arXiv:2305.02241v1 [cs.RO] 3 May 2023

多环境自主驾驶多步动力学建模框架

Jason Gibson[1], Bogdan Vlahov[1],David Fan[2],Patrick Spieler[2],Daniel Pastor[2],Ali-akbar Agha-mohammadi[2],Evangelos A. Theodorou[1]

摘要—对动力学进行建模通常是实现车辆自主驾驶的第一步。尽管道路上的自动驾驶车辆已得到广泛研究，但越野车辆仍带来许多棘手的建模问题。越野车辆会遇到极其复杂且难以建模的地形与车辆交互，同时其自身的车辆动力学也很复杂。这些复杂因素会给有效的高速控制与规划带来挑战。本文提出一种多步动力学预测框架，可显式处理建模误差的累积，并能适用于基于采样的控制器。我们的方法采用经过特殊初始化的长短期记忆网络（LSTM），在有限时间范围内作为混合模型中的学习组件，预测一辆四座全地形车（Polaris S4 1000 RZR）在两种不同环境中的动力学。通过让 LSTM 仅在固定时间范围内进行预测，我们避免了对长期稳定性的需求，而长期稳定性往往是训练循环神经网络时面临的挑战。我们的框架灵活易用，只需里程计信息作为标签。通过大量实验，我们证明，该方法能够在具有挑战性的越野驾驶场景中实时预测数百万条可能的轨迹，时间范围可达五秒。

I. 引言

模型预测控制（模型预测控制）方法多年来一直用于实时控制应用 [1]–[4] ，以实现对复杂系统的实时控制。这些方法能够通过快速重新规划来处理噪声输入和不确定性。然而，这些方法的主要限制因素之一是需要快速且准确的动力学模型。

从历史上看，最常见的动力学模型是通过系统辨识获得的线性近似[5]。现实世界固有的非线性特性促使模型转向非线性或线性化近似，并围绕基于物理的近似这一核心原则展开。不幸的是，这些方程的计算难度极高，需要进行简化假设 [1],，或依赖难以准确确定且代价高昂的参数 [6]。因此，我们通常采用忽略或大幅简化非线性效应的第一性原理模型 [7]。即使存在这一限制，模型预测控制方法仍能够完成具有挑战性的现实任务。通过快速重新规划，即使显著的模型误差也可以得到缓解。

[1]佐治亚理工学院自主控制与决策系统实验室，美国佐治亚州亚特兰大市，邮编30313（通讯作者：*jgib- son37@gatech.edu*）[2] 美国国家航空航天局喷气推进实验室，加利福尼亚理工学院，美国加利福尼亚州帕萨迪纳市。本文提供了补充视频，可通过以下链接观看：https://youtu.be/bzyn-qpRJIk

快速重新规划会带来计算限制；当考虑基于采样的控制器时，这种限制会进一步放大，因为每秒预测所需的前向传播次数可能超过数百万次。不过，在不牺牲计算效率的情况下提高动力学准确度的需求，促使黑箱学习方法在动力学建模中日益普及。学习组件的发展方向大致可分为以下几类方法：高斯过程（GPs） [2], [8], 神经网络 [4], [9], 以及这篇关于动力学模型学习的综述论文中所概述的其他方法 [5]。对于现实世界中的硬件系统，每种方法都可能存在缺点。在为模型预测控制选择模型时，计算量与预测准确度之间的平衡是主要限制因素。使用基于物理的方程来整合预测的加速度或速度，可以简化学习问题，并改善各种系统中的泛化能力 [10] [11]。采用这种结构的模型通常称为混合模型。

虽然模型预测控制中的动力学模型有多种架构和框架，但它们的使用方式往往非常相似，并且存在两个主要缺陷。首先，轨迹通常通过递归方式计算，即使用前几步的输出作为模型下一步的输入。这种固有的递归特性会导致准确度随着时间范围的增加而显著下降，而且轻微的高估就可能迅速放大状态值，进而产生毫无意义的轨迹。其次，单步学习的一个更隐蔽的缺陷是，在学习问题中，较低频率的现象实际上会被视为随机噪声。事实上，先前研究已经表明，在单步预测中，为不同系统选择 Δt 的敏感性 [12] 。因此，解决这两个问题的一个好办法是将轨迹预测问题视为多步预测问题，而不是单步预测问题。

这些模型的递归使用方式使其特别适合采用专为循环计算设计的模型结构。在本文中，我们主要关注长短期记忆网络（LSTM）架构 [13], [14]，因为它能够避免其他循环结构中常见的梯度消失问题。我们将借鉴 [15] 的做法，使用另一个神经网络来初始化 LSTM 的隐藏状态和细胞状态。我们的 LSTM 预测模型只需在固定的时间范围内运行，因此无需针对持续稳定性进行训练。

在这项工作中，我们展示了一种混合架构，该架构能够成功学习同一车辆在两种不同环境中的动力学。我们将

长短期记忆网络 [15] 与积分损失函数 [16]相结合，同时确保神经网络足够小，可以在硬件系统上实时运行。通过使用积分损失函数，我们能够构建一种对环境变化具有鲁棒性的建模架构，同时训练时仅需要相对平滑的里程计估计。这使得人类驾驶数据和自主控制数据能够无缝集成到训练过程中。我们强调，具体的混合设置应被视为一个仍有广泛探索空间的超参数。关键在于积分损失函数能够对任意变换进行预测和集成。据我们所知，这是首项将初始化的长短期记忆网络与积分损失函数相结合的工作。实验结果在真实车辆上得到验证，使用了650，000条驾驶数据轨迹。

