AROLA：面向缩比自主赛车的模块化分层架构

Fam Shihata[1], Mohammed Abdelazim[1],Ahmed Hussein[2]

arXiv:2602.02730v1 [cs.RO] 2 Feb 2026

摘要—自动赛车发展迅速，尤其是在缩比平台上，因此软件栈也必须随之演进。本文介绍了 AROLA，这是一种模块化分层软件架构，将零散且单体式的设计重组为可互换的层与组件，并通过标准化的 ROS 2 接口连接。自动驾驶流程被分解为传感、预处理、感知、定位与建图、规划、行为、控制和执行，从而无需依赖自定义消息定义即可快速替换模块并开展客观基准测试。为支持一致的性能评估，本文还介绍了 Race Monitor 框架。这一轻量级系统可实时记录单圈计时、轨迹质量和计算负载，并生成标准化的赛后分析报告。我们使用 RoboRacer平台，在仿真环境和硬件上对 AROLA 进行了验证，其中包括在 2025年RoboRacer IV25竞赛中的部署。AROLA 与 Race Monitor 共同表明，模块化、透明的接口和系统化评估能够加快开发进程，并提高缩比自主赛车的可复现性。

I. 引言

随着现代基于学习的感知与规划方法的兴起，自动驾驶汽车日益发展；随着这些系统规模扩大，其开发与维护也变得显著更具挑战性 [1]。该领域的传统研究依赖于全尺寸车辆，这带来了高昂成本、安全约束以及可复现性方面的限制。因此，缩比平台（如RoboRacer赛车，前身为F1TENTH）被采用为一种易于获取的替代方案，可实现快速实验、算法的受控验证，以及在真实而可预测的环境中进行系统测试 [2]。

尽管已有大量面向缩比自动驾驶车辆的架构方案，但大多数现有技术栈仍然是单体式、紧密耦合的，或针对特定硬件和竞赛规则量身定制。这种模块化不足限制了可复用性，使基准测试更加复杂，并且常常导致设计各自为政，妨碍不同研究团队之间进行比较。为解决这些问题，本文提出一种开放式分层框架，将自动驾驶流程拆分为通过标准接口连接的模块化组件。这种结构提升了系统的透明性，便于组件互换，并降低了入门门槛，同时保留了高级自动赛车所需的表达能力。

[1]柏林德国国际大学（GIU Berlin），柏林，德国fam@awadlouis.com, mohammed@azab.io[2]IAV GmbH，柏林，德国ahmed.hussein@ieee.org

本文余下部分将介绍 AROLA，这是一种模块化分层框架，旨在解决当前技术栈中的碎片化和可复用性不足问题。文中将介绍其设计原则、核心组件以及配套的 Race Monitor 评估套件。随后将展示在仿真和实体平台上的实验验证，并与相关工作进行比较，讨论能够体现 AROLA 在真实和虚拟赛车环境中有效性的性能指标。

本文的主要贡献如下：
1）面向缩比自主赛车的模块化开放分层软件架构（AROLA）；2）集成的实时监测与基准测试套件（Race Monitor）；以及3）在仿真环境与真实环境中的验证。

II. 相关工作

缩比自动驾驶已得到广泛研究，多项综述对当前技术与平台进行了全面评估。该领域的核心系统是RoboRacer平台 [2]，其开源硬件与基于ROS的软件栈 [3] 为研究提供了有效的试验平台。尽管如此，基于RoboRacer开展的工作范围广泛，导致了碎片化，因为许多团队构建的架构仅针对狭窄的、特定于竞赛的场景进行优化。正如Evans等人（2024年）所指出的，该平台上的研究已变得“广泛而零散”，使公平比较与系统化评估变得更加困难 [4]。Serban等人（2020年）也提出了类似的担忧，他们观察到大量自动驾驶车辆原型以零散方式构建为一次性解决方案，软件架构被视为事后补救，而非严谨设计过程的产物 [5]。

