Visual Navigation With Multiple Goals Based on Deep Reinforcement Learning

Zhenhuan Rao, Yuechen Wu, Zifei Yang, Wei Zhang, Shijian Lu, Weizhi Lu, and ZhengJun Zha, *Member, IEEE*

***Abstract*— Learning to adapt to a series of different goals in visual navigation is challenging. In this work, we present a model-embedded actor–critic architecture for the multigoal visual navigation task. To enhance the task cooperation in multigoal learning, we introduce two new designs to the reinforcement learning scheme: *inverse dynamics model (InvDM)* and *multigoal colearning (MgCl)*. Specifically, InvDM is proposed to capture the navigation-relevant association between state and goal and provide additional training signals to relieve the sparse reward issue. MgCl aims at improving the sample efficiency and supports the agent to learn from unintentional positive experiences. Besides, to further improve the scene generalization capability of the agent, we present an enhanced navigation model that consists of two self-supervised auxiliary task modules. The first module, which is named path closed-loop detection, helps to understand whether the state has been experienced. The second one, namely the state-target matching module, tries to figure out the difference between state and goal. Extensive results on the interactive platform AI2-THOR demonstrate that the agent trained with the proposed method converges faster than state-of-the-art methods while owning good generalization capability. The video demonstration is available at https://vsislab.github.io/mgvn.**

***Index Terms*— Deep reinforcement learning, scene generalization, visual navigation.**

I. Introduction

VISUAL navigation is an intelligent capability to determine the current position and then to plan a path toward some goal location from the image or video inputs [1]–[4]. Due to the limitation of the camera's angle of view, it is hard to navigate with only visual inputs as the environment is partially observable. Inspired by the recent success of deep reinforcement learning in multiple fields, such as Atari games [5], computer Go [6], and robotic manipulation [7], the early efforts were made to train an agent to learn the skill of navigating to a specific goal [8], [9].

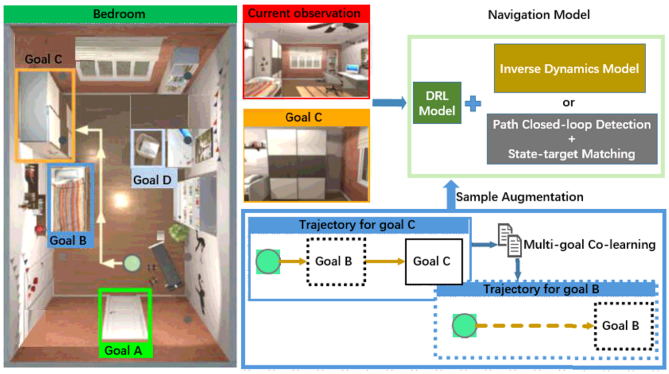


Fig. 1. Illustration of the proposed goal-dependent navigation model. Please enlarge to see details.

Compared to the regular navigation task, goal-dependent navigation is more challenging, which requires the agent to learn to adapt to a series of different goals. That is, after training, the agent is expected to own the capability of navigating to a series of different goals from a randomly sampled position and orientation, as shown in Fig. 1. Therefore, with this technique, it is unnecessary to retrain the model for different goals. There existed some pioneering works trying to solve the problem of goal-dependent navigation based on deep reinforcement learning [10]–[12]. For example, Zhu *et al.* [11] used a Siamese actor–critic architecture with shaped rewards to support navigating to different goals. Mirowski *et al.* [12] introduced an auxiliary task and used a curriculum training scheme to relieve the problem of sparse rewards in goal-dependent navigation tasks. Although these methods have achieved impressive performance, there remain some issues to be addressed for goal-dependent navigation from visual inputs. First, since the agent must learn to navigate to several different goals from a random state, the agent needs to learn the association between state and goal, as well as the association between state and action; second, the agent interacts with environments and generates a number of samples for each goal. However, the specific samples are only used to train their corresponding goals, which is a sample inefficient way to train the agent.

To relieve the above limitations, we present a new model-embedded actor–critic scheme that allows the agent to learn

Manuscript received June 16, 2020; revised January 4, 2021; accepted January 27, 2021. This work was supported in part by the National Key Research and Development Plan of China under Grant 2018AAA0102504; in part by the National Natural Science Foundation of China under Grant U1913204, Grant 61991411, and Grant U19B2038; in part by the Natural Science Foundation of Shandong Province for Distinguished Young Scholars under Grant ZR2020JQ29; and in part by the Shandong Major Scientific and Technological Innovation Project (MSTIP) under Grant 2018CXGC1503. *(Corresponding author: Wei Zhang.)*

Zhenhuan Rao, Yuechen Wu, Zifei Yang, and Weizhi Lu are with the School of Control Science and Engineering, Shandong University, Jinan 250061, China (e-mail: raozhenhuan@mail.sdu.edu.cn; wuyuechen@mail.sdu.edu.cn; zifei@mail.sdu.edu.cn; wzlu@sdu.edu.cn).

Wei Zhang is with the Institute of Brain and Brain-Inspired Science, Shandong University, Jinan 250061, China, and also with the School of Control Science and Engineering, Shandong University, Jinan 250061, China (e-mail: davidzhang@sdu.edu.cn).

Shijian Lu is with the School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798 (e-mail: shijian.lu@ntu.edu.sg).

ZhengJun Zha is with the School of Information Science and Technology, University of Science and Technology of China, Hefei 230026, China (e-mail: zhazj@ustc.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TNNLS.2021.3057424.

Digital Object Identifier 10.1109/TNNLS.2021.3057424

the skill of navigating to multiple goals cooperatively based on visual observation only. First, as shown in Fig. 1, we introduce an inverse dynamics model (InvDM) to the actor–critic architecture, which is trained as an auxiliary task of predicting the last action of the agent given its last and current state. The benefits of such a dynamics model are threefold.

1) The action could be considered as an appropriate criterion to distinguish the difference of the sequential states. After training, the inverse model could better predict the difference between the current state and goal, i.e., the navigation-relevant association between state and goal.
2) Since the auxiliary task of predicting the last action is trained by self-supervised learning, it could be leveraged as guidance to push the agent to explore more efficiently. Thus, the agent training could continue to develop in the absence of extrinsic rewards. That is, such auxiliary task could provide additional training signals to relieve the sparse reward issue, which is a common limitation shared by reinforcement learning methods.
3) Since a goal only changes the reward function instead of the transition structure of the Markov decision process (MDP), the dynamics model can be trained cooperatively by sharing the InvDM when training an agent to adapt to different navigation goals.

Furthermore, to improve the sample efficiency, we introduce a sample augmentation method named *multigoal colearning (MgCl)* that could make the agent learn from an unintentional positive experience when interacting with the environment. The idea can be explained by Fig. 1, where an agent has multiple navigation goals to learn. When the agent is trying to approach goal C, it may pass through goal B unintentionally. Thus, the generated trajectories for navigating to goal C are also valuable for learning to achieve goal B. That is, the samples generated for one goal could be used to train the agent for another navigation goal. Hence, the sample efficiency can be improved significantly. Experimental results show that the proposed architecture could accelerate the learning speed in goal-dependent navigation tasks and maintain robust as the number of the goals increases. Moreover, we allow the agent to learn multiple goals in multiple different environments simultaneously with only binary reward.

