2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

SF-ODNav: Successor Feature Framework for Map-less Target-Driven Outdoor Visual Navigation

Junzhe Wu, Jiaming Zhang, Tingrong Zhang, Ruining Tao, Huy Tran, Girish Chowdhary

Abstract— **Traditional deep reinforcement learning-based visual navigation techniques face challenges in dynamic and unstructured outdoor environments, particularly in the absence of high-resolution maps and GPS signals. This paper presents a deep reinforcement learning-based approach for target-driven visual navigation without explicit localization and mapping in outdoor settings, using the successor feature (SF) framework to enhance the model's transfer learning. This design enables effective knowledge transfer across tasks, allowing the model to adapt to novel environments with zero-shot or few-shot fine-tuning. To facilitate training and evaluation, we design grid-world environments constructed from real-world outdoor images, providing realistic yet controlled conditions for developing and testing deep reinforcement learning-based navigation. Experimental results demonstrate that our method can adapt effectively in outdoor environments, both within the same domain and across different domains. Moreover, despite being trained in a discrete grid-world setting, the model is successfully deployed in real time within the same area, maintaining robust performance and highlighting its strong transferability to continuous, real-world conditions.**

I. INTRODUCTION

Mobile robots have many potential applications in complex outdoor environments, such as agriculture and urban areas [1]. Autonomous navigation of such robots commonly relies on GNSS signals [2], [3], and onboard sensors such as IMU-based odometry, cameras, and LiDAR [4]–[7]. However, depending on GPS is often undesirable in real-world scenarios. For instance, agricultural robots can face localization problems caused by tall crops that block GPS signals [8], while self-driving cars in dense urban areas may encounter weak or unreliable signals in underground street networks [9]. SLAM-based systems [10], [11], which generate a map during exploration, provide an alternative, but they typically require substantial computation and often degrade in outdoor environments due to sparse or ambiguous visual features [12]. Enabling robots to autonomously localize and make motion decisions to reach their target location using only information detected by onboard sensors would greatly improve our ability to deploy mobile robots in such environments.

Deep reinforcement learning (DRL) has shown potential for navigation, with successful applications in indoor environments and outdoor urban areas with explicit landmarks [13], [14]. While some approaches still require map information during training, the actual navigation is performed end-to-end based on the robot's onboard perception [15], [16]. However, extending these methods to unstructured outdoor environments is more challenging, as navigation in such settings demands strong generalization due to their diversity and scale. Ideally, a robot should navigate in new environments with zero-shot or few-shot tuning, but achieving such generalization remains difficult, especially for visual navigation. To address this problem, we propose a goal-conditioned reinforcement learning (GCRL) method [17] that leverages the successor feature (SF) framework [18] to decouple environmental dynamics from rewards, enabling fast adaptation to novel environments. Although prior work has demonstrated the feasibility of SFs for visual navigation [19], [20], these efforts have primarily focused on indoor or simulated domains and provided limited insight into effective training strategies for real-world outdoor environments.

To summarize, the main contributions of this paper are: (1) Grid-world environments constructed from real-world outdoor images are introduced to support the training and evaluation of deep reinforcement learning algorithms for visual navigation. (2) SF-based methods are extended to real-world outdoor environments through systematic investigation of representation learning and SF training strategies. The introduced fully map-less visual navigation algorithm, which relies solely on goal images and current observations, achieves robust real-time deployment and demonstrates strong transferability to continuous outdoor conditions. (3) The transfer learning capacity of our method is validated through extensive experiments, highlighting robust performance across both intra-domain and cross-domain settings.

II. RELATED WORK

A. Goal-conditioned RL

Prior work in goal-conditioned reinforcement learning (GCRL) has largely focused on variants of goal-conditioned value functions. Universal Value Function Approximators (UVFAs) estimate cumulative rewards for arbitrary state–goal pairs [21]–[23]. Mapping State Space (MSS) [24] extends UVFAs to long-horizon tasks by combining them with graph-based planning, but only evaluated in low-dimensional state spaces. LEAP [25] employed goal-conditioned value functions to plan over latent subgoals but overlooked exploration. ARC [26], instead, proposes representations for measuring state similarity via maximum entropy policies, supporting exploration and long-horizon hierarchical RL. However, this approach requires access to

J. Wu and G. Chowdhary are with the Agricultural & Biological Engineering Department, J. Zhang is with the Mechanical Science & Engineering Department, T. Zhang and R. Tao are with the Computer Science Department, H. Tran is with the Aerospace Engineering Department, University of Illinois Urbana-Champaign, Champaign, IL 61820, USA.

This work was supported in part by ONR N00014-20-1-2249.

goal-conditioned policies, which is often impractical in large-scale environments.

B. Outdoor Visual-based Robot Navigation

Visual-based navigation algorithms for robots in outdoor environments have been actively studied over the years. Some recent work has demonstrated kilometer-scale visual navigation with DRL [27]–[29], but these methods often depend on high-resolution maps to localize the robot during execution. To avoid this, several methods propose to construct a topological graph during exploration, where nodes correspond to visual observations, and edges encode predicted traversability [30], [31]. While this eliminates the need for a pre-built map, it introduces other challenges: building and maintaining the graph online demands significant computational resources, especially in large-scale outdoor environments. Furthermore, the algorithm performance is closely related to the robot's online computation and perception capabilities, reducing flexibility and scalability. These limitations underscore the need for more lightweight, generalizable approaches to visual navigation.

