Learning Sampling-based Exploration Planning

Chao Ni, Yuliang Zhong
ETH Zürich
Rämistrasse 101. 8092 Zürich. Switzerland
`chaoni@ethz.ch, yuzhong@student.ethz.ch`

Abstract: To efficiently explore the unknown environment is essential to robot autonomy. Sampling-based methods are usually used to infer the Next-Best-View (NBV) and achieve optimal performance when the samples go infinity. However, it is computationally expensive and would be a bottleneck in real-world applications. We believe that the NBV samples implicitly encode the exploration gain and constraint of the local environment, and are favorable to certain regions given a local map. In this paper, we propose to learn the sampling distribution from the NBV samples and bias the exploration towards the frequently visited areas. Therefore, one can predict the next action by drawing samples from the distribution and achieves faster exploration. We use a Conditional Variational Auto-encoder (CVAE) to learn the distribution. We evaluate our approach on a 2D simulator and show that our planner explores faster than the uniform baseline planner and requires less computational resources to fully explore the map. We also show the generalization capabilities of our approach on a different map.

Keywords: Exploration Planning, CVAE, Informative Path Planning

1 Introduction

Recent years have seen the wide applications of mobile robots. One of the interesting tasks is to explore unseen environments. The robot is equipped with perceptive sensors and needs to make the best decision based on the sensor input. At the same time, due to the limited battery, the robot also needs to explore efficiently in order to fulfill the entire unknown environment.

Sampling-based methods are often used in an exploration planning problem, where various kinds of gains can be evaluated for each sample and the best is chosen as the NBV sample. These samples are drawn uniformly to cover the entire state space and theoretically provide asymptotic optimality. While benefiting from a thorough exploration with a large number of samples, the blind search can take up lots of computational power, which is a bottleneck in real-world exploration. Besides, the uniform sampler assumes a complete state space while the actual robots may have implicit constraints (collision check, torque limitation, etc). As a consequence, the uniform planner ignores the knowledge from the sensor inputs and revisited the experience.

In this paper, we propose to learn from the samples generated by the uniform planner. We believe those samples encode the implicit constraints and imply the gain distribution. By biasing samples towards the learned distribution, the exploration can be accelerated. CVAE [1] is used to learn the underlying distribution because of its capability to reduce the complexity of the high dimensional data and approximate the manifold in a low dimensional space. Conditioned on the robot's local environment, the samples can be drawn from the learned latent space and a prediction of the next action can be inferred.

A thorough evaluation of our approach is provided on a 2D simulator, depicted in Fig. 1 with a rich analysis on the comparison with the NBV uniform sampling planner. We present the Pareto diagram to illustrate the computational issue faced by real robots and compare the exploration speed with the NBV planner. Additionally, we evaluate the generalization capability on an unseen map. This work explores the frontier of exploration planning and makes the following contributions:

- We propose to learn the optimal sampling distribution from the NBV uniform planner and sample from the learned distribution for faster exploration.
- We evaluate our learning sampling-based exploration approach on a 2D simulator and show that it explores faster than the NBV planner with same amout of samples and requires less computational resource to fully explore the map.
- We extend the application of CVAE to the exploration planning tasks. As far as the authors know, this is the first time CVAE is used in an efficient exploration task.

The remainder of this paper is proceeded as follows: Section 2 summarizes related work on the exploration planning and the CVAE. Section 3 explains the background of the problem. Section 4 gives detailed learning sampling-based exploration planning methodology. Section 5 demonstrates the performance of our learned planner and the comparison with the NBV uniform planner and discusses the gain prediction variant. Section 6 concludes our work.

2 Related Work

2.1 Exploration Planning

Exploration planning has attracted a lot of attention and there exists plenty of different approaches, among which sample-based methods are an important family. LaValle et al. [2] proposes a Rapidly-Exploring Random Tree (RRT) to extend the branches biasing toward unexplored space, but this method is limited in its heavy computation cost. Karaman and Frazzoli [3] introduce a provable efficient and optimal online solution RRT* that can be computed in real-time. Schmid et al. [4] reuses the non-executed nodes and keeps a single objective function to achieve global convergence. Bircher et al. [5] utilizes a receding horizon fashion to reuses the cell gain, which also makes exploration capable of online running.

Curiosity-driven exploration has also been well explored since its theory foundation [6][7] and triggered numerous research in both model-based and model-free fields. Houthooft et al. [8] uses a curiosity-driven strategy to bias the exploration to the explored region by maximizing the information gain.

