POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Proactive Human Avoidance in Collaborative Logistics: An RL-Based Dynamic Scheduling Framework

Laurea Magistrale in Automation and Control Engineering - Ingegneria dell'Automazione

Author: Lorenzo Marzi

Advisor: Prof. Paolo Rocco

Co-advisors: Christian Cella, Marco Faroni

Academic year: 2024-2025

1. Introduction

In recent years, the transition toward Human-Robot Collaboration (HRC) has gained significant momentum, aiming to combine human cognitive flexibility with robotic precision. However, standard reactive safety protocols, such as Speed and Separation Monitoring (SSM) [2], frequently cause prolonged idle times and severe throughput degradation by forcing the robot to halt when a human approaches. In response, this thesis develops an intelligent, dynamic task scheduling framework based on Deep Reinforcement Learning (DRL) to optimally manage spatial and temporal coordination in collaborative logistics.

Thesis Purpose While human safety is strictly guaranteed by underlying Speed and Separation Monitoring (SSM) protocols, these reactive systems frequently cause disruptive mid-air halts that degrade productivity. Therefore, the primary objective of this thesis is to design an intelligent robotic orchestrator that restores process fluidity. By modeling the human operator as an autonomous agent governed by a probabilistic Finite State Machine, the robot must dynamically synchronize its schedule with these stochastic movements to proactively avoid triggering safety stops. To effectively navigate this probabilistic environment in real time, this research leverages Reinforcement Learning (RL) to synthesize an optimal, adaptive control policy capable of proactively isolating operations in non conflicting spatial zones.

Thesis Achievements The core contribution of this thesis is the development of a dual-objective robotic orchestrator that successfully balances operational throughput with proactive workspace coordination. To enable this, the collaborative workstation was formally modeled as a Markov Decision Process (MDP) and integrated with a high-fidelity digital twin through a custom software architecture. Driven by the Masked Proximal Policy Optimization (PPO) algorithm, the resulting policy successfully prevents disruptive mid-air safety halts. While a standard greedy baseline achieves a marginally lower theoretical cycle time, it incurs continuous SSM decelerations. In contrast, the RL agent learns proactive spatial coordination specifically *Opposite Side Tasking* and a strategic *Wait* action. This approach preserves a fluid, continuous parallel workflow, reduces mechanical wear, and

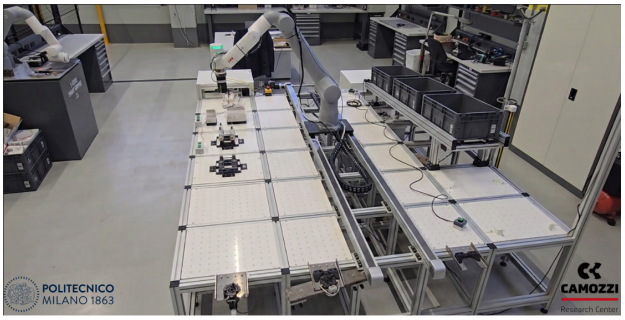
improves ergonomic comfort while maintaining high operational efficiency.

2. State of the Art

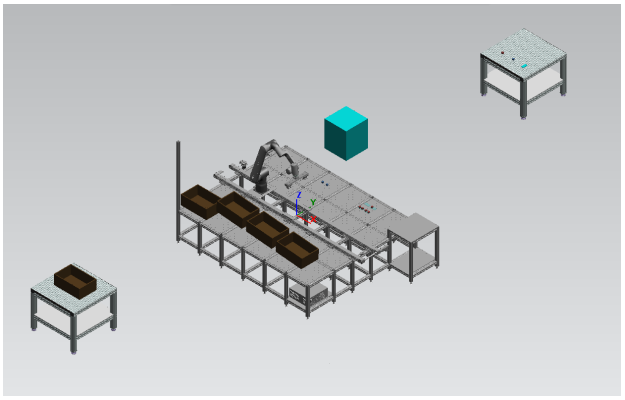
To maintain efficiency under strict safety protocols in Human-Robot Collaboration (HRC), task scheduling is shifting from rigid offline planning to dynamic online execution [3]. Within this shift, single agent Reinforcement Learning (RL) has emerged as a robust tool for fast, real time inference and reactive planning [1].
However, current RL applications primarily focus on collaborative assembly, prioritizing task allocation (deciding *who* performs which task). This thesis addresses a distinct gap by applying these strategies to collaborative logistics. Because task allocation is frequently predetermined in palletizing, the challenge shifts entirely to temporal sequencing and spatial coordination. By dynamically anticipating human movements, the proposed RL orchestrator functions as an intelligent, proactive pre-collision strategy that minimizes disruptive safety halts and ensures continuous process fluidity.

