(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 109783144 B
(45)授权公告日 2022.03.25

(21)申请号 201711116727.6
(22)申请日 2017.11.13
(65)同一申请的已公布的文献号
申请公布号 CN 109783144 A
(43)申请公布日 2019.05.21
(73)专利权人 深圳市创客工场科技有限公司
地址 518000 广东省深圳市南山区学苑大道1001号南山智园C3栋4楼
(72)发明人 李晓峰 颜民革
(74)专利代理机构 深圳市隆天联鼎知识产权代理有限公司 44232
代理人 刘抗美 叶虹
(51)Int.Cl.
G06F 8/71(2018.01)
G06F 8/34(2018.01)
G06F 9/50(2006.01)
G06F 9/54(2006.01)

(56)对比文件
CN 104881330 A,2015.09.02
CN 102576253 A,2012.07.11
US 2011264527 A1,2011.10.27
CN 106681479 A,2017.05.17
CN 107315481 A,2017.11.03
CN 102971723 A,2013.03.13

审查员 陈俊如

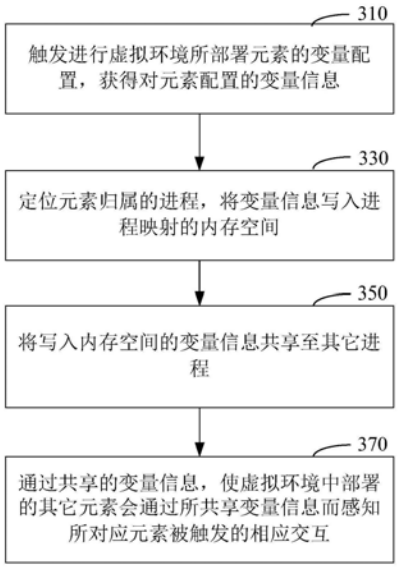
权利要求书2页 说明书13页 附图6页

(54)发明名称

虚拟环境交互实现中变量的处理方法、装置和存储介质

(57)摘要

本发明揭示了一种虚拟环境交互实现中变量的处理方法、装置和计算机可读存储介质。所述方法包括:触发进行虚拟环境所部署元素的变量配置,获得对元素配置的变量信息;定位元素归属的进程,将变量信息写入进程映射的内存空间;将写入内存空间的变量信息共享至其它进程;通过共享的变量信息,使虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互。在元素自定义配置所涉及变量的配置中所引入变量信息共享机制,也得以随之实现所自定义配置的变量在其它元素的共享,由此,使得所自定义配置的元素能够与其它变量相互感知,并触发相应交互,完成进行自定义的元素不再预置所执行的动作,能够进行元素之间的自由交互。



CN 109783144 B

1.一种虚拟环境交互实现中变量的处理方法,其特征在于,所述方法包括:

触发进行虚拟环境所部署元素的变量配置,获得对所述元素配置的变量信息,所述变量信息从所部署元素执行指定动作的控制逻辑提取得到;

定位所述元素归属的进程,将所述变量信息写入所述进程映射的内存空间;

根据被写入所述内存空间的变量信息,所述元素归属的进程发起广播,将所述变量信息在各进程所映射内存空间同步存储,以将所述变量信息共享至其它进程,所广播的数据包携带所述变量信息;

通过共享的所述变量信息,使所述虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互,并通过自身控制逻辑的执行进行响应。

2.根据权利要求1所述的方法,其特征在于,所述触发进行虚拟环境所部署元素的变量配置,获得对所述元素配置的变量信息,包括:

所述元素对应可供配置的动作块中触发进行指定动作块的拖动,获取所述指定动作块形成的控制逻辑;

由所述控制逻辑提取得到对所述元素配置的变量信息。

3.根据权利要求2所述的方法,其特征在于,所述元素包括渲染显示于所述虚拟环境的角色以及所述角色关联的物理硬件;

所述元素对应可供配置的动作块中触发进行指定动作块的拖动,获取所述指定动作块形成的控制逻辑,包括:

所述角色对应可供配置的动作块中,根据所述角色和物理硬件之间交互进行角色所执行动作以及物理硬件所执行动作的动作块配置,确定所述角色和物理硬件分别对应的指定动作块;

由指定动作块和代码信息之间的映射关系以及指定动作块之间的拼接关系,分别获得所述角色和物理硬件对应的控制逻辑。

4.根据权利要求1所述的方法,其特征在于,将写入内存空间的所述变量信息共享至其它进程之后,所述方法包括:

进程所映射内存空间中触发进行变量信息的修改操作,通过所述修改操作的执行在所述内存空间创建变量信息,或者修改变量信息中的数值;

根据执行修改操作而获得的变量信息进行进程之间的广播,以向其它进程共享所述变量信息。

5.一种虚拟环境交互实现中变量的处理装置,其特征在于,包括:

变量配置模块,用于触发进行虚拟环境所部署元素的变量配置,获得对所述元素配置的变量信息,所述变量信息从所部署元素执行指定动作的控制逻辑提取得到;

写入模块,用于定位所述元素归属的进程,将所述变量信息写入所述进程映射的内存空间;

共享模块,用于根据被写入所述内存空间的变量信息,所述元素归属的进程发起广播,将所述变量信息在各进程所映射内存空间同步存储,以将所述变量信息共享至其它进程,所广播的数据包携带所述变量信息;

交互感知模块,用于通过共享的所述变量信息,使所述虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互,并通过自身控制逻辑的执行进

行响应。

6.根据权利要求5所述的装置,其特征在于,所述变量配置模块包括:

动作块配置单元,用于所述元素对应可供配置的动作块中触发进行指定动作块的拖动,获取所述指定动作块形成的控制逻辑;

变量提取单元,用于由所述控制逻辑提取得到对所述元素配置的变量信息。

7.根据权利要求5所述的装置,其特征在于,所述装置还包括:

修改操作执行模块,用于进程所映射内存空间中触发进行变量信息的修改操作,通过所述修改操作的执行在所述内存空间创建变量信息,或者修改变量信息中的数值;

修改广播模块,用于根据执行修改操作而获得的变量信息进行进程之间的广播,以向其它进程共享所述变量信息。

8.一种虚拟环境交互实现中变量的处理装置,其特征在于,包括:

处理器;以及

存储器,所述存储器上存储有计算机可读指令,所述计算机可读指令被所述处理器执行时实现根据权利要求1至4中任一项所述的虚拟环境交互实现中变量的处理方法。

9.一种计算机可读存储介质,其上存储有计算机程序,所述计算机程序被处理器执行时实现根据权利要求1至4中任一项所述的虚拟环境交互实现中变量的处理方法。

虚拟环境交互实现中变量的处理方法、装置和存储介质

技术领域

[0001] 本发明涉及计算机应用技术邻域，特别涉及一种虚拟环境交互实现中变量的处理方法、装置和计算机可读存储介质。

背景技术

[0002] 随着计算机应用的不断发展，在所实现的应用程序中，除了为用户提供各种功能的实现之外，还能够为用户实现虚拟环境，以供用户进行虚拟环境中角色的操控。例如，此虚拟环境，可以是用户进入的对局游戏，所操控的角色即为对局游戏中的虚拟角色。

[0003] 虚拟环境的交互，是各种元素之间交互的实现，即一元素被触发交互之时，其它元素能够对此进行感知。而所指的元素，一方面是被渲染显示于虚拟环境中的角色，另一方面，也一并包含着被虚拟环境中角色关联的物理硬件。

[0004] 随着越来越多能够受控于程序控制的物理硬件，例如，传感器等各种电子元器件的存在，使得虚拟环境的操控不再限于单纯的角色之间，能够由角色而延伸至其所关联的物理硬件。

[0005] 这将使得虚拟环境被操控进行的交互，包括角色之间、角色与物理硬件之间以及由于角色与物理硬件之间交互的进行而引发与其它角色之间的交互，形式多种多样。

[0006] 但是，由于物理硬件中程序控制和虚拟环境的实现，往往是分别通过二者相互独立的开发编程进行的，这将使得角色与物理硬件之间所能够进行的交互，特别是物理硬件所能够执行的动作都是预置的，无法自由实现，从而也进一步限制了虚拟环境交互的实现。

[0007] 这都是由于物理硬件所独立进行的开发编程，以及由此而存在的通信问题造成的限制，虚拟环境并无法完全实现所存在元素的自定义配置，以及元素之间的自由交互。

发明内容

[0008] 为了解决相关技术中虚拟环境并无法完全实现所存在元素的自定义配置以及元素之间的自由交互的技术问题，本发明提供一种虚拟环境交互实现中变量的处理方法、装置和计算机可读存储介质。

