通向可靠安全的自主无人系统

万梓燊

引言和绪论

近年来，随着人工智能和自动化技术的蓬勃发展，自主无人系统（例如自动驾驶、无人机、机器人等）逐渐应用于各种生活生产场景，这种持续部署也对系统可靠性设计提出了新的要求和挑战。自主无人系统是一个集计算和物理信息的复杂系统，传感器和计算栈（计算硬件和软件）有存在潜在错误，且这些错误随着软件复杂度提升和半导体晶体管密度增加而不断出现。因此系统容错能力（例如环境、传感器、软件错误、硬件错误、对抗性攻击）对于自主无人系统的安全性至关重要。近期研究人员陆续提出多种评估标准、整体故障分析框架及轻量级故障缓解技术。本文探究了自主无人系统跨层计算栈的错误来源，讨论了错误对不同规模自主无人系统的影响及缓解方法，并展望了通向下一代可靠安全自主无人系统的挑战与机遇。

自主无人系统技术

闭环跨层自主无人系统计算栈。自主无人系统的计算通常以闭环形式进行。如图 1 所示，为实现智能化，自主无人系统与物理环境不断交互，此闭环循环过程包含了环境感知（输入数据）、规划决策（计算）和控制执行（输出动作），并跨越了算法、系统和硬件计算栈。

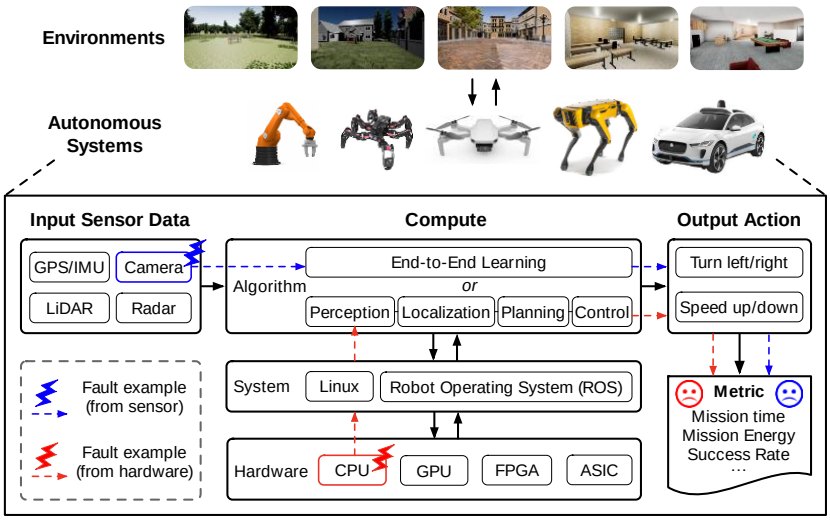

图 1. 闭环跨层自主无人系统计算栈实例（图源[1]）

算法层。一个典型的自主无人系统主要包含感知模块、定位模块和决策模块。

感知模块用于感知动态环境并基于传感器（例如相机、IMU、GPS、激光雷达）构建对环境可靠且详细的表征。感知模块通常包含特征提取、立体视觉、目标检测、场景理解等任务。特征提取通过对图像特征点操作以提高图片处理鲁棒性和计算效率，立体视觉通

过视差计算得到场景的 3D 结构信息。近年来深度学习的快速发展让感知系统具备了更强大计算和推理能力。

定位模块是系统确定自身的位置和方向。SLAM 是一种常用的建图定位算法，以及最近提出的基于滤波的 MSCKF VIO 和 OpenVINS。

决策模块主要包含路径规划和控制。其中规划的目标是找到起始位置到目标位置的最佳无碰撞轨迹，并随环境变化而实时调整。控制器会连续跟踪实际姿态与预定义轨迹姿态的差异以实现鲁棒的自主无人系统运动。基于采样的方法（例如 PRM, RRT）广泛应用于路径规划。