3. System Description

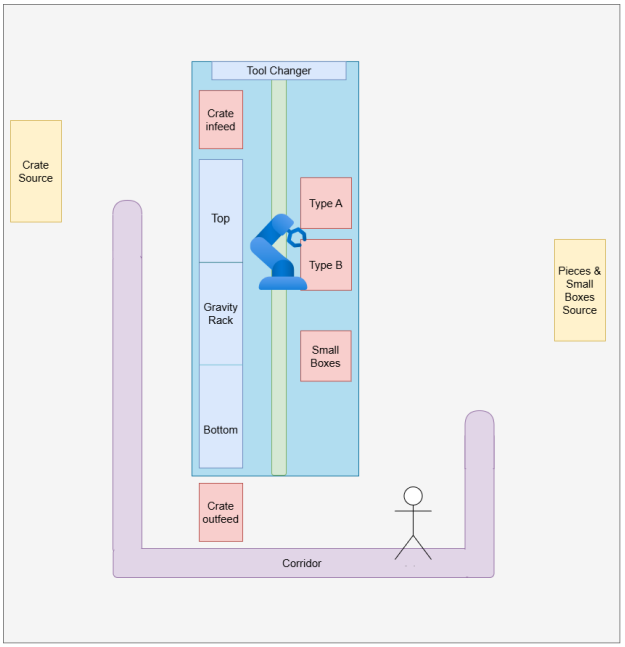
The operational scenario takes place in a collaborative logistics workstation dedicated to material handling, physically realized within the joint laboratory of Camozzi Automation S.p.A. and Politecnico di Milano. The setup features a 7 DoF collaborative robot (a 6 DoF arm mounted on a linear guide) and a human operator sharing a workspace to fulfill a continuous palletizing process.
The workflow requires the manipulation of three primary objects in a hierarchical packaging sequence: *Pieces* (pneumatic components of Type A and Type B), intermediate packaging *Small Boxes*, and macro container *Crates*. To handle these objects of varying dimensions and payload requirements, the robot utilizes an automatic tool changer to seamlessly switch between a specialized Component Gripper and a Crate Gripper.
The division of labor leverages the distinct strengths of both agents. The human operator acts as an autonomous leader, executing flexible tasks such as navigating the complex environment to restock components and manually closing filled boxes. Conversely, the robot performs



(a) Physical workstation in the joint laboratory of Camozzi Automation S.p.A. and Politecnico di Milano.



(b) Tecnomatix Process Simulate digital twin.



(c) Top down view of the workspace.

Figure 1: The collaborative logistics environment. The physical setup (a) is accurately replicated in the high fidelity digital twin (b), where the human is modeled as a kinematic volume (the cube) for training efficiency. The spatial layout (c) provides a clear, top-down schematic of the shared workspace.

the heavy, repetitive pick and place operations. The execution of these operations is governed by strict logical and physical rules. Specifically, a Single Type Constraint dictates that a small box must contain exclusively one type of component (Type A or Type B), and a Homogeneity Constraint enforces that crates can only aggregate small boxes of the identical type. Furthermore, strict precedence and capacity limits apply: a small box can only be transferred to a crate once it is full and manually closed by the operator, and a completed crate can only be removed from the worktable once it reaches maximum capacity.

Beyond these production rules, a critical physical constraint of this shared environment is the underlying Speed and Separation Monitoring (SSM) safety system. This protocol forces the robot to dynamically reduce its speed and eventually halt entirely whenever the human operator breaches the minimum safe separation distance.

To maintain process fluidity and avoid these disruptive SSM halts, the robotic orchestrator must schedule its actions online. The control problem is formalized as a dual objective optimization aimed at balancing operational throughput (efficiency) and human avoidance (social safety). Specifically, the goal is to find the optimal scheduling policy π^* that minimizes a composite objective function $J(\pi)$:

$$\pi^* = \arg\min_{\pi} \mathbb{E}_{\pi} \left[C_{\text{cycle}} + \lambda \sum_{t=0}^{T} \mathcal{P}_{\text{prox}}(s_t, a_t) \right]$$

where C_{cycle} represents the total makespan required to complete the palletizing order, T denotes the episode horizon, $\mathcal{P}_{\text{prox}}(s_t, a_t)$ is a proximity penalty applied when an action a_t violates the safe spatial distance from the human, and λ is a weighting parameter regulating the trade off between productivity and proactive human avoidance. By treating human avoidance as a core performance objective rather than merely a reactive safety constraint, the orchestrator is forced to dynamically adapt its schedule based on the real time state of the workstation.