[0009] 一种虚拟环境交互实现中变量的处理方法，所述方法包括：

[0010] 触发进行虚拟环境所部署元素的变量配置，获得对所述元素配置的变量信息；

[0011] 定位所述元素归属的进程，将所述变量信息写入所述进程映射的内存空间；

[0012] 将写入内存空间的所述变量信息共享至其它进程；

[0013] 通过共享的所述变量信息，使所述虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互。

[0014] 一种虚拟环境交互实现中变量的处理装置，包括：

[0015] 变量配置模块，用于触发进行虚拟环境所部署元素的变量配置，获得对所述元素配置的变量信息；

[0016] 写入模块，用于定位所述元素归属的进程，将所述变量信息写入所述进程映射的

内存空间；

[0017] 共享模块，用于将写入内存空间的所述变量信息共享至其它进程；

[0018] 交互感知模块，用于通过共享的所述变量信息，使所述虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互。

[0019] 一种虚拟环境交互实现中变量的处理装置，包括：

[0020] 处理器；以及

[0021] 存储器，所述存储器上存储有计算机可读指令，所述计算机可读指令被所述处理器执行时实现如前所述的虚拟环境交互实现中变量的处理方法。

[0022] 一种计算机可读存储介质，其上存储有计算机程序，所述计算机程序被处理器执行时实现如前所述的虚拟环境交互实现中变量的处理方法。

[0023] 本发明的实施例提供的技术方案可以包括以下有益效果：

[0024] 进行虚拟环境中元素的部署中，对所部署的元素，将触发进行变量配置获得变量信息，定位此元素归属的进程，将变量信息写入进程映射的内存空间，再次写入内存空间的变量信息共享至其它进程，至此便通过共享的变量信息，使虚拟环境中部署的其它元素会通过共享变量信息而感知所对应元素被触发的相关交互，针对于虚拟环境部署的元素，引入了变量信息的共享机制，使得元素之间能够相互感知，进而在进行虚拟环境中元素的自定义配置所涉及的变量配置所引入变量信息共享机制，也得以随之实现所自定义配置的变量在其它元素的共享，由此，使得所自定义配置的元素能够与其它变量相互感知，并触发相应交互，完成进行自定义的元素不再预置所执行的动作，能够进行元素之间的自由交互。

[0025] 应当理解的是，以上的一般描述和后文的细节描述仅是示例性的，并不能限制本发明。

附图说明

[0026] 此处的附图被并入说明书中并构成本说明书的一部分，示出了符合本发明的实施例，并于说明书一起用于解释本发明的原理。

[0027] 图1根据一示例性实施例示出的本发明所涉及实施环境的示意图；

[0028] 图2是根据一示例性实施例示出的一种装置的框图；

[0029] 图3是根据一示例性实施例示出的一种虚拟环境交互实现中变量的处理方法的流程图；

[0030] 图4是根据图3对应实施例对步骤310的细节进行描述的流程图；

[0031] 图5是根据图4对应实施例示出的对步骤311的细节进行描述的流程图；

[0032] 图6是根据另一示例性实施例示出的一种虚拟环境交互实现中变量的处理方法的流程图；

[0033] 图7是根据一示例性实施例示出的实现虚拟环境的应用程序所部署变量之间的关系示意图；

[0034] 图8是根据一示例性实施例示出的应用程序所构建虚拟环境中部署的角色配置向物理硬件执行的控制动作的界面显示图；

[0035] 图9是根据一示例性实施例示出的进程A和进程B二者之间实现通信全局变量共享的示意图；

[0036] 图10是根据一示例性实施例示出的一种虚拟环境交互实现中变量的处理装置的框图；

[0037] 图11是根据图10对应实施例示出的对变量配置模块的细节进行描述的框图；

[0038] 图12是根据另一示例性实施例示出的一种虚拟环境交互实现中变量的处理装置的框图。

具体实施方式

[0039] 这里将详细地对示例性实施例执行说明，其示例表示在附图中。下面的描述涉及附图时，除非另有表示，不同附图中的相同数字表示相同或相似的要素。以下示例性实施例中所描述的实施方式并不代表与本发明相一致的所有实施方式。相反，它们仅是与如所附权利要求书中所详述的、本发明的一些方面相一致的装置和方法的例子。

[0040] 图1是根据一示例性实施例示出的本发明所涉及实施环境的示意图。本发明所涉及的实施环境，包括控制端110以及物理硬件130。

[0041] 控制端110用于实现虚拟环境以及虚拟环境所相关的控制逻辑。控制端110可以是电脑终端、智能手机、平板电脑等终端设备中的至少一种。控制端110所实现的虚拟环境中，部署的元素可以包括渲染显示于虚拟环境中的角色，以及此角色关联的物理硬件130。

[0042] 换而言之，物理硬件130也是作为虚拟环境中的元素存在的。

[0043] 物理硬件130是脱离于控制端110存在于物理世界中的实体硬件，例如，可以是传感器等各种电子元器件，也可以是装配了各种传感器并由一定形状壳体封装的小车，任一配备了通信连接接口(如，WiFi通信连接接口)的硬件设备。

[0044] 物理硬件130作为物理世界中真实存在的实体硬件，通过所关联的角色接入到控制端110所构建的虚拟环境中，一方面，能够通过虚拟环境中角色被触发的交互，例如对角色的操控，此时物理硬件130感知并响应；另一方面，也能够对物理硬件触发交互，至此而使得虚拟环境中的角色感知并响应。

[0045] 也就是说，在此实施环境中，能够实现角色与角色之间的双向交互，以及角色与物理硬件之间的双向交互，甚至于由于与物理硬件之间的交互而使得角色被触发交互，进而使得虚拟环境，或者其它角色感知并响应。

[0046] 在此应当说明的是，控制端110是唯一存在的，而物理硬件130则可以是一个或者两个以上的，并且可以是单一种类或者多种种类的。

[0047] 图2是根据一示例性实施例示出的一种装置的框图。例如，装置200可以是上述实施环境中的智能手机。

[0048] 参照图2，装置200可以包括以下一个或多个组件：处理组件202，存储器204，电源组件206，多媒体组件208，音频组件210，传感器组件214以及通信组件216。

[0049] 处理组件202通常控制装置200的整体操作，诸如与显示，电话呼叫，数据通信，相机操作以及记录操作相关联的操作等。处理组件202可以包括一个或多个处理器218来执行指令，以完成下述的方法的全部或部分步骤。此外，处理组件202可以包括一个或多个模块，便于处理组件202和其他组件之间的交互。例如，处理组件202可以包括多媒体模块，以方便多媒体组件208和处理组件202之间的交互。

[0050] 存储器204被配置为存储各种类型的数据以支持在装置200的操作。这些数据的示

例包括用于在装置200上操作的任何应用程序或方法的指令。存储器204可以由任何类型的易失性或非易失性存储设备或者它们的组合实现，如静态随机存取存储器(Static Random Access Memory，简称SRAM)，电可擦除可编程只读存储器(Electrically Erasable Programmable Read-Only Memory，简称EEPROM)，可擦除可编程只读存储器(Erasable Programmable Read OnlyMemory，简称EPROM)，可编程只读存储器(Programmable Red-Only Memory，简称PROM)，只读存储器(Read-Only Memory，简称ROM)，磁存储器，快闪存储器，磁盘或光盘。存储器204中还存储有一个或多个模块，该一个或多个模块被配置成由该一个或多个处理器218执行，以完成下述图3、图4、图5和图6任一所示方法中的全部或者部分步骤。

[0051] 电源组件206为装置200的各种组件提供电力。电源组件206可以包括电源管理系统，一个或多个电源，及其他与为装置200生成、管理和分配电力相关联的组件。

[0052] 多媒体组件208包括在所述装置200和用户之间的提供一个输出接口的屏幕。在一些实施例中，屏幕可以包括液晶显示器(Liquid Crystal Display，简称LCD)和触摸面板。如果屏幕包括触摸面板，屏幕可以被实现为触摸屏，以接收来自用户的输入信号。触摸面板包括一个或多个触摸传感器以感测触摸、滑动和触摸面板上的手势。所述触摸传感器可以不仅感测触摸或滑动动作的边界，而且还检测与所述触摸或滑动操作相关的持续时间和压力。屏幕还可以包括有机电致发光显示器(Organic Light Emitting Display，简称OLED)。

[0053] 音频组件210被配置为输出和/或输入音频信号。例如，音频组件210包括一个麦克风(Microphone，简称MIC)，当装置200处于操作模式，如呼叫模式、记录模式和语音识别模式时，麦克风被配置为接收外部音频信号。所接收的音频信号可以被进一步存储在存储器204或经由通信组件216发送。在一些实施例中，音频组件210还包括一个扬声器，用于输出音频信号。

[0054] 传感器组件214包括一个或多个传感器，用于为装置200提供各个方面的状态评估。例如，传感器组件214可以检测到装置200的打开/关闭状态，组件的相对定位，传感器组件214还可以检测装置200或装置200一个组件的位置改变以及装置200的温度变化。在一些实施例中，该传感器组件214还可以包括磁传感器，压力传感器或温度传感器。

