processes

MDPI

*Article*

# Optimization of Smart Textiles Robotic Arm Path Planning: A Model-Free Deep Reinforcement Learning Approach with Inverse Kinematics

**Di Zhao [1,2,3], Zhenyu Ding [3], Wenjie Li [1], Sen Zhao [1] and Yuhong Du [4,*]**

[1] School of Mechanical Engineering, Tiangong University, Tianjin 300387, China; zhaodi@tiangong.edu.cn (D.Z.); 18222828779@163.com (W.L.); zs2014062300@163.com (S.Z.)
[2] Engineering Teaching Practice Training Center, Tiangong University, Tianjin 300387, China
[3] School of Electronics & Information Engineering, Tiangong University, Tianjin 300387, China; 2010910208@tiangong.edu.cn
[4] Innovation College, Tiangong University, Tianjin 300387, China
* Correspondence: duyuhong@tiangong.edu.cn

**Abstract:** In the era of Industry 4.0, optimizing the trajectory of intelligent textile robotic arms within cluttered configuration spaces for enhanced operational safety and efficiency has emerged as a pivotal area of research. Traditional path-planning methodologies predominantly employ inverse kinematics. However, the inherent non-uniqueness of these solutions often leads to varied motion patterns in identical settings, potentially leading to convergence issues and hazardous collisions. A further complication arises from an overemphasis on the tool center point, which can cause algorithms to settle into suboptimal solutions. To address these intricacies, our study introduces an innovative path-planning optimization strategy utilizing a model-free, deep reinforcement learning framework guided by inverse kinematics experience. We developed a deep reinforcement learning algorithm for path planning, amalgamating environmental enhancement strategies with multi-information entropy-based geometric optimization. This approach specifically targets the challenges outlined. Extensive experimental analyses affirm the enhanced optimality and robustness of our method in robotic arm path planning, especially when integrated with inverse kinematics, outperforming existing algorithms in terms of safety. This advancement notably elevates the operational efficiency and safety of intelligent textile robotic arms, offering a groundbreaking and pragmatic solution for path planning in real-world intelligent knitting applications.

**Keywords:** machine learning; deep reinforcement learning; inverse kinematics; path planning; robotic arm



**Citation:** Zhao, D.; Ding, Z.; Li, W.; Zhao, S.; Du, Y. Optimization of Smart Textiles Robotic Arm Path Planning: A Model-Free Deep Reinforcement Learning Approach with Inverse Kinematics. *Processes* **2024**, *12*, 156. https://doi.org/10.3390/pr12010156

Academic Editors: Xiaosong Du and Devina P. Sanjaya

Received: 8 November 2023
Revised: 2 January 2024
Accepted: 5 January 2024
Published: 9 January 2024



## 1. Introduction

The deepening evolution of the Industry 4.0 era is progressively reshaping the face of industrial production, with a pronounced impact in areas like building control [1] and the textile industry [2]. Textile processes, in particular, stand at the forefront of this change. In the realm of intelligent computerized flat knitting, robotic arms bear about 30% of the operational load, undertaking tasks such as yarn fetching, feeding, and knitting. This shift necessitates innovative approaches to robotic arm path planning [3]. The complexity inherent in these tasks not only raises the difficulty of planning and execution but also increases the likelihood of errors, thereby adversely affecting both efficiency and overall production [3]. Traditional path-planning methodologies, which are heavily dependent on offline programming and online instruction, prove inadequate for complex knitting operations, demanding significant labor and time investments [4–6]. Activities like yarn fetching and feeding require multiple adjustments in arm posture, each mandating a reevaluation and replanning of the path. Consequently, the pursuit of efficient, accurate,

and safe path planning for knitting robotic arms has emerged as a vital area of research within the sphere of automated and intelligent manufacturing.

In industrial settings, the trajectory planning of robotic arms typically employs forward and inverse kinematic techniques [7]. Forward kinematics involves manipulating the end effector, TCP, to execute specific tasks by modulating joint angles, velocities, and accelerations. In contrast, inverse kinematics centers on the TCP's path planning, translating this planned trajectory from the path space to the robotic arm joints' operational space through inverse kinematic processes [8]. This paper focuses on exploring model-free path-planning strategies for knitting robotic arms, underpinned by experiential knowledge in inverse kinematics.

DRL distinguishes itself from traditional reinforcement learning methods [9] by adeptly navigating high-dimensional, continuous state and action spaces, thereby exhibiting enhanced representational and generalization skills. This capability enables DRL to identify latent patterns and structures in intricate tasks, facilitating the learning of highly non-linear and complexly associated decisions in unfamiliar environments, as demonstrated in the soft robotics sector [10]. DRL primarily operates through two paradigms: value-function-based and policy-gradient-based reinforcement learning algorithms [11]. The former concentrates on learning the value functions of states or state–action pairs, and evaluating action or state values under prevailing policies. The inherent continuity of action spaces and the infinite variety of action values pose substantial challenges in handling continuous domain issues, with Q-Learning and Deep Q-Network (DQN) being notable examples. Conversely, policy-gradient-based methods, aimed at directly learning policies, are more apt for continuous domains. They produce a probability distribution for each potential action in a given state, refining the policy through parameter optimization to enhance the probabilities of actions yielding higher rewards. Representative algorithms include TRPO [12], DDPG [13], A2C [14], HER [15], PPO [16], SAC [17], and TD3 [18].

These algorithms have all shown remarkable efficacy in path planning for target points. Nevertheless, the non-uniqueness and multiplicity inherent in inverse kinematics can lead to various solutions in the robotic arm joints' operational space [19], creating "multiple solution conflicts". This issue may result in the same processing method being applied to disparate joint states, leading to impractical motion outcomes and compromised stability. Moreover, an excessive emphasis on the TCP point could trap algorithms in local optima, potentially causing unsafe movements, increased mechanical wear, and higher energy consumption. This scenario underscores a lack of robustness and an inability to realize optimal global outcomes [20].

In tackling the challenges posed by the multiplicity of inverse kinematics solutions and the tendency towards local optima due to an excessive focus on the TCP point, this study innovatively introduces Environment Augmentation (EA) and Multi-Information Entropy Geometric Optimization (MIEGO) methodologies within the realm of DRL algorithms. The EA strategy, in particular, enriches an environmental state by incorporating the angles of each robotic arm joint and treats the stability of solutions as an additional reward factor. This technique effectively guides agents towards exploring paths with enhanced stability.

To circumvent the "local optima trap" associated with an overemphasis on the TCP point, we propose the MIEGO approach. This method synergizes the TCP optimization issue with broader global optimization goals, creating a comprehensive multi-objective optimization framework. By shifting the focus from a narrow TCP-centric view to a wider multi-objective perspective, the likelihood of falling into local optima is substantially reduced. In optimizing these multi-objective elements, the study leverages information geometric techniques to regulate multi-information entropy, thereby not only advancing the efficacy of deep reinforcement learning (DRL) algorithms but also amplifying their convergence properties.

Subsequently, the paper delineates the principal work, offering an overview of existing policy-gradient-based reinforcement learning algorithms, followed by an exhaustive delineation of both the EA strategy and MIEGO method. Empirical assessments are conducted to

thoroughly validate each algorithm. Moreover, the simultaneous implementation of these enhancements in the DDPG and SAC algorithms reveals that their integration effectively harmonizes aspects like path length, motion safety, and equipment longevity.

## 2. Main Work

In this study, we developed and implemented innovative EA strategies and MIEGO methods within the framework of DRL algorithms. The primary contributions are outlined as follows:

- The design and analysis of specific industrial scenarios with existing DRL methods: We targeted a specialized application scenario involving intelligent textile robotic arms in textile production lines (depicted in Figure 1), establishing a virtual twin system for simulation training to gather insightful learning data. Furthermore, an extensive examination of current policy-gradient-based reinforcement learning algorithms [21], including TRPO, DDPG, PPO, SAC, and TD3, was undertaken, offering a detailed analysis of their distinct features.
- The introduction of novel strategies to enhance DRL and their experimental validation: To address the challenges of "multiple solution conflicts" in inverse kinematics and the tendency towards local optima due to a concentrated focus on the TCP point, this study introduces novel EA strategies and MIEGO methods tailored for DRL algorithms. Empirical studies were conducted using the virtual twin model of the intelligent textile robotic arm scenario within VREP 3.6.2 software. Our study contrasted these advancements with classical DRL algorithms. The findings indicate a substantial enhancement in the performance of DRL algorithms across several crucial metrics, attributable to the integration of the methodologies proposed herein.

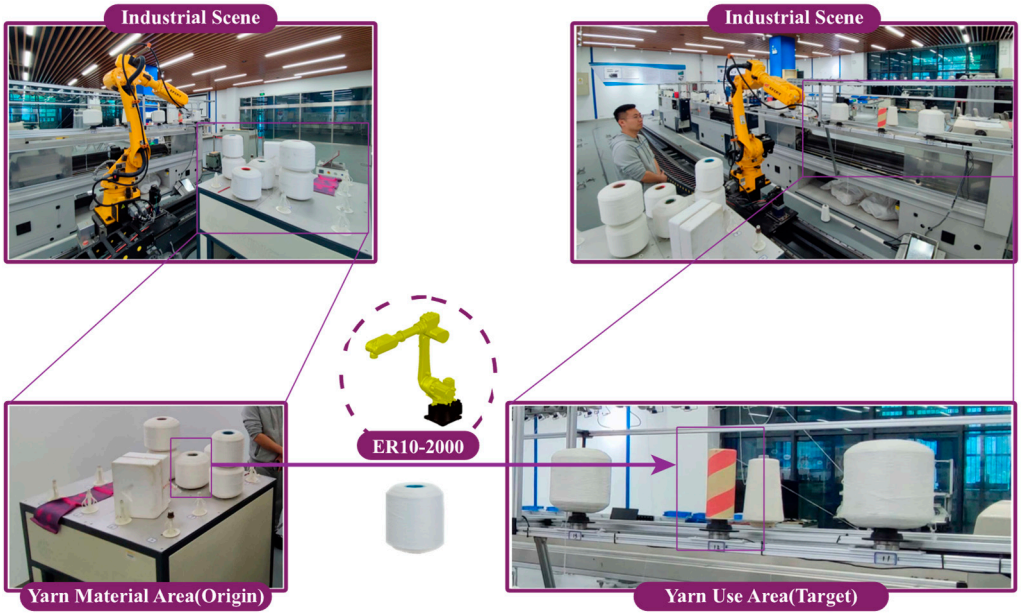



**Figure 1.** The present state of textile production lines.

## 3. Reinforcement Learning Algorithms

This section delves into existing policy-gradient-based reinforcement learning algorithms, which were selected as foundational algorithms for the textile robotic arms discussed in this study. These algorithms are explored and experimented upon in conjunction with the strategies introduced in this research.

