2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Skill Q-Network: Learning Adaptive Skill Ensemble for Mapless Navigation in Unknown Environments

Hyunki Seong*[1] and David Hyunchul Shim[1]

***Abstract*— This paper focuses on the acquisition of mapless navigation skills within unknown environments. We introduce the Skill Q-Network (SQN), a novel reinforcement learning method featuring an adaptive skill ensemble mechanism. Unlike existing methods, our model concurrently learns a high-level skill decision process alongside multiple low-level navigation skills, all without the need for prior knowledge. Leveraging a tailored reward function for mapless navigation, the SQN is capable of learning adaptive maneuvers that incorporate both exploration and goal-directed skills, enabling effective navigation in new environments. Our experiments demonstrate that our SQN can effectively navigate complex environments, exhibiting a 40% higher performance compared to baseline models. Without explicit guidance, SQN discovers how to combine low-level skill policies, showcasing both goal-directed navigations to reach destinations and exploration maneuvers to escape from local minimum regions in challenging scenarios. Remarkably, our adaptive skill ensemble method enables zero-shot transfer to out-of-distribution domains, characterized by unseen observations from non-convex obstacles or uneven, subterranean-like environments. The project page is available at `https://sites.google.com/view/skill-q-net`.**

I. Introduction

Safe and efficient navigation in unknown environments remains a significant challenge in robotics. In scenarios where the robotic agent has access to global maps, finding the optimal path to the goal point is feasible. However, in unfamiliar environments, the agent must rely solely on partially observable, ego-centric information for navigation.

In navigation challenges lacking prior information, two key requirements emerge: 1) the application of adaptive navigation strategies suitable for the current situation, and 2) the adoption of a comprehensive approach to handle unseen, out-of-distribution (OOD) observations. If the agent merely aims towards the goal, it could encounter dead-ends or repeatedly traverse previously visited areas, resulting in inefficient back-and-forth movements. Therefore, beyond a goal-directed strategy, the agent must also adaptively employ exploration skills to navigate out of local minima. Furthermore, in new environments, it is likely to encounter partially-known scenes and unfamiliar OOD terrains, necessitating a robust ability to manage novel situations.

This work was supported by the Technology Innovation Program (Development of drone-robot cooperative multimodal delivery technology for cargo with a maximum weight of 40kg in urban areas) funded by the Ministry of Trade, Industry & Energy (MOTIE), South Korea, under Grant RS-2023-00256794.

*Corresponding author

[1]Hyunki Seong and David Hyunchul Shim are with the School of Electrical Engineering, Korea Advanced Institute of Science and Technology, Daejeon, South Korea. {hynkis, hcshim}@kaist.ac.kr

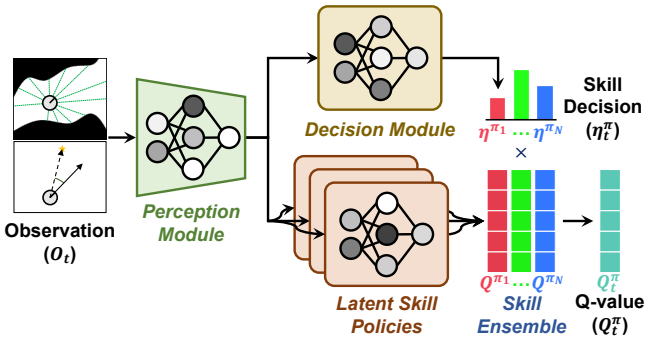

Fig. 1: Overview of the Skill Q-Network.

Recent studies have introduced numerous experience-based reinforcement learning (RL) approaches [1], [2] for navigating environments without maps. These approaches employ policy networks, trained through reward functions with minimal heuristics, to navigate to destinations using egocentric range information about the surrounding terrain and relative positioning information toward the goal. moreover, incorporating memory-based modules like LSTM or GRU enables the development of navigation policies that integrate the temporal characteristics of the ego robot's historical trajectory [3], [4]. However, learning various strategies with a single end-to-end network remains a challenge. Without pre-defining and individually training low-level skills [5], [6], it is still difficult to train a unified policy network that can employ a diverse set of navigation skills.

In this paper, we introduce Skilled Q-Network (SQN), a novel deep Q-learning approach for end-to-end mapless navigation that incorporates an adaptive skill ensemble mechanism. Our network features multiple latent skill policies and a skill decision module, differentiated through module embedding processes. The decision module infers a skill decision, which evaluates importance scores for each skill, facilitating an internal high-level decision-making process within the end-to-end architecture. The skill decision then aggregates the Q-values from the latent skill policies into a single Q-value vector for action selection. This skill-ensembled Q-learning approach, with a latent skill decision mechanism, enables the network to learn adaptive navigation strategies without the need for prior skill-level knowledge.

To train navigation skills, we formulate a tailored reward function with terms that encourage exploration of unknown environments for discovering feasible goal achievement regions and exploitation of goal-reachable situations for successful arrival. By providing these reward signals that balance exploration and goal-directed features, we enable SQN to acquire a diverse set of navigation skills, adept at

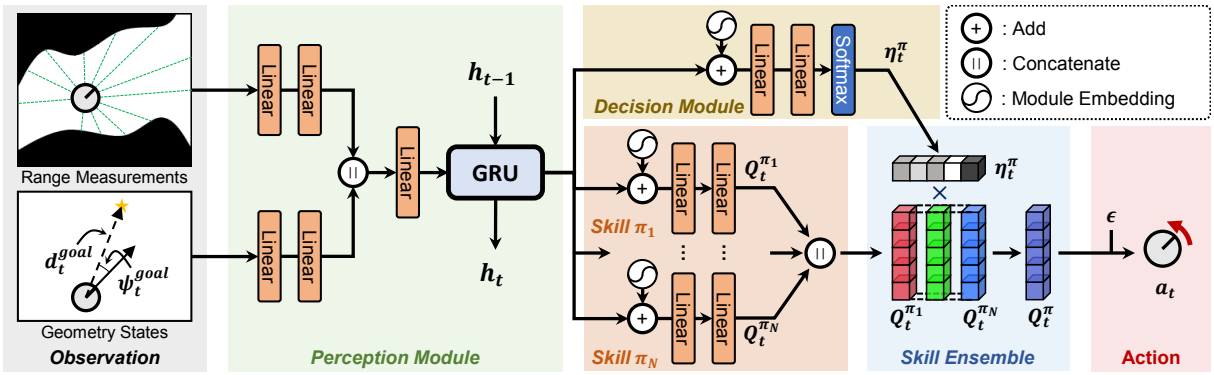

Fig. 2: Detailed network architecture of the Skill Q-Network.

navigating complex, unknown environments.