Apart from the unknown volume space, exploration on a 3D surface with an Unmanned Aerial Vehicle (UAV) also catches attention because of its continuous structure and applications in industrial inspection. Zhu et al. [9] has assumed a spatial correlation between adjacent surface meshes and update the status with manifold Gaussian processes.

2.2 Learning for Path Planning

Machine learning has been widely used in path planning tasks since the rapid development of deep learning frameworks and computational hardware. Due to the non-deterministic property of path planning tasks, learn the probabilistic motion of the robot attracts lots of attention. Ichter et al. [10] learns the distribution of the optimal samples collected from a motion planning problem, where the robot is given an initial position and the goal position. In our work, we are aimed at samples that explore the unknown environment instead of achieving a specific goal.

2.3 CVAE

Representation learning has been experiencing an increase in use in robotics applications. [11] A high dimensional data is processed and encoded by self-supervised or semi-supervised learning methods, and the hidden learned state can be accessed by other tasks such as Reinforcement Learning (RL). As is the nature of a robotics system, having a low-dimensional state is of a crucial role and it has to discard irrelevant information while keeping the most meaningful knowledge. There exist various approaches to infer the hidden state representation. In particular, reconstructing the observation is used to learn the decoder that reproduces the observation solely based on the learned hidden state, which contains essential knowledge. One often imposes constraints on the latent space such as denoising criterion [12], dimension requirement, etc.

Representation learning also benefits from Variational Auto-encoder(VAE) [13], a family of generative models. The VAE uses a self-supervised fashion to learn the hidden state by approximating it with a model while minimizing the observation reconstruction error. VAE has been widely used in robotics applications, Hoeller et al. [14] learns the hidden state representation of the sensor inputs of a mobile robot, and the representation is used in a navigation task.

As a variant of VAE, CVAE adds the component on which the hidden distribution is conditioned, as is shown in Fig. 4 where the local map is the conditioning. CVAE has been proven efficient in a path planning task [10], where the robot is asked to find the best path given the global environment map. The samples drawn from the learned hidden space are biased towards the expert path. While motivated by the success of CVAE in its application in long-horizon motion planning, we believe it is more suitable for a single-step exploration, where the action is conditioned on the local map.

3 Problem Formulation

We simplify exploration in a 2D simulator, as Fig.1 and Fig.2 show. For each step, a sampling-based planner of the robot uniformly samples robot states in free areas of the local map, calculates informative gains of each state, then asks the robot to move to the best one and refresh explored map during movement. The best one is called NBV sample X, or best sample X. The problem we solve in this paper is how to generate X in a computationally efficient way.

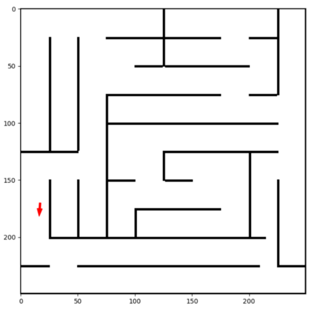

Figure 1: An autonomous robot (red arrow) is exploring a 2D maze-like map. The robot state can be expressed in a 3 dimensional row vector $X_0 = [x_0, y_0, yaw]$, which represents its global position and orientation.

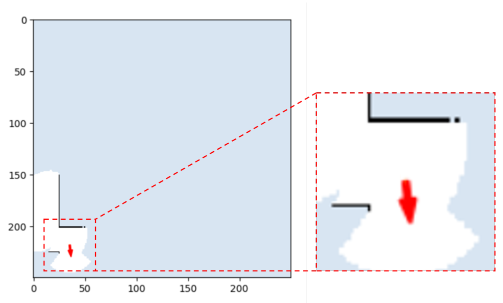

Figure 2: When exploring, the robot can look around via a 90 FOV camera, and draw an explored map. The white, black, blue pixels mean free, wall, and unobserved area. The partial explored map around robot is called local map.

4 Methods

4.1 Overview

We use CVAE framework to learn the latent distribution of the optimal samples generated by the NBV uniform planner. The training samples are collected from simulated explorations on different maps. In the planning, one random variable from the normal distribution is drawn and fed with the local map to the learned decoder, the decoder projects them to the inference action.

4.2 Data Collection

The training data is collected from the NBV uniform planner. As is shown in Fig. 3, each piece of data consists of the best action **X** and the conditioning **Y**. The conditioning local map is an occupancy map, which is one-hot encoded and flattened to a vector. We use 160k data for training and 40k for validation.

