Engineering Science and Technology, an International Journal 70 (2025) 102147

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

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

Full length article

Preference-based deep reinforcement learning with automatic curriculum learning for map-free UGV navigation in factory-like environments

Shunyu Tian [a], Changyun Wei [a,*], Shaojie Jian [a], Ze Ji [b]

[a] College of Mechanical and Electrical Engineering, Hohai University, Changzhou 213000, China
[b] School of Engineering, Cardiff University, Cardiff CF24 3AA, UK

ARTICLE INFO

Keywords:
UGV navigation
Collision avoidance
Deep reinforcement learning
Automatic curriculum learning
Dynamic obstacles

ABSTRACT

Autonomous navigation for unmanned ground vehicles in smart factory environments involves satisfying multiple, often conflicting objectives such as safety, efficiency, and motion smoothness. Traditional reinforcement learning approaches typically rely on fixed, manually weighted reward functions to encode these objectives. However, such static formulations struggle to generalize across varying user preferences and dynamic operational contexts common in real-world factory scenarios. Consequently, they require retraining for every new preference configuration, leading to inefficiency and limited practical deployment. To address this challenge, we propose a novel preference-based reinforcement learning framework that enables a single policy to dynamically adapt its behavior according to a user-defined preference vector that encodes trade-offs among multiple objectives. This allows the agent to modify its navigation strategy on-the-fly without additional retraining. To further improve training efficiency and learning stability, we incorporate automatic curriculum learning, which gradually increases the complexity of training tasks based on the agent's performance, accelerating convergence and robustness. We validate our method in a simulated smart factory environment that reflects realistic navigation constraints. Experimental results demonstrate that our proposed approach ensures faster convergence during training and achieves up to a 93% navigation success rate in challenging factory-like environments compared to recent advances.

1. Introduction

Autonomous navigation is a fundamental capability for Unmanned Ground Vehicles (UGVs) across a wide range of applications, including disaster response [1], construction [2], and autonomous driving [3]. In recent years, UGVs have been increasingly deployed in unstructured environments such as smart factories [4], airport terminals [5], and commercial facilities [6]. In these settings, UGVs are expected to operate safely and efficiently in collaboration with humans and other mobile systems. Due to the dynamic and interactive nature of such shared spaces, effective navigation involves more than just obstacle avoidance or shortest-path planning. It must simultaneously address multiple, often conflicting, objectives—such as safety, travel efficiency, and motion smoothness. The relative importance of these objectives can vary significantly depending on the task [7,8]. For example, a guide UGV operating in a crowded airport terminal must prioritize safety and smoothness to avoid startling nearby passengers, while a logistics UGV in a warehouse environment emphasizes speed and efficiency to maximize throughput. This diversity of task requirements underscores the necessity for flexible navigation strategies that can dynamically adjust to varying objective priorities. However, designing such adaptive navigation systems remains a significant challenge in the field, particularly in complex and dynamic environments like smart factories, where the trade-offs between objectives are often highly context-dependent. As illustrated in Fig. 1, navigation trajectories generated under different preferences for safety, efficiency, and smoothness vary significantly. This observation highlights the urgent need for a navigation framework capable of dynamically balancing multiple objectives in accordance with human-defined preferences, thereby enabling UGVs to meet the nuanced demands of diverse operational contexts.

Previous research on autonomous navigation for UGVs has often failed to adequately account for the dynamic changes in multiple conflicting objectives—such as safety, efficiency, and smoothness—across varying tasks. Traditional studies have largely focused on single-objective optimization, primarily aimed at ensuring safety in static environments [9,10]. In recent years, however, growing demand for navigation in dynamic and human-shared environments has shifted the research focus toward multi-objective navigation problems. These involve a broader set of objectives, including safety, efficiency, trajectory

* Corresponding author.
E-mail addresses: 221319010023@hhu.edu.cn (S. Tian), c.wei@hhu.edu.cn (C. Wei), 231319010004@hhu.edu.cn (S. Jian), jiz1@cardiff.ac.uk (Z. Ji).

https://doi.org/10.1016/j.jestch.2025.102147
Received 9 December 2024; Received in revised form 14 June 2025; Accepted 18 July 2025
Available online 12 August 2025

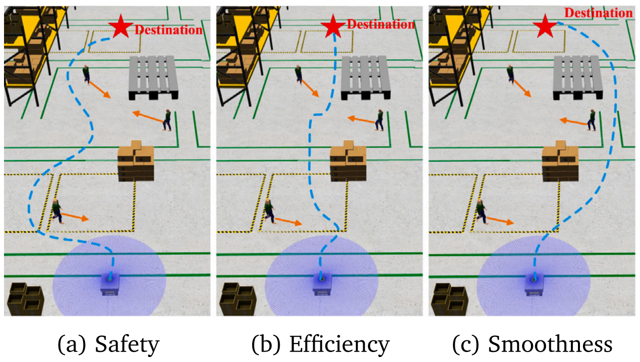


Fig. 1. Sensor-level UGV navigation with various preference-based objectives.

smoothness, continuity, and even energy consumption [11,12]. Unlike single-objective optimization, multi-objective navigation requires balancing trade-offs among multiple goals, often resulting in a set of Pareto-optimal solutions. Preference-based multi-objective navigation introduces a preference vector to encode the relative importance of different objectives, enabling the navigation strategy to adapt dynamically as task requirements evolve. Most existing multi-objective optimization methods rely heavily on expert knowledge to define a fixed preference vector with static weights assigned to each objective [13,14]. This approach has the advantage of transforming the multi-objective problem into a single-objective formulation, making it compatible with conventional reinforcement learning frameworks. However, the use of fixed preference vectors presents notable limitations. First, designing an appropriate preference vector depends on expert insight, which is often difficult to obtain in complex scenarios. As a result, extensive exploration is typically required to refine the preferences, leading to high pre-training costs. Second, static preferences yield only a single solution tailored to a specific task, and the model must be retrained whenever the relative importance of the objectives changes. Therefore, current multi-objective methods for UGV autonomous navigation exhibit limited adaptability in dynamic and evolving environments, highlighting the need for more flexible and adaptive solutions.

To address the challenges outlined above, this paper focuses on dynamic and unstructured factory-like environments. We investigate how UGVs can effectively balance multiple objectives — such as safety, efficiency, and smoothness — based on human preferences that vary across tasks and scenarios. The overall goal is to enable reliable navigation and obstacle avoidance under changing preference conditions. Specifically, the UGV must avoid obstacles in complex environments and reach its goal in a manner consistent with the user's expectations. To this end, we propose a novel Multi-Objective Reinforcement Learning (MORL) framework that directly incorporates human preferences into the learning process. The approach embeds preference information into the state representation and scalarizes the multi-objective reward functions using different preference vectors. These scalarized rewards are then used to iteratively update the policy and value networks during training. This allows the agent to learn policies that yield near-Pareto-optimal performance across a range of preferences. A key advantage of the proposed method lies in its strong generalization capability. Unlike traditional approaches that require training a separate model for each preference setting, our method learns a unified policy network that covers the entire preference space through a single training process. As a result, the UGV can flexibly adapt to diverse and changing human preferences without retraining. The main contributions of this paper are outlined as follows.

- We propose a MORL-based approach that enables a UGV to dynamically adjust its autonomous navigation policy according to human preferences over safety, efficiency, and smoothness. Notably, the proposed method generates a set of solutions that span the entire preference space within a single training process.
- To improve the training efficiency of the MORL approach, we introduce a novel Automatic Curriculum Learning (ACL) algorithm. This algorithm assesses the difficulty of multiple target points using the UGVs's current policy and autonomously selects tasks most suitable for the current learning stage. The goal is to expedite the learning of complex strategies by transitioning from simpler to more challenging tasks.
- The proposed approach has been validated across a range of factory-like scenarios, demonstrating impressive performance compared to baseline methods across a comprehensive range of evaluation metrics, including success rate, collision rate, timeout rate, path length, run-time and average curvature.

The subsequent sections of this paper are organized as follows. Section 2 provides an overview of related work, and Section 3 presents the formalized representation of the MORL problem. Section 4 introduces the detailed design of our navigation approach, covering aspects of the state space, reward function, and automatic curriculum learning algorithms Section 5 conducts algorithm testing in two representative simulation environments to demonstrate the effectiveness of the proposed approach, along with an analysis and discussion of the experiment results. Finally, Section 6 concludes this work.

2. Related work

2.1. UGV navigation in factory-like environments

UGV navigation methods are typically classified into two main categories: map-based and map-free navigation. Map-based approaches often rely on Simultaneous Localization and Mapping (SLAM) [15], where UGVs construct maps using sensor data and utilize classical planning algorithms to generate feasible paths. While effective in static or structured settings, these methods face significant challenges in dynamic environments, such as those involving pedestrian traffic. In particular, SLAM-based methods struggle to maintain accurate maps in rapidly changing scenes due to occlusions, sensor noise, and moving obstacles, and the computational burden of frequent map updates further limits real-time applicability.

