



[12] 发明专利申请公布说明书

[21] 申请号 200480001394.7

[43] 公开日 2009年4月15日

[11] 公开号 CN 101410885A

[22] 申请日 2004.7.30

[21] 申请号 200480001394.7

[30] 优先权

[32] 2004.1.26 [33] US [31] 10/764,961

[86] 国际申请 PCT/US2004/024650 2004.7.30

[87] 国际公布 WO2005/074411 英 2005.8.18

[85] 进入国家阶段日期 2005.5.26

[71] 申请人 微软公司

地址 美国华盛顿州

[72] 发明人 B·斯坦姆 G·C·希契考克
M·J·达根[74] 专利代理机构 上海专利商标事务所有限公司
代理人 钱慰民

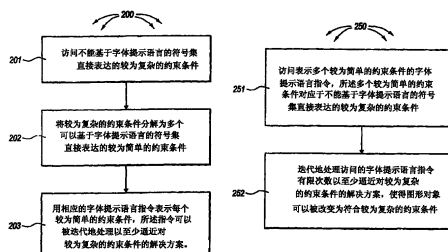
权利要求书4页 说明书15页 附图3页

[54] 发明名称

以字体提示语言交互地解出约束条件

[57] 摘要

本发明的原理涉及以字体提示语言迭代地解出约束条件。计算系统访问不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件，所述较为复杂的约束条件至少约束了轮廓的一部分(201)。所述计算系统将较为复杂的约束条件分解为多个可以基于字体提示语言的符号集直接表达的、较为简单的约束条件(202)。所述计算系统以相应的字体提示语言指令表示每个较为简单的约束条件，所述指令可以被迭代地处理以至少逼近对较为复杂的约束条件的解决方案(207)。在所述计算系统或另一个计算系统处迭代地处理字体提示语言指令，使得图形对象符合(在指定容限内)较为复杂的约束条件。



1. 在访问一组控制点的计算系统中,所述控制点组用于生成图形对象的轮廓,所述轮廓被用于确定如何再现所述图形对象,轮廓的一些部分的位置可能被约束到预定的位置,一种使用字体提示语言表示对约束条件的迭代解决方案的方法,所述方法包含:

访问不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件,所述较为复杂的约束条件约束轮廓的至少一部分;

将较为复杂的约束条件分解为多个可以基于字体提示语言的符号集直接表达的、较为简单的约束条件;

以相应的字体提示语言指令表示每个较为简单的约束条件,所述指令可以被迭代地处理以至少逼近对较为复杂的约束条件的解决方案。

2. 如权利要求1所述的方法,其特征在于,访问不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件包含访问基于幂函数或指数函数的约束条件。

3. 如权利要求1所述的方法,其特征在于,访问不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件包含访问要求同时移动多个控制点的约束条件。

4. 如权利要求1所述的方法,其特征在于,访问不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件包含访问循环依赖的约束条件。

5. 如权利要求1所述的方法,其特征在于,将较为复杂的约束条件分解为多个较为简单的约束条件包含基于幂函数或指数函数将约束条件分解为幂级数的多个部分。

6. 如权利要求1所述的方法,其特征在于,将较为复杂的约束条件分解为多个较为简单的约束条件包含将要求同时移动多个控制点的约束条件分解为多个约束条件,每个约束条件用于移动单独的控制点。

7. 如权利要求 1 所述的方法，其特征在于，将较为复杂的约束条件分解为多个较为简单的约束条件包含将循环依赖约束条件至少分解为：

第一约束条件，它约束第一控制点的位置，符合依赖第二控制点的位置的第一约束条件；

第二约束条件，包含第二控制点的位置，符合依赖第一控制点的位置的第二约束条件；

8. 如权利要求 1 所述的方法，其特征在于，以相应的字体提示语言指令表示每个较为简单的约束条件包含以 TrueType®指令表示每个较为简单的约束条件，所述指令可以被迭代地处理以至少逼近对较为复杂的约束条件的解决方案。

9. 如权利要求 1 所述的方法，其特征在于还包含：

迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案，使得至少轮廓的一部分可以被改变以符合较为复杂的约束条件；

生成图形对象的轮廓，使其至少在指定容限内符合较为复杂的约束条件；以及

基于所述轮廓生成图形对象的象素表示，所述象素表示用于在输出设备上再现。

10. 在被配置成处理字体提示语言指令的计算系统中，一种用于迭代地解出约束条件使得图形对象可以被适当地再现的计算机化的方法，所述方法包含：

接收表示多个较为简单的约束条件的字体提示语言指令，所述多个较为简单的约束条件对应于不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件；以及

迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案，使得图形对象可以被改变以符合较为复杂的约束条件。

11. 如权利要求 10 所述的方法，其特征在于，迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案包含迭代地地处理表示幂级数的部分的字体提示语言指令。

12. 如权利要求 10 所述的方法，其特征在于，迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案包含迭代地处理移动各个的控制点的字体提示语言指令。

13. 如权利要求 10 所述的方法，其特征在于，迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案包含：

基于第一控制点的当前位置，应用第一较为简单的约束条件以便为所述第二控制点计算目标位置；

基于第二控制点的当前位置，应用第二较为简单的约束条件以便为所述第一控制点计算目标位置；

确定第一控制点的目标位置是位于第一控制点的当前位置的指定容限之内；
以及

确定第二控制点的目标位置是位于第二控制点的当前位置的指定容限之内。

14. 如权利要求 13 所述的方法，其特征在于还包含：

使得所述第一控制点的目标位置成为第一控制点的当前位置；以及
使得所述第二控制点的目标位置成为第二控制点的当前位置。

15. 如权利要求 10 所述的方法，其特征在于，迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案包含迭代地处理字体提示语言指令直至执行了指定数目的迭代。

16. 如权利要求 10 所述的方法，其特征在于，迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案包含迭代地处理字体提示语言指令直至控制点的位置位于符合较为复杂的约束条件的阈值容限之内。

17. 如权利要求 10 所述的方法，其特征在于，迭代地处理字体提示语言指令有限次数以至少逼近对较为复杂的约束条件的解决方案包含迭代地处理 TrueType® 指令。

18. 如权利要求 10 所述的方法，其特征在于还包含：
生成所述图形对象的轮廓，使其符合较为复杂的约束条件；以及
基于所述轮廓生成图形对象的像素表示，所述像素表示用于在输出设备处再现。

19. 一种计算机程序产品，用于在访问一组控制点的计算系统中，所述控制点组用于生成图形对象的轮廓，所述轮廓被用于确定如何再现所述图形对象，轮廓的一些部分的位置可能被约束到预定的位置，所述计算机程序产品用于实现使用字体提示语言表示对约束条件的反复解决方案的方法，所述计算机程序产品包含计算机可执行指令，当处理器执行所述指令时使得所述计算系统执行以下：