端到端学习。随着人工智能的发展，端到端学习模式能够通过强化学习或监督学习的方法，使自主无人系统直接从感知信息中学习并做出运动，不需要单独的建图和路径规划阶段。端到端学习简化了系统设计，但同时也有提高可解释性和弥合模拟-现实性能差距等挑战。

系统层。系统层主要包含实时计算操作系统和机器人系统（ROS）。ROS 是专门为机器人应用提供通信和资源分配的系统，是一种分布式处理框架，并将计算封装到数据包和堆栈中。实时操作系统负责将工作负载映射到计算单元并在运行时实时调度任务。

硬件层。自主无人系统搭载不同计算硬件处理负载运算，常见的计算单元包括通用处理器（CPU, GPU）、可编程逻辑器件（FPGA）和专用集成电路（ASIC）。其中 CPU 和 GPU 具有较高的可编程性和灵活性，能处理复杂场景运算，FPGA 和 ASIC 专注于特定应用场景下的能效提升 [2]。随着系统运算复杂度不断提升，片上系统（SoC）和异构处理器架构近年来得到不断发展。对于边缘端系统，低功耗电路设计及架构技术也受到广泛关注 [3]。

算法、系统和硬件计算栈以闭环形式工作，以实现自主无人系统的智能化。

自主无人系统的潜在错误来源

与任何计算系统一样，不同错误源也会对自主无人系统的安全部署和运行产生影响，且无人系统的安全性和可靠性尤为重要。探究错误起源对于分析其在系统中如何传递并影响系统鲁棒性尤为重要。大体有以下四类：

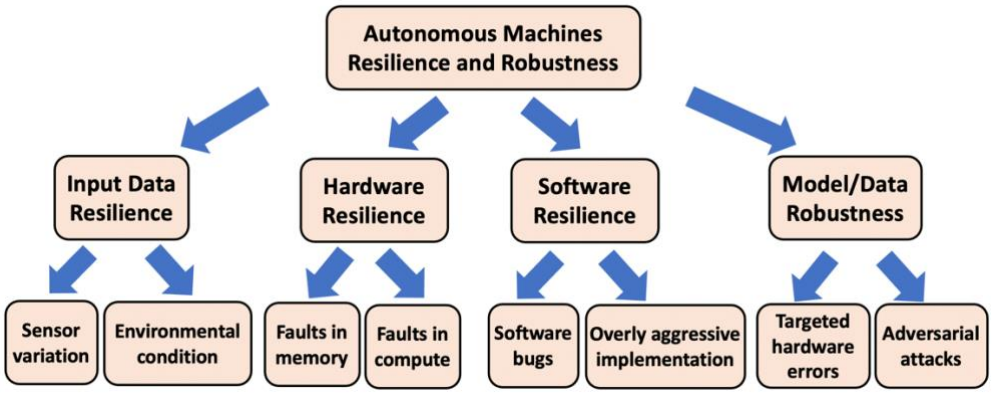

图 2. 自主无人系统错误来源实例（图源 MLCommons Resilience Research Working Group）

输入数据错误：输入数据错误可能来源于环境、传感器和其他智能体等状态。环境的亮度、对比度或者遮挡会给感知图像带来扰动从而影响感知精度。传感器的自身噪声可能改变数据数据并导致自主无人系统对环境的理解出现偏差。交互系统（例如多机系统、车路协同系统）中其他智能体的错误传递信息也可能致使本机出现决策失误。

硬件错误：硬件错误包括由电磁辐射引起的软错误（soft-error），由低电压引起的存储单元比特翻转，由超频或电负载电压下降引起的电路时序错误，由于制造缺陷或者老化引起的固定型故障（stuck-at-fault）。例如最近 IBM 的一项研究[7]表明，降低芯片工作电压会导致芯片上静态随机存取存储器（SRAM）发生位翻转，且会在电压缩放情况下始终存在。这些硬件错误会影响计算和存储单元，并随着技术节点密度和数据流位宽的不断增加及电压的不断降低而加剧。

