(19)中华人民共和国国家知识产权局

(12)发明专利

(10)授权公告号 CN 102197377 B
(45)授权公告日 2014.10.22

(21)申请号 200980143575.6
(22)申请日 2009.10.16
(30)优先权数据
12/258,439 2008.10.26 US
(85)PCT国际申请进入国家阶段日
2011.04.25
(86)PCT国际申请的申请数据
PCT/US2009/060977 2009.10.16
(87)PCT国际申请的公布数据
WO2010/048051 EN 2010.04.29
(73)专利权人 微软公司
地址 美国华盛顿州
(72)发明人 R·L·汤森 X·屠 B·D·斯格特
T·A·托尔塞特 K·W·赛克斯
S·S·普拉丹 J·A·蒂德
(74)专利代理机构 上海专利商标事务所有限公司 31100
代理人 顾嘉运
(51)Int.Cl.
G06F 3/041(2006.01)
G06F 3/0488(2013.01)
G06F 9/44(2006.01)
(56)对比文件
US 2007/0152984 A1,2007.07.05, 全文.
US 2008/0165141 A1,2008.07.10, 全文.
US 2008/0042986 A1,2008.09.21, 全文.
US 2006/0197753 A1,2006.09.07, 全文.
审查员 梁艳

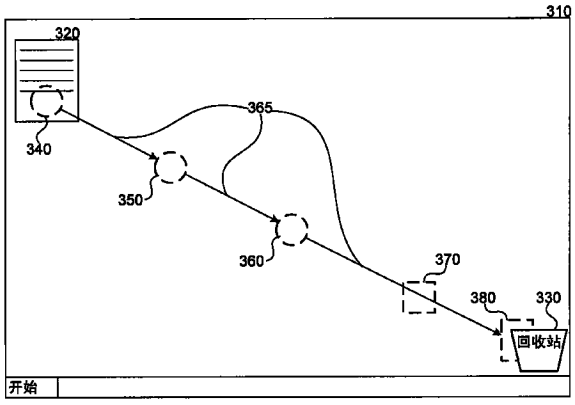
权利要求书2页 说明书11页 附图6页

(54)发明名称

多触摸对象惯性模拟

(57)摘要

惯性系统为应用程序提供一种公用的平台以及应用程序编程接口(API),以延伸从各种多触摸硬件设备接收到的输入,以模拟应用程序对象的现实世界的行为。为自然地移动,应用程序对象应该表现出诸如弹性和减速之类的物理特性。当用户从对象提起所有接触时,惯性系统向应用程序提供额外的操纵事件,以便该应用程序可以处理事件,好像用户仍在利用触摸移动对象。惯性系统基于对对象的行为的模拟,生成事件。如果用户将一个对象移动到到另一对象,则惯性系统模拟对象的边界特征。如此,惯性系统提供用户使用多触摸硬件操纵应用程序对象的更加逼真的移动,API 提供跨多个应用程序之间的操纵的一致的感觉。



CN 102197377 B

1. 一种用于提供使用多触摸输入操纵的对象的逼真移动的计算机实现的方法，所述方法包括：

确定（610）用户已经通过从所述计算机的多触摸输入设备移除一个或多个接触，释放了在所述计算机上执行的应用程序对象，所述应用程序对象被显示在所述计算机的显示器上；

在所述确定之后，调用（640）惯性系统的处理功能以在当前惯性处理周期内处理所述应用程序对象的移动的模拟，其中所述惯性系统提供一种应用程序独立的平台以及惯性API，所述惯性API独立于所述应用程序对象的类型来模拟惯性对象的移动；

由所述应用程序接收（650）在所述确定之后由所述惯性API生成的惯性事件，其中所述惯性事件基于模拟的惯性来描述所述对象的操纵；以及

通过修改所述应用程序对象，基于所述对象的操纵在所述应用程序的上下文中的影响，处理（660）所述接收到的惯性事件并在所述显示器上相应地显示所述应用程序对象。

2. 如权利要求1所述的方法，其特征在于，还包括，在判断所述用户释放了所述对象之后，通过传递描述当所述用户释放所述对象时所述对象的状态的一个或多个参数，初始化所述惯性系统。

3. 如权利要求1所述的方法，其特征在于，接收所述惯性事件包括接收描述所述应用程序对象的2D仿射变换的信息。

4. 如权利要求1所述的方法，其特征在于，通过所述惯性事件所描述的所述操纵包括旋转操纵、平移操纵，以及缩放操纵中的至少一个。

5. 如权利要求1所述的方法，其特征在于，所述接收到的惯性事件是旋转，其中处理所述接收到的惯性事件包括在显示器上旋转所述应用程序对象。

6. 如权利要求1所述的方法，其特征在于，还包括设置确定惯性处理周期长度的计时器，其中所述应用程序在所述计时器的每一个引发时调用所述惯性API。

7. 如权利要求6所述的方法，其特征在于，从所述惯性API接收所述模拟已完成的指示，当接收到所述指示时，使所述计时器过期。

8. 如权利要求1所述的方法，其特征在于，接收所述惯性事件包括通过COM事件接口，接收通知。

9. 一种用于控制计算机系统模拟应用程序对象的移动的方法，所述应用程序对象通过多触摸输入预先移动，所述方法包括：

接收（710）一个或多个初始模拟参数，所述初始模拟参数提供与用户通过停止与所述计算机系统的触摸输入设备的接触来释放所述应用程序对象时对应的所述应用程序对象的状态；

初始化（720）模拟引擎，所述模拟引擎基于所述初始模拟参数来执行计算以模拟所述应用程序对象的惯性；

接收（730）当前模拟周期的指示；

基于所述初始参数、任何前面的模拟引擎的惯性模拟以及自任何前面的模拟周期以来过去的时间，来模拟（740）所述应用程序对象的惯性移动；以及

引发（770）惯性事件以向所述应用程序发送描述所述应用程序对象的当前移动的变换信息，根据所述变换信息将所述应用程序对象显示在计算机系统的显示器上。

10. 如权利要求 9 所述的方法，其特征在于，所述模拟引擎在它通过所述用户触摸所述对象而启动之后模拟所述应用程序对象的逼真减速行为。

11. 如权利要求 9 所述的方法，其特征在于，当所述应用程序对象与显示器上的另一元素重叠时，所述模拟引擎模拟所述应用程序对象的逼真弹性行为。

12. 如权利要求 9 所述的方法，其特征在于，所述接收当前模拟周期到了推进模拟的时候的指示包括从所述应用程序接收对模拟处理函数的调用。

13. 如权利要求 9 所述的方法，其特征在于，还包括判断所述模拟是否完成，如果所述模拟完成，通知所述应用程序，所述模拟完成。

多触摸对象惯性模拟

背景技术

[0001] 平板PC或笔输式计算机，是配备有触摸屏或图形输入板/屏幕混合技术的笔记本或板式(slate shaped)移动计算机，这种技术可使用户利用指示笔、数字笔或指尖而不是键盘或鼠标来操作计算机。平板PC提供更自然的输入形式，因为勾画和手写是比键盘和鼠标熟悉得多的输入形式，特别是对于对计算机不熟悉的新人来说。平板PC也可以更容易访问，因为在物理上不能输入的那些人可以使用平板PC的附加特征能够与电子世界进行交互。

[0002] 多触摸表示可使计算机用户使用多个手指或输入设备（例如，指示笔）控制图形应用程序的一组交互技术。多触摸实现通常包括识别多个同时的触摸点的触摸硬件（例如，屏幕、平板、墙壁(wall)等等）和软件。多触摸是相对于一次只识别一个触摸点的传统的触摸屏（例如，计算机触摸板、ATM、购物自助服务终端）而言的。多触摸硬件可以使用热量、手指压力、高俘获率照像机、红外光、光学捕捉、调谐的电磁感应、超声波接收器、传感麦克风、激光测距仪、阴影捕捉，及其他机制，来感应触摸。存在许多多触摸接口的应用程序，应用程序设计人员和用户正在提出更多。某些用途是专用的（例如，Microsoft Surface、Apple iPhone、HTC Diamond）。作为新的输入方法，多触摸提供了新用户体验的范例的潜力。

