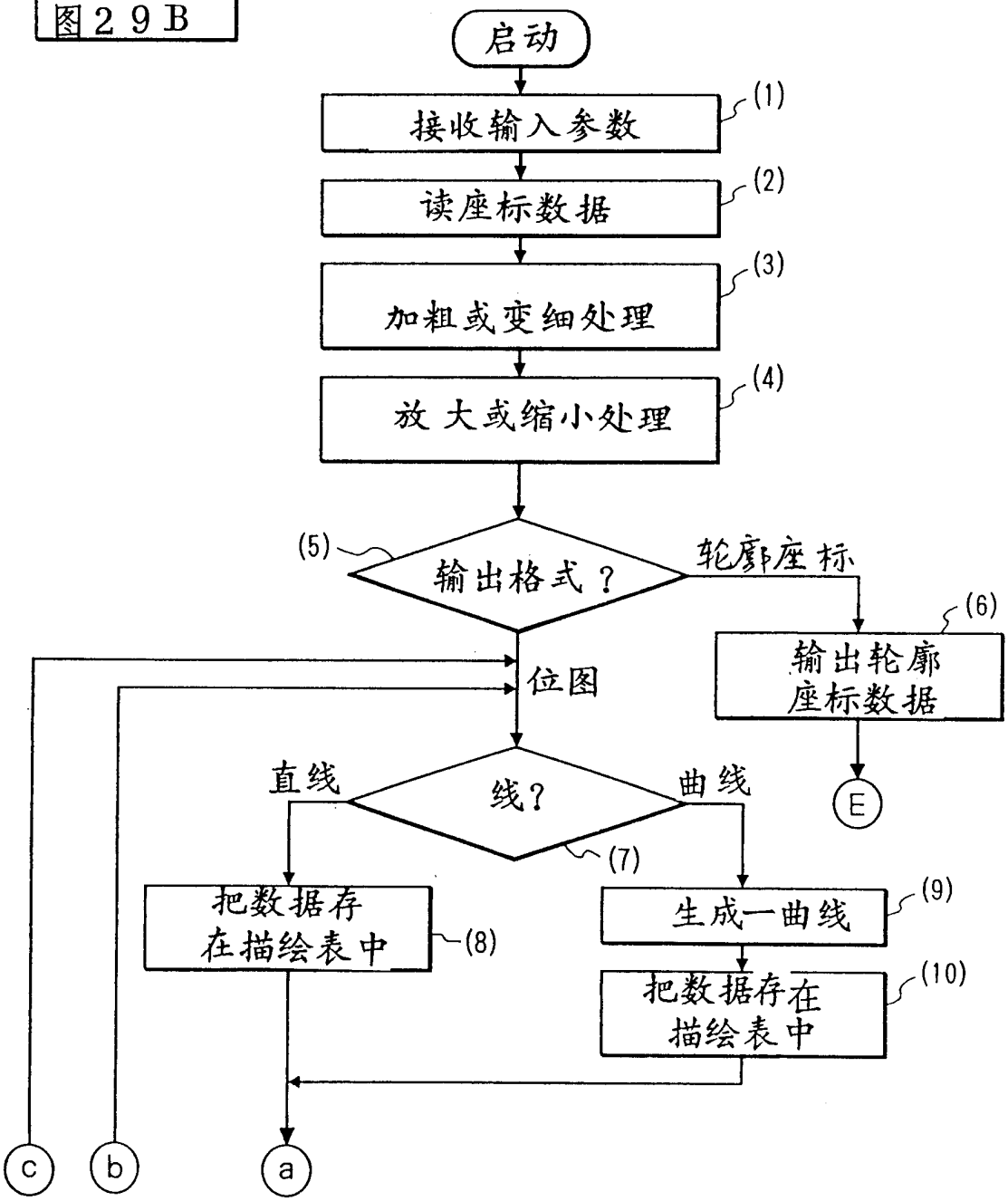


图２９Ｂ

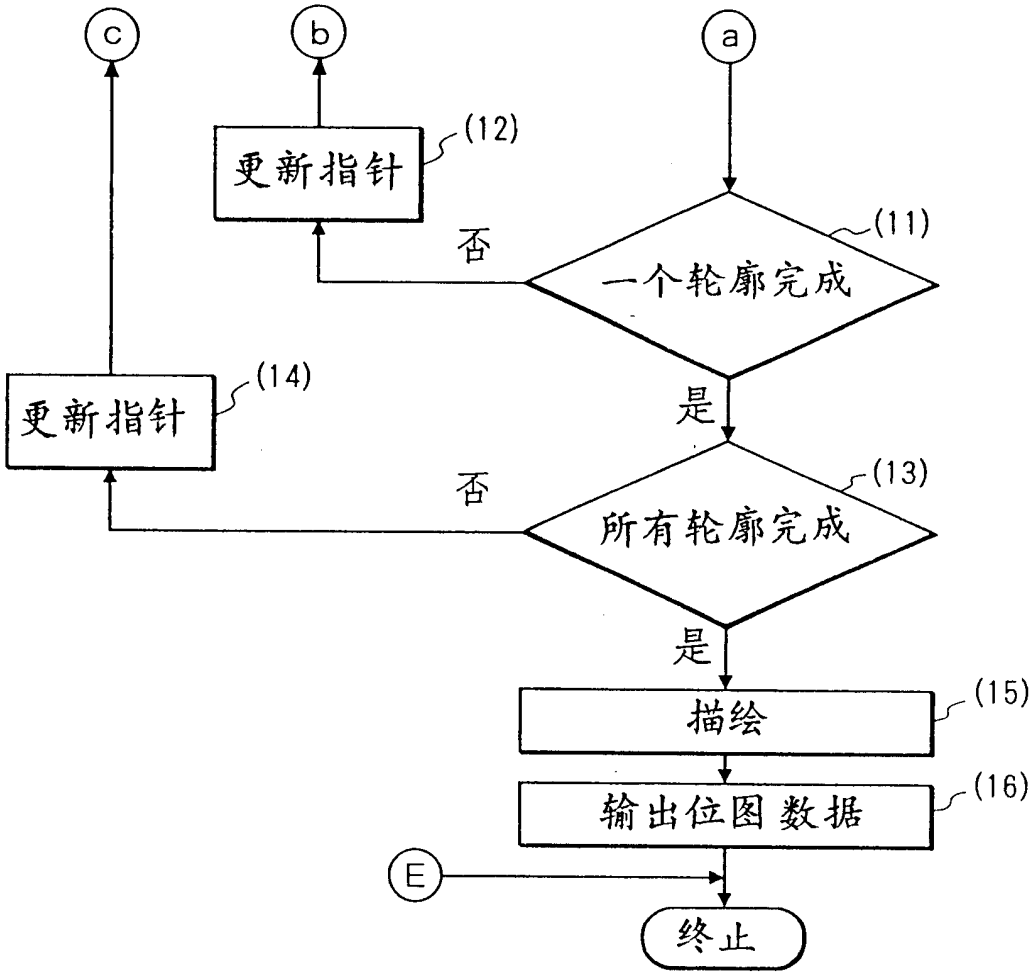

c
b
a
更新指针
(12)
否
一个轮廓完成
(11)
更新指针
(14)
是
否
所有轮廓完成
(13)
是
描绘
(15)
输出位图数据
(16)
E
终止

