Run-Time Assurance for Onboard COTS-Based AI in Safety-Critical LEO Small-Satellite Systems

Omer Khalid[1,2], Jose-Carlos Gamazo-Real[1,2(✉)], and Elena Roibás-Millán[1]

[1] Universidad Politécnica de Madrid, Instituto Universitario "Ignacio Da Riva" (IDR-UPM), 28040 Madrid, Spain
{omer.hkhalid,josecarlos.gamazo,elena.roibas}@upm.es

[2] Universidad Politécnica de Madrid, ETSI de Sistemas Informáticos, 28031 Madrid, Spain

Abstract. Safety-critical systems are increasingly applying Artificial Intelligence (AI) to enhance perception, autonomy, and decision-making. In parallel, the widespread adoption of Commercial Off-The-Shelf (COTS) hardware offers substantial benefits in terms of cost, computational performance, and development speed. However, combining unpredictable, nondeterministic AI models with heterogeneous and uncertified COTS devices introduces significant challenges for guaranteeing reliability, predictability, and certifiability, which are essential requirements in systems where failures can be catastrophic. This article examines these emerging challenges and proposes a layered Run-Time Assurance (RTA) methodology tailored for AI-enabled functions deployed on COTS platforms for safety-critical systems. The approach introduces four complementary monitoring layers, such as input, inference, output, and system-level behavioural supervision, which are designed to detect anomalies, timing overruns, semantic inconsistencies, and elevated mission risks before they escalate into unsafe states. A detailed case study of a Low Earth Orbit (LEO) microsatellite, using the UPMSat-2 from the IDR-UPM research institute as the reference platform, is presented to demonstrate how the proposed RTA framework can maintain safe operation during perception uncertainties, hardware-induced degradations, and collision-risk assessments. The proposed methodology aims to support future certifiable integration of onboard AI pipelines, enabling reliable autonomy in resource-constrained spacecraft and advancing the safe adoption of high-performance COTS computing within New-Space systems.

Keywords: Run-Time Assurance · Artificial Intelligence · Commercial Off-The-Shelf · Safety Monitor · Safety-Critical System · LEO Small Satellite · New Space

1 Introduction

Safety-critical systems are entering a new phase of evolution, driven by the growing demand for higher levels of autonomy onboard in domains such as aerospace, transportation, and industrial robotics [1]. As these systems increasingly operate with reduced

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2027
O. Gervasi et al. (Eds.): ICCSA 2026, LNCS 16767, pp. 445–460, 2027.
https://doi.org/10.1007/978-3-032-30491-9_28

human supervision, they must be able to perceive complex environments, adapt to dynamic conditions, and make reliable decisions in real time, capabilities for which Artificial Intelligence (AI) has emerged as a key enabler [2]. Moreover, the adoption of Commercial Off-The-Shelf (COTS) hardware is gaining popularity across multiple safety-critical domains, because they offer clear advantages such as low cost, high computational performance, shorter development cycles, and faster technological evolution compared to certified or radiation-hardened systems [3, 4]. These benefits are particularly appealing in the New Space sector, where small satellites require faster and less costly development, and testing methodologies apart from coverage, parameters such as availability, mean time between failures, and reliability [5].

Nonetheless, integrating AI-based functionalities in COTS hardware into safety-critical systems also brings extra challenges. AI models executed on COTS processors behave unpredictably and its nondeterministic components whose internal decision mechanisms cannot be fully traced or formally verified [2, 6]. Their performance and outputs could differ based on input, hardware load, thermal conditions, or minor implementation details that make them inherently difficult to certify within traditional assurance frameworks [7]. At the same time, safety-critical systems are built upon principles of deterministic, explainable, and verifiable behaviour, where every operational state must remain within predefined safety margins, and timing guarantees must be respected under all conditions [2, 8]. This creates a mismatch considering that AI-enabled COTS devices offer exceptional computational capability but lack the predictability and certifiability required for systems whose failure modes could bring serious losses. Bridging this gap requires new architectural strategies capable of monitoring the behaviour of AI-driven COTS components while maintaining the strict safety guarantees that these domains require.

To bridge this gap between advanced AI capabilities in COTS devices and strict safety requirements, researchers have drifted towards Run-Time Assurance (RTA) methods. RTA provides continuous online supervision of untrusted components which aren't present in static verification methods. They can supervise AI models, while guaranteeing that safety boundaries are not violated during operation [9, 10]. In its classical form, RTA pairs an unverified high-performance controller with a verified safety controller and employs a real-time monitor to decide when intervention is required. This paradigm is emerging as a practical and scalable method for integration of AI in safety-critical systems, particularly when full formal verification is infeasible or when systems must operate on heterogeneous and potentially unreliable COTS hardware. Preliminary works have explored modular and multilayer RTA structures, including frameworks with multiple controllers and monitoring layers [11], as well as safety-filtering architectures that separate controller, monitor and override layers [12]. However, none introduces a unified four-layers model tailored to AI perception on heterogeneous COTS hardware, making the architecture proposed here distinct for safety-critical space systems. The novelty of the present work lies in proposing a unified four-layer RTA model tailored to onboard AI perception on heterogeneous COTS platforms, integrating different behavioural monitoring within a single safety-assurance architecture.

