Exploring the Task Cooperation in Multi-goal Visual Navigation

Yuechen Wu[1†], **Zhenhuan Rao**[1†], **Wei Zhang**[1*], **Shijian Lu**[2], **Weizhi Lu**[1] and **Zheng-Jun Zha**[3]
[1]School of Control Science and Engineering, Shandong University
[2]School of Computer Science and Engineering, Nanyang Technological University
[3]School of Information Science and Technology, University of Science and Technology of China
{wuyuechen, raozhenhuan}@mail.sdu.edu.cn, {davidzhang, wzlu}@sdu.edu.cn,
shijian.lu@ntu.edu.sg, zhazj@ustc.edu.cn

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 multi-goal visual navigation task. To enhance the task cooperation in multi-goal learning, we introduce two new designs to the reinforcement learning scheme: *inverse dynamics model (InvDM)* and *multi-goal co-learning (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. Extensive results on the interactive platform AI2-THOR demonstrate that the proposed method converges faster than state-of-the-art methods while producing more direct routes to navigate to the goal. The video demonstration is available at: https://youtube.com/channel/UCtpTMOsctt3yPzXqe_JMD3w/videos.

1 Introduction

Visual navigation is an intelligent ability to determine the current position and then to plan a path towards some goal location from image or video inputs [Gupta *et al.*, 2017; Anderson *et al.*, 2018]. Due to the limitation of 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 tracking [Zhang *et al.*, 2018], Atari games [Mnih *et al.*, 2015] and computer Go [Silver *et al.*, 2016], the early efforts were made to train an agent to learn the skill of navigating to a specific goal [Mirowski *et al.*, 2016; Banino *et al.*, 2018].

Compared to the regular navigation task, goal-dependent navigation is more challenging, which requires the agent 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 Figure 1. Therefore, with such 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. Specifically, Zhu *et al.* used a siamese actor-critic architecture with shaped rewards to support navigating to different goals [Zhu *et al.*, 2017]. Mirowski *et al.* introduced an auxiliary task and used a curriculum training scheme to relieve the problem of sparse rewards in goal-dependent navigation tasks [Mirowski *et al.*, 2018]. 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.

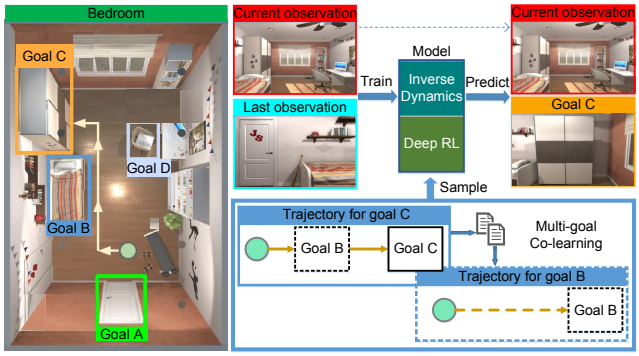

Figure 1: Illustration of the proposed goal-dependent navigation model. Please enlarge to see details.

To relieve the above limitations, we present a new model-embedded actor-critic scheme that allows the agent to learn the skill of navigating to multiple goals cooperatively based on visual observation only. First, as shown in Figure 1, we introduce an inverse dynamics model (InvDM) to the actor-

[†]Equal contribution.
[*]Corresponding author.

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 dynamics model are three-fold: 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 of 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 a guidance to push the agent to explore more efficiently. So 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 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 *Multi-goal Co-learning (MgCl)* which could make the agent learn from unintentional positive experience when interacting with the environment. The idea can be explained by Figure 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. So, 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.

