An Expansive Latent Planner for Long-horizon Visual Offline RL

Robert Gieselmann (robgie@kth.se) and Florian T. Pokorny (fpokorny@kth.se)

Introduction

Sampling-based motion planning algorithms search for global solution paths in geometrically complex environments. We argue that model-based reinforcement learning (RL) under sparse rewards could benefit from such powerful planning strategies. Previous work [2] achieves planning from visual observations by adapting Expansive Space Trees (ESTs) [3] to search for paths that connect states in a learned state embedding. In this work, we extend [2] towards the more general reward-based learning setting.

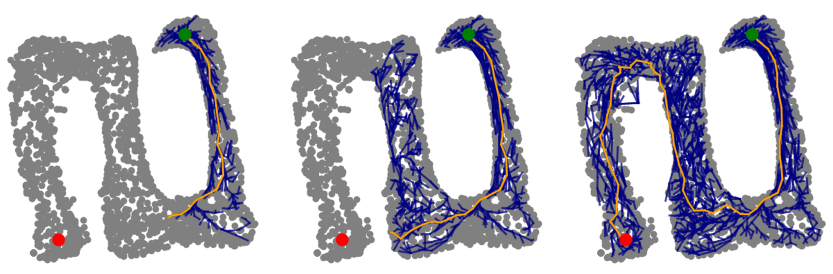

Figure 1. Taken from [2]: A tree (blue) is iteratively grown and optimized while being bound to the estimated latent support region (gray).

Type of Control Tasks

Our goal is to solve MDP tasks with continuous states, actions, and a binary reward function indicating success. We evaluate our method on problems with high-dimensional visual observations (short video sequences) and provide an offline dataset $\mathcal{D}$ consisting of suboptimal trajectories for training.

Our Method - Overview

VELAP presents a model-based RL agent that determines sequences of subgoals towards a global goal (region of positive reward) through tree-based exploration of a latent state space.

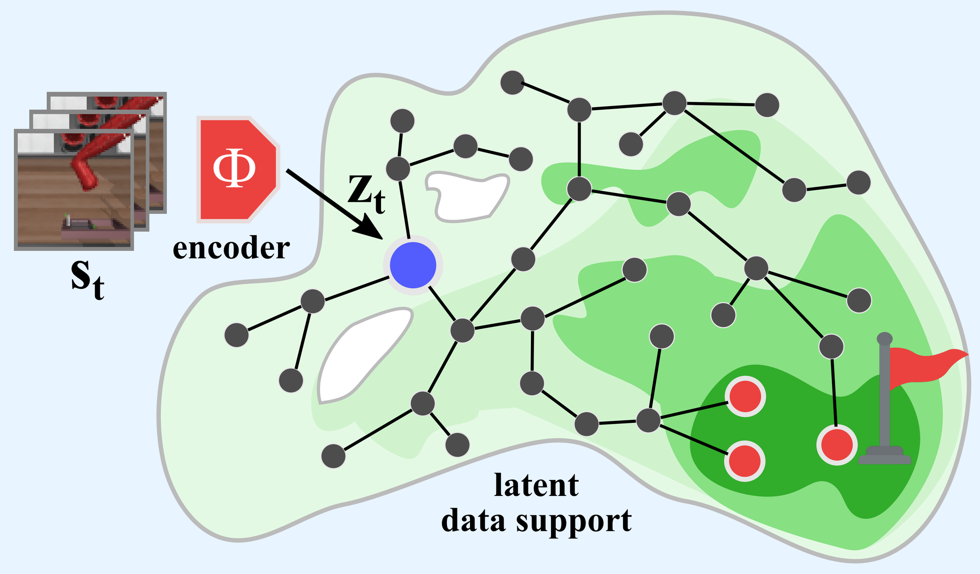

Figure 2. A search tree is grown in the latent space to globally explore reward-maximizing paths (blue:start, red:goal nodes, green: estimated values).