C. Successor Features for Robot Navigation

One promising direction to address the above drawbacks is the use of successor features (SFs), which aim to decouple environment dynamics from task-specific reward functions. In standard deep Q-networks (DQNs) [32], the Q-function must be retrained whenever the task changes within the same environment, making adaptation inefficient. To address this, recent studies have explored SF-based methods for robot navigation with cross-target generalization capabilities [20], [33]. Others have further demonstrated the zero-shot navigation ability of SFs and their adaptation to slight task changes in different rooms [34]. Despite their theoretical appeal, most SF-based navigation methods remain limited to simulation, lacking real-world sensory data or physical deployment. Some efforts have extended SF-based navigation to real-world platforms, such as navigating maze-like environments without explicit localization or mapping [19]. However, these applications are typically confined to structured indoor settings where salient visual features are easily detectable. Extending such SF-based methods to real-world outdoor environments remains underexplored and demands further investigation.

III. BACKGROUND

A. Successor Features

SFs are an approach to reinforcement learning (RL) that allows one to represent a value function in a form that separates the dynamics of an environment from its rewards [18]. More specifically, given a feature mapping $\phi(s,a)$ and weight vector w, a linear reward can be defined as: $r(s,a) = \phi(s,a)^\top w$. The action-value function associated with policy π, $Q^\pi(s,a)$, can be then represented as:

$$\begin{aligned} Q^\pi(s,a) &= \mathrm{E}^\pi[r_{t+1} + \gamma r_{t+2} + \cdots \mid s_t = s, a_t = a] \\ &= \mathrm{E}^\pi\left[\phi_{t+1}^\top w + \gamma \phi_{t+2}^\top w + \cdots \mid s_t = s, a_t = a\right] \\ &= \mathrm{E}^\pi\left[\sum_{i=t}^{\infty} \gamma^{i-t} \phi_{i+1} \mid s_t = s, a_t = a\right]^\top w \\ &= \psi^\pi(s,a)^\top w \end{aligned} \tag{1}$$

where ψ^π are defined as the SFs, and can be thought of as a model that captures the dynamics induced by π in a given environment with the weights w capturing the associated rewards of that environment. SFs satisfy the Bellman equation as follows:

$$\psi^\pi(s_t, a_t) = \phi_t + \gamma E^\pi\left[\psi^\pi\left(s_{t+1}, \pi\left(s_{t+1}\right)\right)\right] \tag{2}$$

B. Successor Feature Similarity

Successor Feature Similarity (SFS) is a metric that represents the similarity between two state-action pairs through their SFs [20]. The Euclidean distance between two state vectors can represent the similarity between those states; however, it ignores the action's response and the dynamics of the environment. Motivated by this idea, SFS can be defined as the dot-product of the SFs of two state-action pairs:

$$\mathrm{SFS}^\pi\left(\left(s_1, a_1\right), \left(s_2, a_2\right)\right) = \psi^\pi\left(s_1, a_1\right)^\top \psi^\pi\left(s_2, a_2\right). \tag{3}$$

The SFS thus contains relevant long-term information between states that can aid in long-term decision making, even for states that were not encountered during training.

IV. METHODOLOGY

A. Grid-World Outdoor Environments

There are two outdoor test environments used in this study. Figure 1 shows the first one. The reachable zone of the test area is divided into 592 grids, each with a side length of 1.5 meters. The environment is made up of 46 rows, with each row containing 6 to 27 grids. There are observations in four directions for each grid, resulting in a total of 2,368 states for the environment. The second outdoor test environment consists of 32 rows, with rows 1–30 containing 27 grids each and rows 31–32 containing 16 grids each, yielding a total of 844 grids and 3376 states. Each state is represented by an image with resolution of 1280 x 480 pixels, resized to 256 x 96 during training to increase batch sizes. Data were collected by recording video in four directions while manually moving the robot back and forth along the center line of each row at a constant speed. The frames were selected at intervals of 1.5 meters based on GPS coordinates. The action space of this environment contains four actions: {moving forward 1.5m ($a = 0$), turning left 90° ($a = 1$), turning right 90° ($a = 3$), and staying still ($a = 2$)}.

B. Pretraining Protocol for the Encoder

The training of our agent is split into two steps. In the first step, ResNet34 [35] is used to extract state-related features $\phi(s)$ from the images, without considering the influence introduced by actions a. As shown in Figure 2, three loss