In addition to solving common problems in the training process of the visual navigation agent, the agent's generalization capability in different navigation scenes also needs to be considered. Although the goal-dependent visual navigation makes it possible that agents can execute different strategies for different goals, these goals must have been seen during the training stage. When facing an unknown scene or new target, the agent can hardly complete navigation tasks due to unfamiliar information (different positions, shapes, and decoration). To tackle multiscenes and multitarget visual navigation tasks, the generalization capability is necessary, which can help reduce training time and directly complete the navigation tasks without retraining in new scenes. If we can build an auxiliary task to extract navigation-related features, the strategies learned from it may have better generalization capability. Therefore, we design two modules: 1) the path closed-loop detection module, to make agents acquire the capability of judging whether the current state is the one that has been experienced and 2) the state-target matching module, to help the agent evaluate the degree of difference between the current state and the target state. These two modules make the model adaptive to different navigation tasks, proving to yield better generalization capability.

II. Related Work

There has been a number of methods focusing on the task of visual navigation [13]–[17]. Gupta *et al.* [18] designed a unified joint architecture for mapping and planning in unknown environments. Hussein *et al.* [19] introduced imitation learning into navigation task. Recently, deep reinforcement learning methods were introduced for navigation in simulated 3-D scenes [20]–[22]. However, due to the sparse reward signals in navigation tasks, the recent navigational agents seek aids from auxiliary supervised tasks for training [23], [24]. Mirowski *et al.* [8] proposed to train agent with auxiliary depth prediction and loop closure classification tasks. Banino *et al.* [9] introduced a grid cell network that was trained as an auxiliary task that enabled the agent to plan direct trajectories to the goals. All these demonstrated auxiliary tasks can be used to facilitate learning [25], [26]. Unlike the previous work relying on external signals to assist training, we advocate using the action signals to explore the difference between the current state and goal and introduce an auxiliary task of action prediction for training.

Recently, some efforts were made to handle the more challenging navigation tasks, the goal-dependent visual navigation [4]. Zhu *et al.* [11] used a feedforward Siamese actor–critic architecture incorporating a pretrained ResNet to support navigation to different goals in a discredited 3-D environment. Mirowski *et al.* [12] presented a dual-pathway agent architecture that enables the agent to navigate to a series of goals in a city-scale, real-world visual environment. Pathak *et al.* [27] presented a method to learn the trajectory between state and goal and then employed the expert demonstration to communicate the goal for navigation. Savinov *et al.* [14] proposed a semiparametric topological memory (SPTM) that is trained in supervised fashion and acts as a planning module in goal-directed navigation. Compared to these methods, we propose a self-supervised InvDM to better predict the difference between the current state and goal. Besides, a sample augmentation scheme, MgCl, is introduced to improve the sample efficiency and make the agent learn from unintentional positive experiences.

It is noted that some previous works have investigated the sample efficiency problem in reinforcement learning [28], [29]. For example, Andrychowicz *et al.* [30] presented a technique coined hindsight experience replay (HER) to improve the sample efficiency when learning multigoal policies [31]. Andrychowicz *et al.* [30] developed a hierarchical extension of HER in order to further speed up training. However, HER is a method that each trajectory is used to generate samples for arbitrarily assumed goals and the samples are stored in a replay buffer, which makes it only applicable to off-policy reinforcement learning scheme. By contrast, the proposed MgCl could train the generated samples immediately

without experience replay. Hence, it is suitable for online methods, such as A3C.

Generalization problem has received much attention in recent years from machine learning [32], [33], computer vision [34], [35], and robotics [36]–[38] communities. Lanctot *et al.* [39] pointed out that models based on reinforcement learning can easily overfit the training distribution. To alleviate the overfitting issue, Farebrother *et al.* [40] proposed to use the regularization in supervised learning, while Mankowitz *et al.* [41] leveraged robust optimization. Packer *et al.* [42] found that random noise such as parameter space noise or environmental randomization can be introduced into the training process to avoid overfitting. However, Zhang *et al.* [43] show that environmental randomization may work only in certain tasks, but not in others and even counterproductive. Hashemzadeh *et al.* [44] proposed model-based learning with subspaces method to achieve generalization and faster learning. However, the effectiveness of the above methods has only been verified in simple tasks. When faced with some complex problems, such as visual navigation, the number of samples required by the above methods increases greatly, and thus, it cannot converge.

III. Multigoal Visual Navigation Model

This section presents the details of the proposed architecture for the task of goal-dependent visual navigation. We propose a shared model and a colearning strategy for the same purpose: enhancing cooperation in multigoal learning.

A. Preliminaries

The goal-dependent reinforcement learning aims to train an agent interacting with an environment to maximize the expected future rewards [45] for a goal. It is concerned with policy optimization in an MDP, which is formalized as a tuple $\mathcal{M}(s, g, a, r, \gamma)$, where $\mathcal{S} = \{s\}$ denotes a set of finite states, $\mathcal{G} = \{g\}$ denotes a set of possible goals, $\mathcal{A} = \{a\}$ denotes a set of actions, r denotes the state-reward function, and $\gamma \in (0, 1]$ is a discount factor. The reward function $r_g(s, a, s')$ depends on the current goal and state. A stochastic policy $\pi(a|s, g)$ maps each pair of state and goal to an action and defines the behavior of the agent.

At each discrete time step t, the agent observes the state s_t and chooses an action a_t according to its policy $\pi(a_t|s_t, g_t)$. One time step later, the agent receives a numerical reward r_t and finds itself in a new state s_{t+1}. The process continues until the agent achieves the given goal. The return R_t is the total accumulated rewards from time step t. The aim of the agent is to learn an optimal control policy π, which maximizes its expected return until the goal is completed. A3C could update both the policy function $\pi(a_t|s_t, g_t; \theta_\pi)$ and the state-value function $V(s_t, g_t; \theta_v)$ by n-step returns. The policy and the value function are updated after every $t_{\max}$ actions or when a given goal is completed. The total accumulated return from time step t is defined as

$$R_t = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{k+t}, g_{k+t}; \theta_v) \tag{1}$$

where k may vary from state to state and is upper bounded by $t_{\max}$.

The entropy H of the policy π [46] is added to the objective function to alleviate premature convergence to suboptimal deterministic policies. The gradient of the full objective function can be regarded as

$$\nabla_{\theta_\pi} \log\pi(a_t|s_t, g_t; \theta_\pi)(R_t - V(s_t, g_t; \theta_v)) + \beta \nabla_{\theta_\pi} H(\pi(s_t, g_t; \theta_\pi)) \tag{2}$$

where β controls the strength of the entropy regularization term. The final gradient update rules are listed as follows:

$$\theta_\pi \leftarrow \theta_\pi + \eta \nabla_{\theta_\pi} \log\pi(a_t|s_t, g_t; \theta_\pi) \times (R_t - V(s_t, g_t; \theta_v)) + \beta \nabla_{\theta_\pi} H(\pi(s_t, g_t; \theta_\pi)) \tag{3}$$

$$\theta_v \leftarrow \theta_v + \eta \nabla_{\theta_v} (R_t - V(s_t, g_t; \theta_v))^2 \tag{4}$$

where η denotes the learning rate.

B. Network Architecture With Multigoal

As illustrated in Fig. 2, we design a new A3C-based deep model for the goal-dependent visual navigation task. The model takes the goal as part of the inputs and enables the agent to learn a series of different goals jointly. Meanwhile, the two pathways of the proposed model may learn two different types of feature representations: general and special. The general feature representation (denoted by green) depends only on the current observation and could be used to support cognitive processes, such as scene understanding. The special feature representation (denoted by red) relies on the current observation and the goal, which is beneficial to long-range planning.

The proposed model takes two inputs, observation of current state x_t and observation of goal x_g, and produces a probability distribution over the action space and a value function. The value function can represent the utility of any state s and given goal g. We train the proposed model through end-to-end deep reinforcement learning by incorporating an auxiliary objective. The aim of the training is to simultaneously maximize the cumulative reward using an actor–critic approach and minimize an auxiliary loss defined by the predicted last action and the true last action, as defined in Section III-C.