[0003] 若没有供应用程序软件从多触摸硬件接收信息的接口，应用程序就不能使用多触摸硬件。令人遗憾的是，每一个多触摸硬件设备都包括其自己的专有的接口，应用程序作者必须具有硬件设备的特定知识来编写同该设备一道工作的软件。例如，多触摸硬件提供商可以提供内核模式驱动程序和用户模式应用程序接口，用户模式软件应用程序可以通过该接口与多触摸硬件进行通信，以接收触摸信息。应用程序作者编写与用户模式应用程序接口进行通信的软件，但是，应用程序作者的软件只与该多触摸硬件一道工作。带有不同的多触摸硬件设备的计算机用户不能使用该应用程序作者的软件，除非应用程序作者编制了正确地与计算机用户的设备一起操作的软件的不同的版本。这会为应用程序作者产生非常有限的潜在市场，降低了编写支持多触摸交互的应用程序的动机，并使具有最大数量的可用应用程序的最流行的设备的成本非常高。

[0004] 另一问题是应用程序难以基于从多触摸硬件接收到的触摸输入来确定用户的意图。触摸输入可以作为坐标列表来接收，其中，硬件在任何给定时间感应触摸输入。每一应用程序都必须包括解释坐标并确定用户的意图的软件。另外，用户的意图可以延伸到接收到的实际触摸输入之外。用户可以预期虚拟对象表现得如它们在物理世界中那样。例如，用户可以预期能够通过快速轻弹(flick)他/她的手指，将文件从桌面的一侧“抛”(toss)到另一侧。这种类型移动不受预期用户将他/她的手指从屏幕的一侧一直拖到另一侧的当前多触摸应用程序的支持。即使一个应用程序为此类型的移动提供支持，其他应用程序也不能从它那里得到好处，如此应用程序作者必须重复第一应用程序作者的工作以在他们的应用程序中提供相同功能。

发明内容

[0005] 惯性系统为应用程序提供一种公用的平台以及应用程序编程接口 (API)，以延伸从各种多触摸硬件设备接收到的输入，以模拟对象的现实世界的行为。由应用程序接收到的操纵只基于与多触摸硬件的接触的移动，来描述对象的移动。然而，为了自然地移动，对象也应该表现出诸如弹性和减速之类的物理特性。当用户从对象提起所有接触时，惯性系统向应用程序提供额外的操纵事件，以便该应用程序可以处理事件，好像用户仍在利用触摸移动对象。然而，惯性系统基于对对象的行为的模拟，实际生成事件。如果用户将一个对象移动入另一对象，则惯性系统基于对象的边界特性来发送操纵事件。如此，惯性系统提供用户使用多触摸硬件操纵的应用程序对象的更加逼真的移动，API 提供跨多个应用程序之间的操纵的一致的感觉。

[0006] 提供本概述是为了以精简的形式介绍将在以下详细描述中进一步描述的一些概念。本概述并不旨在标识出所要求保护的主题的关键特征或必要特征，也不旨在用于限定所要求保护的主题的范围。

附图说明

[0007] 图 1 是示出了一个实施例中的惯性系统的组件的框图。

[0008] 图 2 是示出了一个实施例中的惯性系统的典型的操作环境和组件之间的数据的流动的数据流程图。

[0009] 图 3 是示出了一个实施例中的通过用户触摸操纵的应用程序对象的显示图。

[0010] 图 4 是示出了一个实施例中的使用惯性系统处理操纵事件的多触摸应用程序的输入循环处理的流程图。

[0011] 图 5 是示出了一个实施例中的当系统接收到触摸输入时惯性系统的处理的流程图。

[0012] 图 6 是示出了一个实施例中的使用惯性系统处理惯性事件的多触摸应用程序的处理的流程图。

[0013] 图 7 是示出了一个实施例中的惯性处理系统的模拟组件的处理的流程图。

具体实施方式

[0014] 惯性系统为应用程序提供一种公用的平台以及 API，以延伸从各种多触摸硬件设备接收到的输入，以模拟对象的现实世界的行为。例如，当用户停止推现实世界的对象时，这些对象通常不会立即停止移动，而是表现出某种惯性，并继续移动，直到摩擦力使它们减慢，最后停止。在某些实施例中，触摸输入首先经历将一个或多个接触的移动解释为操纵的过程。操纵比单个触摸输入更加直接地映射到用户意图，并为使用多个触摸接触的对对象的基本变换添加了支持。应用程序可以使用操纵来同时支持旋转、缩放，以及平移多个对象（例如，照片）。可以将操纵描述为包含旋转、缩放（放大或缩小），以及平移（例如，平移）信息的二维 (2D) 仿射变换。

[0015] 对多触摸硬件的每一个触摸都叫做接触。例如，当用户将他 / 她的手指置于多触摸硬件上，四处移动他 / 她的手指，以及提起他 / 她的手指时，该系列事件是单个接触。例如，如果用户将两个接触移动得更靠近到一起或彼此更远离，则系统可以判断用户正在缩

放（例如放大或缩小）一个对象。作为另一个示例，如果用户以圆周运动方式移动多个接触，那么，系统可以将移动解释为对象的旋转。每一应用程序都可以以不同的方式定义相关的对象，如此，该由应用程序将系统的实例（叫做操纵处理器）附加到用户可以使用应用程序内的触摸输入来操纵的每一对象。例如，照片浏览应用程序可以将操纵处理器附加到每一个显示的照片，以便用户可以四处移动照片，缩放照片，旋转照片等等。

[0016] 由应用程序处理的操纵只基于接触的移动，来描述对象的移动。然而，为自然地移动，对象也应该表现出诸如弹性和减速之类的物理特性。当用户从对象提起所有接触时，惯性系统向应用程序提供额外的操纵事件，以便该应用程序可以处理事件，好像用户仍在利用触摸移动对象。然而，惯性系统基于对对象的行为的模拟，实际生成事件。例如，如果用户在对象在特定方向上具有速度时提起接触，那么，惯性系统持续发送指示该对象正在在该方向移动的事件，随时间当对象减速而减慢。如果用户将一个对象移动入另一对象，如屏幕的边缘，则惯性系统基于对象的边界特性来发送操纵事件。例如，如果应用程序作者将两个对象定义为弹性的，那么，当用户向彼此移动两个对象时，这两个对象会从彼此弹回。如此，惯性系统提供用户使用多触摸硬件操纵的应用程序对象的更加逼真的移动，API 提供跨多个应用程序之间的操纵的一致的感觉。

[0017] 图 1 是示出了一个实施例中的惯性系统的组件的框图。惯性系统 100 包括硬件接口 110、一个或多个操纵处理器 120、输入变换组件 130、模拟组件 140，以及应用程序接口 150。此处更详细地描述了这些组件中的每一个。

[0018] 硬件接口 110 与硬件进行通信，以接收触摸接触和移动。硬件接口 110 可包括一起协作以提供触摸输入信息的多个子组件。例如，操作系统可以为多触摸硬件制造商提供共同的驱动程序模型，以为他们的特定硬件提供触摸信息。操作系统可以将通过此模型接收到的触摸信息转换为窗口消息（例如，此处所描述的 WM TOUCH），并将这些消息传递到应用程序。如此，硬件接口 110 可以涉及硬件、硬件驱动程序、以及操作系统层的协调。结果是发往惯性系统的标识特定接触（例如，手指的触摸）以及随时间的接触的坐标的一系列消息。例如，当在多触摸硬件上有新的接触时，操作系统可以提供消息，每当接触移动时，提供消息，以及当接触离开多触摸硬件时，提供消息。

[0019] 一个或多个操纵处理器 120 使用输入变换组件 130 来解释与特定应用程序对象相关联的每一接触的移动。操纵处理器 120 可以判断用户正在使用多个触点以执行单一动作。例如，用户可以利用一只手的全部五个手指触摸照片，并扭曲他 / 她的手以指出旋转照片的意图。操纵处理器 120 接收五个单独的接触（每一个手指，一个接触），并随着用户旋转他 / 她的手，改变每一接触的坐标。操纵处理器 120 判断每一接触正在抓取同一个对象，并执行同一旋转。系统将通知应用程序，用户旋转了对象，但是，应用程序可以忽略用户是使用两个、五个或任何特定数量的手指或其他接触来执行旋转。这大大地简化了对应用程序的创作，因为应用程序作者可以处理与应用程序有关的那些类型的操纵，并将其留给惯性系统来解释从多触摸硬件接收到的每一个低级别的触摸输入的含义。

[0020] 操纵处理器 120 使用输入变换组件 130 来独自或协调地进行关于接收到的各种接触的移动的含义的判断。例如，如果用户正在利用两根手指操纵照片从而产生两个对应的输入接触，那么，操纵处理器 120 使用输入变换组件 130 来确定两个接触之间的相对移动的含义。如果两个接触移动离开，那么，输入变换组件 130 可以判断用户正在缩放对象，以改

