In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Prague, Czech Republic, 27 Sep-01 Oct 2021.

Robotic Occlusion Reasoning for Efficient Object Existence Prediction

Mengdi Li[1], Cornelius Weber[1], Matthias Kerzel[1], Jae Hee Lee[1], Zheni Zeng[2], Zhiyuan Liu[2], Stefan Wermter[1]

***Abstract*— Reasoning about potential occlusions is essential for robots to efficiently predict whether an object exists in an environment. Though existing work shows that a robot with active perception can achieve various tasks, it is still unclear if occlusion reasoning can be achieved. To answer this question, we introduce the task of robotic object existence prediction: when being asked about an object, a robot needs to move as few steps as possible around a table with randomly placed objects to predict whether the queried object exists. To address this problem, we propose a novel recurrent neural network model that can be jointly trained with supervised and reinforcement learning methods using a curriculum training strategy. Experimental results show that 1) both active perception and occlusion reasoning are necessary to successfully achieve the task; 2) the proposed model demonstrates a good occlusion reasoning ability by achieving a similar prediction accuracy to an exhaustive exploration baseline while requiring only about 10% of the baseline's number of movement steps on average; and 3) the model generalizes to novel object combinations with a moderate loss of accuracy.**

I. INTRODUCTION

Indoor assistant robots that are able to perform tasks, such as searching for objects and answering questions about the environment, according to verbal commands from users have promising application prospects. We expect robots to not only complete these tasks correctly but also complete them efficiently, which benefits improving user experience and reduces energy requirements.

The ability of reasoning about potential occlusions of objects is essential for achieving the aforementioned goal. When asked to search for an object, a robot needs to reason whether the target object is possibly occluded by visible objects, and then determine whether to check the occluded space by executing movement actions. However, occlusion reasoning is non-trivial: a robot needs to know the size of the target object from the verbal instruction, and compare it with the size of the visible objects to perform occlusion reasoning. Though existing work has shown that robots with active perception can achieve various tasks [1], [2], [3], [4], in this work we further investigate if robots can efficiently explore environments by performing occlusion reasoning.

To answer this question, we propose a novel robotic object existence prediction (ROEP) task. Fig. 1 shows the task in real scenarios, and a simulation environment built using the robot simulator CoppeliaSim [5]. The robot is the humanoid Pepper[1]

[1]Knowledge Technology, Department of Informatics, University of Hamburg, 22527 Hamburg, Germany. {mli, weber, kerzel, lee, wermter}@informatik.uni-hamburg.de

[2]Department of Computer Science and Technology, Tsinghua University, 100084 Beijing, China. zzn20@mails.tsinghua.edu.cn, liuzy@tsinghua.edu.cn

[1]https://www.softbankrobotics.com/emea/en/pepper.

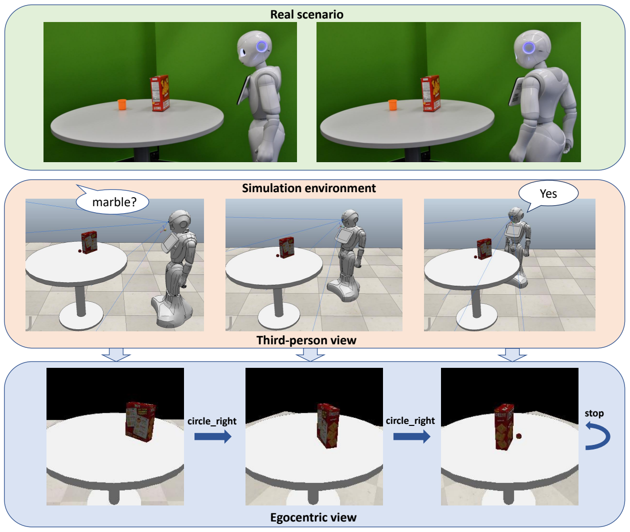

Fig. 1. The task of robotic object existence prediction: given a word instruction (e.g. "marble"), a robot standing by a table needs to execute as few movement steps as possible to give a correct prediction (e.g. yes) whether the queried object exists on the table.

from SoftBank Robotics, which has three omnidirectional wheels for flexible locomotion. The movement of the robot is implemented as a circular motion around the table by 30 degrees clockwise or anticlockwise. The robot receives a word instruction (e.g. "marble"), and is rewarded for correctly predicting whether the target object exists on the table while executing as few movement steps as possible. There are three main challenges behind achieving this goal: 1) the robot needs to connect linguistic concepts with visual representations; 2) the robot needs to memorize the past interactions with the environment to make action selection decisions; and 3) the selected actions and the final prediction functionally interact with each other, which makes the training difficult.

We propose a novel model (see Fig. 2) to address the above challenges. This model is a recurrent neural network consisting of five modules: a visual perception module, a word embedding module, a memory module, an action selection module, and an existence prediction module. The model can be jointly trained with reinforcement learning and supervised learning methods using a curriculum training strategy [6].