[0055] 通信组件216被配置为便于装置200和其他设备之间有线或无线方式的通信。装置200可以接入基于通信标准的无线网络，如WiFi(WIreless-Fidelity，无线保真)。在一个示例性实施例中，通信组件216经由广播信道接收来自外部广播管理系统的广播信号或广播相关信息。在一个示例性实施例中，所述通信组件216还包括近场通信(Near Field Communication，简称NFC)模块，以促进短程通信。例如，在NFC模块可基于射频识别(Radio Frequency Identification，简称RFID)技术，红外数据协会(Infrared Data Association，简称IrDA)技术，超宽带(Ultra Wideband，简称UWB)技术，蓝牙技术和其他技术来实现。

[0056] 在示例性实施例中，装置200可以被一个或多个应用专用集成电路(Application Specific Integrated Circuit，简称ASIC)、数字信号处理器、数字信号处理设备、可编程逻辑器件、现场可编程门阵列、控制器、微控制器、微处理器或其他电子元件实现，用于执行下述方法。

[0057] 图3是根据一示例性实施例示出的一种虚拟环境交互实现中变量的处理方法的流

程图。该虚拟环境交互实现中变量的处理方法,适用于图1所示实施环境所指的控制端110,该控制端110在一个示例性实施例中是图2所示的装置。如图3所示,该虚拟环境交互实现中变量的处理方法,至少包括以下步骤。

[0058] 在步骤310中,触发进行虚拟环境所部署元素的变量配置,获得对元素配置的变量信息。

[0059] 其中,应当明确说明的是,所指的元素是虚拟环境交互实现中能够触发交互的对象,由于虚拟环境交互实现中能够触发交互的对象除了数量并不单一之外,其种类也有多种,因此,所指的元素可以是一个或者两个以上的,并且可以是单一种类或者多种种类的。

[0060] 在一个示例性实施例的具体实现中,所部署的元素至少包括渲染显示在虚拟环境中的角色,以及与此角色关联的物理硬件,当然,除此之外,还可以包括其它角色,其它角色可以是关联于其它物理硬件的角色,也可以是单独存在的角色,在此不进行限定。

[0061] 进一步的,虚拟环境所部署的元素,是指被加载至虚拟环境交互实现的元素,例如,被加载至虚拟环境的角色,又例如,通过所关联角色而被接入至虚拟环境交互中的物理硬件。此元素由于刚完成自身在虚拟环境的部署,而尚未进行相应控制逻辑的配置。

[0062] 针对于虚拟环境所部署的元素,将进行变量配置,由此而获得的变量信息包括变量名称以及赋值,此赋值可以是默认值,也可以是其它数值。

[0063] 变量信息是为元素而配置的控制逻辑中存在的变量,因此,所进行的变量配置过程,存在于控制逻辑的配置过程中。

[0064] 所进行虚拟环境交互的元素部署以及为元素的程序控制而进行的控制逻辑部署都由用户实现,即用户通过图形化编程而实现。在一个示例性实施例中,实现虚拟环境的应用程序中,存在着工具栏,工具栏存在着当前所部署元素可供配置的所有动作块,这些动作块可分类显示于工具栏中,以便用户选取。

[0065] 一动作块,有其所唯一映射的代码信息,而随着用户为所部署元素选取并拼接在一起的动作块的不断增加,在获得动作块各自所分别映射的代码信息之外,根据动作块之间的拼接关系,还获得了代码信息之间的逻辑状态,至此即可获得所部署元素对应的控制逻辑。

[0066] 控制逻辑除了定义所需要执行的动作之外,还对所涉及的变量进行了声明并赋值,因此,随之而获得变量信息。

[0067] 在步骤330中,定位元素归属的进程,将变量信息写入进程映射的内存空间。

[0068] 其中,如前所述的,元素可以是虚拟环境中渲染显示的角色,也可以是物理硬件。无论是角色还是物理硬件,在虚拟环境交互的实现中,都涉及了一定动作的执行,当然所执行的动作必然是所配置控制逻辑定义的动作。

[0069] 动作的执行是由元素所归属的进程控制的,因此,配置而获得的变量信息必然是写入到此进程所映射的内存空间,以便于在执行控制逻辑时读取。

[0070] 也就是说,针对于当前所部署的元素,获得所配置的变量信息之后,所归属的进程便在自身拥有的内存空间执行写操作,以将变量信息写入。

[0071] 应当进一步说明的,对于角色而言所归属的进程即为虚拟环境主进程,物理硬件所归属的进程即为物理硬件中创建的进程,即硬件进程。

[0072] 应当理解的,对于每一进程,都其所映射的内存空间,进程所执行程度控制中涉及

的变量以及变量的赋值等，即变量信息，都被存放于进程所映射的内存空间。

[0073] 变量的配置，包括变量的声明、变量的修改，当然对于变量的修改而言，一方面可以是对原有变量的修改，另一方面也可以是新增变量的实现。

[0074] 随着虚拟环境中角色的加载进入，以及角色所关联物理硬件的接入，对于角色和物理硬件而言，均未存在对应的变量信息，需要对此进行，所配置的变量信息将被创建至所属进程映射的内存空间。

[0075] 在一个示例性实施例中，对于变量信息在内存空间中的存储地址，可以通过哈希分派的方式分配，在完成了存储地址的分配之后，再将变量信息对此存储地址执行写操作，完成变量信息在内存空间中的写入。

[0076] 进一步的，对变量信息所携带的变量名称进行哈希分派，以根据变量名称索引到内存空间中对应的存储地址，在找到对应的存储地址之后即可执行写操作，将变量信息写入。

[0077] 在步骤350中，将写入内存空间的变量信息共享至其它进程。

[0078] 其中，相互作用共同实现虚拟环境交互的所有进程中，其它进程是除了当前所部署元素归属进程的其它进程。在进程之间进行各自所拥有变量信息的同步，以使得变量信息都能够被进程所共享。

[0079] 共享，即使得变量信息在各进程所映射内存空间同步存储，并且后续也将同步更新。一变量信息被配置之后，便随着写操作以及同步存储的执行共享至其它进程，并且在此变量信息发生变化之后执行同步更新，使得其它进程能够感知所发生的变化。

[0080] 在步骤370中，通过共享的变量信息，使虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互。

[0081] 其中，通过前述步骤的执行，为虚拟环境交互的实现引入了变量信息的共享机制，而在虚拟环境交互的实现中，随着元素被触发交互，此元素所对应的某些变量信息将由此而发生变化，至此，由于变量信息共享机制的引入，而使得其它进程所共享的此变量信息随之发生同步更新。

[0082] 对于其它进程，将通过自身所映射内存空间中变量信息同步更新的发生，感知此变量信息所对应元素被触发了交互，此时，将根据配置的控制逻辑对感知到的交互进行响应。

[0083] 在一个示例性实施例的具体实现中，被触发交互的元素是存在于虚拟环境的角色，并且这一角色有着关联的物理硬件。物理硬件归属于一进程，即硬件进程，硬件进程所映射的内存空间中，所关联角色共享的变量信息发生同步更新，通过同步更新的进行得以感知所关联角色被触发的交互，此时，即可通过相应控制逻辑的执行控制物理硬件进行响应。

[0084] 在另一个示例性实施例的具体实现中，被触发交互的元素是一物理硬件，此物理硬件与虚拟环境中的一角色相关联。随着物理硬件被触发的交互，例如自身所装设传感器感知到外部变化，又例如，某一控制按钮被触发等，都将使得自身所配置的相应变量信息发生变化。此时，对于所关联的角色而言，归属进程映射的内存空间中，共享自所关联角色的变量信息将发生同步更新，角色随着同步更新的进行感知到物理硬件被触发的交互，进而通过自身控制逻辑的执行进行响应。

[0085] 至此，便为虚拟环境交互所部署元素实现了自定义控制，并且在变量信息共享机制的作用下，能够自由实现角色与物理硬件之间的交互，进而完全实现元素的自定义配置，同样作为元素存在的物理硬件也能够进行自定义配置。

[0086] 由此，可以看到，作为元素的物理硬件以及角色，同处于一个世界，使得物理硬件和角色之间的交互都在用户的配置下自然实现，打通了角色与物理硬件之间存在的通信困难等诸多不便，进而物理硬件和角色之间能够天然通信，低门槛的自定义实现多种元素，例如角色和物理硬件之间的交互。

[0087] 对于角色以及物理硬件控制的编程而言，由于不再需要考虑角色和物理硬件之间连通的障碍，所以能够以直觉的方式完成控制逻辑的编程，尽可叹有的降低了编程的门槛，非专业开发的用户甚至于适龄儿童都能够完成。