变对象的大小。如果两个接触旋转，那么，输入变换组件 130 可以判断用户正在旋转对象。如果两个接触都在一个特定方向滑动，那么，输入变换组件 130 可以判断用户正在将对象平移到一个新的位置。虽然单独地讨论了每一种类型的移动，但是，注意，用户可以同时进行所有三种类型的移动，并且输入变换处理器可以将总的变换报告到应用程序。例如，用户可以在一个运动中旋转、缩放，和平移一个对象。

[0021] 在用户基于为对象定义的初始化参数和约束停止触摸对象之后，模拟组件 140 模拟应用程序对象的持续的移动。应用程序可以利用与对象相关联的操纵处理器 120 的最终状态初始化模拟组件 140。应用程序还可定义对象的各种特性，诸如对象的边界应该如何表现。模拟组件 140 使用基于物理学的技术来在用户释放了对象之后的时段模拟对象的行为。例如，模拟组件 140 可以如同在用户移动对象时以由应用程序接收到的操纵事件那样同样的形式向应用程序继续引发通知。然后，应用程序可以专注于对对象的移动作出反应而不关心什么动作（用户或物理）导致对象移动。本领域技术人员将认识到，用于以软件模拟虚拟对象的等效物理行为的很多已知的技术。

[0022] 应用程序接口 150 与应用程序进行通信，以接收信息并将操纵变换提供到应用程序。应用程序接口 150 从应用程序接收初始化信息。初始化信息可以指定对于特定对象，应用程序对象支持哪些类型的变换，以及相关联的操纵处理器，以及当用户不再移动对象时用于模拟组件 140 的初始化数据。例如，某些应用程序对象可以支持水平缩放，但不支持旋转。初始化信息还可指定对象的枢轴点。惯性系统通过应用程序接口将操纵变换提供到应用程序。例如，当惯性系统接收系统解释为识别的变换（例如，旋转）低级别的触摸输入时，系统引发将有关操纵的信息通知给应用程序的事件。应用程序处理操纵变换以基于变换来修改对象。例如，如果用户旋转对象，那么，应用程序可以存储对象的新的朝向，以在应用程序下一次显示对象时使用。作为另一个示例，如果基于模拟组件 140 的计算，在用户释放对象之后对象继续旋转，那么，应用程序可以存储对象的新的朝向。

[0023] 在其上面实现了系统的计算设备可包括中央处理单元、存储器、输入设备（例如，键盘和指示设备）、输出设备（例如，显示设备），以及存储设备（例如，磁盘驱动器）。存储器和存储设备是可以用实现系统的计算机可执行指令来编码的计算机可读介质，意味着包含指令的计算机可读介质。另外，数据结构和消息结构可以被存储或通过诸如通信链路上的信号之类的数据传输介质传输。可以使用各种通信链路，如因特网、局域网、广域网、点对点拨号连接、蜂窝电话网络等等。

[0024] 系统的各实施例可以在各种操作环境中实现，操作环境包括个人计算机、服务器计算机、手持式或膝上型设备、多处理器系统、基于微处理器的系统、可编程消费电子产品、数码相机、网络 PC、小型计算机、大型计算机、包括上述系统或设备中的任一个的分布式计算机环境等。计算机系统可以是蜂窝电话、个人数字助理、智能电话、个人计算机、可编程消费电子产品、数码相机等等。

[0025] 可以在由一台或多台计算机或其他设备执行的诸如程序模块之类的计算机可执行的指令的一般上下文中来描述本系统。一般而言，程序模块包括执行特定任务或实现特定抽象数据类型的例程、程序、对象、组件、数据结构等等。通常，程序模块的功能可以按需在各个实施例中进行组合或分布。

[0026] 图 2 是示出了一个实施例中的惯性系统的典型的操作环境和组件之间的数据的

流动的数据流程图。多触摸硬件设备通过硬件接口产生输入 210。例如，硬件可以通过由硬件制造商所提供的软件驱动程序向操作系统发送输入 210。硬件接口向应用程序 230 提供输入事件 220。例如，应用程序可以通知操作系统，应用程序 230 支持多触摸用户输入，并注册以接收与多触摸用户输入相关的消息。应用程序 230 作为输入变化 240 接收低级别的触摸输入信息，并将输入变化 240 转发到操纵系统 250。例如，输入变化 240 可以使用指示每一个接触的当前位置及其他移动特性的一组坐标，描述与硬件的一个或多个触摸接触的每一个移动。操纵系统 250 解释输入变化 240，并将一个或多个操纵事件 260 通知给应用程序 230，操纵事件 260 指示用户正在对显示的对象执行的较高级别的操纵。例如，如果接触的移动指示用户打算旋转对象，则操纵事件 260 指示旋转的程度。

[0027] 当用户完成移动对象（例如，当应用程序接收到已经从触摸硬件中移除了触摸对象的每一个接触的通知时），应用程序 230 向惯性系统 280 发送初始化信息 270。惯性系统 280 确定对象的下一位置，并提供类似于当用户移动对象时操纵系统 250 所提供的操纵事件 260 的惯性事件 290。应用程序 230 还提供驱动计时器，以周期性地调用惯性系统 280，以通过惯性事件 290 提供对象的下一位置。应用程序 230 以类似于操纵事件的方式处理惯性事件。

[0028] 虽然图示出了应用程序首先接收触摸输入并将触摸输入传递到操纵系统和惯性系统，但是，在某些实施例中，这些系统直接从硬件接口接收触摸输入，解释触摸输入，并将解释的操纵事件提供到应用程序。同样，应用程序可能不知道，在用户停止利用触摸移动对象之后，单独的惯性系统 280 提供惯性事件，而是可能在用户移动对象过程中并且然后当对象基于惯性而移动时从一个接口接收到事件。这表示提供类似的结果功能但是给予应用程序对于对输入的处理的较少的控制的替换的体系结构。例如，应用程序可能不能定义系统将单个操纵处理器附加到其中的单个应用程序对象。此处所描述的 RTS 插件是该系统的此替换的体系结构的一个示例。

[0029] 图 3 是示出了一个实施例中的通过用户触摸操纵的应用程序对象的显示图。应用程序可以同时显示并接收对于许多这样的对象的触摸输入。例如，操作系统外壳应用程序可以显示存储在用户的计算机桌面上的一个或多个文档对象。在显示器 310 中，文档对象 320 表示用户的桌面上的用户希望使用触摸拖到回收站 330 中的文档。用户对文档对象 320 执行快速轻弹，导致系统处理第一操纵位置 340、第二操纵位置 350，以及第三操纵位置 360。当用户最初利用一个或多个手指（即，接触）触摸文档对象 320 时，应用程序接收第一操纵位置 340。当用户在屏幕上滑动他 / 她的手指时，应用程序接收第二操纵位置 350。当用户从屏幕上提起他 / 她的手指时，应用程序接收第三操纵位置。箭头 365 表示文档对象 320 的移动的矢量。

[0030] 若没有惯性，文档对象 320 将停止在第三操纵位置 360，这很可能不是用户想要的位置。基于当用户释放文档对象 320 时文档对象 320 的速度，惯性系统向应用程序提供额外的操纵位置，就好像用户仍在触摸和移动文档对象 320。当应用程序初始化惯性系统并第一次调用惯性系统的处理功能时，应用程序接收第一基于惯性的操纵位置 370。当应用程序继续调用惯性系统的处理功能时，应用程序接收第二基于惯性的操纵位置 380。因为文档对象 320 的最后的操纵位置 380 在回收站 330 上方，因此，应用程序处理两个对象之间的接触（例如，通过将文档对象 320 放在回收站 330 中）。在所示出的示例中，尽管惯性系统使

文档对象 320 的移动减速，基于开始时用户移动文档对象 320 的高初始速度，文档对象 320 仍能在显示器 310 上移动相当的距离。

[0031] 图 4 是示出了一个实施例中的使用惯性系统处理操纵事件的多触摸应用程序的输入循环处理的流程图。在框 410 中，应用程序接收低级别的触摸输入。例如，操作系统或惯性系统的实例从多触摸硬件接收触摸接触信息，并将触摸接触信息转发到应用程序。在框 420 中，应用程序标识输入应用到的对象。例如，应用程序可以通过将接收到的输入的坐标与由应用程序显示的每一应用程序对象的坐标进行比较，来点击测试 (hit test) 接收到的输入的坐标。如果触摸输入在显示的应用程序对象的边界内，那么，应用程序判断触摸输入应用于该对象。在框 430 中，应用程序将接收到的触摸输入和有关标识的应用程序对象的信息发送到用于调用惯性系统的操纵 API（参见图 5）。例如，应用程序可以为每一个应用程序对象创建数字标识符，并在每当触摸输入与该对象相对应时将该数字标识符传递到惯性系统。

