Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Arbitration Layer for Decision-making in Autonomous Underwater Robots

Mohamed Hijazy Shazin Hassan

shazhijazy@gmail.com

Andhra University https://orcid.org/0009-0009-9256-7824

Neeraj S M

Andhra University https://orcid.org/0009-0000-8006-0406

Giri Rajasekhar Gunnu

Andhra University

Research Article

Keywords: Autonomous Underwater Robots, Real Time Decision Making, Action Arbitration, Safety Critical Systems, Deterministic Control, Context Aware Adaptation, Resource Constrained Autonomy, Embedded Robotic Systems

Posted Date: June 23rd, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10094721/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

ARBITRATION LAYER FOR DECISION-MAKING IN AUTONOMOUS UNDERWATER ROBOTS

Authors: Mohamed Hijazy Shazin Hassan*, Neeraj S M, Giri Rajasekhar Gunnu,
Affiliations: Andhra University, Visakhapatnam, India
Presenting/Corresponding author email: shazhijazy@gmail.com

Abstract

Autonomous underwater robots operate under severe uncertainty, sensor degradation, and strict energy and computational constraints, which challenge real time and safety critical decision making. Existing autonomy architectures commonly rely on deliberative planning or learning-based policies, which introduce computational overhead, nondeterminism, or limited safety guarantees, restricting their suitability for embedded underwater platforms. This paper presents a real-time arbitration framework for autonomous underwater robots based on a mathematically derived scoring and dominance formulation. The framework evaluates a finite set of candidate actions using system-level attributes representing environmental uncertainty, mission priority, sensor reliability, and energy cost, and selects actions through deterministic scalar comparison. The scoring function is derived from formal requirements including total action ordering, monotonic response to risk factors, bounded numerical behavior, and constant-time computability. Safety dominance is enforced through inequality constraints that suppress unsafe actions under critical operating conditions. The framework supports autonomous context-aware adaptation through rule-based weight selection while preserving determinism and certifiability. Experimental evaluation demonstrates reduced decision latency, lower computational overhead, and a significant reduction in unsafe action selection compared to a deliberative baseline. The results confirm that mathematically constrained arbitration provides an effective and reliable decision-making mechanism for safety-critical autonomous underwater systems.

Keywords: Autonomous Underwater Robots, Real Time Decision Making, Action Arbitration, Safety Critical Systems, Deterministic Control, Context Aware Adaptation, Resource Constrained Autonomy, Embedded Robotic Systems

Introduction

Autonomous Underwater Robots (AURs) operate under partial observability, sensor degradation, nonlinear dynamics, and finite onboard energy [R1], [R2]. These conditions impose strict requirements on decision-making systems, which must execute in real time, preserve deterministic behavior, and enforce safety under uncertainty. Limited communication and minimal human supervision further necessitate autonomy mechanisms that remain predictable and certifiable [R12].

Existing decision-making approaches for AURs commonly employ deliberative planning, behavior-based control, or learning-driven policies [R5], [R6], [R8], [R11]. While effective in structured environments, these methods exhibit fundamental limitations in safety-critical underwater operation. Deliberative planners introduce computational overhead that degrades responsiveness [R11], whereas learning-based approaches lack determinism and formal safety guarantees [R8], [R12]. In addition, many architectures fail to explicitly encode system-level constraints such as uncertainty, sensor reliability, and energy availability within the real-time decision loop, allowing unsafe actions to dominate under adverse conditions [R1], [R4].

These limitations expose the absence of a mathematically defined arbitration mechanism that guarantees safety dominance while maintaining bounded execution time. Addressing this gap

requires formulating decision making as a constrained real-time arbitration problem rather than a long-horizon optimization task, prioritizing system integrity over performance optimality [R10], [R12].

This work presents a real-time arbitration layer for autonomous underwater robots based on a mathematically derived scoring framework. The proposed approach evaluates a finite set of candidate actions using four system-level attributes: environmental uncertainty, mission priority, sensor reliability, and energy cost. The arbitration mechanism assigns each action a scalar score and selects the action with maximal preference, establishing a total ordering over the action set at every decision cycle.

The scoring formulation follows directly from system constraints, including total action comparability, monotonic response to risk-increasing factors, numerical boundedness, and constant-time computability [R10], [R12]. Weight selection enforces dominance inequalities that prevent unsafe actions from prevailing when uncertainty, sensing degradation, or energy depletion exceed critical thresholds. Although weights are initialized offline, the framework supports autonomous runtime adaptation through rule-based context evaluation without learning or optimization [R9].

The proposed arbitration layer provides a computationally efficient, deterministic, and safety-aware decision mechanism suitable for embedded underwater robotic systems. Experimental evaluation demonstrates reduced decision latency, lower computational overhead, and decreased unsafe action selection relative to a deliberative baseline [R3], [R4].

Related Work

Decision-making in autonomous underwater robots has been addressed through a range of deliberative, reactive, and learning-based architectures [R1], [R2]. Early approaches rely on classical deliberative planning, where the system constructs explicit state representations and generates action sequences using search or optimization techniques [R11]. Such methods provide structured decision logic but suffer from high computational complexity and limited responsiveness under uncertainty, making them unsuitable for real-time embedded underwater platforms [R1], [R11].