In our extensive experiments, SQN consistently outperforms baseline models across various mapless environments, demonstrating a 40% improvement in performance compared to conventional models. Additionally, we investigate the adaptive skill decision trajectories of our model, showcasing its ability to effectively combine latent navigational skills to overcome local minimum situations and navigate through complex scenarios. This highlights SQN's significant adaptability and strategic capability in end-to-end mapless navigation. Notably, we observe that our SQN can leverage its adaptive decision mechanism for zero-shot transfer to previously unseen novel environments. These environments encompass scenarios with noisy disturbance conditions, observation noise settings, or unstructured observation patterns, such as open-space non-convex obstacles or subterranean cave-like scenes, demonstrating our method's capacity to handle out-of-distribution situations.

The summary of our contributions is as follows:

- We present Skill Q-Network (SQN), a novel RL method capable of learning multiple navigation skills through adaptive skill ensemble.
- We design a tailored reward function to learn effective mapless navigation in complex environments.
- We empirically demonstrate the effectiveness of our adaptive skill ensemble method in addressing challenging mapless navigation problems across diverse environments including out-of-distribution settings.

II. Related Works

A. Conventional Mapless Navigation

Various approaches have been developed for robotic navigation in unknown environments. Conventional methodologies [7]–[9] often employ Simultaneous Localization and Mapping (SLAM) to generate maps with topological graphs for navigating new terrains. These methods incrementally expand the map as the agent traverses the area, effectively managing the explored regions [10]–[12]. This dynamic process enables the robot to distinguish between explored and unexplored regions in environments, thereby facilitating stable navigation. However, map generation and graph management are both complex and computationally demanding processes. The complexity of these operations escalates significantly with the size of the environment, making it challenging to apply these solutions in expansive or complex areas. Moreover, they depend on handcrafted rules and the heuristics of human engineers to balance exploration in unknown spaces and goal-oriented navigation toward a destination. This reliance can hinder the agent's ability to generalize to new scenarios.

B. Learning-based Mapless Navigation

To overcome existing limitations, recent studies have focused on developing learning-based navigation policies that eliminate the need for map generation, utilizing imitation [13]–[15] and reinforcement learning [1]–[6], [16]–[21] approaches. Many of these studies have introduced mapless policy networks that process ego-centric sensory inputs, such as range measurements, and relative distance and orientation towards the goal. Initial research efforts often employed network architectures based on Multilayer Perceptrons (MLP), which solely rely on current observations for navigation [1], [2], [16], [17]. Although these models are suitable for simple scenarios, they often struggle to navigate complex environments with intricate topologies and fail to avoid getting stuck in unstructured regions.

Some researchers have adopted recurrent neural network modules, such as Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRU), for memory-based, mapless navigation [3], [4], [18], [19]. These methods utilize the temporal features of egocentric observations, facilitating efficient navigation in complex environments without the need to revisit areas. However, they primarily rely on a single policy network, which complicates the learning of adaptable maneuvers across various navigation scenarios, thereby limiting their ability to generalize in new environments.

Recent studies have introduced skill-based learning techniques [5], [6], [20], [21] for acquiring diverse navigation strategies through hierarchical learning. However, they still require separate high-level and low-level learning processes, necessitating tailored schemes for training each low-level skill [5], [6]. Moreover, the utilization of predefined skills introduces heuristics that may compromise generalization performance in new situations. In contrast, our approach learns low-level policies and an adaptive skill combination mechanism without relying on skill-level priors, offering a novel pathway to address the limitations of existing models.

III. METHODOLOGIES

A. State and Action Representation

To represent the surrounding navigation scene, our network receives two types of input state data, range measurements and navigational information. The range measurements $o_t^{range} \in \mathbb{R}^{71}$ constitute a set of distances to occupied areas, each calculated from the ego robot's center point. The maximum observation range is 5 m, and the field-of-view angle spans 350° from left to right, with an angle resolution of 5°. The navigational information $o_t^{goal} \in \mathbb{R}^2$ contains the line-of-sight distance d_t^{goal} and heading angle ψ_t^{goal} toward the goal point. Both data types are ego-centric and do not require prior knowledge of the driving environment while navigating to the goal point. To handle large distances and accommodate environments of various sizes, we normalize all observation values to a range of 0 to 1.

Our agent is assumed to be a differential-wheeled robot controlled by twist commands $c = [v, \omega]$, which include translational (v in m/s) and rotational (ω in rad/s) velocities. Accordingly, the robot agent's action space is represented by the following five discrete actions $a_t \in \mathbb{R}^5$: *no-operation* ($[0, 0]$), *forward* ($[2.5, 0]$), *backward* ($[-1.25, 0]$), *turn-left* ($[0, +\pi]$), and *turn-right* ($[0, -\pi]$).

B. Skill Q-Network

To design a policy network capable of utilizing multiple skill policies, we incorporate the concept of functional modularity, as proposed in [22], and develop the Skill Q-Network (SQN). The SQN is composed of modules categorized into three functions: the perception module, the planning module, and multiple latent skill policy modules (Fig. 2).