[0032] 在框 440 中，应用程序从惯性系统接收描述对已标识的应用程序对象的一个或多个操纵的操纵事件。例如，应用程序可以接收描述对应用程序对象的 2D 仿射变换的事件。注意，为简化说明，在框 430 之后按顺序地示出了框 440。在实践中，在惯性系统利用操纵事件通知应用程序之前，应用程序可以接收许多触摸输入事件。在触摸输入事件与操纵事件之间不需要有一对一映射。因为操纵事件表示对低级别的触摸输入的较高级别的解释，因此，多个触摸输入可以构成单个操纵事件。在框 450 中，应用程序处理接收到的操纵事件。例如，如果接收到的操纵事件是旋转，那么，应用程序可以在屏幕上旋转应用程序对象，并给应用程序对象存储新的位置，供当应用程序对象再次被显示时使用。惯性系统免除应用程序执行特定多触摸硬件设备所特定的步骤，或者甚至不必知道哪一个硬件设备正在提供多触摸输入。另外，惯性系统免除应用程序处理单个接触移动，并允许应用程序专注于处理应用程序对象级别的变换。

[0033] 在框 460 中，应用程序等待下一触摸输入。例如，应用程序可以调用操作系统所提供的消息 API，如 Microsoft Windows 上的 GetMessage，等待要被传送给应用程序的消息队列的下一消息。在判断框 470，如果应用程序接收到下一触摸输入，那么，应用程序循环到框 410，以处理输入，否则，应用程序循环到框 460，以继续等待进一步的输入。当应用程序关闭时，应用程序退出输入循环（未示出）。

[0034] 图 5 是示出了一个实施例中的当系统接收到触摸输入时惯性系统的处理的流程图。在框 505 中，系统接收触摸输入以及标识与触摸输入相关联的应用程序对象的信息。例如，触摸输入可包括一个或多个触摸接触的坐标或其他位置信息，而应用程序对象信息可包括应用程序给多触摸硬件上的触摸输入所在的特定显示的对象指定的标识符。在框 510 中，系统标识与应用程序对象相关联的操纵处理器。在判断框 520，如果系统没有预先将操纵处理器与应用程序对象相关联，那么，系统继续执行框 530，否则，系统继续执行框 540。在框 530 中，系统创建操纵处理器，并将它与应用程序对象相关联，然后，继续执行框 540。

[0035] 在判断框 540，如果接收到的触摸输入指示应用程序接收到了新的接触（例如，touch down 事件），那么，系统继续执行框 550，否则系统继续执行框 560。例如，用户可以用手指与屏幕上的对象进行初次接触，或在以前触摸的对象上放另一手指（即，接触）。在框 550 中，系统向与操纵处理器相关联的接触的列表添加新的接触，然后继续执行框 560。在

判断框 560，如果接收到的触摸输入指示应用程序接收到了触摸接触被移除的通知（例如，触摸提起（touch up）事件），那么，系统继续执行框 570，否则系统继续执行框 580。例如，用户可以从以前触摸的对象提起一个或多个手指。在框 570 中，系统从与操纵处理器相关联的接触的列表中移除接触，然后继续执行框 580。在框 580 中，系统处理触摸输入以确定由触摸输入所表示的任何操纵。例如，触摸移动可以指示旋转或平移操纵，而触摸接触移除可以指示操纵完成。在框 590 中，系统引发向应用程序发送描述操纵的变换信息的操纵事件。例如，系统可以向所述应用程序提供所述对象的角旋转程度。在框 590 之后，这些步骤结束。

[0036] 图 6 是示出了一个实施例中的使用惯性系统处理惯性事件的多触摸应用程序的处理的流程图。在框 610 中，应用程序判断用户已经释放了对象。例如，在对图 4 的框 450 中的操纵事件进行处理之后，应用程序可以接收操纵已完成的指示或用户已经提起触摸应用程序对象的所有接触的指示。在框 620 中，应用程序初始化惯性系统。例如，应用程序可以将引用传递给处理对象的移动及其他初始化信息的操纵处理器。在框 630 中，应用程序设置将驱动惯性系统的惯性处理周期的计时器。例如，应用程序可以设置每隔 100 毫秒将引发以处理对象的下一移动增量的计时器。在框 640 中，应用程序调用惯性系统的处理功能（参见图 7）。例如，惯性系统可以提供“处理”函数 (Process function)，应用程序调用该函数来通知惯性系统：是从自最后的模拟时段以来的周期内执行模拟的时候了。

[0037] 在框 650 中，应用程序接收基于模拟的惯性来描述对对象的操纵的一个或多个惯性事件（例如，旋转、平移，和 / 或缩放）。例如，如果对象在特定方向移动，则应用程序可以接收描述在该方向的平移操纵的惯性事件。作为另一个示例，如果当用户释放对象时该对象正在放大，则应用程序可以接收描述缩放操纵的惯性事件。注意，为简化说明，在框 640 之后按顺序地示出了框 650。在实践中，在惯性系统利用惯性事件通知应用程序之前，应用程序可以多次调用惯性处理函数。在对处理函数和惯性事件的调用之间不需要有一对一映射。另一方面，在对处理函数的单一调用之后，惯性系统可以将多个惯性事件通知给应用程序。

[0038] 在框 660 中，在特定应用程序的上下文中，应用程序基于操纵的含义（例如，效果），来处理接收到的惯性事件。例如，如果接收到的惯性事件是旋转，那么，应用程序可以在屏幕上旋转应用程序对象，并给应用程序对象存储新的位置，以供当应用程序再次显示应用程序对象时使用。在判断框 670，如果惯性事件已完成，那么，这些步骤结束，否则，系统继续执行框 680。惯性系统可以作为来自处理函数的返回值或通过提供到应用程序的通知（例如，通过组件对象模型 (COM) 事件接口），来通知应用程序：特定模拟的操纵已完成。在框 680 中，应用程序等待计时器的下一次引发，然后，循环到框 640，以调用惯性系统处理函数。

[0039] 图 7 是示出了一个实施例中的惯性处理系统的模拟组件的处理的流程图。在框 710 中，组件接收初始模拟参数。例如，当用户停止触摸对象时，应用程序或操纵处理器可以提供应用程序对象的最终状态。在框 720 中，组件初始化基于物理学执行计算的模拟引擎以基于参数确定对象的行为的。例如，模拟引擎可以提供通过用户触摸输入启动的应用程序对象的逼真减速或弹性行为。在框 730 中，组件从应用程序接收处理调用。应用程序或其他组件通过每隔一定的时间间隔反复地调用处理函数以推进模拟，来驱动模拟过程。模

拟组件还可在内部生成计时器。

[0040] 在框 740 中，组件基于初始参数，任何前面的处理，以及自最后的处理调用经过的事件，来模拟对象的移动。处理调用还可提供指示应用程序希望模拟使用的时间的时间戳。这允许应用程序非实时地（例如，用于应用程序测试或调试）模拟应用程序行为。在判断框 750，如果移动完成，那么，组件继续执行框 760，否则，组件继续执行框 770。组件可以基于诸如对象是否仍正在移动或者对象移动是否已经低于某一阈值之类的因素，判断移动已完成。在框 760 中，组件在下一惯性事件上设置完成标志。在框 770 中，组件引发惯性事件以向应用程序发送描述当前移动（例如，作为操纵）的变换信息。例如，系统可以向所述应用程序提供所述对象的角旋转程度。在框 770 之后，这些步骤结束。

[0041] 在某些实施例中，惯性系统从应用程序接收对象约束。例如，应用程序可以定义对象的弹性、摩擦系数（来确定对象如何减速），对象的边界特性，等等。例如，应用程序作者可以定义用户可以移动的刚性对象以及有弹性的应用程序窗口边缘，以便当用户释放对象时，移入窗口边缘的对象从窗口边缘弹回。

[0042] 在某些实施例中，当用户利用触摸输入来操纵对象时，惯性系统从跟踪对象的移动的操纵系统接收初始对象状态信息。例如，操纵系统可以跟踪每一对象的当前位置，对象的历史移动，对象的线速度和角速度等等。应用程序作者可以向惯性系统提供操纵的输出，以初始化惯性系统，以便惯性系统可以基于适当的物理学和对象的特征，平滑地继续对象的经过移动，并使其减慢。

