Object Goal Visual Navigation using Semantic Spatial Relationships*

Jingwen Guo[1,2], Zhisheng Lu*[1], Ti Wang[1,3], Weibo Huang[1], and Hong Liu[1]

[1] Peking University Shenzhen Graduate School
[2] Soochow University
[3] Nanjing University of Science and Technology

Abstract. The target-driven visual navigation is a popular learning-based method and has been successfully applied to a wide range of applications. However, it has some disadvantages, including being ineffective at adapting to unseen environments. In this paper, a navigation method based on Semantic Spatial Relationships (SSR) is proposed and is shown to have more reliable performance when dealing with novel conditions. The construction of joint semantic hierarchical feature vector allows for learning implicit relationship between current observation and target objects, which benefits from construction of prior knowledge graph and semantic space. This differs from the traditional target driven methods, which integrate the visual input vector directly into the reinforcement learning path planning module. Moreover, the proposed method takes both local and global features of observed image into consideration and is thus less conservative and more robust in regards to random scenes. An additional analysis indicates that the proposed SSR performs well on classical metrics. The effectiveness of the proposed SSR model is demonstrated comparing with state-of-the-art methods in unknown scenes.

Keywords: Visual navigation · Semantic graph · Hierarchical relationship.

1 Introduction

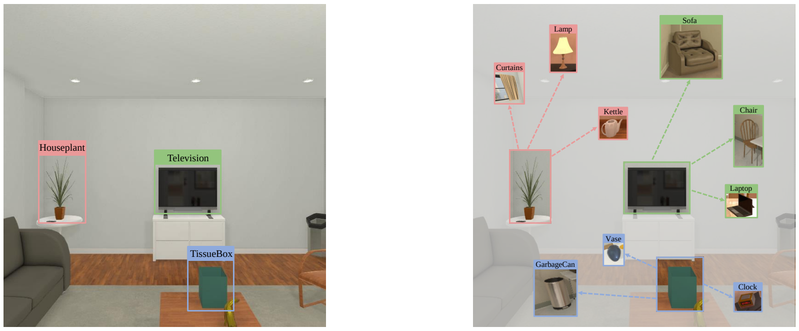
Vision-based mobile robot navigation has produced countless research contributions, both in the field of vision and in the field of control. However, it is difficult for mobile robots to run at high speeds due to the huge radar data and dimensionality disasters when processing real-time status information.

As a research hotspot in machine learning, deep reinforcement learning provides us with an important intelligent control method. It relies on the perception ability of deep learning without model information, and can collect sample data for learning during the navigation process of mobile robots. Interacting with the environment to obtain feedback for strategy is an effective method in field of

* Supported by National Natural Science Foundation of China $No.$62073004, Science and Technology Plan of Shenzhen $No.JCYJ$20190808182209321

mobile robot navigation. In recent years, many visual navigation methods based on reinforcement learning have emerged.

Traditional navigation algorithms firstly build the environment map, and then realize the path planning. Compared with these algorithms, using current environment observation and target information as prior conditional input of model, and planing the optimal path afterwords is a current research hotspot. These methods can be collectively referred to as end-to-end visual navigation. Zhu et al. [29] used a pre-trained residual network as feature extraction module and designed a twin network architecture to improve the goal and scenario generalization performance. Mirowski et al. [17] proposed a dual-path agent structure that uses end-to-end reinforcement learning for training and can handle real-world visual navigation tasks on city-level scale. Pathak et al. [20] exploited the collected samples to train general navigation strategies in a small range, and then used the expert teaching method to transfer the navigation target information to the agent, which can be summarized as an unsupervised learning mode.



(a) (b)

Fig. 1. Semantic Cues Based on Knowledge Graph. The navigation goal for agent is GarbageCan, which is invisible from observation (a). We use locational features as cues such as Television that can be easily detected. From prior semantic knowledge graph, we can learn that the connection between GarbageCan and TissueBox is strong as shown in (b). So their spatial locations should be very close. The agent can navigate to the target simply though it's invisible at the moment.