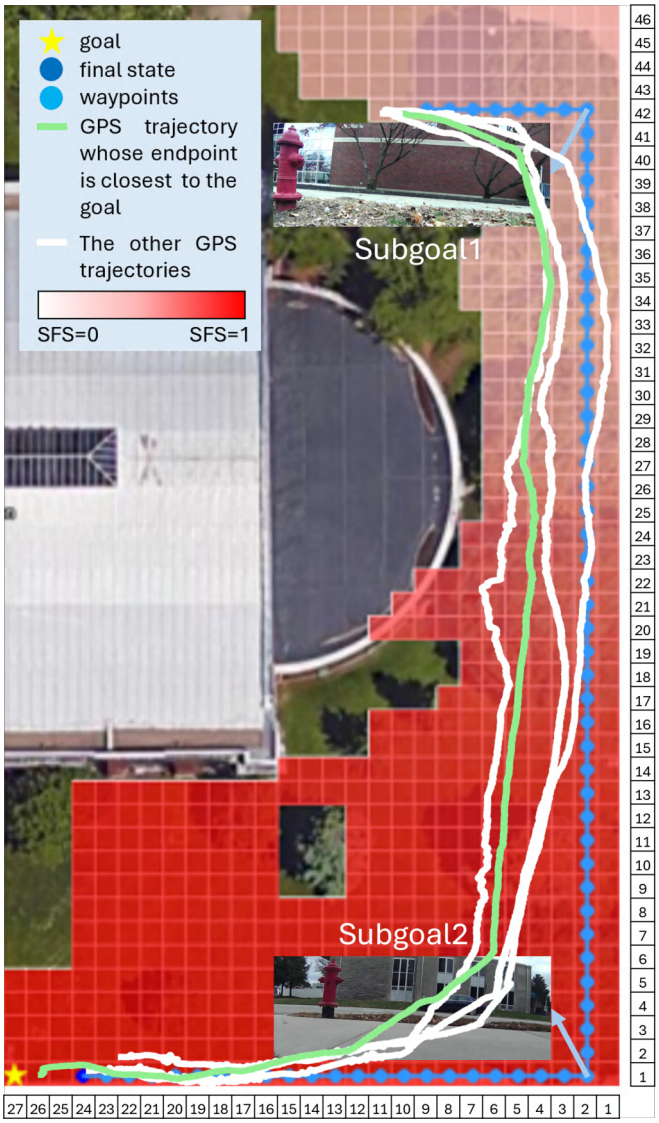


Fig. 1: Map of the first constructed grid-world environment showing SFSs for the marked goal and the longest successful trajectory generated by the trained goal-conditioned policy. The presence of fire hydrants at both turning points suggests that the trained goal-conditioned policy tends to select states with landmark objects as subgoals to enhance navigation success rates over long distances. The green and white lines represent the GPS trajectories of the robot during real-time outdoor execution of our algorithm, while the blue waypoints show the simulated grid-world trajectory.

functions are utilized to train all layers of encoder which are initialized from ImageNet:

1) Reconstruction Loss (L_R): Deconvolutional layers are applied to reconstruct the original image from the extracted 512-dimensional embedding vector. The network is trained with a mean squared error (MSE) loss.

2) Siamese Loss (L_S): Fully-connected layers ending with 2-way softmax are used to determine whether a pair of image observations is achievable within limited steps. The

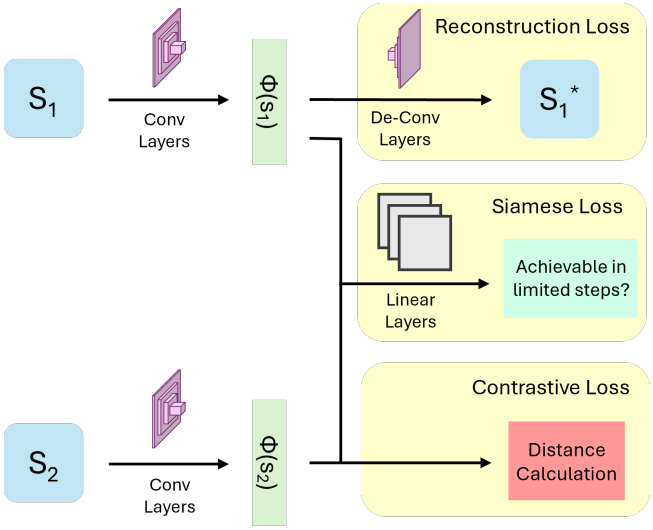


Fig. 2: Three losses applied in the encoder training.

network is trained with a cross-entropy loss.

To generate pairs (s_1, s_2, y) for this loss, each state is set as the origin and then 200 steps are executed. During each step, an action is randomly selected from the action space, excluding the "stay still" option. If an invalid action is encountered, a new action is randomly selected until a valid one is found. States generated within the first 10 steps from each starting point are considered achievable ($y = 0$), while those generated in the last 10 steps are considered unachievable ($y = 1$).

3) Contrastive Loss (L_C): For a pair of image observations, the following contrastive loss is used to minimize the embedding distance when they are achievable ($y = 0$) but maximize the distance otherwise ($y = 1$):

$$L_C\left(s_1, s_2, y, \theta_\phi\right) = \mathrm{E}\Big[(1-y)\left\{\max\left(0, D - m_1\right)\right\}^2 + y\left\{\max\left(0, m_2 - D\right)\right\}^2\Big], \tag{4}$$

where $D\left(s_1, s_2, \theta_\phi\right) = 1 - \cos(\phi(s_1), \phi(s_2))$ and the margin parameters $m_1 = 0.15$ and $m_2 = 1.15$ are defined as the lower bound distance for achievable and unachievable pairs. Compared to a traditional contrastive loss, a new margin m_1 is applied to prevent the embedding vectors of achievable pairs from becoming almost identical. Moreover, cosine distance is used instead of Euclidean distances since the cosine similarity computation uses less memory and usually produces better results [36].

$$L_{total} = L_R + L_S + L_C, \tag{5}$$

Three losses are added together for the training while a genetic algorithm [37] is used to search the hyperparameter space, which includes parameters for data augmentation, learning rate, end factor of linear learning rate scheduler, momentum, and weight decay. Our data augmentation process includes HSV color adjustment, translation, scaling, rotation, and cropping.