To overcome these issues, map-free navigation methods have become increasingly prevalent. These approaches typically fall into two categories: rule-based models and learning-based techniques. Rule-based algorithms, such as Velocity Obstacle (VO) [16], Reciprocal Velocity Obstacle (RVO) [17], and Optimal Reciprocal Collision Avoidance (ORCA) [18], have been widely used in autonomous UGV navigation and extended to multi-agent systems. However, VO assumes perfect knowledge of obstacle velocities, which is rarely feasible in practice. RVO introduces reciprocal interaction models but often leads to oscillatory or unnatural behaviors in dense crowds due to violations of mutual cooperation assumptions. ORCA improves multi-agent scalability but incurs high computational cost and lacks learning capabilities, limiting its adaptability to unfamiliar scenarios. Extensions to these models, such as [19], have incorporated UGV heterogeneity to reduce freezing issues, yet they still rely on hand-tuned parameters and require explicit knowledge of the relative geometry and motion of obstacles—conditions that are difficult to guarantee in real-world deployments.

In contrast to rule-based methods, DRL has emerged as a powerful framework for UGV navigation in complex and uncertain environments [20]. DRL methods learn policies directly from raw sensor data through trial-and-error interaction with the environment [21]. For example, Zhu et al. [22] propose a DRL framework using oriented bounding capsules to model agents. However, the learned policy focuses only on safety and smooth trajectories, and is not suitable for scenarios where task efficiency is also critical. Wang et al. [23] introduce a curriculum-based two-stage learning strategy to address UGV navigation in environments with movable obstacles. While their method improves training efficiency, it still lacks a comprehensive balance among safety, efficiency, and trajectory smoothness during navigation.

In factory-like environments, UGV navigation often requires a delicate balance among multiple objectives, including safety, efficiency, and trajectory smoothness. In recent years, several studies have explicitly incorporated these multi-objective considerations into algorithmic designs to enhance overall navigation performance. For example, Nishimura et al. [24] propose a balanced learning framework that aims to trade off safety and operational efficiency. However, their rule-based switching mechanism lacks adaptability to continuous environmental changes and reduces robustness in dynamic contexts. Similarly, Jain et al. [25] introduce a human–robot collaboration strategy in which the robot autonomously complies with predefined behavioral norms related to human comfort and safety. While this approach enhances human–robot interaction, it lacks adaptability to variations in user preferences or situational contexts. Ravankar et al. [26] present a trajectory generation algorithm that promotes both safety and trajectory smoothness by maintaining safe distances from pedestrians. Although effective in semi-structured environments such as vineyards, this method shows limited generalization in highly dynamic, obstacle-dense indoor settings. Zhu et al. [27] propose a hierarchical DRL framework that enables coordinated optimization of safety and efficiency. Nevertheless, the hierarchical architecture introduces switching delays and increases training complexity.

In summary, although existing methods address objectives like safety, efficiency, and smoothness, they often rely on static weights and rule-based switching, limiting adaptability to dynamic tasks and user preferences. In real-world settings, the importance of these objectives varies across contexts, but current approaches lack the flexibility to adjust accordingly, leading to suboptimal performance. Therefore, developing navigation strategies that can flexibly and autonomously adapt to varying task requirements through preference-aware learning remains an open and pressing challenge—serving as the primary motivation of this paper.

2.2. *Multi-objective reinforcement learning*

MORL seeks to optimize multiple competing objectives simultaneously by extending the scalar reward to a vector form, enabling balanced control across objectives. Recently, MORL has gained significant attention in domains like robotic control [28,29]. Existing approaches fall mainly into two categories: reward aggregation methods and Pareto optimization methods.

Reward aggregation methods combine multiple objective rewards into a single scalar — often via weighted sums — allowing use of standard reinforcement learning algorithms [13,30,31]. For example, He et al. [32] propose a multi-objective Deep Q-Network (DQN) variant incorporating composite rewards and multi-agent games, while Chraibi et al. [33] extend Q-values to vectors for joint optimization of task time and energy in cloud scheduling. These methods are simple and computationally efficient, suitable for low-dimensional, stable-preference tasks. However, they rely heavily on fixed weights, limiting adaptability to dynamic user preferences or task requirements, which often vary in real scenarios—resulting in reduced policy flexibility and robustness.

In contrast, Pareto optimization methods treat each objective independently and aim to find a Pareto-optimal set, where no objective can improve without worsening another [34,35]. Typically structured with an outer loop for policy selection and an inner loop for policy training, such methods better capture objective trade-offs [36]. For instance, Xu et al. [13] use evolutionary learning guided by predictive modeling to approach the Pareto front in continuous robot control, while Zhang et al. [37] develop a continuous Reinforcement Learning (RL) framework for online multi-objective adaptation. Despite their theoretical advantages, Pareto methods face scalability challenges due to evaluating multiple policies and high computational costs, limiting practical use in high-dimensional objective spaces [28,38,39].

To overcome these limitations, we propose a novel Pareto-based MORL method that integrates human preference vectors into state representations. By continuously sampling diverse preference combinations and training a unified policy network, the approach learns an approximate Pareto frontier covering a broad preference space in a single process. This method combines Pareto optimization's objective independence with DRL's adaptability, achieving enhanced generalization and practical performance in dynamic multi-objective settings.

2.3. *Automatic curriculum learning*

ACL enables an agent to autonomously organize and master a sequence of tasks from simple to complex during training [40]. By first accumulating experience on basic tasks, the agent gradually transfers and generalizes knowledge to more challenging ones, improving learning efficiency and policy quality [41].

Within the domain of UGV navigation, most ACL methods adapt either subtask decomposition or two-stage training [42–44]. The subtask decomposition approach incrementally raises task difficulty within a single environment, allowing progressive learning across subtasks of increasing complexity. For example, Matiisen et al. [45] train an agent in the Minecraft environment by sequentially transitioning from avoiding lava to navigating narrow passages, effectively demonstrating progressive policy learning. Alternatively, the two-stage method employs multiple environments with escalating difficulty to progressively enhance navigation skills. Song et al. [46] use a three-stage curriculum in a racing simulator, achieving overtaking performance comparable to experienced human drivers.

These methods generate tasks of moderate difficulty for policy training, avoiding the formation of distorted policies due to the imbalance between positive and negative samples in reinforcement learning. However, the sorting of curriculum difficulty levels in these ACL methods typically requires human expert intervention. Discrepancies between the difficulty levels assigned by human experts and generated automatically can significantly impact algorithm convergence. To overcome this, we propose an evaluation network that enables the agent to autonomously estimate task difficulty and select medium-difficulty tasks during training. This approach reduces reliance on manual intervention and enhances the adaptability and generalization of learned policies.

3. Problem formulation

This section will present our MORL framework proposed to address the challenge of learning multi-objective strategies for UGVs navigating through dynamic, unknown, and human-preference-varying pedestrian environments. Initially, we formulate the UGV's sequential decision-making problem as a Multi-Objective Markov Decision Process (MOMDP). Following this, we conduct a theoretical analysis of the MORL framework based on human preferences. Furthermore, we present the Bellman equation for the multi-objective Q-function, which serves to effectively tackle the multi-objective navigation problem.

3.1. *Multi-objective Markov decision process*

The MOMDP consists of a 7-tuple $(\mathcal{S}, \mathcal{A}, \boldsymbol{P}, \mathcal{R}, \gamma, \boldsymbol{\Omega}, f_{\Omega})$, which includes the state space $\mathcal{S}$, action space $\mathcal{A}$, state transition probability matrix $\boldsymbol{P}$, vector reward function $\mathcal{R}$, discount factor $\gamma \in [0,1]$, human preference space $\boldsymbol{\Omega}$, and preference-guided scalarization function f_{Ω}. In contrast to MDP, MOMDP incorporates the human preference space and preference-guided scalarization function as additional dimensions for considering and trading off multiple objectives. More importantly, in the MOMDP framework, reward values are expressed in vector form, representing distinct returns associated with multiple objectives. The vector reward value is expressed as $\boldsymbol{r}(s,a) = [r_1, r_2, \ldots, r_m | S_t = s, A_t =$

$a]^{\mathrm{T}}$, where m represents the number of objectives. To denote the reward value in the current state, the vector reward function is denoted as $\boldsymbol{R}_\pi(s) = \mathbb{E}_\pi[\boldsymbol{r}_{t+1}|S_t = s]$. The preference space represents the diverse human preferences towards a particular solution, while concurrently addressing multiple objectives that need to be achieved in robotic navigation. The preference-guided scalarization function maps the m-dimensional human preferences and the vector reward function to a scalar value.

Unlike traditional weighted approaches, our proposed scalarization function assigns dynamic weights to various objectives across the human preference space. This method encourages the UGV to explore strategies under different weight combinations in multi-objective tasks. As a result, the UGV can adapt to shifts in objective weights or preferences in dynamic environments, allowing it to develop navigation strategies that are tailored to diverse configurations of weights. This flexibility enables the UGV to effectively address challenges in real-world scenarios that evolve over time. In our study, the preference-guided scalarization function is defined as:

$$f_{\boldsymbol{\omega}}(\boldsymbol{r}(s,a)) = \boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{r}(s,a) \tag{1}$$

where $\boldsymbol{\omega} \in \boldsymbol{\Omega}$, $\boldsymbol{\omega} = [\omega_1, \omega_2, \ldots, \omega_m]^{\mathrm{T}}$ represents a specific human preference and satisfies $\sum_{i=1}^{m} \omega_i = 1$. $\boldsymbol{r}(s,a)$ denotes the multi-objective vector reward associated with a specific state and action.