2 Related Work

There has been a number of methods focusing on the task of visual navigation [Bonin-Font *et al.*, 2008; Savinov *et al.*, 2018]. Gupta $et\ al.$ designed a unified joint architecture [Gupta *et al.*, 2017] for mapping and planning in unknown environments. Recently, deep reinforcement learning methods were introduced for navigation in simulated 3D scenes [Wu *et al.*, 2018a; Pathak *et al.*, 2017]. However, due to the sparse reward signals in navigation tasks, the recent navigational agents seek for aids from auxiliary supervised tasks for training [Dosovitskiy and Koltun, 2016; Jaderberg *et al.*, 2017]. Mirowski $et\ al.$ proposed to train agent with auxiliary depth prediction and loop closure classification tasks [Mirowski *et al.*, 2016]. Banino $et\ al.$ introduced a grid cell network which was trained as an auxiliary task [Banino *et al.*, 2018] that enabled the agent to plan direct trajectories to the goals. All these demonstrated auxiliary tasks can be used to facilitate learning [Xu *et al.*, 2018; Wu *et al.*, 2018b]. Unlike the previous work relying on external signals to assist training, we advocate using the action signals to explore the difference of 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 [Anderson *et al.*, 2018]. Zhu $et\ al.$ used a feedforward siamese actor-critic architecture incorporating a pretrained ResNet [Zhu *et al.*, 2017] to support navigation to different goals in a discredited 3D environment. Mirowski $et\ al.$ [Mirowski *et al.*, 2018] presented a dual-pathway agent architecture that enable agent to navigate to a series of goals in a city-scale, real-world visual environment. Pathak $et\ al.$ presented a method to learn the trajectory between state and goal, and then employed the expert demonstration to communicate the goal for navigation [Pathak *et al.*, 2018]. Savinov $et\ al.$ proposed a semi-parametric topological memory (SPTM) which is trained in supervised fashion and acts as a planning module in goal-directed navigation [Savinov *et al.*, 2018]. Compared to these methods, we propose a self-supervised InvDM to better predict the difference of 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 [Schaul *et al.*, 2015b; Wang *et al.*, 2016]. For example, Andrychowicz $et\ al.$ presented a technique coined Hindsight Experience Replay (HER) [Andrychowicz *et al.*, 2017] to improve the sample efficiency when learning multi-goal policies [Schaul *et al.*, 2015a]. Riedmiller $et\ al.$ developed a hierarchical extension of HER in order to further speed up training [Andrychowicz *et al.*, 2017]. 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 make 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 to the online methods like A3C.

3 Method

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

3.1 Preliminaries

The goal-dependent reinforcement learning aims to train an agent interacting with an environment to maximize the expected future rewards [Sutton and Barto, 2018] for a goal. It is concerned with policy optimization in a Markov Decision Process (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.

