CCIL：面向城市驾驶的上下文条件模仿学习

郭可* † 魏靖†，陈俊博†，潘嘉*
*香港大学，电子邮箱：{kguo、jpan}@cs.hku.hk
† 阿里巴巴集团，电子邮箱：21wjing@gmail.com、chjb@zju.edu.cn

arXiv:2305.02649v1 [cs.RO] 4 May 2023

摘要—**模仿学习在解决复杂的城市自动驾驶任务方面具有巨大潜力，因为经验丰富的人类驾驶员能够轻松应对极具挑战性的场景。尽管行为克隆无需进行具有风险的在线交互，因此成为自动驾驶中广泛使用的模仿学习方法，但它受到协变量偏移问题的困扰。为解决这一局限，我们提出了一种上下文条件模仿学习方法，该方法利用策略将上下文状态映射为自车的未来轨迹，而不是沿用传统的同时基于自车状态和上下文状态预测自车动作的形式。此外，为减少坐标系中隐含的自车信息，我们设计了一种自车扰动目标导向坐标系。该坐标系的原点是自车位置加上零均值高斯扰动，其 x 轴方向指向目标位置。在真实世界的大规模 Lyft 和 nuPlan 数据集上的实验表明，我们的方法显著优于当前最先进的方法。**

I. 引言

在复杂城市环境中为自动驾驶车辆（SDV）规划一条安全、舒适度高且高效的轨迹，是自动驾驶领域一项具有挑战性且至关重要的任务 [1]。不同于高速公路驾驶 [2], , 城市驾驶需要应对各种道路几何形态，例如环岛和交叉路口，同时还要与交通信号灯、行人及其他车辆进行交互。传统的基于规则的方法 [3] 在行业中已经取得了一定成功，但需要大量人力工程才能处理多样化的现实场景。近年来深度学习技术的发展促使研究人员 [4, 5] 使用神经网络对复杂的驾驶策略进行建模。利用人类驾驶员示范进行模仿学习（模仿学习）是一种很有前景的学习这些策略的方法，因为经验丰富的驾驶员能够处理最困难的情况，而且他们的驾驶数据可以大规模轻松收集。

最简单的模仿学习算法是行为克隆（BC）方法，它在自动驾驶领域有广泛应用 [6, 4, 7]。该方法以监督学习的方式，通过最小化学习者采取的动作与专家在专家状态分布中采取的动作之间的差异来学习策略，而无需进行可能存在危险的在线交互。尽管简单易用，BC 方法仍受到协变量偏移 问题的影响 [8], 即学习者策略诱导的状态会随着时间推移不断偏离专家的分布。

为解决协变量偏移 问题，现有方法（如 DAgger [8] 和 DART [9] ）会在学习者状态或受扰动的专家状态下查询监督者的纠正反馈。由于人工监督数据难以收集，近期研究（如 GAIL [10]）旨在通过基于神经网络的判别器提供反馈，以帮助学习者从其策略所引发的分布外状态中恢复。然而，这些数据增强方法要么需要专家监督，要么需要在真实世界中执行学习者策略的滚动运行，这在自动驾驶领域并不切实际。另一种方法是，一些研究人员尝试通过融入控制理论先验知识来约束学习策略的形式，以确保其对策略误差具有鲁棒性，因为现实世界中的系统通常具有鲁棒性。例如，Palan 等人 [11]，Havens 和 Hu[12] 对学习得到的线性策略施加卡尔曼约束或线性矩阵不等式约束，以确保线性时不变系统的闭环稳定性。Yin 等人 [13] 通过将策略构建为简单的前馈神经网络，进一步拓展了这一方法。此外，East [14] 将 [12] 中提出的方法扩展至多项式策略和动力系统。然而，城市驾驶任务过于复杂，无法通过这些简单的线性或多项式策略形式来处理。

为了通过模仿离线的人类示范来学习稳定且具有泛化能力的城市驾驶策略，我们提出了一种**上下文条件模仿学习**（CCIL）方法。与行为克隆不同，我们的方法采用策略网络，仅根据自动驾驶车辆的上下文状态预测其未来轨迹，而不是同时将自车状态和上下文状态作为输入来生成下一步动作 [15, 16]。在我们的方法中，自车状态表示自动驾驶车辆的历史轨迹，而上下文状态则包含所有其他观测到的物体的状态，以及自动驾驶车辆的目标位置。

我们的方法主要基于这样一个事实：自车状态极易受到策略误差的影响。策略输入中的微小分布偏移也可能导致更大的动作误差，并最终引发分布外状态。此外，在自动驾驶任务中，上下文中的静态元素（例如车道或人行横道）不受自动驾驶车辆的影响，而人类驾驶员等动态元素则会尝试从自动驾驶车辆造成的扰动中恢复。交通系统的这一稳定性特征可用于解决分布偏移问题，方法是仅将上下文作为策略输入。此外，从策略输入中移除自车状态也有助于克服因果混淆造成的惯性问题 [17]。例如，当自车停下时，训练数据通常显示其保持静止的概率很高。这会导致低速与不加速之间形成虚假相关性，使模仿策略难以重新起步。然而，我们的

方法会屏蔽自车历史信息，从而避免形成此类虚假相关性。

然而，在实践中，仅从策略输入中移除自车历史信息还不够。策略输入通常会转换到车辆坐标系中，以自动驾驶车辆的后轴中点为原点，并将朝向设为 x 轴方向 [4, 18]。然而，使用这种自车中心坐标系仍可能隐式泄露自车信息。这是因为，自车中心坐标系中的观测分布会随坐标原点的选择而变化，而规划策略可以利用这一点推断自车信息。物理学中的非惯性参考系可以帮助我们理解这一概念。在自车中心参考系中，经历加速度的运动主体（例如汽车）可以根据局部观测推断出自身的一些运动信息，从而泄露自车运动信息。为尽量减少这种隐式信息泄露，我们设计了一种受自车扰动影响较小的目标导向坐标系。该坐标系的原点为自动驾驶车辆当前位置加上零均值高斯扰动，x 轴方向则指向其目标位置。

本文的主要贡献可总结如下：

1. 我们提出了一种新颖的上下文条件模仿学习方法，以解决离线模仿学习中的协变量偏移 问题。我们的方法学习一种策略，仅以自动驾驶车辆的上下文为输入来预测其未来轨迹，并基于上下文稳定性假设提供鲁棒性保证。

2. 我们通过移除策略输入中的显式自车状态信息，并提出自车扰动目标导向坐标系，以最大限度地减少坐标系中的隐式自车信息，从而将我们的方法专门应用于城市驾驶场景。

3. 我们在真实世界的大规模城市驾驶数据集 Lyft [19] 和 nu-Plan [20]，上验证了我们方法的有效性，并达到了最先进的性能基准。我们方法的视频和代码可在 https://sites.google.com/view/contextconditionedil 找到。

II. 相关工作

A. 自动驾驶模仿学习

在自动驾驶中应用模仿学习（IL），旨在通过模仿人类驾驶员行为来教会自动驾驶车辆驾驶。最直接的方法是行为克隆（BC），它通过最小化学习者与专家在专家状态下的动作差异来进行训练，无需额外的人工标注数据或在线交互。早期将行为克隆应用于自动驾驶的案例，如 ALVINN [6] 和 PilotNet [4]，利用大量人类驾驶经验，学习能够将传感器输入直接映射为车辆控制指令的端到端策略。近年来，ChauffeurNet [16]通过利用感知结果生成中间规划，提高了泛化能力和透明度。然而，行为克隆方法往往会导致训练分布与部署分布之间出现协变量偏移。这意味着，即使策略中存在轻微错误，也可能导致车辆偏离专家状态，从而造成更大的错误。为应对这一挑战，现有的模仿学习方法可分为在线方法、离线无模型方法和离线基于模型的方法。

1) 在线模仿学习：在线方法旨在直接匹配专家状态-动作分布，而不是像行为克隆中那样匹配专家状态条件动作分布。例如，[21],中基于 DAgger [8], 提出的方法，会查询学习者访问状态下的监督者动作，并将新数据添加到数据集中。这样可以调整专家状态分布，使其匹配学习者的状态分布，从而消除对交互式专家的要求。为了摆脱对交互式专家的要求，[22] 中基于 GAIL [10]的方法利用判别器衡量学习者与专家的状态-动作分布之间的差异，然后在强化学习中计算奖励。通过提高策略的累积奖励，策略的状态分布将更加接近专家分布。然而，在自动驾驶等安全关键任务中部署此类强化学习循环颇具挑战，因为它要求与环境进行在线交互。相比之下，我们提出的方法匹配以上下文为条件的自车未来轨迹分布，该分布可以通过监督学习进行学习，无需与环境交互，也无需获得专家监督。这样既避免了与环境进行在线交互的需求，也有利于在自动驾驶等安全关键场景中部署。

