(19)中华人民共和国国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 112034730 A
(43)申请公布日 2020.12.04

(21)申请号 202010425806.0

(22)申请日 2020.05.19

(30)优先权数据
16/417,540 2019.05.20 US

(71)申请人 辉达公司
地址 美国加利福尼亚州

(72)发明人 F·T·拉莫斯 D·福克斯

(74)专利代理机构 北京市磐华律师事务所 11336
代理人 高伟

(51)Int.Cl.
G05B 17/02(2006.01)

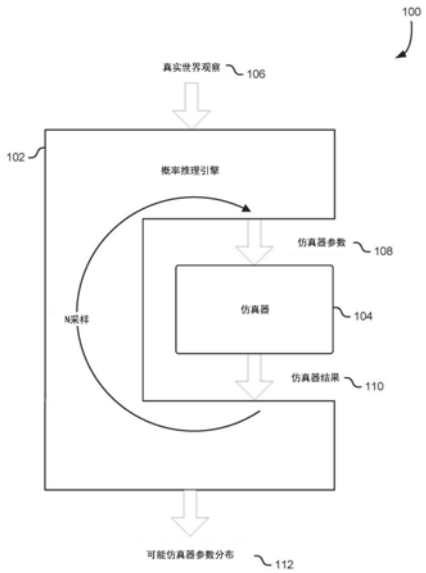
权利要求书2页 说明书24页 附图18页

(54)发明名称

使用机器学习的自主车辆仿真

(57)摘要

本发明实施例公开了一种使用机器学习的自主车辆仿真技术。在一个实施例中，系统计算用于仿真的可能参数的分布以使仿真与真实世界中的经测量的行为相匹配。在一个实施例中，系统基于表示可能参数值的初始估计的统计分布来选择多个仿真参数。在一个实施例中，使用由仿真产生的结果，基于使用傅立叶特征建模的结果的密度来构造可能参数的更新分布。在一个实施例中，可以使用可能参数的更新分布来选择用于仿真的特定参数集，这使得仿真器对经测量的行为进行近似处理。



CN 112034730 A

1.一种处理器,包括一个或更多个算术逻辑单元(ALU),所述处理器被配置为至少部分地基于使用参数值和所述参数值实际出现的频率的函数的一个或更多个仿真来计算参数值的分布。

2.根据权利要求1所述的处理器,其中,所述参数值的分布通过至少部分地基于所述一个或更多个仿真的结果计算密度函数来确定。

3.根据权利要求2所述的处理器,其中,将所述密度函数参数化为一组傅立叶特征。

4.根据权利要求1所述的处理器,其中,所述一个或更多个仿真由根据参数的经预测的先前分布选择的参数集来执行。

5.根据权利要求1所述的处理器,其中,所述参数值的分布表示应用于仿真器的结果使所述仿真器近似于真实世界任务的经测量的结果的参数。

6.根据权利要求5所述的处理器,其中:

所述真实世界任务是由机器人执行的任务;以及

所述仿真器对执行所述任务的机器人进行仿真。

7.一种系统,包括存储器,所述存储器存储指令,作为由一个或更多个处理器的执行所述指令的结果,使所述系统至少部分地基于使用参数值和所述参数值实际出现的频率的函数的一个或更多个仿真来计算参数值的分布。

8.根据权利要求7所述的系统,其中,所述参数值的分布通过至少部分地基于所述一个或更多个仿真的结果计算密度函数来确定。

9.根据权利要求8所述的系统,其中:

所述密度函数被建模为一组傅立叶特征;以及

使用Halton序列选择所述一组傅里叶特征。

10.根据权利要求8所述的系统,其中,所述密度函数被建模为一组随机选择的傅立叶特征。

11.根据权利要求7所述的系统,其中,所述一个或更多个仿真由仿真器使用根据先前生成的仿真参数的分布所选择的参数集来执行。

12.根据权利要求11所述的系统,其中:

所述仿真器近似于由设备执行的真实世界任务;以及

所述仿真器对所述参数集中的各个参数集生成结果。

13.根据权利要求7所述的系统,其中,所述参数值的分布是指示多个参数解的非高斯分布。

14.一种机器可读存储介质,其上存储有一组指令,作为由一个或更多个处理器执行所述指令的结果,使所述一个或更多个处理器至少部分地基于使用参数值和所述参数值实际出现的频率的函数进行的一个或更多个仿真计算参数值的分布。

15.根据权利要求14所述的机器可读存储介质,其中,所述参数值的分布通过至少部分地基于由所述一个或更多个仿真产生的参数结果对计算密度来确定。

16.根据权利要求15所述的机器可读存储介质,其中,所述密度被建模为一组傅立叶特征。

17.根据权利要求16所述的机器可读存储介质,其中,所述一组傅立叶特征根据准蒙特卡洛策略确定。

18.根据权利要求14所述的机器可读存储介质,其中,作为由所述一个或更多个处理器执行所述指令的结果,还使所述一个或更多个处理器使用根据所述参数值的分布所选择的附加仿真来产生参数值的精确分布。

19.根据权利要求14所述的机器可读存储介质,其中,所述一个或更多个仿真采用根据有界一致先验所选择的参数集来执行。

20.根据权利要求14所述的机器可读存储介质,其中,所述一个或更多个仿真采用根据高斯先验选择的参数集来执行。

使用机器学习的自主车辆仿真

背景技术

[0001] 仿真器(Simulators)是用于开发技术和科学发现的重要工具。例如，仿真器可以用于执行机器学习系统的训练，例如自主车辆(autonomous vehicle)控制系统或图像识别系统。它们在诸如宇宙学和生物学等自然科 学中也很有用，其中它们可用于建模(model)自然现象。使用仿真器允许 这些系统以快速且经济高效的方式进行训练，因为它减少了对从真实世界 收集的数据的依赖。但是，仿真器的实用性可能会受到仿真器相对于真实 世界的准确性的限制。如果仿真(simulation)不能准确地代表真实世界，则基于仿真结果得出的结论在应用于真实世界时可能有缺陷或失败。许多 仿真受一组参数控制。例如，对机械系统建模的仿真器可能收到诸如重力、摩擦、空气阻力以及被仿真的各种物体的质量和尺寸参数之类的参数控制。不幸的是，在某些情况下，对正确仿真参数的认识的缺乏，过于简化的仿 真模型或微分方程求解器的数值精度不足可能会阻止将仿真结果无缝地转 移到真实世界的系统中。

附图说明

[0002] 参照附图将描述不同的技术，其中：

[0003] 图1示出了根据实施例的概率推理工具的示例，该概率推理工具确定 再现真实世界观察的可能的仿真参数的分布；

[0004] 图2示出了根据实施例的执行获取-滑动(fetch-slide)任务的机器人 的示例，其中该机器人具有对表的受限访问。

[0005] 图3示出了根据实施例的执行获取-推动(fetch-push)任务的机器人 的示例，其中该机器人具有对表的受限访问。

[0006] 图4示出了根据实施例的执行车-杆平衡任务的机器人的示例，其中，该机器人控制车的运动。

[0007] 图5示出了根据实施例的车-杆问题的极点长度的后验的示例；

[0008] 图6示出了根据实施例的车-杆问题的质杆的后验的示例；

[0009] 图7示出了根据实施例的通过针对获取-滑动问题的不同方法恢复的 后验的示例；

[0010] 图8示出了根据实施例的针对不同方法和问题的不同对数预测的概率；

[0011] 图9示出了根据实施例的通过使用先验的长度参数随机化来训练的对 于车-杆策略的累积奖励的示例；

[0012] 图10示出了根据实施例的通过使用先验的质杆参数随机化来训练的 对于车-杆策略的累积奖励的示例；

[0013] 图11示出了根据实施例的通过使用后验的长度参数随机化来训练的 对于车-杆策略的累积奖励的示例；

[0014] 图12示出了根据实施例的通过使用后验的质杆参数随机化来训练的 对于车-杆策略的累积奖励的示例；

[0015] 图13示出了根据实施例的对于获取-滑动问题的策略的示例；

[0016] 图14示出根据实施例的用于获取-推动问题的策略的示例；

[0017] 图15示出了根据实施例的过程的示例，作为该过程由计算机系统的 处理器执行的结果，使该系统估计仿真参数的分布，当将这些仿真参数应 用于仿真时，使仿真产生期望结果；

[0018] 图16示出了根据实施例的并行处理单元(“PPU”)的示例；

[0019] 图17示出了根据实施例的通用处理集群(“GPC”)的示例；

[0020] 图18示出了根据实施例的存储器分区单元的示例；

[0021] 图19示出了根据实施例的流式多处理器的示例；和

[0022] 图20示出了根据实施例的可以在其中实现各种示例的计算机系统。

具体实施方式

[0023] 本文描述了一种确定仿真参数的系统和方法，当应用这些参数时，使 仿真近似于观察到的真实世界结果。在一个实施例中，参数是控制仿真操 作的值。例如，在一个实施例中，本文描述的贝叶斯推理技术可以用于估 计仿真车-杆(cart-pole)平衡问题的参数，其中，在计算机控制下，带轮 车在平坦表面上来回移动。在一个实施例中，杆使用枢轴连接到车，并且 目标是控制系统以保持杆在直立位置平衡的方式移动车。在一个实施例中， 对于车杆问题，仿真由杆的长度和质量控制以及，在一些示例中，由摩擦 力和空气阻力的附加参数控制。

[0024] 在一个实施例中，系统观察对真实世界任务的尝试(例如，车-杆问题)， 并尝试确定使仿真近似地匹配观察结果的一组参数。在一个实施例中，该 系统生成可能参数的统计分布，该统计分布可以例如在存在多于一个解决 方案的情况下指示多于一个解决方案。在一个实施例中，例如在车-杆问题 中，本文描述的贝叶斯(Bayesian)推理技术可以识别可能参数的非高斯 后验分布，这表明杆长度和杆质量的多种组合可以产生观察到的结果。

[0025] 在一个实施例中，由于仿真器的内部不容易访问，使得产生可能参数 的分布变得更具挑战性。在一个实施例中，该仿真在给定一组参数而不是 反向参数的情况下产生结果(观测值)，但是本文描述的贝叶斯推理技术确 定可能参数的分布，其通过采样来自仿真器的多个参数输出对来产生给定 输出，尽管如此限制。在一个实施例中，至少部分地基于有时被称为先验 (prior)的可能参数的“最佳猜测(best guess)”分布来选择样本。在一个实施例中，例如，先验可以是定值或在可能范围内的常数。在一个实施例 中，先验可基于先前确定的仿真参数的分布。

[0026] 在一个实施例中，将样本转换成表示特定仿真器输出和仿真参数之间 的关系的分布。在一个实施例中，本文所述的贝叶斯推理技术通过对仿真 参数的后验建模来确定参数的分布。在一个实施例中，密度表示参数的期 望分布。在一个实施例中，通过一组傅立叶特征来对密度进行参数化，该 傅立叶特征被示出以提供参数值的更准确的分布，如本文中提供的实验结 果所示。

[0027] 在一个实施例中，随着仿真器变得更加复杂并且能够更准确地表示环 境的动态，诸如运动计划和感知之类的机器人技术中的基本问题可以在转 移到物理机器人上的仿真和解决方案中解决。然而，在一个实施例中，由 于动态模型中的不准确的参数化或简单的

假设,仿真器在某些方面仍可能 不能表示现实。在一个实施例中,本文描述的系统和方法提供了统计框架 以推理关于仿真参数的不确定性。在一个实施例中,给定黑匣子仿真器(或 生成模型)从未知的仿真参数输出状态和动作对的轨迹,然后输出采用本 文所述的贝叶斯推理技术的物理机器人获得的轨迹,本文所述的贝叶斯推 理技术能够开发出无似然(likelihood-free)推理方法,该方法计算仿真参 数的后验分布。在一个实施例中,后验用于域随机化以训练新策略,该新 策略在实际值附近更一致地执行。

[0028] 在一个实施例中,无似然贝叶斯推理被应用于估计机器人仿真器的参 数。在一个实施例中,本文描述的贝叶斯推理技术提供了完整的分布,因 此量化了仿真器关于现实的不确定性。在一个实施例中,作为从机器人仿 真器执行贝叶斯推理的方法的一部分,本文所述的贝叶斯推理技术提供了 一种回归模型,该回归模型使用随机傅里叶特征(“RFF”)和分布的混合 来捕获问题的多模式特性。在一个实施例中,贝叶斯推理技术通过对与先 验相反的后验分布进行随机化来训练策略,即控制器。在各种实施例中, 这提供了在实际环境中表现更好的策略。

[0029] 如本领域技术人员根据本公开将理解的,某些实施例可能能够实现某 些优点,包括以下的一些或全部:(1)通过提供关于仿真参数的分布,本 文描述的贝叶斯推理技术量化了代表现实的仿真器的不确定性,从而允许 识别需要进一步开发的仿真器的组件;(2)通过域随机化,仿真的实现是 从不同的参数化生成,深度学习模型可以从仿真器生成的数据进行训练, 从而大大减少了人工注释;(3)同样,可以在仿真中训练用于在复杂环境 中控制机器人的策略,并在减少训练期间中对机器人造成损害的机会之后, 将其转移到物理系统中,并通过减少需要执行的物理实验数量来节省成本。

[0030] 在一个实施例中,仿真器是使机器人技术中的有效机器学习成为可能 的重要工具。在一个实施例中,物理上精确且逼真的仿真、感知模型和控 制策略可以在被转移到真实的机器人之前更容易地训练,从而节省了运行 复杂实验的时间和成本。然而,在一个实施例中,关于正确的仿真参数的 知识的缺乏,过分简化的仿真模型或微分方程求解器的数值精度不足会产 生与正在仿真的真实系统不够相似的仿真。在一个实施例中,为了改善该问题,使用域随机化(“DR”)。在域随机化中,在训练期间对不同的仿真 参数进行采样,以生成对仿真不确定性具有鲁棒性的模型。

[0031] 在一个实施例中,关于域随机化的一个问题是确定要对哪些仿真参数 进行随机化以及从哪些分布中采样其值。在一个实施例中,这些参数及其 分布是在手动过程中通过迭代测试在随机仿真中学习的模型在实际系统上 是否工作良好来确定的。在一个实施例中,如果模型在真实机器人上不起 作用,则改变随机化参数,使得它们更好地覆盖在真实世界中观察到的条 件。在一个实施例中,为了克服该手动调整过程,在真实机器人上执行的 策略可以用于自动更新采样参数上的高斯分布,以使仿真器更好地匹配现 实。在采样分布限于高斯分布的实施例中,该方法无法对参数之间更复杂 的不确定性和依赖性进行建模。

[0032] 图1示出了根据一个实施例的概率推理工具102的示例,该概率推理 工具102确定再现真实世界观察106的可能仿真参数112的分布。在一个 实施例中,系统100提供一种原理性贝叶斯方法,该方法计算关于仿真器 参数的全部后验(posteriors)。在一个实施例中,100利用贝叶斯分析方法 的无似然性推断来基于在真实系统上获得的少量观察结果更

新关于仿真参 数的后验。在一个实施例中，计算这样的后验的主要困难涉及对似然(likelihood)函数的评估，该似然函数对仿真参数108和对应的仿真器结 果110或真实世界中的观察之间的关系进行建模。在一个实施例中，虽然 仿真器104隐式地定义该关系，但是似然函数使用仿真器模型的逆，即， 如何将观察到的系统行为用于导出对应的仿真参数。在一个实施例中，本 文描述的贝叶斯推理技术不假定访问仿真器104下面的内部微分方程，并 且将仿真器104视为黑盒。

[0033] 在一个实施例中，本文描述的贝叶斯推理技术提供了用于利用机器人 仿真器进行概率推理的通用框架，并且提供了最适合观察数据的仿真参数 的完整空间。相反，各种替代系统提供了近似点解决方案。在一个实施例 中，本文描述的贝叶斯推理技术提供了一种新颖的混合密度随机傅里叶网 络，以通过从由提议先验(proposal prior)和仿真器生成的对$\{\theta_i, x_i^s\}_{i=1}^N$中 直接近似于条件分布$p(\theta|x^r)$。在一个实施例中，通过生成具有域随机化的 策略，其中根据后验将仿真器参数随机化，本文描述的贝叶斯推理技术生 成的策略比直接来自于先验的随机化明显更鲁棒并且更易于训练。

[0034] 在一个实施例中，仿真器104是配置有可执行指令的计算机系统，该 可执行指令实现现实环境、任务或场景的模型。在一个实施例中，计算机 系统包括处理器和存储器，诸如图16-图20所示的那些。在一个实施例中， 仿真器104对包括执行任务的机器人的系统进行建模。在一个实施例中， 机器人是自动驾驶车辆，并且任务是街道导航。在一个实施例中，仿真器 104将一组参数108作为输入，并且这些参数影响仿真器的操作。在一个 实施例中，可以调整该组参数108，以使得仿真器104紧密地逼近真实世 界的环境或产生期望的结果。

[0035] 在一个实施例中，概率推理工具102是配置有与仿真器104交互的可 执行指令的计算机系统。在一个实施例中，概率推理工具102向仿真器提 供一组参数108，并接收相应的仿真器结果110。在一个实施例中，概率推 理工具不能访问仿真器104的内部，并且概率推理工具102多次调用仿真 器104以生成多个样本。在一个实施例中，多个样本中的每个样本是包括 一组输入参数和由该组输入参数产生的对应仿真器结果的值对(value pair)。在一个实施例中，样本由概率推理工具102处理以产生参数的估计分布， 该估计分布产生来自仿真器的给定结果。在一个实施例中，概率推理工具 102将真实世界观察作为输入。在一个实施例中，通过在真实世界中引导 通过仿真近似的任务以及在真实世界中测量该结果来获得真实世界观察。 在一个实施例中，真实世界观察106是目标值或期望值，概率推理工具102 针对该目标值或期望值确定对应的一组参数(或多组参数)。在实施例中， 将对应的参数集确定为指示参数将产生期望结果的机会的分布。

[0036] 在一个实施例中，仿真器用于通过允许更快、高度可扩展和低成本的 数据收集来加速机器学习的影响。在一个实施例中，本系统可以应用于诸 如经济学、进化生物学和宇宙学的领域，其中仿真器提供科学发现方面的 进步。例如，在一个实施例中，机器人的控制系统中可能存在“现实差距”， 并且机器人视觉也受到该问题的影响。在一个实施例中，在来自仿真的图 像上训练的算法可能在不同的真实世界环境中失败，因为真实世界的外观可能与仿真中复制的外观有很大差异。

[0037] 在一个实施例中，在训练控制策略的同时随机化仿真器的动态 (dynamic)减轻了现实差距问题。在一个实施例中，仿真参数从诸如阻尼、摩擦力和物体质量等的物理设置

变化到诸如物体纹理和形状等的视觉参数。在一个实施例中,噪声被添加到系统参数中,而不是从一致先验分布中采 样新参数。在一个实施例中,还可以在机器人移动上看到扰动,其中通过 扰动模型的集合来完成计划。在一个实施例中,在仿真和现实之间进行交 错策略部署也可以很好地作用于旋钉入孔和打开柜子抽屉任务。

[0038] 在一个实施例中,来自数据的仿真的学习模型利用对物理世界的理解,潜在地帮助解决相关问题。在一个实施例中,近似贝叶斯计算(“ABC”) 用于解决这种类型的问题。在一个实施例中,拒绝(Rejection)ABC是其 中如果参数设置在某个指定范围内则接受/拒绝参数设置的方法。在一个实 施例中,该组接受参数近似于真实参数的后验。在一个实施例中,本文描 述的贝叶斯推理技术使用马尔可夫链蒙特卡洛ABC(“MCMCABC”)来扰 动接受的参数,而不是独立地提出新参数。在一个实施例中,本文描述的 贝叶斯推理技术使用顺序蒙特卡洛ABC(“SMC-ABC”)来利用顺序重要 性采样来仿真缓慢变化的分布,其中连续分布是真实参数后验的近似值。 在一个实施例中,本文描述的贝叶斯推理技术使用无∈(∈-free)方法进行 无似然性推断,其中密度随机傅里叶网络的混合通过高斯混合估计真实后验的参数。