访问不能基于字体提示语言的符号集直接表达的、较为复杂的约束条件，所述较为复杂的约束条件至少约束轮廓的一部分；

将较为复杂的约束条件分解为多个可以基于字体提示语言的符号集直接表达的、较为简单的约束条件；

以相应的字体提示语言指令表示每个较为简单的约束条件，所述指令可以被迭代地处理以至少逼近较为复杂的约束条件的解决方案。

以字体提示语言交互地解出约束条件

技术领域

本发明涉及用于操纵用于生成图形对象轮廓的控制数据的系统和方法，尤其涉及以字体提示语言反复地解出约束条件的系统和方法。

背景和相关领域

计算技术改变了我们工作和娱乐的方式。如今，计算系统具有各种各样的形式，包括台式计算机、膝上型计算机、输入板计算机、个人数字助理(PDA)等等。甚至连家用设备(诸如冰箱、烤箱、缝纫机、安全系统等等)也具有不同程度的处理能力，因此可以被认为是计算系统。随着时间的前进，处理能力会被结合到许多传统上不具有处理能力的设备中。相应地，计算系统的多样性很可能会增加。

几乎所有与人类交互的计算系统开始使用显示器来传递信息。在许多情况下，显示器的感染力被认为是计算系统的一个重大属性。历史上，文本信息(例如基于拉丁的字符)显示在阴极射线管(“CRT”)显示设备的单元中。每个单元被分成相等大小的格网位置网格，其中每个格网位置可以被打开或关闭。例如，CRT的每个单元可以是 8×8 的网格，导致每个单元有64个可能的网格位置。

字符组的每个字符做为一个存储图像(位图)存储在CRT显示设备的硬件中(例如在视频适配卡中)。存储图像包含多个二进制值(例如，为了将一个字符显示在 8×8 的网格上有64个二进制值)，其中每个二进制值对应于一个特定的网格位置。一个值(例如，二进制“1”)表明显示当该字符时，相应的网格位置为“打开”，而另一个值(例如，二进制“0”)表明当显示当该字符时，相应的网格位置为“关闭”。在接收表示一个字符的二进制数据(例如，位图)时，CRT将“打开”对应于二进制1的网格位置，而“关闭”对应于二进制0的网格位置以显示该字符。

更近一些，一些计算系统使用比例位图(例如，存储在磁盘上)，所述位图根据要显示的字符在单元尺寸方面不同。例如，在比例位图字符集中，用于字母“i”的单元(例如，3个网格位置的宽度)要比用于字母“h”的单元(例如，6个网格位置的宽度)更狭窄。

然而，作为位图的存储字符(无论是固定或比例)会消耗很多计算系统资源。

因为计算系统会需要以多种不同尺寸显示和打印字体的字符(一般为 256 或更多不同的字符), 会需要存储大量不同尺寸的位图。例如, 会希望有一文字处理器显示和打印大小范围从 4pt 到 72pt 的字体字符。因此, 运行文字处理器的计算系统可能必须存储 68 种(72 减 4)不同大小的位图用于以不同的大小显示字体。

而且, 由于打印机一般相对显示器具有不同的(对于多数来说是较高的)分辨率, 计算系统可能也必须存储相应的 68(72 减 4)种不同大小的位图用于以不同大小打印字体。例如, 8×5 网格的位图(需要 40 比特的存储)会被用于以特定尺寸显示字符, 而 50×30 网格的位图(需要 1500 比特的存储)被用于以特定尺寸打印字符。

当计算设备要从不同的字体显示和打印字符时, 与位图字体相关联的存储要求问题进一步复杂化。即, 计算设备会需要存储位图用于以多种不同大小表示多种不同字体。因此, 在上述例子中, 配置文字处理器以使用 50 种不同字体会导致超过 5000 中不同的位图集(例如, $(68+68) \times 50$)。由于许多字符集包含 256 或更多字符, 这会容易导致超过一百万个单独的位图(例如, 5000×256)。为每种字体的下划线的、黑体的和/或斜体版本存储位图会进一步增加存储需求。此外, 手动产生大量位图非常耗时。

相应地, 更近地, 图形基元被用于描述字体的字符。例如, 一组控制点和用于连接这些点的指令(例如, 用直线、弧线、贝塞尔曲线等连接)可以被用于在任意网格空间(例如, 超过象素设备最高分辨率的任意网格空间)定义字符的轮廓。通常, 会以较大的尺寸描述字符, 接着当字符要以较小的尺寸被再现(或黑体、斜体等等)时在数学术上缩小之(或不同地处理)。因此, 只需要存储减少数量的描述, 可能为一个字符(每种字体)只要一个描述。

为了缩小字符, 控制点的位置可除以比例因数。例如, 要用比例因数 10 缩小字符, 定义该字符(以较高分辨率)的每个控制点的坐标可除以 10。定义在 100×100 网格上显示的字符会被缩小成在 10×10 的网格上显示。因此在网格位置(50, 30)处的控制点会被缩小成在网格位置(5, 3)处的控制点, 在网格位置(70, 70)处的控制点会被缩小成在网格位置(7, 7)处的控制点等等。相应地, 可计算表示字符的较小的轮廓, 为字符存储多种不同尺寸的位图的需要降低了。

接着可以分析较小的轮廓以识别要打开的网格位置和识别要关闭的网格位置(通常被称为“扫描转换”的过程)。一个扫描转换算法确定网格位置的中心是在较小的轮廓之内还是之外的。当网格位置的中心是在较小轮廓之内, 就打开网格位置。另一方面, 如果网格位置的中心在较小轮廓之外, 就关闭网格位置。

同样地，当再现字符时，会要求部分字符符合一个或多个约束条件。约束条件可以表达为根据一个或多个独立参数定义一个或多个依赖参数的算法。对一个控制点的约束条件可以用其它控制点的位置或网格上的位置(例如，大写线)来表示。例如，控制点“P”的位置可以用控制点“Q”的位置来表示，使得P相距Q一固定距离“c”。即， $P=Q+c$ 。因此，当Q移动时，会需要对P的相应移动使得P符合固定距离c。

部分由于不同字体中不同的艺术和技术特征的多样化，约束条件可以适用于个别的字体。通常，约束条件用具有有限的和非常特定的符号集的字体提示语言(例如，TrueType®语言)表达。所述有限和非常特定的符号集简化了从数学概念到字体提示语言的翻译。例如，它一般直接翻译上述约束条件($P=Q+c$)，因为字体提示语言一般包含赋值运算符(例如“=”)和加法算符(例如“+”)。

然而，有限和非常特定的符号集也会阻止可以表达的约束条件类型。例如，基于如超越函数的更复杂的函数，来表达约束条件会是困难的，因为这些较为复杂的数学函数不包含在字体提示语言符号集中，并且会难以加上。缺少较为复杂的数学函数(例如平方根函数)转而使得确定适当的控制点位置以符合约束条件变得困难。因此，即使是在字体提示语言缺少较为复杂的符号集的情况下使用字体提示语言的现有符号集来解出约束条件的机制将是有优势的。

发明内容

本发明的原理克服了对本领域的原先技术中的上述问题，它主要针对以字体提示语言迭代地解出约束条件。计算系统访问较为复杂的约束条件，所述条件不能基于字体提示语言的符号集直接表达，较为复杂的约束条件约束了至少一部分的轮廓。计算系统将较为复杂的约束条件分解为多个较为简单的可以基于提示字体提示语言直接表达的约束条件。计算系统以相应的字体提示语言指令表示了每种较为简化的约束条件，所述指令可以被迭代地处理以至少逼近对较为复杂的约束条件的解决方案。字体提示语言指令在该计算系统或另一个计算系统被迭代处理，以使得图形对象符合(在指定容限内)较为复杂的约束条件。