图 30

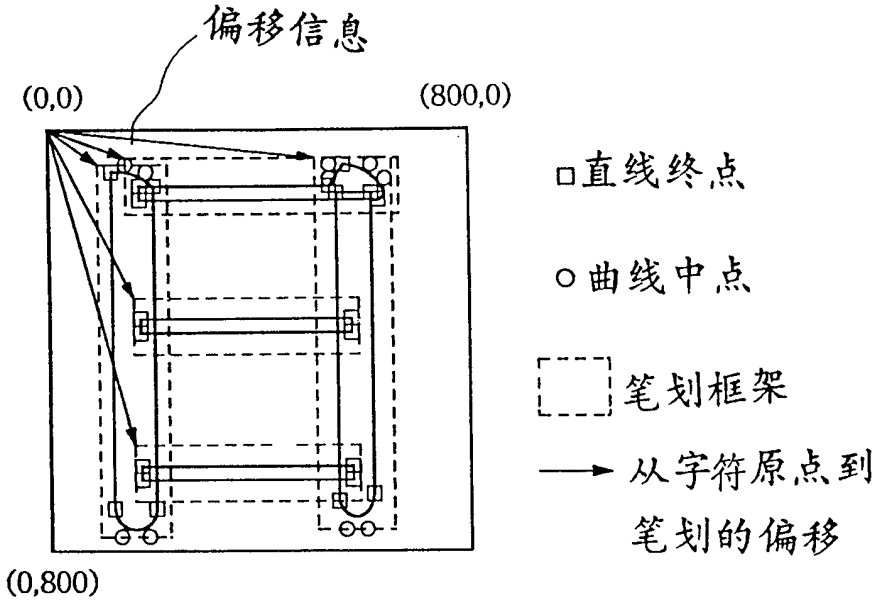


图 31

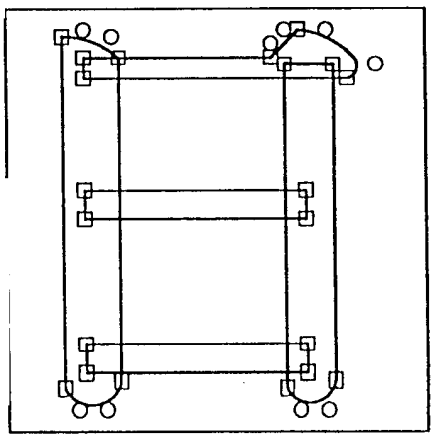


图 32

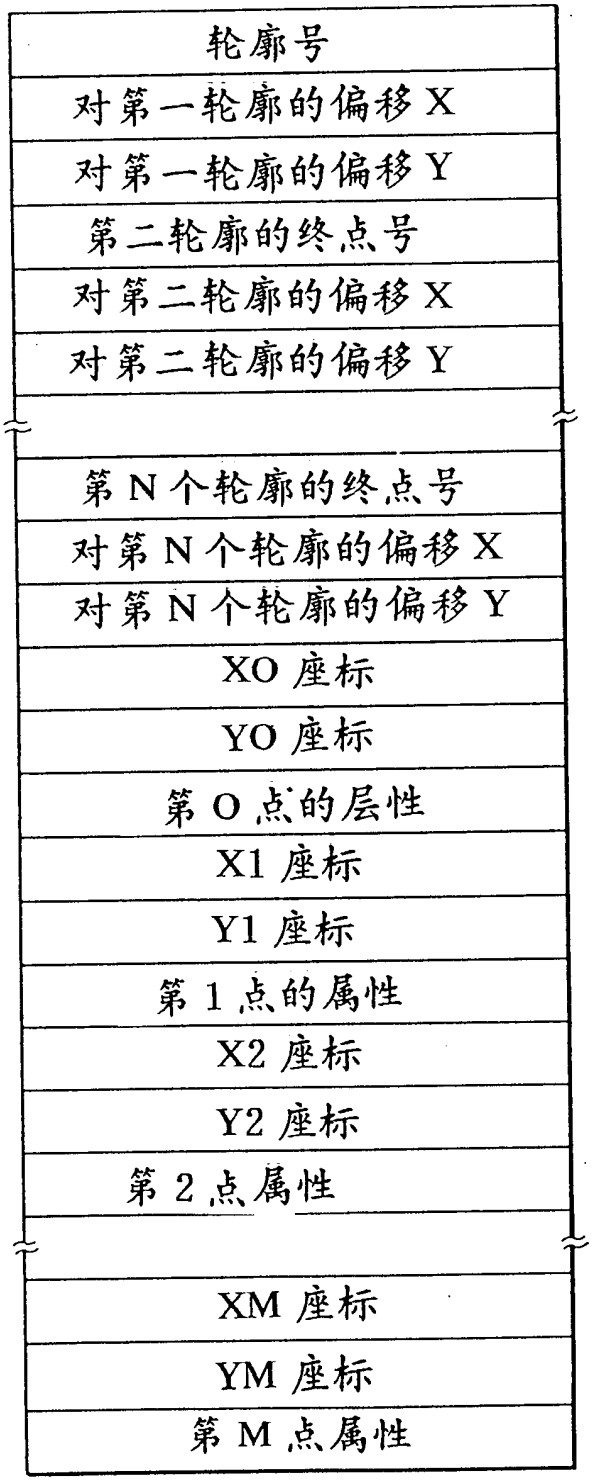

轮廓号
对第一轮廓的偏移 X
对第一轮廓的偏移 Y
第二轮廓的终点号
对第二轮廓的偏移 X
对第二轮廓的偏移 Y
第 N 个轮廓的终点号
对第 N 个轮廓的偏移 X
对第 N 个轮廓的偏移 Y
XO 座标
YO 座标
第 O 点的层性
X1 座标
Y1 座标
第 1 点的属性
X2 座标
Y2 座标
第 2 点属性
XM 座标
YM 座标
第 M 点属性

