Towards generic object property estimation using Unsupervised Renforcement Learning

Maxime Chareyre, Pierre Fournier, Julien Moras, Youcef Mezouar, Jean-Marc Bourinet

Towards generic object property estimation using Unsupervised Renforcement Learning

Maxime Chareyre[1,2], Pierre Fournier[1], Julien Moras[1], Youcef Mezouar[2] and Jean-Marc Bourinet[2]

***Abstract*— Scene understanding for an autonomous agent is essential to perform tasks in an environment often made up of a multitude of objects of complex appearances and dynamics. In this paper, we propose a generic method to train an agent to physically interact with an object to extract several of its physical properties. In a real case, the agent may face unidentified objects with no prior knowledge. For this purpose, we used an algorithm from Unsupervised Reinforcement Learning (URL) to learn such an interaction strategy. The observations acquired by the agent along its trajectory are used to predict several properties with a model learned in a supervised manner. We applied our method in a simulated environment with a mobile robot only capable of interacting through pushing in order to extract dynamic parameters such as mass, shape, capacity to roll and slide. The results show that URL is able to provide diversity in trajectories to extract properties even in noisy conditions.**

I. INTRODUCTION

The autonomy of a robot relies on its ability to perceive, understand and interact with an environment often made up of a multitude of objects of complex appearances and dynamics. It can be necessary to build a knowledge base of these objects to perform tasks. Such knowledge can be obtained through passive or active perception of the objects, but for an embodied agent, these approaches do not leverage its ability to interact physically with the environment. Such physical interactions are all the more advisable that they make it easier to infer properties otherwise hidden, like mass or dynamic properties ("does this object roll when pushed?").

Intuitively, an embodied agent tasked with identifying properties of objects by interacting with them should process in two phases: interact informatively enough with objects, then extract properties from the interaction. This dissociation is mandatory as optimizing the property predictor and the interaction policy together quickly becomes unachievable and requires prior knowledge of the objects' properties. In this work, we propose a new generic object property discovery framework relying on Unsupervised Reinforcement Learning (URL) to learn to interact with unknown objects in such a way that the interactions are rich enough to train a predictive model of objects properties in a supervised manner.

In URL, the environment is a reward-free Markov decision process defined by $M = (\mathcal{S}, \mathcal{A}, \mathcal{P})$ where $\mathcal{S}$ is the state set, $\mathcal{A}$ the action set and $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1]$ the transition distribution function. Following a policy π in M leads to a trajectory τ. The question we seek to answer is: what kind of policy should the agent follow in order to make sure that the resulting trajectory can be used to train a model that predicts some properties of objects unknown at the time of interaction?

In this context, our contributions are as follows:

- we propose to learn this policy with an URL algorithm that maximizes *the entropy of a subset of the state set* S in order to produce the most informative trajectories of interaction with the objects;
- we validate this choice experimentally on a simulated mobile platform tasked with predicting multiple properties of some objects solely by pushing them optimally.

II. RELATED WORK

A. Physical properties estimation

The question of formalizing and learning knowledge about objects properties has been approached from multiple angles.

Supervised learning methods can identify properties such as shape [1], mass [2] or type of material [3] from object appearance of labeled static images. Some of these properties can be visually identified (e.g. shape) while others can be considered hidden (e.g. mass). A promising way of learning properties consists in exploiting the object movements resulting from an interaction. J. Wu et al. infer the physical properties in a self-supervised manner using Markov chain Monte Carlo (MCMC) [4] or Neural Network (NN) based methods [5] by processing videos of an object sliding on a slope. In [6] M. Lohmann et al. interact with some object in a complex realistic environment by applying force of variable magnitude. This way, they train a convolutional model in a self-supervised way to predict properties such as mobility and mass by using the force-induced movements.

Contrary to the previous works, some recent studies focus on using a robot to autonomously interact with objects. Pinto et al. [7] use a robot to build better object visual representation by performing different kind of interaction with its arm. In [8] authors learn a predictive model of the dynamics of an object by observing its motions resulting from poking interactions. Xu and al. [9] created a framework to learn a physical representation of objects upon which NNs are used to extract properties such as friction or mass. A robotic arm performs predefined actions to push or collide with several objects and thus builds a representation model based on pre- and post-interaction observations provided by a camera positioned above the environment.