The perception module extracts sensory features from two types of observations: o_t^{range} and o_t^{goal}. These observations are individually transformed into $z_t^{range} \in \mathbb{R}^{128}$ and $z_t^{goal} \in \mathbb{R}^{128}$, respectively, using a Multi-layer Perceptron (MLP) that consists of two linear layers and ReLU non-linearity (Eq. 1). The resulting features are then concatenated into a single hidden vector. Additionally, to capture temporal dependencies on past observations, a GRU layer is integrated, enabling the network to derive a comprehensive perceptual feature $z_t^p \in \mathbb{R}^{128}$ that incorporates hidden features from previous observations h_{t-1} (Eq. 2).

$$z_t^i = \text{MLP}_i(o_t^i), \quad i = \{range, goal\} \tag{1}$$

$$z_t^p, h_t = \text{GRU}([z_t^{range}; z_t^{goal}], h_{t-1}) \tag{2}$$

The decision module derives a skill decision as an attention score vector for multiple latent skill policies based on the hidden feature z_t^p. To promote functional modularity within the skill decision-making mechanism, we add a learnable module embedding $e_{mod}^{dec} \in \mathbb{R}^{128}$ to the input perceptual feature. This enhanced input is then processed by an MLP block, equipped with two linear layers and ReLU nonlinear activation (Eq. 3). Finally, we apply a softmax operation to generate the skill decision $\eta_t^\pi \in \mathbb{R}^N$, which indicates the importance of each latent skill policy output (Eq. 4).

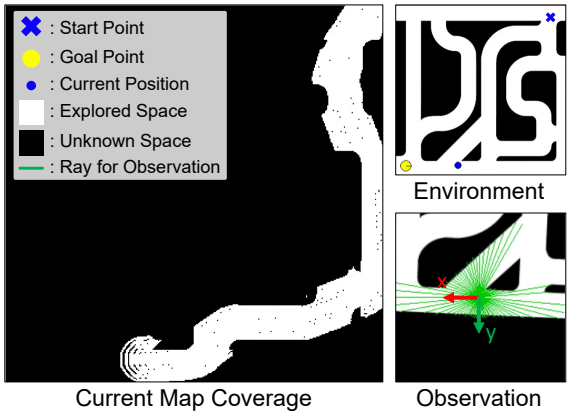

Fig. 3: An example illustrating the map coverage, along with the corresponding status and observation of the ego agent.

$$z_t^{dec} = \text{MLP}_{dec}(z_t^p + e_{mod}^{dec}), \quad e_{mod}^{dec} \in R^{128} \tag{3}$$

$$\eta_t^\pi = [\eta_t^{\pi_1}, ..., \eta_t^{\pi_N}] = \frac{\exp(z_t^{dec})}{\sum_{i=1}^{N} \exp(z_t^{dec}(i))} \tag{4}$$

The latent skill policy modules consist of N skill policy networks, each generating individual Q-values. Similar to the decision module, we apply module embeddings to distinguish the functional roles of the policies. Each latent skill policy is implemented using an MLP with two linear layers and ReLU non-linearity (Eq. 5). The Q-values produced by each skill network ($Q_t^{\pi_i},\ i = 1, \ldots, N$) are aggregated into a single Q-value Q_t^π through multiplication with the skill decision η_t^π from the decision module (Eq. 6). Finally, an epsilon-greedy strategy is employed to select the final discrete action $a_t \in \mathbb{R}^m$.

$$Q_t^{\pi_i} = \text{MLP}_{\pi_i}(z_t^p + e_{mod_i}), \quad i = 1, 2, \ldots, N \tag{5}$$

$$Q_t^\pi = [\eta_t^{\pi_1}, ..., \eta_t^{\pi_N}] \times \begin{bmatrix} Q_t^{\pi_1}(s, a_1), & ..., & Q_t^{\pi_1}(s, a_m) \\ & ... & \\ Q_t^{\pi_N}(s, a_1), & ..., & Q_t^{\pi_N}(s, a_m) \end{bmatrix} \tag{6}$$

Since SQN integrates a recurrent neural network, we adopt the R2D2 [23] reinforcement learning framework, which utilizes a burn-in mechanism. This approach involves executing inference for the initial steps of sequential episode data acquired through batch sampling. This process initializes the GRU's hidden state with the burn-in technique, enabling the computation of objectives based on Q-values under warm-start conditions. R2D2's training method effectively manages variable sequence states, significantly enhancing the stability and efficiency of learning from long-term sequential data. For further details on the training algorithm, we refer to [23].

C. Reward Function

We designed two positive reward terms and one negative penalty term to train diverse navigation skills.

Exploration Term: We compute map coverage using range measurements to represent the explored area on the global map. Subsequently, we calculate the difference between the sizes of the current and subsequent explored areas

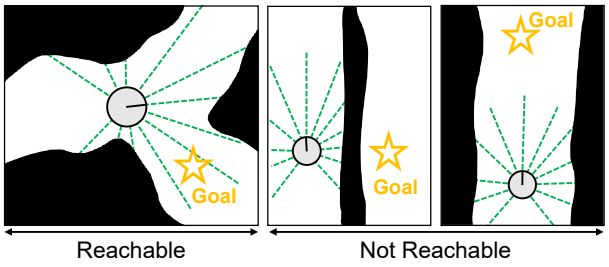

Fig. 4: Examples of goal (non-)reachability. In mapless environments, situations frequently arise where the goal point is close in terms of Euclidean distance yet remains unreachable.

to generate a positive reward signal, as follows:

$$r_{exp} = M(o^{range}_{t+1:0}) - M(o^{range}_{t:0}), \tag{7}$$

where $M(o^{range}_{t:0})$ represents the size of the map coverage up to the current observation o^{range}_{t}. This term encourages the ego agent to explore new regions and enhances navigation progress toward the goal in unknown environments. Additionally, it discourages the robot from revisiting previously explored areas, thus promoting more efficient navigation. Note that the map coverage is used solely as privileged information within the reward signal and does not form part of the agent's input state. Consequently, it allows our model to harness the benefits of map generation-based approaches [12] without the need to construct an expensive SLAM map.

