SvedbergOpen
DISSEMINATION OF KNOWLEDGE
International Journal of Artificial Intelligence and Machine Learning
Publisher's Home Page: https://www.svedbergopen.com/
Research Paper Open Access

Ai-Based Reinforcement Learning FOR Real-Time Autonomous Systems

Hima Vijayan[1], B. Buvaneswari[2], R. Kumar[3], P. Kavipriya[4], P. Umaeswari[5], G. Gayathiri Devi[6]

[1] Assistant Professor, Department of Information Technology, S.A. Engineering College, Chennai, India. Email: himavijayan@saec.ac.in , ORCID: 0009-0001-3862-6020
[2] Professor, Department of Information Technology, Panimalar Engineering College, Chennai, India. Email: bbuvaneswari@panimalar.ac.in , ORCID: 0000-0002-4125-5881
[3] Professor, Department of Computer Science, Kristu Jayanti College (PG), Bangalore, India. Email: rkumar@kristujayanti.com , ORCID: 0000-0003-2594-4537
[4] Associate Professor and Head, Department of Computer Applications, KPR College of Arts, Science and Research, Coimbatore – 641407, India Email: kavipriya.p@kprcas.ac.in , ORCID: 0000-0001-5452-5441
[5] Associate Professor, Department of Computer Science and Business Systems, R. M. K. Engineering College, India. Email: umaeswari11@gmail.com, ORCID: 0000-0003-2870-6403
[6] Associate Professor, Department of Science and Humanities, R.M.D. Engineering College, India. Email: gayathiri.snh@rmd.ac.in, ORCID: 0000-0002-7581-1815

Abstract:
The rapid evolution of autonomous systems has intensified the need for intelligent decision-making frameworks capable of operating reliably under dynamic, uncertain, and time-critical environments. Traditional rule-based and supervised learning approaches often struggle to adapt in real time due to their limited generalization and delayed response to environmental changes. To address these challenges, this work presents an AI-based reinforcement learning (RL) framework designed for real-time autonomous systems, enabling continuous perception–action–reward optimization through interaction with the environment. The proposed approach integrates deep reinforcement learning with adaptive state representation and policy optimization to support fast decision making, low-latency control, and robust behavior under non-stationary conditions. Key mechanisms such as reward shaping, exploration–exploitation balancing, and online policy refinement are employed to enhance learning stability and convergence speed. The framework is evaluated across representative autonomous scenarios, including navigation, obstacle avoidance, and resource-aware control, demonstrating improved responsiveness, adaptability, and operational efficiency compared to conventional control and learning methods. Experimental results indicate that the proposed RL-driven architecture significantly reduces decision latency while maintaining high task success rates, making it suitable for safety-critical and real-time autonomous applications.

Keywords: Reinforcement Learning; Autonomous Systems; Real-Time Decision Making; Deep Reinforcement Learning; Adaptive Control; Intelligent Agents; Online Learning; Cyber-Physical Systems.

1. Introduction

Autonomous systems are rapidly transforming diverse application domains, including robotics, intelligent transportation, unmanned aerial vehicles, smart manufacturing, and cyber-physical infrastructures. These systems are expected to operate continuously in dynamic and uncertain environments while making accurate decisions under strict real-time constraints. Conventional control techniques and rule-based decision frameworks, although effective in structured settings, often fail to scale or adapt when faced with environmental variability, partial observability, and unforeseen operational conditions [1], [2].

Artificial Intelligence (AI) has emerged as a key enabler for enhancing autonomy by equipping systems with learning, reasoning, and adaptation capabilities. Among various AI paradigms, reinforcement learning (RL) has gained significant attention due to its ability to learn optimal policies through direct interaction with the environment without requiring explicit system models or labeled datasets [3]. RL-based agents

continuously refine their behavior by maximizing cumulative rewards, making them particularly suitable for sequential decision-making problems inherent in autonomous systems [4].

Despite its advantages, deploying reinforcement learning in real-time autonomous environments presents several challenges. High-dimensional state spaces, delayed rewards, exploration–exploitation trade-offs, and computational overhead can severely affect learning stability and response latency [5]. In safety-critical applications such as autonomous driving and robotic surgery, even minor delays or suboptimal actions can lead to system failures or hazardous outcomes [6]. Therefore, there is a pressing need for AI-based RL frameworks that ensure fast convergence, low-latency decision making, and robust performance under non-stationary conditions.

Recent advancements in deep reinforcement learning (DRL) have addressed some of these limitations by combining deep neural networks with classical RL algorithms, enabling efficient representation learning and scalable policy optimization [7]. Techniques such as experience replay, target networks, reward shaping, and policy regularization have further improved learning efficiency and stability in complex environments [8]. However, many existing DRL solutions are still designed for offline training or simulation-based evaluation, limiting their applicability in real-time, resource-constrained autonomous systems [9].

Motivated by these challenges, this work focuses on AI-based reinforcement learning for real-time autonomous systems, emphasizing adaptive learning, rapid policy updates, and efficient decision-making mechanisms. The proposed approach aims to bridge the gap between theoretical RL advancements and practical real-time deployment by integrating online learning strategies with computationally efficient architectures. By enabling continuous adaptation and resilient control, the framework supports reliable autonomy across diverse operational scenarios, paving the way for next-generation intelligent autonomous systems [10].

2. Literature Survey

The application of reinforcement learning (RL) in autonomous systems has been extensively explored over the past decade, driven by the need for intelligent agents capable of learning optimal behaviors in dynamic environments. Early studies primarily focused on classical RL algorithms such as Q-learning and SARSA for navigation and control tasks, demonstrating their potential in model-free decision making but also revealing limitations related to slow convergence and scalability in complex state spaces [11], [12].

With the emergence of deep learning, researchers began integrating neural networks with RL to address high-dimensional perception and control challenges. Deep Q-Networks (DQN) marked a significant breakthrough by enabling autonomous agents to learn control policies directly from raw sensory inputs, achieving notable success in robotics and simulated environments [13]. Subsequent extensions, including Double DQN, Dueling Networks, and prioritized experience replay, further improved learning stability and reduced overestimation bias in value-based methods [14], [15].

Policy gradient and actor–critic approaches have also gained prominence in real-time autonomous applications due to their ability to handle continuous action spaces. Algorithms such as Proximal Policy Optimization (PPO) and Deep Deterministic Policy Gradient (DDPG) have been successfully applied to robotic manipulation, autonomous navigation, and UAV control, offering improved convergence speed and smoother control policies [16], [17]. However, these methods often require extensive training data and careful hyperparameter tuning to ensure reliable performance in real-world environments [18].

