International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

Intelligent Embedded Systems for Next-Generation Robotics Integrating Edge AI, Real-Time Control and Hardware-in-the-Loop Validation

Girish Shashikant Pati[1]
[1]*Independent Researcher Universal Instruments Corporation*
Email id: *mailtogirish.patil@gmail.com*[1]

Abstract
Perception, estimation, planning and feedback control on small, embedded computers are now being done by autonomous robots. These systems must be able to support challenging AI tasks without interference of accelerator contention, memory traffic, operating-system jitter, or network delay with closed-loop control. This paper presents a description of an architecture, which integrates edge artificial intelligence, deterministic control, and hardware-in-the-loop validation. The processing stages have defined deadlines, a limit on data-freshness, a resource budget and a fallback response. A mixed-criticality runtime ensures the high-rate control is not mixed with the best-effort AI workloads, and the slack-aware inference manager increases or decreases the accuracy of the models and their rate of update in the validated operating points. HIL environment is then a layered comparison of deployment in the production with real-time models of the plant, sensors, network, thermal conditions and faults. The article defines the schedulability and end-to-end latency requirements, sets quantifiable validation requirements, and uses the methodology to a study of a mobile manipulator. Analytical workload, real-time isolation also brings the modeled-miss ratio to 0.42%; adaptive inference brings it to 0.07% and the average power consumption decreases to 18.7 W instead of 23.5 W. The paper shows how timing contracts, runtime enforcement and HIL evidence can be used to allow predictable normal and degraded operation prior to field trials.

Keywords: *Edge AI, embedded robotics, hard real-time control, mixed-criticality systems, hardware-in-the-loop, digital twin, ROS 2, deterministic networking.*

1. Introduction
The intelligence used in robots is gradually shifting from being based at remote workstations to being incorporated directly onto the robots themselves. Currently, cameras, lidars, tactile arrays, microphones, and proprioceptive sensors are providing data to perception and estimation models which are running on embedded CPUs, GPUs, neural processing units, and field-programmable gate arrays. By carrying out the inference locally, a necessary trip to the cloud is avoided, the amount of sensor data that is exposed is reduced, and operation can go on even if the connection fails. Nevertheless, strict requirements still apply with regard to control timing: the motor-current loops can operate at tens of kilohertz, joint control at nearly one kilohertz, whole-body control at hundreds of hertz, and perception or planning at tens of hertz. If the result of an image classification is delayed this could lower the quality of recognition; if a torque command is sent late then unwanted energy might be introduced into a physical interaction. For these reasons the system has to treat AI tasks that are performance-oriented differently from those that are safety-critical. The main problem facing the system is resource interference. Although Edge AI enhances semantic perception, it has to share caches, memory bandwidth, thermal headroom, DMA channels, and communication links with the control software. The time taken to perform inference depends on the input shape, the level of sparsity, the control flow, the kernel selected, and the degree of accelerator occupancy. For real-time control, on the other hand, what is required are bounded response times, up-to-date state estimates, and predictable actuation. An average frame rate can mask rare yet significant latency spikes, and classical timing analysis can become unreliable if the shared accelerator is throttled or if the neural model changes after the system has been deployed. HITL

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

testing offers a parameterized method of changing test mode to software simulation and real field test. The production controller sends signals at timed points with timestamps to a real-time model of the plant and the bench injects some standard network delay, packet loss, sensor faults, actuator saturation, compute overload, and thermal derating. HIL is most persuasive with the use of the same timing and failure conditions as with the deployed runtime. It is not sufficient to reproduce normal kinematics and the test is not engineered to encompass clock alignment and queueing, resource contention and recovery behavior.

1.1.Research Questions

The study investigates how an embedded robot should divide up the tasks of AI, estimation, and planning in order to remain useful when overloaded. What timing and resource agreements link inference quality to closed-loop stability, as well as how HIL should produce evidence with regard to algorithms, hardware, middleware, and faults. The architecture suggested deals with these questions by incorporating all adaptive functions within a safety envelope that remains unchanged and by assigning each important contract a measurable verdict in the hardware-in-the-loop testing.

1.2.Scope

The system is applicable to mobile robots, manipulators, drones, and other autonomous machines that have diverse onboard computing resources. Rather than dealing with the training of foundation models on the robot, it focuses on inference and presumes the existence of an independently verified safety channel which can issue a controlled stop command. Individual products still require their own mechanical design, formal functional-safety certification, and penetration testing. The contribution consists of a reusable systems and validation framework and it does not rely on any one particular middleware or accelerator.

2. Technical Background and Related Work

2.1.Edge Intelligence in Robotics

By carrying out computation near the source of the data, edge computing decreases the need for distant infrastructure and allows for the delivery of low-latency, location-aware services [1], [2]. In the case of robotics, the edge can refer to the onboard computer, a cell controller, or a nearby server; execution on board provides the highest level of resilience against network failures. The use of compact architectures, pruning, quantization, and accelerator-aware mapping has lowered the arithmetic and memory requirements of deep learning [3]-[6]. MobileNet uses depthwise separable convolutions to run vision tasks in resource-constrained settings [4], and integer-only quantization can be used to efficiently run on fixed-point devices [5]. Although these techniques have enhanced the common latency, the appropriateness of the techniques in real-time application is also dependent on the tail latency and the interference occasioned by other concurrent workloads. Modern embedded modules include multicore Arm processors, GPUs, dedicated deep-learning accelerators, video engines, and high-speed I/O. For instance, the Jetson AGX Orin family claims a maximum of 275 sparse INT8 TOPS across its various power modes [7]. However, achieving peak throughput does not ensure that deadlines will be met, since memory bandwidth, tensor-layout conversion, CPU preprocessing, synchronization, and thermal limits can all have a dominant effect on end-to-end latency. It is therefore necessary to evaluate the entire sensor-to-actuator path when selecting a platform by subjecting it to representative concurrent loads.

2.2.Real-Time Robotics Software

ROS 2 has replaced the centralized communication model of ROS 1 with a middleware abstraction which is usually based on DDS, allowing for configurable settings relating to reliability, history, deadlines, and liveliness [8], [9]. DDS was developed for data-centric, real-time publish-subscribe communication in embedded systems [10]. Although these features are necessary they are not enough since a reliable topic can still come too late and a standard deadline policy does not place a limit on the execution of user code. The real-time programming advice for ROS 2 therefore stresses the importance of avoiding page faults, dynamic allocation, and indefinitely blocking synchronization in the critical path [11].

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

PREEMPT_RT achieves preemption in most of the Linux kernel paths by means of threaded interrupts and locks that are aware of priority inheritance [12]. Isolation, CPU affinity, memory locking, bounded drivers, and stress measurements still remain essential.

2.3.Hardware-in-the-Loop Validation

The HIL system incorporates at least one production hardware or software component into a closed-loop arrangement with a real-time simulation; in the case of robotics this has included the use of ROS and embedded NVIDIA computers [13], agricultural path tracking [14], exoskeleton control [15], network HIL for the control of wireless robots [16], and digital-twin-based nonlinear tracking [17]. These studies show that it is possible to carry out safe and repeatable evaluations of controllers before carrying out expensive physical tests. Yet AI latency, scheduling, network behaviour, plant dynamics, and safety fallback mechanisms are still generally evaluated separately. The framework that has been proposed links these aspects together in a single traceable chain of evidence.

2.4.Research Gap

There are three related gaps in the literature. In studies involving Edge-AI, accuracy, throughput, or energy are often optimized without taking into account a freshness limit that is based on control requirements. Control studies usually assume that computation insvolves a fixed delay. When it comes to HIL studies, the controller may be verified while ignoring both shared-accelerator contention and adaptive model switching. A common agreement is required in order to link each hazard and control objective to a scheduler rule, a middleware configuration, a fault-injection test, and the release criterion.