We evaluate our model by comparing it with three baselines, which are a passive model without any movement, a random model with a stochastic movement selection strategy, and an exhaustive exploration model which takes a maximum number of movements. Experimental results demonstrate that our model can outperform the passive and random baselines by a large margin, and achieve a similar prediction accuracy to the exhaustive exploration model while requiring only about 10% of the baseline's number of movement steps on average. This shows the necessity of active perception and occlusion reasoning to successfully achieve the task, and that a good occlusion reasoning ability is obtained by our model.

As the number of different objects increases, the number of possible combinations of two objects with occlusion increases exponentially. So a good generalization performance on novel combinations of two objects is especially important for occlusion reasoning. We evaluate the generalization performance of our model on novel object combinations held out from the training data, where we show that the generalization to novel object combinations comes with a moderate loss of accuracy while maintaining a small average number of movement steps. Moreover, the generalization performance increases when more kinds of object combinations are included in the training data.

The main contributions of the paper can be summarized as follows: 1) we formulate a novel robotic object existence prediction (ROEP) task, which poses a high requirement of active perception and occlusion reasoning ability for robots; 2) we develop a novel model that can efficiently achieve this task; and 3) we find that the proposed model generalizes to novel object combinations with a moderate loss of accuracy, and that the variety of object combinations in the training data benefits increasing generalization.

II. RELATED WORK

Mobile robots with active perception: Zhu et al. [1] proposed a reinforcement learning model for the task of target-driven visual navigation. The model is expected to navigate towards a visual target in indoor scenes with a minimum number of movement steps by its egocentric visual inputs and the image of the target. Ye et al. [2] studied the problem of mobile robots searching small target objects in arbitrary poses in indoor environments. They proposed a model integrating an object recognition module and a deep reinforcement learning-based action selection module together for the object searching task. Wang et al. [3] focused on the efficiency of robots when searching for target objects. They proposed a scheme to encode the prior knowledge of the relationship between rooms and objects in a belief map to facilitate efficient searching. Instead of focusing on achieving tasks in large-scale indoor environments, we concentrate on the efficiency of the model when encountering occlusion situations.

Object occlusion: The occlusion situation between objects is very common in robotic scenarios. However, the occlusion reasoning ability of autonomous mobile robots has not been studied well. Yang et al. [4] introduced the task of embodied amodal recognition focusing on the visual recognition ability of agents in scenes with occlusion. They proposed a model that can navigate in the environment to perform object classification, location, and segmentation. However, this work did not concentrate on the occlusion reasoning ability of the agent. A recent work on developing robots with the occlusion reasoning ability is [7]. This work introduced the task of answering visual questions via manipulation (MQA), where a robot manipulator needs to perform a series of actions to move objects possibly occluding some small objects on a tabletop, in order to correctly answer visual questions. Similar to the ROEP task, the MQA task also requires the robot to have the ability of occlusion reasoning to perform reasonable exploration actions. However, the robot in MQA is a manipulator that explores the environment by moving objects, while we focus on autonomous mobile robots to explore the environment by active perception.

TABLE I
OBJECTS USED IN THE SIMULATION ENVIRONMENT

Category	Objects			
Large	cracker_box desktop_plant	cleanser wine	laptop teddy_bear	pitcher
Medium	apple rubiks_cube	baseball meat_can	foam_brick coffee_can	mug
Small	bolt card	dice battery	key button_battery	marble

Embodied learning: Robotics research is recently benefiting from achievements in vision and language processing. On the other hand, researchers are also taking advantage of agents situated in 3D environments to conduct multimodal research. It has been proven that an active agent is able to connect linguistic concepts with visual representations of the environment through training to complete action-involved tasks [8], [9]. Hill et al. [10] found that an embodied agent can achieve one-shot word learning when trained with reinforcement learning in a 3D environment. The proposed ROEP task also involves multimodalities, including vision, language, and action. Different from the abovementioned work, our model needs to specifically connect linguistic concepts with visual representations about object size through training to achieve the ROEP task.

III. ROBOTIC OBJECT EXISTENCE PREDICTION

A. Simulation Environment

Existing simulation environments are not suitable for the ROEP task. We create a corresponding tabletop simulation environment using the robot simulator CoppeliaSim [5] (see Fig. 1). The robot can capture egocentric RGB images by a visual sensor mounted on its head, and execute actions selected from (*circle_left*, *circle_right*, and *stop*). By taking the action *circle_left*, the robot circles around the table clockwise by 30 degrees. The action *circle_right* works in the same way but in an anticlockwise direction. When the action *stop* is selected or the maximum number of 6 movement steps is reached, the robot takes no movement action and predicts whether the queried object exists.