Reachability-Aware Navigation Term: We introduce a positive reward signal r_{nav} for the agent's goal navigation. Typically, this signal is designed based on the Euclidean distance between the goal point and the ego robot [2], [24]. However, this method may excessively focus on minimizing the distance to the goal, leading to maneuvers that are susceptible to local minima. On the other hand, offering a sparse reward only [17], [19] upon reaching the goal may encourage the acquisition of various maneuvers, but it tends to make learning navigation more complex compared to the use of dense rewards. Given these considerations, we propose a partially dense reward term that accounts for the distance to the goal point g, incorporating the reachability from the ego robot's current observation space S_{obs} as follows:

$$r_{nav} = \begin{cases} exp(-d^{goal}_{t}) & g \in S_{obs} \\ 0 & g \notin S_{obs} \end{cases}, \tag{8}$$

where the goal g is considered reachable ($g \in S_{obs}$) if it is within the agent's field-of-view and the distance d^{goal}_{t} is less than the observed range measurement in the direction of the goal. This approach prevents the generation of over-optimistic reward signals in situations where the Euclidean distance to the goal point g is small, but the goal is located beyond a non-traversable area (see Fig. 4). As a result, it enables the ego agent to concentrate on learning exploration maneuvers by the signal r_{exp}, which facilitates escaping from local minimum situations through local exploration. Conversely, when the goal point is accessible to the ego robot, this reward term produces a positive signal, encouraging the agent to learn goal-directed maneuvers that prioritize reaching the goal over exploring new environments.

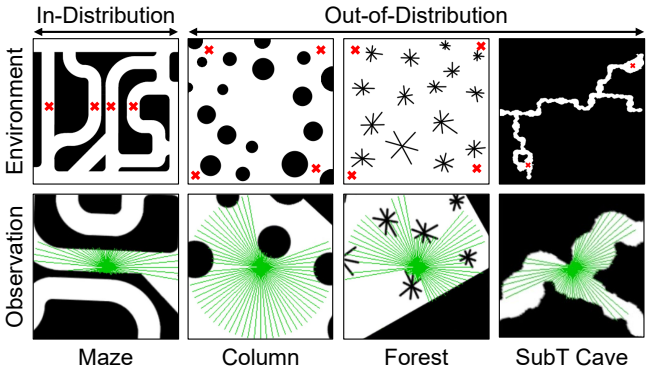

Fig. 5: Various environments and observation patterns for experiments. Observations are visualized with the ego agent's center point as the origin, where the local x-axis points east.

Time Step Penalty Term: Lastly, we define a constant negative reward term, $r_{time} = -1$, to incentivize the agent to perform actions that yield more positive reward signals and compensate for this negative penalty.

The total reward function is a weighted summation of the above reward terms as follows:

$$r_t = 2.0 \times r_{nav} + 1.0 \times r_{exp} + 0.2 \times r_{time} \tag{9}$$

IV. Experiments

A. Environments Setup

We utilize the Multiagent Particle Environments (MPE) [25] in the Pettingzoo framework [26] to create a particle agent-based training environment. Differing from the original setup, we limit the robot's movements to forwards, backwards, and rotations, mirroring the physical constraints of differential-wheeled robots. In the simulation, we only update to the next state if it is traversable; otherwise, we stop the motion to avoid navigating into occupied areas of the global map. Each training episode is composed of 400 time steps, with each step lasting 0.1 seconds, based on a 10 Hz cycle for the ego robot's control system.

We train the policy network in the Maze environment and validate it against challenging scenarios within this environment. Additionally, we create three unseen environments (Column, Forest, SubT Cave) to assess the policy model's performance under OOD conditions. During evaluation, start and goal points are randomly chosen from either the four or two highlighted locations (marked by red 'x's) in Fig. 5, focusing on scenarios that challenge the ego robot to skillfully navigate to the goal point.

Maze: For training and evaluation, we construct the Maze environment (17×17 m), featuring complex topology and multiple intersections. Various spawn points are defined for training. We randomly select two non-overlapping points to establish the ego robot's initial pose and goal point, with both position and orientation being randomly initialized. During evaluation, however, we focus on the four highlighted locations that can yield challenging routes, including multiple intersections between the start and goal points.

Column: The Column scenario is different from the training environment, being an open-space scenario with

TABLE I: Performance Results in a Known Scenario (Maze) and Three Unseen Environments (Column, Forest, SubT Cave)

	Maze		Column		Forest		SubT Cave	
Method	Success (%) ↑	Time Steps (-) ↓	Success (%) ↑	Time Steps (-) ↓	Success (%) ↑	Time Steps (-) ↓	Success (%) ↑	Time Steps (-) ↓
DQN	0.72 ± 0.04	252.98 ± 10.46	0.95 ± 0.03	151.49 ± 12.30	$\mathbf{0.97 \pm 0.01}$	$\mathbf{167.00 \pm 6.74}$	0.38 ± 0.03	1061.35 ± 16.04
R2D2	0.70 ± 0.04	257.82 ± 10.85	$\mathbf{0.99 \pm 0.01}$	172.44 ± 1.59	0.91 ± 0.03	221.78 ± 10.78	0.64 ± 0.02	937.68 ± 11.76
SQN	$\mathbf{0.98 \pm 0.01}$	$\mathbf{181.10 \pm 5.18}$	$\mathbf{0.99 \pm 0.01}$	$\mathbf{148.67 \pm 2.24}$	0.93 ± 0.02	171.43 ± 6.17	$\mathbf{0.82 \pm 0.03}$	$\mathbf{767.51 \pm 32.30}$