Table 1 Integration gap addressed by the proposed architecture

AI	Average latency and accuracy	Tail latency, freshness, energy, confidence, fallback
Control	Fixed sampling and delay	Measured jitter, age of information, mode switching
Platform	Peak compute capability	Contention, throttling, priority, bounded I/O
Validation	Nominal plant response	Timing, network, thermal, sensor and actuator faults

3. System Model and Design Requirements

3.1.Workload Model

Let the robot carry out a series of periodic or sporadic tasks $\tau_i = (C_i, T_i, D_i, J_i, \kappa_i)$, where C_i is a validated upper bound on the execution time, T_i is the minimum inter-arrival period, D_i is the relative deadline, J_i is the release jitter, and κ_i is the criticality class. The A class includes functions whose delay may breach the safety envelope, for example motor control and the independent supervisor. The B class includes functions which affect mission performance, such as state estimation and local planning. The C class includes best-effort mapping, logging, diagnostics, and model updates. Accelerator jobs include a non-preemptive or coarsely preemptive section, and

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

therefore their blocking time must be specified explicitly.

$$R_i = C_i + B_i + \sum_{j \in \mathrm{hp}(i)} \left\lceil \frac{R_i + J_j}{T_j} \right\rceil C_j$$

Equation (1) represents the conventional response-time recurrence formula involving blocking B_i and higher-priority interference hp(i) [18]. It is only useful in cases where both C and B constrain the binary that has been deployed on the deployment platform. In the context of heterogeneous computing, the analysis is carried out on a per-resource basis and then combined with the transfer and synchronization bounds. When an accelerator is unable to preempt a long model, the system will either limit the duration of the kernel, reserve the device, or relocate the safety-critical path.

3.2.End-to-End Freshness

The control action is dependent on the age of an observation. The age of information A = t o t n is the difference between the actuator-commit time t o and sensor timestamp t n and thus includes sampling phase, the transport, the queueing, the inference, the estimation, the planning, the control computation and output transfer. Every control mode has a maximum age Amax, when the data exceeds this value, the consumer must either extrapolate within a validated horizon, reuse a validated estimate, slow down, or enter into a safe state but should not consider stale data to be current.

$$L_{\mathrm{e2e}} = L_{\mathrm{sample}} + L_{\mathrm{transfer}} + L_{\mathrm{AI}} + L_{\mathrm{est}} + L_{\mathrm{plan}} + L_{\mathrm{ctrl}} + L_{\mathrm{output}} \leq D_{\mathrm{e2e}}$$

3.3.Control Requirement

When a discrete plant is given by x 0 k 1 = f (x 0, u 0, w 0) a delayed or stale estimate would cause the value of the input actually given to the plant to be changed. The design therefore considers the delays of bounded computation and communication as being a part of the uncertainty of the plant. Delay-dependent stability analysis or a robust invariant set can be employed in a linearized mode; a control-barrier condition can be imposed in a nonlinear mode. The safety supervisor analyzes the state, magnitude of the command, command rate, level of confidence and age of data before performing an action.

3.4.Nonfunctional Requirements

As shown in Table 2 Requirements and observable acceptance evidence, Figure 1 Layered architecture with contracts propagated across intelligence control platform and validation.

Table 2 Requirements and observable acceptance evidence

R1	Control deadline	99.999th percentile and maximum response	Application-defined; no unhandled miss
R2	Observation freshness	Sensor-to-actuator age	Below mode-specific Amax
R3	Isolation	Added control jitter under AI stress	Within allocated interference budget
R4	Graceful degradation	Time to enter fallback	Before safety-envelope violation
R5	Energy and thermal	Power, temperature, throttle residency	Within sustained mission envelope
R6	Evidence integrity	Clock error and trace loss	Bounded and reported

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

4. Proposed Contract-Driven Architecture

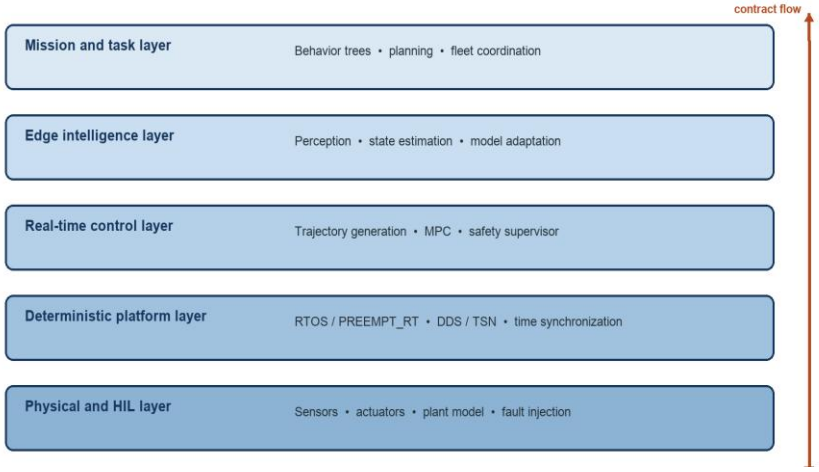

Figure 1 Layered architecture with contracts propagated across intelligence control platform and validation

4.1.Layer Responsibilities

The mission layer is in charge of setting the goals and determining the behaviours, but it cannot circumvent the safety supervisor. The edge-intelligence layer transforms the high-bandwidth observations into detections, semantic features, occupancy information, contact estimates, or uncertainty. The real-time control layer takes these estimates and converts them into bounded trajectories and commands for the actuators. The deterministic platform layer offers the necessary scheduling, memory, device, clock, and network mechanisms. In the physical and HIL layer, the actual sensors and actuators are used when the system is in operation and real-time models that can be replaced are used during verification. Each interface includes data together with a contract header, the header containing the source timestamp, the sequence number, the coordinate frame, the validity horizon, the confidence or covariance, the producer mode, and the integrity status. When a consumer encounters data that is incompatible or has expired, it rejects it. This avoids a typical failure in which the pipeline seems to be working but the age of the observations in the queue keeps on increasing. The contracts also specify a fallback; for instance, a delayed semantic map can cause discretionary navigation to be frozen, while a stale joint position results in an immediate controlled stop. As shown in Figure 2

Example latency budget for a perception to actuation chain.

4.2.Assurance Invariants

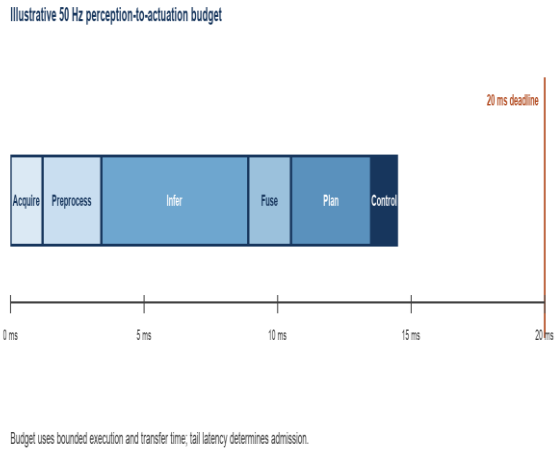

Figure 2 Example latency budget for a perception to actuation chain

There are four invariants that govern adaptation. The first is that the safety supervisor operates using reserved compute and communication resources. The second is that no AI output causes an actuator to move unless it has passed range, rate, freshness,

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

and feasibility checks. The third one states that the alterations to the model or accuracy occur at defined mode limits or due to a limited two-buffered exchange. The fourth is that the system logs enough evidence at the time the command was issued to be able to reassemble all the deadlines, fallback and rejected commands. These invariants decrease the state space that the HIL needs to traverse and have a deterministic core and probabilistic perception.

4.3.Deployment Partitions

At least three partitions are employed when it comes to its implementation. The watchdogs, the emergency-stop logic, the limit checking and a minimal state observer are all included in the safety island. The fieldbus I/O, high-rate estimation, trajectory interpolation and feedback are performed by the real-time partition. The camera preprocessing, inference, mapping and long-horizon planning is done by the AI partition. The different partitions may be assigned to different cores, priorities, memory pools, DMA policies, and bandwidth limits although they may be co-located in the same system-on-chip. In case there is a need to have a higher level of assurance, the safety island can be situated on another microcontroller.

5. Edge AI Pipeline and Adaptive Inference

5.1.Model Selection

The process of selecting a model starts with considering the task loss that has an impact on control, and not just on leaderboard accuracy. When it comes to obstacle avoidance, the failure to detect small objects close to the stopping distance might be more important than the overall mean average precision. In the case of grasping, uncertainty regarding the pose and temporal consistency could be more significant than class accuracy. The candidate models have to be evaluated using the full preprocessing and postprocessing pipeline, together with representative sensor resolutions, the simultaneous control load, the sustained temperature, and the intended numerical precision. The deployment set should contain a cheaper model or an alternative that is not neural and which maintains a specified minimum level of capability.

5.2.Compression and Hardware Mapping

Since quantization leads to a reduction in storage space, memory traffic, and arithmetic cost when it is supported by the target system, post-training quantization is a cheap option but may cause a deterioration of performance in sensitive layers; quantization-aware training generally maintains better accuracy because it simulates the effects of low precision throughout the training process [5]. Structured pruning is more easily exploited by general accelerators than unstructured sparsity. Operator fusion helps to decrease the amount of memory traffic for intermediate results. However, these optimizations have to be checked end to end since the conversion process, the presence of unsupported operators, and CPU fallbacks can nullify the theoretical benefits. Different operator and precision limitations are available through ONNX Runtime, TensorRT, TensorFlow Lite, and the vendor compilers [19], [20].

5.3.Slack-Aware Inference

The inference manager estimates the available slack using the following control deadline, accelerator queue, thermal headroom and the current mission mode. It selects a model m, a precision q, an input scale s and update period TAI based on a finite set, which has been verified. The policy never constructs a tested configuration which has not been tested. The oscillation between the modes is prevented by a timer and the minimum dwell time is required to allow the filters to stabilize once a change in the input resolution or model takes place.

$$(m^*, q^*, s^*, T_{AI}^*) = \arg\max_{m,q,s,T_{AI}} U(Q, \text{confidence}, \text{freshness}) \quad \text{subject to} \quad \hat{L}_{99.9} + \hat{I} \le S, \quad P \le P_{max} \qquad (3)$$

U in formula (3) represents a mission utility, L99.9 is a conservative estimate of the mission latency with the chosen configuration, Î is the interference and transfer allowance, S is the available slack and Pmax is the current power ceiling; a safety margin is added to offset an error in estimation. An explicit unavailable state is declared by the manager and the controller goes into the degraded state described by the unavailable state.

5.4.Confidence and Distribution Shift

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

Confidence and timing have to do with various categories of risk; a high-confidence output may be out of date whereas a timely output may be wrong in case of a distribution shift. This is the reason why the interface contains different fields that represent the indicators of uncertainty, temporal consistency, sensor health, and age. HIL scenarios should consist of scenarios with illumination variations, blur, partial obstruction, calibration drift, sensor drop and problematic geometry. The controller merely takes the reported uncertainty after performing calibration and consistency checks.

6. Real-Time Estimation and Control

6.1.Multirate Control Structure

A single loop rate cannot be efficiently used in the case of heterogeneous robotics. The proposed system is a fast inner loop with a medium-rate loop to estimate the state and generate the trajectory with a slower semantic planning loop. The fast loop reads a finite size of state data in shared memory, reads a time indexed reference, imposes command constraints and sends the results to the fieldbus; it does not wait until neural inference is finished. The estimator accepts measurements asynchronously along with their time stamps and drops all the measurements that are outside the validity horizon of the estimator. The planner gives the trajectories some reasonable look-ahead that ensures that an update is not missed and, as a result, does not hog the command buffer.

$$u_k = \mathrm{sat}_{\mathcal{U}}\{u_{\mathrm{ff},k} + K(\hat{x}_k - x_{\mathrm{ref},k})\} \qquad (4)$$

Equation (4) describes a bounded feedback controller which has a feedforward input uff, a gain K, an estimated state x̂, a reference state xref, and an admissible command set U. It can be replaced by more advanced MPC or whole-body control, on the condition that both solver termination and constraint behaviour remain bounded. If an anytime solver is used, it will return the best feasible iterate by the deadline; if a solution is found to be infeasible, then a precomputed braking or hold trajectory is activated.

6.2.Freshness-Aware State Use

Each state sample is linked with the time at which it was acquired and with the covariance. When the age of the data is still within the prediction horizon, the estimator uses the motion model to propagate the state to the control moment. However, once it goes beyond that horizon, the uncertainty increases and the controller decreases its speed or authority. This is better than carrying out an unconditional last-value hold, since such a procedure would conceal the loss of observability. Moreover, the fact that the data is fresh also affects the collision margins in that greater latency means a longer required stopping distance.

$$u_k = \mathrm{sat}_{\mathcal{U}}\{u_{\mathrm{ff},k} + K(\hat{x}_k - x_{\mathrm{ref},k})\} \qquad (5$$

Equation (5) makes this connection clear. When the speed is v, the perception-to-actuation age is A, the stop-command latency is Tstop, the guaranteed deceleration is amin, and the uncertainty margin is duncertainty, the planner must keep at least dsafe clearance. Although the precise formula will depend on the system in question, the general principle is that the time delay uses up the physical safety margin.

6.3.Policy on overrunning and Watchdog

A missed deadline will cause a given transition of state rather than simply bringing about a log entry. The system is able to drop one late and noncritical frame. When the estimator is slow on many occasions, the speed is slowed down and separation distance is increased. In case of an overrun of the motor-control, the independent safety channel assumes control and substitutes the command by a limited hold or a controlled halt. Watchdogs maintain a vigil on liveness and semantic progress as a task can remain alive even when the task is repeatedly reporting stale data.

6.4.Stability Across Modes

Adaptive inference will lead to different measurement quality and latency. During the offline stage, all the combinations of the model, rate, speed envelope and controller gain that are possible are tested. The switching is carried out with hysteresis and dwell-time constraints when switching or a common Lyapunov certificate is utilized when possible. As shown in Figure 3 Example partitioned schedule that protects the motor control path , Table 3 Example multirate

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

workload and explicit late-data behavior

Table 3 Example multirate workload and explicit late-data behavior

Motor current	10–40 kHz	A	Hardware trip or local hold
Joint or vehicle control	500 Hz–2 kHz	A	Controlled stop
State estimation	100–500 Hz	A/B	Predict, inflate covariance, slow
Perception	10–60 Hz	B	Drop frame or select smaller model
Global planning	1–10 Hz	B/C	Reuse valid plan; request replan

7. Deterministic Runtime and Resource Isolation

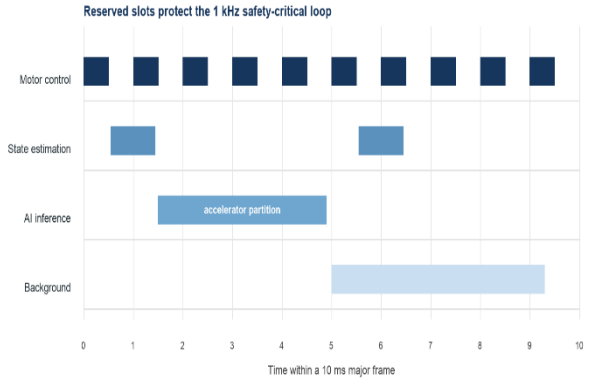

Figure 3 Example partitioned schedule that protects the motor control path

7.1.CPU Scheduling

Class A threads are pinned to separate cores, are given fixed priorities based on verified dependencies, have their memory locked, their stacks are pre-faulted, and no dynamic allocation takes place after the system has been initialized. High-volume camera and storage interrupts are kept away from the control cores by means of interrupt affinities. Although PREEMPT_RT reduces the number of non-preemptible sections in the kernel, firmware, buses, device drivers, system-management interrupts, and shared caches can still cause outliers [12], [21]. Cyclic latency tests therefore serve only as a preliminary check of the platform, not as proof relating to the application path.

7.2.Accelerator Scheduling

Mixed-criticality scheduling is more challenging with the use of GPUs as kernels can be non-preemptive, vendor runtimes might lazily allocate memory, and changing frequency can change the execution time. Prior to enabling motion, the deployment performs a warm-up process, fixed tensor shapes, buffers preallocation and a restriction of the number of simultaneous contexts. In the absence of accelerator partitioning, an admission controller prevents long best-effort kernels to be started near a critical inference window. Small safety-critical networks can alternatively execute on a dedicated DLA, FPGA, DSP or CPU as opposed to the shared GPU.

7.3.Policy on overrunning and Watchdog

Failure to meet a deadline results in a specified state transition instead of merely leading to a log entry. The system can discard a single late and noncritical frame. If the estimator is late repeatedly, the speed is reduced and the separation distance is increased. When a motor-control overrun occurs, the independent safety channel takes over and replaces the command with a bounded hold or a controlled stop. Watchdogs keep an eye on liveness and semantic progress since a task may still be considered alive even if it is repeatedly issuing stale data.

7.4.Thermal and Power Management

Embedded accelerators are usually placed in situations where they are operating close to their maximum power and thermal limits. Since dynamic voltage and frequency scaling can

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

increase average efficiency it is no longer valid to assume a fixed execution time. As a result, the production modes associate a known power profile with each workload that has been validated. Thermal telemetry is passed to the mode manager before any throttling takes place, which enables the resolution or inference rate to be reduced while the control resources stay the same. The qualification process involves cold start, hot soak, fan degradation, and supply droop. As shown in Table 4 Isolation mechanisms and the tests that challenge them.

Table 4 Isolation mechanisms and the tests that challenge the

CPU affinity and RT priority	Scheduler interference	SMI, shared cache, driver stalls	CPU stress plus interrupt storm
Locked preallocated memory	Page faults and allocator delay	Bus and cache contention	Memory-bandwidth saturation
Accelerator admission	Long kernel blocking	Opaque firmware queues	Concurrent model burst
Bandwidth and DMA policy	I/O starvation	Device-specific arbitration	Camera plus logging overload
Fixed power mode	Frequency variability	Thermal throttling	Hot-soak temperature ramp

8. Communication and Time Synchronization

8.1.Middleware Contracts

DDS provides reliability, history depth, deadline handling, lifespan, and ownership policies that correspond naturally to the data flows in robotic systems [10]. When selecting a policy one must take account of the semantics involved; in the case of high-rate camera or pose updates, best-effort delivery with a depth of one is usually suitable since retransmitting an old sample would increase its age. Reliable delivery should be used for configuration and mode commands, provided that the queue depth and the retry behaviour are limited. Deadline and liveliness events are fed to the system monitor, but the applications themselves still check the source timestamps and the sequence continuity.

8.2.Deterministic Networking

EtherCAT, CAN FD, or time-sensitive Ethernet can be used to connect a distributed robot using motor drives, sensors, safety controllers and compute modules. IEEE 802.1 time-sensitive networking profiles incorporate clock synchronization, scheduled traffic, shaping and redundancy features, to regular Ethernet. The emerging DDS-TSN mapping defines how topic flows can be associated with scheduled TSN streams [22]. Determinism depends on an engineered end-to-end path, including switches, queues, endpoint Latency measurements can only be regarded as credible if the clocks involved are synchronized; for this reason, both the HIL bench and the robot nodes make use of a common PTP grandmaster or a characterized hardware trigger. In each test, clock offset, drift, synchronization state, and timestamp position are recorded. The software receive timestamps take into account both driver and queueing delay, while hardware timestamps taken near the media-access layer offer more solid evidence regarding the network. When a clock goes into holdover, the acquisition system either increases the uncertainty bound or declares the run invalid.rk evIn equation (6), εsync is used to bound the relative clock error. The pass/fail limits should include a margin to allow for this uncertainty. It is not credible to claim an improvement of 200 µs when the clock error is ±500 µs.hould reserve margin for thisThe controller has been tested with respect to latency, jitter, loss, duplication, reordering, and burst patterns. It is important to consider independent variations since different closed-loop behaviours can result even when the mean delay is the same. Network HIL includes a

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

controllable impairment bridge between the production endpoints and at the same time keeps the production serialization and middleware stack [16]. Each of the injected conditions is given a seed, a timing trace, and an expected mode transition.ialization and middleware stack [16]. Each injected condition has a seed, timing trace, and expected mode transition. As shown in Table 5 Example communication policies tied to control semantics.

Table 5 Example communication policies tied to control semantics

Joint feedback	Reliable cyclic fieldbus	Fixed slot; bounded retry	Stop on missing cycle
Odometry or pose	Best effort depth 1	Overwrite old sample	Predict within horizon
Camera frames	Best effort bounded pool	Drop newest or oldest by design	Report frame age
Mode command	Reliable transient local	Small bounded queue	Acknowledge state transition
Safety stop	Independent safety channel	Highest priority	Fail-safe on channel loss
Logs and maps	Reliable best effort priority	Backpressure outside RT path	Defer or discard

9. Hardware-in-the-Loop Validation Architecture

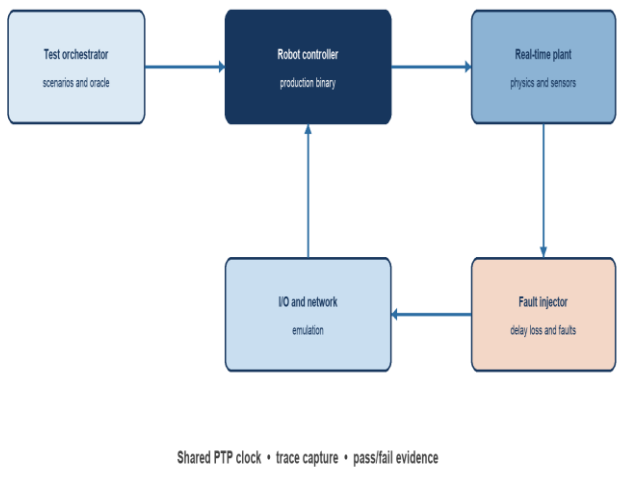

Figure 4 The item being tested consists of the production compute module

Figure 4 The item being tested consists of the production compute module, its operating-system image, the middleware configuration, the inference engine, and the control binary. A different real-time target carries out the integration of plant dynamics at fixed time steps and generates the sensor outputs. Physical I/O or protocol interfaces mimic the behaviour of ADCs, encoders, PWM signals, CAN, EtherCAT, or Ethernet. The orchestrator is responsible for setting up the scenarios, verifying the preconditions, injecting faults, and checking the assertions. Trace collection makes use of a common clock and records sufficient information so that a failure can either be replayed or diagnosed. The plant model has to finish each integration step by the wall-clock deadline, and the level of fidelity is assigned according to its impact on the requirement—such as actuator saturation and delay in the case of tracking, contact stiffness when it comes to manipulation, tire slip for mobile robots, aerodynamic disturbances for drones, and sensor sampling, noise, and dropout for estimation. Detailed geometry is of no benefit if it leads to delays. Co-simulation allows the fast actuator models to be separated from the slower environments, but when crossings occur deterministic exchange and documented

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

extrapolation are required. actuator models from slower The model is assessed using component tests or physical data within its intended operating range. Validation reports include the parameter ranges, the residual error, the unmodelled effects, and the extrapolation limits. HIL results provide evidence regarding the controller-plus-model system, rather than serving as automatic proof of performance in the field. The test plan keeps physical tests for those effects which cannot be properly modelled, and uses HIL to repeatedly examine dangerous, rare, and combinatorial conditions that cannot be modeled credibly, while using HIL to cover dangerous, rare, and combinatorial conditions repeatedly.

9.1.Test Oracles

Each trace is turned into a verdict by the test oracle. Timing claims are deadlines, data age and clock quality. Some of the control assertions are tracking error, constraint violation and energy. To be on the safe side, there are statements regarding detection time and rejecting unsafe commands. The AI claims consider the loss of tasks, confidence behavior, time stability, and the distribution shift. An invalid run is one where evidence required is lost.

10. Validation Strategy and Test Matrix

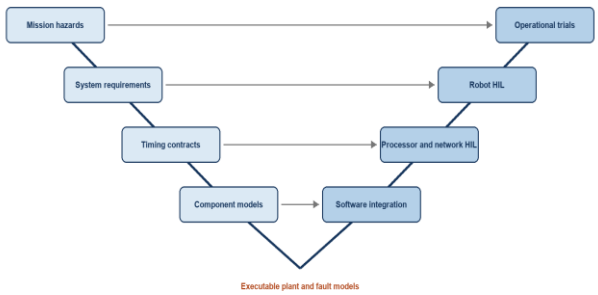

Figure 5 Traceability from requirements to progressively higher-fidelity validation

10.1. Staged Campaign

Validation starts with testing of the deterministic components and then moves on to software-in-the-loop, processor-in-the-loop, network HIL, full robot HIL, and finally to supervised physical trials. The same scenario schema and assertions are used at all the stages. In the early stages the execution is faster than real time in order to carry out an extensive parameter search, whereas in the later stages wall-clock timing and the production interfaces are maintained. If a high-level test fails, it is narrowed down to the lowest stage at which the failure occurs, thus reducing the time taken for diagnosis and avoiding the HIL bench from becoming an opaque integration endpoint.

10.2. Coverage Model

This is not testable completely since the space is too large because of a myriad of factors such as continuous states, environmental conditions, model versions and fault timing. The campaign employs equivalence classes, boundary values, and pairwise combinations, Monte Carlo seeds, and requirement-based adversarial search. The number of repetitions of the tests depends on the risk. The assertions of tail latency must have sufficient data to support the percentile given. Several minutes of normal functioning will not demonstrate the frequency with which there are rare misses. As shown in Table 6 Core HIL campaign matrix.

Table 6 Core HIL campaign matrix

Nominal dynamics	Speed, payload, friction, curvature	Tracking and deadlines	State, commands, timing trace
Compute interference	CPU, GPU, memory, storage stress	Class A isolation	Per-task response and counters

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

Network impairment	Delay, jitter, loss, reorder, burst	Freshness and fallback	Hardware timestamps, QoS events
Sensor faults	Bias, freeze, dropout, calibration error	Detection and degraded mode	Residuals, health state
Actuator faults	Saturation, lag, stuck output	Constraint handling and stop	Plant energy, command limits
AI shift	Lighting, blur, occlusion, novel objects	Uncertainty and safe behavior	Task loss, confidence, trajectory
Thermal and power	Hot soak, fan loss, power cap	No hidden deadline violation	Temperature, clocks, power
Recovery	Restart, clock loss, model reload	Bounded and nonoscillatory recovery	Mode timeline and logs

10.3. Statistical Reporting

The reports include the median, the higher percentiles, the maximum value observed, the number of samples, the run duration, the confidence interval where it is relevant, and the exact workload; each percentile is accompanied by a deadline-miss ratio and a burst analysis. Control error is divided up by mode and fault rather than being averaged over the easy and difficult sections. Each plot indicates whether the values are measured, simulated or analytically generated.

11. Illustrative Mobile-Manipulator Design Study

11.1. Purpose and Reproducibility

The framework described here is that which is used prior to a physical bench being available; the analytical study involves a differential-drive mobile base fitted with a six-axis collaborative arm. The results are based on the workload assumptions given in Table 7 and use a fixed random seed. They show the method of calculation, the structure of the experiment, and the decision logic that will be employed in a later HIL campaign; they do not consist of hardware measurements.

11.2. Vision and Goals

The robot can relocate parts, identify people and obstacles with stereo cameras and lidar, localize itself in a pre-mapped facility and perform guarded pick-and-place operations. The system has a heterogeneous embedded module having six CPU cores which are availed to the application, one GPU and one dedicated inference accelerator. A microcontroller will be needed to complete the loops of the motor-current, and to offer an independent stop feature. The central computer runs a command supervisor with 1 kHz, an estimator with 200 Hz, a local planner with 50 Hz, a nominal detector with 30 Hz and does its best to perform mapping and logging.

11.3. Analytical Workload

The samples of the executions are drawn out of truncated log-normal distributions which are selected to achieve the specified median and P99. A shared memory-contention term is a part of a correlated interference component. In case of thermal throttling, it causes the AI and planning execution times to rise once the temperature has surpassed the temperature threshold. The baseline (which is not controlled) allows sharing CPU cores and launching the GPUs freely. In the isolated setup, there is a reserved two CPU cores as well as limited accelerator admissions. The adaptive configuration uses the inference policy of (3), and chooses to either use nominal INT8, reduced-resolution INT8 or a lightweight obstacle network.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

Table 7 Declared workload assumptions for the illustrative study

Command supervisor	1 ms	0.18 ms	0.34 ms	reserved CPU	0.8 ms
State estimator	5 ms	1.35 ms	2.40 ms	reserved CPU	4.0 ms
Local planner	20 ms	4.8 ms	8.6 ms	CPU	16 ms
Nominal detector	33.3 ms	10.8 ms	16.2 ms	GPU	20 ms
Reduced detector	50 ms	6.4 ms	9.0 ms	GPU/DLA	14 ms
Mapping and logging	sporadic	6.0 ms	14.0 ms	best effort	none

11.4. Acceptance Criteria

- In the 10^7 modeled releases there was no instance of an unhandled command-supervisor deadline miss; whenever a miss was detected it had to trigger the independent stop channel.
- For at least 99.9% of the eligible updates, the perception-to-command age was under 20 ms in nominal mode and under 35 ms in reduced-speed mode.
- The lateral path error was less than 0.12 m for 95% of the samples, the highest error was below 0.25 m, and there was no penetration of the collision envelope.
- Fallback is initiated within 40 ms of a violation of the persistent data age and does not return to nominal mode until five seconds of healthy operation have been achieved.

12. Illustrative Results

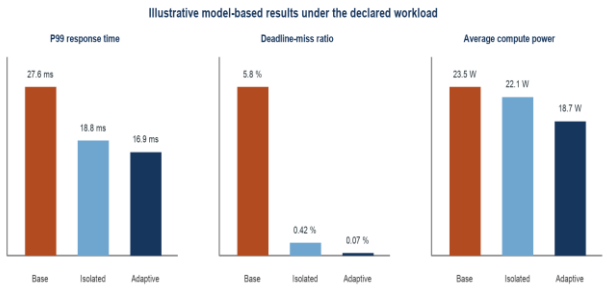

Figure 6 Model-based timing and power comparison; values are illustrative rather than measured

12.1. Timing

This uncontrolled setup would give a modelled P99 sensor-to-command response time of 27.6 ms and would not respond within the 20 ms nominal deadline in 5.8% of the available perception cycles. Most of the missed cases are as a result of simultaneous bursts by the detector, the planner and the logging system. CPU isolation and accelerator admission reduces the P99 response time to 18.8 ms and the miss ratio is reduced to 0.42%. There are some misses that still occur during the modelled thermal transition, when the accelerator frequency varies before the workload manager can react. The adaptive configuration selects the lower detector as slack becomes smaller or the thermal stress becomes higher. The P99 response as modeled is reduced to 16.9 ms and the deadline-miss ratio is decreased to 0.07%. This tradeoff however does has a definite cost: the detector fidelity is decreased and the robot operates at the lower speed limit 8.4% of the time in the scenario. Therefore, the system shows a loss of capability as opposed to maintaining the nominal speed at stale perception.