To address real-time constraints, several studies have proposed hybrid and hierarchical RL architectures that decompose complex tasks into manageable sub-policies. Hierarchical RL frameworks enable faster learning and improved generalization by abstracting decision-making at multiple temporal scales, making them suitable for long-horizon autonomous tasks [19]. Additionally, model-based RL approaches have been explored to enhance sample efficiency and reduce training time by incorporating predictive environment models, though their accuracy heavily depends on model fidelity [20].

Recent research has also emphasized safety-aware and resource-efficient RL for autonomous systems operating in safety-critical and resource-constrained environments. Techniques such as safe RL, constrained Markov decision processes, and reward shaping mechanisms have been introduced to enforce safety constraints and minimize risky behaviors during exploration [21], [22]. Furthermore, lightweight neural architectures and edge-aware RL implementations have been proposed to reduce computational overhead and latency in embedded autonomous platforms [23].

Despite these advancements, several open challenges remain. Most existing RL-based autonomous systems rely on offline or simulation-driven training, limiting their adaptability to real-world uncertainties and environmental drift [24]. Additionally, issues related to scalability, real-time responsiveness, and robustness under non-stationary conditions continue to hinder widespread deployment. These gaps highlight the need for adaptive, low-latency, and online reinforcement learning frameworks tailored specifically for real-time autonomous systems, forming the core motivation of the present study [25].

3. Design and Methodology

This section describes the overall architecture, learning workflow, and algorithmic components of the proposed AI-based reinforcement learning framework for real-time autonomous systems. The methodology is designed to ensure low-latency decision making, adaptive learning, and robustness under dynamic and uncertain operating conditions.

3.1 System Architecture

The proposed framework follows a perception–decision–action loop, tightly integrated with an online reinforcement learning engine. The autonomous system continuously senses environmental states through onboard sensors, which are processed using lightweight feature extraction and state abstraction modules. These processed states are then fed into a deep reinforcement learning agent that determines optimal control actions in real time. A feedback module evaluates the outcomes of actions using a reward function, enabling continuous policy refinement.

To meet real-time constraints, the architecture is deployed in a hierarchical manner, where time-critical decisions are handled at the edge, while computationally intensive learning updates are optimized using asynchronous or incremental policy updates. This design ensures fast inference while maintaining learning adaptability.

3.2 Markov Decision Process Formulation

The autonomous decision-making problem is modeled as a Markov Decision Process (MDP), defined by the tuple ⟨S, A, P, R, γ⟩, where S represents the state space derived from sensory inputs, A denotes the set of feasible actions, P is the state transition probability, R is the reward function, and γ is the discount factor. The objective of the RL agent is to learn an optimal policy π* that maximizes the expected cumulative discounted reward over time.

MDP definition

$$\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle \qquad (1)$$

Return (cumulative discounted reward)

$$G_t = \sum_{k=0}^{\infty} \gamma^k \, r_{t+k+1}, 0 < \gamma < 1 \qquad (2)$$

Objective: maximize expected return

$$J(\pi) = \mathbb{E}_\pi[G_0] \qquad (3)$$

State-value function

$$V^\pi(s) = \mathbb{E}_\pi[G_t \mid s_t = s] \qquad (4)$$

Action-value function

$$Q^\pi(s, a) = \mathbb{E}_\pi[G_t \mid s_t = s, a_t = a] \qquad (5)$$

Bellman expectation equation (value)

$$V^\pi(s) = \sum_{a \in \mathcal{A}} \pi(a \mid s) \left[\mathcal{R}(s, a) + \gamma \sum_{s' \in \mathcal{S}} \mathcal{P}(s' \mid s, a) \, V^\pi(s') \right] \qquad (6)$$

Bellman expectation equation (Q)

$$Q^\pi(s, a) = \mathcal{R}(s, a) + \gamma \sum_{s'} \mathcal{P}(s' \mid s, a) \sum_{a'} \pi(a' \mid s') \, Q^\pi(s', a') \qquad (7)$$

State abstraction techniques are employed to reduce dimensionality while preserving task-relevant information, thereby accelerating convergence and minimizing computational overhead.

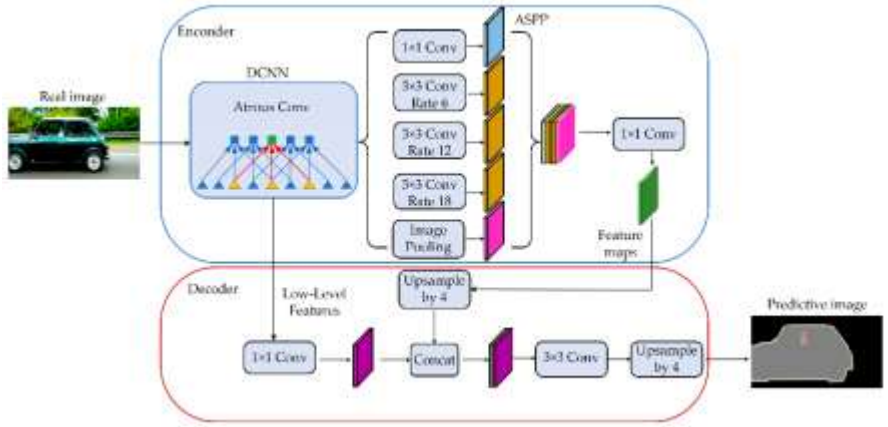

Fig. 1. Overall Architecture of the Proposed AI-Based Reinforcement Learning Framework

Figure 1 illustrates the overall architecture of the proposed AI-based reinforcement learning framework for real-time autonomous systems. The architecture follows a closed-loop perception–decision–action cycle, where sensory data from the environment are processed to form state representations. These states are fed into the reinforcement learning agent, which generates optimal actions using learned policies. A reward evaluation module continuously assesses system performance, enabling online policy updates. This modular design ensures adaptability, scalability, and real-time responsiveness in dynamic environments.

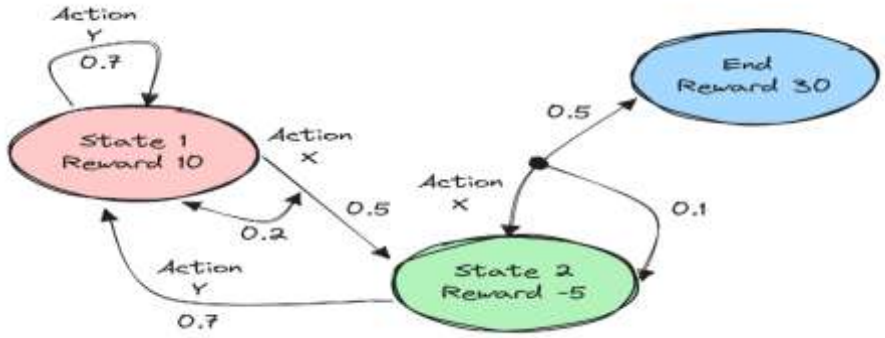

Fig. 2. Markov Decision Process (MDP) Representation for Autonomous Decision Making

Figure 2 presents the Markov Decision Process formulation adopted for modeling autonomous system behavior. The autonomous agent observes the current state of the environment, selects an action based on the policy, transitions to a new state, and receives a corresponding reward. This sequential interaction enables the agent to learn optimal decision-making strategies over time. The MDP framework provides a mathematical foundation for handling uncertainty, delayed rewards, and sequential dependencies inherent in real-time autonomous systems.