4.3 CVAE Architecture

Preliminary We denote x as the continuous random variable, y as the conditioning variable, and z as the latent variable. A general process of CVAE is as follows: the latent variable z is sampled from $Q(z|x, y)$, and the data x is generated from the decoder $P(x|z, y)$.

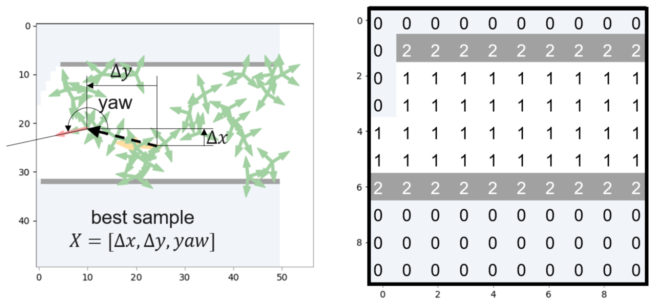

Figure 3: A piece of training sample: best action **X** and local map **Y**. The best action is a relative distance to the current robot position. The local map is an occupancy map which has element of {0,1,2} representing {unknown, free, wall} respectively.

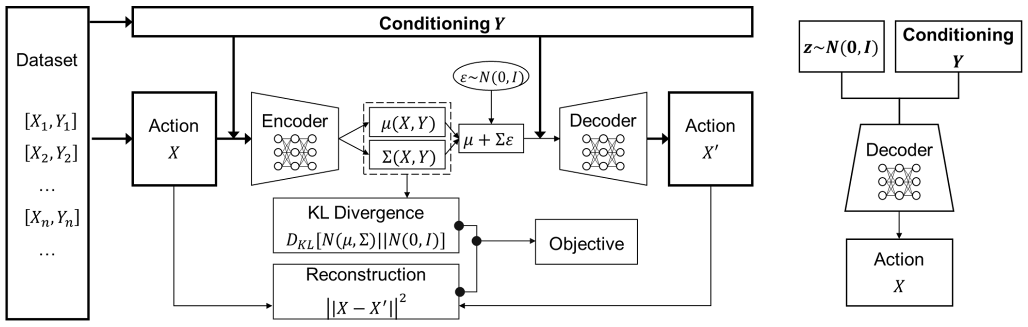

Figure 4: CVAE structure. (a): offline training. The best actions are encoded to the latent space and reconstructed by a decoder conditioned on the local map. (b): online evaluation. A random variable is drawn from the normal distribution and fed into the decoder together with conditioning to generate a sample.

To directly infer the model is intractable in practice, the variational lower bound is optimized instead:

$$\log P(x|y) \geq E_{z\sim Q}[\log P(x|z,y) - D_{KL}[Q(z|x,y)||P(z|y)]] \tag{1}$$

where $D_{KL}(\cdot)$ denotes the KL divergence. The lower bound can be approximated by Monto Carlo methods and efficiently optimized by standard machine learning techniques [15][16].

Architecture The encoder and decoder are parametrized by neural networks. They are trained simultaneously by minimizing the reconstruction error and regularized by the KL divergence. Fig. 4 demonstrates the overall architecture. To evaluate our approach, a random variable is drawn from $\mathcal{N}(0, I)$, conditioned on the local map, and projected to the state space through the learned decoder (Fig. 4 (b)).

4.4 Planning

Once the network is ready, online planning is conducted in a newly generated maze map. The robot starts from a randomly initialized position and explores the map within 20 minutes (Simulation time). At each step, the robot receives the local observation and predicts the next action by sampling the latent space for N times. Each sample is evaluated and the best one is chosen. If the planner fails to find a feasible solution within N samples, it resorts to the NBV planner and proceeds. The overall workflow is illustrated in Algo. 1

5 Results

We evaluate our approach against the NBV baseline planner on a 2D simulator. All of our simulations are run on an Intel® Xeon(R) CPU E3-1280 v5 @ 3.70GHz × 8 cores machine. We use one thread when exploring for both planners.