12.2. Control Behavior

The median tracking target is achieved in the nominal dynamics sweep in the three configurations because the average computation suffices. The change is reflected in jolts of disruption. The maximum lateral error, in the uncontrolled situation, is 0.31 m, more than the upper limit of 0.25 m. The maximum with isolation is 0.22 m and with adaptive inference it is 0.19 m since the mode manager decreases the reference

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

speed prior to freshness exceeding the degraded-mode limit

12.3. Energy and Thermal Behaviour

The average computed power drops from 23.5 W in the unmanaged situation to 22.1 W when isolation is used and to 18.7 W with adaptive inference. Since isolation only alters the scheduling without changing the amount of work, it does not lead to much energy saving. Energy is saved through adaptation by reducing the input resolution and the update rate when the robot is stationary, thermally constrained, or is already working at a reduced speed. This shows that power management should be connected to the mission and timing state rather than being considered a separate controller. As shown in Table 8 Summary of analytically generated design-study results.

Table 8 Summary of analytically generated design-study results

P99 response	27.6 ms	18.8 ms	16.9 ms	< 20 ms nominal
Deadline misses	5.8%	0.42%	0.07%	Report; fallback on persistence
Maximum path error	0.31 m	0.22 m	0.19 m	< 0.25 m
Average compute power	23.5 W	22.1 W	18.7 W	< 25 W sustained
Reduced-mode residency	0%	0%	8.4%	Mission-dependent

13. Fault Injection and Sensitivity Analysis

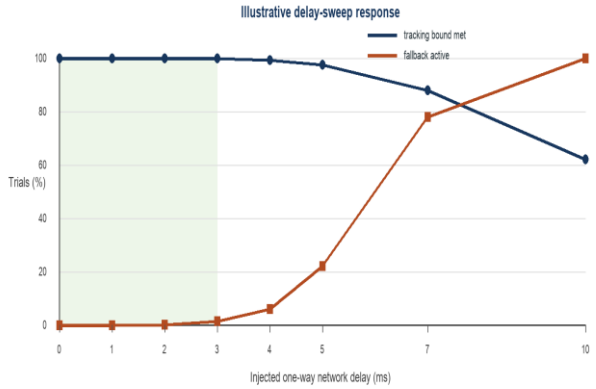

Figure 7 Illustrative response to injected network delay with a 3 ms qualified region

13.1. Delay and Loss

While the delay sweep keeps the plant dynamics and the workload unchanged and at the same time increases the one-way delay of the estimator network, the design achieves its tracking bound in all the trials modelled up to 2 ms and in 99.8% of the trials at 3 ms; beyond the qualified area freshness violations build up and the safety fallback triggers more frequently. The controller fails to maintain the nominal motion by making use of a delay margin which had never been assigned. Since packet-loss bursts cause a more abrupt transition than independent losses having the same average rate, it is because the bursts use up the prediction horizon.

13.2. Compute Overload

Adding high-priority interference constrained to the non-reserved cores, and contention for memory bandwidth across the system, are both sources of CPU stress. Best-effort kernels that are bursty cause overload on the GPU. Due to the isolated design, the changes in CPU utilisation can be tolerated until shared-memory traffic overwhelms it, demonstrating that core affinity alone is inadequate. It is important to note that even though accelerator admission protects critical inference at the cost of starving mapping; hence the use of monitoring policy logs the denial of resources and restricts the backlog rather than retaining old jobs.

13.3. Sensor and Model Faults

All the three configurations are able to reach the median tracking target in the nominal dynamics sweep because average computation is adequate. The disparity manifests itself in times of

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

disturbance bursts. The maximum lateral error in the uncontrolled scenario is 0.31 m (greater than the upper limit of 0.25 m) in clusters of the stale estimates. Isolation the maximum is only 0.22 m and adaptive inference is only 0.19 m since mode manager decreases the reference speed when freshness exceeds the limit of degraded mode. Model faults occur when the model of the system is not consistent with what is occurring. This discrepancy may be caused by the lack of information, misplaced assumptions or alterations in the system that are not reflected in the model. Consequently, the model-based predictions and control actions can be unreliable. These problems can be detected early by updating and checking them on a regular basis. Both sensor and model faults can reduce the reliability and safety of the system. Designers need to keep these problems in mind and use checks to reduce their impact. The sequence and content-change monitors pick up a frozen camera stream that has valid transport headers. It is more difficult to detect slow drifting calibration since each frame looks plausible. After a persistence interval that can be configured, the cross-sensor residuals and the map consistency pick out the drift. The test oracle is able to tell the difference between detection latency and mitigation latency. In one scenario, the AI confidence is intentionally miscalibrated in order to verify that the controller still carries out the geometric and freshness checks.

13.4. Thermal Derating

A temperature ramp causes the intended reduction in inference to take place prior to the point at which the hardware is expected to throttle. When telemetry is delayed or missing, frequency throttling results in a latency excursion and the deadline monitor then activates the same degraded mode. The first approach is preventive and ensures that movement remains smooth. The second is reactive and stays safe although it might necessitate a stop. Both of these procedures have to be tested since sensors and management services are capable of failing. As shown in Table 9 Fault detection and evidence expectations.

Table 9 Fault detection and evidence expectations

Stale but valid message	Age and sequence	Reject; predict or slow	No use beyond validity horizon
GPU queue saturation	Queue age and slack	Gate launch; smaller model	Class A/B latency within contract
Clock loss	PTP state and offset	Expand uncertainty; invalidate run	Bounded timestamp error
Thermal throttle	Temperature and frequency	Reduce AI workload	No hidden control miss
Actuator saturation	Command versus feedback	Anti-windup; safe trajectory	Constraint respected

14. Safety, Security, and Assurance

14.1. Safety Case Structure

The system includes a claim-evidence safety case; the main claim is that the robot stays within the safety boundaries that have been defined for each of its operational modes. Various subclaims cover bounded control timing, the availability of valid state information, constrained actuation, safe degradation, and the ability to stop independently. The evidence consists of analytical studies, a code review, unit tests, schedulability results, HIL traces, the results of fault injection, and physical tests. Standards such as IEC 61508, ISO 13849, ISO 10218, and ISO 3691-4 might apply according to the product and environment [23]-[26]. Compliance is dependent on a lifecycle that is specific to the system and cannot be deduced from the architecture alone.

14.2. AI Assurance Boundary

The neural model functions as a performance component within a deterministic wrapper, the wrapper checking the input's health, confirming

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

that execution has completed, verifying the output values, assessing the uncertainty indicators, ensuring spatial feasibility, and checking the age. When it comes to making safety decisions, conservative geometry is used, independent sensing is employed where necessary, and a bounded fallback is provided. The record of model provenance includes the version of the training data, the architecture, the quantization, the compiler, the calibration set, the target hardware, and the hash. An over-the-air model update is a type of configuration change which involves repeating the relevant qualification tests.

14.3. Cybersecurity

While edge execution decreases the amount of data that has to be made available mandatorily it at the same time increases the attack surface available to the robot. The basic security measures include secure boot, the use of signed software and models, the implementation of least-privilege services, encrypted and authenticated management channels, network segmentation, and protected debug interfaces. It is necessary to take real-time availability into account when carrying out a threat analysis since a valid but costly request, a logging flood, or a malformed sensor stream can result in a deadline failure without any change to the control code. Rate limits and bounded parsing provide protection for the critical path.