In above tasks, with target location and self-centered observations as input, the agent needs to persistently execute one possible action until it reaches the target. Target-driven visual semantic navigation is very challenging since the location and appearance of the target are unknown to the agent. The navigation system needs to estimate both the coordinates of the target and the path to it at the same time. Considering the disadvantages of current navigation methods, we propose an innovative visual navigation algorithm based on deep reinforcement learning and knowledge graph. The main idea is shown in Fig. 1. We capture the global semantic features and local location features of the current observation simultaneously. The prior knowledge graph can be used to perform semantic guidance and encode the spatial relationship of objects naturally, so that agent can infer the general direction of target even if the target is temporarily invisible.

We list the main contributions of this paper: (1) We propose an efficient end-to-end model Spatial Semantic Relationship (SSR) that takes visual images and target semantic labels as model inputs without map of the environment. (2) We use the deep graph neural network to introduce the external semantic prior information between objects, to encode the current visual observation of scene and the target infomation, learning the implicit relationships between these objects. (3) Our model improves the navigation performance and generalization ability to unknown environments and new target objects. (4) We construct the simulation environment as a robot navigation environment for algorithm comparison.

2 Related Work

Visual Navigation Without Map Common navigation tasks are mainly divided into two categories. One task is actively exploring the environment, and the other is exploiting devices such as GPS sensors to send direction signals to the target. Visual navigation has powerful scene recognition capabilities because it can obtain massive amounts of environmental information through visual sensors. There have been some research recently in field of visual navigation. Lu et al. [14] proposed a novel abstract map Markov Network for deep reinforcement learning visual navigation method, and used graph neural network for probabilistic inference. It solved the problem that the agent is restricted in the new environment of acquiring the map and improved the success rate of navigation. Wu et al. [25] proposed a way to incorporate information theory regularization into deep reinforcement learning framework to improve the cross-target and cross-scene versatility of visual navigation. Gupta et al. [6] used visual information and self-motion to navigate in a vast indoor environment in the way of building potential map.

Deep Reinforcement Learning Methods for Navigation Methods based on deep reinforcement learning have been combined with the traditional navigation algorithm [18] and achieved great process recently. Yu et al. [27] proposed a neural network and a hierarchical reinforcement learning mobile robot path planning model, which mapped the robot's actions and current state through hierarchical reinforcement learning. This method made robot perceive the environment and perform feature extraction to solve the problems of autonomous learning in path planning and slow path planning convergence speed. Mousavian et al. [19] proposed a deep reinforcement learning framework that used LSTM-based strategies for semantic target driven navigation, and learned navigation strategies based on capturing spatial layout and semantic contextual clues. Wen et al. [22] proposed an integrated navigation method Active SLAM which combined path planning with SLAM (simultaneous localization and mapping). They used fully convolutional residual network to identify obstacles to obtain depth images. They employed dual DQN algorithm to plan obstacle avoidance path and established 2D map of the environment based on FastSLAM at the same time during navigation process.

Object Goal Navigation Object navigation is a valuable research problem, which refers to the robot autonomously navigating to a specific object. Different from point navigation related methods [23], the goal of object navigation is to navigate to the target category object, not global coordinates. Object navigation means that agent needs to make full use of the prior knowledge of the scene, which is very important for the efficiency of navigation. Chaplot et al. [2] proposed a modular system Goal-Oriented Semantic Exploration, which can effectively explore the environment by constructing episodic semantic maps and using it according to target object categories. Wortsman et al. [24] proposed Self-Adaptive Visual Navigation (SAVN) method, where the agent used meta-learning to adapt to the invisible environment. Martins et al. [15] solved the problem of enhanced metric representation by using semantic information from RGB-D images to construct scenes. They proposed a complete framework to use object-level information to create an enhanced map representation of the environment to assist robots in completing the object goal visual navigation. They exploited CNN-based object detector and 3D model-based segmentation technology to perform instance semantic segmentation, and used Kalman filter dictionaries to complete semantic class tracking and positioning.

Semantic Reasoning Using GNNs Graph Neural Network (GNN) was first proposed by Gori et al. [5]. It is a deep learning method for processing graph data. GNNs have great effect on extracting features of data containing graph structures. Kawamoto et al. [9] proved that untrained GNN can perform well with a simple architecture. Deepmind [7] proved that the graph network supports relational reasoning and combinatorial generalization, which is of great significance for probabilistic reasoning. Guo et al. [28] pointed out that GNN is applied to tasks such as relation extraction and contextual reasoning, and the results are significantly better than other methods. Kim et al. [10] proposed a F-GCN module based on graph convolutional network, which used GNN to extract knowledge from multi-modal context and solve problem reasoning. In our proposed model, we adopt GCN to encode the prior knowledge graph and learn the spatial semantic relationship between agent and target object.

