systems

MDPI

Article

A Deep-Reinforcement-Learning-Based Digital Twin for Manufacturing Process Optimization

Abdelmoula Khdoudi [1,*], Tawfik Masrour [1,2], Ibtissam El Hassani [1,2] and Choumicha El Mazgualdi [1]

[1] Artificial Intelligence for Engineering Science Team, Moulay Ismail University, Meknes 50050, Morocco; c.elmazgualdi@edu.umi.ac.ma
[2] Mathematics, Computer Science and Engineering Department, University of Quebec at Rimouski, Rimouski, QC G5L 3A1, Canada; t.masrour@ensam.umi.ac.ma (T.M.); i.elhassani@ensam.umi.ac.ma (I.E.H.)
* Correspondence: a.khdoudi@edu.umi.ac.ma

Abstract: In the context of Industry 4.0 and smart manufacturing, production factories are increasingly focusing on process optimization, high product customization, quality improvement, cost reduction, and energy saving by implementing a new type of digital solutions that are mainly driven by Internet of Things (IoT), artificial intelligence, big data, and cloud computing. By the adoption of the cyber–physical systems (CPSs) concept, today's factories are gaining in synergy between the physical and the cyber worlds. As a fast-spreading concept, a digital twin is considered today as a robust solution for decision-making support and optimization. Alongside these benefits, sectors are still working to adopt this technology because of the complexity of modeling manufacturing operations as digital twins. In addition, attempting to use a digital twin for fully automatic decision-making adds yet another layer of complexity. This paper presents our framework for the implementation of a full-duplex (data and decisions) specific-purpose digital twin system for autonomous process control, with plastic injection molding as a practical use-case. Our approach is based on a combination of supervised learning and deep reinforcement learning models that allows for an automated updating of the virtual representation of the system, in addition to an intelligent decision-making process for operational metrics optimization. The suggested method allows for improvements in the product quality while lowering costs. The outcomes demonstrate how the suggested structure can produce high-quality output with the least amount of human involvement. This study shows how the digital twin technology can improve the productivity and effectiveness of production processes and advances the use of the technology in the industrial sector.

Keywords: digital twin; smart manufacturing; deep reinforcement learning; twin delayed DDPG algorithm; PPO algorithm; manufacturing process optimization

Citation: Khdoudi, A.; Masrour, T.; El Hassani, I.; El Mazgualdi, C. A Deep-Reinforcement-Learning-Based Digital Twin for Manufacturing Process Optimization. *Systems* **2024**, *12*, 38. https://doi.org/10.3390/systems12020038

Academic Editor: Ed Pohl

Received: 7 December 2023
Revised: 29 December 2023
Accepted: 5 January 2024
Published: 24 January 2024

1. Introduction

1.1. Background and Motivation

The term "Smart Manufacturing" refers to an approach in the manufacturing industry that seeks to combine information technology (IT) and operational technology (OT) in order to develop a production system that is more integrated [1], efficient, and responsive [2]. Digital twin is one of the main technologies that makes smart manufacturing possible (DT).

The definition of a digital twin (DT) originated in the aerospace industry and was originally described as the interactive equivalent of the physical system.

Nowadays, the idea of DT is starting to thrive in production. Going in this path, DT has taken on the role of tracking the overall life cycle of goods [3] or industrial processes [4]. Such advancements support remote sensing and the real-time control and monitoring of equipment and cyber–physical manufacturing elements through network infrastructures, and, thus, provide more direct alignment and coordination between the physical and the virtual worlds.

Several benefits can be derived from the adoption of the DT concept in organizations, and, in particular, the manufacturing sector [5], which provides advantages in an environment in which industrial players are forced to react more quickly, reduce their costs, show great flexibility [6], and be able to predict changes in their business. The aerospace and defense sectors are examples of sectors where the digital twin concept is deployed more. The automotive industry is one of the industries that still need further effort in DT integration.

Despite the diversity of the literature interpretation and definition of the DT concept, which can, furthermore, be very different depending on the sector concerned or on the intended application [7], many researchers share a common understanding of the existence of three levels of DT integration. According to developmental complexity, DT can be sub-categorized as follows: (1) a digital model which represents a static representation of the physical laws of the real system. It is mainly used for offline simulation and scenario studies. (2) A digital shadow is defined as a model that is continuously updated with sensor data [8]. The necessary data are collected by the real system, and these data allow for the virtual model to have a state closer to reality and not to stop on theoretical- or physical-law encoding modeling. This more precise model can be defined as a true DT. (3) A digital twin is where advanced analytics or optimization [5] are computed at the virtual level and then transferred back to the real system for optimization and functional improvement.

1.2. Research Contribution

During our exhaustive review of previous work in the field of manufacturing digital twins, we were surprised to discover certain gaps that we address with innovative contributions. First, while many approaches are proposed from a theoretical point of view without placing particular emphasis on the industrialization phase nor on the adaptability to various types of manufacturing processes, our methodology shifts away from traditional models reliant on physical laws. Instead, we advocate a data-oriented approach that captures the complex behavior of manufacturing systems through real-time data, addressing the need for adaptable solutions across various manufacturing processes. Second, where some methodologies for digital twins are extremely rigid, our research introduces the "Specific-Purpose Digital Twin", focusing on achieving particular objectives such as improving product quality or optimizing manufacturing parameters. This targeted approach enhances the feasibility and value of digital twins in diverse contexts.

Third, we noted a lack of substantial research on the decision-making aspects of digital twins. Our work fills this gap by integrating AI and machine learning for continuous improvement and intelligent decision-making. We emphasize the importance of precise machine parameters and settings, underlining the critical role of process stability and product quality. Our data-driven replica of the physical model and advanced traceability methods for data collection and pre-processing ensure an effective link between process inputs and quality outputs. By employing real-time predictions and model inference, our digital twin adapts proactively to maintain an optimal production quality, demonstrating a significant advancement over previous rigid and theoretical models. Thus, our paper provides a novel, adaptable, and practical framework for deploying digital twins in manufacturing, promising substantial improvements in efficiency, adaptability, and product quality.

1.3. Organization of the Paper

This paper is organized into six sections. Following the actual introduction section, we attempt, in the second section, "Literature Review", to present the results and understandings of our literature review. We derive two sections: in the first, we focus on a review of general works related to the digital twin technology and its various applications. The second part places more emphasis on research in the manufacturing subfield, and, especially, on the plastic injection molding process and how it has been approached as a digital twin use case. The third section of the paper describes our research methodology by focusing on the different steps that should be followed to create an autonomous AI-enabled digital twin for any kind of manufacturing processes. Modeling techniques like supervised

learning, deep reinforcement learning, and optimization are examples of topics that are detailed in this part. In the fourth section, attention is focused on the work results and performance evaluation of the machine learning models, in addition to the validation of the real-time optimization inference. In the fifth section, we interpret the results and their implications. We also highlight some limitations of our methodology and how it can be improved in future work. We then conclude this article by summarizing the research work and main contributions, with a list of some additional applications of our approach.

2. Bibliographic Study on Digital Twin Technologies

2.1. Literature Review on the Digital Twin Technology and Its Applications

In our bibliographic study, we considered 70 research articles available as open-access and published from 2013 to early 2023 on the theme "Digital Twin". We then classified them into four types: review, concept presentation paper, definition paper, or case study (Figures 1 and 2). Looking deeper into these types of contribution, we could classify the application area into eight areas, which are as follows: healthcare, layout planning, maintenance, manufacturing, production planning and control, process design, product lifecycle, and smart city (Figures 3 and 4).

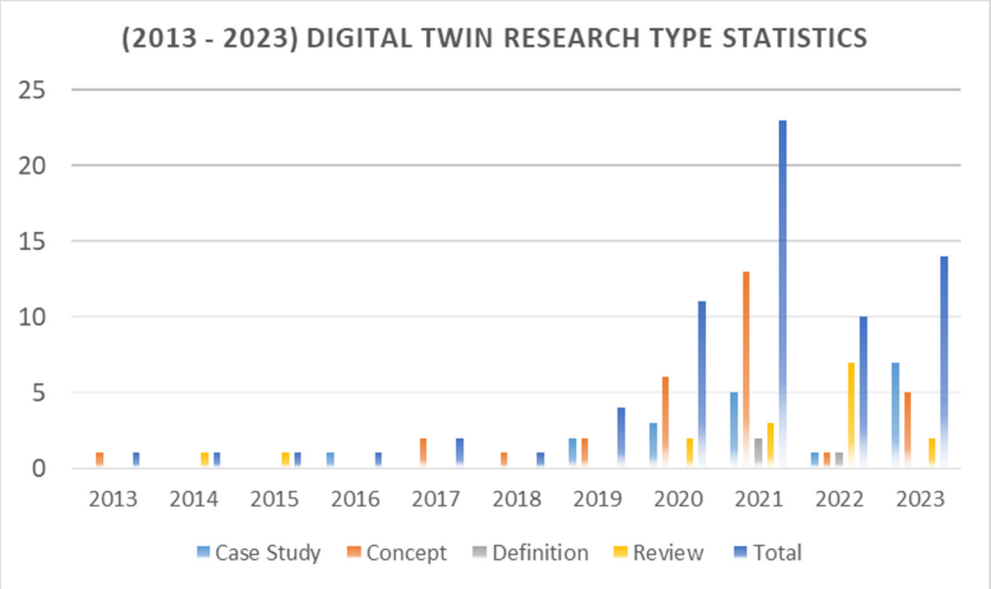

Figure 1. Digital twin research type statistics.

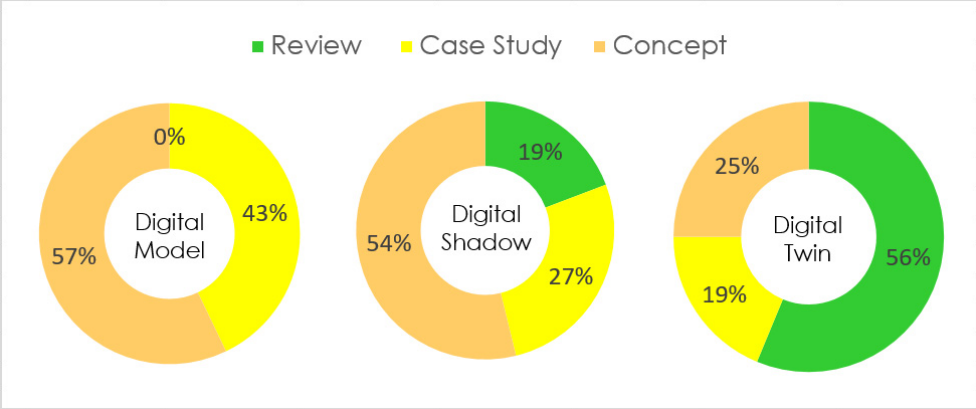

Figure 2. Ratio of research type by the digital twin level.

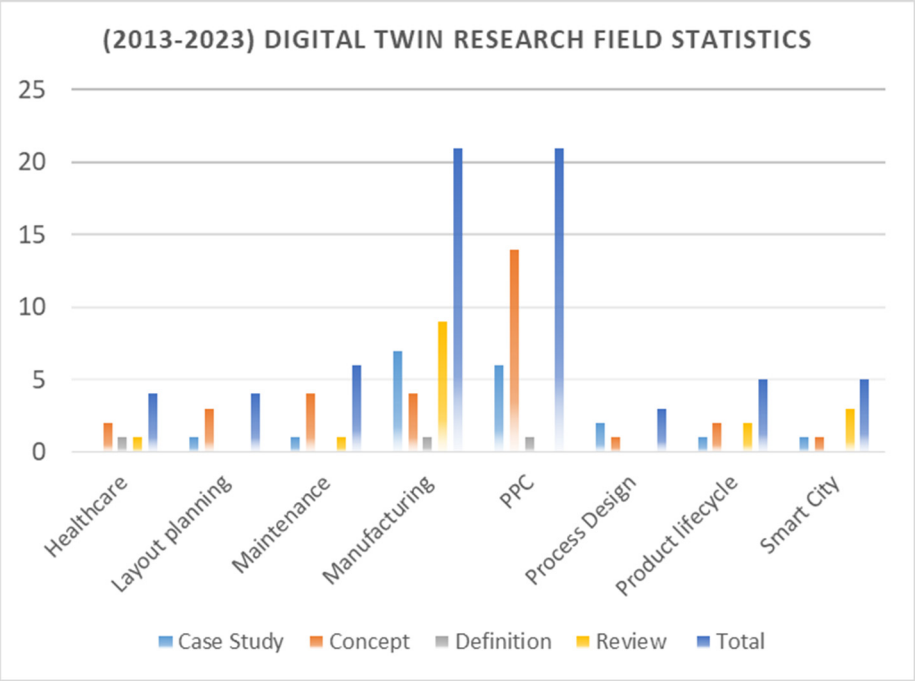

Figure 3. Digital twin research field statistics.

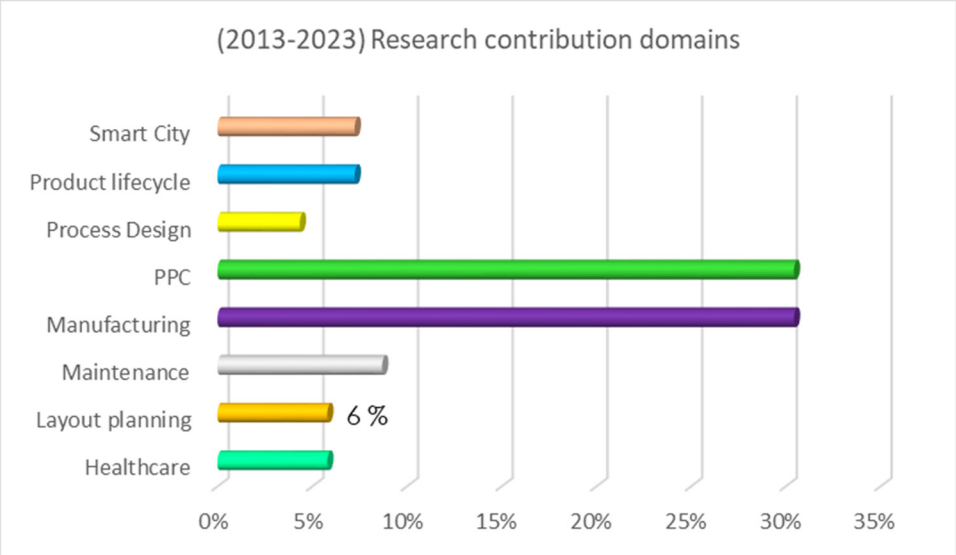

Figure 4. Digital twin research contribution domains.