A total of 21 everyday objects are used in the simulation environment. Some of them are from the YCB dataset [11]. The rest of them are provided by CoppeliaSim or collected online. These objects are divided into 3 categories according to their relative size, as shown in Table I. When fitting these objects into cubes, objects from the *Large* category have a

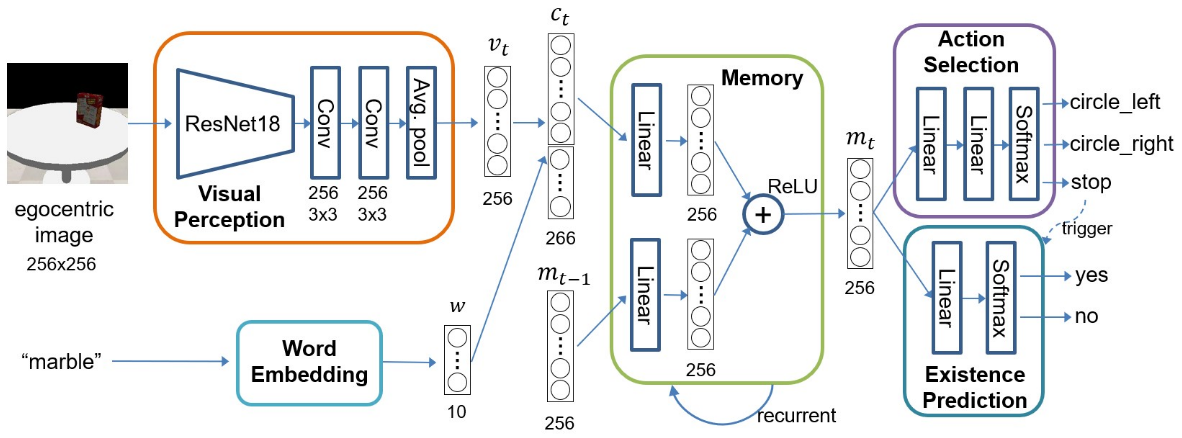

Fig. 2. The architecture of the proposed model.

minimum height of $21cm$ and an average volume of $2905cm^3$. The heights of objects from the *Medium* category are from $5cm$ to $14cm$, and their average volume is $508cm^3$. Objects from the *Small* category have a maximum height of $3cm$ and an average volume of $7cm^3$. There are potential occlusions of objects from different categories.

B. Data Generation

Our data is automatically generated based on predefined rules like the CLEVR [12] and the ShapeWorld [13] datasets. All the samples are generated during training and testing periods. Each data sample is a triplet [*Scene*, *Query*, *Prediction*]. *Scene* is an arrangement of objects on the table. *Query* is a word randomly selected with equal probability from Table I to instruct the robot to search for the referred object in *Scene*. *Prediction* is a ground-truth binary label representing whether the target object exists in *Scene*. It is randomly set as positive or negative with the equal probability of 50%. Based on a determined pair of *Query* and *Prediction*, a corresponding scene is then generated.

There are three different types of scenes: 1) scenes that contain one object; 2) scenes with two objects without occlusion from the initial field of view of the robot; 3) scenes with two objects, one of which is occluded by the other one from the initial field of view of the robot. They account for the same proportion ($1/3$) in the generated data. To generate scenes with one object, the object is randomly placed on the table. To generate scenes with two objects, some geometric calculations using position coordinates of the robot's visual sensor, and both position coordinates and heights of the two objects are applied for controlling whether there is occlusions in generated scenes. It should be noted that the smaller object is not necessarily fully occluded by the larger one in scenes with occlusion.

We have a reasoning table (see Table II) of the ideal action strategy at the first time step in an episode. This table shows whether the robot should move to change its viewpoint or predict the existence of the target object directly when given a query for objects of a specific category (different columns), and the object seen from the initial viewpoint. Except for

TABLE II
REASONING TABLE

	Query		
Visible Object	*Large*	*Medium*	*Small*
One *Large*	predict	move	move
One *Medium*	predict	predict	move
One *Small*	predict	predict	predict

the situation where a *Large* object is queried, or a *Small* object is seen, the robot has to utilize both information from the word instruction and visual perception to make an ideal action decision. Because there are at most two objects on the table, whenever the robot sees two objects, the robot should give an existence prediction directly no matter which object is queried.

IV. METHODOLOGY

A. Model

Our proposed model is inspired by the recurrent attention model [14], which is originally applied to attention-driven image classification tasks. The proposed model is a recurrent neural network overall (see Fig. 2), and can be divided into five parts: 1) a memory module for incrementally building up state representations, 2) a visual perception module for extracting visual representations, 3) a word embedding module for extracting distributed representations of a query word, 4) an action selection module for making action decisions, and 5) an existence prediction module for producing final predictions.