[0039] 在一个实施例中,可以使用深度强化学习(“Deep RL”)技术来解决 广泛的复杂机器人控制问题。在一个实施例中,可以使用具有诸如置信区 域策略优化(“TRPO”)和近端策略优化(“PPO”)等算法的策略搜索来成 功地解决诸如钟摆、山地车,机器人和车-杆等控制问题。在一个实施例中, 使用传统的策略搜索很难解决机器人技术中更复杂的任务,例如操纵任务。 在一个实施例中,本文描述的贝叶斯推理技术可以用于经由域随机化的策 略搜索。

[0040] 在一个实施例中,本文描述的贝叶斯推理技术通过仿真参数θ,黑盒 生成模型或从这些参数生成仿真观测值x^s的仿真器$x^s=g(\theta)$取先验$p(\theta)$, 并从物理世界x^r来计算后验$p(\theta|x^s,x^r)$。在一个实施例中,计算该后验的挑 战涉及对从仿真器隐式定义的似然函数$p(x|\theta)$的评估。在一个实施例中, 仿真器由与数值或分析求解器(solver)相关联的一组微分方程控制,这些 数值或分析求解器通常难以处理并且评估昂贵。在一个实施例中,系统不能直接访问这些方程,因此将仿真器视为黑匣子。在一个实施例中,这允 许该系统与许多机器人仿真器(甚至是封闭源代码的仿真器)一起使用, 但是需要一种方法,其中不能直接评估似然性,而是通过执行前向(forward) 仿真而从采样中评估似然性(likelihood)。在一个实施例中,这被称为无似 然推断。在一个实施例中,一种确定无似然推断的算法家族是近似贝叶斯 计算(“ABC”)。

[0041] 在实施ABC的实施例中,仿真器用于根据参数先验从样本中生成综 合观察。在一个实施例中,当从合成数据计算出的特征或足够的统计数据 与来自从物理实验获得的真实观察的特征或足够的统计数据相似时,接受 该样本。在一个实施例中,作为基于采样的技术,尤其是当参数空间的维 数较大时,ABC可能收敛(converge)较慢。在一个实施例中,ABC使用 贝叶斯规则来近似后验$p(\theta|x=x^r)\propto p(x=x^r|\theta)p(\theta)$。然而,在一个实施 例中,由于似然函数$p(x=x^r|\theta)$不可用,所以不能应用针对贝叶斯推理的其 他方法。在一个实施例中,ABC通过$p(||x=x^r||<\in|\theta)$近似于$p(x=x^r|\theta)$来 解决该问题,其中∈是定义围绕真实观测值x^r球形的小值,并且使用蒙特 卡洛估计其值。在一个实施例中,近似的质量随着∈的减小而增加;但是, 由于大多数仿真都不会落在可接受的范围内,因此计算成本可能会

过高。

[0042] 在一个实施例中，本文描述的贝叶斯推理技术可以应用于机器人技术 中的强化学习和策略搜索。在一个实施例中，本文描述的贝叶斯推理技术 被应用于默认的RL场景，其中代理在离散的时间步长中与环境E交互。 在一个实施例中，在每个步骤t，代理接收观测值o^t，采取动作a^t并接收实 数奖励(reward)r^t。在一个实施例中，机器人技术中的动作是实值为$a^t \in R^D$，并且通常会部分地观察环境，从而观察历史可以由动作对$\eta(\beta)=\{s_t,a_t,o_t\}_{t-0}^{T-1}$表示。在一个实施例中，本文描述的贝叶斯推理技术试 图最大化已打折(discounted)的未来奖励的预期总和，其通过遵循由β参 数化的策略$\pi(a_t|s_t;\beta)$进行，

[0043] $$J(\beta)=\mathbb{E}_\eta[\sum_{t-0}^{T-1}\gamma^t r(s_t,a_t)|\beta]$$ 。

[0044] 在一个实施例中，强化学习中的各种方法利用称为Bellman方程的递 归关系，其中Q^π是动作值函数，描述了在采取动作a_t之后，在状态s_t处， 以及在遵循策略π之后的预期回报，

[0045] $$Q^\pi(s_t,a_t)=\mathbb{E}_{r_t,s_{t+1}}[r(s_t,a_t)+\gamma\mathbb{E}_{a_{t+1}}[Q^\pi(s_{t+1},a_{t+1})]]$$

[0046] 在一个实施例中，将RL方法应用于具有连续动作空间的控制任务。在一 个实施例中，深度确定性策略梯度可以被应用于广泛的控制问题。在一个 实施例中，一个警告(caveat)是DDPG算法依赖有效的经验采样来良好 地执行，因此，改善如何收集经验是重要的主题。在一个实施例中，在奖 励信号离散的机器人任务的指令(repertoire)中，经验回放和优先经验回 放执行不充分。在一个实施例中，后见之明经验回放(Hindsight Experience Replay)(“HER”)在这种情况下表现良好，因为它将单个轨迹/目标分解 为较小的轨迹/目标，从而为策略优化算法提供了更好的奖励(reward)信 号。

[0047] 在一个实施例中，策略搜索算法基于通过置信区域的优化。在一个实 施例中，通过置信区域进行优化对上述经验采样问题不太敏感。在一个实 施例中，用于探索的最大步长由其置信区域确定，然后逐步评估最优点， 直到达到收敛为止。在一个实施例中，更新受限于它们自己的置信区域， 因此，学习速度得到更好的控制。

[0048] 在一个实施例中，近端策略优化和置信区域策略优化在广泛的控制问 题中应用这些思想提供了最先进的性能。在一个实施例中，两种技术在采 样经验的方式上不同。在一个实施例中，第一个是策略外的算法，其中经 验由行为策略生成，第二个是策略上的算法，其中用于生成经验的策略与 用于执行控制任务的策略相同。在一个实施例中，这些算法在不同的机器 人控制场景下具有可比的性能。

[0049] 在一个实施例中，本文描述的贝叶斯推理技术通过直接学习由参数φ参数化的条件密度$q_\phi(\theta|x)$来近似难处理的后验$p(\theta|x=x^r)$。在一个实施例 中，如我们将看到的，$q_\phi(\theta|x)$采取混合密度随机特征网络的形式。在一个 实施例中，为了学习参数φ系统首先生成具有N个对(θ_n,x_n)的数据集， 其中θ_n独立于被称为建议先验的分布$\tilde{p}(\theta)$而绘制。通过使用参数θ_n运行 仿真器获得x_n，使得$x_n=g(\theta_n)$。在一个实施例中，当似然性$\prod_n q_\phi(\theta_n|x_n)$ 最大化w.r.t.φ时，$q_\phi(\theta|x)$与$\frac{\tilde{p}(\theta)}{p(\theta)}p(\theta|x)$成比例。在一个实施例中，使对数 似然性最大化以确定φ的系统

[0050] $\mathcal{L}(\phi)=\frac{1}{N}\log\ q_{\phi}(\theta\ |x_{n})$

[0051] 在一个实施例中，在完成这些之后，通过下式获得后验的估计：

[0052] $\hat{p}(\theta\,|x=x^{r})\propto\frac{\mathrm{p}(\theta)}{\tilde{p}(\theta)}q_{\phi}(\theta\,|x{=}x^{r})$

[0053] 其中p(θ)是期望的先验，可能与建议先验不同。在一个实施例中，当$\tilde{p}(\theta)=p(\theta)$时，由此可见$\hat{p}(\theta\,|x=x^{r})=\ q_{\phi}\ (\theta\,|x=x^{r})$。在一个实施例中，当$\tilde{p}(\theta)\neq p(\theta)$时，系统如下所述调整后验。在一个实施例中，本文描述的贝叶斯推理技 术将条件密度$q_{\phi}(\theta|x)$建模为K高斯的混合，

[0054] $q_{\phi}\ (\theta\,|x)=\sum_{k}\alpha_{k}\mathcal{N}(\theta|\mu_{k},\Sigma_{k})$

[0055] 其中$\alpha=(\alpha_{1},\cdots,\alpha_{k})$是混合系数，$\{\mu_{k}\}$是均值，且$\{\Sigma_{k}\}$是协方差矩阵。在 一个实施例中，当如下所述计算α、μ和Σ时，本文描述的贝叶斯推理技术 使用准蒙特卡洛(QMC)随机傅里叶特征。

[0056] 在一个实施例中，将Ψ(x)表示为特征向量，并且将混合系数计算为

[0057] $\alpha=\mathrm{softmax}(W_{\alpha}\Phi(x)+b_{\alpha})$

[0058] 在一个实施例中，对于i＝1；...；K的运算符(operator) $\mathrm{Softmax}(z)_{i}=\frac{\exp(z_{i})}{\sum_{k=1}^{K}\exp z_{k}}$强制执行系数之和等于1，并且每个系数在0和1之间。在一个 实施例中，均值定义为特征向量的线性组合。在一个实施方案中，对于混 合的每个分量，

[0059] $\mu_{k}=W_{\mu_{k}}\Phi(x)+b_{\mu_{k}}$。

[0060] 在一个实施例走过，本文描述的贝叶斯推理技术采用下式确定协方差 矩阵的参数为对角矩阵：

[0061] $\mathrm{diag}(\Sigma_{k})=\mathrm{mELU}(W_{\Sigma_{k}}\Phi(x)+b_{\Sigma_{k}})$

[0062] 其中，mELU是修正的指数线性单元，定义为：

[0063] $\mathrm{mELU}(z)=\begin{cases}\propto(e^{z}-1)+1 & z\leq 0\\ z+1 & z>0\end{cases}$

[0064] 以强制执行正值。在一个实施例中，对角参数化假定仿真器参数θ的维度 之间是独立的。在一个实施方案中，如果混合中分量的数量足够大，则这 不是过度限制。

[0065] 在一个实施例中，用于混合物密度网络的全组参数为

[0066] $\varphi=(W_{\alpha},b_{\alpha},\{W_{\mu_{k}}\ ,\ b_{\mu_{k}},\ W_{\Sigma_{k}},\ b_{\Sigma_{k}}\}_{k=1}^{K})$

[0067] 在一个实施例中，神经网络特征可以用于对密度建模。在一个实施例 中，本文描述的贝叶斯推理技术可以使用神经网络特征来创建与混合密度 网络相似的模型。在一个实施例中，对于具有两个全连接层的前馈神经网 络，特征采用以下形式：

[0068] $\Phi(x)=\sigma(W_{2}(\sigma(W_{1}x+b_{1}))+b_{2})$

[0069] 其中σ(•)是S型函数(sigmoid function)；我们在本文所述的实验中使用σ (•)＝tanh(•)。在一个实施例中，该网络结构被用于实验中并且与下面描述 的准蒙特卡洛随

机特征进行比较。

[0070] 在一个实施例中，将准蒙特卡洛随机特征用于对密度建模。在一个实 施例中，本文描述的贝叶斯推理技术使用随机傅里叶特征代替神经网络来 对混合密度进行参数化。在一个实施例中，有多个理由使之成为一个好的 选择：1)随机傅里叶特征-QMC特征是其中的一种特殊类型-近似可能无 限的希尔伯特空间，其性质由相关内核的选择来定义。这样，在一个实施 例中，可以便利地通过选择合适的正半定核来合并关于功能空间的性质的 先验信息。2)在一个实施例中，以$\mathcal{O}(1/\sqrt{s})$阶(order)近似收敛到原始希 尔伯特空间，其中s是特征的数量，因此独立于输入维数；3)在一个实施 例中，我们实验验证了具有随机傅立叶特征的混合密度对于不同的初始化 更为稳定，并且在大多数情况下收敛于相同的局部最大值。

[0071] 在一个实施例中，随机傅立叶特征通过有限尺寸特征Φ(x)的点积k(τ) $\approx \Phi(x)^{T}\Phi(x')$近似于移位不变核k(τ)，其中τ＝||x-x′||。在一个实施例中，这 可以通过首先应用下面所述的Bochner定理[33]来说明：

[0072] 定理1(Bochner定理)与正有限测度dμ(ω)相关的位移不变核k(τ)，$\tau \in \mathbb{R}^{D}$可以用其傅立叶变换表示为，

[0073] $$k(\tau)=\int_{\mathbb{R}^D} e^{-i\omega\cdot\tau}\, d\mu(\omega)。$$

[0074] 在一个实施例中，当μ具有密度$\mathcal{K}(\omega)$时，$\mathcal{K}$表示正半定k的频谱分布， 并且在这种情况下，k(τ)和$\mathcal{K}(\omega)$是傅立叶对偶：

[0075] $$k(\tau)=\int \mathcal{K}(\omega) e^{-i\omega\cdot\tau} d\omega。$$

[0076] 在一个实施例中，用带有N个样本的蒙特卡洛估计来对上述方程进行 近似处理，得到：

[0077] $$k(\tau)=\frac{1}{N}\sum_{n=1}^{N}(e^{-i\omega_n x})(e^{-i\omega_n x'})$$

[0078] 其中，ω从密度$\mathcal{K}(\omega)$中采样。

[0079] 在一个实施例中，使用欧拉方程(e^{-ix}＝cos(x)-i sin(x))，将特征恢复 为：

[0080]
$$\Phi(x)=\frac{1}{\sqrt{N}}\,[\cos(\omega_1 x+b_1),\ldots,\cos(\omega_n x+b_n),$$
$$-i\cdot\sin(\omega_1 x+b_1),\ldots,-i\cdot\sin(\omega_n x+b_n)],$$

[0081] 其中，引入偏置项b_i的目的是旋转投影并在捕获正确的频率时提供更大的 灵活性。

[0082] 在一个实施例中，该近似与位移不变核一起使用，以通过为该问题选 择合适的核来在引入现有知识中提供灵活性。在一个实施例中，例如，可 以使用具有ω～(0,$2\sigma^{-2}$I)和$b\sim\mathcal{U}[-\pi,\pi]$的上述特征来对RBF内核进行近似。σ 是一个与内核长度尺度相对应的超参数，通常通过交叉验证进行设置。

[0083] 在一个实施例中，采用准蒙特卡洛策略来对频率进行采样。在一个实 施例中，使用了与标准蒙特卡洛技术相比具有更好的收敛速度和更低的近 似误差的霍尔顿序列。在

本文中，频率的术语函数可用于指代选定的傅立 叶特征，包括随机选择的傅立叶特征、使用蒙特卡洛或准蒙特卡洛技术选 择的傅立叶特征、以及基于霍尔顿序列选择的傅立叶特征。

[0084] 在一个实施例中，恢复后验。在一个实施例中，如从以上等式可以推 断，如果建议先验与期望先验不同，则系统通过以比率$p(\theta)/\tilde{p}(\theta)$对其加权 来调整后验。

[0085] 在一个实施例中，先验是一致的，或者在有限的支持下(在一个范围 内定义，在其他地方为零)，或者在各处使用不适当的恒定值。因此，在一 个实施例中，

[0086] $\hat{p}(\theta|x=x^r)\propto\frac{q_\phi(\theta|x^r)}{\tilde{p}(\theta)}$。

[0087] 在一个实施例中，当建议先验是高斯时，本文描述的贝叶斯推理技术 能够分析地计算混合和单个高斯之间的划分。在一个实施例中，由于 $q_\phi(\theta|x)$是高斯和$\tilde{p}(\theta)\sim\mathcal{N}(\theta|\mu_0,\Sigma_0)$的混合，所以解(solution)由下式给出：

[0088] $\hat{p}(\theta|x=x^r)=\sum_k \alpha'_k((\theta|\mu'_k,\Sigma'_k)$

[0089] 其中，

[0090] $\Sigma'_k=(\Sigma_k^{-1}-\Sigma_0'^{-1})^{-1}$

[0091] $\mu'_k=\Sigma_k^{-1}(\Sigma_k^{-1}\mu_k-\Sigma_0^{-1}\mu_0)$

[0092] $$\alpha'_k=\frac{\alpha_k\exp(-\frac{1}{2}\lambda_k)}{\alpha_{k'}\exp(-\frac{1}{2}\lambda_k')}$$

[0093] 以及系数λ_k通过下式给出：$\lambda_k=\log\det\Sigma_k-\log\det\Sigma_0-\log\det\Sigma'_k+\mu_k^T\Sigma_k^{-1}\mu_k-\mu_0^T\Sigma_0^{-1}\mu_0-\mu_k'^T\Sigma_k'^{-1}\mu'_k$。

[0094] 在一个实施例中，典型问题中的状态和动作对的轨迹可以是长序列，这使得模型的输入维数过大且计算昂贵。在一个实施例中，代替向模型输 入原始状态和动作序列，系统首先计算足够的统计量。在一个实施例中， 形式上，$x=\psi(S,A)$其中$S=\{s^t\}_{t=1}^T$，以及$A=\{a^t\}_{t=1}^T$是从t＝1到T的状态和 动作序列。在一个实施例中，对于时间序列或轨迹数据有许多选项以供充 足的统计信息，例如每个时间序列的均值、对数方差和自相关以及在两个 时间序列之间的互相关。在一个实施例中，系统例如利用自动编码器从数 据中学习这些。在一个实施例中，本文描述的贝叶斯推理技术使用经常应 用于诸如Lotka-Volterra模型的随机动态系统的统计。

[0095] 在一个实施例中，将$\tau=\{s^t-s^{t-1}\}_{t=1}^T$定义为近期状态和当前状态之间 的差，统计

[0096] $\psi(S,A)=(\{\langle\tau_i,A_j\rangle\}_{i=1,j=1}^{D_s,D_a},E[\tau],\mathrm{Var}[\tau])$

[0097] 其中，D_s是状态空间的维数，D_a是动作空间的维数，$\langle\bullet,\bullet\rangle$为点积的符号，$E[\bullet]$

是期望,以及Var[•]是方差。

[0098] 在一个实施例中,OpenAI Gym中可用的Fetch机器人用于执行推动 和滑动任务。在一个实施例中,使用闭环场景,其中手臂总是在整个桌子 的范围内,因此,它可以根据从环境接收的输入来校正其轨迹。在一个实 施例中,使用了更困难的开环情况,其中机器人在将冰球推到其期望的目 标时通常仅具有一次射击。在一个实施例中,对于这两个任务,物体和表 面的摩擦系数在最终结果中起主要作用,因为它们与施加每个力后物体经 过的距离严格相关。在一个实施例中,非常低的摩擦系数意味着物体随着 其更容易滑动而变得难以控制,并且非常高的摩擦系数意味着需要施加更 多的力以使物体运动。

[0099] 图2示出了根据一个实施例的执行获取-滑动(fetch-slide)任务的机 器人200的示例,其中机器人对表的访问受到限制。在一个实施例中,机 器人200附接到基座202。在一个实施例中,第一铰接关节204将基座202 连接到第一臂206。在一个实施例中,第一臂206经由第二铰接关节208 连接到第二臂210。在一个实施例中,第二臂210通过第三铰接关节212 连接到探针214。在一个实施例中,控制计算机系统指导伺服电动机、气 动致动器或液压致动器的操作,从而控制铰接关节的运动。在一个实施例 中,控制计算机系统实现了对获取-滑动问题的解决方案,其中机器人200 试图将冰球(puck)216滑动到目标220。在一个实施例中,机器人200 不能完全访问桌子218,因此,机器人可能无法反复尝试成功完成任务, 因为冰球可能变得无法触及。

[0100] 图3示出了根据一个实施例的执行获取-推动(fetch-push)任务的机 器人300的示例,其中该机器人可以完全访问表318。在一个实施例中, 机器人200附接到基座302。在一个实施例中,第一铰接关节304将基座 302连接到第一臂306。在一个实施例中,第一臂306经由第二铰接关节 308连接到第二臂310。在一个实施例中,第二臂310通过第三铰接关节 312连接到探针314。在一个实施例中,控制计算机系统指导伺服电动机、气动致动器或液压致动器的操作,从而控制铰接关节的运动。在一个实施 例中,控制计算机系统实现了对获取-滑动问题的解决方案,其中机器人 300试图将冰球316推到目标320。在一个实施例中,机器人300可以完全 访问桌子318,这允许机器人重新定位,并多次尝试成功完成任务。