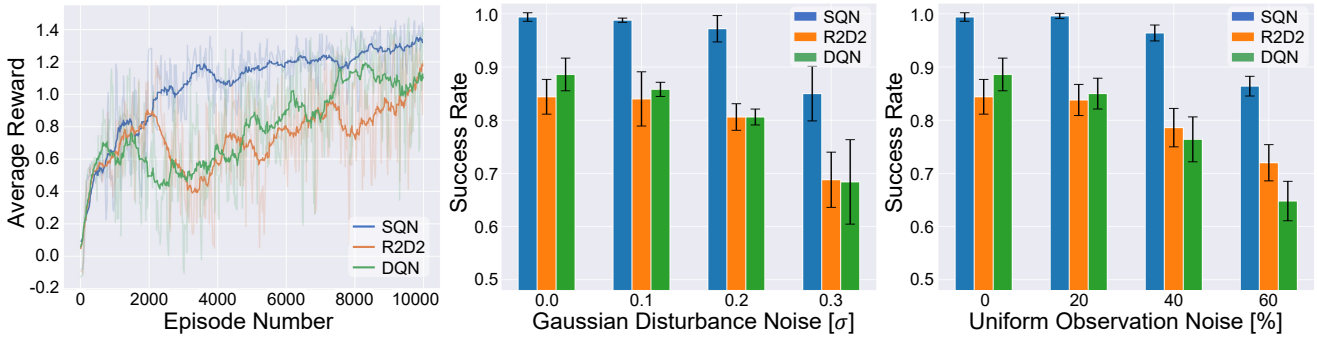

Fig. 6: **Left:** Learning curves of the average reward in evaluation. **Center:** Performance of different evaluation policies with external disturbance noise applied to the agent's motion. **Right:** Performance of the policies with uniform noise added to the agent's observations.

round-shaped obstacles in an unseen environment (17×17 m). The ego robot primarily receives range observations with a convex pattern. There is a wide open space in the center that was not observed in the training environment.

Forest: The Forest is also an open-space environment (17×17 m) and a new scenario with irregular asterisk-shaped obstacles. The ego robot primarily observes range measurements with a non-convex pattern. The asterisk-shaped terrain makes it easy for the ego agent to become trapped.

SubT Cave: This environment is an out-of-distribution, subterranean (SubT), large-scale (51×51 m), cave-like setting with numerous non-convex terrains [12]. The environment contains several forks that often lead the agent into local minimum situations or dead ends. With its realistic terrain, taking a wrong turn into a branch can significantly delay navigation toward the desired goal point. Considering that the spatial scale is three times larger than that of the other environments, we set the maximum number of time steps for each episode in this environment at 1200.

B. Robustness Settings

To evaluate the potential of our trained model for broader navigation tasks, such as navigating through noisy real-world environments, we assess the robustness of our agent by testing its ability to generalize across various modified environments. We deploy the model, trained in the Maze environment, to the following settings without additional retraining: 1) a setting where external Gaussian disturbance is randomly applied to the robot agent's translational and rotational velocities, and 2) a setting where uniform noise is applied to the agent's range measurement observations.

C. Hyperparameter Configuration

We train the SQN model with the following hyperparameters: the batch size is 64, and the layer dimensions in the SQN are all 128. The target network is updated by Polyak updates with a coefficient $\tau = 0.002$. The initial epsilon value is $\epsilon = 0.4$ and linearly decreases to the terminal value of $\epsilon = 0.1$. When updating the network, we sample the batch of 400-step sequence data and perform the *burn-in* of 10 steps for each sequence. We train the network for a maximum of 10,000 episodes. Considering the complexity of the environment and the number of expected strategies, we set the number of latent skill policy modules to 2 ($N = 2$).

D. Baseline Models

- **DQN:** Consistent with the methods in various studies [1], [17], [24], this policy relies solely on current state inputs and employs the same discrete actions as SQN. To compensate for the network capacity difference compared to SQN, which consists of one decision module and two skill policy modules, we increased the size of the DQN network layer after the feature extraction module to three times that of SQN.
- **R2D2:** Similar to previous studies [3], [4], [18], this method incorporates a recurrent layer to capture temporal features. It shares the same backbone network structure as SQN's GRU-based perception module, followed by a single Q-value head. R2D2 is trained following the methodology outlined in [23], using the same sequence length and burn-in steps as SQN.

V. Quantitative Evaluation

A. Learning Curves

Fig. 6 (Left) illustrates the learning curves of the average reward for each policy model. The results indicate that SQN outperforms R2D2 and DQN by 14% and 23%, respectively, achieving a final average reward of 1.35. R2D2 initially surpasses DQN, maintaining a higher performance until the