The **Visual Perception** module takes the egocentric RGB image (256×256 pixels) as input to extract visual representations. It first extracts the 128 $28{\times}28$ image feature maps from the *conv3* layer of a fixed ResNet18 [15] pretrained on ImageNet [16]. The feature is then passed through two CNN layers both with 256 $3{\times}3$ kernels, and an average pooling layer to obtain the visual representations v_t with a length of 256. This process is similar to the visual module of the MAC model [17] designed for visual reasoning on the CLEVR dataset [12].

The **Word Embedding** module maps each word instruction to a 10-dimensional word vector w. The weights of the embedding module are randomly initialized, and updated during training.

The **Memory** module is a recurrent unit that takes the concatenated representations $c_t = (v_t, w)$ as the input, and combines c_t with the internal representations at the previous time step m_{t-1} to produce the new internal representations $m_t \in \mathbb{R}^{256\times 1}$. This process can be formalized as

$$m_t = f_m(m_{t-1}, c_t) = \text{ReLU}(W_m \cdot m_{t-1} + W_c \cdot c_t + b) \quad (1)$$

where $W_m \in \mathbb{R}^{256\times 256}$ and $W_c \in \mathbb{R}^{256\times 266}$ are weight matrices, $b \in \mathbb{R}^{256\times 1}$ is a bias vector, ReLU$(\cdot)$ is the rectified linear activation function. More sophisticated units such as LSTM or GRU are not used for the memory module because a vanishing gradient is not a problem for our task since only few recurrent steps have to be taken.

The **Action Selection** module and **Existence Prediction** module are both classification networks with *softmax* outputs. The action selection module is a fully connected network with one hidden layer (128 hidden units). Its three *softmax* outputs correspond to three movement actions. The existence prediction module has a single linear layer followed by a *softmax* layer with two outputs which correspond to the positive and negative prediction respectively.

B. *Training*

The parameters of our model include parameters of the visual perception module, the word embedding module, the memory module, the action selection module, and the existence prediction module $\theta = \{\theta_v, \theta_w, \theta_m, \theta_a, \theta_p\}$. The model is non-differential overall. We train the model jointly with supervised learning and reinforcement learning methods, where θ_a is trained using reinforcement learning, $\{\theta_v, \theta_w, \theta_m, \theta_p\}$ are trained using supervised learning.

The task can be formalized as a partially observable Markov decision process from the perspective of reinforcement learning. The true state of the environment cannot be fully observed. The action selection module is a reinforcement learning agent, which needs to learn a stochastic policy $\pi(a_t|s_{0:t}; \theta_a)$ with the parameters θ_a, where a_t is one of the three actions in the predefined action set. Executing each movement action except the *stop* action leads the model to obtain a new visual input. $s_{0:t} = w, v_0, a_0, v_1, a_1, ..., v_t$ is the history of past interactions with the environment from time step 0 to t. The internal representations m_t in the memory module is an approximation to $s_{0:t}$.

The model is expected to gain a high reward at the end of each episode. We design a cost-sensitive reward function containing two parts, an accuracy reward r_{acc} and a latency reward r_{lat}. An accuracy reward of 1 is received when a correct prediction is produced. An accuracy reward of -1 is received when an incorrect prediction is produced. The latency reward is

$$r_{lat} = \frac{1}{T+2} \quad (2)$$

where T is the number of movement steps the agent takes in one episode. $T = 0$ means that the *stop* action is selected at time step 0. The total reward at time step T is a summation of these two rewards: $r_T = r_{acc} + r_{lat}$. We use $T + 2$ rather than $T + 1$ as the denominator of r_{lat} to make sure that r_T is negative when the prediction is incorrect. At other time steps $(t = 0, ..., T - 1)$, we set $r_t = 0$.

The agent is expected to maximize the expected reward return $J(\theta_a)$ under the policy $\pi(a_t|s_{0:t}; \theta_a)$.

$$J(\theta_a) = \mathbb{E}_{\pi(a_t|s_{0:t};\theta_a)}\left[\sum_{t=0}^{T} r_t\right] \quad (3)$$

We use Monte-Carlo policy gradient (REINFORCE) [18] to optimize the agent. REINFORCE uses the sample gradient to approximate the actual gradient of $J(\theta_a)$

$$\nabla_{\theta_a} J \approx \sum_{t=0}^{T} \nabla_{\theta_a} \log \pi(a_t|s_{0:t}; \theta_a)(R_t - b_t) \quad (4)$$

where $R_t = \sum_{t'=0}^{T} r_{t'}$ is the accumulated reward following the action a_t, b_t is the estimated reward predicted by a baseline network, which has a single linear layer taking m_t as the input. The estimated reward b_t is used for reducing variance of gradient estimation. The baseline network is trained with a mean squared error loss $\mathcal{L}_b = \frac{1}{T}\sum_{t=0}^{T}(R_t - b_t)^2$.