2）离线无模型模仿学习：目前最流行的无模型模仿学习方法基于 DART [9]，它通过提供如何从偏离状态中恢复的合成示例来避免复合误差。在 [7],中，向轨迹注入时间相关噪声，以模拟车辆逐渐偏离期望轨迹的过程。另一种方法是，ChauffeurNet [16]对自动驾驶车辆的当前位姿施加均匀扰动，即对其后轴中点坐标和朝向进行扰动，并拟合一条新的平滑轨迹，使自动驾驶车辆返回原始目标位置。然而，这些基于规则的轨迹增强方法难以覆盖由学习者策略引发的真实运动分布，而且策略很可能逐渐形成扰动驾驶的倾向。我们的方法也会对自动驾驶车辆的当前位置施加扰动，但扰动的作用是模糊自车的位置信息，而非提供数据增强。因此，我们的方法在训练期间不需要进行轨迹平滑处理。我们的方法同样是无模型的，但旨在通过约束策略的形式，使其具备鲁棒性，而不依赖恢复示例。

3）离线基于模型的模仿学习：为了解决分布偏移问题，基于模型的模仿学习方法会最小化在可微分的学习模型或数据驱动模型中展开的轨迹与专家轨迹之间的差异。例如，PPUU [23]基于变分自编码器 [24]学习数据驱动的动力学模型，并训练策略网络输出能够生成与专家轨迹相似轨迹的动作。由于动力学模型可微分，动作可以接收来自多个未来时间步的梯度，从而使得

对那些未来会导致较大偏差的动作进行惩罚，即使它们的瞬时偏差较小。UrbanDriver [18]，并未学习模型，而是利用记录的感知数据和高精地图构建一个可微分的数据驱动模型，并根据自车的位姿和采集的数据，通过坐标变换计算新的观测。然而，基于模型的方法的性能受限于模型的准确性。

B. 具有鲁棒性的模仿学习

模仿学习与监督学习不同，它会在动力学作用下部署策略，而鲁棒性被视为学习策略从策略误差中恢复的能力。研究者已探索控制理论方法，通过约束策略和系统动力学来学习具有稳定性保证的策略。Taylor Series IL [25] 表明，如果学习者与专家在专家状态下的导数差异较小，那么二者产生的轨迹将十分接近。然而，在数据不足时，很难计算专家策略的高阶导数。其他研究则通过对策略施加约束，为线性动力系统学习鲁棒的线性 [11, 12] 或简单的前馈神经网络 [13] 控制策略。尽管 East [14] 中的作者将鲁棒性保证扩展到了多项式系统和策略，但要为非线性自动驾驶系统获得闭环稳定性保证仍然是一项挑战。近期，CMILe 方法 [26]学习的非线性策略具有与专家相同的安全性保证，但与 DAgger [8]类似，它需要在线访问专家。相比之下，我们的方法将策略约束在接收上下文状态并生成自车未来轨迹的形式中；在对环境动力学作出温和的输入到状态稳定性假设时，可以保证其稳定性。

III. 理论分析

我们首先介绍本文中使用的符号和定义。对于任意向量 $\boldsymbol{x} \in \mathbb{R}^n$，$\|\boldsymbol{x}\|_p$ 表示其 L^p范数，而 $\|\boldsymbol{x}\|$ 表示其 L^2 范数。对于任意方阵$\boldsymbol{A} \in \mathbb{R}^{n\times n}$，$\|\boldsymbol{A}\|$ 表示其诱导 L^2 范数，$\rho(\boldsymbol{A})$表示其谱半径（其特征值绝对值的最大值）。对于每一种诱导矩阵范数，我们都有 $\rho(\boldsymbol{A}) \leqslant \|\boldsymbol{A}\|$。

定义 1 (比较函数)。函数 $\gamma : \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ 属于类 $\mathcal{K}$，当且仅当它连续、严格递增且满足 $\gamma(0) = 0$。函数 $\beta(x, t)$: $\mathbb{R}_{\geqslant 0} \times \mathbb{R}_{\geqslant 0} \to \mathbb{R}_{\geqslant 0}$ 属于类 $\mathcal{KL}$，当且仅当它连续，且对每个 t，$\beta(\cdot, t)$ 属于类 $\mathcal{K}$，并且对每个 x，$\beta(x, \cdot)$ 递减。

定义 2 (输入到状态稳定性（ISS） [27])。如果存在一个类 $\mathcal{KL}$ 函数 β 和一个类 $\mathcal{K}$ 函数γ，使得对于每个有界输入 $\boldsymbol{u}$ 以及初始条件 $\boldsymbol{\xi}$和 $t \in \mathbb{Z}_+$，均满足以下条件，则称离散时间系统 $\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t)$ $\boldsymbol{x}_0 = \boldsymbol{\xi}$ 是输入到状态稳定的:

$$\|\boldsymbol{x}_t(\boldsymbol{\xi}, \boldsymbol{u})\| \leqslant \beta(\|\boldsymbol{\xi}\|, t) + \gamma(\|\boldsymbol{u}\|_\infty), \tag{1}$$

其中 $\|\boldsymbol{u}\|_\infty = \text{上确界}_{t\in\mathbb{Z}_+}(\boldsymbol{u}_t)$ 是输入的上确界范数。如果一个系统满足 *ISS*性质，则任意两条轨迹之间的距离最终必定受其输入信号所界定，且与初始条件无关。

我们考虑在如下非线性离散时间系统中执行模仿学习:

$$\boldsymbol{x}_{t+1} = f(\boldsymbol{x}_t, \boldsymbol{u}_t), \tag{2}$$

其中，$\boldsymbol{x}_t \in \mathbb{R}^n$ 是时间 $t \in \mathbb{N}$ 时的系统状态，$\boldsymbol{u}_t \in \mathbb{R}^u$ 是控制输入。对于自动驾驶系统，其状态可以分为两部分 $\boldsymbol{x}_t = (\boldsymbol{e}_t, \boldsymbol{c}_t)$: 由控制输入 $\boldsymbol{u}$ 控制的自车状态 $\boldsymbol{e}_t \in \mathbb{R}^m$，以及受自车状态影响的上下文状态 $\boldsymbol{c}_t \in \mathbb{R}^{n-m}$。将策略误差视为系统的扰动，我们在专家状态$\boldsymbol{x}^*$ 处对非线性系统进行线性化，得到以下线性子系统:

$$\boldsymbol{c}_{t+1} = \boldsymbol{A}\boldsymbol{c}_t + \boldsymbol{B}\boldsymbol{e}_t, \tag{3}$$

带有解的:

$$\boldsymbol{c}_{t+1} = \boldsymbol{A}^{t+1}\boldsymbol{c}_0 + \sum_{j=0}^{t} \boldsymbol{A}^{t-j}\boldsymbol{B}\boldsymbol{e}_j. \tag{4}$$

这里我们稍微滥用一下符号，用相同的符号 $\boldsymbol{c}_t$, $\boldsymbol{e}_t$表示上下文偏差 $\boldsymbol{c}_t - \boldsymbol{c}_t^*$ 和自车状态偏差$\boldsymbol{e}_t - \boldsymbol{e}_t^*$。

由于没有自车状态偏差 $\boldsymbol{c}_{t+1} = \boldsymbol{A}\boldsymbol{c}_t$ 的现实世界交通系统是稳定的，因此我们有 $\rho(\boldsymbol{A}) < 1$。对于这样的舒尔矩阵，存在常数 $c > 0$ 和 $0 \leqslant \sigma < 1$，使得 $\|\boldsymbol{A}^t\| \leqslant c\sigma^t$ [28]。此外，交通系统中的上下文通常包括不受自车影响的静态地图元素，以及能够迅速从其他车辆扰动中恢复的人类驾驶员。自然可以假设，式 (3) 中的子系统满足 ISS性质，其中 $\beta(r, t) < \sigma^{t-1}r$ 且 $\gamma(r) \leqslant \epsilon r$，其中 $\epsilon \geqslant 0$。结合式 (1) 和式 (4)，我们有:

$$\begin{aligned} \beta(r, t) &= c\sigma^t r < \sigma^{t-1} r, \\ \gamma(r) &= \sum_{j=0}^{\infty} c\sigma^t \|\boldsymbol{B}\| r = \frac{c\|\boldsymbol{B}\|}{1-\sigma} r \leqslant \epsilon r, \end{aligned} \tag{5}$$