一些值得注意的设计为缩比自主赛车的模块化架构提供了借鉴。Kabzan等人（2020）提出了最早的完整流水线自动赛车系统之一，该系统按照感知–思考–行动范式组织，这影响了本文所采用的分解方式 [6]。同样，TUM Autonomous Motorsport系统展示了经过精心分层的架构在高速赛车中的有效性，其设计原则也促成了本文采用的结构 [7]。此外，Demeter等人（2025）提出了一种双层软件栈，将高性能计算与实时控制分离开来，说明架构划分如何简化自动驾驶汽车的开发与执行 [8]。

文献中的另一条研究方向考察了端到端架构。Wang 等人（2024）将自动驾驶技术栈分为传统模块化系统和端到端方法：前者由相互独立的感知、定位、预测、规划和控制子系统组成，后者则将原始传感器数据直接映射为控制命令 [9]。尽管已知存在流水线阶段之间的误差传播等问题，本文仍有意采用模块化架构，以保留可解释性并便于调试。端到端系统能够避免这些传播效应，但通常具有较低的透明性和可维护性 [10]。近期的示例包括 End2Race，该系统使用基于 GRU 的模型，将原始激光雷达测量数据直接映射为控制命令，并实现了较高的安全率和出色的超车性能 [11],；此外，还有 Bosello等人（2022）提出的深度强化学习控制器，该控制器采用基于深度 Q 网络的方法，展示了稳健的仿真到现实迁移能力 [12], [13]。

随着自动化系统日益复杂，系统化评估与基准测试已变得至关重要。仿真如今发挥着关键作用，能够在现实世界部署前进行受控测试。然而，现有框架种类繁多，使得合适评估工具的选择变得复杂。为满足这一需求，业界开发了多个基准测试套件。Grupp 的*evo* 软件包为里程计与同步定位与建图评估提供了标准化指标 [14]。Evans 的 f1tenth 基准测试套件支持对经典赛车算法与基于学习的赛车算法进行直接比较 [15]。TUM*laptime-simulation* 工具还提供了一种准稳态车辆动力学模型，用于评估单圈时间和能耗 [16]。总体而言，这些工具强调了可复现、模块化评估对于推动自动赛车研究的重要性。

这些不足表明，我们需要一个协调统一的模块化框架，以实现接口标准化、支持组件互换，并与成熟的机器人软件实践保持一致。下一节将介绍 AROLA。

III. 提出的方法

所提出的架构采用模块化的开放分层设计，旨在解决缩比自主赛车中单体式和面向特定任务的软件栈所存在的局限。AROLA（自主赛车开放分层架构）将自动驾驶流程划分为八个基本层。每一层均实现为可互换的模块，并通过标准化的 ROS 2 接口连接，以确保透明性、可扩展性和可复现性。

A. 基本组件

1) 传感： 它指系统通过车载传感器获取周围环境信息的能力。现代传感技术使缩比自动驾驶车辆能够生成大量原始数据，为所有下游计算奠定基础 [17]。

2) 预处理：原始传感器数据可能不完整、含有噪声或结构不一致。因此采用预处理来过滤、清洗并融合这些数据流，然后再将其传递给后续模块。集中实现这一功能可避免各节点之间的重复操作，并减少不必要的计算开销。

3) 感知： 感知为传感器采集的数据赋予语义含义。传感负责采集环境信息，而感知则对这些信息进行解读，例如识别赛道边界、障碍物或其他智能体。将这两个层分开有助于提升可诊断性，并支持模块化开发。

4) 定位与建图： 定位与建图被视为一个耦合过程，这与常见的同步定位与建图（SLAM）方法一致 [18]。尽管它们可以看作感知的专门形式，但由于其在自主导航中的核心作用，以及在现有软件栈中常作为独立模块出现，因此将其列为一个独立层。