II. 相关研究

在自动驾驶赛车领域，精确的动力学建模对于实现车辆操控极限下的性能至关重要。 [1] 采用了一种运动学-动力学模型，在名义自行车模型的基础上，纳入了空气阻力、道路坡度效应、转向不足/转向过度特性，以及车辆相对于最优赛车线路的位置。借助该模型，他们在迭代式模型预测控制求解器中实现了实时性能。然而，该模型假设轮胎不会发生侧滑，因此不适用于越野条件。 [17] 展示了一种模型预测控制方法，该方法所用的模型将轮胎力学习为相对于预先选定的轮胎模型库的偏差。这使车辆能够在积雪地形上实现控制，但生成轮胎模型库需要大量领域专有知识。我们的方法降低了构建模型所需的知识量。

多年来，研究人员一直在探索使用学习得到的模型进行模型预测控制，以在准确度与计算复杂度之间取得可接受的平衡。 1995, [4]使用前馈神经网络（FNN），以泵流量为控制量，学习工厂中的 pH 值动力学。他们成功证明，采用单步预测损失训练的模型与模型预测控制结合后，在实现并维持目标 pH 设定值方面，性能优于比例积分控制器。此后， [18]在基于采样的模型预测控制器中使用通过单步预测损失函数训练的神经网络，以捕捉非线性行为，例如车辆转弯时的漂移，从而提升了性能。

一种在学习模型时防止这种递归误差放大的方法，是使用高斯过程 [2], [8]，当预测点接近先前采集的数据点时，其误差仍能保持有界。 [2] 使用稀疏高斯过程（SGP）学习名义自行车动力学模型与从电动赛车采集的真实数据之间的差异。这种组合使他们能够实时运行迭代式模型预测控制求解器，并根据新的环境信息实时更新模型。然而，这类模型仅适用于可以反复经过同一位置的问题，例如赛车场。 [11][19] 使用长短期记忆网络模型，在混合模型中预测动力系统，但没有任何一种方法采用积分损失函数，因此无法应用我们的混合模型架构。

III. 模型学习框架

我们将使用模型预测路径积分控制（MPPI）算法作为基准 MPC 方法进行计算比较。最终，我们希望模型能够以约 15 Hz 的频率计算 18,432 个样本。预测的目标时间范围为 $T = 5.0$ ，$\Delta t = 0.02$。这相当于每秒进行 $\approx 69,000,000$ 次动力学前向传播。我们的 MPPI 具体实现将能够像先前发表的研究 [20]一样，通过 CUDA 利用 GPU。我们具体以 NVIDIA RTX 3080 为目标硬件，但这些神经网络也能在较小的硬件系统上实时运行。我们将遵循先前关于混合动力学模型的研究 [5]，预测我们关注的状态导数，并使用相关运动学方程进行积分。我们的整体架构可总结于图 1。

初始化长短期记忆网络：我们现在简要回顾 [21] ，该研究提出了一种初始化长短期记忆网络的新方法。长短期记忆网络的训练通常需要解决如何初始化其隐藏状态和细胞状态的问题。循环网络最常见的做法是设定一个预热期，在预定义的时间段内让神经网络处理先前的数据 [22]。先前研究将预热期视为超参数；由于无法对预热期内使用的数据计算损失，因此实际上会丢弃这些数据。此外，预热也可能导致训练过程不稳定 [23]。 [21] 将隐藏状态/细胞状态的初始化问题纳入训练过程。具体做法是让一个神经网络利用某个预定义的历史缓冲区来预测另一个神经网络的初始隐藏状态/细胞状态，该历史缓冲区记为 τ。

后续论文 [15] 主要探讨哪种初始化网络最为合适，并对洗出机制、前馈神经网络和循环神经网络（RNN）进行了全面比较。大量实验结果表明，对于规模非常大的神经网络，循环神经网络是最合适的架构。研究还探讨并证明，混合架构能够以更少的权重更准确地预测更远的未来。我们将在越野环境下的自动驾驶场景中继续这一分析。通用的初始化网络架构可表示为以下方程。

$$\begin{bmatrix}\mathbf{c}^I_{t-\tau} & \mathbf{h}^I_{t-\tau}\end{bmatrix}^T = \kappa_I(\mathbf{x}_{t-\tau}, \mathbf{0}, \mathbf{0}) \tag{1}$$

$$\vdots$$

$$\begin{bmatrix}\mathbf{c}^I_t & \mathbf{h}^I_t\end{bmatrix}^T = \kappa_I(\mathbf{x}_t, \mathbf{c}^I_{t-1}, \mathbf{h}^I_{t-1}) \tag{2}$$

$$\begin{bmatrix}\mathbf{c}^P_t & \mathbf{h}^P_t\end{bmatrix}^T = \vartheta_I(\mathbf{h}^I_t, \mathbf{x}_t) \tag{3}$$

$$\begin{bmatrix}\mathbf{c}^P_{t+1} & \mathbf{h}^P_{t+1}\end{bmatrix}^T = \kappa_P(\mathbf{x}_t, \mathbf{c}^P_t, \mathbf{h}^P_t) \tag{4}$$

