Journal of Manufacturing Systems 83 (2025) 157–174

Contents lists available at ScienceDirect

Journal of Manufacturing Systems

journal homepage: www.elsevier.com/locate/jmansys

Technical paper

Digital twin-based framework for an efficient execution of CPPS reconfiguration through human–robot collaboration

Francesco Giuseppe Ciampi [a,b,*], Thierno M.L. Diallo [b], Faïda Mhenni [b], Jean-Yves Choley [b], Stanislao Patalano [a]

[a] Department of Industrial Engineering, University of Naples Federico II, Naples, 80125, Italy
[b] Euler Laboratory, ISAE-Supméca, 3 rue Fernand Hainaut, Saint-Ouen, 93400, France

ARTICLE INFO

Keywords:
Cyber–physical production system
Reconfiguration
Human–robot collaboration
Digital twin
Physics-informed neural network

ABSTRACT

This paper presents a Digital Twin-based framework to support the reconfiguration process of Cyber–Physical Production Systems (CPPSs) through human–robot collaboration and Industry 5.0 enabling technologies. The proposed approach integrates a Mixed Reality (MR) module into the digital twin architecture to enhance human–machine interaction, data visualisation, and robot programming. It also incorporates Physics-Informed Neural Networks (PINNs), a hybrid methodology that combines machine learning and physical modelling to improve prediction accuracy and physical consistency. A proof of concept implementation of the framework is carried out on the reconfiguration of a real-world production line within a research platform. The communication mechanism between system modules is tested and discussed in detail. Additionally, the use of PINNs for predicting the energy consumption of a mobile robotic system involved in the reconfiguration task is implemented and benchmarked. The mobile robotic system integrates an AMR (Autonomous Mobile Robot) and a Cobot (collaborative robotic arm). Results show that the proposed model outperforms conventional physics-based and data-driven methods, significantly enhancing the predictive capabilities of the digital twin and broadening its applicability beyond the specific use case.

1. Introduction

Manufacturers are under growing pressure to deliver mass personalisation, while simultaneously managing frequent disruptions, labour shortages, and increasing demands for sustainable and ethically responsible production practices. These complex and often conflicting demands are driving a transition towards production systems capable of rapidly adapting to change, while ensuring operational continuity and optimising resource efficiency [1]. Industry 4.0 technologies like Internet of Things (IoT), Edge Computing and Artificial Intelligence (AI) have started this transition, enabling real-time data exchange, distributed intelligence, and greater automation. As these innovations evolved, attention has progressively shifted towards incorporating social and environmental goals. It has led to the emergence of Industry 5.0, which builds upon the digital foundation of Industry 4.0 while emphasising human-centricity, resilience, and sustainability [2], contributing to a broader socio-technological transformation of the industrial landscape [1]. This perspective is reflected in a range of enabling technologies, many of which were already fundamental to Industry 4.0, such as Digital Twins (DTs) and Extended Reality (XR), which play a crucial role in bridging the gap between human operators and cyber–physical systems [3]. Within this evolving context, Cyber–Physical Production Systems (CPPSs) serve as the operational core of modern factories. CPPSs are: *"systems of systems of autonomous and cooperative elements connecting with each other in situation dependent ways, on and across all levels of production, from processes through machines up to production and logistics networks, enhancing decision-making processes in real-time, response to unforeseen conditions and evolution along time"* [4]. From this definition, it is evident that resilience and adaptability are not just desirable but essential characteristics of CPPSs [5]. In particular, the ability to reconfigure CPPSs rapidly and effectively supports the resilience required by Industry 5.0, enabling systems to absorb disruptions, accommodate variability, and adapt to emerging constraints. The reconfiguration can be defined as *"the technical view of the process of changing a system, already developed and in operation, in order to adapt it to new requirements, extend functionality, eliminate errors or improve quality characteristics"* [6]. This process typically unfolds in three phases: identifying the need for change, specifying and planning reconfiguration actions, and executing the necessary modifications [7]. It is also among the tasks where the human role remains most prominent in the management of production systems [8]. While automation

* Corresponding author at: Department of Industrial Engineering, University of Naples Federico II, Naples, 80125, Italy.
E-mail address: francescogiuseppe.ciampi@unina.it (F.G. Ciampi).

https://doi.org/10.1016/j.jmsy.2025.09.001
Received 3 June 2025; Received in revised form 29 August 2025; Accepted 1 September 2025
Available online 13 September 2025
0278-6125/© 2025 The Authors. Published by Elsevier Ltd on behalf of The Society of Manufacturing Engineers. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

plays an important role in supporting reconfiguration, the execution still relies on human operators to perform several critical tasks. The complexity of a CPPS requires personnel with deep expertise in both the physical and digital layers of the system. Moreover, reconfigurations often need to be implemented quickly to avoid extended downtime. Under such time constraints, operators may face pressure and stress, increasing the likelihood of errors, especially in the absence of adequate decision-support tools or proper training [7]. A promising approach to improving the reconfiguration process while upholding human-centric principles is the Human–Robot Collaboration (HRC), one of the key aspects of Industry 5.0 [3]. In this setting, collaborative robots (cobots) work safely alongside human operators, assisting with physically demanding, repetitive, or time-sensitive tasks. This allows human workers to concentrate on higher-level decision-making and system supervision. Most importantly, HRC supports a balanced integration of human skills and automation, reinforcing Industry 5.0's goal of empowering rather than replace human workers [3]. While human–robot collaboration offers clear advantages, it also presents challenges related to interface design, intuitive control, and operator safety. Many of these challenges can be effectively addressed through the use of emerging Industry 5.0 technologies such as Mixed Reality (MR), AI-based decision support, and digital twins [9]. This research then focuses on the design and development of a novel framework that involves human–robot collaboration during the execution of the CPPS reconfiguration. The approach integrates Mixed Reality (MR) as a human–machine interface, offering real-time access to essential data, interactive guidance, and seamless cobot programming capabilities. At the core of the framework lies a digital twin, which is not only interfaced with the physical system, but also with a prediction model based on Physics-Informed Neural Networks (PINNs) [10], for task optimisation and decision-making support. The PINNs are a specialised type of neural network designed to solve scientific and engineering problems by embedding physical laws, typically in the form of partial differential equations (PDEs), directly into the learning process. Unlike traditional machine learning models that rely primarily on data, PINNs incorporate these known equations into their loss functions, ensuring that the model's predictions align with the physical constraints to enhance accuracy and interpretability. The Digital Twin at the core of the proposed framework differs from the CPPS that is the object of the reconfiguration. From a general perspective, while both CPPSs and DTs involve cyber–physical integration, their roles are distinct. CPPSs function as the execution layer, emphasising real-time control and interaction with physical components such as sensors and actuators. In contrast, DTs offer a virtual and intelligent representation of the CPPS, enabling advanced simulation, real time monitoring and data-driven decision-making [11]. In this work, the DT framework supports the reconfiguration process by enhancing operator interaction through Mixed Reality and intelligent assistance.

To test and validate the proposed framework, a proof of concept is developed using a real-world case study on a production line within ALIX (Advanced pLatform for Industry X.0), an automated, modular, and reconfigurable cyber–physical production platform. The scenario focuses on a reconfiguration task involving the physical replacement of machine components and change in the positioning of machines. In its current form, the procedure relies on manufacturer-provided documentation, making it time-consuming and difficult to perform without prior training or external assistance. The proposed architecture is designed with a dual-purpose: optimise the reconfiguration process and the collaboration between human and robot through the use of the PINNs and the digital twin, and provide digital, interactive and non-overwhelming guidelines through an MR hands-free device. The full framework is presented and the development of the PINNs-based models are described in detail analysing a case study in predicting the energy consumption of a mobile robotic system used in the collaborative task. The mobile robotic system consists of an Autonomous Mobile Robot (AMR) and a 6 DoF cobotic arm and, for each of them, a Physics-Informed Neural Network is defined based on robot dynamics.

The remainder of the paper is organised as follows: Section 2 provides a literature review on reconfiguration of cyber–physical production systems, analysing also the role of digital twin, mixed reality and PINNs in industrial applications, Section 3 details the proposed framework and its implementation on a research platform (ALIX), Section 4 presents the case study regarding the reconfiguration of the ALIX production line and the implementation of the PINNs to predict the mobile robotic system's energy consumption, Section 5 discusses the results and findings, comparing the proposed solution with traditional approaches, and finally, Section 6 highlights the conclusions and future research directions.

2. Literature review and research gaps

The literature review has been divided into three macro areas: reconfiguration of CPPSs, human–robot collaboration using Industry 5.0 enabling technologies, and application of Physics-Informed Neural Network. Finally, the thread of these three aspects is presented together with the innovative contributions of this work.

2.1. Reconfiguration of cyber–physical production systems

The growing demand for adaptability in modern manufacturing has intensified interest in the reconfigurability of Cyber–Physical Production Systems (CPPSs). CPPSs are often described using the 5C architecture, which structures their development into five hierarchical levels: Connection, Conversion, Cyber, Cognition, and Configuration [12]. When fully integrated, these levels enable real-time data collection, analysis, decision-making, and adaptive system reconfiguration. The Configuration level, in particular, involves Self-X properties: self-adjustment, self-configuration, and self-optimisation, and is widely regarded as the most challenging to implement [13]. Müller et al. [7] formalise the reconfiguration process into three main phases: identifying the need for reconfiguration, planning the new system configuration, and executing the necessary changes. In subsequent works they introduce a model for self-organised reconfiguration, but their focus remains primarily on the first two phases, with execution largely left to manual intervention [8]. Similarly, Macherki et al. [14] assess several algorithmic approaches for self-reconfiguration, including Multi-Agent Systems and Satisfiability Modulo Theories, but align reconfigurability with selecting a new configuration, without addressing its practical deployment on the shop floor. Hengstebeck et al. [15] also note that industrial reconfiguration is still heavily reliant on expert knowledge-driven manual processes. Although they propose a digital assistant to support decision-making, it does not facilitate or automate the execution phase. Napoleone et al. [16] explore how CPS technologies can enhance reconfigurability by categorising their functions across various technology groups. Particularly relevant to the execution phase are the use of simulations and machine learning for knowledge extraction from data, digital twins for system monitoring and decision-making support, and AR/VR technologies for supporting human–machine interaction during actuation. While the paper focuses on enhancing reconfigurability, these capabilities suggest strong potential to support also the execution phase of the reconfiguration. Overall, while the literature offers a solid foundation for understanding reconfiguration in CPPSs, it remains skewed towards planning over execution. However, integrated frameworks based on Industry 5.0 technologies, such as collaborative robotics, may offer valuable support in this area, as they have already demonstrated effectiveness in related domains like assembly [17], where many of the practical and cognitive challenges of executing reconfiguration are also present. This will be discussed in more detail in Section 2.2.

2.2. Human-robot collaboration and industry 5.0 enabling technologies

Human-Robot Collaboration (HRC) is a key aspect of Industry 5.0, combining human cognitive skills with robotic repeatability and precision. However, its industrial application face challenges such as ensuring safe interaction, flexible robot programming, effective interfaces, and collaborative workspace design [18–20]. Currently, three specific safety standards guide industrial HRC practices: ISO 10218-1:2025 [21] and ISO 10218-2:2025 [22] for safety requirements of industrial robots, and ISO-TS 15066:2016 [23] for collaborative robotics. Several review articles provide a general overview of human–robot collaboration in industry [18,20,24–26]. These works highlight a range of challenges, including ensuring human safety through reliable perception and control systems, managing task allocation between humans and robots, achieving intuitive and effective communication interfaces, and addressing the integration of collaborative systems into existing production environments without disrupting workflows or compromising efficiency. They also present some of the enabling technologies of industry 4.0 and 5.0 that can be exploited to enhance the HRC. This technological dimension is the focus of the work from Zafar, Langås, and Sanfilippo [9], that reviews technologies enabling HRC in Industry 5.0, including cobots, Digital Twins (DTs), AR/VR, and AI, emphasising their role in smart, sustainable manufacturing. It analyses the introduction of these technologies with reference to three main features that the collaborative application must possess: perception, control and actuation. While each technology contributes individually to improving collaboration, the authors argue that it is their integrated application that yields the most significant impact. Finally, the study discusses the main challenges associated with this integration, including cybersecurity, implementation costs, skill gaps, lack of standardisation, ethical considerations, and resistance to organisational change. Similarly, Baratta et al. [27] analyse the use of digital twin in human–robot simulation, ergonomic studies, and interoperability, and discuss key development platforms such as Gazebo (ROS), Unity3D, and Delmia (3DX). In a subsequent study [28], they introduce a multi-simulation Digital Twin architecture using AnyLogic and Process Simulate to enhance task allocation in HRC for assembly processes. Other relevant contributions include Zhu et al. [17], who propose a Digital Twin-driven multi-objective optimisation method using an NSGA-II algorithm to support dynamic task allocation and reconfiguration in HRC workcells, and Ma et al. [29], who develop a Digital Twin-based framework for environment-adaptive task allocation, leveraging genetic algorithms for optimal task distribution. Zhang et al. [30] integrate Digital Twins with HRC assembly systems to enable human detection and tracking, enhancing safety and collaboration efficiency. Similarly, Wang et al. [31] propose a deep learning-enhanced Digital Twin framework for synchronised virtual–physical interaction, improving HRC safety and performance through efficient object detection. Another technology used to improve HRC in industry is the Extended Reality (XR), often combined with a digital twin. The term Extended Reality (XR) is a broad term encompassing the Reality-Virtuality Continuum, which includes Augmented Reality (AR), Mixed Reality (MR), and Virtual Reality (VR). AR overlays digital elements onto the real world without full integration, MR blends digital and physical environments for real-time interaction, and VR immerses users in a fully digital space, replacing the real world [32]. However, in literature the distinction between AR and MR is not always clear, and the terms sometimes overlap, especially when AR applications include interactive spatial elements. This study therefore refers to both terms. A first reference for the integration of AR and digital twins in industrial scenarios is provided by C. Li et al. [33] who showcase a system integrating AR and DTs for teleoperation and robot coordination, demonstrating high efficiency. C. Zhang et al. [34], instead, present a Human–Cyber–Physical Assembly System (HCPaS) framework using AR and DTs to improve safety and reduce errors in HRC assembly. C. Li et al. [35] propose a safe HRI approach combining AR, DTs, and AI-based planning for collision avoidance, validated through a prototype. Liu et al. [36] propose an MR framework for industrial collaboration, integrating a cloud–edge-terminal architecture to improve performance. The MR collaboration application is incorporated into the Digital Twin system for real-time monitoring and control in mining. Finally, Wang et al. [37] propose a Digital Twin-based method for adaptive task allocation in human–robot collaborative assembly, leveraging transfer reinforcement learning. The approach integrates augmented reality, simulation, domain randomisation, and knowledge distillation to enable efficient and flexible policy learning in dynamic assembly environments. While the reviewed works offer comprehensive frameworks that advance HRC, mainly through machine learning models and traditional simulation tools, they predominantly focus on assembly operations. Applications addressing the execution phase of reconfiguration remain largely unexplored. Nevertheless, these studies provide valuable insights into the key challenges of human–robot collaboration and highlight the potential benefits of integrating Industry 5.0 enabling technologies.