3 Proposed Approach

Our task is to introduce prior knowledge graph and spatial location information into the target-driven visual semantic navigation system. In order to achieve this goal, our method Spatial Semantic Relationship (SSR) contains three main components as shown in Fig. 2: (1) Spatial relationship between objects building module: we take local features observed by the agent and semantic label of the target object as input to explore the hidden internal spatial relationship and construct the context vector; (2) Semantic scene building module: we take the global visual features observed by agent and prior knowledge graph as input to form semantic scene representation; (3) Reinforcement learning navigation module: we combine the output of first two modules to obtain a joint visual semantic knowledge feature vector. We then input this vector into reinforcement learning navigation network, making the agent interact with environment and generate the next action strategy with judgement.

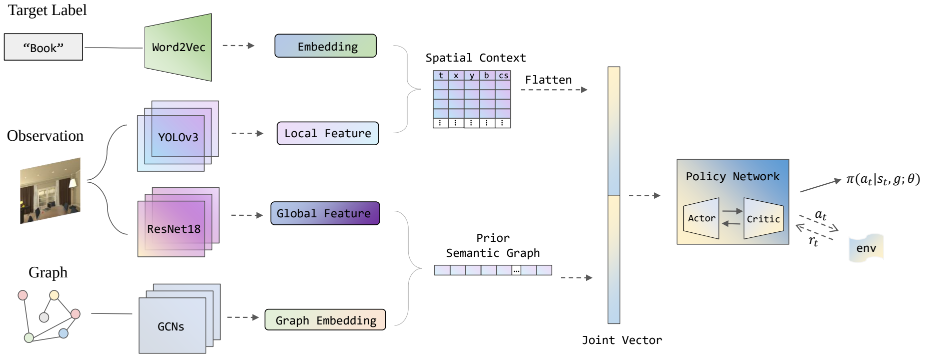


Fig. 2. Illustration of Spatial Semantic Relationship Model

3.1 Task Definition

We aim at training an agent, which receives RGB images as observation information, and semantic tags as target information. The agent is trained to look for an instance of the specificated target category object. During the navigation, agent perceives target location and environment through RGB images and finds the minimum length of action sequence to reach the target location while avoiding obstacles. Given the sequential decision characteristics of visual navigation, we can define this task as a Markov Decision Process (MDP) problem.

Considering $G = \{g_0, g_1, ..., g_n\}$ as the set of target objects, $S = \{s_0, s_1, ..., s_m\}$ as the set of states that agent may exist. s_t is defined by function $s_t = f(o_t; \alpha)$, where o_t is the current agent's observation of the environment from the first perspective, α is the network parameter. At each time step t, the agent needs to act according to the current state s_t and the target object g_i. Agent choose action a_t from set $A = \{MoveAhead, RotateRight, RotateLeft, LookUp, LookDown, DONE\}$ to achieve the maximum policy expectation π^* according to the strategy function $\pi(a_t|s_t, g; \theta)$, which is expressed as follows:

$$\pi^* = \arg\max_{\pi} E_{\pi}\left[\sum_{k}^{N} r_{k+t}|s_t = s, a_t = a\right], \forall s \in S, \forall a \in A, \forall t \geq 0 \qquad (1)$$

The optimal strategy π^* represents the prediction of the maximum cumulative reward value that can be obtained in the future.

We add a special termination action $DONE$ to the agent's action set. If the agent executes $DONE$ and the target object is visible, then the navigation task is considered successful. If the agent executes $DONE$ while the target object is invisible, it is judged as failure. The target object is visible only when it is in the agent's view, and the geodestic distance between them is less than twice the width of the agent [1].

3.2 Construction of Semantic Relation Graph

Global Visual Feature Extraction Visual feature extraction is divided into local visual representation and global feature extration. The global features of image describe the spatial position relationship between objects. In contrast, the local features contain richer information of specific object. Therefore, we use different networks to extract the global and local features of objects at the same time, which are respectively used for the joint representation of spatial relationships and the construction of semantic relationship maps. He et al. [8] proposed a Residual Neural Network (ResNet) to learn features from visual images, which have shown excellent performance in a variety of computer vision tasks. We use ResNet18 [8] pre-trained in the ImageNet [4] to extract global feature vector of environment for each input observation to perceive surroundings. This vector will be input into the GCN later together with the knowledge graph embedding as a node feature to form the construction of prior semantic knowledge graph.