$$\zeta_t = \vartheta_P(\mathbf{h}^P_{t+1}, \mathbf{x}_t) \tag{5}$$

其中， κ_I 表示长短期记忆网络初始化器， ϑ_I 是前馈神经网络初始化输出神经网络， κ_P 表示长短期记忆网络

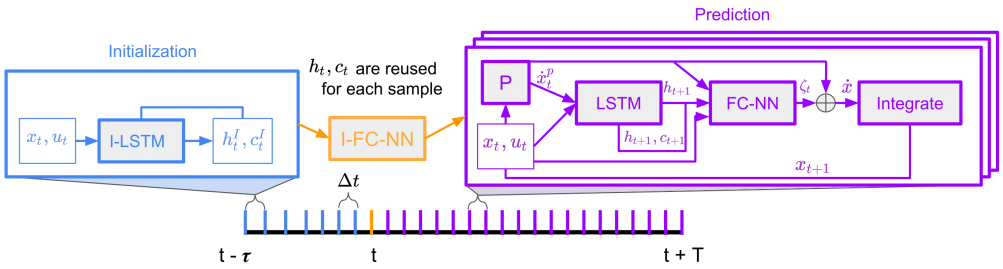

图 1. 展示整体架构的示意图。请注意，初始化网络仅运行一次， h_t， c_t 会在使用预测步骤采样的每条轨迹中重复使用。

预测网络， ϑ_P 表示 FNN 预测输出网络，而$\mathbf{h}_t^I, \mathbf{c}_t^I, \mathbf{h}_t^P, \mathbf{c}_t^P, \mathbf{x}_t, \zeta_t,$ 分别表示时间 t 时的初始化隐藏状态向量和细胞状态向量、预测隐藏状态向量和细胞状态向量、状态向量以及输出向量。需要特别指出的是，初始化网络仅在历史数据上运行，不包含有关车辆未来控制的信息。这意味着基于采样的方法可以在所有样本中重复使用相同的初始化输出。此外，对初始化网络的输入没有任何限制；它可以使用与预测网络不同的输入向量。

积分损失函数：积分损失函数主要通过惩罚位置上的真实状态轨迹偏差来计算损失[16]。本质上，这不仅惩罚了集成的准确度，也惩罚了神经网络的预测结果。 [16] 的主要优势可概括为以下几点。首先，我们只需要平滑的位置估计，因此无需对噪声里程计读数进行微分来获取加速度，也无需计算难以获得的近似输出。损失函数的输入仅为位置估计，因此学习模型可以预测任意值，而为这些值确定标签可能颇具挑战。我们在混合架构中利用这一点，在车轮坐标系中预测车轮上的阻力。其次，我们明确惩罚累积误差，从而能够在遥远的未来进行稳定的长期预测，并考虑动力学的缓慢变化。该方法的两个主要缺点是：在最初几次预测中，可能会出现损失较低但输出不平滑的情况；并且在学习过程中，损失函数的复杂度会增加。第一个问题在本研究中仍持续存在，第二个问题则通过使用现代反向传播库得到缓解 [24]。

IV. 混合模型设置

我们学习的车辆动力学针对的是 Polaris S4 1000 RZR。该车配备了线控驾驶套件，既支持自动驾驶，也能记录人类驾驶员的控制输入。车辆的控制输入包括期望油门、制动和方向盘位置。在自动驾驶过程中，必须由实体执行器转动方向盘并施加制动，这会带来显著的延迟。油门执行的延迟非常小，可以安全地忽略。人类驾驶数据中不存在这些延迟，而且人类操控这些控制装置的速度远快于自动驾驶系统。这带来了一个问题：获取人类驾驶数据的成本较低，而且车辆可以开得更激进。为了利用这些数据，我们设计了一种模型架构，使方向盘角度和制动指令能够与车身加速度预测解耦。模型的输入包括来自里程计的位置和速度，以及根据高程图推导出的横滚和俯仰近似值。我们的模型分为以下三个主要组成部分。

前两个组件预测制动和转向执行延迟。这些组件仅使用自主驾驶数据进行训练，并将真实里程计和控制指令作为输入。如下文所述，这两个系统都可以通过速率受限的一阶滞后与神经网络相结合进行参数化建模，

$$p_b = \min\left(\max\left(\left(u_t^b - b_t^u\right) C_B, -\dot{b}_{lim}^u\right), \dot{b}_{lim}^u\right) \tag{6}$$

$$\zeta_t^b = \vartheta_P^b\left(b_t^u, u_t^b, p_b\right) \tag{7}$$

$$b_{t+1}^u = b_t^u + \left(p_b + \zeta_t^b\right)\Delta t \tag{8}$$

其中， p_b 是具有速率限制 $\dot{b}_{lim}^u$ 和常数 C_B的模型的参数化近似， u_t^b 是制动指令， b_t^u 是当前制动状态。 ζ_t^b 是神经网络的输出，其后的变量表示输入。转向延迟由以下方程建模，

$$p_\delta = \min\left(\max\left(\left(u_t^\delta - \delta_t\right) C_\delta, -\dot{\delta}_{lim}^u\right), \dot{\delta}_{lim}^u\right) \tag{9}$$

$$\zeta_t^\delta = \vartheta_P^\delta\left(v_t^x, u_t^\delta, \delta_t, \dot{\delta}_t, p_\delta\right) \tag{10}$$

$$\delta_{t+1} = \delta_t + \left(p_\delta + \zeta_t^\delta\right)\Delta t \tag{11}$$

其中，符号与上文一致，但 δ 表示转向角位置， v_t^x 表示车辆车体坐标系下的前向速度。上述方程中的参数通过 Adam 优化器 [25] 在纯参数化版本上进行拟合。损失函数为状态的均方误差（δ， b_t^u）。