The methods discussed above focus more on how to process the observations than how to find the policy used

*This work was not supported by any organization

1 are with DTIS, ONERA, 91123 PALAISEAU, France. firstname.lastname@onera.fr

2 are with SIGMA Clermont, 63178 AUBIERE, France. firstname.lastname@sigma-clermont.fr

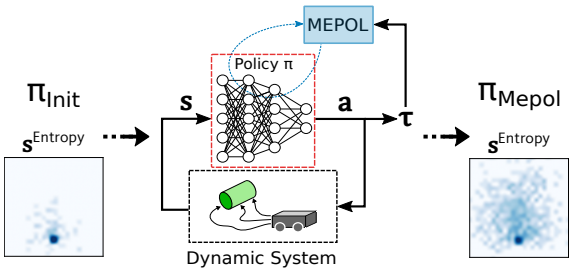

Fig. 1. Learning process of the interaction policy using MEPOL in charge of maximizing the entropy on a well-chosen space. It illustrates the impact of a random policy called π_{init} and of π_{Mepol} on the distribution of a $\mathcal{S}^{Entropy}$ 2-dimensional space.

to acquire them. However, it may be relevant to move around and intelligently interact to get the most informative observations. Approaches based on active perception [10] [11] have shown that a mobile robot is able to identify, without a camera, the shape of an object using only eight laser sensors set up around it. Denil and al. [12] show that interaction policies learn by RL lead to more informative representations. They perform a set of tasks that require agents to estimate properties such as mass and cohesion of objects based on manipulations and observation of the consequences.

B. Interaction policy through Unsupervised RL

Objects properties estimation in the previous works often relies on confined environment, in which agents have the capacity to observe the whole scene allowing to precisely locate objects and to apply predefined actions. In our case, we assume that we do not have any knowledge about objects when learning the policy. Our setup relies on the recently proposed paradigm of URL [13]. Generally, these algorithms are used to pre-train a policy to explore an environment without the use of any extrinsic reward function. The resulting policy is fine-tuned on a specific task using a traditional RL method, and the process is much faster than learning from scratch. Among URL methods, we focus on algorithms known as data based according to [13]. This kind of policies aims to increase the diversity of the agent's observations by maximizing a discrete [14] or a continuous [15] entropy space which might be high dimensional [16]. These diversities generated by the policy will facilitate the extraction of properties.

III. PROPOSED MODEL

A. Finding an exploration policy that promotes the diversity of interactions

Let us define a property p of an object o as a continuous or categorical value to be inferred from the interactions between a given agent and this object. o is simulated by drawing uniformly on multiple parameter spaces such as shape, size or weight. We consider an agent in presence of an object o whose final objective is to interact to estimate l properties of a set $Properties = p_1, p_2, ..., p_l$. In the formalism of URL, the agent in presence of one object o follows a policy

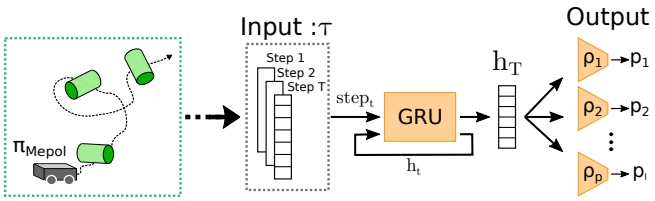

Fig. 2. Multi-prediction properties using τ generated with the π_{Mepol}.

π, potentially interacts with o and observes its environment and the motions o generated by its actions $a \in \mathcal{A}$. Let us denote $s \in \mathcal{S}$ a vector containing the set of observations provided by the agent's sensors.

In this context, we propose to find a policy that maximizes the entropy over a space $\mathcal{S}^{Entropy}$ which should optimally defined to represent properties belonging to specific classes (dynamic or visual). Let $s^{Entropy} \in \mathcal{S}^{Entropy}$ be a subset of the state variables s, and $d_T^{\pi}(s^{Entropy})$ the distribution of $s^{Entropy}$ induced by following π over one trajectory τ of finite horizon T. We define π_θ as a fully connected network of parameters θ. The optimization problem to solve writes:

$$\max_{\theta} \quad \mathcal{H}(d_T^{\pi_\theta}(s^{Entropy})) \tag{1}$$

where $\mathcal{H}$ is a measure of the entropy on a given distribution.

