Human Interaction and Emerging Technologies (IHIET 2026), Vol. 236, 2026, 149–159
https://doi.org/10.54941/ahfe1008091

Toward Reinforcement Learning for Selection and Parameterization of Appropriate Prevention Measures for Collaborative Robotic Applications

Gustavo Novak[1,2,3], Vincent Weistroffer[1], Jonathan Savin[2], and Richard Bearee[3]

[1]University Paris-Saclay, CEA, List, Palaiseau, 91120, France
[2]French National Research and Safety Institute, Vandœuvre-les-Nancy, 54500, France
[3]Arts et Metiers Sciences and Technologies, LISPEN, Lille, 59046, France

ABSTRACT

The deployment of collaborative applications has emerged as a key trend in the manufacturing industry, driven by the synergy of human intelligence and adaptability with robots' endurance, strength, and precision. However, this closer integration of humans with robots heightens the risk of injuries. To prevent these risks, the risk reduction process is performed during design phase of collaborative workstations, ensuring that safety criteria required by current standards are met. In this work, we formulate the risk reduction process as a sequential decision-making problem and propose a Reinforcement Learning-based approach. In a simulation environment, the RL agent learns to select and parameterize prevention measures to reduce collision and crushing risks of collaborative applications. The considered prevention measures include physical barriers, laser safety barriers, and Tool Center Point (TCP) speed monitoring zones parameterized by their geometric and temporal activation interval parameters. The RL agent is trained through trial-and-error interactions within the simulated environment, aiming to maximize a reward function that prioritizes risk reduction. Secondary objectives include minimizing cycle time and reducing workstation footprint, allowing the system to derive optimized solutions that satisfy safety while balancing cost and productivity constraints. A feasibility demonstrator is presented through a use case in which the RL agent proposes prevention measures to reduce identified hazards for a single collaborative application, enabling faster human-centered design of safe collaborative workstations.

Keywords: Reinforcement learning, Robotics safety, Human-robot collaboration, Risk reduction

INTRODUCTION

The growing use of collaborative robotics heightens the risk of injuries. To prevent these risks, it is crucial to ensure, right from the design phase of collaborative workstations, that safety criteria required by current standards are satisfied.

Central to this challenge is the risk reduction process, a critical and complex aspect of implementing collaborative applications. Human-Robot

Received April 29, 2026; Revised June 24, 2026; Accepted July 8, 2026; Available online August 25, 2026

Collaboration (HRC) safety standards require that risk reduction is performed prior to commissioning an industrial robot system (ISO, 2025). The objective is to deploy prevention measures that reduce identified risks.

This paper presents feasibility demonstrator of a reinforcement learning algorithm designed to enhance safety experts' capabilities by proposing prevention measures during the design phase of collaborative applications.

The remainder of this paper is organized as follows. This first section reviews HRC safety standards and outlines the current industrial risk reduction process. The following sections present the reinforcement learning environment and algorithm, followed by a discussion and perspectives.

HRC Safety Standards

The most relevant standards for risk reduction in collaborative applications include ISO 12100 (ISO, 2011), which outlines the risk assessment and reduction processes, and ISO 10218-2:2025 (ISO, 2025), which specifies safety requirements for collaborative applications. These standards address all categories of hazards (e.g., mechanical, electrical, thermal). However, in this study, a simplification is adopted by focusing exclusively on a subset of mechanical hazards: crushing and collision risks. Accordingly, throughout this paper, the term risk refers solely to these hazards. According to ISO 10218-2:2025, mechanical risks in collaborative applications can be reduced through two primary principles:

- Separation: ensure a safe distance is maintained between humans and the robot while it is in motion.
- Contact limiting: ensure that forces and pressures exerted on humans remain within biomechanical acceptable thresholds (ISO, 2025).

In both industry practice and the literature, collaborative applications based on the first principle are commonly referred to as Speed and Separation Monitoring (SSM) applications, while those based on the second principle are known as Power and Force Limiting (PFL) applications. However, in this paper, the terms "separation" and "contact limiting" applications are used to differentiate the safety principles, as SSM and PFL are considered safety functions rather than prevention principles. Therefore, these acronyms will not be used further in this paper.