图 33

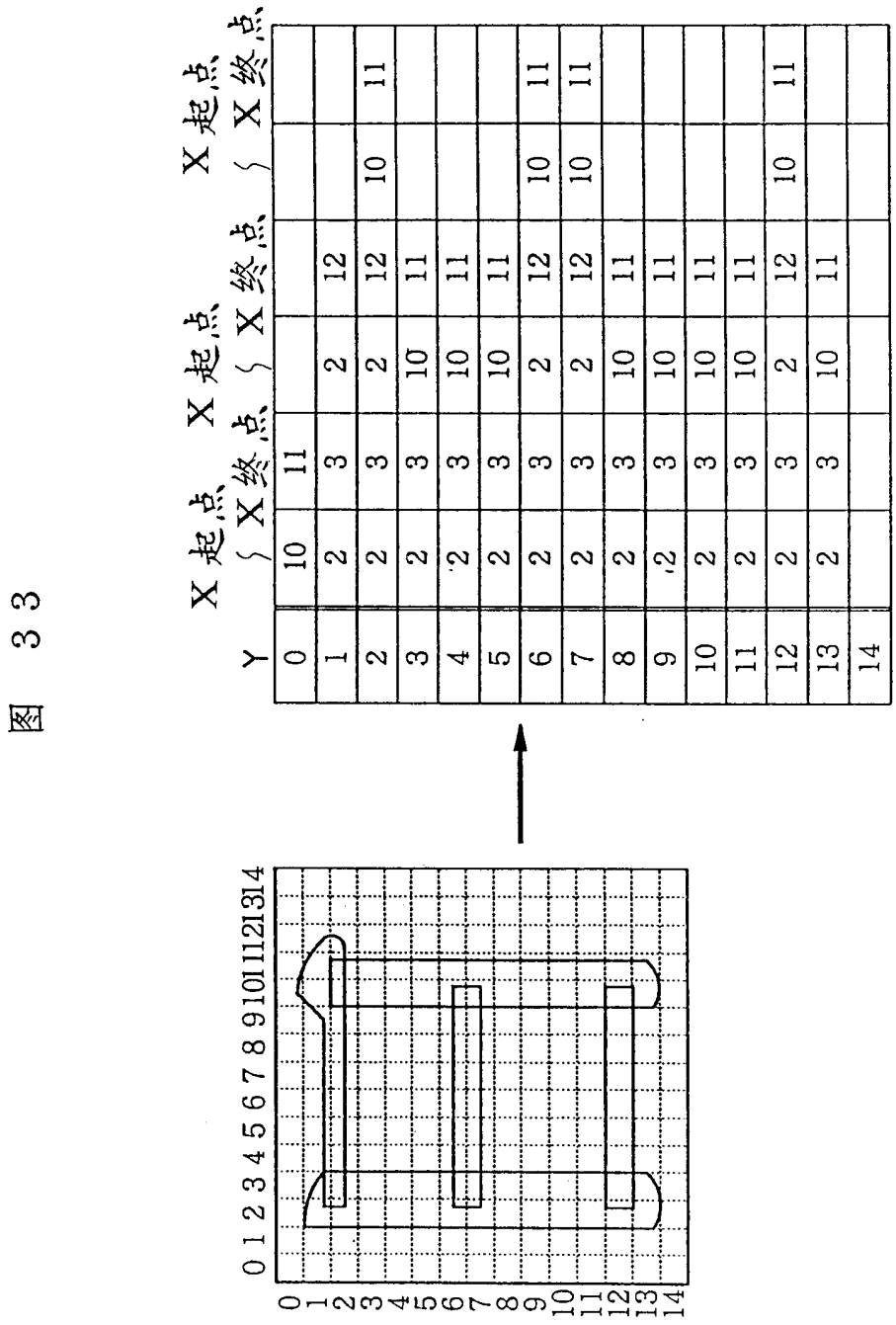

X起点 X起点 X起点
Y ∫X终点 ∫X终点 ∫X终点
0 10 11
1 2 3 2 12
2 2 3 2 12 10 11
3 2 3 10 11
4 2 3 10 11
5 2 3 10 11
6 2 3 2 12 10 11
7 2 3 2 12 10 11
8 2 3 10 11
9 2 3 10 11
10 2 3 10 11
11 2 3 10 11