5) 规划： 规划层基于环境感知和地图信息生成可行轨迹。多个规划器可在不同的时间范围或粒度层级上运行，从而兼顾全局优化与局部避障。

6) 行为： 行为模块负责决策逻辑，通常通过有限状态机、行为树或学习策略实现。将行为与规划解耦，有助于明确车辆应该做什么，以及运动规划将如何实现这一目标。

7) 控制： 控制层执行生成的轨迹，同时补偿由执行器缺陷和环境不确定性引起的偏差。该模块生成低级命令，并确保车辆遵循其动力学特性和执行器限制。

8) 执行： 执行层代表控制器命令的物理执行。转向和电机信号直接发送至硬件，从而闭合感知—思考—行动闭环。

B. 数据流

AROLA 被设计为一种开放式分层架构，支持传统的顺序数据流，同时允许在需要时进行可选的跨层交互。建议谨慎使用此类交互，以保持模块化。图1展示了预期的自上而下的数据流。

标准 ROS 2 接口统一了各层之间的交互。传感器输出、感知结果、规划消息和控制命令均使用标准消息类型进行交换，例如：

- $/scan$发布 sensor_msgs/LaserScan,
- $/odom$发布 nav_msgs/Odometry,
- $/map$ 提供 nav_msgs/OccupancyGrid 或 GetMap 服务,
- $/drive$接受以 ackermann_msgs/AckermannDriveStamped 表示的命令。

ROS2 tf消息在各节点之间提供一致的坐标系对齐。命名约定遵循标准 ROS 2 实践，使用小写的描述性标识符。多智能体运行通过命名空间予以支持（例如，

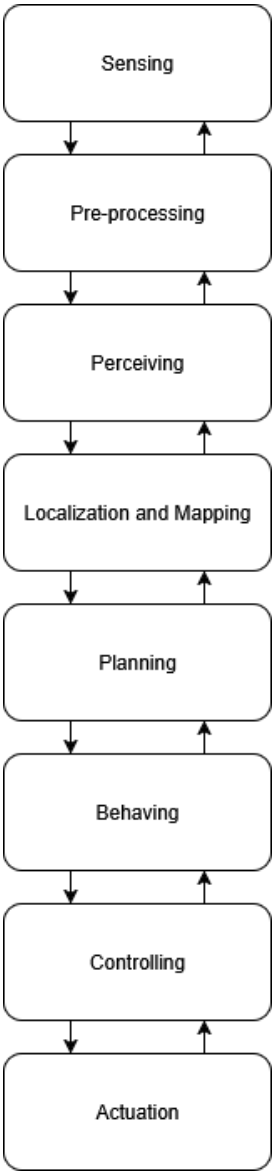

图 1. 完整 AROLA 架构概览

/ego_racecar、/opp_racecar）。自定义消息类型的使用被有意尽量减少，以增强可移植性以及与现有工具的兼容性。

C. 系统集成与部署

AROLA 兼容多种硬件配置。该架构使用 ROS 2 Humble 部署于 RoboRacer平台，集成了 Hokuyo 激光雷达、NVIDIA Jetson 计算单元以及基于 VESC 的电机控制器等组件。其硬件无关设计使团队只需极少修改，即可将架构适配到不同的配置中。

该框架还在多种仿真环境中通过了验证，包括 RoboRacer 模拟器和基于 Forza 的赛车模拟器 [19], [20]。这些部署展示了跨平台的一致性和易于集成的特点。图 2 展示了一个高复杂度示例配置。

D. Race Monitor

为支持对自动赛车性能进行系统化评估，Race Monitor 被集成为一个开放的、基于 ROS 2 的监控框架。它提供单圈计时、轨迹质量和计算负载的实时日志记录，以及使用 *evo* [14] 生成的赛后分析。