We use the Maximum Entropy POLicy (MEPOL) optimization algorithm [15] to learn a policy π that solves objective Eq. (1). This algorithm is suitable for dealing with continuous action and state spaces. At each epoch, MEPOL builds a batch of trajectories τ and updates the set of parameters θ using a k-nearest neighbors approach [17] to solve an estimation of Eq. (1). At the end of the process illustrated in Fig. 1, we call π_{Mepol} the policy learned by MEPOL after N epoch.

B. Extracting properties from trajectories

We use π_{Mepol} to generate K trajectories τ_k of T steps with interaction with object $o_k, k \in [\![1, K]\!]$. At each time step t, the agent observes the environment with its sensors and builds a feature vector $step_t$. This vector contains information about the agent's state, its actions and its observations of the object. The observations over the K trajectories form the dataset D_{train} composed of sample-pairs $(\tau_k = [step_0^k, step_1^k, ..., step_T^k], Properties_k = \{p_1^k, p_2^k, ..., p_l^k\})$. To summarize observations are used to build 3 different spaces which are: $\mathbf{s}$ used by the policy π to decide on the action a, $\mathbf{s^{Entropy}}$ space on which entropy is maximized, $\mathbf{step}$ observations used to predict the properties.

We predict properties from the whole trajectory. For this purpose, as illustrated in Fig. 2, we use a GRU cell. The last hidden state h_T of the cell becomes the input of a model ρ_l used to predict the property p_l. Because properties can be related together, we share the same internal representation provide by h_T to each predictor. The whole set of models ρ_l is jointly trained in a supervised manner using the dataset D_{train} with the following loss function:

$$Loss = \sum_{i}^{l} L_{p_i}$$
$$\text{where} L_{p_i} = \begin{cases} \text{MSELoss}(\hat{p_i}, p_i) & \text{if } p_i \text{ continuous} \\ \text{CrossEntLoss}(\hat{p_i}, p_i) & \text{if } p_i \text{ discrete} \end{cases} \quad (2)$$

IV. PREDICTING PROPERTIES THROUGH PUSHING INTERACTIONS

The objective is to validate the method proposed in the previous section in the specific case of a mobile robot based only on pushing interactions. The PyBullet python package [18] is used with the Bullet physics engine to build our environment and evaluate our agent. All the model implementations are made in PyTorch.

A. Experimental setup

Simulated environment. The agent is a WifiBot[1] mobile robot placed on a planar environment. The agent is positioned as described in Fig. 3, facing one object of random shape $p_{shape} = \{cube, sphere, cylinder\}$. Mass (in kg) is set by a parameter $p_{mass} \sim \mathcal{U}([0.1; 8])$ and the size of magnitude 10 cm is multiplied by a scale factor $p_{size} \sim \mathcal{U}([1; 5])$. Cylinders are all positioned on one of their generatrix to create an object that can either slide or roll making it hard to distinguish them from spheres or cubes. The initial distance between the agent and the object is set to 50cm. The friction parameters as well as the inertial parameters of the object are established by PyBullet considering homogeneous material.

The agent's actions are parameterized by a forward and a rotational speed $a = [v^{forward}, v^{rotation}]$. The agent is only allowed to interact with the object by pushing it. We assume first that the agent is able to accurately estimate the position of the object's center using its sensors, and then relax this assumption by evaluating the robustness of the framework against noise on this estimation. The observation space reported in Fig. 3 includes:

- r: the distance between the agent and the object centers,
- θ^{agent}: the angle between the forward direction of the agent and the object center,
- dx^{object} and dy^{object}: distances covered by the object between two steps along x and y expressed in R_t,
- vx^{object} and vy^{object}: object velocities between two time steps expressed in R_t.

Interaction policy. We define s and $s^{Entropy}$ as follows:

$$s = [r, \theta^{agent}, dx^{object}, dy^{object}]$$
$$s^{Entropy} = [dx^{object}, dy^{object}]$$

Appendix V-A provides the hyperparameters used in MEPOL as well as the policy network architecture. At each time step, the policy outputs two mean values of normal distributions from which are drawn a forward velocity $v^{forward} \in [-1, 1]$ and a rotational velocity $v^{rotation} \in [-0.5, 0.5]$.