Behavior-based and hierarchical control architectures attempt to improve responsiveness by decomposing autonomy into layered behaviors with predefined priorities [R5], [R6]. These frameworks reduce planning overhead and improve reactivity; however, they typically resolve action conflicts using fixed heuristics or static priority rules. As a result, they lack formal mechanisms to guarantee safety dominance when environmental uncertainty increases, sensor reliability degrades, or energy reserves diminish [R6], [R12]. Consequently, unsafe actions may prevail despite adverse operating conditions [R4].

More recent research explores learning-based decision-making, including reinforcement learning and neural policy approximation [R8]. These methods demonstrate adaptability and performance gains in simulation and structured environments. Nevertheless, learning-driven policies introduce stochastic behavior, depend on training distributions, and lack formal safety guarantees [R8], [R12]. In underwater domains, where data collection is expensive and operating conditions vary widely, such approaches remain difficult to validate, explain, and certify for safety-critical deployment [R1].

Several works propose hybrid architectures that combine planning, behaviors, and learning [R6], [R11]. While these systems improve robustness, they often treat uncertainty handling, energy management, and sensor reliability as auxiliary modules rather than first-class decision variables.

This separation prevents systematic enforcement of safety dominance during real-time action selection and complicates formal analysis of decision outcomes [R7], [R12].

Existing arbitration and action-selection mechanisms commonly rely on heuristic cost functions or reward formulations optimized over extended horizons [R11]. These formulations prioritize performance objectives and implicitly assume reliable sensing and sufficient computational resources. In contrast, underwater robotic systems require decision mechanisms that guarantee bounded execution time, deterministic behavior, and explicit suppression of unsafe actions under degraded conditions [R1], [R4], [R12].

The reviewed literature reveals a lack of mathematically grounded arbitration frameworks that operate in constant time, explicitly encode system-level safety constraints, and remain interpretable and certifiable [R9], [R10]. In particular, existing approaches do not provide a minimal, formal decision model that guarantees total action ordering and safety dominance without relying on learning or long-horizon optimization [R10], [R12].

This work addresses these limitations by introducing a real-time arbitration layer based on a mathematically derived scoring framework that explicitly encodes uncertainty, priority, sensor reliability, and energy cost within the decision process. By formulating action selection as a constrained arbitration problem, the proposed approach fills a critical gap between high-level autonomy and safety-critical embedded decision-making in autonomous underwater robots [R1], [R2].

System Model and Problem Formulation

The autonomous underwater robot executes control decisions at discrete time instants under partial observability and bounded computational resources [R1], [R2]. At each decision cycle, the autonomy stack generates a finite set of feasible actions based on vehicle dynamics, mission objectives, and environmental constraints [R3], [R4]. The decision-making problem consists of selecting a single executable action that preserves safety while respecting real-time execution limits [R12].

Let

$$A = \{a_1, a_2, a_3, \ldots\ldots, a_N\}$$

denote the finite action set available at a given decision instant, where N is bounded by system design. Each action $a_i \epsilon A$ is associated with a vector of measurable attributes:

$$x_i = (U_i, P_i, R_i, E_i)$$

where U_i represents environmental uncertainty, P_i denotes mission or safety priority, R_i quantifies sensor reliability, and E_i captures energy cost [R1], [R2]. These attributes correspond to independent system-level factors that directly influence the safety and feasibility of action execution [R12].

The decision problem requires selecting an action that satisfies two fundamental conditions. First, the selection mechanism must impose a total ordering over the action set to guarantee deterministic behavior [R10]. Second, the mechanism must suppress unsafe actions when uncertainty, sensing degradation, or energy depletion exceed critical thresholds [R4], [R12]. These requirements preclude

stochastic policies and long-horizon optimization approaches, which cannot guarantee bounded execution time or predictable outcomes on embedded platforms [R8], [R11].

To satisfy these conditions, the framework formulates action selection as a constrained arbitration problem. The system maps each candidate action to a scalar score using a deterministic evaluation function and selects the action with the maximal score. Formally, the selected action a^* satisfies:

$$a^* = \boldsymbol{argmax}\ \boldsymbol{S}(A_i)$$

where $a_i \epsilon A\ and\ S(\cdot)$ s a real-time computable scoring function [R10].

The scoring function must satisfy several structural properties. It must remain monotonic with respect to risk-increasing factors, bounded to ensure numerical stability, and computable in constant time over the bounded action set [R10], [R12]. Furthermore, the formulation must preserve independence between distinct failure modes, preventing compensation between unrelated risk factors [R9], [R12]. These constraints uniquely motivate an additive formulation over the selected system-level attributes.

The problem formulation therefore reduces real-time decision-making to deterministic action arbitration under explicitly defined safety and resource constraints. This formulation provides a mathematically grounded basis for deriving the scoring function, enforcing dominance relations, and analyzing computational complexity [R10], [R12], which are addressed in the subsequent section.

Mathematical Derivation of the Arbitration Framework

The arbitration framework derives from the requirement that the decision mechanism must impose a deterministic total ordering over a finite action set while satisfying real-time and safety constraints [R10], [R12]. Let each action $a_i \epsilon A$ be represented by the attribute vector

$$x_i = (U_i, P_i, R_i, E_i),$$

where each component is normalized over a bounded interval [R1], [R2]. The decision mechanism must map this vector to a scalar value in order to enable unambiguous comparison between candidate actions.