14.4. HIL Bench Integrity

The test bench is the part of the evidence system and must be configurable. Plant models, scenarios, impairment settings, firmware, controller images, clock settings, and analysis scripts are required to have version identifiers and hashes. The orchestrator makes use of deterministic seeds whenever this is possible and keeps records of the calibration and self-test results prior to each campaign. The result is said to be invalid in case the plant model overruns, the clocks are not synchronized or a needed trace channel cannot provide data.

14.5. Human Interaction

Working together introduces uncertainty regarding perception, ambiguity about intent, and differences in the way contact occurs. Although HIL is able to simulate human motion paths and force profiles, it cannot replicate all types of behaviour or injury mechanisms. Physical testing carried out with suitable safeguards is still necessary. The embedded system has to ensure that the operators can see when the system is operating in a degraded state and must prevent quick mode changes that may be technically safe but are confusing. For recovery to take place, it should be based on stable evidence rather than a single good frame. As shown in Table 10 Connection between assurance concerns controls and evidence.

Table 10 Connection between assurance concerns controls and evidence

Neural misclassification	Independent constraints and safety supervisor	Shift scenarios and unsafe-command rejection
Timing failure	Reserved resources and deadline monitor	Stress HIL with traceable deadline verdicts
Malicious workload	Least privilege and rate limits	Resource-exhaustion security tests
Unsafe update	Signed versioned model and rollback	Regression HIL on target hardware
Bench error	Clock and model self-tests	Calibration record and evidence completeness

15. Implementation Blueprint

15.1. Define Contracts

Begin with a examination of the operational modes, the hazards, and the physical limitations. Determine the loop periods, the end-to-end age limits, the way in which the system behaves when it stops, how it responds to uncertainty, and the minimum level of capability required to keep

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

motion going. Divide each of these limits among the areas of sensing, communication, inference, estimation, planning, and actuation. Find out which measurements verify each of these requirements. This task must come before any decision is made regarding the choice of hardware since it decides whether or not a particular platform can be qualified.

15.2. Research the stack of deployment

Prepare the sensor-to-command production line as quickly as possible. Test each candidate model and controller on the actual hardware, in both cold and hot conditions, including conversions, middleware operations, and memory transfers, and in cases of multiple processes in operation simultaneously. Document the distributions with the worst values that have been seen, and append an engineering margin that is reasonable given the nature of the testing, and your experience with the platform. Remove any unsupported operators and lazy initialization. Make sure that the watchdog and stop paths do not have a single point of failure in common with the AI runtime.

15.3. Configure Isolation

Allocate the criticality classes, cores, priorities, interrupt affinities, memory pools, accelerator queue and network quality of service. Use the tracepoints to track the release, start, completion, publish, receive and actuator commit events. Establish the overload policies in advance of an overload occurring. Make sure that there are limits to all queues and all cases of overflow are predetermined with a policy. Logging should be on a best-effort basis and there should be a small event buffer of protection.

15.4. Put together the HIL evidence chain

Carry out the plant model at a fixed real-time step and then check the accuracy of the model against the available component data. Connect the physical interfaces or use the production protocols. Carry out automated regression tests under normal conditions before introducing any faults. Add one fault dimension at a time, followed by selected combinations according to risk. For each run store the identifiers for the scenario, software, model, hardware, and calibration. Produce both machine-readable verdicts and human-readable summaries from the same trace.

15.5. Release and Monitor

The release candidate successfully completes the entire regression campaign on the target hardware and power mode. The telemetry data collected in the field shows the deadline margins, the amount of time spent in each mode, the rejection of stale data, resource throttling, and safety events, without capturing any unnecessary personal data. Operational monitoring is able to identify workload areas that are not included in the test set. Before any change in control or inference behaviour, new areas become HIL scenarios. As shown in Table 11Practical implementation and release gates.

Table 11 Practical implementation and release gates

Architecture	Timing and safety contracts	Every critical flow has deadline, freshness and fallback
Platform selection	Target profile report	Sustained workload fits with justified margin
Integration	Resource and QoS configuration	Interference tests meet isolation budget
HIL readiness	Qualified plant and calibrated I/O	Real-time step and clock bounds demonstrated
Release	Traceable verification report	All mandatory tests pass; deviations dispositioned

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

15.6. Platform Selection Checklist

- The compute should be sustained, not just at its peak, within the thermal and power limits.
- There is a documented accelerator preemption or a mechanism for enforceable admission.
- Hardware timestamping and deterministic interfaces for sensors and actuators.
- The software will receive long-term support, have a security update path, and feature a reproducible toolchain.
- A standalone watchdog system that follows a safe state path and has verified independence from failure.

16. Discussion

16.1. The Importance of Cross-Layer Contracts

The physical behavior is influenced by changes in the software in a feedback system. This connection is explicitly defined by the cross-layer contracts that state the maximum useful age of data that can be used in perception, blocking budgets of the scheduler, packet deadlines of the networking, a fault condition, and an oracle of validation. Another effect of the contract on impossible designs is that impossible designs can be identified early in the contract; e.g. when the camera period consumes most of the stopping-distance budget, then the lost margin cannot be filled in by scheduler optimization.

16.2. Throughput Versus Predictability

The application of deep queues, batching, dynamic shaping and aggressive frequency scaling tend to achieve improved average utilization of the accelerator, although can also imply longer tail latency, and can make the completion time to be conditional upon unrelated tasks. Rather, the emphasis of robotics platforms should be on maximising the utility of missions considering both timing and safety constraints and not necessarily on the number of frames per second. Although probabilistic deadlines can be accepted by some applications, hard deadline limits are required by some applications; both can be accommodated by the architecture by categorizing tasks into different classes of criticality and allowing the controller to react by explicitly responding when data is late.

16.3. Centralized Compute as Against Distributed Compute

Concentrating attention and control on one module will cause less wiring and fewer serial connections but will cause more common-cause interference. Although distributed controllers are more localized and provide fault localization, they also need clock synchronization, network engineering and configuration management. One common real-world solution is to have the current/torque loops and the emergency response capabilities near the actuators, and the central computing unit to perform state estimation, state planning and AI tasks. The Hardware-in-the-Loop simulation is expected to perform the same partitioning and make use of the same communication paths since a change in one task would impact the timing as well as propagation of failures.

16.4. Limits of HIL

The HIL system cannot verify the effects that its models have omitted. It may be difficult to reproduce accurately contact and tire-soil interaction, flexible cables, electromagnetic interference, lens contamination, and human behaviour. Numerical solvers may lead to the introduction of artifacts and laboratory networks might be cleaner than field networks. For HIL to be used in a credible way it is therefore necessary to set out explicit fidelity limits, calibrate against measurements, carry out a residual-risk analysis, and proceed in stages to physical testing. Although HIL provides repeatability and safe fault coverage it does not replace evidence obtained in the field.

16.5. The Limits of the Illustrative Study

The distributions, correlations, power values and control responses of the case study are ready to show the design. They do not rely on a specific processor, robot or compiler. The findings represent the internal influences of the assumptions and they have to be replaced by measured target traces in an empirical submission. Most importantly, the extremely slender tails of latency, the scheduling of the graphics cards, and thermal behavior, and closed-loop tracking are quite reliant on the hardware and the workload. To prevent the

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

error of confusing simulated precision with experimental fact, these values are explicitly marked in the paper.

16.6. Research Directions

Future work should focus on creating a compositional timing model of CPU-GPU-DLA pipelines, on the implementation of accelerator scheduling that enables analyzable preemption, on control-aware neural architecture search, on creating uncertainty measures that remain calibrated when quantized and on digital twins with provable fidelity guarantees. The generation of the scenarios through automation may also be aimed at addressing the gaps in the safety coverage. The reports to be included in the shared robotics benchmark are as follows: the accuracy of the tasks, the success of the missions, the energy use, the latency of tails, the age of data, the interference, and the recovery behaviour, when running on the same platform.