[0101] 在一个实施例中,本文所述的贝叶斯推理技术通过估计对于车-杆问题 的未知仿真参数来说明。图4示出了根据实施例的执行车-杆平衡任务的机 器人的示例,其中,机器人控制车的运动。在一个实施例中,系统400包 括具有轮子404和406的车402,轮子404和406允许车沿着表面408移 动。在一个实施例中,杆410使用枢轴412连接到车402。枢轴412允许 杆落在垂直于车的移动的轴线上。在一个实施例中,计算机控制系统能够 移动车(如图4的示例中所示向左和向右)以保持杆竖直。在一个实施例 中,质量块414位于杆的顶部。在一个实施例中,用于使杆保持直立的控 制参数主要取决于质量414和杆的长度。

[0102] 在一个实施例中,通过向车402的左侧或右侧施加力来平衡安装在车 402上的杆410。在一个实施例中,杆410的质量414和长度都不可用,并 且我们使用本文描述的贝叶斯推理技术来获得这些参数的后验。在一个实 施例中,系统对两个参数使用一致先验,并遵循r1-zoo策略来收集1000 个仿真以训练系统。在一个实施例中,随着模型的训练,系统可以收集具 有正确参数的10个轨迹以仿真真实观察。图5示出了根据一个实施例的用 于车-杆问题的极点长度的后验的示例500。图6示出了根据一个实施例的 车-杆问题的质点的后验的示例600。在一个实施例中,质量和杆长度表现 出统计依赖性,该统计依赖性产生对其

值的多种解释。在一个实施例中，杆可以具有较低的质量和较长的长度，或者较高的质量和较短的长度。在 一个实施例中，该系统能够对提供准确地表示问题的双峰不确定性的密度 的后验的多峰性质进行恢复。

[0103] 在一个实施例中，系统使用利用通过推理方法获得的后验的策略来执 行域随机化。在一个实施例中，给定从仿真参数$\hat{p}(\theta|x=x^r)$获得的后验，系 统将目标最大化，

[0104] $J(\beta)=\mathbb{E}_\theta[\mathbb{E}_\eta[\sum_{t-0}^{T-1}\gamma^{(t)}r(s_t,a_t)|\beta]]$

[0105] 其中，相对于策略参数β，$\theta\sim\hat{p}(\theta|x=x^r)$。在一个实施例中，后验是高斯的 混合，因此通过伴随α上的分布对混合分量进行采样以获得分量k，然后 对相应的高斯$\mathcal{N}(\theta|\mu_k,\Sigma_k)$进行采样来对第一期望进行近似处理。

[0106] 在一个实施例中，对恢复的后验的准确性进行如下验证。在一个实施 例中，我们针对不同问题和方法获得的后验的质量执行第一分析。在一个 实施例中，本文描述的贝叶斯推理技术使用混合模型下目标的对数概率作 为度量，定义为$\log p(\theta_*||x=x^r)$，其中θ_*是参数的实际值。在一个实施方案 中，我们比较拒绝-ABC作为基线，∈-Free其提供混合模型作为后验，以及 本文所述的贝叶斯推理技术，其使用每层具有24个单元的两层神经网络或 本文所述的具有准随机傅里叶特征的贝叶斯推理技术。

[0107] 在一个实施例中，使用Matern 5/2内核，并且通过交叉验证来建立采 样精度σ。在一个实施例中，针对不同的问题使用了三个不同的仿真器：OpenAI Gym、PyBullet 2和MuJoCo。在一个实施例中，提出了以下问题：车杆(Gym)、摆锤(Gym)、山地车(Gym)、机器人(Gym)、料斗(PyBullet)、获取-推动(MuJoCo)和获取-滑动(fetch-slide)(MuJoCo)。在一个实施 例中，对于方法和参数的所有配置，训练和测试执行五次对数概率取平均 值和计算标准差。在一个实施例中，为了提取真实观察值，用实际参数仿 真环境十次，并且将结果的平均值用于获得x^r。在一个实施例中，通过针 对最多200个时间步长或直到情节(episode)结束执行转出来收集足够的 统计信息。

[0108] 图8示出了根据实施例的针对各种方法和问题的各种对数预测的概率。在一个实施例中，表800示出了对数概率的结果(均值和标准差)。在一个 实施例中，与拒绝ABC相比，本文描述的具有RFF或神经网络特征的贝 叶斯推理技术通常提供更高的对数概率和更低的标准偏差。在一个实施例 中，这表明由本文描述的贝叶斯推理技术提供的后验在该参数的正确值附 近更趋于峰值和集中。在一个实施例中，与∈-Free相比，结果在均值方面 是等效的，但是本文所述的贝叶斯推理技术通常在该方法的多次运行中提 供较低的标准偏差，表明其比∈-Free更稳定。将本文所述的贝叶斯推理技 术的实施例与RFF和NN进行比较，在大多数情况下，RFF特征导致较高 的对数概率，但是使用神经网络的版本具有较低的标准偏差。

[0109] 在一个实施例中，结果表明，当估计仿真参数上的后验分布时，本文 描述的具有RFF或NN的贝叶斯推理技术是优越的。然而，在一个实施例 中，对于下面分析的机器人问题，本文描述的具有RFF的贝叶斯推理技术 提供了优于其他测试方法的明显优越的结果，并且比本文描述的具有NN 的贝叶斯推理技术提供了更好的结果。这由图7中的后验图示出。图7示 出了根据一个实施例的针对获取-滑动(fetch-slide)问题通过不同的方法进

行恢复的后验的示例700。在一个实施例中，本文描述的使用RFF的贝叶 斯推理技术明显地在真实摩擦力值附近更趋于峰值和集中。

[0110] 在一个实施例中，通过比较策略在一致先验和学习后验上的性能来评 估策略的鲁棒性。在一个实施例中，评估是在仿真器设置的预定范围内完 成的，并且在图9-图12中示出了针对每个参数值的平均奖励。图9示出 了根据一个实施例的通过对长度参数的先验进行随机化而训练的车-杆策 略的累积奖励的示例900。图10示出了根据一个实施例的通过针对质杆参 数的先验进行随机化而训练的车杆策略的累积奖励的示例1000。图11示 出了根据实施例的通过针对长度参数的后验进行随机化而训练的车-杆策 略的累积奖励的示例1100。图12示出了根据一个实施例的通过针对质杆 (masspole)参数的后验进行随机化而训练的车-杆策略的累积奖励的示例 1200。

[0111] 在一个实施例中，在一组实验中，车-杆问题用于说明后验随机化的好 处。在一个实施例中，训练了两个策略，如图8所示针对长度和质杆对一 致先验进行第一随机化，以及基于本文所述的具有RFF的贝叶斯推理技术 提供的后验进行第二随机化。在一个实施例中，两种情况都使用PPO以针 对2M时间步长从先验和后验的100个样本中训练策略。在一个实施例中， 结果在图9-图12中示出，采用相应的标准偏差对多次运行取平均值。在 一个实施例中，后验的随机化产生明显更鲁棒的策略，特别是在实际参数 值处。在一个实施例中，对于较低的长度值和较高的质杆值，性能的降低 是显著的。在一个实施例中，由于本文描述的贝叶斯推理技术的动态增加， 当长度较短时，更难控制杆位置。在一个实施例中，当质量基本上增加到 超过其实际训练的值时，控制器努力保持杆平衡。在一个实施例中，如曲线图(plots)中的较低方差所指示的，在跨多个运行中，使用后验学习的 策略似乎更加稳定。

[0112] 在一个实施例中，目标是使用本文描述的贝叶斯推理技术来恢复在摩 擦系数上的后验的良好近似。在一个实施例中，用于数据生成目的的具有 固定摩擦系数的策略将使用具有使用HER采样200个时期(每个时期100 时期/展示)的经验的DDPG来训练。在一个实施例中，使用Adam以0.001 的步长来完成梯度更新。在一个实施例中，以不同的摩擦系数多次运行该 策略，以便对似然函数进行近似处理并恢复整个仿真参数的后验。在一个 实施例中，使用动态模型，本文描述的贝叶斯推理技术使用一些数据来恢 复期望的后验，该数据采样于我们想要从中学习动态的环境。在一个实施 例中，使用前述设置进行训练，但是代替使用固定的摩擦系数，当新的情 节开始时，从其各自的分布中采样新的。

[0113] 根据一个实施例，来自两个任务的结果在图13和图14。图13示出了 用于获取-滑动问题的策略的示例1300。图14示出了根据实施例的用于获 取-推动问题的策略的示例1400。在一个实施例中，一致先验在推动任务 上运转良好。在一个实施例中，发生这种情况是因为如果出现问题，机器 人有机会校正其轨迹。在一个实施例中，在获取-滑动问题中，机器人暴露 于涉及不同动态的广泛场景中，因此机器人随后可以使用环境的输入来执 行校正动作，并且仍然能够实现目标。在一个实施例中，滑动任务使用一 致先验，这导致机器人取得不良性能。在一个实施例中，发生这种情况是 因为机器人没有校正其轨迹的选择。在一个实施例中，本文描述的贝叶斯 推理技术是有用的，因为它们恢复了围绕真实参数的非常高密度的分布， 并因此导致更好的总体控制策略。

[0114] 在一个实施例中，本文提出了机器人仿真参数的贝叶斯处理，结合了 用于策略搜

索的域随机化。在一个实施例中，本文描述的贝叶斯推理技术 使用被集成到框架中的黑匣子生成模型或仿真器。在一个实施例中，还可 以提供先验分布并将其结合到模型中以计算参数上的多峰后验。在一个实 施例中，本文描述的方法与用于贝叶斯推理的其他现有技术的无似然性方 法具有可比性，但是对于不同的初始化更稳定，并且在恢复真实后验时跨多个运行更稳定。在一个实施例中，与在一致先验随机化下训练的策略相 比，具有后验的域随机化使得在多个参数值上的策略更鲁棒。

[0115] 在一个实施例中，本文所述的贝叶斯推理技术可以应用于广泛的问题， 其中仿真器利用全套参数化来表示现实。在一个实施例中，本文描述的框 架可以集成在涉及仿真器的许多其他问题中。

[0116] 图15示出了根据一个实施例的过程1500的示例，作为该过程由计算 机系统的处理器执行的结果，使本文所述的贝叶斯推理技术估计仿真参数 的分布，当该仿真参数应用于仿真时，使仿真产生期望结果。在一个实施 例中，在框1502处，计算机系统指导诸如机器人手臂之类的机器人执行任 务。在一个实施例中，在框1504处，计算机系统获得任务的结果。在一个 实施例中，可以经由位置传感器、照相机、运动检测器或其他传感器来获 得结果，并且该结果可以表示对象的位置或轨迹或物理量或值。在一个实 施例中，在框1506处，计算机系统获得表示预测以产生所获得的结果的仿 真参数的分布的估计。在一个实施例中，分布是恒定值。在一个实施例中， 分布是恒定值，以及在其他任何地方是零。在一个实施例中，从过程1500 的先前执行获得分布。

[0117] 在一个实施例中，在框1508处，计算机系统根据在1506处获得的分 布为仿真器生成参数集(sets of parameters)。在一个实施例中，在框1510 处使用确定的参数集中的每一个来运行仿真器。在一个实施例中，对于每 组参数，仿真器产生相应的结果。在一个实施例中，在框1512处，使用所 得的参数结果对来估计密度。在一个实施例中，使用一组如上所述的傅立 叶特征对密度进行建模。在一个实施例中，在框1514处，计算机系统使用 估计的密度来计算预测的参数的分布，以产生在框1504处观察到的结果。

[0118] 图16示出了根据一个实施例的并行处理单元(“PPU”)1600。在一个 实施例中，PPU1600配置有机器可读代码，该机器可读代码如果由PPU执 行，则使得PPU执行贯穿本公开描述的一些或全部过程和技术。在一个实 施例中，PPU 1600是在一个或更多个集成电路设备上实现的多线程处理器， 并且利用多线程作为设计用于处理在多个并行线程上的计算机可读指令 (也称为机器可读指令或简称为指令)的等待时间隐藏技术。在一个实施 例中，线程是指执行线程，并且是配置成由PPU 1600执行的一组指令的 实例。在一个实施例中，PPU 1600是配置成实现图形渲染管线的图形处理 单元(“GPU”)，以用于处理三维(“3D”)图形数据以便生成二维(“2D”) 图像数据以在诸如液晶显示器(LCD)设备之类的显示设备上显示。在一 个实施例中，PPU1600用于执行诸如线性代数运算和机器学习运算之类的 计算。图16仅出于说明性目的示出了示例并行处理器，并且应被解释为在 本公开的范围内设想的处理器体系结构的非限制性示例，并且可以采用任 何适当的处理器来对其进行补充和/或替代。

[0119] 在一实施例中，一个或更多个PPU配置成加速高性能计算(“HPC”)、数据中心和机器学习应用。在一个实施例中，PPU 1600配置成加速深度学 习系统和应用程序，包括以下非限制性示例：自主车辆平台、深度学习、 高精度语音、图像、文本识别系统、智能视频分

析、分子仿真、药物发现、疾病诊断、天气预报、大数据分析、天文学、分子动力学仿真、金融建模、机器人技术、工厂自动化、实时语言翻译、在线搜索优化以及个性化用户 推荐等。

[0120] 在一个实施例中，PPU 1600包括输入/输出(“I/O”)单元1606、前端 单元1610、调度器单元1612、工作分配单元1614、集线器1616、交叉开 关(“Xbar”)1620、一个或更多个通用处理集群(“GPC”)1618和一个或 更多个分区单元1622。在一个实施例中，PPU 1600通过一个或更多个高 速GPU互连1608连接到主机处理器或其他PPU 1600。在一个实施例中，PPU 1600经由系统总线1602连接到主机处理器或其他外围设备。在一个 实施例中，PPU 1600连接到包括一个或更多个存储设备1604的本地存储 器。在一个实施例中，本地存储器包括一个或更多个动态随机存取存储器 (“DRAM”)设备。在一实施例中，一个或更多个DRAM设备配置成和/ 或可配置成高带宽存储器(“HBM”)子系统，并且每个设备内堆叠有多个DRAM裸晶(die)。

[0121] 高速GPU互连1608可以指代系统使用其来缩放基于有线的多通道通 信链路，并且包括与一个或更多个CPU结合的一个或更多个PPU 1600，支持PPU 1600和CPU之间的高速缓存一致性，以及CPU主控。在一个实 施例中，数据和/或命令由高速GPU互连1608通过集线器1616传送到PPU 1600的其他单元/从PPU 1600的其他单元传出，PPU 1600的其他单元诸如 一个或多个复制引擎、视频编码器、视频解码器、电源管理单元，以及未 在图16中明确示出的其他组件。

[0122] 在一个实施例中，I/O单元1606配置成通过系统总线1602从主机处 理器(图16中未示出)发送和接收通信(例如，命令、数据)。I/O单元 1606直接通过系统总线1602或通过一个或更多个中间设备(例如存储器 桥)与主机处理器通信。在一个实施例中，I/O单元1606可以经由系统总 线1602与一个或更多个其他处理器(例如一个或更多个PPU 1600)通信。 在一个实施例中，I/O单元1606实现外围组件互连快速(“PCIe”)接口，用于通过PCIe总线进行通信。在一个实施例中，I/O单元1606实现用于 与外部设备通信的接口。

[0123] 在一个实施例中，I/O单元1606对经由系统总线1602接收的分组进 行解码。在一个实施例中，至少一些分组表示被配置为使PPU1600执行各 种操作的命令。在一个实施例中，I/O单元1606如命令所指定的那样将解 码的命令发送到PPU 1600的各种其他单元。在一个实施例中，命令被发 送到前端单元1610和/或被发送到集线器1616或PPU 1600的其他单元，例如一个或更多个复制引擎、视频编码器、视频解码器、电源管理单元等 等(在图16中未明确示出)。在一个实施例中，I/O单元1606配置成在PPU 1600的各个逻辑单元之间和之中路由通信。

[0124] 在一个实施例中，由主机处理器执行的程序在缓冲器中对命令流进行 编码，该缓冲器将工作负载提供给PPU 1600以进行处理。在一个实施例 中，工作负载包括指令和要由那些指令处理的数据。在一个实施例中，缓 冲器是主机处理器和PPU 1600两者都可以访问(例如，读/写)的存储器 中的区域-主机接口单元可以配置成访问经由I/O单元1606通过系统总线 1602传输存储器请求而连接到系统总线1602的系统存储器中的缓冲器。在一个实施例中，主机处理器将命令流写入缓冲器，然后将指向命令流开 始的指针发送至PPU 1600，从而前端单元1610接收指向一个或更多个命 令流的指针并管理一个或更多个流，从这些流中读取命令，并向PPU1600 的各个单元转发命令。

[0125] 在一个实施例中，前端单元1610耦合到调度器单元1612，该调度器 单元1612配置

各种GPC 1618以处理由一个或更多个流定义的任务。在一 个实施例中,调度器单元1612配置成跟踪与由调度器单元1612管理的各 种任务有关的状态信息,其中状态信息可以指示任务被分配给哪个GPC 1618,该任务是活跃的还是不活跃的,与任务相关的优先级,等等。在一 实施例中,调度器单元1612管理在一个或更多个GPC1618上的多个任务 的执行。

[0126] 在一个实施例中,调度器单元1612耦合到工作分配单元1614,该工 作分配单元配置成分配要在GPC 1618上执行的任务。在一个实施例中, 工作分配单元1614跟踪从调度器单元1612接收到的多个预定任务和工作 分配单元1614为每个GPC 1618管理未决任务池和活跃任务池。在一个实 施例中,未决任务池包括多个时隙(例如32个时隙),这些时隙包含分配 的任务由特定的GPC 1618处理;活跃任务池可包括用于由GPC 1618主动 处理的任务的多个时隙(例如4个时隙),以便当GPC 1618完成任务的执 行时,该任务将从GPC 1618的活动任务池中逐出并且未决任务池中的其 他任务之一被选择并安排在GPC 1618上执行。在一个实施例中,如果活 跃任务在GPC 1618上是空闲的,例如在等待要解决的数据依赖性时,则 将活动任务从GPC 1618中逐出并奖励到未决任务池,同时选择未决任务 池中的另一个任务并安排其在GPC 1618上执行。

[0127] 在一个实施例中,工作分配单元1614经由XBar 1620与一个或更多 个GPC 1618通信。在一个实施例中,XBar 1620是将PPU 1600的许多单 元耦合到PPU 1600的其他单元的互连网络,并可以配置为将工作分配单 元1614耦合到特定的GPC168。尽管未明确显示,但PPU 1600的一个或 更多个其他单元也可以通过集线器1616连接到XBar1620。

[0128] 这些任务由调度器单元1612管理,并由工作分配单元1614分配给 GPC1618。GPC1618配置为处理任务并生成结果。结果可能会被GPC 1618 中的其他任务消耗,通过XBar 1620路由到其他GPC 1618或存储在内存 1604中。结果可通过分区单元1622写入存储器1604,分区单元实现用于 从存储器1604读取数据和写入数据到存储器1604的存储器接口。结果可 以经由高速GPU互连1608传输到另一个PPU 1604或CPU。在一个实施 例中,PPU 1600包括数量为U的分区单元1622,其等于耦合到PPU 1600 的分离且不同的存储设备1604的数量。下面将结合图11更详细地描述分 区单元1622。

[0129] 在一个实施例中,主机处理器执行驱动程序内核,该驱动程序内核实 现应用程序编程接口(“API”),该应用程序编程接口使在主机处理器上执 行的一个或更多个应用程序以调度用于在PPU 1600上执行的操作。在一 个实施例中,PPU 1600同时执行多个计算应用程序,并且PPU 1600为多 个计算应用程序提供隔离、服务质量(“QoS”)和独立的地址空间。在一 个实施例中,应用程序生成指令(例如,以API调用的形式),该指令使 驱动程序内核生成一个或更多个任务以由PPU 1600执行,并且驱动程序 内核将任务输出到由PPU1600处理的一个或更多个流。在一个实施例中, 每个任务包括一个或更多个相关线程组,其可以被称为线程束(warp)。在 一个实施例中,线程束包括可以并行执行的多个相关线程(例如32个线程)。在一个实施例中,协作线程(cooperating threads)可以指多个线程,包括执行任务并且通过共享存储器交换数据的指令。根据一个实施例,结合图18A 更详细地描述了线程和协作线程。