The details of the architecture are described as follows. First, a feature extractor consists of two convolutional layers and a fully connected layer. It is shared to process the state and goal images to produce the visual features f_s and f_g, respectively. The first convolutional layer has a kernel of size 8×8, a stride of 4×4, and 16 feature maps. The second layer has a kernel of size 4×4, a stride of 2×2, and 32 feature maps. The fully connected layer has 256 hidden units. Rectified linear units (ReLUs) separate the layers. Second, the visual feature of the state $f_s(X_t)$ is concatenated to the visual feature of goal $f_g(X_g)$. The fully connected hidden layer has 256 hidden units with ReLU and outputs hidden activation $h_a(f_s, f_g)$. The action predictor module $g_a(h_a)$ is a fully connected layer with 64 hidden units and could predict the last action a_{t-1} with a softmax layer. Final, the long short-term memory (LSTM) has 256 hidden units and outputs LSTM hidden activation $h_s(f_s)$.

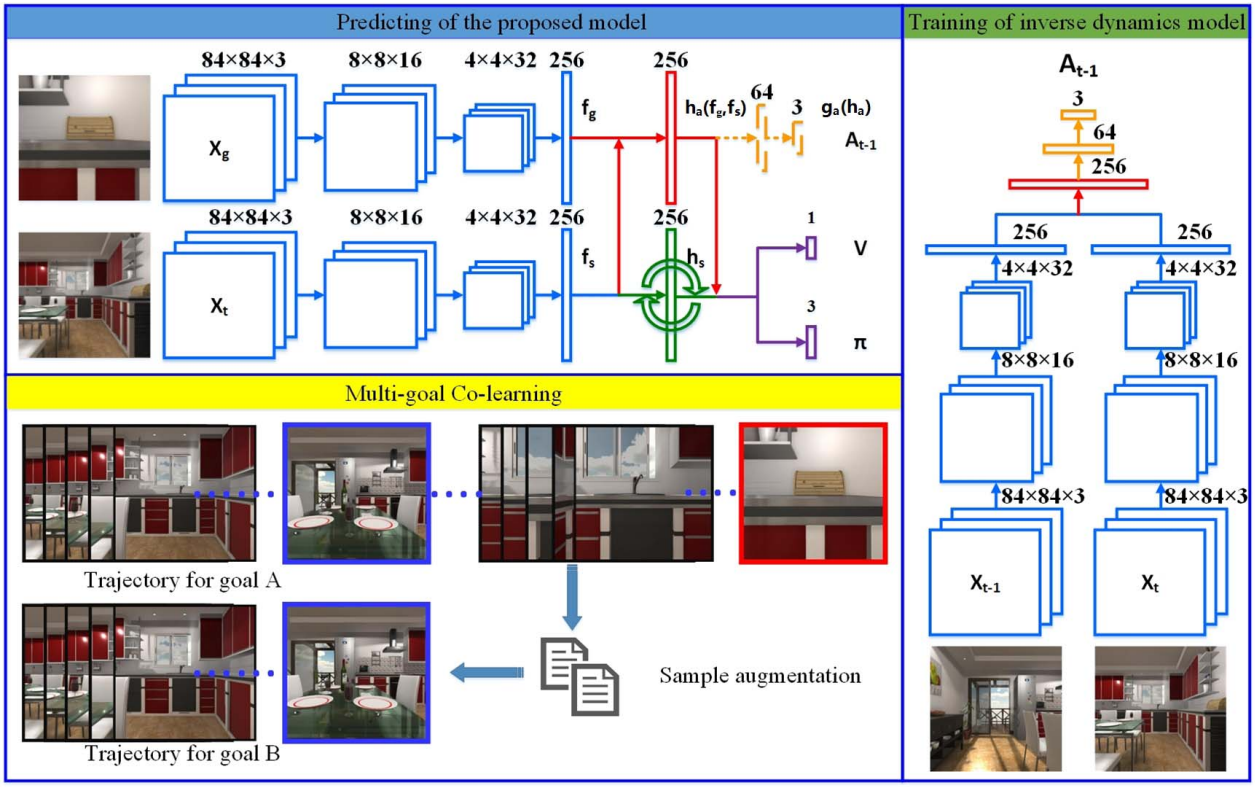


Fig. 2. Details of the proposed deep reinforcement learning model for goal-dependent navigation.

The hidden activation h_a is concatenated to the LSTM hidden activation h_s. The policy π is predicted with the softmax layer. The value function V is yielded by the fully connected layer.

C. Inverse Dynamics Model

For the visual navigation, if capturing the association between the current state and the goal, the agent could well address the interplay between planning and real-time action selection. Hence, as shown in Fig. 2, an InvDM is incorporated into the actor–critic architecture. The InvDM is trained as an auxiliary task of predicting the last action of the agent given its last state and current state. The action prediction could be rendered as an appropriate assessment to determine which is the critical differences between sequential states. Thus, after training, the navigation-relevant difference between the current state and the goal can be predicted by the InvDM, which is valuable for long-range planning.

In the implementation, the auxiliary task is trained in a self-supervised manner and could produce extra continuous gradients. Hence, the sparse reward issue in reinforcement learning could be relieved by such auxiliary tasks that could provide additional training signals. Meanwhile, as aforementioned, one goal only changes the reward function rather than the transition structure of MDP. Thus, the proposed InvDM could be shared and trained cooperatively across all goals. That is, the capability of navigating to one goal may benefit the learning of navigating to another goal by sharing the InvDM.

The training process of InvDM is illustrated in the right part of Fig. 2. It takes an observation of current state x_t and an observation of last state x_{t-1} as inputs and produces a probability distribution over the action prediction. The action is predicted with an extra optimization objective function defined by the cross-entropy classification loss as follows:

$$\mathcal{L}_a = -\sum_i a_i \cdot \log \bar{a}_i \tag{5}$$

where i denotes the index of action, and a and $\bar{a}$ represent the true action and predicted action, respectively.

D. Multigoal Colearning

For the goal-dependent navigation task, the agent interacts with the environment and generates a number of samples for each goal. However, as aforementioned, the generated samples are normally used to train a specific goal in the current methods, which is apparently sample-inefficient for agent training. Herein, we present a sample augmentation strategy *MgCl* to make the agent learn from unintentional yet useful experiences when interacting with the environment. That is, the generated samples of navigating to one goal may benefit the learning of navigating to the other goals. Hence, multiple goals could be learned by the agent cooperatively. Suppose that an agent has a number of goals $g_A, g_B, \ldots \in \mathcal{G} \subseteq \mathcal{S}$, each goal corresponds to a state, and only a goal-achieving reward $r_g(s, a, s' = r_g(s, a, g) = 1$ is provided. When the agent is trying to achieve g_A, it may have a state $s = g_B$ by chance. The reward for g_A is $r_{g_A}(s, a, g_B) = 0$ because the state $s = g_B$ is not the terminal state for g_A, whereas the MDP of navigating to g_A is the same to the MDP of going to g_B,

Algorithm 1 MgCl

Given asynchronous advantage actor-critic (A3C) and a set of goals $\mathcal{G}$.