[0043] 在某些实施例中，惯性系统从应用程序接收对对象的移动的限制。例如，应用程序作者可以对一旦用户释放了对象之后对象可以移动的距离定义上限。作为另一个示例，应用程序作者可以对一旦用户释放了对象之后对象可以移动多长时间定义上限。这些及其他限制允许应用程序作者调整惯性系统以适合由应用程序操纵的对象的类型，并增强用户使用应用程序的体验。

[0044] 在某些实施例中，惯性系统不为具有低于预定义的阈值的移动的对象提供额外的移动。阈值可以是可由应用程序配置的。例如，惯性系统可以具有特定对象线性速度或角速度，低于该速度，系统在用户释放对象之后将不会继续对象的移动。如果对象在用户释放它时没有移动得非常快，则用户可能希望对象将原位不动，不继续移动。该阈值允许应用程序或惯性系统的作者确定提供好的用户体验的操纵之后的移动级别。

[0045] 在某些实施例中，惯性系统从应用程序接收递增模拟移动的指令。例如，惯性系统可以提供“处理”(Process) 或“工作”(DoWork) 函数，应用程序调用这些函数来命令惯性系统执行总的模拟的一部分。惯性系统可以预期应用程序设置计时器或周期性地调用该函数，以使惯性系统根据自然的时间线来模拟某时间段内的移动。应用程序可以通过改变应用程序调用函数的频率来影响由惯性系统所提供的操纵事件的特性。在其他实施例中，惯性系统使用内部计时器来以定期的调度表方式提供操纵事件，直到每一对象都已经停止移动（例如，由于减速或其他模拟的力）。

[0046] 在某些实施例中，惯性系统是基于消息的操作系统的一部分，该系统接收与操作系统从硬件接收到的触摸输入相关的消息。例如，使用类似于对于鼠标消息的 WM_MOUSEMOVE 的范例，Microsoft Windows 的未来版本可以提供 WM_TOUCH 消息，该消息包含从多触摸硬件接收到的低级别的触摸移动信息。操作系统还可提供更精细粒度的消息，如 WM_

TOUCHDOWN(当用多触摸硬件作出新的接触时)、WM_TOUCHMOVE(当现有的接触移动时),以及 WM_TOUCHUP(当从多触摸硬件提起接触时)。接收与 WM_TOUCH 相关的消息的应用程序可以调用惯性系统,并将消息传递到惯性系统供解释和处理。然后,应用程序接收较高级别的事件,这些事件表示惯性系统基于接收到的低级别的触摸移动信息,对由用户打算的操纵的解释。

[0047] 在某些实施例中,惯性系统从诸如实时指示笔之类的专门硬件接收低级别的触摸移动信息。例如,Microsoft 平板 PC 软件开发工具包 (SDK) 提供了应用程序作者可以利用挂钩扩展的实时指示笔 (RTS) 组件。RTS 挂钩接收来自 RTS 硬件的输入,并可以对接收到的输入执行处理。惯性系统可以提供应用程序可以插入到 RTS 组件中的挂钩,以自动处理 RTS 及其他输入,以操纵应用程序对象,如此处所描述的。RTS 挂钩提供了惯性系统接收输入的不同的方式,但是,惯性系统解释输入并向应用程序引发描述由输入所暗示的操纵的事件,如前所述。用户可以使用指示笔和触摸输入的组合。例如,用户可以利用指示笔画一个对象,然后,使用他 / 她的手指旋转对象。

[0048] 在某些实施例中,惯性系统是应用程序可以调用以提供公用用户界面的公用控件的一部分。Microsoft Windows 提供用于显示列表、树、按钮等等的公用控件。同样,惯性系统可以提供用于以此处所描述的方式操纵应用程序对象的基于多触摸的控件。例如,系统可以提供允许用户显示一个或多个对象并操纵对象的分散 (scatter) 控件。分散控件负责低级别的触摸输入的处理并将输入与特定应用程序对象相关联,应用程序从该控件接收事件以处理对应用程序对象的操纵。例如,如果该控件指示用户缩放了对象,那么,应用程序可以存储对象的新大小。

[0049] 在某些实施例中,惯性系统在三维空间执行此处所描述的处理。虽然此处描述了二维多触摸硬件,但是,本领域技术人员将认识到,此处所描述的系统的处理可以同样应用于三维 (3D) 操纵,如果有硬件可用来在三维空间中提供坐标移动。例如,检测压力或使用照像机来检测用户的手指的 3D 移动的硬件可以向惯性系统提供三维空间中的移动的坐标,然后,惯性系统可以产生描述在多个 3D 方向对对象的操纵 (例如,旋转、缩放,以及平移) 的 3D 变换。

[0050] 下表定义了一个 API,惯性系统将其提供到应用程序,用于在对象的基于用户触摸的移动之后,将基于惯性的移动提供到应用程序对象。

[0051]

属性：	
BoundaryBottom	限制目标对象可以向屏幕的底部移动多远。
BoundaryLeft	限制目标对象可以向屏幕的左侧移动多远。
BoundaryRight	限制目标对象可以向屏幕的右侧移动多远。
BoundaryTop	限制目标对象可以向屏幕的顶部移动多远。
DesiredAngularDeceleration	指定目标对象将停止旋转的所希望的速度（以弧度每毫秒表示）。
DesiredDecleration	指定平移操作减速的所希望的速率。
DesiredDisplacement	指定对象将移动的所希望的距离。
DesiredExpansion	指定对象的平均半径的所希望的变化。
DesiredExpansionDeceleration	指定对象将停止展开的速率。
ElasticMarginBottom	指定用于弹回目标对象的底部区域。
ElasticMarginLeft	指定用于弹回目标对象的最左边的区域。
ElasticMarginRight	指定用于弹回目标对象的最右边的区域。
InitialAngularVelocity	指定当移动开始时目标的旋转。
InitialOriginX	获取或放入指定目标对象的水平位置的属性。此属性指定带有惯性的目标的开始平面位置。
InitialOriginY	获取或放入指定目标对象的垂直位置的属性。此属性指定带有惯性的目标的开始垂直位置。
InitialRadius	指定在对象变化之前从目标的边缘到其中心的距离。
InitialTimestamp	指定带有惯性的目标对象的开始时间戳。
InitialVelocityX	指定目标对象在水平轴上的初始移动。
InitialVelocityY	指定目标对象在垂直轴上的初始移动。
方法：	
HRESULT Reset();	利用初始时间戳初始化处理器。
HRESULT Process([out] BOOL* completed);	对于给定记号，执行计算，并可以依赖外插是否已完成来引发增量或完成（Delta or Completed）事件。如果外插在前面的记号结束，方法是空操作。

[0052]

HRESULT ProcessTime([in] DWORD timestamp, [out] BOOL* completed);	对于给定记号，执行计算，并可以依赖外插是否已完成来引发增量或完成（Delta or Completed）事件。如果外插在前面的记号结束，方法是空操作。
HRESULT Complete();	引发完成事件。
HRESULT CompleteTime([in] DWORD timestamp);	处理给定记号，并引发完成事件。
事件：	
HRESULT ManipulationStarted(**[in] FLOAT x,** **[in] FLOAT y** **);**	处理当操纵开始时的事件。
HRESULT ManipulationDelta(**[in] FLOAT x,** **[in] FLOAT y,** **[in] FLOAT translationDeltaX,** **[in] FLOAT translationDeltaY,** **[in] FLOAT scaleDelta,** **[in] FLOAT expansionDelta,** **[in] FLOAT rotationDelta,** **[in] FLOAT cumulativeTranslationX,** **[in] FLOAT cumulativeTranslationY,** **[in] FLOAT cumulativeScale,** **[in] FLOAT cumulativeExpansion,** **[in] FLOAT cumulativeRotation** **);**	处理当操纵的对象变化时发生的事件
HRESULT ManipulationCompleted(**[in] FLOAT x,** **[in] FLOAT y,** **[in] FLOAT cumulativeTranslationX,** **[in] FLOAT cumulativeTranslationY,** **[in] FLOAT cumulativeScale,** **[in] FLOAT cumulativeExpansion,** **[in] FLOAT cumulativeRotation** **);**	处理当操纵结束时的事件。