以下将描述本发明的其它特征和优势，部分会从描述中变得明显，或通过本发明的实践而获得。通过尤其在所附权利要求书中的所提出的装置和组合，会实现和获取本发明的特征和优势。本发明的这些和其它特征会从以下描述和所附权利要求书变得完全地显而易见，或者可以通过下文所提出的本发明的实践中获得。

附图说明

为了描述本发明可以获得的上述和其它优势和特征的方式，将参考其特定实施例提供对以上简述的本发明的较为详细的描述，所述特定实施例在附图中说明。应该理解这些附图只描述了本发明的典型实施例，因此不应该被认为限制其范围，将通过使用附图描述和解释本发明的附加特征和细节，其中：

图 1 示出了用于以字体提示语言迭代地解出约束条件的示例的计算机构架。

图 2A 示出了使用字体提示语言表示对约束条件的迭代地解决方案的示例方法的流程图。

图 2B 示出了用于迭代地解出约束条件使得图形对象可以被适当地再现的示例方法的流程图。

图 3 示出了用于实现本发明的原理的适当的操作环境。

具体实施方式

本发明的原理涉及用于以字体提示语言迭代地解出约束条件的系统、方法和计算机程序产品。计算系统接收一组设计控制点，所述控制点以较大的尺寸（例如，72 点）描述图形对象（例如，文本字符）的轮廓。该组设计控制点可以是非提示的，或者可以包含用于以较小的尺寸显示图形对象的基本提示。该组设计控制点可以与不能基于字体提示语言的符号集直接表达的较为复杂的约束条件相关联。例如，该组设计控制点可以与循环依赖约束条件相关联，诸如，举例而言指示“Z”的对角线笔画的边缘是平行的约束条件。

计算系统访问不能基于字体提示语言的符号集直接表达的较为复杂的约束，所述较为复杂的约束条件约束至少部分的轮廓。计算系统将较为复杂的约束条件分解为多个较为简单的可以基于提示字体提示语言直接表达的约束条件。计算系统以相应的字体提示语言指令表示了每种较为简化的约束条件，所述指令可以被迭代地处理以至少逼近较为复杂的约束条件的解决方案。字体提示语言指令在该计算系统另一个计算系统被迭代地处理，以使得图形对象符合（在指定容限内）较为复杂的约束条件。

在本发明范围内的实施例包含用于携带或具有其上存在计算机可执行指令或数据结构的计算机可读介质。这样的计算机可读介质可以是任何通用或专用计算系统可访问的可用介质。通过举例而非限制，这样的计算机可读介质可以包含物理存

储介质诸如 RAM、ROM、EPROM、CD-ROM 或其它光盘存储、磁盘存储或其它磁性存储设备，或者任何其它可以被用于以计算机可执行指令、计算机可读指令、数据结构的形式传送或存储所期望的程序编码装置的介质，且它们可以由通用或专用计算系统访问。

在这个描述和随后的权利要求中，“网络”被定义为一个或多个数据链路，使得计算设备和/或模块之间能够传输电子数据。当信息在网络或另一个通信连接（有线的、无线的或有线和无线的组合）上被传递到或提供给计算系统时，该连接被适当地视为计算机可读介质。因此，任何这样的连接被适当地定义为计算机可读介质。上述的组合也包含在计算机可读介质的范围内。计算机可执行指令包含，例如使得通用或特殊用途计算系统执行某些功能或功能组的指令和数据。计算机可执行指令可以是，例如二进制、中间形式指令，诸如汇编语言，或者甚至是源代码。

在这个描述和随后的权利要求中，“计算系统”被定义为一个或多个软件模块、一个或多个硬件模块或其组合，它们一起工作对电子数据执行操作。例如，对计算系统的定义包含个人计算机的硬件组件以及软件模块，诸如个人计算机的操作系统。模块的实际配置不重要。计算系统会包含一个或多个经由网络耦合的计算机。同样地，计算系统可包含单个的物理设备（诸如移动电话或个人数据助理“PDA”），其中内部模块（诸如存储器和处理器）一起工作以执行在电子数据上的操作。

如这里所使用的，术语“模块”或“组件”可以指软件对象或在计算系统上执行的例行程序。这里所描述的不同的组件、模块、引擎和服务被实现为在计算系统上执行的对象或过程（例如，作为单独的线程）。虽然这里所描述的系统最好用软件实现，但是用软件和硬件或硬件的实现也是可能的和预期的。

本领域的技术人员会理解，本发明可以在具有多种类型的计算系统配置的网络计算环境中实现，所述计算系统配置包含个人计算机、膝上计算机、手掌设备、多处理器系统、基于微处理器或可编程消费电子产品、网络 PC、小型机、大型机、移动电话、PDA、寻呼机等等。本发明也可以在分布式系统环境中实现，其中通过网络连接（通过有线数据链路、无线数据链路或有线和无线数据链路的结合）的本地和远程计算系统都执行任务。在分布式系统环境中，编程模块可以同时位于本地和远程存储器存储设备中。

图 1 示出了用于以字体提示语言迭代地解出约束条件的计算机构架 100 的例子。在计算机构架 100 中，计算系统 118 包含提示模块 119。一般地，提示模块 119 接收一组表示图形对象（图形对象 131）的控制点（例如，设计控制点 122）。适当时，

提示模块 119 将字体提示语言(例如 TrueType®)的计算机可执行指令(此后称为“提示”)分配给控制点组内的所含的控制点(例如控制点 101—108)。随后,可以处理提示以更适当地再现图形对象(例如,以较小的尺寸)。

提示模块 119 可以包含多个模块,所述多个模块自动标识由控制点组表示的特征,诸如笔划、衬线等等,并自动标识控制点组内控制点的约束条件。一些约束条件,诸如,举例而言将第一控制点限制为离开第二控制点一水平距离,可以使用字体提示语言的现有符号集较为简单地实现。然而,诸如那些要求较为复杂的数学函数其它约束条件不能使用字体提示语言的现有符号集简单地实现(如果它们是完全可以实现的)。例如,当字体提示语言的现有符号集缺少平方根函数,直接计算关于对角边的距离约束条件会是困难的(或甚至是不可能的)。

约束条件识别模块 121 可以基于控制点的位置或包含在设计控制点 122 中的基本提示识别约束条件,包含否则难以解出的约束条件。基本提示可以是用于以较大的尺寸(例如,在目标设备处 72 点)再现图形对象的提示,但是当以其它的尺寸(例如,在目标设备处 12 点)再现图形对象时不是一般适用的。例如,当在特定目标设备处再现为 72 点时,基本提示可以将“T”的竖划的宽度限制为六个像素。然而,以较小的尺寸实现相同的六个像素约束条件降低了其意义,因为当在特定目标设备处以较小的尺寸再现时,图形对象在宽度上甚至不是六个像素。相应地,提示模块 119 可以生成约束条件等式,基本限制图像对象要被显示的尺寸和分辨率。例如,提示模块 119 可以将对“T”的竖划的约束条件配置为“T”被再现的点尺寸的十二分之一。