2.3. Application of Physics-Informed Neural Networks (PINNs)

The complexity of the industrial environment is still one of the biggest challenges that the emergence of AI in industry is facing. Interdependent systems, data uncertainty and robustness requirements, in fact, make classical data-driven models underperform, while traditional physics-based models need large computational resources and are not scalable or flexible [38]. This has led to the definition of some hybrid solutions that can provide the benefits of machine learning models, such as simplicity and speed, without requiring too many data, or losing robustness and physical validity [39]. The integration of these models within digital twins in industry is mentioned by Runkana et al. [40], who highlight the importance of hybrid DT integrating physics-based models, machine learning, and domain knowledge. This paper refers specifically to the Physics-Informed Neural Networks [10], that in recent years have gained a lot of notoriety. Yang et al. [41], indeed, investigate the use of PINNs for digital twins. PINNs demonstrate scalability for complex physics and outperform single-fidelity methods in accuracy and uncertainty quantification, advancing the realisation of robust DT systems. Focusing on industrial applications, many recent works relate to energy system modelling, safety and reliability analysis, and dynamic system modelling and control [42,43]. Regarding specific applications in Human-Robot Collaboration, X. Yang et al. [44] and X. Yang et al. [45] analyse the use of PINNs in modelling collaborative robot joint dynamics during physical Human-Robot Interaction (pHRI). The results highlight the PINN's adaptability to broader dynamic systems. Ciampi et al. [46] explores the use of Physics-Informed Neural Networks (PINNs) to enhance Human-Robot Collaboration (HRC), presenting a case study in collision detection. The model achieves better accuracy and interpretability with less data compared to classical neural networks. Lee [47] explores the use of PINNs to enhance collaborative tasks in contactless delivery involving multiple agents. The proposed method is integrated in a digital twin system and its effectiveness is validated through experimental results and comparisons with existing machine learning methods.

2.4. Research gaps and research contributions

Despite significant advancements in CPPS reconfiguration, human–robot collaboration (HRC), and the integration of advanced AI techniques into digital twins, key research gaps remain. This paper addresses two primary gaps identified in the literature:

- **Execution of CPPS Reconfiguration through Human–Robot Collaboration:** existing research on CPPS reconfiguration predominantly focuses on the first two phases: identifying the need for change, and specifying and planning the reconfiguration actions, while the execution phase remains largely manual and

insufficiently addressed. This results in several limitations, including extended lead times, increased reliance on human operators, and reduced overall efficiency. Although digital twins and extended reality technologies have been explored to enhance human–robot collaboration in industry, there is a lack of frameworks leveraging these technologies to support and optimise the execution phase of a reconfiguration process.

- **Integration of Physics-Informed Neural Networks (PINNs) into DT-based framework for human–robot collaboration:** while digital twins have been increasingly adopted in industrial scenarios, their predictive capabilities still rely on conventional machine learning models or physics-based simulations. The integration of PINNs into digital twins represents a promising solution, yet its application in Human–robot collaboration and task optimisation remains under-explored. Moreover, existing studies involving PINNs often lack a detailed discussion on key aspects such as the selection of appropriate physical models, the integration of the physical model into the neural network, and the tuning of hyper-parameters. In this research, the motivation for incorporating a predictive model stems from one of the main constraints in deploying the proposed framework: the necessity to optimise battery usage of the mobile robotic system involved in collaborative tasks. Therefore, a reliable tool is needed to accurately predict and optimise the robot's energy consumption.

To bridge the identified research gaps, this study introduces a Digital Twin (DT)-enabled framework for supporting the execution phase of CPPS reconfiguration through Human–Robot Collaboration (HRC). The proposed architecture integrates Mixed Reality (MR) to create an immersive interface layer, enabling human operators to interact with the DT for functionalities such as robot programming, real-time monitoring, physics-based simulation, and interactive procedural guidance during reconfiguration tasks. The system architecture is further augmented with Physics-Informed Neural Networks (PINNs) to enhance the predictive capabilities of the DT, embedding domain-specific physical laws (e.g., robot dynamics) directly into the learning process. This allows for data-efficient and physically consistent predictions. The framework is developed as a proof of concept within a research-oriented production line, where the PINN model is used to estimate the battery usage of a mobile robotic systems during collaborative operations, to support energy-aware task allocation and execution.

3. Material and methods

This section presents and discusses a novel framework to enhance the effectiveness of the reconfiguration process of a cyber–physical production system through Human-Robot Collaboration. The proposed architecture leverages key enabling technologies of Industry 5.0, specifically digital twins, mixed reality, and artificial intelligence, to foster a more sustainable, human-centric and resilient manufacturing paradigm. The section presents first the architecture of the proposed framework, its main components, and a brief introduction to Physics-Informed Neural Networks, then focuses on the implementation of the framework onto a real system.

3.1. The proposed architecture

The proposed framework, shown in Fig. 1, revolves around three main components: a physical system, a digital environment, and an operator using a Mixed Reality (MR) device. The physical domain consists of a real-world collaborative workspace, incorporating human operators, collaborative robots (cobots), and cyber–physical systems. This environment serves as the operational domain where the reconfiguration process take place, with direct interactions between human workers and automated systems. The physical environment continuously generates data which are transmitted to its digital counterpart for analysis and optimisation. The digital environment functions as a virtual representation of the physical workspace, encompassing all relevant process elements and enabling real-time bidirectional data exchange. It integrates simulation capabilities to predict system behaviour, evaluate different operational scenarios, and optimise workflows before execution. A key innovation of this framework is the incorporation of Physics-Informed Neural Networks (PINNs), which enhance the accuracy of system modelling by combining data-driven learning with prior physical knowledge. This integration allows the digital environment to perform advanced task optimisation, generate predictive insights, and assess KPIs with improved reliability. Additionally, the digital environment serves as a central hub for process adjustments and real-time programming. The operator, through the use of a Mixed Reality device, plays a pivotal role as the primary decision-maker, interfacing with both the physical and digital environments. The MR interface enables real-time monitoring, step-by-step procedural guidance, and enhanced situational awareness without obstructing the operator's engagement with the physical workspace. This integration supports a more intuitive and efficient workflow, ensuring that human expertise remains central to the reconfiguration process while leveraging the advantages of collaborative robotics and digital intelligence. The connections between the physical environment, digital environment, and the MR device on the operator establish seamless data exchange, process control, and collaborative execution of reconfiguration tasks. The key interactions include:

- **Real-time data acquisition and synchronisation:** the physical environment continuously generates data from different sources. This information is transmitted to the digital environment, where it is processed for simulation, analysis, and predictions. The synchronisation ensures that the digital representation remains an up-to-date mirror of the physical workspace.
- **Simulation and prediction analytics:** the digital environment leverages real-time data to perform simulations. By integrating PINNs, it refines predictions and enhances task optimisation. These insights are transmitted to the operator to assist in decision-making and operational adjustments.
- **User commands**: through the MR interface, the operator modifies operational parameters, requests simulations or predictions, and indirectly command the cobot actions.
- **Online programming of cobots:** the digital environment facilitates direct control over collaborative robots based on operator input. Via the MR interface, the operator triggers modifications in robot programming, that the digital twin transmits to the cobots in the physical environment, enabling real-time task execution and adaptive behaviour in response to evolving process requirements.

The interaction of the operator within the physical environment is also a critical aspect of this framework. The operator, in fact, performs a collaborative task within the physical environment while maintaining continuous interaction with the MR interface. It is therefore crucial that the interaction between all elements is safe and smooth. The proposed framework can also contribute to human-factor modelling by integrating tools that explicitly account for human perception, decision-making, and ergonomics within the reconfiguration process. The Digital Twin environment can be extended to include representations of human operators, enabling the simulation and evaluation of motion trajectories, visibility, reachability, and ergonomic postures before execution. This allows proactive assessment of human task feasibility and physical strain, supporting safer and more efficient task planning. Moreover, the use of Physics-Informed Neural Networks (PINNs) offers the potential to model not only robot dynamics, but also to predict human motion or intent integrating physical constraints [48]. Finally, Mixed Reality (MR)

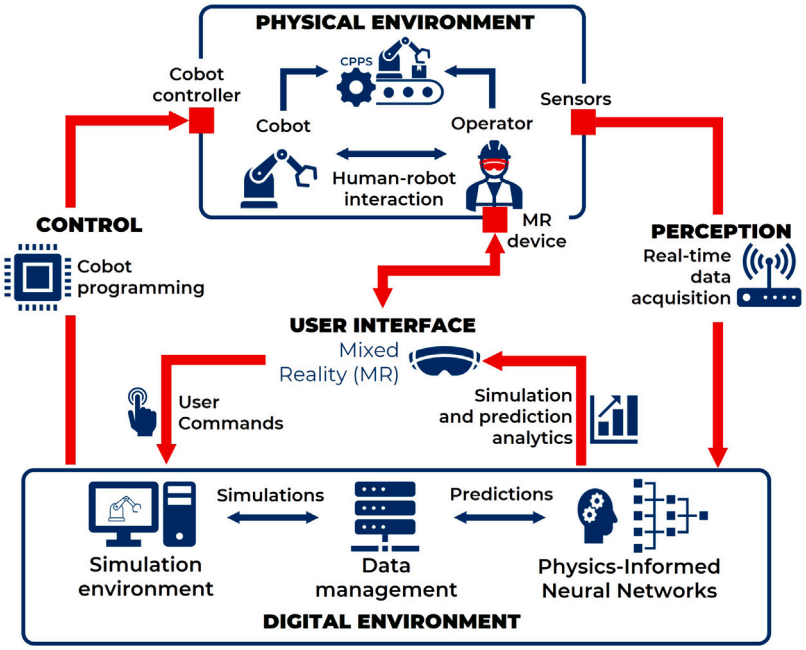

Fig. 1. The proposed framework integrating digital twin, mixed reality and PINNs.

interfaces facilitate intuitive and context-aware human–machine communication, providing visual overlays, haptic cues, and real-time feedback that reduce cognitive load and improve operator awareness. This aligns with prior research showing that well-integrated Human-Robot Collaboration (HRC) systems can outperform manual-only solutions in terms of efficiency, task reliability, and human well-being [49].

3.1.1. The Physics-Informed Neural Networks

Physics-Informed Neural Networks (PINNs), first introduced by Raissi et al. [10], represent a hybrid modelling approach that combines traditional machine learning techniques with physical principles. Unlike conventional data-driven models that rely solely on empirical observations, PINNs integrate known physical laws into the learning process. This integration allows for improved generalisation, especially in scenarios where data is limited, noisy, or difficult to collect. Incorporating prior knowledge into machine learning involves three critical design choices: the selection of the learning model, the method for representing the physical knowledge, and the strategy for integrating it into the training process [50]. Each combination of these elements can result in distinct modelling outcomes. The standard, or "vanilla", form of PINNs employs a feed-forward artificial neural network (FFNN), where the physical knowledge is embedded directly into the loss function using governing equations. Fig. 2 shows the architecture of the vanilla PINNs.