配置通过 YAML 文件指定，其中定义了启用的模块、参考轨迹和日志记录参数。Race Monitor 将遥测数据发布到标准化的 ROS 2 话题（表 I），使包括控制器在内的其他模块能够在不引入耦合的情况下进行订阅。参考轨迹可采用 CSV、TUM 或 KITTI 格式提供，起终点区域则使用 RViz 工具交互式定义。

表 I Race Monitor 发布的 ROS 2 话题

话题	Type
/race_monitor/lap_count	std_msgs/Int32
/race_monitor/lap_time	std_msgs/Float32
/race_monitor/race_进行中	std_msgs/Bool
/race_monitor/race_status	std_msgs/String
/race_监控/总_距离	std_msgs/Float32
/race_monitor/current_trajectory	nav_msgs/Path
/race_monitor/trajectory_指标	std_msgs/String

赛后输出包括单圈指标、轨迹误差可视化，以及针对 PID、模型预测控制或基于强化学习的变体等控制器的性能摘要。该框架同时支持实时自适应与事后评估，从而能够在相同条件下对控制策略进行无偏比较。

IV. 实验工作

在筹备 RoboRacer 2025 竞赛期间，使用 AROLA 部署了一套精简的模块化软件栈。在基于同步定位与建图的地图生成之后，以 CSV 格式生成了参考轨迹。在计时赛阶段之前，有 20 分钟的标定窗口可用于赛道测试。对三种控制器进行了评估：（1）间距跟随器，（2）纯跟踪，以及（3）模型预测控制。使用 Race Monitor 评估其性能，并以单圈时间、跟踪精度和计算统计数据作为评价指标。三种控制器的定量比较汇总于表 II。

间距跟随器表现出很强的反应能力和可靠的避障性能，但产生的单圈时间最慢。MPC 和纯跟踪表现出相近的较低位姿跟踪误差（绝对位姿误差和相对位姿误差）；MPC 的加加速度更低，但计算负载和控制延迟显著更高。纯跟踪产生了最快的单圈时间，因此被选用于后续使用。经过大约 5 圈后，观察到单圈时间一致性下降。Race Monitor 的误差映射轨迹可视化使得识别并修正