3.2. Multi-objective reinforcement learning

The MORL model can be constructed based on multi-objective Markov chains. Similar to single-objective RL, where rewards are incrementally accumulated to define the return, multi-objective returns can be expressed as follows:

$$\boldsymbol{G}_t := \sum_{k=0}^{+\infty} \gamma^k \boldsymbol{r}_{t+k+1} \tag{2}$$

where $\boldsymbol{G}_t$ denotes vector return with m-dimensional human preferences. Let the policy $\pi : \mathcal{S} \times \Omega \rightarrow \mathcal{A}$ be a stationary policy conditioned on a preference vector $\omega \in \Omega$. The corresponding scalarized state-value function is defined as:

$$V^\pi(s,\omega) := \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t f_\omega\left(\boldsymbol{r}(s_t,a_t)\right) \,\middle|\, S_0 = s \right], \tag{3}$$

where $f_\omega(\boldsymbol{r}) = \omega^\top \boldsymbol{r}$ is the scalarization of the vector reward $\boldsymbol{r}(s_t,a_t) \in \mathbb{R}^m$. The corresponding scalarized action-value function is given by:

$$Q^\pi(s,a,\omega) := \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t f_\omega\left(\boldsymbol{r}(s_t,a_t)\right) \,\middle|\, S_0 = s,\ A_0 = a \right]. \tag{4}$$

The optimal Q-function satisfies the Bellman optimality equation:

$$Q^*(s,a,\omega) = f_\omega(\boldsymbol{r}(s,a)) + \gamma \sum_{s'} P(s' \mid s,a) \max_{a'} Q^*(s',a',\omega), \tag{5}$$

and the corresponding greedy policy is defined as:

$$\pi^*(s,\omega) = \arg\max_{a\in\mathcal{A}} Q^*(s,a,\omega). \tag{6}$$

In order to meet the multi-objective optimization of UGV autonomous navigation, it is necessary to define and verify the Pareto optimality and frontier coverage of MORL. Let the vector-valued return under policy π be:

$$\boldsymbol{R}^\pi(s) := \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t \boldsymbol{r}(s_t,a_t) \,\middle|\, s_0 = s \right]. \tag{7}$$

A policy π is said to be Pareto-optimal if there does not exist another policy π' such that:

$$\boldsymbol{R}^{\pi'}(s) \succeq \boldsymbol{R}^\pi(s) \quad \text{and} \quad \boldsymbol{R}^{\pi'}(s) \neq \boldsymbol{R}^\pi(s), \tag{8}$$

where $\succeq$ denotes component-wise dominance (i.e., no objective is worse, and at least one is better). The set of all such non-dominated policies at state s forms the Pareto front, denoted by $\mathcal{P}(s)$.

Under the assumption that the image of the reward function is convex, it is known that for any point on the convex hull of the Pareto front, there exists a weight vector $\omega \in \Omega$ such that the corresponding scalarized policy is optimal:

$$\pi^*(\omega) = \arg\max_\pi\ \omega^\top \boldsymbol{R}^\pi(s). \tag{9}$$

By uniformly sampling ω from Ω and learning a unified policy $\pi(s,\omega)$, MORL framework aims to approximate the entire convex coverage of the Pareto front:

$$\pi(s,\omega) \approx \pi^*(s,\omega), \quad \forall\ \omega \in \Omega. \tag{10}$$

In order to enable the MORL method to obtain a set of Pareto frontiers, it is necessary to prove its convergence. For each fixed $\omega \in \Omega$, the scalarized Q-function is updated using temporal difference methods. Define the Bellman operator as:

$$\mathcal{T}_\omega Q(s,a) := f_\omega(\boldsymbol{r}(s,a)) + \gamma \sum_{s'} P(s' \mid s,a) \max_{a'} Q(s',a',\omega). \tag{11}$$

This operator is a γ-contraction with respect to the supremum norm:

$$\|\mathcal{T}_\omega Q_1 - \mathcal{T}_\omega Q_2\|_\infty \leq \gamma \|Q_1 - Q_2\|_\infty, \tag{12}$$

which guarantees convergence of $Q_k \rightarrow Q^*$ as $k \rightarrow \infty$. Assuming sufficient coverage and representational capacity of the function approximator (e.g., deep networks), the approximation $Q(s,a,\omega) \approx Q^*(s,a,\omega)$ holds uniformly across $\omega \in \Omega$.

By conditioning both actor and critic networks on ω, and sampling ω during training, the learned policy $\pi(s,\omega)$ becomes a preference-adaptive universal policy that approximates the optimal decision behavior for any user-defined trade-off vector. Therefore, under the following assumptions:

- The reward space is bounded and convex;
- The scalarization function $f_\omega(\cdot)$ is linear and normalized;
- The preference vector ω is sampled from a dense support over Ω;
- The deep networks have sufficient capacity;

we conclude that the proposed MORL framework is theoretically sound and converges to a set of policies that approximate the convex portion of the Pareto-optimal solution set.

4. Preference-based navigation approach

In this section, we will present a novel map-free preference-based navigation approach, named Safe-Efficient-Smooth Navigation (SESN), designed to address the multi-objective challenges arising from diverse task requirements. To the best of our knowledge, enabling agents to accomplish multi-objective autonomous navigation based on distinct human preferences poses significant challenges. Therefore, we establish a DRL model based on multi-objective evaluation methods to achieve end-to-end UGV navigation, spanning from sensor-level environment perception to action level speed control. Additionally, we build an automatic curriculum learning algorithm to enhance the learning efficiency of the agent. Unlike conventional manual design methods, the sequence of tasks for multi-objective learning is automatically generated.

4.1. Overall design of the SESN architecture

The proposed SESN employs a multi-objective DRL framework to formulate navigation strategies for UGVs adaptable to diverse human preferences. Specifically, our SESN considers safety, efficiency, and smoothness as three key criteria for UGV navigation, aiming to identify optimal strategies customized to potential human preferences. Moreover, our SESN also incorporates ACL mechanisms to assess the complexity of distinct target locations and, consequently, select targets of suitable challenge levels to expedite the training process. Fig. 2 illustrates the overall architecture of the proposed SESN approach.

The overall architecture includes the following components.

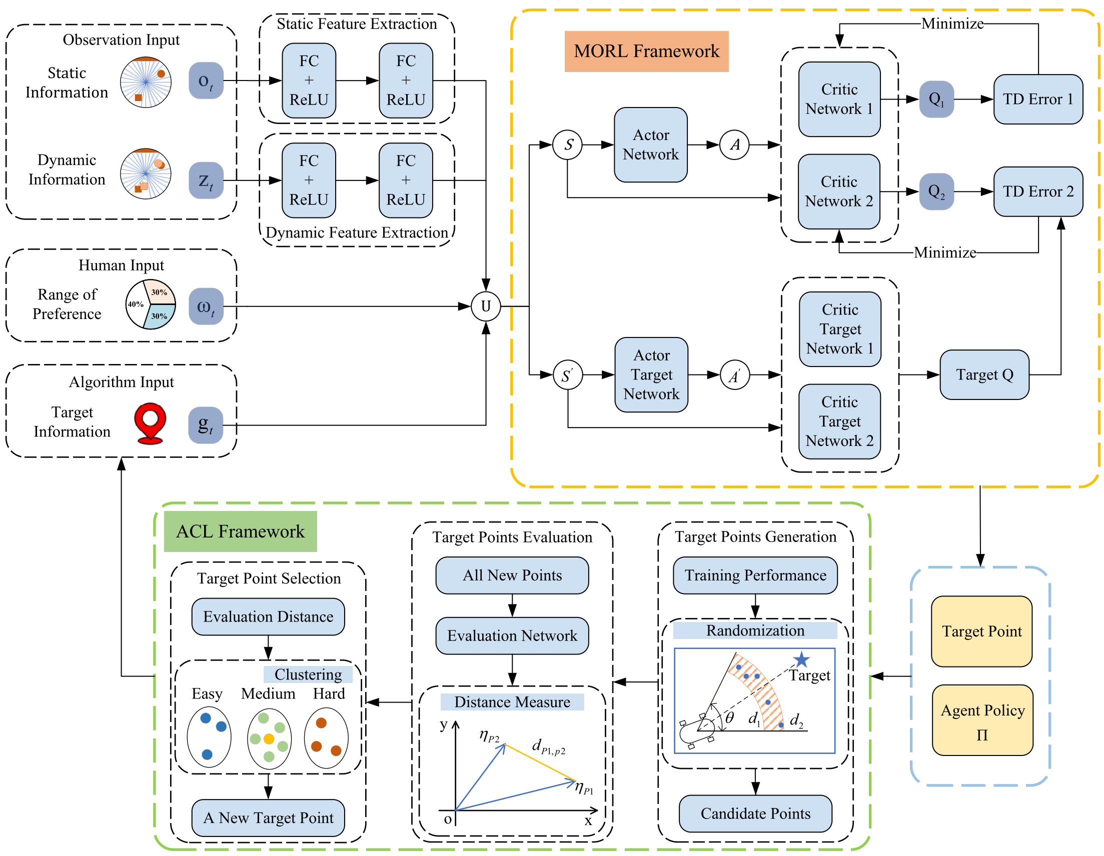


Fig. 2. Illustration of our preference-based SESN architecture.