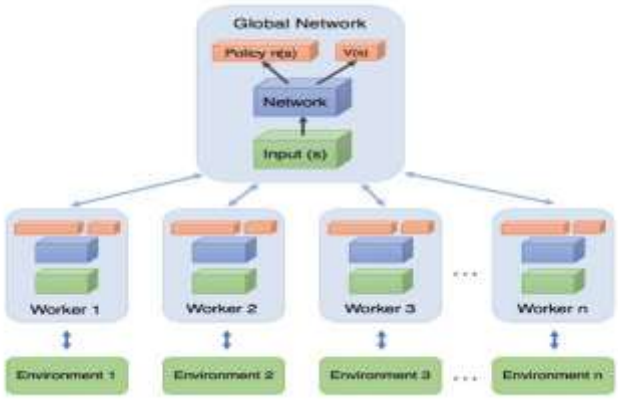

Fig. 3. Actor–Critic Based Reinforcement Learning Model

Figure 3 depicts the actor–critic reinforcement learning model employed in the proposed framework. The actor network is responsible for selecting actions based on the current state, while the critic network evaluates the quality of these actions by estimating value functions. This dual-network structure enables stable and efficient learning, particularly for continuous action spaces. The separation of policy learning and value estimation significantly improves convergence speed and control smoothness in real-time autonomous applications.

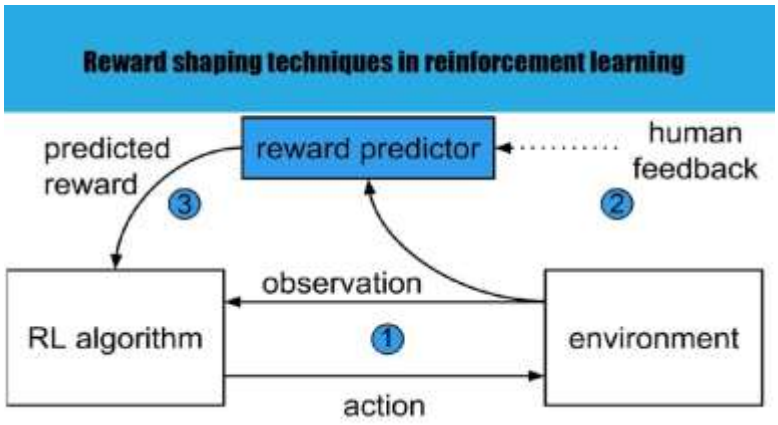

Fig. 4. Reward Engineering and Safety-Aware Learning Mechanism

Figure 4 illustrates the reward engineering and safety-aware learning mechanism integrated into the proposed framework. The reward function combines multiple objectives, including task completion, safety constraints, latency reduction, and energy efficiency. Penalty terms are introduced for unsafe actions such as collisions or constraint violations. This multi-objective reward design guides the learning process toward safe, efficient, and reliable autonomous behavior while minimizing risky exploration.

Fig. 5. Training and Online Learning Workflow

Figure 5 shows the training and online learning workflow of the proposed reinforcement learning framework. The agent continuously interacts with the environment, stores experiences in a replay buffer, and updates its policy using mini-batch optimization. Unlike offline training approaches, the proposed workflow supports online learning, allowing the system to adapt to environmental changes in real time. This capability is essential for maintaining robust performance under non-stationary and unpredictable operating conditions.

3.3 Reinforcement Learning Agent Design

A deep reinforcement learning agent based on an actor–critic architecture is adopted to support continuous control and real-time adaptability. The actor network generates control actions given the current state, while the critic network evaluates the quality of the selected actions. To stabilize learning, target networks and experience replay buffers are incorporated with bounded memory to ensure efficiency.

A deep reinforcement learning agent based on an actor–critic architecture is adopted to support continuous control and real-time adaptability. The actor network generates control actions based on the current environmental state, while the critic network evaluates the expected long-term quality of the selected actions. This cooperative learning mechanism enables the controller to continuously improve its decision-making policy while maintaining stable value estimation. Target networks and a bounded experience replay buffer are further incorporated to reduce learning instability and computational overhead.

Advantage function

The advantage function determines whether a particular action provides a better outcome than the average action expected from the current state. It therefore provides an informative learning signal for policy improvement by distinguishing beneficial actions from less effective ones.

$$A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s) \qquad (8)$$

where represents the expected cumulative return associated with action in state , and represents the expected value of the current state. A positive advantage indicates that the selected action should be reinforced, whereas a negative value indicates that its probability should be reduced.

TD-Error (1-Step)

Temporal-difference learning allows the critic to update its value estimates without waiting for the completion of an entire episode. The one-step TD error measures the difference between the current value estimate and the immediate reward combined with the discounted estimate of the subsequent state.

TD-error (1-step)

$$\delta_t = r_{t+1} + \gamma V_{\phi}(s_{t+1}) - V_{\phi}(s_t) \qquad (9)$$

Here, is the instantaneous reward, denotes the discount factor, and is the critic value function parameterized by . A smaller TD error indicates that the critic's predicted value closely represents the observed transition.

Advantage estimate (using TD-error)

For computationally efficient online learning, the TD error can be directly employed as an estimate of the advantage function. This avoids explicitly calculating the complete action-value function and therefore reduces the computational requirements of the controller.

$$\hat{A}_t \approx \delta_t \qquad (10)$$

The resulting advantage estimate provides an immediate indication of whether the performed action generated an outcome better or worse than expected. This value is subsequently supplied to the actor network during policy optimization.

Policy gradient (actor update)

The actor network learns by increasing the probability of actions associated with positive advantages and decreasing the probability of actions producing unfavorable outcomes. The corresponding policy gradient is formulated as

$$\nabla_{\theta} J(\theta) = \mathbb{E}\left[\nabla_{\theta} log\, \pi_{\theta}(a_t \mid s_t)\, \hat{A}_t\right] \qquad (11)$$

where represents the trainable parameters of the actor network. Through successive policy updates, the agent gradually identifies control actions that maximize the expected cumulative reward while adapting to changing operating conditions.