根据目标设备的分辨率,对应于点尺寸的十二分之一的像素的数目可以变化。在较高分辨率的设备上,诸如,举例而言 300 点每英寸(“dpi”)打印机、点尺寸的十二分之一可以对应于较多数目的像素。另一方面,在较低分辨率的设备上,诸如,举例而言 96dpi 监视器,点尺寸的十二分之一可以对应于较少数目的像素。

提示应用模块 129 可以为设计控制点 122 增加提示(例如 TrueType®指令)以在图形对象随后被再现时实现这样的约束条件。例如,除法操作符可以被用于实现对“T”的竖划的宽度约束条件(笔划的宽度=点尺寸/12)。提示应用模块 129 可以包含用于迭代地解出其它更为复杂的约束条件的提示,诸如要求更为复杂的数学函数的那些约束条件。当随后再现图形对象时,计算系统能够处理所述提示以基于,如图形对象要显示的尺寸迭代地解出约束条件。

计算系统 123 包含缩放模块 124、提示处理器 134 和扫描转换模块 126。缩放

模块 124 能够接收一组以较大的尺寸表示图形对象的轮廓的提示控制点(例如控制点 132)。提示控制点可以经由网络连接接收或从计算机可读介质载入,诸如,磁盘或光盘。缩放模块 124 可以缩放控制点的位置,使得可以以较小或较大的尺寸再现图形对象。

提示处理器 134 可以处理与所提示的控制点组相关联的提示(可能进一步改变缩小的控制点的位置)使得图形对象更适合以较小的尺寸再现。例如,提示处理器 134 能够处理字体提示语言的计算机可执行指令,以迭代地解出更为复杂的约束条件。扫描转换模块 126 将相应像素网格的像素(或子像素)打开或关闭以生成图形对象的像素表示。

因此,用于实现较为复杂的约束条件的提示能够与一组控制点相关联,而无需改变字体提示语言或相应的提示处理器以包含较为复杂的数学函数。相应地,计算系统可以使用现有组件以符合较为复杂的约束条件的方式来再现图形对象。这是有优势的,因为改变字体提示语言的符号集和/或提示处理器的功能会比使用现有字体提示消息符号集来提示图像对象要困难和耗时得多。

图 2A 示出了用于以字体提示语言迭代地解出约束条件的示例方法 200 的流程图。将关于计算机构架 100 中的计算系统、模块和控制点来描述方法 200。方法 200 包含访问较为复杂的约束条件的动作,所述约束条件不能基于字体提示语言的符号集(动作 201)简单地表达。例如,提示模块 119 能够(识别和)方位用设计控制点 122 表示的较为复杂的约束条件。提示模块 119 和/或提示处理器 134 所使用的字体提示语言可能不具有从控制点 122 访问的用于表达较为复杂的约束条件的符号集。

较为复杂的约束条件可以是,如由幂函数或指数函数(X^Y)表达的约束条件、多个循环依赖约束条件或要求多个控制点同时移动的约束条件。然而,这个较为复杂的约束条件列表并非意在是全面的。对于本领域的技术人员应该显而易见的是在看过这个描述之后,除了那些可以被清除描述的约束条件之外,其它较为复杂的约束条件可以被访问。

关于循环依赖约束条件,符合第一约束条件的能够依赖于第一控制点的位置。例如,约束条件识别模块 121 可以识别控制点 107 被约束为距边缘 117 对角线距离 114。符合对角线距离 114 依赖于控制点 103 的位置,因为关于控制点 102 改变控制点 103 的位置会改变对角线距离 114 的方向。

符合第二约束条件能够依赖于第二控制点的位置。例如,约束条件识别模块 121 可以识别控制点 103 被约束为距边缘 116 对角线距离 144。符合对角线距离 144

依赖于控制点 107 的位置,因为关于控制点 106 改变控制点 107 的位置会改变对角线距离 144 的方向。

因此,第一和第二约束条件导致循环依赖性,其中控制点 107 的符合对角线距离 114 依赖于控制点 103,而控制点 103 的符合对角线距离 144 依赖于控制点 107。约束条件识别模块 121 可以识别循环依赖性。

方法 200 包含将较为复杂的约束条件分解成多个较为简单的约束条件的动作,所述较为简单的约束条件可以基于字体提示语言的字符集直接地表达(动作 202)。例如,提示模块 119 可以将从设计控制点 122 访问的较为复杂的约束条件分解为被提示模块 119 和/或提示处理器 134 使用的可以用字体提示语言表达的多个较为简单的约束条件。

分解较为复杂的约束条件可以包含任何数目的用于降低较为复杂的约束条件的复杂性的机制。例如,循环依赖约束条件可以被分解成多个非循环依赖约束条件。基于指数函数或幂函数的约束条件可以被分解成幂级数。要求多个控制点同时移动的约束条件可以被分解成多个约束条件,每个对应于单个控制点。其它较为复杂的约束条件也可以被分解成多个相应的较为简单的约束条件。

方法 200 包含以相应的字体提示语言指令表达每个较为简单的约束条件的动作,所述指令可以被迭代地处理以至少逼近对较为复杂的约束条件的解决方案(动作 203)。例如,提示模块 119 可以以字体提示语言指令 TrueType®指令,表达较为简单的约束条件,这些指令可以在提示处理器 134 处被迭代地处理。

关于循环依赖约束条件,表示较为简单的约束条件可以包含制定字体提示语言指令,用于基于第一控制点的当前位置应用第一约束条件,以便为第二控制点计算目标位置。例如,提示处理器 134 可以基于控制点 103 的当前位置(在缩小以后)为计算对角线距离 114 制定指令,用于为控制点 107 计算目标位置。同样地,表示较为简单的约束条件可以包含制定字体提示语言指令,用于基于第二控制点的当前位置应用第二约束条件,以便为第一控制点计算目标位置。例如,提示处理器 134 可以基于控制点 107 的当前位置(在缩小以后)为计算对角线距离 144 制定指令,用于为控制点 103 计算目标位置。

可以交替应用为表示第一和第二约束条件制定的指令。例如,可以应用第一约束条件,可以应用第二约束条件,可以再一次应用第一约束条件,可以应用第二约束条件等等。交替应用第一和第二约束条件可以继续直至控制点位置适当为止。

相应地,以字体提示语言指令表达的较为简单的约束条件也可以包含制定字

体提示语言指令以确定第二控制点的目标位置位于第二控制点当前位置的指定容限之内。例如，提示模块 134 制定字体提示语言指令用于确定控制点 107 的目标位置是在控制点 107 的当前位置的指定容限之内。指定容限可以是以象素或象素的分数(例如 1/64 的象素)为单位的一距离。同样地，以字体提示语言指令表达较为简单的约束条件可以包含制定字体提示语言指令以确定第一控制点的目标位置是位于第一控制点当前位置的指定容限之内的。例如，提示模块 134 可以制定字体提示语言指令，以确定控制点 103 的目标位置是位于控制点 103 当前位置的指定容限之内的。