12 2 3 2 12 10 11
13 2 3 10 11
14
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

图 3 4

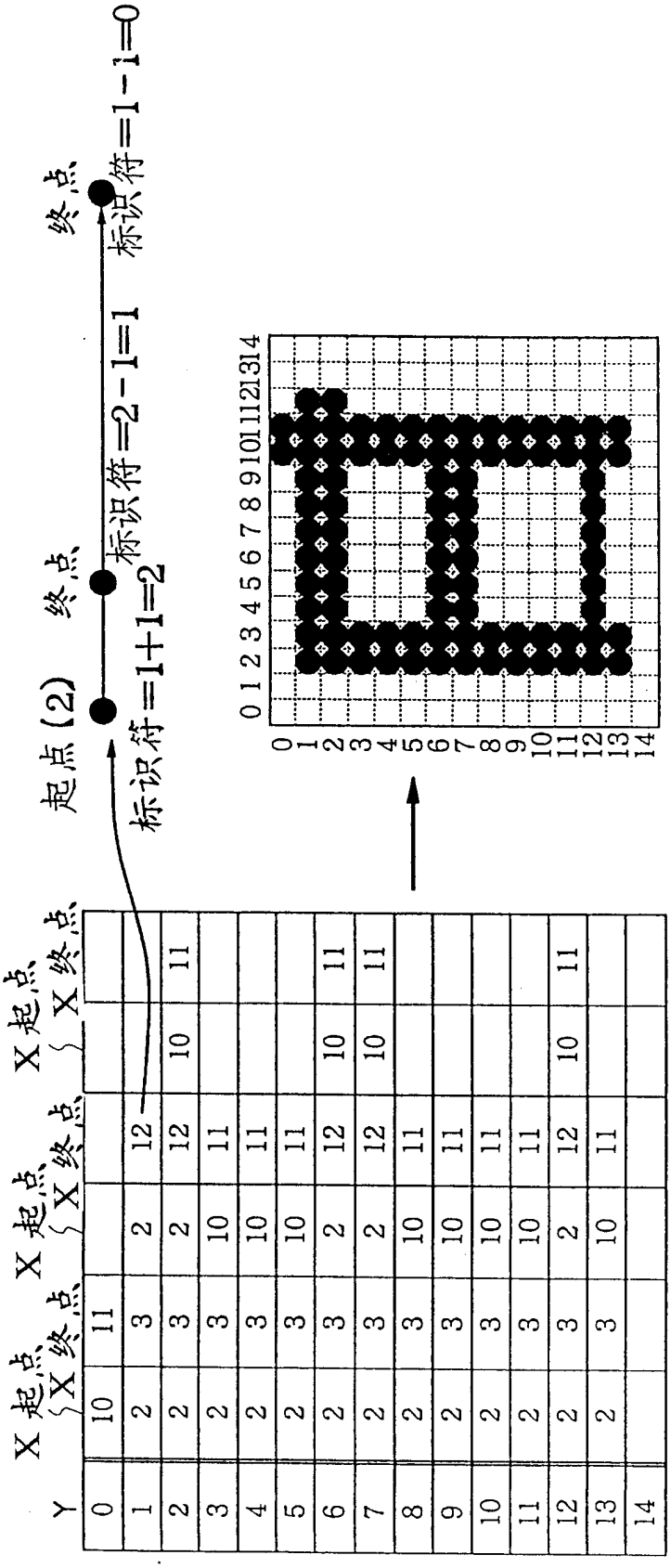

起点(2)
终点
标识符=1+1=2
标识符=2-1=1
标识符=1-1=0
X起点
X终点
Y
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