这意味着 $c < \frac{1}{\sigma}$，且 $\|\boldsymbol{B}\| \leqslant {}^{\epsilon}\!/_{(1-\sigma)\ c}$。

接下来，我们研究在此 ISS 系统下，如何使我们所提出的策略保持稳定。我们学习一种上下文状态反馈控制 $\boldsymbol{e}_t = \boldsymbol{u}_t = f_{\boldsymbol{\theta}}(\boldsymbol{c}_t)$，它将上下文状态直接映射到自车状态。为简单起见，我们考虑线性策略$\boldsymbol{u}_t = \boldsymbol{K}\boldsymbol{c}_t$。于是，式 (3) 中的线性子系统可简化为 $\boldsymbol{c}_{t+1} = (\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K})\boldsymbol{c}_t$，其稳定条件为 $\rho(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}) < 1$。因此，当 $\|\boldsymbol{K}\| < {}^{c}\!/_{(1-c\sigma)\,\epsilon(1-\sigma)}$时，我们有，

$$\begin{aligned} \rho(\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}) &\leqslant \|\boldsymbol{A} + \boldsymbol{B}\boldsymbol{K}\| \leqslant \|\boldsymbol{A}\| + \|\boldsymbol{B}\|\|\boldsymbol{K}\| \\ &\leqslant c\sigma + \frac{\epsilon(1-\sigma)}{c}\|\boldsymbol{K}\| < 1. \end{aligned} \tag{6}$$

该结果表明，只要上下文系统足够稳定，且我们的策略范数足够小，就可以保证闭环自动驾驶的稳定性。

在实践中，我们会在神经网络策略的训练损失中加入 a L^2 范数正则化，使其内化这一稳定性先验。

最后，我们展示即使在上下文 $\boldsymbol{c}_t = 0$ 固定的情况下，朴素的行为克隆策略形式仍可能失效。该形式同时将自车状态和上下文状态作为输入，并输出相对于当前状态的位移。我们考虑一个线性策略：$\boldsymbol{u}_t = \boldsymbol{K}_{bc}\boldsymbol{e}_t$并更新状态 $\boldsymbol{e}_{t+1} = \boldsymbol{e}_t + \boldsymbol{u}_t$。随后，动力学为$\boldsymbol{e}_{t+1} = (\boldsymbol{I} + \boldsymbol{K}_{bc})\boldsymbol{e}_t$。由于 $\|\boldsymbol{I}\| = 1$，我们无法通过限制 $\|\boldsymbol{K}_{bc}\|$ 来获得稳定条件 $\rho(\boldsymbol{I} + \boldsymbol{K}_{bc}) < 1$，这正是我们的策略形式所采用的方法。因此，无法保证系统的稳定性。

这一简单分析支持了我们的决定：在学习过程中忽略自动驾驶车辆的历史轨迹，以避免协变量偏移。我们的实验也将验证这一做法的优势，表明我们的方法能够显著降低驶离道路率。

IV. 方法

在图1中，我们概述了我们的上下文条件模仿学习方法。我们的策略网络由两个编码器组成，即基于 Transformer [29]的空间编码器和时间编码器。在训练期间，我们优化网络参数，以最小化其预测轨迹与真实轨迹之间的 L^1 距离。在评估期间，我们采用线性二次调节器（LQR） [30]控制器，根据策略网络预测的轨迹生成平滑的规划。

A. 策略输入

我们的策略网络将来自 H 个先前时间步的观测上下文作为输入，时间步间隔为 I。我们没有直接从原始传感器数据（例如摄像头图像或激光雷达点云）中学习端到端策略，而是利用中间输入表示来提升泛化能力和可解释性，与 [16] 中的方法类似。

道路网络由人行横道和车道组成，每个都表示为包含多种特征的向量折线序列，如 VectorNet [31]。人行横道向量特征包括其起点和终点的位置及序列顺序，而车道向量还包含交通信号灯状态和车道宽度等其他特征。

对于交通参与者，我们将其分为自车和其他智能体。如前所述，我们忽略自车中可能受到学习策略影响的特征，仅考虑其任务目标位置，而不考虑其历史轨迹。对于其他智能体，我们输入它们在多个过去时间步的类型、尺寸、边界框质心和朝向。

总体而言，我们的方法使用中间向量表示，并忽略自动驾驶车辆的历史轨迹，以提升模型性能。

B. 坐标系

在获取不包含明确自车状态、持续H 个时间步的上下文后，我们将每个上下文转换到其各自当前的坐标系中，以便在闭环评估期间重用特征。为避免泄露可能损害闭环性能的隐式自车信息，我们提出了一种自车扰动目标导向坐标系。该坐标系的原点锚定在自车位置加上零均值高斯扰动处，x轴指向其目标位置。扰动模糊了自车的位置信息，并确保局部可观测性。目标位置消除了自动驾驶车辆航向的影响，也可以替换为任何不受学习策略影响的点。

值得注意的是，我们在所有过去的 H 时间步上分别施加扰动，以模糊或隐藏学习策略中的隐式自车位置信息，而非扩充训练数据。这是因为在训练和评估过程中，几乎不可能出现经过扰动的数据。相比之下，ChauffeurNet [16] 会扰动自动驾驶车辆的当前位姿，将平滑轨迹拟合到扰动后的点以及原始起点和终点，并生成逼真的新训练样本。

C. 策略网络

策略网络旨在仅利用自动驾驶车辆的上下文来预测其未来轨迹。由于在没有自动驾驶车辆历史轨迹的情况下预测其未来轨迹更具挑战性，因此我们需要充分利用自动驾驶车辆与周围环境之间的时空交互。为此，我们使用空间编码器将每个时间步的观测嵌入为观测特征。随后，我们使用时间编码器对空间观测特征进行嵌入并生成预测。Transformer 是该网络的基础，因为它在自动驾驶领域已展现出卓越的性能 [18, 32]。

空间编码器： 为了捕捉同一地图折线中向量之间的空间关系，我们使用局部 Transformer 编码器作为向量间交互编码器。接下来，我们通过最大池化聚合属于同一折线的向量特征，以获得折线级特征。此外，我们利用两个多层感知器（MLP）来获取智能体特征和目标特征。为了对目标、智能体和地图元素之间的高阶交互进行建模，我们还采用了全局Transformer编码器，并将目标嵌入作为观测特征输出 o_t。

时间编码器： 为了捕捉时间信息和交互，我们采用带有因果自注意力掩码的 Transformer 编码器，对 H 历史观测特征进行嵌入。最后，Transformer 的每个隐藏状态都由一个全连接线性层解码，以预测自动驾驶车辆在其相对未来 T 个时间步的位姿。

D. 训练过程

我们采用行为克隆等离线监督学习方法，通过最小化预测轨迹与真实值轨迹之间的 L^1 距离来训练策略网络。这是因为，与常用的均方误差 [33]相比，L^1 指标与驾驶性能的相关性更高。为帮助网络收敛并提升泛化能力，我们还