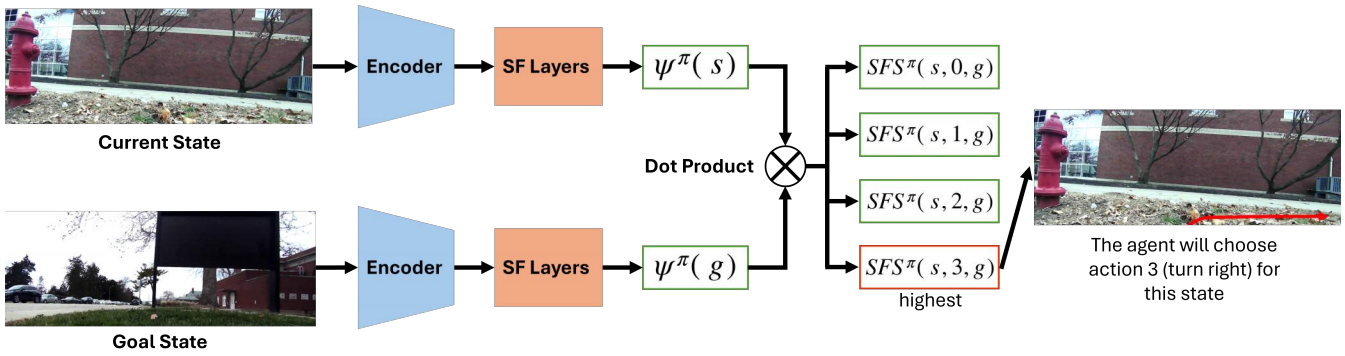


Fig. 3: Flow diagram of the goal-conditioned policy.

C. Deep Successor Feature Network (DSFN)

In the second step of training, a 3-layer fully-connected network is used to predict the SFs $\psi^{\pi}(s,a)$ from the embedding vector ϕ. The network is updated by minimizing the following temporal difference error:

$$L\left(s,a,s',\pi,\theta_{\psi}\right) = \\ \mathrm{E}\left[\left(\phi(s)+\gamma\widehat{\psi}^{\pi}\left(\phi\left(s'\right),\pi\left(s'\right)\right)-\psi^{\pi}(\phi(s),a)\right)^{2}\right] \tag{6}$$

where $\widehat{\psi}^{\pi}$ is a target network that is updated at fixed intervals for stability [32]. For the purpose of transferring knowledge, we want the trained SF function to focus on capturing the underlying dynamics of the environment, without any bias towards agent behaviors caused by a specific policy [20]. Therefore, π is set as a uniform random policy $\bar{\pi}$.

1) Multi-Step Learning: Traditional Q-learning typically derives the target value from the immediate reward and the estimated value of the next state. This approach tends to suffer from slow learning in the early stages, and the same limitation applies to SF-learning. To address this issue, we incorporate multi-step learning, which enables more accurate estimation of SFs in the early phase of training, thereby accelerating the learning process. We modify the loss from Equation (6) as follows:

$$L\left(s_0,a,s_1,s_2,s_3,\bar{\pi},\theta_{\psi}\right) = \\ \mathrm{E}\Big[\Big(\phi(s_0)+\gamma\phi(s_1)+\gamma^2\phi(s_2) \\ +\gamma^3\widehat{\psi}^{\bar{\pi}}\left(\phi\left(s_3\right),\bar{\pi}\left(s_3\right)\right)-\psi^{\bar{\pi}}(\phi(s_0),a)\Big)^{2}\Big] \tag{7}$$

2) Noise Perturbation: To deploy our algorithm in real-world scenarios, it is necessary to better simulate intermediate states between discrete states in our grid-world environment, as well as variations in lighting conditions. Therefore, the same data augmentations used in the encoder training are applied to states within the collected trajectories for SF training. In addition, Gaussian noise is added to the extracted state features to obtain a more accurate estimation of the SFs as follows:

$$\tilde{\phi}(s) = \phi(augment(s)) + \epsilon, \quad \epsilon \sim \mathcal{N}(0,I). \tag{8}$$

To generate trajectories for SF learning, the same method of generating pairs for the Siamese loss is applied, except that "stay still" is included in the action space. A genetic algorithm is again used to search the hyperparameter space, including learning rate, batch size, discount γ, $\widehat{\psi}^{\pi}$ update interval, and gradient clip.

D. Goal-Conditioned Policy

In the grid-world navigation task, the standard reward function assigns a penalty of -0.04 for each step taken and a reward of 1 for reaching the goal. However, this type of sparse reward function can lead to instability in normal Q-learning. To address this issue, instead of learning weights w to predict the sparse reward function in the first step of training, we can set weights w equal to the SFs of the goal state g with action $a = 2$ (stay still) [20] as follows:

$$r(s,a) = \phi(s,a)^{\top}w = \phi(s,a)^{\top}\psi^{\pi}(g,2). \tag{9}$$

This approach transforms the Q-function into the SFS between the current state s and the goal state g as follows:

$$Q^{\pi}(s,a) = \psi^{\pi}(s,a)^{\top}\psi^{\pi}(g,2) = \mathrm{SFS}^{\pi}((s,a),(g,2)). \tag{10}$$

A target-driven navigation policy π_{nav} can then be defined as $\pi_{nav} = \arg\max_a Q^{\pi}(s,a,g)$. This process is summarized in Figure 3.