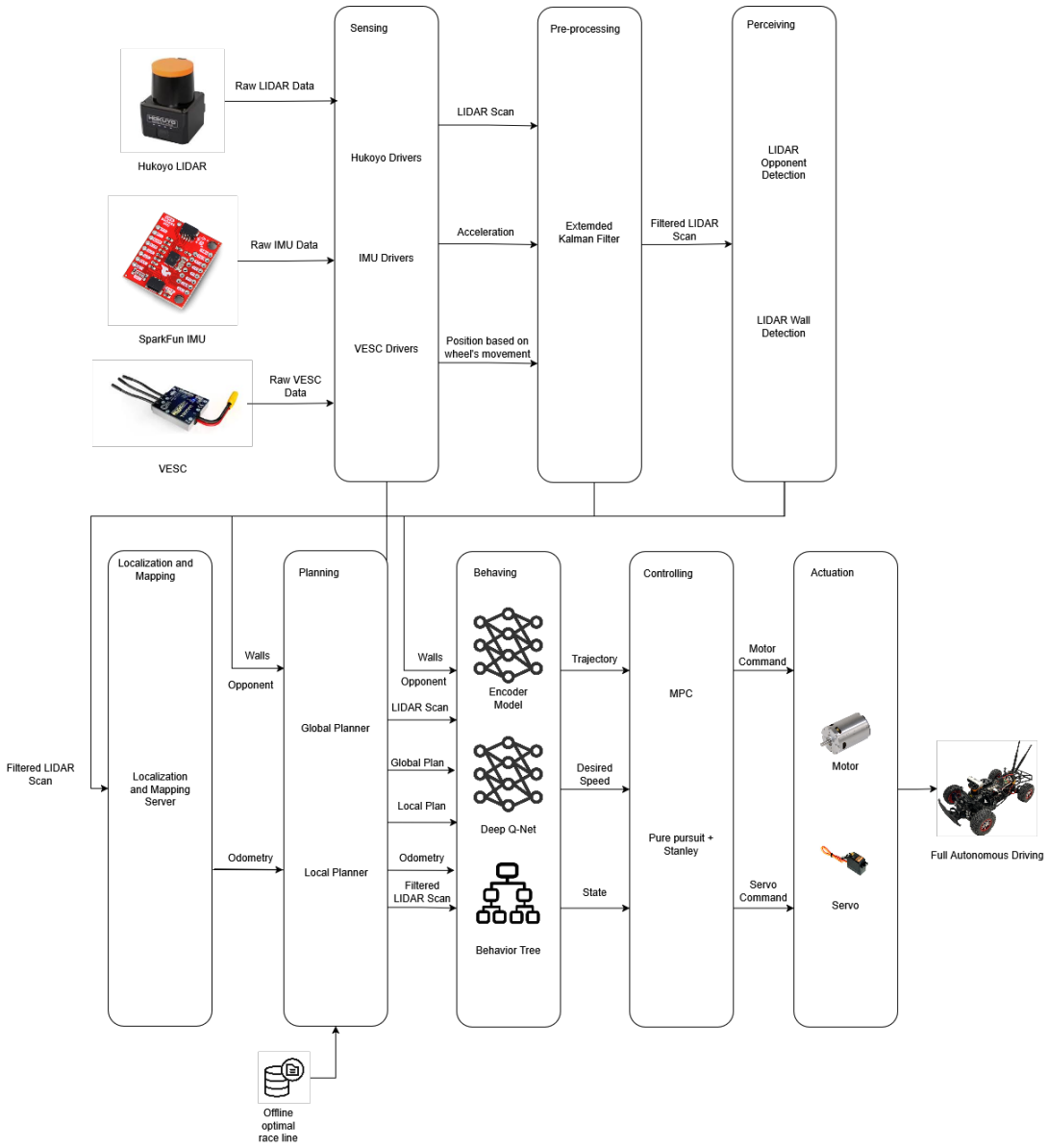

图 2. 使用 AROLA 的系统配置示例

表 II 控制器比较（初步）。

指标	间距跟随器	MPC	纯跟踪
最佳单圈时间（秒）	12.60	10.24	**10.10**
平均单圈时间（秒）	12.85	10.40	**10.35**
一致性评分	–	**0.98**	0.92
平均速度（米/秒）	4.30	4.99	4.95
控制延迟（毫秒）	**12**	42	18
CPU负载（%）	22	**55**	28
RPE 均值	–	4.2	4.3
APE 均值	–	**0.19**	0.22

次优参考路径段成为可能，随后进行了参数调优。相同的最小 AROLA 配置通过官方 RoboRacer 最小配置在仿真中进行了验证。

软件示例。[1]

A. RoboRacer IV25 比赛结果

最终配置获得了第三名，单圈时间为 $10.1\ s$，与第二名相差 $0.02\ s$，与第一名相差 $1.24\ s$。

B. 使用 *Race Monitor* 评估线性二次型调节器控制器

比赛结束后，使用了一张外部地图（“柏林地图”）来评估 LQR 控制器，并扩展 Race Monitor 的功能。在一次评估运行期间，Race Monitor 记录了圈速统计、跟踪性能和控制器行为。LQR 控制器顺利完成了七圈，未出现故障。平均单圈时间为 $19.75\ s$，最佳单圈时间为 $19.53\ s$。

[1]https://github.com/GIU-F1Tenth/minimal sofware example

单圈时间一致性较高，变异系数为 0.15，一致性评分为 0.99。平均速度达到 3.40 m/s，峰值为 3.79 m/s。表 III 总结了单圈时间和跟踪指标。