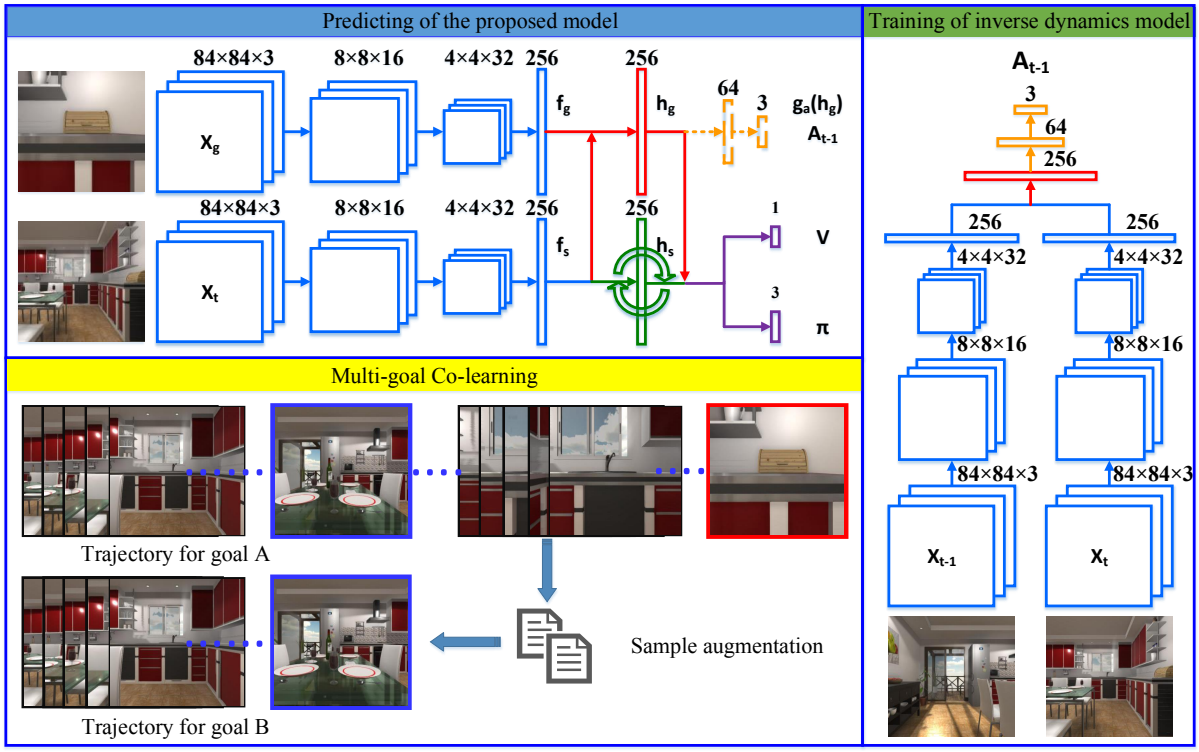

Figure 2: Details of the proposed deep reinforcement learning model for goal-dependent navigation.

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 [Mnih *et al.*, 2016] 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:

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

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

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

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

where η denotes the learning rate.

3.2 Proposed Architecture

As illustrated in Figure 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) is only rely 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, an observation of current state x_t and an 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 in achieving a given goal g. We train the proposed model through end-to-end deep reinforcement learning by incorporating an auxiliary objective. The aim of 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 3.3.

The details of the architecture are described as follows. First, 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 8x8, a stride of 4x4, and 16 feature maps. The second layer has a kernel of size 4x4, a stride of 2x2, and 32 feature maps. The fully connected layer has 256 hidden units. Rectified linear units (ReLU) 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 $h_s(f_s)$. 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.

3.3 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 Figure 2, an inverse dynamics model (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. So, 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 task that could provide additional training signals. Meanwhile, as aforementioned, one goal only changes the reward function rather than the transition structure of Markov Decision Process. So 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 Figure 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 below.

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

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

3.4 Multi-goal Co-learning

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 *multi-goal co-learning (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 an agent has a number of goals $g_A, g_B, \cdots \in \mathcal{G} \subseteq \mathcal{S}$, 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, 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. So 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 to 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. So, 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 contributes more at the beginning of training, and gradually decreases 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. So, it is meaningless to learn to navigate to it. More details of MgCl can be found in Algorithm 1.

4 Experimental Results

This section tests the proposed method on *AI2-THOR* [Kolve *et al.*, 2017], which is an open-source set within the Unity3D game engine, and enables navigation in a set of near-photo-realistic 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 smallest while the *Kitchen* is the biggest scene. The detailed settings of the environment in the experiments are given as follows.

- *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 meters), and the turn ac-