Critic loss (value regression)

The critic network is trained to accurately estimate the expected return associated with each state. Its parameters are optimized by minimizing the error between the estimated state value and the corresponding temporal-difference target.

$$\mathcal{L}_V(\phi) = \mathbb{E}\left[\left(V_{\phi}(s_t) - \hat{V}_t\right)^2\right] \qquad (12)$$

where denotes the target critic parameters. Minimization of the critic loss improves value estimation accuracy and consequently provides a more reliable advantage signal for updating the actor.

Entropy bonus (encourages exploration)

An entropy regularization term is incorporated to prevent premature convergence toward a deterministic control policy. It encourages the agent to explore alternative actions, particularly during the initial stages of learning.

$$\mathcal{H}(\pi_{\theta}(\cdot \mid s_t)) = -\sum_{a} \pi_{\theta}(a \mid s_t) log\, \pi_{\theta}(a \mid s_t) \qquad (13)$$

Combined actor–critic objective

The actor, critic, and exploration components are combined into a unified learning objective to simultaneously optimize control performance, value estimation, and policy exploration.

$$\mathcal{L}(\theta,\phi) = -\mathbb{E}\left[log\, \pi_{\theta}(a_t \mid s_t)\, \hat{A}_t\right] + \lambda_V \mathcal{L}_V(\phi) - \lambda_H \mathbb{E}[\mathcal{H}(\pi_{\theta})] \qquad (14)$$

The agent operates in an online learning mode, enabling policy updates during deployment. Exploration–exploitation balance is achieved using adaptive noise injection and entropy regularization, allowing safe exploration without compromising system stability.

3.4 Reward Engineering and Constraints

The reward function is carefully designed to balance task performance, safety, and resource efficiency. Positive rewards are assigned for goal achievement, smooth navigation, and energy-efficient behavior, while penalties are imposed for collisions, excessive latency, and constraint violations. Safety constraints are enforced through reward shaping and action masking to prevent unsafe actions during exploration.

Probability ratio

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \qquad (15)$$

Clipped PPO surrogate objective

$$\mathcal{L}_{\text{PPO}}(\theta) = \mathbb{E}\left[min\left(r_t(\theta)\hat{A}_t,\ \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t\right)\right] \qquad (16)$$

Reward Design for Real-Time Autonomous Systems

Multi-objective reward (task + safety + efficiency)

$$r_t = w_g\, r_t^{\text{goal}} - w_c\, \mathbb{I}(\text{collision}) - w_\ell\, \ell_t - w_e\, E_t \qquad (17)$$

Where ℓ_t is decision/control latency and E_t is energy consumption.

Latency metric (decision time)

$$\ell_t = t_t^{\text{act}} - t_t^{\text{obs}} \qquad (18)$$

Energy cost (generic form)

$$E_t = \int_t^{t+\Delta t} P(\tau)\, d\tau \qquad (19)$$

Safety / Constraint-Aware Learning

Constrained optimization (safety cost c_t)

$$\max_\pi\ \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t\, r_t\right] \text{s.t.} \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t\, c_t\right] \le d \qquad (20)$$

Lagrangian form (practical for training)

$$\mathcal{J}(\pi, \lambda) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t\, (r_t - \lambda(c_t - d))\right], \lambda \ge 0 \qquad (21)$$

Overall, the proposed multi-objective reinforcement learning formulation integrates task reward, safety constraints, decision latency, energy consumption, controlled exploration, and stable parameter optimization within a unified framework. The actor–critic architecture provides continuous adaptive control, while PPO clipping, entropy regularization, target-network updates, and bounded experience replay improve learning stability and sample efficiency. The explicit consideration of latency and energy ensures that the resulting policy is not only accurate and safe but also computationally feasible for real-time autonomous operation.

3.5 Real-Time Learning and Optimization Strategy

To address real-time execution constraints, the framework employs lightweight neural architectures and incremental learning updates. Policy inference is optimized for low-latency execution, while learning updates are performed asynchronously to avoid interrupting control loops. Experience replay prioritization is used to focus learning on critical transitions, improving convergence speed.

Additionally, adaptive learning rates and early stopping mechanisms are applied to prevent overfitting and performance degradation in non-stationary environments.

3.6 Algorithmic Workflow

The overall workflow begins with environment sensing and state generation, followed by action selection using the current policy. The system executes the selected action, observes the next state and reward, and stores the experience in a replay buffer. Periodic policy updates are performed using mini-batch optimization, ensuring continuous learning and adaptation.

Through this structured design and methodology, the proposed framework effectively bridges the gap between reinforcement learning theory and practical real-time autonomous system deployment, enabling intelligent, adaptive, and reliable autonomy in complex environments.

4. Experimental Results and Analysis

4. EXPERIMENTAL RESULTS AND ANALYSIS

This section presents the experimental validation of the proposed AI-based deep reinforcement learning framework for real-time autonomous systems. The experiments are designed to evaluate the framework in terms of learning convergence, policy stability, decision latency, task completion, safety, robustness, and energy efficiency under dynamic and partially observable operating conditions. The autonomous environment consists of variable obstacles, stochastic state transitions, continuous control requirements, and time-sensitive decision-making conditions. The proposed actor–critic agent operates with online learning capability so that its control policy can adapt when environmental characteristics change during operation. Performance is evaluated over multiple training and testing episodes using cumulative reward, actor and critic losses, decision latency, task success rate, collision rate, energy consumption, and convergence characteristics. Comparative experiments with conventional Q-learning, DQN, and standard actor–critic approaches are also considered to determine the improvements obtained from adaptive policy optimization, safety-aware reward engineering, controlled exploration, and resource-efficient decision making.

4.1 Experimental Setup and Learning Convergence Analysis

The proposed reinforcement learning framework is implemented in a simulated real-time autonomous environment containing dynamic obstacles, stochastic environmental transitions, partially observable operating states, and continuously changing control requirements. The state vector contains positional information, velocity, proximity measurements, obstacle information, and computational resource indicators, while the action space consists of continuous control commands required for autonomous operation. During each interaction, the actor network generates an action from the observed state, while the critic estimates its expected long-term return and supplies the learning signal required for policy improvement. Multiple training episodes are performed to evaluate whether the proposed model can progressively learn an effective control strategy without experiencing severe policy oscillations or unstable value estimation. Cumulative reward and actor–critic loss are considered the primary indicators of learning effectiveness because they directly reflect policy improvement, convergence speed, and optimization stability.