图 35

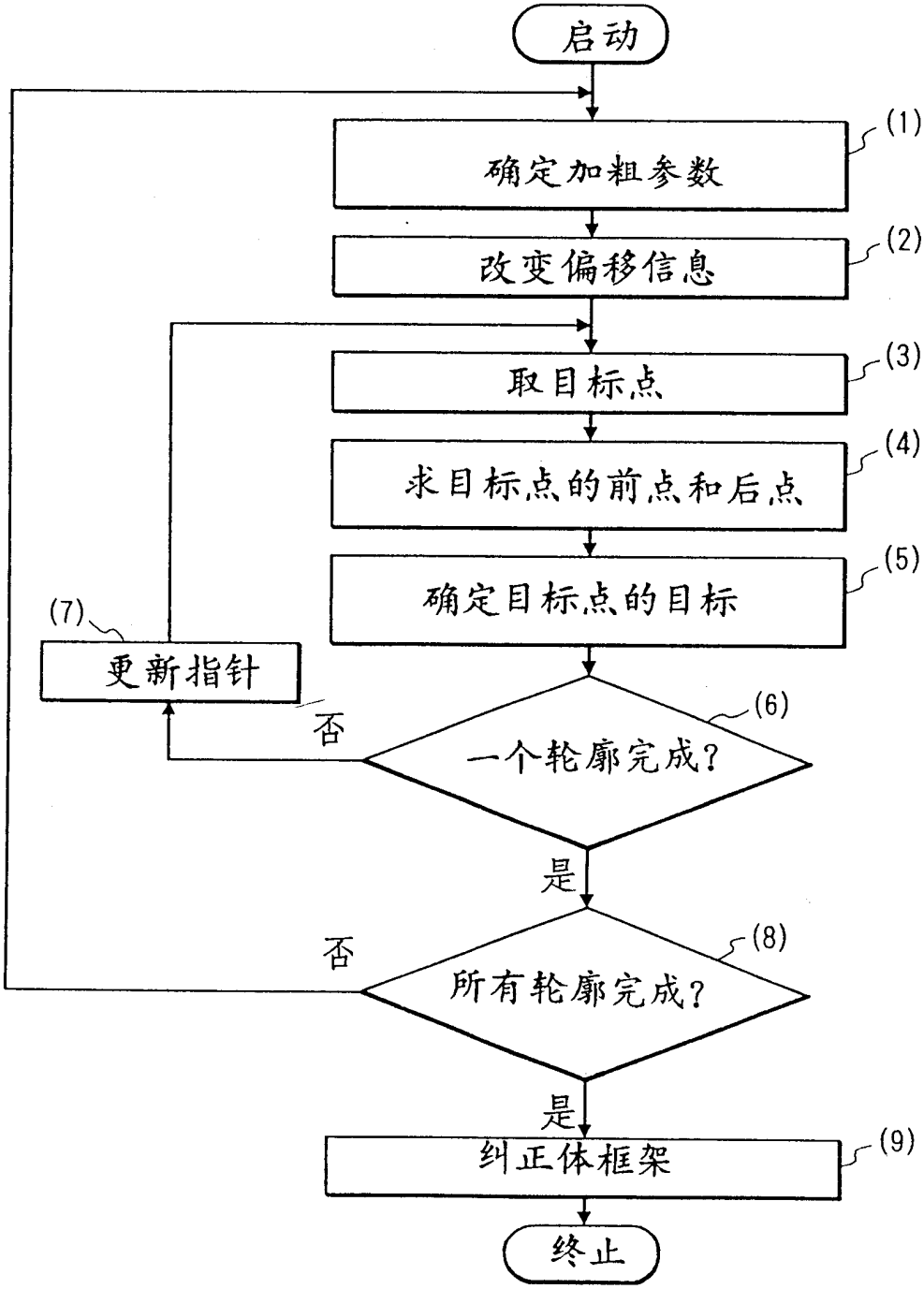

启动
确定加粗参数
(1)
改变偏移信息
(2)
取目标点
(3)
求目标点的前点和后点
(4)
确定目标点的目标
(5)
(7)
更新指针
否
一个轮廓完成?
(6)
是
否
所有轮廓完成?
(8)
是
纠正体框架
(9)
终止