V. EXPERIMENTS AND RESULTS

In our experiments, we initially assessed how well our agent could generalize to different goals within the same environment. We used two metrics: the overall success rate (OSR) and the path ratio (PR) to evaluate the performance of our method compared to several baselines. Subsequently, the agent's transfer learning capacity was examined through experiments involving both within-domain environments and cross-domain combinations. Finally, we also deployed our algorithm in the real-world to assess the transfer learning capacity of our method from a simulated grid-world to a real continuous state space.

A. Overall Success Rate (OSR)

Our OSR metric assumes that every state s in the environment $\mathcal{X}$ can be designated as the goal g, while the remaining

TABLE I: Comparison of algorithm performance.

Method	Environment1		Environment2	
	OSR	PR	OSR	PR
DQN-Finetune [32]	36.16%	1.944	41.36%	1.899
DSFN [19]	38.99%	1.691	42.27%	1.624
DSFN–Siamese [20]	47.65%	1.569	51.38%	1.514
SF-ODNav (ours)	**71.34%**	**1.308**	**90.06%**	**1.198**
SF-ODNav w/o Reconstruction Loss	68.76%	1.339	85.91%	1.231
SF-ODNav w/o Siamese Loss	65.82%	1.347	80.51%	1.294
SF-ODNav w/o Contrastive Loss	66.22%	1.326	81.48%	1.282
SF-ODNav w/o Multi-Step Learning	54.61%	1.511	71.16%	1.308
SF-ODNav w/o Perturbation	68.40%	1.322	87.14%	1.216

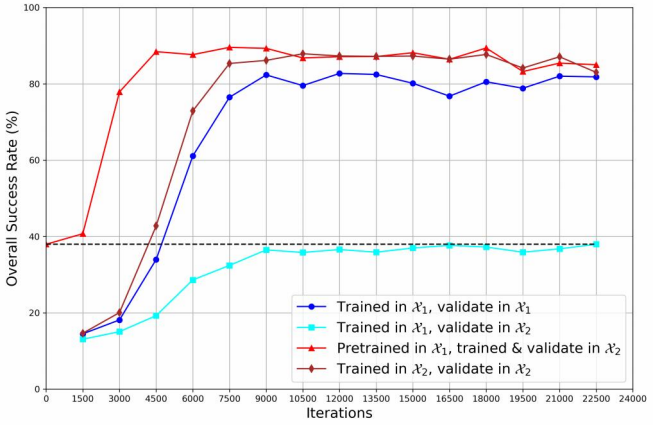


Fig. 4: Comparison of overall success rate between training and validating in different environments within the same domain.

states s' are set as initial states s_o. We then apply our goal-conditioned navigation policy π_{nav} to navigate from s_o to g for each pair (s_o, g). If the action "stay still" ($a = 2$) is selected, the navigation task is considered a "Success" if the distances of the x and y coordinates between the final state and the goal are both less than or equal to 3 grids. If the distance requirement is not met or an invalid action is chosen, the task is deemed a "Fail." The OSR is calculated as the number of successful tasks divided by the total number of pairs.

B. Path Ratio (PR)

For each successful navigation task, the optimal path between the initial state and the goal state is chosen by the A* planner, then the ratio of the actual path length to the optimal path length is calculated. This metric reflects whether the agent is actively navigating toward the target or merely reaching it by chance during random exploration.

C. Generalization to Different Goals

Table I summarizes the navigation performance of our method and various baselines in our two outdoor grid-world settings. Our baselines included Deep Q-Network (DQN) [32] finetuned for each task using the standard sparse reward function, Deep SF-Network (DSFN) trained with only a reconstruction loss [19], and DSFN trained with only a Siamese loss [20]. We see that our baselines exhibit low success rates and high path ratios, indicating a limited ability to produce goal-directed behavior and frequent inefficient paths. In contrast, our algorithm, SF outdoor navigation (SF-ODNav), outperforms the strongest baseline in both test environments, confirming that the proposed combination of architectural and training enhancements leads to robust and efficient navigation in complex outdoor environments. The longest path that successfully reached the goal in the first test environment, generated by our model, is shown in Figure 1.

The lower part of Table I shows results from an ablation study, where we consider SF-ODNav trained without various proposed components. We see that removing the contrastive loss or Siamese loss achieves a similar performance reduction, likely because cross-entropy minimization has been shown to be an approximation of a contrastive loss in low dimensions [38]. Compared with those two losses, training without the reconstruction loss shows a smaller impact on overall model performance, likely because the reconstruction loss primarily captures low-level visual features while neglecting relational similarities and dissimilarities between image pairs. Such relational cues are critical, as illustrated in Figure 1, where the SFS is higher when the agent state is closer to the goal and lower when it is further away. In contrast, both contrastive and cross-entropy losses explicitly aim to maximize mutual information between embedded features and their corresponding labels, thereby enhancing the discriminative capacity of the learned representations.