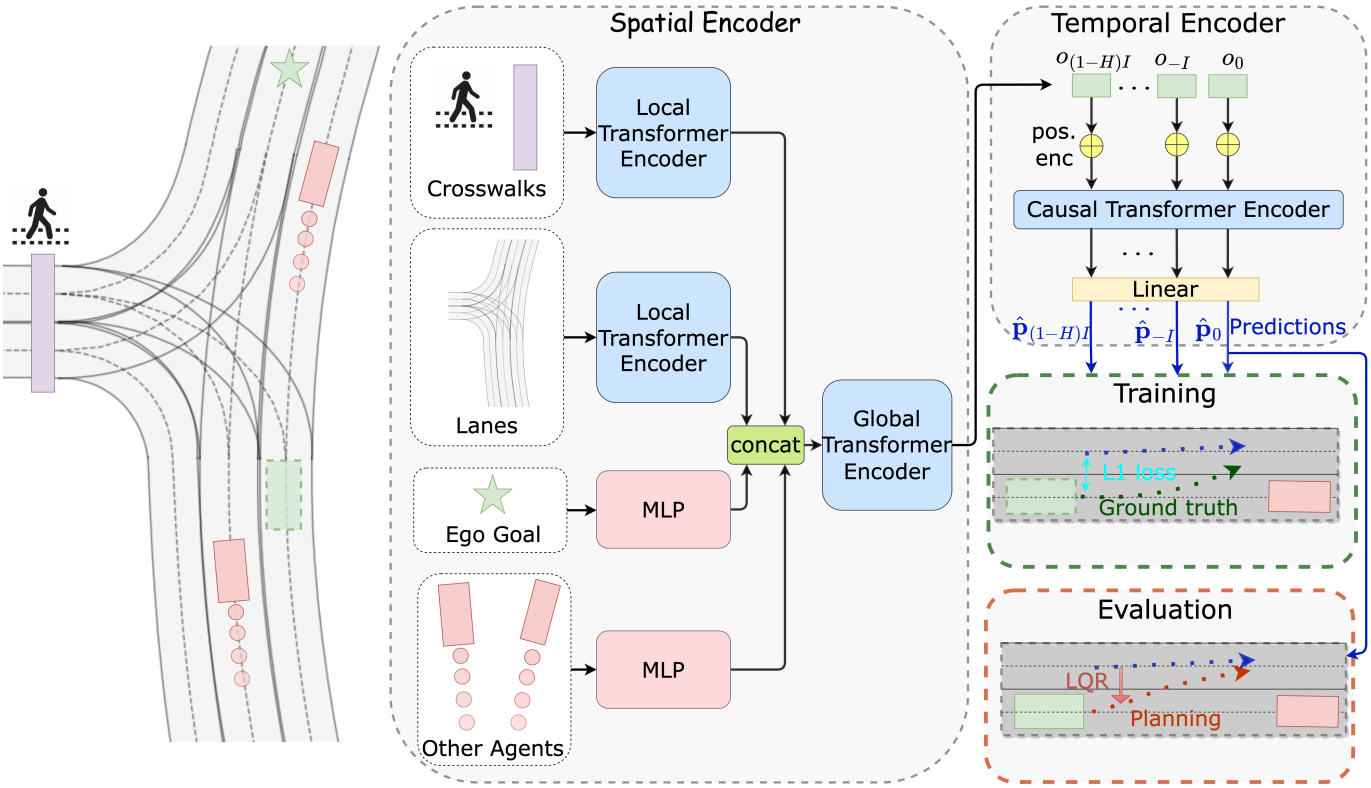

图1. 我们方法的概述。

引入一项辅助任务，以最小化因果Transformer中所有先前时间步的 L^1 误差，并根据公式（6）的启发，对网络参数 $\boldsymbol{\theta}$施加平方 L^2 范数正则化。最终损失为：

$$L = \sum_{t=1}^{T} \left(\|\boldsymbol{p}_t - \hat{\boldsymbol{p}}_0^t\|_1 + \mu \sum_{h=1}^{H-1} \|\boldsymbol{p}_{t-hI} - \hat{\boldsymbol{p}}_{hI}^t\|_1 \right) + \frac{\lambda}{2}\|\boldsymbol{\theta}\|^2, \tag{7}$$

其中， $\boldsymbol{p}_t$ 表示时间步t 的真实自车位姿，$\hat{\boldsymbol{p}}_{hI}^t$ 表示过去 hI 个时间步处对其相对未来 t 个时间步的位姿预测。 μ 是辅助任务超参数， λ 是正则化因子。

E. 评估流程

虽然不输入自车状态的预测更加稳定，但难以确保从当前状态出发的预测轨迹具有平滑性。为解决这一问题，先前的工作 [34, 23]通常会在策略网络中加入可微运动学层，以生成物理上可行的规划。然而，这种可微运动学层会引入自动驾驶车辆（SDV）的当前信息，而这不利于策略学习。因此，我们选择在评估流程中应用线性二次调节器（LQR），以获得平滑轨迹。

线性二次调节器是一种计算效率高的方法，可最小化线性动态系统的总累积二次代价。为简单起见，我们考虑一个有限时域、离散时间线性系统，其动力学描述如下：

$$\begin{bmatrix} \boldsymbol{p}_{t+1} \\ \dot{\boldsymbol{p}}_{t+1} \\ \ddot{\boldsymbol{p}}_{t+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{D} & \boldsymbol{D}^2 \\ 0 & \boldsymbol{I} & \boldsymbol{D} \\ 0 & 0 & \boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{p}_t \\ \dot{\boldsymbol{p}}_t \\ \ddot{\boldsymbol{p}}_t \end{bmatrix} + \begin{bmatrix} \boldsymbol{D}^3 \\ \boldsymbol{D}^2 \\ \boldsymbol{D} \end{bmatrix} \boldsymbol{u}_t, \tag{8}$$

其中， $\boldsymbol{D}$ 是一个对角矩阵，其对角元素为各时间步的间隔； $\dot{\boldsymbol{p}} = (\omega_t, \boldsymbol{v}_t)$ $\ddot{\boldsymbol{p}}_t = (\alpha_t, \boldsymbol{a}_t)$ $\boldsymbol{u}_t = (\zeta_t, \boldsymbol{j}_t)$ 分别表示角速度和位置速度、加速度、加加速度以及控制输入。该系统受二次代价函数约束：

$$J = \sum_{t=1}^{T} \|\boldsymbol{p}_t - \hat{\boldsymbol{p}}_0^t\|^2 + \eta_\omega \omega_t^2 + \eta_\alpha \alpha_t^2 + \eta_{\boldsymbol{a}} \|\boldsymbol{a}_t\|^2 + \eta_{\boldsymbol{j}} \|\boldsymbol{j}_t\|^2, \tag{9}$$

其中， $\boldsymbol{p}_t$ 表示规划位姿，而策略网络预测的位姿$\hat{\boldsymbol{p}}_0^t$ 被视为目标位姿。我们使用权重 η_ω、 η_α、 $\eta_{\boldsymbol{a}}$和 $\eta_{\boldsymbol{j}}$ 来平衡规划轨迹的安全性与平滑性。优化结束时，可生成一条包含位置和航向的平滑轨迹，供自动驾驶车辆跟随。

V. 实验

A. 数据集

为评估我们方法的性能，我们在两个大规模真实世界数据集上进行了实验：

Lyft Level 5 预测数据集 [19]：其中包含约 1,000 小时来自帕洛阿尔托的城市驾驶示范数据，这些数据被划分为时长近 25 秒、频率为 10 Hz 的独立场景。我们使用提供的

100 小时子集（16,265 个场景）训练 UrbanDriver [18]，并使用全部 16,220 个验证场景对其进行测试。

nuPlan 数据集 [20]：该数据集包含在四座城市（波士顿、匹兹堡、拉斯维加斯和新加坡）采集的 1,312 小时人类驾驶数据。由于不同城市的交通规则和模式存在显著差异，我们按照 Phan-Minh 等人 [35] 的做法，提取拉斯维加斯的驾驶数据。随后，按照 Lyft 数据集中的做法，以 10 Hz 的频率将数据分割成时长为 25 秒的独立场景，并筛除不含任务目标的场景。筛选后，我们获得了 63,181 个训练场景、4,774 个验证场景和 6,386 个测试场景。

有关数据准备的更多信息以及另一个**玩具数据集** 可参见附录。

B. 闭环评估

为评估我们方法的闭环性能，我们采用日志回放模拟器，与先前的工作 [18, 36] 一致。在日志回放模拟器的每个步骤中，自动驾驶车辆根据规划轨迹更新其位姿，而其他智能体则假定按照数据集中记录的轨迹行驶。对于这两个数据集，我们以 10 Hz 的频率对方法进行 25 秒的评估，采用以下指标：

碰撞率：如果自动驾驶车辆在某个场景的任意时间步与其他智能体发生碰撞，则该场景被视为碰撞场景。碰撞率通过碰撞场景数量除以场景总数计算得出。

驶离道路率：在 nuPlan 中，我们使用官方的驶离道路指标，该指标考虑自动驾驶车辆边界框的一个角与可行驶区域之间的距离。具体而言，如果该距离超过 0.3 m，则该场景被视为驶离道路。然而，Lyft 数据集中无法获取可行驶区域。因此，遵循 UrbanDriver [18],，如果场景中自动驾驶车辆相对于人类驾驶员真实值的横向偏差超过 2 m，则将该场景视为驶离道路。

