SCIENCE ROBOTICS | RESEARCH ARTICLE

ARTIFICIAL INTELLIGENCE

Learning agile and dynamic motor skills for legged robots

Jemin Hwangbo[1]*, Joonho Lee[1], Alexey Dosovitskiy[2], Dario Bellicoso[1], Vassilios Tsounis[1], Vladlen Koltun[3], Marco Hutter[1]

Legged robots pose one of the greatest challenges in robotics. Dynamic and agile maneuvers of animals cannot be imitated by existing methods that are crafted by humans. A compelling alternative is reinforcement learning, which requires minimal craftsmanship and promotes the natural evolution of a control policy. However, so far, reinforcement learning research for legged robots is mainly limited to simulation, and only few and comparably simple examples have been deployed on real systems. The primary reason is that training with real robots, particularly with dynamically balancing systems, is complicated and expensive. In the present work, we introduce a method for training a neural network policy in simulation and transferring it to a state-of-the-art legged system, thereby leveraging fast, automated, and cost-effective data generation schemes. The approach is applied to the ANYmal robot, a sophisticated medium-dog–sized quadrupedal system. Using policies trained in simulation, the quadrupedal machine achieves locomotion skills that go beyond what had been achieved with prior methods: ANYmal is capable of precisely and energy-efficiently following high-level body velocity commands, running faster than before, and recovering from falling even in complex configurations.

【强化学习 38】RL ANYmal

张楚珩
清华大学 交叉信息院博士在读

这是刚刚发表在Science子刊上的一篇工作，开创性地把强化学习成功应用到了实体的四足机器人ANYmal上。

原文传送门

Hwangbo, Jemin, et al. "Learning agile and dynamic motor skills for legged robots." Science Robotics 4.26 (2019): eaau5872.

特色

迄今，许多强化学习算法产生的控制器在模拟环境下（比如Mujoco）都能很好地控制机器人完成特定的任务了。同时，在真实环境下，机器人也展现出来了极强的机动能力，比如波士顿动力。在人们惊叹于波士顿动力的机器人时，我们需要注意到，他们并没有使用强化学习的算法。这主要由于强化学习得到的控制器在直接迁移到实际系统中时常常会失败，这主要由模拟环境和真是环境的差异造成的，即现实落差（reality gap，我随便翻译的）。本文通过一系列方法克服了现实落差，使得强化学习得到的控制器能够成功迁移到实际系统中。

过程

克服现实落差的途径

文章中提到克服现实落差的方法主要有两类。

一类方法是建立更为贴近实际环境的模拟环境。如果仅仅只是像 Mujoco那样的物理环境，对于特定的刚体结构，给定输出的力矩，模拟给出机器人的运动学特征，这样的环境还比较容易模拟。但是对于实际的机器人来说，机器人的每一个关节都是通过驱动器来控制的。当策略给出一个目标力矩时，驱动器并不能马上达到这个力矩，这中间会有延迟和误差，要让模拟环境能够刻画这些延迟和误差则十分困难。常见的驱动器有液压驱动器（hydraulic cylinder）、伺服电机（servomotor）和 SEA（series elastic actuator，这也是 ANYmal 平台使用的驱动方式），这些驱动器很难用解析

的方法来模拟。为了更好地模拟驱动器的动力学特征，这篇文章使用了数据驱动的方法，我们马上会看到。

另一类方法则是在模拟的时候就考虑到模拟环境和现实环境的不同，期望能够得到更为鲁棒的控制器。这类方法的主要实现方法包括在模拟环境中随机变化模拟参数（比如机器人的质量、质心位置等）、使用随机性策略、在模拟环境中加上噪声、在观察到的状态上加噪声等。