Algorithm 1 Learning Sampling-based Exploration Outline

```
Offline:
 1: Input: Training data size $L$, empty dataset $Dataset$
 2: for $i = 1 \rightarrow i = L$ do
 3:     if local map is fully explored then
 4:         reset simulation
 5:     end if
 6:     call NBV uniform sampling planner
 7:     $Dataset$.append( NBV sample $X_i$, local map condition $Y_i$ )
 8:     robot.moveTo($X_i$)
 9: end for
10: Train the encoder, decoder as in Fig. 4(a)
Online:
11: Input: robot initial position, unknown map, sample fraction $\lambda$
12: repeat
13:     Detect surroundings via camera and get local map $Y$
14:     Generate $\lambda N$ samples from learned planner,
15:     Generate $(1 - \lambda)N$ samples from NBV planner
16:     Check feasibility and evaluate gain of each samples, choose the best one $X$
17:     if no feasible sample found then
18:         call NBV uniform sampling planner for the best sample $X$
19:     end if
20:     robot.moveTo($X$)
21: until Map is fully explored
```

Distribution Capture To demonstrate the behavior of our planner, we sample multiple times from the NBV planner and our planner. As shown in Fig. 5, the left figures are samples from the NBV uniform planner, and the right ones are from our learned planner. Our planner can capture the main distribution of optimal samples despite some outlier samples (top left figure: samples on the left do not appear in the case of NBV planner, as our planner is only conditioned on the local map and the NBV planner prefers samples that are easier to reach, aka, in the same velocity direction). Our planner can also capture the distribution of multi-mode optimal samples as depicted in the bottom right figure.

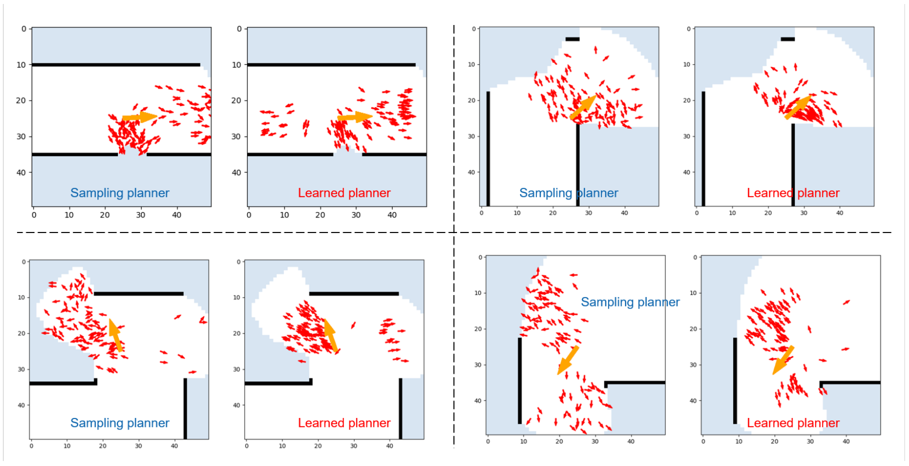

Figure 5: Illustration of the distribution of best samples generated by the NBV uniform planner and our planner. We sample 100 times for each planner and our planner (right) can capture the main distribution of the NBV teacher planner (left).

Exploration Performance To elaborate the performance of our approach, we compare three different planners: Our planner, NBV baseline planner, and a mixture planner, where 30% samples are drawn from the NBV planner and 70% are from our planner. We sample $N \in \{1, 5, 10, 20, 50, 100\}$

times to show the performance of our approach in different scales. As is shown in Fig. 6, our planner is able to explore faster in all cases. When the samples increase, the exploration difference becomes narrower as the uniform planner tends to find the optimal solution with more samples.