[0053] 在上面的表中，惯性系统可以在相同接口（在该接口上，应用程序以前接收基于用户移动的事件）上提供列出的事件。

[0054] 从前面的描述中可以看出，可以理解，此处描述的惯性系统的特定实施例只是为了说明，但是，在不偏离本发明的精神和范围的情况下，可以进行各种修改。例如，虽然在多触摸操纵的上下文中描述了系统，但是，系统提供了可以在诸如游戏之类的其他上下文及通常使用模拟的其他领域中使用的惯性的模拟。相应地，本发明不受限制，只受所附的权利要求书的限制。

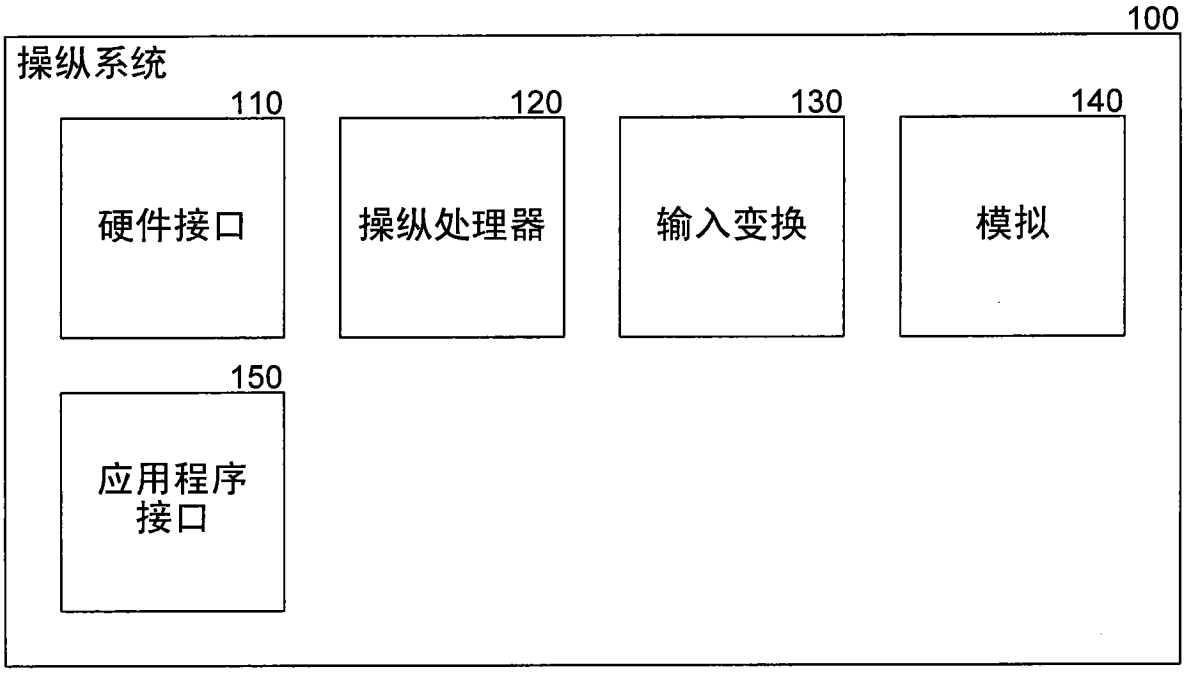

100
操纵系统
110
硬件接口
120
操纵处理器
130
输入变换
140
模拟
150
应用程序
接口

图 1

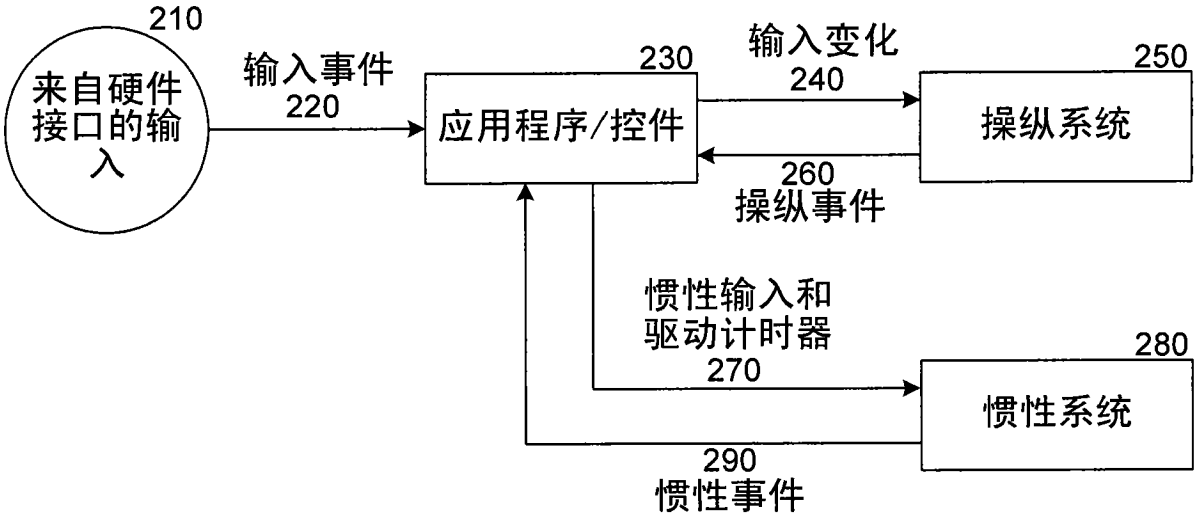

210
来自硬件
接口的输
入
输入事件
220
230
应用程序/控件
输入变化
240
250
操纵系统
260
操纵事件
惯性输入和
驱动计时器
270
280
惯性系统
290
惯性事件