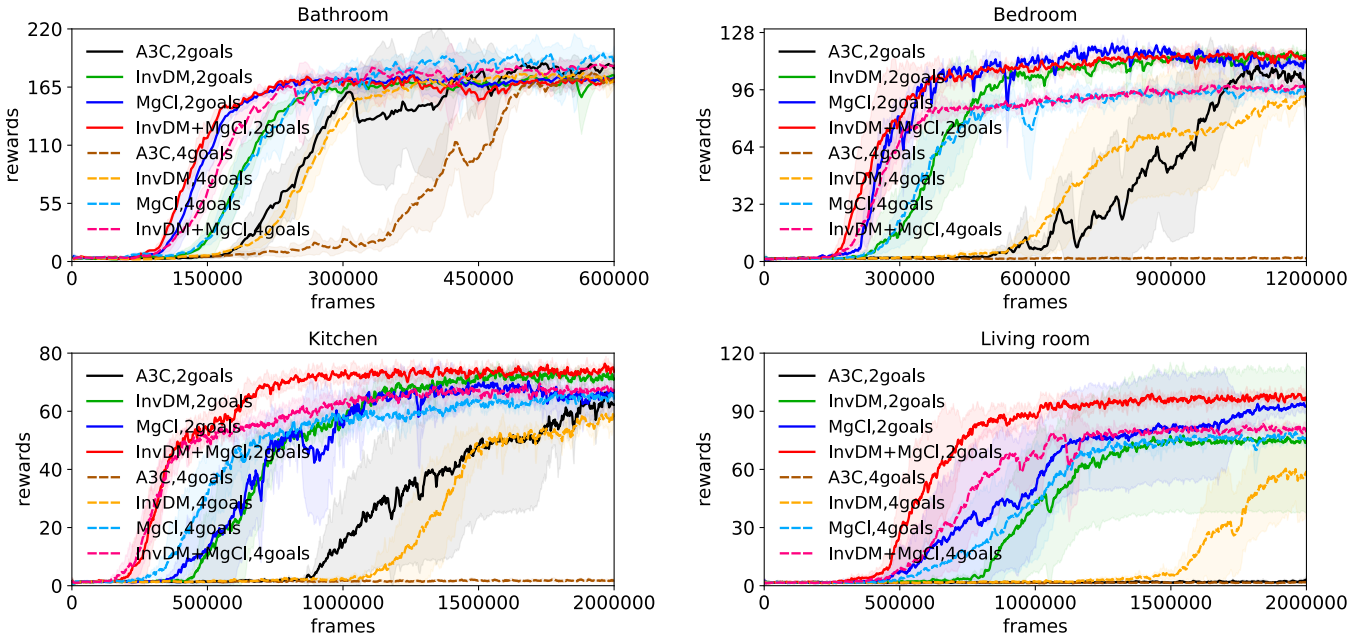

Figure 3: Testing of our model in different scenes. Due to space limit, 'A3C+InvDM' is abbreviated to 'InvDM', so do the other terms.

Algorithm 1: Multi-goal Co-learning (MgCl)

Given asynchronous advantage actor-critic (A3C).
Given a set of goals $\mathcal{G} = \{g_A, g_B, \cdots\}$.
for *each update* **do**
  Interact and generate a trajectory $\mathcal{T}_{\mathcal{A}} = \{s_t, a_t, s_{t+1}, a_{t+1}, \cdots, s_{t+k-1}, a_{t+k-1}, s_{t+k}\}$.
  for $i \in \{0, \cdots, k-1\}$ **do**
    if $s_i == g' \in \mathcal{G}$ **then**
      Generate a pseudo trajectory $\mathcal{T}' = \{s_t, a_t, s_{t+1}, a_{t+1}, \cdots, s_{t+i-1}, a_{t+i-1}, s_{t+i}\}$.
      Set reward $r'(s_{t+i}, a, g') \leftarrow 1$.
      Accumulate gradients for goal g'.
    end
  end
end

tion has a constant angle (90 degree). The scene space is discretized into a grid-world representation.

- *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 down-sampled 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.
- *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 5 actions ($i.e.$, t_{max} = 5). To relieve the bias behaviour, 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 the episodes that terminate in every 2000 frames over all goals. For each goal, we random initialize the starting location of the agent, the episodes will not terminate until the agent reaches the goal. Each test was repeated 5 times with different seeds. We report the average final rewards and plot the mean and standard deviation of the reward statistic.

4.1 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 Figure 3) in the scenes of *Bathroom*, *Bedroom*, *Kitchen* and *Living room* with different number of goals, i.e., 2 goals and 4 goals. It is noted that the A3C here is a goal-dependent variant of the standard baseline [Mnih *et al.*, 2016]. The network architectures of A3C and A3C+InvDM are identical except that A3C+InvDM is trained with an additional auxiliary loss defined by Eq. (5).