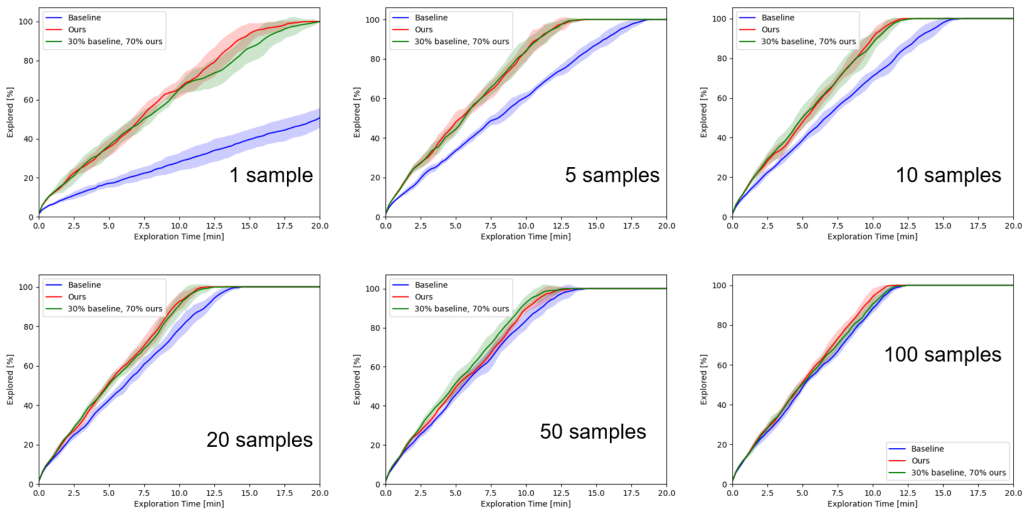

Figure 6: We compare exploration speed of 3 planners in 6 sub-tests: NBV uniform planner (blue), our learned planner (red), and a mixture of both planners (green), where 30% of samples are drawn from the NBV planner and 70 % are from our planner.

Computational Cost In this section, We elaborate on the computational gain of our planner. As our planner explores faster, it takes less time to fully explore the map with the same number of samples each step. Fig. 7 shows us drawing 5 samples each step can achieve similar exploration performance as an NBV planner takes 50 samples, while our planner only takes about one-fifth of the computational time. The NBV planner is Pareto dominated by our learned planner (Fig. 7 (left)) and consumes much less computational resources to obtain similar exploration (Fig. 7 (right)), which makes it ideal for hardware experiments.

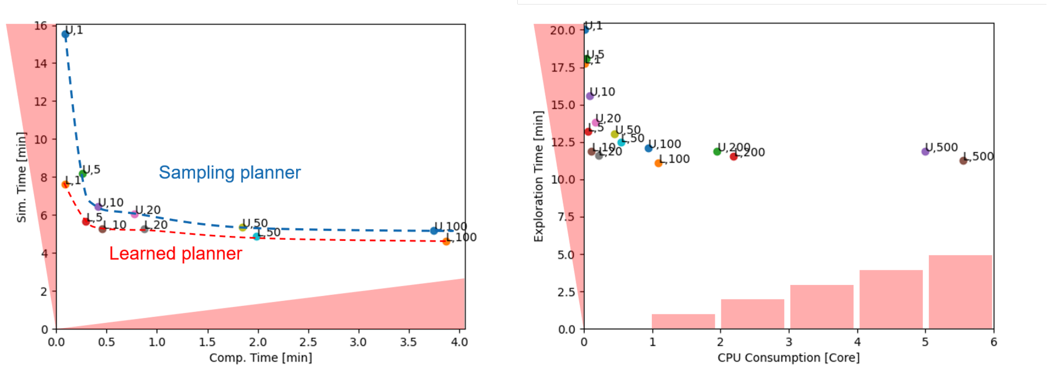

Figure 7: Pareto diagram. L and U represents the learned planner and the NBV uniform planner respectively. Alongside the alphabet is the number of samples used for the planner. The NBV planner is pareto dominated by our learned planner. Less computational resources are needed for our planner to obtain same exploration level.

Generalization Capabilities To show the generalization capabilities of our learned planner, We evaluate on a cubic-and-sphere map with the planner trained on the maze map. Compared to the maze map, this map consists of relatively large obstacles which are very different from walls in the maze map. However, our planner can generalize its performance into this new map and explores faster in most cases. As depicted in Fig. 8.

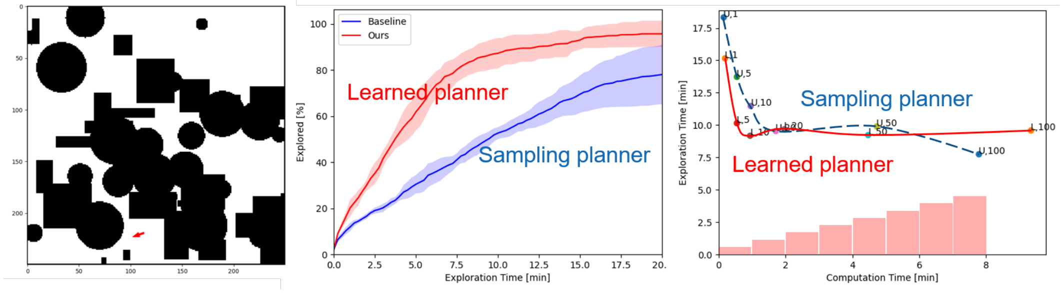

Figure 8: On a different cubic and sphere map, our planner trained with maze also outperforms the NBV uniform planner.