Prior Graph Embedding We exploit the Visual Genome [13] to integrate semantic knowledge in the form of graph representation to construct a knowledge graph, and use GCNs [3] to calculate the relationship characteristics. GCN is an extension of the graph structure of CNN, and its goal is to learn the functional representation of a given graph $G = (V, E)$, where V is the set of nodes and E is the set of edges. The input of each node in V is a feature vector x_i. We summarize the input of all nodes into a matrix $X = \left[x_1, \ldots, x_{|V|}\right] \in \mathbb{R}^{|V| \times D}$, where D represents the dimension of the input feature. The graph structure is represented as a binary adjacency matrix $A \in \mathbb{R}^{|V| \times |V|}$. We normalize A to get $\bar{A}$. The structure of neural network can be expressed as follows:

$$H^{(l+1)} = f(\bar{A} \cdot H^{(l)} \cdot W^{(l)}) \tag{2}$$

where $f(\cdot)$ denotes the activation function, $W^{(l)}$ is the parameter of the lth layer, and l is the index of GCN layers. We construct a knowledge graph by including all object categories that appear in the simulation environment we use. Each object category is represented as a node in the graph. We input the global visual features extracted in the previous section into the GCNs together with the pre-built knowledge graph embedding. Consistent with [26], we use a three-layer GCN. The first two layers output the potential features of the joint input, and the last layer outputs a single value for each node. The final output is a $|V|$-dimensional feature vector. This feature vector basically encodes the global features of the current scene and the semantic priors in the environment.

3.3 Joint Representation of Spatial Context Information

We employ YOLOv3 [21] for local feature representation since it is a advanced real-time object detection system. The image of the current frame obtained by the agent is first input into YOLOv3 for target detection. We use the stacks of $[x, y, b, e]$ to represent the local features of the extracted image. For each detected object, x and y represent the center of the detected bounding box, and b represents the size of the bounding box; e represents the word vector corresponding to the category label of the object. Considering objects that are not in the current frame but may exist in the scene, we form the local feature representation of the current observation image $c_{embed} = [t, x, y, b, e]$, $t = 1$ if the object exists in the current frame. The target tag gets its word embedding through pretrained Word2Vec [16]. We then calculate the $Cosine\,Similarity\,(CS)$ between each object and the target based on the embedding of their labels. The last column in c_{embed} is replaced with CS, so that we obtain the joint feature representation of the spatial relationship with five columns, which also called location-aware vector. After flattening the current local feature representation, we merge it with the global visual feature graph embedding of Section 3.2. The final vector is the visual representation of semantic knowledge of our environment. We introduce this vector into the policy network for decision making later.

3.4 Navigation Driven by Spatial Semantic Relationships of Objects

We use the Asynchronous Advantage Actor-Critic (A3C) [18] algorithm to predict the strategy and reward value of each time step. The input of our A3C module is the output feature of the joint representation, which consists of the current state, the prior relationship, and the semantic task goal. The A3C module produces two outputs, strategy and reward value. The hidden layer of the network consists of several fully connected layers and ReLU layers. The joint input is first mapped to the latent space, and then two branches of the network generate $|A|$ dimension of strategies and values, as shown in the Fig. 2. Different from previous researches using different strategy networks for different scenarios [29], we use a single strategy network for samples in different scenarios. This improves the navigation efficiency and generalization ability in unseen scenes.

4 Experiments

4.1 Datasets

The dataset used in this article is AI2-THOR (The House Of inteRactions) [12]. There are 120 scenes in the AI2-THOR environment, covering four different room categories: Kitchen, Living Room, Bedroom and Bathroom. Each category room has 30 different scenes. Each room has a set of objects that can be found. Certain object types can be found in all scenes of a given category, and certain object types can sometimes only be found in scenes of a specific category. Similar to [24], we use the first 20 scenes of each scene type as the training set, 5 scenes for verification, and the remaining 5 scenes for test. For each current state, the agent will take an action a_t from the set of actions A. For our experiment, we set the same target classes for different types of scenes. The initial position of the agent is randomly generated by the AI2-THOR framework, and then the navigation algorithm is trained and tested. We use 0.5m to discretize the environment, meaning the distance between each location is 0.5m.