Physics-Informed Neural Networks (PINNs) are designed to solve problems governed by general non-linear partial differential equations (PDEs). These equations can be described as in Eq. (1) [51]:

$$\mathcal{F}(u(z);\gamma) = f(z), \quad z \in \Omega, \tag{1}$$

where $\mathcal{F}$ denotes a non-linear differential operator, $u(z)$ is the unknown function of interest, γ represents the physical parameters, and $f(z)$ is the known function. The domain $\Omega \subset \mathbb{R}^n$ is defined over a space–time coordinate vector $z = [x_1, \ldots, x_{n-1}, t]$. In addition to modelling the governing equations, prior physical knowledge can be exploited through boundary or initial conditions as in Eq. (2):

$$\mathcal{B}(u(z)) = g(z), \quad z \in \partial\Omega, \tag{2}$$

where $\mathcal{B}$ is the boundary operator, $g(z)$ defines the known constraints, and $\partial\Omega$ is the boundary of Ω. To approximate the solution $u(z)$, a neural network $\hat{u}_\theta(z)$ is used. The parameters θ describe the neural network and the training process searches for the optimal set of parameters θ^* that minimises a loss function that integrates contributions from the PDE residuals, boundary conditions, and supervised data as shown in Eq. (3) :

$$\theta^* = \arg\min_\theta \left(\omega_\mathcal{F}\mathcal{L}_\mathcal{F}(\theta) + \omega_\mathcal{B}\mathcal{L}_\mathcal{B}(\theta) + \omega_D\mathcal{L}_D(\theta)\right) \tag{3}$$

where $\omega_\mathcal{F}, \omega_\mathcal{B}, \omega_D$ are the weights reflecting the relative importance of each term.

3.2. Implementation of the proposed framework onto the ALIX production line

ALIX (Advanced platform for Industry X.0) facilities set up in our laboratory served as a test-bed for validating the proposed reconfiguration framework. This research platform is an automated, modular and reconfigurable cyber-physics production platform. It consists of a cockpit, an Autonomous Mobile Robot (AMR), another AMR integrated with a 6 DoF cobotic arm, a packaging station, a connected assembly and quality control station and a distribution and measurement (power and compressed air) box. Fig. 3 shows the items of interest for this study: the two production machines and the mobile manipulator. As the production machines are switched off during the reconfiguration, they are considered as static parts of the environment and the data exchange with the physical environment is only related to the mobile robotic system. The reference mobile robotic system has a MiR100 as driving component. The MiR100 (Mobile Industrial Robots A/S, Odense, Denmark) is an autonomous, collaborative mobile robot designed for material transportation in industrial environments. The MiR100 is equipped with lidar sensors, 3D cameras, and ultrasonic sensors, enabling real-time environment mapping, obstacle avoidance, and enhanced depth perception. It uses differential drive wheels for agile navigation and integrates wheel encoders and IMUs to ensure precise localisation and motion control. The MiR100 is integrated with an UR5e robotic arm, constituting a mobile manipulator. The UR5e (Universal Robots A/S, Odense, Denmark) is a six-degree-of-freedom (6-DOF) manipulator that offers high precision, repeatability, and ease of integration. The UR5e includes a built-in force/torque sensor, joint encoders, and safety features that support human–robot collaboration, in

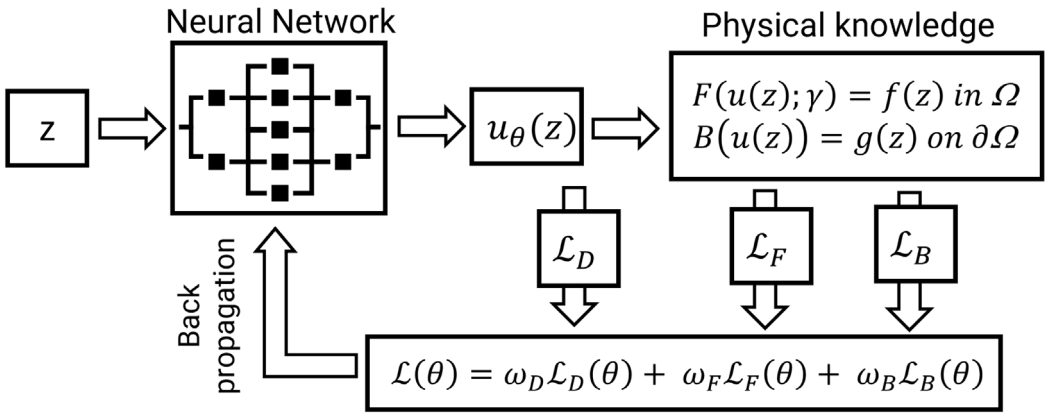

Fig. 2. Architecture of the Physics-Informed Neural Networks (PINNs).

Fig. 3. Machines of the ALIX production line.

compliance with ISO 10218-1:2011 and ISO/TS 15066:2016 standards. The robotic arm is powered directly by the MiR's onboard supply and integrated into its control system. This integration allows the MiR to autonomously trigger predefined UR5e programs, enabling coordinated tasks such as navigating to a station and performing pick-and-place operations without requiring external control input.

This section aims to provide the necessary details for integrating the proposed framework onto a real system, demonstrating the feasibility of implementation, and analysing the data exchange processes. The scope, however, is not to provide a comprehensive description of all modules, but rather a focus on some of the main features that enable integration and communication between them. The implementation, although tested in the ALIX environment, is rather general as it is done using commercial software and common tools for commercial robots.

3.2.1. Development, integration and communication

The proposed framework comprises two primary software components: the Mixed Reality (MR) module and the digital environment. The MR module implementation involves the graphical user interface (GUI) design and the communication protocols. In contrast, the digital environment developments centres on the definition of the simulation environment, the communication with the physical system and MR module, and the integration of Physics-Informed Neural Networks (PINNs).

The Mixed Reality (MR) module is essential to keep the operator at the centre of the decision-making process and to ensure that all the necessary information for task execution is provided. Developed in Unity 6000.39.f1 using C# for interaction handling, the application is deployed on Microsoft HoloLens 2. After exporting the project as a Universal Windows Platform (UWP) build, deployment is completed via Visual Studio. Fig. 4 shows the main menu of the MR application.

The MR module development is divided into front-end and back-end aspects. The front-end focuses on the Graphical User Interface (GUI), while the back-end handles data acquisition and management. Although a complete GUI design is beyond the scope of this work, two main features are discussed. First, the use of the Unity Mixed Reality ToolKit (MRTK), an open-source framework designed for HoloLens development. MRTK provides pre-built components such as hand tracking, voice commands, and spatial UI elements like holographic buttons and sliders. These features support rapid prototyping and ensure compatibility with the HoloLens platform. The UI elements are spatially interactive, allowing users to reposition or anchor menus, improving usability in industrial settings where unobstructed views and spatial awareness are essential. On the other hand, with regard to the visualisation of the simulations, it is achieved through a spatial rendering of a virtual mobile manipulator, embedded in the real environment. The rendering mirrors the behaviour of the robot in the digital twin through the real time collection of joint positions. Fig. 5 shows a screenshot of the testing phase for the manipulator.

On the back-end, the MR module communicates with the digital environment using two protocols: TCP-IP for sending user commands and MQTT for receiving continuous data. Both are implemented in

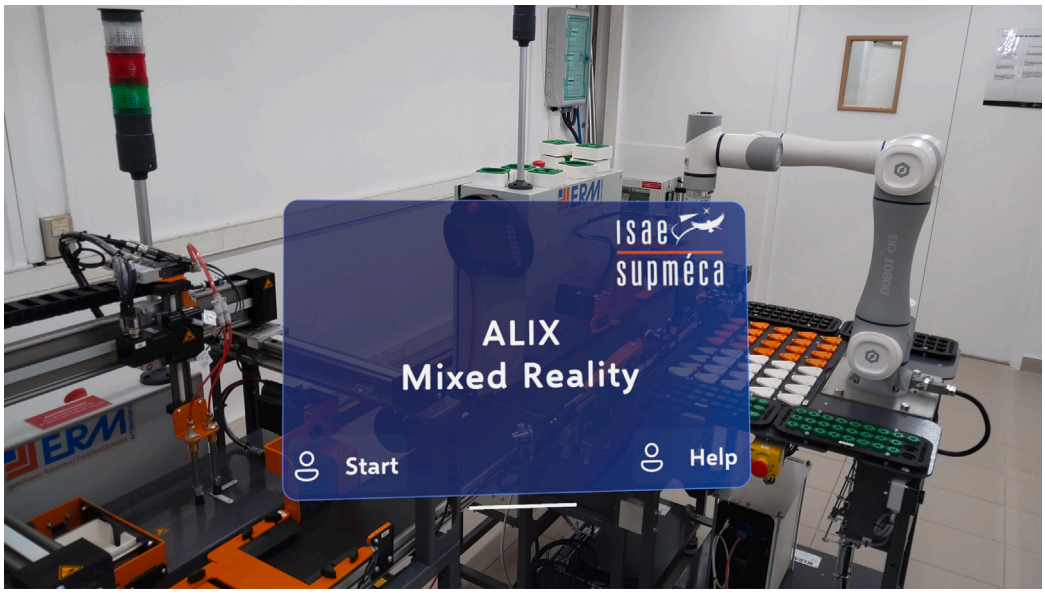

Fig. 4. Main menu of the Mixed Reality (MR) application.

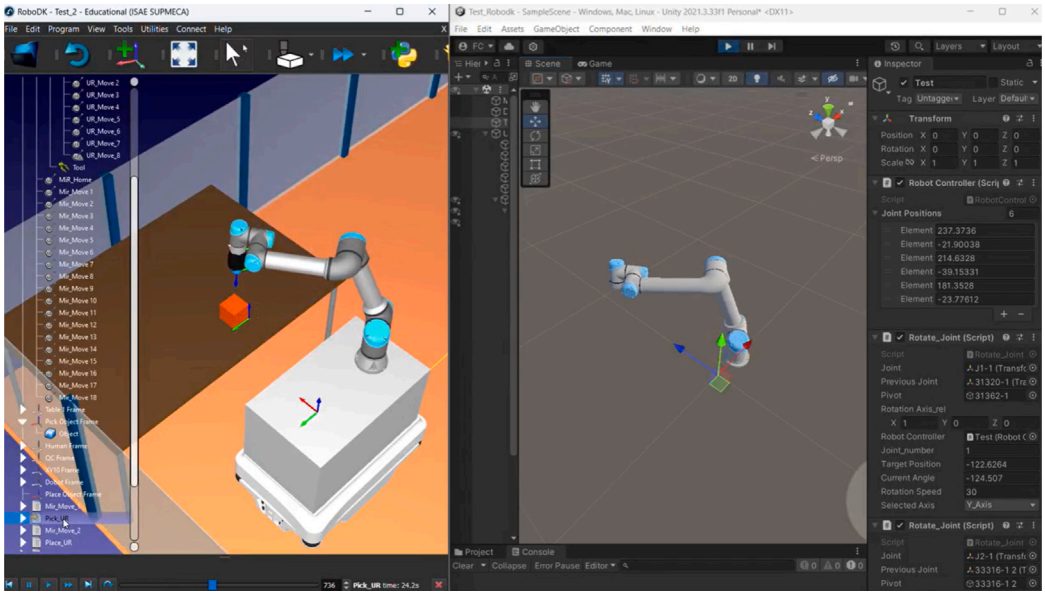

Fig. 5. Rendering of the manipulator motion.

C# scripts. The TCP-IP protocol enables real-time communication from Unity to RoboDK by establishing a client–server connection, where the digital environment acts as the server. The C# script initialises a TCP client and stream messages. It includes also error handling to detect and log issues (Script 1). For data reception, an MQTT client subscribes to a specific topic and receives JSON-formatted data from the digital environment (Script 2).

Concerning the digital environment, it has two important functions: enabling simulation through a virtual replica of the physical system and acting as middleware for integrating models and managing communication protocols. RoboDK 5.9 is used to implement this environment, primarily due to its support for collaborative and mobile robots and its embedded Python IDE. The Python environment enables the development of custom communication protocols, integration of predictive models (e.g., PINNs), and automation of robot tasks. The virtual replica (Fig. 6) is built by importing 3D models following a hierarchical criterion: priority is given to pre-built RoboDK models, followed by manufacturer-provided assets, and finally, custom-developed CAD models when necessary. The behaviour of the mobile robotic systems is modelled in RoboDK using two pre-built mechanism: one for the AMR and the second related to the 6 DOF collaborative robotic arm, implemented as a child element of the AMR to ensure correct movements. The use of pre-built mechanisms allows faster modelling and realistic simulation of kinematics, but does not support robot dynamics, requiring the use of additional module such as the Physics-Informed Neural Networks to expand the functionalities and achieve an high-fidelity digital twin. Task programming within the environment is modular, where complex procedures are decomposed into basic motion primitives (e.g., move, pick, place), simplifying task configuration and execution.