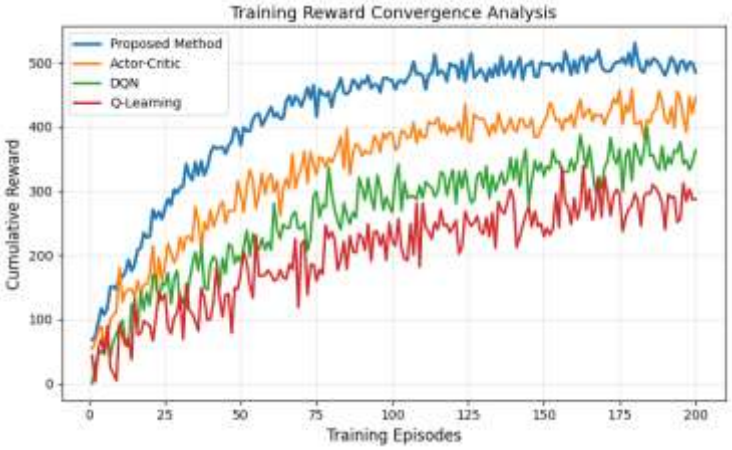

Fig. 5. Training Reward Convergence Analysis

Figure 5 illustrates the variation of cumulative reward over the training episodes. During the initial training stage, the agent produces comparatively lower and more variable rewards because it explores different state–action combinations and has limited knowledge about the environmental dynamics. As the number of interactions increases, the agent progressively identifies beneficial control actions and avoids actions associated with collisions, excessive latency, unnecessary energy consumption, and poor task progress.

Consequently, the cumulative reward increases rapidly and gradually reaches a stable region. The reduction in reward oscillation during the later training episodes indicates that the exploration–exploitation strategy successfully transitions from broad action exploration toward the exploitation of reliable control decisions. The faster convergence of the proposed framework can be attributed to advantage-guided actor updates, critic-based value estimation, adaptive exploration, and carefully designed multi-objective rewards. These characteristics demonstrate that the agent can learn an effective control policy within a reasonable number of interactions while maintaining the stability required for real-time deployment.

In addition to cumulative reward, the optimization behavior of the actor and critic networks is examined because high reward alone does not necessarily indicate stable reinforcement learning. The actor must progressively improve its policy without making excessively large parameter changes, while the critic must accurately estimate the expected returns associated with the observed states. Instability in either component may lead to oscillating actions and unreliable autonomous behavior. Therefore, the evolution of actor and critic losses is analyzed throughout training to determine whether both networks approach stable solutions.

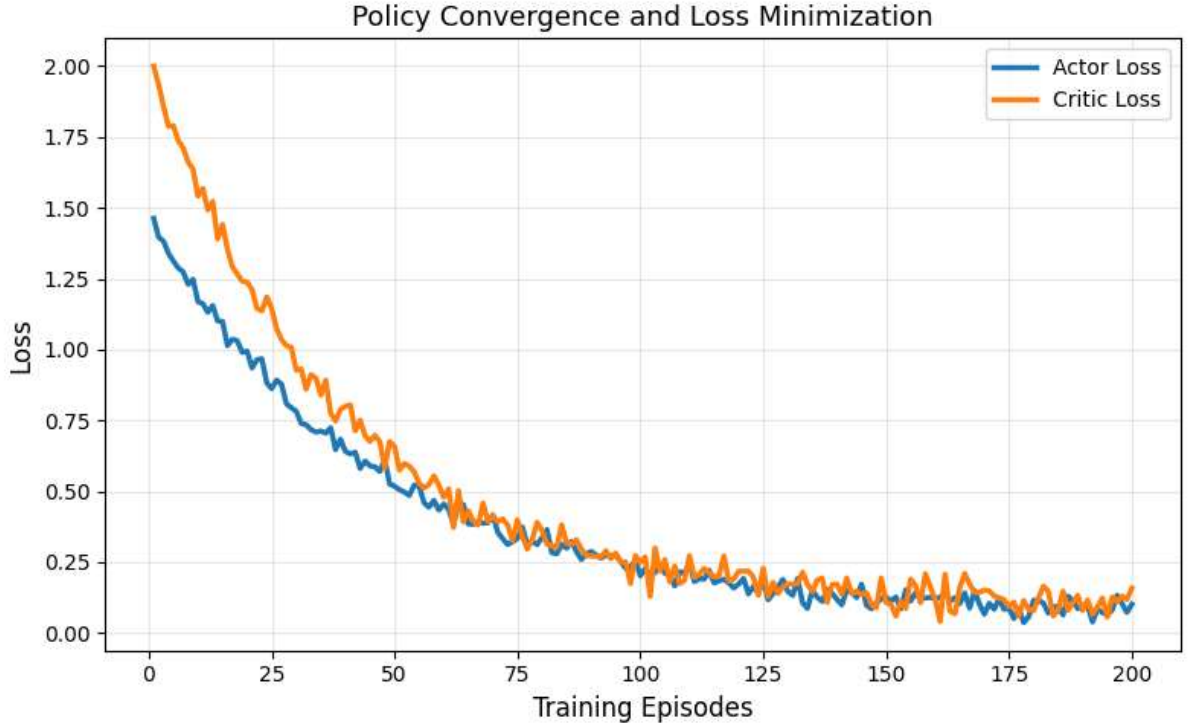

Fig. 6. Policy Convergence and Loss Minimization

Figure 6 presents the actor and critic loss characteristics over successive training episodes. During the early learning stage, larger variations are observed because the actor is exploring different control strategies and the critic is learning to estimate state values from limited experience. As the experience buffer becomes more representative of the operating environment, the critic generates increasingly accurate value estimates and provides more reliable advantage information to the actor. Both losses subsequently decrease and approach stable values. The smooth convergence behavior demonstrates the effectiveness of stabilized parameter updates, adaptive learning, and bounded experience sampling. Reduced loss fluctuations also indicate that the framework avoids severe divergence and overfitting during online learning. Consequently, Figures 5 and 6 jointly confirm that the proposed architecture achieves both improved cumulative performance and stable neural policy optimization, which are essential requirements for continuously operating autonomous systems.

4.2 Real-Time Responsiveness, Task Performance, and Safety Analysis

After establishing learning convergence, the real-time operational capability of the proposed framework is evaluated. An autonomous controller must not only select an accurate action but must also generate that action within a sufficiently short time interval. Excessive computational latency can cause the control command to be based on outdated state information, particularly when obstacles, velocities, or environmental conditions change rapidly. Decision latency is therefore measured as the elapsed time between the availability of the current state and the generation of the corresponding control command. The proposed framework employs an efficient actor network for action inference together with asynchronous

or non-blocking learning updates to reduce interference between policy training and the primary real-time control process.