[0088] 至此，便得以实现了全局变量，由于其是跨进程实现的，因此也是通信全局变量，从变量作用域来看，实现了跨进程和跨物理实体，将能够最大限度的降低实现连接、状态维护和异步调度等的复杂度，也使得用户的逻辑在不同物理实体和应用程序上透明化，对于用户而言，角色和物理硬件的交互是通过同一环境下的控制逻辑实现的。

[0089] 图4是根据图3对应实施例对步骤310的细节进行描述的流程图。该步骤310，如图4所示，至少包括以下步骤。

[0090] 在步骤311中，元素对应可供配置的动作块中触发进行指定动作块的拖动，获取指定动作块形成的控制逻辑。

[0091] 其中，对于当前所部署的一元素而言，将随着可供配置动作块中拖动操作的触发获得指定动作块，指定动作块的数量为一个或者两个以上。在指定动作块为一个时，所映射的代码信息便形成控制逻辑；在指定动作块为两个以上时，应当理解，两个以上的指定动作块是拼接在一起的，由此便可由指定动作块所映射的代码信息以及拼接关系形成控制逻辑。

[0092] 应当理解，所形成的控制逻辑，用于指示当前所部署元素执行的动作。换而言之，后续将通过所形成控制逻辑的执行而控制当前所部署的元素执行指定动作，进而实现交互。

[0093] 在步骤313中，由控制逻辑提取得到对元素配置的变量信息。

[0094] 其中，应当理解，控制逻辑是由代码信息集合在一起形成的，将不可避免的进行着变量的声明，以创建变量并赋值，也会进行着相应的读取操作以及修改操作，进而为控制逻辑的执行提供必备参数，所以能够由控制逻辑提取得到用户对当前所部署元素配置的变量信息。

[0095] 图5是根据图4对应实施例示出的对步骤311的细节进行描述的流程图。元素包括渲染显示于虚拟环境的角色以及角色关联的物理硬件，该步骤311，如图5所示，至少包括以下步骤。

[0096] 在步骤3111中，角色对应可供配置的动作块中，根据角色和物理硬件之间交互进行角色所执行动作和物理硬件所执行动作的动作块配置，确定角色和物理硬件分别对应的指定动作块。

[0097] 其中，应当进一步说明的是，角色和物理硬件之间的关联性，包括虚拟环境中角色由于用户操作等外部变化的施加而被触发交互，产生自身动作变化，进而带动物理硬件响

应而执行一定的动作;也包括物理硬件被触发交互,而执行自身动作,并使得角色受到影响而执行一定的动作。

[0098] 例如,物理硬件中装配的传感器感知到外部变化,例如,温度、光线等的变化,便将相关的信息传送至自身具备的主控模块,由主控模块向虚拟环境中关联的角色反馈,进而使关联的角色响应所感知的外部变化。

[0099] 又例如,物理硬件是马达以及各种零部件的组合,在对虚拟环境中关联角色施加用户操作之后,即可在角色对物理硬件的控制动作下使得物理硬件在自身马达的驱动下执行各种机械运动。

[0100] 物理硬件基于自身与角色的关联性接入虚拟环境,至此便可通过物理硬件和虚拟环境二者之间的通信,具体而言,即为硬件进程与虚拟环境主进程之间的通信,实现物理硬件与虚拟环境中角色的相互控制。

[0101] 在步骤3113中,由指定动作块和代码信息之间的映射关系以及指定动作块之间的拼接关系,分别获得角色和物理硬件对应的控制逻辑。

[0102] 其中,动作块实质是角色所相关程序控制的一条或者几条代码信息所对应的图形存在形式,因此,由指定动作块与代码信息之间的映射关系,以及指定动作块之间的拼接关系,获得角色所对应的控制逻辑。

[0103] 角色与物理硬件二者之间存在着关联关系,亦即二者是相互作用的,故角色所对应控制逻辑的执行所实现的动作,还包括了对物理硬件的控制动作,因此,基于角色所对应的控制逻辑即可获得物理硬件对应的控制逻辑。

[0104] 在一个示例性实施例的具体实现中,对于物理硬件所关联角色在虚拟环境的部署,相对于并未关联物理硬件的虚拟角色而言,是通过虚拟环境中扩展加载的执行实现的。

[0105] 物理硬件是可进行程序控制的实体硬件,例如,传感器、显示屏、马达等各种零部件的组合,物理硬件能够通过程序控制实现动作执行。虚拟环境是由运行于终端设备的程序构建的,用户将通过对虚拟环境中各种角色的操控而使得角色被触发交互。物理硬件存在于虚拟环境外部,实质上是以角色的形式存在于虚拟环境中的。

[0106] 物理硬件所关联角色在虚拟环境中的扩展加载,是通过预先进行的角色定义实现的。具体而言,将定义角色在虚拟环境中的显示样式以及所能够执行的动作,以获得配置文件,此配置文件包括了角色载入虚拟环境所需要的资源文件以及可供配置的动作块相关信息,资源文件将用于定义角色在虚拟环境中的显示样式。可供配置的动作块相关信息将用于指示所定义角色所能够在工具栏中布设的动作块。

[0107] 具体的,对应于角色的配置文件,用于定义并描述角色的显示样式以及所能够执行的动作。例如,关联于物理硬件的角色,其配置文件除了定义并描述角色在虚拟环境中的显示样式,和在虚拟环境执行各种可实现的动作之外,还定义了对物理硬件的控制动作,以及对物理硬件被施加的外部变化而执行的响应动作。

[0108] 关联于物理硬件的角色,通过被定义生成的配置文件,即可加载至虚拟环境中,在此虚拟环境中渲染显示,获得角色在虚拟环境中的图形显示。

[0109] 此时,对于关联于角色的物理硬件而言,将被控制创建自身归属的进程,即硬件进程,并将创建的硬件进程与角色归属的虚拟环境主进程建立连接。

[0110] 随着动作块被配置于角色上,将随之为此动作块创建相应的线程,该线程归属于

虚拟环境主进程，进而通过所创建的线程控制角色实现相应控制逻辑的执行。

[0111] 应当说明的是，虚拟环境构建显示之初便为此而创建了虚拟环境主进程，通过虚拟环境主进程而实现虚拟环境所相关程序控制的执行。在一个示例性实施例的，在虚拟环境所相关程序控制的执行中，对于所产生的任务，例如，一角色所触发动作的控制，可通过创建线程的形式实现，即在虚拟环境主进程下采用多线程的模式。

[0112] 随着物理硬件所关联角色在虚拟环境中的加载，将在虚拟环境主进程下为此角色所触发动作创建相应的线程；而对于所连接的物理硬件，将控制其创建进程，即硬件进程，并建立硬件进程和虚拟环境主进程之间的连接，此时，便使得物理硬件是接入虚拟环境的。

[0113] 物理硬件在虚拟环境中的接入，使得物理硬件和虚拟环境中关联角色的通信，是通过硬件进程和虚拟环境主进程进行的，在此基础上再经由相应的线程直连至关联角色，通过多线程模式保证虚拟环境中各种角色所触发动作被并发执行的同时，也优化了物理硬件和所关联角色之间的联动性。

[0114] 例如，角色被触发交互而执行物理硬件的控制动作，由于此控制动作是角色所预先定义的，故存在着相应的线程。此时，只需在相应线程的作用下生成或者修改此控制动作对应的变量信息，线程将变量信息传送至虚拟环境主进程，进而通过虚拟环境主进程与硬件进程之间的连接传送至物理硬件，使得动态变化的变量信息被同步至物理硬件，在同步的变量信息作用下驱使物理硬件执行指定动作，至此，对于与物理硬件关联的角色而言，便完成了其控制动作的执行。

[0115] 通过此实现过程，便使得角色和物理硬件之间的交互能够直接、顺畅的进行，在为用户所拥有物理硬件自由实现所关联角色在虚拟环境中载入的同时，也为角色和物理硬件之间的相互控制，相互联动奠定了基础。

[0116] 由此，对于角色的动作执行以及物理硬件的动作执行而言，都得以通过动作块的配置实现相应的控制逻辑，在为用户实现图形化编程的同时，也使得用户能够在所进行的图形化编程过程中根据自己的期望直观的完成编程，实现的难度非常低，并且能够为用户提供即时的控制效果，即一旦完成了动作块的拼接，即可直接进行角色的控制，甚至于物理硬件的控制。

[0117] 在一个示例性实施例中，图3所对应实施例中的步骤350包括：根据被写入内存空间的变量信息，元素归属的进程发起广播，所广播的数据包携带变量信息。

[0118] 其中，在完成了变量信息的写入之后，即可对所写入的变量信息执行同步。所写入变量信息的同步，通过进程之间广播的进行实现。也就是说，在完成变量信息的写入之后，即可进行进程之间的广播，以使得其它进程均能够同步至变量信息，进而实现所有进程之间变量信息的共享，方便了进程之间通信和控制的实现效率。