Properties predictions. We validate our framework by predicting three categorical and one continuous properties of objects: shape p_{shape}, whether the object can roll p_{roll} or slide p_{slide}, and mass p_{mass}. We define :

$$step_t^k = [r_t, \theta_t^{agent}, vx_t^{object}, vy_t^{object}, v_t^{forward}, v_t^{rotation}]$$

π_{Mepol} is used to generate 2000, 500 and 500 trajectory samples to respectively build the sets D_{train}, D_{val} and D_{test}.

We choose a GRU cell with latent state h_t of dimension 40. This latent state is then decoded in a dense network with hidden layers of shape (32, 16, dim(p_l)). We use ReLU between each hidden layer. The parameters of the predictive model are updated minimizing the error from Eq. (2). ADAM is used as optimizer with a learning rate $lr = 10^{-4}$. The model trains for 3000 epochs with mini-batch of size 128.

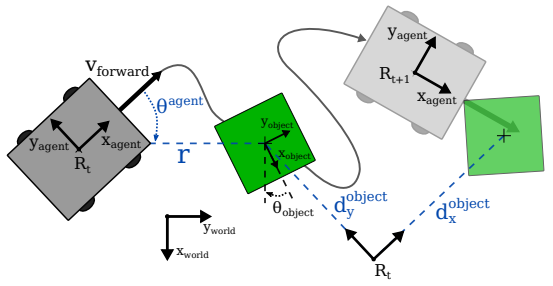

Fig. 3. Parametric setting of the robot in its environment. vx^{object} and vy^{object} are derived from dx^{object} and dy^{object} considering the period of a simulation step. At initialization, θ^{object} and θ^{agent} are set to 0°.

B. Results

The model introduced in Section III is trained with the experimental setup just presented. The performance is assessed from the scores on a single set D_{test} of 5 models trained on independent sets D_{train}, with optimal parameters calibrated from a single set D_{val}. The score results presented in this section are presented in terms of their means and variances.

Maximizing entropy over a well-chosen space gives better interaction trajectories and a better prediction model.

Fig. 4 plots the performances of properties predictors trained on datasets obtained from interaction policies of increasing entropy over $\mathcal{S}^{Entropy}$ with trajectories length $T = 400$ steps. Before training for entropy maximization, the policy gives 40% of correct predictions. After training, the policy allows the predictors to reach more than 98% accuracy on classification tasks and an RMSE around 1kg for the mass estimation estimate.

The trained policy leads to efficient interactions for the further estimation of the object properties.

The upper plots of Fig. 5 show how the $\mathcal{S}^{Entropy}$ space is explored through $T = 400$ steps trajectories based on policies trained with 3 or 15 epochs of MEPOL. We also give the confusion matrices in the lower plots of Fig. 5. The way the agent finds to maximize this space is to push the object on its different sides to move it further and further in different directions. It is easier to cover the space when the object is rolling. The agent therefore looks for an angle of approach

[1]https://www.wifibot.com/

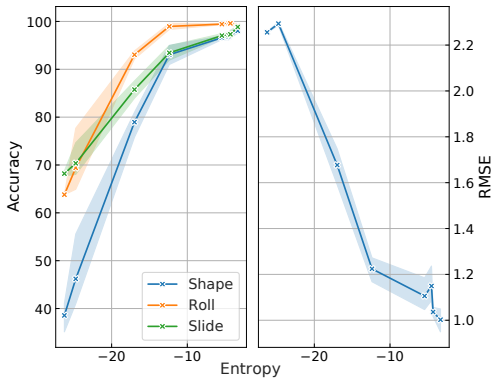

Fig. 4. Impact of policies leading to varying entropy measurements on the performance of the property predictor. Entropy is measured on the space $\mathcal{S}^{Entropy}$ for trajectories of length $T = 400$ steps. Left: prediction accuracy for discrete properties, right: averaged RMSE for mass property.