Current HRC Risk Reduction Process

In current industrial practices, risk reduction is performed mainly manually by integrators (i.e the multidisciplinary team of experts responsible for designing and deploying the collaborative application) in accordance with internal guidelines and standards previously cited.

The first part of the risk reduction process consists in risk assessment. It relies on three steps. First, reasonably foreseeable mechanical hazards are identified, including collisions or crushing. Second, each hazard is evaluated qualitatively using four factors: severity of harm, frequency of exposure,

possibility of avoidance, and probability of occurrence. These factors are combined using a predefined risk matrix to determine a risk level. Since these matrices are not standardized, results may vary between organizations.

Once all hazards have been assigned a risk level, the risk reduction process is performed. Hazards are to be addressed one by one, in order of risk level. High risks must be eliminated or reduced by design, while moderate risks require recognized technical measures; lower risks may be mitigated through organizational measures or personal protective equipment.

Risks linked to collisions and crushing may be reduced using one of the two primary principles introduced earlier. The separation principle prevents contact through physical or sensing barriers such as fences, light curtains, or laser scanners (Gopinath et al., 2018). The contact-limiting principle, defined in ISO 10218-2:2025, limits force and pressure during contact by monitoring TCP speed, force, and joint parameters (SICK AG, 2018).

As any implemented measures modify the system, the whole process (risk assessment followed by risk reduction) is to be iteratively run until acceptable safety is achieved.

REINFORCEMENT LEARNING PROBLEM

This paper focuses on transforming the risk reduction process into a reinforcement learning problem and solving it using a state-of-the-art RL algorithm (PPO, detailed in the Reinforcement Learning Algorithm section). The problem is formulated as a Deterministic Markov Decision Process (DMDP), defined by the tuple (S, A, P, R) where S denotes the set of states, A the set of actions, P the deterministic state transition function and R the deterministic reward function.

The RL algorithm solves the problem by finding the policy $\pi_{\boldsymbol{\theta}}(a \mid s)$ that maximizes the cumulative reward. An episode corresponds to a finite sequence of interactions between the RL agent and the environment, starting from an initial state and evolving through successive actions until a termination condition is met. During training, multiple environments—i.e., independent simulation instances—are executed in parallel on a GPU, enabling efficient data collection.

Action Space

In this study, for simplicity of implementation, the set of possible actions (i.e., prevention measures) is limited to physical barriers, detection barriers, and TCP speed monitoring safety zones. Physical and detection barriers are modeled as a set of two-dimensional line segments $\{(x_1, y_1), \ldots, (x_n, y_n)\}$. Detection devices are controlled by programmable safety systems, such as safety PLCs, that can enable or disable them as required. Consequently, each detection barrier must be associated with a time interval (t_{start}, t_{end}) during which it is active. In this study, only a single activation interval is considered per detection barrier for simplicity.

Robot TCP speed monitoring safety functions vary across manufacturers; for simplicity, we consider a single axis-aligned cuboid safety zone defined by

two 3D points $\{(x_1, y_1, z_1), (x_n, y_n, z_n)\}$. The TCP speed limit is not treated as an action parameter, as it can be analytically derived from the current state to reduce collision and crushing risks to an acceptable level, as described in the State Transition section.

The three types of heterogeneous actions defined above involve different parameter sets. To represent them, the action space is modeled as a sequential parameterized action space, building on the concept of parameterized actions (Masson et al., 2015), where a discrete action is associated with a real-valued parameter vector. The discrete component a_t allows the agent to select one type of action among the set $\{PB, DB, TCP, P_+, END\}$ where PB and DB initiate the construction of physical and detection barriers, respectively, TCP initiates a TCP speed monitoring zone, P_+ adds a point to the prevention measure currently being defined, and END terminates the action sequence.

The parameters, denoted p, define the geometric or temporal characteristics of each action. Depending on the action type, $p \in \mathbb{R}^2$or $\mathbb{R}^3$, representing either a time interval or 2D/3D spatial coordinates. This sequential formulation enables the agent to build prevention measures with variable structures, such as barriers with an arbitrary number of sides (see Figure 1).