不适度：为量化规划轨迹的舒适度和可行性，我们计算所有时间步中绝对加速度值超过 3 m/s^2 的比例。

L2：我们使用滚动轨迹与人类驾驶员真实值之间的平均 L2 位置误差，来量化我们的结果与人类驾驶之间的相似度。

C. 性能评估

在 Lyft 数据集上，我们将所提方法与三种最先进的方法进行比较，以展示所提框架的优势：

Raster-perturb：一种官方基线方法，通过行为克隆在 Lyft 数据集 [19], 上训练50 [37], 。该方法基于 ResNet，接收以自驾车辆为中心的场景鸟瞰图（BEV）表示，并规划具有位置和偏航位移的轨迹。为扩充数据，对当前自驾车辆位置施加扰动，然后生成一条新的、满足运动学约束的轨迹，使其到达原始终点，方法与 ChauffeurNet 类似。

BC-perturb：一种由 UrbanDriver 提供的 BC 方法，采用与 **Raster-perturb** 相同的轨迹扰动和输出但其地图和智能体输入以向量形式表示，并由 VectorNet 和 Transformer 处理。此外，自动驾驶车辆的历史信息还加入了随机失活。

UrbanDriver：一种离线策略梯度方法，通过利用可微数据驱动模拟器来模仿专家策略，其输入和模型结构与**BC-perturb**相同。

在 nuPlan 数据集上，由于缺乏其他已有研究，我们将所提方法与一种通过行为克隆训练的**官方**基线方法进行比较：

LaneGCN-perturb：一种基于向量的模型，利用一系列多层感知机对自驾车辆和智能体输入进行编码，并利用 LaneGCN [38]对矢量地图元素进行编码；随后通过融合网络和注意力层，捕捉车道与智能体之间及其内部的交互。为扩充训练数据，对自驾车轨迹施加扰动，并随机丢弃其他智能体。

除了先前工作提供的这些模型外，我们还在两个数据集上，采用行为克隆和离线强化学习中的代表性方法训练了我们的模型：

TD3+行为克隆 [39]：该方法将行为克隆项添加到双延迟深度确定性策略梯度（TD3） [40] 的策略更新中，以实现隐式策略约束。我们通过应用轨迹扰动数据增强构建离线强化学习数据集，并在奖励设计中考虑碰撞和舒适度。

Vector-Chauffeur：我们的网络采用与 ChauffeurNet 相同的数据增强方法进行训练，包括轨迹扰动和自车过去运动随机失活。我们还使用相同的坐标系表示数据，以自动驾驶车辆的位置作为原点，并将经均匀噪声扰动后的航向作为朝向。

如表 I 所示，我们提出的方法在两个数据集上均显著优于所有先前工作。由于场景更加复杂且包含更多道路智能体，nuPlan 数据集的碰撞率高于 Lyft 数据集。官方基线 **LaneGCN-扰动** 在 nuPlan 数据集上的表现较差，是因为其使用了未经微调的官方超参数，而其训练过程耗时较长。

D. 消融研究

我们在 Lyft 数据集上进行了以下消融实验，以揭示模型中不同组件的重要性，结果如表 II 所示：

网络输入：我们首先研究从网络输入中移除显式自车信息的重要性。我们考虑了两种重新引入显式自车输入的方法。一种是直接输入自动驾驶车辆的过去位置，然后以与处理目标位置相同的方式处理这些位置。另一种是在训练期间，像 ChauffeurNet 一样，对自车输入额外施加 50% 的随机失活。请注意，我们的方法与随机失活方法不同，因为随机失活仅在训练期间应用，而我们始终移除自车状态。我们可以观察到，在这两种显式输入自车信息的

表 I Lyft 和 nuPlan 数据集上闭环性能与基线方法的比较。

模型	参数数量	碰撞（%）	驶离道路(%)	不适度(%)	L2（米）
Raster-perturb*	23.6M	15.48	5.06	**4.00**	5.90
BC-perturb*	1.8M	9.38	6.77	39.10	4.77
UrbanDriver*	1.8M	13.28	7.27	39.41	5.74
TD3+行为克隆	2.8M	22.53±1.76	15.21±0.97	4.86±0.47	6.34±0.41
Vector-Chauffeur	1.5M	10.12±0.23	3.40±0.32	5.42±0.44	5.03±0.43
CCIL（我们的方法）	**1.5M**	**3.32**±0.15	**0.62**±0.13	4.33±0.22	**1.23** ± 0.08
LaneGCN-perturb	2.0M	60.63±2.34	34.25±1.65	17.26±1.80	21.21±1.81
TD3+行为克隆	2.8M	39.12±2.21	18.59±1.04	10.56±0.95	15.04±1.62
Vector-Chauffeur	1.5M	24.12±1.37	10.11±0.62	12.53±1.17	6.12±0.87
CCIL（我们的方法）	**1.5M**	**6.91**±0.11	**3.08**±0.11	**1.16**±0.05	**3.68**±0.04

*Raster-perturb 中不存在方差，BC-perturb 和 UrbanDriver 中不存在方差，因为我们在确定性模拟器中评估确定性预训练模型 .

表 II Lyft 数据集上的闭环性能消融实验

模型	扰动	Ego	碰撞(%)	驶离道路(%)	不适度(%)	L2（米）
带显式自车	✓	✓	20.29±0.88	19.18±2.98	**0.57**±0.15	5.50±0.55
采用自车随机失活	✓	✓	14.05±1.53	5.02±0.88	0.63±0.20	4.15±0.47
采用自车坐标		✓	11.31±1.44	9.79±1.34	0.95±0.05	3.87±0.11
std=0			7.08±0.35	2.86±0.25	0.89±0.10	3.46±0.34
std=1	✓		3.39±0.17	1.00±0.16	1.99±0.15	2.10±0.05
std=2（我们的方法）	✓		**3.32**±0.15	0.62±0.13	4.33±0.22	1.23±0.08
std=3	✓		3.42±0.12	**0.49**±0.10	7.35±0.26	<escape>**0.91**±0.04</escape>
不使用因果 Trans	✓		4.28±0.25	1.43±0.25	6.53±0.26	1.63±0.10
无 LQR	✓		3.81±0.14	2.07±0.12	89.05±0.35	1.02±0.02
不使用正则化	✓		4.07±0.16	1.05±0.14	4.96±0.30	1.23±0.09
不含辅助项	✓		4.56±0.29	1.02±0.06	3.23±0.31	1.92±0.07

方式中，由于协变量偏移问题，碰撞率、驶离道路率和 L2 距离都急剧下降。

坐标系：为分析坐标系中隐式自车信息的影响，我们首先考虑将 ChauffeurNet 中的自车中心坐标系替换为我们的自车扰动目标导向坐标系，并让朝向围绕航向均匀分布。我们发现，由于存在隐式自车信息，Chauffeur Net 坐标系会导致较差的闭环性能。此外，在我们的坐标系中，我们在自动驾驶车辆的当前位置上添加零均值高斯噪声，以获得原点。通过增大高斯噪声的标准差（std），可以减少隐式自车信息。我们观察到，随着 std 增大，不适度也随之增加，而驶离道路率和闭环 L2 则有所下降。这表明，尽管隐式自车信息可以提升瞬时规划精度，却可能降低长期闭环驾驶性能。我们推测，这是因为在闭环驾驶过程中，自回归模型可能会使规划误差产生时间相关性，导致误差持续偏向某一方向，进而引发驶离道路事件。然而，时间上不相关的噪声可能会导致车辆围绕专家轨迹轻微振荡，但仍能实现成功驾驶。离线瞬时规划精度与实际长期驾驶质量之间相关性较弱这一现象，Codevilla 等人也曾指出 [33]。

模型架构：为了证明时间信息的重要性，以及因果 Transformer在捕捉时间交互方面的有效性，我们将其替换为多层感知机。我们的研究结果表明，因果 Transformer通过整合空间和时间信息，取得了更好的性能。此外，我们对线性二次调节器模块进行了消融研究，以考察其对舒适度的影响。我们发现，移除线性二次调节器机制后，舒适度显著下降，碰撞率和驶离道路率也略有上升。不过，L2 指标略有改善。这凸显了线性二次调节器模块在提升轨迹平滑性方面的重要作用，同时也表明评估期间引入的自车信息仍可能损害闭环性能的 L2 指标表现。