Initialize:

- The global shared network parameter vectors θ and global shared counter $T = 0$,
- The thread-specific weights θ' and thread step counter $t \leftarrow 1$.

while $T \leqslant \mathrm{T}_{\max}$ **do**
  Reset gradients $d\theta \leftarrow 0$
  Synchronize weights $\theta' = \theta$
  $t_{start} = t$
  Get state s_t and goal g_t
  //sample
  while not terminal s_t **and** $t - t_{start} \neq t_{max}$ **do**
    Perform a_t according to policy $\pi(a_t|s_t, g_t; \theta')$
    Receive reward r_t and the new state s_{t+1}
    $g_{t+1} \leftarrow g_t$
    $t \leftarrow t + 1$
    $T \leftarrow T + 1$
  end
  //train for goal g_t

$$R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, g_t, \theta') & \text{for non-terminal } s_t \end{cases}$$

  for $i \in \{t-1, \ldots, t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$
    Accumulate gradients wrt $d\theta \leftarrow d\theta'$ using equ. 3, equ. 4 and equ. 5.
  end
  //train for other goal g'
  for $i \in \{t_{start}, \ldots, t\}$ **do**
    for $g' \in \mathcal{G}$ **do**
      if $g' == s_i$ **then**
        Set reward $r' \leftarrow r(s, a, g')$.
        Generate trajectory of goal g'.
        Accumulate gradients for goal g'.
      end
    end
  end
  Perform asynchronous updates of θ using $d\theta$ and η.
end

i.e., $\mathcal{M}(s, g_A, a, r_{g_A}, \gamma)$ and $\mathcal{M}(s, g_B, a, r_{g_B}, \gamma)$ are identical, except for the reward function. Thus, parts of the trajectory generated for g_A can also be served as the ones generated for g_B. Especially, since the reward of g_B in the state $s = g_B$ is $r_{g_B}(s, a, g_B) = 1$, it is quite useful for the agent to learn to achieve g_B. That is, the samples generated for g_A could be used to train the agent to navigate g_B.

However, it does not imply that additional trajectories would be generated certainly by the proposed MgCl. It takes effect when the agent drops into a state where another goal is achieved. Thus, as the agent becomes more skilled, the agent would navigate to the goal with more direct routes and hardly drop into other unintentional goals. That is, the proposed MgCl would contribute more at the beginning of training and gradually decrease as the training goes on. Meanwhile, for the goal-dependent visual navigation, not every state could be regarded as a goal or a terminal state, e.g., the white wall, a common interior structure, and has few discriminative features. Thus, it is meaningless to learn to navigate to it. More details of MgCl can be found in Algorithm 1.

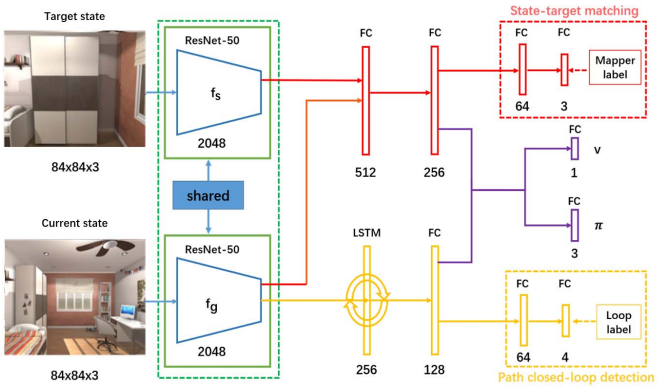


Fig. 3. Details of the proposed model for scene generalization of goal-dependent navigation.

IV. Enhanced Navigation Model for Generalization

Similar to the inverse model auxiliary task, the auxiliary task can help agents acquire good generalization capability while training. Hence, we propose two auxiliary modules to improve the agent's capability of scene generalization as follows.

A. Network Architecture for Scene Generalization

As illustrated in Fig. 3, we design a new module-based deep model for visual navigation. The model mainly consists of four parts: feature extraction module, modular auxiliary task module, A3C module, and value function module. We train the proposed model through end-to-end deep reinforcement learning combined with some modular auxiliary tasks. The aim of the training is to simultaneously maximize the cumulative reward using an actor–critic approach and minimize two auxiliary losses defined in (6) and (7). We use the alternate update to approximate the optimal solution. When calculating the two different auxiliary losses, respectively, the inputs are described in Sections IV-B and IV-C. When training via A3C, the proposed model has two inputs: an observation of current state x_t and observation of goal x_g. The outputs are a probability distribution over the action space and a value function.

The details of the architecture are described briefly as follows. First, the feature extractor consists of two ResNet-50 layers with shared weights. They process the current state and goal images, respectively, and generate the corresponding visual features f_s and f_g, both of which are 2048-D vectors. Second, the path closed-loop detection module includes a layer of LSTM network with 256 hidden units and a three-layer fully connected network. The numbers of hidden layers in the middle two layers of a fully connected network are 128 and 64, respectively. The activation function of the above

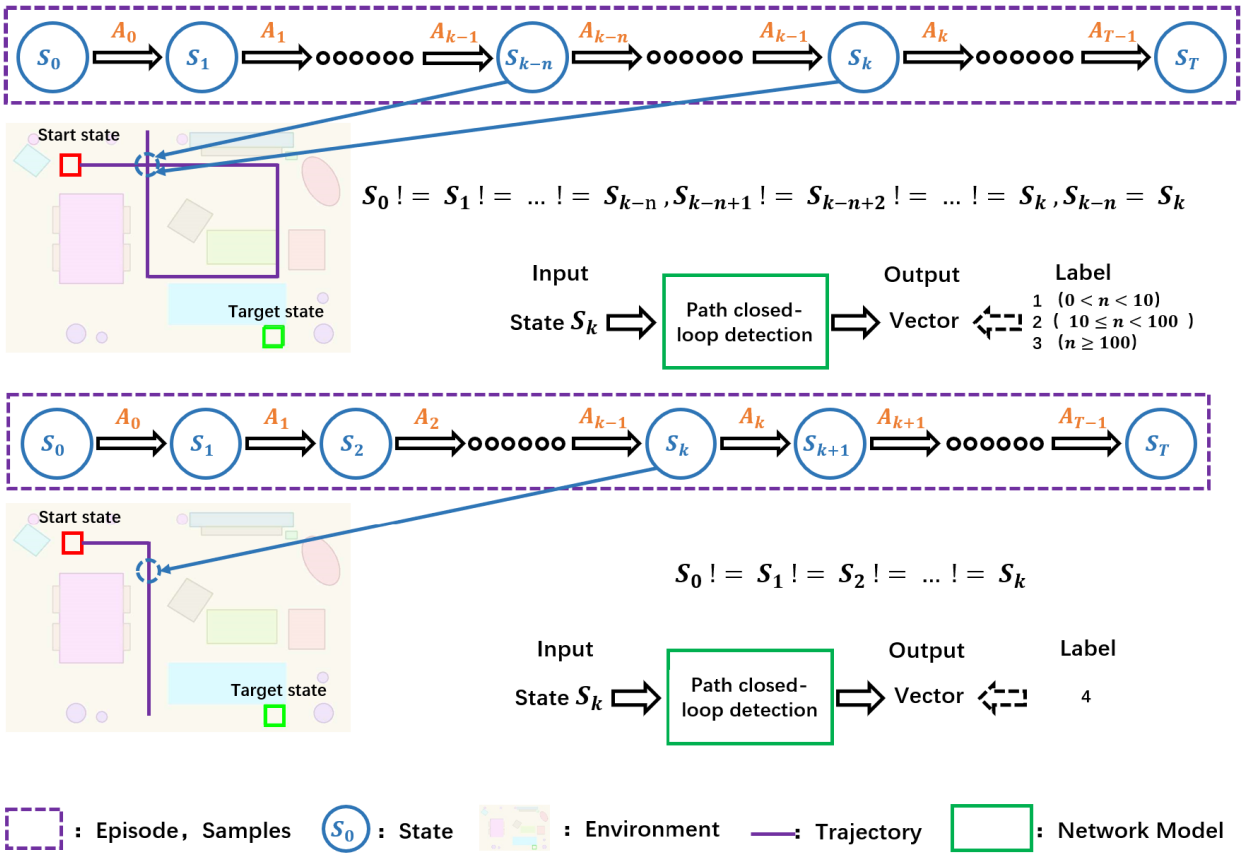


Fig. 4. Path closed-loop detection module workflow.