本文同时使用了这两类方法来克服现实落差，具体的做法我们将一一介绍。

总体框架

总体框架如下图所示。为了构建一个好的模拟环境，首先需要对 ANYmal 机器人进行建模，用于机器人的物理模拟；同时还需要对于驱动器的电动特性进行建模，这里使用了数据驱动的方法，即训练了一个神经网络来进行驱动器特性的模拟。最后，把训练好的策略神经网络直接部署到机器人上，这篇工作比较特色的一点是神经网络直接部署之后不需要进行太多的调节就能有比较好的效果。

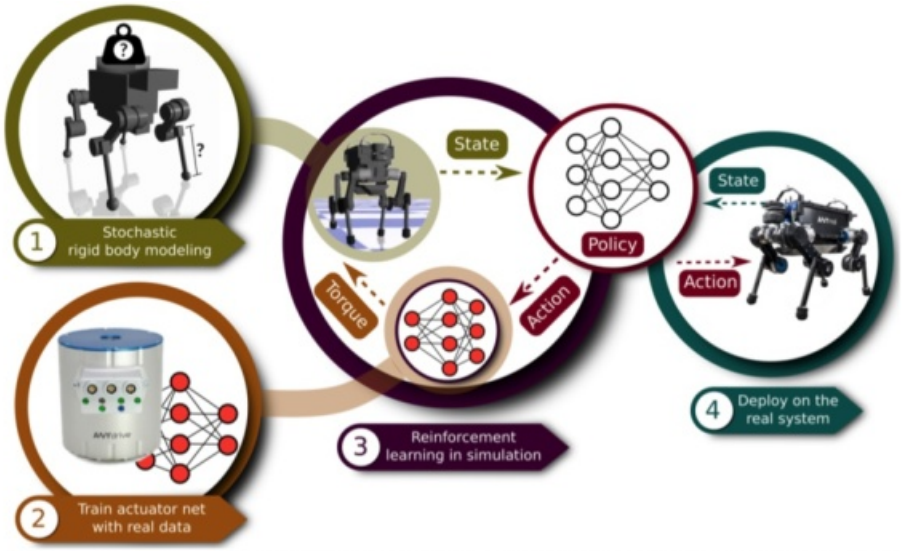


Fig. 1. Creating a control policy. In the first step, we identify the physical parameters of the robot and estimate uncertainties in their properties. In the second step, we train an actuator net that models complex actuator/software dynamics. In the third step, we train a control policy using the model produced in the first two steps. In the fourth step, we deploy the trained policy directly on the physical system.