损失项：为展示附加损失项的有效性，我们分别移除了辅助项损失和正则化损失项。我们发现，这两者都有助于提升性能，尽管范数正则化带来的提升有限，这可能是因为我们的策略网络参数量适中，即使没有该损失项，也具有良好的正则化效果。

E. 定性分析

我们在两个 Lyft 场景中对比并可视化了我们的方法与基线方法的闭环轨迹。如图2所示，我们的方法生成了可行轨迹

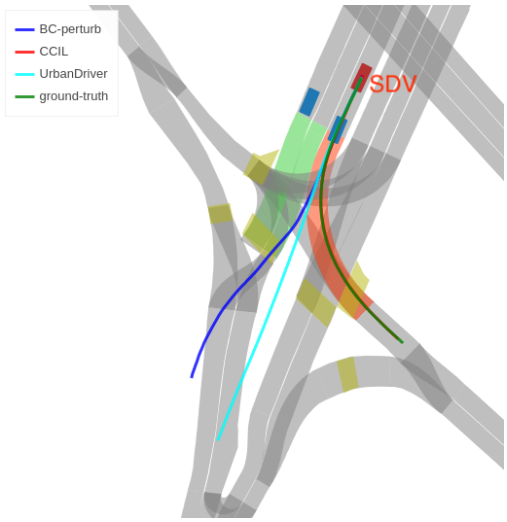

图2. 我们的方法与基线方法在道路驾驶任务中的定性比较。

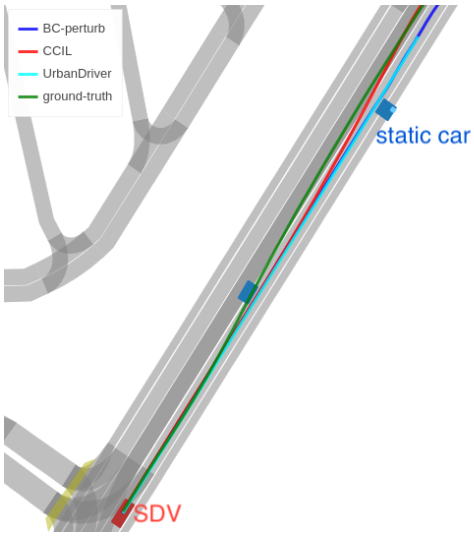

图3. 我们的方法与基线方法在避碰方面的定性比较。

与人类驾驶员的轨迹相似，而 BC-perturb 和 UrbanDriver 方法却使自动驾驶车辆驶离道路。从图3中可以看出，我们的方法能够通过变道避开路肩处的静止车辆，而基线方法则继续直行，最终导致碰撞。我们推测，当策略在转弯点或变道点输出错误动作（例如继续直行）时，就会发生驶离道路或碰撞事件。在这种情况下，策略在分布外历史信息下可能会产生更大的误差，导致驶离道路或与其他物体发生碰撞。BC-perturb 中的轨迹扰动以及 UrbanDriver 中的轨迹分布匹配都无法帮助自动驾驶车辆从错误状态中恢复，这可能是因为训练数据中，过去的直行轨迹与直行动作之间存在很强的相关性。

我们还对模型进行了定性分析

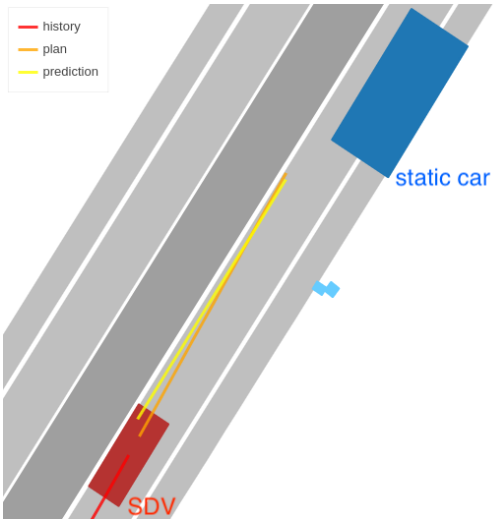

图4。我们模型预测与规划性能的定性分析。请参阅图3了解整个场景。

图3所示碰撞规避场景中的预测与规划。在图4中，我们展示了模型能够基于策略网络的预测规划出一条平滑轨迹。当前方路肩上有一辆静止车辆时，我们的策略网络推断自动驾驶车辆的未来位置应更靠近道路中心。尽管从自动驾驶车辆当前状态来看，该预测在动力学上不可行，但与其足够接近。这表明我们的策略能够根据上下文推断出合理的自车状态。

VI. 结论

我们提出了一种新的离线模仿学习方法，以缓解行为克隆中的分布偏移。我们的方法通过训练神经网络来预测自车的未来位姿，而不接收显式的自车信息。为消除隐式自车信息并确保局部可观测性，我们引入了一种新的自车扰动目标导向坐标系，用于表示观测输入。为应对这一具有挑战性的学习任务，我们设计了一种基于Transformer的网络，能够有效利用历史时空上下文数据。最后，我们使用两个真实世界的大规模数据集验证了该方法的有效性，并取得了最先进性能。

致谢

本研究得到阿里巴巴集团通过阿里巴巴创新研究计划和阿里巴巴研究实习生计划提供的支持。

参考文献

[1]Ekim Yurtsever、Jacob Lambert、Alexander Carballo 和 Kazuya Takeda。自动驾驶综述：常见实践与新兴技术。*IEEE Access*，8:58443–58469，2020年。