A comparison between models trained with and without multi-step learning shows that incorporating this technique significantly improves both success rate and path efficiency, highlighting the advantage of leveraging longer temporal dependencies in SF estimation. Beyond multi-step learning, we also examine the role of noise perturbation. Although it has little effect on navigation performance in the constructed grid-world environment, it plays a crucial role in real-world deployment. When encountering observations that differ significantly from the training states, models trained without noise perturbation often lose control and exit the test field, with failures occurring particularly during turning movements.

D. Transfer Learning within the Same Domain with a Fixed encoder

In order to test the transfer learning ability within the same domain, the first test environment $\mathcal{X}$ is divided into two parts: $\mathcal{X}_1$ (representing rows 1 to 14 from Figure 1, totaling 1320 states) and $\mathcal{X}_2$ (representing rows 15 to 46, totaling 1048 states). For this test, only the SF layers are re-trained, while the same encoder used in Table I is kept fixed to obtain the embedding vector ϕ.

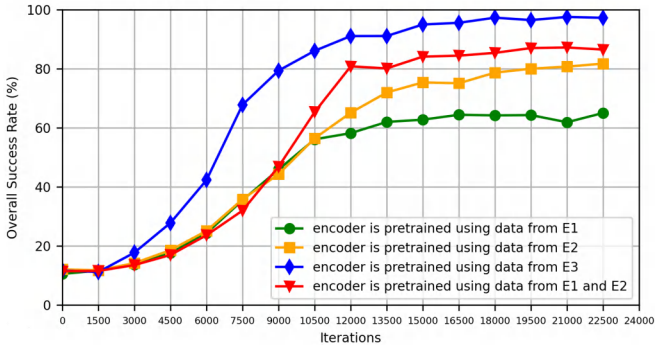


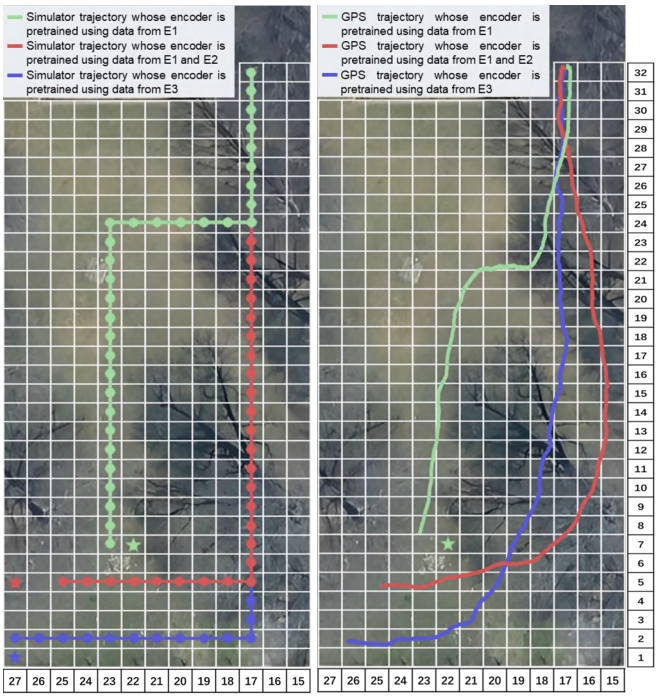
Fig. 5: Comparison of navigation performance in E2 with encoders trained in different environments.

As shown in Figure 4, the dark blue and light blue curves share the same weights trained in $\mathcal{X}_1$ at each interval, but the model is validated in both $\mathcal{X}_1$ and $\mathcal{X}_2$ to obtain different OSRs. From the light blue curve, we observe that our method struggles to transfer to a new environment within the same domain without additional re-training of the SF model. The red curve shows results for pretraining the SF model in $\mathcal{X}_1$ and then re-training it in the new environment $\mathcal{X}_2$. Compared to the brown curve, which did not use the pre-trained model for initialization, the red curve converges faster and achieves a higher OSR. The comparison shows that pretraining on the same domain could improve model performance. It is important to note that the OSR of the trained model typically increases when the environment becomes smaller, as the probability of successfully completing a navigation task increases when the origin and goal are closer in a smaller environment. This is why the OSRs in Figure 4 are higher than the values in Table I.

E. Transfer Learning across Different Domains with Different Encoders

To evaluate how an encoder pretrained in different environments impacts navigation in a new target environment, we designed experiments across three environments: the second test environment was split into two regions: the forest area (**E2**, columns 1–16, 2048 states) and the grassland area (**E3**, columns 17–27, 1328 states), in addition to the first test environment (**E1**). The trajectories generated in **E3** are used for training the SF layers and the overall navigation performance of models whose encoders were trained on data from different environments were then assessed in **E3**.