对于基于幂函数的较为复杂的限制条件，表示较为简单的限制条件可以包含制定字体提示语言指令以解幂级数。对于要求多个控制点同时移动的约束条件，表示较为简单的约束条件可以包含制定字体提示语言指令以分别地移动控制点。表示对应于其它类型的较为复杂的约束条件的较为简单的约束条件也可以包含制定适当的字体提示语言指令。

在一些实施例中，表示多个较为简单约束条件的字体提示语言指令在制定字体提示语言指令的计算系统处处理。例如，提示模块 119 可以处理先前在提示模块 119 处制定的字体提示语言指令(例如与控制点 132 相关联的)。在其它实施例中，表示多个较为简单约束条件的字体提示语言指令被发送到另一个计算系统用于处理。例如，计算系统 118 可以将字体提示语言指令(例如与控制点 132 相关联的)发送到计算系统 123 用于处理。

图 2B 说明了以字体提示语言迭代地解出约束条件的示例方法 250 的流程图。将关于计算机构架 100 中的计算系统、模块和控制点来描述方法 250。可以在制定表示多个较为简单的约束条件的字体提示指令的计算系统(例如，计算系统 118)处或在接收表示多个较为简单的约束条件的制定的字体提示指令的计算系统(例如，计算系统 123)处执行方法 250。

方法 250 包含访问表示多个较为简单的约束条件的字体提示语言指令的动作，所述多个较为简单的约束条件对应于不能基于字体提示语言的符号集直接表达的较为复杂的约束条件(动作 251)。例如，提示模块 119 可以访问表示多个较为简单的约束条件的字体提示语言指令(例如，TrueType®指令)，所述多个较为简单的约束条件对应于一与控制点 132 相关联的较为复杂的约束条件。相似地，提示处理器 134 可以访问表示多个较为简单的约束条件的字体提示语言指令(例如，TrueType®指令)，所述多个较为简单的约束条件对应于一与控制点 132 相关联的较为复杂的

约束条件。

方法 250 包含有限次数迭代处理字体提示语言指令以至少逼近对较为复杂的约束条件的解决方案,使得图形对象可以被改变以符合较为复杂的约束条件(动作 252)。例如,适当时,提示模块 119 或提示处理器 132 可以迭代地处理字体提示语言指令以至少逼近对较为复杂的约束条件的解决方案。

对于要求多个控制点同时移动的较为复杂的约束条件,提示模块 119 或提示处理器 132 处理多个字体提示语言指令以移动各个的控制点(对单独控制点的移动被视为一次迭代)直至所有的控制点都被移动。对于包含指数函数或幂函数的约束条件,提示模块 119 或提示处理器 132 处理多个表示部分幂级数的字体提示语言指令(对每个部分的处理被视为一次迭代)直至每个部分都被处理。部分基于较为复杂的约束条件的类型和如何分解复杂的约束条件,指定部分表示较为简单的约束条件的字体提示语言指令的处理可以被示为一次迭代。

对于循环依赖约束条件,迭代可以继续直至控制点的位置位于符合约束条件的阈值容限之内或执行了指定数目的迭代。例如,当控制点 103 和 107 的目标位置分别位于相对应的控制点 103 和 107 的当前位置的指定容限之内,控制点 103 和 107 的当前位置被视为足够精确。另一方面,当控制点 103 和 107 的任一目标位置不是分别位于相对应的控制点 103 和 107 的当前位置的指定容限之内,控制点 103 和 107 的当前位置不被视为足够精确。

若当前位置不足够精确,控制点 103 和 107 的当前位置被分别设置为等于对控制点 103 和 107 计算的目标位置。再一次应用第一和第二约束条件(使用计算目标位置作为当前位置)以便为控制点 103 和 107 计算第二目标位置。即,计算目标位置被用于迭代第二目标位置。可以继续迭代地将从先前计算所得的计算目标位置用作新的计算的当前位置(例如,将第二目标位置用作迭代第三目标位置的当前位置等等)直至某次迭代的计算目标位置位于当前位置的指定容限之内。或者,迭代计算可以继续直至发生阈值次数的迭代。例如,循环指令可以指明在十次迭代后停止迭代。

以下伪码表示可以制定和/或执行以便于迭代地解出约束条件的算法的例子。在该伪码的例子中,在控制点 132 被缩小和提示后, P_1 可以指控制点 107 的位置, P_2 可以指控制点 106 的位置。类似地在伪码例子中,在控制点 132 被缩小和提示后, Q_1 可以指控制点 102 的位置, Q_2 可以指控制点 103 的位置。可以用多种字体提示语言制定伪码算法,并用相应的提示处理器处理它(例如提示处理器 134)。在伪码例

子中，分号(“;”)后面的文本表示注释，用于描绘伪码所执行的功能。

第一行: $n = \text{迭代的次数}$; 初始化计时器
 第二行: $P_1' = P_1$; 记录 P_1 的当前位置
 第三行: $Q_2' = Q_2$; 记录 Q_2 的当前位置
 第四行: $P_1 \leftarrow Q_1 Q_2 + c$; 如同 Q_2 位于其目标位置那样执行约束条件
 第五行: $Q_1 \leftarrow P_2 P_1 + c$; 如同 P_1 位于其目标位置那样执行约束条件
 第六行: $n \leftarrow n - 1$; 计数器减 1
 第七行: $n = 0?$; 发生了阈值次数的迭代?
 是, 跳转到第 10 行
 否, 继续
 第八行: $|P_1 - P_1'| < 1/64?$; P_1 和 P_1' 之间的距离在容限之内?
 否, 跳转到第 2 行
 是, 继续
 第九行: $|Q_2 - Q_2'| < 1/64?$; Q_2 和 Q_2' 之间的距离在容限之内?
 否, 跳转到第 2 行
 是, 继续
 第十行: 结束

在伪码的例子中，执行第 4 行以应用第一约束条件，诸如，对角线距离 114，并执行第 5 行以应用第二约束条件，诸如对角线距离 144。在一些实施例中，所示的功能和第 4 和第 5 行被进一步改进。

例如，在第 4 行中， P_1 被从边 $Q_1 Q_2$ 垂直测量的距离 c 限制。在这些改进的实施例中，这种约束条件被分为两个部分。在距离部分， P_1 被移动到符合距离 c 。随后，在单独的角度部分，将 Q_2 移动一足以重建边 $P_1 P_2$ 和 $Q_1 Q_2$ 之间角度的量。即，如果这些边被设计成平行的，那么在执行角度部分后它们将再次平行。移动 Q_2 以符合角度部分会导致额外的不符合距离部分。然而，随着执行更多的迭代，由移动 Q_2 以符合角度部分而引入的额外的不符合距离的量将变得更小且越来越不重要(在一些次数的迭代后将变得实质上不相关)。

在第 5 行中，可以执行类似的“再平行”。第一， Q_1 被移动到符合约束条件的距离部分，接着移动 P_1 以符合约束条件的角度部分。这些改进的实施例便利了对可用控制点的限制，以此模仿限制笔划，这可以包含限制笔划的边之间的角度。相应地，当可用的控制数据包含控制点以及如何连接这些控制点的指令(直线, Bezier