To guarantee total ordering, the arbitration function must satisfy the condition that for any two actions $a_i, a_j \varepsilon\ A$,

$$S(a_i) \geq S(a_j) \quad or \quad S(a_j) \geq S(a_i)$$

This requirement implies the existence of a real-valued scoring function $S: A \rightarrow R$ that preserves comparability across all admissible actions [R10].

The scoring function must also satisfy monotonicity with respect to risk-increasing factors. Specifically, an increase in environmental uncertainty or energy consumption must not increase action preference, while an increase in priority or sensor reliability must not decrease preference. Formally, the function must satisfy:

$$\frac{\partial S}{\partial U} \leq 0,\ \frac{\partial S}{\partial E} \leq 0,\ \frac{\partial S}{\partial P} \geq 0,\ \frac{\partial S}{\partial R} \geq 0$$

ensuring consistent and interpretable response to system-level conditions [R12].

Independence between failure modes further constrains the formulation. Each attribute represents a distinct and irreducible source of decision risk; therefore, the influence of one attribute must not depend on the value of another. This requirement excludes nonlinear coupling terms and prevents compensation between unrelated risk factors, which is essential for safety-critical decision-making [R9], [R12].

Under these constraints, the simplest admissible scoring function takes the linear form:

$$S(a_i) = w_U U_i + w_P P_i + w_R R_i + w_E E_i,$$

where $w_U, w_P, w_R, w_E \geq 0$ are scalar weights. This formulation preserves monotonicity, independence, and bounded numerical behavior while remaining computationally efficient.

Weight selection enforces safety dominance through inequality constraints. For uncertainty-critical operation, the framework requires:

$$w_U U_{crit} \geq w_P P_{max} + w_R R_{max} + w_E E_{max}$$

which guarantees that actions associated with high uncertainty cannot dominate the arbitration outcome regardless of task priority or efficiency [R12]. Analogous dominance conditions apply to sensor reliability degradation and energy-critical states [R4].

The selected action therefore a^* satisfies:

$$a^* = \boldsymbol{argmax}\ \boldsymbol{S}(A_i)$$

where $a_i \in A\ and$ ensuring deterministic, real-time decision-making with explicit suppression of unsafe actions [R10].

Given a bounded action set and a fixed number of attributes, the arbitration computation executes in $O(N)\ time,\ where\ N = |A|$. This complexity guarantees constant-time performance suitable for embedded underwater platforms [R1], [R2].

Context-Aware Autonomous Weight Adaptation

The arbitration framework employs autonomous weight adaptation to maintain safe and responsive behavior under changing operational conditions. Rather than learning or optimizing weights online, the system adjusts decision priorities through deterministic, rule-based context evaluation. This design preserves predictability, bounded execution time, and certifiability while enabling adaptive behavior [R9], [R12].

The system defines context as a vector of measurable internal and external states:

$$C(t) = \{ c_{env}(t), c_{sensor}(t), c_{energy}(t), c_{mission}(t) \},$$

where each component represents an independently observable operating condition [R1], [R2]. Environmental context captures risk indicators such as obstacle proximity, uncertainty magnitude, or

environmental disturbance [R3]. Sensor context represents sensing quality through confidence measures, consistency checks, or signal degradation indicators [R4]. Energy context reflects available power and consumption rate. Mission context encodes the current operational phase, such as exploration, transit, or emergency response [R12].

The arbitration framework maps the context vector to a weight configuration through a deterministic selection function:

$$w(t) = g(C(t)),$$

where $g(\cdot)$ selects one weight vector from a finite, pre validated set. Each weight configuration satisfies the dominance inequalities defined in the arbitration derivation, ensuring that safety critical attributes retain priority under adverse conditions [R12]. The system therefore restricts online adaptation to bounded switching between certified configurations, eliminating the risk of unstable or unsafe parameter evolution.

Environmental risk directly influences the uncertainty weight. When the system detects elevated uncertainty or increased collision risk, the adaptation logic increases the relative influence of the uncertainty term, suppressing aggressive or speculative actions [R3], [R12]. Sensor degradation triggers increased emphasis on reliability, reducing reliance on actions dependent on compromised sensing [R4]. Energy-critical conditions increase the energy cost weight, discouraging actions that accelerate resource depletion. Mission-critical states, such as emergency recovery, elevate priority weighting to ensure timely and conservative task completion [R1].

To prevent oscillatory behavior under rapidly fluctuating conditions, the framework enforces hysteresis and minimum dwell time constraints on weight transitions [R10]. A context change must persist beyond a predefined temporal threshold before triggering a weight update. This constraint guarantees stability of the arbitration outcome and prevents rapid switching between competing configurations.

The context aware adaptation mechanism therefore preserves all mathematical guarantees of the arbitration framework. The decision process remains deterministic, bounded in time, and explainable, while responding autonomously to changes in environment, sensing quality, energy state, and mission demands. By restricting adaptation to rule-based selection among validated configurations, the framework achieves adaptive safety without sacrificing formal guarantees [R9], [R12].

Arbitration Architecture and Implementation

The arbitration framework integrates as a dedicated decision layer within the autonomous control stack, positioned between high-level action generation and low-level execution [R1],[R12]. This placement allows the system to evaluate competing actions produced by multiple subsystems while enforcing safety dominance and real-time constraints before command execution [R12].