[0130] 图17示出了根据一个实施例的GPC 1700,例如图16的PPU 1600中 示出的GPC。在一个实施例中,每个GPC 1700包括用于处理任务的多个 硬件单元,并且每个GPC 1700包括管线管理器1702、光栅前操作单元 (“PROP”)1704、光栅引擎1708、工作分配交叉开关

(“WDX”)1716、存储器管理单元(“MMU”)1718、一个或更多个数据处理群集(“DPC”) 1706，以及部件的任何合适组合。将理解的是，图17的GPC 1700可以配 置为包括除了图17中所示的单元之外的其他硬件单元，也可以包括代替图 17中所示的单元的其他硬件单元。

[0131] 在一个实施例中，GPC 1700的操作由管线管理器1702控制。管线管 理器1702管理一个或更多个DPC 1706的配置，以处理分配给GPC1700 的任务。在一个实施例中，管线管理器1702将一个或更多个DPC 1706中 的至少一个配置为实现图形渲染管线的至少一部分。在一个实施例中，DPC 1706配置成执行在可编程流传输多处理器(“SM”)1714上执行的顶点着 色程序。在一个实施例中，管线管理器1702配置成将从工作分配接收的分 组路由到GPC 1700内的适当逻辑单元，并且可以将一些分组路由到PROP 1704和/或光栅引擎1708中的固定功能硬件单元，而其他的分组路由到 DPC 1706，以供基元引擎1712或SM1714处理。在一实施例中，管线管 理器1702配置一个或更多个个DPC 1706中的至少一个，以实现神经网络模型和/或计算管线。

[0132] 在一个实施例中，PROP单元1704配置成将由光栅引擎1708和DPC 1706生成的数据路由到存储器分区单元中的光栅操作(“ROP”)单元，如 上面更详细地描述的。在一个实施例中，PROP单元1704配置成执行用于 颜色混合的优化、组织像素数据、执行地址转换等等。在一个实施例中， 光栅引擎1708包括配置为执行各种光栅操作的多个固定功能硬件单元，并 且光栅引擎1708包括设置引擎、粗糙光栅引擎、剔除引擎、剪切引擎、精 细光栅引擎、瓦片合并引擎及其任何合适的组合。在一个实施例中，设置 引擎接收变换后的顶点并生成与由这些顶点定义的几何本基元相关的平面 方程；平面方程式被传送到粗糙光栅引擎以生成图元的覆盖率信息(例如， 瓦片的x、y覆盖率掩模)；粗糙光栅引擎的输出将传输到剔除引擎，在剔 除引擎中，将与z测试失败的图元相关联的片段剔除，并将其传输到剪切 引擎，在剪切引擎中，位于视锥范围之外的片段将被剪切。在一个实施例 中，将经过剪裁和剔除的片段传递给精细光栅引擎，以基于由设置引擎生 成的平面方程式生成像素片段的属性。在一个实施例中，光栅引擎1708 的输出包括将由任何合适的实体(例如，由在DPC 1706内实现的片段着 色器)处理的片段。

[0133] 在一个实施例中，GPC 1700中包括的每个DPC 1706包括M-管线 (M-Pipe)控制器(“MPC”)1710；基元引擎1712；一个或更多个SM 1714； 及其任何合适的组合。在一个实施例中，MPC 1710控制DPC 1706的操作， 将从管线管理器1702接收的分组路由到DPC 1706中的适当单元。在一个 实施例中，与顶点相关联的分组被路由到基元引擎1712，基元引擎1712配置为从存储器中获取与顶点关联的顶点属性；相反，以将与着色器程序 相关联的分组可发送到SM 1714。

[0134] 在一个实施例中，SM1714包括可编程流处理器，其配置成处理由多 个线程表示的任务。在一个实施例中，SM1714是多线程的并且配置成同 时执行来自特定线程组的多个线程(例如32个线程)，以及实现SIMD(单 指令、多数据)架构，其中一组线程(例如，线程束)中的每个线程配置 成基于相同的指令集来处理不同的数据集。在一个实施例中，线程组中的所有线程执行相同的指令。在一个实施例中，SM1714实现SIMT(单指令 多线程)架构，其中一组线程中的每个线程配置成基于相同的指令集来处 理不同的数据集，但是其中允许线程组中的各个线程在执行期间发散。在 一个实施例中，为每个线程束维护程序计数器、调用栈和执行状态，从而 当线程束中的线程发散时，实现线程束和线程束内的串行执行之间

的并发。在另一个实施例中,为每个单独的线程维护程序计数器、调用堆栈和执行 状态,从而使得在线程束和线程束之间的所有线程之间具有相等的并发性。在一个实施例中,为每个单独的线程维持执行状态,并且执行相同指令的 线程可以被收敛并并行地执行以获得最大效率。在一个实施例中,下面将 更详细地描述SM1714。

[0135] 在一个实施例中,MMU1718在GPC1700与存储器分区单元之间的 接口,并且MMU1718提供虚拟地址到物理地址的转换、存储器保护、和 储存器请求的仲裁。在一个实施例中,MMU 1718提供一个或更多个转换 后备缓冲器("TLB"),用于将虚拟地址转换为存储器中的物理地址。

[0136] 图18示出了根据一个实施例的PPU的存储器分区单元。在一个实施 例中,存储器分区单元1800包括光栅操作("ROP")单元1802;二级("L2") 高速缓存1804;存储器接口1806;及其任何合适的组合。存储器接口1806 耦合到存储器。存储器接口1806可以实现32、64、128、1024位数据总线 等,用于高速数据传输。在一个实施例中,PPU包括U个存储器接口1806, 每对分区单元1800一个存储器接口1806,其中每对分区单元1800连接到 对应的存储设备。例如,PPU最多可以连接到Y个存储设备,例如高带宽 存储堆栈或图形双数据速率,版本5,同步动态随机存取存储器("GDDR5 SDRAM")。

[0137] 在一个实施例中,存储器接口1806实现了HBM2存储器接口,并且 Y等于U的一半。在一个实施例中,HBM2存储器堆栈与PPU位于同一物 理封装上,与传统的GDDR5 SDRAM系统相比,节省了大量的功率和面 积。在一个实施例中,每个HBM2堆栈包括四个存储器管芯,并且Y等于 4,而HBM2堆栈包括每个管芯两个128位通道,总共8个通道和1024位 的数据总线宽度。

[0138] 在一个实施例中,存储器支持单错误校正双错误检测("SECDED") 错误校正码("ECC")以保护数据。ECC为对数据损坏敏感的计算应用程 序提供了更高的可靠性。在PPU处理非常大的数据集和/或长时间运行应 用程序的大规模集群计算环境中,可靠性尤其重要。

[0139] 在一个实施例中,PPU实现多级存储器层次结构。在一个实施例中, 存储器分区单元1800支持统一存储器以为CPU和PPU存储器提供单个统 一虚拟地址空间,从而实现虚拟存储器系统之间的数据共享。在一个实施 例中,追踪PPU对位于其他处理器上的存储器的访问频率,以确保将存储 器页面移动到更频繁地访问页面的PPU的物理存储器。在一个实施例中, 高速GPU互连1608支持地址转换服务,该地址转换服务允许PPU直接访 问CPU的页表并提供由PPU对CPU存储器的完全访问。

[0140] 在一个实施例中,复制引擎在多个PPU之间或在PPU与CPU之间 传送数据。在一个实施例中,复制引擎可以为未被映射到页面表中的地址 生成页面错误,并且存储器分区单元1800然后为页面错误提供服务,将地 址映射到页面表中,之后复制引擎执行传输。在一个实施例中,为多个处 理器之间的多个复制引擎操作固定(即,不可分页)存储器,从而实质上 减少了可用存储器。在一个实施例中,由于硬件页面故障,可以将地址传 递给复制引擎,而不必考虑存储页面是否驻留,并且复制过程是透明的。

[0141] 根据一个实施例,来自图16的存储器或其他系统存储器的数据由存 储器分区单元1800提取并存储在L2高速缓存1804中,该L2高速缓存1804 位于芯片上并且在各种GPC之间共享。在一个实施例中,每个存储器分区 单元1800包括与对应的存储器设备相关联的L2

高速缓存1760的至少一 部分。在一个实施例中,在GPC内的各个单元中实现较低级别的高速缓存。 在一个实施例中,每个SM1840可以实现一级(“L1”)高速缓存,其中L1 高速缓存是专用于特定SM1840的私有存储器,并且获取来自L2高速缓存 1804的数据并将其存储在每个L1高速缓存用于在SM1 840的功能单元中 进行处理。在一个实施例中,L2高速缓存1804耦合到存储器接口1806和 XBar 1620。

[0142] 在一个实施例中,ROP单元1802执行与像素颜色有关的图形光栅操 作,诸如颜色压缩、像素混合等。在一个实施例中,ROP单元1802与光 栅引擎1825一起实施深度测试,从光栅引擎1825的剔除引擎接收与像素 片段相关联的样本位置的深度。在一个实施例中,针对与片段相关联的样 本位置,相对于深度缓冲区中的相应深度的深度进行测试。在一个实施例 中,如果片段通过了针对样本位置的深度测试,则ROP单元1802更新深 度缓冲器,并将深度测试的结果发送至栅格光栅引擎1825。将理解的是, 分区单元1800的数量可以与GPC的数量不同,因此,在一个实施例中, 每个ROP单元1802可以耦合到每个GPC。在一个实施例中,ROP单元1802追踪从不同GPC接收到的分组,并确定通过Xbar将ROP单元1802 生成的结果路由到哪个GPC。

[0143] 图19示出了根据一个实施例的诸如图17的流式多处理器之类的流式 多处理器。在一个实施例中,SM1900包括:指令高速缓存1902;一个或 更多个调度器单元1904;寄存器文件1908;一个或更多个处理核心1910; 一个或更多个特殊功能单元(“SFU”)1912;一个或更多个加载/存储单元 (“LSU”)1914;互连网络1916;共享存储器/L1高速缓存1918;及其任何合适的组合。在一个实施例中,工作分配单元调度任务以在PPU的GPC 上执行,并且每个任务被分配给GPC内的特定DPC,并且,如果该任务 与着色器程序相关联,则该任务被分配给SM 1900。在一个实施例中,调 度器单元1904从工作分配单元接收任务,并管理调度分配给SM 1900的 一个或更多个线程块的指令。在一个实施例中,调度器单元1904调度线程 块以作为并行线程的线程束执行,其中每个线程块被分配至少一个线程束。 在一个实施例中,每个线程束执行线程。在一个实施例中,调度器单元1904 管理多个不同的线程块,将线程束分配给不同的线程块,然后在每个时钟 周期将来自多个不同的协作组的指令分配给各个功能单元(例如,核心 1910、SFU 1912、和LSU 1914)。

[0144] 协作组可以指用于组织通信线程组的编程模型,该模型允许开发人员 表达通信线程的粒度,从而能够表达更丰富、更有效的并行分解。在一个 实施例中,协作启动API支持线程块之间的同步以执行并行算法。在一个 实施例中,常规编程模型的应用提供了用于同步协作线程的单一、简单的 构造:跨线程块的所有线程的屏障(例如,syncthreads()函数)。但是, 编程人员通常希望以小于线程块粒度的方式定义线程组,并在定义的组内 进行同步,以实现更高的性能、设计灵活性以及以集体范围内的功能接口 的形式来复用软件。合作组使编程人员能够在子块(即,小到单个线程) 和多块粒度上明确定义线程组,并在协作组中对线程执行集体操作(例如 同步)。编程模型支持跨软件边界的干净构成,以便库和实用函数可以在其 本地上下文中安全地同步,而不必对收敛进行假设。合作组基元启用了新 的合作并行模式,包括生产者-消费者并行、机会主义并行以及跨整个线程 块网格上的全局同步。

[0145] 在一个实施例中,分派单元1906配置成将指令发送到一个或更多个 功能单元,并且调度器单元1904包括两个分派单元1906,所述两个分派 单元1906使得来自相同线程束

的两个不同指令能够在每个时钟周期中被 分派。在一个实施例中,每个调度器单元1904包括单个分派单元1906或 额外的分派单元1906。

[0146] 在一个实施例中,每个SM1900包括寄存器文件1908,该寄存器文 件1908为SM1900的功能单元提供一组寄存器。在一个实施例中,寄存器 文件1908被划分在每个功能单元之间,使得每个功能单元被分配寄存器文 件1908的专用部分。在一实施例中,寄存器文件1908在由SM 1900执行 的不同线程束之间划分,并且寄存器文件1908为连接到功能单元的数据路 径的操作数提供临时存储。在一个实施例中,每个1SM 900包括多个(L 个)处理核心1910。在一个实施例中,SM1900包括大量(例如128个或 更多个)不同的处理核心1910。在一个实施例中,每个核心1910包括全 管线、单精度、双精度和/或混合精度处理单元(包括浮点算术逻辑单元和 整数算术逻辑单元)。在一个实施例中,浮点算术逻辑单元实现用于浮点算 术的IEEE 754-2008标准。在一个实施例中,核心1910包括64个单精度 (32位)浮点核心、64个整数核心、32个双精度(64位)浮点核心和8 个张量核心。

[0147] 张量核心配置成根据实施例执行矩阵运算。在一个实施例中,一个或 更多个张量核被包括在核心1910中。在一个实施例中,张量核心配置成执 行深度学习矩阵算术,例如用于神经网络训练和推理的卷积运算。在一个 实施例中,每个张量核在4×4矩阵上操作并且对矩阵进行乘法和累加操作 D=A×B+C,其中A、B、C和D为4×4矩阵。

[0148] 在一个实施例中,矩阵乘法输入A和B是16位浮点矩阵,并且累加 矩阵C和D是16位浮点或32位浮点矩阵。在一个实施例中,张量核心对 16位浮点输入数据和32位浮点进行累加运算。在一个实施例中,该16位 浮点乘法需要64次运算,并产生全精度乘积,然后使用32位浮点加法将 其与其他中间乘积相加,以进行4×4×4矩阵乘法。在一个实施例中,张 量核心用于执行更大的二维或更高维的矩阵运算,这些运算由这些较小的 元素构成。在一个实施例中,诸如CUDA 9C++API之类的API公开专门 的矩阵加载、矩阵乘法和累加以及矩阵存储操作以有效地使用来自 CUDA-C++程序的张量核心。在一个实施例中,在CUDA级别,线程束级 接口假定l 6×16大小的矩阵跨线程束的所有32个线程。

[0149] 在一个实施例中,执行特定功能(例如,属性评估、倒数平方根等) 的每个SM1900包括M个SFU1912。在一个实施例中,SFU1912包括配置 成遍历分层树数据结构的树遍历单元。在一个实施例中,SFU1912包括配 置成执行纹理图过滤操作的纹理单元。在一个实施例中,纹理单元配置成 从存储器中加载纹理贴图(例如,纹理的2D阵列)以及对纹理图进行采样,以生成采样的纹理值,以供SM1900执行的着色器程序使用。在一个 实施例中,纹理图存储在共享存储器/LI高速缓存中。根据一个实施例,纹 理单元实施纹理操作,例如使用mip-maps(例如,具有细节的变化级别的 纹理图)进行过流操作。在一个实施例中,每个SM1900包括两个纹理单 元。

[0150] 在一个实施例中,每个SM1900包括N个LSU1854,LSU实现在共 享存储器/LI高速缓存1918和寄存器文件1908之间的加载和存储操作。 在一个实施例中,每个SM 1900包括互连网络916,其将每个功能单元连 接到寄存器文件908并且将LSU914连接到寄存器文件908、共享存储器/LI 高速缓存918。在一个实施例中,互连网络1916是可以配置为将任何功能 单元连接到寄存器文件1908中的任何寄存器并将LSU1914连接到寄存器 文件和共享存储器/LI高速缓存1918中的存储器位置的交叉开关。

[0151] 在一个实施例中,共享存储器/L1高速缓存1918是片上存储器的阵 列,其允许

SM1900和基元引擎之间以及SM1900中的线程之间的数据存 储和通信。在一个实施例中,共享存储器/LI高速缓存1918包括128KB的 存储容量,并且在从SM1900到分区单元的路径中。在一个实施例中,共 享存储器/LI缓存1918用于缓存读取和写入。共享存储器/LI高速缓存1918、L2高速缓存和存储器中的一个或多个是后备存储器。

[0152] 在一个实施例中,将数据高速缓存和共享存储器功能组合到单个存储 器块中,为两种类型的存储器访问提供了改进的性能。在一个实施例中, 该容量被不使用共享内存的程序使用或用作高速缓存,例如,如果共享内 存配置为使用一半的容量,纹理以及加载/存储操作可以使用剩余容量。根 据一个实施例,在共享存储器/LI高速缓存1918内的集成使共享存储器/LI 高速缓存1918能够用作用于流传输数据的高吞吐量管道,同时提供对频繁重用的数据的高带宽和低延迟访问。当配置用于通用并行计算时,与图形 处理相比,可以使用更简单的配置。在一个实施例中,固定功能图形处理 单元被绕过,从而创建了更简单的编程模型。在一个实施例中,在通用并 行计算配置中,工作分配单元直接将线程的块分派和分配给DPC。根据一 个实施例,块中的线程执行相同的程序,在计算中使用唯一的线程ID以确 保每个线程生成唯一的结果,使用SM1900执行程序和执行计算,共享存 储器/L1高速缓存1918在线程之间进行通信,以及LSU1914通过共享存储 器/L1高速缓存1918和存储器分区单元1914来读取和写入全局存储器。 在一个实施例中,当被配置用于通用并行计算时,SM 1900写入命令,调 度器单元可以使用该命令来在DPC上启动新工作。

[0153] 在一个实施例中,PPU包括在台式计算机、膝上型计算机、平板计 算机、服务器、超级计算机、智能电话(例如,无线、手持设备)、个人数 字助理(“PDA”)、数码相机、车辆、头戴式显示器、手持式电子设备等中 或与之耦合。在一个实施例中,PPU被封装在单个半导体衬底上。在一个 实施例中,PPU与一个或更多个其他设备(例如附加的PPU、存储器、精 简指令集计算机(“RISC”)CPU)、存储器管理单元(“MMU”)、数模转 换器(“DAC”)等一起包括在片上系统(“SoC”)中。

[0154] 在一个实施例中,PPU可以被包括在包括一个或更多个存储设备的 图形卡上。图形卡可以配置为与台式计算机的主板上的PCIe插槽交互。在 又一个实施例中,PPU可以是主板的芯片组中包括的集成图形处理单元 (“iGPU”)。

[0155] 图20示出了根据一个实施例的可以在其中实现各种架构和/或功能的 计算机系统2000。在一个实施例中,计算机系统2000配置成实现贯穿本 公开描述的各种过程和方法。

[0156] 在一个实施例中,计算机系统2000包括至少一个中央处理单元2002, 该中央处理单元2002连接到使用任何合适的协议(例如PCI(外围组件互 连)、PCI-Express、AGP(加速图形端口)、超传输(HyperTransport)或任 何其他总线或点对点通信协议)实现的通信总线2010。在一个实施例中, 计算机系统2000包括主存储器2004和控制逻辑(例如,实现为硬件、软 件或其组合),并且数据被存储在主存储器2004中,其可以采取随机存取 存储器(“RAM”)的形式。在一个实施例中,网络接口子系统2022向其 他计算设备和网络提供接口,以用于从计算机系统2000接收数据并将数据 从计算机系统2000传输到其他系统。

[0157] 在一个实施例中,计算机系统2000包括输入设备2008、并行处理系 统2012和显示设备2006,显示设备2006可以使用常规CRT(阴极射线管)、 LCD(液晶显示器)、LED(发光二极管)、等离子显示器或其他合适的显 示技术。在一个实施例中,从诸如键盘、鼠标、触摸