曲线等等), 但不明显地表示这些笔划时, 这些改进的实施例有利于限制笔划。

第 7 行执行检查以确定是否发生了指定阈值数目的迭代。当发生了指定阈值数目的迭代, 算法终止(即使没有计算出对于约束条件足够精确的解决方案)。另一方面, 如果未发生指定阈值数目的迭代, 算法继续到第 8 行。

第 8 行执行检查以确定 P_1 (第一控制点的目标位置) 是否在 P_1' (第一控制点的当前位置) 的指定容限(像素的 $1/64$) 之内。当 P_1 和 P_1' 之间的距离不在指定容限之内, 算法继续到第 2 行(并开始另一个迭代)。另一方面, 当 P_1 和 P_1' 之间的距离在指定容限之内, 算法继续到第 9 行。

第 9 行执行检查以确定 Q_2 (第二控制点的目标位置) 是否在 Q_2' (第二控制点的当前位置) 的指定容限(像素的 $1/64$) 之内。当 Q_2 和 Q_2' 之间的距离不在指定容限之内, 算法继续到第 2 行(并开始另一个迭代)。另一方面, 当 Q_2 和 Q_2' 之间的距离在指定容限之内, 算法终止。当 P_1 和 P_1' 之间的距离在容限之内且 Q_2 和 Q_2' 之间的距离在指定容限之内时, P_1 和 Q_2 被视为有足够精确以再现相应的图形对象。

在一些实施例中, 迭代地解出约束条件被用于补偿由于符合多个依赖约束条件而发生的不适当的再现。例如, 当控制点具有多个与它们相关联的约束条件, 且这些约束条件中的一些依赖于其它约束条件, 其它约束条件反过来依赖于其它约束条件等等, 适当地再现相应的图形对象将变得更为困难。例如, 对于图形对象 131, 一组限制条件 C_1 会将边 151 带入到最近全像素(例如, 接近“大写线”)。另一组约束条件 C_2 会将边 152 保持在离开边 151 指定的距离。相似的约束条件组 C_1' 和 C_2' 会与控制点 106、105、103 和 104 相关联。又另一组约束条件 C_3 会将边 116 保持在离开边 117 指定的距离。符合 C_1 和 C_1' 的约束条件会分别影响 C_2 和 C_2' 的约束条件, 它们转而又会影响 C_3 的约束条件。因此, 符合 C_1 、 C_1' 、 C_2 和 C_2' 的约束条件会导致控制点 103 和/或 107 不再符合 C_3 的约束条件(例如, 对角线距离 114 和 144)。然而, 本发明的实施例可以被用于迭代地解出 C_3 的约束条件, 使得图形对象仍然符合 C_1 、 C_1' 、 C_2 和 C_2' 的约束条件。

在按照接收到的提示改变控制点位置之后, 扫描转换模块 126 会在像素网格上打开适当的像素(或子像素), 使得图形对象 131 能够以较小的尺寸被再现。扫描转换模块 126 可以生成像素表示 127, 接着所生成的像素表示被提供给显示设备 128。显示设备 128 可以是彩色或单色的监视器。适当时, 例如, 当显示设备 128 是彩色监视器时, 扫描转换模块仅打开一些像素的部分(例如像素的红、绿或蓝子组成部分)以更好地再现图形对象 131。

图 3 和以下的讨论意在提供本发明可以在其中实现的、合适的计算环境的简要、一般描述。虽然不是要求的，本发明会在计算机可执行指令的背景中描述，诸如计算机执行的程序模块。一般地，程序模块包含例程序、程序、对象、组件、数据结构等等，它们执行特定任务或实现特定抽象数据类型。与数据结构相关联的计算机可执行指令和程序模块表示了这里所揭示的方法的执行动作的程序代码装置的例子。

参见图 3，用于实现本发明的示例系统包含以计算机系统 320 形式示出的通用计算设备，包含处理单元 321、系统存储器 322 以及把包括系统存储器 322 在内的各种系统组件耦合至处理单元 321 的系统总线 323。处理单元 321 可以执行设计以实现计算机系统 320 的特征(包含本发明的特征)的计算机可执行指令。系统总线 323 可以是若干总线结构类型中的任一种，包括存储器总线或存储器控制器、外围总线以及使用多种总线构架的局部总线的任一种。系统存储器包含只读存储器(“ROM”)324 以及随机存取存储器(“RAM”)325。基本输入/输出系统(“BIOS”)存储在 ROM 324 内，它包括如启动时帮助在计算机 320 内的元件间传输信息的基本例程。

计算机系统系统 320 还包含读写磁硬盘 339 的磁硬盘驱动器 327、从可移动磁盘 329 的磁盘驱动器 328、读写 CD-ROM 或其它光学介质这样的可移动光盘 331 的光盘驱动器 330。磁硬盘驱动器 327、磁盘驱动器 328 和光盘驱动器 330 分别通过硬盘驱动器接口 332、磁盘驱动器接口 333 和光盘驱动器接口 334 连接到系统总线 323。这些驱动器和它们相关联的计算机可读介质提供了对计算机可执行指令、数据结构、程序模块和计算机系统 320 的其它数据的非易失性存储。虽然这里所描述的示例环境采用了磁性硬盘 339、可移动光盘 331，可以使用用于存储数据的其它类型的计算机可读介质，包含盒式磁带、闪存卡、数字多用途盘、Bernoulli 盒式磁带、RAM、ROM 等等。

包含一个或多个程序模块的程序代码装置可以被存储在硬盘 339、磁盘 329、光盘 331、ROM 324 或 RAM 325 上，包含操作系统 335、一个或多个应用程序 336、其它程序模块 337 和程序数据 338。用户可以通过键盘 340、定位设备 342 或其它输入设备(未示出)将命令和信息输入计算机系统 320，所述其它输入设备诸如麦克风、操纵杆、游戏垫、扫描仪等等。这些和其它的输入设备通过与系统总线 323 耦合的输入/输出接口 346 连接到处理单元 321 上。输入/输出接口 346 逻辑上表示多种类型不同接口种的任意一种，诸如串行接口、PS/2 接口、并行接口、通用串

行总线(“USB”)接口、或电气和电子工程师协会(“IEEE”)1394接口(即,高速率行连接总线(Fire Wire)接口)或者甚至是逻辑上表示不同接口的组合。

监视器 347 或其它类型的显示设备也通过视频接口 348 连接到系统总线 323。监视器 347 可以显示图形对象,包含由计算机系统 320 产生的文本。其它外围设备(未示出)诸如扬声器、打印机和扫描仪也可以连接到计算机系统 320。连接到计算机系统 347 的打印机可以打印图形对象,包含由计算机系统 320 产生的文本。

计算机系统 320 可以连接到网络,诸如办公室范围或公司范围的计算机网络、家庭网络、内联网和/或因特网。计算机系统 320 可以在这样的网络上与外部信源交换数据,诸如远程计算机系统、远程应用程序和/或远程数据库。