第三个组成部分由两个不同的神经网络构成。发动机模型用于预测发动机施加在车轮上的力， ζ_t^e，地形动力学模型用于预测与环境相关的阻力，ζ_t^t。坐标系如图所示。2。由于车辆质量在所有方程中均保持不变，因此我们对其进行归一化。神经网络的输出乘以 10 进行缩放，该数值被选为神经网络预期输出的最大加速度

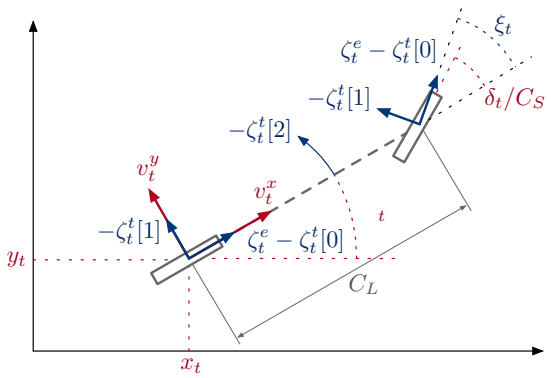

图 2。混合模型坐标系定义

值。这使预期输出范围缩小至 $[-1, 1]$，从而避免输出神经网络使用 tanh 激活函数时出现问题。

转向、制动和发动机模型在所有环境中共享，但地形动力学模型会根据当前所处的地形进行切换。最初，我们训练一个单一的地形动力学网络，随后在学习过程中逐步进行专门化。我们假设环境类别是先验地已知的，并在运行时加载正确的模型。环境是一个宽泛的定义，例如沙漠、林区等。我们认为，在未来的工作中，根据感知输入来预测环境是合理的，而且不会造成限制。组合方程概述如下，其中包含 $g = 9.81m/s^2$，

$$\xi_t = \tan\left(\delta_t / C_W\right) \tag{12}$$

$$p_t^{\psi} = \frac{v_t^x}{C_L} \tan\left(\frac{\delta_t}{C_S}\right) \tag{13}$$

$$\zeta_t^e = \vartheta_P^e\left(v_t^x, u_t^t, b_t^u\right) \tag{14}$$

$$\zeta_t^t = \vartheta_P^t(v_t^x, v_t^y, \dot{\psi}_t, \delta_t, \dot{\delta}_t, g\sin\left(\phi_t\right), g\sin\left(\theta_t\right), p_t^{\psi}, \xi_t, \zeta_t^e) \tag{15}$$

$$\dot{v}_t^x = \zeta_t^e - \zeta_t^t[0] + \left(\zeta_t^e - \zeta_t^t[0]\right)\cos\left(\xi_t\right) + \zeta_t^t[1]\sin\left(\xi_t\right) \tag{16}$$

$$\dot{v}_t^y = -\zeta_t^t[1] - \zeta_t^t[1]\cos\left(\xi_t\right) + \left(\zeta_t^e - \zeta_t^t[0]\right)\sin\left(\xi_t\right) \tag{17}$$

$$\dot{\psi}_t^t = p_t^{\psi} - \zeta_t^t[2] \tag{18}$$

其中，u_t^t 为指令油门，ϕ_t 为车辆横滚角，θ_t 为俯仰角，v_t^x 和 v_t^y 分别为车身纵向速度和横向速度。我们按照以下详细说明对上述神经网络预测结果进行积分，

$$x_{t+1} = x_t + \left(v_t^x\cos(\psi) - v_t^y\sin(\psi)\right)\Delta t \tag{19}$$

$$y_{t+1} = x_t + \left(v_t^x\sin(\psi) + v_t^y\cos(\psi)\right)\Delta t \tag{20}$$

$$\psi_{t+1} = \psi_t + \dot{\psi}_t^t\Delta t \tag{21}$$

$$v_{t+1}^x = v_t^x + \dot{v}_t^x\Delta t \tag{22}$$

$$v_{t+1}^y = v_t^y + \dot{v}_t^y\Delta t \tag{23}$$

首先使用 Adam 在完整数据集上拟合参数常数，以获得最佳的纯参数化近似，但在神经网络训练过程中允许这些常数发生变化。通常，这些模型的数值与初始值相差不超过 5%，期间进行训练。C_S 和 C_W 是非常相似的常数，不同之处在于 C_S 通过最小化 ψ 上的误差进行拟合，使用真实值和公式（13）。C_W 经过测量，能够在各种转向条件下正确近似车轮角度。

V. 实验设置

用于生成所有数据的车辆是一辆经过改装的 Polaris S4 1000 RZR，具体描述见第 IV 节。该车辆配备了车载传感器、计算设备和电源系统，旨在实现完全自主运行，并能够在多公里范围内完成越野 traversals。车载计算设备包括一颗 AMD Threadriper 3990X 处理器、8 条 32GB DDR4 内存，以及 4 块各配备 10GB 显存的 NVIDIA RTX 3080 显卡（其中仅分配一块 RTX 3080 用于模型预测控制计算）。车载传感器包括 3 台激光雷达（2 台朝前，1 台朝后，型号为 Velodyne VLP-32C）、4 台立体/RGB 相机（3 台朝前，1 台朝后，型号为 CRL S27），以及其他本研究中未使用的各类传感器。此外，该车辆还配备了惯性测量单元、GPS，以及车轮和悬架编码器。

