Curriculum Reinforcement Learning From Avoiding Collisions to Navigating Among Movable Obstacles in Diverse Environments

Hsueh-Cheng Wang, *Member, IEEE*, Siao-Cing Huang, Po-Jui Huang, Kuo-Lun Wang, Yi-Chen Teng, Yu-Ting Ko, Dongsuk Jeon, *Member, IEEE*, and I-Chen Wu, *Senior Member, IEEE*

***Abstract*—Curriculum learning has proven highly effective to speed up training convergence with improved performance in a variety of tasks. Researchers have been studying how a curriculum can be constituted to train reinforcement learning (RL) agents in various application domains. However, discovering curriculum sequencing requires the ranking of sub-tasks or samples in order of difficulty, which is not yet sufficiently studied for robot navigation problems. It is still an open question what navigation strategies can be learned and transferred during multi-stage transfer learning from easy to hard. Furthermore, despite of some attempts of learning real robot manipulation tasks using curriculum, most of existing works are limited to toy or simulated settings rather than realistic scenarios. To address those issues, we first investigated how the model convergence in diverse environments relates to the navigation strategies and difficulty metrics. We found that only some of the environments can be trained from scratch, such as in a relatively open tunnel-like environment that only required wall following. We then carried out a two-stage transfer learning for more difficult environments. We found such approach effective for goal navigation, but failed for more complex tasks where movable obstacles may be on the navigation path. To facilitate more complex policies in the navigation among movable obstacles (NAMO) task, another curriculum with distance and pace functions appropriate to the difficulty of the environment was developed. The proposed scheme was proved effective and the strategies learned were discussed via comprehensive evaluations conducted in simulated and real environments.**

***Index Terms*—Collision avoidance, curriculum learning, deep reinforcement learning, movable obstacles, search and rescue robots.**

Manuscript received 30 August 2022; accepted 9 February 2023. Date of publication 1 March 2023; date of current version 31 March 2023. This letter was recommended for publication by Associate Editor P.-C. Lin and Editor H. Moon upon evaluation of the reviewers' comments. This work was supported in part by Taiwan's National Science and Technology Council (NSTC) under Grants 111-NU- E-A49-001-NU, 111-2634-F-A49-013, and 111-2623-E-A49-007 and in part by Qualcomm through the Taiwan University Research Collaboration Project. *(Corresponding author: Hsueh-Cheng Wang.)*

Hsueh-Cheng Wang, Siao-Cing Huang, Po-Jui Huang, Kuo-Lun Wang, Yi-Chen Teng, Yu-Ting Ko, and I-Chen Wu are with the National Yang Ming Chiao Tung University (NYCU), Hsinchu 30010, Taiwan (e-mail: hchengwang@nycu.edu.tw; yellow.gdr09g@nctu.edu.tw; rayhuang.ee09@nycu.edu.tw; kuolun wang.ee08@nycu.edu.tw; m88215.en11@nycu.edu.tw; yutingk.ee11@nycu.edu.tw; icwu@cs.nctu.edu.tw).

Dongsuk Jeon is with the Graduate School of Convergence Science and Technology, the Research Institute for Convergence Science, and the Inter-University Semiconductor Research Center, Seoul National University, Seoul 08826, South Korea (e-mail: djeon1@snu.ac.kr).

Supplementary materials can be found at https://arg-nctu.github.io/projects/curl-navi.html.

Digital Object Identifier 10.1109/LRA.2023.3251193

I. Introduction

INTELLIGENT agents capable of operating in the human environment have been studied for decades; however, existing systems based solely on navigation are unable to find a free path through areas blocked by movable obstacles. Actions such as opening doors and pushing movable obstacles require additional manipulators. Some of the initial studies on navigation among movable objects (NAMO) problems [1], [2], [3], [4] used the geometry of objects and scenes to select a set of action primitives; however, autonomously controlling the mobile base and manipulators involves a complex series of decisions.

RL makes it possible for agents to maximize future expected rewards through the selection of actions specific to the scenarios, based on incoming sensory observations. Recent studies have sought to train general-purpose neural networks using RL and leverage prior knowledge to alleviate the difficulties in learning new tasks. RL has been used to automate a wide variety of robotic tasks. However, most modern RL schemes learn every task from scratch, which requires large amounts of data and considerable time to process it. It might be possible to side-step the process of exploration by collecting human demonstrations; however, collecting sufficient high-quality data to attain high-level performance would be very difficult. Furthermore, the best performance of the resulting model would be limited by the best demonstration in a given dataset. Imitation learning via behavior cloning [5] is a simple and effective approach to efficient learning; however, the resulting policies are prone to "compounding errors," in which even small mistakes can spiral out of control and force the agent into unexpected states. Controlling a mobile platform and a manipulator jointly requires more complex controls. Such difficulty of using a high-dimensional action space has prompted some researchers to adopt a series of sub-goals and/or sample-based motion planning for the interactive navigation problem [6], which keeps tracking the states of a target object. Other researchers have combined high-level policies (to generate sub-goals) to guide low-level policies (to complete the action) [7]. In this study, we use curriculum learning to solve the NAMO problem, which aims at navigating the agent to a goal by pushing away movable obstacles but not tracking them. We seek a sample efficient method of exploration when using continuous control signals to form a continuous action space.