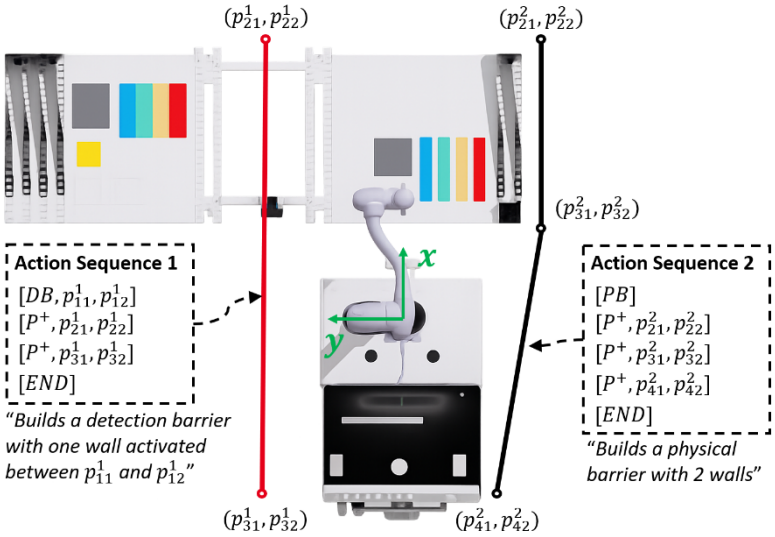

Figure 1: Workstation top view illustration of the process of constructing prevention measures through sequential parametric actions.

State Space

The environment state s is defined by the set of tensors $\{s_{2d}, p_{tcp}, s_{tcp}, s_{as}\}$ where s_{2d} represents a set of stacked 2D grids describing: the Workstation Components (WC), Stopped Robot Occupancies (SRO), Robot Occupancies (RO), Physical Barriers Occupancies (PBO), Detection Barriers Occupancies (DBO), Human Accessible Areas (HAA), Mandatory Human Accessible Areas ($MHAA$) and Risk Levels grids (R_{grid}). Each of these components is a

list of 2D N_H by N_W grids of size N_T where N_T denotes the total number of trajectory frames:

- ***WC*:** Boolean grids indicating areas occupied by static workstation elements (e.g., tables, conveyors).
- ***RO*:** Boolean grids representing the robot's spatial footprint at each nominal trajectory frame, obtained by projecting the 3D robot mesh vertices onto the ground plane.
- ***SRO*:** Boolean grids representing the robot's spatial footprint after a complete safety stop, evaluated at each trajectory frame. For a given t-th trajectory frame, the robot's stopped pose is first estimated from its current state using stopping time and distance charts provided by the manufacturer (Novak et al., 2026). The occupancy is then obtained by projecting the robot mesh vertices at this stopped pose onto the ground plane.
- ***PBO*:** Boolean grids indicating regions occupied by physical barriers, generated via geometry rasterization.
- ***DBO*:** Boolean grids indicating regions occupied by detection barriers, generated via geometry rasterization.
- ***HAA*:** Boolean grids denoting regions that are accessible to humans, accounting for a minimum required clearance for human access.
- ***MHAA*:** Boolean grids specifying areas that must remain accessible to humans, particularly useful in direct collaboration scenarios.
- $\boldsymbol{R}_{\mathbf{grid}}$**:** Floating-point grids representing the spatial distribution of robot surface risks, obtained by projecting the risk value at each 3D mesh vertex onto the ground plane.

The tensor p_{tcp} denotes the robot Tool Center Point (TCP) position at each trajectory frame, while $\mathrm{s}_{\mathrm{tcp}} \in [0,1]^{N_T, N_{TCP}, 6}$ represents the set of TCP speed monitoring cuboid zones active at each trajectory frame, where N_{TCP} is the maximum number of zones activated at a single trajectory frame, and the 6-dimensional zone descriptor concatenates the two bounding 3D points. Finally, $s_{as} \in \mathbb{R}^{N_{max}, 4}$ represents the current sequence of actions being executed by the RL agent, where N_{max} is the maximum number of actions per sequence, and each action is encoded as the concatenation of the discrete action type a_t and the continuous parameters p, where entries corresponding to actions not yet executed are zero-padded.