We could easily see that the highest number of related studies was conducted from the end of the year 2019. Additionally, the number of conceptual studies and review articles was greater than real case study articles until 2022, where we could see an advancement in case studies compared to previous years. These positive changes in case studies contributions may be linked to the recent accelerated digital strategies launched by various companies and businesses across the world [9]; this would not have been possible before an in-depth development of a concept well established in previous years.

The research landscape for digital twins in manufacturing is vast and dynamic, with numerous studies proposing innovative strategies and methodologies. In [10], the authors suggest a data-driven model connecting process factors to performance metrics in thermal production, demonstrating increased effectiveness over current techniques. In [11], the authors propose a data-driven architecture using machine learning and process mining

to facilitate model creation and validation, thereby enhancing output and reducing costs. In [12], the authors focus on automatically generating digital twins from data logs for improved decision-making in supply chain management and production planning.

Ref. [13] delves into data ownership in industry, identifying reliable data sources for metal-based product manufacturing, while Refs. [10,14] explore the integration of various models into a digital twin for manufacturing, particularly LPBF, to enhance adaptability, optimization, and uncertainty management. Ref. [15] introduces a real-time scheduling technique for intelligent manufacturing, termed twins learning, which combines reinforcement learning with digital twins to maximize efficiency and responsiveness.

Ref. [16] presents a novel digital twin integrated into a multi-agent CPS for anomaly detection and bottleneck identification, improving human resource utilization rates by 30%. Ref. [17] suggests a digital twin combined with IoT for efficient fault diagnosis in safe production systems. Ref. [18] improves production-progress prediction precision using a digital twin model with a synchronous evolution based on dynamic sample data.

Ref. [19] presents a case study on digital twin applications in automobile fault prediction, utilizing real-time sensor data for future failure predictions. Ref. [20] studied tool wear prediction in milling machines using vibration data and a deep stacked GRU model. Ref. [21] explored the use of augmented reality (AR) for digital twin data exploration and control in CNC milling machines, enhancing operator interaction and monitoring.

Ref. [22] details the implementation of dynamic clamping and positioning in a diesel engine cylinder head production line, developing a digital twin model for decision-making. Ref. [23] models a machining process as digital-control features for machine tools' design improvement and operation optimization. Finally, Ref. [24] proposes a digital twin model for closed-loop feedback optimization in aircraft spraying systems, improving planning and operation efficiency through knowledge engineering and the OPC-UA protocol.

This body of work collectively advances the field of digital twins in manufacturing, offering diverse approaches to enhance process control, efficiency, and adaptability across various industry applications.

2.2. Related Work on Digital Twin for Injection Molding Process

Researchers have carried out numerous works related to the design and implementation of a DT in the specific area of the industrial injection molding process. Ref. [25] presented a general connectivity and business value relationship between mold design, mold manufacturing, and production process execution. In their work, they emphasized the importance of connecting the three stages in order to share knowledge and benchmarking. The work focused on information flow and data gathering without going further into data mining or insight generation.

On other hand, Ref. [26] explores the benefits of a digital twin system in the monitoring of the plastic injection molding of micro parts. They took on the complexity of ensuring production quality in the micro-manufacturing industry as a challenge, which justified the implementation of a digital twin facilitating data collection and process monitoring. As a result, the work did not explore or study performance optimization or the decision-making mechanism of the digital twin.

Ref. [27] investigated the degree of digitation and intelligence of the injection molding industry in China, which showed low production performances. One of the main goals of this work is to propose an industrial Internet system architecture as a solution for smart-factory-based digital twins [28] in order to manage intelligent equipment, intelligent production lines, workshops, factories, and formats.

The work presented in this article is well detailed from a connectivity point of view and can be considered an excellent conceptual foundation for intelligent manufacturing internet architecture. However, it is still far from being applicable in practice and requires additional engineering steps.

On the other hand, such work would be classified as a conceptual connectivity architecture, which is a different research field than our present work. In the present paper,

we try to fill the research gap by focusing on the use of artificial intelligence and machine learning techniques to develop an intelligent digital twin for automated decision machines. To overcome the complexity of modeling the physical laws of the real system, we detail our data-driven approach to create a big data technology based on special-purpose digital twins coupled with supervised model training to learn the specific phenomenon. The learned phenomenon is then predicted in real time and optimized for the case of deviation using a deep reinforcement-learning algorithm.

3. Digital Twin Design Methodology

3.1. General Description

The current section describes our methodology for designing a transparent AI-based digital twin for manufacturing processes. The goal of this data-driven approach is to go beyond the standard methodology described in several research studies which asserts the need to model the physical laws of the real system within the virtual representation. Today's complex manufacturing systems are difficult to model numerically using physical equations and require a different type of behavior coding based on real-time data.

In a typical manufacturing line, one or several processes apply different transformations to an input material in order to reach the desired final product. This product should validate the customer's quality requirements. For such reasons, a good process design and control should be established and maintained. Some of the core aspects of process stability are mainly related to machine parameters and settings, settings that should be precisely defined and constantly monitored to minimize deviations.

During the production flow, the physical system is made up of one or more process steps that add value to the product. The product quality is also assessed after each step or at the end of the production flow using automated inspection systems [29] or, in some cases, human-based inspection. A product's quality inspection process (with respect to quality insurance) is used to make several types of measurements for the production specification.

The process output quality ("product quality") is generally impacted positively or negatively by any improper changes to the raw material, the production environment, the work methodology, the machine parameters, or the machine conditions. For the cases of complex industrial processes where little human intervention is needed, production quality depends to a large extent on the processing parameters (machine parameters), especially when dozens or even hundreds of processing variable inputs are involved. A large number of manufacturing parameters makes it difficult or impossible for humans to perceive and master. In Figure 5, we presents the architecture of our proposed specific-purpose DT.

Digital Shadow Section:

Data Ingestion and Processing:
This area manages the input of real-time process and product data, including batch and sensor data. The data are processed and prepared for further analysis using data cleaning, aggregation, and preprocessing techniques.
Supervised Learning Model Training:
Taking data as batches (in a periodic manner), the machine learning models are trained with the ingested data to predict product quality as process outcomes. The models are updated continuously with new data to keep the latest representation of the physical-system behavior.
Inference Model for Real-Time Prediction of Production Quality:
The trained models are used to make real-time predictions about production quality.
Decision Engine Section:
Deep Reinforcement Learning Environment:
This area represents an environment where deep reinforcement learning (DRL) models are trained through interactions with the supervised learning models. These models are used to make decisions that optimize the manufacturing process (taking the quality optimization as the "specific-purpose" of this DT example).
Deep Reinforcement Learning Training:

This is the training phase for the DRL models where they learn to make decisions that will eventually optimize production quality and process efficiency. The training phase of the DRL models is triggered once a new environment is created (new supervised learning model generated).
Optimization Inference Section:
DRL Inference Model:
The trained DRL model is deployed in this section to make inferences. The inference at this stage means the proposal of process adjustment settings to achieve a better product quality.
Processing Settings Optimization:
Based on the inferences made by the DRL model, the processing settings of the manufacturing equipment can be optimized.
Integration of Changes Within the Physical System:
This is the most valuable part where the "autonomy" arises. The optimized process settings propagate down to the physical system through the "write" methods of the industrial communication protocols (OPC UA/MAQTT).
Deviation from Quality Target:
To guide the decision model, a function that computes the difference between the predicted product quality and the quality targets is established, providing continuous feedback for further optimization.
Part-Level/Batch-Level Traceability System:
The goal of this system is to maintain the records of each part (part-level) or batch (batch-level) through the manufacturing process for quality assurance and traceability.
Feedback Loop:
This is the mechanism by which the system learns and improves over time. The parts' quality results and production quality assessment feed back into the system to refine the decision-making models.
Connectivity and Data Flow:

Various types of data flows are represented by different colored lines:

- Blue lines indicate material flow through the process steps.
- Green lines show traceability data flowing through the system.
- Purple lines suggest actions taken in the physical system based on the optimization.
- Orange lines represent cleaned and processed data flowing to the prediction model.
- Black lines indicate the generated supervised model (that serves as a DRL environment).
- The gray lines indicate the internal model's variable exchanges.

3.2. The "Specific-Purpose Digital Twin" as Key for Fast Adoption

In today's factories, the adoption and implementation of concrete digital twin solutions is still rare despite the rapid spread of the theoretical concept. According to the literature, such slow adoption is due to the existence of conceptual ambiguities in addition to several implementation challenges.

A general-purpose DT requires a full and "loyal" representation of the physical system regardless of the searched value and usage [30]. This breaks the feasibility of such project at the first modeling constraint (e.g., modeling the physical laws behind a complex industrial machine). Once we try to overcome such challenges by simplifying the large constraints (skipping the physical aspects that cannot be modeled), we get confronted by the loss of fidelity and the drop in accuracy and value.

Making the DT concept more purpose-specific is our core idea behind this work in order to narrow the implementation target while increasing the expected value and making it more accurate.

In this direction, improving product quality, reducing process energy consumption [31], and optimizing the manufacturing process parameters [32] become examples of goals that require a specific design and implementation of a digital twin. By this, a specific-purpose digital twin becomes a system that is designed to achieve a specific objective or function. In this way, it becomes affordable to model a function of a system instead of

modeling the entire system. In the present study, we demonstrate our approach on a real case where we try to model and implement an autonomous DT for a manufacturing process (plastic injection molding as a use case), taking as target the optimization of process values to get the optimal production quality.

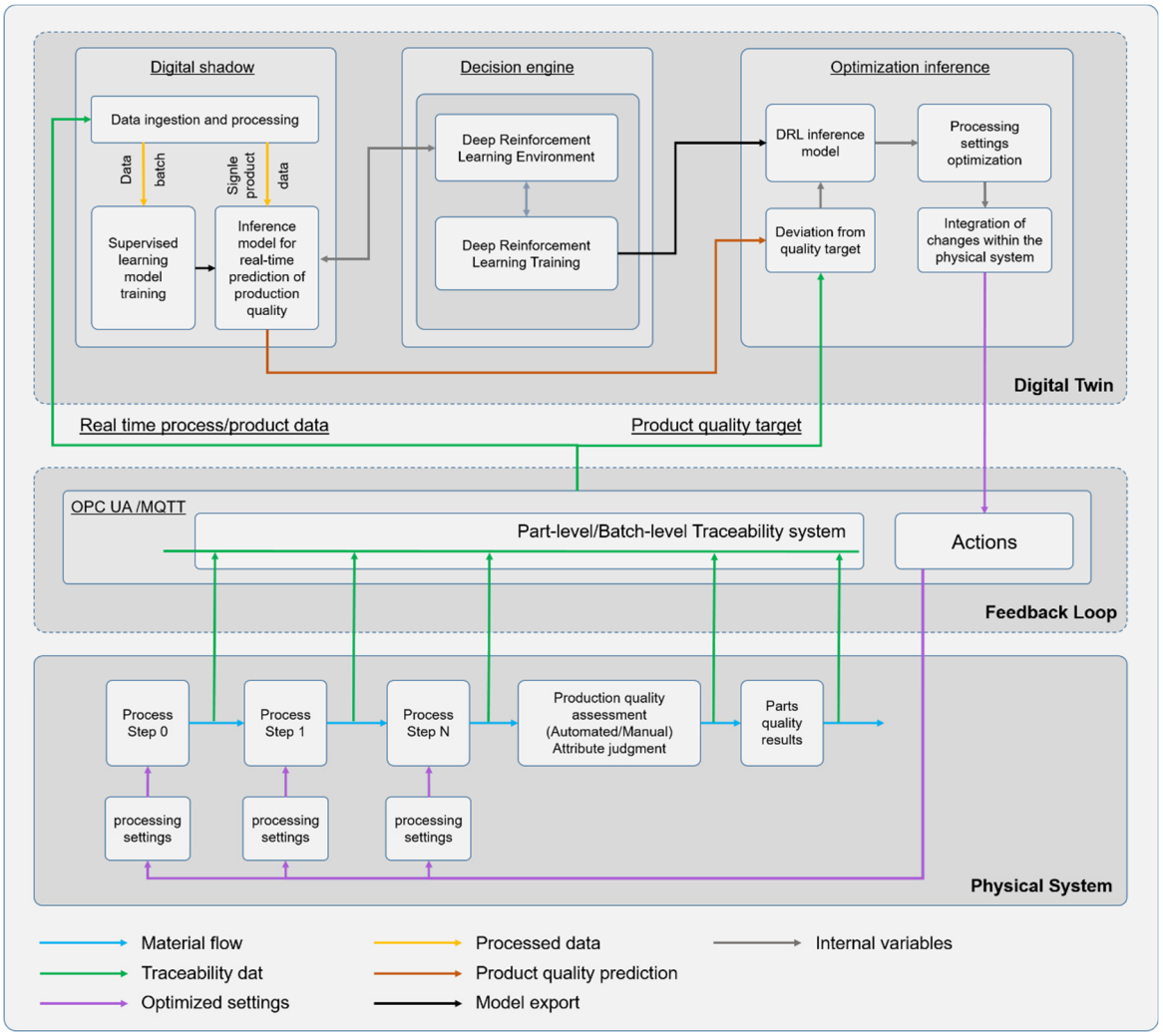

Figure 5. Our proposed architecture for specific-purpose digital twin.