1. **Sensor-Level Environmental Perception** (as depicted in the left side of Fig. 2): the agent employs a 2-D laser radar to capture environmental information. Then, the sensor-level data is processed by a feature extraction network, which can generate environmental representations of static and dynamic features.
2. **Preference-Based MORL**: the agent utilizes both static and dynamic extracted features of the environment, along with pre-specified target locations and human preferences, as inputs for the reinforcement learning process. Unlike traditional DRL, our approach introduces stochastic preferences within the state space, using these preferences as reward weights for various objectives. This innovative design facilitates the learning of navigation strategies that accommodate a wide range of preferences. By allowing the policy network to thoroughly explore the entire preference space during training, the approach facilitates the generation of a diverse set of navigation strategies that are tailored to different preferences. This flexibility enhances the UGV's ability to adapt to varying objectives in dynamic environments.
3. **The ACL Framework**: this component encompasses the generation, evaluation, and selection of the target locations. The policy learned from navigating to the current target location is fed into the ACL network, which then selects the most suitable next target location for subsequent stage training.

In addition, we have designed a multi-objective network based on the Twin Delayed Deep Deterministic Policy Gradient (TD3) [47] to learn policies tailored to various human preferences. Our proposed SESN approach is a generic method, indicating that the TD3 algorithm can be substituted with other frameworks such as Soft Actor–Critic (SAC) or Asynchronous Advantage Actor–Critic (A3C), thereby demonstrating the versatility of the proposed method.

4.2. Sensor-level state space representation

To address the navigational requirements related to safety, efficiency, and smoothness, the accurate representation of the UGV's state at each time step is crucial. A clear and efficient state representation can accelerate model convergence during the RL training process. In our proposed SESN approach, at each time step t, the UGV's state is defined as a combination of sensor-level environmental perception, target information, and human preferences.

Sensor-level environmental perception refers to the capacity of a UGV to acquire environmental information through its onboard sensors and subsequently analyze the acquired information. As the UGV advances toward its destination, it is crucial for it to acquire local spatial information within its sensing range, which includes both dynamic and static obstacles. Furthermore, the capability to comprehend the relative motion of dynamic obstacles in close proximity plays a pivotal role in facilitating trajectory selection. Hence, we consider both obstacle positions and relative motion information for environmental perception. Specifically, the UGV's environmental perception is defined as $e_t = (o_t, z_t)$. Positional observations o_t are measured through a two-dimensional laser radar with a field of view. Relative motion observations z_t represent the differences between the current laser radar observation and previous one, denoted as $z_t = o_t - o_{t-1}$.

With the maximum sensing range, the UGV can perceive whether a dynamic obstacle is moving away from it or approaching it based on the positional changes between continuous time steps. Therefore, by concurrently considering both positional and relative motion information, the UGV can identify the dynamic variations in the surrounding obstacles, thereby contributing to the extraction of features from the dynamic environment.

The target information g_t depends on the destination of various UGV navigation tasks. Target information g_t consists of two components: one is the relative distance d_t between the target position and the current location, and the other is the relative angle θ_t between the target direction and the current velocity. Based on the UGV current location (x_t, y_t) in Cartesian coordinates, we use d_t and θ_t to denote the relative distance and angle between the UGV and the target position, and v_t and ϕ_t to represent the linear and angular velocities, respectively. In order to ensure motion stability, we also consider that the linear and angular velocities should adhere to the UGV's dynamic constraints by $v_t \in [0, v_{tmax}]$ and $\phi_t \in (-\phi_{tmax}, \phi_{tmax})$.

The determination of human preferences relies on a comprehensive consideration of three key factors: safety, efficiency, and smoothness. Preferences remain consistent within the same task but may vary across different tasks due to specific requirements. For example, in tasks that prioritize safety, the navigation strategy may adopt a more conservative path to avoid potential collisions. Conversely, in scenarios emphasizing efficiency, the UGV may select a more direct route to minimize task completion time. The preference values assigned to different tasks influence the UGV's navigation strategy through a preference-guided scalar function during the training process. Consequently, once the MORL model is trained and deployed on the UGV, it can utilize these preference values within the state space to inform its various navigation strategies. In this manner, preferences effectively guide the navigation strategies, enabling the UGV to better adapt to complex environments and diverse human needs.

To ensure the rationality of preferences, we establish preference ranges for the three factors based on the reward function configuration. Specifically, the preference range for safety is set at $[0.45, 0.55]$, aiming to prioritize the avoidance of collisions and potential hazards during task execution. The preference range for efficiency is defined as $[0.35, 0.45]$, encouraging the UGV to complete tasks efficiently in the shortest possible time, thereby enhancing overall performance. The preference range for smoothness is set between $[0, 0.2]$ to ensure the stability of the UGV's movements, avoiding abrupt turns and stops.

4.3. Preference-based reward function

In multi-objective UGV navigation tasks, we need to consider a wider range of performance criteria compared to standard navigation tasks. It usually includes safety, efficiency, and trajectory smoothness. The rationale for selecting these metrics stems from their ability to comprehensively evaluate the key performance aspects of UGV navigation [48,49]. Safety is crucial as it ensures that the UGV can effectively avoid collisions and potential hazards in both static and dynamic environments, thereby safeguarding the UGV and surrounding objects. Efficiency assesses the UGV's ability to complete tasks in the shortest possible time using optimal pathways, which helps save both energy and time. Lastly, trajectory smoothness ensures fluid movement by minimizing abrupt turns and the effects of inertia, thereby enhancing control precision and operational stability. To address different navigation scenarios based on human preferences, we have designed independent reward functions for each objective as:

$$r(t) = [r_s(t), r_e(t), r_m(t)]^{\mathrm{T}} \tag{13}$$

Here, $r_s(t)$ represents safety, imposing penalties in situations where the UGV approaches obstacles or exceeds speed limits. This mechanism ensures safe navigation by applying strong penalties to discourage collisions and hazardous conditions. The term $r_e(t)$ optimizes efficiency by minimizing travel time or path length, encouraging the UGV to select optimal routes through appropriate speed adjustments. Lastly, $r_m(t)$ focuses on trajectory smoothness, penalizing sharp turns or sudden accelerations to promote fluid, natural movement.

The reward function for safety is formulated as $r_s = r_c + r_{vl}$. The first component r_c represents the penalty associated with the proximity and potential collision between the UGV and environmental obstacles. The second component r_{vl} pertains to the penalty incurred when the UGV exceeds the limited maximum velocity within a given operational environment. The calculation for r_c and r_{vl} are expressed as follows:

$$r_c = \begin{cases} -10, & \text{if } l_{min} \le d_c \\ -10(\frac{d_s - l_{min}}{d_s - d_c})^2, & \text{if } d_c < l_{min} \le d_s \\ 0, & \text{if } d_s < l_{min} \end{cases} \tag{14}$$

$$r_{vl} = \begin{cases} 0, & \text{if } v \le v_{max} \\ -10, & \text{if } v_{max} < v \end{cases} \tag{15}$$

where l_{min} represents the minimum 2-D LiDAR value, indicating the closest distance between the UGV and an obstacle. d_c signifies the minimum safe distance between the UGV and obstacles, while d_s is the minimum distance that ensures human comfort. v_{max} represents the maximum permissible speed within the current environmental context. Specifically, penalties are applied when $l_{min} \le d_c$ indicating potential UGV-obstacle collisions. Proportional penalties are also imposed when the UGV intrudes upon the social distance maintained by humans, emphasizing the importance of a harmonious human–robot interaction. To ensure the safety of both the UGV and humans, any instance of the UGV exceeding the maximum permissible speed defined by the environment will result in penalties.

We define r_e as an efficiency-driven reward function, expressed as $r_e = r_p + r_g + r_{ve}$. This composite function includes three components. r_p represents the living cost assigned to the UGV at each time step, and in this work, we have designed $r_p = -1$ at each time step. Comparatively, r_g is used to indicate whether the UGV is navigating closer to the target location,

$$r_g(t) = \gamma(d_{rg}(t-1) - d_{rg}(t)) \tag{16}$$

where γ serves as a positive parameter linked to the duration of each time step, and $d_{rg}(t)$ denotes the Euclidean distance between the UGV and its target location at time t. In order to address the efficiency, we use r_{ve} to denote a penalty to regulate the UGV's speed with respect to a minimum threshold $v_{\min}$,

$$r_{ve} = \begin{cases} 0, & \text{if } v_{min} < v \\ -5, & \text{if } v \le v_{min} \end{cases} \tag{17}$$

Moreover, we define the reward function r_m to evaluate the smoothness of UGV navigation. It consists of two components: r_{mv} and $r_{m\phi}$, corresponding to rewards for linear velocity variation and angular velocity variation, respectively. To enhance navigation smoothness in specific scenarios, penalties are imposed when the variations in linear velocity and angular velocity exceed predetermined thresholds, as follows:

$$r_{mv} = \begin{cases} 0, & \text{if } |v(t) - v(t-1)| \le v_c \\ -2, & \text{if } v_c < |v(t) - v(t-1)| \end{cases} \tag{18}$$

$$r_{m\phi} = \begin{cases} 0, & \text{if } |\phi(t) - \phi(t-1)| \le \phi_c \\ -2, & \text{if } \phi_c < |\phi(t) - \phi(t-1)| \end{cases} \tag{19}$$

Here, $v(t)$ and $\phi(t)$ represent the current linear velocity and angular velocity of the UGV, respectively. Parameters v_c and ϕ_c denote the maximum allowable changes in linear velocity and angular velocity that the UGV can endure while maintaining a smooth trajectory.

In summary, the preference-guided scalarization function for multi-objective UGV navigation tasks is expressed as:

$$f_{\boldsymbol{\omega}}(\boldsymbol{r}(s,a)) = \boldsymbol{\omega}_{sem}^{\mathrm{T}} \cdot [r_s, r_e, r_m]^{\mathrm{T}} \tag{20}$$

where $\boldsymbol{\omega}_{sem}$ represents the preference weights assigned to the three objectives: safety, efficiency, and smoothness. The vector $[r_s, r_e, r_m]$ contains the corresponding reward values obtained during training.