Our method consists of the following components:

$$
\begin{aligned}
&\text{State encoder:} && \phi : \mathcal{S} \to \mathcal{Z} \\
&\text{Dynamics:} && h : \mathcal{Z} \times \mathcal{A} \to \mathcal{Z} \\
&\text{Action model:} && g : \mathcal{Z} \times \mathbb{R}^m \to \mathcal{A} \\
&\text{Local policy:} && \pi^l : \mathcal{Z} \times \mathcal{Z} \to \mathcal{A} \quad Q^l : \mathcal{Z} \times \mathcal{Z} \times \mathcal{A} \to \mathbb{R} \\
&\text{Global policy:} && \pi^g : \mathcal{Z} \to \mathcal{A} \quad Q^g : \mathcal{Z} \times \mathcal{A} \to \mathbb{R}
\end{aligned} \tag{1}
$$

Offline Model Learning

We train the state encoder jointly with our dynamics model and local/global policy/Q-function (TD3-BC [1]). The local policy learns state reaching behavior for which we synthesize a dataset $\mathcal{D}'$ using hindsight goal relabeling. For $\mathcal{L}_h$, we use a contrastive objective similar to CPC [4].

$$\mathcal{L}_{\text{model}} = \mathcal{L}_{Q^l} + c_0 \cdot \mathcal{L}_{Q^g} + c_1 \cdot \mathcal{L}_h \tag{2}$$

$$\mathcal{L}_{Q^l} = \mathbb{E}_{\mathcal{D}'}[(Q^l(z_t, z^g, a_t) - (r_t + \gamma Q^l(z_{t+1}, z^g, \pi^l(z_{t+1}, z^g))))^2] \tag{3}$$

$$\mathcal{L}_{Q^g} = \mathbb{E}_{\mathcal{D}}[(Q^g(z_t, a_t) - (r_t + \gamma Q^g(z_{t+1}, \pi^g(z_{t+1}))))^2] \tag{4}$$

References

[1] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[2] Robert Gieselmann and Florian T. Pokorny. Latent planning via expansive tree search. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[3] D. Hsu, J.-C. Latombe, and R. Motwani. Path planning in expansive configuration spaces. In *Proceedings of International Conference on Robotics and Automation*, volume 3, pages 2719–2726 vol.3, 1997.

[4] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[5] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

Planning Algorithm

Our planner creates a tree in the latent state space by iterating between (a) randomly choosing an existing node z_{exp} in the tree (b) generating a new node z_{new} from z_{exp} using the learned dynamics model.

The tree is sparsified by rejecting nodes that are too close to existing ones. We avoid exploration outside the data support by discarding unlikely transitions.

Algorithm 1 Node sampling and tree expansion

Given: z_{init}, n_{iter}, n_{sim}, $\tau^{\text{neigh}}_{\text{discard}}$, $\tau^{\text{std}}_{\text{discard}}$ g, h, π^g, Q^g, Q^l_k, π^l
Initialize: $\mathcal{V} \leftarrow \{z_{\text{init}}\}$, $\mathcal{E} \leftarrow \emptyset$
for n_{iter} steps **do**
  Sample node z_{exp} from $\mathcal{V}$ given $P_{\text{node}}(\mathcal{V})$
  $z_{\text{new}} \leftarrow z_{\text{exp}}$
  Simulate forward using dynamics for n_{sim} steps
  for n_{sim} steps **do**
    Sample action $a \sim g(.|z_{\text{new}})$ (or $a = \pi^g(z_{\text{new}})$)
    $z_{\text{new}} \leftarrow h(z_{\text{new}}, a)$
  end for
  Reject node if too close to existing one in the tree, too far from expansion node or if the value uncertainty is too high
  if $Q^{\text{exp,new}}_{\text{min}} > \tau^{\text{low}}_{\text{discard}}$ and $Q^{\text{exp,new}}_{\text{std}} < \tau^{\text{std}}_{\text{discard}}$ **then**
    if $\max\{Q^{i,\text{new}}_{\text{min}} | z_i \in \mathcal{V}\} < \tau^{\text{high}}_{\text{discard}}$ **then**
      Add new node to tree
      $V \leftarrow V \cup \{z_{\text{new}}\}$; $E \leftarrow E \cup \{z_{\text{exp}\to\text{new}}\}$
    end if
  end if
end for

Biased node+action sampling is introduced to ensure efficient and goal-direction exploration (see full article).