板、麦克风等的输入 设备2008接收用户输入。在一个实施例中，每个前述模块可以位于单个半 导体平台上以形成处理系统。

[0158] 在本说明书中，单个半导体平台可以指唯一的统一的基于半导体的集 成电路或芯片。应当注意，术语“单个半导体平台”也可以指具有增加的 连通性的多芯片模块，其仿真芯片上的操作，并且通过利用传统中央处理 单元(“CPU”)和总线实现方式进行了实质性的改进。当然，根据用户的 需求，各个模块也可以单独放置或以半导体平台的各种组合放置。

[0159] 在一个实施例中，以机器可读的可执行代码或计算机控制逻辑算法的 形式的计算机程序被存储在主存储器2004和/或辅助存储器中。如果由一 个或更多个处理器执行，则计算机程序使系统2000能够执行根据一个实施 例的各种功能。主存储器2004、存储器和/或任何其他存储器是计算机可读 介质的可能示例。辅助存储可以指任何合适的存储设备或系统，例如硬盘 驱动器和/或可移动存储驱动器，表现为软盘驱动器、磁带驱动器、光盘驱 动器、数字多功能磁盘(“DVD”)驱动器、记录设备、通用串行总线(“USB”) 闪存。

[0160] 在一个实施例中，各种先前附图的架构和/或功能是在中央处理器 2002、并行处理系统2012、具有中央处理器2002和并行处理系统2012两 者的至少一部分能力的集成电路、芯片集(例如，被设计为工作并作为执 行相关功能的单元出售的集成电路组等)；以及任何合适的集成电路组合的 上下文中实现。

[0161] 在一个实施例中，各种先前附图的架构和/或功能在通用计算机系统、 电路板系统、专用于娱乐目的的游戏控制台系统、应用程序专用系统等的 背景中实现。在一个实施例中，计算机系统2000可以采取台式计算机、膝 上型计算机、平板计算机、服务器、超级计算机、智能电话(例如，无线、 手持设备)、个人数字助理(“PDA”)、数码相机、车辆、头戴式显示器、 手持式电子设备、移动电话设备、电视、工作站、游戏机、嵌入式系统和/ 或任何其他类型的逻辑的形式。

[0162] 在一个实施例中，并行处理系统2012包括多个PPU 2014和相关联 的存储器2016。在一个实施例中，PPU经由互连2018和开关2020或多路 复用器连接到主机处理器或其他外围设备。在一个实施例中，并行处理系 统2012跨(一个或者更多个)PPU 2014分布计算任务，计算任务可以是 并行的，例如，作为跨多个GPU线程块的计算任务分布的一部分。在一个 实施例中，存储器尽管在整个PPU 2014或所有PPU 2014之间共享和整个 PPU 2014或所有PPU 2014由访问(例如，用于读和/或写访问)，尽管相 对于使用本地存储器和驻留在PPU中的寄存器而言，这样的共享存储器可 能招致性能损失。在一个实施例中，通过使用诸如_syncthreads()之类的 命令来同步PPU2014的操作，该命令要求块中的所有线程(例如，跨多个 PPU 2014执行)必须在继续执行之前到达代码的特定执行点。

[0163] 因此，说明书和附图应被认为是说明性的而不是限制性的。但是，很 明显，在不脱离权利要求书所提出的本发明的更广泛精神和范围的前提下，可以对其做出各种修改和改变。

[0164] 其他变型在本公开的精神内。因此，尽管所公开的技术易于进行各种 变构和替代构造，但是某些示出的实施例由此在附图中示出并且已经在上 面进行了详细描述。然而，应当理解，无意将本发明限制为所公开的特定 形式或形式，相反，其意图是涵盖如所附权利要求所定义的属于本发明的精 神和范围内的所有修改、替代构造和等同形式。

[0165] 在描述所公开的实施例的上下文中(特别是在所附权利要求的上下文 中)术语“一”、“一个”和“所述”以及类似指代的使用应被解释为涵盖 单数和复数,除非本文另有说明或与上下文明显矛盾。除非另外指出,否 则术语“包括”、“具有”、“包含”和“含有”应解释为开放式术语(即, 意思是“包括但不限于”)。术语“连接”(未经修改且指的是物理连接),应理解为完全或部分地包含在,附加到或连接在一起,即使有某物介入。本文中数值范围的引用仅旨在用作一种简写方法,除非本文另有说明,否 则分别指代落入该范围内的每个单独值,并且每个单独值都被并入说明书 中,就如同在此单独引用一样。术语“集合”(例如,“项目的集合)”或“子 集”的使用,除非上下文另有说明或与之矛盾,否则应解释为包含一个或 更多个成员的非空集合。此外,除非上下文另有说明或与之矛盾,否则相 应集合的术语“子集”不是必需表示相应集合的适当子集,但是该子集和 相应集合可以相等。

[0166] 连接的语言,例如“A、B和C中的至少一个”或“A、B和C至少 一个”形式的短语,除非另有明确说明或与上下文明显矛盾否则,否则可 以作为通常使用与上下文一起理解以呈现项目、条款等,可以是是A或B 或C,也可以是A和B以及C的集合的任何非空子集。例如,在具有三个 成员的集合的示例性示例中,连接短语“A、B和C中的至少一个”和“A, B和C中的至少一个”是指以下任意集合:{A}、{B}、{C}、{A,B}、 {A,C}、{B,C}、{A,B,C}。因此,这种连接语言通常并不旨在暗示 某些实施例需要至少一个A、至少一个B和至少一个C,他们每一个用于 呈现。另外,除非另有说明或与上下文矛盾,否则术语“多个”表示复数 的状态(例如,“多个项目”表示复数个项目)。“多个”中项目的数量至少 是两个,但是当明确地或通过上下文指示时可以是更多。此外,除非另有 说明或从上下文中另外可知,否则短语“基于”是指“至少部分基于”而 不是“仅基于”。

[0167] 可以以任何合适的顺序来执行本文描述的过程的操作,除非本文另有 指示或与上下文明显矛盾。在一个实施例中,诸如本文描述的那些过程(或 其变型和/或组合)的过程在一个或更多个计算机系统中的一个控制下通过 硬件或其组合执行,一个或更多个计算机系统配置有可执行指令并且被实 现为在一个或多个处理器上共同执行的代码(例如,可执行指令、一个或 更多个计算机程序或一个或更多个应用程序)。在一个实施例中,代码以计 算机程序的形式存储在计算机可读存储介质上,该计算机程序包括可由一 个或多个处理器执行的多个指令的计算机程序。在一个实施例中,计算机 可读存储介质是非暂时性计算机刻度存储介质,其不包括暂时信号(例如, 传播的瞬态电或电磁传输)但包括瞬时信号的收发器内的非暂时性数据存 储电路(例如,缓冲器、高速缓存和队列)。在一个实施例中,代码(例如, 可执行代码或源代码)被存储在其上存储有可执行指令(或其他存储器以存储可执行指令)的一组一个或更多个非暂时性计算机可读存储介质上, 该可执行指令在由被计算机系统的一个或更多个处理器执行时(例如,(即, 作为被执行的结果),使计算机系统执行本文所述的操作。在一个实施例中, 该组非暂时性计算机可读存储介质包括多个非暂时性计算机可读存储介质, 以及多个非暂时性计算机可读存储介质中的一个或更多个单独的非暂时性 存储介质缺少全部代码,而多个非暂时性计算机可读存储介质共同存储所 有代码。在一个实施例中,可执行指令被执行,使得不同的指令被不同的 处理器执行-例如,非暂时性计算机可读存储介质存储指令,并且主CPU 执行一些指令,而图形处理器单元执行其他指令。在一个实施例中,计算 机系统的不同组件具有独立处理器,以及不同处理器执行指令的不同子集。

[0168] 因此，在一个实施例中，计算机系统配置成实现单独或共同执行本文 所述的过程的操作的一个或更能多个服务，并且这样的计算机系统配置有 能够使操作的执行的适用的硬件和/或软件。此外，实现本公开的实施例的 计算机系统是单个设备，并且在另一实施例中，是一种分布式计算机系统， 其包括以不同方式操作的多个设备，使得该分布式计算机系统执行本文所 述的操作，并且使得单个设备不执行所有操作。

[0169] 除非另外要求，本文提供的任何和所有示例或示例性语言(例如，“诸 如”)的使用仅旨在更好地阐明本发明的实施例，而不对本发明的范围构成 限制。本说明书中的语言不应解释为表示任何未要求保护的要素对于实施 本发明是必不可少的。

[0170] 这里描述了本公开的实施例，包括发明人已知的用于实施本发明的最 佳方式。这些实施例的变型对于本领域普通技术人员而言，在阅读了前述 说明之后将变得显而易见。发明人期望熟练的技术人员适当地采用这种变 型，并且发明人希望本公开的实施例可以不同于本文具体描述的方式实施。因此，本公开的范围包括所列举的主题的所有修改和等同形式。此外，除 非本文另外指出或与上下文明显矛盾，否则上述元件在其所有可能的变型 中的任何组合都包括在本公开的范围之内。

[0171] 本文引用的所有参考文献，包括出版物、专利申请和专利，均以引用 的方式并入本文，其程度如同每个参考文献被单独且具体地指示以引用方 式并入本文一样，并在此全文进行阐述。

[0172] 在说明书和权利要求书中，可以使用术语“耦合”和“连接”及其派 生词。应当理解，这些术语可能不旨在作为彼此的同义词。相反，在特定 示例中，“连接”或“耦合”可用于指示两个或更多个元件彼此直接或间接 物理或电接触。“耦合”也可能意味着两个或多个元素彼此不直接接触，但 仍彼此协作或交互。

[0173] 除非另有说明，否则应理解，在整个说明书中，诸如“处理”、“计算 处理”、“计算”、“确定”等术语均指计算机或计算系统或类似的电子计算 设备的动作和/或过程，这些电子设备将在计算系统的寄存器和/或存储器中 表示为物理量(例如电子)的数据(例如电子)操作和/或转换为类似地表 示为计算系统的存储器、寄存器或其他此类信息存储器、传输或显示设备 中的物理量的其他数据。

[0174] 在类似的方式中，术语“处理器”可以指处理来自寄存器和/或存储 器的电子数据并将该电子数据转换成可以存储在寄存器和/或存储器中的 其他电子数据的任何设备或设备的一部分。作为非限制性示例，“处理器” 可以是中央处理单元(CPU)或图形处理单元(GPU)。“计算平台”可以 包括一个或更多个处理器。如本文所使用的，“软件”过程可以包括例如随 时间执行工作的软件和/或硬件实体，诸如任务、线程和智能代理。同样， 每个过程可以指代多个过程，以依次或并行，连续或间歇地执行指令。术 语“系统”和“方法”在本文中可互换使用，以达到一种该系统可以体现 一种或更多种方法并且该方法可以被认为是系统的程度。

[0175] 在本文档中，可以参考获得、获取、接收或将仿真或数字数据输入子 系统、计算机系统或计算机实现的机器。可以以多种方式来完成获得、获 取、接收或输入仿真和数字数据的过程，例如通过接收作为函数调用或对 应用程序接口的调用的参数的数据。在一些实施方式中，获得、获取，接 收或输入仿真或数字数据的过程可以通过经由串行或并行接口传输数据来 完成。在另一个实施方式中，获得、获取、接收或输入仿真或数字数据的 过程

可以通过经由计算机网络将数据从提供实体转移到获取实体来完成。也可以参考提供、输出、传输、发送或呈现仿真或数字数据。在各种示例 中，提供、输出、传输、发送或呈现仿真或数字数据的过程可以通过将数 据作为函数调用的输入或输出参数、应用程序编程接口或进程间通信机制 的参数进行传输来完成。

[0176] 尽管以上讨论阐述了所描述的技术的示例实施方式，但是其他架构可 以用于实现所描述的功能，并且意图在本公开的范围内。此外，尽管出于 讨论目的在上面定义了具体的职责分配，但是根据情况，各种功能和职责 可能以不同的方式分配和划分。

[0177] 此外，尽管已经以结构特征和/或方法动作专用的语言描述了主题， 但应理解，所附权利要求书中定义的主题不必限于所描述的特定特征或动 作。而是，将特定特征和动作公开为实现权利要求的示例性形式。