### 3.1. Trust Region Methods

Trust Region Policy Optimization (TRPO), introduced by Schulman et al. in 2015 [12], represents a sophisticated method in the domain of reinforcement learning, designed to optimize control policies with a guarantee of monotonic advancements. TRPO's primary principle involves identifying a trajectory that enhances policy efficacy while ensuring minimal deviations in policy alterations. This approach is exceptionally effective for the optimization of extensive, non-linear policy frameworks.

Within the TRPO framework, a policy, $\pi$, is conceptualized as a probability distribution governing the selection of an action, $a$, in a given state, $s$, expressed as $\pi(a|s)$. The overarching aim of this policy is the maximization of the expected cumulative rewards. TRPO primarily focuses on optimizing the following objective function:

$$maximize_\theta \; E_{s\sim p_{\theta'}, a\sim\pi_{\theta'}}\left[\frac{\pi_\theta(a|s)}{\pi_{\theta'}(a|s)} A_{\theta'}(s,a)\right] \tag{1}$$

Here, $\theta$ and $\theta'$ represent the policy parameters of the current and previous iterations, respectively. $A_{\theta'}(s,a)$ is the advantage function for state $s$ and action $a$ under policy $\pi_{\theta'}$.

In addition, Trust Region Policy Optimization (TRPO) employs a Kullback–Leibler (KL) divergence constraint to ensure that the magnitude of policy updates remains moderate [12]. The KL divergence is a measure used to quantify the difference between two probability distributions. In this context, it is utilized to quantify the variation in a policy before and after an update. Specifically, TRPO controls the magnitude of policy updates through the following KL divergence constraint:

$$s.t. E_{s\sim p_{\theta'}}[D_{KL}(\pi_{\theta'}(\cdot|s))] \leq \delta_{KL} \tag{2}$$

Here, $D_{KL}$ represents the Kullback–Leibler (KL) divergence, and $\pi_{\theta'}(\cdot|s)$ denotes the probability distribution obtained under state $s$ using the policy with parameters $\theta'$. The value $\delta_{KL}$ is a predefined threshold. By employing this approach, TRPO can effectively control the magnitude of changes during policy updates, preventing excessively drastic alterations. This ensures the stability and reliability of the learning process.

Proximal Policy Optimization (PPO), proposed by Schulman et al. in 2017 [16], is an improvement of TRPO. PPO simplifies the optimization process of TRPO and enhances sample utilization efficiency. PPO optimizes a policy through the following objective function:

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{old}(a_t|s_t)} \tag{3}$$

$$L(\theta) = \hat{\mathbb{E}}\left[\min\left(r_t(\theta)\hat{A}_t,\; clip(r_t(\theta),\, 1-\epsilon,\, 1+\epsilon)\hat{A}_t\right)\right] \tag{4}$$

In this context, $r_t(\theta)$ is the probability ratio, representing the likelihood of selecting an identical action under both the new and the old policies. $\hat{A}_t$ is the estimated value of the advantage function, and $\epsilon$ is a small constant utilized to limit the magnitude of policy updates. The 'clip' function confines the probability ratio, $r_t(\theta)$, within predetermined bounds of $1-\epsilon$ and $1+\epsilon$. This constraint on the probability ratio aids in preventing excessive oscillations during policy updates, thereby ensuring stability and reliability in the learning process. When the value of $r_t(\theta)$ falls below $1-\epsilon$, the 'clip' function adjusts it to $1-\epsilon$; conversely, when it exceeds $1+\epsilon$, it is corrected to $1+\epsilon$. Within these boundaries, the value of $r_t(\theta)$ remains unchanged.

Equation (3) defines the probability ratio, $r_t(\theta)$, which measures the ratio of the probabilities of choosing the same action under new and old policies. This ratio is the foundation for calculating the objective function of the PPO algorithm. Following this, Equation (4) presents the core objective function, $L(\theta)$, of the PPO algorithm, utilizing the probability ratio, $r_t(\theta)$, from Equation (3), in conjunction with the estimated value of the advantage function, $\hat{A}_t$, and the clip function, to optimize the policy.

PPO utilizes this approach to balance exploration and exploitation while diminishing reliance on extensive datasets, enhancing its suitability for practical applications.

### *3.2. Deterministic Policy Methods*

In addition to stochastic policies, the Deep Deterministic Policy Gradient (DDPG) emerges as a deterministic policy gradient approach [13], extending the success of DQN to the realm of continuous control. The DDPG adheres to the following update mechanism:

$$Q(s_t,\ a_t) = r(s_t,\ a_t) + \gamma Q(s_{t+1}, \pi_\theta(s_{t+1})) \tag{5}$$

This equation conveys that the estimated value of the function $Q$, given the current state, $s_t$, and action, $a_t$, is the sum of the immediate reward, $r(s_t,\ a_t)$, and the discounted value of future expected returns. The discount factor, $\gamma$, plays a critical role in weighting the significance of immediate versus future rewards. The term $Q(s_{t+1}, \pi_\theta(s_{t+1}))$ denotes the anticipated return associated with the optimal action as per policy θ in the subsequent state $s_{t+1}$. DDPG operates as an off-policy actor–critic algorithm adept at formulating effective policies across a spectrum of tasks.

The Twin Delayed DDPG (TD3) builds upon the DDPG [18], aiming to mitigate some of its inherent constraints. TD3 utilizes a dual-structured framework with two actor networks and two critic networks. These networks are co-trained to synergize and optimize the agent's performance. The twin actor networks are designed to yield disparate action outputs for each state, subsequently appraised by the dual critic networks [22]. Such an arrangement allows TD3 to more accurately navigate environmental uncertainties and curtail the propensity for value overestimations typical in DDPG.

### *3.3. Entropy-Regularized Policy Gradient*

Soft Actor–Critic (SAC) [17] is a sophisticated off-policy gradient technique that forms a nexus between DDPG and stochastic policy optimization strategies. SAC integrates the concept of clipped Double Q-learning, with its maximum entropy DRL objective function articulated as:

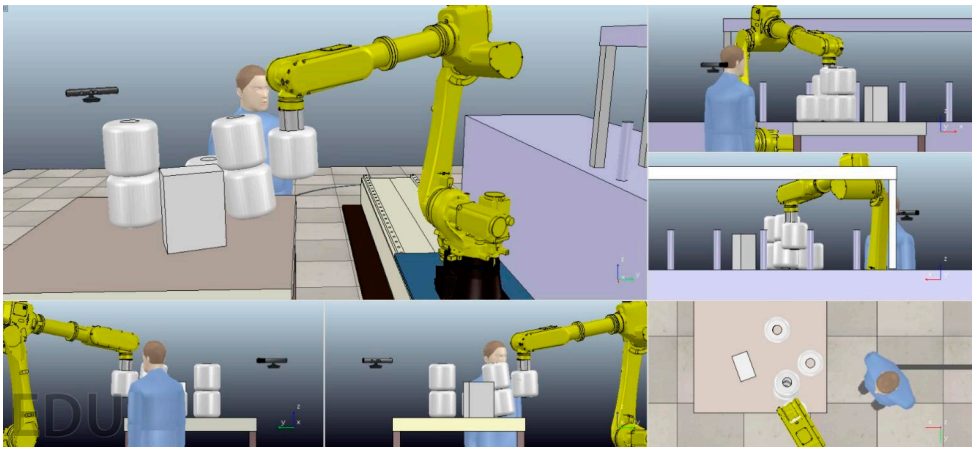
$$J(\pi) = \sum_{t=0}^{T} \mathbb{E}_{(s_t,a_t)\sim\rho_\pi}[r(s_t,a_t) + \alpha H(\pi(\cdot|s_t))] \tag{6}$$

In this context, $r(s_t,a_t)$ denotes the immediate reward obtained from executing action $a_t$ in state $s_t$. $H(\pi(\cdot|s_t)$ symbolizes the policy's entropy, reflecting the inherent randomness or uncertainty within a specific state. The $\alpha$ parameter serves as a crucial balance factor, calibrating the significance of the reward against entropy, thus effectively managing the policy's exploration–exploitation dynamics. The core objective of this approach is to optimize the policy $\pi$, maximize expected returns, and facilitate the optimal choice of action $a_t$ in any given state, $s_t$. SAC's objective function uniquely features entropy regularization, aiming to optimize a policy through a strategic equilibrium between entropy and anticipated returns. Entropy here quantifies the level of stochasticity in policy decisions, mirroring the critical balance between exploration and exploitation. Elevating entropy levels not only fosters increased exploration but also expedites the learning trajectory. Crucially, this mechanism also safeguards against a policy's convergence to suboptimal local extremums.

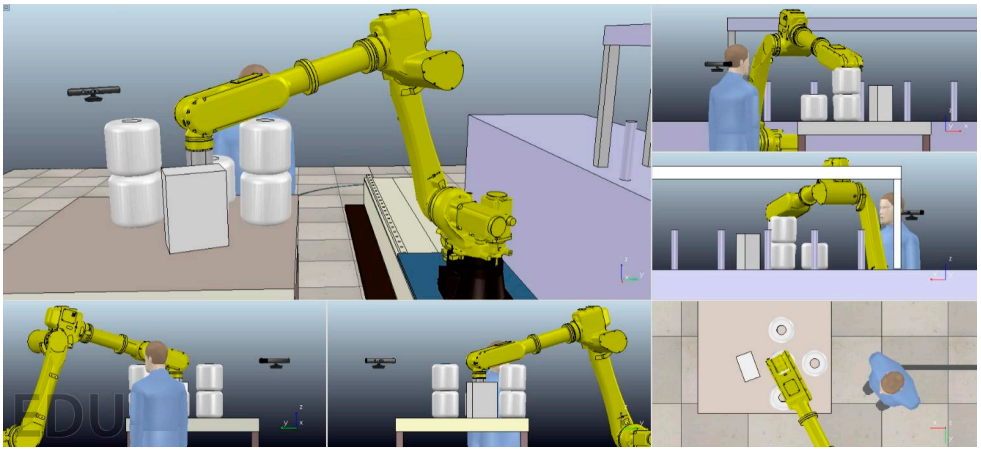
## 4. Environment Augmentation Strategy

In the field of robotics, trajectory planning for tasks is a pivotal endeavor. It encompasses the movement of objects from one point to another or the execution of intricate maneuvers. For this purpose, robots require an efficient trajectory planning strategy. Utilizing DRL for trajectory planning presents a highly promising approach. It leverages the dynamics of Markov Decision Processes (MDPs) to derive optimal strategies from multifaceted environments. Such a strategy enables robots to make optimal decisions based on real-time status during task execution, thereby crafting trajectories that fulfill specified criteria.

In conventional path-planning algorithms integrating inverse kinematics, intelligent textile robotic arms typically align directly above the center of a target yarn spool, subsequently inserting their grippers into the central hole of the spool and then opening them. However, this method can result in collisions between the robot arm and the upper inside of the yarn spool, as illustrated in Figure 2a. Such collisions can become unacceptably hazardous due to environmental variations. To counter this, we introduced an EA strategy tailored to refining the path-planning algorithm of DRL under the paradigm of inverse kinematics integration. This approach addresses the issue of overconcentration on the tool center point (TCP) and the consequent neglect of environmental data, thereby facilitating safer and more efficient trajectory planning. By incorporating MIEGO, the focus is expanded from a singular TCP issue to a broader spectrum of multi-objective optimization, leveraging environmental data to circumvent scenarios of local optima, as shown in Figure 2b.



(**a**)



(**b**)

**Figure 2.** Unintended collisions stemming from local optima. (**a**) Scenario 1; (**b**) scenario 2.

It is crucial first to clarify that in DRL, the definition of the environmental state is of utmost importance. In typical reinforcement-learning-based trajectory planning tasks, the environmental state encompasses: (1) the position of the TCP of the end effector and (2) relative positional data between the TCP and the target location. Typically, the state $s_t$ for trajectory planning is defined as:

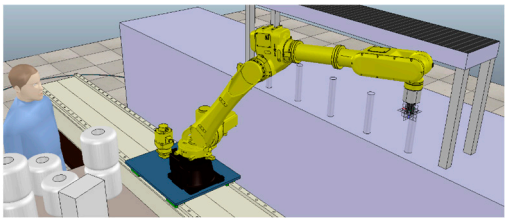
$$s_t = [\boldsymbol{TCP}, \delta, Done] \tag{7}$$

In this framework, the ***TCP*** is represented as a vector in the global coordinate system, capturing the position of the end effector. The term $\delta$ represents the offset vector between the end effector and its targeted object. The boolean variable *Done* is set to 1 when the end effector successfully reaches its target position; otherwise, it remains 0.

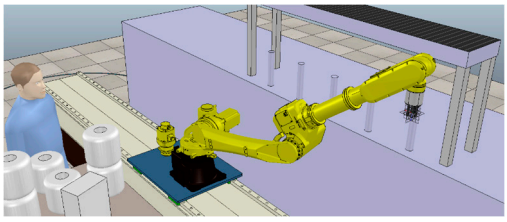
Throughout the task execution phase, the DRL model makes action decisions based on both the prevailing state and the strategy it has learned. To ensure adaptability across various robots and end effectors, the action $\boldsymbol{a_t}$ is defined in terms of the end effector's translational movement:

$$\boldsymbol{a_t} = [\Delta x, \Delta y, \Delta z] \tag{8}$$

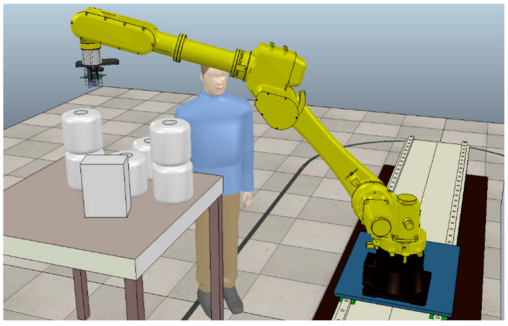
This approach to defining states and actions steers the continuous modification in the position and orientation of the robot's end TCP, crafting a seamless motion trajectory. However, despite facilitating the autonomous planning of task trajectories by the agent, the inherent multiplicity in inverse kinematics suggests multiple potential configurations of robotic arm joint angles (states) for any given TCP position and orientation. Figure 3 depicts multiple-solution conflict scenarios, a consequence of inverse kinematics, in two distinct cases.



(**a**)

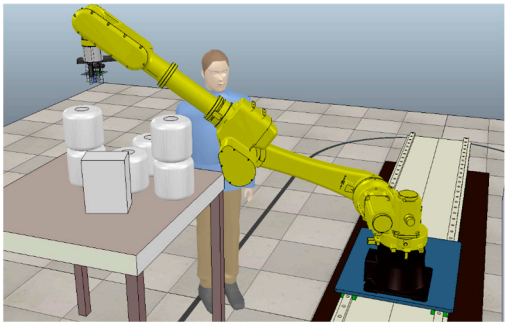


(**b**)



(**c**)

**Figure 3.** *Cont.*



(**d**)

**Figure 3.** Phenomenon of multiple-solution conflicts. (**a**) Scenario 1: normal solution; (**b**) scenario 1: abnormal solution; (**c**) scenario 2: normal solution; and (**d**) scenario 2: abnormal solution.

In the realm of DRL, models typically formulate a distinct policy (action) for each specified state. However, a complication arises when joint angles are not factored in and states are solely defined by the TCP position. This can lead to different states of the robotic arm corresponding to the same TCP position, yet the DRL model may produce identical policies. Such a scenario introduces several challenges:

1. For robotic arms with identical TCP positions but varying joint states, the ideal action could vary. Generating the same policy for these divergent states by the DRL model might compromise task execution efficiency or even result in task failure.
2. In tasks necessitating maneuvers in complex settings or confined spaces, the specific joint angle configurations of the robotic arm can critically influence task success. Policies derived solely from TCP positions may fall short of meeting these nuanced requirements.
3. Given that a robotic arm's state is influenced by both the TCP position and joint angles, overlooking joint angles means the state cannot comprehensively represent a robot's actual condition, potentially impeding the learning outcomes.

To surmount these DRL challenges, this study introduces a novel EA strategy, as delineated in Figure 4. This strategy centers around enhancing the state representation of the robotic arm within its environment, synergized with a thoughtfully constructed reward function. Initially, the enhanced state and the predefined reward function are fed into the agent. This step is followed by updating the actor–critic network parameters within the agent. Subsequently, the deep network, utilizing these updated parameters, computes the information for the robotic arm's subsequent target pose (TCP). Post-inverse kinematics processing, these data inform the action angles for the robotic arm's joints. These angles are then communicated to the robotic arm, directing it to perform the planned actions. This iterative process facilitates the ongoing refinement of the robotic arm's control strategy.

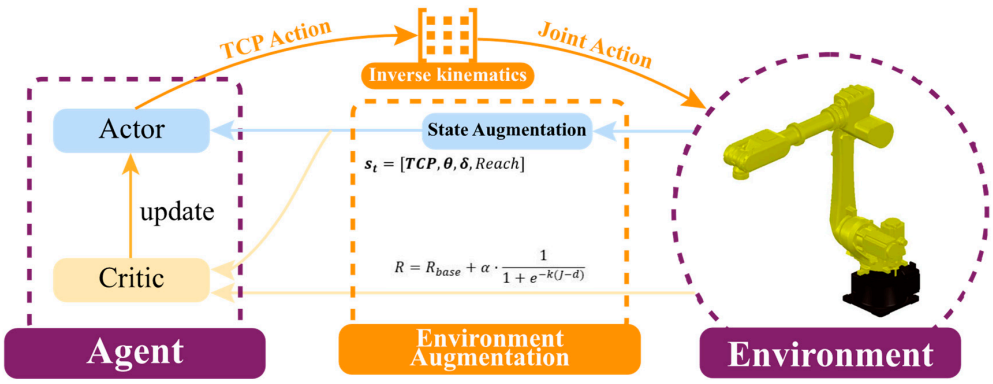



**Figure 4.** Environmental enhancement strategies.

In DRL, the State Augmentation strategy involves introducing the states of the robotic arm's joint angles into the environmental state. This approach provides a more accurate reflection of a robotic arm's actual state, thereby generating more effective strategies. The state $\boldsymbol{s_t}$ is defined as:

$$\boldsymbol{s_t} = [\boldsymbol{TCP}, \boldsymbol{\theta}, \delta, Done] \tag{9}$$

Here, the definitions of $\boldsymbol{TCP}$, $\delta$, and $Done$ remain consistent with earlier descriptions, while $\boldsymbol{\theta}$ represents the angles of the robotic arm's joints, providing a more accurate representation of the robot's current state.

Given that the robotic arm used in the scenarios of this paper is the six-DOF ER10-2000 model (EFORT Intelligent Equipment Co., Ltd., Foshan, China), the state incorporates the rotational angles of the robotic arm's six axes to offer additional information for learning. In DRL, the effectiveness of model learning often depends on the amount of available information. Introducing joint angles provides more data for the model, potentially aiding in learning better strategies. Furthermore, for robots with redundant degrees of freedom, multiple joint angle configurations can achieve the same TCP position. Incorporating joint angles allows the strategy to better utilize these redundant degrees of freedom, potentially resulting in a greater variety of actions.

This paper also uses the change in inverse solutions as a reference index for measuring the stability of solutions. This index serves as an additional reward element, guiding the agent to prioritize exploring paths with higher stability and achieving smoother trajectories that avoid collisions and enhance efficiency.

Initially, $\theta(t) \in \mathbb{R}^6$ represents the robotic arm's joint angles at the current time step, and $\theta(t-1) \in \mathbb{R}^6$ represents the joint angles at the previous time step. The corresponding planned TCP positions are $q(t) \in \mathbb{R}^3$ and $q(t-1) \in \mathbb{R}^3$. The following formula is established:

$$\theta(t) = f(q(t)) \tag{10}$$

With the corresponding differential relationship:

$$\dot{\theta}(t) = \frac{\partial f(q(t))}{\partial t} = J(q(t)) \tag{11}$$

Here, $J(q(t)) \in \mathbb{R}^6$ is the Jacobian matrix for a six-DOF robotic arm, representing the rotational velocity of each joint. For differential mobile robotic arms, it signifies the rate of change in angles or the rotational acceleration of each joint.

To measure the stability of solutions without excessively rewarding minor changes, this paper employs a variant of the sigmoid function for the nonlinear mapping of stability rewards, defined as:

$$S = \frac{1}{1 + e^{-k(J-d)}} \tag{12}$$

Here, $k$ is a positive parameter controlling the slope of the function, and $d$ is a parameter controlling the center of the function.

When $J$ is close to 0, $S$ will be a small value. As $J$ increases, $S$ will smoothly increase. The advantage of this function is that it does not overly reward very small changes, and as the change magnitude reaches a certain level, the reward increases but at a slower rate.

If the stability measure dominates the reward function, the agent might minimize the change in solutions during learning to maximize rewards, leading to local optima or inaction. To avoid this, the paper introduces a weight parameter and incorporates this stability measure into the overall reward function:

$$R = R_{base} - \alpha \cdot \frac{1}{1 + e^{-k(J-d)}} \tag{13}$$

Here, $R_{base}$ is the basic reward, such as the reward for reaching the target position or other task-related rewards; $\alpha$ is a weight parameter, controlling the importance of the stability measure in the overall reward.

## 5. Multi-Information Entropy Geometric Optimization Method

This chapter delves into the core aspects of the MIEGO method. This includes establishing multi-objective optimization problems, incorporating information entropy, and utilizing natural gradients for geometric optimization. Figure 5 illustrates the basic framework of this method.

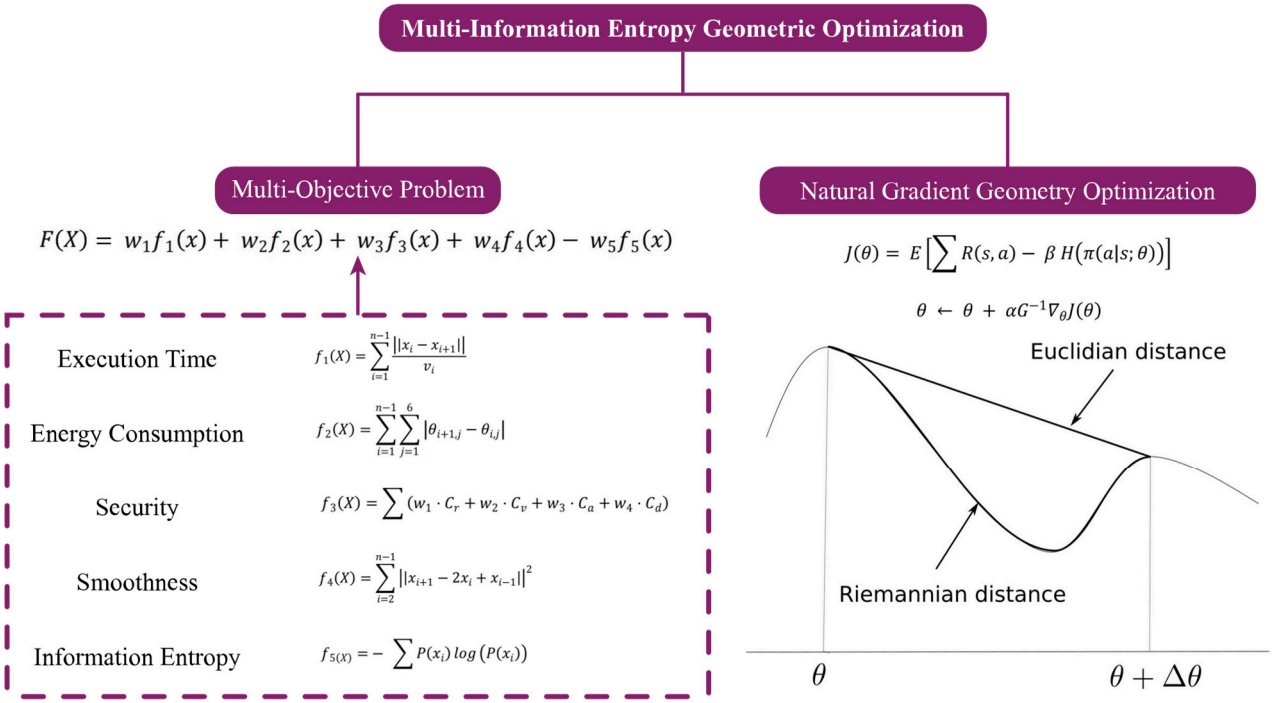



**Figure 5.** Schematic of MIEGO method.

### *5.1. Construction of Multi-Objective Optimization Problems*

The primary goal of robotic arm path-planning tasks is to identify viable motion paths that satisfy a set of predefined objectives within given constraints. These objectives typically encompass minimizing motion time, reducing energy consumption, enhancing safety, optimizing motion smoothness, improving precision, adhering to specific constraints, maximizing work efficiency, and minimizing mechanical stress [23]. In this chapter, we initially provide a detailed definition of these objectives and then explore how to construct and solve a multi-objective optimization problem encompassing these goals.

#### 5.1.1. Motion Time

Motion time refers to the total time required for a robotic arm to move from its initial state to the target state. In intelligent textile manufacturing environments, tasks assigned to textile robotic arms often need to be completed within a certain timeframe to ensure process continuity and production efficiency. We define a time cost function, $f_1(X)$, to quantify the motion time of the robotic arm. Specifically, $f_1(X)$ can be defined as the total time needed to move from the initial state to the target state. However, the per-step motion time, $\Delta t$, obtained through different DRL algorithms may vary due to differences in step sizes [24]. Therefore, we define motion time as the sum of the ratios of the step lengths to their velocities:

$$f_1(X) = \sum_{i=1}^{n-1} \frac{||x_i - x_{i+1}||}{v_i} \tag{14}$$

Here, $X = [x_1, x_2, x_3, \ldots, x_n]$ represents a path composed of n points from the initial state to the target state. $x_i (i = 1, 2, \ldots, n)$ describes the vector of the robotic arm's TCP, including its position and rotation state. $v_i$ is the motion velocity at each step. When the TCP sequence, $X$, of the robotic arm minimizes $f_1(X)$, we say that the optimal motion time path has been found.

5.1.2. Energy Consumption

Energy consumption is another objective considered for sustainable development and energy conservation. We define an energy consumption cost function, $f_2(X)$, to quantify the energy usage of the robotic arm. Similar to $f_1(X)$, $f_2(X)$ can be defined as the total energy consumed by the robotic arm in moving from the initial state to the target state [25]. We define energy consumption as the total angular mileage of joint rotations:

$$f_2(X) = \sum_{i=1}^{n-1} \sum_{j=1}^{6} \left| \theta_{i+1,j} - \theta_{i,j} \right| \tag{15}$$

where $X$ is the same as mentioned above, and $\theta_{i,j}$ represents the absolute angle of joint $j$ in the $i$-th step of the sequence, $X$. When the TCP sequence, $X$, of the robotic arm minimizes $f_2(X)$, we say that the optimal energy consumption path has been found.

5.1.3. Safety

In many scenarios where robotic arms share a workspace with humans or other machines, safety is an indispensable consideration. We define a safety cost function, $f_3(X)$, based on factors such as the robotic arm's range of motion, velocity, acceleration, and minimum distance to people or objects to quantify the safety of the robotic arm:

$$f_3(X) = \sum (w_1 \cdot C_r + w_2 \cdot C_v + w_3 \cdot C_a + w_4 \cdot C_d) \tag{16}$$

Generally, the safety cost should be a non-negative function that decreases as safety increases. Moreover, it should be as smooth as possible, meaning its derivatives are continuous across the entire domain, as many optimization algorithms, like gradient descent, require the function's derivatives [26]. Based on this analysis, we define the following sub-functions for the safety cost function: $C_r$ is the cost function for the range of motion:

$$C_r = \sum_{i=1}^{6} \exp\left( -\frac{(\theta_{max,i} - \theta_i)(\theta_i - \theta_{min,i})}{\sigma^2} \right) \tag{17}$$

Here, $\theta_i (i = 1, 2, 3, 4, 5, 6)$ represents the angle of the $i$-th joint, with a safe motion range set as $[\theta_{min}, \theta_{max}]$. This function rapidly increases as joint angles approach their limits, with $\sigma$ being a parameter that regulates the growth rate of the function.

$C_v$ and $C_a$ are cost functions for velocity and acceleration, respectively:

$$C_v = \sum_{i=1}^{n} \left( \frac{\dot{\theta}_i}{v_{max,i}} \right)^2 \tag{18}$$

$$C_a = \sum_{i=1}^{n} \left( \frac{\ddot{\theta}_i}{a_{max,i}} \right)^2 \tag{19}$$

Here, $\dot{\theta}_i$ and $\ddot{\theta}_i$ represent the absolute angular velocity and acceleration of the i-th joint, respectively, with $v_{max}$ and $a_{max}$ denoting the maximum velocity and acceleration for each joint. The cost increases rapidly when the velocity or acceleration exceeds its maximum allowable value.

$C_d$ is the cost function for the range of motion:

$$C_d = \frac{1}{1 + \exp\left(\frac{d_{min} - d_{safe}}{\delta}\right)} \tag{20}$$

Here, $d_{min}$ is the minimum distance between the robotic arm and the closest person or object in the environment, $d_{safe}$ is a predefined safe distance, and $\delta$ is a parameter that adjusts the growth rate of the function. The function is monotonic, increasing in value as the distance between the robotic arm and other objects or humans in the environment decreases.

Therefore, when the TCP sequence, $X$, of the robotic arm minimizes $f_3(X)$, we say that the optimal safety path has been found.

#### 5.1.4. Motion Smoothness

We aim for the robotic arm's motion to be as smooth as possible to minimize shocks and vibrations, thereby enhancing the quality and stability of operations. We define a smoothness cost function, $f_4(X)$, to quantify the motion smoothness of the robotic arm:

$$f_4(X) = \sum_{i=2}^{n-1} ||x_{i+1} - 2x_i + x_{i-1}||^2 \tag{21}$$

Here, $x_i$ is defined as in previous sections. This formula essentially calculates the sum of the squares of the discrete acceleration of the TCP trajectory. In cases of smooth motion, the acceleration should be as minimal as possible; hence, the cost function decreases with smoother motion. The advantage of this method is its direct computability from the sequence of TCP points, facilitating network optimization in conjunction with inverse kinematics [27]. Therefore, when the TCP sequence, $X$, of the robotic arm minimizes $f_4(X)$, we say that the optimal smooth path has been found.

Based on the definitions above, we can construct a multi-objective optimization problem to simultaneously satisfy all these objectives under given constraints. Specifically, we seek to find a solution, $X$, that minimizes the following objective function, $F(X)$:

$$F(X) = w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x) + w_4 f_4(x) \tag{22}$$

Here, $w_i$ represents weight coefficients used to balance the importance of different objectives. In practical applications, the choice of $w_i$ typically depends on specific task requirements and environmental conditions. For instance, if safety is prioritized over motion time, one might choose $w_3 > w_1$; if motion smoothness is more critical than energy consumption, then $w_4 > w_2$ could be selected.

### *5.2. Information Entropy and Geometric Optimization*

In the process of seeking optimal solutions, we introduce a crucial concept: information entropy. Originally proposed by Claude Shannon in his study of information transmission in communication systems, information entropy serves as a significant tool to measure the uncertainty or randomness of information.

#### 5.2.1. Mathematical Definition of Information Entropy

Information entropy is a concept widely used in probability theory and information theory, measuring the uncertainty of a random variable. For a random variable, $X$, its information entropy, $H(X)$, is defined as:

$$H(X) = -\sum P(x_i) log(P(x_i)) \tag{23}$$

Here, $P(x_i)$ represents the probability of the random variable X assuming the value $x_i$. Since the calculation of information entropy is based on a probability distribution, for

continuous random variables, their probability density functions are required to compute the information entropy.

#### 5.2.2. Application of Information Entropy in Robotic Arm Path Planning

In our context, robotic arm path planning is viewed as finding the optimal solution within all possible solution spaces. Each potential solution can be considered a random event, with its occurrence probability represented by the corresponding objective function value. Therefore, the uncertainty of solutions can be gauged by calculating the information entropy of the solution space.

Generally, greater information entropy indicates a higher uncertainty of the solution, and vice versa. In practical applications, we often seek solutions with high certainty, i.e., lower information entropy. However, in multi-objective optimization problems, conflicting objectives may lead to multiple solutions that satisfy all objectives, forming a Pareto optimal set. In such cases, we aim to find solutions that not only meet all objectives but also possess high certainty, that is, the solutions with the highest information entropy in the Pareto optimal set.

Here, we introduce a new objective function, $f_5(X)$, to quantify the information entropy of solutions:

$$f_5(X) = -\sum P(x_i) log(P(x_i)) \tag{24}$$

In this equation, $P(x_i)$ represents the probability of the solution $x_i$, which can be indicated by the objective function value. Thus, our multi-objective optimization problem becomes finding a solution $X$ that minimizes the following objective function, $F(X)$:

$$F(X) = w_1 f_1(x) + w_2 f_2(x) + w_3 f_3(x) + w_4 f_4(x) - w_5 f_5(x) \tag{25}$$

In this reformulated optimization problem, we seek solutions that not only satisfy all objectives but also have the maximum information entropy.

#### 5.2.3. Information Entropy in DRL

In DRL, policy networks typically represent the behavioral strategies of robots. The output of a policy network is a probability distribution, indicating the likelihood of executing various actions in a given state [28]. During training, the goal is to find a policy that maximizes cumulative rewards. However, in practical applications, it is often desirable for a robot's behavior to exhibit a degree of randomness, enabling exploration in unknown environments [29]. One study [17] suggests increasing the randomness of a policy by maximizing its information entropy.

Thus, an information entropy term is incorporated into the objective function of reinforcement learning as follows:

$$J(\theta) = E\left[\sum R(s,a) - \beta\, H(\pi(a|s;\theta))\right] \tag{26}$$

Here, $R(s,a)$ represents the reward function, $H(\pi(a|s;\theta))$ is the policy's information entropy, and $\beta$ is a hyperparameter controlling the balance between rewards and information entropy.

This approach not only makes the robot's behavior more random but also stabilizes the policy. During training, too low an information entropy might lead to overfitting, whereas too high an entropy could degrade policy performance. Therefore, by maximizing information entropy, policy stability is enhanced while ensuring performance.

#### 5.2.4. Natural Gradient Geometric Optimization

Conventional gradient methods, such as gradient ascent or descent, often consider only the gradient direction at the current position, overlooking the local geometric structure of the parameter space. This can lead to excessively large or small steps in parameter updates, resulting in unstable learning or low efficiency. Hence, this paper employs

"natural gradients" for optimization, a method that explicitly considers the Riemannian structure of the policy parameter space, focusing more on the local geometric structure of the parameter space.

Specifically, the natural gradient method first defines a Riemannian metric, setting the local distance in the parameter space. Then, it updates parameters according to this metric, maintaining the smoothness of policy changes throughout the update process.

The natural gradient update rule is applied as follows:

$$\theta \leftarrow \theta + \alpha G^{-1} \nabla_\theta J(\theta) \tag{27}$$

In this formula, $\alpha$ is the learning rate, $\nabla_\theta J(\theta)$ is the gradient of the objective function, $J(\theta)$, with respect to parameters $\theta$, and $G$ is the policy's Fisher information matrix. $G^{-1}\nabla_\theta J(\theta)$ represents the natural gradient.

By combining information entropy and natural gradient methods, we can make the robot's behavior more random and better consider the geometric structure of the policy parameter space, leading to more effective optimization.

## 6. Experimental Analysis

This chapter first establishes a virtual twin experimental environment for the intelligent textile robotic arm, as illustrated in Figure 6, and defines the task objectives. Subsequently, this paper employs four policy-gradient-based DRL algorithms, DDPG, TD3, SAC, and PPO, and integrates our EA strategy and MIEGO method into these algorithms. A series of experiments are then conducted to evaluate the proposed optimization strategies and methods, demonstrating their superior performance.

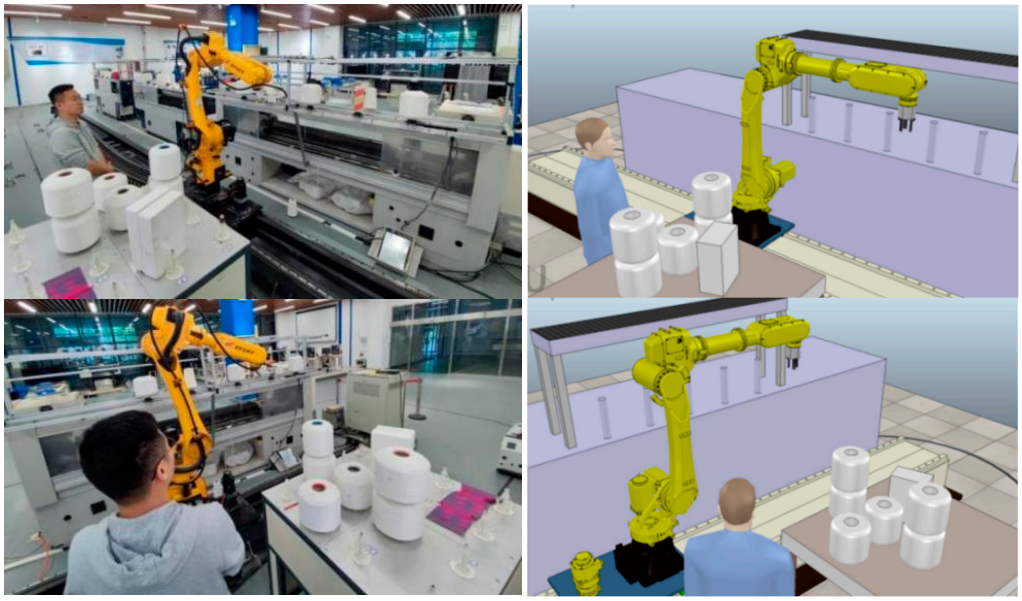


**Figure 6.** Virtual twin system.

### *6.1. Intelligent Textile Robotic Arm and Its Virtual Twin*

The intelligent textile robotic arm model involves the geometric structure of industrial robot arms, joint parameters, and kinematic modeling. Tables 1 and 2 and Figure 7 present the parameters and motion range of the real-world textile robot hand ER10-2000. Based on these parameters, a model of the textile robot hand is developed in VREP on a 1:1 scale ratio, as shown in Figure 8, achieving a high-fidelity simulation of the real-world scenario.

**Table 1.** Overall parameters of robotic arm ER10-2000.

| **Model** | ER10-2000 |
|---|---|
| **Number of Axes** | 6 axis |
| **Wrist Load Capacity** | 10 kg |
| **Repeatability** | ±0.06 mm |
| **Weight** | 221 kg |
| **Maximum Reach** | 2019 mm |
| **Drive Type** | AC Servo Motor |

**Table 2.** Detailed data of robotic arm and hand ER10-2000.

| | **Wrist Allowable Torque** | **Wrist Allowable Inertia Torque** | **Maximum Single-Axis Speed** | **Motion Range of Each Axis** |
|---|---|---|---|---|
| Joint 1 | - | - | 170°/s | ±175° |
| Joint 2 | - | - | 150°/s | +64°/−142° |
| Joint 3 | - | - | 146°/s | +165°/−81° |
| Joint 4 | 22 N·m | 1 kg·m$^2$ | 360°/s | ±178° |
| Joint 5 | 22 N·m | 1 kg·m$^2$ | 360°/s | ±128° |
| Joint 6 | 16 N·m | 0.3 kg·m$^2$ | 550°/s | ±720° |

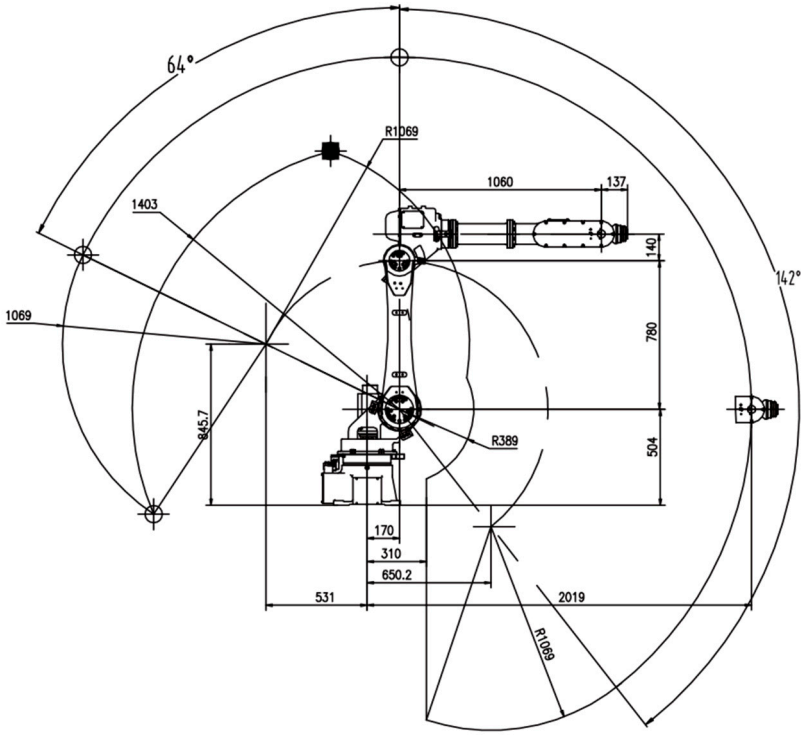



**Figure 7.** Motion range of robotic arm ER10-2000.

To acquire more learnable data in realistic textile yarn-gripping robotic hand scenarios, this study constructs a virtual twin system. This system accurately replicates the 1:1 kinematic model of the robotic arm and hand, enabling extensive training within the simulation environment. This approach allows for the collection of a substantial amount of high-quality training data, which can be directly applied to the trajectory generation of the actual robot.

The introduction of the virtual twin system not only optimizes the efficiency and effectiveness of DRL training for the robotic arm and hand but also reduces the costs and safety risks associated with physical training. By simulating different production scenarios

and environmental conditions, the virtual twin system can provide more comprehensive and accurate training data for the actual robot, testing its robustness and reliability in various settings. Furthermore, safety constraints can be tested and optimized in the virtual environment, ensuring the safety and stability of the robot in actual operations. Thus, the virtual twin technology offers a more efficient and reliable solution for the textile production and processing industry.

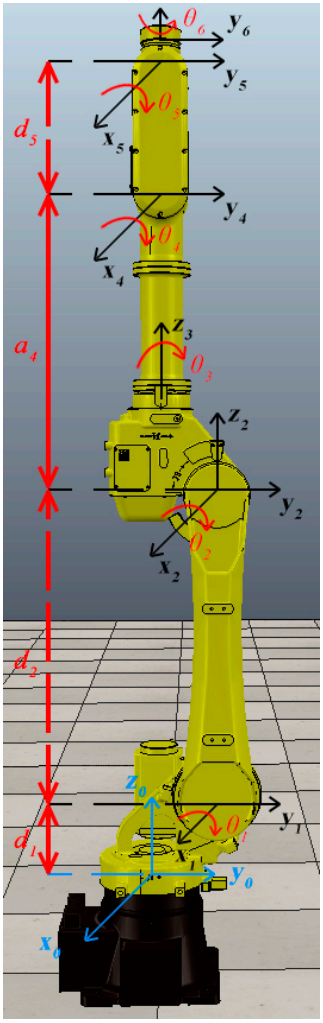



**Figure 8.** Constructed model of robotic arm and hand.

This paper utilizes VREP as the simulation platform, specifically V-REP PRO EDU Version 3.6.2. This software provides a powerful simulation environment, supporting a variety of robot models and algorithms and facilitating high-quality training of the robotic arm and hand in the virtual environment. Using VREP, this study can quickly set up and test the virtual twin system, laying a solid foundation for the practical application of the robotic arm and hand.

### *6.2. Experimental Setup*

Figure 9 displays the continuous trajectory planning scenario for the textile robotic hand within the virtual twin platform, with the task described as follows.

At the start of the task, the robotic arm is positioned at the initial location, point A, and its vicinity. To the right of the robotic hand is a table with various obstacles of different sizes and dimensions, which will be detailed in subsequent chapters. Also placed on the table is a yarn spool, the object to be grasped and relocated. Above the yarn spool is point B,

where the robotic hand's end effector needs to be positioned vertically downward during grasping. The end effector inserts its grippers into the central hollow of the yarn spool and opens them, lifting the spool using friction before placing it in the designated area, point C, on the left platform with the correct posture. To generalize the problem, only one yarn spool in the middle is selected on the virtual twin platform, but it can be positioned in different areas before each simulation to accommodate more possibilities.

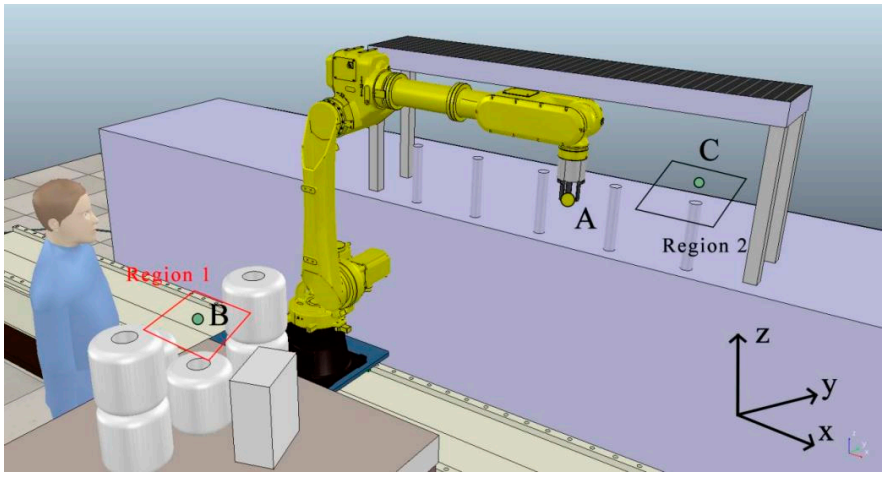



**Figure 9.** Continuous trajectory planning task for textile robotic hand.

Assume the end effector of the robotic hand is initially at point A, and the center of the yarn spool is located above point B (and its vicinity). To successfully lift the spool, the end effector of the robotic hand must first move a short distance along the $-x$ axis, then move along the $-y$ axis, continue moving along the $-x$ axis, and approach the yarn spool with the appropriate posture before lifting it along the $z$-axis. It is foreseeable that if the robotic hand moves directly from point A to point B, the robotic arm's joints (or links) will inevitably collide with the beams in the environment. Therefore, the grasp–place task is divided into the following steps, with the complete trajectory composed of continuous segments:

1. The yarn spool appears at any location on the table, and the TCP of the robotic arm moves from the initial position, A, along trajectory segment 1 to the preparation position, B, with the appropriate posture. Point B is located directly above the center of the yarn spool along the $z$-axis and may appear anywhere in Region 1.
2. The robotic arm moves along trajectory segment 2 from the preparation position, B, to the placement position, C (which can be randomly specified in Region 2), retracts its gripper, and places the yarn spool on the platform.
3. The arm returns to the vicinity of the initial position, A, from the placement position, C.

In this paper's experiments, the six-DOF intelligent textile robotic arm model on the virtual twin platform is used to complete the task along trajectory 1. In this environment, the coordinates of point A (the starting point) are (1.3019, 0.0319, 1.5801), and those of point B (the center of the yarn spool) are (0.7500, −1.3000, 1.2000). Therefore, the distance between A and B is approximately 0.7228 m. A step length, $a \in (0, 0.03)$, is chosen, so the straight-line distance between A and B requires at least 24 steps (all distance units are in meters).

### *6.3. Experimental Results*

In our experiments, we compared the performance of PPO, SAC, DDPG, and TD3 and their corresponding environment-augmented strategy algorithms, EA-PPO, EA-SAC, EA-DDPG, and EA-TD3. The specific parameters for each algorithm are detailed in Table 3.

**Table 3.** Key parameters of the employed DRL algorithms.

| | PPO | SAC | DDPG | TD3 |
|---|---|---|---|---|
| learning_rate | 0.0003 | 0.0003 | 0.001 | 0.001 |
| n_steps | 2048 | - | - | - |
| batch_size | 64 | 256 | 100 | 100 |
| gamma | 0.99 | 0.99 | 0.99 | 0.99 |
| tau | - | 0.005 | 0.005 | 0.005 |
| action_noise | - | - | Gaussian | Gaussian |

The training comparison data presented in Figure 10 show that both PPO and EA-PPO were successful in learning appropriate paths over the course of training, ultimately converging to similar reward levels. Notably, due to the requirement of EA-PPO to explore the environment more extensively, its convergence rate was relatively slower. For SAC and EA-SAC, their performance was generally similar, but the step count results indicate that EA-SAC required more steps in the process of exploring suitable paths. The difference between DDPG and EA-DDPG was more pronounced, with EA-DDPG spending more time exploring quality paths due to the influence of inverse solution stability rewards and the uncertainty of environmental information. Additionally, EA-DDPG exhibited a larger error band compared to DDPG, indicating increased fluctuation in reward values across multiple experiments due to the EA strategy. Since TD3 is an improved version of DDPG, in the experiments, relative to DDPG, both TD3 and EA-TD3 achieved convergence faster, although EA-TD3 still required more steps to find ideal paths.

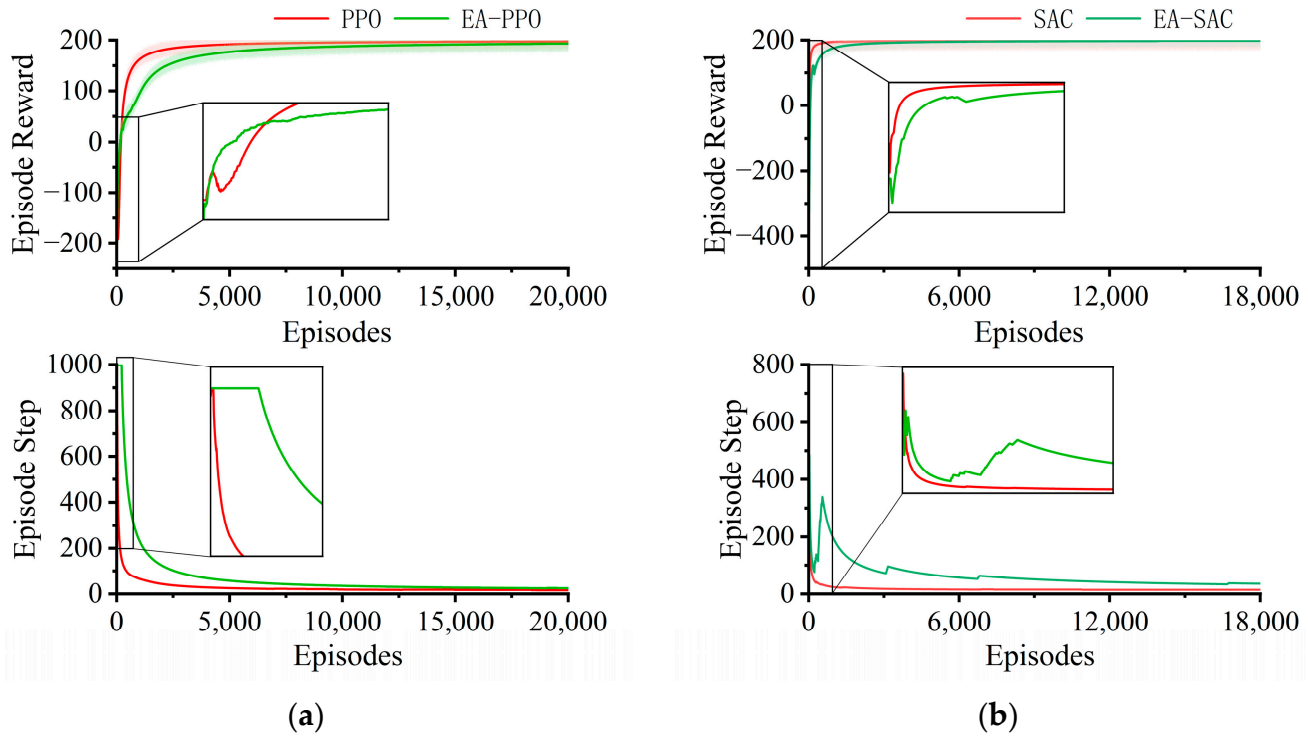



**Figure 10.** *Cont.*

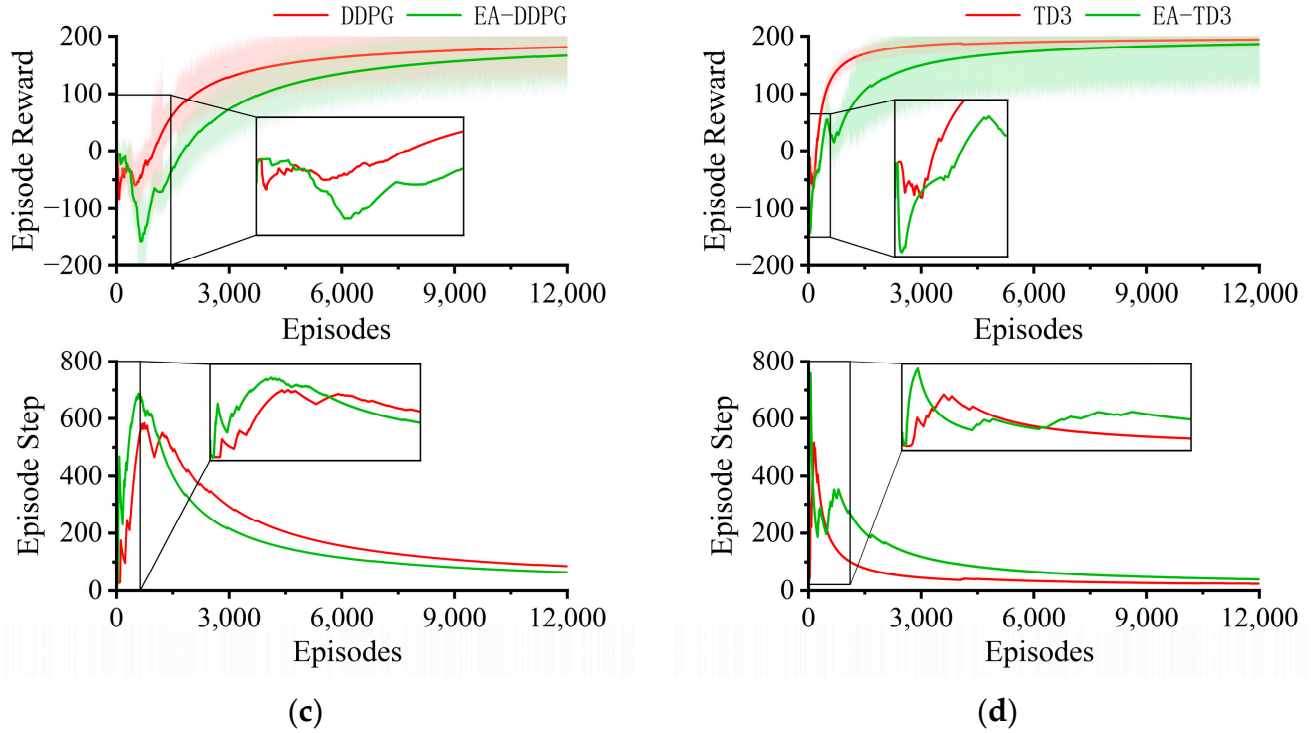



**Figure 10.** Training comparison between standard algorithms and EA strategy algorithms. (**a**) Comparison of PPO and EA-PPO training curves; (**b**) Comparison of SAC and EA-SAC training curves; (**c**) Comparison of DDPG and EA- DDPG training curves; and (**d**) Comparison of TD3 and EA- TD3 training curves.

Figure 11 reveals the impact of the EA strategy on the stability metric of the solutions, especially the stability measure referenced by the S value. As training progressed, a gradual decrease in the S value was observed. This trend indicates that the algorithms produced more stable solutions over time. Furthermore, this stability was reflected in the smoothness of the search paths. In other words, as the S value decreased, the exploration paths became smoother. This is particularly important in practical applications, as smoother paths typically mean more predictable and controllable decision-making processes.

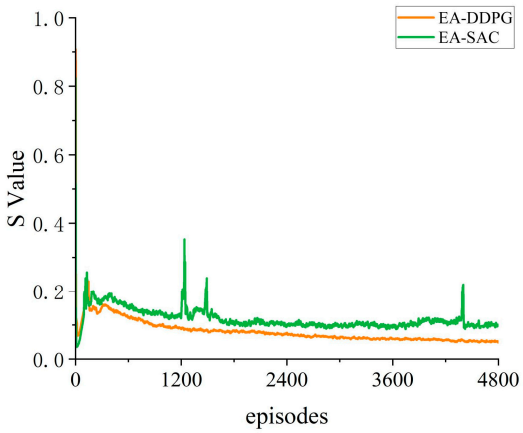



**Figure 11.** Stability reward variation curve of the solutions.

Figure 12 illustrates significant differences between the environment-augmented strategy versions of PPO, SAC, DDPG, and TD3 and their original DRL algorithms across various performance indicators. At first glance, the paths planned by the versions with the EA strategy seem longer with extended execution times, giving an initial impression of

lower efficiency. However, this reflects a key feature of the EA strategy: a deeper and more comprehensive exploration of the environment. The strategy aims to thoroughly analyze the environmental structure, uncover potential risks, and thus provide more comprehensive and refined decision making.

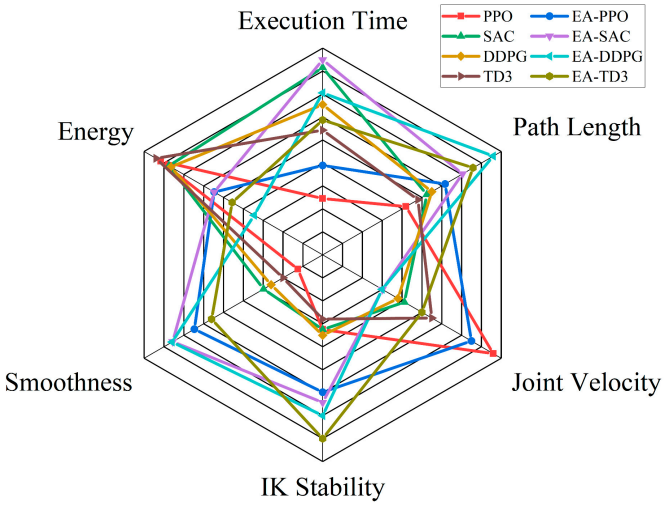



**Figure 12.** Radar chart comparing the performance of standard algorithms and EA strategy algorithms.

The experiments validate the effectiveness of our EA strategy. During training, our strategy continually optimizes stability indicators, producing higher-quality solutions. Ultimately, this enhancement in stability and improvement in path smoothness provide strong support for the practical application of our EA strategy.

Furthermore, the EA strategy emphasizes robust control policies, prioritizing precise and cautious movements over speed. This strategy not only reduces joint speeds (except for EA-PPO, where all environment-augmented versions have lower joint speeds than their original versions) but also enhances inverse kinematics stability. By choosing more robust and precise inverse kinematics solutions, the environment-augmented versions can achieve smoother path planning and more stable joint movements, effectively reducing energy consumption. Specifically, the EA strategy brings benefits in terms of energy saving and also helps to reduce the overall lead time. Although this strategy may result in more steps and longer execution times in the initial stages of a task, it effectively reduces repetitions or adjustments due to environmental changes or imprecise path planning over the entire task cycle. This ability to reduce repetitive actions, in the long run, not only lowers energy consumption but also reduces the overall completion time of a task. In other words, the EA strategy enhances efficiency and safety in the long term by improving the accuracy and stability of path planning, thereby avoiding operational failures and safety incidents due to environmental changes or misinterpretations [30].

In summary, while the EA strategy may show weaker performance in terms of steps and execution time compared to the original algorithms, its deep exploration and understanding of the environment, along with its enhanced focus on operational safety and stability, reveal its significant advantages and resilience in dealing with complex environments and unknown risks.

Additionally, this paper selects the widely applied DDPG and SAC algorithms in continuous space path planning as benchmarks for comparison and conducts an in-depth analysis of the path planning results. We designed and implemented an improved algorithm combining the EA strategy and MIEGO method (referred to as the EM algorithm), specifically for training on the first path. Comparative observations of the final paths planned by these algorithms reveal the effectiveness and innovation of the EM algorithm

in path optimization, further confirming the practical value of the EA strategy in intelligent textile robotic arm path planning.

Figure 13 shows the paths planned by four different algorithms, from left to right: EM-SAC, SAC, DDPG, and EM-DDPG. It is observed that the paths planned by EM-SAC and EM-DDPG are more circuitous and maintain a greater distance from potential obstacles. This strategy reduces safety risks that could arise due to minor environmental changes, electrical control factors, etc., during actual movement. More importantly, the paths planned by these two algorithms are more streamlined, better suited for smooth motor operation, and can effectively avoid frequent motor starts and stops, thus extending the motor's lifespan [31].

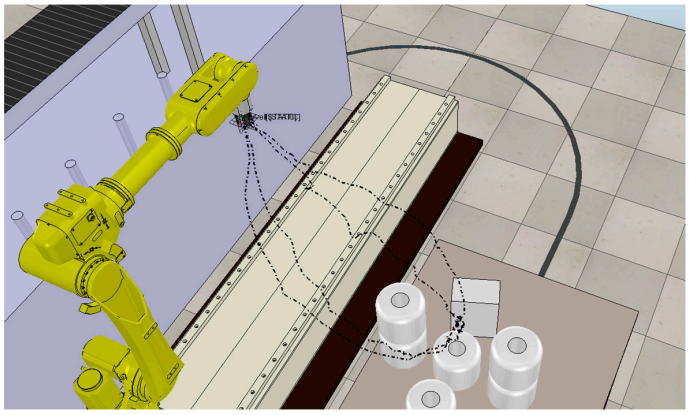


**Figure 13.** Final paths planned by original algorithms and EM algorithm.

Figure 14 presents the operational conditions of joint angles in the optimal paths planned by the original algorithms and the EM algorithm. In the figure, (a) and (b) correspond to the EM-SAC and SAC algorithms, while (c) and (d) correspond to DDPG and EM-DDPG, respectively. In comparison, it is noticeable that the variation curves in (a) and (d) are significantly smoother and exhibit less jitter, almost negligible. This phenomenon indicates that the paths planned through the EM algorithm have a higher degree of smoothness. This feature of high smoothness means that in practical applications, the robotic arm's movement will be considerably less jittery, thus achieving more precise positioning. Additionally, smoother motion paths can reduce the robotic arm's instantaneous acceleration, thereby mitigating wear and tear and prolonging its lifespan. Furthermore, smoother movement paths also reduce the robotic arm's demand for electricity, consequently lowering energy consumption [4]. Therefore, the excellent performance of the EM algorithm in this regard further confirms its superior performance and broad application prospects in real-world applications.

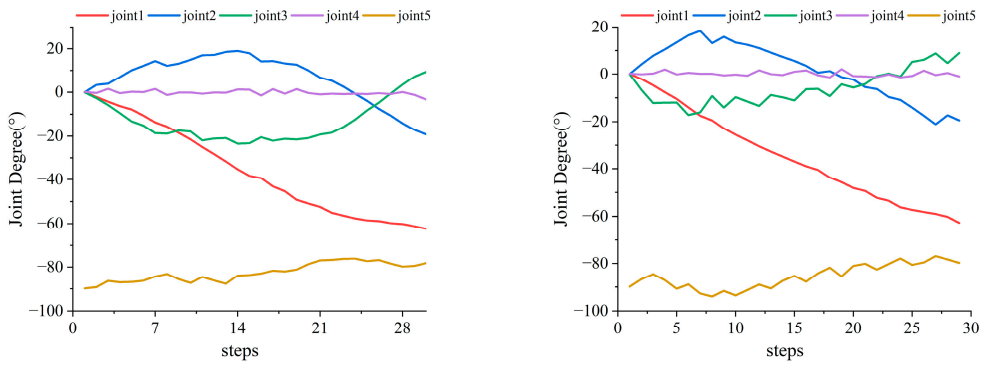



**Figure 14.** *Cont.*

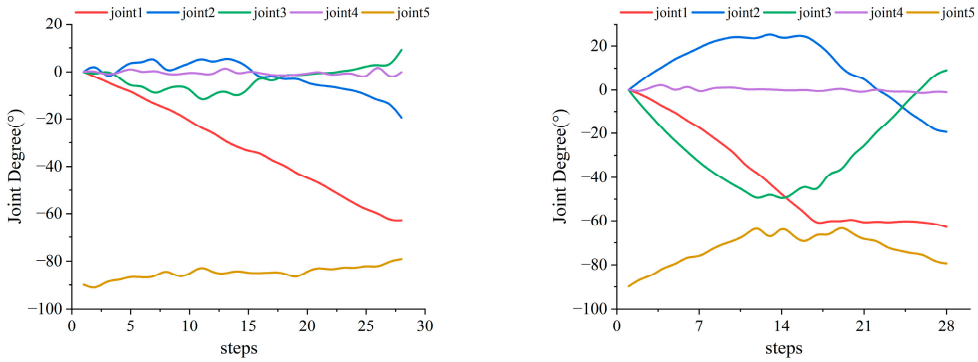



**Figure 14.** Operational conditions of joint angles in optimal paths planned by original algorithms and EM algorithm. (**a**) EM-SAC; (**b**) SAC; (**c**) DDPG; (**d**) EM-DDPG.