[0119] 图6是根据另一示例性实施例示出的一种虚拟环境交互实现中变量的处理方法的流程图。该虚拟环境交互实现中变量的处理方法，在步骤350之后，如图6所示，还包括以下步骤。

[0120] 在步骤410中，进程所映射内存空间中触发进行变量信息的修改操作，通过修改操作的执行在内存空间创建变量信息，或者修改变量信息中的数值。

[0121] 其中，在此所指的修改操作，是作用于内存空间中进程所控制元素对应的变量信息的，而并未作用于进程共享其它进程的变量信息。此修改操作是由于进程所控制元素被

触发交互所引发的。修改操作包括了变量信息中赋值的修改、变量的新增和删除，修改操作是使得原本所存在变量信息发生变化的操作。

[0122] 应当理解，被触发交互的元素，所对应存储的变量信息会被触发修改操作，将自身所存储变量信息修改为对应于所进行交互的新变量信息，这是元素自身对触发交互的响应。在一个示例性实施例中，随着自身所对应变量信息的变化，被触发交互的元素将执行的动作，在此元素为关联于物理硬件的角色时，所执行的动作还将包括对物理硬件的控制动作。

[0123] 例如，对虚拟环境中的角色触发触碰操作，此时，其所对应属性状态信息被相应触发修改操作，使得角色所对应变量信息发生变化，随着此变化的发生，也将使得角色在虚拟环境中进行响应触碰操作的动态显示，并且变化的变量信息也将被同步至物理硬件。

[0124] 在一个示例性实施例中，对于修改操作的触发，也可根据所作用的变量名称定位存储地址，此存储地址即为触发修改操作的变量信息被写入的地址，将在此存储地址上对写入的变量信息执行修改操作。

[0125] 在步骤430中，根据执行修改操作而获得的变量信息进行进程之间的广播，以向其它进程共享变量信息。

[0126] 其中，随着变量信息被执行修改操作而获得新的变量信息，便发起进程之间的广播，使得其它进程获得新的变量信息，进而得以感知此变量信息所对应元素被触发的交互，可以根据所感知的交互而执行指定的动作，以响应所感知的交互。

[0127] 通过此示例性实施例，由于进程间变量信息同步机制的引入，使得所有元素能够处于相同运行环境，也就是说，对于所存在二者相互关联的角色和物理硬件而言，可以一并实现物理硬件和角色中程序控制的编程，不再需要分别基于角色和物理硬件编程，进而必然也不再需要解决二者之间通信的实现，在用户层面，将使得用户能够随意实现角色和物理硬件之间的交互。

[0128] 通过如上所述的示例性实施例，将使得用户，甚至于低龄用户，例如儿童，也能够通过编程实现自身与虚拟环境中角色的互运算，以及与物理硬件的互动，所期望角色和物理硬件实现的动作，都可以通过自己所进行的快速配置实现，例如，在物理硬件被开启时，虚拟环境中关联的角色输出指定的语音，在物理硬件被关闭时，虚拟环境中关联的角色则进入睡眠状态。

[0129] 又例如，在虚拟环境的状态发生改变时，例如，用户操控物理硬件关联的角色在对局游戏中成功闯关，则物理硬件会闪灯并播放音乐。

[0130] 再例如，物理硬件感知到外部温度降低，则其在虚拟环境中关联的角色将控制实现下雪等事件。

[0131] 物理硬件也可接受语音，使得其在虚拟环境中关联的角色将语音转换成文字。

[0132] 以上，都可通过用户对虚拟环境中角色和物理硬件的自由配置快速实现，不需要花费大多的学习成本。

[0133] 下面以一物理硬件为例，结合一具体场景来描述上述虚拟环境交互实现中变量的处理过程。此具体场景中，以物理硬件必须传输至所关联角色，即需由物理硬件这一物理实体而传输至虚拟环境所在终端设备的变量处理为例，此变量为通信全局变量，将在虚拟环境交互实现中通过本发明所示实施例实现通信全局变量的处理。

[0134] 首先应当说明的是，实现虚拟环境的应用程序，将存在着三种变量，即局部变量、全局变量和通信全局变量。图7是根据一示例性实施例示出的实现虚拟环境的应用程序所部署变量之间的关系示意图。

[0135] 如图7所示的，实现虚拟环境的应用程序，其运行中存在着的三种变量，拥有着各不相同的变量作用域。局部变量是在函数体内作用的变量，全局变量是在整个程序进程内能够使用的变量，而通信全局变量则是能够跨进程跨物理设备的变量，将用于实现虚拟环境中角色与物理硬件二者之间的相互控制。

[0136] 由此可以看到，应用程序中存在的通信全局变量是本发明前述示例性实施例所指变量的具体实例。

[0137] 图8是根据一示例性实施例示出的应用程序所构建虚拟环境中部署的角色配置向物理硬件执行的控制动作的界面显示图。应用程序所构建的虚拟环境中，存在着物理硬件所关联的角色，经由此角色而实现对物理硬件的控制。

[0138] 具体的，结合参阅图8，角色通过动作块的拼接，已经为物理硬件配置了当按下轻触开关A时，对运行时间进行计时的控制逻辑，即动作块拼接区域510所示的。

[0139] 而在此控制逻辑的实现中，必然要进行角色和物理硬件之间的通信。因此，控制逻辑所配置的变量为前述所指的通信全局变量。

[0140] 可视化编程的创建区域510中，为此而进行通信全局变量的创建，此通信全局变量至少包括运行时间。具体的，指定动作块被拖动至可视化编程的创建区域510，并根据所映射的代码信息而相互拼接在一起，即可完成运行时间这一通信全局变量的创建。

[0141] 至此，对于角色而言，将通过所归属的虚拟环境主进程，在此称之为进程A，传送至物理硬件所对应的硬件进程，在此称之为进程B。

[0142] 各进程之间独立进行所数据的缓存，因此，分别映射了不同的内存空间，所需要缓存的数据将通过哈希分派的方式配置至内存空间中的存储地址。

[0143] 因此，各进程都创建了hash(哈希)表，所创建的通信全局变量都将其变量名称所对应的hash存储至hash表中，进而索引至所对应的存储地址，通信全局变量便在这一存储地址上执行写操作。

[0144] 在此，将涉及变量声明、读取变量以及修改变量三大阶段。首先，对于变量声明，是在hash表所索引存储地址上建立变量，并赋予默认值的过程；其次，对于读取变量，将从hash表根据变量名索引到存储地址，以执行读取的过程；最后，对于修改变量，将包括建立变量并赋值、修改赋值两大过程，具体而言，同步存在于各进程的变量，也将进行同步修改，在hash表中查找变量，如果没有，则建立并赋值，如果存在，则修改赋值。

[0145] 图9是根据一示例性实施例示出的进程A和进程B二者之间实现通信全局变量共享的示意图。当然，应当进一步强调的，对于应用程序所实现的虚拟环境交互而言，由于能够接入各种物理硬件，因此，所存在的进程并不仅限于进程A和进程B，在此仅仅是以进程A和进程B为例进行说明。

[0146] 请结合参阅图9，内存空间以cache表的形式存在。无论是进程A还是进程B，在其运行之初都将进行cache表的初始化。

[0147] 完成初始化之后，各种进行通信全局变量的创建，分别声明变量，将变量放入cache表中，并赋予默认值，将此同步至另一进程。

[0148] 后续即可进行变量的读取和修改过程，具体如后续时序所示的。在需要进行变量读取时，可由cache表中取得所需要的变量，在需要修改变量时，将进行两个进程之间的同步。

[0149] 至此，便使得角色和物理硬件的交互不再存在通信的困难点，特别是对于新手而言，通过通信全局变量的作用，使得用户很容易就能够实现不同进程，不同物理实体之间的通信和响应，以最为直观的方式降低了编程门槛。

[0150] 通过这一实现过程，能够方便地在多个环境中，网络与物理实体间建立通信，降低理解和实现的成本，为用户所提供的图形化编程界面中，用户所配置的控制逻辑能够同时执行在软件和硬件上，对于用户所希望对多个物理实体和角色进行的交互，通常的编程概念中，需要实现连接，状态维护、异步调用等复杂的过程，增加了用户的学习理解成本，而使用跨物理域全局变量的方法，使得用户的逻辑在不同物理硬件和软件上透明化。

[0151] 由于通信全局变量不但在不同程序控制中保持一致，在跨物理实体上仍然保持一致，因此能够自动维护连接同步过程，简化了编程中的代码实现，也增强了角色和物理硬件之间的联动性，对于用户而言，甚至能够简单快速的通过虚拟环境实现物理硬件的状态面板、遥控端和控制台等功能。