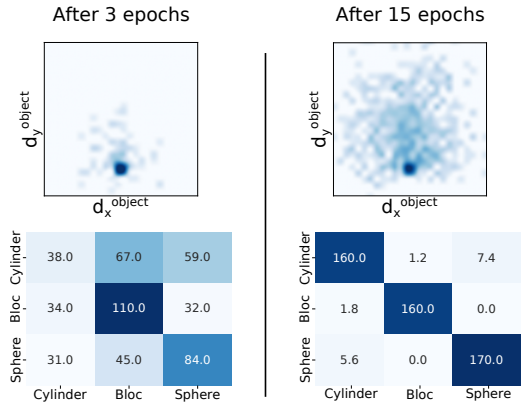

Fig. 5. Visit of the $\mathcal{S}^{Entropy}$ space after 3 or 15 epochs of MEPOL measured on 5 trajectories (top). Confusion matrices of the shape predictions associated to these policies with $T = 400$ steps. Calculations performed on D_{test} of 500 objects averaged on 5 trained models.

which makes the object roll, which seems an ideal policy for predicting the shape of an object. This is qualitatively confirmed by the results obtained after 15 epochs.

Increasing the number of interactions improves the accuracy of predictions.

We evaluate in Fig. 6 the impact of the number of interactions between the agent and the object by carrying out experiments with various maximum trajectory length (on average, $\{3, 6, 12, 25, 38, 50\}$ interactions are observed respectively for trajectories of $\{50, 100, 200, 400, 600, 800\}$ steps). For a trajectory length of 50 steps, the shape classification already reaches 90% accuracy, indicating an efficient exploration policy. In particular, the trajectory length has a significant impact on the mass estimation, which is the hardest task.

The accuracy of predictions remains satisfactory with noisy observations.

Our framework heavily relies on the agent ability to estimate the object's positions. In a realistic scenario, such an estimation is bound to be noisy. We thus evaluate the robustness of the framework against noise by adding a Gaussian noise of standard deviation of 10cm spread over

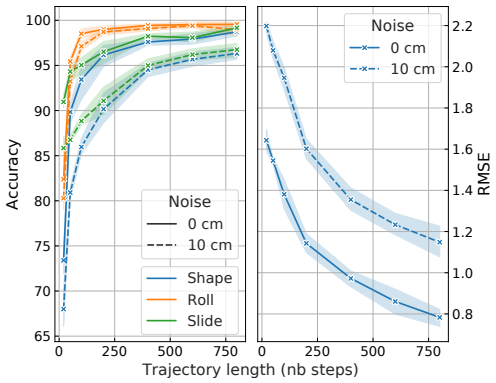

Fig. 6. Property prediction accuracy versus trajectory length (solid lines). Effect of a Gaussian noise applied on all observations of *step* (dotted lines).

all the observations of the vector $step_t$ except for the agent actions. The use of noise also prevents the agent from exploiting too much precise measurement and discerning objects from slight oscillations of the objects upon contact. The prediction accuracy in the presence of noisy measurements is also plotted in Fig. 6. Results show that noise has a moderate impact on the prediction accuracy and that increasing the trajectory length compensates for noisy sensors in order to reach the same performance as a noise-free trained model. The mass estimation is more challenging, as the agent predicts the mass of the object from its post-contact velocity. However, the velocity of the object and the distance traveled are also affected by the friction coefficients and the object/ground contact surface. This means that the larger the contact surface, the more difficult it is to predict the mass. This is confirmed by our results: the RMSE for a cube is on average 2 times higher than that of a sphere.

V. CONCLUSIONS

A two-step process was presented for : 1) learn to interact with an object of unknown physics and 2) use the observations of these interactions to extract several physical properties. This method was validated by means of simulations on a mobile robot without cameras whose only interaction capability is pushing. The results indicated that entropy maximization on a well-chosen space leads to an efficient policy for mass and shape identification on spherical, cylindrical and cubic objects even with noisy observations. The proposed method is intended to be generic and can be applied to different mobile or non-mobile robots with other interactions capabilities, such as a manipulator arm.

The current method shows very promising results when focusing on objects one at a time. Robotic scenarios where the agent can alternate between multiple objects raise interesting challenges: how should the agent pick an object to explore through interaction? How should the agent process the resulting disjointed sequences of interaction associated with each selected object? The last question will be addressed in a follow-up work focused on property estimation from *sets* of interaction sequences with *multiple* objects, instead of full trajectories with one object.