On the digital environment side, communication with the MR module is managed through Python scripts implementing both TCP/IP and MQTT protocols. For TCP/IP, the script initialises a server within RoboDK, listens for incoming user commands, and triggers corresponding actions (Script 3). The MQTT script, instead, continuously collects scene

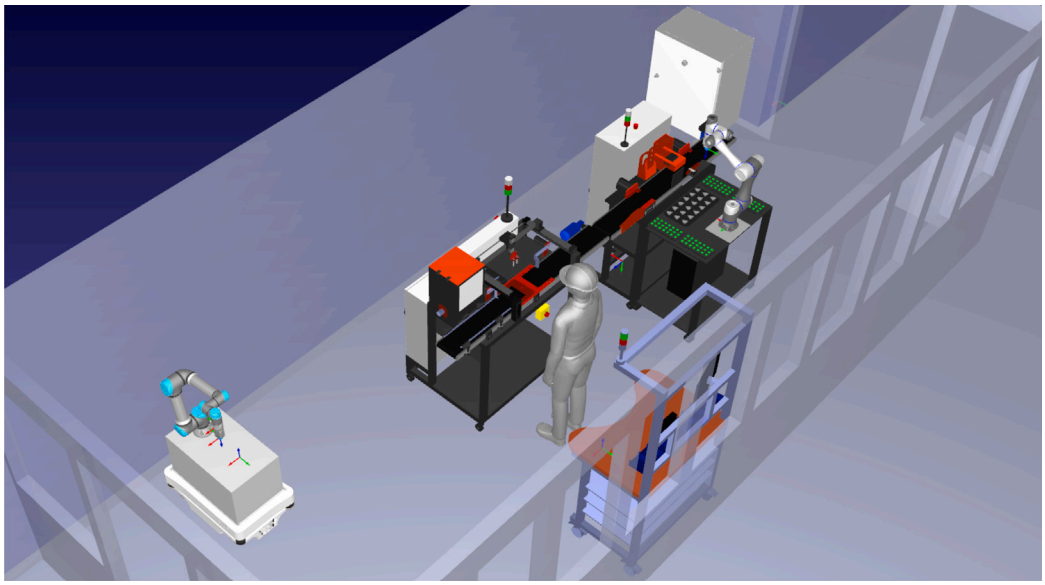

Fig. 6. Digital environment in RoboDK.

data, packages it, and publishes it to a predefined topic at regular intervals (Script 4). Additionally, RoboDK's Python API is leveraged for the integration of Physics-Informed Neural Networks (PINNs), enabling the collection of simulation or real-robot data for model training or real-time inference directly within the environment (Script 5). Finally, communication with the physical system is handled through the MiR100's onboard Wi-Fi router, enabling remote access via its RESTful API. This interface uses standard HTTP methods (GET, POST, PUT, DELETE) to exchange JSON-formatted data for real-time monitoring and control. Authentication is managed via Basic Authentication with encoded user credentials. Through the API, it is possible to monitor robot status (e.g., position, battery level, state), manage missions (e.g., initiating or cancelling tasks via mission IDs), handle navigation and mapping (e.g., querying or updating maps and way-points), and retrieve error logs and event history (Script 6). the MiR API also allows the launch of predefined UR5e programs. However, program modifications or data management of the UR are not possible from the MiR interface. Therefore, a direct connection to the UR is required to enable the data retrieval and on-line programming. The UR5e supports multiple communication protocols; in this work, the Real-Time Data Exchange (RTDE) protocol is tested for high-frequency TCP/IP communication. RTDE enables real-time monitoring of robot state, sensor inputs, and triggering of events such as gripper actions (Script 7). Data collection from both the UR and MiR follows the same logic across the MR and digital environments, despite differing tools and languages. However, online programming is exclusively handled by the digital environment. User commands are routed through RoboDK, which connects directly to the robot controller. Using the RoboDK API, a Python script invokes the virtual robot, its driver, and the corresponding program (Script 8), enabling execution in simulation, on the physical robot, or synchronously in both.

Finally, Fig. 7 summarises the implementation of the framework and all communications onto the ALIX facilities. The Mixed Reality module interfaces with the digital environment for visualisation and control. The digital environment acts as a middleware, coordinating simulation, predictions, data exchange, and robot communication. Communication is secured over a dedicated Wi-Fi network using standardised protocols.

To ensure robustness during execution, the framework includes multiple mechanisms to handle unexpected events or communication failures. Communication between the Digital Twin, Mixed Reality interface, and robotic systems is monitored, and automatic reconnection is triggered in case of disconnection. Additionally, real-time alerts are issued to the operator through the MR interface. Safety features are embedded in the robotic agents: the AMR is equipped with obstacle detection and collision avoidance strategies, and the manipulator is designed in compliance with ISO 10218-1 and ISO/TS 15066 standards for collaborative operation. Moreover, both the AMR and the manipulator are equipped with emergency stop functions. As the system follows a human-centric design, the operator retains full operational authority and can make informed decisions during abnormal conditions, supported by the visual and contextual feedback provided by the MR interface.

4. Case study

This section presents first the context of the reconfiguration of the ALIX production line, then proposes a case study on the optimisation of the mobile manipulator's energy consumption based on the use of Physics-Informed Neural Networks. Finally, the implementation details for PINNs are presented.

4.1. The reconfiguration of the ALIX production line

The ALIX production line described in Section 3 has two operating modes:

- The **Assembly mode** produces a planetary gearbox, consisting of a case, a ring gear, a carrier frame, three planetary gears and a sun gear. The product has two variants based on the reduction ratio.
- The **Packaging mode** produces a box containing a defined number of pots. The product has three main variants: a box of four large pots with personalised lid, a box of six tall pots with personalised over-lid, and a box of six shallow pots with personalised over-lid.

The reconfiguration process of ALIX involves mainly changes in the composition and implementation of the two production machines to switch from one operating mode to another. The change of the variant in production (e.g., from shallow to tall pots) is usually done by modifying some parameters at software level and without physical intervention on the machines, therefore leading to a reduced downtime. On the other hand, the change of the operating mode (e.g., from assembly to packaging) involves the replacement of some parts of the machines. Given the specificity of the process, it is therefore necessary to provide additional detail on the two reference machines:

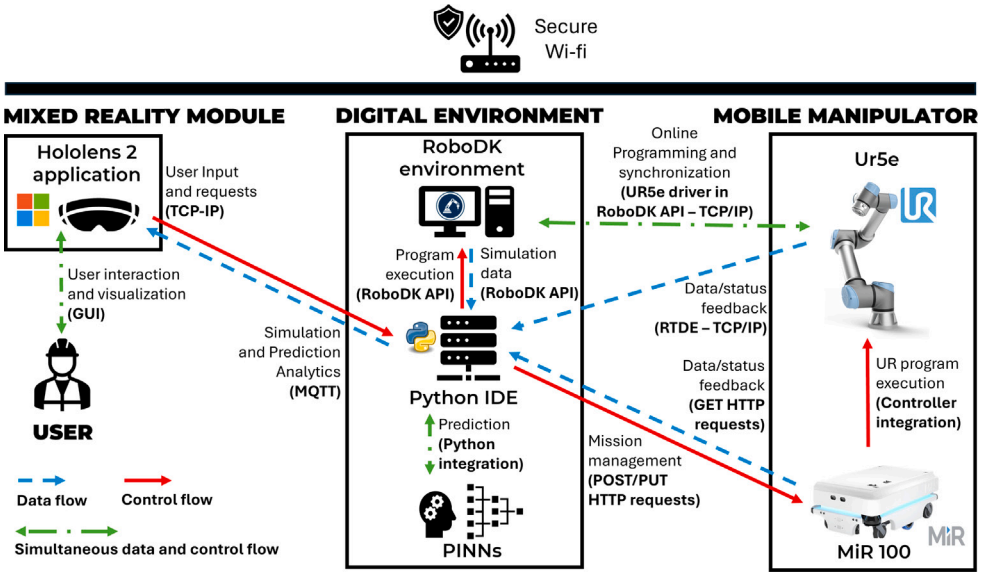

Fig. 7. Implementation of the proposed architecture and communication protocols.

- The first production machine is constituted by a Cartesian robot equipped with a double ended vacuum gripper, a conveyor, a vertical depot, pneumatic actuators, and RFID generator. The machine places the ring gear, picked from the conveyor, inside the case, picked from the vertical depot. In packaging mode, the robot places the pots, picked from the conveyor, inside the box, picked from the vertical storage. The pneumatic actuators handle the queue and proper positioning of products, while the RFID generator writes the RFID code on the case.
- The second production machine is constituted by a 6 DOF robotic arm equipped with a single ended vacuum gripper, multiple storages, a conveyor, a camera, pneumatic actuators, and an RFID reader. In assembly mode, the machine reads the RFID and the robot, based on the variant in production, picks the right carrier frame, planetary gears and sun gear from the storages and places them inside the semi-finished product constituted by the case and the ring gear. In packaging mode, the robot picks the lids and over-lids from the storages and places them on the pods. The camera enables the detection of faulty products, while the pneumatic actuators handle queuing, proper positioning of the product and disposal of defective products.

The reconfiguration, at the current stage, must be performed by a human operator using digital manuals, presenting a considerable down-time and requiring extensive knowledge of the system. An example of the reconfiguration process for the Cartesian robot is presented in Table 1. It describes the operations to change from packaging to assembly mode. It also shows any constraint of precedence associated with the specific step. The reconfiguration process has 5 main phases: preliminary phase of security checks and inventory (X.x), replacement of the components into the vertical depot (A.x), replacement of the vacuum gripper (B.x), replacements of the components onto the conveyor (C.x), and a final phase of regulation, testing and initialisation (Z.x). Although many of the steps are not designed to be carried out by robots, a robotic manipulator is very useful for two main tasks: to manage the handling of parts and tools from the warehouse to the workspace and to hold the parts in position to allow the operator a smoother and faster fastening process. In addition, some of the steps can be performed in parallel expediting the whole process (e.g. while the operator assembles one component, the robot goes to the warehouse to pick up the next one).

4.2. Optimisation of the robot's energy consumption

The objective of this work is therefore to consider some representative tasks of the ALIX reconfiguration process and predict the energy consumption of the mobile manipulator by analysing different operating conditions. This allows the optimisation of the usage of the mobile manipulator by choosing the most efficient scenario. However, given the complexity of accurate physical models, the choice has been to exploit the potential of Physics-Informed Neural Networks. Combining the machine learning capabilities with the possibility of enforcing the physical knowledge, PINNs allow to simplify the physical formulation and achieve faster and more efficient modelling while respecting the physical constraints. The energy consumption of the mobile manipulator is represented by the battery discharge rate of the MiR100 that powers the system. Neglecting the self-discharge rate, the state of charge of the battery can be expressed as in Eq. (4):

$$\mathrm{SoC}(t) = \mathrm{SoC}(t_0) - \frac{1}{\eta}\frac{E_{\mathrm{tot}}(t)}{C_{\mathrm{bat}}} \tag{4}$$

where $\mathrm{SoC}(t)$ is the battery state of charge at time t, $\mathrm{SoC}(t_0)$ is the initial state of charge, η is the battery efficiency, $E_{\mathrm{tot}}(t)$ is the total energy consumed from time t_0 to time t, and C_{bat} is the nominal battery capacity. The energy consumption then can be decomposed based on the operating modes: energy consumption at rest and in motion of the AMR, and energy consumption at rest (with mechanical brake enabled) and in motion of the manipulator. The energy consumption at rest depends mostly on the control, diagnostic and safety systems and is considered constant. When robots are in motion, dynamic contributions related to the electric motors are to be added. To reflect the actual system behaviour, the energy consumption model assumes non-simultaneous motion between the AMR and the robotic manipulator. While certain operations (e.g., part transport and placement) involve both agents, their actions are executed in a coordinated but sequential manner. This constraint is primarily imposed for safety, as the AMR lacks the ability to dynamically detect or predict the manipulator's position, increasing the risk of collision in semi-structured environments. Additionally, the task structure does not require parallel execution, and the mission-handling system enforces sequential execution through task queuing. Finally, avoiding simultaneous operation helps to manage power draw more effectively, since both robots share a common energy source. The robot, therefore, presents 3 operating modes: idle, AMR moving,

Table 1
Example of reconfiguration steps to switch to assembly mode.

Step	Description of the operation	Preceded by
X.0	Check that all the necessary parts and tools for the reconfiguration are available. Assembly mode parts have a T carved onto the surface.	
X.1	Check that the machine is switched off and all valves of compressed air are closed. Remove all products from the depot and the conveyor.	
X.2	Disconnect the power cable and the compressed air hose.	X.1
X.3	Take the necessary parts and tools from storage and bring them to the workstation.	
A.1	Remove the locking cylinder at the bottom of the depot by loosening the four-wing screw.	
A.2	Change the pusher on the locking cylinder.	All X, A.1
A.3	Remove the depot by sliding it upwards.	A.1
A.4	Loosen the 6 wing screws securing the exit of the depot.	A.3
A.5	Change the rails at the exit of the depot mounting the new ones in the outer notches and re-tighten the 6 wing screws.	A.4
A.6	Install the new depot in the stack.	A.5
A.7	Refit the locking cylinder by re-tightening the four wing screws.	A.2, A.6
A.8	Stock the depot with the new product.	A.7
B.1	Disconnect the compressed air hose from the clamp.	
B.2	Change the gripper. It is secured with two 3 mm hex head cap screws.	B.1
C.1	Change the housing of buffer components on the conveyor. It is secured with four wing screws.	
C.2	Change the product separator and adjust its positioning using the slotted holes. It is secured by two wing screws.	
C.3	Adjust the conveyor rails. They are secured with eight 4 mm hex head cap screws.	
C.4	Change the two adjusters for correct positioning of the final product by spacing them as widely as possible. They are secured with four wing screws into slotted holes for regulation.	
C.5	Change the stopper for correct positioning of the final product by backing it up to the maximum. It is secured with two wing screws into slotted holes for regulation.	
Z.1	Move the machine to the new position by lifting the anti-vibration rubber feet and loading the weight on the wheels. Secure the rubber feet in the new position. They are regulated by a screw and nut-screw system adjustable without tools.	All A,B,C
Z.2	Reconnect the power cable and the compressed air hose. Verify that the pressure is 6 Bar otherwise adjust it by using the regulation knob.	Z.1
Z.3	Switch on the machine. Unlock all the emergency stops. Reset and initialise the machine.	Z.2
Z.4	Regulate the parameters for the new configuration.	
Z.5	Using the step-by-step mode, regulate the positioning of the product separator, the adjusters and the stopper. All these components have slotted holes for regulation.	Z.4
Z.6	Using the automatic modes, test the process at least three times.	Z.5

manipulator moving. Eq. (5) shows the three cases.