Once the RL agent executes the *END* action, s_{as} is reset to an empty sequence. The tensors s_{2d}, p_{tcp} and s_{tcp} share common first dimension indexed by the trajectory frame, such that $s_d = \{s_{2d,1}, \ldots, s_{2d,N_T}\}$.

State Transition Function

The transition function defines the logic used to update the environment state based on the executed action. Its primary complexity arises from updating HAA, R_{grid}, and the SRO grids.

Human Accessible Areas are updated based on the configuration of physical and detection barriers. The update is performed by propagating accessibility from the grid boundaries using flood-fill algorithm while enforcing a minimum clearance required for human access.

The Risk Levels grids are computed using the algorithm described in our previous work (Novak et al., 2026) which estimates, for each robot mesh vertex, the risk level associated with each reasonably foreseeable human area and body part. When TCP speed monitoring is activated, the TCP speed is adjusted to ensure that the risk level at each robot mesh vertex remains within acceptable threshold when the TCP lies within the monitored cuboid zone, which in turn affects the SRO grids.

An episode terminates successfully once all risks have been reduced to an acceptable level, and it is considered truncated if the number of executed actions exceed the predefined maximum.

Reward Function

Before defining the reward function, we introduce the relevant notation. The apostrophe symbol ($'$) denotes the value of a variable after the execution of an action. The normalized change in Human Accessible Areas is defined as

$$\Delta HAA = \frac{\sum_{t,h,w}(HAA'_{t,h,w} - HAA_{t,h,w})}{N_T + N_H + N_W}$$

where $HAA \in \{0,1\}^{N_T,N_H,N_W}$ is a binary tensor indicating whether a grid cell (h, w) at the t-th trajectory frame is accessible to humans. This term ΔHAA is always non-positive, since the actions cannot increase the human accessible area.

Another concept is the set of reasonably foreseeable human areas $RFHA \in \{0,1\}^{N_{RF},N_T,N_H,N_W}$. Unlike HAA, which dynamically tracks ground accessibility as prevention measures are added, each of the N_{RF} areas represent a fixed ground region associated with a specific human type (e.g., operator, third party) where human presence is reasonably expected. Only hazards occurring within these areas need to be considered for risk level estimation, since contacts outside them are not reasonably foreseeable and are thus excluded from the risk assessment.

We also consider the normalized change in risk levels, denoted by ΔR, computed from the tensor $R \in [0,5]^{N_{RF},N_B,N_T,N_N}$, where N_B is the number of human body parts, N_N the number of robot vertices. Each element represents the risk level associated with a specific reasonably foreseeable human area, body part, trajectory frame, and robot vertex. The normalized change in risk level is defined analogously to ΔHAA and is also non-positive. Note that R differs from the tensor $R_{grid} \in [0,5]^{N_T,N_H,N_W}$ which represents the maximum risk value at each grid cell, aggregated over all reasonably foreseeable human areas and body parts.

The tensor $PO \in [0,1]^{N_B, N_T, N_N}$ represents the probability of contact between each body part and robot vertex over the trajectory. Its normalized variation, denoted ΔPO, is computed in the same way and is likewise non-positive. To penalize actions that slow down the trajectory and increase the cycle time, we introduce ΔCT which approximates the normalized change in cycle time:

$$\Delta CT = \frac{\sum_t \left(\frac{1}{SR_t^{'}} - \frac{1}{SR_t} \right)}{N_T}$$

where SR_t is the speed ratio relative to original trajectory speed at the t-th trajectory frame. Although approximate, this term captures the overall impact of speed reductions on cycle time.

Finally, let V_{TCP} denote the total activated volume of TCP speed monitoring zones, computed as the sum of the products of activation time and cuboid volume across all monitoring zones. Its variation is denoted ΔV_{TCP}.