network is ReLu. The last output layer has four hidden units, and the activation function is softmax. Third, in the state-target matching module, the visual feature of the state $f_s(X_t)$ is concatenated to the visual feature of another state $f_g(X_g)$ and then used as the input of a four-layer fully connected network. The numbers of hidden units in this network are 512, 256, 64, and 3, respectively. The activation function of the first three layers is ReLu, and one of the last layers is softmax. For the strategy and the value function, we employ the LSTM network to deal with the current state's feature $f_s(X_t)$, pass through a full connection neural network with 128 hidden units, and, finally, produce an output vector. At the same time, the two feature vectors $f_s(X_t)$ and $f_g(X_g)$ are connected with a two-layer full connected network, which has 512 and 256 hidden units. The fully connected network produces another output vector. The two output vectors are further coupled to produce the final strategy (dimension 3) and value function (dimension 1). The activation function of the output layer of the strategy is softmax. The output layer of the value function has no activation function, and the others are ReLu layers.

B. Path Closed-Loop Detection Module

In visual navigation tasks, an episode often consists of hundreds or even thousands of steps, and some of which are repeated. For the optimal path to the target, it is often impossible to repeatedly reach or stay in a certain state. Determining whether a state has occurred during training is of great help for completing navigation tasks. Such capability could ensure that no matter how the navigation scene changes, each state occurs only once in the optimal path.

Based on this observation, we store all the samples and their sequence in the current episode of the training process. When entering a new state, we compare it with the samples in the memory to determine whether the agent has ever reached this state and the time of its previous visit. This information can be used as a training sample and label to train auxiliary tasks in a supervised manner.

Since the agent's repeated arrival to a certain state is largely related to the previous action sequence, the previous action sequence should also be used as part of the input. The sequence of actions grows dynamically with the length of the episode. A general forward neural network cannot process this variable sequence, so a recurrent neural network (RNN) is introduced to extract the temporal information. In addition, we hope that the closed-loop detection is only related to the state, and thus, the navigation target is excluded from the input information. With this insight, as shown in Fig. 4, we design a path closed-loop detection module to make the agent have the capability of distinguishing whether the current state has occurred or not.

The module is composed of an LSTM network and a three-layer fully connected network. Its input is the output feature

of the current state by the feature extractor. The intermediate output will also participate in the final strategy selection and value function estimation. The supervised learning method is used to train the module, and its label is the repeated information of the current state in the current episode. The specific labels are set as follows.

1) If the current state occurs more than once in the current episode and the time interval between the last repeated and the current state is less than 10 steps, the label is set to 1.
2) If the state occurs more than once and the time interval is between 10 and 100 steps, the label is set to 2.
3) If the state occurs more than once and the time interval is greater than 100, the label is set to 3.
4) If the current state is unique, the label is set to 4.

This is a supervised classification task with labels in four-value one-hot encoding. The loss function is defined by cross-entropy classification loss as follows:

$$\mathcal{L}_b = -\sum_i b_i \cdot \log \bar{b}_i \tag{6}$$

where i denotes the index of current state category, and a and $\bar{a}$ represent the true category and predicted category, respectively.

C. State-Target Matching Module

By matching the current state with the target state, the agents could judge whether the current state is approaching the target step by step and determine the next action. This implies that how to evaluate the proximity between the current state and the target is the key to navigation success. Therefore, a state-target matching module can be trained to evaluate the similarity between the two pictures, so as to help the agent identify the distance between the current state and the target. Such capability is independent of the environment and the target and also provides the possibility to improve the scene generalization capability of agents in navigation.

As shown in Fig. 5, we employ the auxiliary task to embed a state-target matching module into the model. In an episode, the closer the two samples are, and the higher their similarity is. Therefore, the state and the distance between states in the episode stored in memory can be used as training data. The input of the model should contain two states of an episode (maybe the same state), and the output label is the distance between these two states in the episode. This structure also does not require additional signals from the environment.

We cast it as a classification task via supervised learning to train this module. The training data and labels come from the memory for the episode. At each training step, the data include the two states sampled from the same episode and their interval in the episode as the distance label. The details are as follows.

1) If the distance between the two states is less than 10, the label is set to 1.
2) If the distance between the two states is greater than or equal to 10 and less than 30, the label is set to 2.

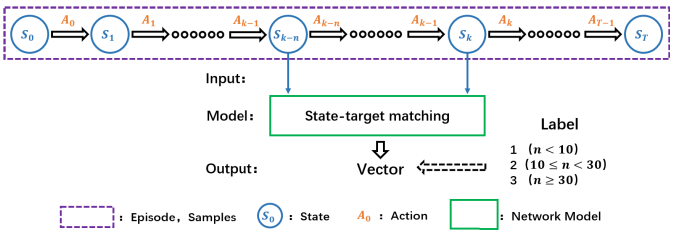


Fig. 5. State-target matching module workflow.

3) If the distance between the two states is greater than or equal to 30, the label is set to 3.

The labels are one-hot-encoded 1×3 vector. The loss function is defined by cross-entropy classification loss as follows:

$$\mathcal{L}_c = -\sum_i c_i \cdot \log \bar{c}_i \tag{7}$$

where i denotes the index of category, and a and $\bar{a}$ represent the true category and the predicted category, respectively.

V. Experimental Results

This section tests the proposed method on *AI2-THOR* [47], which is an open-source set within the Unity3D game engine, and enables navigation in a set of near-photorealistic indoor scenes. Here, we choose four different categories of scenes: *Bathroom*, *Bedroom*, *Kitchen*, and *Living room* to illustrate the navigation performance. It is worth noting that the size of *Bathroom* is the smallest, while the *Kitchen* is the biggest scene. One example of the *bedroom* scenes that the agent can navigate in and interact with is shown in Fig. 1. The detailed settings of the environment in the experiments are given as follows.

1) *Action Space:* There are three actions to learn: moving forward, turning left, and turning right. The move action has a constant step length (0.5 m), and the turn action has a constant angle (90°). The scene space is discretized into a grid-world representation.
2) *Observations and Goals:* Both observations and goals are images taken in the first-person view. The actual inputs to the agent are 84×84 images downsampled from the original size 300×300. Given a target image, the goal is to navigate to the location and viewpoint where the target image is taken.
3) *Reward Design:* The environments only provide a goal-achieving reward (1.0) upon task completion.

In the implementation, we set the discount factor $\gamma = 0.99$, the RMSProp decay factor $\alpha = 0.99$, the exploration rate $\epsilon = 0.1$, and the entropy regularization term $\beta = 0.01$. Besides, we used 16 threads and performed updates after every five actions (i.e., $t_{\max} = 5$). To relieve the bias behavior, all the goals and scenes were trained, in turn, in each thread in the experiments. The performance could be evaluated quantitatively in terms of the number of episodes that terminate in every 2000 frames over all goals. For each goal, we randomly initialize the starting location of the agent, and the episodes will not terminate until the agent reaches the goal. Hence, the number of episodes may indicate how many times the