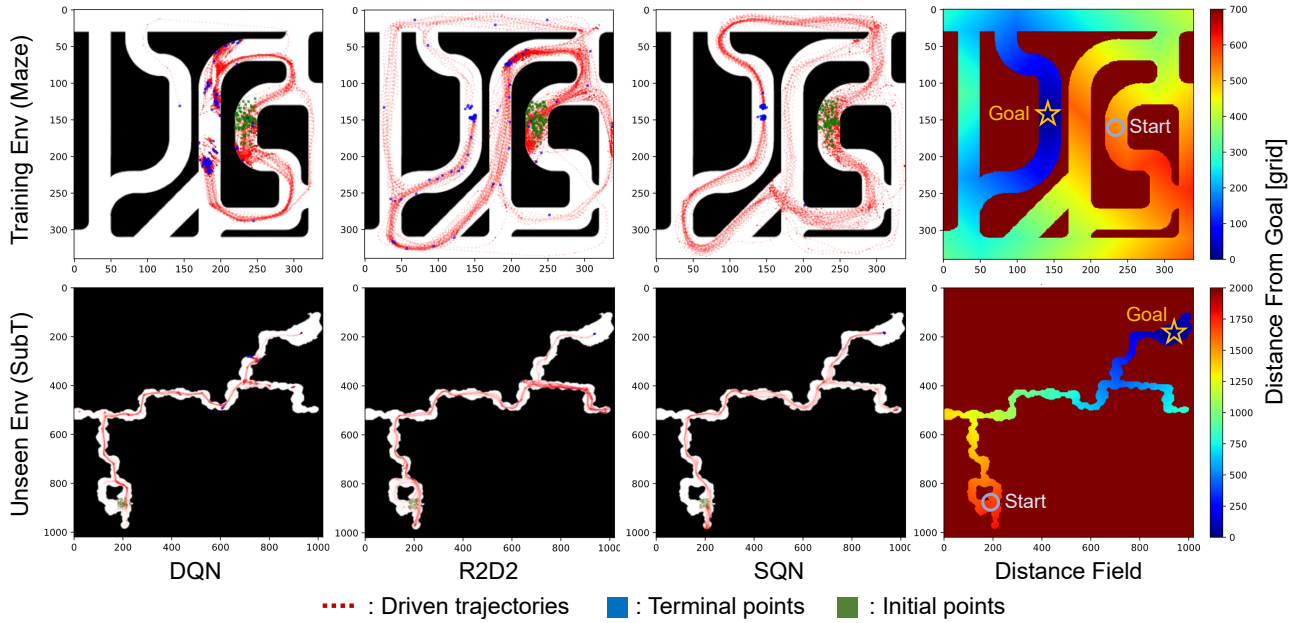

Fig. 7: Trajectory results of evaluation policies in the evaluation environments, with distance fields visualizing the true distance cost from the goal point.

first 2000 episode before experiencing a temporary decline. It recovers to a performance level of 1.18 after episode 3100. Similarly, DQN shows improvement until episode 1500, after which it faces a temporary drop, ultimately reaching a performance level of 1.10. The temporary performance drops observed in both models can be attributed to the evaluation scenarios, which feature routes prone to leading into local minimum regions before reaching the goal point. These drops can result from the models' predominant tendency to learn a single policy, causing temporary overfitting in areas characterized by local minima. Unlike these methods, our SQN model consistently demonstrates performance improvements without significant declines, benefiting from the adaptive ensemble of skill policies.

B. Performance Comparison

For the quantitative comparison, we evaluate two metrics: Success (success rate) and Time Steps (the number of steps taken in an episode). We assess the success rate of reaching the goal point for all policies across 100 episodes with 5 different seeds, as well as the number of steps taken to reach the goal. We then calculate the mean and standard deviation of these metrics. In the Column, Forest, and SubT Cave environments, characterized by OOD observations and scenarios distinct from Maze, we deploy evaluation policies via zero-shot transfer, without the need for additional training.

Table I summarizes the overall performance of the policy models. In challenging scenarios within the Maze environment, our SQN demonstrates superior navigation performance, achieving a success rate up to 40% higher (0.98) and requiring up to 42% fewer time steps (181.20 steps) compared to baseline models. Despite being trained in the Maze, the models still encounter challenges, mainly due to the numerous intersections between start and goal points. These challenges result in lower performance for the two baseline models equipped with a single policy network pipeline, with success rates around 0.70 and requiring over 250 steps to complete. In Column and Forest environments, where there are no dead ends between the start and goal, all the policies exhibit high success rates exceeding 0.90. In SubT Cave, characterized by uneven terrain requiring long-term exploration, the DQN, relying solely on current states, exhibits the lowest performance at 0.38. R2D2, utilizing sequential capabilities alongside exploration and reachability-based reward signals, achieves performance exceeding 0.50 in the novel environment but is capped at 0.64. In contrast, our SQN achieves remarkable zero-shot transfer performance of 0.82 even when deployed in scenarios with OOD range measurements, owing to its skill-based navigational policy.

C. Robustness

We analyze the robustness of policy models in the Maze environment under two sets of modified conditions. All experiments are conducted with 5 seeds, with each run consisting of 100 episodes to assess the success rate.

1) External Disturbance: Fig. 6 (Center) shows the performance variation of the evaluation policies when Gaussian disturbance noise with a mean of 0 is applied to the ego agent's translational and rotational velocity. We evaluate the policies across three settings ($\sigma = 0.1, 0.2, 0.3$) by incrementing the noise's standard deviation, σ, by 0.1. The results for the $\sigma = 0.0$ setting represent disturbance-free baseline performance. Our SQN closely matches the disturbance-free result up to the $\sigma = 0.2$ setting, demonstrating the highest success rate of 0.97. When subjected to a strong disturbance with $\sigma = 0.3$, our SQN maintains a high performance level of up to 0.85. In contrast, the two baseline models experience a significant drop in performance, falling below 0.70.

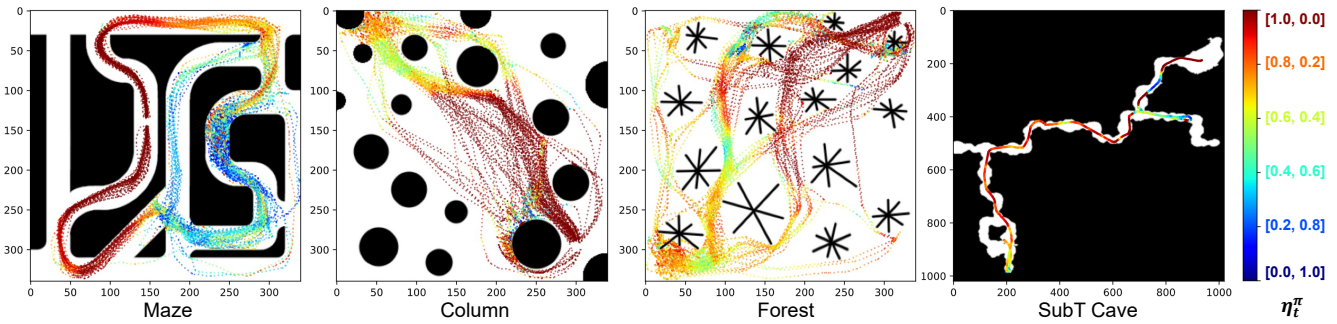

Fig. 8: Skill trajectories of SQN across various environments. For brevity, a single trajectory is represented in SubT Cave.