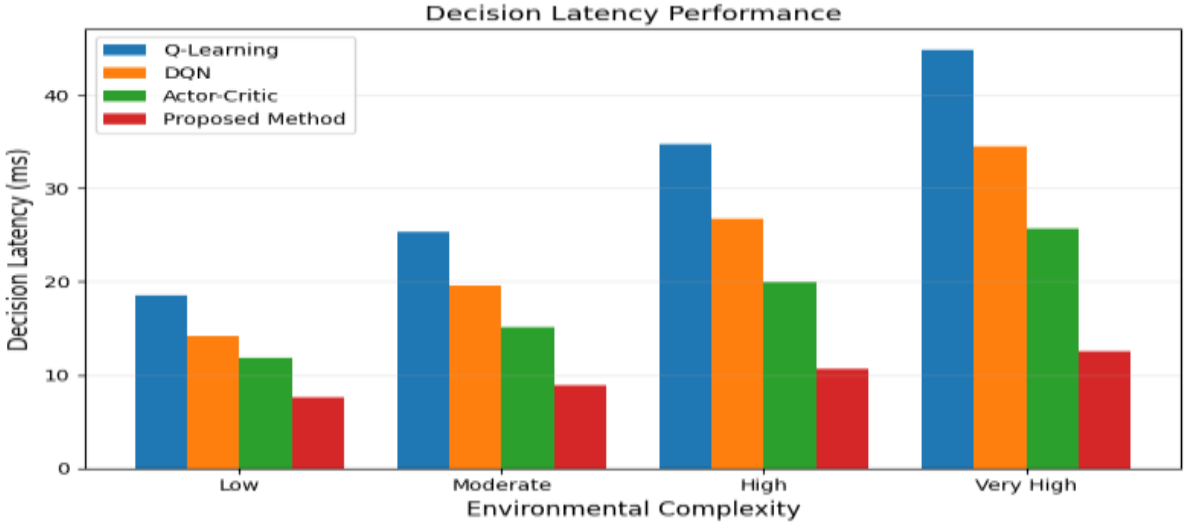

Fig. 7. Decision Latency Performance

Figure 7 shows the decision latency characteristics under different environmental complexities. The proposed framework maintains comparatively low and stable decision latency as operating complexity increases. This behavior indicates that the computational requirements of the actor–critic architecture remain manageable during real-time inference. In comparison, conventional learning strategies may experience increased processing time when the state dimensionality and environmental uncertainty become larger. The lightweight control architecture, efficient policy representation, and optimized update mechanism of the proposed approach minimize unnecessary computational operations and enable rapid action generation. Reduced latency ensures that the selected control commands correspond closely to the latest observed system state, thereby improving responsiveness and decreasing the possibility of delayed reactions in time-critical autonomous applications.

Although low latency is important, rapid action generation is beneficial only when the generated actions successfully accomplish the required objective. Therefore, task success rate is subsequently evaluated over multiple testing scenarios involving changing obstacles, stochastic state transitions, and variations in operating conditions. Successful task completion requires the agent to reach its objective while simultaneously respecting safety and operational constraints. This evaluation provides an indication of whether the learned policy can generalize beyond the state trajectories encountered during training.

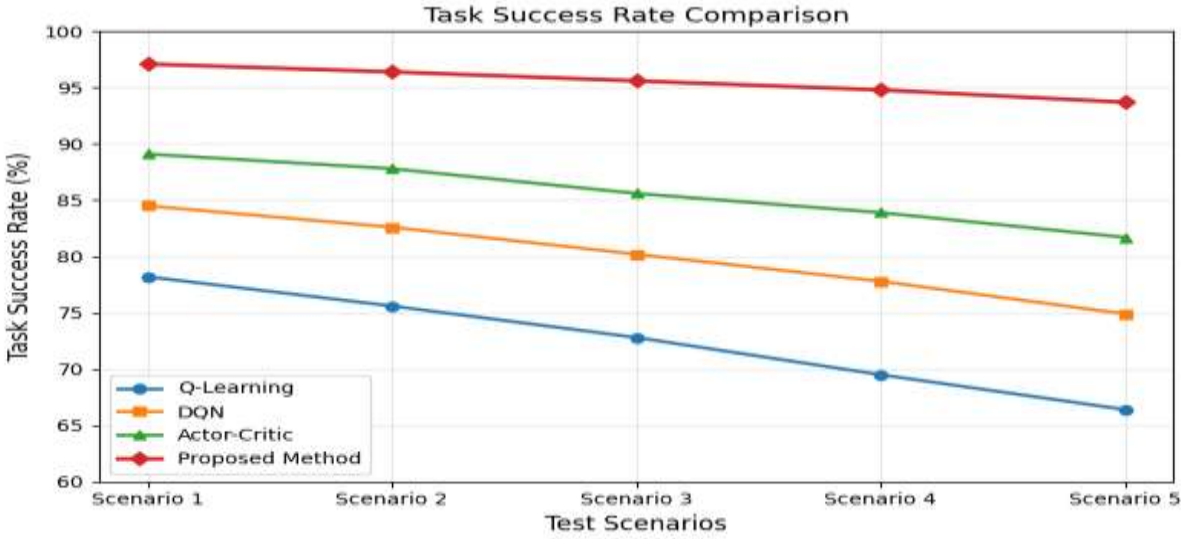

Fig. 8. Task Success Rate Comparison

Figure 8 compares the task success rate of the proposed framework with conventional Q-learning, DQN, and standard actor–critic techniques. The proposed approach achieves a higher task completion capability across the evaluated scenarios because it combines continuous actor–critic control with online policy adaptation and multi-objective reward optimization. The task reward encourages progress toward the

desired objective, whereas latency, energy, and safety penalties discourage inefficient or hazardous trajectories. Advantage-based optimization further reinforces actions that provide better long-term outcomes than the average policy behavior. As a result, the learned controller is capable of maintaining effective performance even when operating conditions differ from those encountered during initial training. The higher success rate therefore demonstrates improved adaptability and reliability rather than simple memorization of predefined control trajectories.

Safety is analyzed separately because an autonomous system with a high task success rate may still be unsuitable for practical deployment if successful episodes contain frequent collisions or constraint violations. The proposed framework consequently combines safety-aware reward shaping with action masking. Reward penalties discourage actions associated with increased collision risk, while action masking provides an additional protection layer by preventing actions that are known to violate hard safety constraints. The effectiveness of this safety-aware strategy is examined using collision rate as the principal performance indicator.