3.3. Data Collection and Pre-Processing

Our approach makes usage of the traceability concept for the process and quality inspection data collection. This ensures accurate mapping between process inputs "adjustment process values" and process output "product quality measurements" through a product or batch identification. Several product/lot identification techniques are used in manufacturing today; the most common ones can be categorized as below:

Parts identification by labels: This technique makes usage of label printers to create 1D or 2D codes that are stuck on each product or batch for later identification. The goal is typically to hold basic information such as the production date and time, the product's unique ID, and any additional information that can help identify and trace the product in later processing steps. Such labels can also hold textual identification of the product.

Parts marking: Unlike the previous technique which creates a new element to stick onto the product, part marking uses the surface of the product itself to hold the information using inkjet or laser marking. This technique is very useful for parts where the presence of labels can affect product quality or can simply be damaged during further manufacturing steps (e.g., hot environment, harsh manipulation, etc.).

Parts wireless sensing: Modern techniques like RFID product tracking are widely used in production facilities where the product should hold active information. Identification using an RFID tag can allow us to read/write the identifier's content based on the processing condition, which makes it more adapted to flexible production lines [33].

Virtual tracking: This is used for some special types of products where no additional labels can be held or where no surface marking or smart tags are used. Virtual tracking can be ensured using the manufacturing PLC (programmable logic controller) as a memory-based transfer of part identifiers (part IDs). This concept is based on machine-to-machine communication where each cycle-start triggers the creation of a new unique part identifier. The virtual identifier is transferred using legacy data communication between manufacturing steps during the flow of production parts. This method becomes limited when the parts should be physically transferred outside a connected production flow (e.g., shipped to customers) [34]. In this case, mapping back a non-physically identified part to a past virtual identification history becomes non-feasible.

For an effective traceability system design, the read/write step of product identification should be supported at the entry and exit of each process step. This allows for a good tracking and mapping of the processing parameters to the quality results of the product through the corresponding unique identification.

3.4. Data-Driven Replica of the Physical Model

In the proposed AI-based digital twin with a specific-purpose approach, data are collected from the production process and encapsulated with respect to traceability format where keys represent product identifiers and columns with sub-columns represent the process steps with the corresponding parameter values. Each product identifier row is assigned the quality inspection results for each step (or the end-of-line global result). This data flow is ensured through modern industrial internet standards and protocols, mainly the OPC UA and MQTT for our practical architecture. The previously described components ensure the bottom-up flow of data and information, which constitutes the first part of the digital twin feedback loop.

The full flow of the real-time process and quality data is consumed by the digital shadow according to different frequencies. One is the batch transfer of cleaned data to re-train a supervised learning algorithm in order to learn the relationship between the process values and the quality output. The training process is performed periodically to keep an updated representation of the physical system with respect to a specific context. The second flow "also called single-part data" is used for the real-time prediction and model inference of new parts.

The trained machine learning model is then used for the real-time prediction of product quality result. In a case where the predicted value is far from the target physical value, the physical parameters must be adjusted to meet the targeted prediction, a step where the optimization loop is triggered.

3.5. Deep Reinforcement Learning for Automated Decision-Making

The optimization inference is performed by the deep reinforcement learning model which is trained to find the optimal policy to reach the best set of input parameters that result in the desired product quality [35].

This strategy involves iterative adjustments and learning from the system's performance to continuously refine the manufacturing process, aiming for the highest-quality outcomes. The model's ability to adapt and optimize based on real-time data exemplifies the power of integrating advanced AI techniques in industrial settings.

The DRL algorithm uses the supervised model as the training environment where the observation space is composed from the manufacturing process values and the corresponding quality output. The action space is the set of discrete or continuous actions performed to iterate over the parameter modifications, while the reword function is set to maximize the model convergence to the best production quality. The present work explores

two main RL algorithms, which are PPO (proximal policy optimization) and twin-delayed deep deterministic policy gradient (or TD3).

This research delves into the intricacies and effectiveness of these advanced algorithms, dissecting their mechanisms and potential impacts in enhancing decision-making processes within sophisticated systems.

As far as our knowledge, this is the first digital twin-related work that explores the performance of the twin delayed DDPG algorithm. Taking the gap between the real target of the quality attribute and predicted value coming from the digital shadow, the DRL inference model proposes the adjusted manufacturing input parameters to reach the best production quality. This approach not only identifies discrepancies but actively works to rectify them by suggesting optimal adjustments, thereby ensuring the manufacturing process consistently meets the desired standards of quality.

The suggested values are then processed in the top–down flow in a feedback loop and reach the physical machines through the IT/OT network, closing the loop of the full-duplex digital twin system.

4. Digital Twin Implementation Methodology

4.1. Theoretical Background: Deep Reinforcement Learning

In the field of artificial intelligence, reinforcement learning (RL) is a promising class of learning methods that address complex sequential decision-making challenges. The main idea of RL algorithms is about providing an artificial agent with the ability to learn how to behave in a given environment through trial-and-error interactions. Thus, the artificial agent learns from interacting with the environment how to map situations into actions that maximize the long-run numerical reward signal without having to assume full knowledge of the environment.

Essentially, most RL algorithms are formalized using the Markov decision process (MDP) [36,37]. An MDP is usually described by the tuple (S, A, P, R, γ), where S is the state space and A denotes the state actions. P consists of the transition function such that $P(s_t, a_t, s_{t+1}) = p(s_{t+1}|s_t, a_t)$ is the transition probability from state s_t to the following state s_{t+1} taking action a_t. Finally, R stands for the reward function, and γ is the discount factor. Formally, RL refers to the process by which an artificial agent learns an optimal policy π for solving MDP problems. At each time step t, the agent interacts with the environment by applying an action a_t according to the policy π, and the environment returns the next state s_{t+1} and the reward r_{t+1}. Using the experience gathered, the agent updates its policy such that the expected reward is maximized.

Based on the methods used, reinforcement learning can be broadly divided into value-based reinforcement learning and policy gradient algorithms. Value-based approaches learn the optimal policy by optimizing the value function. On the other hand, policy gradient algorithms learn the optimal policy through computing policy gradients.

4.1.1. Value Optimization

Value-based reinforcement learning typically involves alternating between estimating the state-value function under the current policy and optimizing the policy with the estimated value function. The state-value function $V_\pi(s)$ is defined as the expected return when starting in state s and acting according to policy π:

$$V_\pi(s) = E[R_t|s_t = s] \tag{1}$$

From the definition of the expected return, the optimal value function can be defined as:

$$V^*(s) = \max_{\pi \epsilon \Pi} V_\pi(s) \tag{2}$$

Similarly, the state-action *value* function or quality function $Q_\pi(s, a)$ is defined as the expected return starting in state s, taking action a, while following policy π. This function

provides a more complete estimate since it includes the action to perform for an arbitrary state. The state-action value function can be expressed as follows:

$$Q_\pi(s,a) = E[R_t|s_t = s, a_t = a] \tag{3}$$

In the same way, the optimal Q-value function can be expressed as follows:

$$Q^*(s,a) = \max_{\pi \epsilon \Pi} Q_\pi(s,a) \tag{4}$$

Therefore, the optimal policy can be obtained directly based on the previous equation:

$$\pi^*(s) = \underset{a \epsilon A}{\operatorname{argmax}} Q^*(s,a) \tag{5}$$

Typical value-based algorithms include TD learning [38], Q-learning [39], and their variants.

Q-Learning Methods

The Q-learning algorithm introduced by Watkins (1989) is considered to be the basic and common technique for estimating the Q-value function. It consists of representing the Q-values in a lookup table in which each state–action pair has one entry. This setting makes use of the Q-learning limited to finite and discrete action–state spaces. Thus, for addressing more complex and relatively large-scale problems, function approximation approaches are required to represent the Q-value function.

Deep Q-Network (DQN)

The Deep Q-Network (DQN) [40] uses a deep neural network (DNN) to approximate the value function $Q(s,a)$ with a parameterized value function $Q(s,a,\theta)$ satisfying $Q^*(s,a) \approx Q(s,a,\theta)$. Thus, the optimal value function $Q^*(s,a)$ is learned by minimizing the following loss function:

$$L_i(\theta_i) = E\Big[\big(r + \gamma max Q(s',a',\theta_{i-1}) - Q(s,a,\theta_i)\big)^2\Big] \tag{6}$$

where θ refers to the neural network parameters.

Combining the RL with the DNN provides robustness and better performance over a wide range of complex decision-making tasks. However, this combination induces some disadvantages such as the instability of the learning process and nonstationary data distribution. For handling these issues, the DQN makes use of two key components, namely, the replay buffer and the target network. The replay buffer provides learning stability by storing experience transitions and then sampling them uniformly to update the policy. The target network is a separate network parameterized using a copy of the regular network and updated in a delayed manner to avoid the problem of nonstationary data distribution.

4.1.2. Policy Optimization

Policy Gradient (PG) Methods

Policy gradient (PG) methods are commonly used model-free policy search algorithms, optimizing a policy with a gradient estimator of the expected return [41].

In contrast to value-based methods, policy gradient methods directly optimize the agent's policy without learning the value function [42]. They consists of representing the stochastic policy by a parametric probability distribution $\pi_\theta(a,s) = P[a|s;\theta]$; then, an estimator of the expected return gradient with respect to θ is obtained from sample trajectories. Finally, by adjusting the policy parameters in the direction of the estimated gradient, the policy is updated with respect to the below surrogate performance objective:

$$L^{PG}(\theta) = \hat{E}_t\big[\log \pi_\theta(a_t|s_t)\hat{A}_t\big] \tag{7}$$

where $\hat{A}_t$ is the estimated advantage function of state s and action a in time step t. Generally, it is equal to the discounted rewards minus a baseline estimate. Intuitively, the baseline estimate is replaced by the value function V(s). Thus, the advantage function provides a measure of comparison for each action to the expected return following policy π.

Deterministic Policy Gradient (DPG)

The deterministic policy gradient is a special case of the stochastic policy gradient (SPG) [43] described above. While in the SPG the policy function $\pi_\theta(a, s)$ is represented by a probability distribution, the DPG instead models the policy as a deterministic action $a = \mu_\theta(s)$. In this sense, the DPG extends the standard policy gradient theorems to deterministic policies under certain conditions [44]. By integrating only over the state space, the DPG can be defined as the expected gradient of the action-value function [43,44]. From a practical point of view, the gradient in the deterministic case can be estimated much more efficiently than in the stochastic case, requiring fewer samples, especially if the action space has many dimensions.

A key challenge of policy gradient methods is about estimating the proper step size to perform policy updates; small values may negatively impact the convergence rate, while large values may imply the oscillation divergence of the policy. Since most PG-based methods suffer from the step size pitfall, a trade-off between learning speed and learning stability is still a promising area to explore for improving PG methods.

Trust Region Methods

To further improve PG-based methods, the trust region policy optimization (TRPO) method, an extension of the policy gradient method, was developed. It consists of imposing a trust region constraint onto the objective function to control the step size of the policy update using the Kullback–Leibler (KL) divergence [45]. The use of a KL divergence constraint limits the difference between successive policies, which leads to monotonic performance improvements [46]. The objective function of TRPO is given by:

$$\max_\theta \hat{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right] \tag{8}$$

$$Subject\ to : \hat{E}_t \left[KL\left[\pi_{\theta_{old}}(.|s_t), \pi_\theta(.|s_t) \right] \right] \leq \delta \tag{9}$$

where $\pi_{\theta_{old}}$ is the old policy before updating, and π_θ is the new policy.

Despite its good performance and stability, TRPO is still a complicated algorithm since it relies on second-order approximations, which makes it computationally expensive and difficult to extend to complex network architectures.

Proximal Policy Optimization (PPO)

The proximal policy optimization (PPO) [47] is an enhancement of the TRPO algorithm, adopting a clipped surrogate instead of using the KL divergence constraint to reduce the TRPO complexity. The clipped surrogate function is obtained by approximating the probability ratio between the actual policy and the old policy and then applying a clipping range into the original objective function to prevent large policy updates. The PPO is considered a simplified version of the TRPO, providing faster convergence, easier implementation, and better sample efficiency. The clipped surrogate objective function of the PPO is expressed as follows:

$$L^{Clip}(\theta) = \hat{E}_t \left[min\left(r_t(\theta)\hat{A}_t, clip(r_t(\theta), 1-\varepsilon, 1+\varepsilon)\hat{A}_t \right) \right] \tag{10}$$

where the probability ratio $r_t(\theta)$ is defined as:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{11}$$

By clipping the objective function, the policy update is limited to prevent large changes. Compared to the KL divergence [48], the surrogate objective of the PPO has been shown to offer improved performance while constraining the step size update in a much simpler manner, thus achieving state-of-the-art results on a variety of challenging reinforcement learning tasks.

4.1.3. Actor–Critic Methods

The combination of value-based and policy-based optimization forms is one of the most effective and popular architectures in reinforcement learning called actor–critic [49]. The actor network and the critic network are trained simultaneously by employing the optimization of the value function as a baseline for policy improvement. The actor interacts with the environment and learns to select the best actions using the critic's feedback [50]. On the other hand, the critic evaluates the actions selected by the actor network by estimating the value function of the current state of the environment [44]. In doing so, the actor–critic method absorbs the merits of both value-based and policy-based methods providing improved stability and convergence allied with the ability to handle high-dimensional state and action spaces in a robust way.

Advantage Actor–Critic (A2C)

One of the most popular and effective actor–critic variants is the synchronous advantage actor–critic (A2C) [51] focusing on parallel training.

Figure 6 shows the architecture of a synchronous advantage actor–critic (A2C) (on the left) and its asynchronous version (A3C) (on the right).