To use gradient descent algorithms for optimizing the agent, we define loss $\mathcal{L}_a = -J(\theta_a)$. It should be noted that gradients of $\mathcal{L}_a$ and $\mathcal{L}_b$ are not backpropagated to the memory, visual perception, and word embedding module.

We train these modules along with the existence prediction module using supervised learning methods to optimize the binary cross-entropy loss

$$\mathcal{L}_p = -y \log \hat{y} - (1 - y) \log(1 - \hat{y}) \quad (5)$$

where y is the labeled ground-truth prediction (1 for yes, 0 for no), $\hat{y}$ is the estimated probability of the prediction yes. Gradients of $\mathcal{L}_p$ are backpropagated to update parameters of the existence prediction θ_p, memory θ_m, visual perception θ_v, and word embedding θ_w module.

The total loss function is a weighted summation of the three losses, as

$$\mathcal{L}_{total} = \mathcal{L}_p + \alpha \cdot \mathcal{L}_a + \beta \cdot \mathcal{L}_b \quad (6)$$

where α and β are weight coefficients of $\mathcal{L}_a$ and $\mathcal{L}_b$ respectively.

C. *Training Details*

We found that it is hard to train the model from scratch on data with all three different types of scenes, which corresponds to the finding of [4] that joint training perception and policy networks from scratch is difficult. We resort to a curriculum training strategy to train the model on data with 4 levels of increasing difficulty. We refer to data only containing scenes with one object as *L1-1-vis*, data only containing scenes with two objects without occlusion as *L2-2-vis*, data only

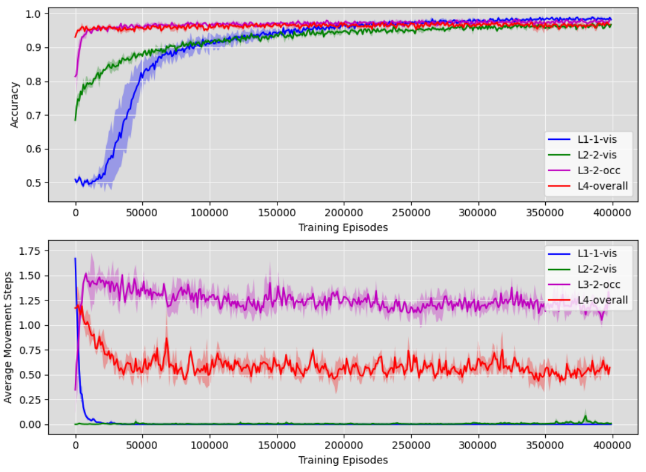

Fig. 3. Training curves of the proposed model in different training stages. The model is sequentially trained on *L1-1-vis*, *L2-2-vis*, *L3-2-occ*, and *L4-overall* data. The parameters obtained from one training stage are loaded as the initial parameters for the next training stage.

containing scenes with two objects with occlusion as *L3-2-occ*, and data containing all types of scenes as *L4-overall*, in which three types of scenes occupy the same proportion. The model is trained on these four levels of data sequentially. The parameters obtained from one training stage are loaded as the initial parameters for the next training stage.

We use the Adam optimizer with a learning rate of $1e-4$. The weight coefficients in the total loss function (Eq. 6) is set as $\alpha = 1e-2$, $\beta = 1$ for training stage on the first three levels. A smaller weight coefficient $\alpha = 1e-4$ is used for the last training level to make the training process stabler.

V. EXPERIMENTS

A. Curriculum Training

Our model is trained using a curriculum training strategy. Specifically, the model is trained sequentially on *L1-1-vis*, *L2-2-vis*, *L3-2-occ*, and *L4-overall* data with a fixed number of episodes ($900k$, $900k$, $400k$, and $400k$ respectively) in our experiments. The total training process takes about four days using one GPU (NVIDIA Titan RTX). We noticed that it is unnecessary to train the model to achieve the best performance in the first three training stages if we are only interested in the final model. We repeat the experiment three times to avoid the effect of randomness. The accuracy of correct predictions and the average number of movement steps are used as metrics to evaluate the performance.

Fig. 3 shows the training curves in different training stages. In the first two training stages on *L1-1-vis* and *L2-2-vis* data, the accuracy increases stably until reaching a plateau of over 97%, while the average number of movement steps stay near 0. In the third training stage on *L3-2-occ* data, the accuracy rapidly increases in the first $30k$ episodes with the rapid increase of the average movement steps. In the last two training stages on *L3-2-occ* and *L4-overall* data, the average movement steps continuously decrease after the accuracy has reached a plateau.