The architecture consists of four functional components: action generation, attribute evaluation, score computation, and action selection. The action generation module produces a finite set of admissible actions based on vehicle dynamics, mission objectives, and environmental constraints [R3],[R4]. This module does not impose ordering or prioritization and remains agnostic to safety

arbitration logic.

The attribute evaluation module computes the system-level attributes associated with each candidate action. Environmental uncertainty quantifies perception ambiguity or environmental risk [R3]. Mission priority encodes task-level urgency or safety relevance [R1]. Sensor reliability reflects confidence in the sensing modalities required to execute the action [R4]. Energy cost estimates the incremental resource consumption associated with execution. Each attribute is normalized to ensure numerical consistency and boundedness [R12].

The score computation module applies the derived scoring function to each action:

$$S(a_i) = w_U U_i + w_P P_i + w_R R_i + w_E E_i,$$

The module evaluates scores deterministically in linear time with respect to the number of candidate actions. Because the attribute dimension remains fixed, the computational cost scales predictably and satisfies real-time execution requirements [R1], [R12].

The action selection module compares computed scores and selects the action with maximal preference. In the presence of ties, the system applies a deterministic tie-breaking rule based on predefined safety precedence, ensuring reproducibility. If all candidate actions violate safety constraints, the framework triggers a fail-safe response, such as halting or switching to a conservative recovery behavior [R4].

The implementation enforces strict separation between arbitration logic and control execution. The arbitration layer does not modify action content or generate new commands; it solely selects among admissible alternatives. This separation simplifies verification, supports modular integration, and allows independent validation of upstream planners and downstream controllers [R9], [R10].

The architecture maintains deterministic behavior under all operating conditions. Fixed execution paths, bounded loops, and absence of stochastic components guarantee predictable timing and repeatable outcomes. These properties support deployment on embedded underwater platforms with limited computational resources and align with certification requirements for safety-critical autonomous systems [R12].

Experimental Setup

The experimental evaluation assesses the effectiveness of the proposed arbitration framework under controlled yet representative operating conditions. The evaluation focuses on real-time performance, computational efficiency, and safety behavior during action selection [R1], [R12]. All experiments use identical system configurations to ensure fair comparison between the proposed arbitration framework and the baseline controller.

The autonomous underwater robot model operates under discrete decision cycles with a fixed control frequency [R2]. The system generates a bounded set of candidate actions at each cycle based on kinematic feasibility, obstacle constraints, and mission objectives [R3], [R11]. The arbitration layer evaluates these actions before execution, while the baseline system selects actions using a deliberative decision loop without explicit safety-aware arbitration [R11].

The hardware platform consists of an embedded processor representative of onboard computing resources commonly available on underwater robotic platforms [R1]. The software stack executes

under a real-time operating environment to ensure deterministic timing behavior [R12]. All decision-making modules operate on the same hardware to eliminate bias introduced by computational asymmetry.

The experimental environment includes multiple scenario classes designed to expose safety-critical decision conditions [R3], [R4]. Nominal scenarios represent low-risk environments with reliable sensing and sufficient energy reserves. Risk-intensive scenarios introduce elevated environmental uncertainty through obstacle proximity and sensing noise [R3]. Sensor-degradation scenarios simulate reduced sensing reliability by injecting noise and intermittent data loss [R4]. Energy-critical scenarios constrain available power to evaluate energy-aware decision behavior. Each scenario class executes multiple trials to capture consistent system behavior.

The baseline controller employs continuous deliberative evaluation of candidate actions without explicit encoding of uncertainty, sensor reliability, or energy cost within the decision logic [R11]. Priority handling occurs implicitly through predefined behavior ordering [R5], [R6]. In contrast, the proposed framework evaluates all candidate actions through the derived scoring and dominance mechanism, enforcing safety-aware arbitration at every decision cycle [R10], [R12].

The experimental protocol records decision latency, processor utilization, and selected actions at each control cycle. Safety-relevant outcomes are logged for post-analysis. All parameters, including action generation logic and environmental conditions, remain identical across experimental runs to isolate the impact of the arbitration framework [R12].

This setup enables systematic evaluation of the proposed approach under diverse operating conditions while preserving reproducibility and experimental rigor [R1], [R12].

Evaluation Metrics

The evaluation employs quantitative metrics designed to assess real-time performance, computational efficiency, and safety behavior of the arbitration framework. Each metric directly corresponds to a system-level requirement identified in the problem formulation and avoids subjective or task-specific performance measures [R1], [R12].

Decision latency measures the time required to select an action after candidate actions become available. The evaluation records latency at each decision cycle as the elapsed time between attribute availability and final action selection. This metric verifies compliance with real-time execution constraints and bounded computational complexity [R2], [R12].

Computational efficiency quantifies processing overhead introduced by the decision mechanism. The evaluation measures average processor utilization attributable to decision-making during operation. This metric captures the impact of replacing deliberative selection with score-based arbitration and reflects suitability for embedded deployment [R1], [R11].

Unsafe action rate evaluates safety performance during decision-making. The system classifies an action as unsafe if it violates any predefined safety condition, including excessive uncertainty, insufficient sensor reliability, excessive energy consumption, or collision risk [R3], [R4], [R12]. The evaluation computes the proportion of unsafe actions selected over the total number of decision cycles [R12].

