Curiosity-driven Exploration for Mapless Navigation with Deep Reinforcement Learning

Oleksii Zhelo[1], Jingwei Zhang[1], Lei Tai[2], Ming Liu[2], Wolfram Burgard[1]

***Abstract*— This paper investigates exploration strategies of Deep Reinforcement Learning (DRL) methods to learn navigation policies for mobile robots. In particular, we augment the normal external reward for training DRL algorithms with intrinsic reward signals measured by curiosity. We test our approach in a mapless navigation setting, where the autonomous agent is required to navigate without the occupancy map of the environment, to targets whose relative locations can be easily acquired through low-cost solutions (e.g., visible light localization, Wi-Fi signal localization). We validate that the intrinsic motivation is crucial for improving DRL performance in tasks with challenging exploration requirements. Our experimental results show that our proposed method is able to more effectively learn navigation policies, and has better generalization capabilities in previously unseen environments. A video of our experimental results can be found at https://goo.gl/pWbpcF.**

I. Introduction

Deep Reinforcement Learning (DRL), deploying deep neural networks as function approximators for high-dimensional RL tasks, achieves state of the art performance in various fields of research [1].

DRL algorithms have been studied under the context of learning navigation policies for mobile robots. Traditional navigation solutions in robotics generally require a system of procedures, such as Simultaneous Localization and Mapping (SLAM) [2], localization and path planning in a given map, etc. With the powerful representation learning capabilities of deep networks, DRL methods bring about the possibility of learning control policies directly from raw sensory inputs, bypassing all the intermediate steps.

Eliminating the requirement for localization, mapping, or path planning procedures, several DRL works have been presented that learn successful navigation policies directly from raw sensor inputs: target-driven navigation [3], successor feature RL for transferring navigation policies [4], and using auxiliary tasks to boost DRL training [5]. Many follow-up works have also been proposed, such as embedding SLAM-like structures into DRL networks [6], or utilizing DRL for multi-robot collision avoidance [7].

In this paper, we focus specifically on mapless navigation, where the agent is expected to navigate to a designated goal location without the knowledge of the map of its current environment. We assume that the relative pose of the target is easily acquirable for the agent via cheap localization solutions such as visible light localization [8] or Wi-Fi signal localization [9]. Tai *et al.* [10] successfully applied DRL for mapless navigation, taking as input sparse laser range readings, as well as the velocity of the robot and the relative target position. Trained with asynchronous DRL, the policy network outputs continuous control commands for a nonholonomic mobile robot, and is directly deployable in real-world indoor environments.

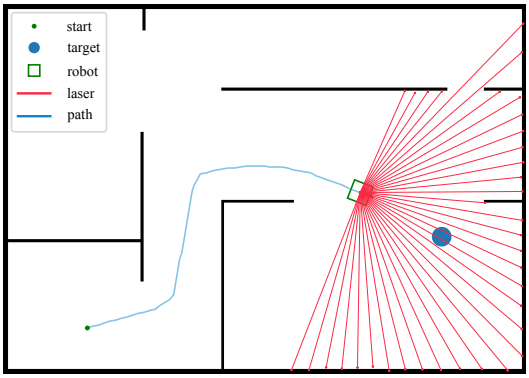


Fig. 1: We study the problem of mapless navigation, where without a given the map of the environment, the autonomous agent is required to navigate to a target whose relative location can be easily acquired by cheap localization solutions. Structures like long corridors and dead corners make it challenging for DRL agents to learn optimal navigation policies. Trained with reward signals that are augmented by intrinsic motivation, our proposed agent efficiently tackles this task, and shows that its learned policy generalizes well to previously unseen environments.

Most of the aforementioned methods, however, either rely on random exploration strategies like ϵ-greedy, or on state-independent exploration by maximizing the entropy of the policy. As the previous works present experiments in environments which do not impose considerable challenges for the exploration of DRL algorithms, we suspect that these exploration approaches might not be sufficient for learning efficient navigation policies in more complex environments.

Proposed by Pathak *et al.* [11], the agent's ability to predict the consequences of its own actions can be used to measure the novelty of the states, or the intrinsic curiosity. The *Intrinsic Curiosity Module* (ICM) is employed to acquire this signal during the process of reinforcement learning, and its prediction error can serve as an intrinsic reward. As illustrated in their experiments, the intrinsic reward, learned completely in a self-supervised manner, can motivate the agent to better explore the current environment, and make

[1]Department of Computer Science, Albert Ludwig University of Freiburg. oleksii.zhelo@saturn.uni-freiburg.de, {zhang, burgard}@informatik.uni-freiburg.de

[2]Department of Electronic and Computer Engineering, The Hong Kong University of Science and Technology. {ltai, eelium}@ust.hk

use of the structures of the environment for more efficient planning strategies.

Similar intrinsic signals have also been used as auxiliary tasks to encourage exploration by Khan *et al.* [12], where a planning agent is developed using differentiable memory and self-supervised state, reward and action prediction.

In this paper, we investigate exploration mechanisms for aiding DRL agents to learn successful navigation policies in challenging environments that might contain structures like long corridors and dead corners. We conduct a series of experiments in simulated environments, and show the sample-efficiency, stability, and generalization ability of our proposed agent.

II. Methods

A. Background

We formulate the problem of autonomous navigation as a *Markov Decision Process* (MDP), where at each time step t, the agent receives an observation of its current state $\mathbf{s}_t$, takes an action $\mathbf{a}_t$, receives a reward R_t, and transits to the next state $\mathbf{s}_{t+1}$ following the transition dynamics $p(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ of the environment. For the mapless navigation task that we consider, the state $\mathbf{s}_t$ consists of laser range readings $\mathbf{s}_t^l$ and the relative pose of the goal $\mathbf{s}_t^g$. The task of the agent is to reach the goal position g without colliding with obstacles.

B. Extrinsic Reward for Mapless Navigation

We define the extrinsic reward R^e (the traditional reinforcement signal received from the environment) at timestep t as follows (λ^p and λ^ω denote the scaling factors):

$$R_t^e = r_t + \lambda^p r_t^p + \lambda^\omega r_t^\omega, \tag{1}$$

where r_t imposes the main task ($\mathbf{p}_t$ represents the pose of the agent at t):

$$r_t = \begin{cases} r_{\text{reach}}, & \text{if reaches goal,} \\ r_{\text{collision}}, & \text{if collides,} \\ \lambda^g \left(\left\| \mathbf{p}_{t-1}^{x,y} - g \right\|_2 - \left\| \mathbf{p}_t^{x,y} - g \right\|_2 \right), & \text{otherwise,} \end{cases}$$

and the position- and orientation-based penalties r_t^p and r_t^ω are defined as:

$$r_t^p = \begin{cases} r_{\text{position}}, & \text{if } \left\| \mathbf{p}_{t-1}^{x,y} - \mathbf{p}_t^{x,y} \right\|_2 = 0, \\ 0, & \text{otherwise,} \end{cases}$$

$$r_t^\omega = \left\| \text{atan2}(\mathbf{p}_t^y - g^y, \mathbf{p}_t^x - g^x) - \mathbf{p}_t^\omega \right\|_1 .$$

C. Intrinsic Reward for Curiosity-driven Exploration

On top of the normal extrinsic reward R^e, we use intrinsic motivation measured by curiosity, rewarding novel states to encourage exploration and consequently improve the learning and generalization performance of DRL agents in environments that require more guided exploration.

Following the formulation of [11], we measure the intrinsic reward R^i via an *Intrinsic Curiosity Module* (ICM) (depicted in Fig. 2). It contains several feature extraction layers ϕ, a *forward model* (parameterized by ψ^f), and an *inverse model* (parameterized by ψ^i).

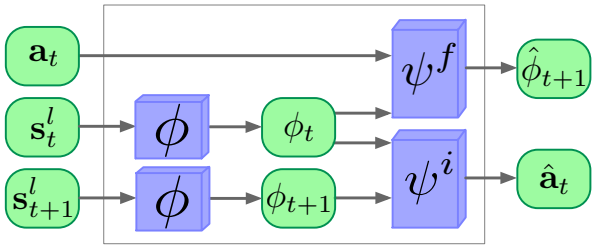


Fig. 2: ICM architecture. $\mathbf{s}_t^l$ and $\mathbf{s}_{t+1}^l$ are first passed through the feature extraction layers ϕ, and encoded into ϕ_t and ϕ_{t+1}. Then ϕ_t and ϕ_{t+1} are input together into the *inverse model* ψ^i, to infer the action $\hat{\mathbf{a}}_t$. At the same time, $\mathbf{a}_t$ and ϕ_t are together used to predict $\hat{\phi}_{t+1}$, through the *forward model* ψ^f. The prediction error between $\hat{\phi}_{t+1}$ and ϕ_{t+1} is used as the intrinsic reward R^i.

First, $\mathbf{s}_t^l$ and $\mathbf{s}_{t+1}^l$ are passed through ϕ, encoded into their corresponding features ϕ_t and ϕ_{t+1}. Then, ψ^i predicts $\hat{\mathbf{a}}_t$ from ϕ_t and ϕ_{t+1} (trained through cross-entropy loss for discrete actions with the ground truth $\mathbf{a}_t$), while ψ^f predicts $\hat{\phi}_{t+1}$ from ϕ_t and $\mathbf{a}_t$ (trained through mean squared error loss with the ground truth ϕ_{t+1}). ϕ, ψ^f and ψ^i are learned and updated together with the actor-critic parameters θ^π and θ^v (Sec. II-D) during the training process.

The overall optimization objective is

$$\min_{\theta_\phi, \theta_{\psi_f}, \theta_{\psi_i}} ((1-\lambda^f)(-\sum_{j=1}^{3} \mathbf{a}_j \log \hat{\mathbf{a}}_j) + \lambda^f \frac{1}{2} \left\| \hat{\phi}_{t+1} - \phi_{t+1} \right\|_2^2),$$

where the first part corresponds to the inverse model training objective, and the second part to the forward model prediction error. These losses are weighted by the λ^f parameter, which helps control the trade-off between the characteristics both models are conferring to the feature extraction layers: 1) the inverse model seeks to extract information useful to predict the action taking the agent between consecutive states, and 2) the forward model strives to encourage extracting more predictable feature embeddings. To avoid learning constant features for all states (which would be trivially predictable), the magnitude of forward model's contribution to the learning process needs to be balanced by the weight λ^f. Both of these influences, however, help to exclude environmental factors which are unaffected by and do not affect the agent, so it would not stay constantly curious of irrelevant or inherently unpredictable details of the environment.

The prediction error between $\hat{\phi}_{t+1}$ and ϕ_{t+1} through ψ^f is then used as the intrinsic reward R^i:

$$R^i = \frac{1}{2} \left\| \hat{\phi}_{t+1} - \phi_{t+1} \right\|_2^2 . \tag{2}$$

Following this intrinsic reward, the agent is encouraged to visit novel states in the environment, which is crucial in guiding the agent to get out of local minimums or premature convergence of sub-optimal policies. This exploration strategy allows the agent to make better use of the learned environment dynamics to accomplish the task at hand.

D. Asynchronous Deep Reinforcement Learning

For training the navigation policies, we follow the asynchronous advantage actor-critic (A3C) algorithm [1], with the weighted sum of the external reward and the intrinsic reward as the supervision signal:

$$R = R^e + \lambda^i R^i, \tag{3}$$

where $\lambda^i > 0$ is a scaling coefficient.

A3C updates both the parameters of a policy network θ^π and a value network θ^v, to maximize the cumulative expected reward. The value estimate is bootstrapped from n-step returns with a discount factor γ (where K represents the maximum number of steps for a single rollout):

$$G_t = \gamma^{K-t} V(\mathbf{s}_K; \theta^v) + \sum_{\tau=t}^{K-1} \gamma^{\tau-t} R_\tau. \tag{4}$$

Each learning thread accumulates gradients for every experience contained in a K step rollout, where the gradients are calculated according to the following equations:

$$d\theta^\pi = \nabla_{\theta^\pi} \log \pi(\mathbf{a}_t|\mathbf{s}_t; \theta^\pi)(G_t - V(\mathbf{s}_t; \theta^v)) \tag{5}$$

$$+ \beta \nabla_{\theta^\pi} H(\pi(\mathbf{a}_t|\mathbf{s}_t; \theta^\pi)), \tag{6}$$

$$d\theta^v = \partial (G_t - V(\mathbf{s}_t; \theta^v))^2 / \partial \theta^v, \tag{7}$$

where H is the entropy of the policy, and β represents the coefficient of the entropy regularization. This discourages premature convergence to suboptimal deterministic policies, and is the default exploration strategy that has been used with A3C.

III. Experiments

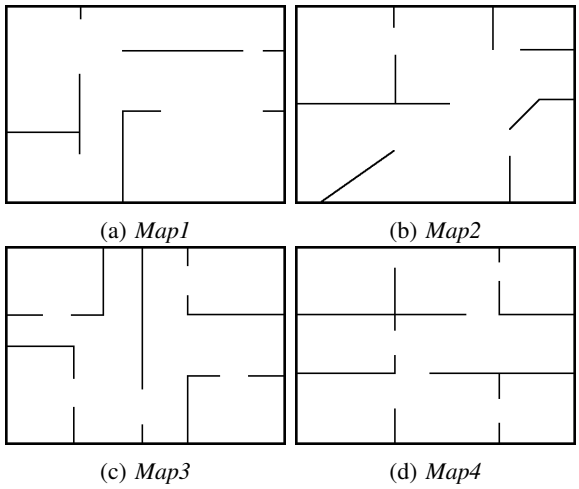
A. Experimental setup

We conduct our experiments in a simulated environment, where a robot is equipped with a laser range sensor, and is navigating in the $2D$ environments shown in Fig. 3. At the beginning of each episode, the starting pose of the agent $\mathbf{p}_0$ and the target position g are randomly chosen such that a collision-free path is guaranteed to exist between them. An episode is terminated after the agent either reaches the goal, collides with an obstacle, or after a maximum of 7000 steps during training and 400 for testing.

The state $\mathbf{s}_t$ consists of 72-dimensional laser range readings $\mathbf{s}_t^l$ (with a maximum range of $7\,\mathrm{m}$), and a 3-dimensional relative goal position $\mathbf{s}_t^g$ (relative distance, and $\sin$ and $\cos$ of the relative orientation in the agent's local coordinate frame). At each timestep, the agent can select one of three discrete actions: {go straight for $0.06\,\mathrm{m}$, turn left $8°$, turn right $8°$}.

The hyper-parameters regarding the reward function (II-B,II-C) are the following: $\gamma = 0.99$, $\lambda^p = 1$, $\lambda^\omega = \frac{1}{200\pi}$, $\lambda^g = 0.15$, $r_{\text{reach}} = 1$, $r_{\text{collision}} = -5$, $r_{\text{position}} = -0.05$, and the scale for the intrinsic reward λ^i is set to 1.

The actor-critic network uses two convolutional layers of 8 filters with stride 2, and kernel sizes of 5 and 3 respectively, each followed by ELU nonlinearities. These are followed by two fully connected layers with 64 and 16 units, also ELU nonlinearities, to transform the laser sensor readings $\mathbf{s}_t^l$ into a 16-dimensional embedding. This representation is concatenated with the relative goal position $\mathbf{s}_t^g$, and fed into a single layer of 16 LSTM cells. The output is then concatenated with $\mathbf{s}_t^g$ again, and used to produce the discrete action probabilities by a linear layer followed by a softmax, and the value function by a linear layer.



(a) *Map1* (b) *Map2* (c) *Map3* (d) *Map4*

Fig. 3: Different floor plans we considered in our experiments. *Map1,3,4* have similar wall structures, while imposing gradually more exploration challenges for DRL agents to effectively learn navigation policies. *Map2*, while holding a level of challenge for exploration similar to that of *Map1*, has some structural variations such as the angled walls. We only train agents on *Map1*. The performance of the trained agents is later evaluated on *Map1*, and tested on *Map2,3,4* for their generalization capabilities. The maps are all of size $5.33\,\mathrm{m} \times 3.76\,\mathrm{m}$.

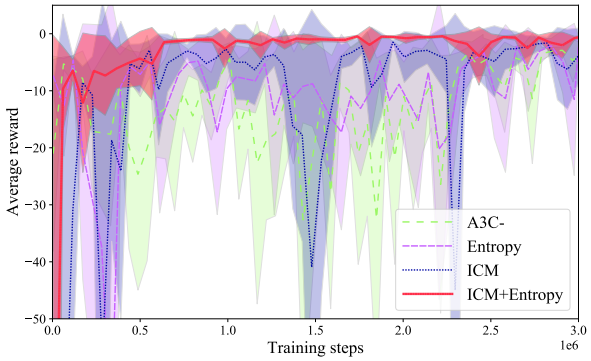
The ICM inverse model ψ^i consists of three fully connected layers with 128, 64, and 16 units, each followed by an ELU nonlinearity, producing features ϕ_t and ϕ_{t+1} from $\mathbf{s}_t^l$ and $\mathbf{s}_{t+1}^l$. Those features are then concatenated and put through a fully connected layer with 32 units and an ELU nonlinearity, the output of which predicts $\hat{\mathbf{a}}_t$ by a linear layer followed by a softmax. The ICM forward model ψ^f accepts the true features ϕ_t and a one-hot representation of the action $\mathbf{a}_t$, and feeds them into two linear layers with 64 and 32 units respectively, followed by ELU and a linear layer that predicts $\hat{\phi}_{t+1}$.

We train A3C with the Adam optimizer with statistics shared across 22 learner threads, with a learning rate of $1\mathrm{e}{-4}$. The rollout step K is set to 50. ICM is learned jointly with shared Adam with the same learning rate and $\lambda^f = 0.2$. Each training of 3 million iterations takes approximately 15 hours using only CPU computation.

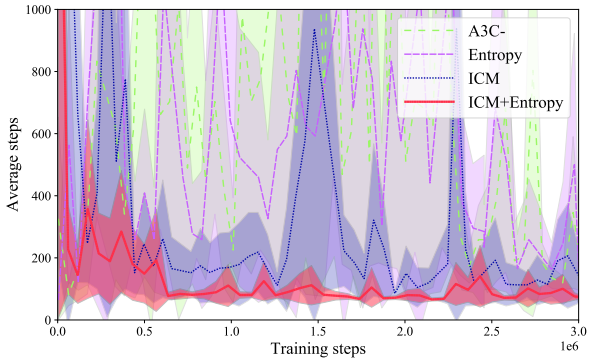
We only train agents on *Map1* (Fig. 3a), varying exploration strategies (Sec. III-B). *Map2,3,4* are used to test the generalization ability of the trained policies (Sec. III-C).

B. Training and Evaluation on Map1

We experiment with 4 variations of exploration strategies by switching on and off the entropy loss and the intrinsic



(a) Average reward.



(b) Average steps.

Fig. 4: Comparison of (Fig. 4a) average reward and (Fig. 4b) steps for different exploration strategies that we considered (switching on/off the entropy loss or the intrinsic reward). The mean and confidence interval for each configuration is calculated over the statistics of 3 independent runs.

reward: 1) *A3C-*: A3C with $\beta = 0.$; 2) *Entropy*: A3C with $\beta = 0.01$; 3) *ICM*: A3C with $\beta = 0.$, with ICM; 4) *ICM+Entropy*: A3C with $\beta = 0.01$, with ICM.

The average reward and steps obtained in the evaluation during training are shown in Fig. 4. We can clearly observe that without any explicit exploration mechanisms, *A3C-* exhibits unstable behavior during learning. *ICM* alone is much more effective in guiding the policy learning than *Entropy* alone, since the exploration imposed by the latter is state-independent, while with *ICM* the agent is able to actively acquire new useful knowledge from the environment, and learn efficient navigation strategies for the environmental structures that it has encountered. *ICM+Entropy* stably and efficiently achieves the best performance.

We run evaluations on the same set of 300 random episodes on *Map1* after training, and report the resulting statistics in Table I. In addition, we conducted another set of experiments where we remove the LSTM layers in the actor-critic network. We report their evaluation results in Table I as well. Although without LSTM none of the configurations manage to converge to a good policy (so we omit their training curves in Fig. 4), we can still observe a clear trend of performance improvements for the policies trained both with and without LSTM, brought about by ICM exploration.

TABLE I: Evaluation on *Map1*.

	Exploration Strategy	Success Ratio (%)	Steps (mean±std)
Map1 (w/o LSTM)	Entropy	34.3	313.787±135.513
	ICM	65.7	233.147±143.260
	ICM+Entropy	**73.3**	**162.733±150.291**
Map1 (w/ LSTM)	A3C-	88.3	173.063±123.277
	Entropy	96.7	102.220±90.230
	ICM	98.7	91.230±62.511
	ICM+Entropy	**100**	**75.160±52.075**

TABLE II: Generalization tests on new maps.

	Exploration Strategy	Success Ratio (%)	Steps (mean±std)
Map2	A3C-	66.0	199.343±145.924
	Entropy	85.3	115.567±107.474
	ICM	**87.7**	**101.160±102.789**
	ICM+Entropy	82.7	109.690±107.043
Map3	A3C-	51.3	197.200±141.292
	Entropy	69.7	175.667±139.811
	ICM	63.7	152.627±135.982
	ICM+Entropy	**72.7**	**139.423±120.144**
Map4	A3C-	44.3	273.353±148.527
	Entropy	34.3	240.197±164.565
	ICM	45.0	236.847±169.169
	ICM+Entropy	**55.0**	**209.500±161.767**

C. Generalization Tests on Map2,3,4

To test the generalization capabilities of the learned policies, we deploy the networks trained on *Map1*, collect statistics on a fixed set of 300 random episodes on *Map2,3,4*, and report the results in Table II.

As discussed earlier in Fig. 3, *Map2* is relatively simple but contains unfamiliar structures; *Map3* and *Map4* are more challenging, but the structures inside these floorplans are more similar to the training environment of *Map1*. With the exploratory behaviors brought by both *ICM* and *Entropy*, the agent learns well how to exploit structures similar to those it has encountered during training. Thus from Table II we can observe that, although the full model *ICM+Entropy* is slightly worse than *ICM* exploration alone on *Map2*, it achieves better performance in the more challenging environments of *Map3* and *Map4*.

D. Discussion

As can be seen from our results, the ICM agents, even if they are not always more successful in reaching the targets, tend to find shorter paths to them. We believe that this can be viewed as evidence that exploratory behaviors due to curiosity help escape local minima of the policy function (in form of solutions that choose inefficient ways of reaching the target). We speculate that the reason for this improvement lies in the ICM agents seeing a bigger part of the environment, as the intrinsic motivation attracts them towards novel and currently less predictable states, while random exploration does little to avoid re-exploring the same areas again and again, getting stuck in structures which are difficult to escape by pure chance.

IV. Conclusions

We investigate exploration strategies for learning mapless navigation policies with DRL, augmenting the reward signals with intrinsic motivation. Our proposed method achieves better sample efficiency and convergence properties during training than all the baseline methods, as well as better generalization capabilities during tests on previously unseen maps. Our future work includes studying how to better adapt the weighting of the different components in the reward function, as well as real-world robotics experiments.

References

[1] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016, pp. 1928–1937.

[2] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT Press, 2005.

[3] 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 *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3357–3364.

[4] J. Zhang, J. T. Springenberg, J. Boedecker, and W. Burgard, "Deep reinforcement learning with successor features for navigation across similar environments," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 2371–2378.

[5] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu, *et al.*, "Learning to navigate in complex environments," *arXiv preprint arXiv:1611.03673*, 2016.

[6] J. Zhang, L. Tai, J. Boedecker, W. Burgard, and M. Liu, "Neural SLAM," *arXiv preprint arXiv:1706.09520*, 2017.

[7] P. Long, T. Fan, X. Liao, W. Liu, H. Zhang, and J. Pan, "Towards optimally decentralized multi-robot collision avoidance via deep reinforcement learning," *arXiv preprint arXiv:1709.10082*, 2017.

[8] Q. Liang and M. Liu, "Plugo: a VLC systematic perspective of large-scale indoor localization," *arXiv preprint arXiv:1709.06926*, 2017.

[9] Y. Sun, M. Liu, and M. Q.-H. Meng, "WiFi signal strength-based robot indoor localization," in *Information and Automation (ICIA), 2014 IEEE International Conference on*. IEEE, 2014, pp. 250–256.

[10] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*. IEEE, 2017, pp. 31–36.

[11] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *International Conference on Machine Learning (ICML)*, vol. 2017, 2017.

[12] A. Khan, V. Kumar, and A. Ribeiro, "Learning Sample-Efficient Target Reaching for Mobile Robots," *ArXiv e-prints*, Mar. 2018.