计算机系统 320 包括网络接口 353,计算机系统 320 通过所述网络接口 353 从外部信源接收数据和/或发送数据至外部信源。如图 3 所示,网络接口 353 便利了通过链路 351 与远程计算机系统 383 交换数据。网络接口 353 可以逻辑上表示一个或多个软件和/或硬件模块,诸如网络接口卡和相应的网络驱动程序接口规范(“NDIS”)栈。链路 351 表示网络的一部分(例如,以太网网段),远程计算机系统 383 表示网络的节点。例如,链路 351 能够表示计算机系统 118 和 123 之间的网络连接。

同样地,计算机系统 320 包含输入/输出接口 346,计算机系统 320 通过所述输入/输出接口 346 从外部信源接收数据和/或发送数据至外部信源。输入/输出接口 346 耦合到调制解调器 354(例如,标准调制解调器、电缆调制解调器或数字用户线路(“DSL”)调制解调器)上,计算机系统 320 通过所述调制解调器 354 从外部信源接收数据和/或发送数据至外部信源。如图 3 所示,输入/输出接口 346 和调制解调器 354 便利了通过链路 352 与远程计算机系统 393 交换数据。链路 352 表示网络的一部分,远程计算机系统 393 表示网络的节点。

虽然图 3 示出了本发明使用的操作环境,但是可以在能够实现本发明的原理的任何系统(如有必要作适当的修改)中采用本发明的原理。图 3 所示的环境只是说明性的,决不是表示即使只是多种可以实现本发明的原理的环境中的一部分。

依照本发明模块,诸如提示模块 119 和提示处理器 134 以及相关程序数据,诸如设计控制点 122、控制点 132 和象素表示 127 可以被存储在与计算机系统 320 相关联的任一计算机可读介质上并可从中读取。例如,部分这样的模块和部分相关联的程序数据可以包含在操作系统 335、应用程序 336、程序模块 337 和/或程序数据 338 中,用于存储在系统存储器中。

当大容量的存储设备(例如,磁性硬盘 339)耦合到计算机系统 320 上,这样的

模块和相关联的程序数据也可以被存储在所述大容量存储设备中。在网络环境中，关于计算机系统 320 描述的模块或其部分可以被存储在远程存储器存储设备中，诸如，与远程计算机系统 383 和/或远程计算机系统 393 相关联的系统存储器和/或大容量存储设备。这样模块的执行可以在分布式环境中进行。

本发明可以以其它特定的形式实施而不背离其精神和主要的特征。无论从那方面来看，所述的实施例应该被认为是说明性的而非限制性的。因此，本发明的范围是由所附的权利要求书而非上述描述所指出的。所有在权利要求书的等价的含义和等价范围之内内的变化是在它们的范围之内内的。