[2]Mikael Henaff、Alfredo Canziani 和 Yann LeCun。通过不确定性正则化进行模型预测策略学习，以实现在密集交通中驾驶。发表于*ICLR*，2019年。[3]Haoyang Fan、Fan Zhu、Changchun Liu、Liangliang Zhang、Li Zhuang、Dong Li、Weicheng Zhu、Jiangtao Hu、Hongye Li 和 Qi Kong。百度阿波罗 EM 运动规划器。*arXiv* 预印本 *arXiv:1807.08048*，2018年。[4]Mariusz Bojarski、Davide Del Testa、Daniel Dworakowski、Bernhard Firner、Beat Flepp、Prasoon Goyal、Lawrence D Jackel、Mathew Monfort、Urs Muller、Jiakai Zhang 等。端到端学习自动驾驶汽车。*arXiv* 预印本 *arXiv:1604.07316*，2016年。[5] Yunpeng Pan、Ching-An Cheng、Kamil Saigol、Keuntaek Lee、Xinyan Yan、Evangelos A Theodorou 和 Byron Boots。灵巧自动驾驶的模仿学习。《国际机器人研究杂志》，39(2-3)：286–302，2020年。[6]Dean A Pomerleau。ALVINN：基于神经网络的自主陆地车辆。*NeurIPS*，1，1988年。[7]Felipe Codevilla、Matthias M¨uller、Antonio L´opez、Vladlen Koltun 和 Alexey Dosovitskiy。通过条件模仿学习实现端到端驾驶。发表于*ICRA*，第4693–4700页，2018年。[8] St´ephane Ross、Geoffrey Gordon 和 Drew Bagnell。将模仿学习和结构化预测归约为无悔在线学习。发表于*AISTATS*，第627–635页，2011年。[9] Michael Laskey、Jonathan Lee、Roy Fox、Anca Dragan 和 Ken Goldberg。DART：通过噪声注入实现鲁棒模仿学习。发表于*CoRL*，第143–156页，2017年。[10] Jonathan Ho 和 Stefano Ermon。生成对抗模仿学习。*NeurIPS*，29，2016年。[11] Malayandi Palan、Shane Barratt、Alex McCauley、Dorsa Sadigh、Vikas Sindhwani 和 Stephen Boyd。使用卡尔曼约束拟合示范数据的线性控制策略。发表于动力学与控制学习，第374–383页，2020年。[12] Aaron Havens 和 Bin Hu。线性控制策略的模仿学习：通过线性矩阵不等式条件强制施加稳定性和鲁棒性约束。发表于*ACC*，第882–887页，2021年。[13] He Yin、Peter Seiler、Ming Jin 和 Murat Arcak。具有稳定性和安全性保证的模仿学习。《*IEEE*控制系统快报》，6：409–414，2021年。[14] Sebastian East。非线性系统稳定化策略的模仿学习。《欧洲控制杂志》，第100678页，2022年。[15] Felipe Codevilla、Eder Santana、Antonio M L´opez 和 Adrien Gaidon。探索行为克隆在自动驾驶中的局限性。发表于*CVPR*，第9329–9338页，2019年。[16]Mayank Bansal、Alex Krizhevsky 和 Abhijit Ogale。ChauffeurNet：通过模仿最优方案并合成最差方案来学习驾驶。发表于*RSS*，2019年。[17] Pim De Haan、Dinesh Jayaraman 和 Sergey Levine。模仿学习中的因果混淆。*NeurIPS*，32，2019年。[18] Oliver Scheel、Luca Bergamini、Maciej Wolczyk、Bła˙żej Osi´nski 和 Peter Ondruska。Urban Driver：利用策略梯度从真实世界示范中学习驾驶。在*CoRL*，第718–728页，2022年。[19] John Houston、Guido Zuidhof、Luca Bergamini、Yawei Ye、Long Chen、Ashesh Jain、Sammy Omari、Vladimir Iglovikov 和 Peter Ondruska。一千零一小时：自动驾驶运动预测数据集。在*CoRL*，2020年。[20] Holger Caesar、Juraj Kabzan、Kok Seang Tan、Whye Kit Fong、Eric Wolff、Alex Lang、Luke Fletcher、Oscar Beijbom 和 Sammy Omari。NuPlan：面向自动驾驶车辆的基于机器学习的闭环规划基准。在*CVPR ADP3*研讨会，2021年。[21]Jiakai Zhang 和 Kyunghyun Cho。端到端模拟驾驶的查询高效模仿学习。在*AAAI*，第2891–2897页，2017年。[22]Pin Wang、Dapeng Liu、Jiayu Chen、Hanhan Li 和 Ching-Yao Chan。通过增强对抗逆强化学习实现自动驾驶决策。在*ICRA*，第1036–1042页。IEEE，2021年。[23] Brandon Amos、Ivan Jimenez、Jacob Sacks、Byron Boots 和 J Zico Kolter。用于端到端规划与控制的可微模型预测控制。*NeurIPS*，31，2018年。[24] Diederik P Kingma 和 Max Welling。自编码变分贝叶斯。*arXivpreprintarXiv:1312.6114*，2013年。[25] Daniel Pfrommer、Thomas TCK Zhang、Stephen Tu 和 Nikolai Matni。TaSIL：泰勒级数模仿学习。*arXivpreprint arXiv:2205.14812*，2022年。[26]Stephen Tu、Alexander Robey、Tingnan Zhang 和 Nikolai Matni。稳定性约束模仿学习的样本复杂度。在动力学与控制学习会议，第180–191页，2022年。[27] Eduardo D Sontag。输入到状态稳定性：基本概念与结果。在非线性与最优控制理论，第163–220页。施普林格，2008年。[28] Zhong-Ping Jiang 和 Yuan Wang。离散时间非线性系统的输入到状态稳定性。*Automatica*，37(6)：857–869，2001年。[29]Ashish Vaswani、Noam Shazeer、Niki Parmar、Jakob Uszkoreit、Llion Jones、Aidan N Gomez、Łukasz Kaiser 和 Illia Polosukhin。Attention Is All You Need。在*NeurIPS*，第5998–6008页，2017年。[30] Karl Johan ˚Astr¨om 和 Richard M Murray。反馈系统：科学家与工程师导论。普林斯顿大学出版社，2021年。[31] Jiyang Gao、Chen Sun、Hang Zhao、Yi Shen、Dragomir Anguelov、Congcong Li 和 Cordelia Schmid。VectorNet：通过向量化表示对高清地图和智能体动力学进行编码。在*CVPR*，第11525–11533页，2020年。[32] Jiquan Ngiam、Vijay Vasudevan、Benjamin Caine、Zheng-dong Zhang、Hao-Tien Lewis Chiang、Jeffrey Ling、Re-becca Roelofs、Alex Bewley、Chenxi Liu、Ashish Venu-

gopal 等人。场景变换器：一种用于预测多个智能体未来轨迹的统一架构。发表于国际学习表征会议，2022年。

[33] Felipe Codevilla、Antonio M Lopez、Vladlen Koltun 和 Alexey Dosovitskiy。基于视觉的驾驶模型的离线评估。载于欧洲计算机视觉国际会议，第 236–251 页，2018年。[34] Matt Vitelli、Yan Chang、Yawei Ye、Ana Ferreira、Ma- ciej Wołczyk、Bła˙zej Osi´nski、Moritz Niendorf、Hugo Grimmett、Qiangui Huang、Ashesh Jain 等。SafetyNet：使用机器学习策略为真实世界自动驾驶车辆进行安全规划。载于*ICRA*，第 897–904 页，2022年。[35] Tung Phan-Minh、Forbes Howington、Ting-Sheng Chu、Sang Uk Lee、Momchil S Tomov、Nanxiang Li、Caglayan Dicle、Samuel Findler、Francisco Suarez-Ruiz、Robert Beaudoin 等。在现实世界中通过逆强化学习驾驶。*arXiv* 预印本 *arXiv:2206.03004*，2022年。[36] Zhiyu Huang、Haochen Liu、Jingda Wu 和 Chen Lv。用于自动驾驶的可学习代价函数与可微集成运动预测和规划。*arXiv* 预印本 *arXiv:2207.10422*，2022年。[37] Kaiming He、Xiangyu Zhang、Shaoqing Ren 和 Jian Sun。用于图像识别的深度残差学习。载于*CVPR*，第 770–778 页，2016年。[38] Ming Liang、Bin Yang、Rui Hu、Yun Chen、Renjie Liao、Song Feng 和 Raquel Urtasun。学习用于运动预测的车道图表示。载于欧洲计算机视觉国际会议，第 541–556 页，2020年。[39] Scott Fujimoto 和 Shixiang Shane Gu。离线强化学习的极简方法。神经信息处理系统进展，34:20132–20145，2021年。[40] Scott Fujimoto、Herke van Hoof 和 David Meger。应对演员-评论家方法中的函数逼近误差。载于国际机器学习会议，第 80 卷，第 1587–1596 页，2018年。

附录

A. 地图预处理

在 Lyft 中，地图元素只有两种类型：车道和人行横道。在 nuPlan 中，地图元素包括车道、车道连接段、交叉口、停止线、人行横道、人行道和停车场。我们将它们分为折线元素（包括车道和车道连接段）以及多边形元素（包括停止线、人行横道、交叉路口、人行道和停车场）。

对于每个折线元素，我们以 3 m 的间隔将其近似表示为向量序列。对于这两个数据集，我们都将折线地图预处理为图，以利用其中的拓扑信息。对于每个向量，如果存在与其最近的左侧、右侧和下一个向量，我们就将它们连接起来。最近的左侧或右侧向量，是其可达的左侧或右侧折线中距离最近的向量。向量之间的距离以其起点之间的欧几里得距离表示。构建向量图后，我们可以使用迪杰斯特拉算法计算任意两个向量之间的行进距离。对于这两个数据集，折线向量具有共同特征，包括起点和终点的坐标、到左右侧向量的距离、到任务目标的距离、交通灯状态（红色或绿色）以及序列顺序。不过，Lyft 中的车道向量具有车道宽度特征，因为这些向量近似表示计算得到的车道中线；而 nuPlan 中的车道或车道连接段向量具有车道左侧和宽度特征，因为这些向量近似表示标注基线。此外，nuPlan 中的折线向量还具有额外的限速和类型特征。

对于每个多边形元素，我们也用向量序列对其进行近似。在 Lyft 数据集中，人行横道向量直接通过连接原始连续标注坐标构造而成。但由于交叉路口等元素的标注点数量过多，我们使用固定的 20 个向量来近似每个多边形。对于这两个数据集，每个多边形向量的特征都包括起点和终点的坐标及其顺序序列，而 nuPlan 中的向量还包含额外的类型特征。

我们的神经网络输入由两种地图元素（折线和多边形）以及两种智能体元素（其他智能体和自车目标）组成。缺失的输入以零填充，并在计算注意力时屏蔽。原点是经过扰动的自驾车辆位置。