[0152] 下述为本发明装置实施例，可以用于执行本发明上述虚拟环境交互实现中变量的处理方法实施例。对于本发明装置实施例中未披露的细节，请参照本发明虚拟环境交互实现中变量的处理方法实施例。

[0153] 图10是根据一示例性实施例示出的一种虚拟环境交互实现中变量的处理装置的框图。该虚拟环境交互实现中变量的处理装置，如图10所示，至少包括：变量配置模块610、写入模块630、共享模块650和交互感知模块670。

[0154] 变量配置模块610，用于触发进行虚拟环境所部署元素的变量配置，获得对元素配置的变量信息。

[0155] 写入模块630，用于定位元素归属的进程，将变量信息写入进程映射的内存空间。

[0156] 共享模块650，用于将写入内存空间的变量信息共享至其它进程。

[0157] 交互感知模块670，用于通过共享的变量信息，使虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互。

[0158] 图11是根据图10对应实施例示出的对变量配置模块的细节进行描述的框图。该变量配置模块610，如图11所示，至少包括：动作块配置单元611和变量提取单元613。

[0159] 动作块配置单元611，用于元素对应可供配置的动作块中触发进行指定动作块的拖动，获取指定动作块形成的控制逻辑。

[0160] 变量提取单元613，用于由控制逻辑提取得到对元素配置的变量信息。

[0161] 图12是根据另一示例性实施例示出的一种虚拟环境交互实现中变量的处理装置的框图。该虚拟环境交互实现中变量的处理装置，如图12所示，还包括：修改操作执行模块710和修改广播模块730。

[0162] 修改操作执行模块710，用于进程所映射内存空间中触发进行变量信息的修改操作，通过修改操作的执行在内存空间创建变量信息，或者修改变量信息中的数值。

[0163] 修改广播模块730，用于根据执行修改操作而获得的变量信息进行进程之间的广播，以向其它进程共享所述变量信息。

[0164] 可选的，本发明还提供一种虚拟环境交互实现中变量的处理装置，该虚拟环境交互实现中变量的处理装置可以前述所示实施环境中，执行图3、图4、图5和图6任一所示的虚拟环境交互实现中变量的处理方法的全部或者部分步骤。所述装置包括：

[0165] 处理器；

[0166] 用于存储处理器可执行指令的存储器；

[0167] 其中，所述处理器被配置为执行：

[0168] 触发进行虚拟环境所部署元素的变量配置，获得对所述元素配置的变量信息；

[0169] 定位所述元素归属的进程，将所述变量信息写入所述进程映射的内存空间；

[0170] 将写入内存空间的所述变量信息共享至其它进程；

[0171] 通过共享的所述变量信息，使所述虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互。

[0172] 该实施例中的装置的处理器执行操作的具体方式已经在有关该虚拟环境交互实现中变量的处理方法的实施例中执行了详细描述，此处将不做详细阐述说明。

[0173] 在示例性实施例中，还提供了一种存储介质，该存储介质为计算机可读存储介质，例如可以为包括指令的临时性和非临时性计算机可读存储介质。该存储介质例如包括指令的存储器204，上述指令可由装置200的处理器218执行以完成上述方法。

[0174] 应当理解的是，本发明并不局限于上面已经描述并在附图中示出的精确结构，并且可以在不脱离其范围执行各种修改和改变。本发明的范围仅由所附的权利要求来限制。

