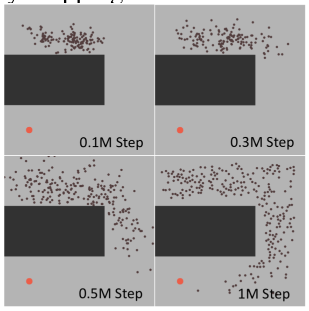
We thank all reviewers for the constructive comments. We will carefully proofread the paper and add suggested references. Some suggested papers are from unpublished arXiv papers ([2,3] suggested in R3's review). According to the review policy, reviews should not be based on them, but nonetheless, we will still comment them for clarification.

How to learn a structured model for model-based RL is a hot topic. We attempt to solve goal-conditioned policies in large MDPs with ***sparse*** rewards, in which exploration and routing across remote states are both extremely challenging. Our approach maintains a ***dynamic landmark-based map*** abstracting the ***visited state space*** to facilitate the routing and exploration. To build the map, selecting landmarks and adding edges among them are both based on ***learned*** information of the environment. Experimentally we showed that the approach can solve long-range goal reaching problems better than model-free methods and hierarchical RL methods, for a number of challenging games.

Significant Concerns:

Novelty in Map Construction and Replay Buffer Usage (R1&R3). Instead of trivially mapping the whole state space by uniform sampling (R3's review), our landmarks are dynamically sampled from the replay buffer by iterative FPS algorithm using distances estimated by the UVFA Q-function (LN203-204) and get updated at the beginning of every episode. The FPS sampling tends to find states at the boundary of the visited space, which implicitly helps exploration (LN50-53). Figure on the right shows selected landmarks at four training stages on AntMaze with a fixed starting point at the upper-left corner. Landmarks expand gradually towards the goal (red dot), even if it only covers a small proportion of states at the beginning. FPS has much higher success rate than uniform sampling: 0.4 vs 0.15 at 0.5M training step, 0.7 vs 0.3 at 0.75M training step, and 0.95 vs 0.7 at 1M training step. ***This map construction strategy is adaptive to the capability of the policy network, the UVFA network, and the task (rewards), without assuming domain-specific knowledge.***

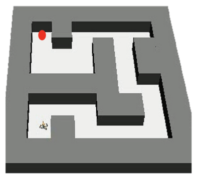


Comparison with [1,2,3] suggested by R3. [1] and [2] (unpublished) learned environment models using domain knowledge with explicit supervision. They required user-defined attributes and learned the state-attribute mapping in a supervised way. [3] (unpublished) proposed a complex system of relevant high-level idea with many hyper-parameters (19) for very different problem settings and solutions.According to the review policy, discussions are left in future work.

No Comparison with HRL (R3). Sec 4 of supplementary compared our method with SOTA HRL algorithms (HIRO and HAC with authors' published code) on big AntMaze of size (24×24). The game is quite challenging for mixing control (30D) and navigation (2D) in states. With sparse rewards, neither HIRO nor HAC can solve it, but ours can solve it efficiently. With dense rewards, our method outperforms HIRO and HAC significantly (recent results on HAC).

Applicability to General MDPs (R1 & R2). Our framework is theoretically applicable to general MDPs. Indeed, the state space of a few MDPs in our experiments mixes information NOT about location for navigation: AntMaze and PointMaze use 28D and 4D for the control of robot body, FetchReach uses 7D for the control of gripper, and FetchPush is a robot-object interaction task that has 15D for the object under manipulation and 7D for gripper control. Note that we do not use any information (such as control cost) other than the sparse rewards for goal reaching. This set of environments have basically covered the commonly used games for goal-conditioned MDPs.

We also recently tested on a non-navigational control task (Acrobot), an Atari game (MonteZuma's Revenge), and a complex locomotion+navigation task (complex AntMaze). Note that the original Acrobot and MonteZuma's Revenge are not universal goal-reaching tasks, but we re-wrap them by sampling goals. For Acrobot, goals are states that the end-effector is above a certain line at specific joint angles and velocities. For MonteZuma's Revenge, goals are sampled according to the RND-based novelty value. *The modification makes them much harder than the original version for demanding the ability to reach arbitrary goals.* One Acrobot, our method achieved a higher success rate than HER (0.65 vs 0.25 after 0.5M steps). On MonteZuma's Revenge, our method outperformed HER+RND by a large margin and can successfully reach the key in the first room while HER fails. Note that our backbone algorithm, DQN, is well-known to perform poorly on this game. Given more time, we would explore other favorable backbone algorithms, like PPO. Finally, we evaluated our method on a larger and more complex AntMaze, which requires 1500 steps (HIRO and HAC both fail). Our method can occasionally achieve the goal at an early stage ($\sim$0.5M training steps), and converged at 0.85 success rate before 3M steps.



Individual Concerns:

Figure 3 and Figure 4(b) (R1). For Figure 3, our method will also use the planner to collect data at training time. The curve shows the evaluation performance. For Figure 4(b), it shows that a harder goal requires more steps to reach, and the performance of HER is more unstable.

State Space and Goal Space (R2). Usually it is hard and unnecessary to precisely realize intermediate subgoals for long-term planning. In AntMaze, the landmarks include only the target position but not any locomotion information, and we found that this is enough for the long-term planning. However, for other environments like the 2DPush and FetchPush, the goal vector only describes the target objects' location but loses information of the gripper, which is essential for planning. So in these environments, the landmarks are selected in the original state space. A second HER from the last landmark to the goal (the object) is learned to complete the planning.

Actor Network and DQN (R3). For DQN, the action is generated from the maximal Q-value, which is different from DDPG. But the solutions of using Q-value to connect landmarks are similar.