The reward function evaluates the quality of an action by encouraging risk reduction while balancing secondary objectives. In particular, the RL agent is encouraged to reduce risk—favoring separation over the contact-limiting principle—while maximizing human accessibility, limiting increases in cycle time, and minimizing the extent of TCP monitoring zones. The reward is defined as:

$$r = -\Delta PO - \Delta R + \beta \left(\Delta HAA - \Delta CT - \Delta V_{TCP} \right)$$

The ΔPO term is incorporated into the reward function alongside ΔR to prioritize risk reduction through separation principle over contact-limiting.

REINFORCEMENT LEARNING ALGORITHM

Proximal Policy Optimization (PPO) algorithm is used to solve the reinforcement learning problem. It was selected for this study because it is a widely adopted policy gradient method in deep reinforcement learning (Schulman et al., 2017). The neural network architecture consists in three components: a state-action transformer, an actor, and a critic. The following sections briefly describe each component of the neural network.

State-Action Transformer

Before selecting an action and parameterizing it, the RL agent must process entire raw state s to decide next action. This is the role of the state-action transformer (Vaswani et al., 2017), which transforms the heterogeneous raw state into two fixed-size tensors z_a and z_s. The first is a representation of the next action and the second a representation of the current environment state.

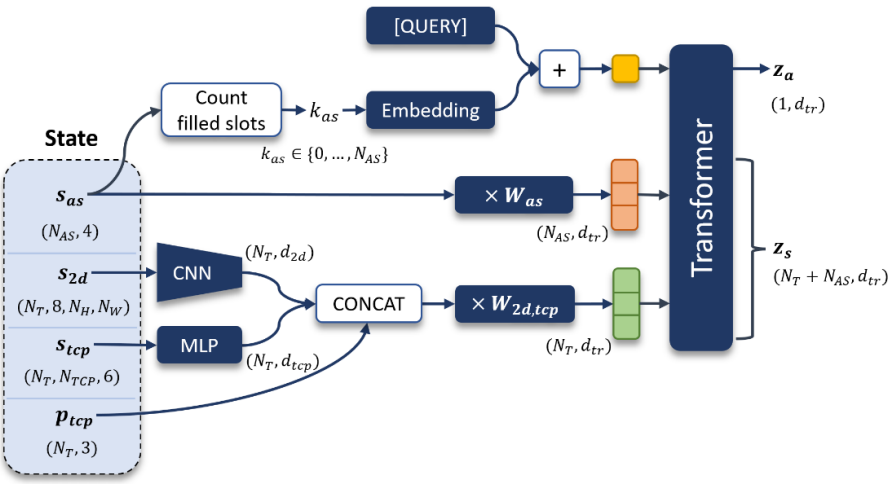

Figure 2: Neural network architecture of the state-action transformer. Components shown in dark blue contain learnable parameters. The terms d_{2d}, d_{tcp} and d_{tr} denote the output dimensions of the CNN, MLP, and transformer, respectively. [QUERY] denotes a learnable query token that interacts with all state tokens and produces the global output π_t.

This design is inspired by the CLS input token mechanism used in BERT (Devlin et al., 2019), Vision Transformers (Dosovitskiy et al., 2021), and Octo (Ghosh et al., 2024), where a dedicated token aggregates a global output representation of the input sequence while the remaining tokens provide local context through self-attention.

Actor and Critic Heads

The actor policy π_{θ} adopts two specialized output heads to handle the parametric sequential action space $\{a_t, p\}$. Given the action representation z_a produced by the state-action transformer, a discrete multilayer perceptron (MLP) head first selects the action type a_t by sampling from a categorical distribution, while an independent continuous MLP head generates the action parameters p by sampling from a normal distribution — both taking z_a as their sole input.

The critic estimates the value function $V(s)$ from the environment state representation z_s through adaptive pooling followed by a dedicated MLP head. Adaptive pooling is preferred over other pooling strategies due to its robustness to input variation and noise (Brothers, 2025), contributing to more stable value estimates and, consequently, more stable PPO training.