RL has also been combined with curriculum learning to train agents via a series of tasks that gradually increase in difficulty. In [8], curriculum learning was used to find increasingly distant destinations in a city from a small area until it covered the entire city. In [9], the authors proposed a 3-stage RL curriculum to

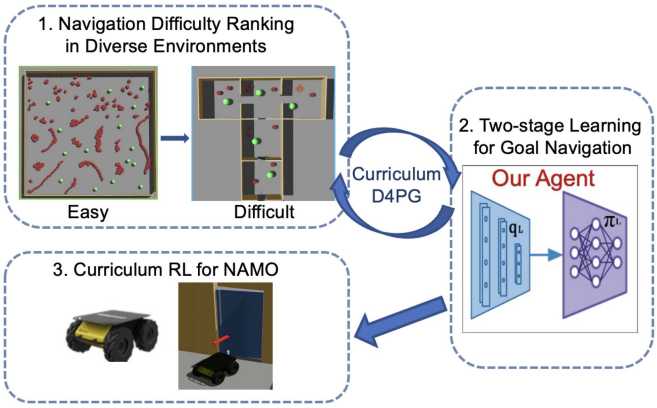


Fig. 1. Our approach. 1) Navigation difficulty ranking in diverse Environments, 2) two-stage transfer learning for goal navigation, and 3) curriculum RL for a more complex navigation among movable obstacles (NAMO) problem.

tackle the problem of autonomous overtaking in racecar simulators. The trained policy outperformed the built-in AI, achieving overtaking performance comparable to that of an experienced human driver. In [10], an agent was trained via curriculum learning to navigate through a Minecraft simulation environment, gradually progressing from simple to more complex navigation problems, including avoiding lava and passing through narrow gates. The authors proposed a teacher-student framework, wherein the student is trained as a RL agent and the teacher algorithm assigns suitable training sub-tasks in accordance with the training score associated with the student. Our work is built upon the above studies including: 1) a curriculum of increasing distance to goal, 2) a multi-stage learning paradigm, and 3) a T-shaped maze similar to [10].

Note, however, that curriculum learning requires the ranking of sub-tasks in order of difficulty, which researchers have yet to explore. To our knowledge, none of the previous works address the curriculum (sequence of training samples/tasks/environments) of navigation and NAMO problems in large-scale, unseen environments. Fig. 1 illustrates the proposed approaches. The contributions of this study are outlined as follows:

- *Investigations of model convergence with respect to navigation difficulty for a two-stage curriculum RL:* We collected a set of diverse simulation environments, and calculate navigation difficulty using the metrics in [11]. We found the deep RL model convergence of training from scratch correlates to those metrics. For goal navigation tasks, the two-stage easy-to-hard curriculum training process allows the agent to reach high rewards, which is not achieved if the model were trained from scratch.
- *Curriculum RL for a complex NAMO task:* We carried out two curriculum designs, including two-stage training and curriculum sub-steps in additional environments with movable obstacles. The results revealed that both designs were required to optimize the policy of more complex tasks (e.g., passing narrow gates and interacting with doors).
- *Comprehensive evaluations in simulated and real environments compared to a state-of-the-art method:* The proposed deep RL agent was evaluated in unseen environments for goal navigation and NAMO tasks. We found competitive results compared with a state-of-the-art local planner [12]. Strategies learned by the policy were discussed.

II. Related Work

Curriculum learning was inspired by the way that humans learn, wherein the complexity of the underlying architecture increases as learning progresses [13]. Model training can be accelerated by eliminating the need to deal with noise, while providing guidance toward higher reward regions of the parameter space. The benefits of a "starting small" strategy for machine learning were elucidated in [14]. In that study, they determined that curriculum learning facilitates model training by minimizing the time spent dealing with noise, while guiding the training process toward better regions within the parameter space. Curriculum learning provides two important benefits: 1) increased convergence speed in training; and 2) improved performance. The intuition behind why curriculum learning helps is that the switching of an "easy" data source to a "difficult" one shakes up the basins of attractions of the loss landscape during training as discussed in [15]. There have been a wide range of training problems that have been successfully dealt with using curriculum learning (see [16], [17] for surveys).

A number of studies on robotics tasks have focused on the application of RL in conjunction with Hindsight Experience Replay (HER), which was classified as task-level curriculum learning in [16]. HER is a simple off-policy RL algorithm using *sparse rewards* involved in manipulating the replay buffer to enhance the efficiency with which the agent learns universal policies. In [18], a Curriculum-guided HER (CHER) is considered for goal proximity and diversity-based curiosity in the selection of hind-sight experiences in replay. The authors show that CHER improves challenging robotics tasks in OpenAI Gym simulated environments. In [19], the authors implemented DDPG [20] with HER and estimated the difficulty of goals by masking combinations of sub-goals using masks which were assigned specific difficulty levels. We conceptually adopted the approach described in HER, wherein sub-tasks of increasing difficulty experiences stored in the replay buffer are managed via a curriculum scheduler to enhance sample efficiency. Note that our work uses *dense reward* that could be obtained via well-established state estimation during training or inference stages. Therefore it is unnecessary to enrich the replay buffer via manipulating and re-using the training trajectories of the failed experiences.

As discussed in [16], [18], the curriculum generation of ranking difficulty of stages could be done either by a human expert or an adaptive algorithm in pre-defined or self-paced fashions. In this study, we seek to speed up training without excessive effort in collecting human demonstrations by employing sub-tasks and introducing prior knowledge. Our curriculum is generated by a few navigation difficulty metrics suggested in [11]. The proposed curriculum RL training process makes it possible for an agent to push a door opened and navigate through narrow passages as it moves toward a specific goal. We found that efficiency can be improved by building such a curriculum to narrow the distribution of tasks assigned to the agent.

III. Methodology

This section introduces the problem formulation and its reward function design, and describes how curriculum can be

combined with off-policy Distributed Distributional Deep Deterministic Policy Gradient (D4PG) [21] methods for training the neural network policy. The selection of model architecture backbone is motivated by our previous work [22], where D4PG outperforms other models such as Deep Deterministic Policy Gradient (DDPG) [23] and Recurrent Deterministic Policy Gradient (RDPG) [24] in navigation tasks. Our proposed curriculum learning tackles complex mapping (combining both navigation and arm movements) from sensor measurements to velocity commands that usually require a large number of samples to explore high-reward regions by training from scratch. We provide the agents with basic obstacle avoidance capability before initiating the training phase for navigation among movable obstacles.

The methodologies were designed to address the following research questions:

- How model convergence in diverse environments relates to the navigation strategies and difficulty metrics?
- Can curriculum learning speed up training and improve sample efficiency in comparison with standard training from scratch for NAMO problem?

A. Problem Formulation

The problem of sequential decision making can be formulated as a Partially Observed Markov Decision Process (POMDP) to be solved using deep RL. During training, the agent executes actions received from the actor network for use in storing transitions (i.e., state-action-reward pairs) within replay buffer B as experiences. The agent then updates its actor and critic networks by randomly sampling transitions from B. N-step transition pairs are sampled from B in each update cycle to compute actor and critic updates.

1) Goal Navigation Without Collisions: Goal point navigation involves two main objectives: minimizing the distance between the agent and the goal, and avoiding collisions between the agent and other static, dynamic, and movable obstacles. The optimal navigation strategy defines the best trade-off between these two competing objectives.

The reward settings (1) were expected to encourage the agent to move towards the goal and avoid collisions simultaneously to obtain high rewards. The agent should receive some rewards to keep moving and closing to the target. We established a dense reward function as follows: (1) heading toward the goal r_{hg}, (2) reaching the goal r_{rg}, (3) maintaining movement r_{mv}, (4) mobile base colliding to static (e.g., walls) and dynamic obstacles (pedestrians) r_{bc}, and (5) manipulator end-effector colliding r_{ec} to static and dynamic obstacles.

2) Navigation Among Movable Obstacles (NAMO) Problem: Existing systems based solely on collision avoidance are unable to find a free path through areas blocked by movable obstacles. Actions such as pushing movable obstacles away require additional manipulators. Some of the initial studies on navigation among movable objects (NAMO) problems [1], [2], [3], [4] used the geometry of objects and scenes to select a set of action primitives. However, autonomously controlling the mobile base and manipulators involves a complex series of decisions, involving both object interactions and obstacle avoidance.

Our agent is initially designed to push the movable obstacles (MO) out of the desired path towards the goal only by a manipulator, but not the mobile platform. However, a penalty reward for the mobile platform contacting MO may prevent the agent from moving close to any MO. On the other hand, a positive reward toward MO may encourage the agent always move toward any MO, even not blocking the path to the goal. Therefore, we did not design rewards for handling MO. The overall designs prohibit the agent from contacting other static and dynamic obstacles by the mobile platform with high penalty (-10), and with minor decreased reward (-0.5) by the end-effector.

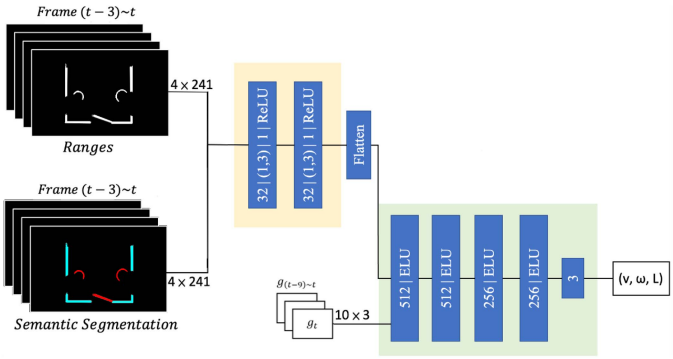


Fig. 2. Network architectures of D4PG network.

$$R(s_t, a_t, g) = r_{hg} + r_{rg} + r_{mv} + r_{bc} + r_{ec} \tag{1}$$

$$r_{hg} = \begin{cases} 0.2 & \text{if Heading toward the goal} \\ -0.2 & \text{Otherwise} \end{cases}$$

$$r_{rg} = \begin{cases} 200 & \text{if Reaching the goal} \\ 0 & \text{Otherwise} \end{cases}$$

$$r_{mv} = \begin{cases} 0.1 & \text{if Position variation} > 0.03 \\ -0.1 & \text{Otherwise} \end{cases}$$

$$r_{bc} = \begin{cases} -10 & \text{if Mobile base collision} \\ 0 & \text{Otherwise} \end{cases}$$

$$r_{ec} = \begin{cases} -0.5 & \text{if End-effector collision} \\ 0 & \text{Otherwise} \end{cases}$$

B. Observation and Action

Similar to prior work [27], [28], [29], we consider LiDAR measurement range inputs to achieve robust sim-to-real performance. As shown in Fig. 2, we converted a 3-dimensional LiDAR Point Cloud into a 2-dimensional laser scan with a total range of 241 values sampled from $-120°$ to $120°$. The resolution of the laser scan was one degree per ray.

We also provide a layer of *semantic segmentation of movable obstacles* as observation inputs. Each point in the laser scan has a segmentation label. Points that refer to movable obstacles (e.g., such as doors) are labeled 1, and otherwise 0. The input state representations include both ranges and their semantic segmentation stacked of consecutive 4 frames, fed into two 1D convolutional layers using ReLU as activation function. The extracted features of the surroundings are then concatenated with the stacked relative poses of goal points as a representation of the state. We concatenated four consecutive frames of range data and 10 frames of the position relative to the goal as a single observation space for goal navigation policy. A fully connected network serves as a decision-making mechanism.

Multi-dimensional action space involving both navigation and arm movement commands is presented as a 1×3 dimension

TABLE I
SUMMARY OF THE DIVERSE ENVIRONMENTS, DIFFICULTY METRICS, AND MODEL CONVERGENCE

		Geometry Properties					Start/Goal-point Pairs			Training			Evaluations	
	Source	Dim	Dist	Vis	Disp	CD	N	Len.	Tor.	TFS	FT-NA	Cu-MO	MO-S	MO-R
Cave	[25]	120 x 96	2.14	6.01	8.33	3.76	39	23.36	1.06	✓				
Campus	[26]	300 x 276	3.58	9.84	2.89	9.87	13	54.25	1.23	✗	✓		✓	
Indoor	[26]	144 x 120	1.83	6.15	7.27	3.35	10	43.87	1.62	✗	✓			
Garage	[26]	96 x 153	2.45	11.31	2.76	6.32	10	18.87	1.07	✗	✓			
Forest	[26]	153 x 153	3.87	14.69	1.41	13.08	10	24.17	1.01	✗	✓			
BARN-3X3	Ours	36 x 36	2.18	8.62	2.09	5.99	9	9.39	1.12	✓				
T-Maze	Ours	13 x 6	1.90	5.37	4.10	4.72	4	10.25	1.24	✗	✗	✓		
Auditorium	MP	75 x 45	2.13	7.22	4.56	5.78	5	23.09	2.21	✗	✓		✓	
Office	MP	54 x 30	1.51	5.65	8.08	3.38	2	16.90	1.84	✗	✓		✓	
Nuclear	[25]	102 x 102	2.07	6.95	3.69	5.56	1	78.38	1.57				✓	
NYCU EE6F	Ours	102 x 84	0.99	4.46	12.7	1.10	1	7.79	1.51				✓	✓

MP: models built by 3D scans of Matterport3D; TFS: training from scratch; FT-NA: finetune for goal navigation task; Cu-MO: curriculum learning for NAMO task; MO-S: evaluating NAMO task in simulated environments; MO-R: evaluating NAMO task in real environments.

matrix (ν, ω, L), used for linear velocity, angular velocity of the mobile platform and the forward displacement of the end-effector on the arm. Actions related to the twist command and displacement of the end-effector were designated as linear or angular actions. Linear actions were normalized to [0,1] to prevent the agent from moving backward, whereas angular actions were normalized to $[-1, 1]$. To enable smooth motion, angular velocity was represented as a continuous rather than discrete variable. Forward displacement actions were normalized to [0,0.5], due to the limits of the robot in terms of reach.

C. Navigation Difficulty Based on Geometric Properties

We collected a variety of distinct environmental models. Each environment required a set of metrics by which to benchmark the degree of difficulty based on [11]. These metrics derived from the occupancy grid map are used to quantify the difficulty of each environment. When we implemented the occupancy grid map used to calculate these metrics, we used a resolution of 0.6 meters (i.e., matching the width of the Husky robot). We then calculated the values for all cells in the occupancy grid map and visualized the results using a heatmap. In calculating the average value in each stage, we summed the values of all cells intersected by the path connecting the start point and goal point. The metrics are summarized as follows, which are summarized in Table I.

1) *Distance to closest obstacle (Dist): Cave* is provided for the DARPA Subterranean Challenge Virtual Competition [25] with many closed walls (*Dist:* 2.14). Similarly, *Indoor* (provided by [26]) contains long, narrow corridors connecting lobby areas with obstacles, and *Office* (created using Matterport3D) with crowded furnishings and dynamic pedestrian traffic. The average *Dist* are 1.83 and 1.51, respectively.
2) *Visibility (Vis): Forest* includes a cluster of trees with highest *Vis* (14.69), the average distance to an obstacle along each ray in a 360° scan, which is relatively open.
3) *Dispersion (Disp): Disp* captures the number of changes in potential paths out of a given location. Thus, a high dispersion value (such as at a T-junction or intersection) would indicate a more complex environment. *Campus* and *Garage* were created from large sections of the Carnegie Mellon University campus featuring low average *Disp* (2.89 and 2.76) and high average *Vis* (9.84 and 11.31), which indicate several long corridors.

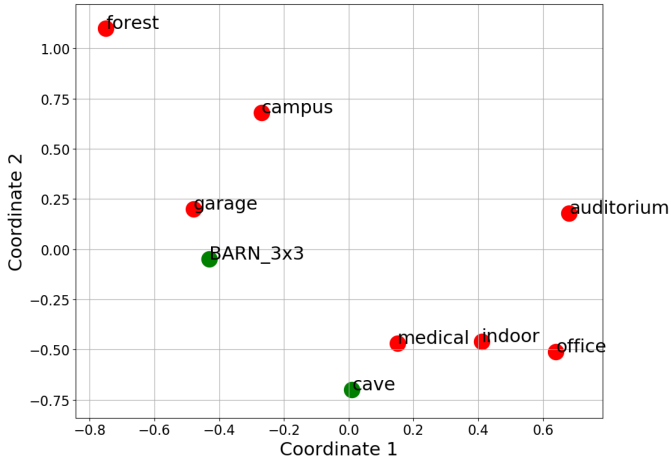


Fig. 3. The visualization of the geometric similarities of individual environments via MDS; the axes represent the two scaling coordinates reduced from the 5 metrics in [11]. We found that model convergence in diverse environments relates to the metrics. Some environments (green points) can be trained from scratch, whereas in other environments (red points) required easy-to-difficult finetune.

4) *Characteristic dimension (CD): CD* can be defined as the visibility of the axis through the cell with the lowest visibility, which captures the tightness of space. We also included *Auditorium*, which is created using Matterport3D featuring a lecture hall and complex architectural structures.
5) *Tortuosity (Tor): Tor* relates to the trajectories of the selected start/goal point pairs. We ran the TARE algorithm [12] and calculated *Tor*, which captures bends in the trajectory that could conceivably hinder navigation (*Auditorium*), compared to navigation along a relatively straight path (such as in *Garage*, *Forest*, and *Cave*).

D. Binary Difficulty Categorizations of Diverse Environments

We visualizes the individual environments in Fig. 3 via Multidimensional scaling (MDS). MDS method is used to visualize the level of similarity/distance of individual cases mapping from multiple dimensions into an abstract Cartesian two dimensional space. The two axes represent the relative scaling, labeled as "Coordinate 1 and 2" respectively without units. The environments distributed in the lower left corner (green points) were easier to navigate. We further systematically evaluate the

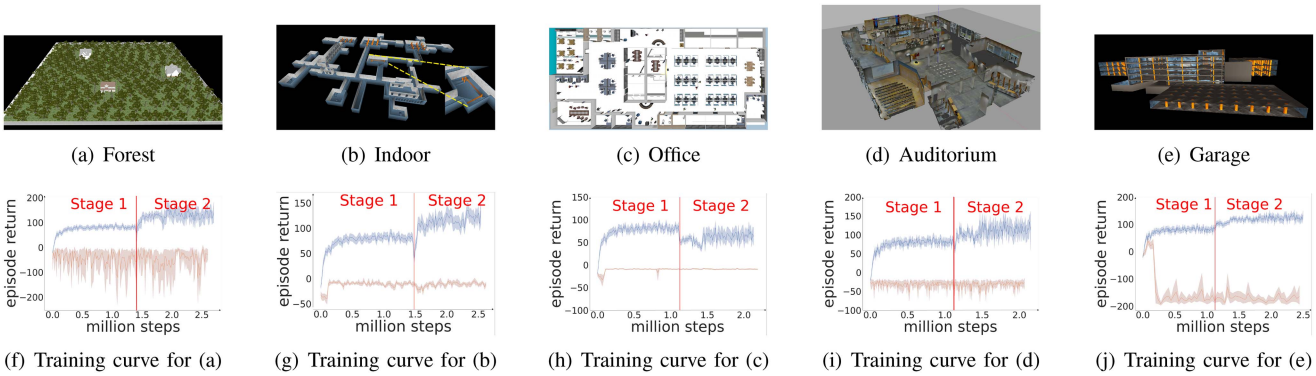


Fig. 4. Learning curves in various environments. Learning in these environments was unable to proceed beyond very low rewards by training from scratch (orange curves; average reward: -50.27). In these cases, the two-stage curriculum learning with pre-trained weights from *Cave* are effective (blue curves; average reward: 199.73).

relationship between model convergence and each metric or their combinations. We found that *Tor* and *Dist* may be the key metrics for successful training from scratch. Those analyses are shown in supplementary materials.

The combination of the five metrics serves as a predictor that guided our work to design a new, easy-to-navigate environment *BARN-3x3*, which has a higher probability of successful training from scratch. We then verified that *BARN-3x3* was able to be trained from scratch. Based on whether or not an environment can be trained from scratch, we divided the environments into two categories: E_{easy} and E_{hard}.

E. Two-Stage Learning for Goal Navigation

In this subsection, we describe the use of diverse environments for training. We discovered that agents are able to learn navigation skills without supervision; however, this requires an enormous number of samples in order for the agent to explore high-reward regions in complex environments. Note that some environments are so difficult that agents would never be able to map states to suitable actions when *training from scratch*. The deep RL navigation policy could be trained from scratch in E_{easy} (e.g., *Cave*), whereas in E_{hard} (e.g., *Forest*), it was unable to proceed beyond very low rewards. In such cases, curriculum learning would be helpful.

We implemented a two-stage, from easy to difficult, curriculum setup to overcome the inefficiency of agents exploring complex environments. We referred Stage-1 to the training of allowing the agents basic navigation ability in a relatively easy-to-navigate environment, such as *Cave*. It appears that training in *Cave* from scratch reached stable convergence under high-reward scenarios. In Stage-2, we loaded the model trained in Stage-1 and finetuned in more complex environments.

All training processes were carried out in virtual training environments using Gazebo software. We employed the OpenAI Gym framework in integrating our gym environment with Gazebo, including the analysis of agent observations, reward calculations, assigning actions, and resetting the agent. Each training environment included a set of pre-defined start/goal point pairs, and one pair was randomly selected during an episode. The robot learns the probability of transition implicitly and attempts to reach the highest accumulated reward during the training phase. A discount factor is used to determine the degree to which the agent focuses on the distant future rather than on the immediate future. In this work, the discount factor was set at 0.99 as suggested in [21]. We found that most of the environments, which failed to train from scratch, were able to be finetuned from a pre-trained weight. The Stage-2 finetuning results are shown in Table I, and parts of the training curves are demonstrated in Fig. 4.

F. Curriculum Learning for NAMO

1) Training Environments: We selected *BARN-3x3* from E_{easy} and designed *T-Maze* as E_{hard} for NAMO tasks. The training environments were designed to interact with the movable obstacles (e.g., how to use an arm to push the door or balls) and the skills of passing through the narrow gate. Our *BARN-3x3* has similar geometric properties to *Cave*, and has larger dimensions compared to the ones in [11]. *BARN-3x3* was generated using the toolkits developed in [11] with static obstacles, and we included additional movable obstacles. While the positions of static obstacles were fixed, the position of movable objects was randomly generated when each new training episode reset. *T-Maze* was designed similar to prior work [10] including several rooms in an indoor environment. The intuitive of such design is similar to Hindsight Experience Replay (HER), but the robot must travel through a number of two adjacent rooms with movable obstacles. Each room was separated by walls but connected by either a door or a narrow gate. Moreover, some random movable balls were placed, at least one in each room. Within each room of confined space, the agents inevitably encountered the movable obstacles and learned to push them away.

2) The Proposed Curriculum RL Algorithm: As shown in Algorithm 1, the proposed training algorithm includes two curriculum schedulers:

- *Two-stage curriculum scheduler (S_1):* The training process was similar to the two-stage learning described in Section III-E, except that S_1 includes a criterion C_1 to switch the training environments from E_{easy} (*BARN-3x3*) to E_{hard} (*T-Maze*). The goal point of each episode in *BARN-3x3* was randomly sampled. C_1 was to set to 70% average success rate of the last 100 records in the experience replay buffer $\mathcal{B}$.
- *Curriculum sub-steps (S_2):* We designed four curriculum steps by start and goal point pairs $p_1, \ldots, p_4$ that travelled through 2 to 4 rooms in *T-Maze*. The difficulty was sorted by the distance between the start and end points, where short distance indicated easy step. S_2 included a criterion

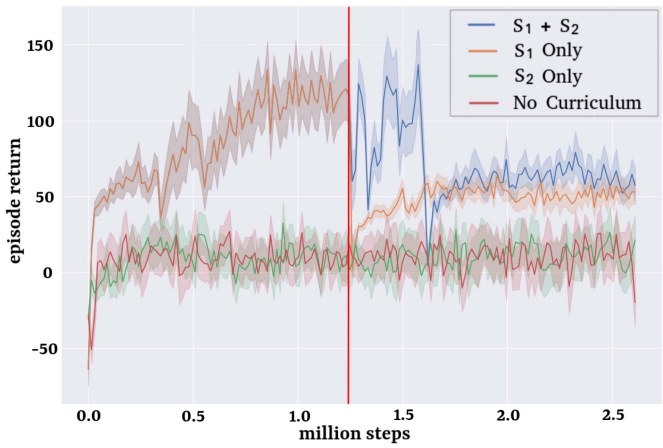


Fig. 5. The learning curves obtained using the four conditions of curriculum schedulers. The vertical red dashed line indicates the step between the two stage training from *BARN-3x3* to *T-Maze*. The training via both curriculum schedulers ($S_1 + S_2$) is more sample-efficient tackling difficult tasks such as NAMO.

Algorithm 1: The Proposed Curriculum RL for NAMO.

Input: The D4PG off-policy RL model M with actor and critic network parameters (θ, ω) and experience replay buffer $\mathcal{B}$.

Training environments E_{easy}, E_{hard} selected by two-stage curriculum scheduler S_1 and criterion C_1.
Curriculum steps by start and goal point pairs $p_1, \ldots, p_4$ selected by curriculum sub-step scheduler S_2 and criterion C_2.

Output: Learned $M(\theta, \omega)$

Initialize $M \leftarrow (\theta_0, \omega_0)$ at random, $\mathcal{B} \leftarrow \emptyset$
repeat
  Select training environment E by S_1 and C_1
  if $E = E_{easy}$ **then** $\triangleright$ E_{easy} = *BARN-3x3*
    Sample a goal g and an initial state s_0 randomly
  end if
  if $E = E_{hard}$ **then** $\triangleright E_{hard}$ = *T-Maze*
    Select curriculum step p_n by S_2 and C_2
    Sample a goal g and an initial state s_0
  end if
  for $t = 1,..., T-1$ **do**
    Sample an action a_t from θ policy of M
    Execute action a_t and observe a new state s_{t+1};
  end for
  for $t = 1,..., T-1$ **do**
    $r_t = R(s_t, a_t, g)$ $\triangleright$See Equation (1)
    Store the tuple $(s_t|g, a_t, r_t, s_{t+1}|g)$ in $\mathcal{B}$
  end for
  Train M and update parameters (θ, ω)
until Trigger the next by S_1, S_2 or terminate

C_2 by the average success rate greater than 50% based on the last 50 records in the buffer. The scheduler was triggered to switch to the next step when C_2 was satisfied.

3) Effects of Curriculum Schedulers: As shown in Fig. 5, we compared a total of 4 conditions either with or without each design to understand the effects of the proposed two curriculum designs of S_1 and S_2. For *No Curriculum* condition we trained a model to travel through four rooms in *T-maze* over a period of 3 days for about 2 million steps from scratch, without basic obstacle avoidance and goal navigation skills trained in S_1. We found it unsuccessful for NAMO with low reward close to 0 (dark red curve). Similarly the trained model of the S_2 *Only* condition was incapable of interacting with movable obstacles (green curve). We found that the designs with S_1 were required to complete the NAMO tasks, but we found that $S_1 + S_2$ (blue curve) was more sample efficient than $S_1 Only$ (Orange curve). We also observe a drop of $S_1 + S_2$ at 1.6 million step while the robot attempted to travel through *T-Maze* with a turn. It appeared that even with S_2 travelling through multiple movable obstacles with turns was challenging. In the evaluation section, we used the model trained with $S_1 + S_2$ for 2.5 million steps.

IV. Evaluations

Experiments were designed to investigate the navigation performances and the strategies learned by our approach compared to the state-of-the-art model-based method.

A. Metrics and Methods

Robot Navigation was evaluated according to the rate of success and average collision counts per trial. The success rate showed that the robot successfully reached the goal. We set a timeout period to report the robot got trapped and failed to reach the goal. Collisions represented the robot colliding on static obstacles such as walls, and we calculated the average collision times of a route.

1) Baseline TARE Local Planner (TARE-L): We selected the local planner of the state-of-the-art TARE [12] as our baseline. Note that *TARE-L* was not designed to interact with movable obstacles, which will always avoid contact. While the movable obstacle was presented normally, TARE tended to wander back and forth in front of the door until timeout. We intentionally forced *TARE-L* to ignore all detected movable obstacles in NAMO tasks.

2) The Proposed Approach (Ours): We selected the deep RL policy trained with *BARN-3x3* followed by *T-maze* for the evaluations in the goal navigation and NAMO tasks. Note that such a model did not train in any of the environments in evaluations. We tested our deep RL policy (*Ours*) and *TARE-L* in several unseen, large-scale virtual environments.

B. NAMO Evaluations in Simulated Environments (MO-S)

1) Environments: We set up a few movable obstacles (i.e., doors) in *Nuclear*. Apart from the origin virtual environment, we added a few movable obstacles (hinged doors) to a few constrained passages. *NYCU EE6F* was a scanned 3D environment with narrow indoor corridors and classrooms, created using Matterport3D. The robot must have to pass through a door, which was added as a simulated hinged door into Gazebo. Together with another 2 indoor environments *Auditorium* and *Office*, those 4 environments were used to evaluate NAMO.

2) Results and Discussions: As shown in Table II, *TARE-L* was able to pass through movable obstacles by ignoring them, but the success rates was lower than our proposed approach due to such policy caused some trapped, collisions or even the vehicle overturned. In *Office* there was no success passing through, caused by the narrow gate *TARE-L* could not pass even by ignoring movable obstacles. Our purposed deep RL policy achieved good performance, and the agent could not only push away movable obstacles but also navigating in large-scale environment with some minor collisions.

TABLE II
WE COMPARED THE PROPOSED POLICY (OURS) AND THE BASELINE STATE-OF-THE-ART APPROACH (*TARE-L*) WITH GOAL NAVIGATION AND NAMO TASKS IN SIMULATED AND THE REAL ENVIRONMENTS

Task	Environment	Success Rate		Avg. Collision	
		Ours	TARE-L	Ours	TARE-L
MO-S	Auditorium	**1.0**	0.4	0.6	0.4
	Office	**1.0**	0	1.5	2.0
	Nuclear	**1.0**	0.8	0.8	0.2
	NYCU EE6F	0.8	0.7	0.4	0.8
MO-R	NYCU EE6F	0.9	-	0.2	-

(a) Matterport 3D Environments

(b) NAMO - Push Balls Away

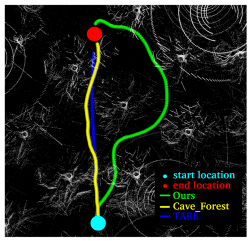
(c) NAMO - Push Doors Open

Fig. 6. NAMO Evaluations in a real office environment and its scanned simulated Matterport 3D environment.

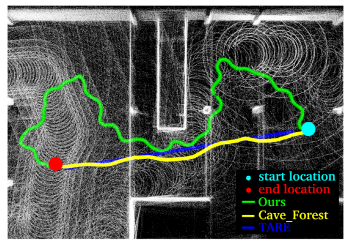
C. NAMO Evaluations in Real Environments (MO-R)

1) Environments: As shown in Fig. 6, the real environment of *NYCU EE6F* was used for running the our proposed policy on real robot. We selected a few starting and goal points, and added some movable obstacle (yoga balls) and a cardboard door. To prevent the robot from damaging we did not carry out *TARE-L*. The real experiments used the same routes as in the Matterport 3D experiments. We also carried out a semantic segmentation algorithm to detect movable obstacles (blue balls and doors), and a SLAM algorithm (Lego-LOAM [30]). Both performed well in this real environment.

2) Results and Discussions: We received a success rate of 0.9 that the real robot system was able to correctly navigate to the goal. The agent tended to first make contact by the arm and end effector to avoid collisions with the mobile platform when encountering movable obstacles. Our deep RL policy performed consistently on virtual and real robots. Note that the settings of *TARE-L* in the NAMO experiments encouraged the robot to make contacts to movable obstacles by the mobile base instead of the manipulator. Although the robot moving speed was not fast, in real-world pilot study we have observed that the robot was stuck by movable objects. Such settings also caused confusion about the timing of when an experimenter should carry out emergency stops. Note that the use of *TARE-L* ignoring movable obstacles is designed as a baseline method in our experiment, and we do not consider it a viable method in real-world NAMO tasks for safety considerations. To prevent the robot from damaging itself we did not carry out *TARE-L* in MO-R.



(a) Trajectories in *Forest*



(b) Trajectories in *Garage*

Fig. 7. In a constrained environment our agent carried out a similar trajectory as *TARE-L*. However, our method took a higher tortuosity trajectory, possibly learnt the wall following strategy of avoiding obstacle.

D. General Discussions

1) Wall Following Strategy Was Learned in Stage-1: We found that the agent (trained in *BARN-3x3* and *T-maze*) learned to avoid collisions by following walls in Stage-1, and such behavior was carried to Stage-2 training. As shown in Fig. 7, for a relatively open space in (a) *Forest* and (b) *Garage*, our approach tended to follow a longer path close to walls instead of a shorter straight path. This was less efficient compared with *TARE-L*, which carried out shortest path straight toward the goal. We further performed a model trained in *Cave* and *Forest* (*Cave_Fores*), which corrected the tendency of wall following.

2) Learning Without Forgetting: One drawback of finetuning a deep network in literature was that the finetuned models tend to be optimized to the new task and perform poorly in the original task. In this work we did not observe a performance degradation of a finetuned model in originally trained environments, such as *Cave* and *BARN-3x3*. Instead, we found a good generalization of collision avoidance capabilities in diverse **unseen** environments. This might be caused by that the more difficult to navigate environments still required the basic abilities trained in Stage-1, instead of a conflict toward a different objective.

V. CONCLUSION

In this letter, we proposed a curriculum learning approach, consisting of a deep RL agent for a mobile robot with a manipulator to carry out navigation among movable obstacles (NAMO). We first associated the navigation difficulties with model convergences among a set of diverse environments. The agent was trained from easy to difficult through continuous control via a sample-efficient curriculum learning settings. For a more complex NAMO task, we found that both designs of two-stage learning and curriculum sub-steps were required. By starting training from basic navigation skills such as wall following, obstacle avoidance, and goal navigation, the agent learned toward more complex skills about pushing away movable obstacles.

We carried out comprehensive evaluations for goal navigation and NAMO tasks by comparing the deep RL policy with a state-of-the-art local planner *TARE-L* as a baseline. We verified that our method performed competitive results in a goal navigation task, and better than the baseline method when dealing with movable obstacles. In real-robot experiments, the qualitative results of our method showed that our method could deploy in real environment scenarios. For future work we suggest that force-torque or touch sensors installed on the manipulator end-effector or around the mobile base will be useful for conducting real-world experiments interacting with movable objects.

Finally, we suggested that the learned policies could operate robustly in large-scale, **unseen** environments in the virtual and real environments.

V. Acknowledgment

The authors would like to thank Chi-An Lee and Zi-Yan Liu for providing assistance in early stage developments.

References

[1] M. Stilman, J.-U. Schamburek, J. Kuffner, and T. Asfour, "Manipulation planning among movable obstacles," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2007, pp. 3327–3332.