This article proposes an approach for safely integrating AI-based functionalities into safety-critical systems operating on COTS hardware. It proposes a layered RTA

architecture and safety monitors that supervise AI components through input, inference, output, and system-level behavioural monitors. The motivation of this study is the need to exploit AI and COTS hardware in resource-constrained New Space missions without compromising safety. The approach ensures that mission safety is preserved even when AI models exhibit uncertainty, nondeterminism, or timing violations associated with heterogeneous COTS platforms. A detailed Low Earth Orbit (LEO) microsatellite case study, using the UPMSat-2 from the IDR-UPM (Instituto Universitario "Ignacio Da Riva") as the reference platform [13], demonstrates how the proposed approach maintains safe operation during perception anomalies and hardware-induced degradations. The work contributes to a practical path towards certifiable AI deployment in resource-restricted and critical domains such as aerospace. The proposed framework is analytical and architectural in scope; no dedicated analysis software was used at this stage. Instead, it was developed through traditional engineering analysis.

The remainder of this article is structured as follows. Sect. 2 introduces the motivation for contextualising the increasing adoption of AI in safety-critical systems, beside the outline of the challenges faced when using resource-constrained COTS devices rather than certified ones. Then, in Sect. 3, a description of RTA systems and their operations is presented. In Sect. 4, the RTA layered methodology is presented and described with a detailed case study involving a scientific LEO microsatellite equipped with an AI-based perception pipeline demonstrating how this approach can be instantiated in a real mission scenario. Finally, Sect. 5 presents the conclusions and outlines future directions of research.

2 Motivation Scenario: Integration of COTS Devices in Safety-Critical Applications

Safety-critical systems like as avionics controllers, spacecrafts, self-driving vehicles, industrial systems, and medical equipment increasingly rely on complex software as they integrate advanced sensing, perception, and decision-making capabilities while running on tightly constrained ad-hoc hardware. These must meet strict real-time and reliability requirements while operating with limited Central Processing Unit (CPU) capacity, memory, and energy. However, with the rapidly increasing in processing capabilities, the possibility of upgrading the hardware to meet these demands is infeasible due to weight constraints, higher certification burdens, increased energy consumption, and cost considerations. This creates a pressing need for techniques to optimise resource usage without compromising safety guarantees.

2.1 Safety Considerations in Hardware and Software Architectures

The emergence of edge-computing in safety-critical systems introduces new demands on how hardware and software architectures ensure safety. Unlike traditional centralised architectures, edge systems must perform computations under strict latency, energy, and reliability constraints. Hardware platforms deployed at the edge often combine different processing units, such as multicore CPUs, Field Programmable Gate Arrays (FPGA),

Graphics Processing Units (GPU), and Deep Learning (DL) accelerators, each with different timing behaviour and resource-sharing patterns [14]. This heterogeneity enables high performance; however, it complicates predictability and affects robustness and reliability in safety-critical contexts. As a result, determining how current hardware architectures provide deterministic behaviour and ensure overall reliability becomes a central question in safety-critical systems. On the software side, modern software systems rely on complex middleware, virtualisation, containerisation, and distributed computing [15]. These architectures must coordinate tasks with different levels of criticality while sharing limited hardware resources in real time. It introduces additional layers of nondeterminism and raises questions about whether they can maintain the safety guarantees required in safety-critical systems. Moreover, the growing adoption of AI adds further difficulties, as it black-box behaviour and high computational demands, can produce unexpected outputs, and complicate the certification process, which sacrifices and affects the system reliability [16].

2.2 COTS Devices Integration for Safety-Critical Edge Computing

The increasing interest in using of COTS hardware in safety-critical edge systems in the edge is influenced by its obvious advantages: lower cost, higher performance, shorter development timelines, and rapid progression of computing technologies [3]. These benefits are especially attractive in domains such as aerospace, where traditional hardware is expensive, slow to certify, and often lags cutting-edge commercial processors with respect to computational capabilities [17]. However, COTS components adoption into this sector introduces significant challenges for systems that must meet strict safety, reliability, and robustness requirements [18]. COTS hardware is typically not designed with determinism or long-term reliability in mind. its limitations can become even more exposed when deployed at the edge in harsh environments with strong thermal, radiation, or vibration constraints. For aerospace systems in particular, disadvantages such as radiation susceptibility and uncertainty in long-term component behaviour raise concerns about how safety can be assured when using COTS hardware [19]. Figure 1 summarizes the key concepts of safety-critical edge AI-based systems, as it includes domains such as autonomous vehicles, aerospace, and space technologies, all of which require high-safety considerations to ensure reliable operation. AI introduces additional challenges due to its behaviour, especially when deployed on resource-constrained edge devices.