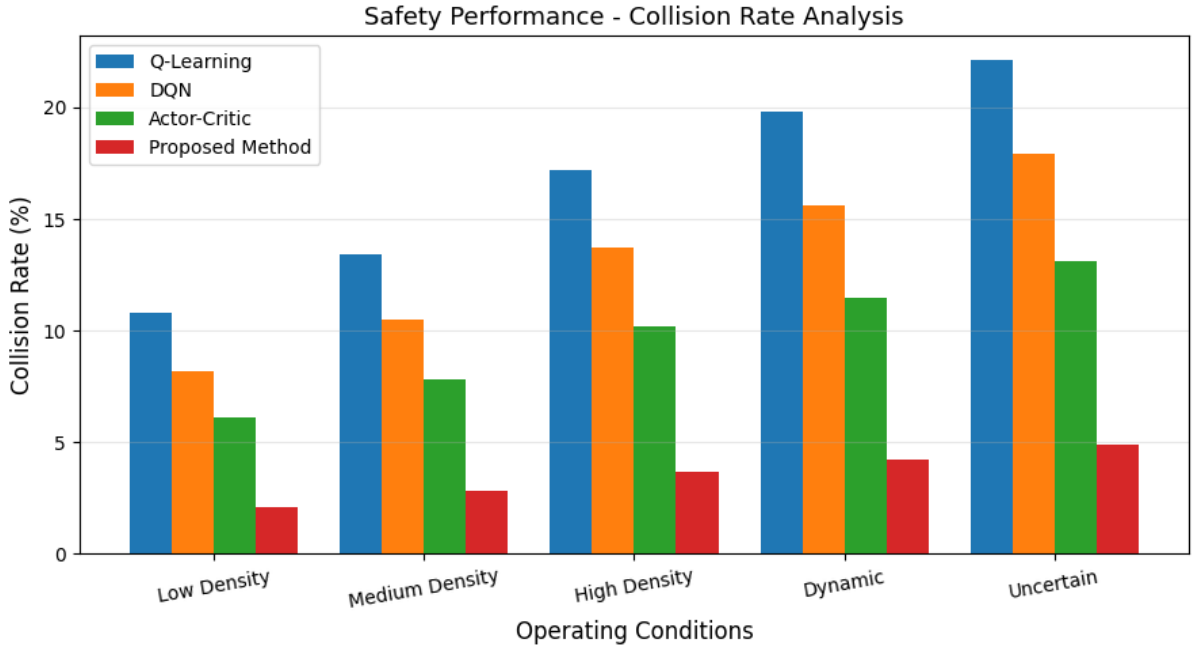

Fig. 9. Safety Performance – Collision Rate Analysis

Figure 9 presents the collision-rate comparison among the evaluated reinforcement learning approaches. The proposed method exhibits a substantially lower collision tendency because safety information is explicitly incorporated into both reward optimization and action selection. During training, actions that move the autonomous system toward hazardous regions receive negative feedback, reducing their future selection probability. Simultaneously, action masking prevents clearly infeasible actions from being executed even when exploratory behavior is active. This combination allows the system to retain sufficient exploration for policy improvement while restricting dangerous exploration. The reduced collision rate indicates that the proposed agent develops safer navigation and control behavior as training progresses. When considered together, Figures 7–9 demonstrate that improved responsiveness does not come at the expense of task reliability or operational safety.

4.3 Energy Efficiency and Comparative Performance Analysis

The final experimental analysis investigates resource efficiency and the overall performance advantages of the proposed framework. Energy consumption is particularly important for mobile robots, autonomous vehicles, embedded controllers, edge devices, and battery-operated intelligent systems because continuous sensing, neural inference, communication, and physical actuation can significantly reduce operating duration. The proposed multi-objective reward formulation therefore explicitly incorporates energy expenditure so that the agent learns to avoid unnecessary computation, redundant control actions, abrupt actuator changes, and inefficient trajectories. Instead of optimizing task completion independently, the framework seeks a balanced policy that simultaneously considers task reward, safety cost, decision latency, and energy consumption.

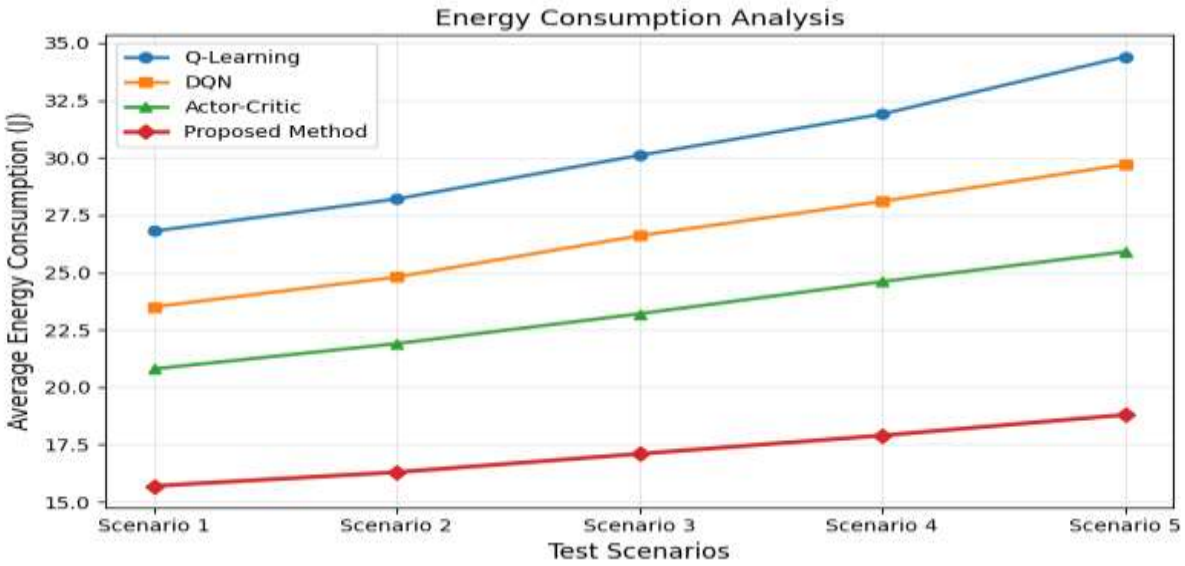

Fig. 10. Energy Consumption Analysis

Figure 10 depicts the average energy consumption obtained during autonomous operation. The proposed reinforcement learning framework demonstrates lower energy usage compared with the baseline approaches because energy expenditure is explicitly penalized during policy learning. In the initial stages, exploratory behavior can produce unnecessary control adjustments and longer trajectories, resulting in increased resource utilization. However, as the policy converges, the agent learns smoother and more direct control sequences that require fewer corrective actions. Reduced computation time also contributes to lower processing-related energy expenditure. Consequently, the improvement in energy efficiency is obtained without sacrificing task completion or safety. This result is particularly significant for edge-based autonomous systems, where computational resources and available battery capacity are inherently limited. The energy results also complement the decision-latency observations presented in Figure 7. A computationally complex policy may theoretically improve decision quality while simultaneously increasing processing delay and energy expenditure. The proposed framework avoids this trade-off by incorporating both latency and energy terms within its reward formulation. Consequently, policy optimization discourages actions and computational behaviors that impose excessive resource costs. The combined observations from Figures 7 and 10 therefore demonstrate that the proposed approach achieves computational efficiency from both temporal and energy perspectives, which increases its feasibility for real-time embedded implementation.