软件错误：软件错误主要来源于计算机中的程序错误或者由近似或低精度计算导致的结果失准。网络通信路径中的时序错误（例如数据丢包、数据无序或延迟传输）也可能导致系统出现计算错误。

攻击型错误：攻击性错误包括对抗性攻击和针对性硬件错误等。对抗性攻击会影响输入数据或者神经网络模型的准确性，例如篡改感知图像或者误导输出动作，并延长计算延迟及功耗从而降低系统性能。针对性硬件错误，例如针对芯片存储单眼的 Rowhammer 攻击，针对时序的 Clkscrew 攻击会污染存储数据和计算结果，降低系统安全性和鲁棒性。

自主无人系统鲁棒性分析

近期 Google 和 Meta（原 Facebook）两家公司首次发文揭示了硬件计算栈错误对其数据中心的处理性能和安全性带来的负面影响，并指出这种影响随着半导体制程的缩放有加剧趋势[5,6]。因此本文以硬件计算栈错误为切入点，着重分析其对自主无人系统计算性能和可靠性的影响。

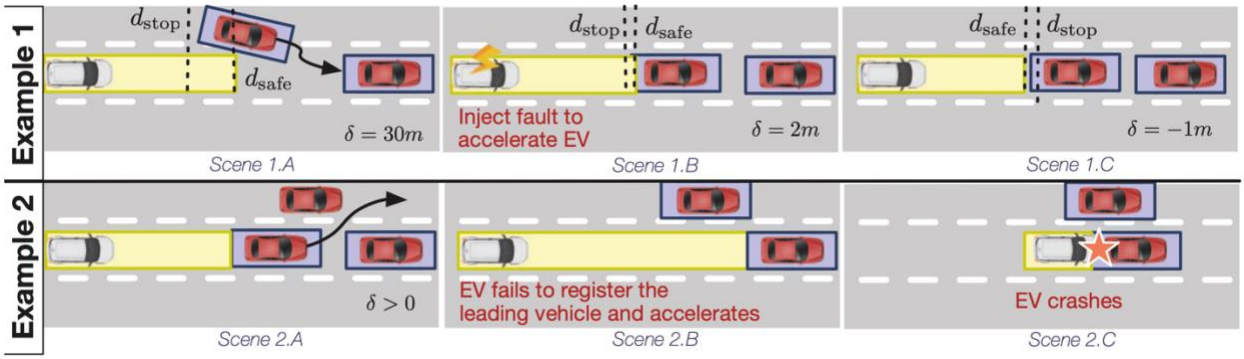

图 3. 示例场景：（1）故障注入实验导致危险行车状态；（2）特斯拉自动驾驶仪的真实例子与注入故障类似 （图源[7]）

计算栈错误影响自主无人系统运行安全。自主无人系统通常有复杂的计算栈系统和硬件来支持感知、规划及机器学习任务，计算栈错误会对系统安全运行产生严重威胁。以无人车为例，来自伊利诺伊大学香槟分校的研究人员[7]对百度 Apollo3.0 和英伟达专用自动驾驶系统 DriveAV 进行了鲁棒性研究，通过将系统硬件计算栈中的错误注入仿真器并收集

无人车的响应数据，研究人员发现了 561 个关键安全故障，并揭示这些计算栈错误会导致自动驾驶系统的油门、转向、车道类型判断值等出现错误，并导致行车事故。计算栈错误对无人机系统安全性同样带来威胁，近期来自佐治亚理工学院、哈佛大学和卡耐基梅隆大学的研究人员在开源无人机仿真器 PEDRA 重探索了硬件计算栈错误对基于端到端学习的无人机系统的影响，并发现 0.01%的计算栈错误便可致使自主导航飞行距离降低 10%[8]，这种负面影响在多机系统中会随着机间通信而浮动甚至加剧[9]。