The learning curves in Figure 3 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 the 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 InvDM has good potential in

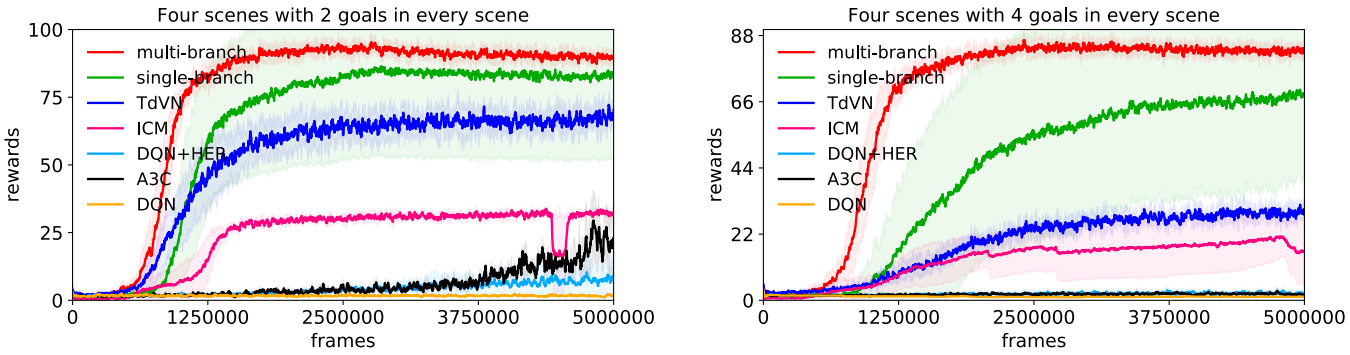

Figure 4: Comparison to state-of-the-art methods in multiple scenes with different number of goals.

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 w.r.t the number of goals could be proved.

4.2 Ablation Study of the Multi-goal Co-learning

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. 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 agent 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 Figure 3, InvDM+MgCl significantly outperforms InvDM and MgCl in all scenes, in terms of both convergence speed and rewards.

4.3 Results of Learning in Multiple Scenes

In this section, the proposed method was tested to learn multiple goals in different scenes simultaneously. Different scenes may have different environmental dynamics. So when learning in multiple scenes at the same time, the network parameters of InvDM could be trained and shared in two manners: full share with a single-branch model and partial share with a multi-branch model as described below.

- *Single-branch:* All parameters of InvDM are shared by all scenes, i.e., full share.
- *Multi-branch:* 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 Figure 4, the multi-branch model outperforms the single-branch one in both two indexes in the two different configurations. We hypothesize the partial share scheme make it possible to capture the particular characteristics in a scene that may vary across scene instances. It can also be found that the multi-branch 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 [Mnih *et al.*, 2016], DQN [Mnih *et al.*, 2015]), ICM [Pathak *et al.*, 2017], TdVN [Zhu *et al.*, 2017] and HER [Andrychowicz *et al.*, 2017]. As shown in Figure 4, DQN and A3C suffers from slow convergence. It seems difficult for them to train with sparse and binary rewards. DQN+HER also performed poor 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 multi-branch model performed better than the others. The results indicate that by enhancing the cooperation in multi-goal learning, our model can learn multiple goals in different scenes simultaneously with only binary reward, while yields better performance than existing works.

5 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. Experimental results on the interactive *AI2-THOR* platform demonstrated the superiority of the proposed architecture over existing methods in goal-dependent navigation task.

Acknowledgments

This work was supported by the National Key Research and Development Plan of China under Grant 2017YFB1300205, NSFC Grant no. 61573222, and Major Research Program of Shandong Province 2018CXGC1503.

References

[Anderson *et al.*, 2018] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton van den Hengel. Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments. In *CVPR*, volume 2, 2018.

[Andrychowicz *et al.*, 2017] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *NIPS*, pages 5048–5058, 2017.