图 2

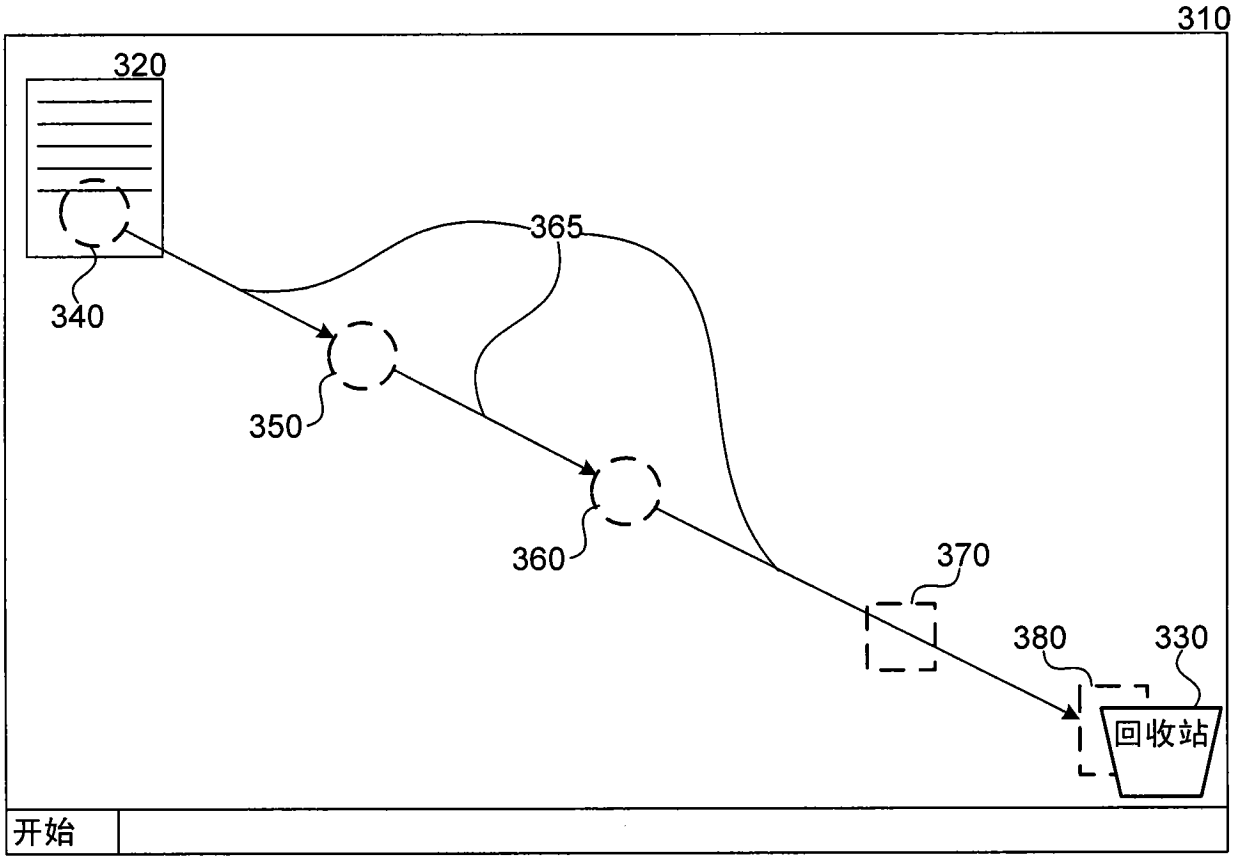


图 3

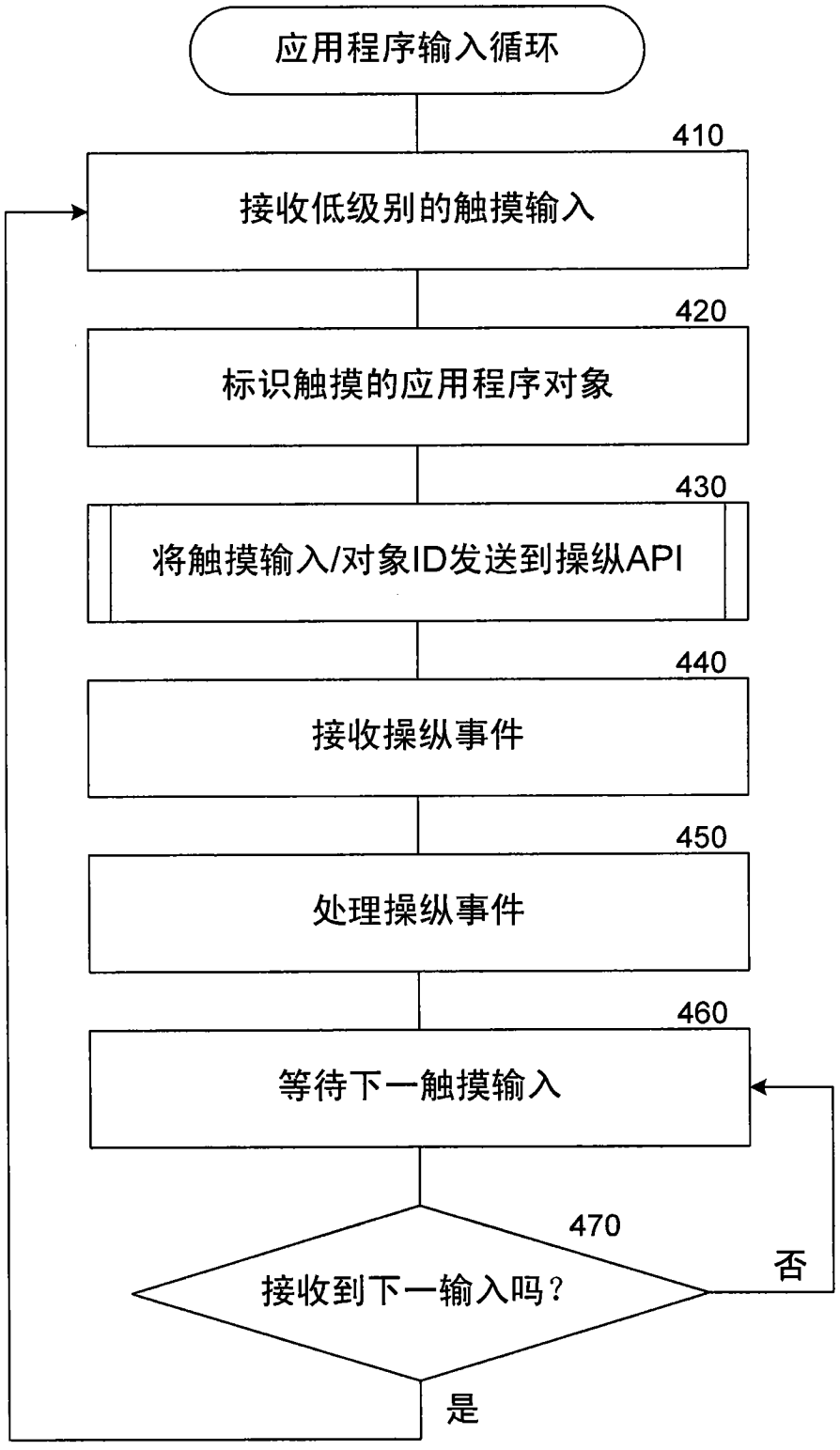

应用程序输入循环
410
接收低级别的触摸输入
420
标识触摸的应用程序对象
430
将触摸输入/对象ID发送到操纵API
440
接收操纵事件
450
处理操纵事件
460
等待下一触摸输入
470
接收到下一输入吗？
否
是

图 4

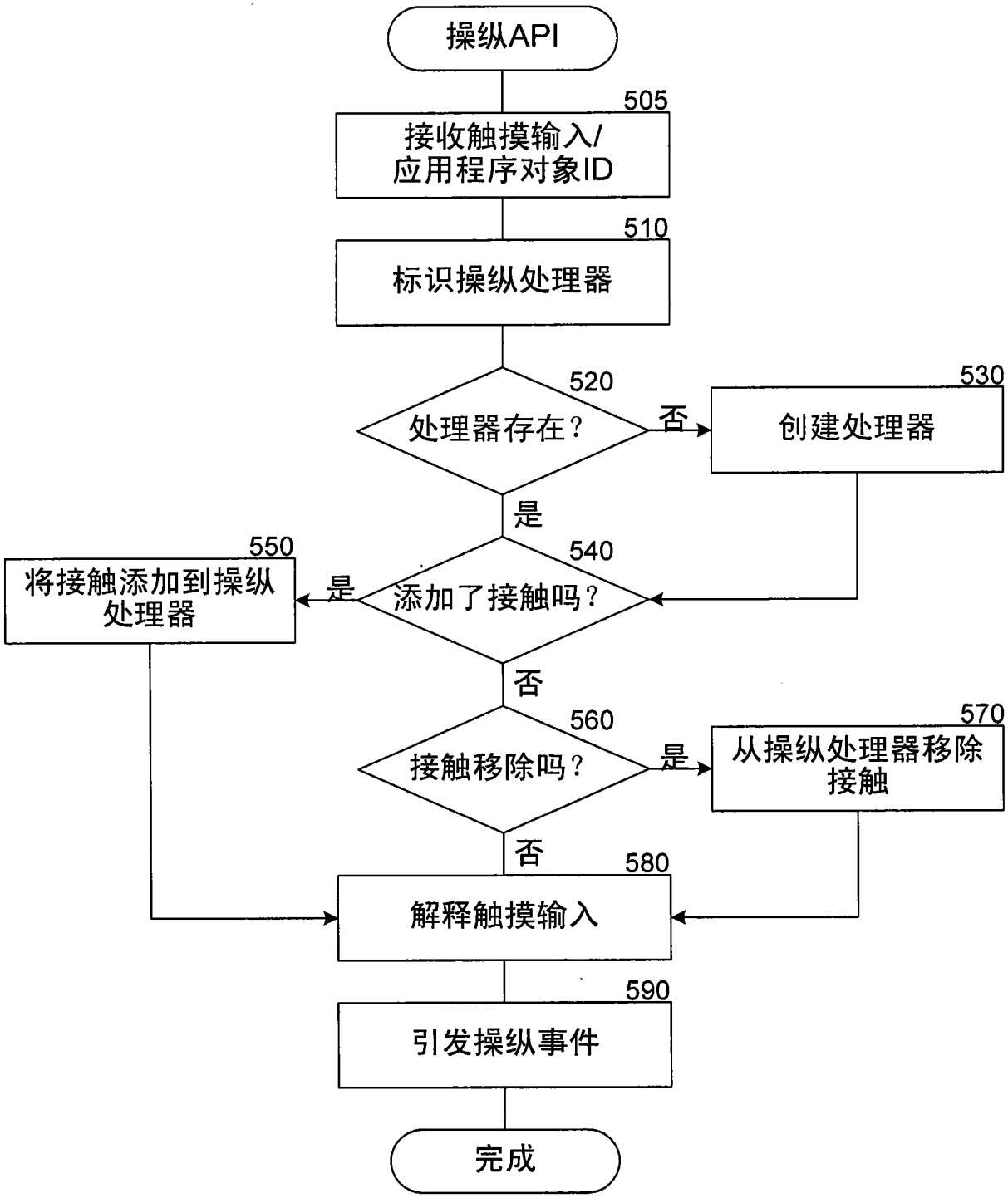

操纵API
505
接收触摸输入/
应用程序对象ID
510
标识操纵处理器
520
处理器存在？
否
530
创建处理器
是
550
将接触添加到操纵
处理器
是
540
添加了接触吗？
否
560
接触移除吗？
是
570
从操纵处理器移除
接触
否
580
解释触摸输入
590
引发操纵事件
完成