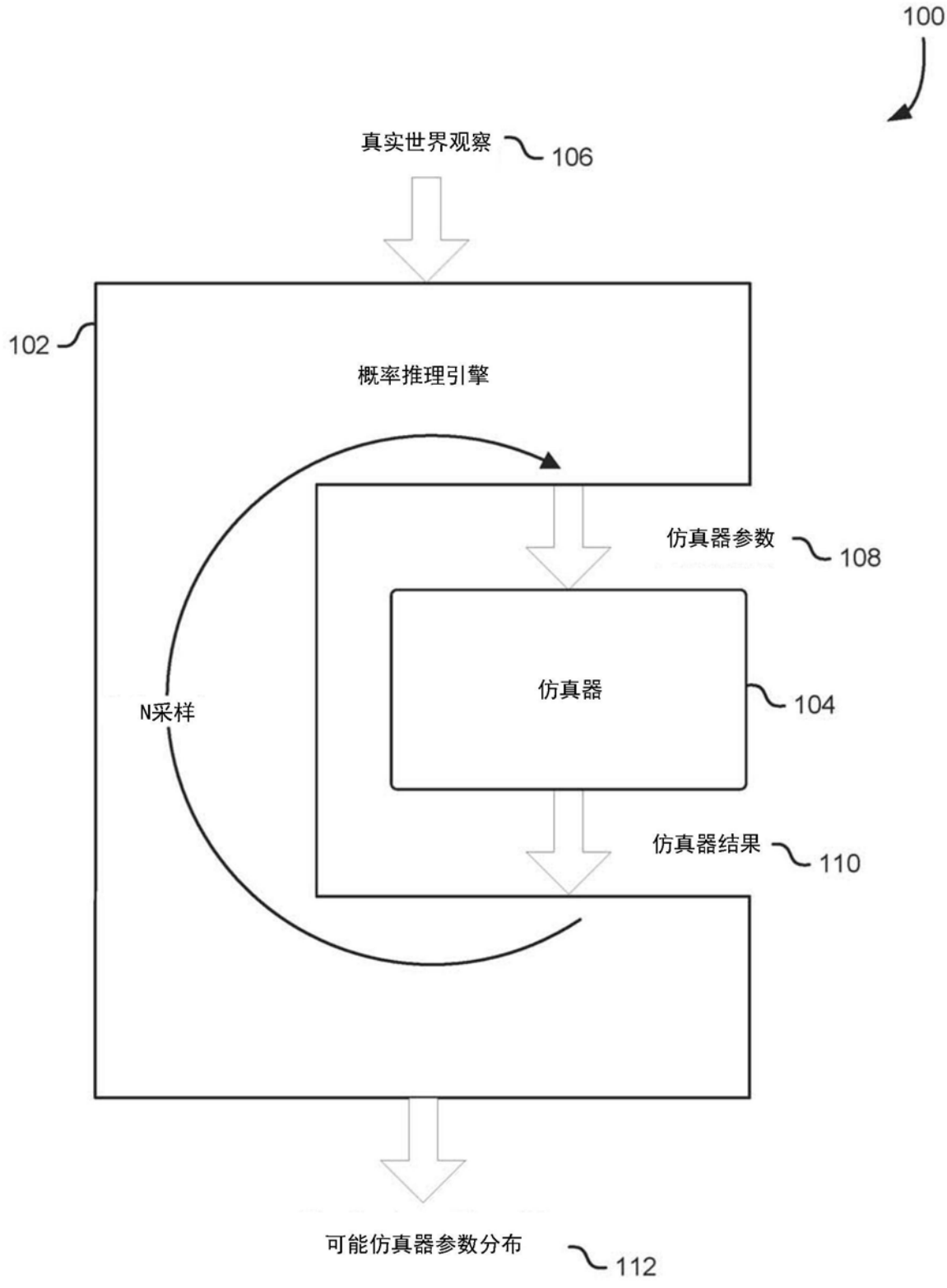

100
真实世界观察
106
102
概率推理引擎
仿真器参数
108
仿真器
104
N采样
仿真器结果
110
可能仿真器参数分布
112

图1

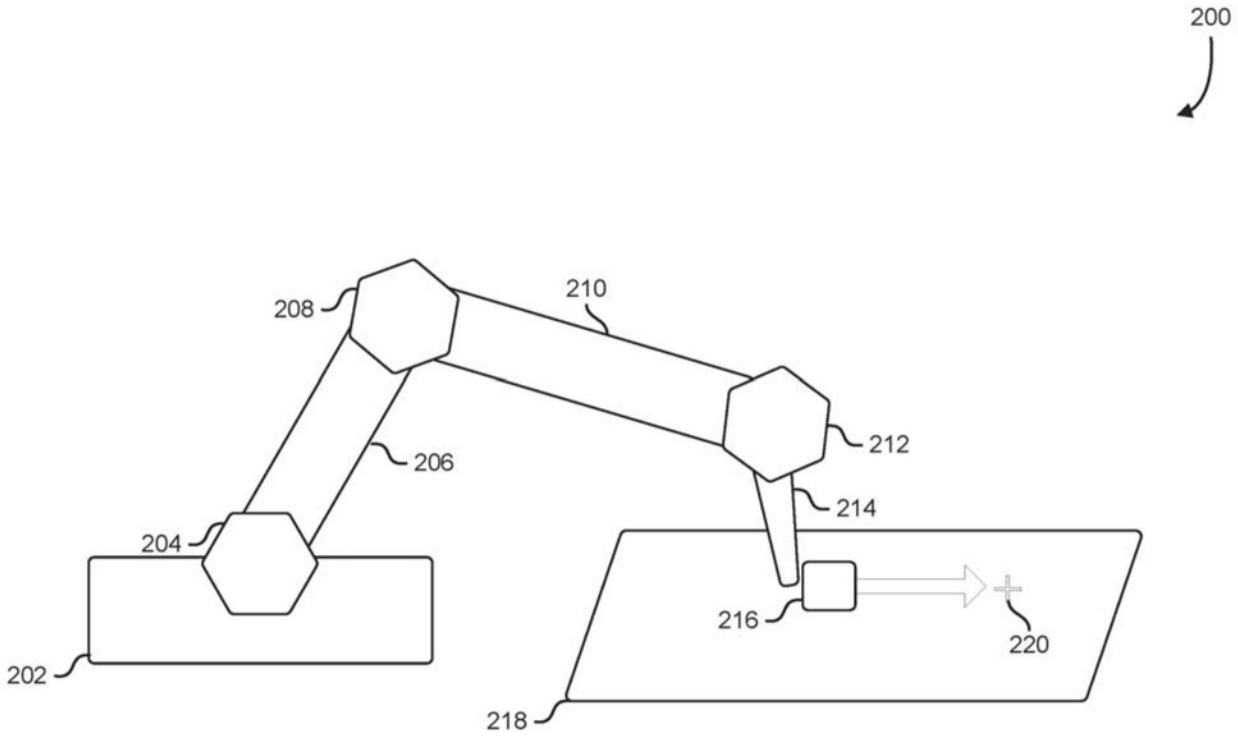

200
208
210
212
206
214
204
216
220
202
218

图2

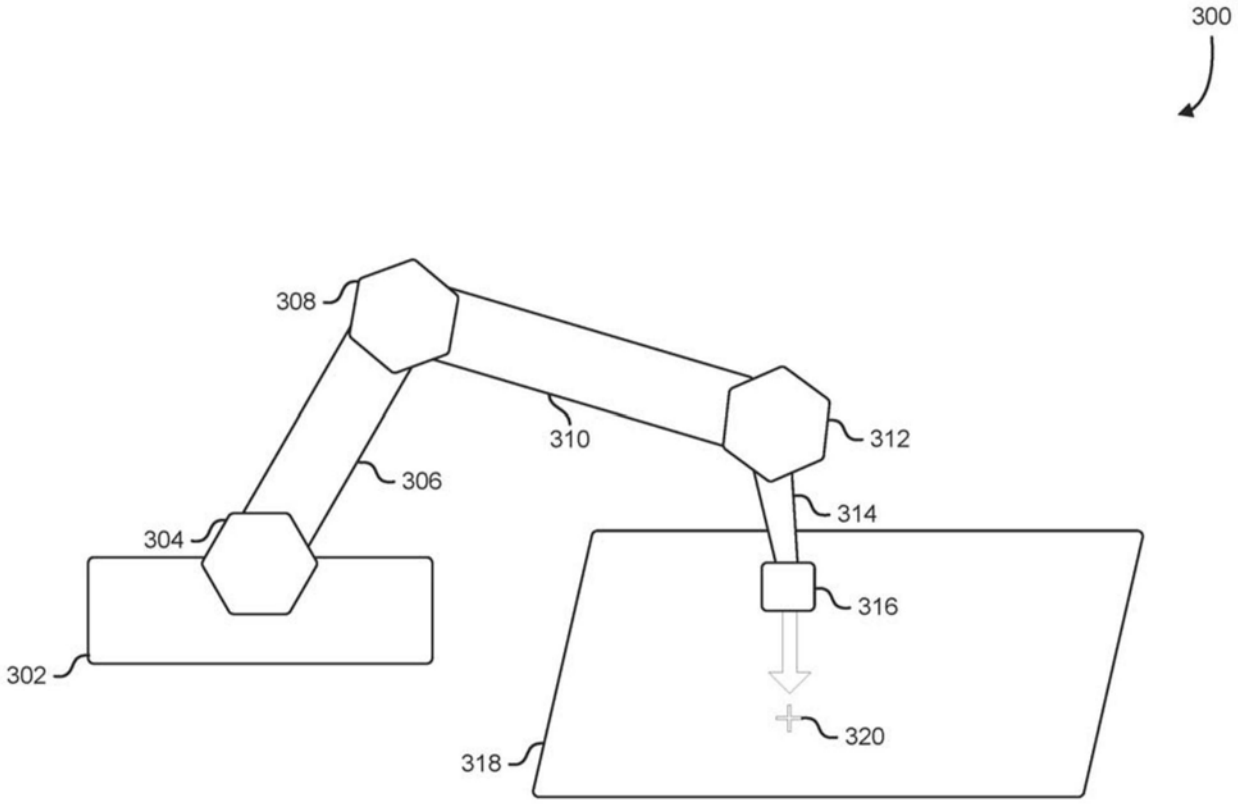

300
308
310
312
306
314
304
316
320
302
318

图3

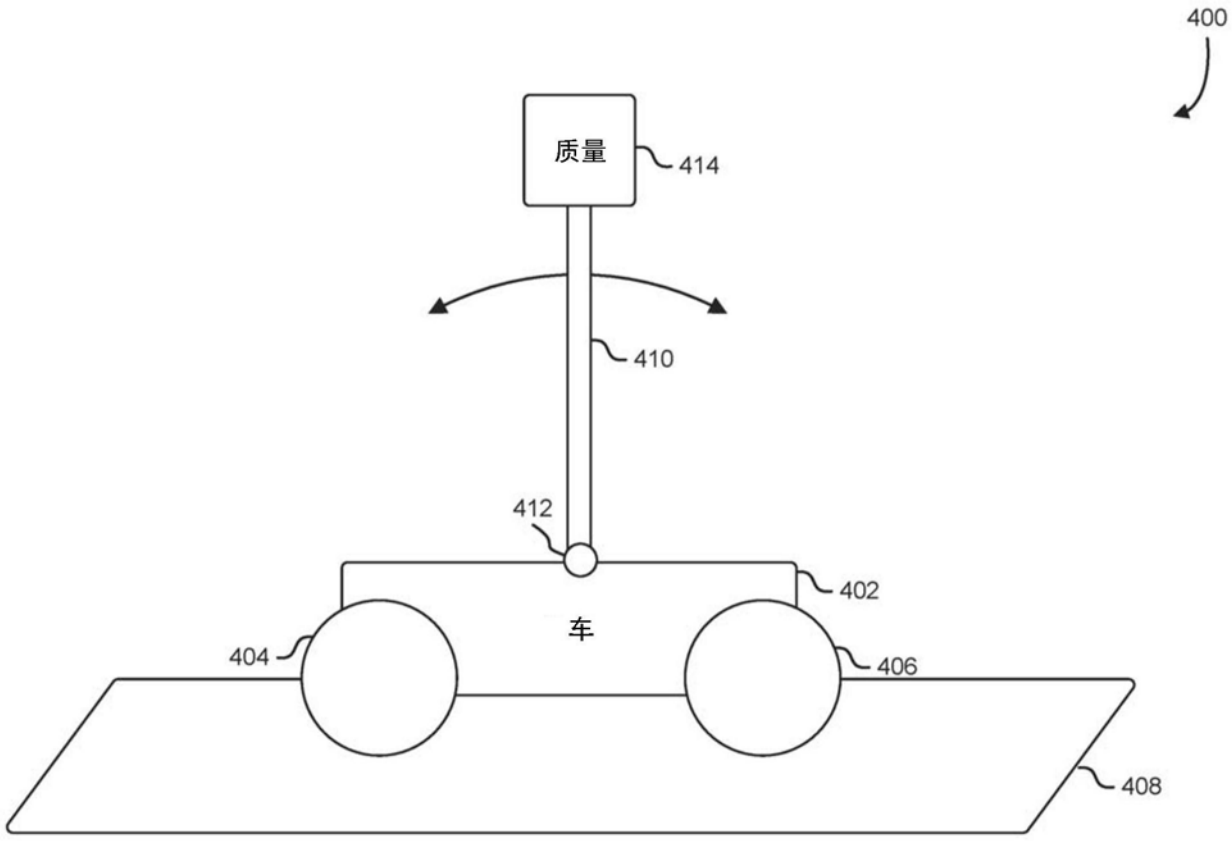


图4

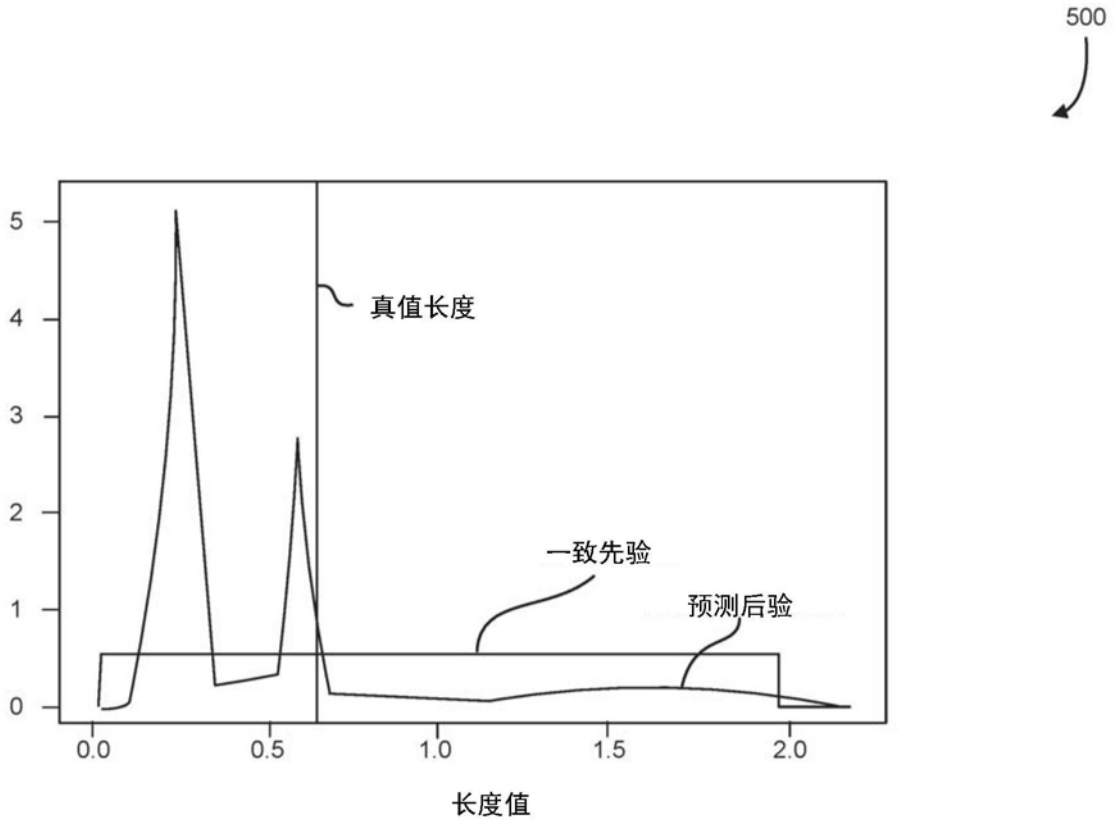


图5

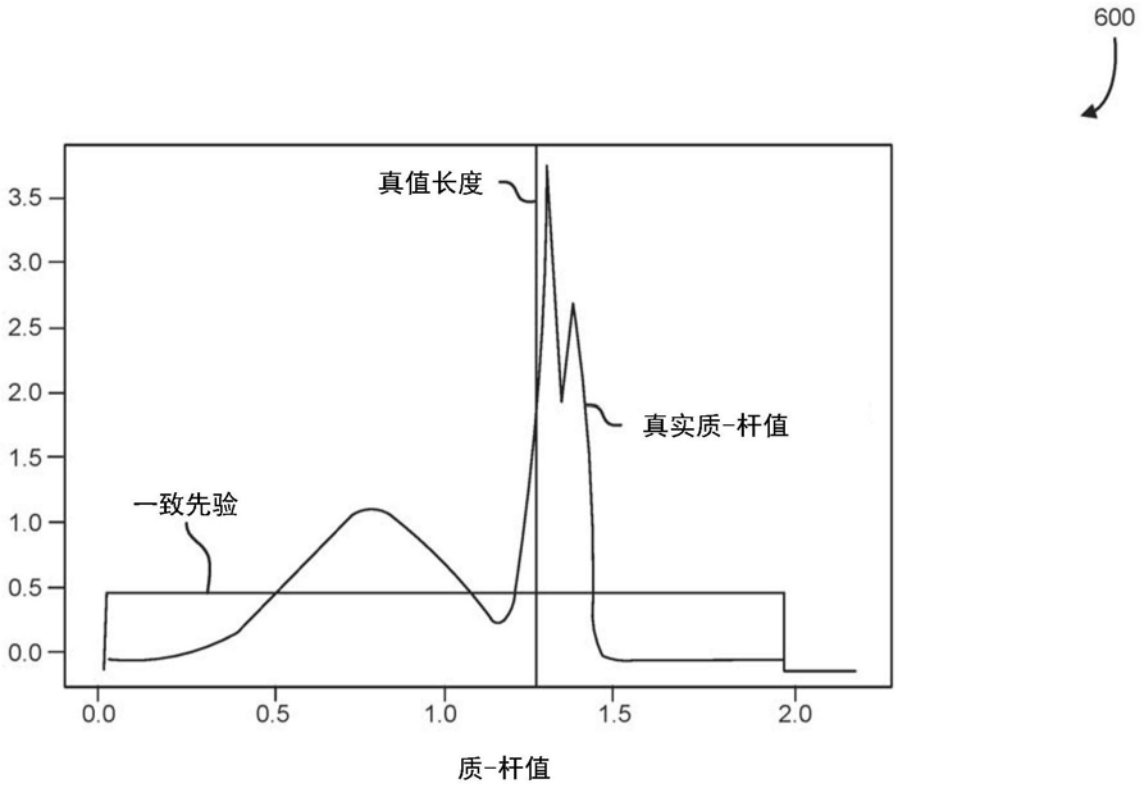


图6

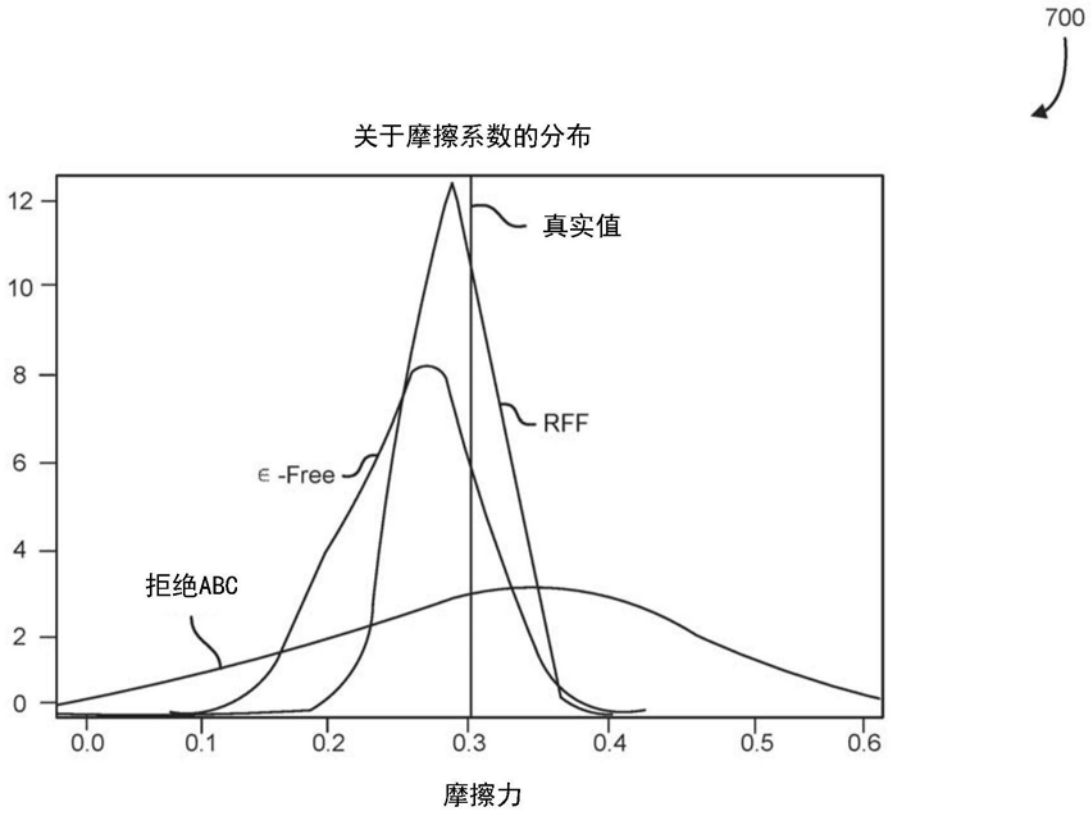


图7

800

问题	参数	一致先验	拒绝ABC	ϵ-Free	RFF	NN
车杆	杆长度	[0.1, 2.0]	-0.342±0.15	**-0.211±0.07**	-0.609±0.39	-0.657±0.25
	杆质量	[0.1, 2.0]	0.032±0.21	0.056±0.14	**0.973±0.26**	0.633±0.52
摇摆	dt	[0.01, 0.3]	2.101±1.04	2.307±0.84	3.192±0.30	**3.199±0.17**
山地车	功率	[0.0005, 0.1]	3.69±1.21	3.800±1.06	3.863±0.52	**3.901±0.2**
机器人	链接质量 1	[0.5, 2.0]	1.704±0.82	1.883±0.79	**2.046±0.37**	1.331±0.22
	链接质量 2	[0.5, 2.0]	1.832±0.93	**2.237±0.76**	0.321±1.85	1.513±0.39
	链接长度 1	[0.1, 1.5]	**2.421±0.75**	2.135±0.50	2.072±0.76	1.856±0.18
	链接长度 2	[0.5, 1.5]	-0.521±0.36	-0.703±0.16	**-0.148±0.19**	-0.672±0.09
漏斗	横向摩擦力	[0.3, 0.5]	3.032±0.43	3.154±0.81	2.2622±0.64	**3.391±0.08**
获取-推动	摩擦力	[0.1, 1.0]	1.332±0.54	2.013±0.09	**2.423±0.07**	2.404±0.05
获取-滑动	摩擦力	[0.1, 1.0]	1.014±0.38	1.614±0.12	**2.391±0.06**	2.111±0.03

图8

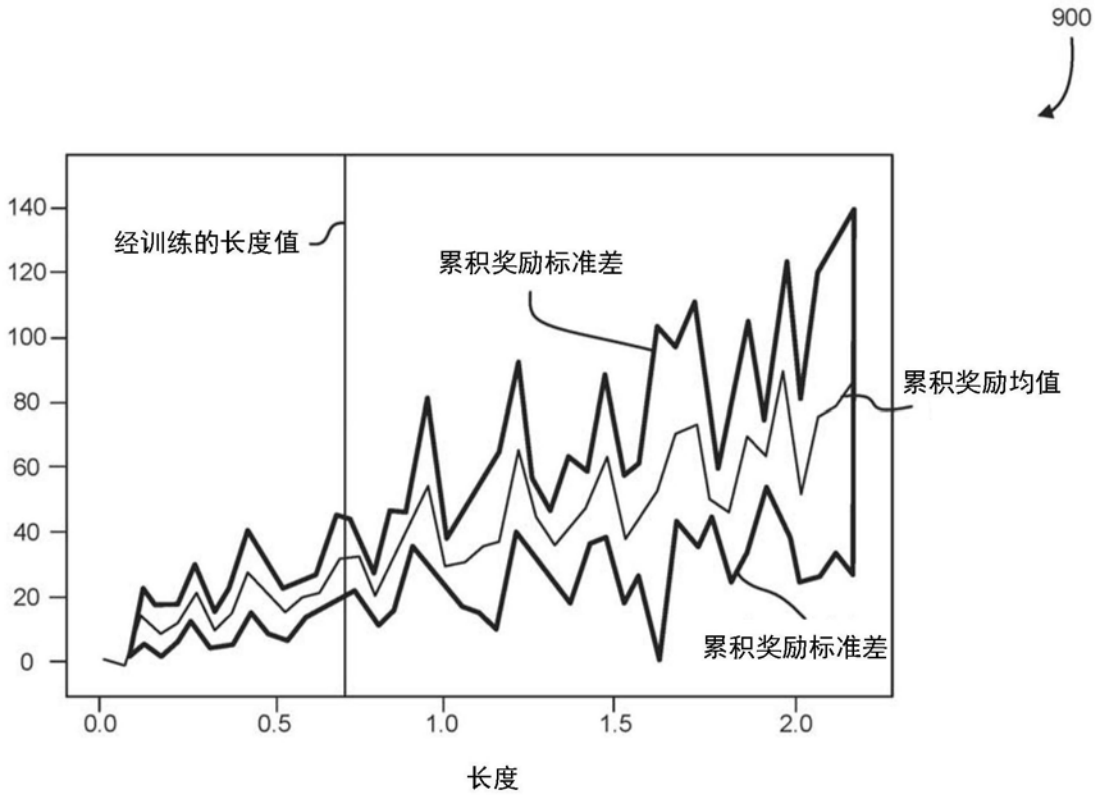

900
140
120
100
80
60
40
20
0
经训练的长度值
累积奖励标准差
累积奖励均值
累积奖励标准差
0.0
0.5
1.0
1.5
2.0
长度