输入：我们简要介绍动力学模型的两种外部输入，即里程计和高程图。里程计采用 [26] 中所述的方法生成。里程计的输入包括 IMU 和激光雷达数据。我们采用基于因子图的方法融合多种测量来源，包括惯性测量单元积分和 LIO-SAM[27]，，从而以 400Hz 的频率生成低延迟、高准确度的里程计更新，延迟为 <50ms，适用于捕捉车辆的剧烈运动。

高程图采用一种结合几何信息与语义信息的方法生成，该方法利用激光雷达和相机数据创建稠密且局部一致的环境地图。在本研究中，我们需要从传感器数据中可靠地估计地面高度和坡度。这个问题颇具挑战性，尤其是在植被茂盛程度各异的环境中，因为植被既可压缩，又会遮挡视线。仅依赖聚合激光雷达扫描生成的纯几何地图，不足以预测车辆穿越地形时轮胎将停驻的位置。基于先前研究（见 [28]），我们采用结合语义信息和几何信息的方法，利用语义线索从地面高度估计中去除可压缩植被。下面简要介绍该方法，更多细节超出了本文的讨论范围。首先，使用里程计对齐激光雷达点云和相机图像。然后对这些数据进行语义分类，以识别草、灌木、树木及环境中的其他可压缩元素。接着，采用基于 GPU 的光线追踪和体素建图方法 [29]，将激光雷达数据聚合到局部地图中。通过体素地图中的最小高度生成初始高程估计，再结合方差加权平均、基于语义的启发式方法和插值对该估计进行优化。

Helendale：第一个数据集包含加利福尼亚州 Helendale 附近的越野和非铺装道路驾驶数据，该地区位于莫哈韦沙漠（图3a）。数据集涵盖的具体区域内有许多约 10 米高的山丘，且非常干燥，



(a) Helendale (b) 帕索罗布尔斯

图3. 数据集采集所在两个环境的图片。

地面覆盖着松散的沙子。该数据集包含 $\approx$ 300，000 条人类驾驶数据轨迹，以及 $\approx$ 100，000 条自动驾驶数据轨迹。

帕索罗布尔斯：第二个数据集来自加利福尼亚州帕索罗布尔斯附近的越野小径和越野驾驶数据。该区域主要由干燥草地和橡树林地组成，称为灌木丛环境（图3b）。这部分数据包含大型陡峭斜坡，坡度最高达 40°，高度超过100米。地面完全覆盖着茂密的干草，车辆很容易失去牵引力。我们的数据集包含$\approx$ 100,000 条人工驾驶轨迹，以及 $\approx$ 150,000条自动驾驶数据轨迹。

VI. 结果

本节将介绍在 RZR车辆上进行的结果，该车辆采用 $T = 5$秒时间范围和 $\Delta t = 0.02$秒进行训练。主要结果汇总于表I。该表比较了基于全部组合数据集训练的5种不同模型架构。训练集约包含 $\approx$ 500,000 条轨迹，测试集包含$\approx$ 150,000 条轨迹。同一轨迹内可能存在重叠，因为从 $[t_0, t_T]$出发的一条轨迹可能包含另一条来自 $[t_1, t_{T+1}]$的轨迹数据。重叠仅限于验证集或测试集内部。这两个数据集均取自同一天的测试数据，但除此之外彼此独立。

H 表示第 IV 节概述的混合模型，而 K 表示仅运动学模型，I 表示使用初始化网络进行预测，因此 HI-LSTM 是一种带有初始化网络的混合架构。如果没有初始化网络，则预测网络的初始隐藏状态/细胞状态设为零。 K 神经网络直接预测输出 $\dot{v}_t^x$、$\dot{v}_t^y$、$\dot{\psi}$，不采用混合架构。K-LSTM 是一种不带初始化网络的长短期记忆网络，仅使用来自里程计和高程图的输入，但仍使用运动学公式 (19) 至 (23) 进行集成。这将 K-LSTM 限制为使用 (v_t^x、v_t^y、$\dot{\psi}_t$、δ_t、$\dot{\delta}_t$、ϕ_t、θ_t、u_t^t、u_t^b、η_t) 作为输入，其中 η_t 对帕索罗布尔斯为 -1，对 Helendale 为 1。这意味着 K-神经网络确实能够获取其运行所在的环境信息。初始化网络和预测网络采用相同的输入值。

我们的神经网络使用 PyTorch [24] 实现，用于训练和数据比较。实时结果则通过使用 C++/CUDA 编写的自定义长短期记忆网络（LSTM）实现获得。所有神经网络均训练 10 个轮次，其中最后 3 个轮次允许地形动力学网络针对每种环境进行专门化。每种架构的网络规模采用以下符号表示：

$$[\kappa_I \text{ 输入维度}][\kappa_I \text{ 隐藏维度}]\times[\vartheta_I \text{ 架构}] \dashv [\kappa_P \text{ 输入维度}][\kappa_P \text{ 隐藏维度}]\times[\vartheta_P \text{ 架构}].$$

其中，符号沿用公式（1）至（5）中的定义。对于输出神经网络 ϑ，数字表示各层的大小。例如 5、10、3 表示一个以前馈神经网络为形式、输入 5 个值、中间层包含 10 个采用 tanh激活函数的神经元，并输出 3 个值的神经网络。

- *K-LSTM* 和*KI-LSTM*:
  - 延迟：$[2][60] \times [62, 100, 10] - [2][5] \times [7, 10, 1]$⊣ 转向：$[4][60] \times [64, 100, 10] - [4][5] \times [9, 5, 1]$⊣ 发动机/地形动力学组合：$[10][60] \times [70, 100, 60] \dashv [10][30] \times [40, 40, 3]$