前端计算单元高延迟高鲁棒性，后端计算单元低延迟低鲁棒性。不同自主无人系统都大致由前端（感知定位）和后端（决策控制）计算单元组成，但二者展示出截然不同的计算复杂度和鲁棒性。以无人车为例，近期来自罗切斯特大学和普思英察的研究者[10]在基于 Autoware 的自动驾驶模拟器上证明前端计算节点的计算延迟多位于 15 至 42 毫秒，而后端计算节点延迟仅需不到 1 毫秒，前端延迟占比超过 95%。但当硬件计算栈系统中发生软错误时，若软错误发生在前端计算单元，其传递比例仅为 1%左右，但当软错误发生在后端计算单元，其传递比例可高达 50%以上，对车辆安全行驶带来巨大威胁。无人机系统也展示出类似特征，近期来自哈佛大学和佐治亚理工学院的研究人员[11]在基于开源无人机模拟器 MAVBench 中的研究也表明无人机前端计算单元延迟占比 79%，但软错误只导致<1%的自主导航任务失败，相反后端计算单元延迟占比 21%，却导致高达 9%的自主导航任务失败。这主要是由于自主无人系统前端存在内在的错误屏蔽机制（如传感器融合和神经网络激鲁棒性），而后端计算简单但直接控制输出动作。

智能故障注入和检测方法对自主无人系统安全性分析至关重要。故障注入是一种用于测试计算和物理网络系统在错误下的弹性和处理能力的成熟方法，但在自主无人系统可靠性评估中却带来新挑战。由于软硬件计算栈的集成，传统的随机故障注入方法无法保证检测到所有关键安全故障，且将耗时数月甚至数年时间。因此加速故障注入实验且减小开销至关重要。核心优化思想是通过人工智能手段降低故障注入测试空间并有效挖掘关键安全场景。例如可以利用机器学习模型预测错误在自主无人系统计算栈中的传递概率，并优化随机故障注入为二叉树类故障注入方法以提高挖掘效率。还可以将传统单级故障分析方法转换为多层级故障分析，或利用贝叶斯网络进行因果推理和反事实推理，发现关键安全情况和故障，可比传统随机故障注入实现>3000 倍的加速[7]。

自主无人系统鲁棒性提升方法

针对计算栈错误对自主无人系统带来的可靠性影响，设计者可以采取不同软硬件方法来检测并尝试消除其负面影响。本文着重分析四种传统计算机系统常用的鲁棒性提升方法，指出其在自主无人系统中的应用和挑战，并提出对下一代轻量级混合自适应保护技术的展望。

异常检测。异常检测通常指利用数据挖掘手段识别数据中的异常点。设计者可以利用自主无人系统通常处理连续输入输出的特性进行异常点检测。例如，传感器的输入图像序列通常表现出时序一致性和连续性，正常行驶的车辆路径规划不太可能发出急转指令。因此连续时序输出在无故障情况下通常是有界的，而系统错误可能会打破这种时间一致性从而被检测到。异常检测可直接在软件计算栈中插入代码，因此开销较小，但有时会由于系

统输出动作的正确突变造成检测假阳性现象并替代正确指令，导致无法完全消除计算栈错误影响。

软件冗余。软件冗余（时间冗余）是指利用同一硬件多次执行部分代码，因为硬件计算栈软错误非常短暂，冗余执行可以帮助减轻有软错误影响，同时传感器数据中的时间冗余性也可用于检测计算栈错误[12]。软件冗余不需对硬件进行改动开销较小，但这种时间冗余方法会增加计算延迟，对自主无人系统的实时性带来挑战。

硬件冗余。硬件冗余（空间冗余）是指在两个或多个硬件上执行相同代码，例如 Tesla 的全自动驾驶芯片 FSD 中引入了双模冗余（DMR），两套完全相同的硬件计算栈同步处理相同信息并检测结果一致性。其他常用方法还有三重冗余（TMR），输出端表决电路会根据结果屏蔽故障硬件 。硬件冗余增加了硬件开销和成本，但对计算延迟基本没有影响，且可消除硬件软错误影响。对于边缘端小型系统（例如无人机），硬件冗余带来的负载变化可能会降低系统的机动性能[13]。