图 36

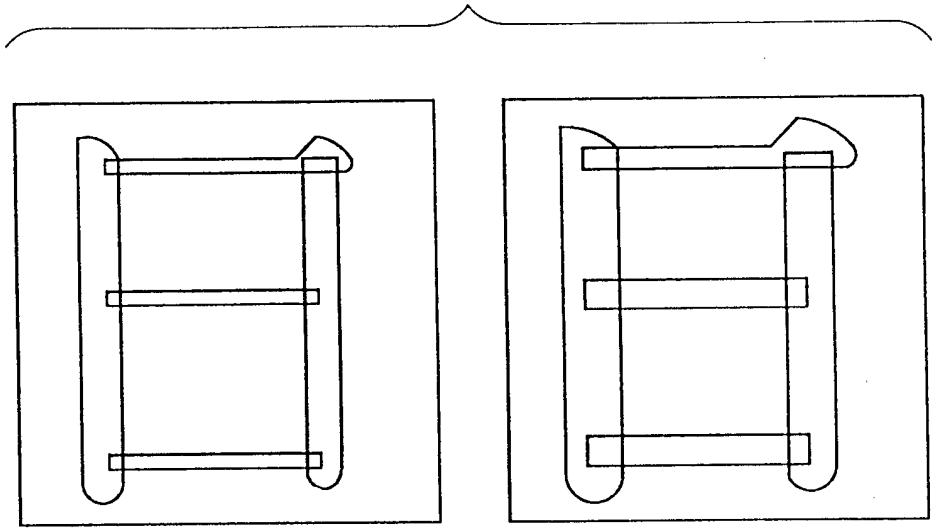


图 37

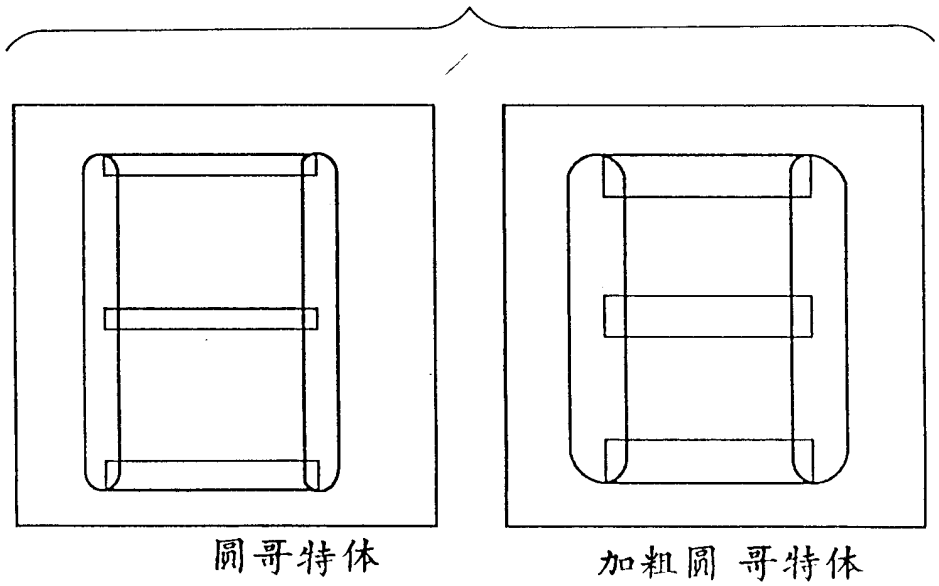


图３８

字体形式	明体		圆哥特体	
x/y	x	y	x	y
很细	6	10	10	10
细	9	15	15	15
标准	12	20	20	20
粗	15	25	25	25
很粗	18	30	30	30

图３９Ａ

字体形式	明 体			圆哥特体		
x/y	x	y	存在	x	y	存在
权值很细 1	3	5		5	5	
细 2	5	10		10	10	
标准 细 3	7	15	○	15	15	
标准 4	10	20		20	20	
标准粗 5	12	25		25	25	○
粗 6	14	30		30	30	
很粗 7	16	35	○	35	35	

图３９Ｂ

字体形式	圆哥特体			存在		
x/y	x	y	存在	x	y	存在
权值很细 1	5	5		3	5	
细 2	10	10		6	10	
标准细 3	15	15		9	15	
标准 4	20	20		12	20	○
标准粗 5	25	25		15	25	
粗 6	30	30	○	18	30	
很粗 7	35	35		21	35	○