4.4. Automatic curriculum learning algorithm

Inspired by the observation that humans tend to learn more rapidly when presented with moderately challenging tasks, our aim is to empower UGVs to autonomously select new tasks of moderate difficulty

Algorithm 1 Automatic curriculum learning (ACL) for generating, evaluating and selecting appropriate target locations.

```
1: Initialize actor network $\pi_\phi$, target actor network $\pi'_\phi$, critic network $Q_\theta$, target critic network $Q'_\theta$, human preference space $\Omega$ and replay buffer $B$.
2: Set final target location $p_t$, initial target location $p_s$, current target location $g_t$.
3: while training episodes $\leq$ preset episodes do :
4:     Acquire sensor observations $o_t$, $z_t$ and set human preference $\omega_t$;
5:     if $g_t$ == None then:
6:         $g_t = p_s$;
7:     end if;
8:     if the number of success in 10 consecutive training episodes $\geq$ 8 then:
9:         if $\|g_t - p_t\|_2 \leq 10$ then:
10:            $g_t = p_t$;
11:        else
12:            Generate the set of potential target location points $p_1, p_2, \ldots, p_K$;
13:            for $i = 1$ to $K$ do:
14:                Choose the $i$-th potential target point as the current target point;
15:                for $j = 1$ to number of tests $M$ do:
16:                    Load current target actor network $\pi'_{\phi t}$;
17:                    Obtain performance data for a single round of testing;
18:                end for;
19:                Integrate performance data for the same target point to form a performance metric vector $\eta_{pi}$;
20:                Normalize performance metrics of all potential points as $\eta^*_{pi}$;
21:            end for;
22:            Calculate Euclidean distance $d_{p1,pj}$ between evaluation vectors of all potential target points and the current one;
23:            According to difficulty level, select the target point with moderate difficulty as the next training target point $g_t$;
24:        end if;
25:    end if;
26:    Select action with noise $a \sim \pi_\phi(s,\omega) + \epsilon$, $\epsilon \sim N(0,\sigma)$ and store transition tuple $(s,a,r,\omega,s')$ in $B$;
27:    Sample mini-batch of N transition $(s,a,r,\omega,s')$ from $B$;
28:    $\tilde{a} \leftarrow \pi_{\phi'}(s',\omega) + \epsilon, \epsilon \sim clip(N(0,\tilde{\sigma}), -c, c)$;
29:    $y \leftarrow r + \gamma \min_{i=1,2} Q_{\theta'_i}(s', \tilde{a}, \omega)$;
30:    Update critic network $\theta_i \leftarrow \arg\min \theta_i N^{-1} \sum (y - Q_{\theta_i}(s,a,\omega))^2$;
31:    Update actor network $\phi$ by the deterministic policy gradient $\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_1}(s,a,\omega)|_{a=\pi_\phi(s,\omega)} \nabla_\phi \pi_\phi(s,\omega)$;
32:    Update target networks $\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$, $\phi'_k \leftarrow \tau\phi_k + (1-\tau)\phi'_k$;
33: end while.
```

during the training process. To achieve the automatic selection of target locations in learning the navigation policies, we have devised the ACL algorithm consisting of three modules: target location generation, target location evaluation and target location selection. The pseudocode for the ACL algorithm is depicted in Algorithm 1.

4.4.1. Target location generation

To facilitate the gradual learning of navigation strategies by an agent, we have pre-constructed a specialized training map in advance. This map features a gradual increase in obstacles and pedestrian density from the bottom left corner to the top right corner. Such preprocessing not only confines the range of target point generation effectively but also provides robust support to the agent's steady progress towards the goal.

Before proceeding to generate candidate target points for the next stage, we also need to establish the triggering mechanism for the subsequent phase of curriculum learning. In ACL, our aim is to achieve a delicate equilibrium between learning efficiency and knowledge acquisition. Thus, we have chosen a straightforward yet effective strategy: initiating the transition to the subsequent phase of curriculum learning once successful navigation has been achieved in 8 out of 10 consecutive attempts. In our work, the ultimate navigation objective for the ACL mechanism is defined as $p_t(x_t, y_t)$, which reflects the intricate configuration of our specialized map. Following the completion of the curriculum phase with the current target point $g_t(x_c, y_c)$, we can generate a set of candidate target points $\boldsymbol{P} = \{p_1, p_2, \ldots, p_K\}$, where K denotes the total number of candidate points derived per iteration. Here, the selection criterion for the ith candidate target point adheres to the following guidelines:

$$\begin{cases} x_i = x_c + d \times \cos(\arctan \frac{y_t - y_c}{x_t - x_c} + \theta) \\ y_i = y_c + d \times \sin(\arctan \frac{y_t - y_c}{x_t - x_c} + \theta) \end{cases} \tag{21}$$

Here, both d and θ are generated through random sampling, where $d \in [5, 10]$ and $\theta \in [-\frac{\pi}{4}, \frac{\pi}{4}]$. Furthermore, we validate these generated points to ensure their compliance with the boundaries of the map. By employing this method of generating target points, we are able to exert independently control over the difficulty spans between curriculum stages by adjusting d and θ.

4.4.2. Target location evaluation

To evaluate the difficulty level of new tasks under the current learning strategy $\pi_{\tau 1}$, we employ a comprehensive evaluation strategy considering four performance metrics: success rate, average path length, average time, and average curvature. Subsequently, we test the learned strategy on all generated target points $(p_1, p_2, \ldots, p_k)$, where p_1 denotes the target point for the current completed learning strategy, while p_2 to p_k represent candidate target points. Through experimental testing, we obtain performance measurement vectors for both the completed learning target points and the candidate target points, denoted as $\eta_{p1}, \eta_{p2}, \ldots, \eta_{pk}$. The $\eta_{pi} = [\eta_{i1}, \eta_{i2}, \eta_{i3}, \eta_{i4}]^T$ corresponds to the four performance indicators. It is important to note that η_{pi} represents the evaluation vector corresponding to the $(i-1)$th candidate target point. However, due to the disparate ranges of these performance metrics, direct comparisons may inadvertently overlook the significance of metrics with smaller value ranges. Hence, we normalize all performance metrics and scale them to the interval [0,1] to obtain the normalized evaluation vector:

$$\eta^*_{pi} = [Norm(\eta_{i1}), Norm(\eta_{i2}), Norm(\eta_{i3}), Norm(\eta_{i4})]^T \tag{22}$$

where

$$Norm(\eta_{ij}) = \frac{\eta_{ij} - \min_{i=1}^{i=k} \eta_{ij}}{\max_{i=1}^{i=k} \eta_{ij} - \min_{i=1}^{i=k} \eta_{ij}} \tag{23}$$

For the evaluation of new target points, it is evident that we can assess the difficulty of these new target points by computing the distance between the evaluation vectors of the new target points and the completed learning target point. Specifically, we define $d_{p1,pj}$ as follows:

$$d_{p1,pj} = \|\eta^*_{p1} - \eta^*_{pj}\|_2 \tag{24}$$

Here, d_{p_1,p_j} represents the Euclidean distance between the evaluation vector η^*_{p1} of the completed learning target point p_1 and the evaluation vector η^*_{pj} of the new target point p_j. This metric serves as an indicator of the difficulty level of the new target points, where a larger value of $d_{p1,pj}$ indicates a higher level of difficulty.

4.4.3. Target location selection

To enhance the learning efficiency of multi-objective UGV navigation, it is essential to choose a new task point of moderate difficulty for the previously learned strategy π. As depicted in Fig. 2, the Target Point Selection module involves clustering points of varying difficulty levels and identifying a task point that falls within the moderate difficulty range. Initially, we employ the K-means algorithm [50] to classify new target points based on their evaluation distances, categorizing them into three tiers: easy, moderate, and difficult. This classification primarily considers two factors. First, simple task points often lack sufficient challenge, which may not effectively promote further learning. Conversely, excessively difficult task points may impose an undue burden on existing strategies, potentially resulting in failure or suboptimal performance. Therefore, focusing on task points of moderate difficulty ensures that the selected tasks provide an appropriate level of challenge, thereby enhancing the UGV's navigation skills. Once clustering is complete, we sort these clusters in ascending order based on their evaluation distance values. From the cluster representing moderate difficulty, we select the target point closest to the cluster center as the next training task for the learning strategy π.

5. Experiments and results

In this section, we present the training and testing results of our proposed SESN approach on navigation tasks. We first demonstrate the training performance of the SESN. Subsequently, we validate the learned policies in two unknown maps with varying obstacle densities. Finally, we evaluate the performance of the SESN under different human preferences and showcase the effectiveness of the preference-based multi-objective approach. The video demonstration is available at https://www.bilibili.com/video/BV1nx4y1a71E.

5.1. Results of training process

We have developed a simulated environment in Gazebo to facilitate the training of UGVs in navigating among heterogeneous obstacles. Taking account of real-world scenarios, the UGV training environment includes a large number of pedestrians and static obstacles. Fig. 3 illustrates the entire training map, spanning dimensions of 40×60 m. The training map features five types of static obstacles such as shelving units, stacked goods, and stationary pedestrians, as well as fifteen moving pedestrians. Each pedestrian has unique motion speeds and trajectories to enrich the environment's realism. Moreover, in order to fulfill the requirement of automatic curriculum learning in the SESN approach, densities of obstacles and pedestrians progressively increase from the lower-left corner to the upper-right corner of the training map.