As illustrated in Figure 5, training the encoder directly in the target environment **E3** achieves the highest OSR, indicating that environment-specific visual cues are crucial for the encoder to extract informative features that support effective navigation. In contrast, training on **E1**, which shares a similar grassland terrain, leads to moderate performance improvements, suggesting that similarity in scene appearance can help transfer useful features to the target domain. The encoder trained on **E2** alone performs slightly better than **E1**, likely because both regions belong to the same overall domain, although the visual differences in forest textures still



(a) (b)

Fig. 6: Comparison of farthest successful navigation goals starting from same position of three models in E3. For easier comparison, columns 15-16 belonged to E2 are also put in this figure. Plot (a) illustrates the corresponding path generated in the simulator, and plot (b) shows the reproduced GPS trajectories recorded during real-time outdoor deployment.

limit full generalization. Combining data from **E1** and **E2** achieves a higher OSR than training on either environment alone, confirming that multi-domain training provides more diverse cues and improves overall robustness. Figure 6(a) illustrates the farthest goal positions in **E3** that the three models can successfully reach from the same starting point (top-right corner), along with their corresponding trajectories. This visualization more intuitively highlights the improvement gained by combining data from **E1** and **E2** compared to training only on **E1**. For this particular challenge, the environment is relatively simple, and the model trained on combined data from **E1** and **E2** performs similarly to the model trained solely on **E2**. Therefore, the latter result is not included in the figure. Overall, these findings highlight that pretraining the encoder on heterogeneous datasets enhances cross-domain generalization. Moreover, training only the SF layers (15 minutes) is substantially more efficient than retraining the full encoder for each environment (12 hours), making this approach both effective and computationally practical.

F. Real-World Deployment and Generalization Across Different Domains and Continuous State Spaces

To evaluate real-world applicability, the model was deployed in real time across different seasons within the same outdoor environment. To support continuous state spaces,

velocity-based commands were used instead of fixed-distance or fixed-angle instructions. Each command is defined as a tuple (x, y, θ), where x, y are normalized linear velocities and θ is the angular velocity around the vertical axis, all scaled by constant linear and angular speeds. The actions used include forward $(1, 0, 0)$, rotate left $(1, 0, -1)$, and rotate right $(1, 0, 1)$. The linear speed was set to $v = 0.5$ m/s and the angular speed to $\omega = 1.0$ rad/s.

As illustrated in Figure 1 and Figure 6(b), the GPS trajectories recorded during deployment closely follow the planned navigation path generated in the simulated grid-world environment, confirming that the robot was able to execute the learned policy with high spatial accuracy. At turning points, the robot selected a more optimal path than the original simulated trajectory. This demonstrates the robustness of the learned representation and its ability to generalize to continuous, real-world conditions without retraining.

VI. CONCLUSIONS

This paper presents a target-driven visual navigation approach for outdoor environments that does not rely on explicit localization or mapping. To enable effective training and evaluation, grid-world environments constructed from real-world outdoor images are developed, offering discrete yet realistic conditions for learning robust navigation policies. Complementing this environment, a carefully designed encoder is used to support visual representation learning, incorporating reconstruction, Siamese, and contrastive losses to extract task-relevant features from image observations. Building on these components, the proposed method leverages the successor feature (SF) framework within a goal-conditioned reinforcement learning (GCRL) policy. This design enables the decoupling of environment dynamics from task-specific rewards, thereby facilitating efficient knowledge transfer and enabling the agent to generalize effectively to new goals and unseen environments. Experimental results demonstrate that combining multi-step SF learning with noise perturbation significantly improves both navigation success rate and path efficiency. Furthermore, real-time deployment experiments confirm that the learned policy generalizes well across domain shifts.

Further research directions include studying alternative image encoders, such as the transformer, to better extract key features from images and conducting additional experiments to better understand the real-world transfer learning ability of the proposed method. More work on domain adaptation technology could also be investigated to improve the ability of transfer learning.

REFERENCES

[1] A. Ghobadpour, G. Monsalve, A. Cardenas, and H. Mousazadeh, "Off-road electric vehicles and autonomous robots in agricultural sector: trends, challenges, and opportunities," *Vehicles*, vol. 4, no. 3, pp. 843–864, 2022.

[2] S. Cebollada, L. Payá, M. Flores, A. Peidró, and O. Reinoso, "A state-of-the-art review on mobile robotics tasks using artificial intelligence and visual data," *Expert Systems with Applications*, vol. 167, p. 114195, 2021.

[3] H. Cui, J. Zhang, and W. R. Norris, "A real-time embedded drive-by-wire control module for self-driving cars with ros2," *International Journal of Mechatronics and Automation*, vol. 9, no. 2, pp. 61–71, 2022.

[4] C. Xu, Z. Liu, and Z. Li, "Robust visual-inertial navigation system for low precision sensors under indoor and outdoor environments," *Remote Sensing*, vol. 13, no. 4, p. 772, 2021.

[5] J. Zhang, H. Cui, J. Wu, and W. R. Norris, "Simulator and transplantable control architecture development for an electric autonomous vehicle," in *2024 IEEE International Conference on Mechatronics and Automation (ICMA)*, pp. 275–280, IEEE, 2024.

[6] C. Tao, S. Cheng, Y. Zhao, F. Wang, and N. Hovakimyan, "An optimization-based planner with b-spline parameterized continuous-time reference signals," *arXiv preprint arXiv:2404.00133*, 2024.

[7] J. Liang, P. Gao, X. Xiao, A. J. Sathyamoorthy, M. Elnoor, M. C. Lin, and D. Manocha, "Mtg: Mapless trajectory generator with traversability coverage for outdoor navigation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2396–2402, IEEE, 2024.