MPC Evaluation in Simulation

We embed our planner into a MPC loop. After every replanning step, we identify the set of tree nodes with close vicinity to the goal and pick the one associated with the minimum travel cost. The local policy achieves navigation between the waypoints of the planned paths.

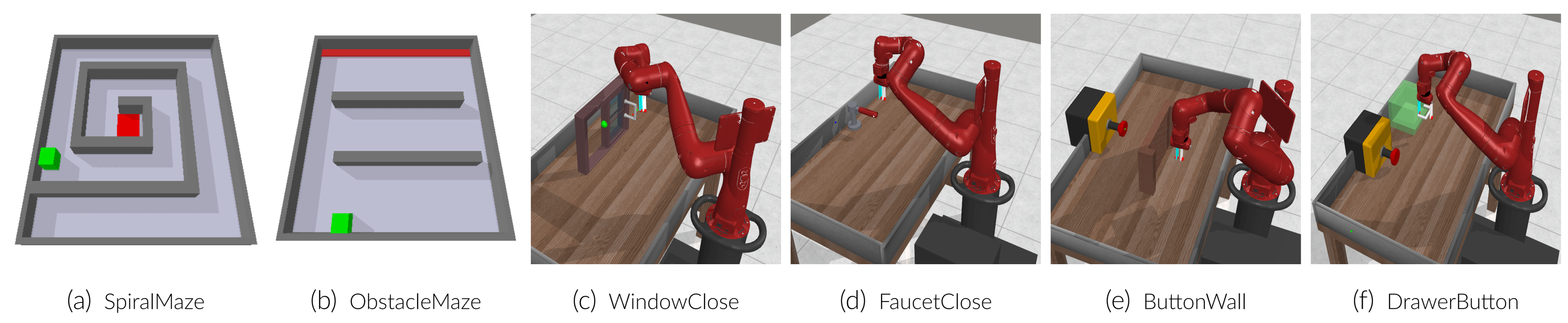

Figure 3. Vision-based control environments(c-f adapted from meta-world benchmark [5]).

Table 1. Success rates (%) on test scenarios.

Method	BC	BC ($\mathcal{D}^*$)	TD3-BC	MPPI	MBOP	IRIS	IRIS (multi-step)	VELAP
Spiral Maze	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	15 ± 31	**94 ± 3**
Obstacle Maze	0 ± 0	15 ± 6	35 ± 22	83 ± 11	40 ± 25	50 ± 25	62 ± 14	**97 ± 2**
Window	0 ± 0	34 ± 11	16 ± 8	70 ± 7	23 ± 4	69 ± 3	43 ± 20	**78 ± 4**
Faucet	0 ± 0	36 ± 6	13 ± 7	41 ± 7	33 ± 2	10 ± 2	3 ± 1	**51 ± 12**
ButtonWall	0 ± 0	0 ± 0	2 ± 2	9 ± 10	0 ± 0	35 ± 5	8 ± 8	**76 ± 9**
DrawerButton	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 0	5 ± 3	0 ± 0	**11 ± 3**