图 40

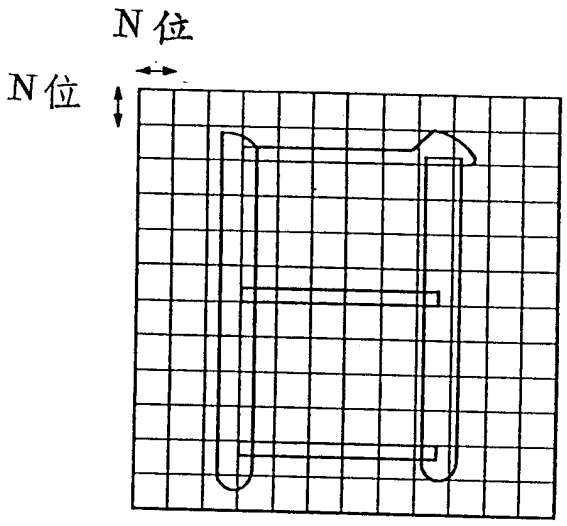


图42A

1	1	1	1
1	1	1	1
1	1	1	1
1	1	1	1

图42B

0.8	0.8	0.8	0.8
0.8	1.6	1.6	0.8
0.8	1.6	1.6	0.8
0.8	0.8	0.8	0.8

图42C

1.1	1.1	1.1	1.1
1.1	0.7	0.7	1.1
1.1	0.7	0.7	1.1
1.1	1.1	1.1	1.1

图４１

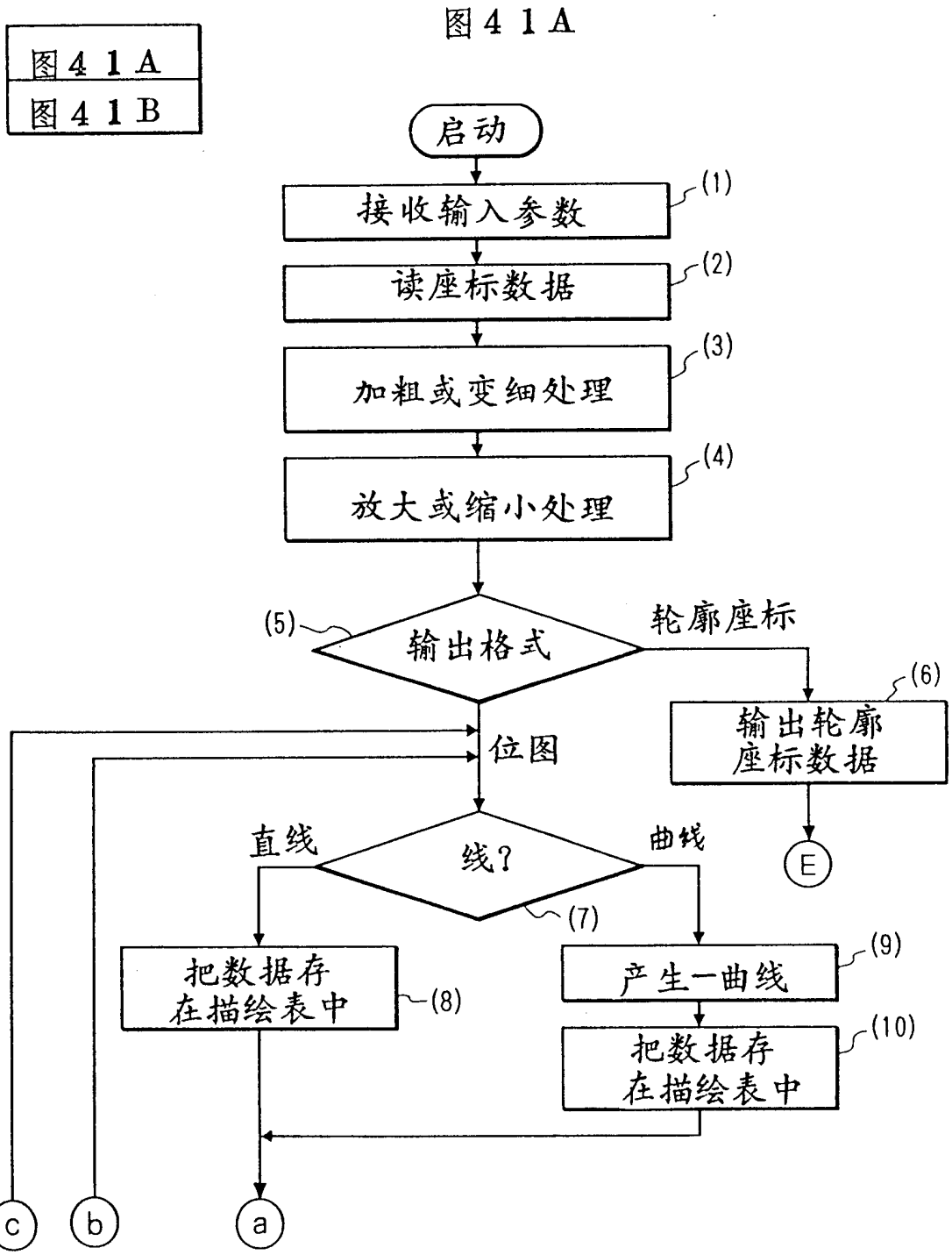


图４１A
图４１B
图４１A
启动
接收输入参数
(1)
读座标数据
(2)
加粗或变细处理
(3)
放大或缩小处理
(4)
(5)
输出格式
轮廓座标
(6)
输出轮廓
座标数据
位图
直线
线?
曲线
(7)
E
把数据存
在描绘表中
(8)
产生一曲线
(9)
把数据存
在描绘表中
(10)
c
b
a