- *H-LSTM* 和*HI-LSTM*:
  - 延迟：$[2][60] \times [62, 100, 10] - [3][5] \times [8, 10, 1]$⊣ 转向：$[4][60] \times [64, 100, 10] - [5][5] \times [10, 5, 1]$⊣ 发动机：$[3][60] \times [63, 100, 10] - [3][5] \times [8, 20, 1]$⊣ 地形动力学：$[10][60] \times [70, 100, 20] - [10][10] \times [20, 20, 3]$

请注意，*K* 版本中的发动机与地形动力学组合网络具有显著更多的权重 $\approx 3X$。权重增加是为了弥补缺少参数化近似来进行预测的问题。

优化 CUDA 代码性能的关键在于减少并行线程对内存的访问。为了并行化神经网络的计算，除了权重之外还需要使用一些额外内存。由于初始化网络只需对所有样本运行一次，因此可以保留在 CPU 上。只有预测网络需要在 GPU 上并行运行。CUDA 中各模型的内存使用量和运行时间如表 II所示。请注意，计算运动学网络所需的内存远超混合架构，但两者的结果相近。

作为基线对比所使用的参数化模型，是将发动机和地形动力学模型的线性近似与式（19）至（23）中概述的参数化分量相结合，并使用式（13）计算偏航角速度。该模型没有任何 v_t^y，只是对$\dot{v}_t^x$的线性近似，

$$\dot{v}_t^x = C_T u_t^t + C_B b_t^u - C_V v_t^x \tag{24}$$

其中 $C_T = 4.0$，$C_B = 10$，$C_V = 0.2$。这是采集自主驾驶数据期间使用的主要动力学模型，与模型预测路径积分配合时表现良好。

发动机/地形动力学性能：为了进行不受学习到的执行器延迟影响的比较，我们将模型与制动状态b_t^u、转向角 δ 和转向速率 δ的真实值进行了比较。这些模型在来自自动驾驶和人工驾驶的组合数据上运行。

表 I 组合和组合自主的平均误差与误差的第一标准差

Data Type	神经网络	距离 误差（米） t = 1	Yaw 误差（弧度） t = 1	距离 误差（米） t = 5	Yaw 误差（弧度） t = 5	距离 误差（米） 时间 = 10	Yaw 误差（弧度） 时间 = 10
C	P	0.377 ± 0.127	0.036 ± 0.002	4.203 ± 10.461	0.175 ± 0.054	11.153 ± 77.551	0.339 ± 0.184
	K-LSTM	$\mathbf{0.270} \pm 0.088$	$\mathbf{0.033 \pm 0.001}$	1.845 ± 2.309	$0.117 \pm \mathbf{0.013}$	5.023 ± 18.833	0.208 ± 0.046
	KI-LSTM	0.277 ± 0.087	0.036 ± 0.001	1.975 ± 2.476	$\mathbf{0.113} \pm 0.014$	$\mathbf{4.931} \pm 18.095$	$\mathbf{0.195 \pm 0.045}$
	H-LSTM	0.284 ± 0.096	0.042 ± 0.001	1.995 ± 2.708	0.138 ± 0.021	5.459 ± 21.833	0.249 ± 0.074
	HI-LSTM	$0.278 \pm \mathbf{0.078}$	0.049 ± 0.002	$\mathbf{1.765 \pm 2.021}$	0.132 ± 0.021	$4.981 \pm \mathbf{17.143}$	0.229 ± 0.065
CA	K-LSTM	0.539 ± 0.136	0.239 ± 0.026	7.259 ± 28.837	0.775 ± 0.307	18.159 ± 202.346	1.055 ± 0.520
	KI-LSTM	0.472 ± 0.109	0.177 ± 0.015	7.040 ± 21.287	0.813 ± 0.0307	21.015 ± 155.293	1.431 ± 0.688
	H-LSTM	0.376 ± 0.071	$\mathbf{0.036 \pm 0.001}$	2.465 ± 3.737	0.149 ± 0.016	6.239 ± 26.865	$0.276 \pm \mathbf{0.049}$
	HI-LSTM	$\mathbf{0.272 \pm 0.042}$	0.040 ± 0.001	$\mathbf{2.089 \pm 3.463}$	$\mathbf{0.145 \pm 0.016}$	$\mathbf{5.722 \pm 22.936}$	$\mathbf{0.268} \pm 0.051$

表 II 在 NVIDIA RTX 3080 上使用 18432 个样本进行 1 次模型预测路径积分迭代的平均 CUDA 运行时间

架构	CUDA 内存（kB）	CUDA 运行时间（ms）
P	0.024	23.327
K-LSTM 和 KI-LSTM	62.468	83.708
H-LSTM 和 HI-LSTM	38.08	59.346

驾驶。结果见表格 I的“组合”部分。所有模型在使用真实值时的表现都比较相似，大多数模型在 10 秒轨迹上的位置误差约为 5 米。这表明，长短期记忆网络能够预测长度达到训练时域两倍的稳定轨迹。对于发动机/地面动力学建模，混合架构似乎并未显著提升学习得到的网络的性能，这表明混合架构可能无助于学习。混合架构和集成的设计空间很广，不同的设置可能会带来性能提升。此外，初始化网络对准确度的影响有限。增大初始化网络的规模可能会提高准确度；未来的一个研究方向是详细分析规模变化对准确度的影响。