$$E(t) = \begin{cases} P_{\text{idle}} \cdot t & \text{if idle state} \\ P_{\text{idle}} \cdot t + \int P_{\text{MiR,dyn}}(t)\, dt & \text{if AMR moving} \\ P_{\text{idle}} \cdot t + \int P_{\text{UR,dyn}}(t)\, dt & \text{if manipulator moving} \end{cases} \tag{5}$$

The constant term (P_{idle}) is obtained by measuring the power of the system in idle state with both robots switched on. For what concern the dynamical terms, instead, Physics-Informed Neural Networks are implemented. Fig. 8 shows the proposed approach. It is structured into two different Physics-Informed Neural Networks. For the mobile manipulator, the current has been selected as the output of the PINN model to simplify the training with more accurate labelled data. Subsequently, the power is calculated as the sum of the power of each DC motor considering an equivalent circuit for each motor (Eq. (6)):

$$\begin{aligned} V_i &= k_{t,i}(N_i \dot{q}_i) + R_i I_i + L \frac{dI_i}{dt} \\ P_{\text{UR,dyn}} &= \sum_{i=0}^{5} V_i I_i = \sum_{i=0}^{5} \left[k_{t,i} I_i (N_i \dot{q}_i) + R_i I_i^2 + L I_i \frac{dI_i}{dt} \right] \end{aligned} \tag{6}$$

where R_i is the electrical resistance of motor armature for joint i, L_i is the inductance of motor armature for joint i, I_i is the current of the motor for joint i, k_i is the motor constant for joint i, N_i is reduction ratio of the joint i, and $\dot{q}_i$ is the velocity of the joint i. Many of the above parameters are values known or experimentally calculable. The problem therefore narrows down to defining a PINN model which, given the positions, velocities and accelerations of the joints, predicts the motor currents. The choice of using positions and velocities as input derives from the lack of robot dynamic models in RoboDK, which instead only provides kinematic parameters. On the other hand, for the PINN model implemented for the AMR, the output is directly the power and, for the same reasons of the manipulator model, the input are the linear and angular velocity of the vehicle, that can be obtained through RoboDK.

4.3. Implementation of the PINN models

The proposed approach therefore requires the implementation of two Physics-Informed Neural Networks, one for the manipulator and one for the AMR. Both models are implemented in PyTorch [52]. Regarding the manipulator, the physical knowledge implemented in the PINN model is the robot dynamics, shown in Eq. (7) [46]:

$$I = \frac{1}{N k_t} \left[M(q)\ddot{q} + C(q,\dot{q})\dot{q} + g(q) + \tau_f(\dot{q}) \right] \tag{7}$$

where I represents the vector of motor currents, N is the vector of gear ratios, and k_t is the vector of motor constant. Also, q, $\dot{q}$, and $\ddot{q}$ represent respectively the joint position, velocity, and acceleration vectors. $M(q)$ is the inertia matrix of the manipulator, $C(q,\dot{q})$ represent the Coriolis matrix, $g(q)$ is the gravitational torque vector, and $\tau_f(\dot{q})$ models the joint friction torque. Given the complexity of calculation of the matrices and vectors in Eq. (7) ($M, C, g,$ and τ_f), the Robotic Toolbox for Python [53] has been used to support the implementation of this equation. Table 2 reports the Denavit–Hartenberg (DH) matrix of the manipulator, and the mass and centre of mass of each link.

The additional dynamical parameters required for the physical modelling such as motor constant and reduction ratios are taken from [54]. Regarding the friction model for the manipulator, it includes viscous and Coulomb terms, expressed as:

$$\tau_f = F_v \cdot \dot{q} + F_c \cdot \text{sign}(\dot{q}) \tag{8}$$

with torques referenced at the gearbox output. The friction parameters F_v and F_c were experimentally tuned starting from the values reported

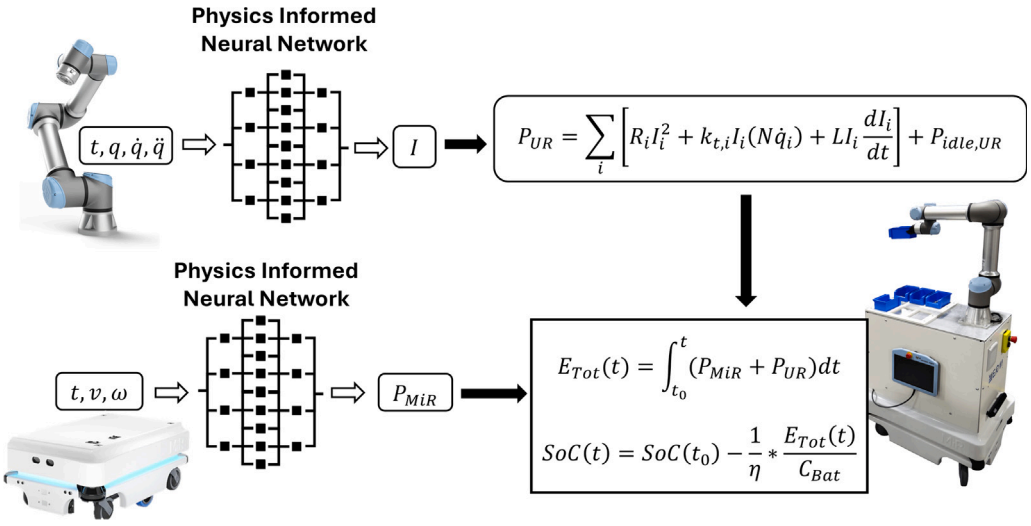

Fig. 8. Scheme of model implementation.

Table 2
DH matrix, masses and centres of mass for the UR5e robot.

Joint	θ_i	a_i [m]	d_i [m]	α_i [rad]	Link M [kg]	Link G [m]
0	q_0	0	0.1625	$\frac{\pi}{2}$	3.761	[0, −0.02561, 0.00193]
1	q_1	−0.425	0	0	8.058	[0.2125, 0, 0.11336]
2	q_2	−0.3922	0	0	2.846	[0.15, 0.0, 0.0265]
3	q_3	0	0.1333	$\frac{\pi}{2}$	1.37	[0, −0.0018, 0.01634]
4	q_4	0	0.0997	$-\frac{\pi}{2}$	1.3	[0, 0.0018, 0.01634]
5	q_5	0	0.0996	0	0.365	[0, 0, −0.001159]

Table 3
Motor constants, gear ratios, and friction parameters for the UR5e robot.

Joint	Motor constant [K_m]	Gear ratio [N]	Friction [F_v, F_c]
0	0.11	100	F_v = 5e−3, F_c = ±5e−4
1	0.11	100	F_v = 5e−3, F_c = ±1e−3
2	0.11	100	F_v = 5e−3, F_c = ±4e−4
3	0.08	100	F_v = 1e−3, F_c = ±3e−4
4	0.08	100	F_v = 3e−3, F_c = ±1e−4
5	0.08	100	F_v = 1e−3, F_c = ±2e−4

in [54] and refined across datasets to match the system's behaviour. All these parameters are resumed in Table 3.

On the other hand, the AMR dynamics is modelled assuming a flat operating surface with uniform friction, consistent with the conditions of the experimental lab environment. Ground slope or non-uniform contact properties are excluded. As with the manipulator, the PINN compensates for discrepancies between the simplified physical assumptions and real-world behaviour through data-driven correction. The Eq. (9) represent, therefore, a simplified model of the differential drive. It is based on the definition of three contributions: the idle power and the motion power from the two motors (left and right wheels). The calculation of the traction force takes into account the acceleration (without regenerative brake), the rolling friction and the drag force. These last two terms are equally distributed over the two motors.

$$\begin{aligned} P_{\text{motor,r}} &= \frac{F_r * v_r}{\eta} = \frac{1}{\eta}\left(ma_r + \frac{1}{2}\mu mg + \frac{1}{4}C_p A v^2\right)\left(v + \frac{D}{2}\omega\right) \\ P_{\text{motor,l}} &= \frac{F_l * v_l}{\eta} = \frac{1}{\eta}\left(ma_l + \frac{1}{2}\mu mg + \frac{1}{4}C_p A v^2\right)\left(v - \frac{D}{2}\omega\right) \\ P_{\text{MiR}} &= P_{\text{motor,r}} + P_{\text{motor,l}} + P_{\text{idle,MiR}} \end{aligned} \tag{9}$$

where F_r and F_l are the traction forces on the two wheels, v_r and v_l are the velocities of the wheels, η is the motor efficiency, m is the vehicle mass, a_r, a_l are the linear accelerations of the wheels, μ is the friction, g is the gravity acceleration, C_p is the drag coefficient, A is the area, v is the linear velocity of the vehicle, D is the track width, and ω is the angular velocity of the vehicle. Table 4 presents all the constant parameters used for the implementation of the AMR dynamics.

Table 4
Parameters for the AMR dynamics.

Symbol	Description	Value
D	Wheel separation (m)	0.508
μ	Rolling resistance coefficient	0.010
C_p	Drag coefficient	1.0
A	Frontal area (m^2)	0.4
m	Mass (kg)	100
g	Gravity (m/s^2)	9.81
η	Motor efficiency	0.9
Capacity	Battery capacity (Wh)	960

Table 5
Model hyper-parameters for manipulator and AMR.

Hyper-parameters	Manipulator	AMR
Architecture	FNN [19–2 × 100–6]	FNN [3–2 × 100–1]
Activation function	$ReLU$	$ReLU$
Learning rate	1e−4	1e−3
Optimiser	Adam	Adam
Batch size	20 000	4000
Loss function weights	[1,...,1]	[0.05, 1]

Regarding the implementation of the two neural networks, Table 5 presents the hyper-parameters for the two models. The training process for them has been done separately and then tested in a complete scenario.

During the training, validation and testing of the models the following Key Performance Indicators (KPIs) have been used (Eq. (10))

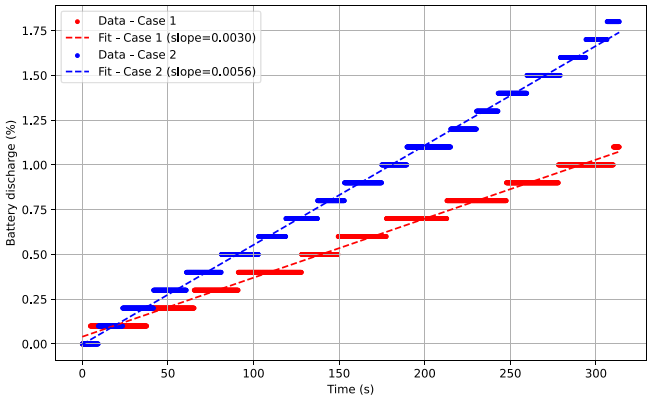

Fig. 9. Idle SoC in cases 1 and 2.

Table 6
Comparison of metrics for current prediction of all joints of the manipulator for a batch size of 10 000 samples.

KPI	Model	J0	J1	J2	J3	J4	J5	Mean
NRMSE	PINN	0.023	0.059	0.057	0.020	0.011	0.010	0.030
	Physics	0.026	0.056	0.081	0.022	0.014	0.016	0.036
	MLP	0.032	0.105	0.044	0.020	0.015	0.015	0.039
R^2	PINN	0.921	0.894	0.773	0.803	0.503	0.931	0.804
	Physics	0.902	0.904	0.540	0.753	0.277	0.835	0.702
	MLP	0.849	0.664	0.866	0.804	0.096	0.851	0.688
NMAE	PINN	0.018	0.048	0.041	0.016	0.010	0.009	0.024
	Physics	0.021	0.049	0.056	0.019	0.012	0.014	0.029
	MLP	0.027	0.086	0.035	0.016	0.013	0.013	0.032

to compare the PINNs with the physics-based model and with a data-driven approach: Multi-Layer Perceptron (MLP).

$$\begin{aligned} \text{NRMSE} &= \frac{\sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_{\text{true},i}-y_{\text{pred},i}\right)^2}}{y_{\text{true,max}}-y_{\text{true,min}}}, \\ R^2 &= 1-\frac{\sum_{i=1}^{n}\left(y_{\text{true},i}-y_{\text{pred},i}\right)^2}{\sum_{i=1}^{n}\left(y_{\text{true},i}-\bar{y}_{\text{true}}\right)^2}, \\ \text{NMAE} &= \frac{\frac{1}{n}\sum_{i=1}^{n}\left|y_{\text{true},i}-y_{\text{pred},i}\right|}{y_{\text{true,max}}-y_{\text{true,min}}} \end{aligned} \tag{10}$$

5. Results and discussion

The results are presented focusing first on the training and validation of the two models and then testing the entire approach on a complete scenario referring to some of the tasks of the reconfiguration process. To define the models, however, it is required the average consumption when the system is in idle. There are to cases of idle: the first with the manipulator switched off (Case 1) and the second with the manipulator switched on and the mechanical brakes released (Case 2). Fig. 9 shows the measured data and the linear interpolation function for the two cases. Considering a battery capacity of 960 Wh and the values of the slopes, it is possible to calculate the idle power in the two cases, that are approximately 100 W and 190 W.

5.1. Manipulator model: training and validation

For the manipulator model, the first step is to train and validate the PINN model in predicting the current of each joint. Table 6 shows the results in terms of metrics. It highlights how PINNs perform better than both the physics-based and data-driven model with an average RMSE of 3%.