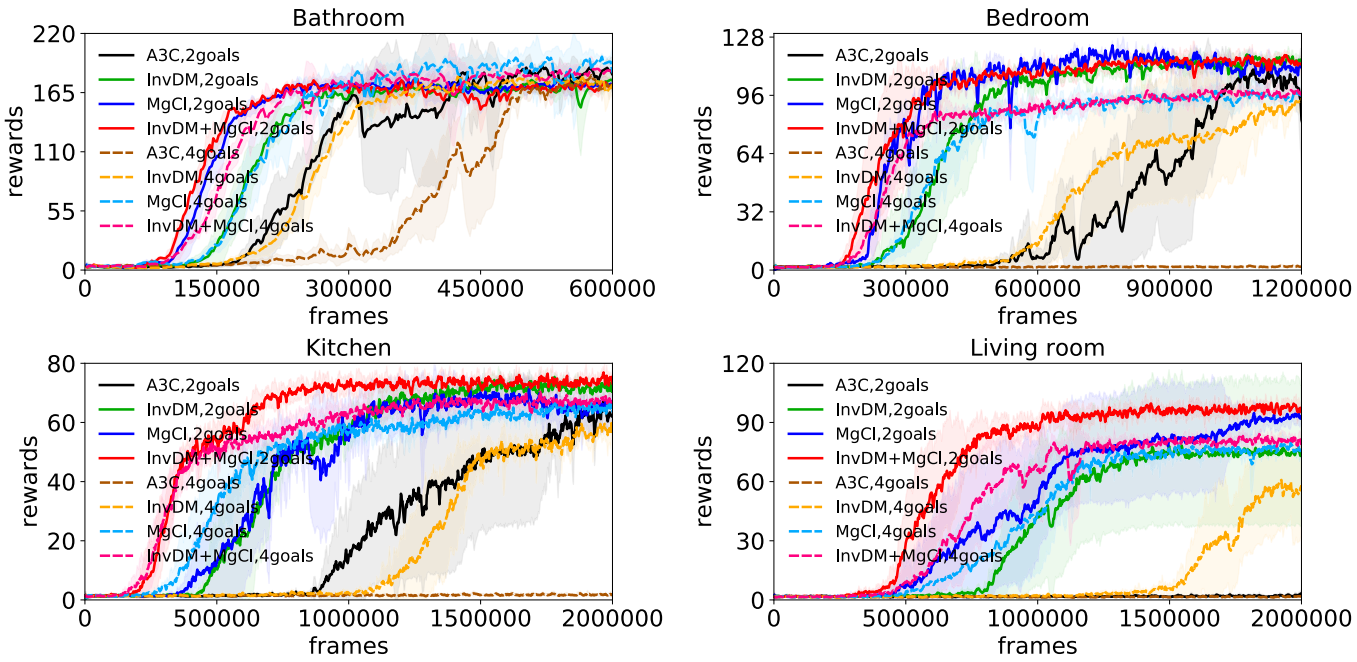


Fig. 6. Testing of our model in different scenes. Due to space limit, "A3C + InvDM" is abbreviated to "InvDM," so do the other terms.

agent reaches the goal, which could indicate how well the agent is trained. Each test was repeated five times with different seeds. We report the average final rewards and plot the mean and standard deviation of the reward statistic.

A. Ablation Study of the Inverse Dynamics Model

To show the benefit of the proposed InvDM for the goal-dependent visual navigation, an ablation study is made between A3C and A3C + InvDM (abbreviated to "InvDM" in Fig. 6) in the scenes of *Bathroom*, *Bedroom*, *Kitchen*, and *Living room* with different number of goals, i.e., two goals and four goals. It is noted that the A3C here is a goal-dependent variant of the standard baseline [46]. The network architectures of A3C and A3C + InvDM are identical except that A3c + InvDM is trained with an additional auxiliary loss defined by (5).

The learning curves in Fig. 6 show that A3C + InvDM obtained better performance than A3C in all cases, in terms of both convergence speed and rewards. This might because InvDM can well capture the navigation-relevant association between state and goal and encourage cooperation when training an agent to multiple goals. Meanwhile, the faster convergence speed may also attribute to the denser training signals offered by InvDM, which could relieve the common sparse reward issue in reinforcement learning.

Besides, it is also found that the improvement of A3C + InvDM over A3C in *Kitchen* is more obvious than that in *Bathroom*. As *Kitchen* is bigger and more clustered than *Bathroom*, such result implies that InvDM has good potential in challenging scenes. In addition, as the number of navigation goals increases, A3C + InvDM maintains good performance, while A3C drops rapidly. Hence, the robustness of InvDM with respect to the number of goals could be proved.

B. Ablation Study of the Multigoal Colearning

A similar study was conducted to investigate the effect of MgCl for goal-dependent visual navigation tasks. We can observe that A3C + MgCl converges faster than A3C while achieving better final performance in all cases. Taking the *Kitchen* with four goals, for example, A3C + MgCl produced the performance of about 58 rewards after training with 1.25 million frames, while A3C has about two rewards. Comparing *Bathroom* with *Kitchen*, the advantage of A3C + MgCl over A3C in *Kitchen* is also more significant than that in *Bathroom*, which indicates that MgCl could improve the agent training in complex scenes. In addition, similar to InvDM, MgCl could help A3C maintain stable performance as the number of goals increases. This proves that MgCl makes the training of agents benefit from the training of agents with other navigation goals, which finally leads to the improvement in sample efficiency.

Moreover, to further prove the benefit of the combination of InvDM and MgCl, we compare the performance of InvDM, MgCl, and InvDM + MgCl in the scenes of *Bathroom*, *Bedroom*, *Kitchen*, and *Living room* with two goals and four goals, respectively. As shown in Fig. 6, InvDM + MgCl significantly outperforms InvDM and MgCl in all scenes, in terms of both convergence speed and rewards.

C. Results of Learning in Multiple Scenes

In this section, the proposed method was tested to learn multiple goals in different scenes simultaneously. Similarly, the experiments were conducted in the four typical scenes of *Bathroom*, *Bedroom*, *Kitchen*, and *Living room* with two goals and four goals, respectively. Different scenes may have different environmental dynamics. Thus, when learning in multiple scenes at the same time, the network parameters

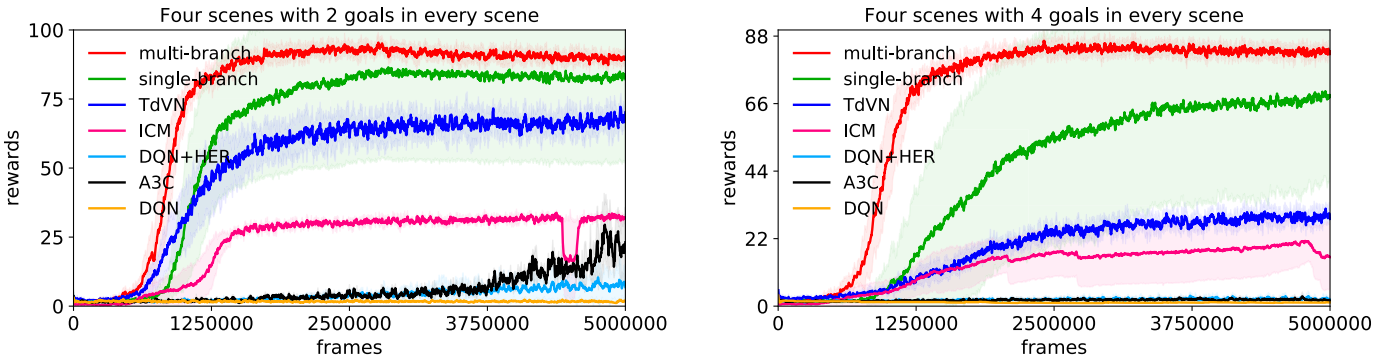


Fig. 7. Comparison to state-of-the-art methods in multiple scenes with different number of goals.

of InvDM could be trained and shared in two manners: full share with a single-branch model and partial share with a multibranch model, as described in the following.

1) *Single-Branch:* All parameters of InvDM are shared by all scenes, i.e., full share.
2) *Multibranch:* All parameters of InvDM are shared, except the fully connected layers, i.e., partial share.