图 5

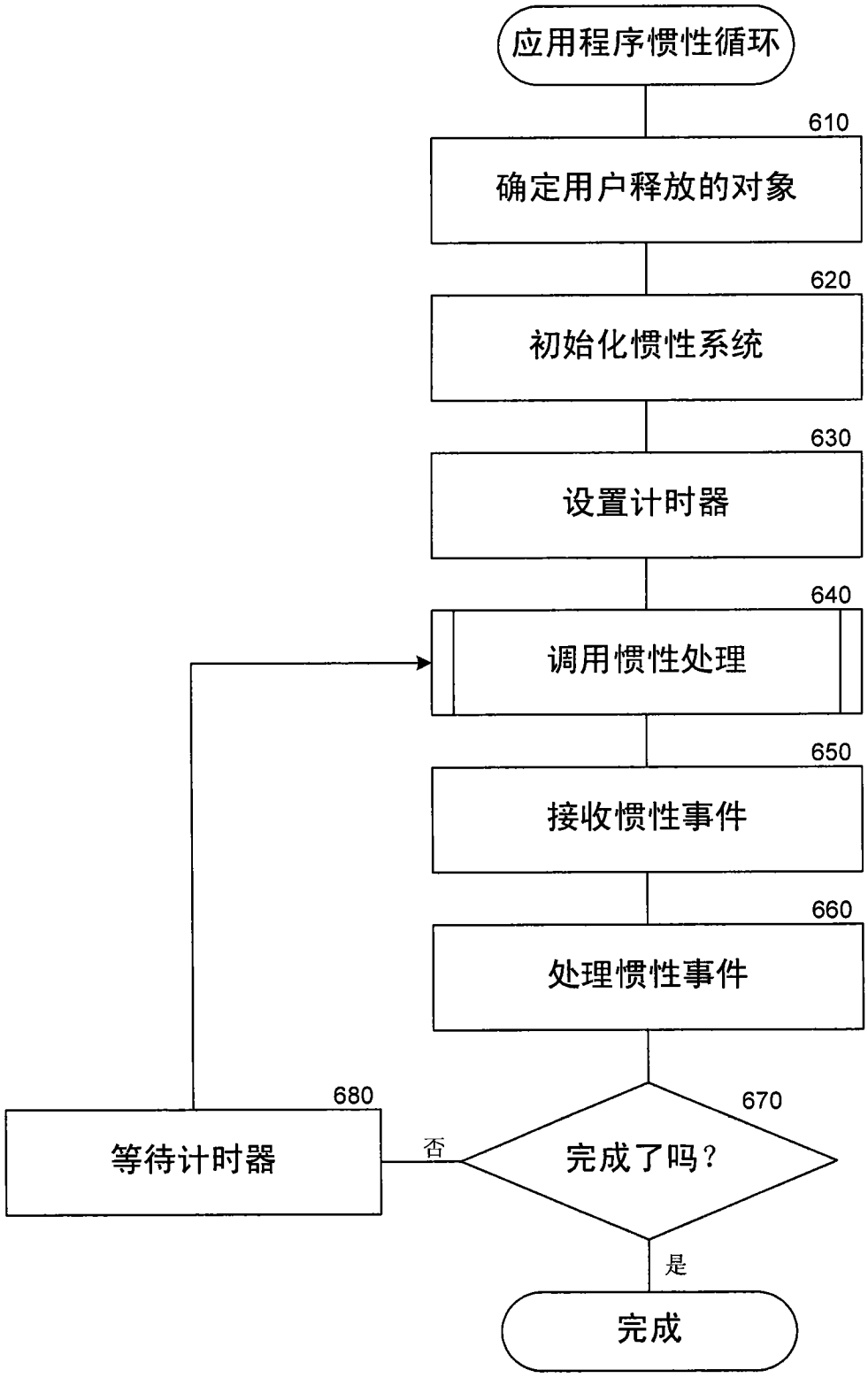

应用程序惯性循环
610
确定用户释放的对象
620
初始化惯性系统
630
设置计时器
640
调用惯性处理
650
接收惯性事件
660
处理惯性事件
670
完成了吗？
否
680
等待计时器
是
完成

图 6

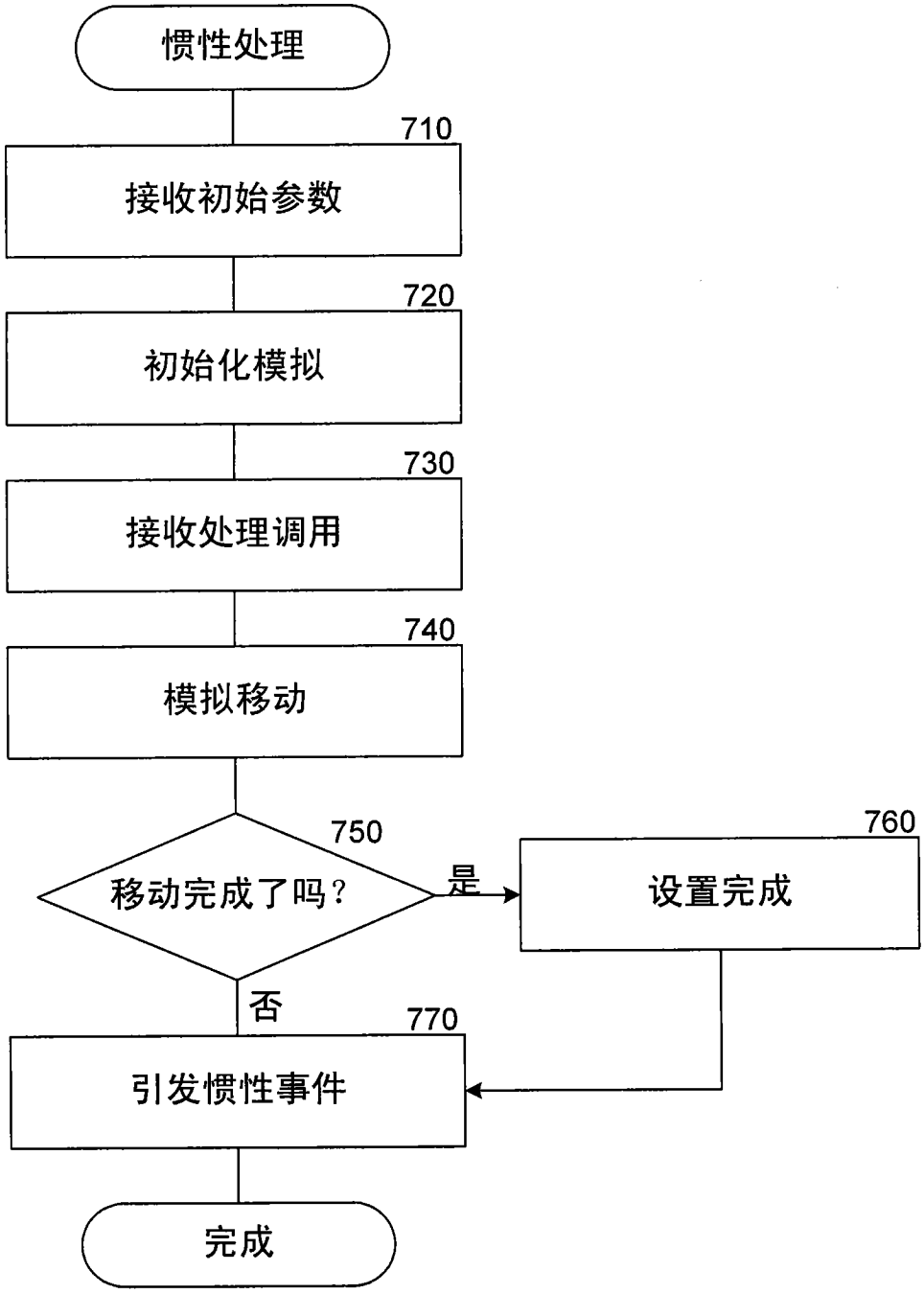

惯性处理
710
接收初始参数
720
初始化模拟
730
接收处理调用
740
模拟移动
750
移动完成了吗？
是
760
设置完成
否
770
引发惯性事件
完成

图 7