Conclusion

The next-generation robots have to integrate edge intelligence with deterministic control when operating on a limited platform. Having a fast accelerator and a real-time kernel is not enough by itself. The system needs to link physical safety margins to data freshness, assign end-to-end deadlines to the various resources, isolate critical tasks, show when the system is overloaded, and establish useful degraded modes. The five-layer architecture suggested encodes these requirements as explicit data and configuration contracts. The method is completed by subjecting the actual production deployment to real-time plant and fault models. As part of a credible campaign, it is necessary to take into account operating-system and accelerator interference, network delay and loss, thermal derating, data from sensors that are out of date but still plausible, actuator limitations, changes to the model, and recovery procedures. The evidence system itself is also verified. The example design study demonstrates how isolation and adaptive inference can be assessed without having to claim that the experiments in question have not been carried out; in a physical implementation the workload assumptions would be replaced by the actual measurements and the same metrics and acceptance logic would be maintained. The design guidelines that result are specific: they set out the boundaries of the information age and specify how safe mode transitions should be handled. The safety channel must remain separate from the best-effort intelligence. Adaptation of the AI should take place only within the available timing slack. Each contract must be tested at its boundary in HIL before it is deployed in the field. Owing to these measures, the robot is able to use capable AI models while at the same time maintaining predictable behaviour during computing, communication, sensing, and in the face of environmental disturbances.

Appendix A Minimal Reproducibility Record

A conference artifact or internal verification package should contain the following items; the list is deliberately kept short so that each HIL run can be reconstructed and compared between software releases. As shown in Table 12 Minimum metadata for a reproducible HIL result.

Table 12 Minimum metadata for a reproducible HIL result

System	Robot configuration, compute module, carrier, sensors, actuators and wiring revision
Software	OS and kernel, middleware, drivers, model hash, inference engine, controller build
Timing	Clock source, offset bound, timestamp points, priorities, affinities and power mode
Model	Plant equations, solver, step size, parameters, calibration data and fidelity limits
Scenario	Initial state, environment, workload, faults, random seed and duration
Evidence	Raw trace, oracle version, exclusions, verdicts, plots and anomaly disposition

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

Appendix B Suggested Reporting Metrics

- Control: the distribution of tracking error, the maximum violation of constraints, the stopping distance, the recovery time, and the energy injected during contact.
- Timing: response-time distribution
- AI: task-specific loss, uncertainty calibration
- Platform: CPU and accelerator utilization, memory bandwidth, power, temperature

References

[1]. W. Shi, J. Cao, Q. Zhang, Y. Li, and L. Xu, "Edge computing: Vision and challenges," IEEE Internet of Things Journal, vol. 3, no. 5, pp. 637–646, 2016, doi: 10.1109/JIOT.2016.2579198.

[2]. [2] M. Satyanarayanan, "The emergence of edge computing," Computer, vol. 50, no. 1, pp. 30–39, 2017, doi: 10.1109/MC.2017.9.

[3]. [3] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," Proceedings of the IEEE, vol. 105, no. 12, pp. 2295–2329, 2017, doi: 10.1109/JPROC.2017.2761740.

[4]. A. G. Howard et al., "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv:1704.04861, 2017.

[5]. [5] B. Jacob et al., "Quantization and training of neural networks for efficient integer-arithmetic-only inference," in Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018, pp. 2704–2713, doi: 10.1109/CVPR.2018.00286.

[6]. S. Han, H. Mao, and W. J. Dally, "Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding," in Proc. International Conference on Learning Representations, 2016.

[7]. NVIDIA, "Jetson AGX Orin series technical brief," ver. 1.2, 2022. Available: https://www.nvidia.com/content/dam/en-zz/Solutions/gtcf21/jetson-orin/nvidia-jetson-agx-orin-technical-brief.pdf.

[8]. S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, architecture, and uses in the wild," Science Robotics, vol. 7, no. 66, eabm6074, 2022, doi: 10.1126/scirobotics.abm6074.

[9]. Y. Maruyama, S. Kato, and T. Azumi, "Exploring the performance of ROS2," in Proc. 13th International Conference on Embedded Software, 2016, pp. 1–10, doi: 10.1145/2968478.2968502.

[10]. Object Management Group, "Data Distribution Service specification, version 1.4," formal/2015-04-10, 2015. Available: https://www.omg.org/spec/DDS/1.4.

[11]. Open Robotics, "Real-time programming in ROS 2," ROS 2 Documentation. Available: https://docs.ros.org/en/rolling/Tutorials/Demos/Real-Time-Programming.html.

[12]. [12] Linux Kernel Documentation, "Real-time preemption: Theory of operation." Available: https://docs.kernel.org/core-api/real-time/theory.html.

[13]. C.-C. Dosoftei, C. Bîrsan, A.-T. Pleşcan, P.-R. Stan, and P.-M. Găîţă, "Hardware in the loop topology for an omnidirectional mobile robot using MATLAB in a Robot Operating System environment," Symmetry, vol. 13, no. 6, 969, 2021, doi: 10.3390/sym13060969.

[14]. J. A. Guerrero et al., "Real-time hardware-in-the-loop emulation of path tracking in low-cost agricultural robots," Vehicles, vol. 5, no. 3, pp. 1094–1119, 2023, doi: 10.3390/vehicles5030049.

[15]. F. Aggogeri et al., "A rapid control prototyping and hardware-in-the-loop approach for upper limb robotic exoskeletons control," Applied Sciences, vol. 14, no. 5, 2017, 2024, doi: 10.3390/app14052017.

[16]. M. B. Alatise et al., "Impacts of wireless on robot control: The network hardware-in-the-loop simulation framework and real-life comparisons," IEEE Internet of Things Journal, vol. 10, no. 12, pp. 10731–10744,

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5902 - 5922
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0761

2023, doi: 10.1109/JIOT.2022.3228082.

[17]. J. de la Cruz et al., "Electronic control unit and digital twin based on Raspberry Pi 4 for testing the remote nonlinear trajectory tracking of a P3-DX robot," Actuators, vol. 14, no. 8, 376, 2025, doi: 10.3390/act14080376.

[18]. M. Joseph and P. Pandya, "Finding response times in a real-time system," The Computer Journal, vol. 29, no. 5, pp. 390–395, 1986, doi: 10.1093/comjnl/29.5.390.

[19]. ONNX Runtime, "Quantize ONNX models," Microsoft. Available: https://onnxruntime.ai/docs/performance/model-optimizations/quantization.html.

[20]. TensorFlow, "Post-training quantization," TensorFlow Lite documentation. Available: https://www.tensorflow.org/model_optimization/guide/quantization/post_training.

[21]. D. B. de Oliveira and T. Cucinotta, "Demystifying the real-time Linux scheduling latency," in Proc. 32nd Euromicro Conference on Real-Time Systems, 2020, doi: 10.4230/LIPIcs.ECRTS.2020.9.

[22]. Object Management Group, "DDS extensions for time sensitive networking," DDS-TSN 1.0 Beta 2, 2024. Available: https://www.omg.org/spec/DDS-TSN/1.0/Beta2.

[23]. IEC, IEC 61508: Functional Safety of Electrical Electronic Programmable Electronic Safety-related Systems, Geneva, Switzerland.

[24]. ISO, ISO 13849-1: Safety of Machinery Safety-related Parts of Control Systems Part 1 General Principles for Design, Geneva, Switzerland, 2023.

[25]. ISO, ISO 10218-1: Robotics Safety Requirements Part 1 Industrial Robots, Geneva, Switzerland, 2025.

[26]. ISO, ISO 3691-4: Industrial Trucks Safety Requirements and Verification Part 4 Driverless Industrial Trucks and Their Systems, Geneva, Switzerland, 2023.