The lower-left corner of Fig. 3 depicts the initial position of the UGV for each training episode. Equipped with a 2-D laser scanner boasting a 360° field of view, the UGV can perceive its surroundings. At each time step, the laser scanner outputs a 180-dimensional vector to capture distances surrounding the UGV with a precision of 2°, a maximum detection span extending to 3.5 m, and a sampling frequency of 0.1 s.

In the training scenarios, we have implemented two baselines for comparison: improved DDPG algorithm [51], and classical algorithm based on TD3 [52]. We select these algorithms because we adopt a combination reward approach to achieve multi-objective reinforcement

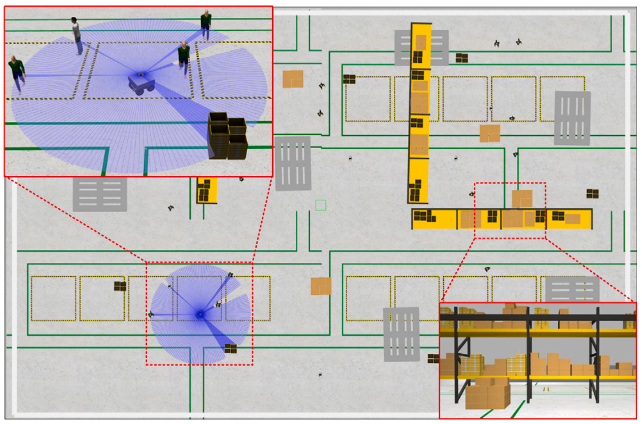


Fig. 3. An illustration of the designed training map with various obstacles in a factory-like environment.

Table 1
The training hyper-parameters of algorithms.

Hyper-parameter	Value
Training Episodes	3000
Steps per Episode	1500
Memory Pool Size	1500000
Batch Size	512
Discount Factor	0.99
Noise Scale Factor	0.2
Learning Rate for A-network	0.0003
Learning Rate for C-network	0.0003
C-target Network Update Frequency	10
A-target Network Update Frequency	5

learning. By transforming multiple objectives into a single-objective reward, we can effectively leverage the single-objective reinforcement learning framework to address multi-objective problems. Therefore, comparing the SESN method with algorithms based on single-objective reinforcement learning (SORL) is both reasonable and appropriate. Additionally, in order to further evaluate the effectiveness of the ACL, we have also conducted ablation experiments by excluding the ACL framework in the SESN architecture. To ensure equitable evaluations of algorithmic efficacy, all algorithms are subjected to the same state spaces and reward functions, with network updates executed using identical hyper-parameters detailed in Table 1. It should be noted that the DDPG-based algorithm excludes noise and delayed updates. Compared to other algorithms, our SESN also considers human preferences by integrating the reward values of different objectives using the preference-guided scalarization function. To ensure fairness in comparisons with other algorithms, we set the preference value range such that the expected value of the scalarized reward aligns with the reward values of the comparison methods. Additionally, during training, the preference values are uniformly sampled from a predefined range, ensuring stability throughout the process.

Fig. 4 illustrates the learning curves of the four algorithms throughout the training process. Each algorithm performs three random initializations with different random seeds. Each training of algorithms runs 3000 episodes, amounting to approximately 32 h in total. Notably, the shaded region in Fig. 4 represents the 95% confidence interval, while the bold lines indicate the average values. To mitigate the impact of task randomness on rewards, the episode rewards are plotted using the average values over 200 consecutive episodes.

As shown in Fig. 4, we can find that the rewards of all four algorithms converge steadily to better values, and, thus, we can say that each algorithm converges to a near-optimal policy. At the end of training, the 0.95 confidence intervals of the rewards are relatively small, which indicates that the policies learned in the training episodes

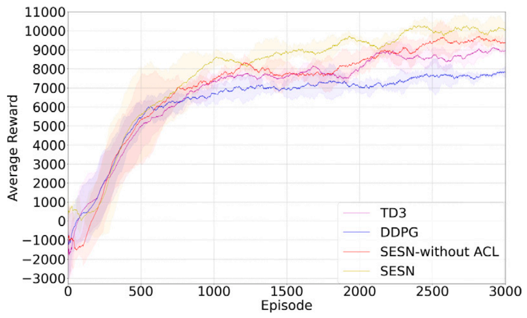


Fig. 4. Average rewards during the learning process of four algorithms.

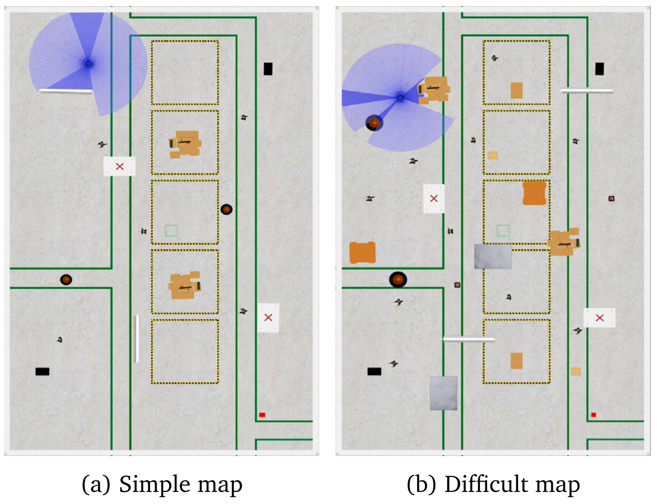


Fig. 5. Testing Scenarios with different obstacle densities.

have similar patterns. By comparing the convergence patterns of the four algorithms, several conclusions can be drawn. Firstly, our proposed SESN achieves superior cumulative rewards in contrast to alternative approaches. This indicates that through a multi-objective reinforcement learning framework, UGVs can adeptly learn policies that harmonize with human preferences. Secondly, our SESN demonstrates accelerated step-wise convergence after 600 episodes in comparison with the approaches without ACL. This observation also reflects the effectiveness of the ACL structure embedded within our SESN.

5.2. Results of testing in two scenarios

To evaluate the navigation performance of the proposed SESN, we carry out experiments in two distinct pedestrian environments characterized by varying obstacle densities. Fig. 5 illustrates two test scenarios, each measuring 25×40 m. The simpler scenario consists of 10 static obstacles of various sizes and 5 pedestrians with different speeds. In the more challenging scenario, the number of obstacles is doubled compared to the simpler one.

To quantitatively evaluate the performance of the algorithms, we considers the following six metrics:

- **Success Rate (SR)**: the percentage of successful UGV arrivals at the goal without collision.
- **Collision Rate (CR)**: the percentage of UGV collisions with pedestrians or static obstacles.
- **Timeout Rate (TR)**: the percentage of UGV failures to reach the goal within the specified number of steps without collision.
- **Average Time (AT)**: the average time taken by the UGV to reach the goal.

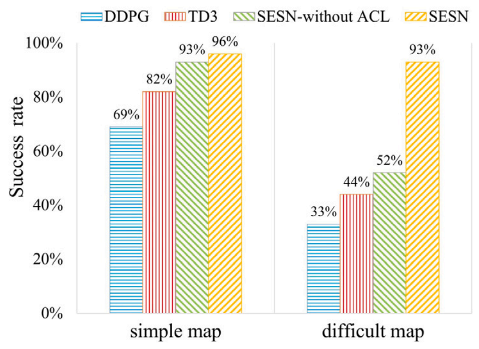


Fig. 6. Comparison of the success rates using different navigation strategies.

- **Average Path Length (AP)**: the average path length taken by the UGV to reach the goal.
- **Average Curvature (AC)**: the average curvature of the path taken by the UGV to reach the goal.

Table 2 presents the performance metrics of each strategy for 100 tests conducted in both simple and difficult maps. It is worth noting that the values in parentheses represent the standard deviation across 100 test iterations.

In terms of navigation success rate, the DDPG algorithm exhibits high failure rates due to its insufficient exploration, resulting in close-range evasion behavior when encountering pedestrians. We can find that TD3 demonstrates improved adaptation to pedestrian environments by integrating noise and delayed update strategies into the agent's behavior policy. In contrast, our SESN outperforms both the above by guiding the agent's behavior at each decision step based on human preferences, consequently achieving superior navigation performance. Moreover, we can see that our SESN achieves success rates averaging 109.0% higher than DDPG and 63.1% higher than TD3.

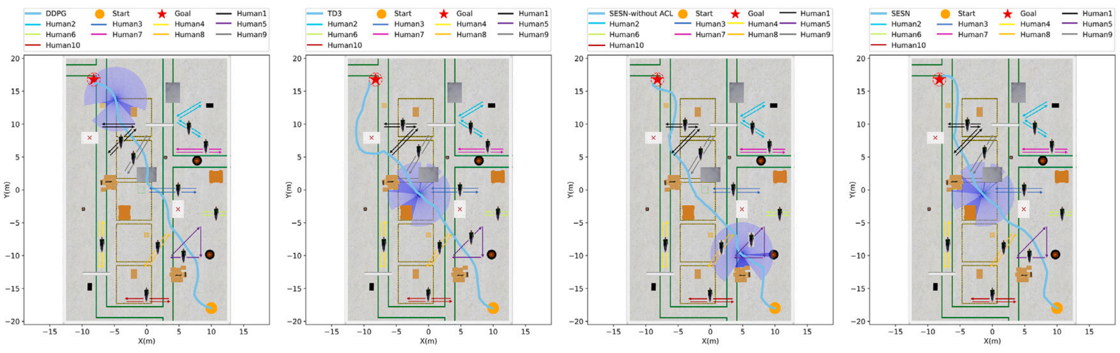
In the simple map, our SESN achieves higher navigation success rates by enhancing safety and reducing smoothness through adjustments in human preferences. Thus, such adjustments inevitably result in an increase in the average path curvature. In the difficult map, our SESN prioritizes safety over efficiency, demonstrating outstanding performance in success rate and average curvature while making sacrifices in efficiency. Although our SESN's successful paths are longer compared to the baselines, it demonstrates the capability to navigate more complex crowd environments, where the baseline algorithms often fail. Fig. 6 shows the comparison of navigation success rates under different strategies.