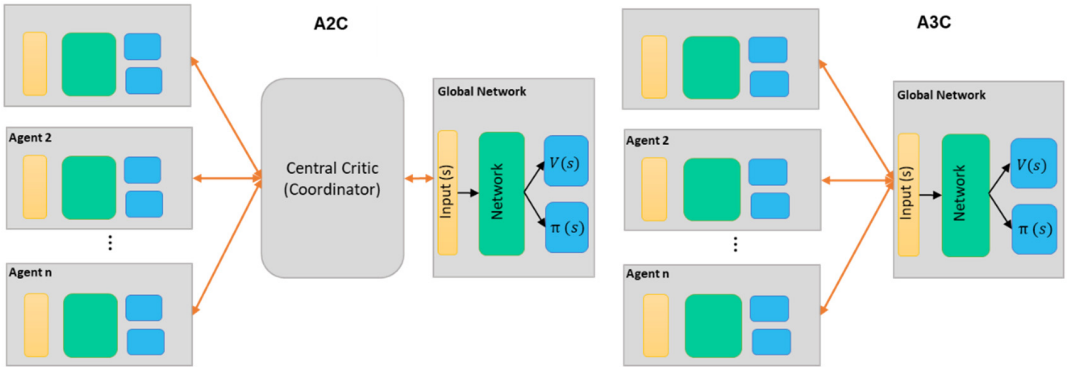

Figure 6. Simplified advantage actor–critic architecture (A2C on the left, A3C on the right).

A2C: This model has multiple agents (Agent 1 to Agent n) that interact with the environment and send their experiences to a central critic (coordinator). The central critic evaluates the actions of each agent based on a global policy ($\pi(s)$) and a value function (V(s)), which estimates the expected return. This information helps the agents learn the optimal policy.

A3C: Similar to A2C but asynchronous. Each agent has its own instance of the global network and interacts with its own copy of the environment independently. The agents asynchronously update a global network which also includes a policy and value function. This allows for faster and more diverse learning since the experiences of the agents are less correlated.

In the two versions, the learning is based on the actor–critic approach using an advantage function to update the policy. The advantage function represents the quality of taking an action in the current state while considering the average expected return. The two algorithms use parallelism to accelerate the learning process, and then all the agents communicate with the global network also called the master node. The main difference between A2C and A3C is that, for A2C, we have a coordinator that receives experiences from all the agents, and once completed, it starts a one-step learning of the global network

(the actor and the critic at the same time), thus providing a synchronous learning process using one shared environment. On the other hand, in the A3C algorithm, the global network is updated asynchronously by the agents that interact, each with its own copy of the environment, and then send their experiences independently to the master node.

Deep Deterministic Policy Gradient (DDPG)

The deep deterministic policy gradient (DDPG) [52] is an extension of the DPG method combining actor–critic and DQN algorithms to train a deterministic policy actor using a learned estimate of the state–action value function provided by the critic. Compared to other actor–critic methods, the DDPG shows improved stability and convergence by using the two key components of the DQN, namely, the replay buffer and the target network. As mentioned in the DQN section, the replay buffer is used for storing the agent's experience transitions and then sampling them uniformly to update the actor and critic networks. The target networks stand for separate networks parameterized simultaneously with a copy of the actor and critic networks and updated in a delayed manner to provide learning stability. The DDPG consists of an iterative boucle of updating between the actor and critic networks. The critic network is updated using sampled experiences from the buffer replay by minimizing the following loss function:

$$(\theta_Q) = E\Big[\big(y - Q(s, a; \theta_Q)\big)^2\Big] \tag{12}$$

where y is the target value based on the target critic network and expressed as follows:

$$y = r + \gamma Q'\big(s', \pi'(s'; \theta_{\pi'}); \theta_{Q'}\big) \tag{13}$$

where r is the immediate reward; θ_Q, and $\theta_{Q'}$ are the parameters of the critic and target critic, respectively. Similarly, θ_π, and $\theta_{\pi'}$ are the parameters of the actor and target actor, respectively.

On the other hand, the actor network updates its parameters by maximizing the expected Q-value of the current policy:

$$L(\theta_\pi) = E\big[Q\big(s, \pi(s; \theta_\pi); \theta_Q\big)\big] \tag{14}$$

In addition, the DDPG uses a soft update for updating the target networks' parameters:

$$\theta_{Q'} \leftarrow (1 - \tau)\theta_{Q'} + \tau\theta_Q \tag{15}$$

$$\theta_{\pi'} \leftarrow (1 - \tau)\theta_{\pi'} + \tau\theta_\pi \tag{16}$$

where $\tau \ll 1$.

Twin Delayed DDPG (TD3)

The twin delayed deep deterministic (TD3) policy gradient algorithm [53] is an enhancement of the DDPG algorithm by introducing three key techniques to improve stability and convergence. The combination of three powerful deep reinforcement learning methods, namely, policy gradient, double deep Q-learning, and actor–critic, makes TD3 one of the most robust and efficient actor–critic algorithms. First, TD3 incorporates the idea of clipped double Q-learning by establishing two separate critic networks to compute the value of the next state. Twin critic networks help reduce the overestimation of Q-values and improve learning stability thanks to the clipping function. Second, it uses target policy smoothing regularization which consists of adding a small amount of noise to the target policy to smoothen the Q-value function and prevent overfitting. Finally, it applies delayed policy updates to optimize the update of the actor and critic networks by updating the policy and the target network less frequently than the critic network, which provides reduced variance of the Q-value function.

4.2. Industrial Use Case: Plastic Injection Molding

4.2.1. Problem Description

Our real-case application takes place in the plastic injection molding process of an automotive manufacturing company. This selection was motivated by the analysis of the market's future development and trends. The growing demand for plastic components from various end-use industries including automotive, packaging, home appliances, electrical and electronics, and medical devices is anticipated to drive the market's growth. The largest sectors that make intensive use of the injection molding process are the automotive and transportation, packaging, consumables, and electronics sectors. According to the Grand View Research platform (Injection Molded Plastics Market Size Report, 2022), today's market size value is USD 303.7 billion, with a growth forecast of USD 423.7 billion in the year 2030.

From a manufacturing perspective, the plastic injection molding process is a forming operation using molds (Figure 7). A material, such as synthetic resin (plastic), is heated and melted and then transferred to a mold where it is cooled to the desired shape. This process owes its name to its resemblance to the injection of liquids using syringe. Plastic injection takes place as follows: the material is melted and poured into the mold, where it cools; then, the products are extracted and finished.

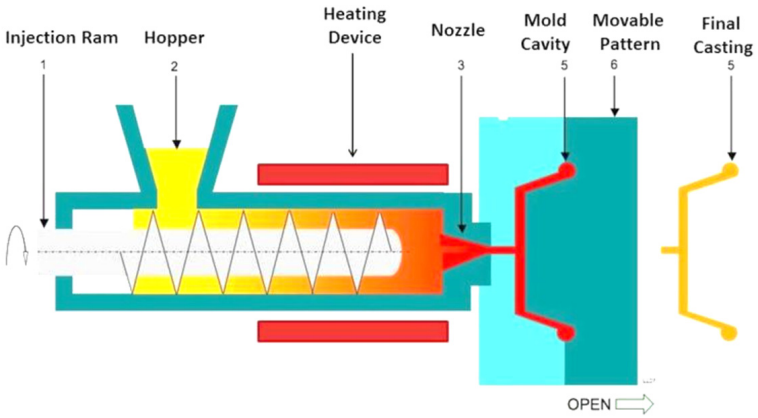

Figure 7. The main components of an injection molding machine.

Getting a stable production quality for molded parts requires great mastering of the injection molding process. This is a common requirement for any manufacturing process where the material, settings, methods, human, and medium each have their impact on the stability of the process. At this stage, the molding parameter settings play a vital role and must be monitored and optimized over time to achieve a stable product quality.

Below are some of the molding parameters that require special attention due to their high impact on the quality output:

Mold Filling Time: This parameter is defined by the speed of the rotary injection screw. It represents the time needed to fill the mold cavity with the melt plastic. The filling time is a value that is influenced by a variety of factors like the mold design and the plastic material type and properties. A long filling time can cause several quality issues including warping, burning, or discoloration. On the other hand, a short filling time can lead to an incomplete filling of the mold cavity, which gives an incomplete part or a part with voids, sink marks, or other defects. It is, thus, crucial to optimize the discussed parameter.

Mold Temperature: The mold is the name of the tool used inside the injection machine to hold and give the final shape to the injected part. It is generally made using a kind of steel with a cavity inside to receive the melt material. The mold temperature is an important parameter with a high impact on the product quality. This comes from the fact that it directly influences the cooling time after the injection operation. In addition, the shrinkage of the part and the final appearance are also aspects that are improved or degraded by the adjustment of the mold temperature.

Material Melt Temperature: This is a measure that represents the temperature at which the plastic used in injection molding melts and can be injected into the mold. The material melt temperature impacts product quality by affecting material flow, mold filling, and drying time. Warping, sink lines, and holes can occur if the melt temperature is too high or low. Thus, to produce high-quality molded parts with uniform qualities, material melt temperature must be controlled.

Injection Cycle Time: This refers to the total amount of time it takes for the plastic material to be injected into the mold, allowed to cool, and then released as a finished product. Defects such as sinks, warps, and colorations are kinds of consequences of a not-controlled cycle time [54]. The cycle time value is highly correlated with other aspects of the injection molding like the cooling time and the mold filling time.

Mold Closing Force: Injection molding requires a certain level of power to close the mold halves. The mold closing force affects product quality by affecting the mold parting line, dimensional accuracy, and flare or burrs. A large force can cause the deformation of the mold of cracks, while a low force can cause an incomplete closure of the mold, which can lead to the occurrence of burrs and flashes.

Injection Pressure: When molten plastic gets injected into the mold, the "Injection Pressure" parameter determines the pressure. Product quality is affected by injection pressure, which affects mold filling behavior, parting line, and part dimensions. Incomplete mold filling or gaps can occur if the filling pressure is too low. Otherwise, a high pressure value can cause excessive flashes or burrs, damage the mold or machinery, and increase production costs.

Other important parameters like the injected volume, the torque of the injection screw, and more also play very important roles, and all should be controlled and optimized continuously for a constant production quality and line performance.

4.2.2. Data Collection and Exploration

The dataset involved in this case study was collected through a traceability system that was connected to several machines of the ANGEL brand, model CC-300. The data exchange between the machines and the traceability system occurs via the integrated OPC servers within the machine's PLC (programmable logic controller). The data were collected over several days; they represent the machine parameters (Table 1) and the related product quality for 1451 produced items (of various production references).

Table 1. Preview of the training data collected and used in this study.

Index	MOLD TEMP	FILL TIME	PLAST TIME	CYCLE TIME	CLOSING FORCE	CLAMP FORCE PEAK	TORQUE PEAK VAL
1	126.04	11.04	4.96	115.96	1412.05	1440.73	177.94
2	126.23	10.64	4.93	115.97	1399.81	1425.85	186.78
3	124.96	10.64	5.92	115.97	1374.7	1387.72	186.78
4	125.64	10.64	4.99	115.99	1399.81	1454.21	183.52
5	125.9	10.8	4.93	115.96	1411.59	1437.01	177.48
6	124.96	10.64	4.93	115.97	1363.09	1387.72	197.32
7	126.28	10.64	4.95	115.96	1374.7	1390.04	183.52
8	125.8	10.64	4.93	115.96	1408.33	1435.77	176.55
9	126.3	10.8	4.94	115.97	1366.33	1391.75	182.59
10	126.08	10.8	4.93	115.97	1427.04	1401.2	197.32
11	126.01	10.64	4.9	115.96	1371.13	1400.43	185.38

Table 1. *Cont.*

Index	MOLD TEMP	FILL TIME	PLAST TIME	CYCLE TIME	CLOSING FORCE	CLAMP FORCE PEAK	TORQUE PEAK VAL
12	124.96	11.04	4.93	115.97	1364.00	1398.6	183.52
13	126.11	10.8	4.94	115.96	1426.33	1435.77	162.13
14	124.64	10.8	4.94	115.97	1372.19	1390.04	181.2
15	126.23	10.8	4.94	115.97	1424.72	1454.21	186.78
Index	**TORQUE MEAN VAL**	**BACK PRESS PEAK VAL**	**INJECT PRESS PEAK VAL**	**SCREW HOLD POS**	**SHOT VOLUME**	**QUALITY INDEX**	
1	162.13	225.84	1431.74	13.69	29.03	0.128	
2	162.6	229.56	1446.31	13.66	29.42	0.296	
3	162.6	225.99	1436.08	13.58	29	0.288	
4	162.6	226.3	1442.28	13.64	29.02	0.3	
5	164.3	227.54	1437.94	13.69	29.05	0.168	
6	162.6	227.39	1367.57	13.67	29.06	0.284	
7	164.61	226.3	1367.57	13.7	29.03	0.28	
8	162.6	226.3	1453.44	13.72	29.02	0.284	
9	162.6	225.53	1445.22	13.72	29.02	0.168	
10	161.67	228.78	1355.63	13.66	29	0.288	
11	166.78	226.46	1427.4	13.69	29.03	0.28	
12	144.31	228.78	1453.44	13.45	29.09	0.288	
13	167.09	226.46	1453.44	13.64	29.03	0.212	
14	165.1	227.85	1432.05	13.64	29.09	0.296	
15	164.92	226.3	1367.57	13.66	29.28	0.252	

For each product, the process variables below were reported (Table 2), where "Quality Index" is the output variable representing the product's quality level:

Table 2. The list of the studied manufacturing process variables.

Parameter Name	Min	Max	Mean	STD
MOLD_TEMP	121.53	127.35	126.05	0.66
FILL_TIME	9.43	17.41	11.56	2.61
PLAST_TIME	4.31	10.25	5.01	0.53
CYCLE_TIME	115.91	117.47	116.59	0.67
CLOSING_FORCE	1358.89	1442.43	1398.06	17.20
CLAMP_FORCE_PEAK	1386.94	1467.08	1424.99	16.70
TORQUE_PEAK_VAL	146.01	201.97	180.91	7.79
TORQUE_MEAN_VAL	118.58	178.1	161.45	7.44
BACK_PRESS_PEAK_VAL	224.44	233.28	226.65	1.24
INJECT_PRESS_PEAK_VAL	1209.78	1461.65	1396.51	39.55
SCREW_HOLD_POS	12.91	14.04	13.65	0.15
SHOT_VOLUME	28.69	29.81	29.07	0.14
QUALITY_INDEX	0.1	0.75	0.39	0.15

The quality level of the parts can be explained as shown below (Table 3):

Table 3. Meaning the of quality index value.