[8] K. Baxevani, I. Yadav, Y. Yang, M. Sebok, H. G. Tanner, and G. Huang, "Resilient ground vehicle autonomous navigation in gps-denied environments," *Guidance, Navigation and Control*, vol. 2, no. 04, p. 2250020, 2022.

[9] Q. Luo, Y. Cao, J. Liu, and A. Benslimane, "Localization and navigation in autonomous driving: Threats and countermeasures," *IEEE Wireless Communications*, vol. 26, no. 4, pp. 38–45, 2019.

[10] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *Journal of intelligent and robotic systems*, vol. 53, pp. 263–296, 2008.

[11] G. Ren, C. Ai, Q. Xu, Z. Wang, Z. Wang, and D. Geng, "Research on indoor and outdoor navigation technology based on the combination of differential gnss and lidar slam," in *2020 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pp. 134–139, IEEE, 2020.

[12] L. Yang and L. Wang, "A semantic slam-based dense mapping approach for large-scale dynamic outdoor environment," *Measurement*, vol. 204, p. 112001, 2022.

[13] Y. D. Yasuda, L. E. G. Martins, and F. A. Cappabianco, "Autonomous visual navigation for mobile robots: A systematic literature review," *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, pp. 1–34, 2020.

[14] J. Kulhánek, E. Derner, and R. Babuška, "Visual navigation in real-world indoor environments using end-to-end deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 3, pp. 4345–4352, 2021.

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

[16] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pp. 31–36, IEEE, 2017.

[17] M. Liu, M. Zhu, and W. Zhang, "Goal-conditioned reinforcement learning: Problems and solutions," *arXiv preprint arXiv:2201.08299*, 2022.

[18] A. Barreto, W. Dabney, R. Munos, J. J. Hunt, T. Schaul, H. P. van Hasselt, and D. Silver, "Successor features for transfer in reinforcement learning," *Advances in neural information processing systems*, vol. 30, 2017.

[19] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard, "Deep reinforcement learning with successor features for navigation across similar environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2371–2378, IEEE, 2017.

[20] C. Hoang, S. Sohn, J. Choi, W. Carvalho, and H. Lee, "Successor feature landmarks for long-horizon goal-conditioned reinforcement learning," *Advances in neural information processing systems*, vol. 34, pp. 26963–26975, 2021.

[21] A. W. Moore, L. Baird, and L. P. Kaelbling, "Multi-value-functions: E cient automatic action hierarchies for multiple goal mdps," in *Proceedings of the international joint conference on artificial intelligence*, pp. 1316–1323, 1999.

[22] T. Schaul, D. Horgan, K. Gregor, and D. Silver, "Universal value function approximators," in *International conference on machine learning*, pp. 1312–1320, PMLR, 2015.

[23] R. S. Sutton, J. Modayil, M. Delp, T. Degris, P. M. Pilarski, A. White, and D. Precup, "Horde: A scalable real-time architecture for learn-

ing knowledge from unsupervised sensorimotor interaction," in *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768, 2011.
[24] Z. Huang, F. Liu, and H. Su, "Mapping state space using landmarks for universal goal reaching," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[25] S. Nasiriany, V. Pong, S. Lin, and S. Levine, "Planning with goal-conditioned policies," *Advances in neural information processing systems*, vol. 32, 2019.
[26] D. Ghosh, A. Gupta, and S. Levine, "Learning actionable representations with goal-conditioned policies," *arXiv preprint arXiv:1811.07819*, 2018.
[27] H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine, "The ingredients of real-world robotic reinforcement learning," *arXiv preprint arXiv:2004.12570*, 2020.
[28] D. Shah, B. Eysenbach, G. Kahn, N. Rhinehart, and S. Levine, "Ving: Learning open-world navigation with visual goals," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13215–13222, IEEE, 2021.
[29] D. Shah and S. Levine, "Viking: Vision-based kilometer-scale navigation with geographic hints," *arXiv preprint arXiv:2202.11271*, 2022.
[30] D. Shah, A. Sridhar, N. Dashora, K. Stachowicz, K. Black, N. Hirose, and S. Levine, "Vint: A foundation model for visual navigation," *arXiv preprint arXiv:2306.14846*, 2023.
[31] A. Sridhar, D. Shah, C. Glossop, and S. Levine, "Nomad: Goal masked diffusion policies for navigation and exploration," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 63–70, IEEE, 2024.
[32] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[33] J. Hu, Y. Ma, H. Jiang, S. He, G. Liu, Q. Weng, and X. Zhu, "A new representation of universal successor features for enhancing the generalization of target-driven visual navigation," *IEEE Robotics and Automation Letters*, 2024.
[34] S. Siriwardhana, R. Weerasakera, D. J. Matthies, and S. Nanayakkara, "Vusfa: Variational universal successor features approximator to improve transfer drl for target driven visual navigation," *arXiv preprint arXiv:1908.06376*, 2019.
[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
[36] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 815–823, 2015.
[37] M. Mitchell, *An introduction to genetic algorithms*. MIT press, 1998.
[38] M. Boudiaf, J. Rony, I. M. Ziko, E. Granger, M. Pedersoli, P. Piantanida, and I. B. Ayed, "A unifying mutual information view of metric learning: cross-entropy vs. pairwise losses," in *European conference on computer vision*, pp. 548–564, Springer, 2020.