However, looking at Fig. 10, there is a limitation in predicting currents in Joint 4 mainly due to the physical side of the model that does not follow the measured data trend. Moreover, the model prioritises correct modelling of joints with higher current values, and being joint 4 the one with the lowest values, it tends to be less considered in the training process.

Once obtained the current predictions these values are used to calculate the power consumed by the manipulator through Eq. (6). Considering a standard range of 0.05–0.5 mH for the inductance, it results to be always negligible. On the other hand, electrical resistance is relevant and an average value of 0.5 Ω is considered. Moreover, for the idle power of the manipulator, it is considered a value of 30 W. As shown in Fig. 11, the predicted power closely matches the actual measures, achieving a RMSE of approximately 3 W on the prediction of the power and 40 J in predicting the energy consumption.

Table 7
Comparison of metrics for battery consumption (%) of the AMR for a batch size of 3500 samples.

Model	Normalised RMSE	R-squared	Normalised MAE
PINN	0.0443	0.9745	0.0378
Physics-based	0.0597	0.9536	0.0514
MLP	0.0702	0.9359	0.0606

Table 8
Comparison of metrics for battery consumption (%) of the manipulator model for a batch size of 3100 samples.

Model	Normalised RMSE	R-squared	Normalised MAE
PINN	0.0445	0.9774	0.0367
Physics	0.0581	0.9615	0.0485

5.2. AMR model: training and validation

For the AMR model, Table 7 shows a comparison of metrics between the PINN, the physics-based and the data-driven models. Again, the model based on PINNs performs better. The specific values of the metrics, however, have little relevance as the SoC measurement are discretised with an accuracy of ±0.1%. For the same reason, a low weight has been assigned to the loss function on labelled data (5%) when training the PINN model.

Analysing the graphs of prediction in Fig. 12, it can be observed that the PINN model tends to respect the constraints of physical laws such as the monotonicity of the discharge rate, which, instead, is not respected in the data-driven model despite the good prediction accuracy.

5.3. Testing of the mobile manipulator model

Finally, once both models have been implemented and validated, an operational scenario of the mobile manipulator can be tested. In this case, it is considered the task X.3 of the reconfiguration process that involves pick and place operations to move equipment and components required for the reconfiguration in the workspace. Fig. 13 shows an example of one of this pick and place operations.

Table 8 shows the metrics of prediction for the PINNs-based approach and for the physics-based model, confirming the higher accuracy of the PINNs-based model.

Moreover, Fig. 14 shows the consumption associated to the manipulator and to the AMR. The component related to the manipulator tends to be linear as, throughout the process, the manipulator has to counter gravity and the motors are always running, so during its operational phase consumption is not very different. Finally, comparing the graphs of the physics and PINN model, it is possible to conclude that although the influence of physical constraints on the model can be clearly observed, its physical interpretability remains limited and the behaviour predicted by the physical model (e.g. in the transition between idle and active phases) is not reflected clearly in the model based on PINNs.

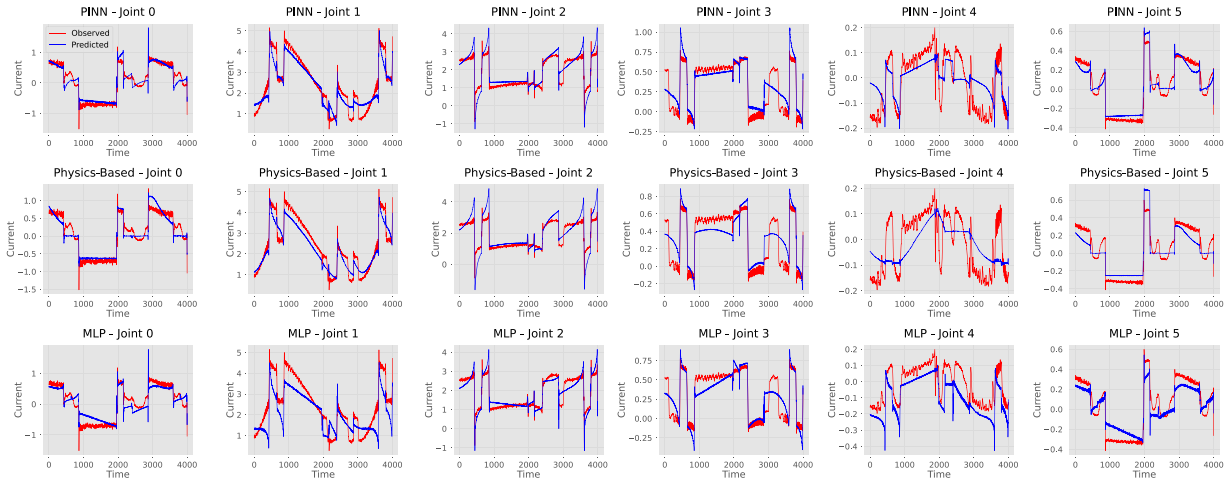

Fig. 10. Comparison of prediction graphs between the three tested models.

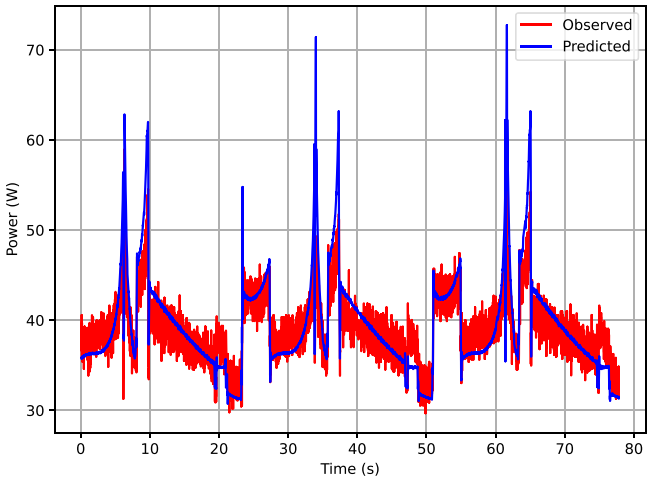

Fig. 11. Prediction of the power consumed by the manipulator.

The results, from a general point of view, show that an approach based on PINNs effectively enables the integration of physical constraints in the use of machine learning models, improving the accuracy and compatibility with physics of the results. A further advantage is that the proposed model, being structured on system dynamics, can be exploited for multiple purposes, including cycle-time optimisation, trajectory planning, and collision detection. This, supported by integration with a digital twin, allows the use of both simulated and real data to obtain more realistic and complete simulations, or additional data useful for the optimisation of human–robot collaborative tasks.

6. Conclusion

This paper presented a novel Digital Twin-based framework for an efficient execution of CPPS reconfiguration, emphasising human–robot collaboration and guided by Industry 5.0 principles. The framework integrates Mixed Reality (MR) to enable real-time, hands-free human–machine interaction and seamless robot programming, while a Physics-Informed Neural Network (PINN) model enhances predictive capabilities by embedding physical laws into the learning process. This framework was implemented as a proof of concept on ALIX, a modular and reconfigurable research platform, focusing on a use case about the execution of the reconfiguration to change operating mode. In this scenario, a communication mechanisms between the different modules of the proposed framework has been tested. Moreover, a detailed case study focused on the use of PINNs to predict the energy consumption of the mobile manipulator employed in the reconfiguration. It is composed of an Autonomous Mobile Robot (AMR) and a collaborative robotic arm and the approach employed a dedicated PINN for each subsystem, integrating their respective dynamic models directly into the training process. This method outperformed conventional data-driven and physics-based models, providing more accurate and physically consistent solutions than classical neural networks. These results confirm the potential of the PINNs integrated into a digital twin to support efficient, intelligent reconfiguration and decision-making, while enhancing human–robot collaboration. Future research will focus on developing a demonstrator for the complete reconfiguration process using the Digital Twin framework. Potential extensions of the simulation environment and PINNs include applications in human motion prediction, collision detection, and ergonomic analysis. Previous studies have indicated the feasibility of such approaches, but their transferability has not yet been empirically validated within the proposed framework. Moreover, it is planned to integrate evaluation metrics for operator workload and task complexity, as well as to quantitatively assess the time and efficiency gains achieved through the proposed framework, compared to manual-only reconfiguration workflows.

CRediT authorship contribution statement

Francesco Giuseppe Ciampi: Writing – review & editing, Writing – original draft, Methodology, Investigation, Data curation, Conceptualization. **Thierno M.L. Diallo:** Writing – review & editing, Writing – original draft, Validation, Formal analysis, Data curation, Conceptualization. **Faïda Mhenni:** Writing – review & editing, Supervision, Formal analysis. **Jean-Yves Choley:** Writing – review & editing, Supervision, Resources, Funding acquisition. **Stanislao Patalano:** Writing – review & editing, Supervision, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The presented considerations and results were discussed within the Suniswell cooperation between ISAE-Supméca (Paris, France), University of Applied Sciences Upper Austria (Wels, Austria) and University of Naples Federico II (Naples, Italy). The authors thank all partners for their valuable contributions.

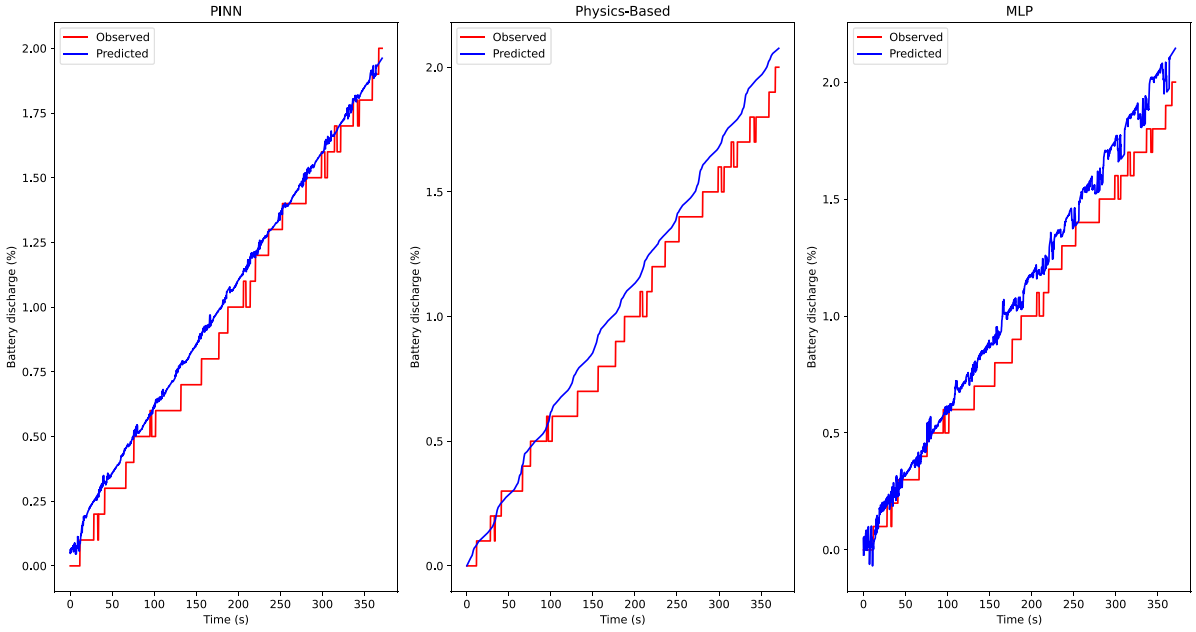

Fig. 12. Comparison of prediction graphs between the three tested models.

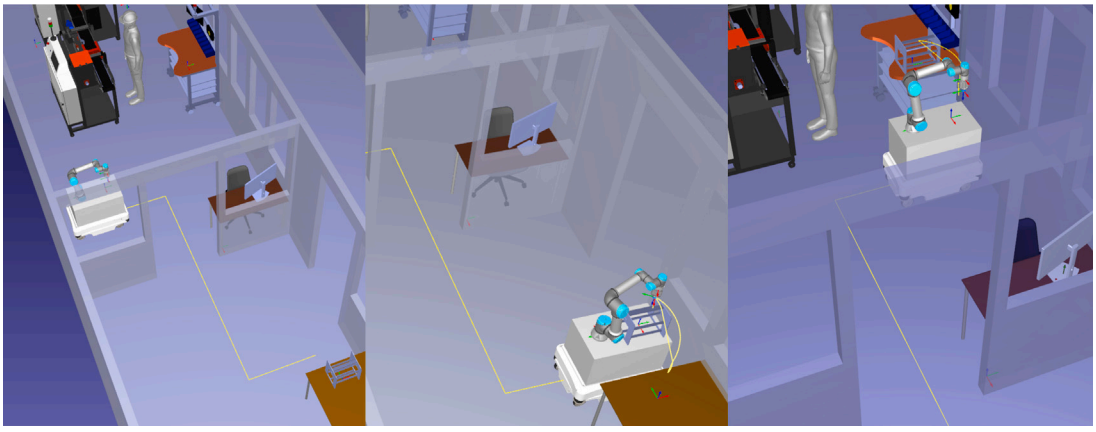

Fig. 13. Representation of a typical operation used in the test phase.