In conclusion, the EA strategy's ability to deeply explore and understand the environment, ensuring operational safety and stability, makes it highly adaptable and superior for tasks involving complex environments and unknown risks. The integration of the EA strategy with the MIEGO method (EM algorithm) in path planning effectively balances factors such as path length, motion safety, and equipment durability, providing an effective strategy and methodology for achieving efficient and safe robotic arm movements in complex environments.

## 7. Conclusions

In this paper, we successfully addressed a series of challenges associated with inverse kinematics integrated path-planning algorithms in the context of knitting robotic arms through the innovative application of the EA strategy and MIEGO method. These challenges include the non-uniqueness and multiple solution characteristics of inverse kinematics, as well as issues related to local optima caused by an excessive focus on the TCP. Our experimental results revealed that the EA strategy effectively guided agents in conducting deep environmental exploration, enhancing the comprehensiveness and safety of path planning. Additionally, our MIEGO method successfully shifted the focus from the singular issue of TCP to multi-objective optimization, effectively avoiding the risk of falling into local optima.

By integrating the EA strategy and MIEGO method, we proposed an improved algorithm (referred to as the EM algorithm) applicable to the widely used DDPG and SAC algorithms in continuous space path planning. The experimental results showed that the EM algorithm not only planned more circuitous paths, steering clear of potential obstacles and reducing safety risks, but also produced more streamlined paths conducive to smooth motor operation, thereby effectively extending the motor's lifespan. This research provides new perspectives and methodologies for path planning for intelligent textile robotic arms, offering robust support for future advancements in intelligent manufacturing and automation technologies. Future research directions include the further optimization of the EA strategy and MIEGO method to suit a broader range of industrial applications, the exploration of integration with other advanced machine learning techniques to enhance the intelligence level of path planning, and the extension of our research methods to other types of robotic arms and automation equipment. Additionally, adaptability to noise and uncertainty in real industrial environments remains an important area for future research.

**Author Contributions:** Conceptualization, D.Z.; Methodology, D.Z.; Software, Z.D.; Validation, Z.D.; Formal analysis, Z.D.; Investigation, W.L.; Resources, D.Z.; Data curation, W.L.; Writing—original draft, S.Z.; Writing—review & editing, D.Z. and S.Z.; Visualization, W.L.; Supervision, Y.D.; Project administration, D.Z.; Funding acquisition, D.Z. and Y.D. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the Tianjin Science and Technology Bureau, grant number 20YDTPJC00740. This research was funded by the Ministry of Education of the People's Republic of China, grant number 220603032241503.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** Data are contained within the article.

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Fu, Q.; Li, Z.; Ding, Z.; Chen, J.; Luo, J.; Wang, Y.; Lu, Y. ED-DQN: An event-driven deep reinforcement learning control method for multi-zone residential buildings. *Build. Environ.* **2023**, *242*, 110546. [CrossRef]
2. Chen, Z.; Xing, M. Upgrading of textile manufacturing based on Industry 4.0. In *Proceedings of the 5th International Conference on Advanced Design and Manufacturing Engineering, Shenzhen, China, 19–20 September 2015*; Atlantis Press: Amstelkade, AV, USA, 2015; pp. 2143–2146.
3. Wang, J.; Xu, C.; Zhang, J.; Zhong, R. Big data analytics for intelligent manufacturing systems: A review. *J. Manuf. Syst.* **2022**, *62*, 738–752. [CrossRef]
4. Sanchez, V.; Walsh, C.J.; Wood, R. Textile technology for soft robotic and autonomous garments. *Adv. Funct. Mater.* **2021**, *31*, 2008278. [CrossRef]
5. Wang, J.; Liang, F.; Zhou, H.; Yang, M.; Wang, Q. Analysis of Position, pose and force decoupling characteristics of a 4-UPS/1-RPS parallel grinding robot. *Symmetry* **2022**, *14*, 825. [CrossRef]
6. She, Q.; Hu, R.; Xu, J.; Liu, M.; Xu, K.; Huang, H. Learning high-DOF reaching-and-grasping via dynamic representation of gripper-object interaction. *ACM Trans. Graf.* **2022**, *41*, 1–14. [CrossRef]
7. Niku, S.B. *Introduction to Robotics: Analysis, Systems, Applications*; Prentice Hall: Englewood Cliffs, NJ, USA, 2001.
8. Hroncová, D.; Miková, Ľ.; Prada, E.; Rákay, R.; Sinčák, P.J.; Merva, T. Forward and inverse robot model kinematics and trajectory planning. In Proceedings of the 2022 20th International Conference on Mechatronics-Mechatronika (ME), Pilsen, Czech Republic, 7–9 December 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 1–9.
9. Zhang, C.; Zhou, L.; Li, Y. Pareto optimal reconfiguration planning and distributed parallel motion control of mobile modular robots. In *IEEE Transactions on Industrial Electronics*; IEEE: Piscataway, NJ, USA, 2023.
10. Qu, J.; Yuan, Q.; Li, Z.; Wang, Z.; Xu, F.; Fan, Q.; Zhang, M.; Qian, X.; Wang, X.; Xu, M.; et al. All-in-one strain-triboelectric sensors based on environment-friendly ionic hydrogel for wearable sensing and underwater soft robotic grasping. *Nano Energy* **2023**, *111*, 108387. [CrossRef]
11. Linh, K.; Cox, J.; Buiyan, T.; Lambrecht, J. All-in-one: A drl-based control switch combining state-of-the-art navigation planners. In Proceedings of the 2022 International Conference on Robotics and Automation (ICRA), Philadelphia, PA, USA, 23–27 May 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 2861–2867.
12. Schulman, J.; Levine, S.; Abbeel, P.; Jordan, M.; Moritz, P. Trust region policy optimization. In Proceedings of the International Conference on Machine Learning, Lille, France, 6–11 July 2015; PMLR: London, UK, 2015; pp. 1889–1897.
13. 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.
14. Mnih, V.; Badia, A.P.; Mirza, M.; Graves, A.; Lillicrap, T.; Harley, T.; Silver, D.; Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016; PMLR: London, UK, 2016; pp. 1928–1937.
15. Andrychowicz, M.; Wolski, F.; Ray, A.; Schneider, J.; Fong, R.; Welinder, P.; McGrew, B.; Tobin, J.; Pieter Abbeel, O.; Zaremba, W. Hindsight experience replay. *arXiv* **2017**, arXiv:1707.01495v1.
16. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal policy optimization algorithms. *arXiv* **2017**, arXiv:1707.06347.
17. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; PMLR: London, UK, 2018; pp. 1861–1870.
18. Fujimoto, S.; Hoof, H.; Meger, D. Addressing function approximation error in actor-critic methods. In Proceedings of the International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018; PMLR: London, UK, 2018; pp. 1587–1596.
19. Reiter, A.; Müller, A.; Gattringer, H. On higher order inverse kinematics methods in time-optimal trajectory planning for kinematically redundant manipulators. *IEEE Trans. Ind. Inform.* **2018**, *14*, 1681–1690. [CrossRef]
20. Morel, G.; Valckenaers, P.; Faure, J.-M.; Pereira, C.E.; Diedrich, C. Manufacturing plant control challenges and issues. *Control. Eng. Pract.* **2007**, *15*, 1321–1331. [CrossRef]
21. Sivanathan, K.; Vinayagam, B.; Samak, T.; Samak, C. Decentralized motion planning for multi-robot navigation using deep reinforcement learning. In Proceedings of the 2020 3rd International Conference on Intelligent Sustainable Systems (ICISS), Thoothukudi, India, 3–5 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 709–716.

22. Dugas, D.; Nieto, J.; Siegwart, R.; Chung, J.J. Navrep: Unsupervised representations for reinforcement learning of robot navigation in dynamic human environments. In Proceedings of the 2021 IEEE International Conference on Robotics and Automation (ICRA), Xi'an, China, 30 May–5 June 2021; IEEE: Piscataway, NJ, USA, 2021; pp. 7829–7835.
23. Karaman, S.; Frazzoli, E. Sampling-based algorithms for optimal motion planning. *Int. J. Rob. Res.* **2011**, *30*, 846–894. [CrossRef]
24. Lee, C. Robot arm kinematics, dynamics, and control. *Computer* **1982**, *15*, 62–80. [CrossRef]
25. Hao, W.G.; Leck, Y.Y.; Hun, L.C. 6-DOF PC-Based Robotic Arm (PC-ROBOARM) with efficient trajectory planning and speed control. In Proceedings of the 2011 4th International Conference on Mechatronics (ICOM), Kuala Lumpur, Malaysia, 17–19 May 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–7.
26. Park, S.-O.; Lee, M.C.; Kim, J. Automation; Systems. Trajectory planning with collision avoidance for redundant robots using jacobian and artificial potential field-based real-time inverse kinematics. *Int. J. Control. Autom. Syst.* **2020**, *18*, 2095–2107. [CrossRef]
27. Huang, J.; Hu, P.; Wu, K.; Zeng, M. Optimal time-jerk trajectory planning for industrial robots. *Mech. Mach. Theory* **2018**, *121*, 530–544. [CrossRef]
28. Ahmed, Z.; Le Roux, N.; Norouzi, M.; Schuurmans, D. Understanding the impact of entropy on policy optimization. In Proceedings of the International Conference on Machine Learning, Long Beach, CA, USA, 10–15 June 2019; PMLR: London, UK, 2019; pp. 151–160.
29. Xu, G.; Meng, Z.; Li, S.; Sun, Y. Collision-free trajectory planning for multi-robot simultaneous motion in preforms weaving. *Robotica* **2022**, *40*, 4218–4237. [CrossRef]
30. Zacharias, F.; Schlette, C.; Schmidt, F.; Borst, C.; Rossmann, J.; Hirzinger, G. Making planned paths look more human-like in humanoid robot manipulation planning. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1192–1198.
31. Zacharias, F.; Sepp, W.; Borst, C.; Hirzinger, G. Using a model of the reachable workspace to position mobile manipulators for 3-d trajectories. In Proceedings of the 2009 9th IEEE-RAS International Conference on Humanoid Robots, Paris, France, 7–9 December 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 55–61.