下面这个图更为详细地描述了这个流程。

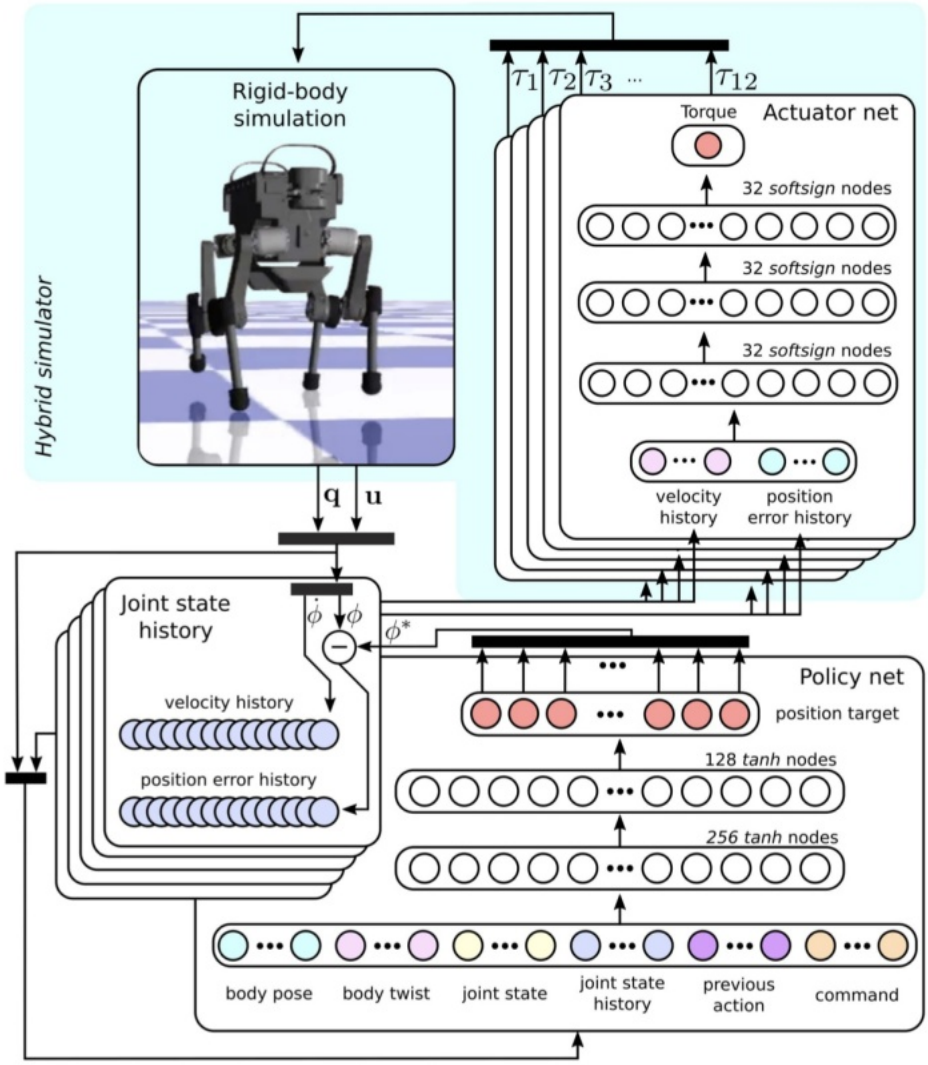


Fig. 5. Training control policies in simulation. The policy network maps the current observation and the joint state history to the joint position targets. The actuator network maps the joint state history to the joint torque, which is used in rigid-body simulation. The state of the robot consists of the generalized coordinate q and the generalized velocity u. The state of a joint consists of the joint velocity $\dot{\phi}$ and the joint position error, which is the current position ϕ subtracted from the joint position target ϕ^*.

1. 对于机器人平台的建模

为了克服现实落差，在对于机器人进行建模的时候，对于质心位置、关节处的质量和关节处位置都做了一定的随机化处理（在某个范围内均匀采样）。个人理解，关节处的质量应该指的是机器人每个关节处的那个驱动器质量，因为通过图片可以看到机器人的每个关节处有一个比较重的驱动器，它的质量应该还挺影响机器人的平衡的。关节处的位置应该指的就是腿长，机器人在不平的地面上可能会踩空什么的，对于腿长进行随机化估计能增大这种情况下的鲁棒性。

2. 驱动器网络（Actuator Net）

对于驱动器来说，一个重要的问题是驱动器需要接受一个怎样的指令。有两类广泛使用的指令方式，一种是告诉驱动器需要输出多大的力矩（torque），另一种是告诉驱动器需要把关节保持在什么位置（position）。这里采用了后一种，其优势在于在训练的初期，如果指令是一个恒定的位置，机器人不容易倒；反之，如果指令是一个恒定的力矩，那么机器人很容易不平衡。

驱动器网络的目的就是学习对于一个给定的目标关节位置（target position），输出一个目标力矩让驱动器来实现。本文使用了数据驱动的方法，即在 ANYmal 机器人上采集关节位置和关节力矩的数据，然后用监督学习的方式来训练一个神经网络完成关节位置到关节力矩的函数关系映射。

在使用的过程中，对于驱动器来说，它的工作流程是先把指令用 PD 控制器转化为目标力矩，再使用 PID 控制把目标力矩转化为目标电流，最后使用 field-oriented controller 把产生相应的电压控制，从而在关节上产生一个实际的力矩。而这里训练得到的驱动器网络就是用来模拟驱动器上从目标位置到实际力矩的全过程。