3 Run-Time Assurance and Safety Monitoring Methodology

3.1 RTA Methodology

Safety-critical systems operate under strict safety constraints, where failures may have severe consequences including death, environmental damage, or significant economic losses; therefore, they demand high guarantees of correct and reliable behaviour. Meeting these requirements has become increasingly challenging with the integration of advanced AI components, who's opaque (black-box) nature and limited interpretability make it difficult to fully verify their operation under all operational conditions. This challenge is

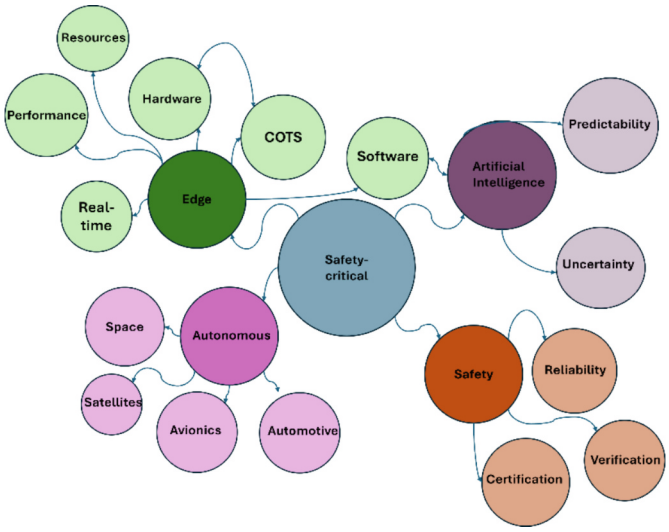


Fig. 1. Relationships among key concepts in safety-critical systems and AI-COTS edge computing.

further magnified by the widespread adoption of COTS hardware and software, which, although cost-effective and high-performance, are not certified for the timing, reliability, or functional constraints demanded by safety-critical systems. Their use may cause performance variability and unnoticed interactions that may alter or degrade the behaviour of AI algorithms deployed in such environments.

As shown in Fig. 2, the architecture of an RTA framework typically comprise three main components: an untrusted or experimental controller that carries out the primary mission objectives; a safety controller that is simpler, verified, and capable of keeping the system within a guaranteed safe region; and a safety monitor that decides when to shift authority between the two. This monitor evaluates real-time measurements, derived state estimates, and predicted trajectories to determine whether the system is approaching unsafe regions of its operational space. If a violation of safety constraints is imminent, the monitor ensures a deterministic low-latency switch to the safety controller, which maintains the autonomous system within predefined boundaries until normal operation can resume safely.

RTA provides a promising approach to mitigating these risks, particularly those arising from performance variability and hidden interactions associated with the use of COTS technology, while increasing the level of assurance in such systems. It is traditionally used when an untrusted or experimental controller is deployed within an autonomous platform, enabling a seamless transition to a conventional or certified safety controller when necessary. In this context, the RTA mechanism continuously monitors the operational state of the system, ensuring that AI-driven actions and their interactions with COTS components remain within a predefined safety margin. When potential safety violations are detected, the control is immediately transferred to a verified backup controller or a safe operational mode, thus preserving overall system safety despite the presence

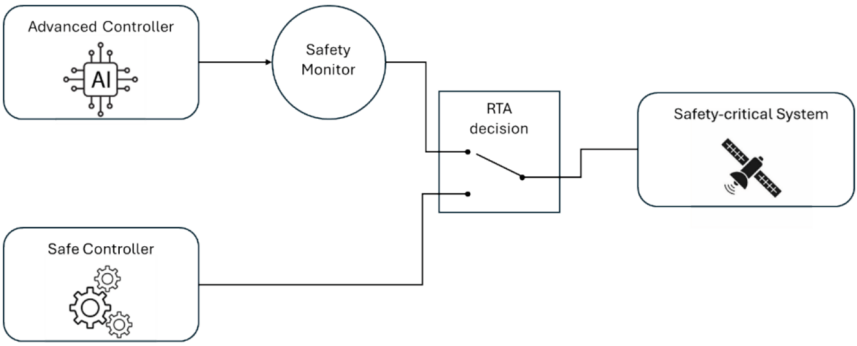


Fig. 2. Conventional architecture of an RTA framework.

of opaque AI components and uncertified COTS elements. The main objective in RTA is to design a switching policy that determines, at each decision point, whether control should reside with the verified safe controller or the untrusted one. This strategy must guarantee the continued safety of the system while simultaneously striving to optimise the relevant performance criterion.

3.2 RTA Variations

Several variations can be considered in the RTA methodology, but they all follow the core idea of combining an untrusted controller with a verified safe controller under the supervision of a run-time monitor. In the academic literature, RTA architectures are typically classified into the following main types:

- Simple-switching RTA (Simplex): A monitor switches from an untrusted controller to a certified safe controller whenever the system approaches unsafe conditions, ensuring immediate protection with minimal architectural complexity [20].
- Shielding-Based RTA: A safety shield filters or corrects unsafe control actions instead of switching controllers, allowing continuous operation while preventing unsafe behaviour in real time [12, 21].
- Predictive or Look-ahead RTA: The monitor predicts future trajectories under proposed actions and intervenes early if a safety violation is likely, enabling anticipatory rather than reactive assurance [12].
- Recovery-Based RTA: The verified safe controller gradually guides the system back to safe conditions, supporting smoother transitions and stable operation, as opposed to sudden switching [22].

3.3 Deployment of RTA and Safety Monitors for Safety Critical System

RTA has great potential as a practical approach to enable the deployment of advanced software-intensive functionalities, particularly those that rely on AI in safety-critical systems. Unlike offline verification techniques, which attempt to exhaustively prove correctness before deployment, RTA operates continuously during execution. It supervises system behaviour in real time, detecting deviations from safe operating boundaries and

intervening before unsafe states can propagate. This characteristic makes RTA particularly attractive for safety-critical systems, and especially for small satellites and other New-Space platforms. In these environments, budget constraints, compressed development schedules, and the increasing adoption of commercial processors introduce a non-negligible likelihood of unexpected behaviours once the system is in orbit.

Designing an RTA begins with representing the system in terms that allow the safety monitor to predict how future behaviour may evolve under different controllers. The operating conditions of the system are divided into a collection of discrete quantised states, and each controller defines how these states change over time. An RTA design must then specify how to choose between the two controllers at every moment, so the system remains within its safe operating region while still attempting to achieve mission objectives. To do this, the designer establishes a set of unsafe states that the system must never enter and defines a monitoring logic that can detect when the untrusted controller is likely to drive the system toward those unsafe regions. The switching logic then replaces the untrusted controller with the safe controller whenever necessary. Because different choices of controller affect mission performance, such as efficiency, data quality, or energy consumption, the design may also incorporate a way to evaluate how desirable a given trajectory is and adjust how aggressively the untrusted controller is used. Ultimately, the goal is to construct a switching strategy that keeps the system safe while allowing the highest mission value possible.

The deployment of RTA in safety-critical systems introduces additional considerations. Due to the limited resources of the autonomous onboard computers, they should be capable of handling the computational overhead associated with continuous monitoring, and the monitor itself must be immune to the same faults it has been designed to detect. Integrating RTA with COTS hardware also requires ensuring that timing determinism can be maintained when running on heterogeneous processing architectures such as GPUs or multicore systems. These challenges highlight the importance of lightweight and certifiable RTA methodologies that can function in a reliable manner within the constraints of small-satellite platforms. Table 1 shows examples of RTA methods developed in different safety-critical domains.

Table 1. RTA methods developed in different safety-critical domains.

Name	Description	Use case/Domain implementation	Limitation with respect to the presented work	Ref.
Simplex RTA	Trusted safety controller supervises untrusted controller; monitor switches control when safety is threatened.	Automatic vehicle braking system	Does not explicitly monitor AI perception nor COTS effects	[20]

(*continued*)

Table 1. (*continued*)

Name	Description	Use case/Domain implementation	Limitation with respect to the presented work	Ref.
Safety filtering RTA	Online monitor filters or corrects unsafe commands in real time, ensuring safety with untrusted controllers.	Practical method for safety-critical Cyber-Physical Systems (CPS)	Primarily focused on control and command	[12]
Automatic-GCAS (Ground Collision Avoidance System)	Real deployed RTA system that monitors aircraft state and automatically overrides pilot control to avoid ground collision.	Aviation systems	No AI, no perception uncertainty, certified hardware	[23]
ULGEN Framework	RTA modules with multiple safe and advanced controllers plus a decision module	Robotic CPS platforms, such as case studies with real robot implementation	Does not explicitly monitor AI perception nor COTS effects	[11]

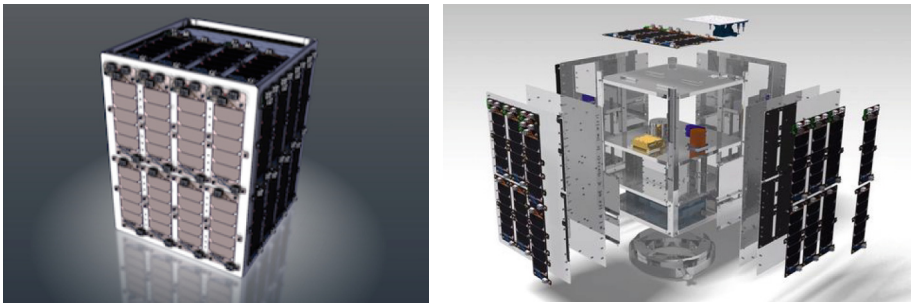
4 Case Study: RTA-Enabled Safety Monitoring in Scientific LEO Small Satellites

4.1 System Overview of LEO Small Satellites

LEO satellites typically operate between 160 and 1,500 Km above the earth surface [24], making it one of the most rapidly evolving sectors in modern space technology. Their proximity to earth allows low-latency communication [25], along with their lower cost, smaller mass, and higher scalability compared to both Medium Earth Orbit (MEO) and geostationary Orbit (GEO) satellites [26], making them highly suitable for modern missions. These features are well-suited for applications requiring hard real-time connectivity, for instance, broadband internet, IoT, mobility services and earth observation [27]. In recent years, LEO small-satellite constellations have gained more interest as there are some projects of large-scale satellite constellation including Airbus, OneWeb, and SpaceX Starlink. These initiatives demonstrate not only the feasibility of deploying mass-produced satellites, but also the potential of integrating AI capabilities on board at scale. However, LEO platforms also introduce a set of technical challenges that AI systems on board must help address, such as shorter operational lifetimes, constrained energy resources, and strict real-time pro-cessing requirements. Consequently, AI models designed for LEO small satellites must be compact, energy-efficient, and operationally robust, while simultaneously ensuring safety objectives of the mission.

4.2 UPMSat-2 Microsatellite as a Reference Platform

UPMSat-2 is a microsatellite developed by the IDR-UPM research institute of the Universidad Politécnica de Madrid [13]. Its approximate mass is 50 kg, with dimensions of 0.5x0.5x0.6 m, and a power output of 15 W, as illustrated in Fig. 3. It was put into orbit on September 3, 2020, thanks to the In-Orbit Demonstration and Validation (IOD/IOV) program from the European Commission. UPMSat-2 operates in a polar orbit at an altitude of approximately 600 km, within the LEO regime, and has an estimated operational lifespan of two years. However, after more than five years in orbit, it successfully completed its mission after re-entering Earth's atmosphere on October 11, 2025.



(a) (b)

Fig. 3. Images of the envelope of the UPMSat-2 microsatellite (a) and the deployment of the electro-mechanical elements that constitute the spacecraft (b).

The satellite is designed as a technology demonstrator, carrying a payload of several experiments from research groups and industry, mainly focused on assessing equipment performance in space. The system architecture incorporates a magnetic attitude control system and various sensors, such as voltage, intensity, and temperature, which are used for platform monitoring, experimentation, and over-all spacecraft operations. Communications with the ground station are established through two radio links using the UHF 400 MHz band, and the satellite includes an onboard mission computer responsible for all data handling, control functions, and experiment management. The computer utilises a Gaisler LEON3 processor and incorporates peripherals such as memories and analogue/digital interfaces. Its primary functions include attitude determination and control (ADCS), platform monitoring, data storage, telemetry and telecommand (TMTC) management, and payload operation. The TMTC subsystem plays an important role in maintaining communication with the ground station through the telecommunications hardware, and the importance of the ADCS in ensuring the correct orientation of the satellite relative to the Earth. The RTA and safety monitoring approach proposed in this article is intended for integration into future iterations of UPMSat-2, which is considered the reference platform for the case study.

4.3 Proposed Application of RTA and Safety Monitors

This case study demonstrates how a layered RTA and safety monitor can be instantiated for a LEO small satellite that uses AI-based image processing on board. This functionality can be implemented on a COTS device to support conjunction assessment and autonomous collision avoidance.

The proposed system consists of a scientific LEO small satellite equipped with an AI-based image processing algorithm on board capable of tracking debris and other space objects. This algorithm estimates their relative motion to predict the probability of an intersection with the satellite's trajectory and to prevent possible collisions. To enable these functions, the satellite is equipped with (i) a wide field of view (FoV) optical camera to capture the surrounding space environment, serving as the raw input for the AI model, (ii) an ADCS to determine the spacecraft orientation that ensure images are correctly orientated, (iii) data from receivers of the Global Navigation Satellite System (GNSS), inertial sensors, and ground-based tracking to provide position, velocity, and absolute time (PVT), and (iv) a COTS System-on-a-Chip (SoC) module executing an AI perception pipeline. The processed perception output can provide the onboard mission computer (OBC) with an estimate of the probability of collision (*Pc*), enabling the satellite to remain within its safety envelope (*Pc* below a mission-defined threshold) and to execute a minimal-risk avoidance burn if the risk increases or the AI becomes untrustworthy at runtime. Fig. 4 demonstrating the high-level architecture of the proposed system, showing how sensor inputs, mission information and AI perception outputs are combined.

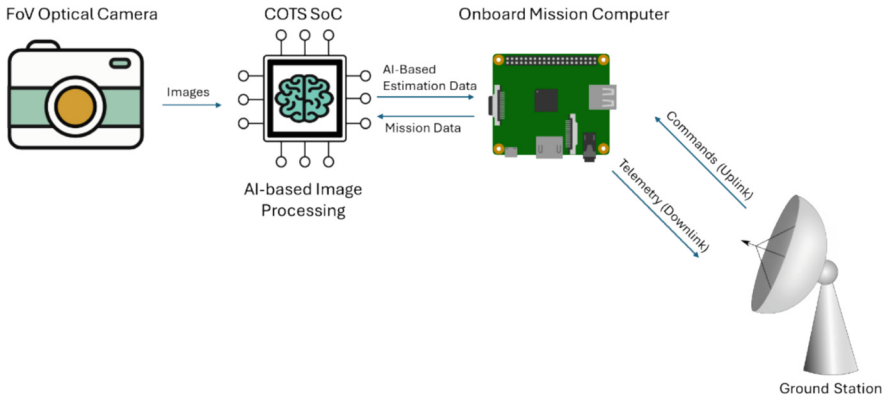


Fig. 4. Proposed onboard architecture of the AI-based perception system.

Given the safety-critical nature of satellites, their operation must remain safe in all circumstances. However, the behaviour of AI-based systems, characterised by black-box properties and inherent uncertainties, can make them difficult to fully trust. In addition, employing COTS components in harsh environments introduces additional reliability concerns. As summarised in Table 2, the failure modes associated with the use of black-box AI models and COTS devices are mapped to their corresponding system hazards. Both failure categories have the potential to jeopardise the hard real-time deadlines

associated with perception and guidance tasks. RTA and mode switching are therefore essential to maintaining safety without requiring modifications to the black-box AI models.

Table 2. Analysis of failure modes and reliability hazards of the proposed system.

Failure mode (source)	Description	Hazard/effect
Missed detection (AI)	Undetected object from image	Underestimated *Pc*, causing collision risk
False-positive detection (AI)	Unreliable collision prediction	Unreliable prediction
Latency overruns (AI/COTS)	Inference after passing the deadline	Degraded *Pc* accuracy
Input corruption (camera)	Blur, saturation and other effects	False negative, positive predictions
Compute degradation (COTS)	GPU overheating, memory pressure	Monitor dropout, loss of determinism

4.4 RTA System Proposal

Proposed Layered RTA Architecture. The proposed RTA and safety monitor for LEO small satellites, shown in Fig. 5, is currently under consideration for implementation. It shows that different monitors cooperate to detect anomalies before they propagate and lead to unsafe decisions for the spacecraft. It is designed to serve as a safety mechanism capable of maintaining satellite operation despite failures in the perception and estimation system. It is composed of four monitoring layers (input, inference, semantic, and behaviour layers) and their associated signals, which feed the safety system. The design can follow established taxonomies of perception-safety monitoring and the RTA literature [28].



Fig. 5. Schematic of the proposed layered RTA architecture.

The four monitoring layers that constitute the proposed system are described in detail below:

- Layer 1: Sensors or input monitors. This layer is used to validate the optical inputs before the AI inference, providing an input health output $h_{in} \in [0,1]$ and event flags.

- Layer 2: Inference monitors. This layer is in charge of ensuring the COTS pipeline meets compute and timing constraints, providing an output of compute health score h_{cmp} and timing OK/Overrun, thermal and memory warnings.
- Layer 3: Output monitors. This layer is responsible for enforcing physics and geometry-based checks on AI tracking, assessing motion, size, orbital behaviour, and cross-sensor consistency, producing a vector and count of contract violations.
- Layer 4: System-behaviour monitors. This layer is responsible for checking safety envelopes at mission level, providing a risk score assessment R_{sys} and recommending safe actions (e.g., "Hold" action to maintain current state of AI navigation and "ManeuverNow" action to transfer control to the safe controller).

The closed-loop system, comprising the satellite, perception module, and guidance controller, can be modelled as a finite and discrete-time plant as follows, taking into account that the finite-state and discrete-time abstractions are a common way to pose RTA switching as a computational problem over a plant [28].

$$M = (Q, q_0, A, D) \tag{1}$$

where Q is a large and finite set of quantized states, $q_0 \in Q$ is the initial state bundles orbital/relative motion and monitor summaries (input health h_{in}, compute health h_{cmp}, semantic-violation count v_{sem}, and system risk R_{sys}), A is a finite action set, and $D{:}Q \times A \to Q$ is a deterministic transition function over a fixed sampling period to specify the next state based on the current state Q and the action A taken. This representation allows the RTA switching problem to be formulated in terms of quantized system states and discrete controller actions.

With respect to the actions (A), the representation can be performed using the two-action RTA formulation:

$$A = \{U, S\} \tag{2}$$

where U selects the untrusted (AI-dependent) controller and S selects the safe classical controller, defining the two possible RTA actions: maintaining the AI-dependent controller or switching to the verified safe controller

For the switching policy π, a memoryless policy can be adopted, enabled by the layered monitors so that switching depends only on the current quantized state:

$$\pi : Q \to \{U, S\} \tag{3}$$

Operation Modes. The proposed RTA system can enable switching among three modes, such as nominal, suspicious, and safe, as described below:

- Nominal (AI-trusted): This is the nominal mode where the AI inference is generated in trustful manner, and all safety contracts are satisfied and no hard violations; during this mode it uses the AI-enhanced fully navigation. It operates when the following conditions are satisfied:

$$h_{in} \geq \theta_{in} \tag{4}$$

$$h_{cmp} \geq \theta_{cmp} \tag{5}$$

$$v_{sem} \leq \alpha \tag{6}$$

$$Pc \leq Pc_{low} \tag{7}$$

where θ_{in} is the threshold for a good image input, θ_{cmp} is the threshold for compute reliability, v_{sem} is the semantic violation count produced by Layer 3 of monitor layers, α is the allowed number of soft violations, and Pc_{low} is the threshold used to decide if the system remains safe.

- Suspicious (degraded trust): It happens with soft anomalies, such as $h_{cmp} < \theta_{cmp}$, repeated soft safety contract violations, or Pc trending upward. During this mode, AI outputs are down weighted and request ground confirmation.
- Safe (safe verified): It occurs at hard violations, such as timing greater than L_{max} (deadline miss), thermal limit exceeds, persistent semantic violations, or $Pc \leq Pc_{high}$, where Pc_{high} is the upper bound on acceptable collision probability. During this mode the AI influences are disabled and the control will be handled back to the verified controller.

5 Conclusion and Future Work

This article considers the growing reliance on AI and COTS hardware in safety-critical systems and the resulting challenges in guaranteeing reliability, determinism, and certifiability. To address these issues, a system based on layered RTA and safety monitors is proposed, incorporating multilevel safety monitors to maintain safe operation even when deploying computationally demanding and opaque AI models on heterogeneous COTS platforms. The effectiveness of multi-layer monitoring combined with a structured switching policy is shown through a LEO microsatellite case study, using the UPMSat-2 from the IDR-UPM as the reference platform, hence demonstrating safe behaviour under nominal, degraded, and unsafe conditions. The findings demonstrate that RTA offers a practical approach for integrating advanced image processing pipelines into resource-constrained space systems without requiring full formal verification.

Future work will be centred on the practical implementation of the proposed layered RTA architecture and multi-level safety monitoring within an onboard AI image-processing payload for LEO small satellites. This effort aims to prove that mission safety can be preserved even when relying on computational components that exhibit black-box behaviour and limited interpretability. Through the deployment of RTA-supervised AI modules on heterogeneous COTS platforms in orbit, future developments intend to provide practical and thorough evidence that advanced perception algorithms can operate safely under real mission conditions despite timing variability, hardware uncertainty, or inference anomalies. On a broader horizon, this research direction contributes to the development of certification pathways capable of integrating AI-driven subsystems in safety-critical space missions.

Acknowledgments. This work was partially funded by the IDR-UPM (Instituto Universitario "Ignacio Da Riva") research institute of the Universidad Politécnica de Madrid (UPM) under the project "MLPLUS-CM: Consolidación de una Plataforma de Microsatélite y Tecnologías para Investigación Científica Espacial Avanzada" (Ref.: TEC-2024/TEC-263) of the Spanish regional call for proposals 2024 titled "Grants for R&D projects carried out in collaboration between research groups belonging to universities and research organisms of the Community of Madrid" in the modality of R&D Activity Programs in Technologies. The research work was also partially funded by the "Programa propio de investigación de la Escuela Técnica Superior de Ingeniería de Sistemas Informáticos (ETSISI)" within the call "Convocatoria de ayudas para la presentación de trabajos en congresos de investigación y reuniones científico-técnicas 2026" of the ETSISI of the UPM. The authors also acknowledge the collaboration of the ETSISI through its doctoral program in Computer Science and Technologies for Smart Cities.

References

1. Perez-Cerrolaza, J. et al.: Artificial intelligence for safety-critical Systems in Industrial and Transportation Domains: a survey. ACM Comput. Surv. **56** (2024). https://doi.org/10.1145/3626314
2. Wang, Y., Chung, S.H.: Artificial intelligence in safety-critical systems: a systematic review. Ind. Manag. Data Syst. **122**, 442–470 (2021). https://doi.org/10.1108/IMDS-07-2021-0419
3. Budroweit, J., Patscheider, H.: Risk Assessment for the Use of COTS Devices in Space Systems under Consideration of Radiation Effects. Electronics **10**, 1008 (2021). https://doi.org/10.3390/electronics10091008.
4. Schwierz, A., Forsberg, H.: Assurance case to structure COTS hardware component assurance for safety-critical avionics. In: IEEE/AIAA 37th Digital Avionics Systems Conference (DASC), pp. 1-10. Institute of Electrical and Electronics Engineers Inc. (2018). https://doi.org/10.1109/DASC.2018.8569774.
5. Gamazo-Real, J.C., Zamorano-Flores, J.R., Sanz-Andrés, Á.: Integration of COTS processing architectures in small satellites for onboard computing using fault injection testing methodology. In: Software Engineering and Formal Methods (SEFM 2021 Collocated Workshops), pp. 333–347. Springer (2022). https://doi.org/10.1007/978-3-031-12429-7_24
6. Perez-Cerrolaza, J., Abella, J., Kosmidis, L., Calderon, A.J., Cazorla, F., Flores, J.L.: GPU devices for safety-critical systems: a survey. ACM Comput. Surv. **55**, 147 (2023). https://doi.org/10.1145/3549526
7. Lee, S., Kang, W., Bertogna, M., Chwa, H.S., Lee, J.: Timing guarantees for inference of AI models in embedded systems. In: Real-Time Systems, pp. 259–267. Springer (2025). https://doi.org/10.1007/s11241-025-09445-9
8. Raj, C.A.: Model-based approaches in safety-critical embedded system design. European J. Comput. Sci. Inform. Technol. **13**, 30–41 (2025) https://doi.org/10.37745/ejcsit.2013/vol13n203041
9. Rahman, Q.M., Corke, P., Dayoub, F.: Run-time monitoring of machine learning for robotic perception: a survey of emerging trends. IEEE Access. **9**, 20067–20075 (2021). https://doi.org/10.1109/ACCESS.2021.3055015
10. Neto, A.V.S., Camargo, J.B., Almeida, J.R., Cugnasca, P.S.: Safety Assurance of Artificial Intelligence-Based Systems: a systematic literature review on the state of the art and guidelines for future work. IEEE Access. **10**, 130733–130770 (2022). https://doi.org/10.1109/ACCESS.2022.3229233

11. Yalcinkaya, B., Torfah, H., Desai, A., Seshia, S.A.: Ulgen: a runtime assurance framework for programming safe cyber-physical systems. IEEE Trans. Comput.-Aided Design Integr. Circ. Syst. **42**, 3679–3692 (2023). https://doi.org/10.1109/TCAD.2023.3246386
12. Hobbs, K.L., Mote, M.L., Abate, M.C.L., Coogan, S.D., Feron, E.M.: Runtime Assurance for Safety-Critical Systems: an introduction to safety filtering approaches for complex control systems. IEEE Control. Syst. **43**, 28–65 (2023). https://doi.org/10.1109/MCS.2023.3234380
13. Instituto Universitario "Ignacio Da Riva" (IDR-UPM) of the Universidad Politécnica de Madrid: UPMSat-2 Microsatellite, https://www.idr.upm.es/en/upm-sat2. Accessed 19 Mar 2026
14. Barbirotta, M. et al.: Fault-Tolerant Hardware Acceleration for High-Performance Edge-Computing Nodes. Electronics. **12**, 3574 (2023) https://doi.org/10.3390/electronics12173574
15. Urblik, L., Kajati, E., Papcun, P., Zolotová, I.: Containerization in Edge Intelligence: A Review. Electronics **13**, 1335 (2024). https://doi.org/10.3390/electronics13071335.
16. Biondi, A., Nesti, F., Cicero, G., Casini, D., Buttazzo, G.: A safe, secure, and predictable software architecture for deep learning in safety-critical systems. IEEE Embed. Syst. Lett. **12**, 78–82 (2020). https://doi.org/10.1109/LES.2019.2953253
17. Furano, G. et al.: Towards the use of artificial intelligence on the edge in space systems: challenges and opportunities. IEEE Aerosp. Electron. Syst. Mag. **35**, 44–56 (2020). https://doi.org/10.1109/MAES.2020.3008468
18. Ma, K. et al.: Radiation-induced degradation analysis and reliability modeling of COTS ADCs for space-borne miniature fiber-optic gyroscopes. IEEE Trans. Instrum. Meas. **70** (2021). https://doi.org/10.1109/TIM.2021.3054419
19. Brunetti, G., Campiti, G., Tagliente, M., Ciminelli, C.: COTS devices for space missions in LEO. IEEE Access. **12**, 76478–76514 (2024). https://doi.org/10.1109/ACCESS.2024.3405373
20. Slagel, J.T., White, L.M., Dutle, A., Muñoz, C.A., Crespo, N.: A formal verification framework for runtime assurance. In: NASA Formal Methods (NFM), pp. 322–328. Springer (2024). https://doi.org/10.1007/978-3-031-60698-4_19
21. Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. In: 32nd AAAI Conference on Artificial Intelligence, AAAI 2018. pp. 2669–2678. AAAI press (2018). https://doi.org/10.1609/aaai.v32i1.11797.
22. Nigam, V., Talcott, C.: Automating recoverability proofs for cyber-physical systems with runtime assurance architectures. In: Theoretical Aspects of Software Engineering (TASE), pp. 1–19. Springer (2023). https://doi.org/10.1007/978-3-031-35257-7_1
23. Swihart, D.E., Griffin, E.: Automatic ground collision avoidance system (AutoGCAS). In: Proceedings of the 13th WSEAS international conference on Systems (2009).
24. Angirasa, D., Ayyaswamy, P.S.: Review of evaluation methodologies for satellite exterior materials in low earth orbit. J. Spacecr. Rocket. **51**, 750–761 (2014). https://doi.org/10.2514/1.A32742
25. Wang, Q., Chen, X., Qi, Q., Li, M., Gerstacker, W.: Multiple-satellite cooperative information communication and location sensing in LEO satellite constellations. IEEE Trans. Wirel. Commun. **24**, 3346–3361 (2025). https://doi.org/10.1109/TWC.2025.3530083
26. Osoro, O.B., Oughton, E.J.: A techno-economic framework for satellite networks applied to low earth orbit constellations: assessing Starlink, OneWeb and Kuiper. IEEE Access. **9**, 141611–141625 (2021). https://doi.org/10.1109/ACCESS.2021.3119634
27. Wei, Y., Li, H., Du, X.: An efficient LEO global navigation constellation design based on walker constellation. In: 2020 IEEE Computing, Communications and IoT Applications (ComComAp), pp. 1–6. Institute of Electrical and Electronics Engineers Inc. (2020). https://doi.org/10.1109/ComComAp51192.2020.9398888

28. Miller, K. et al.: Optimal runtime assurance via reinforcement learning. In: ACM/IEEE 15th International Conference on Cyber-Physical Systems (ICCPS), pp. 67–76. Institute of Electrical and Electronics Engineers Inc. (2024). https://doi.org/10.1109/ICCPS61052.2024.00013