[Banino *et al.*, 2018] Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429, 2018.

[Bonin-Font *et al.*, 2008] Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. Visual navigation for mobile robots: A survey. *Journal of intelligent and robotic systems*, 53(3):263–296, 2008.

[Dosovitskiy and Koltun, 2016] Alexey Dosovitskiy and Vladlen Koltun. Learning to act by predicting the future. *arXiv preprint arXiv:1611.01779*, 2016.

[Gupta *et al.*, 2017] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *CVPR*, pages 2616–2625, 2017.

[Jaderberg *et al.*, 2017] Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. Reinforcement learning with unsupervised auxiliary tasks. 2017.

[Kolve *et al.*, 2017] Eric Kolve, Roozbeh Mottaghi, Daniel Gordon, Yuke Zhu, Abhinav Gupta, and Ali Farhadi. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[Mirowski *et al.*, 2016] Piotr Mirowski, Razvan Pascanu, Fabio Viola, Hubert Soyer, Andrew J Ballard, Andrea Banino, Misha Denil, Ross Goroshin, Laurent Sifre, Koray Kavukcuoglu, et al. Learning to navigate in complex environments. *arXiv preprint arXiv:1611.03673*, 2016.

[Mirowski *et al.*, 2018] Piotr Mirowski, Matthew Koichi Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Koray Kavukcuoglu, Andrew Zisserman, and Raia Hadsell. Learning to navigate in cities without a map. *arXiv preprint arXiv:1804.00168*, 2018.

[Mnih *et al.*, 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*, pages 1928–1937, 2016.

[Pathak *et al.*, 2017] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*, volume 2017, 2017.

[Pathak *et al.*, 2018] Deepak Pathak, Parsa Mahmoudieh, Guanghao Luo, Pulkit Agrawal, Dian Chen, Yide Shentu, Evan Shelhamer, Jitendra Malik, Alexei A Efros, and Trevor Darrell. Zero-shot visual imitation. In *ICLR*, 2018.

[Savinov *et al.*, 2018] Nikolay Savinov, Alexey Dosovitskiy, and Vladlen Koltun. Semi-parametric topological memory for navigation. 2018.

[Schaul *et al.*, 2015a] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *ICML*, pages 1312–1320, 2015.

[Schaul *et al.*, 2015b] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[Silver *et al.*, 2016] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484, 2016.

[Sutton and Barto, 2018] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[Wang *et al.*, 2016] Jane X Wang, Zeb Kurth-Nelson, Dhruva Tirumala, Hubert Soyer, Joel Z Leibo, Remi Munos, Charles Blundell, Dharshan Kumaran, and Matt Botvinick. Learning to reinforcement learn. *arXiv preprint arXiv:1611.05763*, 2016.

[Wu *et al.*, 2018a] Yi Wu, Yuxin Wu, Georgia Gkioxari, and Yuandong Tian. Building generalizable agents with a realistic and rich 3d environment. *arXiv preprint arXiv:1801.02209*, 2018.

[Wu *et al.*, 2018b] Yuechen Wu, Wei Zhang, and Ke Song. Master-slave curriculum design for reinforcement learning. In *IJCAI*, pages 1523–1529, 2018.

[Xu *et al.*, 2018] Dan Xu, Wanli Ouyang, Xiaogang Wang, and Nicu Sebe. Pad-net: Multi-tasks guided prediction-and-distillation network for simultaneous depth estimation and scene parsing. In *CVPR*, June 2018.

[Zhang *et al.*, 2018] Wei Zhang, Ke Song, Xuewen Rong, and Yibin Li. Coarse-to-fine uav target tracking with deep reinforcement learning. *TASE*, 2018.

[Zhu *et al.*, 2017] Yuke Zhu, Roozbeh Mottaghi, Eric Kolve, Joseph J Lim, Abhinav Gupta, Li Fei-Fei, and Ali Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *ICRA*, pages 3357–3364. IEEE, 2017.