TABLE III

PERFORMANCE EVALUATION ON DIFFERENT TEST DATA

	$Model_{L1}$		$Model_{L2}$		$Model_{L3}$		$Final\ Model$	
Test Data	Acc.	Steps	Acc.	Steps	Acc.	Steps	Acc.	Steps
L1-1-vis	**99.4%**	0.0	90.9%	0.02	88.7%	1.20	99.0%	0.74
L2-2-vis	68.3%	0.0	**97.4%**	0.01	91.2%	0.64	97.1%	0.39
L3-2-occ	74.4%	0.0	77.7%	0.07	**98.3%**	1.23	96.9%	1.02
L4-overall	80.8%	0.0	88.5%	0.03	92.6%	1.00	**97.2%**	0.71

We refer to models obtained from the first three training stages at the $900k$, $900k$, $400k$ episodes as $Model_{L1}$, $Model_{L2}$, $Model_{L3}$ respectively. The final model is obtained from the last training stage at the $400k$ episodes, and denoted as *Final Model*. Performance of each model when tested on different test data ($10k$ episodes) is presented in Table III. The results show that each model scores well on the test data that corresponds to the training statistics (diagonal in bold font), and that the final model performs nearly as well as the individual models on their test data. Fig. 4 shows examples when there is only one object, which is larger than the target object, visible from the initial perspective of the agent. A video showing the experimental results is available at `https://youtu.be/L4p7yo8dMmQ`.

B. Comparison with Baselines

We compare the proposed model with three baselines that have the same architecture as the proposed model, but with different action selection strategies. These baselines include a passive model without any movement, a random model with a stochastic movement selection strategy, and an exhaustive exploration model that executes the *circle_left* action for a maximum number of movement steps before producing a prediction. The average movement steps of the three baselines are 0, 1.82, and 6 respectively.

The prediction accuracy of these baselines and our final model when tested on different test data is presented in Table IV. The passive model and the random model are able to achieve a performance close to that of the exhaustive model on *L1-1-vis*, and *L2-2-vis* data, but performs poorly on *L3-2-occ* data. This reveals that active perception is necessary to address the ROEP task. Our model can achieve a similar accuracy on all test data to the exhaustive model while requiring only 11.8% of the baseline's number of movement steps on average (0.71 steps by our model, 6 steps by the exhaustive model). This demonstrates that our model has obtained a good occlusion reasoning ability. However, there are still some challenges remaining: 1) The model learns to always choose one direction to move, rather than choose the optimal direction according to the orientation of the visible object or partial occlusion to check the occluded space (see Fig. 5); 2) The model moves 0.39 steps on average in scenes without occlusion (*L2-2-vis*), which is unnecessary.

C. Generalization Evaluation

A good generalization performance on novel combinations of two objects is especially important for occlusion reasoning. To evaluate the generalization performance, we train our network on two different sets of training data excluding some

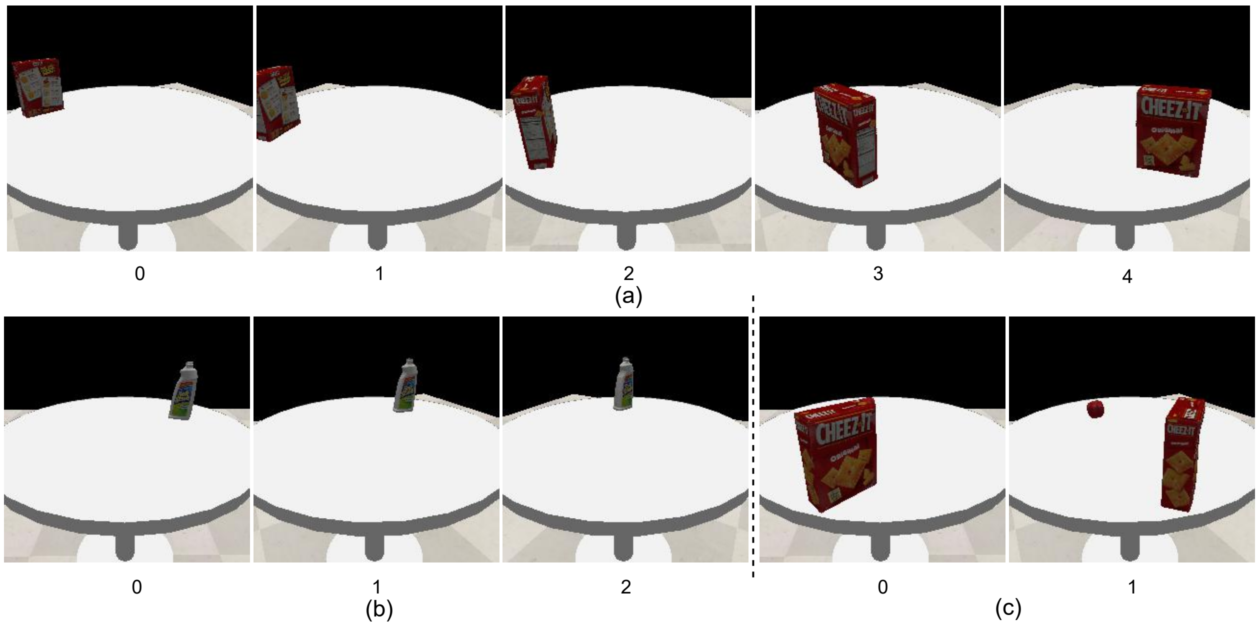

Fig. 4. Examples of egocentric images in an episode when the query is "apple". Numbers below the images indicate the time steps in an episode. (a), (b): only one object larger than the target object exists; (c): the target object is occluded by a larger object. In all cases, the agent moves to check the occluded space and provides the correct answer after the last shown frame.