4.2 Implementations and Evaluation Metircs

We use PyTorch to implement the framework partly based on the public implementation [29]. When learning navigation strategies, we use -0.01 to punish each action step. If an agent reaches a goal and sends a termination signal $DONE$, we will reward the agent with a reward value of 10. We use the Adam optimizer [11] to update the network parameters with a learning rate of $1e^{-4}$. In order to make meaningful comparison, we use the same hyperparameters in each experiment, such as episode number and reward function. We also use pretrained models like ResNet18 during the training process to speed up the process.

In this article, we refer to [1] and use two indicators to evaluate our method: $Success\,Rate\,(SR)$ and $Success\,weighted\,by\,Path\,Length\,(SPL)$. SR is defined as the ratio of the number of times that the agent successfully navigates to the target to the total number of episodes:

$$SR = \frac{1}{N}\sum_{i=1}^{N} S_i \tag{3}$$

where N is the total number of episodes, S_i is the binary indicator of whether the ith episode is successful.

SPL is called normalized inverse path length weighted success, which considers both success rate and optimal path length:

$$SPL = \frac{1}{N}\sum_{i=1}^{N} S_i \frac{l_i}{p_i} \tag{4}$$

where N is the number of evaluations. $S_i = 1$ if the evaluation is successful, otherwise 0. l_i represents the length of the shortest path between the agent's starting position and one of its successful states, and p_i is the length of the current episode. The length used here is the number of operations, which means that performing an operation will increase the length by 1. This indicator can balance the length of the episode and the success rate.

4.3 Comparison Models

Here we describe models that are evaluated and compared in experiments. The following models are used: **Random Policy.** The agent randomly selects an action from the action set A at each time step, which is the simplest navigation method. **Pure DRL agent.** The classical deep reinforcement learning algorithm A3C is used to complete the navigation task. **Target-Driven.** This corresponds to the visual navigation model proposed by Zhu et al. [29]. They use the visual features from the last observation and the target image as input to predict the next action. **Scene Priors.** It uses prior knowledge in the form of a knowledge graph of object relationships to navigate. **SAVN.**The agent constantly understands its environment through the interactive loss function [24] in this model, even during inference time.

Table 1. Comparison with state-of-the-art models. We use the metrics of *Success Rate* (%) and *Success weight by Path Length* (%).

Methods	ALL		L≥5	
	SR	SPL	SR	SPL
Random Policy	10.1	2.2	0.8	0.3
Pure DRL	21.8	6.9	12.1	6.5
Target-Driven [29]	37.0	11.9	25.7	11.2
Scene Priors [26]	35.6	10.7	22.9	10.4
SAVN [24]	37.2	11.2	26.7	10.3
Ours	**55.9**	**19.5**	**49.5**	**19.1**

4.4 Results

Quantitative Results We show the evaluation results of five comparison models in Table 1, where $L \geq 5$ means that the length of the optimal navigation path is more than 5 steps. In order to make a fair comparison, when measuring the performance of this methods, we use the same episodes for training and test them on the same test set. The starting position of the agent is random. Table 1 shows the result on unseen tasks. We can see that the best results are obtained by using our SSR model, and its performance is better than Scene Priors and Target-Driven models. The SR of our model in unknown scene is 56%, SPL to 19%, which is better than other methods.

Table 2. The effect of action *DONE* on navigation in different scenes.

Settings	Kitchen		Living Room		Bedroom		Bathroom		Average	
	SR	SPL	SR	SPL	SR	SPL	SR	SPL	SR	SPL
with *DONE*	65.2	21.8	50.8	17.7	39.1	14.1	79.9	24.8	55.9	19.5
without *DONE*	84.0	45.8	72.8	39.8	60.7	33.1	88.9	51.7	76.6	42.6

Since we use unseen scenes as test set, the experimental results in Table 1 also verify the generalization ability of our model for unknown situations. In fact, Target-Driven model's original extraction of visual features for positioning ability gradually lost because of multi-layer full connection layer. The generalization performance of the model for new targets is also reduced. Scene priors uses the knowledge graph of object relations to extract object relations, but fails to consider the hierarchical relationships between objects. However, we use different modules to learn local and global visual features, and combine them with target semantic information and prior knowledge graph respectively. By using the implicit spatial semantic relations, our model can overcome the shortcomings of previous methods and make the object search easier.