折线：选择 30 条拓扑距离最近的折线，其向量的起点距原点在 35 米以内。原点与一条折线之间的拓扑距离，是原点与该折线各向量之间距离的最小值。

多边形：选择边界距原点在 35 米以内的 20 个多边形。如果多边形超过 20 个，则根据其类型的重要性进行选择，依次为：停止线、人行横道、交叉口、人行道和停车场。

智能体：距离原点 50 m 以内的定向框中心点最近的 30 个智能体。Lyft 中的智能体特征包括其中心点坐标、偏航角、形状、类型，以及过去 2 个时间步和当前时间步的相对时间。nuPlan 中的智能体还具有速度特征，因为数据中提供了这些特征。

表 III 两个数据集的超参数

超参数	值
未来步数 T	15
所有 Transformer 随机失活	0.1
所有 Transformer 头数	8
所有 Transformer 隐藏维度	128
局部 Transformer 层数	3
全局 Transformer 层数	6
因果 Transformer 层数	3
因果 Transformer 历史长度 H	15
因果 Transformer 间隔 I	2
辅助权重 μ	0.3
正则化权重 λ	0.0001
LQR 角速度权重 η_ω	0.1
LQR 角加速度权重 η_α	0.1
LQR 位置加速度权重 $\eta_{\boldsymbol{a}}$	0.1
LQR 位置加加速度权重$\eta_{\boldsymbol{j}}$	0.01

目标：场景中自动驾驶车辆任务目标的 x，y 坐标。在 Lyft 中，任务目标未提供，因此我们将场景最后一个时间步自动驾驶车辆所在车道的终点视为任务目标。在 nuPlan 中，我们直接将场景最后一个时间步提供的任务目标作为场景任务目标。

B. 模型

对于这两个数据集，我们采用相同的模型架构，其超参数列于表 III。

C. 训练

我们的模型使用 Adam 优化器进行训练，学习率为 0.0005，线性预热 10000 步，β =(0.9,0.999)，批量大小为 128。训练 30 个训练轮次后停止，并选择验证集碰撞率最低的模型进行评估。除 Lyft 数据集中的预训练模型外，我们对所有模型分别独立训练 3 次，然后报告其性能的均值和标准差。

D. 评估

对于 Lyft 的基线方法，我们直接评估 Lyft 数据集和 UrbanDriver 提供的预训练模型。Raster-perturb 是 https://github.com/woven-planet/l5kit/blob/master/examples/planning/train.ipynb 上**在 train.zarr 上训练 2 个训练轮次的模型**。BC-perturb 和 Urban- Driver 分别是 https://github.com/woven-planet/l5kit/blob/master/examples/urban driver/train.ipynb 上的**带历史信息的开环**和 **UrbanDriver** 。

我们从第二个时间步开始评估所有 Lyft 模型，并使用位置信息作为输入，通过假设初始加速度为零的线性二次调节器来计算当前速度。

对于 nuPlan 基线方法，我们使用提供的超参数自行训练官方模型，并从第一个时间步开始进行评估，因为已提供速度和加速度信息。

E. 运行时间

我们在 Lyft 数据集上使用单块 Nvidia GeForce GTX 1080 GPU 和 Intel i7-8700@3.2GHz CPU 进行运行时实验。我们测量了各方法在一次评估的所有时间步上的平均运行时间和标准差。表 IV 中的运行时结果涵盖了各模型的所有组成部分，包括数据预处理、模型推理和控制。我们发现，与其他方法相比，我们的架构具有更高的数据处理效率和中等的模型推理效率。由于额外加入了 LQR 控制模块（此前的工作中并不存在该模块，因为它们只关注优化位置精度，而不考虑舒适度），我们的方法总执行时间更长。不过，我们的方法仍可在这套硬件上实时运行。

F. 玩具实验

我们设计了一项玩具实验，以直观展示我们的方法降低误差累积的能力。在实验中，我们使用来自一个极为理想且简化的场景的合成数据：如图 5 所示，一辆自动驾驶车辆以 1 m/s 的固定速度在环形道路网络中行驶，频率为 1 Hz。训练期间，环形道路的半径是一个变量，范围为 10 m 到 100 m；圆形车道由一系列间隔近乎 1 m 的固定车道点表示。

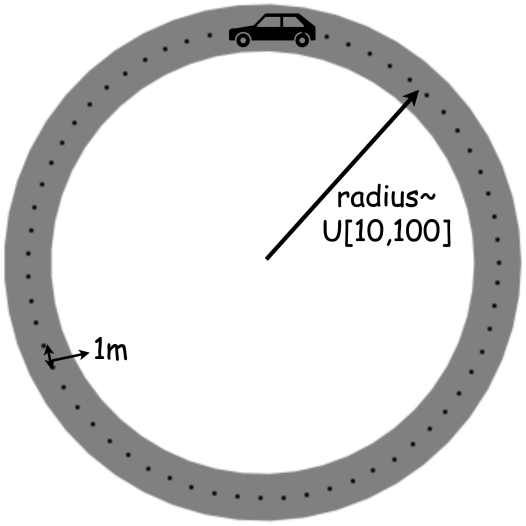

图 5. 环形道路

我们将 CCIL 方法与前文介绍的几种基线方法进行比较，包括行为克隆、BC-perturb 和 UrbanDriver。这些基线方法的输入由两部分组成：自车状态（过去 10 个时间步内的位置和朝向）以及自车坐标系中的上下文（最近的 10 个车道点）；而我们的方法仅将自车扰动中心定向坐标系中过去 10 个时间步的上下文作为输入。自车扰动中心定向坐标系是指将自动驾驶车辆的位置加上均值为零、标准差为一的高斯扰动作为原点，并将

表 IV 各方法中各组件的单帧平均运行时间

模型	数据处理时间（毫秒）	模型推理（毫秒）	控制（毫秒）	总计（毫秒）
Raster-perturb	6.03±0.61	**4.62**±0.16	-	**10.65**±0.64
BC-perturb	6.69±0.72	6.78±5.26	-	13.47±5.65
UrbanDriver	6.33±1.41	12.78±8.80	-	19.11±9.23
CCIL	**4.92**±0.63	6.74±0.34	11.09±0.27	22.75±1.07

指向环形道路中心的方向作为 x 轴方向。BC-perturb 方法通过轨迹扰动来增强数据。对于输出，BC 和 CCIL 方法会生成下一时间步的相对位置和偏航角，而 BC-perturb 方法会生成接下来 10 个时间步的结果。在 UrbanDriver 方法中，我们将策略展开 32 个时间步。

在每种方法中，我们都采用隐藏层大小为 128 的双层多层感知机作为策略网络。我们使用 Adam 优化器训练神经网络，学习率设为 0.0001，并采用随机初始权重训练 100 次。训练 10000 步后，我们从半径为 50 m 的圆上的一个随机起点展开策略，运行 100 个时间步。图 6 展示了每种方法的 100 条闭环轨迹。可以观察到，由于协变量偏移问题，基线方法中的一些轨迹偏离了道路，而我们的方法中的轨迹始终沿着路线行驶。

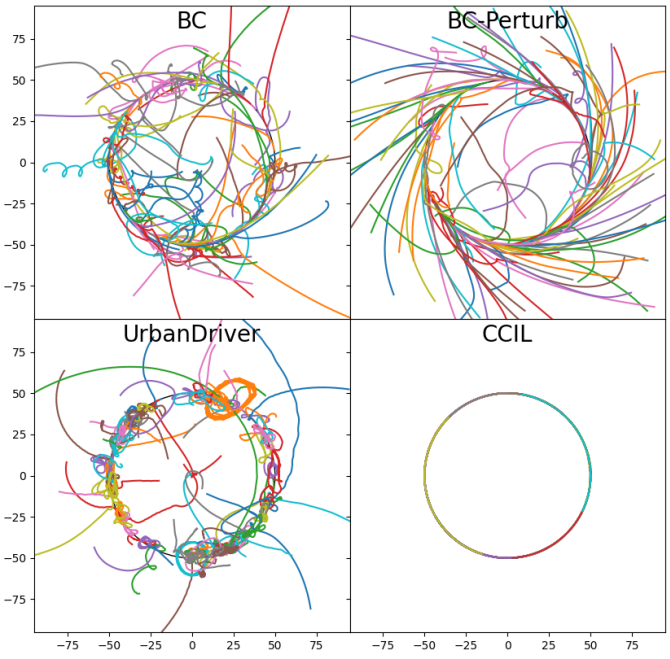

图 6. 在玩具数据集上训练的各个模型的闭环轨迹。