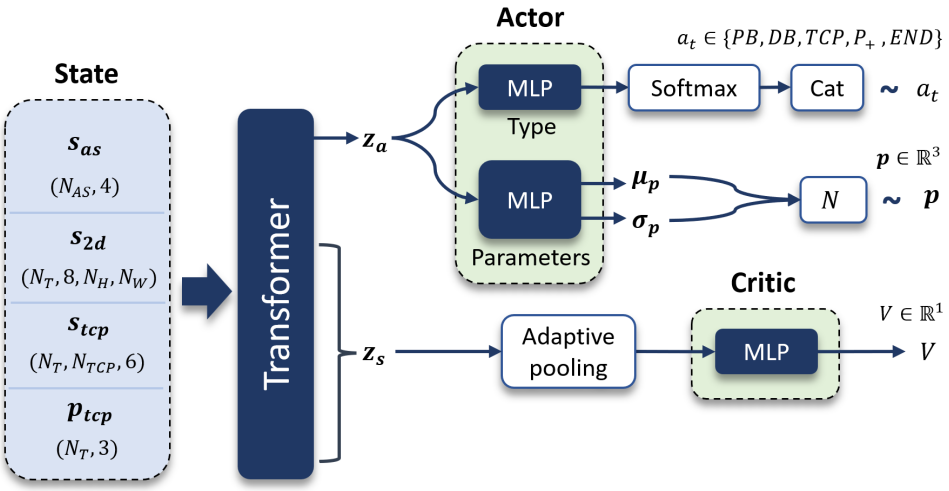

Figure 3: Actor and critic neural networks architectures. The shared transformer is the previously introduced state-action transformer.

Preliminary Results

The flexible workstation used in the proof of concept of our previous work (Novak et al., 2026) was considered in this feasibility demonstration. This workstation can operate in two different scenarios: workspace sharing and direct collaboration. The previous described PPO RL agent is trained to propose a sequence of prevention measures for the direct collaboration scenario that maximizes the cumulative reward. The rewards coefficient multipliers β, α_{PO}, α_{HAA}, α_{CT}, α_V were set to 0.1, 5.0, 5.0, 0.1, 1.0 respectively. These values were manually tuned to obtain the expected behavior: prioritizing risk reduction through the separation principle, while not neglecting secondary objectives such as cycle time and workstation footprint.

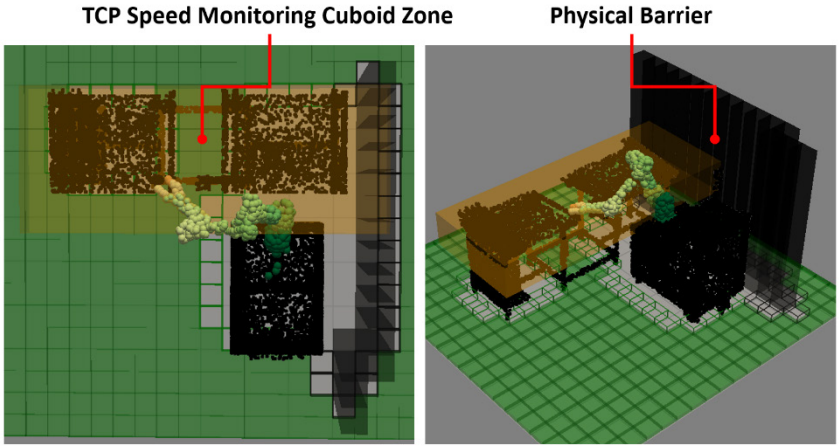

Figure 4: Prevention measures selected and parameterized by RL agent.

During training, the agent converges to a sequence of actions (Figure 4) that includes adding a physical barrier on the right side of the robot, which reduces—using separation principle—the risk of collision between the robot wrist and a third person approaching from the right of the robot, as well a TCP speed monitoring cuboid zone activated the entire trajectory. It reduces the remaining risks of collision and crushing between the robot, third persons, and the operator.

The RL agent is initialized to promote high exploration. The actor type head is initialized with a constant output that assigns equal probability to all action types. The actor parameters head is initialized with the mean at the center of the action space and a high standard deviation, encouraging broad exploration of the parameter space during the early stages of training.