Visualizations

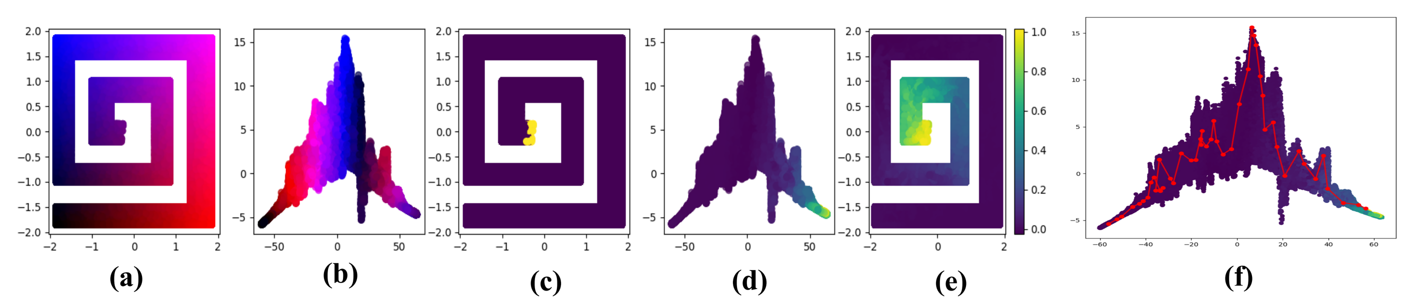

Figure 4. *SpiralMaze*: (a) xy coordinates of robot (b) 2d-embedding of latent space (c) environment reward (d) learned Q-values (latent space) (e) learned Q-values (xy) (f) example scenario (g) latent path

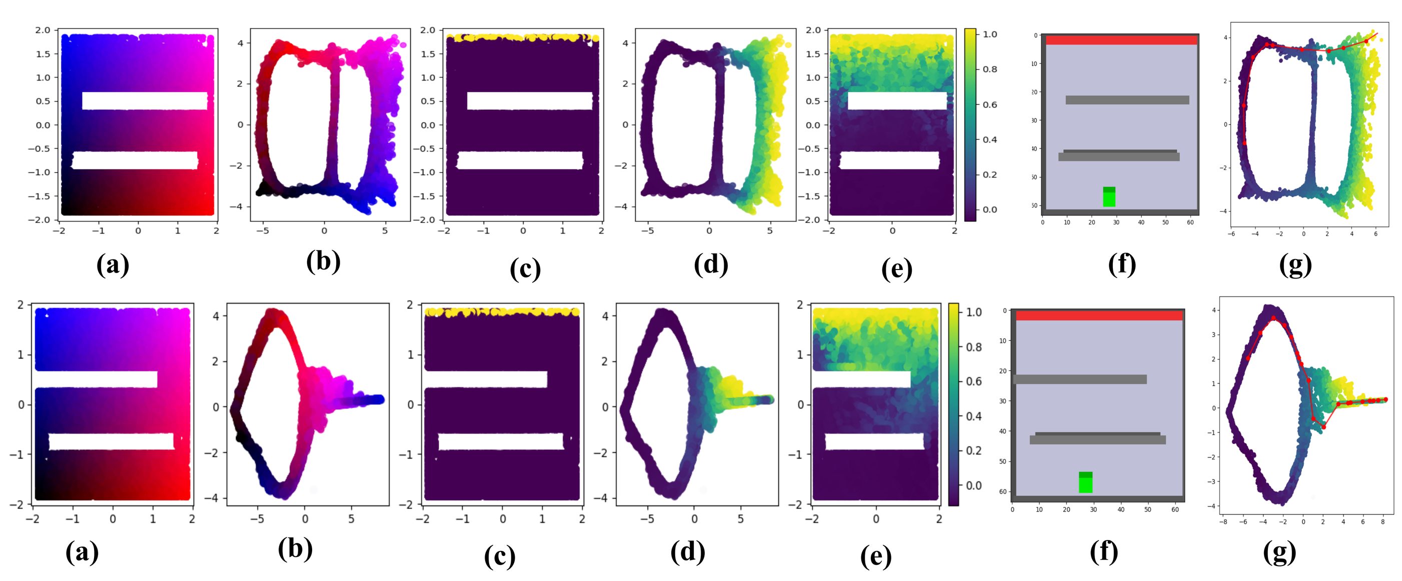

Figure 5. *ObstacleMaze*: (a) xy coordinates of robot (b) 2d-embedding of latent space (c) environment reward (d) learned Q-values (latent space) (e) learned Q-values (xy) (f) example scenario (g) latent path

Limitations and Future Directions

VELAP is currently limited to fully observable states. We plan to extend our method to partially observable states by adapting recursive state estimation. Other future directions include planning strategies that account for uncertainty, or multimodal state representations (e.g. include proprioceptive information).