图9

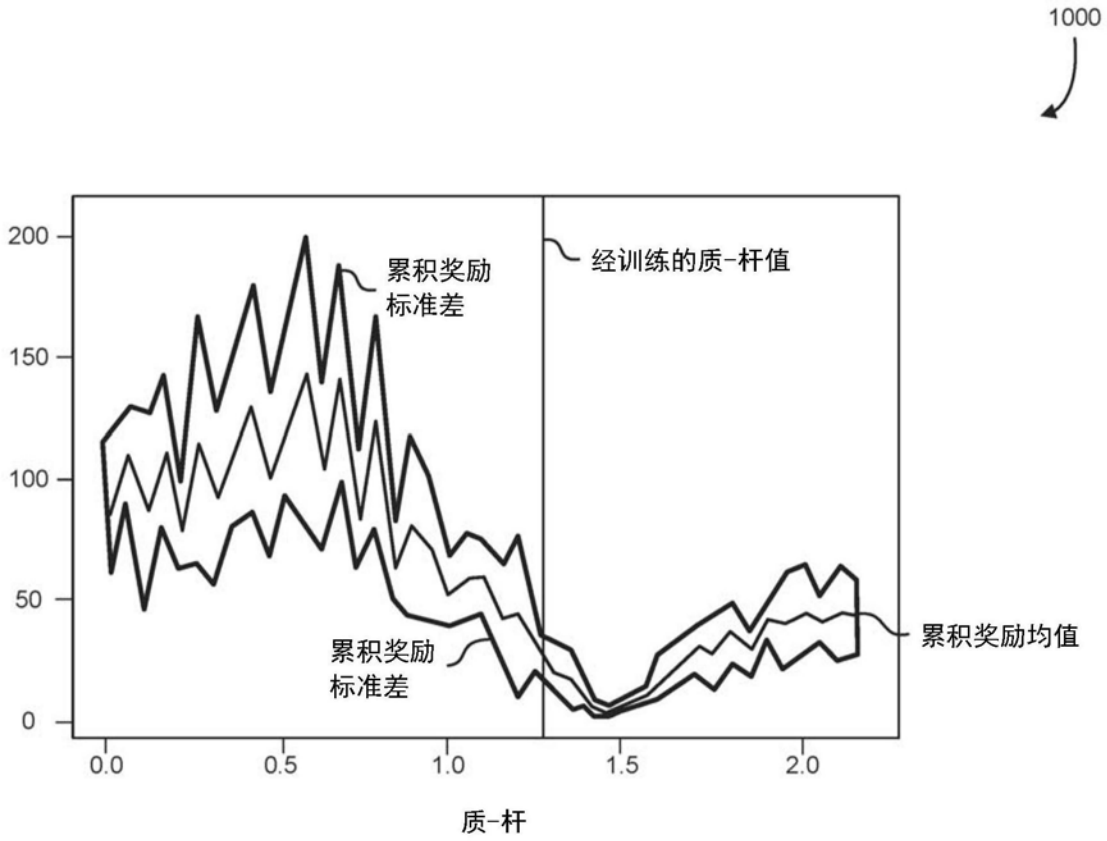

1000
200
150
100
50
0
累积奖励
标准差
经训练的质-杆值
累积奖励
标准差
累积奖励均值
0.0
0.5
1.0
1.5
2.0
质-杆

图10

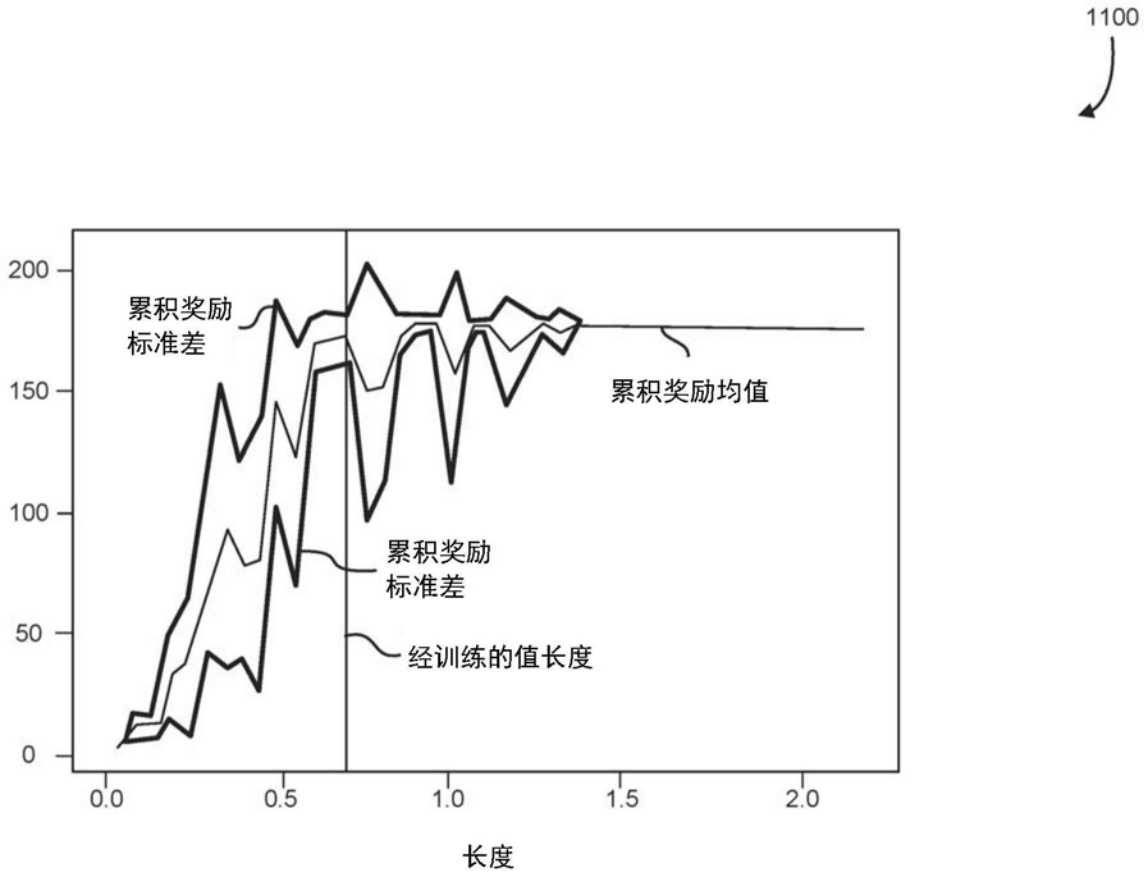

1100
200
150
100
50
0
0.0
0.5
1.0
1.5
2.0
累积奖励
标准差
累积奖励均值
累积奖励
标准差
经训练的值长度
长度

图11

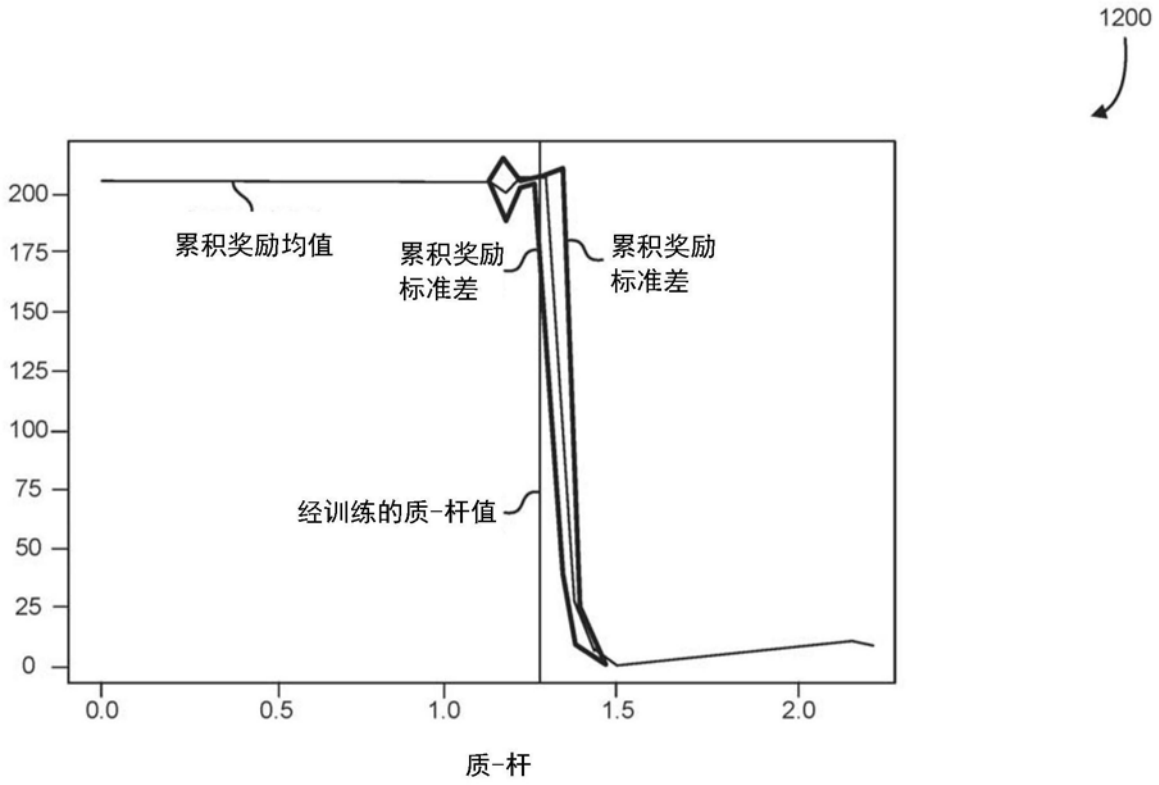

1200
200
175
150
125
100
75
50
25
0
累积奖励均值
累积奖励
标准差
累积奖励
标准差
经训练的质-杆值
0.0
0.5
1.0
1.5
2.0
质-杆

图12

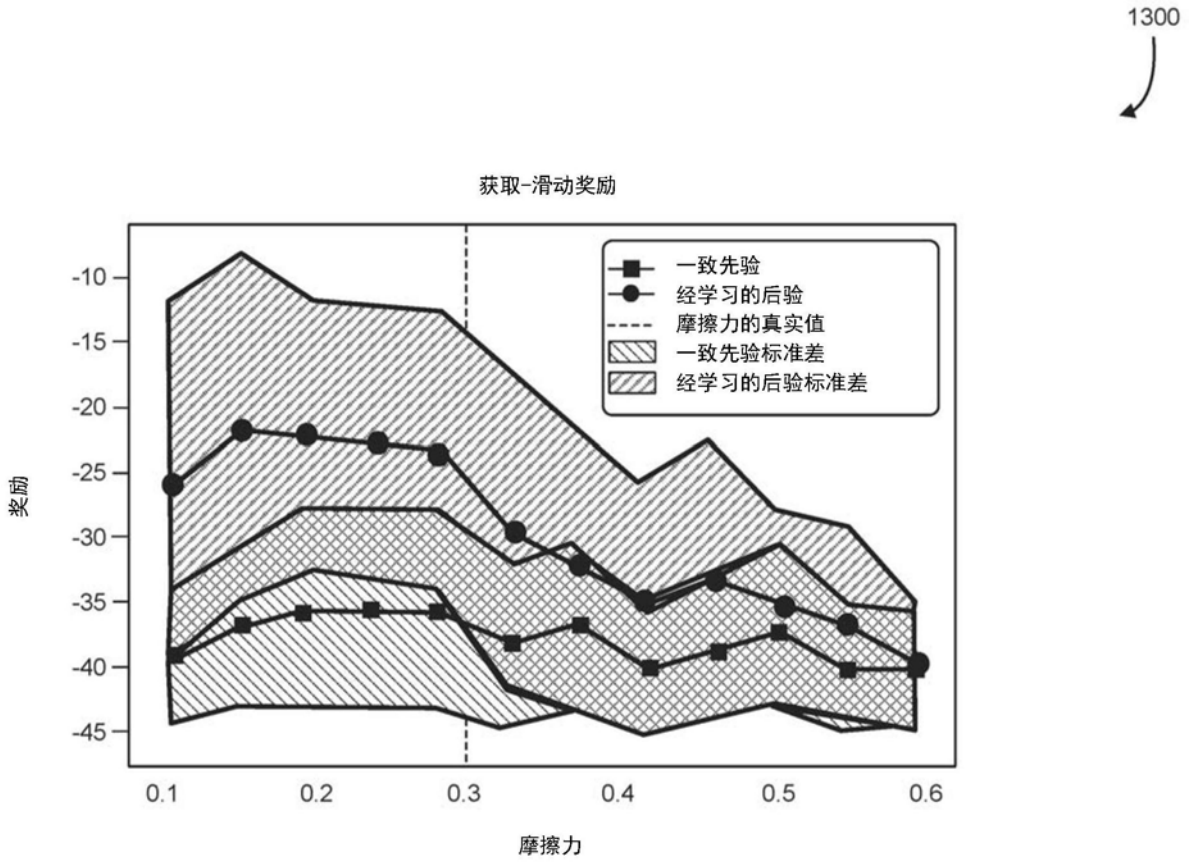

1300
获取-滑动奖励
一致先验
经学习的后验
摩擦力的真实值
一致先验标准差
经学习的后验标准差
奖励
-10
-15
-20
-25
-30
-35
-40
-45
0.1
0.2
0.3
0.4
0.5
0.6
摩擦力

图13

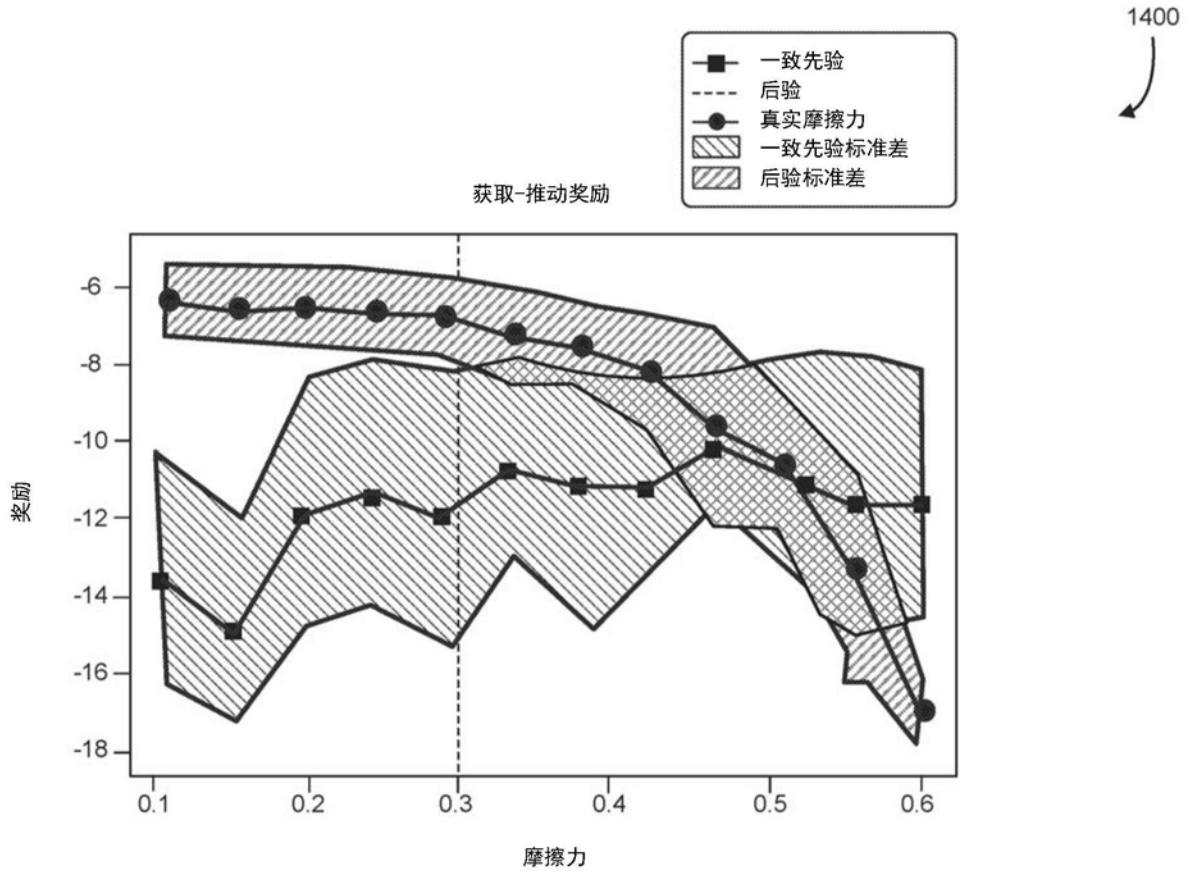

1400
一致先验
后验
真实摩擦力
一致先验标准差
后验标准差
获取-推动奖励
奖励
-6
-8
-10
-12
-14
-16
-18
0.1
0.2
0.3
0.4
0.5
0.6
摩擦力

图14

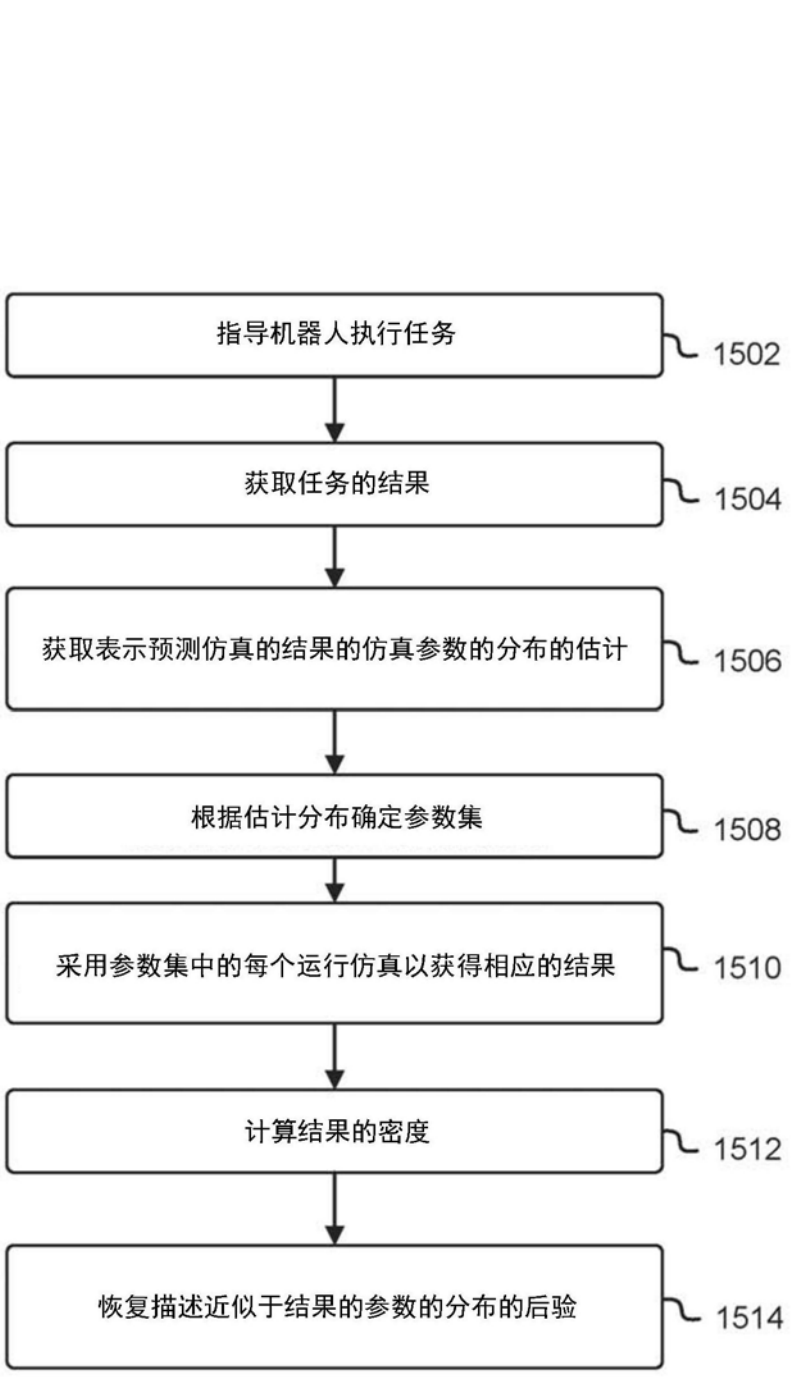

1500
指导机器人执行任务
1502
获取任务的结果
1504
获取表示预测仿真的结果的仿真参数的分布的估计
1506
根据估计分布确定参数集
1508
采用参数集中的每个运行仿真以获得相应的结果
1510
计算结果的密度
1512
恢复描述近似于结果的参数的分布的后验
1514