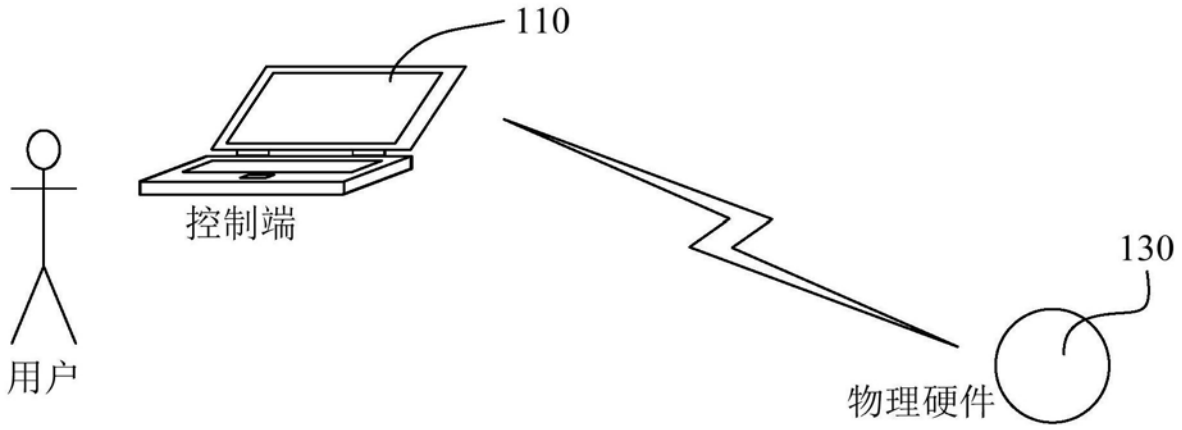

110
控制端
用户
130
物理硬件

图1

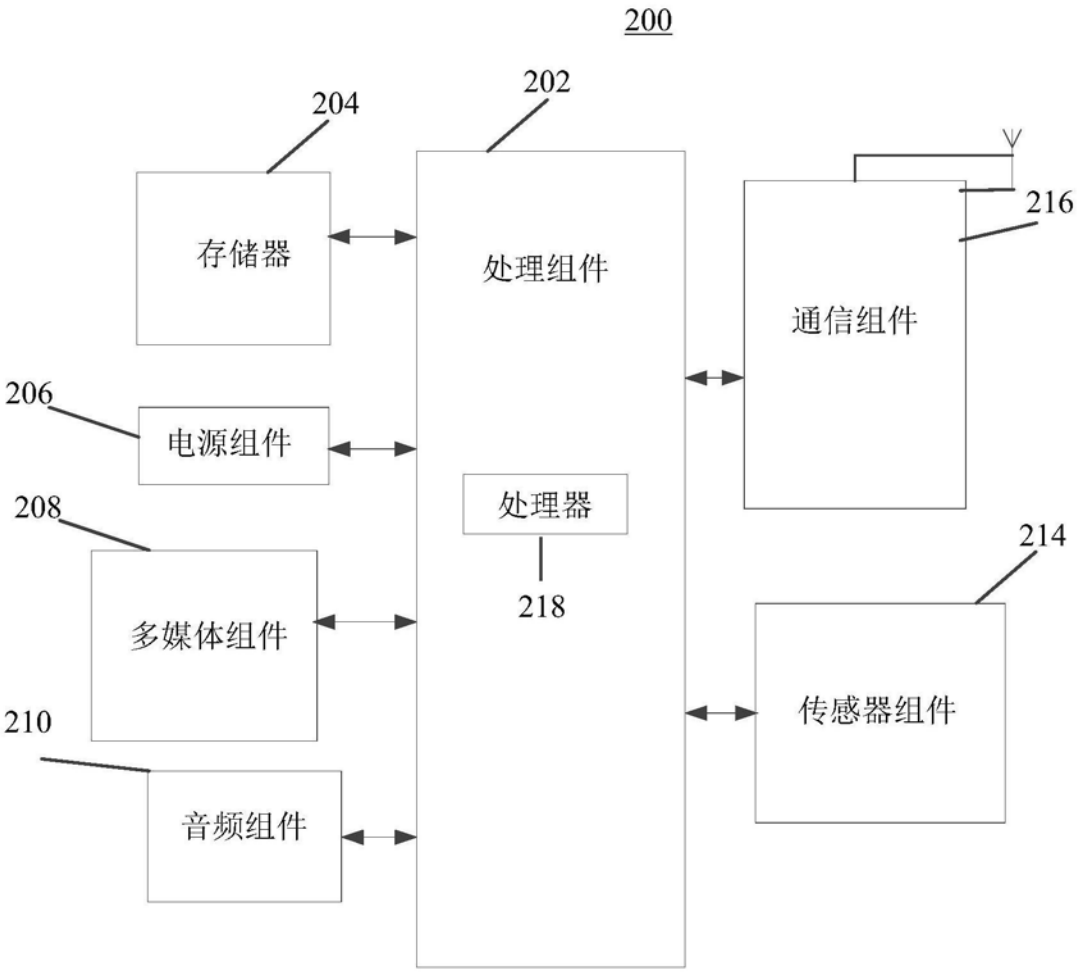

200
202
204
存储器
处理组件
通信组件
216
206
电源组件
处理器
218
208
多媒体组件
214
传感器组件
210
音频组件

图2

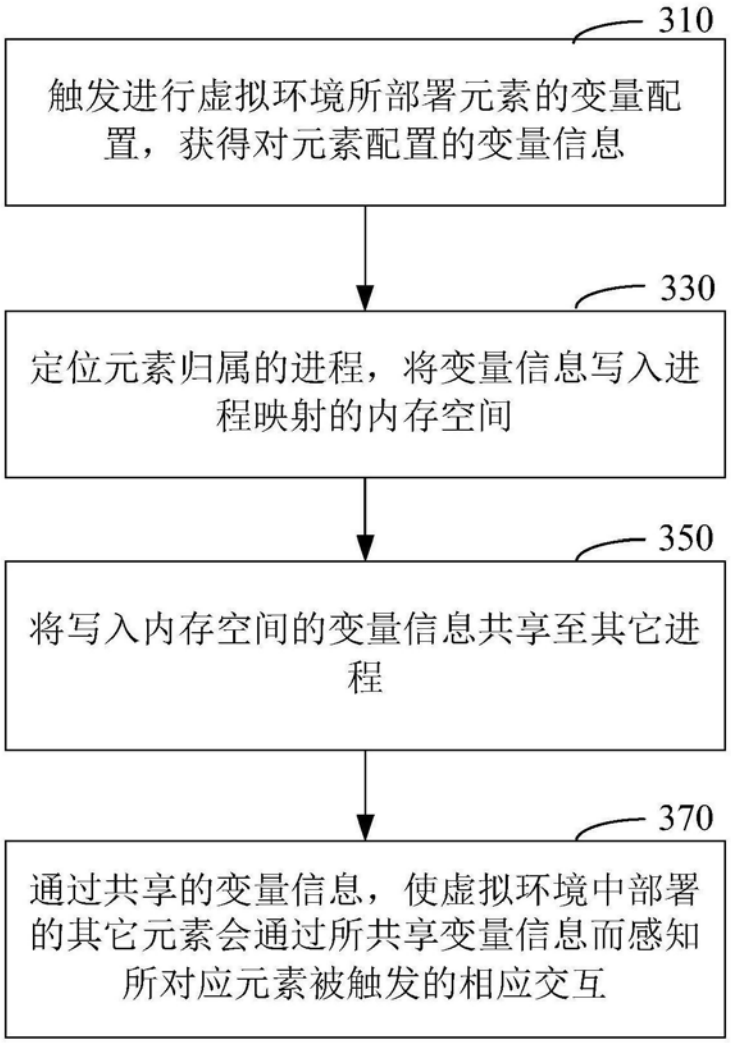

310
触发进行虚拟环境所部署元素的变量配置，获得对元素配置的变量信息
330
定位元素归属的进程，将变量信息写入进程映射的内存空间
350
将写入内存空间的变量信息共享至其它进程
370
通过共享的变量信息，使虚拟环境中部署的其它元素会通过所共享变量信息而感知所对应元素被触发的相应交互

图3

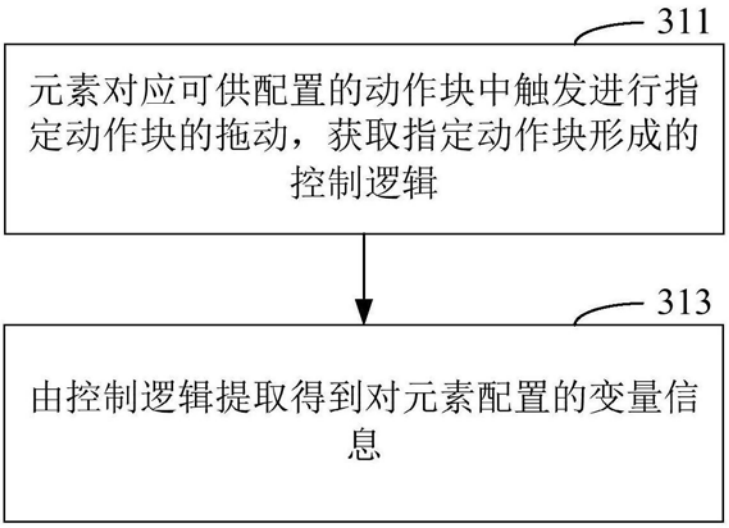

311
元素对应可供配置的动作块中触发进行指定动作块的拖动，获取指定动作块形成的控制逻辑
313
由控制逻辑提取得到对元素配置的变量信息

图4

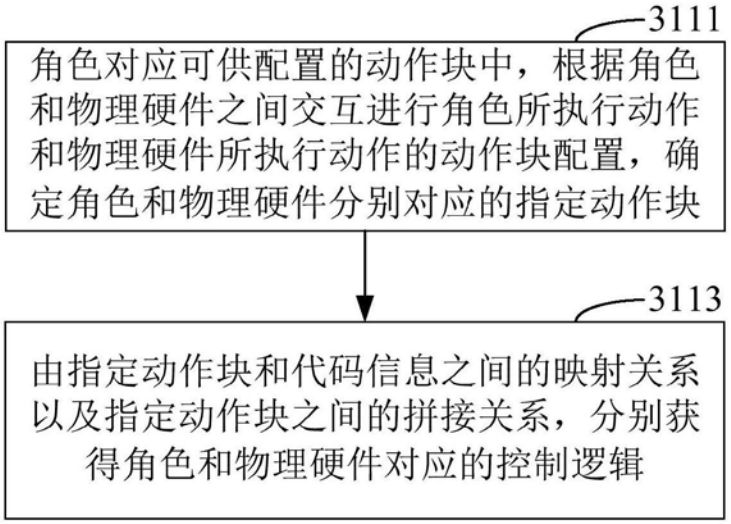

3111
角色对应可供配置的动作块中，根据角色和物理硬件之间交互进行角色所执行动作和物理硬件所执行动作的动作块配置，确定角色和物理硬件分别对应的指定动作块
3113
由指定动作块和代码信息之间的映射关系以及指定动作块之间的拼接关系，分别获得角色和物理硬件对应的控制逻辑

图5

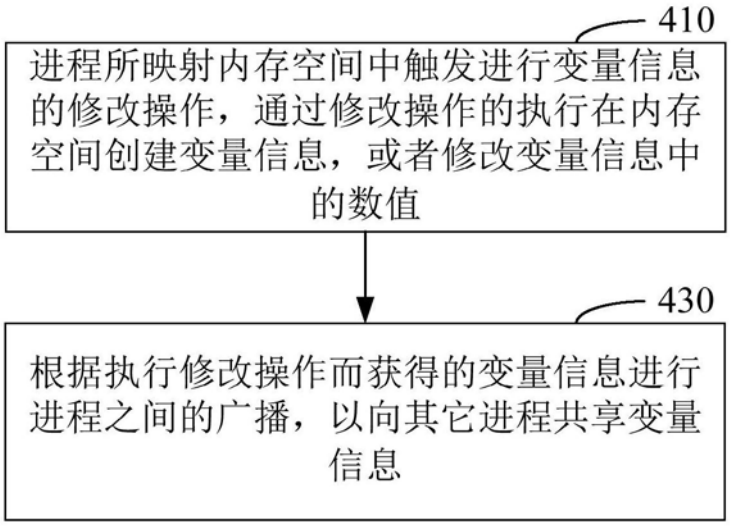

410
进程所映射内存空间中触发进行变量信息的修改操作，通过修改操作的执行在内存空间创建变量信息，或者修改变量信息中的数值
430
根据执行修改操作而获得的变量信息进行进程之间的广播，以向其它进程共享变量信息