表 III LQR 控制器（实验 011）的比赛监控汇总结果。

指标	值
总圈数	7
最佳单圈时间（秒）	19.53
平均单圈时间（秒）	19.75
最差单圈时间（秒）	19.94
单圈时间标准差（秒）	0.15
一致性评分	0.99
平均速度（米/秒）	3.40
最高速度（米/秒）	3.79
APE 均值	4.044
RPE 均值	4.42
控制器	LQR
实验编号	实验 011

轨迹评估显示相对位姿误差（RPE）较低，为4.42。总体绝对位姿误差（APE）较高，原因在于参考轨迹原点处存在尺度失配，而非控制器精度不足。局部跟踪精度在所有圈次中保持稳定。图 3 展示了参数优化过程中使用的误差映射轨迹。

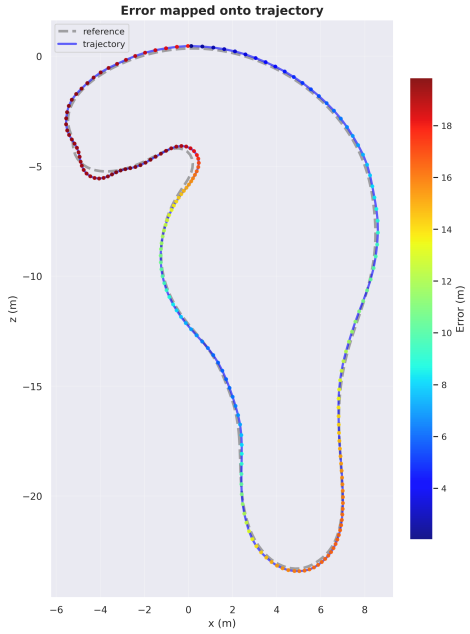

图 3. 误差映射轨迹，显示沿线性二次型调节器控制器路径的偏差。

七个已记录圈次的每圈计时结果见表 IV，圈速趋势见图 4，各轴平移误差见图 5，航向（偏航）误差见图 6。

表 IV LQR 控制器（实验 011）的各圈时间。

Lap	时间（秒）
1	19.67
2	19.73
3	19.94
4	19.84
5	**19.53**
6	19.58
7	19.94

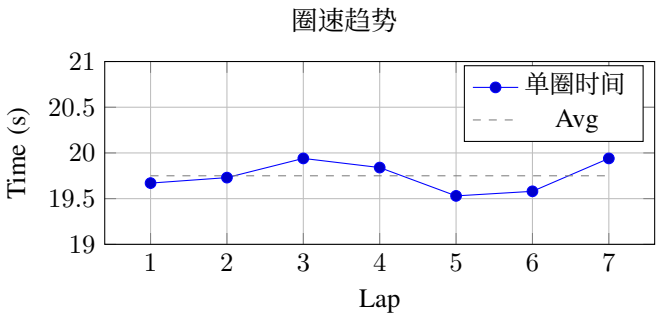

图 4. LQR 控制器的圈速趋势。

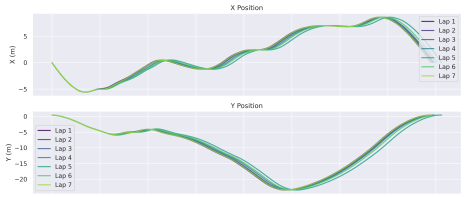

图 5. LQR 控制器（实验 011）相对于参考轨迹的各轴平移误差（XY）。Z 保持不变。

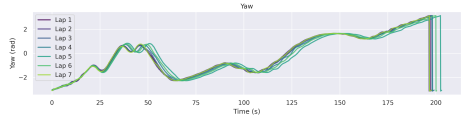

图 6. LQR 控制器（实验 011）的偏航误差分布与参考轨迹的比较。横滚和俯仰保持不变。

V. 总结与结论

AROLA 被提出作为一种模块化软件架构，旨在降低开发工作量，并改善组件在各类缩比自动驾驶车辆项目中的复用方式。它基于 RoboRacer平台构建，将驾驶软件栈划分为清晰的层次——传感、预处理、感知、定位或建图、规划、行为、控制与执行。每一层都依赖标准 ROS 2 接口，从而使系统易于扩展，并能够在不重构整个软件栈的情况下替换组件。