We compare the proposed two models in terms of convergence speed and rewards. From Fig. 7, the multibranch model outperforms the single-branch one in both two indexes in the two different configurations. We hypothesize that the partial share scheme makes it possible to capture the particular characteristics in a scene that may vary across scene instances. It can also be found that the multibranch model is robust to the increasing of the number of scenes, indicating that it is easy to apply our method to the situation with more scenes.

Furthermore, we compare the proposed model with state-of-the-art methods, such as A3C [46], DQN [5]va, ICM [21], TdVN [11], and HER [30]. As shown in Fig. 7, DQN and A3C suffer from slow convergence. It seems difficult for them to train with sparse and binary rewards. DQN + HER also performed poorly though HER could improve sample efficiency. ICM leads to a better performance than the three methods above, as ICM trains an agent with intrinsic curiosity-based motivation and, thus, offers an efficient way for exploring. TdVN showed the second best performance, benefiting from its complicated deep Siamese actor–critic network architecture incorporating a pretrained ResNet. With the proposed InvDM and MgCl, either the single-branch or multibranch model performed better than the others. The results indicate that, by enhancing the cooperation in multigoal learning, our model can learn multiple goals in different scenes simultaneously with only binary reward, while yields better performance than existing work.

D. Results of Scene Generalization

In this section, we test the scene generalization performance of agents trained by different models. The competent models here include Vanilla A3C, A3C + InvDM, A3C + Loop, A3C + Mapper, and A3C + Loop + Mapper. To show the influence of the number of rooms during the training on the generalization performance, the number of rooms was set to 4, 8, 12, and 16 in turn, and there are five targets in each room.

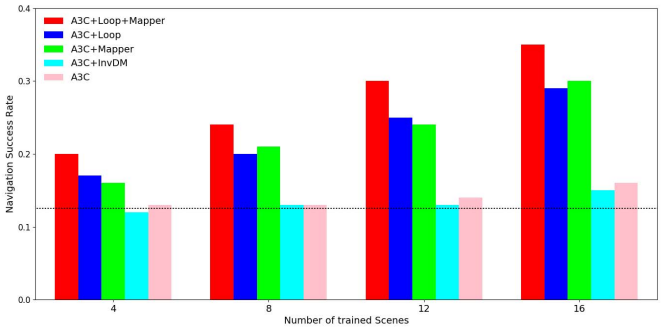


Fig. 8. Scene generalization results.

We trained each model for the same 20M steps. The test was conducted in four additional rooms, with five targets in each room, and the average navigational success rate was used as the performance metric. As shown in Fig. 8, the success rate of random strategy is about 14%, which is indicated by the black dotted line of the figure. With the increase of the number of training scenes, the success rate of Vanilla A3C and A3C + InvDM did not improve much, even lower than that of the random strategy in some cases. Although InvDM has been proven to greatly improve the speed of the training process, it has weak generalization capability for unknown scenes. Compared with the random strategy, Vanilla A3C, and A3C + InvDM, the agents trained by A3C + Loop and A3C + Mapper are more likely to succeed in the unknown test scenes. The above results also prove that by understanding whether the state has been experienced and by figuring out the difference between state and goal, both path closed-loop detection and state-target matching can improve the generalization capability of agents. Moreover, A3C + Loop + Mapper yielded the best generalization performance in all cases, indicating that the two modules could be well combined. Please refer to the website mentioned in the Abstract for the video demonstration results.

VI. Conclusion

In this work, we proposed a model-embedded actor–critic scheme to enable the agent to learn the skill of navigating to multiple goals cooperatively. We introduced two components to the A3C architecture: InvDM and MgCl to enhance the

task cooperation in visual navigation. To further improve the scene generalization capability of the agent, we designed two additional modules, path closed-loop detection and state-target matching, which were set as auxiliary tasks for navigation. Experimental results on the interactive *AI2-THOR* platform demonstrated the superiority of the proposed architecture over existing methods in the goal-dependent navigation task.

REFERENCES

[1] Z. Yang, K. Merrick, L. Jin, and H. A. Abbass, "Hierarchical deep reinforcement learning for continuous action control," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 11, pp. 5174–5184, Nov. 2018.

[2] Y. Lu and D. Song, "Visual navigation using heterogeneous landmarks and unsupervised geometric constraints," *IEEE Trans. Robot.*, vol. 31, no. 3, pp. 736–749, Jun. 2015.

[3] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher–student curriculum learning," *IEEE Trans. neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3732–3740, Sep. 2019.

[4] P. Anderson *et al.*, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 3674–3683.

[5] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.

[6] D. Silver *et al.*, "Mastering the game of go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, p. 484, 2016.

[7] S. Li, H. Wang, and M. U. Rafique, "A novel recurrent neural network for manipulator control with improved noise tolerance," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 5, pp. 1908–1918, May 2018.

[8] P. Mirowski *et al.*, "Learning to navigate in complex environments," 2016, *arXiv:1611.03673*. [Online]. Available: http://arxiv.org/abs/1611.03673

[9] A. Banino *et al.*, "Vector-based navigation using grid-like representations in artificial agents," *Nature*, vol. 557, no. 7705, p. 429, May 2018.

[10] Y. Wu, Z. Rao, W. Zhang, S. Lu, W. Lu, and Z.-J. Zha, "Exploring the task cooperation in multi-goal visual navigation," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 609–615.

[11] Y. Zhu *et al.*, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3357–3364.

[12] P. Mirowski *et al.*, "Learning to navigate in cities without a map," in *Proc. Adv. Neural Inf. Process. Syst.*, 2018, pp. 2419–2430.

[13] F. Bonin-Font, A. Ortiz, and G. Oliver, "Visual navigation for mobile robots: A survey," *J. Intell. Robotic Syst.*, vol. 53, no. 3, pp. 263–296, Nov. 2008.

[14] N. Savinov, A. Dosovitskiy, and V. Koltun, "Semi-parametric topological memory for navigation," 2018, *arXiv:1803.00653*. [Online]. Available: http://arxiv.org/abs/1803.00653

[15] H. Li, Q. Zhang, and D. Zhao, "Deep reinforcement learning-based automatic exploration for navigation in unknown environment," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2064–2076, Jun. 2020.

[16] L. Xie *et al.*, "Learning with stochastic guidance for robot navigation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 1, pp. 166–176, Jan. 2021.

[17] A. Hussein, E. Elyan, M. M. Gaber, and C. Jayne, "Deep reward shaping from demonstrations," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, May 2017, pp. 510–517.

[18] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive mapping and planning for visual navigation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 2616–2625.

[19] A. Hussein, M. M. Gaber, and E. Elyan, "Deep Active Learning for Autonomous Navigation," in *Engineering Applications of Neural Networks*, C. Jayne and L. Iliadis, Eds. Cham, Switzerland: Springer, 2016, pp. 3–17.

[20] Y. Wu, Y. Wu, G. Gkioxari, and Y. Tian, "Building generalizable agents with a realistic and rich 3D environment," 2018, *arXiv:1801.02209*. [Online]. Available: http://arxiv.org/abs/1801.02209

[21] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 2778–2787.

[22] S.-H. Hsu, S.-H. Chan, P.-T. Wu, K. Xiao, and L.-C. Fu, "Distributed deep reinforcement learning based indoor visual navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Oct. 2018, pp. 2532–2537.

[23] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," 2016, *arXiv:1611.01779*. [Online]. Available: http://arxiv.org/abs/1611.01779

[24] M. Jaderberg *et al.*, "Reinforcement learning with unsupervised auxiliary tasks," 2016, *arXiv:1611.05397*. [Online]. Available: http://arxiv.org/abs/1611.05397

[25] P. Agrawal, J. Carreira, and J. Malik, "Learning to see by moving," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 37–45.