4. Methodology

To derive the optimal control policy for the robotic orchestrator, the collaborative workstation is formally modeled as a Markov Decision Process (MDP), defined by the tuple $\langle S, A, R, P, \gamma \rangle$. The policy is trained using the Masked Proximal Policy Optimization (PPO) algorithm [5], a state of the art Deep Reinforcement Learning method capable of handling complex, constrained environments.

4.1. Observation Space

Within this environment, the human operator acts as an autonomous, independent agent. To accurately reflect real world variability, human behavior is modeled using a probabilistic Finite State Machine (FSM). The FSM governs the human's stochastic transitions between active logistics tasks, such as line replenishment and manually closing the lid of the small boxes. For simulation efficiency, the human is abstracted as a kinematic volume, which is sufficient to compute the spatial occupancy and absolute distance required for collision avoidance.

The system's observation space (S) is a normalized 63 dimensional vector designed to provide a comprehensive snapshot of the environment at any decision step. It is composed of three subsets:

- **Robot State:** The Tool Center Point (TCP) coordinates, the robot's position along the linear guide, and the currently mounted tool.
- **Line and Process State:** The logistics status, including the fill levels of the staging areas and the specific capacity and types of the small boxes and crates.
- **Human State:** The spatial position of the human operator within the environment, along with an encoded representation of their current target zone.

4.2. Action Space

The action space (A) consists of 9 discrete high level operations, encompassing component transfers, crate handling, tool changes, and a strategic *Wait* action. This deliberate pause serves a dual purpose: it allows the robot to proactively hold its position to avoid spatial interference with the human operator, and it ensures the resolvability of the decision process. If all physical tasks are temporarily restricted, the *Wait* action guarantees that a valid choice is always available to the agent, preventing dead-

locks.
To ensure the RL agent strictly adheres to logical and physical production constraints, dynamic Action Masking is employed. At each decision step, a binary validity mask $M(s_t)$ filters out impossible actions (e.g., attempting to place a component in a full box, or executing a transfer without the correct tool). This mechanism drastically reduces the exploration space, preventing the agent from learning invalid behaviors and significantly accelerating algorithmic convergence.

4.3. Decision Events

To synchronize the RL agent with the continuous industrial simulation, the decision making process is triggered exclusively by specific discrete events rather than at fixed time intervals [1]. The agent observes the environment and selects a new action under exactly two conditions:

1. **Action Completion (Atomic Tasks):** All physical manipulation tasks (e.g., picking a component, placing a box, or changing a tool) are executed as atomic, uninterruptible operations. The agent must wait for the low level controller to completely finish the physical trajectory before a new decision step is triggered.
2. **Conditional Interrupt (Wait Action):** If the agent selects the deliberate *Wait* action, it does not stand idle for a predetermined duration. Instead, the simulation continuously monitors the human's position. The moment the human moves safely away from the active work zone, the wait action is immediately interrupted. This early stop mechanic triggers an instant decision step, allowing the robot to resume operations without wasting time.

4.4. Reward Function Design

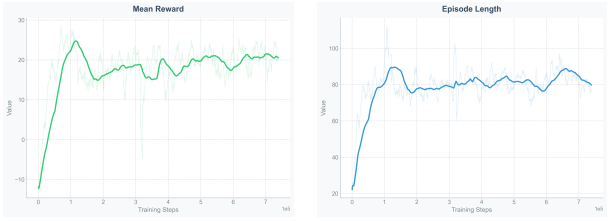
The reward function (R) is designed to balance the dual objective of the thesis. It is formulated as a composite signal comprising four main components:

$$r_t = w_{\text{time}} \cdot t_{\text{duration}} + r_{\text{safety}} + r_{\text{progress}} + r_{\text{penalty}}$$

Where the first term acts as a continuous living penalty to minimize the overall makespan (C_{cycle}), r_{safety} enforces the theoretical proximity constraint ($\mathcal{P}_{\text{prox}}$) via a severe terminal penalty for violating the safe human to robot distance, r_{progress} provides dense positive reinforcement for completing useful logistics tasks, and r_{penalty} discourages inefficient behaviors such as redundant tool changes or excessive waiting.