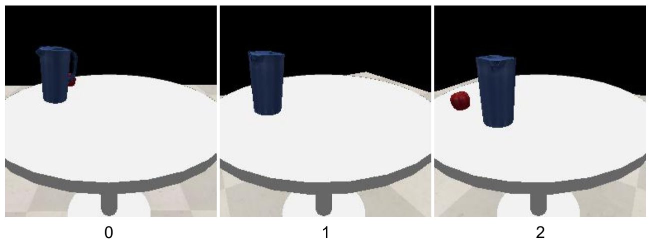

Fig. 5. In this example, an apple is partially occluded by a pitcher. When the query is "apple", the agent does not choose the optimal action *circle_right*, instead it chooses the action *circle_left*.

TABLE IV
PERFORMANCE COMPARISON WITH BASELINES

Test Data	Passive Model	Random Model	Exhaustive Model	Our Model
L1-1-vis	98.0%	97.6%	**99.2%**	99.0%
L2-2-vis	96.1%	92.6%	96.4%	**97.1%**
L3-2-occ	77.6%	83.3%	**97.4%**	96.9%
L4-overall	90.3%	91.1%	**97.4%**	97.2%

object combinations, which are called holdout combinations. That means scenes with some specific object combinations, e.g. [mug, battery], are not included in the training data.

There are three types of combinations of two different size categories, namely [*Large*, *Medium*], [*Large*, *Small*], [*Medium*, *Small*], and 147 $(7 \times 7 \times 3)$ possible combinations of two objects from different size categories. In the first training set, 21 object combinations (7 for each category combination) are held out only for testing, which accounts for 14.3% of all possible combinations. In the second set, 42 object combinations (14 for each category combination) are held out, which accounts for 28.6% of all possible combinations. Holdout combinations are determined by randomly selecting from all possible object combinations before the start of training. Every object in Table I is shown in the training data. Experiments are repeated three times with different holdout combinations and random initialization.

TABLE V
GENERALIZATION EVALUATION

	21 holdout		*42 holdout*	
Test Data	Acc.	Steps	Acc.	Steps
L1-1-vis	98.6%	0.651	98.8%	0.629
L2-2-vis (training)	96.8%	0.262	97.2%	0.197
L3-2-occ (training)	94.7%	0.873	96.2%	0.892
L2-2-vis (holdout)	95.8%	0.399	92.7%	0.317
L3-2-occ (holdout)	91.5%	0.895	86.7%	0.841

Table V presents the test results of the models trained on aforementioned two sets of training data, denoted as *21 holdout* and *42 holdout* respectively. Test data *L2-2-vis (training)* and *L3-2-occ (training)* contain scenes with object combinations used for training. Test data *L2-2-vis (holdout)* and *L3-2-occ (holdout)* only contain scenes with holdout object combinations. The results show that the two models can achieve similar high performance on scenes with object combinations used for training. When tested on *L2-2-vis (holdout)* and *L3-2-occ (holdout)*, the model trained on *21 holdout* can still work well with an accuracy of over 90% and a small average number of movement steps. The performance of the model trained on *42 holdout* drops moderately to 86.7% accuracy when tested on *L3-2-occ (holdout)*, where occlusion reasoning on novel combinations is necessary.

VI. Discussion

Experimental setup: The current experimental setup is simplified. There is a strong prior that there are at most two objects existing on the table, which limits the complexity of potential occlusion situations. An interesting extension is to generate scenes with more objects on the table and extend the task to counting objects. Moreover, the action space of the robot is small. The actions of *circle_left* and *circle_right* used in the current experimental setting limits the generalization capability to environments with tables of different sizes or shapes. More complex robot actions also involving *move_ahead*, *rotate_left*, *rotate_right*, etc., will be used in future work. On the one hand, it will be feasible to transfer a robot with these more complex actions to other environments. On the other hand, it will make the task more challenging, as the robot has greater flexibility in its movements, which places higher demands on action planning.

Training complexity: The current training process is complex, since the curriculum training strategy involves four sequential training stages to obtain the final model. A possible solution to simplify training is using unsupervised learning [19] instead of curriculum learning to learn good visual and word representations.

Sim-to-real transfer: In this paper, we validate the effectiveness of the proposed model in a simulation environment. We can imagine that directly transferring the resulting model trained in a simulation environment to a real-world scenario (see Fig. 1) would result in a certain performance loss. Some techniques, such as fine-tuning the model in a more photo-realistic simulation environment with randomized lighting conditions of the real environment, may mitigate the performance degradation.