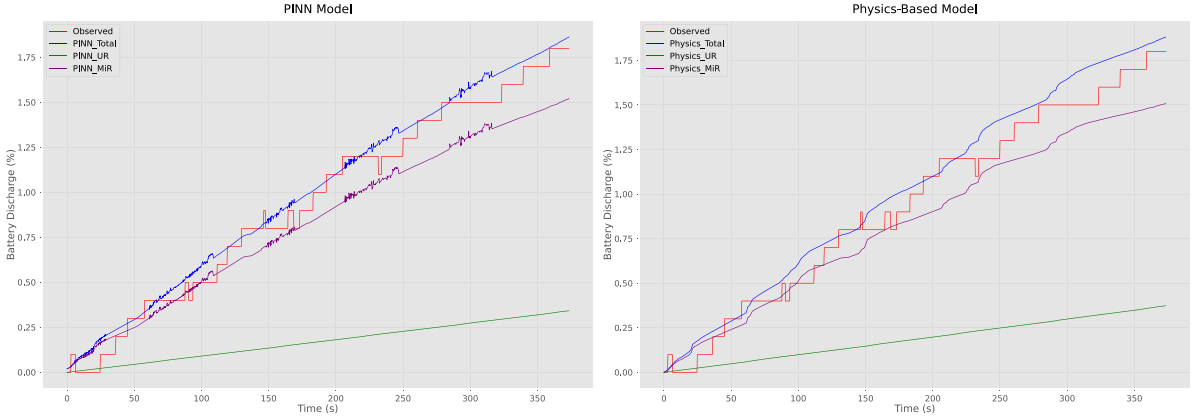

Fig. 14. Comparison of prediction graphs between the PINN model and the physics-based model for the final test.

Appendix A. Pseudocodes

```
DECLARE host AS STRING := "IP_ADDRESS"
DECLARE port AS INTEGER := PORT_NUMBER
DECLARE client AS TcpClient
DECLARE stream AS NetworkStream

PROCEDURE connect()
    TRY
        client := NEW TcpClient(host,
            port)
        stream := CALL client.GET_STREAM
            ()
        PRINT "Connected to server"
    CATCH CONNECTION_EXCEPTION AS
        EXCEPTION
        PRINT "Connection error: " +
            EXCEPTION.MESSAGE
    END TRY
END PROCEDURE

PROCEDURE send_message(message AS STRING
    )
    IF client IS NOT NULL AND stream IS
        NOT NULL THEN
        DECLARE data AS BYTE_ARRAY :=
            ENCODE_UTF8(message)
        CALL stream.WRITE(data, 0,
            LENGTH(data))
        PRINT "Sent: " + message
    ELSE
        PRINT "Not connected to server"
    END IF
END PROCEDURE

PROCEDURE disconnect()
    IF stream IS NOT NULL THEN CALL
        stream.CLOSE()
    IF client IS NOT NULL THEN CALL
        client.CLOSE()
    PRINT "Connection closed"
END PROCEDURE

PROCEDURE execute_command()
    CALL connect()
    CALL send_message("USER_COMMAND")
    CALL disconnect()
END PROCEDURE
```

Script 1: Pseudocode to send user commands from Unity to RoboDK.

```
DECLARE host AS STRING := "
    BROKER_ADDRESS"
DECLARE topic AS STRING := "TOPIC/
    SUBTOPIC"
DECLARE client AS MessagingClient
DECLARE data AS DATA_TYPE
DECLARE keep_running AS BOOL := TRUE

PROCEDURE initialise_data()
    ... # Initialise data structure
END PROCEDURE

PROCEDURE on_message_received(sender,
    message)
    DECLARE received AS STRING :=
        DECODE_UTF8(message)
    data := PARSE_JSON(received)
END PROCEDURE

PROCEDURE subscribe()
    client := NEW MessagingClient(host)
    client.OnMessage := CALLBACK
        on_message_received
    DECLARE client_id AS STRING :=
        GENERATE_ID()
    CALL client.CONNECT(client_id)
    CALL client.SUBSCRIBE([topic], [
        QOS_LEVEL])
END PROCEDURE

PROCEDURE unsubscribe()
    IF client IS NOT NULL THEN
        CALL client.UNSUBSCRIBE([topic])
        CALL client.DISCONNECT()
        PRINT "Disconnected from broker"
    END IF
END PROCEDURE

PROCEDURE main_loop()
    CALL initialise_data()
    CALL subscribe()
    WHILE keep_running DO
        WAIT(0.1)
    END WHILE
    CALL unsubscribe()
END PROCEDURE
```

Script 2: Pseudocode for subscribing an MQTT topic in Unity.

```
DECLARE host AS STRING := "IP_ADDRESS"
DECLARE port AS INTEGER := 5000
DECLARE server_socket AS TcpServer
DECLARE client_socket AS TcpClient
DECLARE is_running AS BOOL := TRUE

DECLARE system AS RobotSystem := NEW
    RobotSystem()
DECLARE robot AS Robot := system.GetItem
    ("ROBOT")

PROCEDURE initialise_server()
    server_socket := NEW TcpServer(host,
         port)
    CALL server_socket.Listen()
    PRINT "TCP server listening at " +
        host + ":" + TO_STRING(port)
END PROCEDURE

PROCEDURE handle_command(command AS
    STRING)
    IF command = "COMMAND_1" THEN
        ... # Perform action 1
    ELSE IF command = "COMMAND_2" THEN
        ... # Perform action 2
    ELSE
        PRINT "Unknown command: " +
            command
    END IF
END PROCEDURE

PROCEDURE start_server_loop()
    CALL initialise_server()
    WHILE is_running DO
        client_socket := server_socket.
            Accept()
        PRINT "Client connected"
```

```
        WHILE client_socket.IsConnected
            () DO
            DECLARE message AS STRING :=
                 client_socket.Receive()
            IF message IS EMPTY THEN
                BREAK
            END IF
            PRINT "Received: " + message
            CALL handle_command(message)
        END WHILE
        CALL client_socket.Close()
        PRINT "Client connection closed"
    END WHILE

    CALL server_socket.Close()
    PRINT "Server shut down"
END PROCEDURE

CALL start_server_loop()
```

Script 3: Pseudocode for receiving commands via TCP in RoboDK.

```
DECLARE host AS STRING := "HOST_ADDRESS"
DECLARE topic AS STRING := "TOPIC_NAME"
DECLARE client AS MessagingClient
DECLARE system AS RobotSystem
DECLARE robot_list AS LIST
DECLARE keep_running AS BOOL := TRUE

PROCEDURE initialise()
    system := NEW RobotSystem()
    robot_list := CALL system.
        GET_ALL_ROBOTS()
    client := NEW MessagingClient()
    TRY
        CALL client.CONNECT(host)
    CATCH CONNECTION_EXCEPTION AS
        EXCEPTION
        PRINT "Connection error: " +
            EXCEPTION.MESSAGE
        EXIT PROCEDURE
    END TRY
END PROCEDURE

PROCEDURE publish_states()
    DECLARE payload AS DICTIONARY :=
        EMPTY_DICTIONARY()
    FOR EACH robot IN robot_list DO
        payload[robot.ID] := CALL robot.
            GET_JOINT_POSITIONS()
    END FOR

    DECLARE message AS STRING :=
        ENCODE_JSON(payload)
    CALL client.PUBLISH(topic, message)
    PRINT "Published robot states"
END PROCEDURE

PROCEDURE terminate()
    CALL client.DISCONNECT()
    PRINT "Messaging client disconnected
        "
END PROCEDURE

PROCEDURE main_loop()
    CALL initialise()
    WHILE keep_running DO
        CALL publish_states()
        WAIT(0.1)
    END WHILE
    CALL terminate()
END PROCEDURE
```

Script 4: Pseudocode for publishing onto an MQTT topic in RoboDK.

```
DECLARE system AS RobotSystem := NEW
    RobotSystem()
DECLARE robot AS Robot := system.GetItem
    ("ROBOT_NAME")
DECLARE program AS Program := system.
    GetItem("PROGRAM_NAME")
DECLARE time_step AS FLOAT := TIME_STEP
DECLARE data_log AS LIST

PROCEDURE simulate_robot_program()
    CALL program.Run()
    WAIT(0.1)
    DECLARE result := program.
        GetJointInstructionList(flags=
        SIMULATION_FLAG, time_step=
        time_step)
    data_log := result.data
END PROCEDURE

PROCEDURE extract_features()
    ... # extract from data_log the
        input for the ML model
    RETURN features
END PROCEDURE

PROCEDURE train_model(train_input,
    train_output)
    ... # define PINNs and train
    RETURN Trained_model
END PROCEDURE

PROCEDURE inference(model, test_input)
    DECLARE prediction := model.Predict(
        test_input)
    RETURN prediction
END PROCEDURE

CALL simulate_robot_program()
DECLARE X:= extract_features()
DECLARE X_train, X_test := split_data(X,
     Y)
DECLARE model := train_model(X_train)
CALL inference(model, X_test)
```

Script 5: Pseudocode for simulating a robot program and use the data for a PINN model.

```
DECLARE url AS STRING := "API_URL"
DECLARE headers AS DICTIONARY := {
    "Content-Type": "application/json",
    "Authorization": "Basic access_token
        "
}
DECLARE data AS DATA_TYPE
DECLARE keep_running AS BOOL := TRUE

PROCEDURE get_data()
    TRY
        response := CALL GET_REQUEST(url
            , headers)
        IF response.STATUS == 200 THEN
            RETURN response.BODY
        ELSE
```

```
            PRINT "API error " +
                response.STATUS
            RETURN NULL
        END IF
    CATCH REQUEST_EXCEPTION AS EXCEPTION
        PRINT "Request failed: " +
            EXCEPTION.MESSAGE
        RETURN NULL
    END TRY
END PROCEDURE

PROCEDURE main_loop()
    WHILE keep_running DO
        DECLARE raw := CALL get_data()
        IF raw IS NOT NULL THEN
            data := PARSE_JSON(raw)
        END IF
        WAIT(0.1)
    END WHILE
END PROCEDURE
```

Script 6: Pseudocode for Requesting Data from MiR API.

```
DECLARE host AS STRING := "IP_ADDRESS"
DECLARE port AS INTEGER := 30004
DECLARE config_file AS STRING := "
    rtde_config.xml"
DECLARE frequency AS FLOAT := FREQUENCY
DECLARE keep_running AS BOOL := TRUE

PROCEDURE run_rtde()
    DECLARE con AS RTDEClient := NEW
        RTDEClient(host, port)
    CALL con.CONNECT()

    CALL con.GET_CONTROLLER_VERSION()
    DECLARE names, types := con.
        GET_RECIPE("out")

    IF NOT con.CONFIGURE_OUTPUT(names,
        types, frequency) THEN
        PRINT "Failed to configure RTDE
            output"
        EXIT PROCEDURE
    END IF

    IF NOT con.START() THEN
        PRINT "Failed to start RTDE"
        EXIT PROCEDURE
    END IF

    WHILE keep_running DO
        DECLARE state := con.RECEIVE()
        IF state IS NOT NULL THEN
            # Process RTDE state data
        END IF
    END WHILE

    CALL con.DISCONNECT()
    PRINT "RTDE connection closed"
END PROCEDURE
```

Script 7: Pseudocode for implementing the RTDE protocol.

```
DECLARE system AS RobotSystem
DECLARE robot AS Robot
DECLARE program AS RobotProgram

PROCEDURE execute_program()
    system := NEW RobotSystem()
    robot := CALL system.GET_ROBOT("
        RobotName")
    program := CALL system.GET_PROGRAM("
        ProgramName")

    IF NOT robot.CONNECT() THEN
        PRINT "Robot connection failed"
        EXIT PROCEDURE
    END IF

    CALL system.SET_EXECUTION_MODE(
        RUN_ON_ROBOT)
    CALL program.SET_EXECUTION_TARGET(
        RUN_ON_ROBOT)

    CALL program.RUN()

    WHILE program.IS_RUNNING() DO
        WAIT(0.1)
    END WHILE

    PRINT "Program execution completed"
END PROCEDURE
```

Script 8: Pseudocode for robot program execution with RoboDK API.

Appendix B. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.jmsy.2025.09.001.

References

[1] Xu X, Lu Y, Vogel-Heuser B, Wang L. Industry 4.0 and Industry 5.0—Inception, conception and perception. J Manuf Syst 2021;61:530–5. http://dx.doi.org/10.1016/j.jmsy.2021.10.006.

[2] Zizic MC, Mladineo M, Gjeldum N, Celent L. From Industry 4.0 towards Industry 5.0: A review and analysis of paradigm shift for the people, organization and technology. Energies 2022;15(14):5221. http://dx.doi.org/10.3390/en15145221.

[3] Leng J, Sha W, Wang B, Zheng P, Zhuang C, Liu Q, Wuest T, Mourtzis D, Wang L. Industry 5.0: prospect and retrospect. J Manuf Syst 2022;65:279–95. http://dx.doi.org/10.1016/j.jmsy.2022.09.017.

[4] Cardin O. Classification of cyber-physical production systems applications: Proposition of an analysis framework. Comput Ind 2019;104:11–21. http://dx.doi.org/10.1016/j.compind.2018.10.002.

[5] Bordoloi SK, Cooper WW, Matsuo H. Flexibility, adaptability, and efficiency in manufacturing systems. Prod Oper Manage 1999;8(2):133–50. http://dx.doi.org/10.1111/j.1937-5956.1999.tb00366.x.

[6] Matevska J. Rekonfiguration komponentenbasierter Softwaresysteme. In: Matevska J, editor. Rekonfiguration komponentenbasierter softwaresysteme zur laufzeit. 2010, p. 73–81. http://dx.doi.org/10.1007/978-3-8348-9780-0_5.

[7] Müller T, Jazdi N, Schmidt J-P, Weyrich M. Cyber-physical production systems: enhancement with a self-organized reconfiguration management. Procedia CIRP 2021;99:549–54. http://dx.doi.org/10.1016/j.procir.2021.03.075.