2) Observation Noise: Fig. 6 (Right) depicts the performance changes when different degrees of uniform noise are applied to the ego agent's range measurements. We increase the magnitude of the uniform noise by 20% and evaluate the policies at three different settings $(20\%, 40\%, 60\%)$, compared to the noise-free performance (0%). The result of SQN remains close to that of the noise-free setting up to a noise level of 40%, achieving the highest success rate of 0.96. In contrast, the other two baseline models exhibit a significant drop in performance as the magnitude of observation noise increases, marked by a large standard deviation in their performance results. SQN sustains a performance level exceeding 0.80 with a minimal deviation of 0.02, even at the highest noise level of 60%.

VI. Qualitative Analysis

A. Learned Navigation Behavior

Fig. 7 visualizes the navigation trajectories of the policies in the environments, Maze and SubT Cave, accompanied by the corresponding distance fields. These fields depict the actual distance to the goal point in each evaluation scenario.

Referring to the distance field, we observe that SQN primarily navigates towards regions that reduce the distance cost to the goal. This result demonstrates that SQN achieves near-optimal navigation performance in the complex Maze environment. In the case of DQN, the method lacks access to previously encountered features, leading it to navigate solely towards the goal point. This often results in the DQN becoming trapped in areas of local minima without reaching the goal. R2D2 can access the previous state features during navigation, achieving a higher success rate than DQN. However, with only a single Q-head, the model struggles to learn a policy that can handle the diverse intersections encountered en route to the goal. This often results in unnecessary revisits and failure to reach the goal point within the maximum allowable steps. Leveraging the skill decision, SQN can employ various skill policies to efficiently navigate complex topological regions en route to the goal point. It also demonstrates the ability to quickly detour and reach the goal, even after entering sub-optimal areas.

In the SubT Cave environment, which features unseen terrain observations, the performance gap between SQN with its adaptive skill ensemble mechanism and other baseline models is more pronounced. On this map, a fork near the (700, 400) location leads to two branches before the upper-right goal point. Choosing the wrong branch leads to a dead end, requiring a long-term return maneuver not encountered in the training scenarios. In such an environment, DQN suffers from the unstructured rough terrain near the fork, often leading to it getting stuck and failing to reach the goal point. R2D2, leveraging the ego robot's state sequence, is more proactive than DQN. However, its frequent oscillation between the fork and the dead end, along with occasional failures to exit the dead end, results in few successful trajectories to the goal. SQN, on the other hand, demonstrates efficient navigation with minimal back-and-forth maneuvers before arriving at the goal. It also performs swift return maneuvers to the fork without unnecessary revisits, even when encountering a dead end, thereby navigating towards the correct branch leading to the goal point.

B. Learned Skill Trajectories

To analyze how SQN utilizes its learned skills, we visualize the trajectories of the skill decision while navigating the four evaluation environments with fixed goal points (Fig. 8). The results show that our SQN can learn diverse combinations of latent low-level skills through adaptive skill ensemble. In the Maze environment, SQN initially produces skill decisions focusing more on the second latent skill policy, π_2. These decisions enable the agent to perform exploration maneuvers to escape from the initial spawn point (near the (235, 162) location), even as the Euclidean distance to the goal point increases. Upon reaching regions near the bottom-left or top-right corner, SQN shifts its emphasis to skill decisions that prioritize the first latent skill policy, π_1, facilitating goal-directed maneuvers. In the Column and Forest maps, when the agent is randomly initialized at the start point (Column: top-left, Forest: bottom-left), SQN generates decisions that balance the two skill policies. Once the route toward the goal becomes more straightforward, our method transitions to prioritizing the first skill, π_1. Furthermore, upon encountering obstacle areas near the goal point, it demonstrates adaptability by temporarily making decisions that highlight more on π_2 to navigate out of the blocked areas. In the SubT Cave episode, the ego agent infers decisions that give more weight to π_1 when navigating towards the goal point along uneven terrain. However, the agent adjusts its decisions to increase the weight on π_2 in situations where it

needs to return to the fork after taking the branch leading to a dead end. These results demonstrate that our SQN can learn an adaptive skill decision mechanism that ensembles latent low-level policies, effectively handling multiple local minima and sub-optimal situations encountered while navigating new environments.

VII. Conclusion

In this paper, we introduce Skill Q-Network, a policy network method based on skill ensemble mechanisms. We present a tailored reward function designed to learn exploration and goal-directed navigation strategies in mapless environments. Our proposed method demonstrates remarkable navigation performance compared with those of other baseline models in four complex unstructured scenarios. Furthermore, our empirical experiments demonstrate the adaptability and robustness of our method when transferred to novel OOD environments in a zero-shot manner. These environments include scenarios such as open spaces with non-convex obstacles or uneven terrain with multiple branches and dead ends. In future work, we aim to explore the capabilities of our skill ensemble mechanism to achieve not only observation-level generality but also dynamics feature-level versatility, leveraging domain randomization schemes [27], [28]. This extension will allow us to further investigate and address the challenge of robust sim-to-real transfer.