整个系统的性能：在本节中，我们沿用之前相同的神经网络，但为制动和转向延迟模型加入了对应的架构。因此，$\dot{K}$ 神经网络将直接使用长短期记忆网络预测 b_t^u 和 $\dot{\delta}$，而不借助任何参数化辅助。这仅针对自主驾驶数据集中的一个子集，该子集包含 $\approx 50,000$ 条轨迹，用作测试集。在这里，我们看到混合方法与运动学方法之间存在鲜明对比。在黑箱函数难以预测的情况下，制动和转向神经网络的表现不如其参数化对应模型。平均而言，它们的转向平均误差约为后者的 2 倍，而制动模型的预测准确度则非常接近。不过，制动预测误差具有误导性，因为大多数轨迹的制动状态始终为零，完全没有施加制动。观察各条状态轨迹可以发现，神经网络产生了轻微扰动，而这些扰动可能会显著影响性能。

较差的转向和制动预测性能会导致偏航误差随时间大幅增加，在较高速度下产生非常大的距离误差。值得注意的是，输出此前平滑的曲线往往会变得更加曲折。神经网络很可能依赖于 $\dot{\cdot}$ 对于 δ，δ，b_t^u 的精确值。一旦这些值由于建模误差而开始变得不准确，神经网络就会进入训练过程中未曾遇到过的输入序列，从而产生较差的结果。混合架构通过提供一个独立于长短期记忆网络内部状态的合理估计，降低了这种依赖性。

对于所有数值，神经网络都倾向于在单个时间步内将预测值从较大的正值振荡到较大的负值。$\dot{\psi}$ 和 $\dot{v}_x$ 更容易出现较大振荡，这可能是由于里程计标签中的噪声较大。该方法对漂移以及由 IMU 预积分约束下延迟的激光惯性里程计优化所导致的里程计小幅跳变很敏感。从逻辑上说，积分损失函数不会惩罚那些看起来不连续、但与里程计位置吻合良好的状态轨迹。漂移被视为神经网络需要预测的潜在运动，因此我们经常会看到速度接近零的短暂动力学不可行轨迹。通过增加针对速度的额外损失，这种情况在轨迹中有所减少，但仍需进一步研究里程计噪声的影响。总体而言，这些模型展现出很强的能力，能够对未来做出稳定的预测，但也倾向于过度平滑输出。由于速度中的里程计噪声若直接积分可能导致位置不准确，因此我们只在误差较大时才对速度损失进行惩罚。

VII. 结论

在本研究中，我们提出了一种适用于复杂越野环境的动力学模型学习框架。我们的方法将经过特别初始化的长短期记忆网络与积分损失函数相结合，从而支持灵活的学习输出。通过数十万条轨迹，我们证明了该方法能够泛化到两种不同的越野环境。我们的方法计算效率高，可在基于采样的模型预测控制方法中于我们的硬件上实时运行。