We also evaluated our model with different stop criteria. In this case, the agent does not just rely on its *DONE* actions to learn to terminate. On the contrary, it stops even when the environment signals that the target object has been found. Table 2 shows the evaluation results with and without *DONE* signal respectively. If we don't use *DONE* signal, the average SR of our model in four

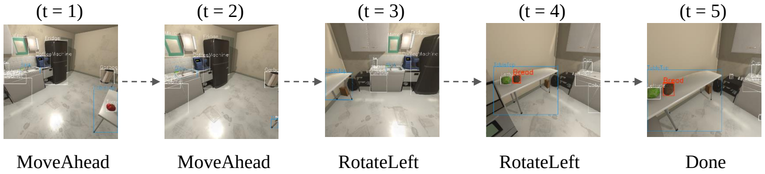


Fig. 3. Navigation example of SSR model.

scenarios is about 76.6%. That's because in this simple environment, the agent will stop automatically when it reaches the goal.

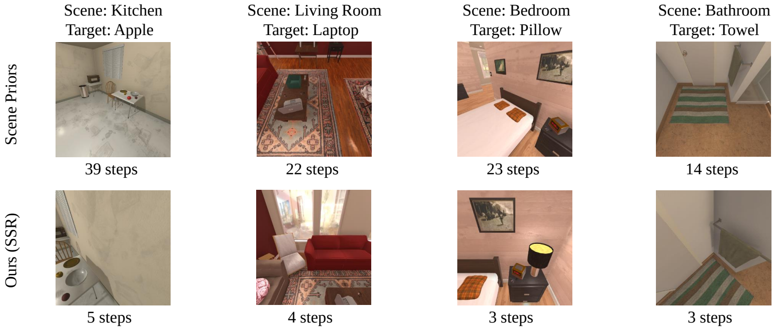


Fig. 4. Compsrison between Ours (SSR) and Scene Priors

Case Study Fig. 3 is an example of the agent's view sequence when navigating to Bread in the Kinchen scene. The target object is displayed in red rectangle, and the objects detected in current observation are displayed in white box. If the target object exists in the object list of the current frame, the geometric distance between the agent and the target is less than the threshold, and the next decision of the agent is $DONE$, the navigation is considered successful and the episode is ended. Otherwise, the similarity between the object detected in the current frame and the target object will be calculated, and the spatial relation vector will be modified to make the next decision. Fig. 4 shows the number of steps required for an agent to navigate to the same target object in four types of scenes. We compare our SSR model with Scene Priors model in unseen environment. Under the same setting, our model can achieve the target position with less operations.

5 Conclusion

We propose an effective target-driven visual navigation method. By learning the spatial visual semantic features and the prior relationship knowledge graph of the scene, our agent is capable of localizing target effectively. In our method, we extract the global and local visual features separately of the observation image through different network modules. Thus we get a joint representation of the spatial relationship to learn the potential connection between the target and the observation. Experiments demonstrate that our method provides obvious advantages for generalizing invisible scenes and targets in navigation.