Fig. 7 displays the trajectories of successful navigation under different algorithms in the challenging environment. We can see that DDPG consistently moves towards the goal along the shortest path, demonstrating a stronger sense of purposefulness. When encountering pedestrians, TD3 chooses to maintain a safe distance and exhibits noticeable turning behavior. On the other hand, our SESN and SESN-without ACL tend to decelerate and maneuver early when encountering pedestrians. As a result, while DDPG achieves shorter navigation distances, it also has a higher likelihood of colliding with pedestrians, whereas our SESN achieves better safety attributes by slightly extending the navigation distance.

Furthermore, we also conduct ablation studies on the ACL component embedded in our SESN. The results show that SESN-without ACL still demonstrates excellent average curvature and the success rate of up to 93% in the simple map. However, in the complex map, SESN-without ACL achieves only a 52% success rate. This indicates that SESN-without ACL can effectively handle navigation tasks in simple environments, but cannot deal with dense pedestrians in the same level. It also highlights the effectiveness of ACL in enhancing SESN's exploration capabilities across the entire human preference space, which enables the algorithm to converge to a set of solutions under different preferences.

Table 2
Quantitative navigation results with different learned policies.

Scenario	Method	AC↓	AP↓	AT↓	SR↑	CR↓	TR↓
Simple map	DDPG [51]	0.61(0.21)	**41.86(1.24)**	47.90(1.81)	69%	31%	**0%**
	TD3 [52]	0.60(0.16)	43.47(1.52)	**47.59(1.55)**	82%	18%	**0%**
	SESN-without ACL	**0.56(0.07)**	43.18(0.53)	63.39(1.54)	93%	7%	**0%**
	SESN(Ours)	0.67(0.11)	43.18(1.36)	50.36(2.09)	**96%**	**4%**	**0%**
Difficult map	DDPG [51]	0.90(0.35)	**42.68(0.90)**	55.20(1.57)	33%	49%	18%
	TD3 [52]	0.66(0.30)	44.25(1.50)	55.66(2.43)	44%	56%	**0%**
	SESN-without ACL	1.50(0.41)	43.98(0.73)	67.30(2.92)	52%	48%	**0%**
	SESN(Ours)	**0.58(0.05)**	43.34(0.52)	**55.04(1.78)**	**93%**	**7%**	**0%**



(a) DDPG trajectory (b) TD3 trajectory (c) SESN-without ACL trajectory (d) SESN trajectory

Fig. 7. Navigation trajectories using different learned policies in the difficult map.

Table 3
Quantitative results with different preferences.

Method	Preference	AC (↓)	AP (↓)	AT (↓)	SR (↑)	CR (↓)	TR (↓)
SESN	[0.45, 0.45, 0.1]	**0.57 (0.05)**	43.34 (0.80)	**54.11 (1.13)**	85%	15%	**0%**
	[0.50, 0.40, 0.1]	0.60 (0.13)	**43.32 (0.80)**	54.39 (1.66)	87%	13%	**0%**
	[0.55, 0.35, 0.1]	0.58 (0.05)	43.34 (0.52)	55.04 (1.78)	**93%**	**7%**	**0%**

5.3. Influences of different human preferences

In order to thoroughly investigate the efficacy of the proposed approach across a comprehensive range of human preferences, we also conduct rigorous tests on the SESN approach under varying human preference configurations in the challenging environment. Experimental evaluations are carried out on the trained policies under diverse human preference settings, and the outcomes are outlined in Table 3.

The three parameters of the human preference vector correspondingly dictate the weighting of rewards for safety, efficiency, and smoothness. Remarkably, as the human preference transitions from efficiency to safety, there is a marked enhancement in the average success rate of the UGV, increasing from 85% to 93% over 100 testing rounds. Concurrently, a marginal increase in the average task completion time is observed. This modest increase stems from the fact that the policies derived from training already approximate optimality with regard to the efficiency objective. Nonetheless, the substantial improvement in the success rate underscores the adaptability of the UGV's learned policies to the human preference space, effectively meeting diverse navigation requirements across various scenarios characterized by distinct human preference configurations.

6. Conclusions and discussion

This work presents a novel preference-based multi-objective reinforcement learning approach for UGV navigation, considering diverse human preferences such as safety, efficiency, and smoothness. Our proposed preference-based method efficiently obtains a set of solutions across the entire preference space in a single training process. Additionally, an automatic curriculum learning algorithm is introduced to expedite the training process. Comparative experiments demonstrate the outstanding performance of our proposed approach that can achieve competitive results in success rate and offer solutions superior to other baseline algorithms across the preference space.

Certainly, our research has certain limitations. Our approach primarily relies on 2D LiDAR for environmental perception. However, real-world factory environments often contain obstacles that can be traversed or have heights below the LiDAR's detection plane, such as traversable shelves and low-stacked goods. These characteristics can hinder 2D LiDAR from accurately detecting critical environmental information, which may affect the performance and reliability of autonomous navigation. Consequently, the limitations of light-weight LiDAR-based perception could become a significant factor in the failure of the UGV's strategy in complex factory settings.

To address this issue, our future research will focus on equipping UGVs with cameras and 2D LiDAR by employing multi-sensor fusion technology to enhance overall environmental perception. By integrating 2D LiDAR with visual data, we aim to capture more comprehensive environmental details of traversable and hard-to-detect obstacles. This approach will enable the UGV to better navigate diverse obstacles in factory environments and improve its performance and adaptability in complex scenarios.

CRediT authorship contribution statement

Shunyu Tian: Writing – original draft, Visualization, Validation, Software, Resources, Methodology, Formal analysis, Conceptualization. **Changyun Wei:** Writing – review & editing, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis. **Shaojie Jian:** Visualization, Validation, Software, Resources, Data curation. **Ze Ji:** Writing – review & editing, Validation, Software, Conceptualization.

Declaration of competing interest

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

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 52371275. We thank all the anonymous reviewers who generously contribute their time and efforts. Their professional recommendations can greatly enhance the quality of the manuscript.