检查点 （Checkpoint）。检查点是指在系统执行过程中定期存储处理器状态，可在计算栈出现错误时利用回滚（rollback）将处理器恢复到先前的安全状态。检查点和回滚机制通常会带来很大计算延迟，一般并不适用于实时自主无人系统的保护。

混合自适应故障保护。根据计算栈错误对自主无人系统影响及不同软硬件故障保护方法的分析，我们认为下一代自主无人系统需要一种融合软硬件方法的轻量级自适应保护模式。例如利用前端高鲁棒性后端低鲁棒性的特征，设计者可以在前端感知定位计算单元应用软件保护方法，在后端决策规划计算单元应用硬件保护方法，从而实现整体保护机制的低延迟和低开销。

总结与展望

随着人工智能技术的蓬勃发展，无人车无人机等自主无人系统广泛与各种场景，其容错和检测错误的能力对于确保系统运行安全性、可靠性和弹性至关重要。本文介绍了闭环跨层自主无人系统计算栈及其潜在错误来源，并探索了计算栈错误对系统计算性能和可靠性的影响及几种主要保护方法。展望人工智能新十年，通向下一代安全可靠自主无人系统需要具备智能故障分析方法、量化比较和鲁棒性基准测试框架和轻型自适应保护技术，并持续提升在集群智能和车联网场景中自主无人系统的安全可靠性。

参考文献

[1] Wan, Zishen, et al. "Analyzing and Improving Resilience and Robustness of Autonomous Systems. " *Proceedings of the 41st IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*. 2022.
[2] Liu, Shaoshan, et al. "Robotic computing on fpgas." *Synthesis Lectures on Computer Architecture* 16.1, pp. 1-218, 2021
[3] Wan, Zishen, et al. "Circuit and system technologies for energy-efficient edge robotics." *2022 27th Asia and South Pacific Design Automation Conference (ASP-DAC)*. IEEE, 2022.
[4] Chandramoorthy, Nandhini, et al. "Resilient low voltage accelerators for high energy efficiency." *2019 IEEE International Symposium on High Performance Computer Architecture (HPCA)*. IEEE, 2019.
[5] Dixit, Harish Dattatraya, et al. "Silent data corruptions at scale." *arXiv preprint arXiv:2102.11245*, 2021.
[6] Hochschild, Peter H., et al. "Cores that don't count." *Proceedings of the Workshop on Hot Topics in Operating Systems*. 2021.

[7] Jha, Saurabh, et al. "Ml-based fault injection for autonomous vehicles: A case for bayesian fault injection." *2019 49th annual IEEE/IFIP international conference on dependable systems and networks (DSN)*. IEEE, 2019.
[8] Wan, Zishen, et al. "Analyzing and improving fault tolerance of learning-based navigation systems." *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021.
[9] Wan, Zishen, et al. "Frl-fi: Transient fault analysis for federated reinforcement learning-based navigation systems." *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2022.
[10] Gan, Yiming, et al. "Braum: Analyzing and protecting autonomous machine software stack." *2022 IEEE 33rd International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2022.
[11] Hsiao, Yu-Shun, et al. "Mavfi: An end-to-end fault analysis framework with anomaly detection and recovery for micro aerial vehicles." *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 2023
[12] Jha, Saurabh, et al. "Exploiting temporal data diversity for detecting safety-critical faults in AV compute systems." *2022 52nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*. IEEE, 2022.
[13] Krishnan, Srivatsan, et al. "Roofline model for uavs: A bottleneck analysis tool for onboard compute characterization of autonomous unmanned aerial vehicles." *2022 IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, IEEE, 2022