Gain Prediction In our previous experiments, we evaluate each sample by checking the feasibility and calculating its gain, which is expensive and takes the majority of computing time. If we are able to infer the gain for each sample directly, then we can rank the samples by gain and pick the highest one.

We extend the robot state to $\tilde{x} = [x, G(x)]$ where $G(\cdot)$ evaluates the gain. A preliminary result (Fig. 9) shows the potential benefit predicting gain can bring us. When the sample number is low, our planner outperforms the baseline NBV planner. However, the learned planner does not perform well when we use more samples. We think that may be due to following reasons: 1) The gain of each piece of training data is rather random and difficult to reconstruct; 2) The implicit Gaussian assumption may not valid for the latent distribution, as multi-mode optimum exists in exploration.

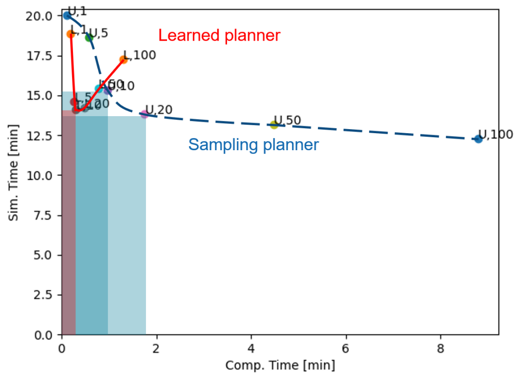

Figure 9: Gain Prediction: reconstruct the gain for each sample directly and pick the sample by sorting the predicted gain value.

6 Conclusion

In this paper, we presented a learning sampling-based approach to efficiently explore the unknown environment with limited computational resources. We utilized CVAE to learn the hidden distribution from the NBV samples collected by a uniform planner. The learned planner explores faster compared to the Next-Best-View (NBV) uniform planner and requires less time to fully explore the environment. We showed the generalization of our approach by applying the learned planner on an unseen map and achieves faster exploration. Furthermore, we extend the application of CVAE to the exploration planning field and our learning scheme can be readily adapted to other exploration tasks. Future research includes hardware validation of our approach and the improvement of our gain prediction variant.

References

[1] K. Sohn, H. Lee, and X. Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28:3483–3491, 2015.

[2] S. M. LaValle et al. Rapidly-exploring random trees: A new tool for path planning. 1998.

[3] S. Karaman and E. Frazzoli. Sampling-based algorithms for optimal motion planning. *The international journal of robotics research*, 30(7):846–894, 2011.

[4] L. Schmid, M. Pantic, R. Khanna, L. Ott, R. Siegwart, and J. Nieto. An efficient sampling-based method for online informative path planning in unknown environments. *IEEE Robotics and Automation Letters*, 5(2):1500–1507, 2020.

[5] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Siegwart. Receding horizon" next-best-view" planner for 3d exploration. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1462–1468. IEEE, 2016.

[6] J. Schmidhuber. Curious model-building control systems. In *Proc. international joint conference on neural networks*, pages 1458–1463, 1991.

[7] Y. Sun, F. Gomez, and J. Schmidhuber. Planning to be surprised: Optimal bayesian exploration in dynamic environments. In *International Conference on Artificial General Intelligence*, pages 41–51. Springer, 2011.

[8] R. Houthooft, X. Chen, Y. Duan, J. Schulman, F. De Turck, and P. Abbeel. Vime: Variational information maximizing exploration. *arXiv preprint arXiv:1605.09674*, 2016.

[9] H. Zhu, J. J. Chung, N. R. Lawrance, R. Siegwart, and J. Alonso-Mora. Online informative path planning for active information gathering of a 3d surface. *arXiv preprint arXiv:2103.09556*, 2021.

[10] B. Ichter, J. Harrison, and M. Pavone. Learning sampling distributions for robot motion planning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7087–7094. IEEE, 2018.

[11] T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat. State representation learning for control: An overview. *Neural Networks*, 108:379–392, 2018.

[12] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, P.-A. Manzagol, and L. Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010.

[13] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pages 1278–1286. PMLR, 2014.

[14] D. Hoeller, L. Wellhausen, F. Farshidian, and M. Hutter. Learning a state representation and navigation in cluttered and dynamic environments. *IEEE Robotics and Automation Letters*, 6 (3):5081–5088, 2021.

[15] C. Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[16] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.