VII. CONCLUSION

In this work, we introduced the task of robotic object existence prediction (ROEP), which is complementary to existing robotic tasks that require active perception. Different from existing tasks, ROEP focuses on the occlusion reasoning ability, which helps a robot explore its environments more efficiently. As such, it can be used for example as a probing task for existing models that are designed for general tasks.

To solve ROEP we proposed a novel recurrent neural network which is trained end-to-end jointly with reinforcement learning and supervised learning methods using a curriculum training strategy. We showed empirically that the proposed model can efficiently achieve the ROEP task compared with the baselines. We also showed that generalization to novel object combinations comes with a moderate loss of accuracy, while including more kinds of object combinations in the training data can increase the generalization performance. This finding, which is related to the finding in [20], can be considered as a recommendation when training a model for tasks that implicitly involve occlusion reasoning (e.g., object goal navigation [21]).

Acknowledgement. The authors gratefully acknowledge support from the China Scholarship Council (CSC) and the German Research Foundation DFG under project CML (TRR 169). We would like to thank Erik Strahl for his support with the experimental setup.

REFERENCES

[1] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, “Target-driven visual navigation in indoor scenes using deep reinforcement learning,” in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 3357–3364.

[2] X. Ye, Z. Lin, H. Li, S. Zheng, and Y. Yang, “Active object perceiver: Recognition-guided policy learning for object searching on mobile robots,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 6857–6863.

[3] C. Wang, J. Cheng, J. Wang, X. Li, and M. Q.-H. Meng, “Efficient object search with belief road map using mobile robot,” *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3081–3088, 2018.

[4] J. Yang, Z. Ren, M. Xu, X. Chen, D. Crandall, D. Parikh, and D. Batra, “Embodied amodal recognition: Learning to move to perceive objects,” in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 2040–2050.

[5] E. Rohmer, S. P. N. Singh, and M. Freese, “CoppeliaSim (formerly V-REP): a versatile and scalable robot simulation framework,” in *Proceedings of The International Conference on Intelligent Robots and Systems (IROS)*, 2013, www.coppeliarobotics.com.

[6] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, “Curriculum learning,” in *Proceedings of the 26th Annual International Conference on Machine Learning (ICML)*, 2009, pp. 41–48.

[7] Y. Deng, D. Guo, X. Guo, N. Zhang, H. Liu, and F. Sun, “MQA: Answering the question via robotic manipulation,” in *Proceedings of Robotics: Science and Systems (RSS)*, 2021.

[8] K. M. Hermann, F. Hill, S. Green, F. Wang, R. Faulkner, H. Soyer, D. Szepesvari, W. M. Czarnecki, M. Jaderberg, D. Teplyashin, *et al.*, “Grounded language learning in a simulated 3D world,” *arXiv preprint arXiv:1706.06551*, 2017.

[9] D. S. Chaplot, K. M. Sathyendra, R. K. Pasumarthi, D. Rajagopal, and R. Salakhutdinov, “Gated-attention architectures for task-oriented language grounding,” in *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, vol. 32, no. 1, 2018.

[10] F. Hill, O. Tieleman, T. von Glehn, N. Wong, H. Merzic, and S. Clark, “Grounded language learning fast and slow,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021.

[11] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, “Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols,” *Proceedings of the 2015 IEEE International Conference on Advanced Robotics (ICAR)*, 2015.

[12] J. Johnson, B. Hariharan, L. Van Der Maaten, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick, “CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2901–2910.

[13] A. Kuhnle and A. Copestake, “ShapeWorld - A new test methodology for multimodal language understanding,” *arXiv preprint arXiv:1704.04517*, 2017.

[14] V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu, “Recurrent models of visual attention,” in *Proceedings of the International Conference on Neural Information Processing Systems (NeurIPS)*, 2014, p. 2204–2212.

[15] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[16] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, *et al.*, “ImageNet large scale visual recognition challenge,” *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[17] D. A. Hudson and C. D. Manning, “Compositional attention networks for machine reasoning,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[18] R. J. Williams, “Simple statistical gradient-following algorithms for connectionist reinforcement learning,” *Machine Learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[19] D. Ha and J. Schmidhuber, “World models,” *arXiv preprint arXiv:1803.10122*, 2018.

[20] F. Hill, A. Lampinen, R. Schneider, S. Clark, M. Botvinick, J. L. McClelland, and A. Santoro, “Environmental drivers of systematicity and generalization in a situated agent,” in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[21] D. S. Chaplot, D. Gandhi, A. Gupta, and R. Salakhutdinov, “Object goal navigation using goal-oriented semantic exploration,” in *In Neural Information Processing Systems (NeurIPS)*, 2020.