致谢

这项研究部分在加州理工学院喷气推进实验室开展，依据与美国国家航空航天局签订的合同（80NM0018D0004）。

参考文献

[1] E. Pagot、M. Piccinini 和 F. Biral，“采用最短时间机动和新型运动动力学模型的自动驾驶遥控车实时最优控制”，载于2020年IEEE/RSJ智能机器人与系统国际会议（IROS），（美国内华达州拉斯维加斯），第 2390 2396, 2020 1, 2 页–IEEE，2019 年 10 月。J. Kabzan 、 L. Hewing 、 A. Liniger 和 M. N. Zeilinger，“ 用于自动驾驶赛车的学习型模型预测控制”，IEEE机器人与自动化快报，第4卷，第3363–3370页，2019年10月。[3] G. Williams、P. Drews、B. Goldfain、J. M. Rehg 和 E. A. Theodorou，“信息论模型预测控制：理论及其在自动驾驶中的应用”，IEEE机器人学汇刊，第 34, 6, 1603 1622, 2018 1 卷，第 34, 6, 1603 1622, 2018 1 期，第 34, 6, 1603 1622, 2018 1 页–。A. Draeger、S. Engell 和 H. Ranke，“使用神经网络的模型预测控制”，IEEE控制系统杂志，第15卷，第61–66页，10月 1995 1, 2 。D. Nguyen-Tuong 和 J. Peters，“机器人控制的模型学习：综述”，认知加工，第 12, 319 40, 04 2011 1, 2 卷，第 12, 319 40, 04 2011 1, 2 页–。H. Hjalmarsson，“复杂结构化系统的系统辨识”，欧洲控制期刊，第15卷，310，2009 3 4，275 1，第 12, 319 40, 04 2011 1, 2 期，第 12, 319 40, 04 2011 1, 2 页–。[7] A. Freddi、A. Lanzon 和 S. Longhi，“四旋翼飞行器容错的反馈线性化方法”，IFAC会议论文集（IFAC-PapersOnline），第44卷，5418，2011 1 1，5413 1，第PART期，第[7] 页–。[8] L. Hewing、J. Kabzan 和 M. N. Zeilinger，“基于高斯过程回归的谨慎模型预测控制”，IEEE控制系统技术汇刊，第28卷，第6期，第2736–2743页，2017年。[9] G. R. Williams、B. Goldfain、K. Lee、J. Gibson、J. M. Rehg 和 E. A. Theodorou，“用于自适应模型预测控制的局部加权回归伪回放”，载于机器人学习会议，第 969 978, 2020 1 页–。Y. Yu、H. Yao 和 Y. Liu，“采用新型基于物理的学习方法进行飞机动力学仿真”，《航空航天科学与技术》，第 87, 254 264, 2019 1 卷，第 87, 254 264, 2019 1 页–。S. K. Singh、R. Yang、A. Behjat、R. Rai、S. Chowdhury 和 I. Matei，“Pi-LSTM：物理信息增强长短期记忆网络”，载于2019年第18届IEEE国际机器学习与应用会议（ICMLA），第 34 41, 2019 1, 2 页–。Y. Liu、J. N. Kutz 和 S. L. Brunton，“分层深度学习多尺度微分方程时间步进器”，《皇家学会哲学汇刊A辑》，第 1 22, 2020 1 页–。T. Zia 和 U. Zahid，“用于乌尔都语声学建模的长短期记忆循环神经网络架构”，《国际语音技术杂志》，第 22, 1, 21 30, 2019 1 卷，第 22, 1, 21 30, 2019 1 期，第 22, 1, 21 30, 2019 1 页–。S. Hochreiter 和 J. Schmidhuber，“长短期记忆”，《神经计算》，第 9, 1735 1780, 1997 1 卷，第 9, 1735 1780, 1997 1 页–。N. Mohajerin 和 S. L. Waslander，“使用循环神经网络对动力系统进行多步预测”，IEEE神经网络与学习系统汇刊，第30卷，第11期，第3370– 3383, 2019 1, 2 页。N. Seegmiller、F. Rogers-Marcovitz、G. Miller 和 A. Kelly，“通过集成预测误差最小化进行车辆模型辨识”，《国际机器人研究杂志》，第32卷，第8期，第912– 931, 2013 2, 3 页。K. Berntorp、R. Quirynen、T. Uno 和 S. Di Cairano，“通过摩擦自适应非线性模型预测控制实现变化路面上的自动驾驶车辆轨迹跟踪”，《车辆系统动力学》，第 58, 705 725, 2020 2 卷，第 58, 705 725, 2020 2 页–, 5月。G. Williams、P. Drews、B. Goldfain、J. M. Rehg 和 E. A. Theodorou，“采用模型预测路径积分控制的激进驾驶”，载于国际机器人与自动化会议（ICRA），第 1433 1440, 2016 2 页–IEEE，。Q. Teng 和 L. Zhang，“使用多步CLDNN进行数据驱动的非线性动力系统辨识”，《AIP进展》，第9卷，085311，08 2019 2 p.。[20]M. S. Gandhi、B. Vlahov、J. Gibson、G. Williams 和 E. A. Theodorou，“鲁棒模型预测路径积分控制：分析与性能保证”，IEEE机器人与自动化快报，第6卷，第2期，第1423–1430页，2021年。2

[21] N. Mohajerin、M. Mozifian 和 S. Waslander，“用于多步预测的四旋翼飞行动力学模型深度学习”，载于 *2018* 年 *IEEE* 国际机器人与自动化会议 *(ICRA)*，第 2454– 2459, 2018 2 。H. Jaeger，“循环神经网络训练教程，涵盖 bppt、rtrl、ekf 以及 echo state network 方法”，2002 2 。H.-G. Zimmermann、C. Tietz 和 R. Grothmann，“循环神经网络预测：12 个技巧”，载于 神经网络：技巧荟萃：第二版(G. Montavon、G. B. Orr 和 K.-R. M¨uller 编)，第 687–707 页，柏林、海德堡：Springer Berlin Heidelberg，2012 年。 2[24] A. Paszke、S. Gross、F. Massa、A. Lerer、J. Bradbury、G. Chanan、T. Killeen、Z. Lin、N. Gimelshein、L. Antiga、A. Desmaison、A. Kopf、E. Yang、Z. DeVito、M. Raison、A. Tejani、S. Chilamkurthy、B. Steiner、L. Fang、J. Bai 和 S. Chintala，“PyTorch：一种命令式风格的高性能深度学习库”，载于 神经信息处理系统进展 *32* (H. Wallach、H. Larochelle、A. Beygelzimer、F. d'Alch´e-Buc、E. Fox 和 R. Garnett 编)，第 8024–8035 页，Curran Associates, Inc.，2019 3, 5 。D. P. Kingma 和 J. Ba，“Adam：一种随机优化方法”，2014 3 。S. Fakoorian、K. Otsu、S. Khattak、M. Palieri 和 A. A. Agha-mohammadi，“ROSE：通过在线协方差自适应实现鲁棒状态估计”，国际机器人研究基金会 *(ISRR)*，2022 年。 4[27] T. Shan、B. Englot、D. Meyers、W. Wang、C. Ratti 和 R. Daniela，“LIO-SAM：通过平滑与建图实现紧耦合激光雷达惯性里程计”，载于 *IEEE/RSJ* 国际智能机器人与系统会议 *(IROS)*，第 5135 5142, 2020 4 –IEEE，。D. D. Fan 、 K. Otsu 、 Y. Kubo 、 A. Dixit 、 J. Burdick 和 A.-A. Agha-Mohammadi，“Step：面向风险感知越野导航的随机可通行性评估与规划”，载于 机器人学：科学与系统，第 1 21, 2021 4 –RSS Foundation，。T. Overbye 和 S. Saripalli，“G-VOM：GPU 加速的体素越野建图系统”，载于 *2022* 年 *IEEE* 智能车辆研讨会 *(IV)*，第 1480–1486 页，2022 年。 4