驱动器网络的输入包括机器人过去的速度以及过去一段时间位置的跟踪误差；网络的输出则是各个

关节的力矩。为了训练这个网络，本文使用一个比较垃圾的控制器来控制该机器人走路，然后采集了机器人的位置、速度和力矩数据，用于驱动器网络的训练。

3. 策略网络（Policy Net）

策略网络的训练主要使用的就是强化学习的框架，其输入的状态（state）包括机器人的质量、姿态、控制指令以及各种历史信息。其中控制指令主要由纵向速度、横向速度、自转角速度三个部分构成，可以理解指令主要来自和遥控汽车遥控器类似的手柄。神经网络输出动作（action）就是各个关节的目标位置。策略网络的训练中，可以把驱动器网络和刚体模拟打包在一起看做是一个模拟环境。强化学习的奖励（reward）是针对不同任务人为规定的损失函数。

值得注意的一点是，对于机器人跑步的任务来说，奖励一般包括任务本身的描述（比如向前移动的速度），同时考虑到实际系统中的机器人都是欠驱动的系统，各个关节对于最大输出力矩和功耗都有一定的要求，因此奖励通常还包括功率消耗等惩罚。如果一开始就强调功耗惩罚，训练得到的控制器很可能会让机器人出于静止不动的局部极小值点处。为了克服这种情况，这篇文章使用了课程学习（curriculum learning）的方法，先让机器人在较小的功耗惩罚下学习到一个能够达到目标任务的策略，然后再慢慢增大功率惩罚以获得能效较高的策略。

同时，注意到策略网络的输入包含各种速度项，但是速度并不是能够直接观测到的物理量，它们通常是通过差分方法估计得到的，误差较大。为了弥补这部分观测的误差，在训练的时候，对于速度的观测值都增加了一定的噪声。

策略网络的训练使用 TRPO 算法，主要考虑到该方法对于超参数比较鲁棒，不需要经过太多的参数调整。训练过程不需要太大的算力，一个 GPU 若干小时就能搞定，使用部署在机器人身上的的计算机做 prediction 的时间只需要 25us。

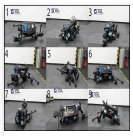
实验结果

这篇工作主要做了 ANYmal 四足机器人的控制、快速奔跑、摔倒后站起这三个实验。

其中控制主要体现对于给定的目标速度（纵向速度、横向速度、转动角速度）之后，机器人能够很好地跟踪，并以目标速度行走。比较有意思的点在于给定速度较慢的时候机器人展现走路的步态，速度加起来之后，机器人能够比较自然地切换到奔跑的步态。

快速奔跑的实验主要说明使用强化学习方法能够在 ANYmal 这个实验平台上得到更快的奔跑速度（1.6m/s）。这个速度没有波士顿动力的那些机器人跑得快，不过跑步速度在不同机器人硬件平台上比较也没啥意义，这和相应的功耗、驱动方式、机器人自身质量有很大关系。

摔倒后站起这个实验结果个人感觉最为惊艳。给定一个好的初态，让机器人奔跑不是很难的事情，但是让机器人能够在摔倒之后站起来则比较困难。从下面的这个视频可以看到，对于不同的卧倒状态，机器人都能找到比较高效的方式站起来，这一点在非强化学习控制的机器人身上个人还没有看到。



ANYmal 能够从卧倒状态爬起来 https://www.zhihu.com/video/1073344872864

总结

本文主要使用了以下方法来克服现实落差（reality gap）

- 使用监督学习方法来训练了一个驱动器网络（actuator net），从而更为准确地对于环境进行模拟；
- 在机器人刚体建模中加入随机化的参数，从而使得训练得到的策略能够更为稳定地面对实际系统中机器人的参数变化；
- 在训练过程中对于机器人的部分观测量增加噪声，弥补实际中对于这部分观测量的观测误差。

编辑于 2019-01-28

强化学习 (Reinforcement Learning) 机器人控制 机器人

文章被以下专栏收录

强化学习前沿
读呀读paper

进入专栏