图15

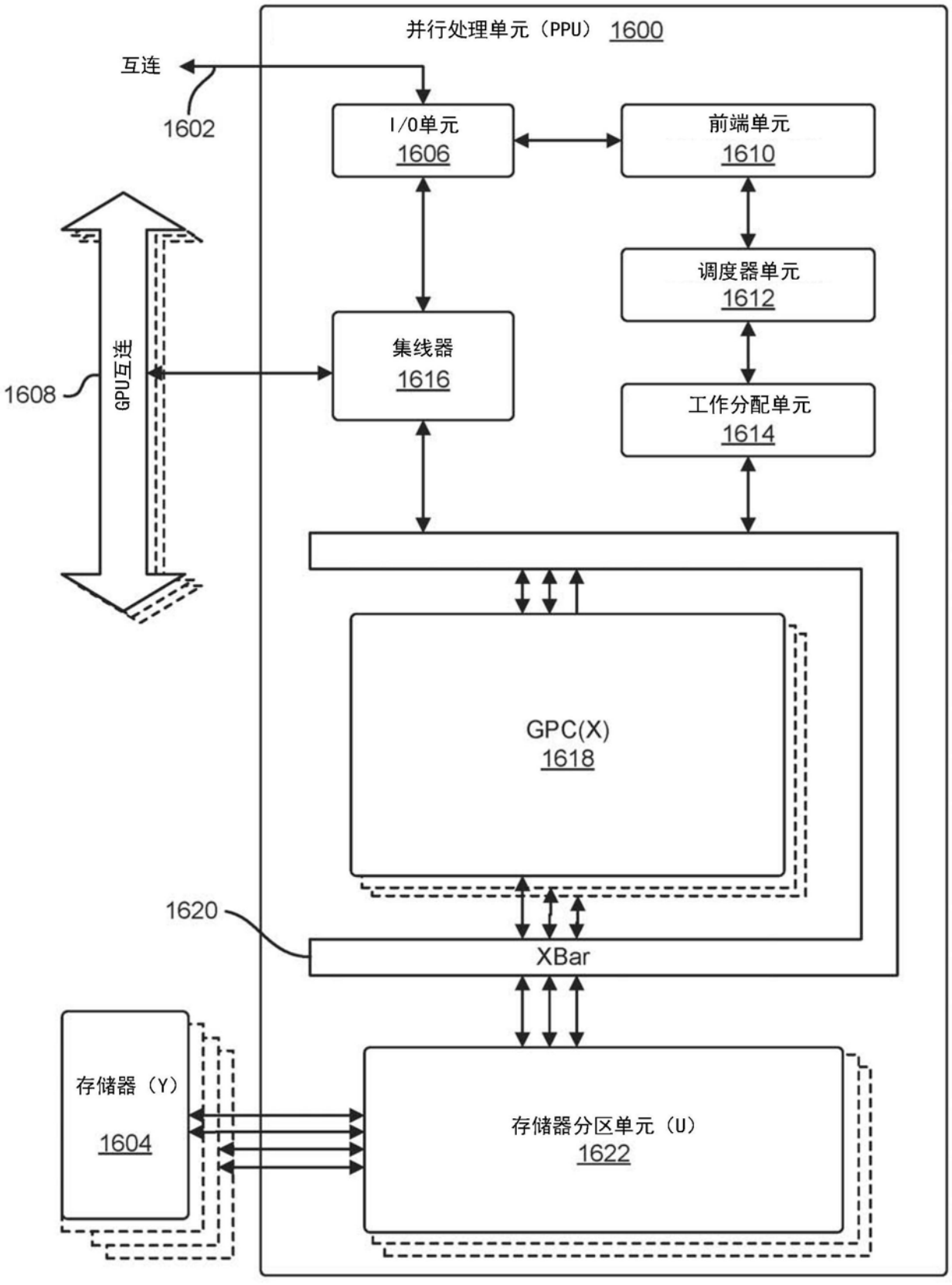

并行处理单元（PPU） 1600
互连
1602
I/O单元
1606
前端单元
1610
调度器单元
1612
集线器
1616
工作分配单元
1614
1608
GPU互连
GPC(X)
1618
1620
XBar
存储器（Y）
1604
存储器分区单元（U）
1622

图16

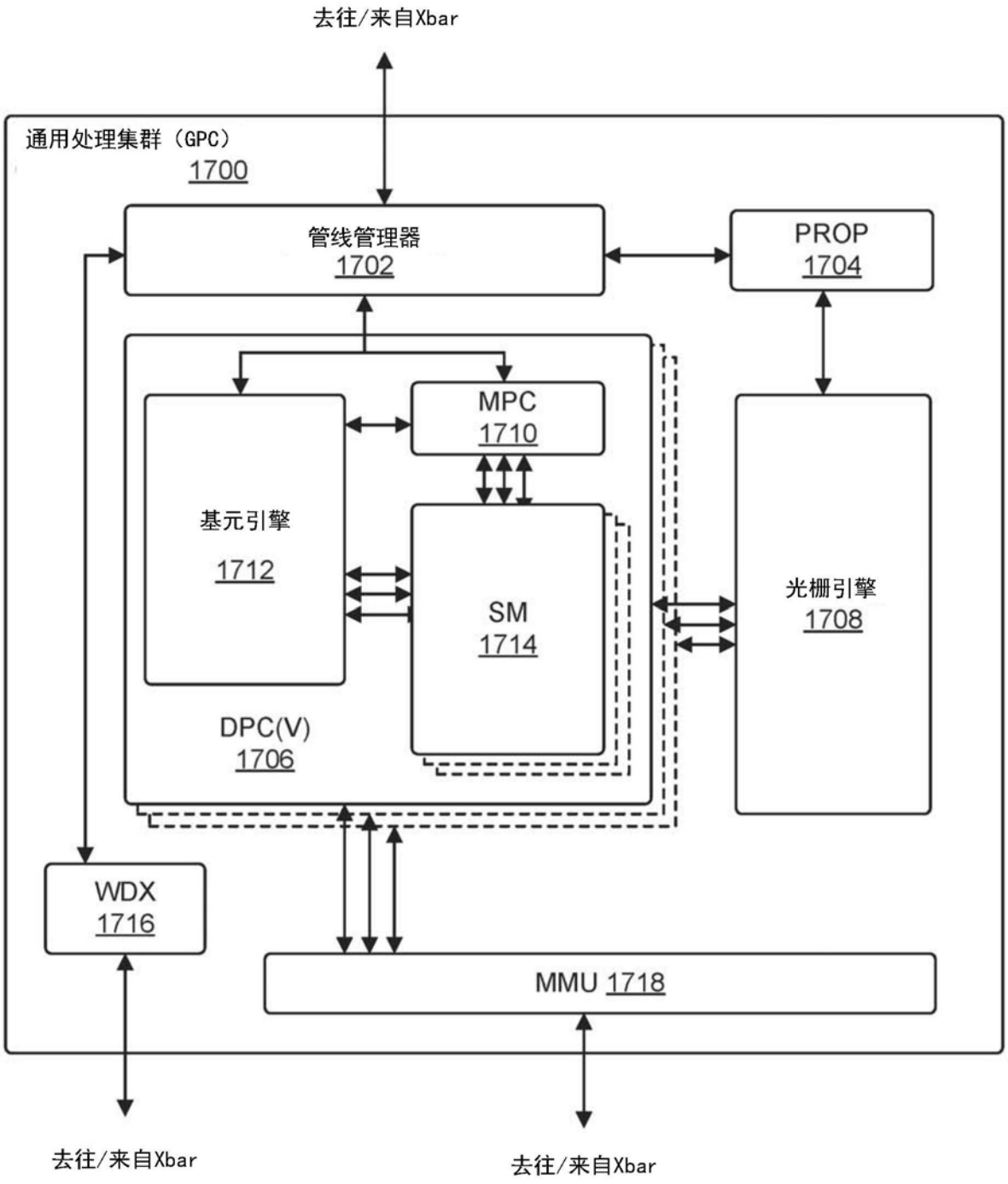

去往/来自Xbar
通用处理集群（GPC）
1700
管线管理器
1702
PROP
1704
MPC
1710
基元引擎
1712
SM
1714
光栅引擎
1708
DPC(V)
1706
WDX
1716
MMU 1718
去往/来自Xbar
去往/来自Xbar

图17

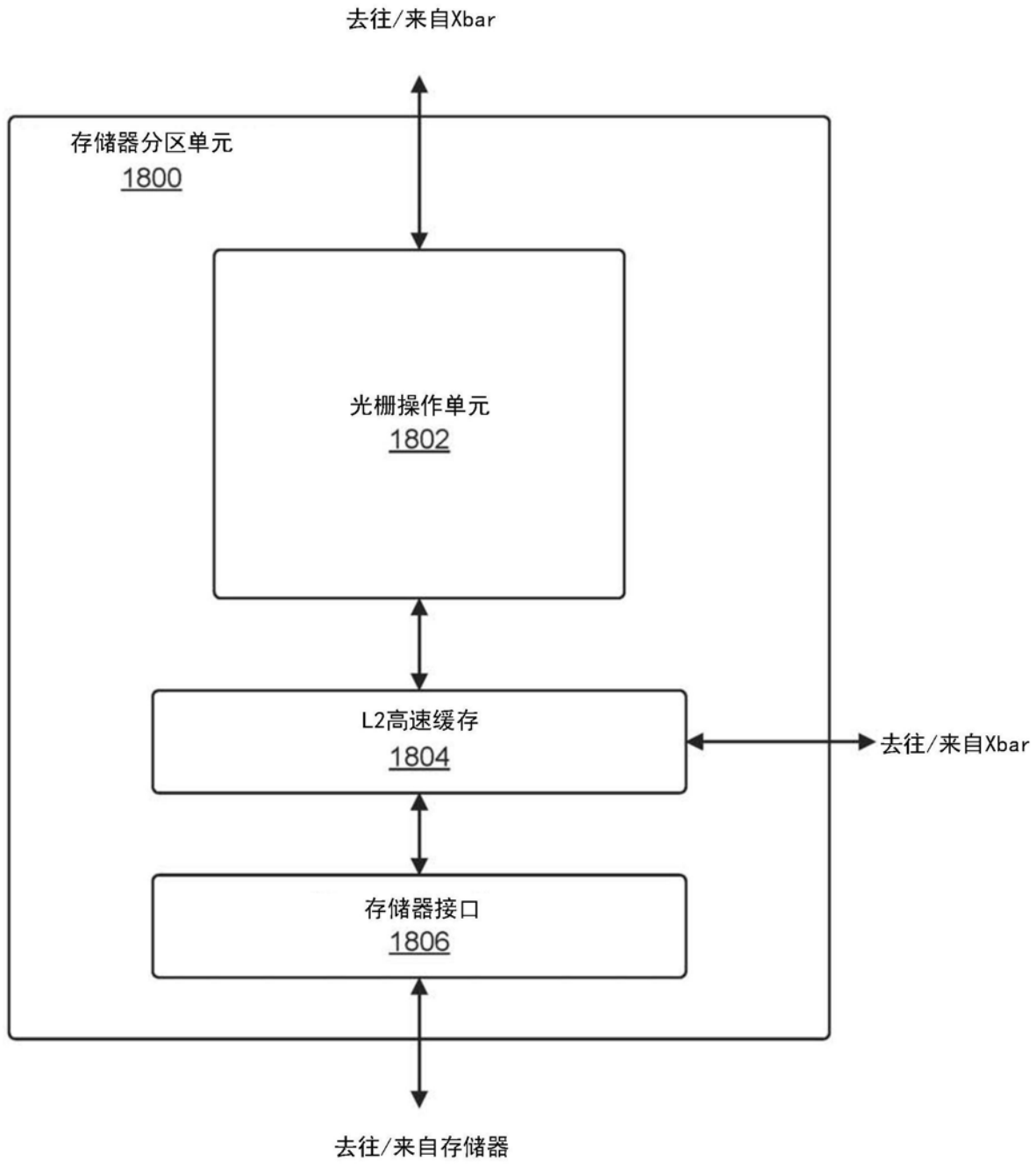

去往/来自Xbar
存储器分区单元
1800
光栅操作单元
1802
L2高速缓存
1804
去往/来自Xbar
存储器接口
1806
去往/来自存储器

图18

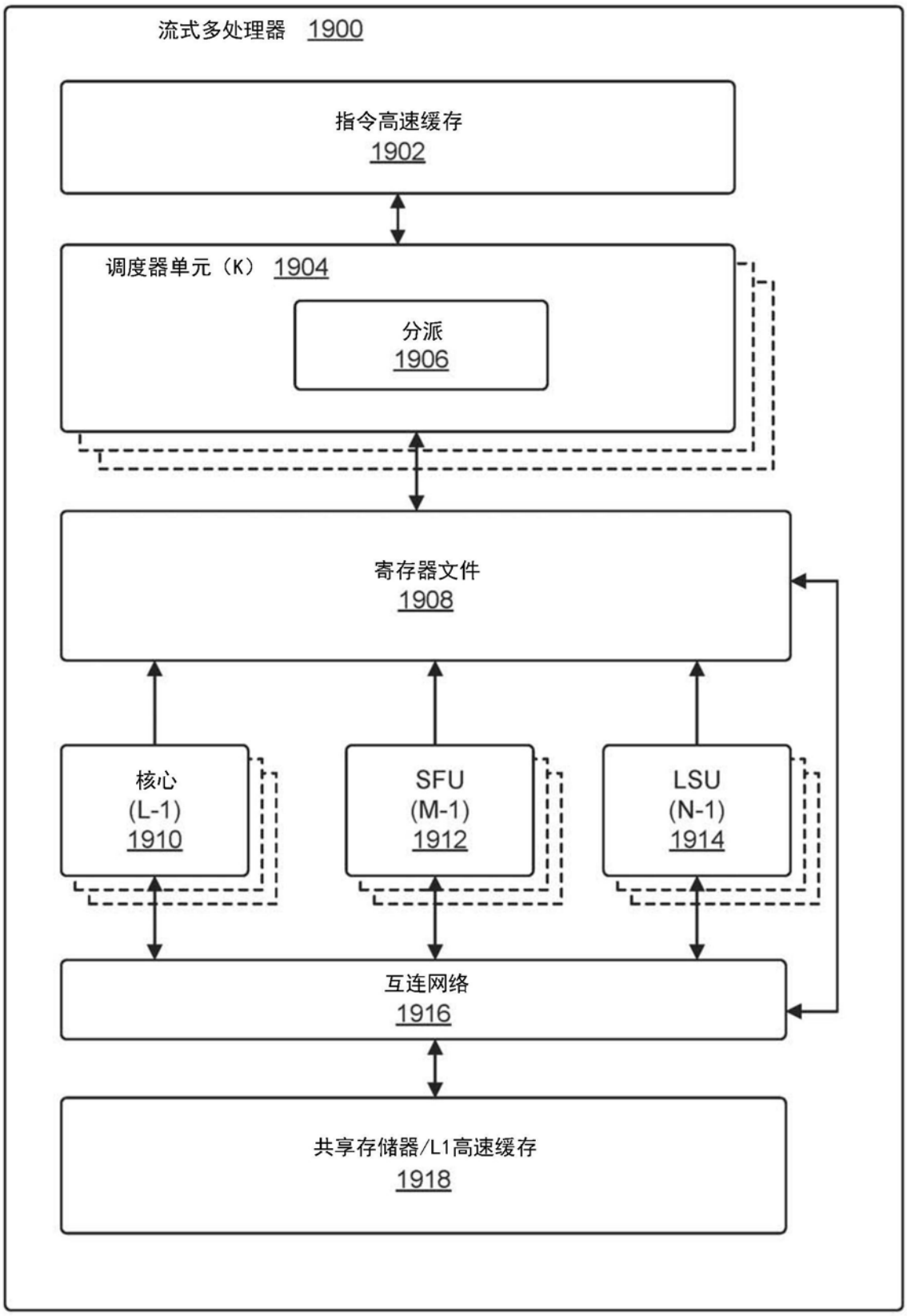

流式多处理器 1900
指令高速缓存
1902
调度器单元（K） 1904
分派
1906
寄存器文件
1908
核心
(L-1)
1910
SFU
(M-1)
1912
LSU
(N-1)
1914
互连网络
1916
共享存储器/L1高速缓存
1918

图19

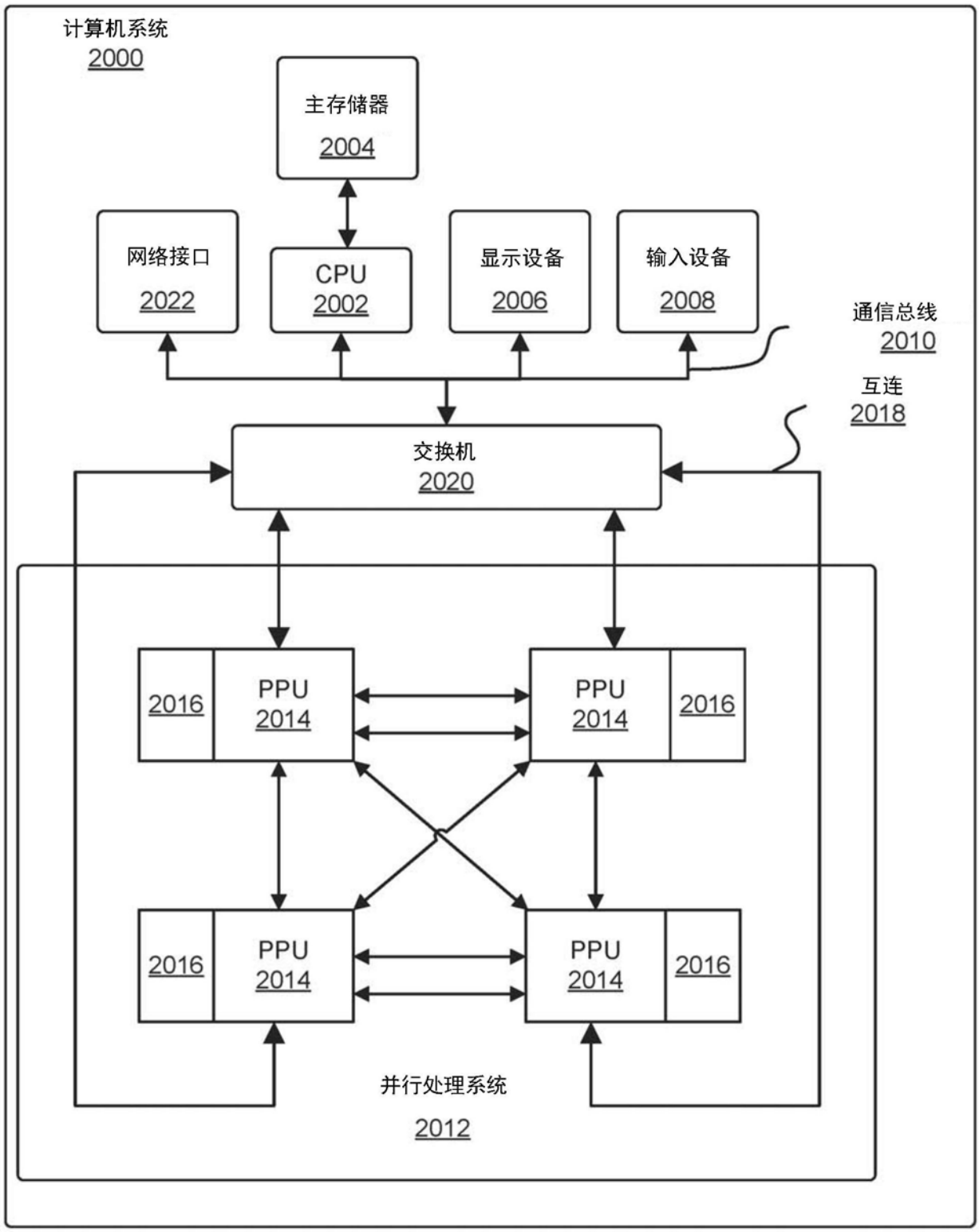

计算机系统
2000
主存储器
2004
网络接口
2022
CPU
2002
显示设备
2006
输入设备
2008
通信总线
2010
互连
2018
交换机
2020
2016
PPU
2014
PPU
2014
2016
2016
PPU
2014
PPU
2014
2016
并行处理系统
2012

图20