References

[1] O. Zhelo, J. Zhang, L. Tai, M. Liu, and W. Burgard, "Curiosity-driven exploration for mapless navigation with deep reinforcement learning," *arXiv preprint arXiv:1804.00456*, 2018. 1, 2, 5

[2] S. Lv, Y. Li, Q. Liu, J. Gao, X. Pang, and M. Chen, "A deep safe reinforcement learning approach for mapless navigation," in *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2021, pp. 1520–1525. 1, 2, 4

[3] M. Dobrevski and D. Skočaj, "Deep reinforcement learning for mapless goal-driven robot navigation," *International Journal of Advanced Robotic Systems*, vol. 18, no. 1, p. 1729881421992621, 2021. 1, 2, 5

[4] Z. Wei, W. Xiao, L. Yuan, T. Ran, J. Cui, and K. Lv, "Memory-based soft actor–critic with prioritized experience replay for autonomous navigation," *Intelligent Service Robotics*, pp. 1–10, 2024. 1, 2, 5

[5] X. Huang, Z. Li, Y. Xiang, Y. Ni, Y. Chi, Y. Li, L. Yang, X. B. Peng, and K. Sreenath, "Creating a dynamic quadrupedal robotic goalkeeper with reinforcement learning," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 2715–2722. 1, 2

[6] K. Lee, S. Kim, and J. Choi, "Adaptive and explainable deployment of navigation skills via hierarchical deep reinforcement learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 1673–1679. 1, 2

[7] C. Witting, M. Fehr, R. Bähnemann, H. Oleynikova, and R. Siegwart, "History-aware autonomous exploration in confined environments using mavs," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9. 2

[8] T. Dang, F. Mascarich, S. Khattak, C. Papachristos, and K. Alexis, "Graph-based path planning for autonomous robotic exploration in subterranean environments," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 3105–3112. 2

[9] B. Lindqvist, A.-A. Agha-Mohammadi, and G. Nikolakopoulos, "Exploration-rrt: A multi-objective path planning and exploration framework for unknown and unstructured environments," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3429–3435. 2

[10] F. Yang, D.-H. Lee, J. Keller, and S. Scherer, "Graph-based topological exploration planning in large-scale 3d environments," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 12 730–12 736. 2

[11] X. Chen, B. Zhou, J. Lin, Y. Zhang, F. Zhang, and S. Shen, "Fast 3d sparse topological skeleton graph generation for mobile robot global planning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 10 283–10 289. 2

[12] B. Kim, H. Seong, and D. H. Shim, "Topological exploration using segmented map with keyframe contribution in subterranean environments," *arXiv preprint arXiv:2309.08397*, 2023. 2, 4, 5

[13] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *2017 ieee international conference on robotics and automation (icra)*. IEEE, 2017, pp. 1527–1533. 2

[14] C.-Y. Tsai, H. Nisar, and Y.-C. Hu, "Mapless lidar navigation control of wheeled mobile robots based on deep imitation learning," *IEEE Access*, vol. 9, pp. 117 527–117 541, 2021. 2

[15] C. Yan, J. Qin, Q. Liu, Q. Ma, and Y. Kang, "Mapless navigation with safety-enhanced imitation learning," *IEEE Transactions on Industrial Electronics*, vol. 70, no. 7, pp. 7073–7081, 2022. 2

[16] T. Fan, X. Cheng, J. Pan, D. Manocha, and R. Yang, "Crowdmove: Autonomous mapless navigation in crowded scenarios," *arXiv preprint arXiv:1807.07870*, 2018. 2

[17] R. B. Grando, J. C. de Jesus, V. A. Kich, A. H. Kolling, N. P. Bortoluzzi, P. M. Pinheiro, A. A. Neto, and P. L. Drews, "Deep reinforcement learning for mapless navigation of a hybrid aerial underwater vehicle with medium transition," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1088–1094. 2, 4, 5

[18] X. Sun, Q. Zhang, Y. Wei, and M. Liu, "Risk-aware deep reinforcement learning for robot crowd navigation," *Electronics*, vol. 12, no. 23, p. 4744, 2023. 2, 5

[19] H.-C. Wang, S.-C. Huang, P.-J. Huang, K.-L. Wang, Y.-C. Teng, Y.-T. Ko, D. Jeon, and I.-C. Wu, "Curriculum reinforcement learning from avoiding collisions to navigating among movable obstacles in diverse environments," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2740–2747, 2023. 2, 4

[20] J. Wu and H. Li, "Deep ensemble reinforcement learning with multiple deep deterministic policy gradient algorithm," *Mathematical Problems in Engineering*, vol. 2020, pp. 1–12, 2020. 2

[21] C. Zhao, Y. Yan, X. Liu, S. Zhang, and B. Ouyang, "Learning adaptive mapless navigation skills through evidential deep learning," in *2023 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. IEEE, 2023, pp. 402–407. 2

[22] H. Seong and H. Shim, "Self-supervised interpretable sensorimotor learning via latent functional modularity," in *eXplainable AI approaches for Deep Reinforcement Learning*, 2024. 3

[23] S. Kapturowski, G. Ostrovski, J. Quan, R. Munos, and W. Dabney, "Recurrent experience replay in distributed reinforcement learning," in *International conference on learning representations*, 2018. 3, 5

[24] E. Marchesini and A. Farinelli, "Discrete deep reinforcement learning for mapless navigation," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 688–10 694. 4, 5

[25] I. Mordatch and P. Abbeel, "Emergence of grounded compositional language in multi-agent populations," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018. 4

[26] J. Terry, B. Black, N. Grammel, M. Jayakumar, A. Hari, R. Sullivan, L. S. Santos, C. Dieffendahl, C. Horsch, R. Perez-Vicente *et al.*, "Pettingzoo: Gym for multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 15 032–15 043, 2021. 4

[27] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810. 8

[28] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *Conference on Robot Learning*. PMLR, 2020, pp. 1162–1176. 8