References

1. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., et al.: On evaluation of embodied navigation agents. arXiv preprint arXiv:1807.06757 (2018)
2. Chaplot, D.S., Gandhi, D.P., Gupta, A., Salakhutdinov, R.R.: Object goal navigation using goal-oriented semantic exploration. Advances in Neural Information Processing Systems **33** (2020)
3. Defferrard, M., Bresson, X., Vandergheynst, P.: Convolutional neural networks on graphs with fast localized spectral filtering. arXiv preprint arXiv:1606.09375 (2016)
4. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. IEEE (2009)
5. Gori, M., Monfardini, G., Scarselli, F.: A new model for learning in graph domains. In: Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005. vol. 2, pp. 729–734. IEEE (2005)
6. Gupta, S., Davidson, J., Levine, S., Sukthankar, R., Malik, J.: Cognitive mapping and planning for visual navigation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2616–2625 (2017)
7. Hamrick, J.B., Allen, K.R., Bapst, V., Zhu, T., McKee, K.R., Tenenbaum, J.B., Battaglia, P.W.: Relational inductive bias for physical construction in humans and machines. arXiv preprint arXiv:1806.01203 (2018)
8. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
9. Kawamoto, T., Tsubaki, M., Obuchi, T.: Mean-field theory of graph neural networks in graph partitioning. Journal of Statistical Mechanics: Theory and Experiment **2019**(12), 124007 (2019)
10. Kim, D., Kim, S., Kwak, N.: Textbook question answering with multi-modal context graph understanding and self-supervised open-set comprehension. arXiv preprint arXiv:1811.00232 (2018)
11. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
12. Kolve, E., Mottaghi, R., Han, W., VanderBilt, E., Weihs, L., Herrasti, A., Gordon, D., Zhu, Y., Gupta, A., Farhadi, A.: Ai2-thor: An interactive 3d environment for visual ai. arXiv preprint arXiv:1712.05474 (2017)
13. Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.J., Shamma, D.A., et al.: Visual genome: Connecting language and vision using crowdsourced dense image annotations. International journal of computer vision **123**(1), 32–73 (2017)
14. Lu, Y., Chen, Y., Zhao, D., Li, D.: Mgrl: Graph neural network based inference in a markov network with reinforcement learning for visual navigation. Neurocomputing **421**, 140–150 (2021)
15. Martins, R., Bersan, D., Campos, M.F., Nascimento, E.R.: Extending maps with semantic and contextual object information for robot navigation: a learning-based framework using visual and depth cues. Journal of Intelligent & Robotic Systems **99**(3), 555–569 (2020)
16. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. arXiv preprint arXiv:1301.3781 (2013)

17. Mirowski, P., Grimes, M.K., Malinowski, M., Hermann, K.M., Anderson, K., Teplyashin, D., Simonyan, K., Kavukcuoglu, K., Zisserman, A., Hadsell, R.: Learning to navigate in cities without a map. arXiv preprint arXiv:1804.00168 (2018)
18. Mnih, V., Badia, A.P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., Kavukcuoglu, K.: Asynchronous methods for deep reinforcement learning. In: International conference on machine learning. pp. 1928–1937. PMLR (2016)
19. Mousavian, A., Toshev, A., Fišer, M., Košecká, J., Wahid, A., Davidson, J.: Visual representations for semantic target driven navigation. In: 2019 International Conference on Robotics and Automation (ICRA). pp. 8846–8852. IEEE (2019)
20. Pathak, D., Mahmoudieh, P., Luo, G., Agrawal, P., Chen, D., Shentu, Y., Shelhamer, E., Malik, J., Efros, A.A., Darrell, T.: Zero-shot visual imitation. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops. pp. 2050–2053 (2018)
21. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767 (2018)
22. Wen, S., Zhao, Y., Yuan, X., Wang, Z., Zhang, D., Manfredi, L.: Path planning for active slam based on deep reinforcement learning under unknown environments. Intelligent Service Robotics pp. 1–10 (2020)
23. Wijmans, E., Kadian, A., Morcos, A., Lee, S., Essa, I., Parikh, D., Savva, M., Batra, D.: Dd-ppo: Learning near-perfect pointgoal navigators from 2.5 billion frames. arXiv preprint arXiv:1911.00357 (2019)
24. Wortsman, M., Ehsani, K., Rastegari, M., Farhadi, A., Mottaghi, R.: Learning to learn how to learn: Self-adaptive visual navigation using meta-learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6750–6759 (2019)
25. Wu, Q., Xu, K., Wang, J., Xu, M., Gong, X., Manocha, D.: Reinforcement learning-based visual navigation with information-theoretic regularization. IEEE Robotics and Automation Letters **6**(2), 731–738 (2021)
26. Yang, W., Wang, X., Farhadi, A., Gupta, A., Mottaghi, R.: Visual semantic navigation using scene priors. arXiv preprint arXiv:1810.06543 (2018)
27. Yu, J., Su, Y., Liao, Y.: The path planning of mobile robot by neural networks and hierarchical reinforcement learning. Frontiers in Neurorobotics **14** (2020)
28. Zhang, Y., Guo, Z., Lu, W.: Attention guided graph convolutional networks for relation extraction. arXiv preprint arXiv:1906.07510 (2019)
29. Zhu, Y., Mottaghi, R., Kolve, E., Lim, J.J., Gupta, A., Fei-Fei, L., Farhadi, A.: Target-driven visual navigation in indoor scenes using deep reinforcement learning. In: 2017 IEEE international conference on robotics and automation (ICRA). pp. 3357–3364. IEEE (2017)