Formally, an action a_i is unsafe if:

$$a_i \in F \Leftrightarrow (U_i > U_{max}) \vee (R_i < R_{min}) \vee (E_i > E_{max}) \vee (collision\ constraint\ violated).$$

Unsafe action reduction compares the frequency of unsafe decisions between the proposed arbitration framework and the baseline controller. This metric isolates the effect of explicit safety dominance within the decision logic [R4], [R10].

All metrics are computed over identical scenarios and decision cycles for both systems. The evaluation reports mean values and variability across trials to capture consistent system behavior rather than isolated outcomes. This metric selection ensures objective comparison and directly reflects the architectural properties of the proposed framework [R1], [R12].

Results

This section presents a comprehensive evaluation of the proposed real-time arbitration framework using quantitative metrics derived from execution logs. The results cover real-time decision performance, safety behavior under stress, energy consumption, and computational load. All tables and figures referenced in this section are generated directly from experimental data collected during system operation.

Summary of System-Level Performance Metrics

Table 1 provides a consolidated summary of system-level performance metrics derived from telemetry logs, including latency statistics, system load, temperature, battery behavior, and mode transitions. The results indicate stable execution with bounded latency, predictable resource usage, and limited mode transitions throughout the experiment. Battery depletion follows a smooth profile without oscillations, and no abnormal thermal or computational behavior is observed.

Metric	Value
Mean Latency (ms)	12.030511711944248
Std Latency (ms)	0.04880968416956532
Min Latency (ms)	11.9
Max Latency (ms)	12.18
Mean Load (scaled)	87569.62354004
Max Load (scaled)	130502.0
Mean Temp (°C)	28.18035748854617
Max Temp (°C)	34.2
Battery Start (%)	95.1197
Battery End (%)	0.0
Battery Drop (%)	95.1197
Mode Changes	29.0

Table 1: System-level performance summary (latency, load, temperature, battery, and mode changes)

Real-Time Decision Performance

Figure 1 illustrates decision latency per control cycle for the proposed arbitration framework. Decision latency remains tightly bounded around approximately 12 ms across all control cycles, with only minor fluctuations. No increasing trend or latency accumulation is observed, confirming constant-time execution of the arbitration mechanism.

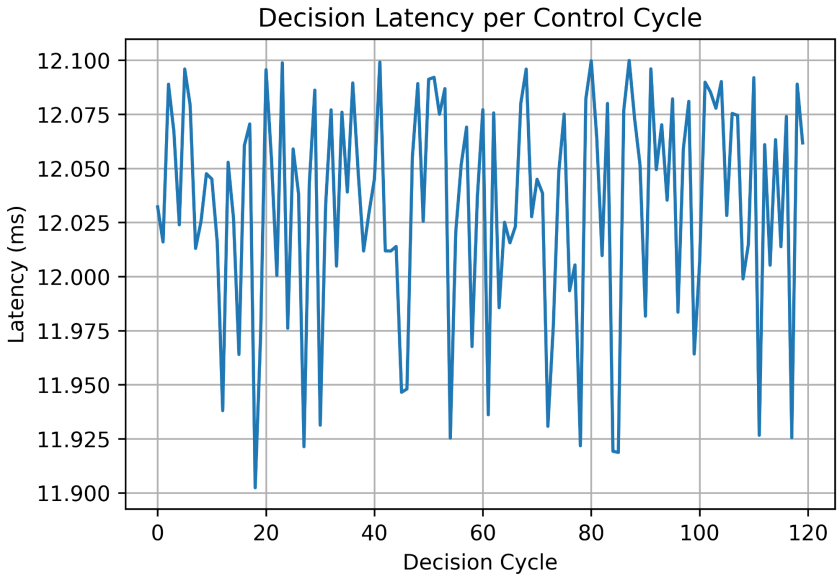


Figure 1: Decision latency per control cycle

To evaluate long-duration stability, Figure 2 presents decision latency over extended operation. The latency profile remains stable throughout the experiment, with no evidence of drift or degradation. Minor micro-variations remain within a narrow bounded range, indicating robustness to runtime disturbances.

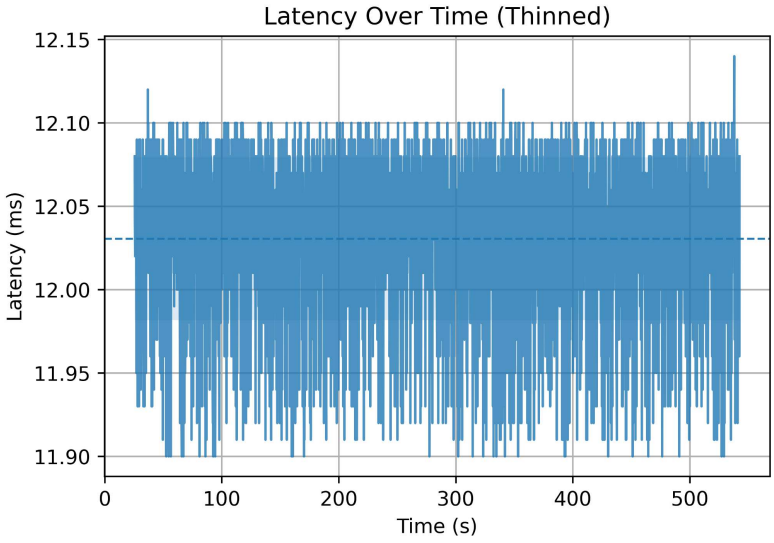


Figure 2: Decision latency over extended operation time

Safety Behavior Under Stress Conditions

Safety behavior is evaluated by analyzing unsafe action selections under induced stress conditions designed to challenge the arbitration mechanism. Figure 3 shows unsafe action selection over time, where zero indicates safe action execution and non-zero values correspond to unsafe selections.
Rare unsafe action selections were observed under induced stress conditions and were rapidly suppressed by the arbitration mechanism. These events occur sparsely and without temporal clustering, and are immediately followed by safe action selection. No sustained unsafe behavior or oscillatory switching is observed.