QUALITY_INDEX		
From	**To**	**Meaning**
0	0.2	Good
0.21	0.35	Acceptable
0.36	0.45	Critical
0.46	-	Bad

More data exploration and visualization of some selected process variables can be seen in Figure 8, which shows the line chart and violin plot of the quality index, plasticization time, and mold temperature for the 1451 sets of data. The line-chart plot shows a sorted quality index, where the other plot shows the existence of bimodal distribution. This comes from the fact that our dataset contains two types of product families peculiar to this study. On the other hand, Figure 9 shows the plots of the same variables' distributions as histograms and raincloud plots.

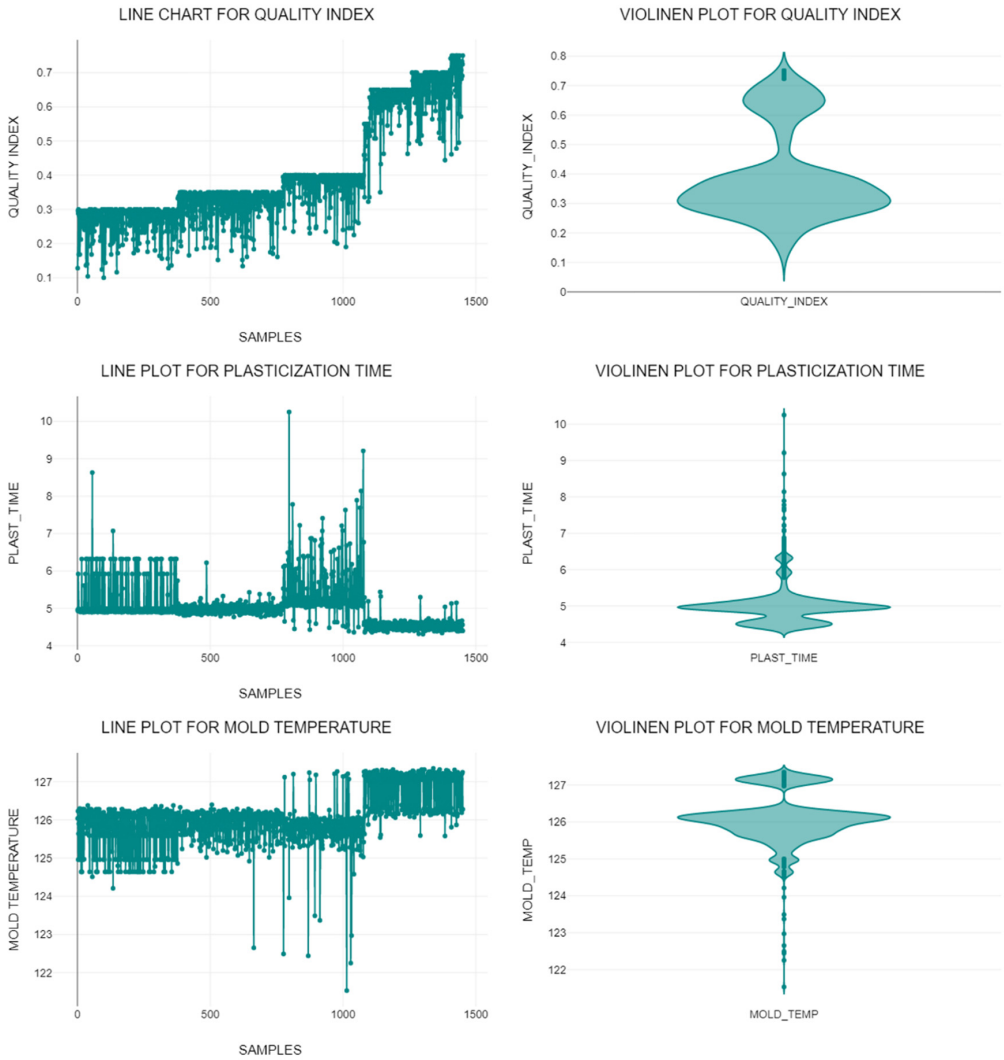

Figure 8. Line plot and violin chart for sample process variables.

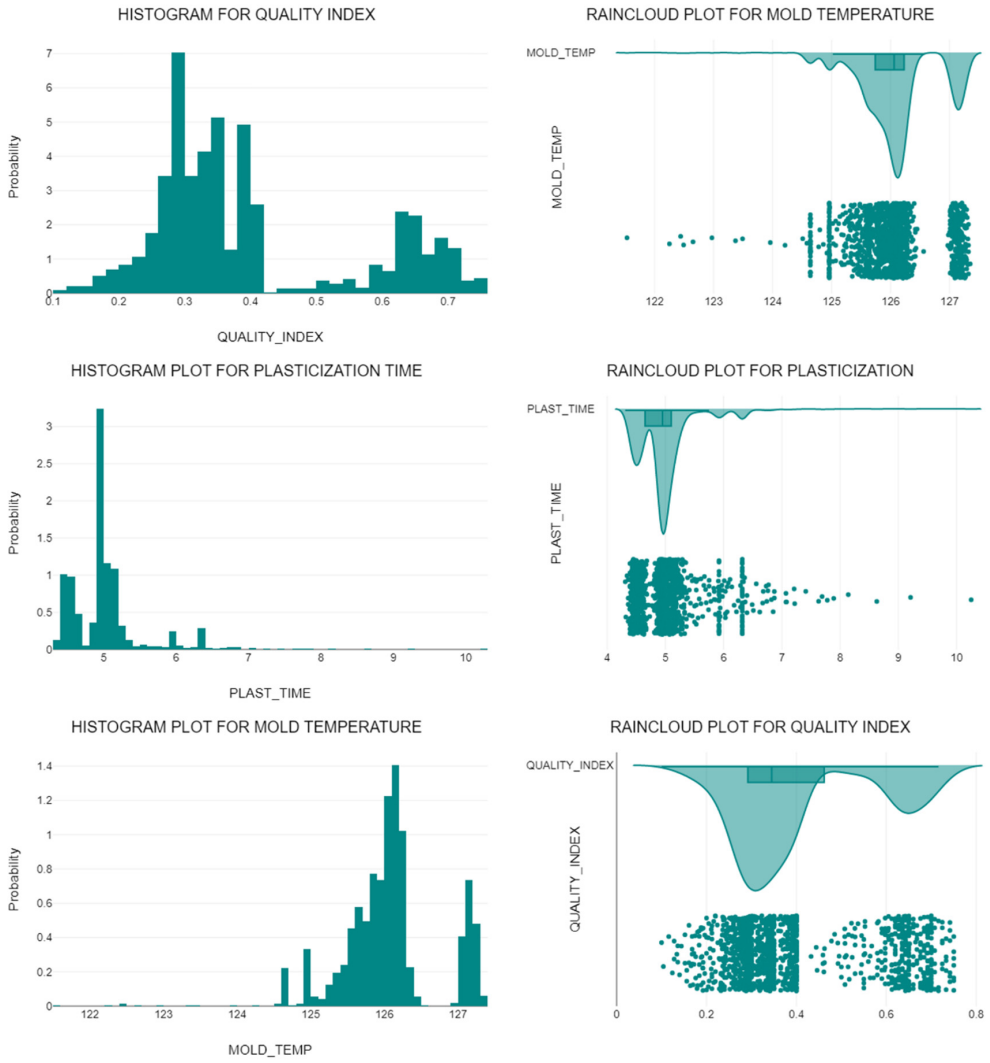

Figure 9. Histogram and raincloud chart for sample process variables.

4.2.3. Training a Supervised Learning Algorithm as Digital Shadow and DRL Environment

The previous steps of data collection, data cleaning, and data preprocessing are necessary steps within the dataflow from the physical system to its digital representation. Designing the digital version of the physical system called "Digital Shadow" within our proposed architecture implies the creation of a replication of the physical system output's change behavior based on the changes in the system input. This concept can be greatly implemented using a supervised machine learning model, where the input variables represent the manufacturing process parameters, and the output label or target represents the physical system output which is the produced item and their searched quality specification. Since we are defending the approach of a special-purpose digital twin, the goal is, thus, to try cloning the system behavior to generate and optimize a specific product quality (quality index). Following this logic, the chances of getting a high and accurate representation of the physical system can be improved by reaching a high machine learning model accuracy or a very low training and validation error.

We can describe three main targets for such a development:

- Real-time prediction of the quality index based on the actual manufacturing settings;
- Testing of hypothesis by manually changing the process values and observing the impact on the product quality;
- Playing the role of a virtual training environment for the deep reinforcement learning algorithm that is used in later steps to make process optimization through trial and error.

Our dataset was composed of 1451 products, which were produced on a manufacturing process that can be described by 12 variables. The measured value at the process exit is the product quality index, which represents the dependent variable for our machine learning model. In real-life manufacturing, the quality index used for this use case is the optical transmission characteristic of the resulting product. This index is measured in the quality department laboratory using some dedicated instruments. The value can range from "Bad" to "Good" following the measured value. Using a train–test–validate split of the data, several regression models were assessed and tuned in order to get the results shown in (Table 4), which are ordered by the mean absolute error value (MAE).

Table 4. Supervised learning models' metrics comparison.

Model Name	MAE	MSE	RMSE	R2	MAPE
Extra Trees Regressor	0.0306	0.0033	0.056	0.8547	0.096
Random Forest Regressor	0.0318	0.0038	0.0598	0.8357	0.0992
CatBoost Regressor	0.0312	0.0036	0.0585	0.8431	0.0995
Gradient Boosting Regressor	0.0353	0.004	0.0614	0.828	0.1123
Extreme Gradient Boosting	0.0357	0.0044	0.0644	0.8109	0.108
Light Gradient Boosting Machine	0.0364	0.0043	0.0639	0.8157	0.1094
Decision Tree Regressor	0.0437	0.0068	0.0806	0.7073	0.1284
K Neighbors Regressor	0.0451	0.0066	0.0806	0.7104	0.1409
AdaBoost Regressor	0.0529	0.0056	0.074	0.7594	0.1529
Linear Regression	0.0565	0.0066	0.0806	0.7129	0.1736

The selected model for this use case is based on the extra trees regression algorithm with an MAE of 0.0306.

Figure 10 demonstrates a preview of predicted values compared to real values from the validation set. The selected model very well fits the process data and even generalizes the validation data well, which is a simulation of unseen data. It is also visible in Figure 11 that some manufacturing parameters (e.g., cycle time) have a greater impact on the results compared to others.

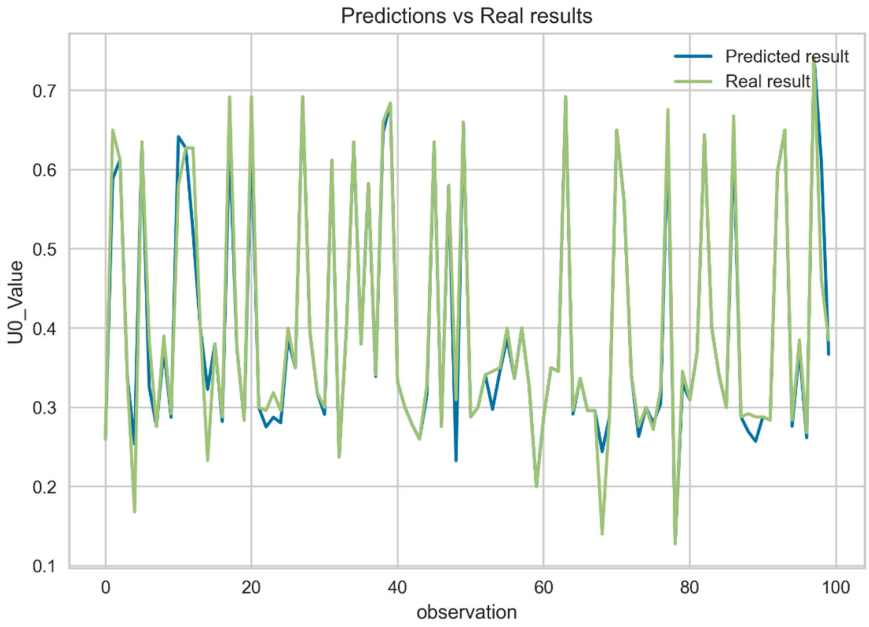

Figure 10. Plot of the real quality index values vs. the predicted values for 100 random products.

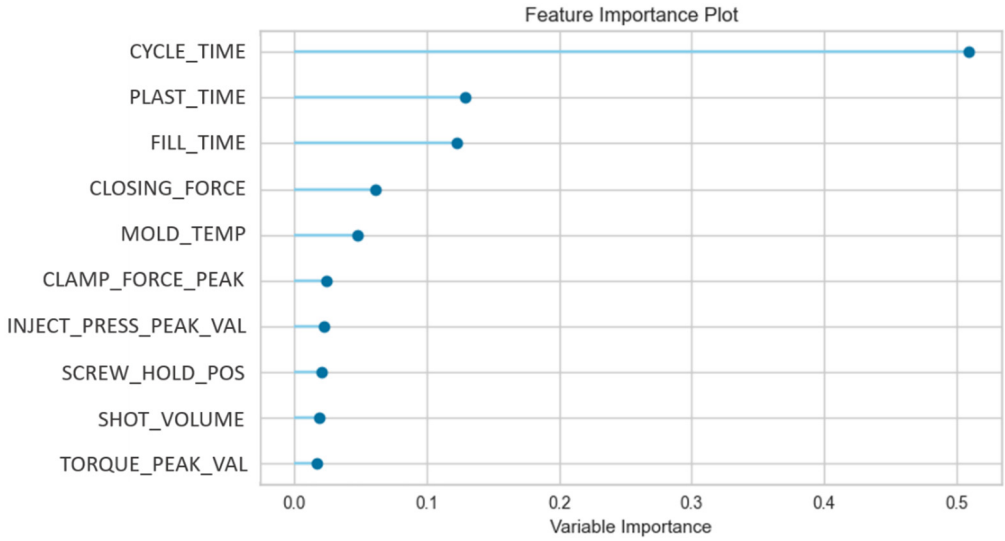

Figure 11. Plot of feature importance.