As training progresses, the agent gradually reduces exploration and increasingly exploits action sequences that yield higher cumulative rewards. Consequently, the policy transitions from predominantly stochastic behavior to more deterministic action selection, resulting in an increase in the average episodic cumulative reward (Figure 5). At each parallel environments step, the agent selects and executes one action in each of the 2500 parallel environments.

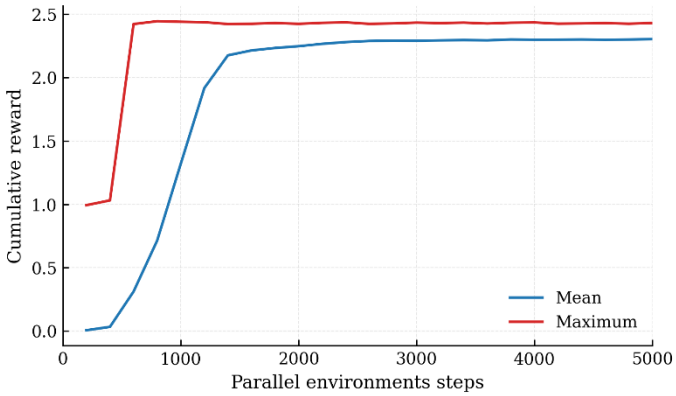

Figure 5: Episode cumulative reward mean and maximum computed across the 1000 latest episodes.

DISCUSSION AND PERSPECTIVES

The algorithm presented in this paper learns a sequence of prevention measures that reduces mechanical risks for a specific collaborative workstation. In this regard, the proposed approach can be viewed as analogous to a classical optimization approach, where the goal is to identify an optimal configuration of prevention measures for a given collaborative application. However, this feasibility study represents only an intermediate step toward a final objective.

The final goal of this research is to develop a reinforcement learning algorithm capable of selecting and parameterizing appropriate prevention measures for collaborative robotic applications in general, rather than

optimizing prevention measures for a single workstation. In such an algorithm, the RL agent would be able to infer suitable prevention strategies across a wide range of collaborative scenarios involving different tasks, layouts, and human–robot collaboration patterns.

Compared with traditional optimization-based approaches, the proposed RL-based approach offers an important advantage: once trained, the RL agent can propose prevention measures with very low inference time.

Future work will therefore focus on improving the generalization capability of the agent. In particular, the training process will be extended to include multiple collaborative robotic applications with varying layouts, tasks, and interaction conditions. By exposing the agent to a diverse set of environments during training, we aim to enable it to learn generalized prevention strategies that can be transferred to previously unseen workstations.

REFERENCES

A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby (2021) An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. International Conference on Learning Representations (ICLR).

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, I. Polosukhin (2017) Attention Is All You Need. Advances in Neural Information Processing Systems (NeurIPS).

G. Brothers (2025) Robust Noise Attenuation via Adaptive Pooling of Transformer Outputs. International Conference on Machine Learning (ICML).

Gopinath, V. Johansen, K. Ölvander, J. (2018) Risk Assessment for Collaborative Operation: A Case Study on Hand-Guided Industrial Robots. In: Svalova, V. (Ed.) Risk Assessment, InTech.

ISO (2011). ISO 12100:2011: Safety of machinery - General principles for design - Risk assessment and risk reduction. (Standard)

ISO (2025). ISO 10218:2025: Robots and robotic devices - Safety requirements for industrial robots - Part 2: Robot systems and integration (Standard)

J. Devlin, M.-W. Chang, K. Lee, K. Toutanova (2019) BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov (2017) Proximal Policy Optimization Algorithms.

M. Masson, P. Ranchod, G. Konidaris (2015) Reinforcement Learning with Parameterized Actions.

Novak, V. Weistroffer, J. Savin, R. Bearee, A. Sghaier, D. Tihay, G. Personeni (2026) PATRAS: Pragmatic Assistive Tool for Mechanical Risk Assessment of Collaborative Applications

Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, S. Levine (2024) Octo: An Open-Source Generalist Robot Policy. Robotics: Science and Systems (RSS).

SICK AG (2018) Safe Robotics – Safety in Collaborative Robot Systems. White Paper, SICK AG.