References

[1] M.P. Manuel, M. Faied, M. Krishnan, M. Paulik, Robot platooning strategy for search and rescue operations, Intell. Serv. Robot. 15 (1) (2022) 57–68.
[2] P. Pauwels, R. de Koning, B. Hendrikx, E. Torta, Live semantic data from building digital twins for robot navigation: Overview of data transfer methods, Adv. Eng. Inform. 56 (2023) 101959.
[3] C. Wang, X. Cui, S. Zhao, X. Zhou, Y. Song, Y. Wang, K. Guo, Enhancing vehicle ride comfort through deep reinforcement learning with expert-guided soft-hard constraints and system characteristic considerations, Adv. Eng. Inform. 59 (2024) 102328.
[4] R. Hercik, R. Byrtus, R. Jaros, J. Koziorek, Implementation of autonomous mobile robot in smartfactory, Appl. Sci. 12 (17) (2022) 8912.
[5] K. Shen, C. Li, D. Xu, W. Wu, H. Wan, Sensor-network-based navigation of delivery robot for baggage handling in international airport, Int. J. Adv. Robot. Syst. 17 (4) (2020) 1729881420944734.
[6] L. Huber, J.-J. Slotine, A. Billard, Avoiding dense and dynamic obstacles in enclosed spaces: Application to moving in crowds, IEEE Trans. Robot. 38 (5) (2022) 3113–3132.
[7] S.K. Jha, S. Prakash, R.S. Rathore, M. Mahmud, O. Kaiwartya, J. Lloret, Quality-of-service-centric design and analysis of unmanned aerial vehicles, Sensors 22 (15) (2022) 5477.
[8] H. Kivrak, F. Cakmak, H. Kose, S. Yavuz, Social navigation framework for assistive robots in human inhabited unknown environments, Eng. Sci. Technol. Int. J. 24 (2) (2021) 284–298.
[9] F.A. Cosío, M.P. Castañeda, Autonomous robot navigation using adaptive potential fields, Math. Comput. Modelling 40 (9–10) (2004) 1141–1156.
[10] W. Gueaieb, M.S. Miah, An intelligent mobile robot navigation technique using RFID technology, IEEE Trans. Instrum. Meas. 57 (9) (2008) 1908–1917.
[11] G. Cheng, Y. Wang, L. Dong, W. Cai, C. Sun, Multi-objective deep reinforcement learning for crowd-aware robot navigation with dynamic human preference, Neural Comput. Appl. 35 (22) (2023) 16247–16265.
[12] K. Suresh, R. Venkatesan, S. Venugopal, Mobile robot path planning using multi-objective genetic algorithm in industrial automation, Soft Comput. 26 (15) (2022) 7387–7400.
[13] J. Xu, Y. Tian, P. Ma, D. Rus, S. Sueda, W. Matusik, Prediction-guided multi-objective reinforcement learning for continuous robot control, in: International Conference on Machine Learning, PMLR, 2020, pp. 10607–10616.
[14] A. Abdolmaleki, S. Huang, L. Hasenclever, M. Neunert, F. Song, M. Zambelli, M. Martins, N. Heess, R. Hadsell, M. Riedmiller, A distributional view on multi-objective policy optimization, in: International Conference on Machine Learning, PMLR, 2020, pp. 11–22.
[15] S. Thrun, Probabilistic robotics, Commun. ACM 45 (3) (2002) 52–57.
[16] P. Fiorini, Z. Shiller, Motion planning in dynamic environments using velocity obstacles, Int. J. Robot. Res. 17 (7) (1998) 760–772.
[17] J. Van den Berg, M. Lin, D. Manocha, Reciprocal velocity obstacles for real-time multi-agent navigation, in: IEEE International Conference on Robotics and Automation, IEEE, 2008, pp. 1928–1935.
[18] J. Van Den Berg, S.J. Guy, M. Lin, D. Manocha, Reciprocal n-body collision avoidance, in: Robotics Research: The 14th International Symposium, Springer, 2011, pp. 3–19.
[19] J. Huang, J. Zeng, X. Chi, K. Sreenath, Z. Liu, H. Su, Velocity obstacle for polytopic collision avoidance for distributed multi-robot systems, IEEE Robot. Autom. Lett. (2023).
[20] E. Montero, N. Pico, M. Ghergherehchi, H.S. Song, Memory-driven deep-reinforcement learning for autonomous robot navigation in partially observable environments, Eng. Sci. Technol. Int. J. 62 (2025) 101942.
[21] N. Pico, E. Montero, A. Amirbek, E. Auh, J. Jeon, M.S. Alvarez-Alvarado, B. Jamil, R. Algabri, H. Moon, Human and environmental feature-driven neural network for path-constrained robot navigation using deep reinforcement learning, Eng. Sci. Technol. Int. J. 64 (2025) 101993.
[22] K. Zhu, B. Li, W. Zhe, T. Zhang, Collision avoidance among dense heterogeneous agents using deep reinforcement learning, IEEE Robot. Autom. Lett. 8 (1) (2022) 57–64.
[23] H.-C. Wang, S.-C. Huang, P.-J. Huang, K.-L. Wang, Y.-C. Teng, Y.-T. Ko, D. Jeon, I.-C. Wu, Curriculum reinforcement learning from avoiding collisions to navigating among movable obstacles in diverse environments, IEEE Robot. Autom. Lett. 8 (5) (2023) 2740–2747.
[24] M. Nishimura, R. Yonetani, L2b: Learning to balance the safety-efficiency trade-off in interactive crowd-aware robot navigation, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2020, pp. 11004–11010.
[25] A. Jain, D. Chen, D. Bansal, S. Scheele, M. Kishore, H. Sapra, D. Kent, H. Ravichandar, S. Chernova, Anticipatory human-robot collaboration via multi-objective trajectory optimization, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2020, pp. 11052–11057.
[26] A. Ravankar, A.A. Ravankar, A. Rawankar, Y. Hoshino, Autonomous and safe navigation of mobile robots in vineyard with smooth collision avoidance, Agriculture Basel 11 (10) (2021).
[27] W. Zhu, M. Hayashibe, A hierarchical deep reinforcement learning framework with high efficiency and generalization for fast and safe navigation, IEEE Trans. Ind. Electron. 70 (5) (2023) 4962–4971.
[28] A. Ramezani Dooraki, D.-J. Lee, A multi-objective reinforcement learning based controller for autonomous navigation in challenging environments, Machines 10 (7) (2022) 500.
[29] C.F. Hayes, R. Rădulescu, E. Bargiacchi, J. Källström, M. Macfarlane, M. Reymond, T. Verstraeten, L.M. Zintgraf, R. Dazeley, F. Heintz, et al., A practical guide to multi-objective reinforcement learning and planning, Auton. Agents Multi-Agent Syst. 36 (1) (2022) 26.
[30] N. Deshpande, D. Vaufreydaz, A. Spalanzani, Navigation in urban environments amongst pedestrians using multi-objective deep reinforcement learning, in: IEEE International Intelligent Transportation Systems Conference, ITSC, IEEE, 2021, pp. 923–928.
[31] A. Agrawal, B.R. Bakshi, H. Kodamana, M. Ramteke, Multi-objective optimization of food-energy-water nexus via crops land allocation, Comput. Chem. Eng. 183 (2024) 108610.
[32] Z. He, K.P. Tran, S. Thomassey, X. Zeng, J. Xu, C. Yi, Multi-objective optimization of the textile manufacturing process using deep-q-network based multi-agent reinforcement learning, J. Manuf. Syst. 62 (2022) 939–949.
[33] A. Chraibi, S. Ben Alla, A. Touhafi, A. Ezzati, A novel dynamic multi-objective task scheduling optimization based on dueling DQN and PER, J. Supercomput. 79 (18) (2023) 21368–21423.
[34] H. Mossalam, Y.M. Assael, D.M. Roijers, S. Whiteson, Multi-objective deep reinforcement learning, 2016, arXiv preprint arXiv:1610.02707.
[35] M. Zuluaga, A. Krause, et al., e-pal: An active learning approach to the multi-objective optimization problem, J. Mach. Learn. Res. 17 (104) (2016) 1–32.
[36] R.S. Rathore, S. Sangwan, S. Prakash, K. Adhikari, R. Kharel, Y. Cao, Hybrid WGWO: whale grey wolf optimization-based novel energy-efficient clustering for EH-WSNs, EURASIP J. Wirel. Commun. Netw. 2020 (2020) 1–28.
[37] K. Zhang, S. McLeod, M. Lee, J. Xiao, Continuous reinforcement learning to adapt multi-objective optimization online for robot motion, Int. J. Adv. Robot. Syst. 17 (2) (2020) 1729881420911491.
[38] A. Abels, D. Roijers, T. Lenaerts, A. Nowé, D. Steckelmacher, Dynamic weights in multi-objective deep reinforcement learning, in: International Conference on Machine Learning, PMLR, 2019, pp. 11–20.
[39] R. Yang, X. Sun, K. Narasimhan, A generalized algorithm for multi-objective reinforcement learning and policy adaptation, Adv. Neural Inf. Process. Syst. 32 (2019).
[40] R. Portelas, C. Colas, L. Weng, K. Hofmann, P.-Y. Oudeyer, Automatic curriculum learning for deep rl: A short survey, 2020, arXiv preprint arXiv:2003.04664.
[41] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M.E. Taylor, P. Stone, Curriculum learning for reinforcement learning domains: A framework and survey, J. Mach. Learn. Res. 21 (181) (2020) 1–50.
[42] S.D. Morad, R. Mecca, R.P. Poudel, S. Liwicki, R. Cipolla, Embodied visual navigation with automatic curriculum learning in real environments, IEEE Robot. Autom. Lett. 6 (2) (2021) 683–690.
[43] M. Sánchez, J. Morales, J.L. Martínez, Reinforcement and curriculum learning for off-road navigation of an UGV with a 3D LiDAR, Sensors 23 (6) (2023) 3239.
[44] Y. Yin, Z. Chen, G. Liu, J. Yin, J. Guo, Autonomous navigation of mobile robots in unknown environments using off-policy reinforcement learning with curriculum learning, Expert Syst. Appl. 247 (2024) 123202.
[45] T. Matiisen, A. Oliver, T. Cohen, J. Schulman, Teacher-student curriculum learning, IEEE Trans. Neural Netw. Learn. Syst. 31 (9) (2020) 3732–3740.
[46] Y. Song, H. Lin, E. Kaufmann, P. Dürr, D. Scaramuzza, Autonomous overtaking in gran turismo sport using curriculum reinforcement learning, in: IEEE International Conference on Robotics and Automation, IEEE, 2021, pp. 9403–9409.

[47] S. Fujimoto, H. Hoof, D. Meger, Addressing function approximation error in actor-critic methods, in: International Conference on Machine Learning, PMLR, 2018, pp. 1587–1596.

[48] C. Wang, X. Chen, C. Li, R. Song, Y. Li, M.Q.-H. Meng, Chase and track: Toward safe and smooth trajectory planning for robotic navigation in dynamic environments, IEEE Trans. Ind. Electron. 70 (1) (2022) 604–613.

[49] R.W. Liu, Y. Lu, Y. Gao, Y. Guo, W. Ren, F. Zhu, F.-Y. Wang, Real-time multi-scene visibility enhancement for promoting navigational safety of vessels under complex weather conditions, IEEE Trans. Intell. Transp. Syst. (2024).

[50] P.S. Bradley, U.M. Fayyad, Refining initial points for k-means clustering, in: ICML, vol. 98, Citeseer, 1998, pp. 91–99.

[51] P. Li, X. Ding, H. Sun, S. Zhao, R. Cajo, Research on dynamic path planning of mobile robot based on improved DDPG algorithm, Mob. Inf. Syst. 2021 (2021) 1–10.

[52] J. Jia, X. Xing, D.E. Chang, Gru-attention based td3 network for mobile robot navigation, in: 22nd International Conference on Control, Automation and Systems, ICCAS, IEEE, 2022, pp. 1642–1647.