[26] D. Xu, W. Ouyang, X. Wang, and N. Sebe, "PAD-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 675–684.

[27] D. Pathak *et al.*, "Zero-shot visual imitation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2018, pp. 2050–2053.

[28] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," 2015, *arXiv:1511.05952*. [Online]. Available: http://arxiv.org/abs/1511.05952

[29] J. X Wang *et al.*, "Learning to reinforcement learn," 2016, *arXiv:1611.05763*. [Online]. Available: http://arxiv.org/abs/1611.05763

[30] M. Andrychowicz *et al.*, "Hindsight experience replay," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5048–5058.

[31] T. Schaul, D. Horgan, K. Gregor, and D. Silver, "Universal value function approximators," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 1312–1320.

[32] Z. Yu, J. Yu, C. Xiang, J. Fan, and D. Tao, "Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 5947–5959, Dec. 2018.

[33] X. Wu, J. Zhang, and F.-Y. Wang, "Stability-based generalization analysis of distributed learning algorithms for big data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 3, pp. 801–812, Mar. 2020.

[34] L. Niu, W. Li, D. Xu, and J. Cai, "Visual recognition by learning from Web data via weakly supervised domain generalization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, no. 9, pp. 1985–1999, Sep. 2017.

[35] F. He, T. Liu, and D. Tao, "Why ResNet works? Residuals generalize," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 12, pp. 5349–5362, Dec. 2020.

[36] C. Yang, C. Chen, W. He, R. Cui, and Z. Li, "Robot learning system based on adaptive neural control and dynamic movement primitives," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 3, pp. 777–787, Mar. 2019.

[37] W. Zhu, X. Guo, Y. Fang, and X. Zhang, "A path-integral-based reinforcement learning algorithm for path following of an autoassembly mobile robot," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 11, pp. 4487–4499, Nov. 2020.

[38] A. Devo, G. Mezzetti, G. Costante, M. L. Fravolini, and P. Valigi, "Towards generalization in target-driven visual navigation by using deep reinforcement learning," *IEEE Trans. Robot.*, vol. 36, no. 5, pp. 1546–1561, Oct. 2020.

[39] M. Lanctot *et al.*, "A unified game-theoretic approach to multiagent reinforcement learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4190–4203.

[40] J. Farebrother, M. C. Machado, and M. Bowling, "Generalization and regularization in DQN," 2018, *arXiv:1810.00123*. [Online]. Available: http://arxiv.org/abs/1810.00123

[41] D. J. Mankowitz *et al.*, "Robust reinforcement learning for continuous control with model misspecification," 2019, *arXiv:1906.07516*. [Online]. Available: http://arxiv.org/abs/1906.07516

[42] C. Packer, K. Gao, J. Kos, P. Krähenbähl, V. Koltun, and D. Song, "Assessing generalization in deep reinforcement learning," 2018, *arXiv:1810.12282*. [Online]. Available: http://arxiv.org/abs/1810.12282

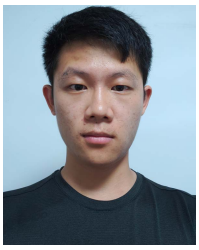
[43] C. Zhang, O. Vinyals, R. Munos, and S. Bengio, "A study on overfitting in deep reinforcement learning," 2018, *arXiv:1804.06893*. [Online]. Available: http://arxiv.org/abs/1804.06893

[44] M. Hashemzadeh, R. Hosseini, and M. N. Ahmadabadi, "Exploiting generalization in the subspaces for faster model-based reinforcement learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 6, pp. 1635–1650, Jun. 2019.

[45] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. Cambridge, MA, USA: MIT Press, 2018.

[46] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1928–1937.

[47] E. Kolve *et al.*, "AI2-THOR: An interactive 3D environment for visual AI," 2017, *arXiv:1712.05474*. [Online]. Available: http://arxiv.org/abs/1712.05474



Zhenhuan Rao received the B.S. degree from the School of Automation, Huazhong University of Science and Technology, Wuhan, China, in 2017, and the M.S. degree from the School of Control Science and Engineering, Shandong University, Jinan, China, in 2020.

His research interests include reinforcement learning and visual navigation.

Yuechen Wu received the B.S. degree from the School of Astronautics, Harbin Institute of Technology, Harbin, China, in 2016, and the M.S. degree from the School of Control Science and Engineering, Shandong University, Jinan, China, in 2019.

He is currently an Algorithm Engineer with Fuxi AI Lab, NetEase, Hangzhou, China. His research interests include reinforcement learning, computer vision, and intelligence testing.

Zifei Yang received the B.S. degree from the School of Electrical Engineering and Automation, Shandong University of Science and Technology, Qingdao, China, in 2019. She is currently pursuing the M.S. degree with the School of Control Science and Engineering, Shandong University, Jinan, China.

Her research interests include visual representation learning in computer vision and robotics.

Wei Zhang received the Ph.D. degree in electronic engineering from The Chinese University of Hong Kong, Hong Kong, in 2010.

He is currently with the School of Control Science and Engineering, Shandong University, China. He has authored over 80 papers in international journals and refereed conferences. His research interests include computer vision, image processing, pattern recognition, and robotics.

Dr. Zhang has served as a program committee member and a reviewer for various international conferences and journals in image processing, computer vision, and robotics.

Shijian Lu received the Ph.D. degree in electrical and computer engineering from the National University of Singapore, Singapore, in 2005.

He is currently an Assistant Professor with the School of Computer Science and Engineering, Nanyang Technological University, Singapore. His research interests include image and video analytics, visual intelligence, and machine learning. He has published more than 100 international refereed journal articles and conference papers and coauthored over ten patents in these areas.

Dr. Lu has served on the program committee of a number of conferences, e.g., the Area Chair for the International Conference on Document Analysis and Recognition (ICDAR) 2017 and 2019 and the Senior Program Committee of the International Joint Conferences on Artificial Intelligence (IJCAI) 2018 and 2019. He is also an Associate Editor of the journal *Pattern Recognition* (PR).

Weizhi Lu received the B.E. and M.E. degrees from Shandong University, Jinan, China, in 2007 and 2010, respectively, and the Ph.D. degree from the National Institute of Applied Sciences, Rennes, France, in 2014, all in electronic engineering.

From 2015 to 2018, he was a Post-Doctoral Researcher with the Graduate School at Shenzhen, Tsinghua University, Beijing, China. He is currently an Associate Professor with the School of Control Science and Engineering, Shandong University. His research interests include signal and image processing, compressed sensing, and machine learning.

ZhengJun Zha (Member, IEEE) received the B.E. and Ph.D. degrees from the University of Science and Technology of China, Hefei, China, in 2004 and 2009, respectively.

He was a Researcher with the Hefei Institutes of Physical Science, Chinese Academy of Sciences, Beijing, China, from 2013 to 2015, a Senior Research Fellow with the School of Computing, National University of Singapore (NUS), Singapore, from 2011 to 2013, and a Research Fellow with NUS from 2009 to 2010. He is currently a Full Professor with the School of Information Science and Technology, University of Science and Technology of China, where he is also the Executive Director of the National Engineering Laboratory for Brain-Inspired Intelligence Technology and Application. His current research interests include multimedia analysis, retrieval and applications, and computer vision. He has authored or coauthored more than 100 articles in these areas with a series of publications on top journals and conferences.

Dr. Zha was a recipient of multiple paper awards from prestigious multimedia conferences, including the Best Paper Award and the Best Student Paper Award in ACM Multimedia and so on. He also serves as an Associate Editor for the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY.