



(12) 发明专利申请

(10) 申请公布号 CN 113785301 A

(43) 申请公布日 2021. 12. 10

(21) 申请号 202080002314.9

(22) 申请日 2020.09.21

(30) 优先权数据

16/864,708 2020.05.01 US

(85) PCT国际申请进入国家阶段日

2020.10.15

(86) PCT国际申请的申请数据

PCT/US2020/051878 2020.09.21

(87) PCT国际申请的公布数据

W02021/221713 EN 2021.11.04

(71) 申请人 尤帕斯公司

地址 美国纽约州

(72) 发明人 D·丹斯

(74) 专利代理机构 北京市金杜律师事务所

11256

代理人 辛鸣

(51) Int.Cl.

G06K 9/00 (2006.01)

G06K 9/20 (2006.01)

G06K 9/62 (2006.01)

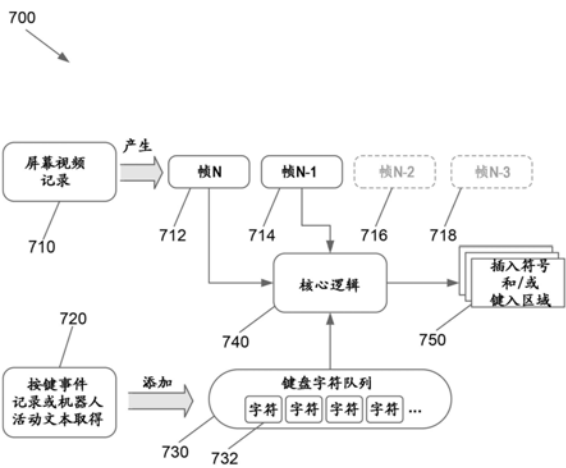
权利要求书4页 说明书22页 附图29页

(54) 发明名称

文本检测、插入符号追踪以及活动元素检测

(57) 摘要

公开了对用于机器人流程自动化 (RPA) 的机器人执行的屏幕响应验证。可以识别在RPA机器人正在与之交互的计算系统上执行的应用中是否发生文本、屏幕变化、图像和/或其他预期视觉动作。可以确定机器人正在键入的位置,并且可以基于一个或多个字符、图像、窗口等出现的当前分辨率来在屏幕上提供物理位置。这些元素的物理位置或缺少这些元素的物理位置可以允许确定机器人正在键入哪个(些)字段以及相关的应用是什么,以用于验证应用和计算系统是否按预期响应的目的。当未发生预期屏幕改变时,机器人可以停止并且引发异常,返回并且再次尝试预期交互,重新启动工作流,或者采取其他适当的措施。



1. 一种包括被实施在非瞬态计算机可读介质上的用于机器人流程自动化 (RPA) 的屏幕响应验证逻辑的计算机程序, 所述程序被配置为使至少一个处理器:

确定当前屏幕截图帧与前一屏幕截图帧之间是否存在差异, 所述当前屏幕截图帧和所述前一屏幕截图帧包括整个用户界面或所述用户界面的一部分;

当所述当前屏幕截图帧与所述前一屏幕截图帧之间存在所述差异、并且基于RPA机器人的工作流中的活动而预期所述当前屏幕截图帧与所述前一屏幕截图帧之间的差异时:

基于所述RPA机器人的所述工作流的所述活动对照一个或多个预期屏幕改变来验证所述当前屏幕截图帧与所述前一屏幕截图帧之间的一个或多个改变, 以及

当所述验证失败时, 引发异常或发起补救动作。

2. 根据权利要求1所述的计算机程序, 其中当所述当前屏幕截图帧与所述前一屏幕截图帧没有不同、并且基于所述RPA机器人的所述工作流中的所述活动而预期所述当前屏幕截图帧与所述前一屏幕截图帧之间没有差异时, 所述程序还被配置为使所述至少一个处理器:

取得下一屏幕截图帧; 以及

使用所述下一屏幕截图帧作为所述当前屏幕截图帧、并且使用所述当前屏幕截图帧作为所述前一屏幕截图帧, 重复所述过程。

3. 根据权利要求1所述的计算机程序, 其中当所述当前屏幕截图帧与所述前一屏幕截图帧没有不同、并且基于所述RPA机器人的所述工作流中的所述活动而预期所述当前屏幕截图帧与所述前一屏幕截图帧之间出现差异时, 所述程序还被配置为使所述至少一个处理器:

引发异常或发起补救动作。

4. 根据权利要求1所述的计算机程序, 其中所述补救动作包括: 所述RPA机器人在驱动级别检查针对所述机器人试图与之交互的应用的窗口是否为活动窗口, 检查鼠标指针或插入符号相对于预期发生改变的图形元素的位置, 检查正确的图形元素是否为活动元素, 或这些的任何组合。

5. 根据权利要求1所述的计算机程序, 其中所述补救动作包括: 所述RPA机器人将正确的窗口设置为活动的聚焦的窗口, 将焦点设置为活动元素, 移动鼠标并且引起鼠标点击事件, 或这些的任何组合。

6. 根据权利要求1所述的计算机程序, 其中所述异常的所述引发包括: 停止所述RPA机器人的操作, 在屏幕上显示消息, 发送指示所述RPA机器人未能实现所述一个或多个预期屏幕改变的消息, 列出一个或多个失败的动作, 或这些的组合。

7. 根据权利要求1所述的计算机程序, 其中所述程序还被配置为使所述至少一个处理器:

生成键字符队列, 所述键字符队列包括由所述RPA机器人创建的、在时间窗口期间发生的按键事件。

8. 根据权利要求7所述的计算机程序, 其中所述RPA机器人被配置为在按键事件之间等待一段时间。

9. 根据权利要求7所述的计算机程序, 其中当基于所述RPA机器人的所述工作流中的所述活动而预期所述当前屏幕截图帧与所述前一屏幕截图帧之间出现所述差异、并且基于所

述RPA机器人的所述工作流的所述活动的所述预期屏幕改变包括一个或多个字符时,所述程序还被配置为使所述至少一个处理器:

在所述当前屏幕截图帧与所述前一屏幕截图帧不同的一个或多个位置上执行光学字符识别(OCR),其中

所述验证包括将由所述OCR识别的字符与所述键字符队列中的字符匹配。

10.根据权利要求9所述的计算机程序,其中所述程序还被配置为使所述至少一个处理器:

从所述键字符队列删除匹配的字符。

11.根据权利要求7所述的计算机程序,其中所述键字符队列是先进先出(FIFO)队列,所述先进先出(FIFO)队列包括:按键事件的字符、以及在所述时间窗口期间针对所述按键事件中的每个按键事件发生所述按键事件的时间。

12.根据权利要求1所述的计算机程序,其中当所述当前屏幕截图帧与所述前一屏幕截图帧不同、并且基于所述RPA机器人的所述工作流中的所述活动而预期所述当前屏幕截图帧与所述前一屏幕截图帧之间出现所述差异时,所述程序还被配置为使至少一个处理器:

确定所述当前屏幕截图帧与所述前一屏幕截图帧不同的一个或多个区域;以及

在所确定的所述一个或多个区域中提取一个或多个连接成员,并且对所提取的所述一个或多个连接成员执行光学字符识别(OCR),从而产生一个或多个识别字符、一个或多个识别字符序列或两者,以及相应的位置,其中

所述验证包括将所述一个或多个识别字符、所述一个或多个识别字符序列或两者与所述键字符队列或者与基于所述活动而预期出现的一个或多个字符、一个或多个字符序列或两者比较。

13.根据权利要求1所述的计算机程序,其中所述程序还被配置为使所述至少一个处理器:

基于所述RPA机器人的所述工作流中的所述活动来确定目标图形元素和要被输入到所述目标图形元素中的内容;

验证所述当前屏幕截图帧与所述前一屏幕截图帧之间的所述一个或多个改变是否与所述目标图形元素的位置相匹配、以及所述一个或多个改变是否与要被输入到所述目标图形元素中的所述内容相匹配;以及

当所述位置、所述一个或多个改变或两者不匹配时,引发所述异常或发起所述补救动作。

14.根据权利要求1所述的计算机程序,其中所述活动包括将文本录入图形元素中,将插入符号移动到所述图形元素,将内容粘贴到所述图形元素中,或这些的组合。

15.根据权利要求1所述的计算机程序,其中基于所述RPA机器人的所述工作流的所述活动的所述一个或多个预期屏幕改变包括:表单的提交、新应用窗口的打开或当前应用窗口的改变。

16.一种包括被实施在非瞬态计算机可读介质上的用于机器人流程自动化(RPA)的屏幕响应验证逻辑的计算机程序,所述程序被配置为使至少一个处理器:

基于RPA机器人的工作流中的活动来确定目标图形元素和要被输入到所述目标图形元素中的内容;

确定当前屏幕截图帧与前一屏幕截图帧之间是否存在一个或多个差异,所述当前屏幕截图帧和所述前一屏幕截图帧包括整个用户界面或所述用户界面的一部分;以及

当所述当前屏幕截图帧与所述前一屏幕截图帧之间不存在所述一个或多个差异时,引发异常或发起补救动作。

17.根据权利要求16所述的计算机程序,其中当所述当前屏幕截图帧与所述前一屏幕截图帧之间存在所述一个或多个差异时,所述程序还被配置为使所述至少一个处理器:

验证所述当前屏幕截图帧与所述前一屏幕截图帧之间的变化是否与所述目标图形元素的位置相匹配、以及所述变化是否与要被输入到所述目标图形元素中的所述内容相匹配;以及

当所述位置、所述改变或两者不匹配时,引发异常或发起补救动作。

18.根据权利要求16所述的计算机程序,其中所述程序还被配置为使所述至少一个处理器:

应用计算机视觉算法以确定所述用户界面中的图形元素和相关联的边界矩形,所确定的所述图形元素包括所述目标图形元素。

19.根据权利要求16所述的计算机程序,其中所述程序还被配置为使所述至少一个处理器:

生成键字符队列,所述键字符队列包括由所述RPA机器人创建的、在时间窗口期间发生的按键事件。

20.根据权利要求19所述的计算机程序,其中当所述当前屏幕截图帧与所述前一屏幕截图帧之间存在所述一个或多个差异时,所述程序还被配置为使所述至少一个处理器:

在所述当前屏幕截图帧与所述前一屏幕截图帧不同的一个或多个位置上执行光学字符识别(OCR),其中

所述当前屏幕截图帧与所述前一屏幕截图帧之间的所述变化是否与要被输入到所述目标图形元素中的所述内容相匹配的所述验证包括:将由所述OCR识别的字符与所述键字符队列中的字符匹配。

21.根据权利要求16所述的计算机程序,其中所述活动包括将文本录入图形元素中,将插入符号移动到所述图形元素,将内容粘贴到所述图形元素中,或这些的组合。

22.根据权利要求16所述的计算机程序,其中基于所述RPA机器人的所述工作流的所述活动的所述一个或多个预期屏幕改变包括:表格的提交、新应用窗口的打开或当前应用窗口的改变。

23.一种用于执行用于机器人流程自动化(RPA)的屏幕响应验证的计算机实现的方法,包括:

由RPA机器人确定当前屏幕截图帧与前一屏幕截图帧之间是否存在差异,所述当前屏幕截图帧和所述前一屏幕截图帧包括整个用户界面或所述用户界面的一部分;以及

当所述当前屏幕截图帧与所述前一屏幕截图帧没有不同、并且基于所述RPA机器人的工作流中的活动而预期所述当前屏幕截图帧与所述前一屏幕截图帧之间出现差异时,由所述RPA机器人引发异常或发起补救动作。

24.根据权利要求23所述的计算机实现的方法,其中当所述当前屏幕截图帧与所述前一屏幕截图帧之间存在所述差异、并且基于所述RPA机器人的工作流中的活动而预期所述

当前屏幕截图帧与所述前一屏幕截图帧之间的差异时,所述方法还包括:

由所述RPA机器人基于所述RPA机器人的所述工作流的所述活动对照一个或多个预期屏幕改变来验证所述当前屏幕截图帧与所述前一屏幕截图帧之间的一个或多个改变;以及当所述验证失败时,由所述RPA机器人引发异常或发起补救动作。

文本检测、插入符号追踪以及活动元素检测

[0001] 相关申请的交叉引用

[0002] 本申请要求于2020年5月1日提交的美国非临时专利申请第16/864,708号的权益。该早前提交的申请的主题以其整体内容通过引用并入于此。

技术领域

[0003] 本发明总体上涉及机器人流程自动化 (RPA)，并且更具体地涉及用于RPA的机器人执行的屏幕响应验证。

背景技术

[0004] RPA机器人可以以类似于用户的方式与计算系统交互。例如，机器人可以移动鼠标、输入文本、单击按钮等。但是，与人类用户不同，RPA机器人无法“看到”计算系统对其交互做出的响应，并且RPA机器人可能会继续其逻辑，即使计算系统未按预期响应。因此，用于监测和验证RPA机器人与计算系统的交互的改进方法可能是有益的。

发明内容

[0005] 本发明的某些实施例可以提供对于尚未被当前的RPA技术完全识别、理解或解决的本领域的问题和需求的解决方案。例如，本发明的一些实施例涉及用于RPA的机器人执行的屏幕响应验证。

[0006] 在一个实施例中，一种包括用于RPA的屏幕响应验证逻辑的计算机程序被实现在非瞬态计算机可读介质上。该程序被配置为使至少一个处理器确定当前屏幕截图帧与前一屏幕截图帧之间是否存在差异。当前屏幕截图帧和前一屏幕截图帧包括整个用户界面或用户界面的一部分。当当前屏幕截图帧与前一屏幕截图帧之间存在差异并且基于RPA机器人的工作流中的活动而预期当前屏幕截图帧与前一屏幕截图帧之间存在差异时，该程序还被配置为使至少一个处理器：基于RPA机器人的工作流的活动对照一个或多个预期屏幕改变来验证当前屏幕截图帧与前一屏幕截图帧之间的一个或多个改变，并且当验证失败时，引发异常或发起补救动作。

[0007] 在另一实施例中，一种包括用于RPA的屏幕响应验证逻辑的计算机程序被实现在非瞬态计算机可读介质上。该程序被配置为使至少一个处理器基于RPA机器人的工作流中的活动确定目标图形元素和要被输入到目标图形元素中的内容，并且确定当前屏幕截图帧与前一屏幕截图帧之间是否存在一个或多个差异。当前屏幕截图帧和前一屏幕截图帧包括整个用户界面或用户界面的一部分。当当前屏幕截图帧与前一屏幕截图帧之间不存在一个或多个差异时，该程序还配置为使至少一个处理器引发异常或发起补救动作。

[0008] 在又一实施例中，一种用于执行用于RPA的屏幕响应验证的计算机实现的方法包括：由RPA机器人确定当前屏幕截图帧与前一屏幕截图帧之间是否存在差异。当前屏幕截图帧和前一屏幕截图帧包括整个用户界面或用户界面的一部分。当当前屏幕截图帧与前一屏幕截图帧没有不同并且基于RPA机器人的工作流中的活动而预期当前屏幕截图帧与前一屏

幕截图帧之间出现差异时,该计算机实现的方法包括由RPA机器人引发异常或发起补救动作。

附图说明

[0009] 为了容易地理解本发明的某些实施例的优点,将通过参考在附图中图示的特定实施例来呈现上文简要描述的本发明的更具体的描述。虽然应理解,这些图仅描绘了本发明的典型实施例,并且因此不应该被认为是对本发明的范围的限制,但是本发明将通过附图的使用以附加的特异性和细节来描述和阐释,在附图中:

[0010] 图1是图示了根据本发明的实施例的机器人流程自动化(RPA)系统的架构图。

[0011] 图2是图示了根据本发明的实施例的部署的RPA系统的架构图。

[0012] 图3是图示了根据本发明的实施例的设计器、活动与驱动之间的关系架构图。

[0013] 图4是图示了根据本发明的实施例的RPA系统的架构图。

[0014] 图5是示出根据本发明的实施例的被配置为执行用于RPA的机器人执行的屏幕响应验证的计算系统的架构图。

[0015] 图6A和图6B是示出根据本发明的实施例的用于RPA的机器人执行的屏幕响应验证的过程的流程图。

[0016] 图7是示出根据本发明的实施例的用于RPA的机器人执行的屏幕响应验证的过程的架构图。

[0017] 图8A图示了根据本发明的实施例的分别包括帧N-1和N的校友捐赠窗口的屏幕截图的部分。

[0018] 图8B图示了根据本发明的实施例的被划分成正方形网格的屏幕截图N-1。

[0019] 图8C图示了根据本发明的实施例的被划分成正方形网格的屏幕截图N。

[0020] 图8D图示了根据本发明的实施例的具有包括来自屏幕截图N-1的变化的亮行(highlighted row)的屏幕截图N。

[0021] 图8E图示了根据本发明的实施例的在包括来自屏幕截图N-1的变化的亮行中具有亮单元的屏幕截图N。

[0022] 图8F图示了根据本发明的实施例的具有包括来自屏幕截图N-1的变化的亮区域的屏幕截图N。

[0023] 图9是图示了根据本发明的实施例的用于检查分辨率变化、检查所连接的显示器的范围的变化以及设置插入符号追踪视频逻辑以适应变化的过程的流程图。

[0024] 图10是图示了根据本发明的实施例的用于录像机的过程的流程图。

[0025] 图11是图示了根据本发明的实施例的用于执行插入符号追踪的过程的流程图。

[0026] 图12A至图12D是示出根据本发明的实施例的用于执行用于RPA的机器人执行的屏幕响应验证的过程的流程图。

[0027] 图13A至图13C是示出根据本发明的实施例的用于执行粘贴的文本跟踪和验证的过程的流程图。

[0028] 图14A图示了根据本发明的实施例的在已经被组合成 32×32 区域的四个 16×16 块中所包括的字母“E”的示例二进制矩阵。

[0029] 图14B图示了根据本发明的实施例的针对图14A的区域的二进制矩阵中所包括的

成员(字母“E”)的剪裁后的 7×9 二进制矩阵。

[0030] 图15是示出根据本发明的实施例的使用CV和按键队列来确定活动元素并且验证机器人动作的过程的流程图。

[0031] 图16是示出根据本发明的实施例的被配置为执行用于RPA的机器人执行的屏幕响应验证的系统的架构图。

[0032] 图17是示出根据本发明的实施例的用于执行用于RPA的机器人执行的屏幕响应验证的过程的流程图。

[0033] 除非另有指示,否则类似的附图标记在整个附图中始终指示对应特征。

具体实施方式

[0034] 一些实施例涉及用于RPA的机器人执行的屏幕响应验证。一些实施例识别在RPA机器人正在与之交互的计算系统上执行的应用中是否发生文本、屏幕变化、图像和/或其他预期视觉动作。这样的实施例可以寻找机器人在与计算系统相关联的屏幕上已经在哪里键入并且基于一个或多个字符、图像、窗口等出现的当前分辨率(例如,以坐标表示)在屏幕上提供物理位置。这些元素的物理位置可以允许确定机器人正在键入哪个(些)字段以及相关的应用是什么,以验证应用和计算系统是否按预期响应。通过这个分析可以确定是否缺少字符、图像、窗口等,或者这些元素是否出现在错误的位置或具有类型错误,并且机器人可以停止并且引发异常,如果计算系统上的应用未实现工作流的功能,则返回并且再次尝试预期交互,重新启动工作流或采取任何其他适当的措施,而没有背离本发明的范围。

[0035] 一些实施例在反馈循环过程中实现,该反馈循环过程连续地或周期性地将当前屏幕截图与先前屏幕截图比较,或者将当前屏幕截图的一部分与先前屏幕截图的一部分比较(例如,与可见应用窗口相关联的屏幕截图的一部分),以标识是否发生改变,并且如果发生改变,则确定改变是什么。在某些实施例中,比较与窗口相关联的屏幕截图的部分,当机器人是与用户并肩工作的有人值守机器人时,该算法可以适应于窗口位置、窗口大小、窗口形状、窗口的缩放变量、其组合等的改变,以便算法识别出屏幕截图的当前部分映射到屏幕截图的先前部分。

[0036] 在键入的文本验证的情况下,可以标识屏幕上发生视觉变化的位置,并且可以在发生变化的位置执行光学字符识别(OCR)。然后将OCR的结果与按键队列的内容(例如,如由RPA机器人生成的按键事件所确定的)比较以确定是否存在匹配。发生改变的位置可以通过将当前屏幕截图中的像素框与先前屏幕截图中同一位置中的像素框比较来确定。

[0037] 某些实施例可以被部署用于机器人流程自动化(RPA)。图1是图示了根据本发明的实施例的RPA系统100的架构图。RPA系统100包括允许开发者设计和实现工作流的设计器110。设计器110可以提供用于应用集成以及自动化第三方应用、管理信息技术(IT)任务和业务IT过程的解决方案。设计器110可以支持自动化项目的开发,该自动化项目是业务过程的图形表示。简而言之,设计器110支持工作流和机器人的开发和部署。

[0038] 自动化项目通过向开发者提供对在工作流中开发的一组定制步骤(在本文中被定义为“活动”)之间的执行顺序和关系的控制来实现基于规则的过程的自动化。设计器110的实施例的一个商业示例是UiPath Studio™。每个活动可以包括动作,诸如点击按钮、读取文件、写入日志面板等。在一些实施例中,工作流可以被嵌套或嵌入。

[0039] 一些类型的工作流可以包括但不限于序列、流程图、FSM和/或全局异常处理程序。序列可能特别适合于线性过程,从而能够在不使工作流混乱的情况下从一个活动流向另一活动。流程图可能特别适合于更复杂的业务逻辑,从而实现通过多个分支逻辑运算符以更为不同的方式进行的决策的集成以及活动的连接。FSM可能特别适合于较大工作流。FSM可以在其执行中使用有限数量的状态,这些状态由条件(即,过渡)或活动触发。全局异常处理程序可能特别适合于在遇到执行误差时确定工作流行为且特别适合于调试过程。

[0040] 一旦工作流在设计器110中被开发,业务过程的执行就由指挥器(conductor)120编排,该指挥器编排执行设计器110中所开发的工作流的一个或多个机器人130。指挥器120的实施例的一个商业示例是UiPath Orchestrator™。指挥器120支持对环境中的资源的创建、监测以及部署的管理。利用第三方解决方案和应用,指挥器120可以充当集成点、或充当聚合点中的一者。

[0041] 指挥器120可以管理一队机器人130,从而从集中点连接并执行机器人130。可以管理的机器人130的类型包括但不限于:有人值守的(attended)机器人132、无人值守的机器人134、开发机器人(类似于无人值守的机器人134,但用于开发和测试目的)以及非生产型机器人(类似于有人值守的机器人132,但用于开发和测试目的)。有人值守的机器人132由用户事件触发并且与人类一起在相同计算系统上操作。有人值守的机器人132可以与指挥器120一起用于集中过程部署和记录介质。有人值守的机器人132可以帮助人类用户完成各种任务,并且可以由用户事件触发。在一些实施例中,无法从这种类型的机器人上的指挥器120开启过程和/或这些过程无法在锁定屏幕下运行。在某些实施例中,有人值守的机器人132只可以从机器人托盘或根据命令提示而被开启。在一些实施例中,有人值守的机器人132应该在人类监督下运行。

[0042] 无人值守的机器人134在虚拟环境中以无人值守的方式运行并且可以使许多过程自动化。无人值守的机器人134可以负责远程执行、监测、调度并且为工作队列提供支持。在一些实施例中,可以在设计器110中运行针对所有机器人类型的调试。有人值守的机器人和无人值守的机器人两者都可以使各种系统和应用自动化,该系统和应用包括但不限于大型主机、web应用、VM、企业应用(例如由 **SAP®**、**SalesForce®**、**Oracle®** 等生产的那些企业应用)以及计算系统应用(例如台式机和笔记本应用、移动设备应用、可穿戴计算机应用等)。

[0043] 指挥器120可以具有各种能力,包括但不限于供应(provisioning)、部署(deployment)、版本化(versioning)、配置(configuration)、排队(queueing)、监测(monitors)、日志记录(logging)和/或提供互连性(interconnectivity)。供应可以包括在机器人130与指挥器120(例如web应用)之间创建和维护连接。开发可以包括确保将包(package)版本正确递送至所指派的机器人130以供执行。在一些实施例中,版本化可以包括某一过程或配置的唯一实例的管理。配置可以包括维护和递送机器人环境和过程配置。排队可以包括提供对队列和队列项的管理。监测可以包括追踪机器人标识数据和维护用户权限。日志记录可以包括将日志存储和编入索引至数据库(例如SQL数据库)和/或另一存储机制(例如 **ElasticSearch®**, 其提供用于存储和快速查询较大数据集的能力)。指挥器120可以通过充当第三方解决方案和/或应用的通信的集中点来提供互连性。

[0044] 机器人130是运行内置于设计器110中的工作流的执行代理。(多个)机器人130的一些实施例的一个商业示例是UiPath Robots™。在一些实施例中,机器人130默认安装Microsoft Windows®服务控制管理器(SCM)管理的服务。结果,此类机器人130可以在本地系统账户下打开交互式Windows®会话,并且具有Windows®服务的权限。

[0045] 在一些实施例中,机器人130可以在用户模式下被安装。针对此类机器人130,这意味着其具有与已经安装了给定机器人130的用户相同的权限。此特征也可以供高密度(HD)机器人使用,该高密度机器人确保了每台机器以其最大潜力的充分利用。在一些实施例中,任何类型的机器人130可以在HD环境中被配置。

[0046] 在一些实施例中,机器人130被分成分别专用于特定自动化任务的若干组件。在一些实施例中,机器人组件包括但不限于SCM管理的机器人服务、用户模式机器人服务、执行器、代理以及命令行。SCM管理的机器人服务管理和监测Windows®会话,并且充当指挥器120与执行主机(即,机器人130在其上被执行的计算系统)之间的委托。这些服务受机器人130信任并且管理机器人130的凭证。控制台应用在本地系统下由SCM启动。

[0047] 在一些实施例中,用户模式机器人服务管理和监测Windows®会话,并且充当指挥器120与执行主机之间的委托。用户模式机器人服务可以受机器人130信任并且管理机器人130的凭证。在未安装SCM管理的机器人服务的情况下,Windows®应用可以被自动启动。

[0048] 执行器可以在Windows®会话下运行给定作业(即,执行器可以执行工作流。执行器可以知道每个监测器的每英寸点数(DPI)设置。代理可以是Windows® Presentation Foundation(WPF)应用,其在系统托盘窗口中显示可用的作业。代理可以是服务的客户端。代理可以请求开启或停止作业以及改变设置。命令行是服务的客户端。命令行是可以请求开启作业并等待其输出的控制台应用。

[0049] 如上文所阐释一般使机器人130的组件分开帮助开发者、支持用户以及计算系统更容易地运行、标识和追踪每个组件正在执行的内容。特殊行为可以以这种方式按组件来配置,诸如针对执行器和服务设置不同的防火墙规则。在一些实施例中,执行器可以始终知道每个监测器的DPI设置。结果,不论创建工作流的计算系统的配置如何,工作流都可以在任何DPI处被执行。在一些实施例中,来自设计器110的项目也可以独立于浏览器缩放级别。在一些实施例中,针对DPI不知道的或有意地标记为不知道的应用,DPI可以被禁用。

[0050] 图2是图示了根据本发明的实施例的部署的RPA系统200的架构图。在一些实施例中,RPA系统200可以是图1的RPA系统100或可以是该RPA系统的一部分。应注意,在不脱离本发明的范围的情况下,客户端侧、服务器侧或两者都可以包括任何期望数量的计算系统。在客户端侧,机器人应用210包括执行器212、代理214和设计器216。然而,在一些实施例中,设计器216可以不在计算系统210上运行。执行器212正在运行过程。如图2中所示,若干业务项目可以同时运行。在该实施例中,代理214(例如Windows®服务)是所有执行器212的单个联系点。该实施例中的所有消息被日志记录到指挥器230中,该指挥器经由数据库服务器240、索引器服务器250或两者进一步处理这些消息。如上文相对于图1所讨论的,执行器212

可以是机器人组件。

[0051] 在一些实施例中,机器人表示机器名称与用户名之间的关联。机器人可以同时管理多个执行器。在支持同时运行的多个交互式会话的计算系统(例如Windows®服务器2012)上,多个机器人可以同时运行,每个机器人在使用唯一用户名的单独Windows®会话中。上文将此称为HD机器人。

[0052] 代理214还负责发送机器人的状态(例如周期性地发送指示机器人仍在运作的“心跳”消息)并且下载要被执行的包的所需版本。在一些实施例中,代理214与指挥器230之间的通信始终由代理214发起。在通知场景中,代理214可以打开WebSocket信道,稍后由指挥器230使用该WebSocket信道来向机器人发送命令(例如开启、停止等)。

[0053] 在服务器侧,包括了呈现层(web应用232、开放数据协议(OData)代表性状态转移(REST)应用编程接口(API)端点234以及通知和监测236)、服务层(API实现/业务逻辑238)以及持久层(数据库服务器240和索引器服务器250)。指挥器230包括web应用232、OData REST API端点234、通知和监测236以及API实现/业务逻辑238。在一些实施例中,用户在指挥器220的接口中(例如经由浏览器220)执行的大多数动作是通过调用各种API来执行的。在不脱离本发明的范围的情况下,此类动作可以包括但不限于在机器人上开启作业、在队列中添加/移除数据、调度作业以便以无人值守的方式运行等。Web应用232是服务器平台的视觉层。在该实施例中,web应用232使用超文本标记语言(HTML)和JavaScript(JS)。然而,可以在不脱离本发明的范围的情况下使用任何期望标记语言、脚本语言或任何其他格式。在该实施例中,用户经由浏览器220与来自web应用232的web页面交互,以便执行各种动作来控制指挥器230。例如,用户可以创建机器人群组、向机器人指派包、分析每个机器人和/或每个过程的日志、开启和停止机器人等。

[0054] 除了web应用232之外,指挥器230还包括暴露OData REST API端点234的服务层。然而,在不脱离本发明的范围的情况下可以包括其他端点。REST API由web应用232和代理214两者消耗。在该实施例中,代理214是客户端计算机上的一个或多个机器人的监督器。

[0055] 在该实施例中,REST API覆盖配置、日志记录、监测以及排队功能性。在一些实施例中,配置端点可以用于定义和配置应用用户、权限、机器人、资产、发布以及环境。日志记录REST端点可以用于日志记录不同信息,诸如(例如)误差、由机器人发送的显式消息以及其他环境特定信息。如果在指挥器230中使用了开启作业命令,那么可以由机器人使用部署REST端点来查询应该被执行的包版本。排队REST端点可以负责队列和队列项管理,诸如将数据添加至队列、从队列中获得事务、设置事务的状态等。

[0056] 监测REST端点可以监测web应用232和代理214。通知和监测API 236可以是用于注册代理214、向代理214递送配置设置并且用于从服务器和代理214发送/接收通知的REST端点。在一些实施例中,通知和监测API 236还可以使用WebSocket通信。

[0057] 在该实施例中,持久层包括一对服务器——数据库服务器240(例如SQL服务器)和索引器服务器250。在该实施例中,数据库服务器240存储机器人、机器人群组、相关联的过程、用户、角色、调度等的配置。在一些实施例中,该信息通过web应用232来管理。数据库服务器240可以管理队列和队列项。在一些实施例中,数据库服务器240可以存储由机器人(除了索引器服务器250之外或代替索引器服务器250)日志记录的消息。

[0058] 在一些实施例中可选的索引器服务器250存储由机器人日志记录的信息并且将该信息编入索引。在某些实施例中,索引器服务器250可以通过配置设置来禁用。在一些实施例中,索引器服务器250使用ElasticSearch[®],其为开源项目全文搜索引擎。由机器人日志记录的消息(例如使用比如日志消息或写入行的活动)可以通过(多个)日志记录的REST端点被发送到索引器服务器250,在索引器服务器250中这些消息被编入索引以供将来使用。

[0059] 图3是图示了根据本发明的实施例的设计器310、活动320、330与驱动(driver)340之间的关系300的架构图。根据上文,开发者使用设计器310来开发由机器人执行的工作流。工作流可以包括用户定义的活动320和UI自动化活动330。一些实施例能够标识图像中的非文本视觉分量,在本文中称作计算机视觉(CV)。与此类分量相关的一些CV活动可以包括但不限于点击、键入(type)、取得文本、悬停、元素存在、刷新范围、高亮等。在一些实施例中,点击使用例如CV、光学字符识别(OCR)、模糊文本匹配以及多锚(multi-anchor)来标识元素,并且点击该元素。键入可以使用上述内容来标识元素和该元素中的类型。取得文本可以标识特定文本的位置并且使用OCR来扫描该位置。悬停可以标识元素并且悬停于元素上方。元素存在可以使用上文所描述的技术来检查在屏幕上是否存在元素。在一些实施例中,可以存在可以在设计器310中被实现的数百个甚至数千个活动。然而,在不脱离本发明的范围的情况下,任何数量和/或类型的活动都可以是可用的。

[0060] UI自动化活动330是利用较低级代码编写的特殊的较低级活动(例如CV活动)的子集并且有助于与屏幕的交互。UI自动化活动330经由允许机器人与期望软件交互的驱动340来支持这些交互。例如,驱动340可以包括OS驱动342、浏览器驱动344、VM驱动346、企业应用驱动348等。

[0061] 驱动340可以与OS进行低级交互,进而寻找挂钩、监测键等。驱动器可以有助于与Chrome[®], IE[®], Citrix[®], SAP[®]等的集成。例如,“点击”活动经由驱动340在这些不同应用中扮演相同角色。

[0062] 图4是图示了根据本发明的实施例的RPA系统400的架构图。在一些实施例中,RPA系统400可以是或包括图1和/或图2的RPA系统100和/或200。RPA系统400包括运行机器人的多个客户端计算系统410。计算系统410能够经由在其上运行的web应用与指挥器计算系统420进行通信。指挥器计算系统420又能够与数据库服务器430和可选的索引器服务器440进行通信。

[0063] 相对于图1和图3,应注意,虽然在这些实施例中使用了web应用,但可以在不脱离本发明的范围的情况下使用任何合适的客户端/服务器软件。例如,指挥器可以运行服务器侧应用,该服务器侧应用与客户端计算系统上的非基于网络的客户端软件应用进行通信。

[0064] 图5是示出根据本发明的实施例的被配置为执行用于RPA的机器人执行的屏幕响应验证的计算系统500的架构图。在一些实施例中,计算系统500可以是本文中所描绘和/或描述的计算系统中的一者或多者。计算系统500包括总线505或用于传送信息的其他通信机制、以及被耦合至总线505以处理信息的(多个)处理器510。(多个)处理器510可以是任何类型的通用或专用处理器,包括中央处理单元(CPU)、专用集成电路(ASIC)、现场可编程门阵列(FPGA)、图形处理单元(GPU)、其多个实例和/或其任何组合。(多个)处理器510还可以具

有多个处理核心,并且至少一些核心可以被配置为执行特定功能。在一些实施例中,可以使用多并行处理。在某些实施例中,(多个)处理器510中的至少一者可以是包括模仿生物神经的处理元件的神经形态电路。在一些实施例中,神经形态电路可以不需要冯·诺依曼计算架构的典型组件。

[0065] 计算系统500还包括用于存储将由(多个)处理器510执行的信息和指令的存储器515。存储器515可以由随机存取存储器(RAM)、只读存储器(ROM)、闪速存储器、缓存、静态存储装置(诸如磁盘或光盘)或任何其他类型的非瞬态计算机可读介质或其组合的任何组合组成。非瞬态计算机可读介质可以是可由(多个)处理器510访问的任何可用介质,并且可以包括易失性介质、非易失性介质或两者。介质也可以是可移除的、不可移除的或两者。

[0066] 附加地,计算系统500包括通信设备520(诸如收发器)以经由无线和/或有线连接提供对通信网络的访问。在一些实施例中,在不脱离本发明的范围的情况下,通信设备520可以被配置为使用频分多址(FDMA)、单载波FDMA(SC-FDMA)、时分多址(TDMA)、码分多址(CDMA)、正交频分复用(OFDM)、正交频分多址(OFDMA)、全球移动系统(GSM)通信、通用分组无线电服务(GPRS)、通用移动通信系统(UMTS)、cdma2000、宽带CDMA(W-CDMA)、高速下行分组接入(HSDPA)、高速上行分组接入(HSUPA)、高速分组接入(HSPA)、长期演进(LTE)、LTE高级(LTE-A)、802.11x、Wi-Fi、Zigbee、超宽带(UWB)、802.16x、802.15、家用节点B(HnB)、蓝牙、射频标识(RFID)、红外数据协会(IrDA)、近场通信(NFC)、第五代(5G)、新无线电(NR)、其任何组合和/或任何其他现有或将来实现的通信标准和/或协议。在一些实施例中,在不脱离本发明的范围的情况下,通信设备520可以包括一个或多个天线,这些天线是单数的、阵列的、相控的、切换的、波束成形的、波束导向的、其组合和/或任何其他天线配置。

[0067] (多个)处理器510进一步经由总线505被耦合到显示器525,诸如等离子体显示器、液晶显示器(LCD)、发光二极管(LED)显示器、场发射显示器(FED)、有机发光二极管(OLED)显示器、柔性OLED显示器、柔性基板显示器、投影显示器、4K显示器、高清显示器、**Retina[®]**显示器、面内切换(IPS)显示器或用于向用户显示信息的任何其他合适的显示器。显示器525可以被配置为使用电阻性、电容性、表面声波(SAW)电容性、红外、光学成像、色散信号技术、声学脉冲识别、受抑全内反射等的触摸(触觉)显示器、三维(3D)触摸显示器、多输入触摸显示器、多触摸显示器等。可以在不脱离本发明的范围的情况下使用任何合适的显示设备和触觉I/O。

[0068] 键盘530和光标控制设备535(诸如计算机鼠标、触摸板等)进一步被耦合到总线505以使得用户能够与计算系统500接口连接。然而,在某些实施例中,可以不存在物理键盘和鼠标,并且用户可以仅通过显示器525和/或触摸板(未示出)与设备交互。输入设备的任何类型和组合可以被用作设计选择的问题。在某些实施例中,不存在物理输入设备和/或显示器。例如,用户可以经由与之通信的另一计算系统与计算系统500远程地交互,或计算系统500可以自主地操作。

[0069] 存储器515存储在被(多个)处理器510执行时提供功能性的软件模块。模块包括计算系统500的操作系统540。模块还包括被配置为执行本文中所描述的过程或其衍生物的全部或部分的视频处理模块545。计算系统500可以包括一个或多个附加功能模块550,该功能模块包括附加功能性。

[0070] 本领域的技术人员应了解,在不脱离本发明的范围的情况下,“系统”可以被实施

为服务器、嵌入式计算系统、个人计算机、控制台、个人数字助理 (PDA)、手机、平板计算设备、量子计算系统或任何其他合适的计算设备或设备的组合。将上文所描述的功能呈现为由“系统”执行并不旨在以任何方式限制本发明的范围,而是旨在提供本发明的许多实施例的一个示例。实际上,本文中所公开的方法、系统和装置可以按照与包括云计算系统的计算技术一致的本地化和分布式形式来实现。

[0071] 应注意,本说明书中所描述的一些系统特征已经作为模块而被呈现,以便更具体地强调其实现独立性。例如,模块可以被实现为包括定制的超大规模集成 (VLSI) 电路或门阵列、现成半导体 (诸如逻辑芯片、晶体管或其他分立组件) 的硬件电路。模块也可以被实现在可编程硬件设备中,诸如现场可编程门阵列、可编程阵列逻辑、可编程逻辑设备、图形处理单元等。

[0072] 模块也可以至少部分地被实现在软件中以便由各种类型的处理器执行。例如,可执行代码的所标识的单元可以包括计算机指令的一个或多个物理块或逻辑块,这些计算机指令可以例如被组织成对象、过程或功能。虽然如此,但所标识的模块的可执行文件不需要在物理上位于一处,而是可以包括存储在不同位置中的全异的指令,这些指令在以逻辑方式被结合在一起时包括该模块并且实现了该模块的既定目的。此外,模块可以被存储在计算机可读介质上,在不脱离本发明的范围的情况下,该计算机可读介质可以是例如硬盘驱动器、闪存设备、RAM、磁带和/或用于存储数据的任何其他此类非瞬态计算机可读介质。

[0073] 实际上,可执行代码的模块可以是单个指令或许多指令,并且甚至可以分布在若干不同的代码段上、分布在不同程序之间以及跨多个存储器设备分布。类似地,操作数据在本文中可以被标识和图示为在模块内,并且该操作数据可以被实施为任何合适的形式并且被组织在任何合适类型的数据结构内。操作数据可以被收集为单个数据集,或可以分布于包括在不同存储设备上的不同位置上,并且可以至少部分地仅作为系统或网络上的电子信号存在。

[0074] 图6A和图6B是示出根据本发明的实施例的用于RPA的机器人执行的屏幕响应验证的过程600的流程图。执行按键记录602和屏幕记录604 (例如,记录屏幕截图并且进行实时比较,可能会由服务器稍后处理的较长系列的屏幕截图的屏幕视频记录等),以分别确定由机器人“按下”的按键 (例如,由机器人在计算系统中使用的按键事件、应用级API调用等) 和发生图形改变的屏幕上的位置。但是,在某些实施例中,由于可以根据机器人的活动确定机器人试图输入的文本,因此不执行按键记录。尽管本文中将由机器人输入的字符称为“按键”,而将步骤602称为“按键记录”,但RPA机器人实际上并未在物理键盘上键入。而是,由于用户物理地在键盘上键入,RPA机器人生成的系统和/或应用级事件与由应用、操作系统或两者生成的事件相同或相似。在一些实时或接近实时的实施例中,由于帧差可以在运行中计算,因此可以不记录和存储视频数据。

[0075] 按键记录602可以通过监测来自操作系统的按键事件来执行。但是,可以使用任何按键事件或为任何编程语言和任何操作系统 (例如,移动、PC、Mac等) 提供按键信息的功能,而没有背离本发明的范围。

[0076] 按键事件可以包括关于哪个字符与被按下的键 (例如字母“a”、数字“2”、“%”符号等) 相关联、发生按键事件的时间等的信息。键字符的队列 (例如先进先出 (FIFO) 队列) 可以在时间窗口 (例如500毫秒 (ms)、一秒等) 内被存储,以考虑到在键被按下的时间与在屏幕上

出现对应字符的时间之间的延迟。时间窗口通常长于在用户按下键的时间与在屏幕上出现键的时间之间的典型时间延迟(例如50ms的字符出现延迟和500ms的缓冲窗口)。

[0077] 队列还可以用于捕获同时出现在屏幕上的多个字符,这由于快速的现代计算硬件所致的机器人操作速度而经常发生。例如,如果机器人极为快速连续地(例如在1ms内)按下“abc”,但每秒仅25帧被捕获(即,每40ms捕获一帧),那么文本“abc”可能会在下一个屏幕截图中才能全部出现。通过在队列中具有“a”、“b”和“c”,当文本识别找到这些字符和/或序列时,算法可以搜索这些字符和/或其序列中的每一者。例如,在一些实施例中,如果机器人键入“abc”,并且“ab”在下一帧中出现,那么可以假设按键队列中的顺序与屏幕上出现的顺序相同。

[0078] 在某些实施例中,机器人可以在按键之间提供延迟。例如,机器人可以创建按键事件,等待5ms,创建另一按键事件,等等。这可能导致机器人与计算系统进行更多类似用户的交互,并且可能还有助于验证,因为从一个屏幕截图到另一屏幕截图的改变可能不太极端并且更接近于人机交互的速度。附加地或备选地,机器人可以在输入文本与“按下”提交按钮、发送电子邮件按钮等之间等待足够的时间,以便允许改变在屏幕截图中出现以进行验证,然后在提交之后消失。

[0079] 在不脱离本发明的范围的情况下,屏幕记录604可以由任何合适的硬件、软件或其任何组合来执行。例如,视频记录可以通过外部摄像机、内部图形卡、监测来自显示器的视频流(经由或不经由所连接的计算系统)的远程计算系统等来执行。所记录的屏幕截图可以按照任何期望的格式被存储,无论像素是否完美(例如JPEG、BMP、PNG等)。

[0080] 在一种实现中,屏幕截图以Format16bppRgb555格式被存储为呈16比特颜色深度的BitmapData。将屏幕截图压缩为有损格式可能会降低准确性,这是由于一个或多个像素的变化可以改变以某些格式被传播至整个图像的颜色图,影响颜色深度,减少/移位总体细节或因压缩伪影(例如呈JPEG格式)的存在而影响图像像素。为了适应此情况,可以使用公差。屏幕记录604以特定帧速率产生帧,该帧速率可以根据计算系统的当前处理负载(例如每秒30帧)而变化。

[0081] 然后,在606处,将当前屏幕截图帧和紧邻的前一帧彼此比较以确定它们之间是否存在差异(例如,其中的至少一些对应像素彼此不同)。根据机器人正在执行工作流的当前操作,可能会或可能不会预期对应屏幕改变。在608处没有发生屏幕改变并且在610处基于工作流执行而预期发生屏幕改变的情况下,该过程在612处前进到下一屏幕截图,并且在606处返回以将该下一屏幕截图与先前的当前屏幕截图比较。

[0082] 如果在608处没有发生任何屏幕改变,并且在610处发生了预期改变,或者如果在608处发生了屏幕改变,但在614处没有发生预期改变,则在616处,系统将引发异常或者指示机器人采取补救动作。在引发异常的情况下,机器人的操作可以停止,并且可以在屏幕上显示消息或向计算系统的用户发送消息(例如,经由文本或电子邮件),以指示机器人未能实现期望动作。在某些实施例中,可以将失败的动作提供给用户,以便他或她可以尝试对问题进行故障排除,或者用户可以报告将由RPA开发人员解决的问题。

[0083] 除了引发异常,另一选项是让机器人采取补救动作。可以向机器人通知工作流的哪个(些)步骤失败,采取某种纠正措施,然后再次尝试该步骤。这可以包括但不限于机器人在驱动级别检查机器人尝试与之交互的针对应用的窗口是否为活动窗口,检查鼠标指针或

插入符号相对于相关图形元素的位置,检查正确的图形元素是否为活动元素,其任何组合等。机器人可以将正确的窗口设置为活动的聚焦的窗口,将焦点设置为期望的活动元素,移动鼠标并且单击,等等。然后再次尝试该步骤。如果补救动作成功,则该过程可以返回到步骤612。但是,如果补救动作失败,则可以引发异常。

[0084] 如果在608处确实发生了屏幕改变,并且在614处基于工作流而预期改变,则该过程进行到图6B。在618处确定屏幕改变的区域并且确定区域中的字符(如果有的话)。在某些实施例中,这可以包括确定期望的窗口出现或消失,确定图像出现或改变,确定图形组件出现或改变(例如,文本框、文本字段、表格等),其组合等。例如,这样的改变可以使用CV来确定。

[0085] 在某些实施例中,可以使用多个帧来进一步提高准确性。例如,如果在键入字符与在屏幕上出现字符之间存在变化的不同步(例如,从30ms变为42ms变为24ms等),则使用多个帧可以帮助标识键入的文本。然后,视觉变化的(多个)位置可以被隔离,并且算法在发生了变化的位置上被运行以识别字符。该算法可以使用OCR、针对某些字体的字符的布尔数组映射进行的像素区域比较等。在一些实施例中,字符识别仅在相对较小的区域上被运行,在该相对较小的区域中,所发生的变化被隔离和分析,而剩余区域被丢弃。这有助于确保算法可以在计算系统上实时运行,在该计算系统中,在整个屏幕(例如 3840×2160 像素分辨率)上运行OCR对于计算系统而言在计算上可能过于昂贵,以致于无法跟上在屏幕上出现字符的速度。然而,对于具有足够的处理能力的计算系统,整个屏幕可以在不解析出未首先发生变化的区域的情况下被分析。

[0086] 根据上文,在某些实施例中,视频帧比较计算并非是像素完美的,而是使用公差。如果颜色强度、亮度和/或其他因素落在特定公差内,那么颜色强度、亮度和/或其他因素可以被视为相同的。例如,如果像素值中的一者或多者的变化小于某个数字、某个百分比等,那么像素可以被视为相同的。红色、绿色、蓝色以及亮度的小于5、小于1%等的变化可以被视为相同的。在某些实施例中,这些变量中的一者或多者可以具有不同公差。例如,或许亮度变化可能需要大于或小于颜色变化以指示真实的像素变化。在某些实施例中,模糊图像匹配可以被执行,以标识相似性/差异。

[0087] 在一些实施例中,模糊图像匹配考虑了亮度、图像模板、边缘比较、二值化、缩减(downscale)和比特下降(bit reduction)、扩张、应用内核模糊、其组合等,以便更准确地标识匹配。将公差应用于RGB值的像素间RGB匹配可以被使用,因此不完全相同的接近值可以被标识为匹配。比特深度和/或色标可以被减小,并且像素间RGB或灰度匹配可以被应用。来自图像的边缘可以被检测和比较。二值化可以被应用于图像(例如二进制阈值、Otsu阈值、自适应阈值等),并且像素间匹配可以被应用于二进制图像上。图像的尺度(scale)可以被减小,并且像素间匹配可以被执行。图像的扩张可以被执行,并且然后像素间匹配可以被应用。关键点可以从图像中被提取(例如最大稳定极值区域(MSER)描述符),并且所提取的关键点可以使用特征匹配器(例如蛮力匹配、k最近邻(kNN)匹配等)比较。

[0088] 存在基于公差的计算可能有利的各种原因。例如,如果在捕获到帧之后对图像进行压缩,那么在计算中应涉及公差,这是由于有损压缩可以影响像素值。此外,可以(例如在经由仿真器启动虚拟计算系统并且仿真器对虚拟计算机屏幕内容进行压缩时)使用有损压缩在捕获之前对原始视觉源进行压缩。这可以是因为图像从远程机器(例如服务器)被广播

到本地计算系统而发生。

[0089] 一旦在618处标识出发生了变化和/或发生了其他图形变化的屏幕区域的字符,就将这些字符与与按键事件相对应的存储字符队列比较。如果找到匹配,则在620处提取匹配位置的屏幕坐标以及其他检测到的图形变化的屏幕坐标。在一些情况下,字符识别算法可能无法识别屏幕上的字符的实际含义。例如,OCR算法可以将屏幕上的字母“O”识别为数字“0”。在那种情况下,在一些实施例中,算法追踪屏幕上的插入符号的位置。这可以通过使用图像检测(例如CV)等将各种插入符号形状的图像图案与屏幕进行比较来确定。在一些实施例中,如果这样做,那么算法可以考虑插入符号闪烁。

[0090] 在某些实施例中,模糊匹配可以用于将OCR结果与队列中的字符进行比较。模糊匹配逻辑可以识别出字母“O”看起来与数字“0”类似,并且可以将这些字符标识为匹配。如果在队列中不存在其他类似字符,那么可以确认匹配。

[0091] 在某些实施例中,插入符号追踪被执行。对图像的(多个)改变的区域的分析可以被执行,以创建可以与插入符号相对应的候选的列表(例如候选呈现为细的竖直线或类似的事物)。可以执行验证以标识候选正随着时间而闪烁,并且然后可以标识真实的插入符号。可以执行进一步的验证,以核实插入符号出现在能够进行文本输入的图形元素(例如文本框、文字处理器文档、文本字段等)内。

[0092] 如果在屏幕上的其他位置并未发生任何变化、或其他变化与队列中除了缺失的字符以外的字符相匹配,那么算法然后可以推断出因为这是唯一未经标识的变化,所以其一定是缺失的字符。算法然后可以推断出所识别的字母“O”实际上为例如字符队列中的另外未经标识的“0”,并且在620提取匹配位置的屏幕坐标。这可以提高算法的准确性。

[0093] 在一些实施例中,当在预定时间过去(例如500ms、一秒等)之后在屏幕上找到字符或字符序列并且唯一地标识该字符或字符序列时,可以从队列中移除字符,基于预定大小(例如20个字符)的队列在队列末尾弹出字符等。为了移除落在时间窗口外部的字符,队列可以存储具有被键入的字符和时间戳的变量。算法可以将队列中的按键变量(潜在地开始于“先进(first in)”端)的时间戳与当前时间周期性地进行比较。如果在队列中找到晚于时间窗口的按键变量,那么变量可以被移除。在某些实施例中,一旦找到落在时间窗口内的按键变量,就可以假设队列中的所有其他变量都在时间窗口内,并且处理可以针对该迭代而停止。

[0094] 在620处提取具有最近键入的字符和/或标识的图形元素的屏幕区域的坐标之后,在622处,将坐标与正在运行的应用比较,并且基于所提取的坐标落入哪个元素来确定活动元素。坐标可能不与图形应用中的任何潜在活动元素相对应,或者,坐标与错误的活动元素相对应。输入的文本也可能不完整或错误,或者也可能没有发生期望的屏幕切换。因此,在624处通过检查对应机器人活动或工作流中的活动并且确保发生活动元素、输入的文本和/或其他期望的屏幕变化来执行验证。这可以包括但不限于确定机器人要执行的对应活动或一系列活动中的动作(例如,选择字段,在该字段中输入文本,单击按钮等),确定屏幕改变应当基于一个活动或一系列活动(例如,预期将打开新的应用窗口,预期该应用窗口将改变为新格式,预期该应用将呈现某些文本,等等)而发生,或任何其他合适的验证动作,而没有背离本发明的范围。验证可以由机器人本身执行,也可以由监测机器人的执行的其他应用或机器人执行,或者以这两种方式执行。如果在626处验证成功,则该过程在图6A的步骤612

中进行到下一屏幕截图。但是,如果验证失败(例如,机器人应当在发票的字段中输入“\$1, 234.50”,但是该字段显示在“公司名称”字段中,或者输入的值有误,因为缺少字符(例如,“\$1,34.50”)),则该过程进行到在图6A的步骤616采取补救动作时引发异常。

[0095] 在一些实施例中,屏幕记录、按键记录和/或其处理由RPA机器人执行。在某些实施例中,记录器应用将机器人活动记录为屏幕截图或视频以及一系列按键,并且将这一信息保存以供稍后处理,或将该信息传递给在计算系统或另一计算系统上运行的另一应用以进行实时或近实时处理。CV可以在视频和按键的处理之前,在视频和按键的处理之后被立即应用或稍后应用,以提供一组识别图形元素,包括其边界矩形。然后,如果在图形元素边界矩形与插入符号/文本的坐标之间找到交集,那么特定元素可以被认为当前为活动的或“受到关注的”(即,“活动元素”)。

[0096] 在一些情况下,屏幕的从一帧到下一帧的变化可以是显著的。例如,当机器人关闭窗口时,屏幕的绝大部分可以改变。因此,在一些实施例中,变化阈值被确定并且被应用于确定是否完全比较时间相邻的屏幕截图(例如超过2%的屏幕已改变,超过5%、10%、30%、50%、70%等)。当满足或超过该阈值时,帧比较过程可以被跳过,直到两个时间相邻的帧之间的差低于阈值为止。

[0097] 在一些实施例中,机器人可以在导致改变计算系统的屏幕的工作流步骤之后暂停,直到接收到确认。这样可以防止机器人在上一步失败时继续进行下一步,下一步和下一步。因此,错误的操作可以被迅速地标识并且停止。

[0098] 在一些实施例中,监测和机器人验证过程可以由试图控制计算系统的同一机器人执行。这为机器人提供了一种验证其自身操作的机制。在某些实施例中,在同一计算系统上操作或在远程接收视频和按键信息的不同计算系统(例如,服务器)上操作的另一机器人可以执行监测和验证。在一些实施例中,本地或远程操作的非机器人应用执行监测和验证。在某些实施例中,执行视频和按键分析的应用或机器人与执行验证的机器人或应用是不同的。

[0099] 图7是示出根据本发明的实施例的用于RPA的机器人执行的屏幕响应验证的过程700的架构图。屏幕记录710被执行,以产生帧N 712、N-1 714、N-2 716以及N-3 718。帧可以是整个屏幕、与运行的应用相关联的窗口等。在某些实施例中,用于多个运行的窗口化应用的帧集可以被存储和比较,或者这些帧可以从屏幕截图中被拉取。按键事件记录或机器人活动文本取得720还被执行,以产生包括与按键相关联的字符732的时间窗口化键字符队列730。在一些实施例中,当与由机器人尝试的字符相关联的按键超过预定使用年限时,这些字符从字符队列中被移除。在某些实施例中,在屏幕上找到字符之后,若字符是唯一的,那么从队列中移除这些字符。然而,如果字符不是唯一的(例如用户快速连续地按下“a a a a”),那么在一些实施例中可以从队列中移除“a”的最旧实例。

[0100] 核心逻辑740接收当前帧和紧邻的前一帧(在这种情况下为帧712、714)以及键字符队列730。核心逻辑740可以执行上文在一些实施例中相对于图6A和图6B所描述的逻辑中的一些或者所有逻辑。例如,核心逻辑740可以将帧712、714彼此进行比较,并确定其中已发生视觉变化的(多个)区域。例如,这可以通过比较相应像素的红色/绿色/蓝色(RGB)值是否相对于彼此超过阈值来确定。

[0101] 例如,核心逻辑740可以隔离视觉变化的位置,执行字符识别以识别该位置中的字

符,并且基于键字符队列730的内容确定字符是否对应于预期出现的字符。核心逻辑740还可以将所识别字符与键字符队列730中的字符732相匹配。当找到匹配时,匹配位置的屏幕坐标可以被提取并且被提供作为插入符号和/或键入区域750。

[0102] 除了使用字符识别之外或替代使用字符识别,在一些实施例中,图像识别(例如CV)可以用于检测新出现或消失的图像或图标。除了在区域差异分析阶段具有这种替换或补充逻辑外,过程可以另外与上文所描述的过程相同。例如,这对于确定机器人正在将图像或文本剪切和粘贴到文档或字段中的位置可以是有用的。

[0103] 在剪切和粘贴文本的情况下,当从剪贴板中清除文本时,单独字符并未被捕获作为按键事件。在这种情况下,可以捕获剪贴板的内容和用户正在按下CTRL+V的时间。然后,剪贴板的内容可以与OCR结果进行比较,类似于如何标识单独字符和字符序列。然而,算法并非审阅键入字符队列,而是将处理在发生CTRL+V之前被存储于剪贴板中的字符串。

[0104] 在一些实施例中,对于机器人按下导致屏幕改变的特定按键的情况,可以保存当前活动元素。例如,如果机器人按下Enter键,则可能导致表单提交和窗口关闭。在另一示例中,如果机器人在Web浏览器的URL输入字段中按下Enter键,则可能导致Web浏览器开始导航到网站。该算法可以识别出这种情况,并且等待一定时间,然后才能在这些情况下查看活动元素的位置,因为可能会发生重大屏幕改变。一旦屏幕变得相对静态(例如,仅屏幕的相对较小部分从一个屏幕截图到下一屏幕截图发生改变),则键入的文本检测和插入符号跟踪可以再次恢复以找到新的活动元素。因此,了解当机器人按下Enter键、退出键等时焦点集中在哪个元素上可能会有所帮助。此外,如果当前操作系统提供一种可靠的方式来本地获取当前前景窗口的窗口边界矩形(例如,在Windows®的user32.dll中可用的“GetForegroundWindow”),则前景窗口边界矩形可以用于限制将用于屏幕捕获和帧处理的区域。

[0105] 在一些实施例中,如果确定了聚焦或活动元素,则可以基于活动元素的性质来确定按下不会导致文本出现的按键的行为(例如,输入、转义、控制、控制和/或alt和一个或多个字符的组合等)。例如,当按下这样的“热键”(不可见键或键组合)时,可以对“热键”触发的动作进行分类。如果聚焦的元素标记为“密码”,并且机器人按下“Enter”,则可以推断出按下“Enter”代表“登录”动作。这可能比仅仅知道机器人已经按下“Enter”更具描述性。

[0106] 图8A图示了根据本发明的实施例的分别包括帧N-1和N的校友捐赠窗口的屏幕截图800、810的部分802、812。如在图8A中可看出,帧N-1中的所有字段都为空,但在帧N中,机器人已经在名字字段中键入了字母“E”。为了确定是否应当已经发生了这一变化,一些实施例的算法缩放屏幕截图并将其标准化为正方形。在这种情况下,屏幕截图800(参见图8B)和屏幕截图810(参见图8C)被标准化为64个正方形乘以48个正方形的网格,该网格的27×24部分在图8B至图8F中被示出。这些正方形或区域可以被表示为64×48二进制矩阵。

[0107] 然后,针对每个屏幕截图的行彼此进行比较,以查看其中所包含的任何像素是否已经改变,并且针对具有变化的每一行,矩阵的值可以被设置为“1”。这可以通过检查每一行中是否存在非零元素来完成。如在图8D中所看出,当算法检查矩阵的第4行(在本文中被指定为814)时,在其中检测到变化。如在图8E中所看出,算法然后进入第4行,并且在第7列中的被指定为816的正方形包括新输入的字母“E”的一部分。算法继续标识包括变化的正方形,并将相邻的正方形818拼接在一起,作为要提交的区域以使OCR在该部分上运行。参见图

8F。这在本文中被指示为“区域”。在一些实施例中，如果组成区域的一个或一些正方形不具有变化，那么这些正方形可以被设置成完全为一种颜色，例如以使OCR处理更快地运行。在某些实施例中，如果有足够的处理资源可用，则可以标识屏幕截图之间的差异区域，可以在差异区域上执行OCR以识别文本，并且可以将该文本与键字符队列比较以搜索匹配。

[0108] 在预期“E”出现在姓名字段中的情况下，处理和验证可以进行到下一帧。但是，如果字母“E”不是正确的输入（例如，预期应当使用“Ed”而不是“E”），“E”出现在错误的字段中，或者两者，则验证逻辑可以识别出这一点。然后，机器人可以尝试输入以删除不正确的文本，然后将正确的文本输入相同的字段或不同的字段，或者采取其他补救动作。如果补救动作失败，则机器人可以暂停或停止其工作流的执行，将错误通知用户或开发人员，等等。

[0109] 有时，显示分辨率可以发生变化，可以挂接其他监测器，等等。一些实施例检测并且适应这些变化，从而使得检测和验证过程保持准确。图9是示出根据本发明的实施例的用于检查分辨率变化、检查所连接的显示器的范围的变化、以及设置帧捕获逻辑以适应变化的过程900的流程图。该过程开始于在910处检查计算系统的一个或多个所连接的显示器，并且在920处将（多个）所连接的显示器与先前的（多个）所连接的显示器进行比较。这可以涉及检查是否存在不同显示设备、检查显示设备的分辨率是否已改变等。在一些实施例中，“所连接的”显示器可以是与计算系统集成的显示器（例如针对智能电话、膝上型计算机等通常如此）。

[0110] 如果在930处（多个）所连接的显示设备和/或分辨率已经改变，那么在940处针对每个所连接的显示器获得分辨率和尺度。在950，针对将被捕获的屏幕截图的屏幕截图区被设置为乘以尺度并且与期望大小的倍数（例如8、16等）对准的完整显示尺寸。倍数可以有助于将屏幕截图划分成正方形，如本文中稍后进一步详细讨论的。然后，在960处，设置帧捕获逻辑（例如，重新启动，重新初始化，提供新的显示设置，等等）。

[0111] 图10是图示了根据本发明的实施例的用于录像机的过程1000的流程图。该过程开始于在1010截取屏幕截图。在一些实施例中，这可以使用针对 Windows® 的图形设备接口 (GDI) CopyFromScreen() 指令在C#中完成。然后在1020，屏幕截图作为帧被添加至缓冲器。例如，这可以通过将屏幕截图作为C#中的位图对象添加至缓冲器来完成。如果在1030处该过程仍在运行（例如该过程尚未因关闭应用、屏幕分辨率变化等而停止），那么可以针对下一个屏幕截图重复屏幕截图捕获。应注意，虽然提供了C#示例，但针对本文中所公开的过程1000和其他过程，可以在不脱离本发明的范围的情况下使用任何合适的编程语言、操作系统、API以及功能。

[0112] 一些实施例执行插入符号追踪，以更准确地标识机器人正在关注屏幕的哪个元素。例如，如果插入符号出现在正出现文本的图形元素中，那么键字符队列中的新添加的文本有可能是正出现在该图形元素中的内容。图11是图示了根据本发明的实施例的用于执行插入符号追踪的过程1100的流程图。通常，插入符号将在用户最近点击的位置处或附近出现并且开始闪烁。因此，一些实施例存储最终鼠标点击的坐标，并且在此位置附近搜索插入符号。这可以减少为了定位插入符号而处理的屏幕截图的量，并可以进一步提高准确性。在一些实施例中，使用鼠标点击或单个最近点击位置的历史缓冲器。在某些实施例中，例如，如果机器人“按下”tab键，那么系统可以假设插入符号可能已经移动到屏幕上的下一个图形元素，并且可以将搜索的重点重新放到该位置（若已知）上，否则搜索整个屏幕截图。

[0113] 在1105,针对屏幕截图中的发生了变化的每个区域计算像素变化并且将这些区域投影到二进制矩阵。二进制矩阵为对区域的像素是否已经改变的表示,并且可以针对在屏幕截图之间不具有变化的像素包括“0”,而针对改变的像素包括“1”。在一些实施例中,“区域”是发生了变化的正方形,其可以包括来自屏幕截图的多个正方形。然而,可以在不脱离本发明的范围的情况下使用任何其他合适的形状(例如矩形、六边形等)。在一些实施例中,根据计算系统的处理能力,支持固定数量的区域进行分析。例如,一些实施例支持两个区域、三个区域、十个区域等的提取和OCR。一些实施例可以寻找插入符号和键入或粘贴文本两者。当在当前屏幕截图与前一个屏幕截图之间找到超过数量为L个变化区域时,可以处理找到的前L个区域,或可以完全忽略屏幕截图。这可以有助于忽略用户已启动不同窗口或另外改变了屏幕的足够部分的屏幕,以使得在捕获到下一个屏幕截图之前,OCR可能无法及时完成。

[0114] 针对每个二进制矩阵,在1110闪烁的插入符号区域候选被提取,并且在1115二进制矩阵成员被结合。如本文中所使用,“成员”是存在于二进制矩阵中的所连接的形状,诸如表示字母、光标等的形状。例如,矩阵成员的结合可以使用“连接分量(connected components)”算法来执行,其中分量是8向连接的。连接分量是每个像素都被连接至所有其他像素的一组像素。

[0115] 在1120从矩阵成员结合结果中提取形状,并且在1125验证形状。例如,该形状通常应该是完美的矩形,例如其可以包括线。在1130,存储经验证的形状候选并且将其与确认队列(confirmation queue)进行比较。可以将插入符号候选的位置、大小和形状连同时间戳一起存储。插入符号闪烁的频率应该在公差范围内(例如5%)。由于插入符号闪烁,因此候选应被存储以进行分析,以便查看这些候选是否与插入符号的预期属性(即,位置、大小和频率)相匹配。当在多个屏幕截图(例如20个)中进行比较时,这可以有助于确定插入符号候选是否以特定频率闪烁。如果用户在新字段上点击鼠标、按下Tab键等之后,插入符号再次出现在其他位置,那么该信息也可以有助于标识插入符号。自然地,在一些实施例中,在过程开始时,确认队列为空。

[0116] 如果在1135基于插入符号候选的出现/消失、大小和位置来确认给定插入符号候选正在闪烁,那么在1140针对闪烁的插入符号产生插入符号追踪数据。这可以包括插入符号在屏幕上的位置、该插入符号所驻留的图形元素(即,活动元素)等。例如,然后在1145将经验证的候选区域和对应成员二进制矩阵数据保存到确认队列,以供稍后验证。在一些实施例中,可以针对每个新屏幕截图重复图11的过程。

[0117] 图12A至图12D是示出根据本发明的实施例的用于执行用于RPA的机器人执行的屏幕响应验证的过程1200的流程图。在过程1200之前,可以对分辨率变化进行检查,并且可以设置插入符号追踪视频逻辑以适应变化。例如,参见图9。在某些实施例中,过程1200可以与录像机同时运行。例如,参见图10。过程1200是使用C#和 Windows® 操作系统的示例。然而,可以在不脱离本发明的范围的情况下使用任何合适的编程语言、操作系统、相关联的API、格式以及功能。

[0118] 该过程开始于在1202处使用格式Format16bppRgb555对视频项N(例如屏幕截图、屏幕的一部分、应用窗口等)执行LockBit,以便为N创建BitmapData对象。LockBit锁定位图的矩形部分,并且提供可以用于以指定格式读取或写入像素数据的临时缓冲器。

BitmapData存储位图的属性。

[0119] 然后在1204处, BitmapData N和BitmapData N-1 (即, 前一项的BitmapData对象) 被划分成高度为16个像素的水平行。然而, 可以在不脱离本发明的范围的情况下使用用于过程1200的该步骤和其他步骤的任何期望高度 (例如8个像素、32个像素等)。在1206, 针对在相同竖直位置中 (即, 在相同“行”中——参见图8A至图8F) 的BitmapData N和BitmapData N-1的每个水平行执行MPCMP指令, MPCMP指令执行字节数组的快速比较。MEMCMP提供了对行是否相同的指示。

[0120] 如果在1208处BitmapData N与BitmapData N-1之间的所有行都是相同的 (即, 至少一个对应行中存在差异), 并且在1210处基于机器人正在执行的当前活动而预期没有差异, 则该过程进行到图12D的步骤1252并且转到下一屏幕截图。然而, 如果在1210处预期存在差异, 则该过程进行到图12D的步骤1256以尝试纠正错误。如果在1208处的BitmapData N与BitmapData N-1之间的所有行都不相同 (即, 至少一个对应行中存在差异), 并且在1212中预期没有差异, 则该过程进行到图12D的步骤1256以尝试纠正错误。但是, 应当注意, 在一些实施例中, 如果BitmapData N与BitmapData N-1之间的比较结果不是预期的 (即, 当预期没有变化时发生了意外的变化, 或者在预期有变化时没有发生变化), 则过程步骤1214-1250或其子集仍然可以被执行。

[0121] 如果在1208处BitmapData N与BitmapData N-1之间的所有行都不相同 (即, 至少一个对应行中存在差异), 并且在1212处预期存在差异, 则在1214处提取MEMCMP结果不等于0的同一行中的BitmapData N和BitmapData N-1的水平行, 然后在1216处将所提取的水平行划分为大小 16×16 个像素。然后在1214, 提取MEMCMP结果不等于0的相同行中的BitmapData N和BitmapData N-1的水平行, 并且在1216, 将所提取的水平行划分成大小为 16×16 的像素。例如, 参见图8C。然后针对BitmapData N和BitmapData N-1的每个 16×16 像素块, 在1218提取这些像素之间存在差异的块。例如, 参见图8F。这可以使用循环Intel **Intrinsics®** 指令或一些其他合适的能力的长XOR函数的组合来执行。

[0122] 根据上文, 在一些实施例中, 将可以被处理的区域的数量限制为预定数量L。在某些实施例中, 可以限制在每个区域中可以包括的块的数量。例如, 可以应用4个正方形、9个正方形、16个正方形等的限制, 以确保OCR可以在获得下一个屏幕截图之前在每个区域上运行。这可以是“优化阈值”, 其可以包括对已经改变的区域的数量限制、对在给定改变区域中所包含的正方形的数量的限制或两者。

[0123] 在1220, 将每个区域中的所提取的 16×16 像素块的总计数、区域的数量或两者与优化阈值进行比较。如果在1220满足优化阈值, 那么过程继续进行至图12D的步骤1252并进行下一个屏幕捕获。在一些实施例中, 如果在步骤1220并未超过阈值, 那么在1222使用连接分量算法结合邻近 16×16 像素块, 在一些实施例中, 该算法可以是8向连接的连接分量算法。这确定了哪些块是邻居。

[0124] 一旦确定了所连接的邻居, 就在1224确定每组邻近块的边界矩形, 从而形成区域。这可以由极值算法来确定, 在该极值算法中包括具有最高x值和最低x值的块 (即, (多个) 最左侧块和最右侧块) 以及具有最高y值和最低y值的块 (即, (多个) 最高块和最低块)。此示例可以从图8F中看出。

[0125] 在1226针对区域的每个边界矩形, 计算像素变化并将其投影到二进制矩阵。例如,

在图14A中示出了在已经组合成 32×32 区域的四个 16×16 块中所包括的字母“E”的示例二进制矩阵1400。

[0126] 在大多数情况下,区域将大于其中所包含的(多个)成员(例如字母、插入符号、改变了像素的其他形状等)。为了提高OCR算法的速度,在1228针对每个二进制矩阵,(例如使用连接分量算法)确定每个区域中所包括的(多个)成员,并且针对每个成员剪裁二进制矩阵。这将针对每个区域中的每个成员产生剪裁后的矩阵。在图14B中示出了从二进制矩阵1400中产生的字母“E”的示例剪裁成员矩阵1410。在一些实施例中,剪裁还可以使用极值算法来执行。

[0127] 然后在1230,从成员矩阵中提取闪烁的插入符号区域候选。例如,在一些实施例中,候选可以具有矩形形状,其潜在地包括具有单个像素的宽度的竖直线。然后在1232,将所提取的闪烁的插入符号区域候选和对应成员矩阵数据与确认队列进行比较,从而潜在地分析闪烁的大小、位置和频率。如果在1234发生闪烁,那么在1236针对闪烁的插入符号产生插入符号追踪数据。然后在1238,将闪烁的插入符号区域及其对应成员矩阵数据保存到确认队列。在一些实施例中,过程1200的该部分可以与图11的过程1100相同或类似。

[0128] 在一些实施例中,成员二进制矩阵仅指示给定像素是否已从屏幕捕获N-1改变为屏幕捕获N。因此,在1240针对已经改变的每个像素,从BitmapData N中获取像素数据。然后在1242生成成员矩形,并且这些成员矩形被准备进行OCR。这可以包括:针对每个改变的像素填充像素数据;消除插入符号像素;处理背景(例如将未经改变的像素设置为空值或高度收缩的值)等。在消除了插入符号像素的情况下,可以假设此时利用二进制矩阵成员的特定位置、形状和集合来检测插入符号本身。可以出于插入符号追踪目的来存储该信息。然后在1244针对所准备的成员矩形像素数据执行OCR

[0129] 在1246处,确定和验证RPA工作流中的当前机器人活动的预期变化。预期变化可以通过分析键字符队列中的按键事件、应当基于RPA工作流中的机器人活动而显示的变化等来确定。

[0130] 如果在1248处验证成功并且使用键字符队列,则可以在1250处从键字符队列删除与OCR区域相匹配的键字符队列项目(例如,按键事件)。在存在多个字符实例的情况下,例如,可以删除键字符队列中该字符的最旧实例。然后,在1252处对BitmapData N-1执行UnlockBits,以从系统内存中解锁该位图,然后在1254处将BitmapData N移动到位置N-1。然后,过程1200可以针对下一捕获项目返回到开始。

[0131] 但是,如果在1248处验证失败(例如,未发生预期改变或仅部分发生改变),则在1256处尝试采取补救动作。例如,可以向机器人通知工作流的哪个(些)步骤失败,采取某种纠正措施,然后再次尝试该步骤。这可以包括但不限于机器人在驱动级别检查机器人尝试与之交互的针对应用的窗口是否为活动窗口,检查鼠标指针或插入符号相对于相关图形元素的位置,检查正确的图形元素是否为活动元素,其任何组合等。机器人可以将正确的窗口设置为活动的聚焦的窗口,将焦点设置为期望的活动元素,移动鼠标并且单击,等等。然后再次尝试该步骤。在采取补救动作的同时,在一些实施例中,帧比较过程可以被暂停。如果在1258处补救动作成功,则该过程可以进行到步骤1252。

[0132] 但是,如果在1258处补救操作失败,则可以在1260处执行错误逻辑。错误逻辑可以包括但不限于引发异常,停止执行机器人并且在屏幕上显示消息或向人类发送消息(例如,

经由文本或电子邮件),以指示机器人未能完成期望动作。在某些实施例中,可以将失败的动作提供给用户,以便他或她可以尝试对问题进行故障排除,或者用户可以报告将由RPA开发人员解决的问题。

[0133] 图13A至图13C是示出根据本发明的实施例的用于执行粘贴的文本跟踪和验证的过程1300的流程图。该过程开始于在1305处可选地执行按键记录并且在1310处执行屏幕捕获(例如,视频记录,捕获屏幕的全部或一部分的屏幕截图,等等)以分别确定被按下的键和屏幕上发生图形变化的位置。然后,在1315处解释机器人动作。机器人动作可以包括但不限于鼠标单击、按下CTRL+V,右键单击以及从菜单中选择粘贴,单击主页按钮并且粘贴到应用中,等等。如果机器人单击某个位置并且快速从剪贴板粘贴,则可能会丢失插入符号,并且可能需要以其他方式重新构造机器人的动作。如果在1320处没有出现来自剪贴板的粘贴,则在1325处执行键入的文本检测和插入符号跟踪逻辑,在一些实施例中,这有可能沿着图12A至图12D的过程1200的线。

[0134] 然而,如果在1320剪贴板包括来自机器人的最近粘贴的数据(例如在最后500ms、最后一秒等内粘贴),那么在1330计算帧N与N-1之间的像素差。然后在1335提取帧N与N-1之间发生变化的所预测的相同位置的区域。在1340针对每个区域,计算像素变化并且将变化投影到二进制矩阵。

[0135] 在1345,例如,针对每个二进制矩阵,使用连接分量算法确定成员,并且针对每个成员确定成员矩阵。在1350,针对成员矩阵中的每个改变的像素提取像素数据,并且生成成员矩形。在1355,每个成员矩形被准备进行OCR,并且在1360,在每个所准备的成员矩形上运行OCR。在1365,使用由剪贴板监测器在1370提供的剪贴板内容针对每个OCR结果执行OCR结果与剪贴板内容的模糊匹配比较。在一些实施例中,使用Clipboard.GetText()从System.Windows.Forms.dll中获得剪贴板文本内容。如果在1375找到匹配,那么产生粘贴文本的坐标(例如以矩形形式),并且在1380清除剪贴板,并且过程针对下一帧返回到步骤1305和1310。

[0136] 但是,如果在1375处未找到匹配(例如,预期的粘贴内容未出现或仅部分出现),则在1385处尝试采取补救动作。例如,可以向机器人通知工作流的哪个(些)步骤失败,采取某种纠正措施,然后再次尝试该步骤。这可以包括但不限于机器人在驱动级别检查机器人尝试与之交互的针对应用的窗口是否为活动窗口,检查鼠标指针或插入符号相对于相关图形元素的位置,检查正确的图形元素是否为活动元素,其任何组合等。机器人可以将正确的窗口设置为活动的聚焦的窗口,将焦点设置为期望的活动元素,移动鼠标并且单击,等等。然后再次尝试该步骤。在采取补救动作的同时,在一些实施例中,帧比较过程可以被暂停。如果在1390处补救动作成功,则该过程可以进行到步骤1380。

[0137] 但是,如果在1390处补救动作失败,则可以在1395处执行错误逻辑。错误逻辑可以包括但不限于引发异常,停止执行机器人并且在屏幕上显示消息或向人类发送消息(例如,经由文本或电子邮件),以指示机器人未能完成期望动作。在某些实施例中,可以将失败的动作提供给用户,以便他或她可以尝试对问题进行故障排除,或者用户可以报告将由RPA开发人员解决的问题。

[0138] 图15是示出根据本发明的实施例的用于使用CV和机器人活动的预期结果来确定活动元素并且验证机器人动作的过程1500的流程图。该过程开始于在1510确定帧是否相较

于前一帧改变了超过预定阈值的量。这可以包括检查是否超过特定部分的帧(例如超过2%)已经改变;是否超过预定数量的像素(例如超过200个)已经改变;在准许文本输入的帧中的图形元素的位置外部是否发生了变化等。

[0139] 如果在1520超过了阈值,那么屏幕上的至少一些图形元素有可能也已经改变。在1530执行CV预处理以标识可以被存储在存储器中的图形元素类型和位置。在一些实施例中,当执行CV预处理时,可以暂停帧比较过程。

[0140] 在1520并未超过阈值的情况下或在1530完成了CV预处理之后,在1540确定在前一帧与当前帧之间添加的键字符队列中的新出现的元素。如果在键字符队列中存在新出现的元素,那么可以假设这些元素出现在屏幕上的合适的图形元素中。然后在1550确定当前帧中屏幕改变的(多个)位置,并且在1560尝试将变化与图形元素的位置相匹配。如果仅在图形元素中的一者中发生变化,那么在1570将匹配的图形元素设置为活动元素。但是,如果在多个图形元素中发生改变,或者在图形元素中未找到任何改变,则在1580处执行补救操作(例如,从不应当显示内容的字段删除内容,并且尝试将内容插入正确的字段在)。然后在1590处取得下一帧,并且重复该过程。

[0141] 图16是示出根据本发明的实施例的被配置为执行用于RPA的机器人执行的屏幕响应验证的系统1600的架构图。系统1600包括用户计算系统,诸如台式计算机1602、平板电脑1604和智能电话1606。然而,可以在不脱离本发明的范围的情况下使用任何期望的计算系统,包括但不限于智能手表、膝上型计算机、物联网(IoT)设备、车辆计算系统等。

[0142] 每个计算系统1602、1604、1606上运行有数字过程1610,该过程记录屏幕截图、击键、正在运行的应用、应用视觉元素、视觉元素位置、应用位置、机器人工作流和正在执行的当前步骤等有关其自身或独立机器人的操作的信息。实际上,可以在不脱离本发明的范围的情况下记录与屏幕图形、用户输入、显示元素等相关的任何期望信息。在某些实施例中,最初仅捕获视频和击键记录,并且随后使用CV来确定其他信息。然而,附加信息可以有助于关注并改进CV过程。在不脱离本发明的范围的情况下,数字过程1610可以是经由RPA设计器应用生成的机器人、操作系统的一部分、个人计算机(PC)或智能电话的可下载应用生成的机器人,或者可以是任何其他软件和/或硬件。实际上,在一些实施例中,数字过程1610中的一个或多个数字过程的逻辑部分地或完全地经由物理硬件被实现。

[0143] 数字过程1610经由网络1620(例如局域网(LAN)、移动通信网络、卫星通信网络、互联网、其任何组合等)向服务器1630发送所记录的屏幕截图、击键、运行的应用、应用元素和位置、其任何组合等。在一些实施例中,服务器1630可以运行指挥器应用,并且数据可以作为心跳消息的一部分周期性地被发送。服务器1630将来自数字过程1610的信息存储在数据库1640中。

[0144] 服务器1630运行用于服务器1630从其接收数据的计算系统(即,计算系统1602、1604、1606)的屏幕响应验证逻辑的实例1632。服务器可以分析来自实例1632的结果,以确定机器人在与它们的计算系统交互时正在做什么。可以映射工作流中的步骤,并且使其与视频和按键按下时间同步,以确保实例1632可以确定机器人正在尝试执行的操作。关于验证是成功还是失败、失败的性质(如果发生了)、机器人所处的步骤等的信息可以被发送回相应用户计算系统1602、1604、1606,以验证由数字过程1610监测的机器人的执行。在一些实施例中,服务器1630的验证可以实时或接近实时地发生,因此当RPA机器人在相应计算系

统上执行时,过程1610可以执行验证。

[0145] 图17是示出根据本发明的实施例的用于执行用于RPA的机器人执行的屏幕响应验证的过程1700的流程图。该过程开始于在1705处应用CV算法以确定用户界面中的图形元素和相关联的边界矩形。在一些实施例中,所确定的图形元素可以包括用于RPA活动的目标图形元素。在1710处,确定当前屏幕截图帧与前一屏幕截图帧之间是否存在差异。当前屏幕截图帧和前一屏幕截图帧可以包括整个用户界面或用户界面的一部分。当在1715处当前屏幕截图帧与前一屏幕截图帧之间存在差异并且在1720处基于RPA机器人的工作流中的活动而预期当前屏幕截图帧与前一屏幕截图帧之间存在差异时,在1725处基于RPA机器人的工作流的活动对照一个或多个预期屏幕变化验证当前屏幕截图帧与前一屏幕截图帧之间的一个或多个改变。在一些实施例中,验证包括在当前屏幕截图帧与前一屏幕截图帧不同的一个或多个位置上执行OCR,并且将由OCR识别的字符与键字符队列中的字符匹配。

[0146] 当在1730处验证成功时,该过程进行到下一帧(即,下一帧为当前屏幕截图帧,当前屏幕截图为前一屏幕截图帧),并且进入工作流中的下一活动或受同一活动影响的下一改变。然而,当验证失败时,在1740处引发异常或采取补救动作。如果在1740处采取补救动作并且该补救动作成功,则该过程可以进行到步骤1740。

[0147] 返回到步骤1710,如果在1715处没有出现差异,并且在1745处预期没有差异,则该过程进行到步骤1735。但是,如果在1745处预期存在差异,则该过程进行到步骤1740。

[0148] 在一些实施例中,补救动作包括RPA机器人在驱动级别检查机器人尝试与之交互的针对应用的窗口是否为活动窗口,检查鼠标指针或插入符号相对于预期发生改变的图形元素的位置,检查正确的图形元素是活动元素,或这些的任何组合。在某些实施例中,补救动作包括RPA机器人将正确的窗口设置为活动的聚焦的窗口,将焦点设置为活动元素,移动鼠标并且引起鼠标点击事件,或这些的任何组合。在一些实施例中,异常的引发包括:停止RPA机器人的操作,在屏幕上显示消息,发送指示RPA机器人未能实现一个或多个预期屏幕改变的消息,列出一个或多个失败的动作,或这些的组合。

[0149] 在一些实施例中,在1750处生成包括由RPA机器人创建的在时间窗口期间发生的按键事件的键字符队列。在某些实施例中,在1725处的验证期间,从键字符队列删除匹配字符。在一些实施例中,键字符队列是包括按键事件的字符以及在时间窗口期间针对按键事件中的每个按键事件发生按键事件的时间的FIFO队列。

[0150] 在一些实施例中,验证包括确定当前屏幕截图帧与前一屏幕截图帧不同的一个或多个区域,在确定的一个或多个区域中提取一个或多个连接成员,并且对所提取的一个或多个连接成员执行OCR,以产生一个或多个识别字符、一个或多个识别字符序列或两者、以及相应的位置。验证可以包括将一个或多个识别字符、一个或多个识别字符序列或两者与键字符队列或者基于活动而预期出现的一个或多个字符、一个或多个字符序列或两者比较。在某些实施例中,基于RPA机器人的工作流中的活动确定目标图形元素和要被输入到目标图形元素中的内容,并且验证包括验证当前屏幕截图帧与前一屏幕截图帧之间的一个或多个改变是否与目标图形元素的位置相匹配、以及一个或多个改变是否与要被输入到目标图形元素中的内容相匹配。在一些实施例中,活动包括将文本录入图形元素中,将插入符号移动到图形元素,将内容粘贴到图形元素中,或这些的组合。在某些实施例中,基于RPA机器人的工作流的活动的一个或多个预期屏幕改变包括表单的提交、新应用窗口的打开或当前

应用窗口的改变。

[0151] 根据本发明的实施例,图6A、图6B、图7、图9至图13B、图15以及图17中执行的过程步骤可以由计算机程序执行,该计算机程序编码有指令,该指令用于使(多个)处理器执行6A、图6B、图7、图9至图13B、图15以及图17中所描述的(多个)过程的至少一部分。计算机程序可以被实施在非瞬态计算机可读介质上。计算机可读介质可以是但不限于硬盘驱动器、闪存设备、RAM、磁带和/或用于存储数据的任何其他此类介质或介质的组合。计算机程序可以包括用于控制计算系统的(多个)处理器(例如图5的计算系统500的(多个)处理器510)以实现6A、图6B、图7、图9至图13B、图15以及图17中所描述的过程步骤中的全部或一部分的编码指令,这些编码指令还可以被存储在计算机可读介质上。

[0152] 计算机程序可以被实现在硬件、软件或混合实现中。计算机程序可以由彼此可操作地通信并且被设计成传递信息或指令以进行显示的模块组成。计算机程序可以被配置为在通用计算机、ASIC或任何其他合适的设备上操作。

[0153] 将容易地理解,如本文中的诸图中大体上描述和图示的,本发明的各种实施例的组件可以以多种不同配置来布置和设计。因此,如附图中所表示,对本发明的实施例的详细描述并不旨在限制所要求保护的本发明的范围,而是仅仅表示本发明的所选择的实施例。

[0154] 贯穿本说明书描述的本发明的特征、结构或特性可以在一个或多个实施例中以任何合适的方式组合。例如,贯穿本说明书,对“某些实施例”、“一些实施例”或类似语言的参考是指结合实施例所描述的特定特征、结构或特性被包括在本发明的至少一个实施例中。因此,贯穿本说明书,短语“在某些实施例中”、“在一些实施例中”、“在其他实施例中”或类似语言的出现并不一定全都是指同一组实施例,并且所描述的特征、结构或特性可以在一个或多个实施例中以任何合适的方式组合。

[0155] 应注意,贯穿本说明书,对特征、优点或类似语言的参考并不意味着可以用本发明实现的所有特征和优点都应该在或都在本发明的任何单个实施例中。相反,涉及特征和优点的语言被理解是指结合实施例所描述的特定特征、优点或特性被包括在本发明的至少一个实施例中。因此,贯穿本说明书,对特征和优点以及类似语言的讨论可以但并不一定是指相同实施例。

[0156] 此外,本发明的所描述的特征、结构和特性可以在一个或多个实施例中以任何合适的方式组合。相关领域的技术人员应认识到,可以在不具有特定实施例的特定特征或优点中的一者或多者的情况下实践本发明。在其他情况下,可以不存在于本发明的所有实施例中的附加特征和优点在某些实施例中可能被识别。

[0157] 本领域的普通技术人员应容易地理解,可以利用不同顺序的步骤和/或利用与所公开的配置不同的配置的硬件元件来实践如上文所描述的本发明。因此,尽管已经基于这些优选实施例描述了本发明,但对于本领域的技术人员而言将明显的是,某些修改、变型和备选构造将是明显的,同时仍在本发明的精神和范围内。因此,为了确定本发明的界限和边界,应当参考所附权利要求书。

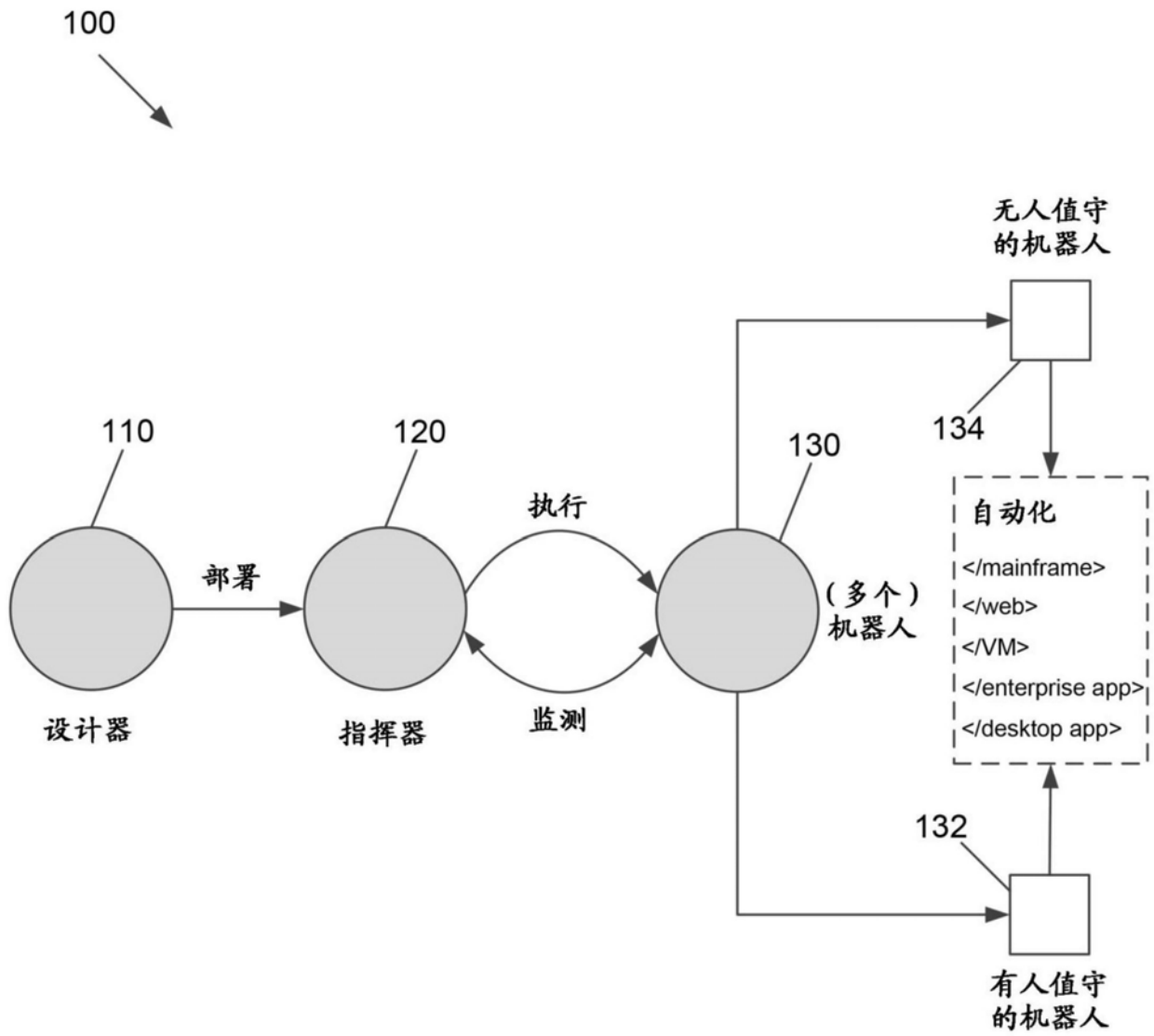


图1

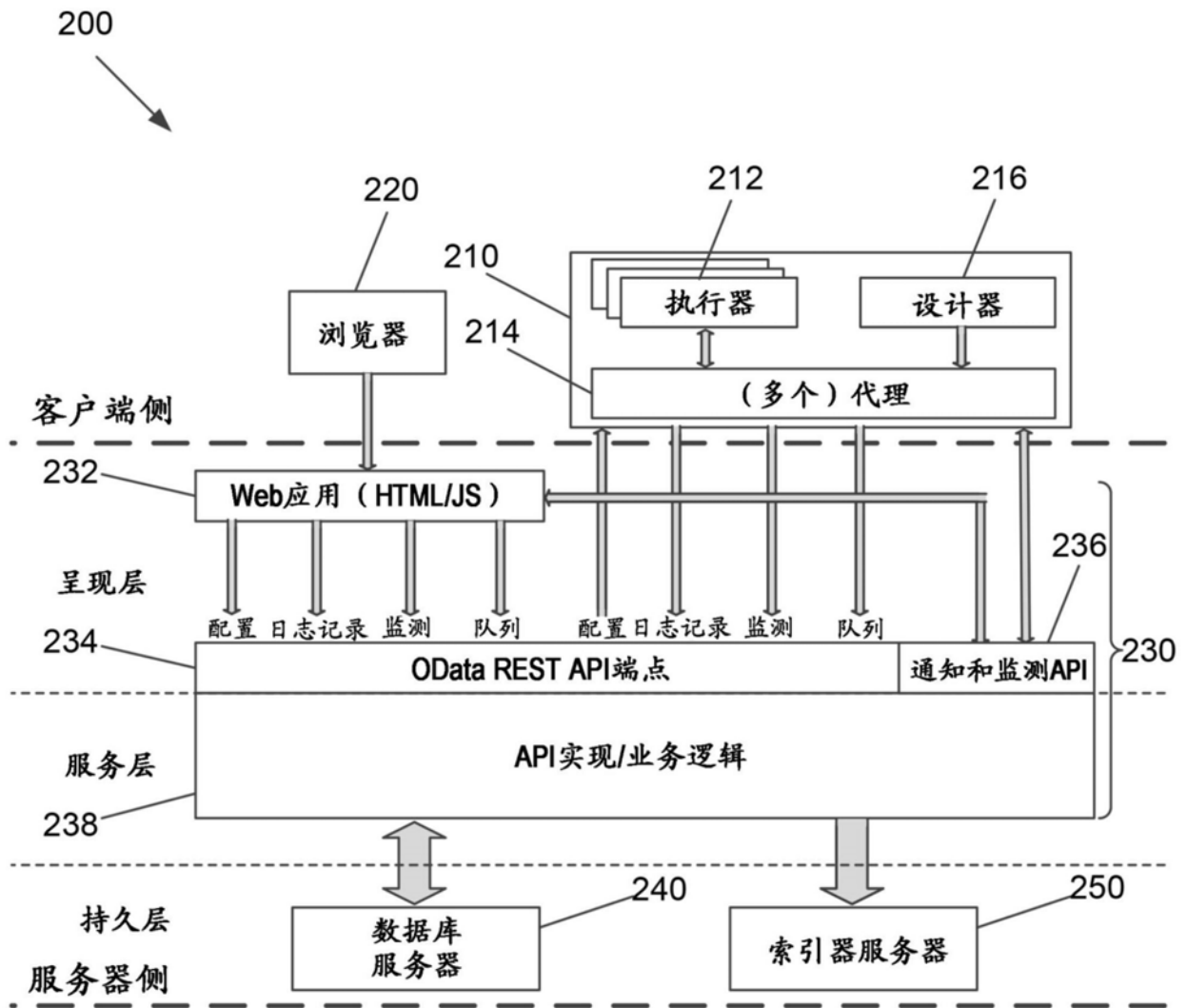


图2

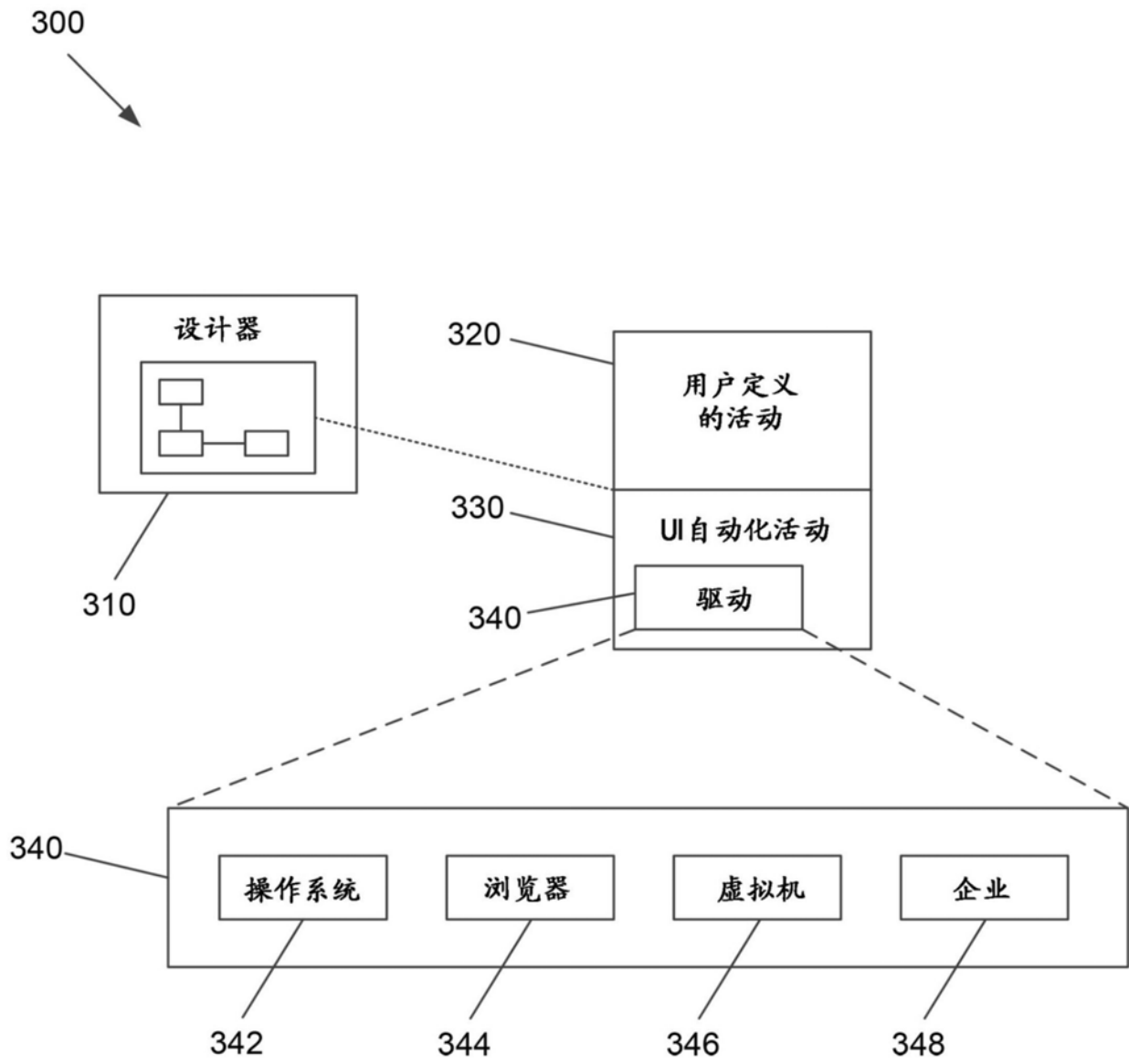


图3

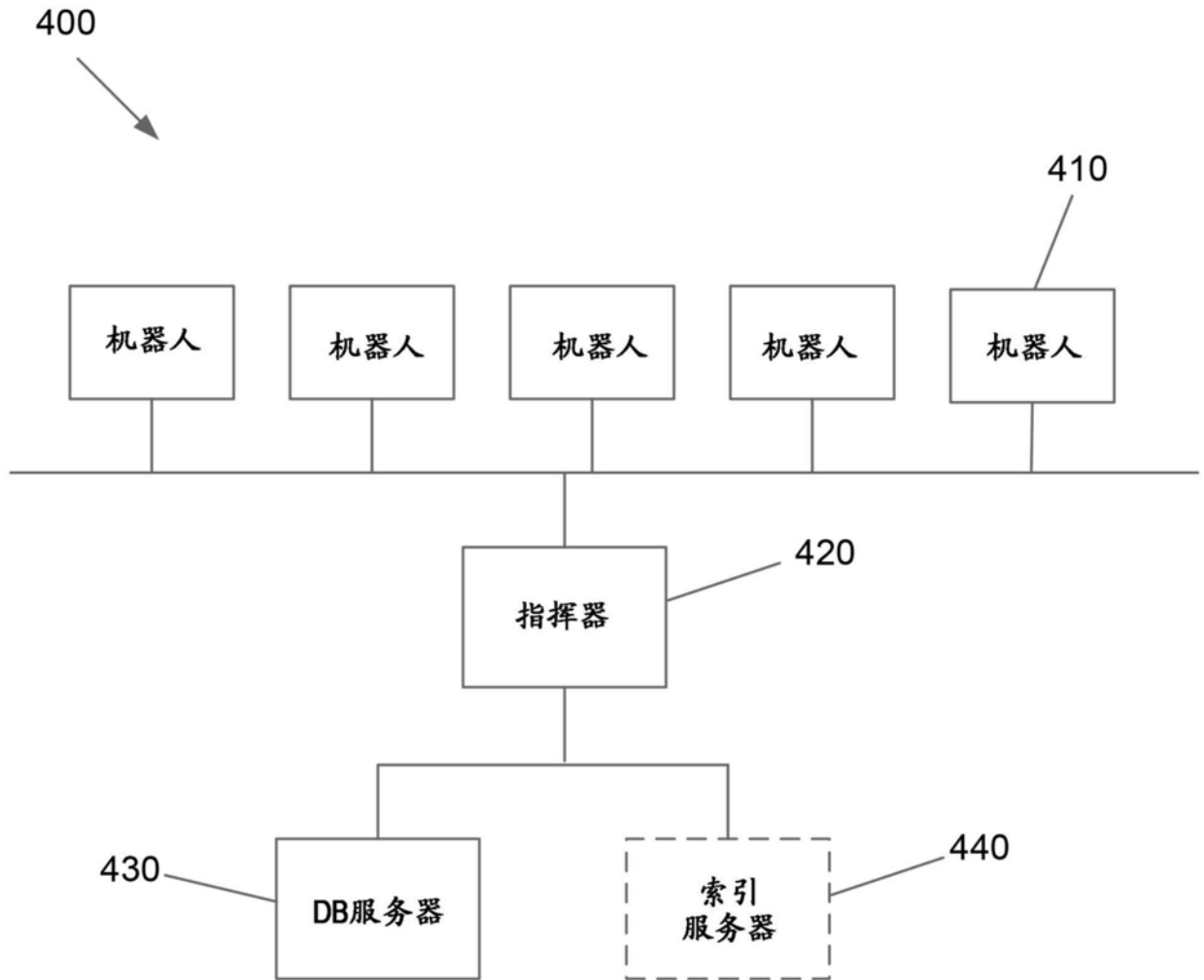


图4

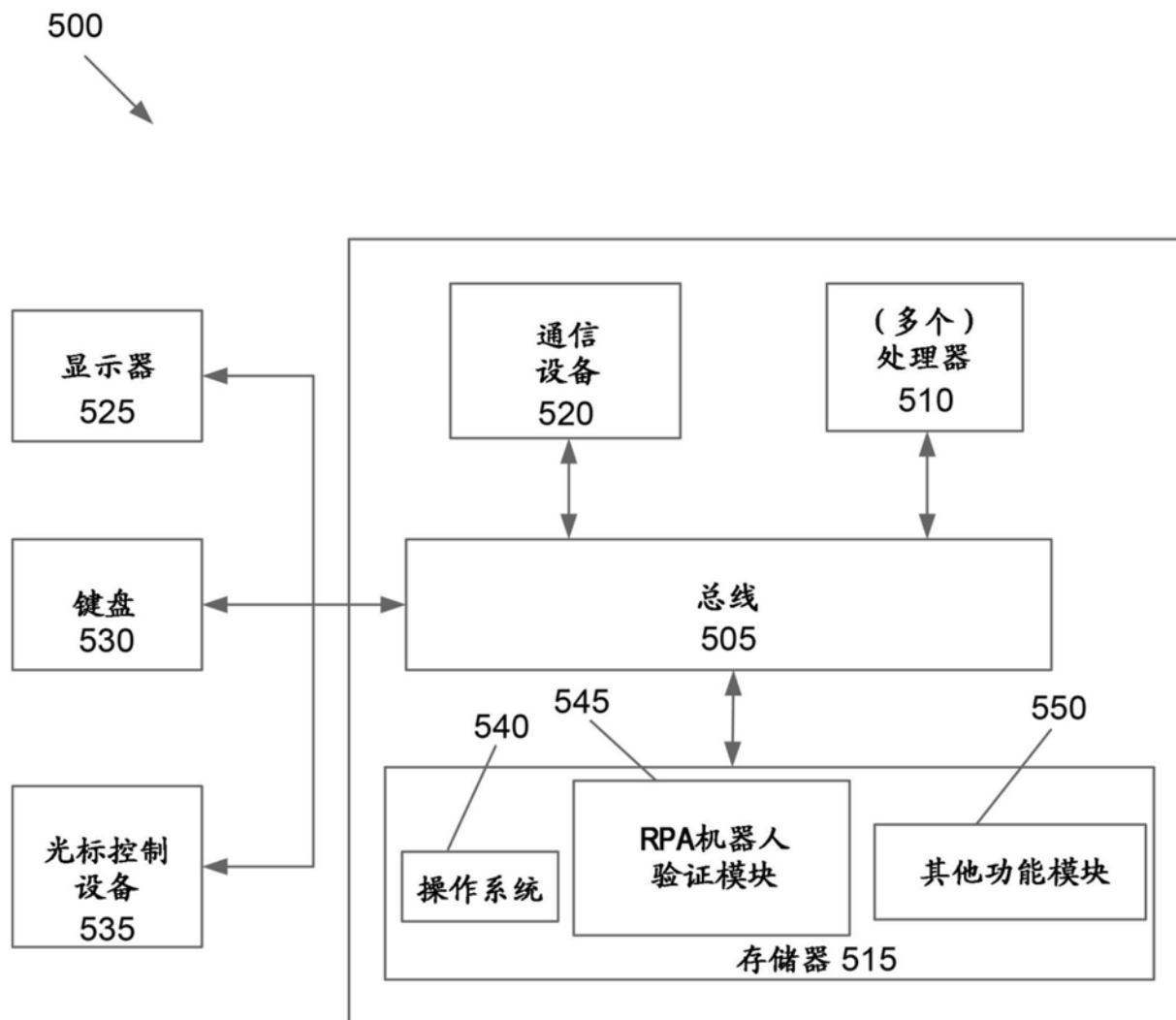


图5

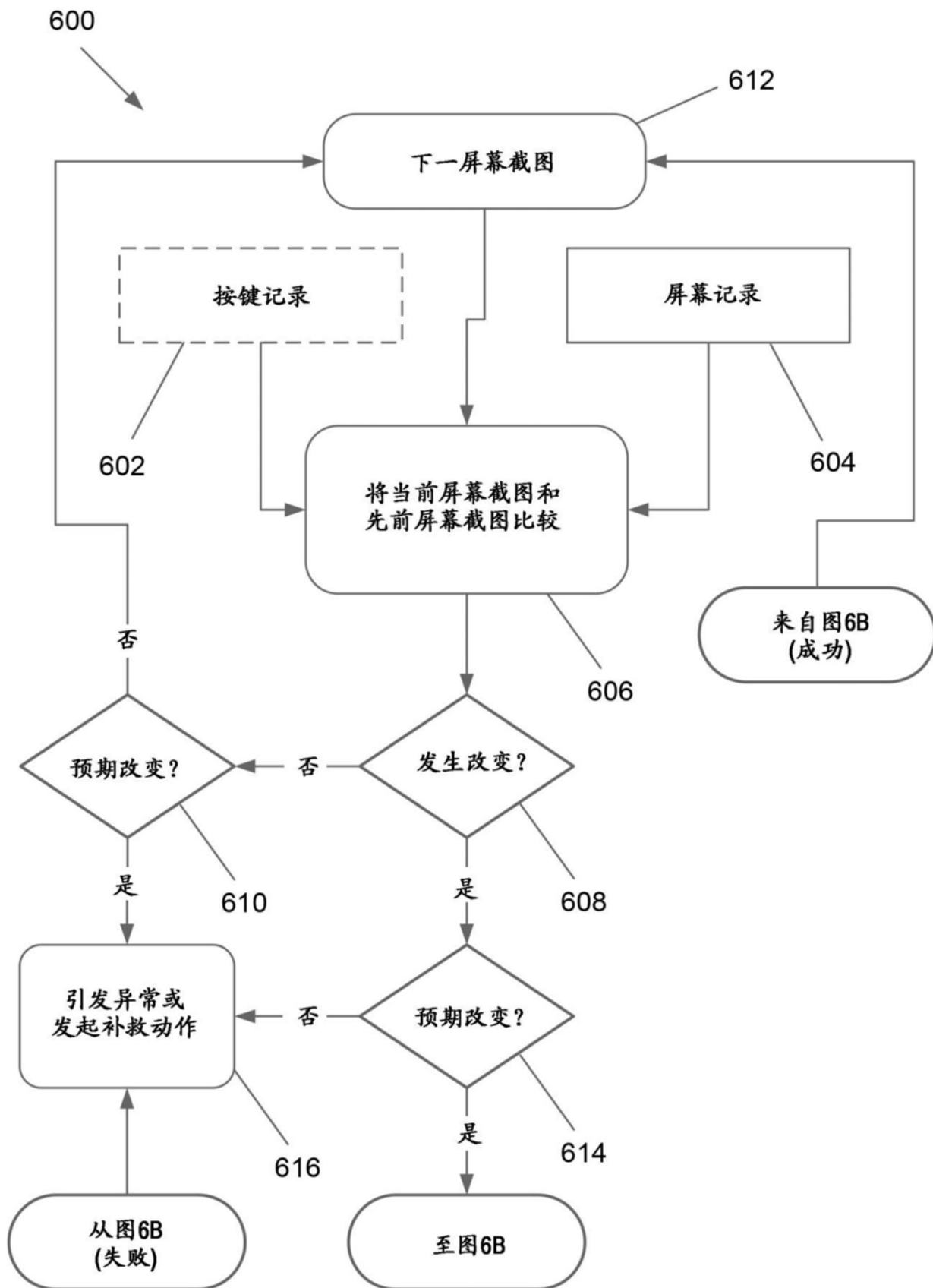


图6A

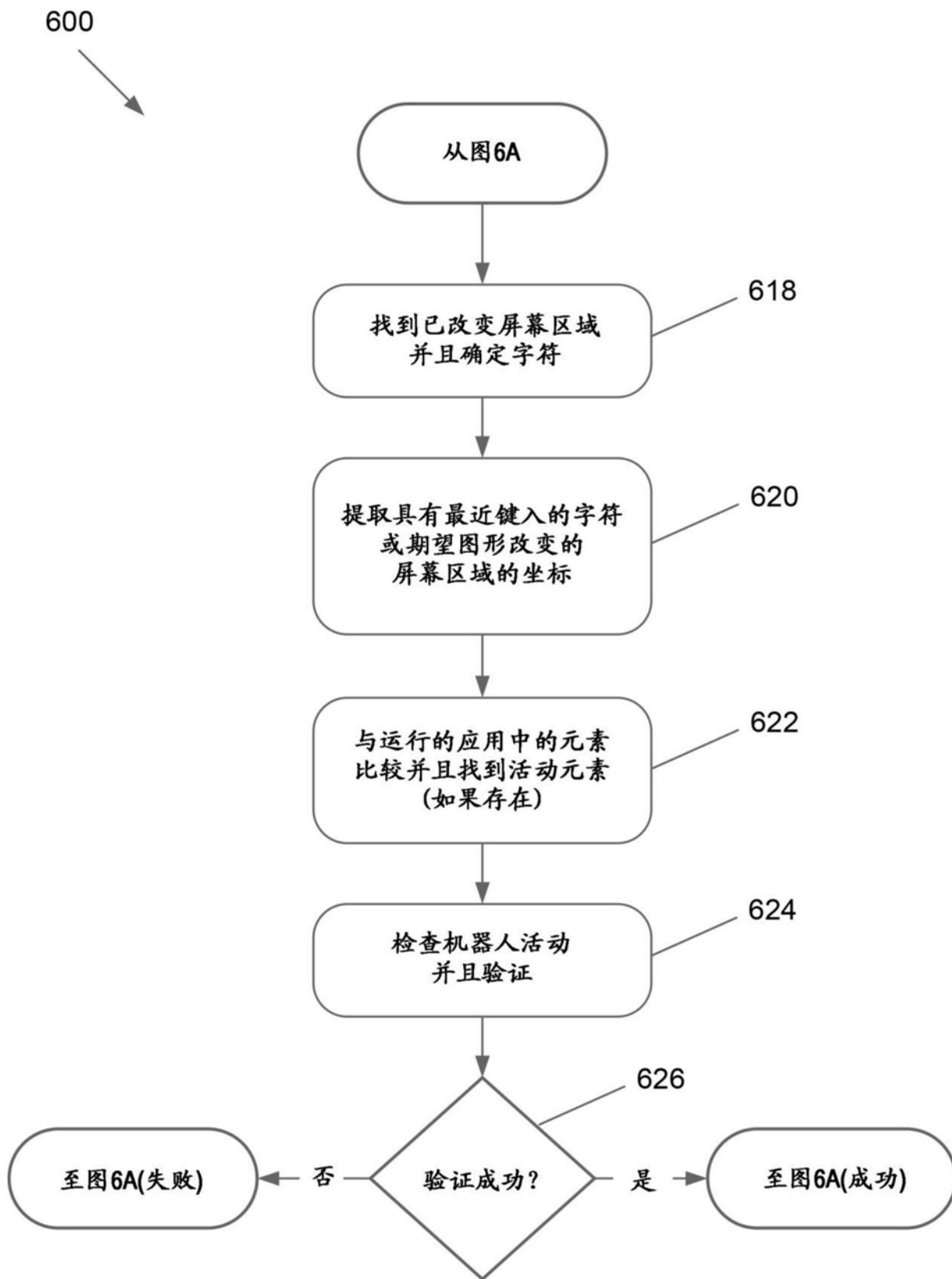


图6B

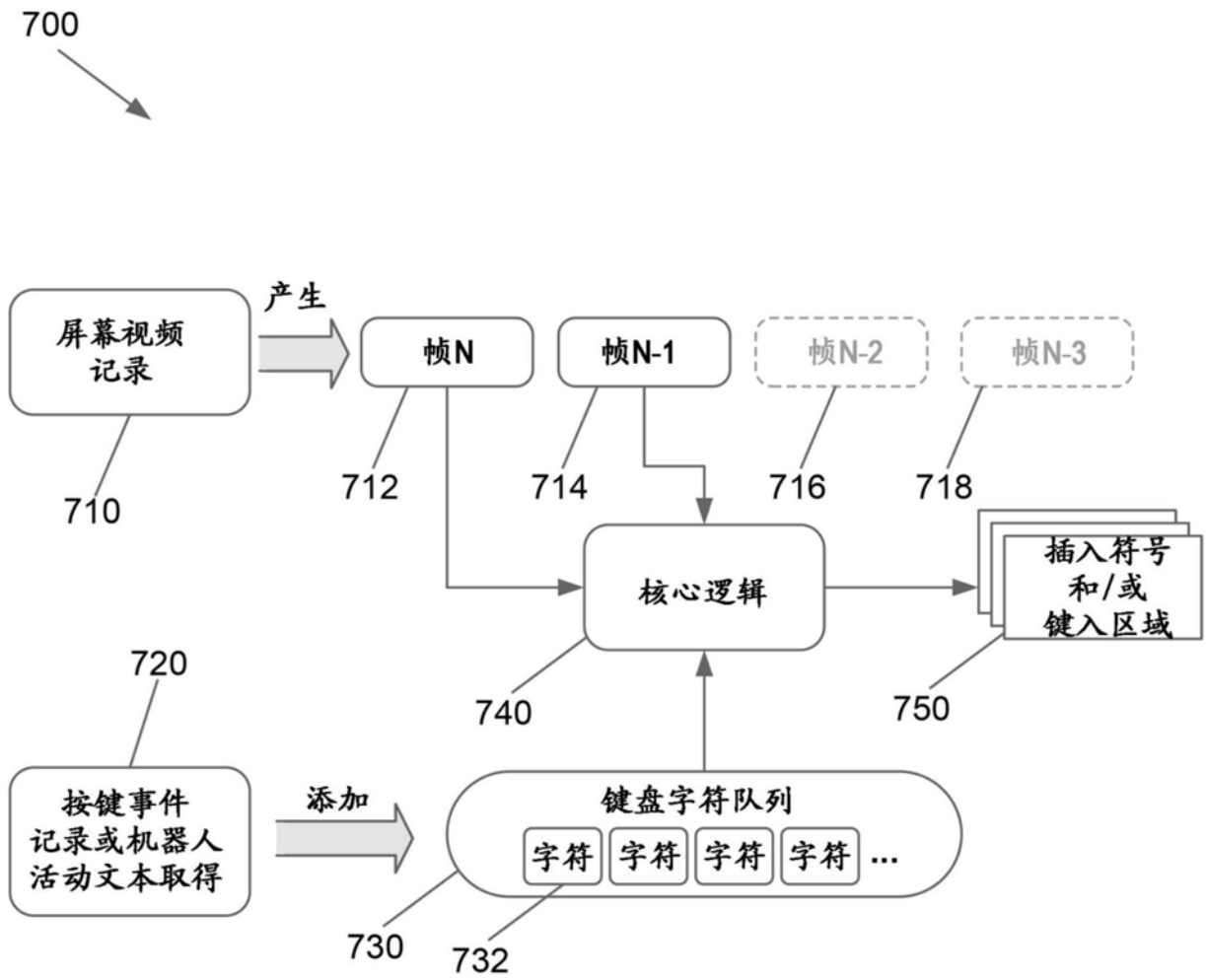


图7

802

帧 N - 1

812

帧 N

校友捐赠

姓名 *

名字 姓氏

电子邮箱 *

电话 *

您希望捐赠的金额

\$

您希望以其他人的名义进行此次捐赠吗?

☐ 是 ☐ 否

捐赠

校友捐赠

姓名 *

名字 姓氏

电子邮箱 *

电话 *

您希望捐赠的金额

\$

您希望以其他人的名义进行此次捐赠吗?

☐ 是 ☐ 否

捐赠

图8A

800

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
1																											
2																											
3																											
4																											
5																											
6																											
7																											
8																											
9																											
10																											
11																											
12																											
13																											
14																											
15																											
16																											
17																											
18																											
19																											
20																											
21																											
22																											
23																											
24																											

图8B

810
↙

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
1																											
2																											
3																											
4																											
5																											
6																											
7																											
8																											
9																											
10																											
11																											
12																											
13																											
14																											
15																											
16																											
17																											
18																											
19																											
20																											
21																											
22																											
23																											
24																											

图8C

810

814

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
1																											
2																											
3																											
4																											
5																											
6																											
7																											
8																											
9																											
10																											
11																											
12																											
13																											
14																											
15																											
16																											
17																											
18																											
19																											
20																											
21																											
22																											
23																											
24																											

图8D

810

816

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27							
1																																		
2							校友捐赠																											
3							姓名 *																											
4																																		
5																																		
6							名字					姓氏																						
7							电子邮箱 *																											
8																																		
9							电话 *																											
10																																		
11																																		
12							您希望捐赠的金额																											
13							\$																											
14							您希望以其他人的名义进行此次捐赠吗?																											
15							☐ 是					☐ 否																						
16																																		
17																	捐赠																	
18																																		
19																																		
20																																		
21																																		
22																																		
23																																		
24																																		

图8E

810

818

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
1																											
2																											
3																											
4																											
5																											
6																											
7																											
8																											
9																											
10																											
11																											
12																											
13																											
14																											
15																											
16																											
17																											
18																											
19																											
20																											
21																											
22																											
23																											
24																											

图8F

900

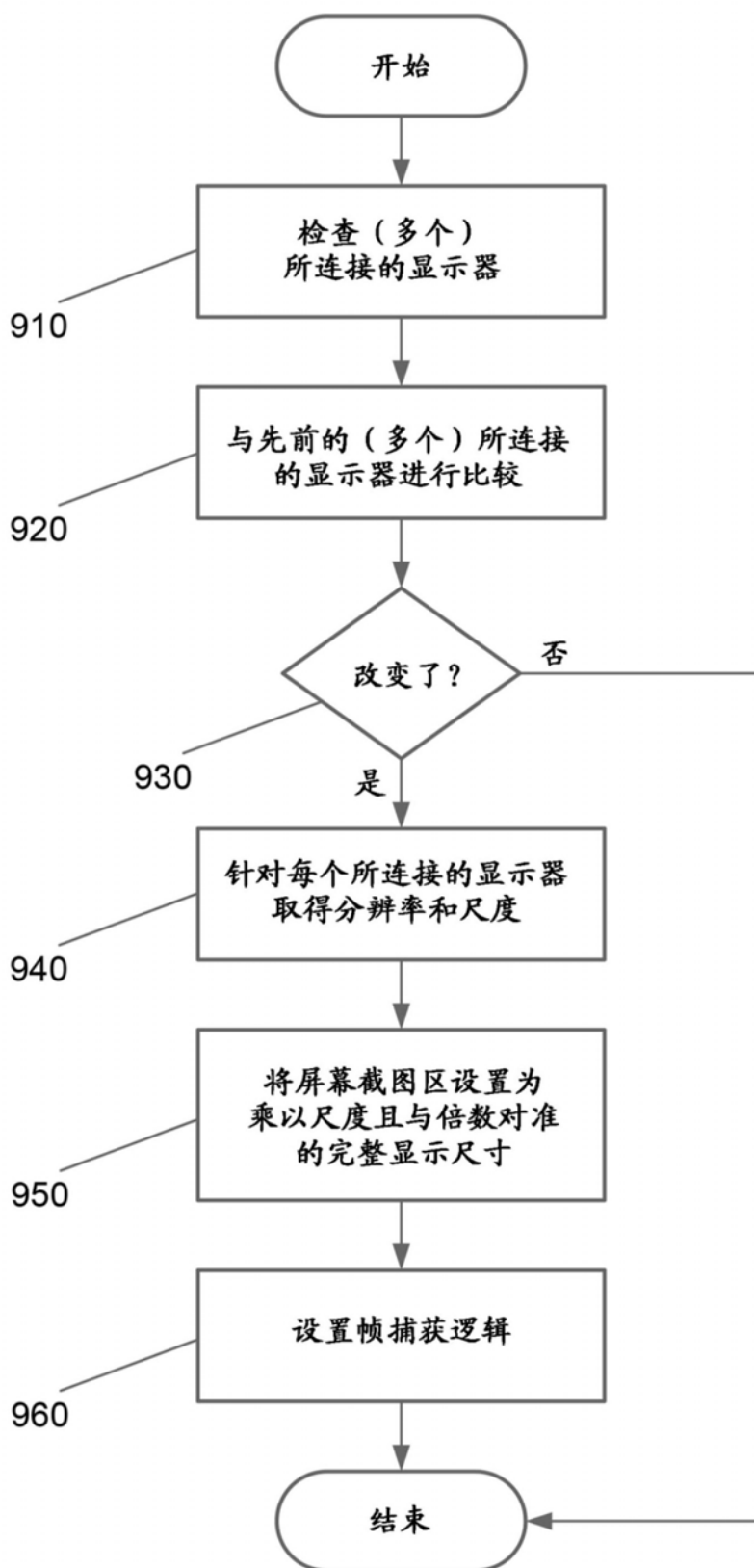


图9

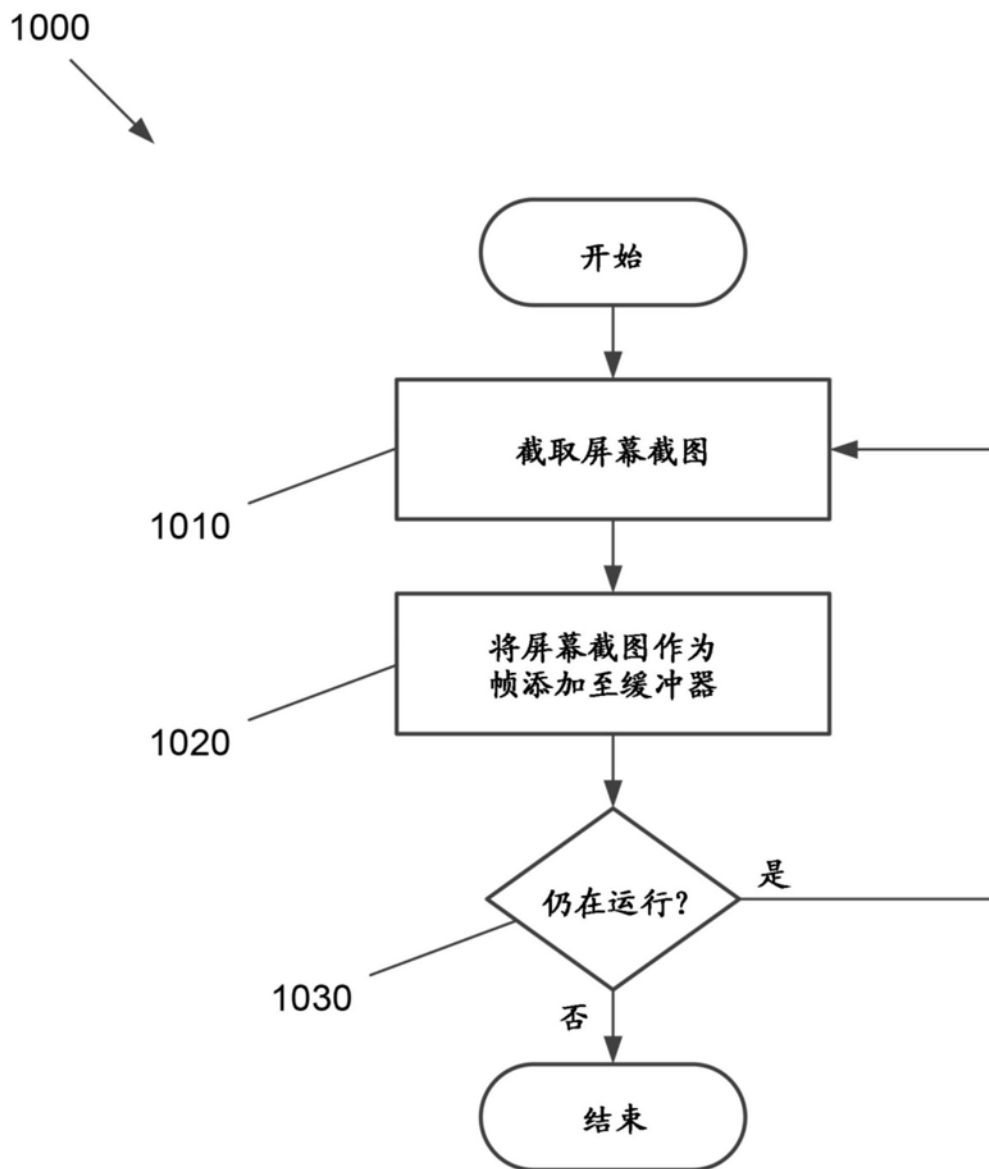


图10

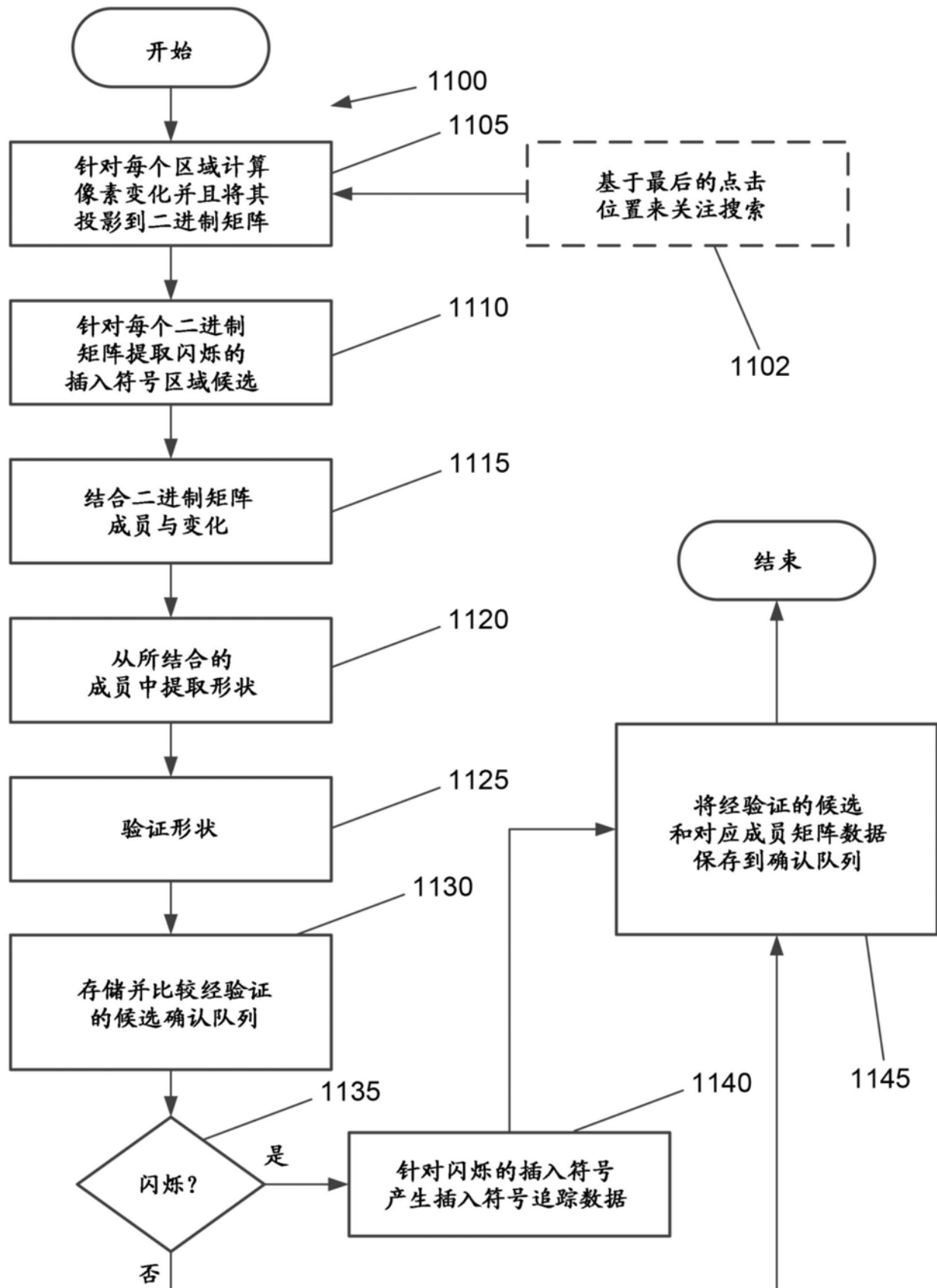


图11

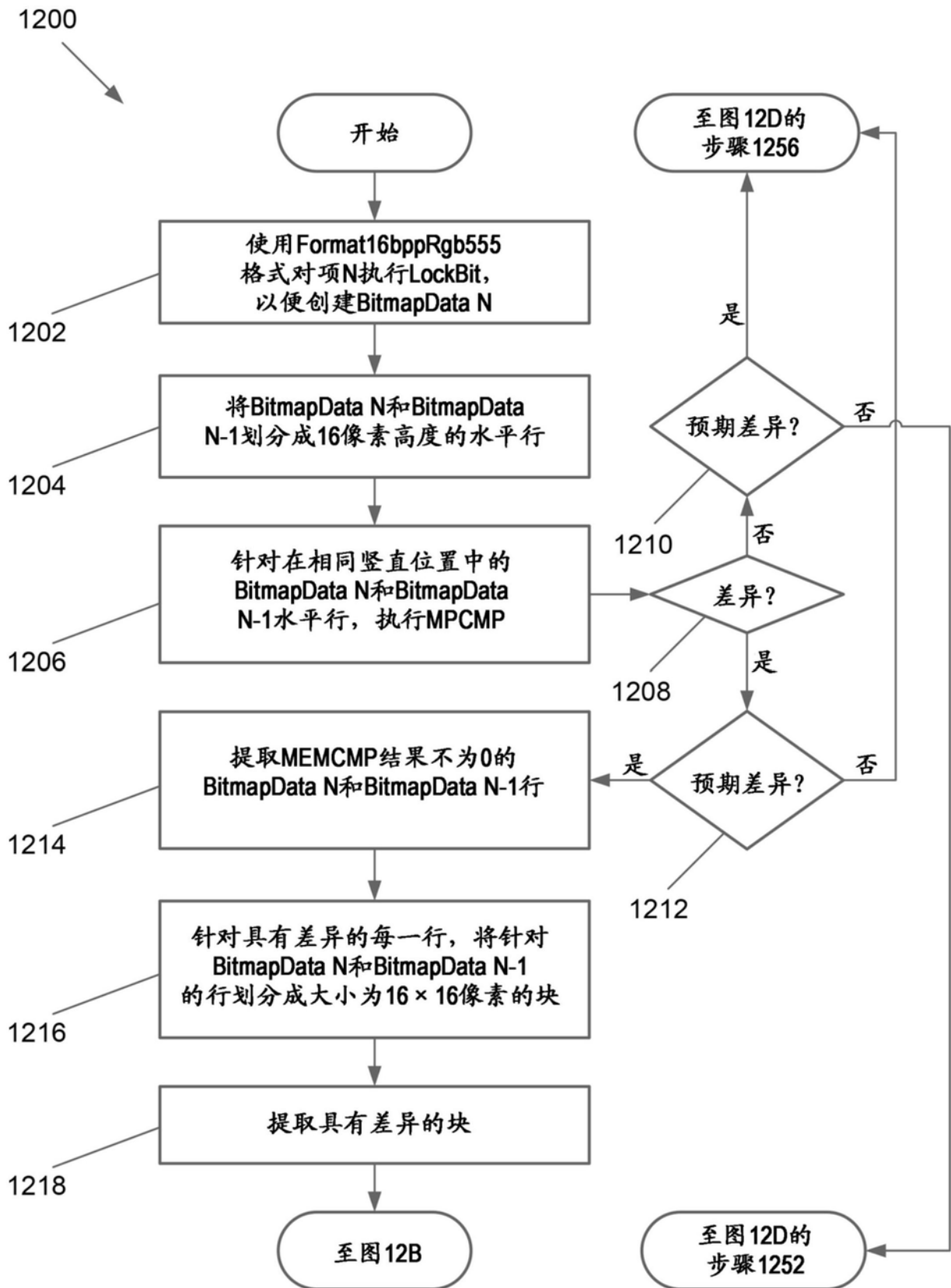


图12A

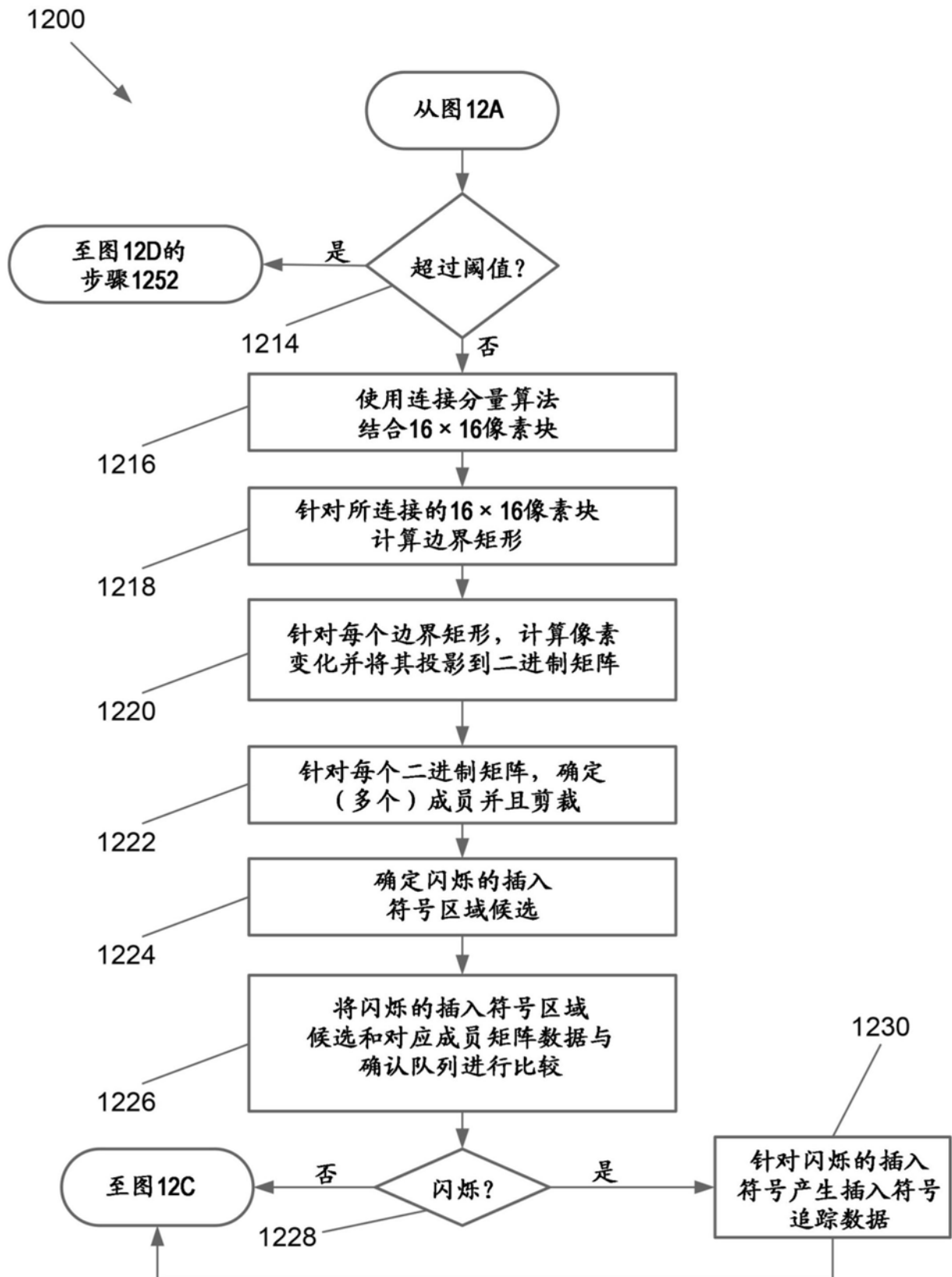


图12B

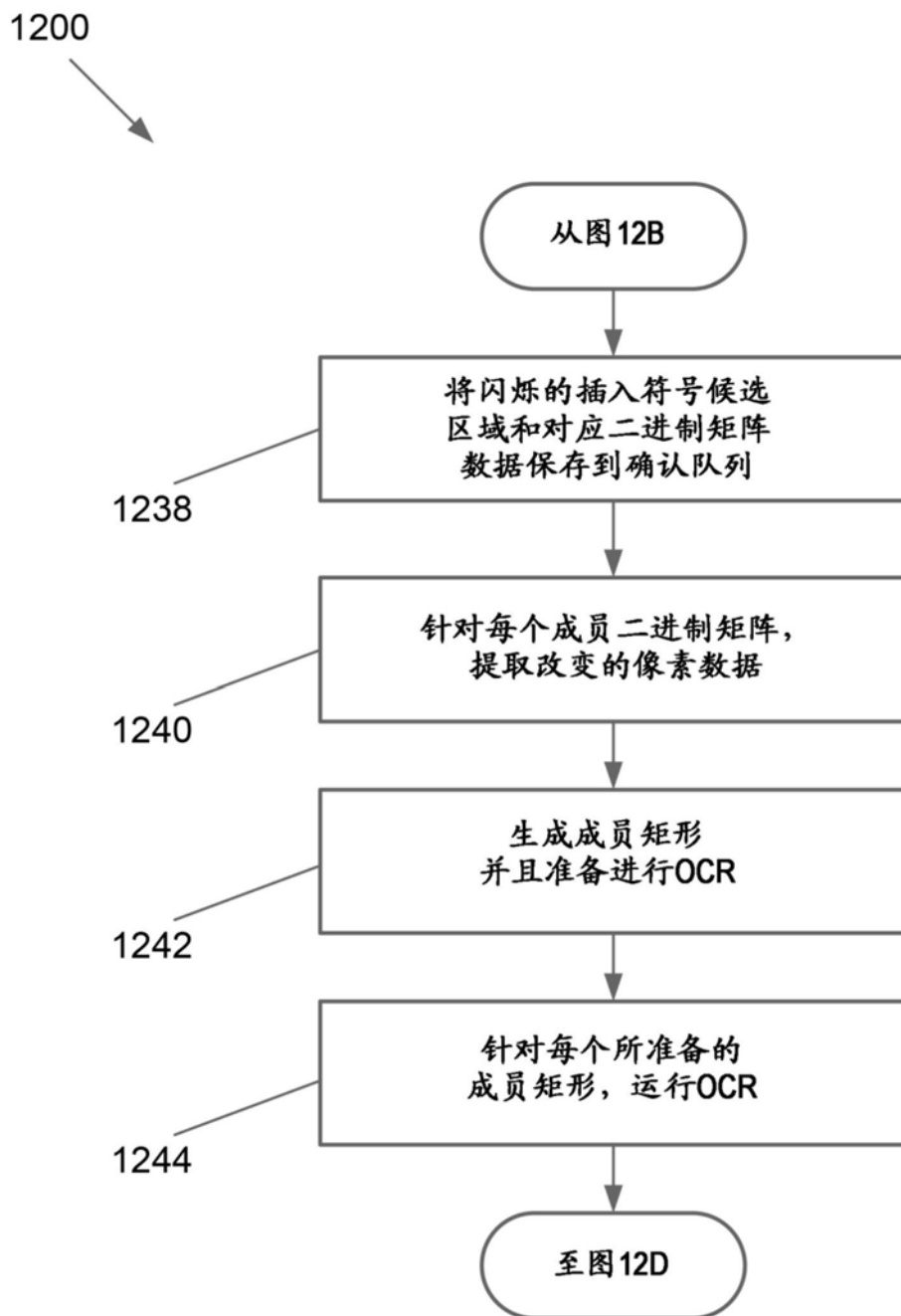


图12C

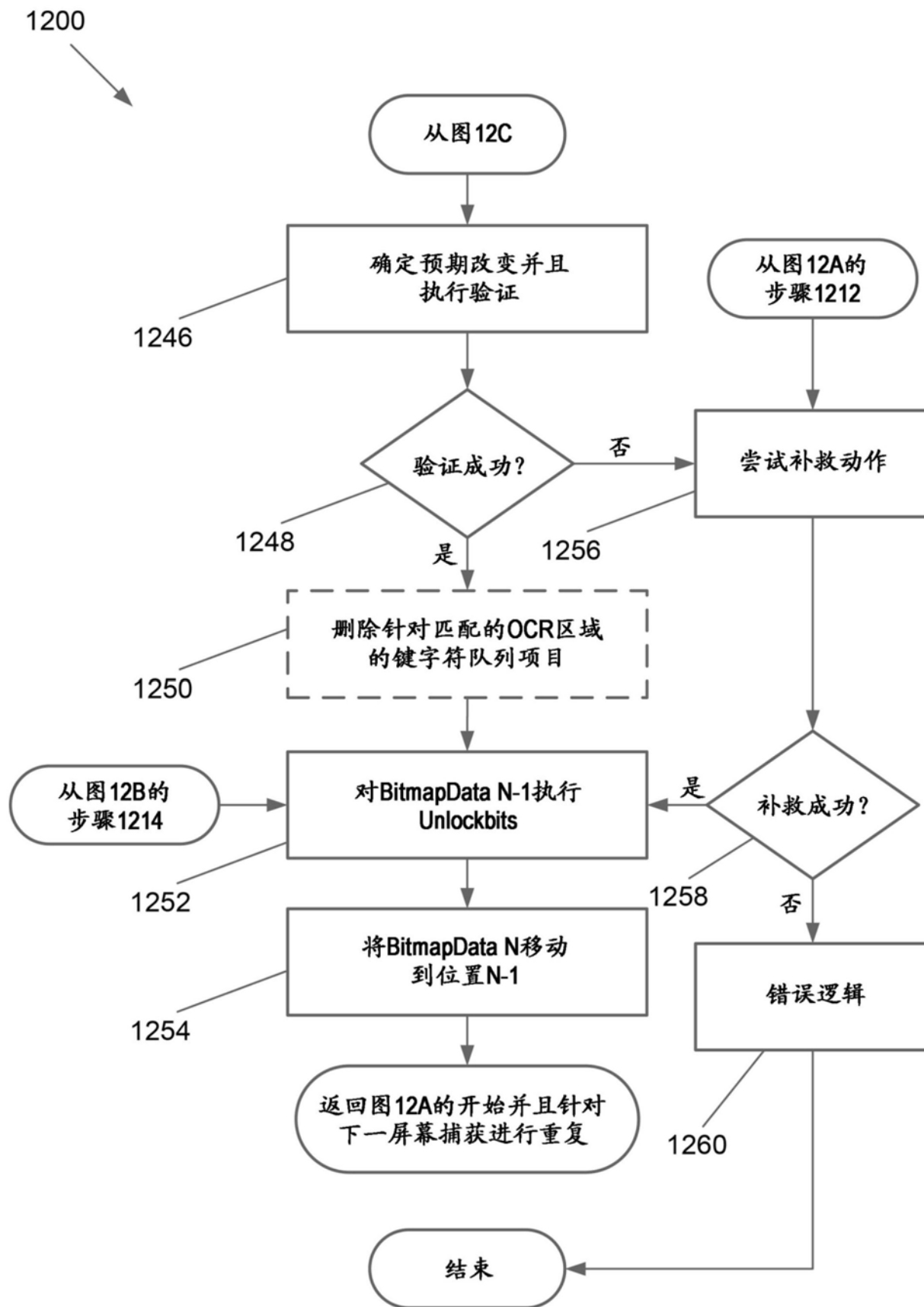


图12D

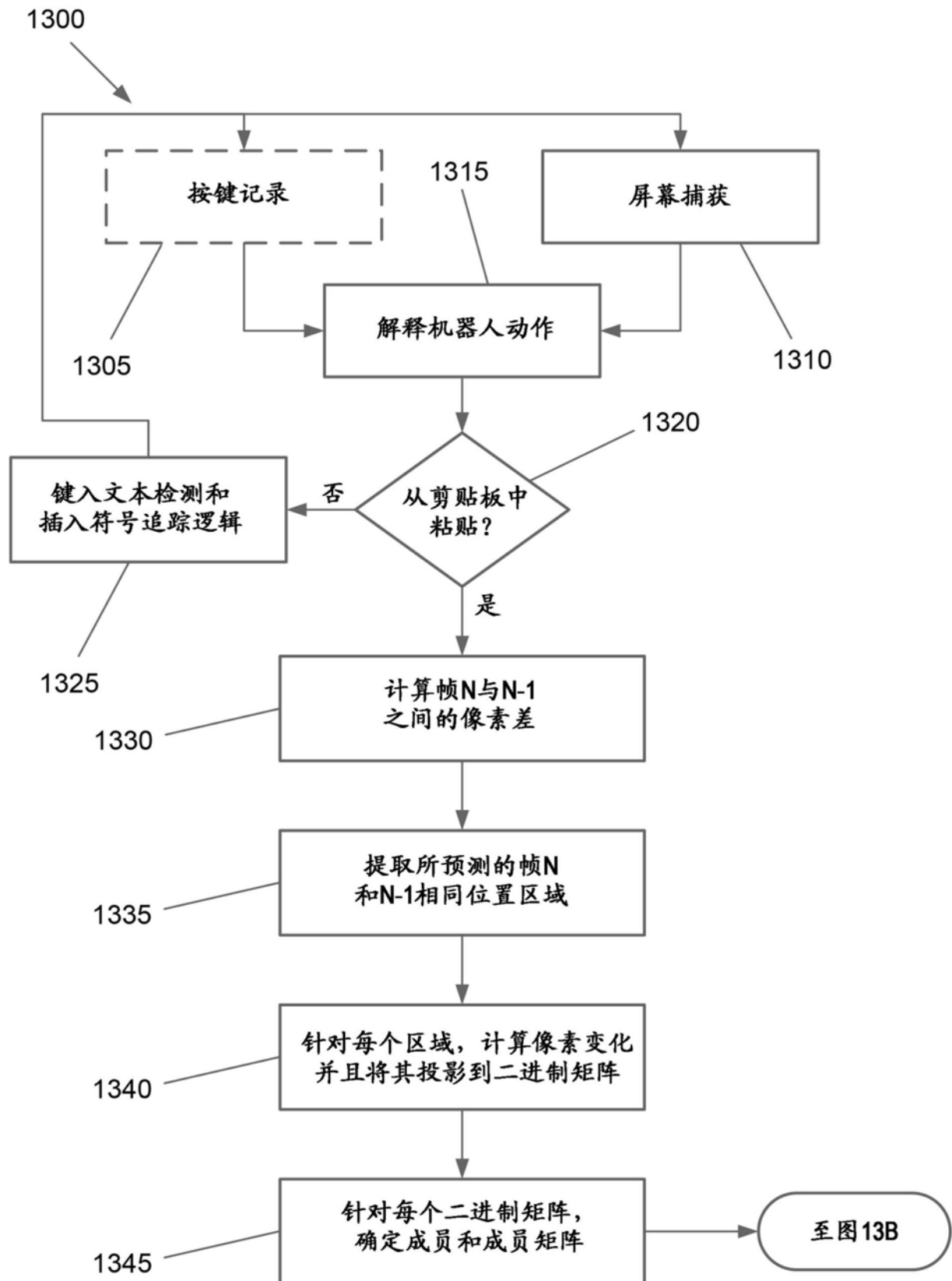


图13A

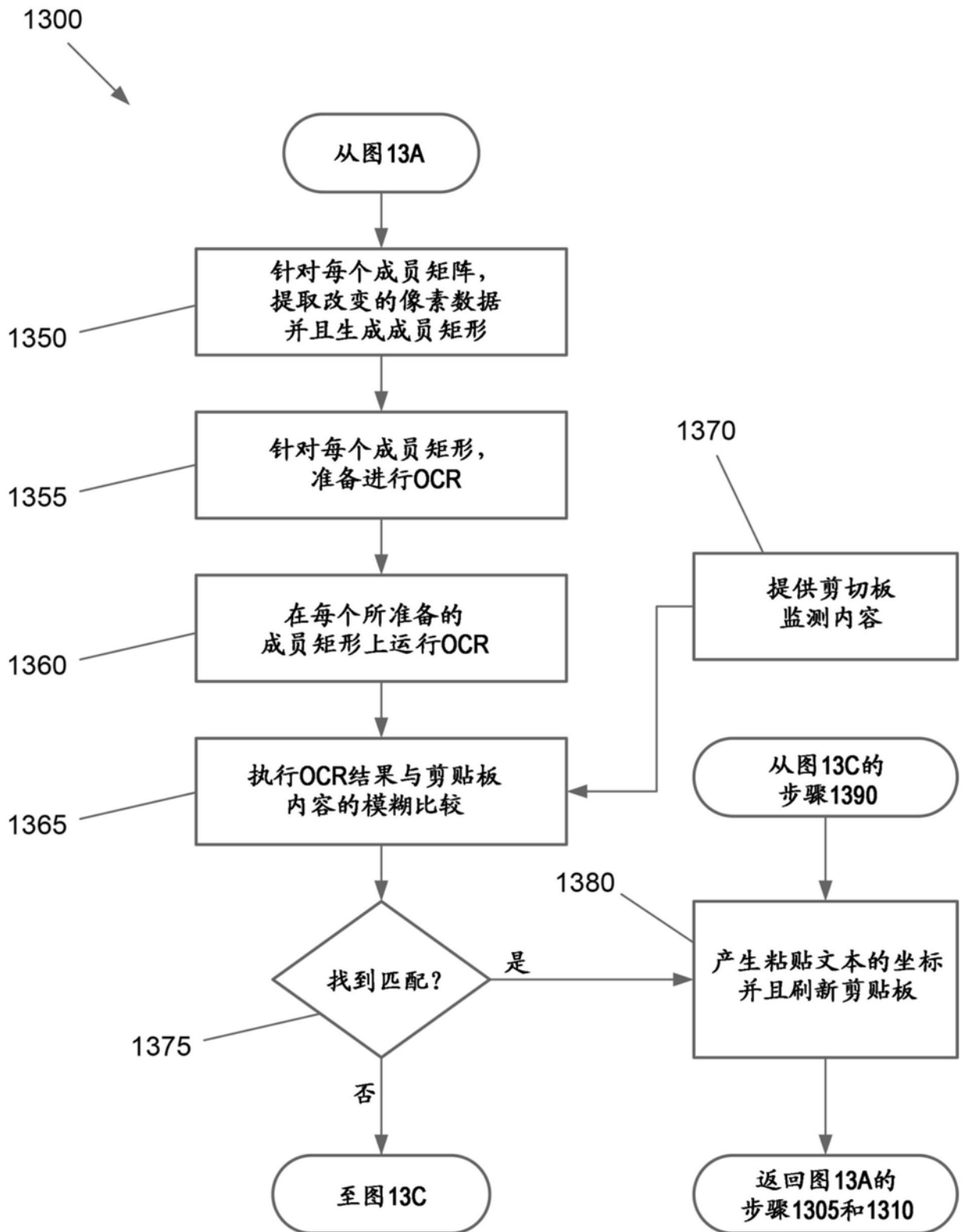


图13B

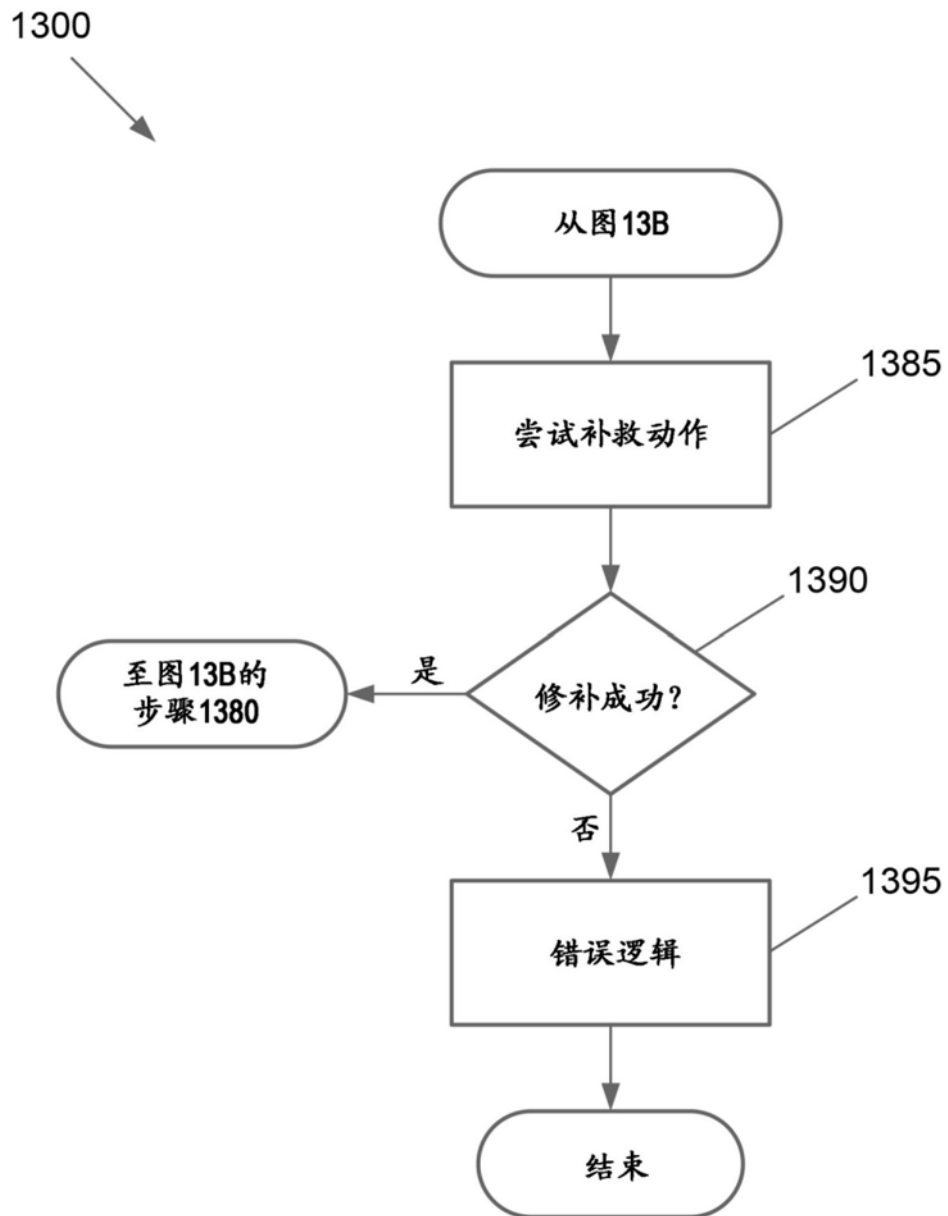


图13C

1410



1	1	1	1	1	1	1
1	0	0	0	0	0	0
1	0	0	0	0	0	0
1	0	0	0	0	0	0
1	1	1	1	0	0	0
1	0	0	0	0	0	0
1	0	0	0	0	0	0
1	0	0	0	0	0	0
1	1	1	1	1	1	1

图14B

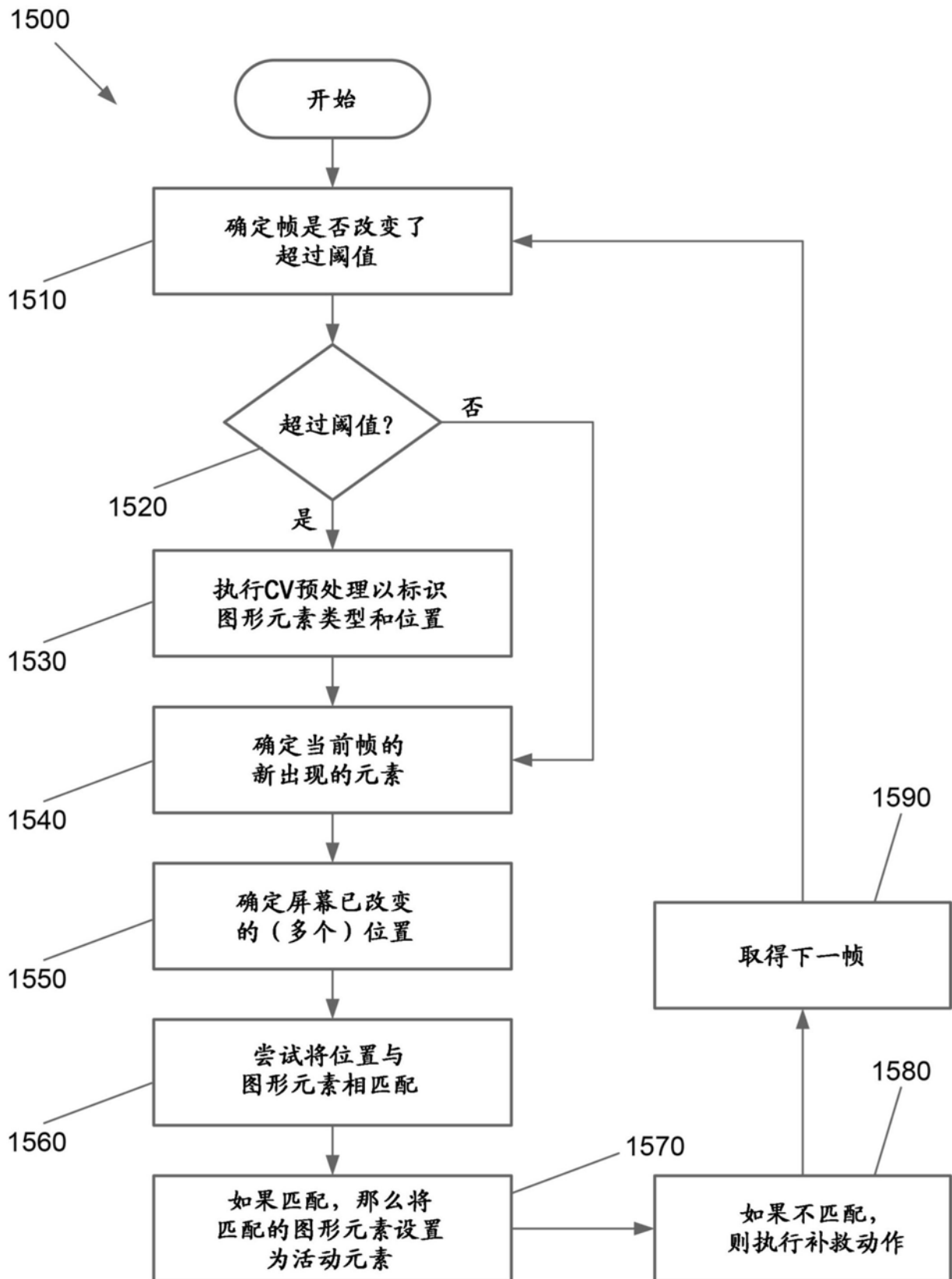


图15

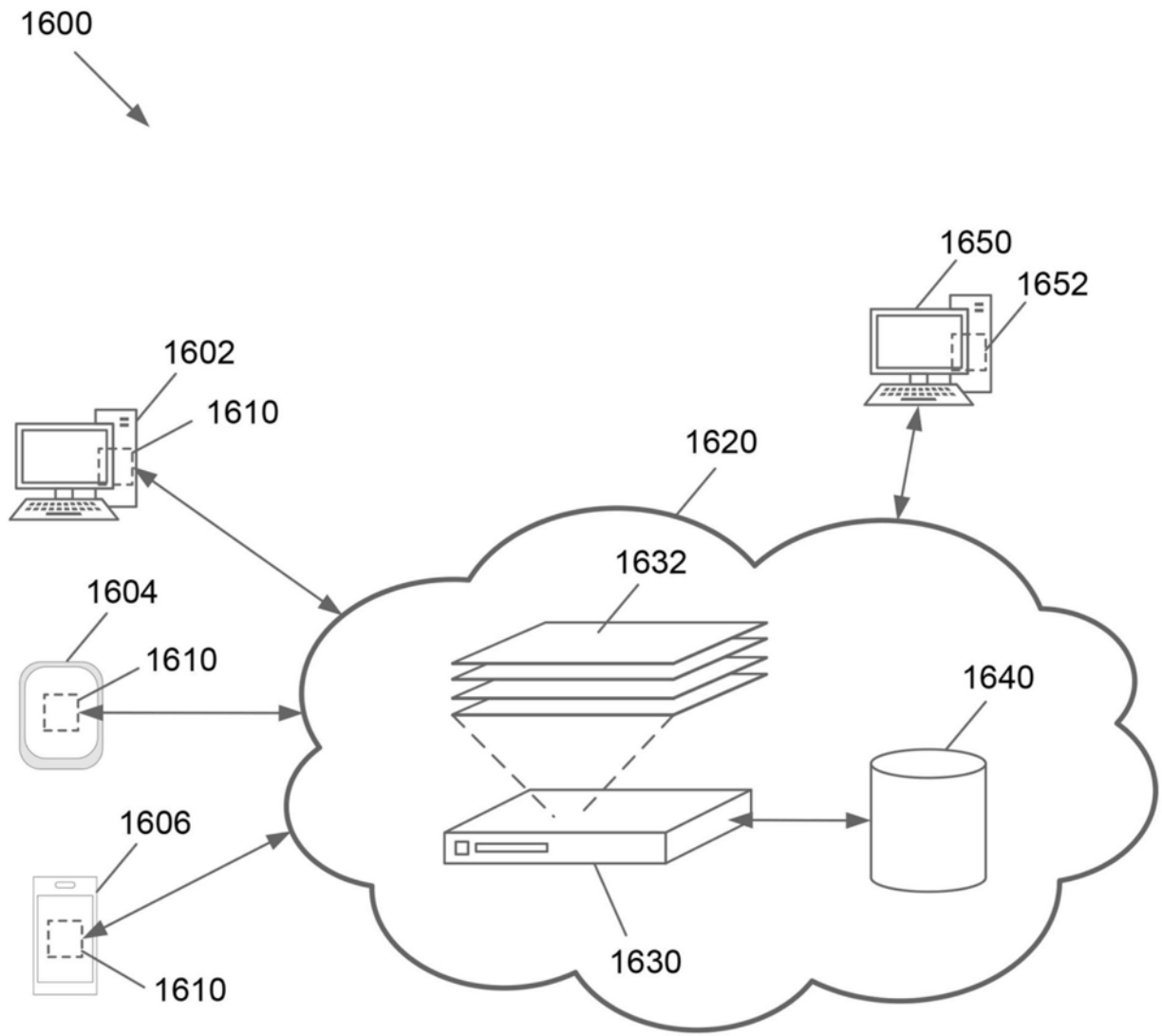


图16

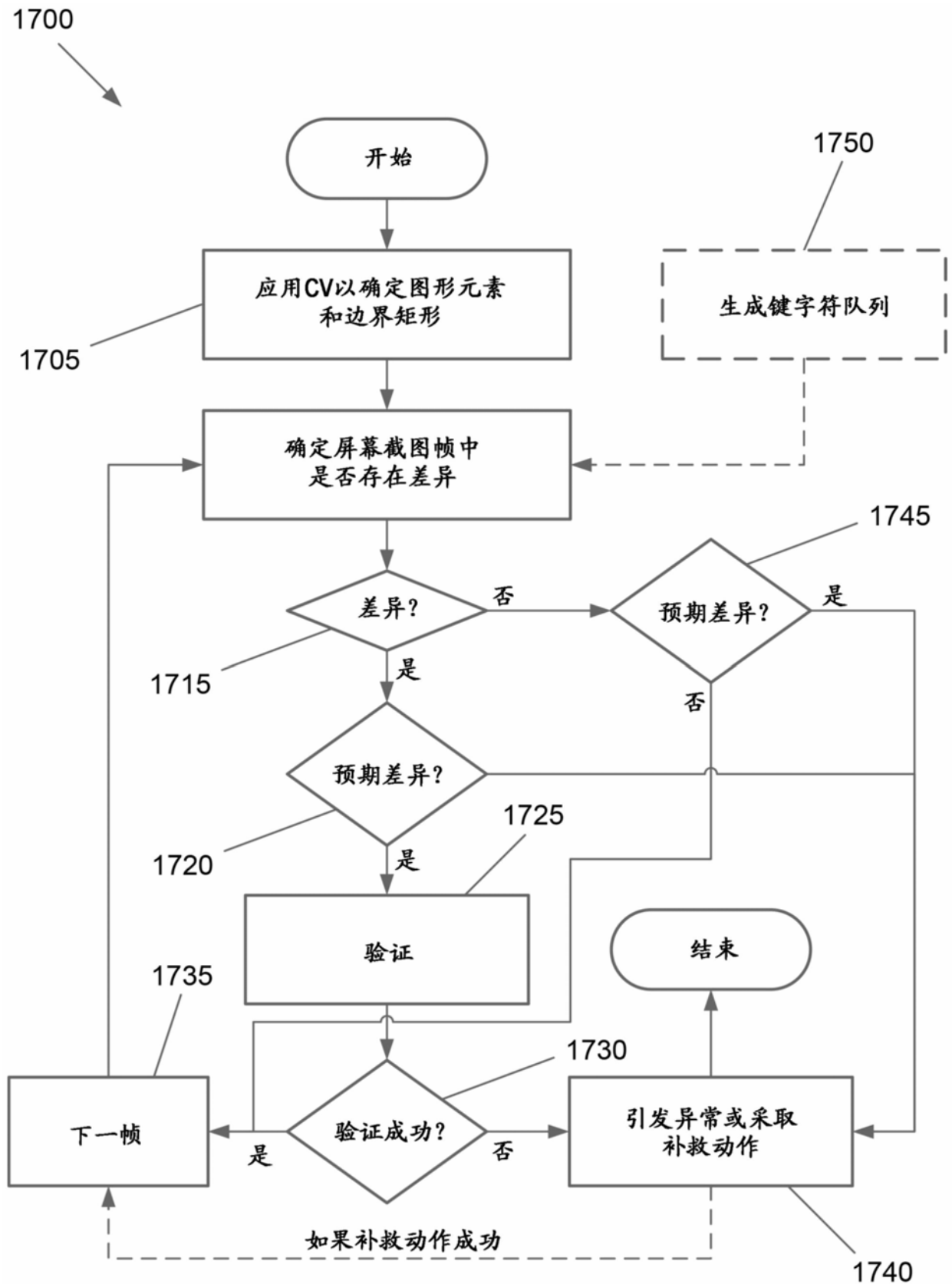


图17