Our selection of tree-based regressors, including extra trees, random forest, CatBoost, gradient boosting, extreme gradient boosting, light gradient boosting machine, and decision tree regressor, was predicated on their strong interpretability, which is crucial for discerning feature relevance within the complex landscape of manufacturing data. These models excel in capturing the intricate, non-linear interactions typical of industrial datasets, and their ensemble nature—pooling multiple models to form a consensus—enhances the prediction accuracy while mitigating overfitting. Moreover, their inherent robustness to outliers and variable data distributions is invaluable in a manufacturing context where such anomalies are prevalent. We also acknowledged the necessity of benchmarking against simpler algorithms such as K neighbors regressor, AdaBoost regressor, and linear regression to provide a thorough evaluation of various machine learning strategies. This comprehensive approach confirms the superior performance of tree-based models in handling the multifaceted and sizable datasets characteristic of our domain.

In the next section, the supervised learning model is used as a training environment for a deep reinforcement learning algorithm. The latest interacts with the environment by taking actions which are mainly the changes in the manufacturing process values and then getting the predicted quality of the product. This resulting quality is then used to evaluate the quality of the model's actions by assigning a reward. The model states then get changed from the previous state to the actual states. The modeling of the DRL models is described in the next section.

4.3. Deep Reinforcement Learning for Automated Decision-Making

As mentioned earlier, the trained self-prediction model works as a simulated environment for the actual industrial use case (plastic injection molding). This allows for a more accurate testing. Because of this, the final process entails the creation of a DRL agent that interacts with the virtual environment rather than the physical environment. This DRL agent's mission is to discover the most effective strategy for improving the product quality index from a non-optimized process setting that comes from manufacturing random deviations. In this research, we investigated two modeling approaches of the RL environment in terms of the agent action space. The first approach tried to evaluate the optimization results using a discrete action space where the changes on each manufacturing parameter can be reduced as three known actions (increase, decrease, keep unchanged), while in the second approach, a continuous actions space was experimented. In the second approach, the changes in the process values are expressed as a sampling of real values from a change range. The structure of the investigated decision model is illustrated in Figure 12.

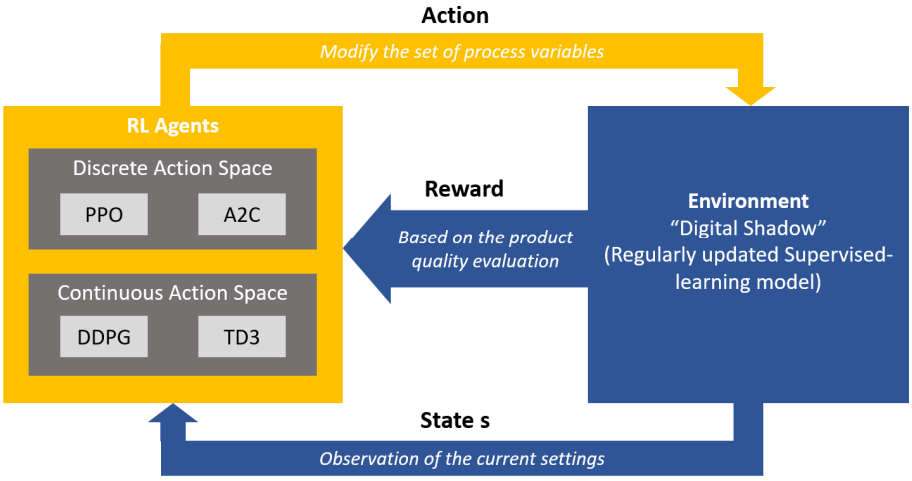

Figure 12. Proposed structure of RL decision model.

Modeling steps of our DRL models are described as below:

State Representation: This can be considered as the first step of modeling an RL problem. It consists of defining the state representation of the agent. In our case, we defined as states the values of each separate process variable in addition to the current product quality level.

The state vector is composed of twelve values:

$$St = (mt, ft, pt, ct, cf, cfp, tpv, tmv, bppv, ippv, shp, sv, qi\ ) \in S$$

where mt, ft, pt, ct, cf, cfp, tpv, tmv, $bppv$, $ippv$, shp, and sv represent the values of the process variables as described in Section 4.2.2, and "qi" is the equivalent product quality.

Action Space: The decision-making agent should be able to optimize the manufacturing process through various types of actions applied to different input process variables. We experimented with two types of action spaces (multi-discrete and continuous) in this study. In the discrete action space, the agent takes three possible actions, "increase", "decrease", or "keep unchanged", on each single process variable.

$$\boldsymbol{A} = \boldsymbol{a}_1 \times \boldsymbol{a}_2 \times ... \times \boldsymbol{a}_{12}$$

where $a_i = \{+2, -2, 0\}$ is the set of possible actions for the $i - th$ process variable.

On the other hand, the experienced continuous action space is a subset of the 12-dimensional interval $[-1, 1]$, which means that each action consists of 12 float values, each falling between -1 and 1, which we can denote as:

$$\boldsymbol{A} \subseteq [-1, 1]^{12}$$

Since the process values have different range magnitudes, state normalization was applied to avoid the situation of action saturation. This is also enhanced by the choice of an action space between -1 and 1. The states normalization is applied after getting the predicted quality index from the supervised learning model and just before supplying the new states to the deep neural network.

Reward Function: The reward function should be designed to optimize the quality index of the plastic injected part. From the RL perspective, this function is used to evaluate the agent's performance. For our problem, the agent should get a high reward for the cases where the proposed process parameters, once supplied to the supervised model, give a predicted quality index that is close to the targeted value. We selected the value of 0.03 as an accepted deviation from the target, which means the agent will get a negative reward in case the proposed set of values lead to a prediction that is outside the accepted deviation

from the target. We also tried to make the positive and the negative rewards proportional to the distance from the target values. By this, the reward function is given by:

$$r_e = \begin{cases} \frac{1}{0.5+\left|qi_{pred}-qi_{target}\right|} \; if \left|qi_{pred}-qi_{target}\right| \leq 0.03 \\ -\frac{1}{0.5+\left|qi_{pred}-qi_{target}\right|} \; Otherwise \end{cases}$$

where qi_{pred} represents the predicted quality index value, and qi_{target} is the target value.

We would like to emphasize that our primary objective is not the direct adaptation of existing deep reinforcement learning (DRL) algorithms to the plastic injection molding process. Rather, our goal is to conceptualize and demonstrate how this complex manufacturing process can be effectively modeled as a DRL problem. This conceptual framework serves as a cornerstone for understanding and exploring the potential of DRL in enhancing and optimizing various aspects of the injection molding process.

By representing the plastic injection molding process through the lens of DRL, we aim to illustrate the versatility and power of DRL algorithms in learning and adapting to complex, dynamic environments. This approach allows us to leverage the inherent capabilities of DRL to make informed decisions and improvements in the manufacturing process, potentially leading to significant advancements in efficiency, quality, and sustainability.

We believe that establishing this foundational understanding is crucial. It paves the way for future research to delve into more intricate adaptations and refinements of DRL algorithms, specifically tailored to the nuanced needs and challenges of plastic injection molding and similar sophisticated manufacturing processes. Our research seeks to ignite a conversation and spur further investigation into this promising interdisciplinary application of DRL, thereby contributing to the broader field of intelligent manufacturing systems.

In the next section, the results of the DRL models are presented and compared.

5. Research Results and Discussion

5.1. Evaluation Metrics for the Models' Performance

In this section, a detailed comparison for the performance of each model is discussed. According to our initial modeling approach, we intend to compare how models perform in a continuous action space and discrete action space. The result of this comparison will provide some insights into the suitable modeling methodology for any future similar projects. This includes comparing the DDPG and TD3 results with the PP0 and A2C results. In addition, for each modeling approach, we evaluate which model performs the best in terms of convergence speed, training loss, and agent reward.

In our research, we utilized the deep reinforcement learning (DRL) models provided by the Stable Baselines library, specifically focusing on proximal policy optimization (PPO), twin delayed DDPG (TD3), advantage actor–critic (A2C), and deep deterministic policy gradient (DDPG). For each model, we adhered to the default hyper parameters as a baseline for comparison and consistency. For the PPO, this included a learning rate of 0.00025, n_steps of 128, and a batch size of 64. TD3 was implemented with a learning rate of 0.001 and a batch size of 100. A2C utilized a learning rate of 0.0007 and n_steps of 5. Lastly, the DDPG was configured with a learning rate of 0.001 and a batch size of 100. These parameters were chosen to maintain a standard setting, allowing for a clear and unbiased evaluation of each model's performance under common conditions. By using the default settings, we ensure that our results are comparable with other studies and provide a solid baseline for further experimentation and refinement.

In Figure 13, we show the training evolution in terms of the episodes' mean reward over 190,000 steps for the four tested algorithms, while Figure 14 shows the episodes' mean length. The curves show different learning behaviors which can provide much information about the model's suitability for our problem.

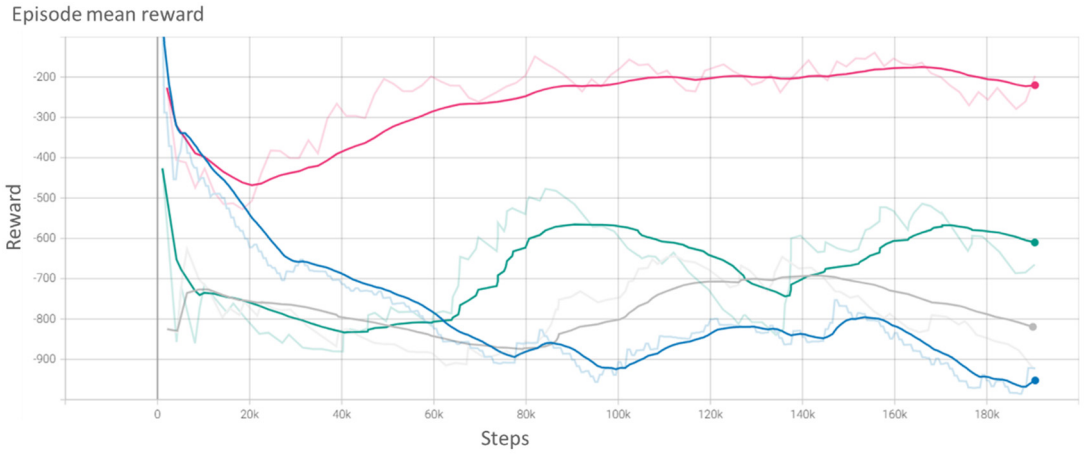

Figure 13. Episodes' mean reward comparison of PPO (pink), A2C (blue), DDPG (gray), and TD3 (green) algorithms.

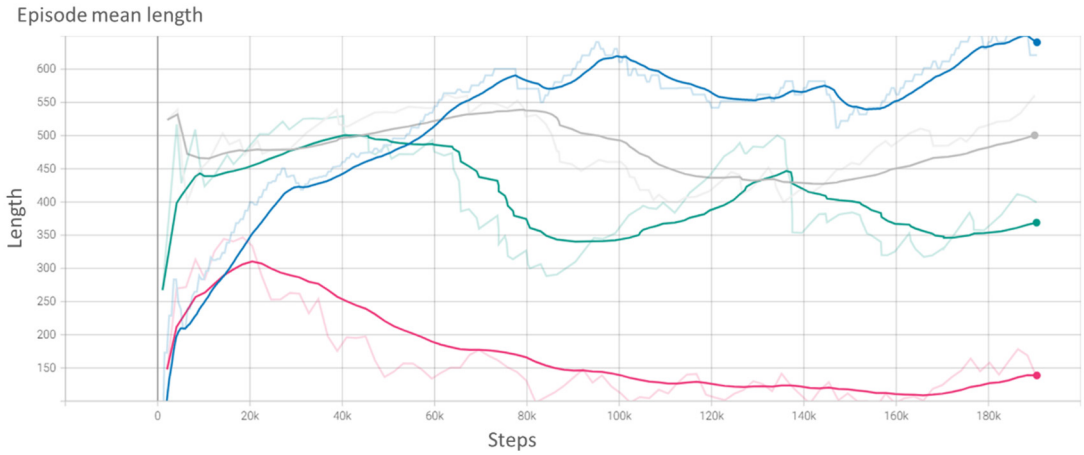

Figure 14. Episodes' mean length comparison of PPO (pink), A2C (blue), DDPG (gray), and TD3 (green) algorithms.

First, it is worth noting that the continuous action space used for TD3 and the DDPG is more complex than the discrete action space used for the PPO and A2C. The continuous action space being a subset of a 12-dimensional interval may be the reason why it is more difficult to optimize. In addition, the fact that the process values have different magnitudes gives more complexity to the problem. However, the normalization of the states that we apply before supplying them to the deep neural network may help achieve a better result.

The second factor that is, perhaps, creating different model behaviors is the reward function. We cannot be sure without a deeper analysis if the reward function we defined helps some of the models to converge faster than others, considering the nature of each of the algorithms. The reward function was made to encourage the agent to reach a target quality index with a small tolerance (0.03). This kind of reward function might favor the PPO, which has a rising trend, because it shows that the agent is always getting better at what it does. On the other hand, the fact that A2C is getting worse shows that it is struggling to meet the quality index goal. We also observe that TD3 and the DDPG are having a fluctuating trend, which indicates that they continue to learn, without converging, how to optimize the continuous action space. Trying to allow more training time may reveal the start of some convergence over time.

In general, when attempting to explain the reported differences in performance that can be seen between the four algorithms, there are several variables to take into consideration. It is highly likely that the combination of variables, such as action space difficulty,

reward function, supervised learning model selection, and hyper-parameters used for each algorithm, all play a role in the determination of the effectiveness of each compared to the others.

It is possible that the differences in the hyper-parameters that are used for each algorithm also contribute to the differences in performance that have been observed. To achieve the best possible outcomes, it is essential to guarantee that the hyper-parameters of each algorithm and environment are meticulously adjusted.