4.5. Simulation Framework

Training an RL agent requires thousands of interactions, making physical deployment impractical for the learning phase. Therefore, the training pipeline integrates a Python based implementation of Masked PPO (via the Stable Baselines3 library [4]) with a high fidelity digital twin developed in Siemens Tecnomatix Process Simulate.
To bridge the learning algorithm with the simulation, a comprehensive MDP wrapper was developed in C# and .NET by leveraging the simulator's native API. This custom architecture essentially transforms the industrial digital twin into a standard RL environment by implementing dedicated modules for state observation, action execution, reward computation, and episode resets. At each decision step, the Python agent transmits the selected action via standard TCP/IP libraries to the C# interface. The interface executes the robot's physical trajectory within the digital twin while the human's probabilistic FSM runs continuously in parallel, ultimately returning the updated state observation and reward once the next decision event is triggered.

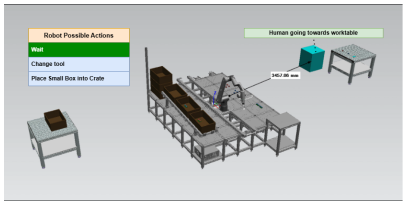


(a) Mean cumulative reward. (b) Mean episode length.

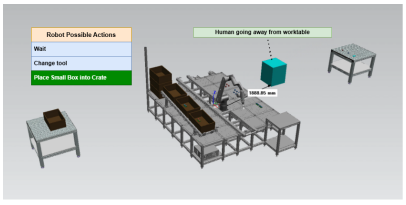
Figure 2: Training metrics: mean cumulative reward (a) and mean episode length (b).

5. Results and Discussion

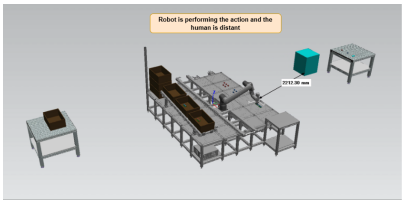
The performance of the trained Reinforcement Learning agent was evaluated quantitatively through its learning metrics, qualitatively by analyzing its emergent behaviors within the



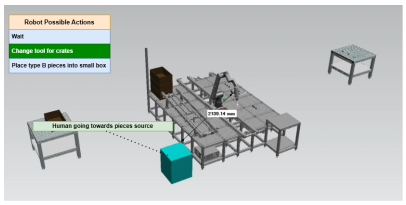
(a) Wait Action: Human approaches.



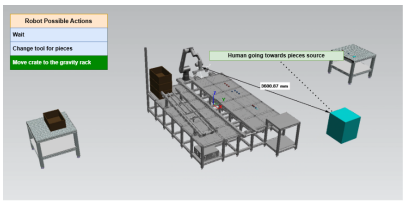
(b) Human retreats; robot selects action.



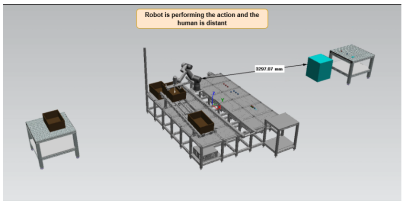
(c) Robot performs action safely.



(d) Robot equips crate gripper as human approaches.



(e) Robot selects task on the opposite (left) side.



(f) Robot executes action safely on the left.

Figure 3: Proactive Spatial Coordination strategies. The top row (a-c) illustrates the strategic *Wait* action. The bottom row (d-f) demonstrates emergent *Opposite Side Tasking*, where the robot dynamically adapts to a human on the right by safely shifting its operations to the left.

Siemens Tecnomatix digital twin, and comparatively against a more reactive baseline. The Masked PPO algorithm was trained for 750×10^3 timesteps. Thanks to the dynamic action masking strictly bounding the exploration space, the agent avoided exploring invalid operations, resulting in a relatively stable convergence.

5.1. Quantitative Convergence

Figure 2 illustrates the agent's learning curves. During the initial training phases, the policy exhibits low cumulative rewards and short episode lengths. This occurs because the untrained agent frequently violates the human safety distance, triggering immediate episode terminations (safety halts). However, as training progresses, both metrics steadily increase and eventually plateau. While this plateau indicates convergence to a local rather than a global optimum, it confirms the agent learned a stable and time efficient policy. The resulting behavior maintains high operational throughput while adhering to the spatial constraints.