[8] Müller T, Kamm S, Löcklin A, White D, Mellinger M, Jazdi N, Weyrich M. Architecture and knowledge modelling for self-organized reconfiguration management of cyber-physical production systems. Int J Comput Integr Manuf 2023;36(12):1842–63. http://dx.doi.org/10.1080/0951192x.2022.2121425, Publisher: Informa UK Limited.

[9] Zafar MH, Langås EF, Sanfilippo F. Exploring the synergies between collaborative robotics, digital twins, augmentation, and industry 5.0 for smart manufacturing: A state-of-the-art review. Robot Comput-Integr Manuf 2024;89:102769. http://dx.doi.org/10.1016/j.rcim.2024.102769.

[10] Raissi M, Perdikaris P, Karniadakis G. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J Comput Phys 2019;378:686–707. http://dx.doi.org/10.1016/j.jcp.2018.10.045.

[11] Tao F, Qi Q, Wang L, Nee A. Digital twins and cyber–physical systems toward smart manufacturing and Industry 4.0: Correlation and comparison. Engineering 2019;5(4):653–61. http://dx.doi.org/10.1016/j.eng.2019.01.014, Publisher: Elsevier BV.

[12] Lee J, Bagheri B, Kao H-A. A cyber-physical systems architecture for Industry 4.0-based manufacturing systems. Manuf Lett 2015;3:18–23. http://dx.doi.org/10.1016/j.mfglet.2014.12.001, Publisher: Elsevier BV.

[13] Wu X, Goepp V, Siadat A. Concept and engineering development of cyber physical production systems: a systematic literature review. Int J Adv Manuf Technol 2020;111(1–2):243–61. http://dx.doi.org/10.1007/s00170-020-06110-2.

[14] Macherki D, Diallo TML, Guizani A, Barkallah M, Choley J-Y, Haddar M. Effective implementation of CPPS self-reconfiguration functionality: Research review. In: Walha L, Jarraya A, Djemal F, Chouchane M, Aifaoui N, Chaari F, Abdennadher M, Benamara A, Haddar M, editors. Design and modeling of mechanical systems - V. 2023, p. 260–8. http://dx.doi.org/10.1007/978-3-031-14615-2_30.

[15] Hengstebeck A, Barthelmey A, Deuse J. Reconfiguration assistance for cyber-physical production systems. In: Schüppstuhl T, Tracht K, Franke J, editors. Tagungsband des 3. Kongresses montage handhabung industrieroboter. 2018, p. 177–86. http://dx.doi.org/10.1007/978-3-662-56714-2_20.

[16] Napoleone A, Negri E, Macchi M, Pozzetti A. How the technologies underlying cyber-physical systems support the reconfigurability capability in manufacturing: a literature review. Int J Prod Res 2023;61(9):3122–44. http://dx.doi.org/10.1080/00207543.2022.2074323.

[17] Zhu Q, Huang S, Wang G, Moghaddam SK, Lu Y, Yan Y. Dynamic reconfiguration optimization of intelligent manufacturing system with human-robot collaboration based on digital twin. J Manuf Syst 2022;65:330–8. http://dx.doi.org/10.1016/j.jmsy.2022.09.021.

[18] Villani V, Pini F, Leali F, Secchi C. Survey on human–robot collaboration in industrial settings: Safety, intuitive interfaces and applications. Mechatronics 2018;55:248–66. http://dx.doi.org/10.1016/j.mechatronics.2018.02.009.

[19] Hentout A, Aouache M, Maoudj A, Akli I. Human–robot interaction in industrial collaborative robotics: a literature review of the decade 2008–2017. Adv Robot 2019;33(15–16):764–99. http://dx.doi.org/10.1080/01691864.2019.1636714.

[20] Kumar S, Savur C, Sahin F. Survey of human–robot collaboration in industrial settings: awareness, intelligence, and compliance. IEEE Trans Syst Man, Cybern: Syst 2021;51(1):280–97. http://dx.doi.org/10.1109/TSMC.2020.3041231.

[21] for Standardization IO. ISO 10218-2:2025, Robotics — Safety requirements — Part 2: Industrial robot applications and robot cells. 2025.

[22] for Standardization IO. ISO 10218-1:2025, Robotics - Safety requirements - Part 1: Industrial robots. 2025, URL https://www.iso.org/standard/73933.html.

[23] for Standardization IO. ISO\TS 15066:2016. Robots and Robotic Devices: Collaborative Robots. 2016.

[24] Matheson E, Minto R, Zampieri EGG, Faccio M, Rosati G. Human–robot collaboration in manufacturing applications: A review. Robotics 2019;8(4):100. http://dx.doi.org/10.3390/robotics8040100.

[25] Othman U, Yang E. Human–robot collaborations in smart manufacturing environments: Review and outlook. Sensors 2023;23(12):5663. http://dx.doi.org/10.3390/s23125663.

[26] Li W, Hu Y, Zhou Y, Pham DT. Safe human–robot collaboration for industrial settings: a survey. J Intell Manuf 2024;35(5):2235–61. http://dx.doi.org/10.1007/s10845-023-02159-4.

[27] Baratta A, Cimino A, Longo F, Nicoletti L. Digital twin for human-robot collaboration enhancement in manufacturing systems: Literature review and direction for future developments. Comput Ind Eng 2024;187:109764. http://dx.doi.org/10.1016/j.cie.2023.109764.

[28] Baratta A, Cardamone M, Cimino A, Longo F, Nicoletti L, Padovano A, Sammarco C. Advancing task allocation in human-robot collaboration with a multi-simulation based digital twin system. Procedia Comput Sci 2025;253:3257–67. http://dx.doi.org/10.1016/j.procs.2025.02.050, Publisher: Elsevier BV.

[29] Ma X, Qi Q, Tao F. A digital twin–based environment-adaptive assignment method for human–robot collaboration. J Manuf Sci Eng 2024;146(3). http://dx.doi.org/10.1115/1.4064040, Publisher: ASME International.

[30] Zhang Z, Ji Y, Tang D, Chen J, Liu C. Enabling collaborative assembly between humans and robots using a digital twin system. Robot Comput-Integr Manuf 2024;86:102691. http://dx.doi.org/10.1016/j.rcim.2023.102691.

[31] Wang S, Zhang J, Wang P, Law J, Calinescu R, Mihaylova L. A deep learning-enhanced Digital Twin framework for improving safety and reliability in human–robot collaborative manufacturing. Robot Comput-Integr Manuf 2024;85:102608. http://dx.doi.org/10.1016/j.rcim.2023.102608.

[32] Rokhsaritalemi S, Sadeghi-Niaraki A, Choi S-M. A review on mixed reality: Current trends, challenges and prospects. Appl Sci 2020;10(2):636. http://dx.doi.org/10.3390/app10020636.

[33] Li C, Zheng P, Li S, Pang Y, Lee CK. AR-assisted digital twin-enabled robot collaborative manufacturing system with human-in-the-loop. Robot Comput-Integr Manuf 2022;76:102321. http://dx.doi.org/10.1016/j.rcim.2022.102321.

[34] Zhang C, Zhou G, Ma D, Wang R, Xiao J, Zhao D. A deep learning-enabled human-cyber-physical fusion method towards human-robot collaborative assembly. Robot Comput-Integr Manuf 2023;83:102571. http://dx.doi.org/10.1016/j.rcim.2023.102571.

[35] Li C, Zheng P, Yin Y, Pang YM, Huo S. An AR-assisted deep reinforcement learning-based approach towards mutual-cognitive safe human-robot interaction. Robot Comput-Integr Manuf 2023;80:102471. http://dx.doi.org/10.1016/j.rcim.2022.102471.

[36] Liu C, Zhang Z, Tang D, Nie Q, Zhang L, Song J. A mixed perception-based human-robot collaborative maintenance approach driven by augmented reality and online deep reinforcement learning. Robot Comput-Integr Manuf 2023;83:102568. http://dx.doi.org/10.1016/j.rcim.2023.102568, Publisher: Elsevier BV.

[37] Wang G, Zhang G, Zhou X, Zhang Y. Distributed multi-robot task dynamic allocation in digital-twin factory towards industry 5.0. Int J Prod Res 2025;1–24. http://dx.doi.org/10.1080/00207543.2025.2499866, Publisher: Informa UK Limited.

[38] Wang J, Li Y, Gao RX, Zhang F. Hybrid physics-based and data-driven models for smart manufacturing: Modelling, simulation, and explainability. J Manuf Syst 2022;63:381–91. http://dx.doi.org/10.1016/j.jmsy.2022.04.004.

[39] Rai R, Sahu CK. Driven by data or derived through physics? A review of hybrid physics guided machine learning techniques with cyber-physical system (CPS) focus. IEEE Access 2020;8:71050–73. http://dx.doi.org/10.1109/ACCESS.2020.2987324.

[40] Runkana V, Majumder S, Desai VJ, Arunprasath J, Kumar R, Nistala SH, Parihar MS, Singh K, Kumar V. Digital twins for optimization of ironmaking operations. CSI Trans ICT 2024;12(1–3):57–70. http://dx.doi.org/10.1007/s40012-024-00395-4.

[41] Yang S, Kim H, Hong Y, Yee K, Maulik R, Kang N. Data-driven physics-informed neural networks: A digital twin perspective. Comput Methods Appl Mech Engrg 2024;428:117075. http://dx.doi.org/10.1016/j.cma.2024.117075.

[42] Ciampi FG, Rega A, Diallo TML, Patalano S. Physics-informed neural networks for industrial applications: A case study in thermal power prediction of an AHU for a topcoat process. In: Di Stefano P, Gherardini F, Nigrelli V, Rizzi C, Sequenzia G, Tumino D, editors. Design tools and methods in industrial engineering IV. Springer Nature Switzerland; 2025, p. 339–47. http://dx.doi.org/10.1007/978-3-031-76597-1_36.

[43] Antonelo EA, Camponogara E, Seman LO, Jordanou JP, De Souza ER, Hübner JF. Physics-informed neural nets for control of dynamical systems. Neurocomputing 2024;579:127419. http://dx.doi.org/10.1016/j.neucom.2024.127419.

[44] Yang X, Zhou Z, Li L, Zhang X. Collaborative robot dynamics with physical human–robot interaction and parameter identification with PINN. Mech Mach Theory 2023;189:105439. http://dx.doi.org/10.1016/j.mechmachtheory.2023.105439.

[45] Yang X, Du Y, Li L, Zhou Z, Zhang X. Physics-informed neural network for model prediction and dynamics parameter identification of collaborative robot joints. IEEE Robot Autom Lett 2023;8(12):8462–9. http://dx.doi.org/10.1109/LRA.2023.3329620.

[46] Ciampi FG, Diallo TML, Mhenni F, Patalano S, Choley J-Y. Enhancing human – robot collaboration in the industry 5.0 framework with physics-informed neural networks: Application to collision detection. In: Dassisti M, Madani K, Panetto H, editors. Innovative intelligent industrial production and logistics. vol. 2372, Cham: Springer Nature Switzerland; 2025, p. 305–18. http://dx.doi.org/10.1007/978-3-031-80760-2_20.

[47] Lee H. Physics-based cooperative robotic digital twin framework for contactless delivery motion planning. Int J Adv Manuf Technol 2023;128(3–4):1255–70. http://dx.doi.org/10.1007/s00170-023-11956-3.

[48] Halim MY, Awad MI, Maged SA. Hybrid physics-infused deep learning for enhanced real-time prediction of human upper limb movements in collaborative robotics. J Intell Robot Syst 2025;111(1):38. http://dx.doi.org/10.1007/s10846-025-02237-0.

[49] Ajoudani A, Zanchettin AM, Ivaldi S, Albu-Schäffer A, Kosuge K, Khatib O. Progress and prospects of the human–robot collaboration. Auton Robots 2018;42(5):957–75. http://dx.doi.org/10.1007/s10514-017-9677-2.

[50] Kim SW, Kim I, Lee J, Lee S. Knowledge Integration into deep learning in dynamical systems: an overview and taxonomy. J Mech Sci Technol 2021;35(4):1331–42. http://dx.doi.org/10.1007/s12206-021-0342-5.

[51] Cuomo S, Di Cola VS, Giampaolo F, Rozza G, Raissi M, Piccialli F. Scientific machine learning through physics–informed neural networks: Where we are and what's next. J Sci Comput 2022;92(3):88. http://dx.doi.org/10.1007/s10915-022-01939-z.

[52] Paszke A, Gross S, Massa F, Lerer A, Bradbury J, Chanan G, Killeen T, Lin Z, Gimelshein N, Antiga L, Desmaison A, Köpf A, Yang E, DeVito Z, Raison M, Tejani A, Chilamkurthy S, Steiner B, Fang L, Bai J, Chintala S. PyTorch: An imperative style, high-performance deep learning library. 2019, http://dx.doi.org/10.48550/ARXIV.1912.01703, Version Number: 1.

[53] Corke P, Haviland J. Not your grandmother's toolbox – the robotics toolbox reinvented for python. In: 2021 IEEE international conference on robotics and automation. ICRA, Xi'an, China: IEEE; 2021, p. 11357–63. http://dx.doi.org/10.1109/ICRA48506.2021.9561366.

[54] Clochiatti E, Scalera L, Boscariol P, Gasparetto A. Electro-mechanical modeling and identification of the UR5 e-series robot. Robotica 2024;42(7):2430–52. http://dx.doi.org/10.1017/S0263574724000833.