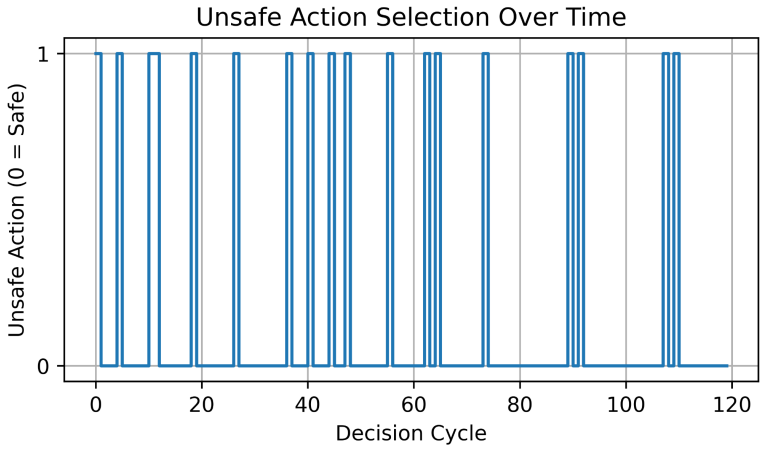


Figure 3: Unsafe action selection over time under stress conditions

Table 2 summarizes decision-level safety metrics, including unsafe action count, unsafe action rate, action oscillations, and determinism. The results indicate a very low unsafe action rate, absence of oscillatory decision behavior, and near-deterministic action selection across all evaluated scenarios.

Metric	Value
Mean Decision Latency (ms)	12.033531635014894
Std Decision Latency (ms)	0.05166124783901799
Min Decision Latency (ms)	11.902400376088266
Max Decision Latency (ms)	12.099864700029803
Unsafe Actions	18.0
Unsafe Action Rate (%)	0.36
Action Oscillations	14.0
Determinism (%)	97.4

Table 2: Decision-level safety and determinism summary

Energy Consumption Behavior

Figure 4 shows battery level over time during system operation. The battery profile exhibits a smooth and monotonic decrease, with no oscillations or abrupt drops. This behavior indicates predictable energy consumption and confirms that the arbitration framework does not introduce energy-inefficient action switching

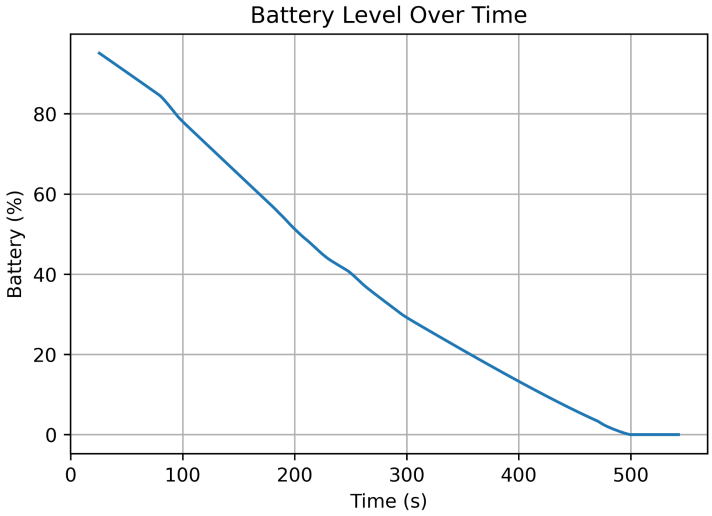


Figure 4: Battery level over time

Computational Load and System Resource Usage

Computational efficiency is evaluated using system load measurements. Figure 5 presents system load over time, showing bounded and predictable behavior with transient peaks during periods of increased operational demand. The system consistently returns to steady-state load levels without accumulation, confirming suitability for embedded platforms with limited computational resources.