Finally, it is possible that the supervised learning model that was used to make the prediction of the quality score is another element that contributes to the variations in performance. It is conceivable that certain algorithms are better adapted to the predictive precision of the model than others, which could contribute to variations in performance.

Trying to interpret additional PPO model metrics gives more insights about the model robustness and learning consistency. In Figure 15, we observe different trends of Value Loss, Policy Gradient Loss, Training Loss, and Entropy Loss. The first metric measures the difference between the predicted value of the current state and the actual value obtained from the environment. We consider that getting a low value means that the state values get predicted with good accuracy by the model. Our curve goes down after step 160,000, which means that the model improves its ability to make good decisions. Allowing for more training steps can help us to understand how this ability will improve over time. Meanwhile, the Policy Gradient Loss measures the difference between the predicted actions and the actions that would have been taken according to the actual policy. Getting better rewards the relay on predicting the best actions, which generates a lower loss. Combining the previous two losses gives the Training Loss, which gives an overall overview of the model's training. Lastly, the Entropy Loss measures the randomness of the actions taken by the model. With a model exploring different actions (which is important for discovering optimal policies), the loss gets higher over time. We observe that the curve of the PPO model starts to flatten at the end of the training period, meaning that our model is doing less random actions and may be converging to an optimal policy.

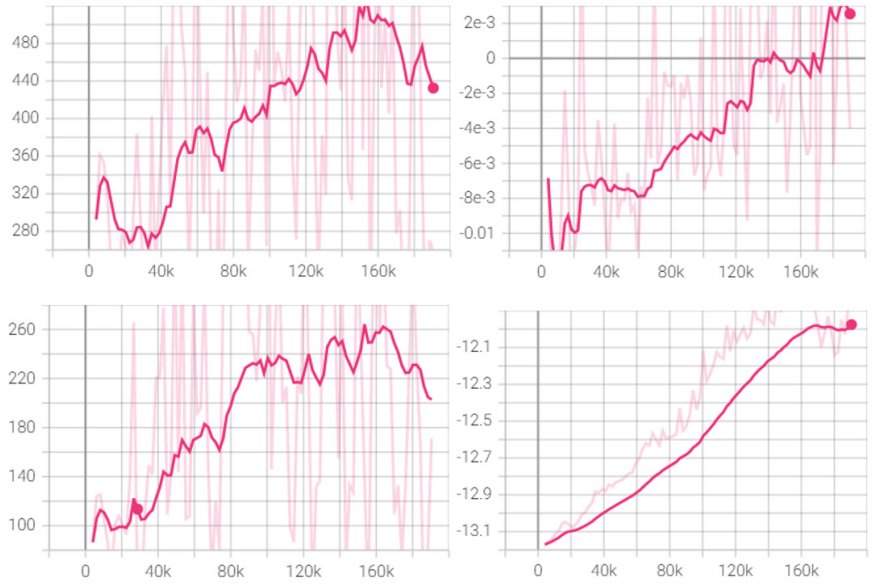

Figure 15. Other PPO metrics from top-left to bottom-right: Value Loss, Policy Gradient Loss, Training Loss, Entropy Loss.

The A2C seems to converge to an optimal policy very fast (before step 40,000) since the Value Loss and the Entropy Loss reach zero and stay steady during the remaining training time (Figure 16). However, this may indicate that A2C is the best model for our problem because the reward result is low for the selected policy compared to the PPO model. This means that the PPO still the best choice in this case.

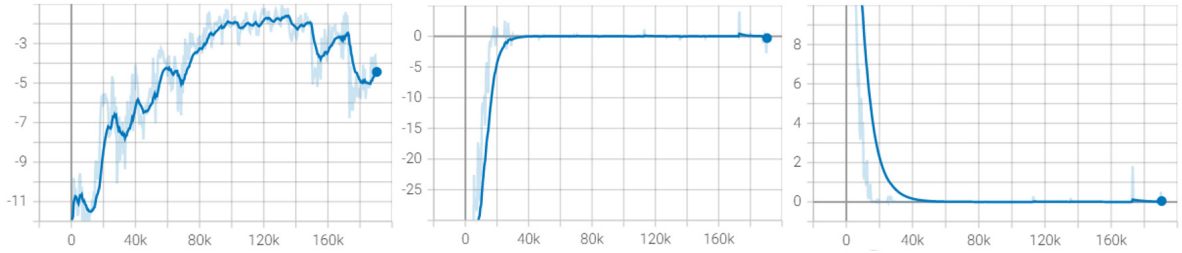

Figure 16. Other A2C metrics from left to right: Entropy Loss, Policy Loss, Value Loss.

The performance of the actor and critic networks of the TD3 algorithms can be discussed in terms of loss, which indicates how they are performing, respectively, in terms of policy function and value function. A low actor loss indicates that the policy function makes the best action in each state, while a low critic loss indicates that the expected return for a given state and action is predicted accurately. In Figure 17, we can see that both the actor loss and critic loss increase over time until the last steps, where they started to decrease. This means that the policy function does not improve as much as it should. However, the smooth curve indicates a steady progress toward a better policy. The same observation is made about the value function side, which can indicate some difficulties in convergence in the allowed training time. In conclusion, both curves continue to increase and start to flatten at the end of the training period, which means that the model perhaps approaches an optimal policy, which may be reached if more training steps are allowed.

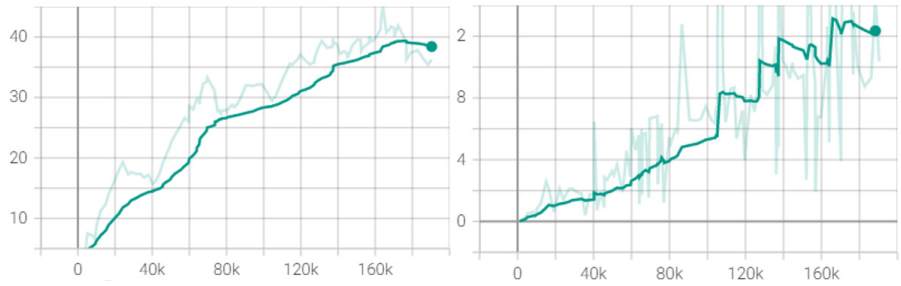

Figure 17. Other TD3 metrics; **Left**: Actor Loss; **Right**: Critic Loss.

For the DDPG model (Figure 18), the loss of the actor and critic networks gets fixed during the first 70,000 steps, meaning that there is no improvement during this learning phase. Then, the model starts to improve quickly and converge towards a better policy.

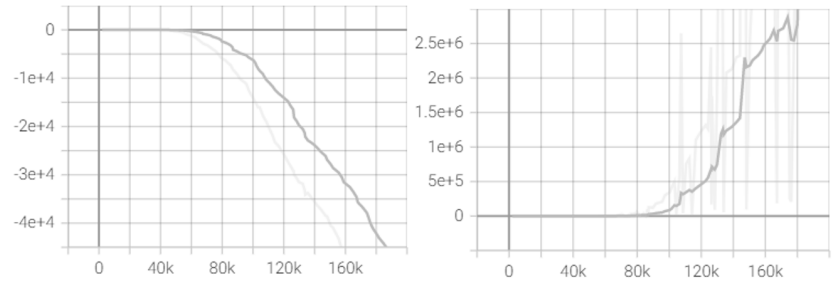

Figure 18. Other DDPG metrics; **Left**: Actor Loss; **Right**: Critic Loss.

In the next section, we describe how all the previous pieces of machine learning models should be tied together into a one specific-purpose digital twin in order to ensure a good manufacturing process performance.

5.2. Run-to-Convergence Comparison between PPO and TD3

While both algorithms performed well, the PPO seemed to have an edge over TD3 in terms of short training sessions. This raised the question of whether TD3 required a longer training to converge due to the complexity of continuous action spaces. To explore this further, we decided to conduct a second run of experiments with both algorithms. We increased the training time for both algorithms and added a stopping mechanism to ensure that the algorithms did not train endlessly. The results of the new experiments showed that TD3 outperformed the PPO in terms of mean episode reward, which confirmed our previous hypothesis. However, it is important to note that the difference in performance was not very significant. Furthermore, we found that TD3 required significantly more training time than the PPO to achieve its optimal performance. While the PPO only took eight hours of training, TD3 needed twenty hours to achieve greater result. This suggests that TD3 may be a more computationally expensive algorithm compared to the PPO. Nonetheless, the results showed that TD3 is a promising algorithm for continuous action spaces, and with sufficient training, it can outperform the PPO (Figure 19).

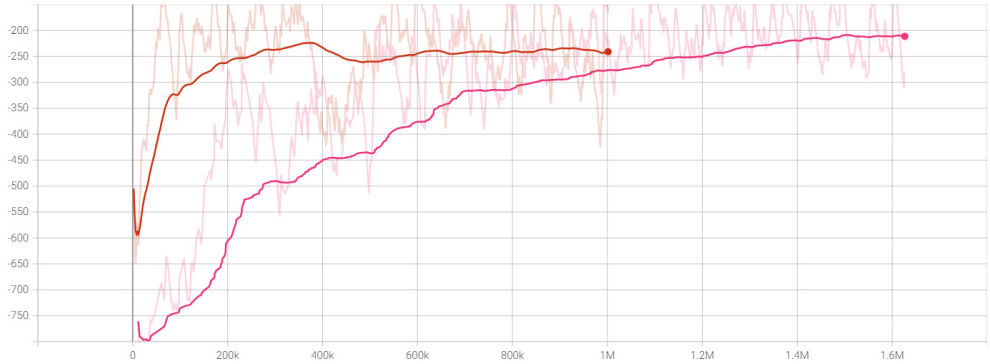

Figure 19. Run-to-convergence comparison between PPO (pink) and TD3 (red).

In the next section, we describe how all the previous pieces of machine learning models should be tied together into one specific-purpose digital twin in order to ensure a good manufacturing process performance.

5.3. Inference Architecture and Dataflow for the Specific-Purpose Digital Twin

We must connect all the machine learning models and incorporate them into an inference system in order to build the specific-purpose digital twin for the manufacturing process. This system's objectives include data collection, decision-making based on the data obtained, and real-time manufacturing process control.

The sensors and equipment on the manufacturing floor that produce the raw data are the first to enter the inference system's dataflow. The edge devices, which are in charge of pre-processing and filtering the data, then gather and process these data. The pre-processed data are subsequently transmitted from the edge devices to the cloud, where the decision-making engine further analyzes and makes use of them.

The system's smooth operation depends on the communication protocols that are used for data collection and transfer. The MQTT protocol is what we opted to employ because it handles enormous volumes of data effectively and simply. The edge devices operate as publishers, and the cloud acts as the subscriber, in a publish–subscribe pattern for data collection and transmission. The deep reinforcement learning algorithm serves as the foundation for the decision-making engine, which is in charge of selecting choices based on the pre-processed data. The algorithm takes the environment's status information as input and produces an action that will improve the manufacturing process. The algorithm continuously develops its decision-making skill over time by using historical data from the manufacturing process as training data.

A digital shadow Is used to replicate the manufacturing process in order to ensure that the judgments made by the algorithm are accurate and effective. Before being used in the actual manufacturing process, the algorithm's judgments can be verified and assessed in a virtual environment because of the digital shadow.

Overall, the specific-purpose digital twin for the manufacturing process is a complicated system that necessitates the fusion of many machine learning models and communication protocols. The decision-making engine depends on the dataflow from the sensors and machines to operate, and the usage of a digital shadow guarantees the accuracy and efficacy of the algorithm's conclusions.

5.4. Bridging Research Gaps with an Autonomous AI-Enabled Digital Twin Framework

In the introduction section, we highlighted three main limitations in the current research on the digital twin technology for manufacturing processes. Firstly, many approaches are proposed without a clear emphasis on the industrialization phase or adaptability to different types of manufacturing processes. Secondly, some practical methodologies for digital twins are too rigid and difficult to apply in different contexts. Thirdly, there is a lack of research on the decision-making aspect of digital twins, particularly in integrating modern artificial intelligence capabilities.

To address these limitations, we proposed a framework for implementing an autonomous AI-enabled digital twin for any type of manufacturing process. This framework utilizes machine learning techniques such as supervised learning, deep reinforcement learning, and optimization to create a data-driven approach to modeling and decision-making. In the discussion section, we evaluate the results of our framework and its implications for future research.

We highlight the importance of developing adaptable and scalable digital twin systems that can be applied to different manufacturing processes and industries. We also discuss the potential for integrating other advanced technologies such as the Internet of Things (IoT) to enhance the capabilities of digital twins. Overall, our framework and its implementation in the plastic injection molding use case demonstrate the potential for the digital twin technology to revolutionize manufacturing processes and decision-making support.

6. Conclusions

This paper introduces a novel methodology for designing an AI-based, specific-purpose digital twin to enhance manufacturing processes. Shifting from traditional models that rely on physical laws, our approach leverages real-time data and machine learning for a more dynamic and accurate representation of complex systems. We emphasize the importance of process stability and product quality, underlining the critical role of precise machine settings and parameters. Our concept of a specific-purpose digital twin addresses the challenges of a broader adoption by targeting specific operational objectives, thereby improving feasibility and value. We detail effective data collection and pre-processing methods, ensuring traceability and accurate mapping between process inputs and quality outputs. Our data-driven replica of the physical model captures real-time changes and continuously updates for relevance and accuracy. By integrating supervised learning algorithms, the digital twin continuously learns and adapts, optimizing manufacturing parameters in real time to maintain a high product quality. Overall, our contributions provide a strategic framework for deploying digital twins in manufacturing, promising significant improvements in quality, efficiency, and adaptability.

We provide a thorough analysis of the use of deep reinforcement learning algorithms in manufacturing processes in this work. For a particular manufacturing issue, we tested four distinct DRL algorithms: PPO, A2C, TD3, and DDPG. We also offer interpretations of the metrics and the effectiveness of these models.