APPENDIX

A. Details on MEPOL parameters

We train our agent using an implementation of MEPOL provided by the authors[2] that we plug on our custom gym environment. Our MEPOL hyperparameters are similar to those used in the GridWorld experiment reported in their paper [15]. These parameters either relevant or adjusted are reported in Table I.

TABLE I

MEPOL PARAMETERS.

	Value
Number of epochs	100
Horizon (T)	200
Learning rate	10^{-5}
Policy hidden layer sizes	(300,300)
Policy hidden layer act. function	ReLU
Policy output act. function	Tanh
Noises on s	0cm

ACKNOWLEDGMENT

REFERENCES

[1] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, 2017, pp. 2463–2471.

[2] T. Standley, O. Sener, D. Chen, and S. Savarese, "image2mass: Estimating the mass of an object from its image," in *1st Annual Conference on Robot Learning, CoRL 2017, Mountain View, California, USA, November 13-15, 2017, Proceedings*, 2017, pp. 324–333.

[3] S. Bell, P. Upchurch, N. Snavely, and K. Bala, "Material recognition in the wild with the materials in context database," in *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, 2015, pp. 3479–3487.

[4] J. Wu, I. Yildirim, J. J. Lim, B. Freeman, and J. B. Tenenbaum, "Galileo: Perceiving physical object properties by integrating a physics engine with deep learning," in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, 2015, pp. 127–135.

[5] J. Wu, J. J. Lim, H. Zhang, J. B. Tenenbaum, and W. T. Freeman, "Physics 101: Learning physical object properties from unlabeled videos," in *Proceedings of the British Machine Vision Conference 2016, BMVC 2016, York, UK, September 19-22, 2016*, 2016.

[6] M. Lohmann, J. Salvador, A. Kembhavi, and R. Mottaghi, "Learning about objects by learning to interact with them," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[7] L. Pinto, D. Gandhi, Y. Han, Y. Park, and A. Gupta, "The curious robot: Learning visual representations via physical interactions," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*, 2016, pp. 3–18.

[8] P. Agrawal, A. Nair, P. Abbeel, J. Malik, and S. Levine, "Learning to poke by poking: Experiential learning of intuitive physics," in *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, 2016.

[9] Z. Xu, J. Wu, A. Zeng, J. B. Tenenbaum, and S. Song, "Densephysnet: Learning dense physical object representations via multi-step dynamic interactions," in *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*, 2019.

[2]https://github.com/muttimirco/mepol

[10] M. Gouko, Y. Kobayashi, and C. H. Kim, "Online exploratory behavior acquisition model based on reinforcement learning," *Appl. Intell.*, pp. 75–86, 2015.

[11] M. Gouko, C. H. Kim, and Y. Kobayashi, "Active perception model extracting object features from unlabeled data," in *18th International Conference on Advanced Robotics, ICAR 2017, Hong Kong, China, July 10-12, 2017*, 2017, pp. 518–523.

[12] M. Denil, P. Agrawal, T. D. Kulkarni, T. Erez, P. W. Battaglia, and N. de Freitas, "Learning to perform physics experiments via deep reinforcement learning," in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

[13] M. Laskin, D. Yarats, H. Liu, K. Lee, A. Zhan, K. Lu, C. Cang, L. Pinto, and P. Abbeel, "URLB: unsupervised reinforcement learning benchmark," in *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021.

[14] E. Hazan, S. M. Kakade, K. Singh, and A. V. Soest, "Provably efficient maximum entropy exploration," in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, 2019, pp. 2681–2691.

[15] M. Mutti, L. Pratissoli, and M. Restelli, "A policy gradient method for task-agnostic exploration," *CoRR*, 2020.

[16] H. Liu and P. Abbeel, "Behavior from the void: Unsupervised active pre-training," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, 2021, pp. 18 459–18 473.

[17] J. Ajgl and M. Šimandl, "Differential entropy estimation by particles," *IFAC Proceedings Volumes*, pp. 11 991–11 996, 2011.

[18] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2022.