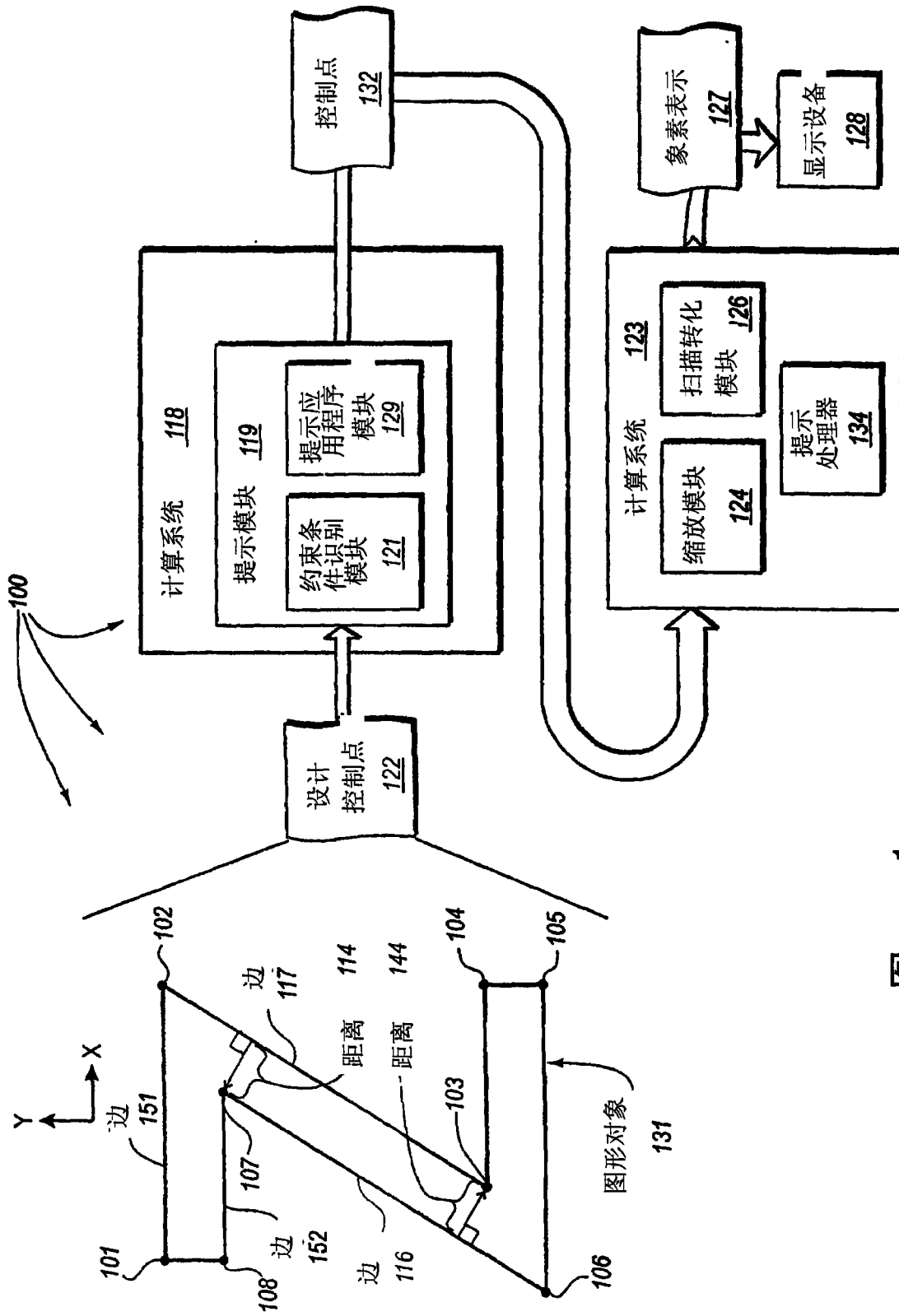


图 1

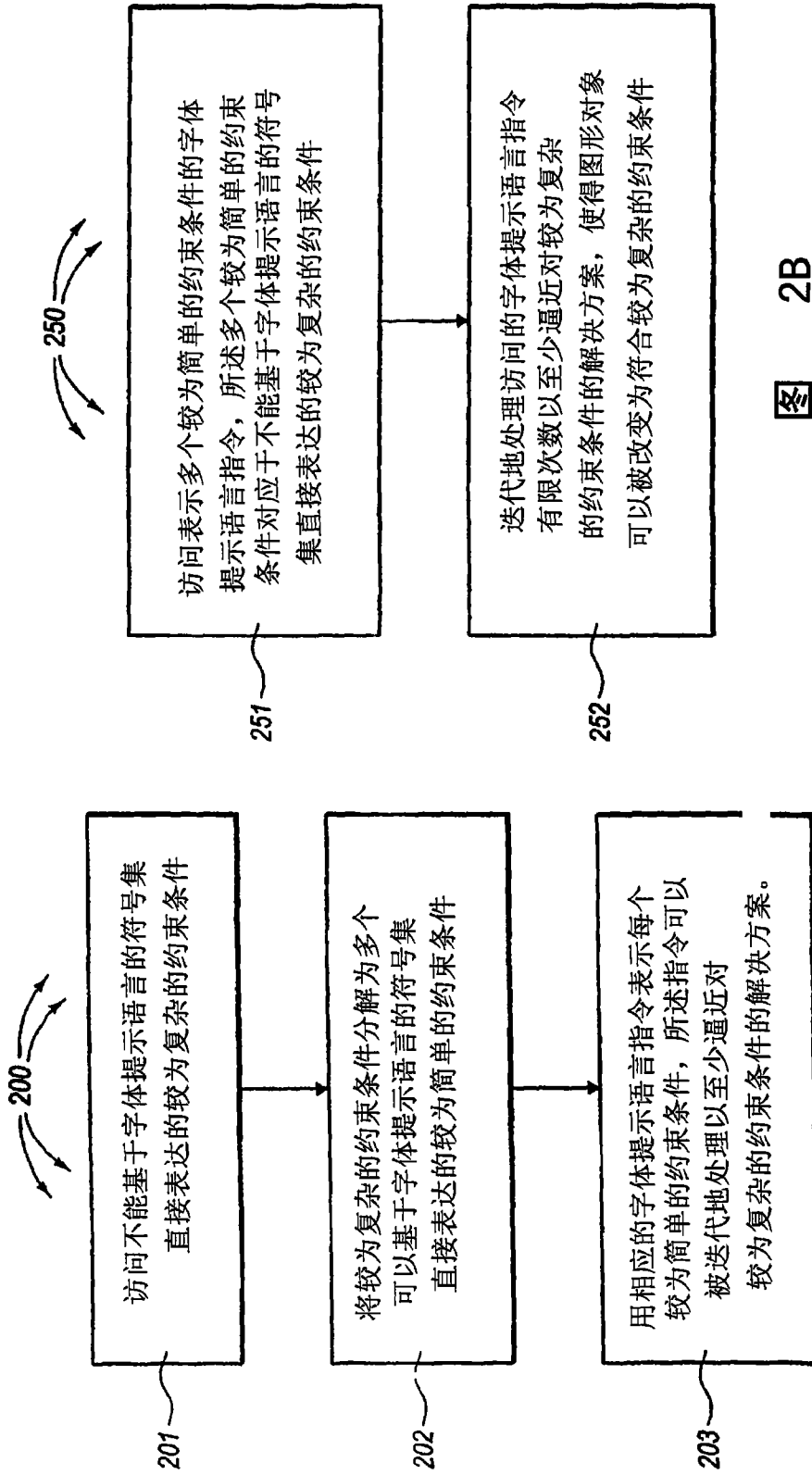


图 2A

图 2B

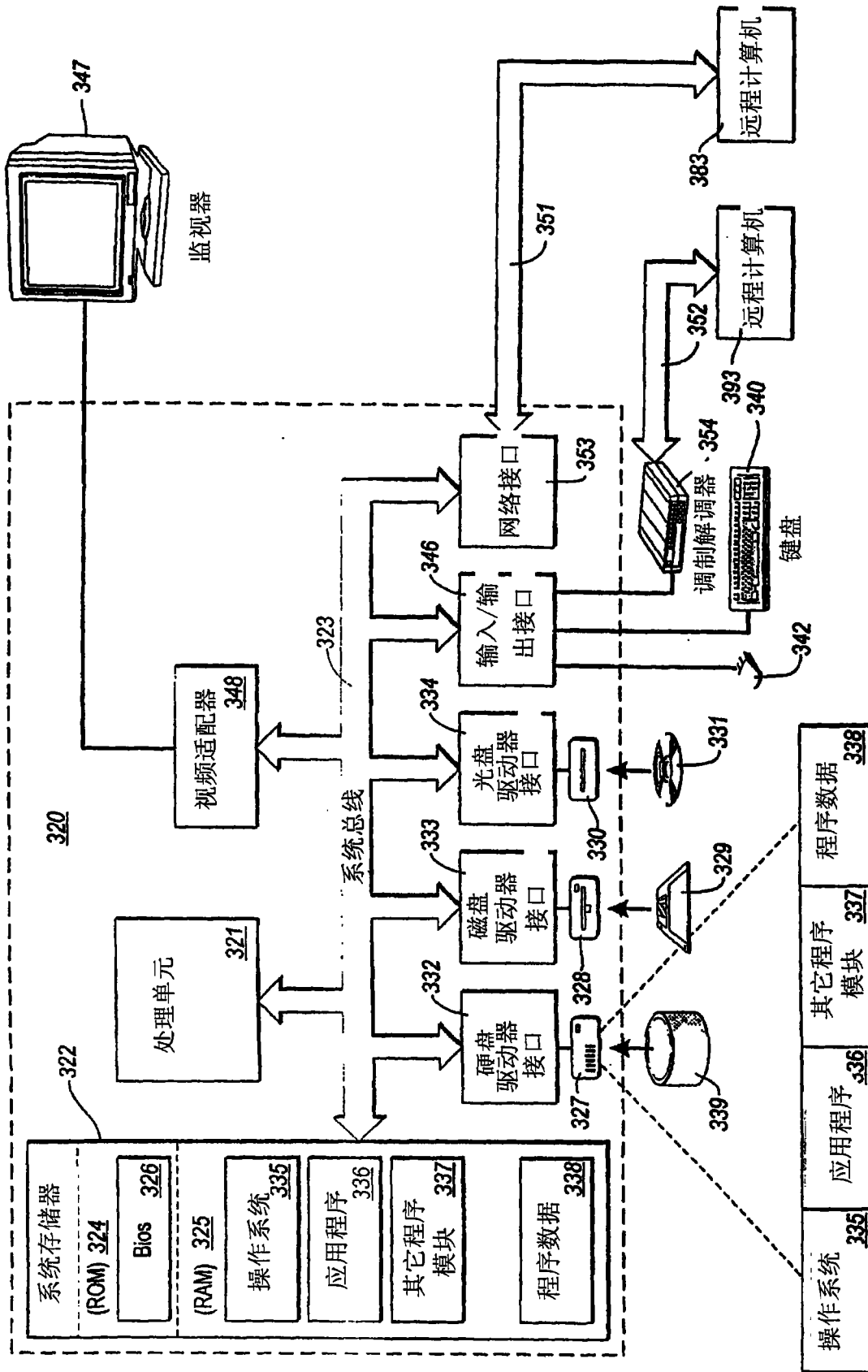


图 3