[2] M. Stilman and J. Kuffner, "Planning among movable obstacles with artificial constraints," *Int. J. Robot. Res.*, vol. 27, no. 11/12, pp. 1295–1307, 2008.

[3] J. V. D. Berg, M. Stilman, J. Kuffner, M. Lin, and D. Manocha, "Path planning among movable obstacles: A probabilistically complete approach," in *Algorithmic Foundation of Robotics VIII*. Berlin, Germany: Springer, 2009, pp. 599–614.

[4] M. Levihn, J. Scholz, and M. Stilman, "Hierarchical decision theoretic planning for navigation among movable obstacles," in *Algorithmic Foundations of Robotics X*. Berlin, Germany: Springer, 2013, pp. 19–35.

[5] M. Pfeiffer et al., "Reinforced imitation: Sample efficient deep reinforcement learning for map-less navigation by leveraging prior demonstrations," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 4423–4430, Oct. 2018.

[6] F. Xia, C. Li, R. Martín-Martín, O. Litany, A. Toshev, and S. Savarese, "ReLMoGen: Integrating motion generation in reinforcement learning for mobile manipulation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 4583–4590.

[7] C. Li, F. Xia, R. Martin-Martin, and S. Savarese, "HRL4IN: Hierarchical reinforcement learning for interactive navigation with mobile manipulators," in *Proc. Conf. Robot Learn.*, 2020, pp. 603–616.

[8] P. Mirowski et al., "Learning to navigate in cities without a map," in *Proc. 32nd Int. Conf. Neural Inf. Process. Syst.*, 2018, pp. 2424–2435.

[9] Y. Song, H. Lin, E. Kaufmann, D. Peter, and D. Scaramuzza, "Autonomous overtaking in GRAN turismo sport using curriculum reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, Xi'an, China, 2021, pp. 9403-9409.

[10] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher-student curriculum learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3732–3740, Sep. 2020 .

[11] D. Perille, A. Truong, X. Xiao, and P. Stone, "Benchmarking metric ground navigation," in *Proc. IEEE Int. Symp. Saf., Secur., Rescue Robot.* 2020, pp. 116–121.

[12] C. Cao, H. Zhu, H. Choset, and J. Zhang, "TARE: A hierarchical framework for efficiently exploring complex 3D environments," in *Proc. Robot. Sci. Syst. Conf.*, 2021.

[13] J. L. Elman, "Learning and development in neural networks: The importance of starting small," *Cognition*, vol. 48, no. 1, pp. 71–99, 1993.

[14] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, 2009, pp. 41–48.

[15] A. Safa, T. Verbelen, I. Ocket, A. Bourdoux, F. Catthoor, and G. G. Gielen, "Fail-safe human detection for drones using a multi-modal curriculum learning approach," *IEEE Robot. Automat. Lett.*, vol. 7, no. 1, pp. 303–310, Jan. 2022.

[16] P. Soviany, R. T. Ionescu, P. Rota, and N. Sebe, "Curriculum learning: A survey," *Int. J. Comput. Vis.*, vol. 130, no. 6, pp. 1526–1565, 2022.

[17] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum learning for reinforcement learning domains: A framework and survey," *J. Mach. Learn. Res.*, vol. 21, no. 1, pp. 7382–7431, 2020.

[18] M. Fang, T. Zhou, Y. Du, L. Han, and Z. Zhang, "Curriculum-guided hindsight experience replay," in *Proc. 33rd Int. Conf. Neural Inf. Process. Syst.*, 2019, pp. 12623–12634.

[19] M. Eppe, S. Magg, and S. Wermter, "Curriculum goal masking for continuous deep reinforcement learning," *Joint IEEE 9th Int. Conf. Develop. Learn. Epigenetic Robot.*, 2018, pp. 183–188.

[20] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2016. [Online]. Available: https://dblp.org/db/conf/iclr/iclr2016.html

[21] G. Barth-Maron et al., "Distributional policy gradients," in *Proc. Int. Conf. Learn. Representations*, 2016. [Online]. Available: https://openreview.net/forum?id=SyZipzbCb

[22] C. Lu et al., "A heterogeneous unmanned ground vehicle and blimp robot team for search and rescue using data-driven autonomy and communication-aware navigation," *Field Robot.*, vol. 2, pp. 557–594, 2022.

[23] T. P. Lillicrap et al., "Continuous control with deep reinforcement learning," in *Proc. Int. Conf. Learn. Representations*, 2018. [Online]. Available: http://dblp.uni-trier.de/db/conf/iclr/iclr2016.html#LillicrapHPHETS15

[24] N. Heess, J. J. Hunt, T. P. Lillicrap, and D. Silver, "Memory-based control with recurrent neural networks," in *Proc. NIPS Deep Reinforcement Learn. Workshop*, 2015. [Online]. Available: https://rll.berkeley.edu/deeprlworkshop/

[25] V. Orekhov and T. Chung, "The DARPA subterranean challenge: A synopsis of the circuits stage," *Field Robot.*, vol. 2, pp. 735–747, 2022.

[26] C. Cao et al., "Autonomous exploration development environment and the planning algorithms," in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 8921–8928.

[27] M. Pfeiffer, M. Schaeuble, J. Nieto, R. Siegwart, and C. Cadena, "From perception to decision: A data-driven approach to end-to-end motion planning for autonomous ground robots," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 1527–1533.

[28] M. Pfeiffer et al., "Reinforced imitation: Sample efficient deep reinforcement learning for mapless navigation by leveraging prior demonstrations," *IEEE Robot. Automat. Lett.*, vol. 3, no. 4, pp. 4423–4430, Oct. 2018.

[29] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2017, pp. 31–36.

[30] T. Shan and B. Englot, "LeGO-LOAM: Lightweight and ground-optimized lidar odometry and mapping on variable terrain," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2018, pp. 4758–4765.