图6

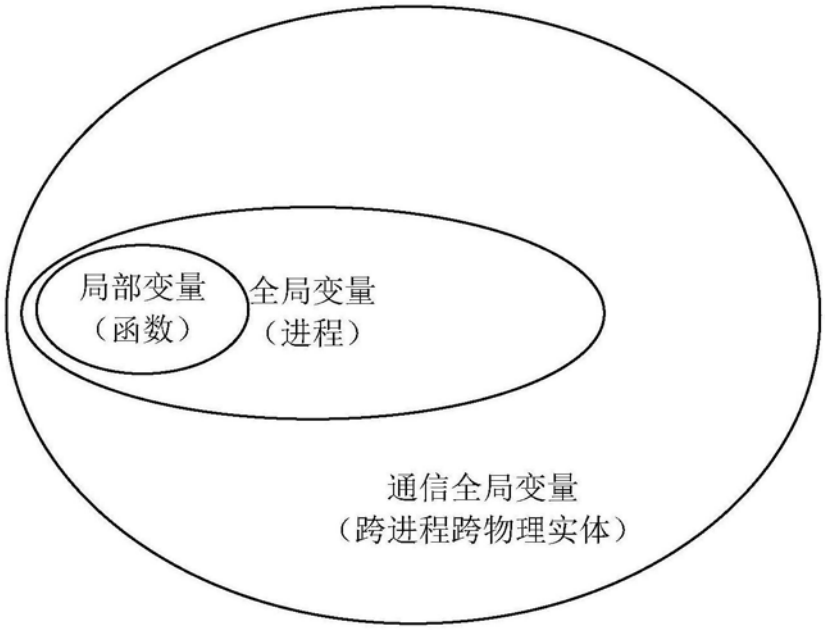

局部变量
（函数）
全局变量
（进程）
通信全局变量
（跨进程跨物理实体）

图7

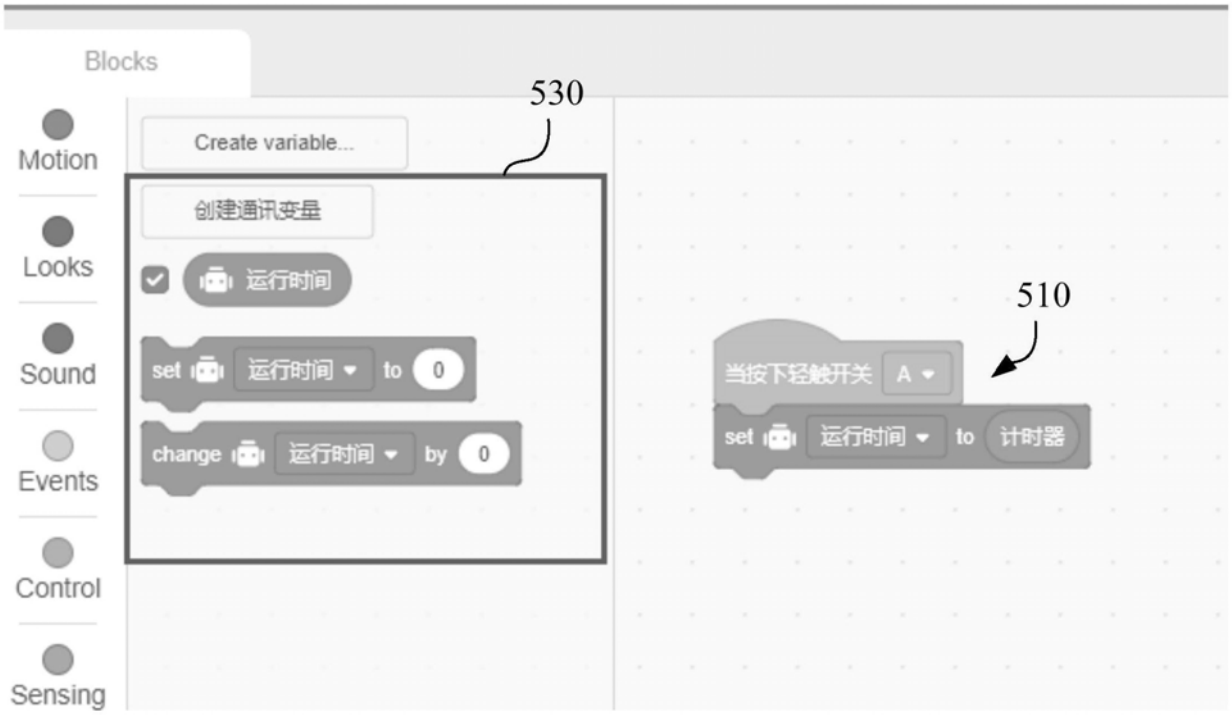

Blocks
Motion
Looks
Sound
Events
Control
Sensing
530
Create variable...
创建通讯变量
运行时间
set 运行时间 to 0
change 运行时间 by 0
510
当按下轻触开关 A
set 运行时间 to 计时器

图8

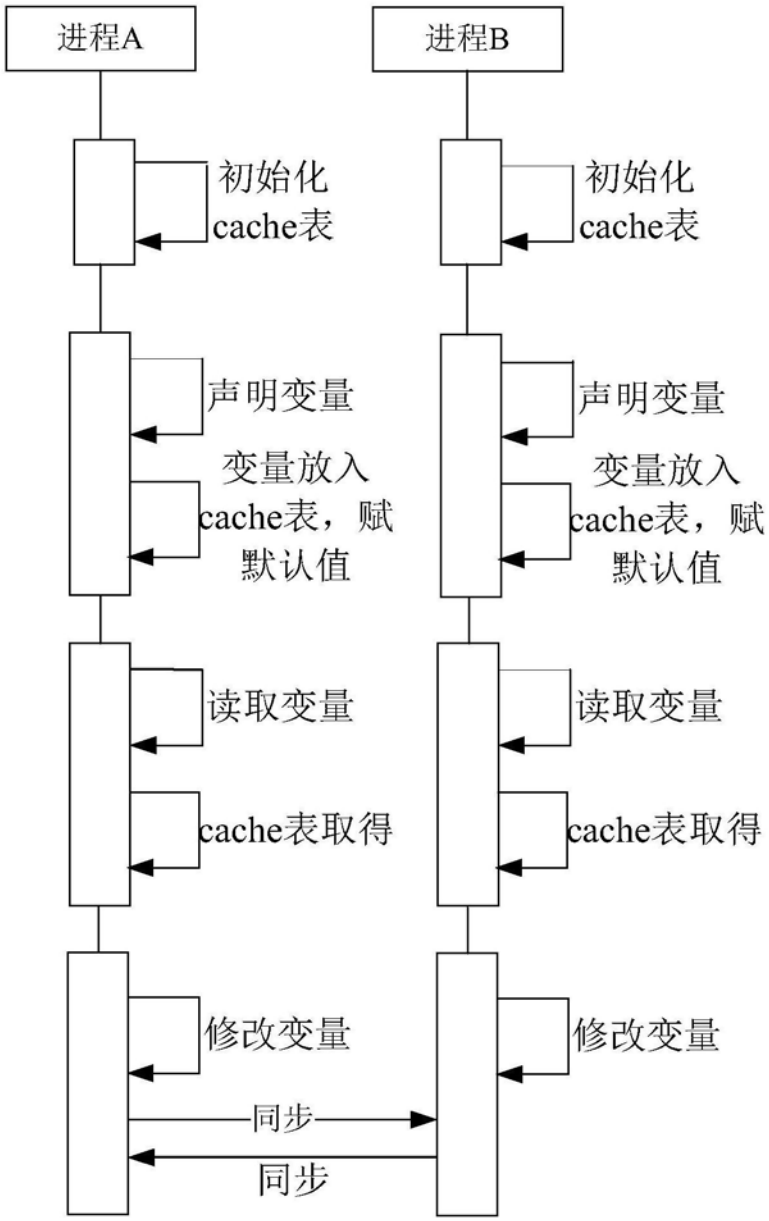

进程A
进程B
初始化cache表
初始化cache表
声明变量
变量放入cache表，赋默认值
声明变量
变量放入cache表，赋默认值
读取变量
cache表取得
读取变量
cache表取得
修改变量
修改变量
同步
同步

图9

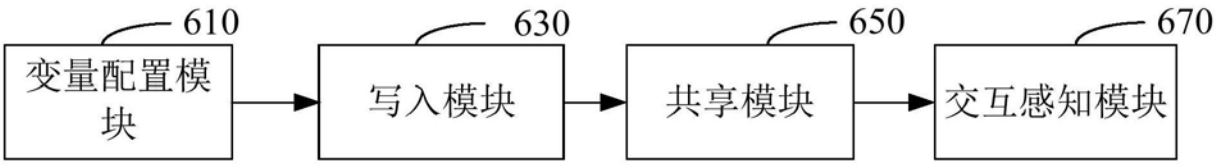

610
变量配置模块
630
写入模块
650
共享模块
670
交互感知模块

图10

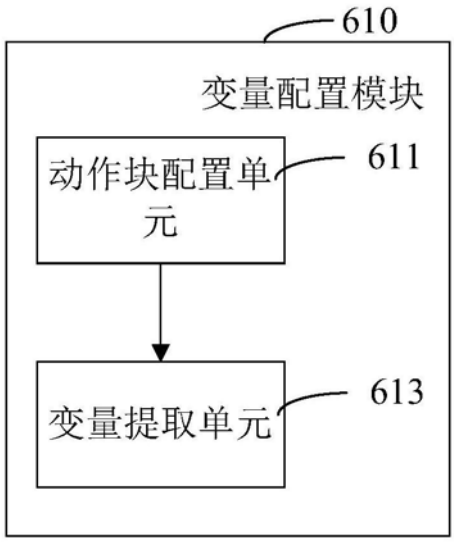


图11

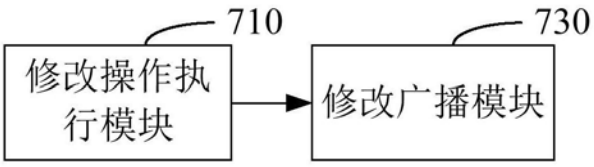


图12