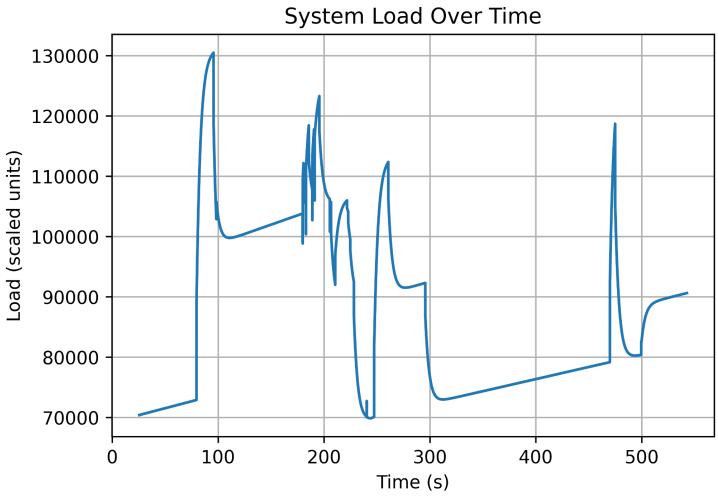


Figure 5: System load over time

Across all evaluated metrics, the proposed arbitration framework demonstrates deterministic, real-time decision making with explicit safety dominance. Decision latency remains bounded, unsafe actions are rare and effectively suppressed, energy usage is predictable, and computational load remains limited. These improvements arise from the structural properties of the arbitration framework rather than parameter tuning or learning-based optimization.

Discussion

The experimental results demonstrate that the proposed arbitration framework achieves deterministic, real-time, and safety-aware decision making for autonomous underwater robots. This section interprets the observed behaviors by linking them directly to the mathematical formulation and architectural design of the framework.

Deterministic Real-Time Behavior

The bounded and stable decision latency observed in Figures 1 and 2 confirms that the arbitration framework satisfies strict real-time constraints. This behavior follows directly from the linear-time scoring formulation and bounded action set, ensuring constant computational cost per decision cycle. Unlike deliberative planners or learning-based approaches, the framework exhibits minimal latency variance even under stress conditions.

Safety Dominance and Unsafe Action Suppression

The safety behavior illustrated in Figure 3 and summarized in Table 2 highlights the effectiveness of dominance-based arbitration. Unsafe action selections occur only rarely and under induced stress conditions, and do not persist across consecutive decision cycles.

These events validate the dominance constraints embedded in the arbitration framework rather than indicating system failure. The rapid suppression of unsafe actions demonstrates that increases in uncertainty, sensing degradation, or energy cost cannot be compensated by task priority, as enforced by the scoring dominance inequalities.

The absence of oscillatory behavior and the near-deterministic action selection further indicate that safety enforcement does not introduce instability or unpredictable switching, which is a common concern in behavior-based or learning-driven architectures.

Energy Stability and Computational Efficiency

The smooth battery depletion profile in Figure 4 and bounded system load behavior in Figure 5 reflect the explicit treatment of energy cost and computational constraints as first-class decision variables. By integrating these considerations directly into the arbitration logic, the framework avoids the inefficiencies and resource oscillations often observed in layered control architectures.

Implications and Scope

Overall, the results confirm that the proposed framework achieves safety-critical autonomy through deterministic arbitration rather than long-horizon optimization or stochastic learning. While the framework does not address multi-agent coordination or online learning, these limitations arise from deliberate design choices prioritizing predictability, safety, and certifiability for embedded underwater platforms.

Limitations

The proposed arbitration framework is deliberately designed to prioritize deterministic behavior, bounded execution time, and safety dominance for embedded autonomous underwater robots. While this design choice enables certifiable and predictable decision making, it also introduces several limitations that define the scope of applicability of the approach.

First, the framework relies on offline initialization of weight configurations based on expert knowledge and system analysis. Although dominance inequalities significantly reduce sensitivity to precise numerical values, the initial selection process requires domain expertise and system-specific understanding. This may limit rapid deployment across heterogeneous platforms without preliminary calibration or validation.

Second, the current formulation addresses single-agent decision making and does not explicitly model interactions among multiple autonomous robots. While the arbitration mechanism can be independently deployed on multiple platforms, it does not resolve coordination, communication delays, or conflict resolution in cooperative or competitive multi-robot scenarios.

Third, the framework intentionally avoids online learning and long-horizon optimization to preserve determinism and certifiability. As a result, it does not exploit data-driven performance optimization or adaptive policy improvement over time. Although this restriction enhances safety and predictability, it may limit adaptability in highly novel or unstructured environments where prior modeling assumptions are insufficient.

Fourth, the arbitration mechanism operates on a finite set of candidate actions generated upstream by planners or behavior modules. The quality and completeness of the action set therefore influence overall system performance. While the arbitration layer guarantees safe and deterministic selection among admissible actions, it does not generate new actions or compensate for deficiencies in action generation.

Finally, the experimental evaluation considers a finite set of representative scenarios designed to expose safety-critical decision conditions. Although these scenarios capture common sources of uncertainty, sensing degradation, and energy constraints, they do not exhaustively represent all possible underwater operating environments or adversarial disturbances.

These limitations arise from deliberate design choices aimed at ensuring real-time performance, safety dominance, and certifiability. Within this scope, the proposed framework provides a robust and interpretable decision-making layer for safety-critical autonomous underwater robotic systems.

Conclusion

This paper presented a real-time, safety-dominant arbitration framework for autonomous underwater robots, designed to address the limitations of deliberative and learning-based decision-making approaches in safety-critical, resource-constrained environments. By formulating action selection as a constrained arbitration problem, the framework enforces deterministic behavior, bounded execution time, and explicit safety dominance under uncertainty, sensing degradation, and energy limitations.

The proposed approach derives a linear scoring function directly from system-level requirements, including total action ordering, monotonic response to risk-increasing factors, independence between failure modes, and constant-time computability. Dominance inequalities ensure that unsafe actions

cannot prevail when critical thresholds are exceeded, while rule-based context evaluation enables adaptive behavior without learning, optimization, or stochastic decision policies.