5.2. Emergent Behaviors

A visual inspection of the simulated environment confirms that the mathematical convergence translates into logical task execution. As a direct consequence of the severe r_{safety} penalty designed to minimize spatial interference, the robot learns to avoid disruptive Speed and Separation Monitoring (SSM) halts through two proactive strategies:

1. **Strategic Waiting:** If no safe actions are available, the robot utilizes the conditional *Wait* action. Instead of triggering a reactive safety stop and halting mid execution while in mid air, the robot safely holds its home position and continuously monitors the human. It immediately resumes operations the moment the operator clears the shared zone, as illustrated in the top row of Figure 3 (a-c).
2. **Opposite Side Tasking:** This is a significant emergent behavior. When the human approaches a specific section of the assembly line to perform restocking or finishing tasks, the orchestrator proactively selects a task on the opposite side of the workbench. As shown in the bottom row of Figure 3 (d-f), the agent dynamically adapts its tool selection and schedule based on the human's real-time trajectory. This intelligent zone separation maximizes physical distance, preventing safety stops and ensuring a continuous, parallel workflow.

5.3. Proactive vs. Reactive

To quantitatively demonstrate the value of the proposed approach, the RL orchestrator was evaluated against a greedy baseline. This baseline was trained without the proximity penalty in the reward function, effectively representing a traditional reactive system that prioritizes absolute speed and relies entirely on the underlying Speed and Separation Monitoring (SSM) system to prevent collisions.

The comparison, summarized in Table 1, highlights the fundamental trade-off between raw execution speed and process fluidity. While the greedy baseline achieves a faster cycle time by taking the most direct spatial paths, it suffers a severe penalty in fluidity, triggering an average of nearly 20 disruptive SSM safety halts per episode.

In contrast, the proposed proactive RL policy virtually eliminates these sudden mid-air stops through its strategic use of the *Wait* action and *Opposite Side Tasking*. Consequently, while the proactive policy requires slightly more time to complete the order, this trade-off is decisively justified. By preventing continuous deceleration cycles, the orchestrator guarantees a fluid parallel workflow, significantly reduces mechanical wear on the robot's joints, and fosters a more predictable, ergonomic environment for the human operator.

	Time (min)	Safety Halts
Greedy	24.3	19.8
Proactive	26.7	0.5

Table 1: Mean performance comparison between the proactive RL policy and the greedy baseline.

6. Conclusions

This thesis successfully addressed the complex challenge of dynamic task scheduling and spatial coordination within a collaborative logistics workstation. By formalizing the operational environment as a Markov Decision Process (MDP), a Reinforcement Learning orchestrator was developed and trained using the Masked Proximal Policy Optimization (PPO) algorithm within a high fidelity Siemens Tecnomatix digital twin. The strategic implementation of dynamic action masking ensured strict adherence to physical and logical production constraints, significantly accelerating algorithmic convergence.

The simulation results demonstrate that the learned policy successfully optimizes the dual objective problem: maximizing system throughput while minimizing spatial interference with the autonomous human operator. Rather than relying on standard reactive safety controllers that halt production upon human proximity, the orchestrator learned highly proactive spatial deconfliction behaviors. By intelligently utilizing a conditional *Wait* action and employing *Opposite Side Tasking*, the robot dynamically adapts its schedule to maintain a safe spatial separation, thereby preserving process fluidity.

While these results prove the theoretical viability and efficiency of the orchestrator, future work must focus on the Sim-to-Real transfer of the learned policy to the physical 7 DoF robotic workstation. A primary limitation of the current framework is the reliance on explicit human state observations provided by the simulation's probabilistic Finite State Machine. In a real world physical deployment, the human's internal intent cannot be directly queried. Therefore, future research must implement a robust perception layer capable of probabilistic human intent recognition and goal inference based on continuous kinematic trajectories. Bridging this simulation to reality gap is the next step for deploying the orchestrator alongside human operators on a physical factory floor.

References

[1] G. Giacomuzzo et al. Decaf: a discrete-event based collaborative human-robot framework for furniture assembly. In *Proc. of IEEE/RSJ IROS*, pages 7085–7091, 2024.

[2] International Organization for Standardization. ISO/TS 15066: Collaborative robots. Technical report, 2016.

[3] C. Petzoldt et al. Review of task allocation for human-robot collaboration in assembly. *Int. J. Comput. Integr. Manuf.*, 36:1675–1715, 2023.

[4] A. Raffin et al. Stable-baselines3: Reliable reinforcement learning implementations. *JMLR*, 22:1–8, 2021.

[5] C.-Y. Tang et al. Implementing action mask in proximal policy optimization (ppo) algorithm. *ICT Express*, 6:200–203, 2020.