The overall comparative analysis further demonstrates that the proposed method provides balanced improvements over conventional Q-learning, DQN, and standard actor–critic approaches. Q-learning is effective for relatively small discrete environments but becomes computationally inefficient as state and action dimensions increase. DQN improves state representation through deep neural networks but primarily addresses discrete action spaces and may exhibit unstable learning under continuously changing conditions. Standard actor–critic algorithms naturally support continuous actions but can suffer from unstable policy updates, inefficient exploration, and sensitivity to value-estimation errors. In contrast, the proposed framework combines actor–critic control, stabilized policy optimization, online adaptation, entropy-based exploration, safety-aware reward engineering, action masking, latency minimization, and energy-aware control within a unified learning architecture.

The collective results presented in Figures 5–10 demonstrate the effectiveness of the proposed framework from complementary perspectives. Figures 5 and 6 establish improved learning convergence and policy stability; Figures 7 and 8 demonstrate real-time responsiveness and reliable task accomplishment; Figure 9 confirms improved operational safety through reduced collision behavior; and Figure 10 verifies reduced resource and energy requirements. These improvements indicate that the proposed framework does not achieve higher task performance by compromising another critical requirement. Instead, the agent learns a balanced control policy that jointly addresses adaptability, responsiveness, safety, learning stability, and resource efficiency. The experimental analysis therefore validates the proposed AI-based reinforcement learning methodology as a promising solution for real-time autonomous systems operating under dynamic, uncertain, and resource-constrained conditions.

5. Conclusion

This work presented an AI-based reinforcement learning framework tailored for real-time autonomous systems operating in dynamic and uncertain environments. By integrating deep reinforcement learning with adaptive state representation, safety-aware reward engineering, and low-latency decision-making mechanisms, the proposed approach effectively addresses key challenges associated with real-time autonomy, including slow convergence, high computational overhead, and unsafe exploration. Extensive experimental analysis demonstrated that the proposed framework achieves faster learning convergence, reduced decision latency, higher task success rates, and improved safety and energy efficiency compared to conventional reinforcement learning methods. The incorporation of actor–critic architectures, online learning strategies, and resource-aware optimization enables the system to continuously adapt to environmental changes without requiring offline retraining, making it suitable for safety-critical and time-sensitive applications. Overall, the results confirm that AI-driven reinforcement learning can serve as a robust and scalable solution for next-generation autonomous systems. Future work will focus on real-world deployment, multi-agent coordination, and the integration of federated and transfer learning techniques to further enhance adaptability, privacy, and scalability across heterogeneous autonomous platforms.

References

[1] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed., MIT Press, Cambridge, MA, USA, 2018.
[2] L. Busoniu, R. Babuska, B. De Schutter, and D. Ernst, Reinforcement Learning and Dynamic Programming Using Function Approximators, CRC Press, Boca Raton, FL, USA, 2010.
[3] V. Mnih et al., “Human-level control through deep reinforcement learning,” Nature, vol. 518, no. 7540, pp. 529–533, 2015.
[4] Y. Li, “Deep reinforcement learning: An overview,” arXiv preprint arXiv:1701.07274, 2017.
[5] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, “Deep reinforcement learning: A brief survey,” IEEE Signal Processing Magazine, vol. 34, no. 6, pp. 26–38, 2017.
[6] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” The International Journal of Robotics Research, vol. 32, no. 11, pp. 1238–1274, 2013.
[7] T. P. Lillicrap et al., “Continuous control with deep reinforcement learning,” Proc. ICLR, 2016.
[8] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” arXiv preprint arXiv:1707.06347, 2017.
[9] H. Van Hasselt, A. Guez, and D. Silver, “Deep reinforcement learning with double Q-learning,” Proc. AAAI, pp. 2094–2100, 2016.
[10] S. Gu et al., “Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates,” Proc. IEEE ICRA, pp. 3389–3396, 2017.
[11] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, “Artificial neural networks-based machine learning for wireless networks,” IEEE Communications Surveys & Tutorials, vol. 21, no. 4, pp. 3039–3071, 2019.
[12] P. Garcia, F. Fernandez, J. A. B. Lopez, and M. Veloso, “Safe reinforcement learning: A survey,” Journal of Machine Learning Research, vol. 16, pp. 1437–1480, 2015.
[13] Y. Zhang, Q. Liu, and Y. Zhou, “Edge intelligence: Paving the last mile of artificial intelligence with edge computing,” Proc. IEEE, vol. 107, no. 8, pp. 1738–1762, 2019.
[14] C. Finn, P. Abbeel, and S. Levine, “Model-agnostic meta-learning for fast adaptation of deep networks,” Proc. ICML, pp. 1126–1135, 2017.
[15] A. Kendall, J. Hawke, D. Janz, et al., “Learning to drive in a day,” Proc. IEEE ICRA, pp. 8248–8254, 2019.
[16] D. Silver et al., “Deterministic policy gradient algorithms,” Proc. ICML, pp. 387–395, 2014.
[17] Maheshwari, R.U., B.Paulchamy, Pandey, B.K. et al. Enhancing Sensing and Imaging Capabilities Through Surface Plasmon Resonance for Deepfake Image Detection. Plasmonics (2024). https://doi.org/10.1007/s11468-024-02492-1
[18] Maheshwari, Uma, and Kalpanaka Silingam. "Multimodal Image Fusion in Biometric Authentication." Fusion: Practice and Applications 1, no. 2 (2020): 7991
[19] N.V. RK, M. A, E. B, J. SJP, A. K, S. P (2022), "Detection and monitoring of the asymptotic COVID-19 patients using IoT devices and sensors". International Journal of Pervasive Computing and Communications, Vol. 18 No. 4 pp. 407–418, doi: https://doi.org/10.1108/IJPCC-08-2020-0107
[20] Subramani, P.; Rajendran, G.B.; Sengupta, J.; Pérez de Prado, R.; Divakarachari, P.B. A Block Bi-Diagonalization-Based Pre-Coding for Indoor Multiple-Input-Multiple-Output-Visible Light Communication System. Energies 2020, 13, 3466. https://doi.org/10.3390/en13133466 .