The PPO algorithm performs the best in terms of generating the highest cumulative reward, according to our results. Although the A2C algorithm's final reward is lower than the PPO algorithm's, it exhibits a faster convergence to an optimal policy. Although they

could need more training steps to completely converge, the TD3 and DDPG algorithms also showed consistent progress in the direction of a superior policy. We also discuss how these DRL models, together with the dataflow, communication protocols, and decision-making engine, can be integrated into a dedicated digital twin for manufacturing operations. Utilizing digital twins can lower operating expenses and enhance the performance of manufacturing processes. Even though our research offers insightful information about the application of DRL to manufacturing processes, there are still some things that could be done better. One drawback of our research is that we only tested a small selection of DRL algorithms; other algorithms might be able to perform even better on this particular manufacturing problem. The models might not generalize well to other manufacturing problems or environments because we considered a fixed environment. Future studies might concentrate on investigating additional DRL algorithms and analyzing how well they work when applied to various manufacturing issues. Investigating ways to strengthen these models' extension and increase their capacity for environmental adaptation would also be intriguing. Last but not least, the incorporation of other technologies like computer vision and IoT sensors may offer extra information and insights for production process optimizations. The findings of our research show, in summary, how DRL algorithms can improve the efficiency of manufacturing processes. Manufacturers can streamline their operations and lower operating costs by integrating these models into a specific-purpose digital twin, which can boost revenue and competitiveness.

Author Contributions: Conceptualization, A.K., T.M., I.E.H. and C.E.M.; Methodology, A.K., T.M. and C.E.M.; Software, A.K.; Validation, T.M. and I.E.H.; Formal analysis, A.K.; Investigation, A.K. and I.E.H.; Writing—original draft, A.K. and C.E.M.; Writing—review & editing, T.M.; Supervision, T.M. and I.E.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Link to supporting data and code can be found at the Gihub repo: https://github.com/KHDOUDI/DT, accessed on 6 December 2023.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cheng, K. Digital-Twins-Driven Semi-Physical Simulation for Testing and Evaluation of Industrial Software in a Smart Manufacturing System. *Machines* **2022**, *10*, 388. [CrossRef]
2. Leng, J.; Wang, D.; Shen, W.; Li, X.; Liu, Q.; Chen, X. Digital twins-based smart manufacturing system design in Industry 4.0: A review. *J. Manuf. Syst.* **2021**, *60*, 119–137. [CrossRef]
3. Abramovici, M.; Göbel, J.C.; Savarino, P. Virtual twins as integrative components of smart products. In *Product Lifecycle Management for Digital Transformation of Industries: 13th IFIP WG 5.1 International Conference, PLM 2016, Columbia, SC, USA, 11–13 July 2016, Revised Selected Papers*; Springer International Publishing: Berlin/Heidelberg, Germany, 2016; Volume 13, pp. 217–226.
4. Rosen, R.; von Wichert, G.; Lo, G.; Bettenhausen, K.D. About the importance of autonomy and digital twins for the future of manufacturing. *IFAC-Paper* **2015**, *48*, 567–572. [CrossRef]
5. Zhang, Q.; Liu, Z.; Duan, J.; Qin, J. A Novel Method of Digital Twin-Based Manufacturing Process State Modeling and Incremental Anomaly Detection. *Machines* **2023**, *11*, 151. [CrossRef]
6. Lee, D.; Kim, C.-K.; Yang, J.; Cho, K.-Y.; Choi, J.; Noh, S.-D.; Nam, S. Digital Twin-Based Analysis and Optimization for Design and Planning of Production Lines. *Machines* **2022**, *10*, 1147. [CrossRef]
7. Xiao, W.; He, M.; Wei, Z.; Wang, N. SWLC-DT: An Architecture for Ship Whole Life Cycle Digital Twin Based on Vertical–Horizontal Design. *Machines* **2022**, *10*, 998. [CrossRef]
8. Tang, Y.-M.; Ho, G.T.S.; Lau, Y.-Y.; Tsui, S.-Y. Integrated Smart Warehouse and Manufacturing Management with Demand Forecasting in Small-Scale Cyclical Industries. *Machines* **2022**, *10*, 472. [CrossRef]
9. Caccamo, C.; Pedrazzoli, P.; Eleftheriadis, R.; Magnanini, M.C. Using the Process Digital Twin as a tool for companies to evaluate the Return on Investment of manufacturing automation. *Procedia CIRP* **2022**, *107*, 724–728. [CrossRef]
10. Stavropoulos, P.; Papacharalampopoulos, A.; Michail, C.K. Digital twin-driven multi-variable process control of thermal manufacturing processes. *Procedia CIRP* **2022**, *107*, 752–757. [CrossRef]
11. Friederich, J.; Francis, D.P.; Lazarova-Molnar, S.; Mohamed, N. A framework for data-driven digital twins of smart manufacturing systems. *Comput. Ind.* **2022**, *136*, 103586. [CrossRef]
12. Lugaresi, G.; Matta, A. Automated manufacturing system discovery and digital twin generation. *J. Manuf. Syst.* **2021**, *59*, 51–66. [CrossRef]

13. Bazaz, S.M.; Lohtander, M.; Varis, J. Availability of Manufacturing data resources in Digital Twin. *Procedia Manuf.* **2020**, *51*, 1125–1131. [CrossRef]
14. Papacharalampopoulos, A.; Michail, C.K.; Stavropoulos, P. Manufacturing resilience and agility through processes digital twin: Design and testing applied in the LPBF case. *Procedia CIRP* **2021**, *103*, 164–169. [CrossRef]
15. Zhang, L.; Yan, Y.; Hu, Y.; Ren, W. Reinforcement learning and digital twin-based real-time scheduling method in intelligent manufacturing systems. *IFAC-Paper* **2022**, *55*, 359–364. [CrossRef]
16. Latsou, C.; Farsi, M.; Erkoyuncu, J.A. Digital twin-enabled automated anomaly detection and bottleneck identification in complex manufacturing systems using a multi-agent approach. *J. Manuf. Syst.* **2023**, *67*, 242–264. [CrossRef]
17. Gopal, L.; Singh, H.; Mounica, P.; Mohankumar, N.; Challa, N.P.; Jayaraman, P. Digital twin and IOT technology for secure manufacturing systems. *Meas. Sens.* **2023**, *25*, 100661. [CrossRef]
18. Qian, W.; Guo, Y.; Zhang, H.; Huang, S.; Zhang, L.; Zhou, H.; Fang, W.; Zha, S. Digital twin driven production progress prediction for discrete manufacturing workshop. *Robot. Comput. Integr. Manuf.* **2023**, *80*, 102456. [CrossRef]
19. Balakrishnan, P.; Babu, K.R.; Naiju, C.D.; Madiajagan, M. Design and implementation of digital twin for predicting failures in automobiles using machine learning algorithms. In *SAE Technical Paper*; SAE International: Warrendale, PA, USA, 2019.
20. Qianzhe, Q.; Wang, J.; Ye, L.; Gao, R. Digital Twin for Machining Tool Condition Prediction. *Procedia CIRP* **2019**, *81*, 1388–1393.
21. Zhu, Z.; Liu, C.; Xu, X. Visualisation of the digital twin data in manufacturing by using augmented reality. *Procedia CIRP* **2019**, *81*, 898–903. [CrossRef]
22. Liu, J.; Du, X.; Zhou, H.; Liu, X.; Li, L.; Feng, F. A digital twin-based approach for dynamic clamping and positioning of the flexible tooling system. *Procedia CIRP* **2019**, *80*, 746–749. [CrossRef]
23. Yao, Y.; Liu, M.; Du, J.; Zhou, L. Design of a machine tool control system for function reconfiguration and reuse in network environment. *Robot. Comput. Integr. Manuf.* **2019**, *56*, 117–126. [CrossRef]
24. Qiu, S.; Liu, S.; Kong, D.; He, Q. Three-dimensional virtual-real mapping of aircraft automatic spray operation and online simulation monitoring. *Virtual Real. Intell. Hardw.* **2019**, *1*, 611–621. [CrossRef]
25. Liau, Y.Y.; Lee, H.; Ryu, K. Digital Twin concept for smart injection molding. *IOP Conf. Ser. Mater. Sci. Eng.* **2018**, *324*, 012077. [CrossRef]
26. Modoni, G.E.; Stampone, B.; Trotta, G. Application of the Digital Twin for in process monitoring of the micro injection moulding process quality. *Comput. Ind.* **2022**, *135*, 103568. [CrossRef]
27. Wang, Z.; Feng, W.; Ye, J.; Yang, J.; Liu, C. A Study on Intelligent Manufacturing Industrial Internet for Injection Molding Industry Based on Digital Twin. *Complexity* **2021**, *2021*, 8838914. [CrossRef]
28. Da Cunha, C.; Cardin, O.; Gallot, G.; Viaud, J. Designing the Digital Twins of Reconfigurable Manufacturing Systems: Application on a smart factory. *IFAC-Paper* **2021**, *54*, 874–879. [CrossRef]
29. Krückemeier, S.; Anderl, R. Concept for Digital Twin Based Virtual Part Inspection for Additive Manufacturing. *Procedia CIRP* **2022**, *107*, 458–462. [CrossRef]
30. Dvorak, J.; Cornelius, A.; Corson, G.; Zameroski, R.; Jacobs, L.; Penney, J.; Schmitz, T. A machining digital twin for hybrid manufacturing. *Manuf. Lett.* **2022**, *33*, 786–793. [CrossRef]
31. Langlotz, P.; Klar, M.; Yi, L.; Hussong, M.; Sousa, F.J.P.; Aurich, J.C. Concept of hybrid modeled digital twins and its application for an energy management of manufacturing systems. *Procedia CIRP* **2022**, *112*, 549–554. [CrossRef]
32. Huang, Z.; Fey, M.; Liu, C.; Beysel, E.; Xu, X.; Brecher, C. Hybrid learning-based digital twin for manufacturing process: Modeling framework and implementation. *Robot. Comput. Integr. Manuf.* **2023**, *82*, 102545. [CrossRef]
33. Fan, Y.; Yang, J.; Chen, J.; Hu, P.; Wang, X.; Xu, J.; Zhou, B. A digital-twin visualized architecture for Flexible Manufacturing System. *J. Manuf. Syst.* **2021**, *60*, 176–201. [CrossRef]
34. García, Á.; Bregon, A.; Martínez-Prieto, M.A. Towards a connected Digital Twin Learning Ecosystem in manufacturing: Enablers and challenges. *Comput. Ind. Eng.* **2022**, *171*, 108463. [CrossRef]
35. Li, S.E. Deep reinforcement learning. In *Reinforcement Learning for Sequential Decision and Optimal Control*; Springer Nature: Singapore, 2023; pp. 365–402.
36. Puterman, M.L. *Markov Decision Processes—Discrete Stochastic Dynamic Programming*; John Wiley & Sons, Inc.: New York, NY, USA, 1994.
37. Szepesvari, C.; Sutton, R.S.; Modayil, J.; Bhatnagar, S.; Models, U.O. In Proceedings of the 28th Annual Conference Processing Systems, Montreal, QC, Canada, 8–13 December 2014; pp. 990–998. Available online: https://search.worldcat.org/zh-cn/title/advances-in-neural-information-processing-systems-27-28th-annual-conference-on-neural-information-processing-systems-2014-nips-december-8-13-2014-montreal-canada-proceedings-of-the-2014-conference-vol-2/oclc/931952337 (accessed on 6 December 2023).
38. Dayan, P. Temporal differences: TD(λ) for general λ. *Mach. Learn.* **1992**, *8*, 341–362. [CrossRef]
39. Watkins, C.J.C.H.; Dayan, P. Q-learning. In *Machine Learning*; Springer: Berlin/Heidelberg, Germany, 1989; Volume 8, pp. 279–292.
40. Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A.A.; Veness, J.; Bellemare, M.G.; Graves, A.; Riedmiller, M.; Fidjeland, A.K.; Ostrovski, G.; et al. Human-level control through deep reinforcement learning. *Nature* **2015**, *518*, 529–533. [CrossRef] [PubMed]
41. Peters, J.; Schaal, S. Reinforcement learning of motor skills with policy gradients. *Neural Netw.* **2008**, *21*, 682–697. [CrossRef]
42. Sutton, R.S.; McAllester, D.; Singh, S.; Mansour, Y. Policy gradient methods for reinforcement learning with function approximation. *Adv. Neural Inf. Process. Syst.* **1999**, *12*, 1058–1063.

43. Silver, D.; Lever, G.; Heess, N.; Degris, T.; Wierstra, D.; Riedmiller, M. Deterministic policy gradient algorithms. In *International Conference on Machine Learning*; PMLR: Beijing, China, 2014; pp. 387–395.
44. Dong, H.; Ding, Z.; Zhang, S. *Deep Reinforcement Learning*; Springer: Singapore, 2020.
45. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In *International Conference on Machine Learning*; PMLR: Lille, France, 2015; pp. 1889–1897.
46. Kurutach, T.; Clavera, I.; Duan, Y.; Tamar, A.; Abbeel, P. Model-ensemble trust-region policy optimization. *arXiv* **2018**, arXiv:1802.10592.
47. Wang, Y.; He, H.; Tan, X.; Gan, Y. Trust region-guided proximal policy optimization. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 32.
48. Konda, V.; Tsitsiklis, J. Actor-critic algorithms. *Adv. Neural Inf. Process. Syst.* **1999**, *12*, 7.
49. Peters, J.; Schaal, S. Natural actor-critic. *Neurocomputing* **2008**, *71*, 1180–1190. [CrossRef]
50. Bhatnagar, S.; Sutton, R.S.; Ghavamzadeh, M.; Lee, M. Natural actor–critic algorithms. *Automatica* **2009**, *45*, 2471–2482. [CrossRef]
51. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.P.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*; PMLR: New York City, NY, USA, 2016; pp. 1928–1937.
52. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous control with deep reinforcement learning. *arXiv* **2015**, arXiv:1509.02971.
53. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*; PMLR: Vienna, Austria, 2018; pp. 1587–1596.
54. Azamfirei, V.; Psarommatis, F.; Lagrosen, Y. Application of automation for in-line quality inspection, a zero-defect manufacturing approach. *J. Manuf. Syst.* **2023**, *67*, 1–22. [CrossRef]