Race Monitor 是与 AROLA 同步开发的，用于跟踪仿真和真实环境运行期间的表现。它会实时记录系统行为，并提供一致的方法来比较和调整控制器。该工具在整个测试过程中都得到了使用，也为最终取得

2025年 RoboRacer IV25 赛事第三名的方案做出了贡献。实验表明，AROLA 的结构让评估不同控制器、有效调整参数并可靠地重复实验变得更加容易。

安装和部署流程力求简便，着重于简化配置并兼容不同的传感器和计算配置。总体而言，AROLA 为自动赛车研究提供了实用基础，有助于快速迭代、公平评估性能，并可靠地扩展规模。

VI. 局限性

尽管所提出的架构展现出很强的实用性，并成功部署于仿真和硬件赛车场景中，但其评估范围仍然有限。实验结果主要聚焦于少数几种控制器，尚未探索该架构旨在支持的更广泛的感知、定位和规划模块。与大多数模块化自动驾驶技术栈一样，该系统仍容易受到流水线阶段间误差传播的影响，感知或定位中的误差可能导致下游规划和控制性能下降。尽管该架构旨在实现硬件无关性和可扩展性，其实证验证仍局限于特定平台和传感器配置。此外，赛车性能指标侧重于单圈时间和轨迹误差，却忽略了其他重要的赛车考量，例如安全裕度、多智能体交互以及在干扰下的稳健性。

VII. 未来工作

未来的工作将致力于改进轨迹、地图和坐标系对齐工具，以减少评估伪影并提升可复现性。通过在多种硬件平台、传感器、模拟器和赛道上扩展验证，将进一步增强关于可移植性和通用性的论据。此外，未来的工作还将着眼于构建看门狗系统，以检测与架构不一致的情况并协助预防故障。此外，Race Monitor 将扩展为基于网络的用户界面，提供系统状态的全局视图，并支持更丰富的评估，包括新增指标，用于对感知、定位和规划模块以及控制性能进行基准测试。最后，在保留标准化接口的同时，将基于学习或端到端组件集成到模块化框架中，是一条有望兼顾可解释性、灵活性和性能的方向，可用于自动赛车系统。因此，该架构将扩展为支持部分端到端组件。

参考文献

[1] J. Li、R. Zong、Y. Wang 和 W. Deng，《自动驾驶测试中城市交叉路口场景的复杂度评估：方法与验证》，《应用科学》，第 14 卷，第 22 期，文章编号 10451，2024年。[2] I. Charles、H. Maghsoumi 和 Y. Fallah，《推进自动赛车：RoboRacer（F1TENTH）平台综合综述》，*arXiv* 预印本 *arXiv:2506.15899*，2025年。