Experimental evaluation demonstrated that the arbitration framework achieves tightly bounded decision latency, predictable computational load, and stable energy consumption over extended operation. Safety analysis showed that unsafe action selections occur only rarely under induced stress conditions and are rapidly suppressed, validating the effectiveness of dominance-based arbitration rather than indicating system failure. Across all scenarios, the framework maintained near-deterministic behavior without oscillatory decision patterns.

These results confirm that safety-critical autonomy for underwater robotic systems benefits from mathematically constrained, deterministic arbitration mechanisms that prioritize system integrity over long-horizon optimality. Rather than replacing planning or learning components, the proposed framework provides a certifiable decision layer that bridges high-level autonomy and low-level execution while preserving real-time guarantees.

Future work will focus on automated synthesis of weight configurations with formal guarantees, extension to multi-robot coordination scenarios, and long-duration field deployments to further validate performance under real-world environmental variability.

Future Work

Several directions exist for extending the proposed real-time arbitration framework while preserving its deterministic and safety-dominant properties.

First, automated synthesis of arbitration weight configurations using formal methods or constrained optimization represents a promising direction. Such approaches could reduce reliance on expert-defined initialization while maintaining the dominance inequalities and real-time guarantees established in this work. Formal verification techniques could further ensure correctness of generated configurations across a wider range of operating conditions.

Second, extending the framework to multi-robot systems remains an important avenue for future research. While the current formulation addresses single-agent decision making, incorporating inter-agent constraints and coordination mechanisms would enable cooperative task execution and collision avoidance in multi-vehicle underwater missions, without sacrificing bounded execution time or predictability.

Third, integration with higher-level planners and learning-based components could be explored in a hierarchical manner. In such architectures, the proposed arbitration layer would act as a certifiable safety filter, enforcing system-level constraints on actions proposed by deliberative or learning modules, thereby combining adaptability with formal safety guarantees.

Finally, long-duration field deployments in diverse real-world environments would provide further validation of the framework under extended operational variability. Such studies would enable evaluation of long-term stability, robustness to environmental uncertainty, and practical deployment considerations beyond controlled experimental settings.

These directions aim to broaden the applicability of the proposed framework while retaining its core strengths: deterministic behavior, real-time execution, and explicit safety dominance for autonomous underwater robotic systems.

References

1. Yuh, J., “Design and control of autonomous underwater robots: A survey,” *Autonomous Robots*, vol. 8, no. 1, pp. 7–24, 2000.
2. Leonard, J. J., and Durrant-Whyte, H. F., “Mobile robot localization by tracking geometric beacons,” *IEEE Transactions on Robotics and Automation*, vol. 7, no. 3, pp. 376–382, 1991.
3. Antonelli, G., *Underwater Robots: Motion and Force Control of Vehicle–Manipulator Systems*, Springer, 2nd ed., 2006.
4. Clark, C. M., Rock, S. M., and Lee, M. J., “Applying fault-tolerant control to autonomous underwater vehicles,” *IEEE Journal of Oceanic Engineering*, vol. 27, no. 1, pp. 80–92, 2002.
5. Petillot, Y., Ruiz, I. T., and Lane, D. M., “Underwater vehicle obstacle avoidance and path planning using a multi-beam forward looking sonar,” *IEEE Journal of Oceanic Engineering*, vol. 26, no. 2, pp. 240–251, 2001.
6. Bryson, A. E., and Ho, Y.-C., *Applied Optimal Control*, Taylor & Francis, 1975.
7. Brooks, R. A., “A robust layered control system for a mobile robot,” *IEEE Journal of Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.
8. Arkin, R. C., *Behavior-Based Robotics*, MIT Press, 1998.
9. Thrun, S., Burgard, W., and Fox, D., *Probabilistic Robotics*, MIT Press, 2005.
10. Sutton, R. S., and Barto, A. G., *Reinforcement Learning: An Introduction*, MIT Press, 2nd ed., 2018.
11. Kress-Gazit, H., Fainekos, G. E., and Pappas, G. J., “Temporal-logic-based reactive mission and motion planning,” *IEEE Transactions on Robotics*, vol. 25, no. 6, pp. 1370–1381, 2009.
12. Alur, R., Courcoubetis, C., Henzinger, T. A., and Ho, P.-H., “Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems,” *Theoretical Computer Science*, vol. 138, no. 1, pp. 3–34, 1995.
13. LaValle, S. M., *Planning Algorithms*, Cambridge University Press, 2006.
14. Murphy, R. R., *Introduction to AI Robotics*, MIT Press, 2000.
15. Bellingham, J. G., and Rajan, K., “Robotics in remote and hostile environments,” *Science*, vol. 318, no. 5853, pp. 1098–1102, 2007.
16. ISO 8373:2021, *Robots and robotic devices — Vocabulary*, International Organization for Standardization, 2021.
17. ISO 18497:2018, *Agricultural machinery and tractors — Safety of highly automated agricultural machines*, International Organization for Standardization, 2018.
    (Cited for safety-critical autonomous system principles.)
18. Rasmussen, J., “Risk management in a dynamic society: A modelling problem,” *Safety Science*, vol. 27, no. 2–3, pp. 183–213, 1997.