图 4 1 B

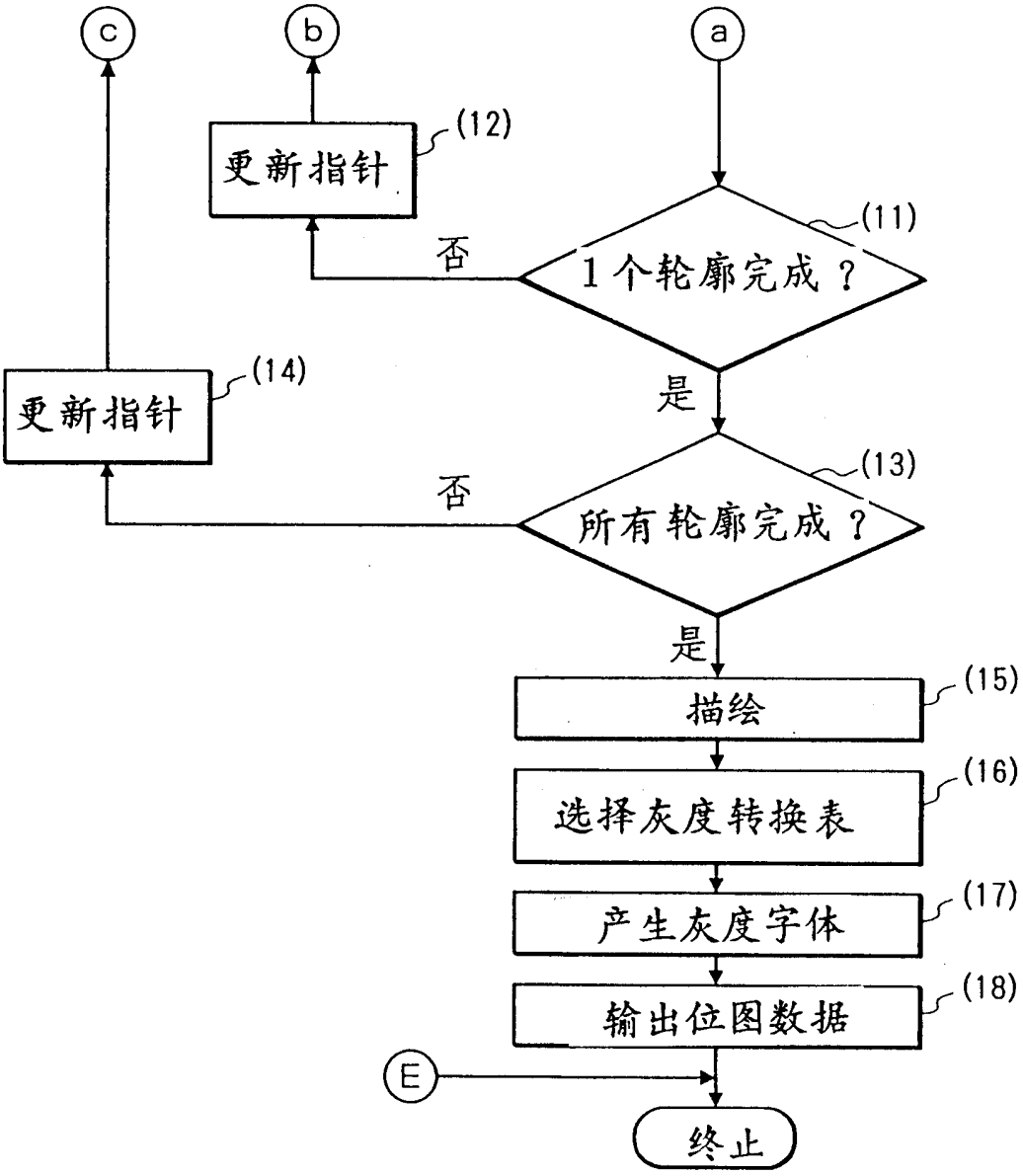

c
b
a
更新指针
(12)
否
1个轮廓完成？
(11)
是
更新指针
(14)
否
所有轮廓完成？
(13)
是
描绘
(15)
选择灰度转换表
(16)
产生灰度字体
(17)
输出位图数据
(18)
E
终止

图 4 3

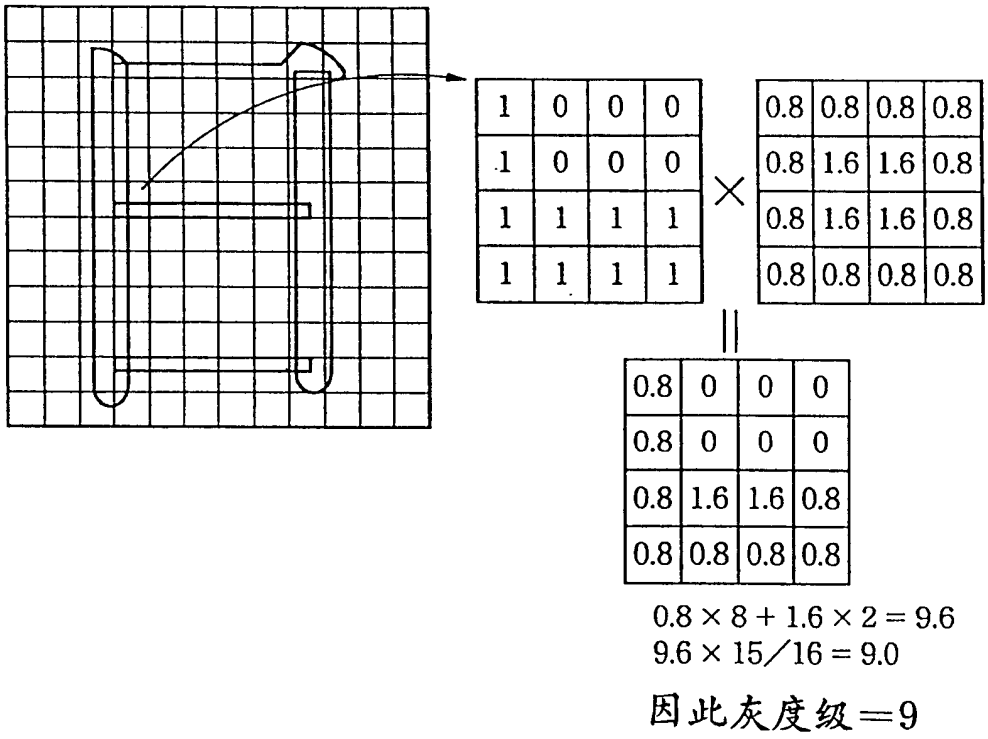


图 4 4

0	0	0	0	0	0	0	0	0	0	0	0
0	0	10	9	8	8	8	9	13	5	0	0
0	0	12	3	0	0	0	0	12	4	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	9	8	8	8	9	13	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	3	0	0	0	0	12	3	0	0
0	0	12	9	8	8	8	8	12	2	0	0
0	0	2	1	0	0	0	0	0	0	0	0