[3] S. Macenski、T. Foote、B. Gerkey、C. Lalancette 和 W. Woodall，《机器人操作系统 2：设计、架构及其实际应用》，《科学机器人学》，第 7 卷，第 66 期，文章编号 eabm6074，2022年。[4] B. D. Evans、R. Trumpp、M. Caccamo、F. Jahncke、J. Betz、H. W. Jordaan 和 H. A. Engelbrecht，《统一 F1TENTH自动赛车：综述、方法与基准》，arXiv 预印本 arXiv:2402 18558，2024年。.[5] A. Serban、E. Poll 和 J. Visser，《面向全自动驾驶汽车的标准驱动软件架构》，《汽车软件工程杂志》，第 1 卷，第 1 期，第 20–33 页，2020年。[6] J. Kabzan、M. I. Valls、V. J. Reijgwart、H. F. Hendrikx、C. Ehmke、M. Prajapat、A. B¨uhler、N. Gosala、M. Gupta、R. Sivanesan 等，《AMZ Driverless：完整的自动赛车系统》，《野外机器人学杂志》，第 37 卷，第 7 期，第 1267–1294 页，2020年。[7] J. Betz、T. Betz、F. Fent、M. Geisslinger、A. Heilmeier、L. Hermansdorfer、T. Herrmann、S. Huch、P. Karle、M. Lienkamp 等，《TUM Autonomous Motorsport：印第安纳波利斯自动驾驶挑战赛的自动赛车软件》，《野外机器人学杂志》，第 40 卷，第 4 期，第 783–809 页，2023年。[8] Z. Demeter、L. Pusk´as、B. Kov´acs、 ´A. Matkovics、M. N´adas、B. Tuba、Z. Farkas、 ´A. Bog´ar-N´emeth 和 G. B´ari，《FRED-003C自动驾驶软件栈：迈向全尺寸自动赛车的开发历程》，arXiv 预印本 arXiv:2504.18439，2025年。[9] X. Wang、M. A. Maleki、M. W. Azhar 和 P. Trancoso，《迈向未来：自动驾驶软件与硬件系统综述》，arXiv 预印本 arXiv:2411.10291，2024年。[10] P. Singh Chib 和 P. Singh，《基于深度学习的端到端自动驾驶最新进展：综述》，arXiv 电子印本，第 arXiv–2307 页，2023年。[11] Z. Qiao、H. Li、Z. Cao 和 H. X. Liu，《End2Race：用于实时F1TENTH赛车的高效端到端模仿学习》，arXiv 预印本 arXiv:2509.16894，2025年。[12] M. Bosello、R. Tse 和 G. Pau，《奥地利训练，蒙特卡洛比赛：用于跨赛道F1TENTH激光雷达赛车的广义强化学习》，载于 2022年 IEEE 第 19 届年度消费者通信与网络会议（CCNC）。IEEE，2022年，第 290–298 页。[13] V. Mnih、K. Kavukcuoglu、D. Silver、A. A. Rusu、J. Veness、M. G. Bellemare、A. Graves、A. Riedmiller、A. K. Fidjeland、G. Ostrovski 等，《通过深度强化学习实现人类水平的控制》，《自然》，第 518 卷，第 7540 期，第 529–533 页，2015年。[14] M. Grupp，《evo：用于评估里程计和同步定位与建图的 Python 软件包》，2017年，GitHub代码库。访问日期：2017-01-01。 [在线]。网址：https://github.com/MichaelGrupp/evo [15] B. D. Evans，《F1TENTH基准测试：常见F1TENTH自动赛车算法的实现及基准测试结果》，https://github.com/BDEvan5/f1tenth benchmarks，2023年，GitHub代码库。[16] 慕尼黑工业大学 Fastwheel Motorsports，《laptime-simulation：准稳态单圈时间仿真》，https://github.com/TUMFTM/ laptime-simulation，2020年，GitHub代码库。[17] H. A. Ignatious、M. Khan 等，《自动驾驶汽车传感器概述》，《计算机科学程序汇刊》，第 198 卷，第 736–741 页，2022年。[18] U. Frese，《关于同步定位与建图的讨论》，《自主机器人》，第 20 卷，第 1 期，第 25–42 页，2006年。[19] M. O'Kelly、H. Zheng、D. Karthik 和 R. Mangharam，《F1TENTH：用于连续控制和强化学习的开源评估环境》，载于 NeurIPS 2019竞赛与演示赛道。PMLR，2020年，第 77–89 页。[20] N. Baumann、E. Ghignone、J. K¨uhne、N. Bastuck、J. Becker、N. Imholz、T. Kr¨anzlin、T. Y. Lim、M. L¨otscher、L. Schwarzenbach 等，《ForzaETH Race Stack——基于完全商用现成硬件的缩比自动双车对抗赛车》，《野外机器人学杂志》，2024年。