We thank the reviewers for their helpful comments. To address **R2** and **R3** concerns, we modified the manipulation task to be more challenging and better test generalization. We added two new state-of-the-art baselines (PETS [1] and HER [2]). We also present preliminary HIRO [6] comparisons, as requested by **R3** . To address **R3** , we added an experiment studying different optimizer choices. These additional experiments should address the primary concerns raised by the reviewers. We summarize the important points below.

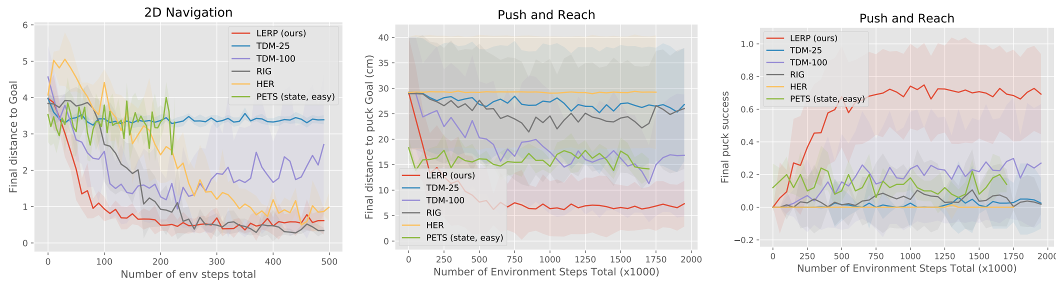

Figure 1: We made Push and Reach more challenging and added new baselines (PETS, HER). MPC results are omitted for clarity.

R2 , **R3** : Regarding additional comparisons, the baselines now include PETS, a model-based method, and HER, a goal-conditioned method. The model-free TDM in the submission is already trained on the VAE state representation, as **R3** requested. Figure 1 shows that LERP significantly outperforms these methods.

R3 : To address questions of generalization, we have modified the Push and Reach task. We now varied the initial state configuration on the pushing task during test time to include 5 rather than 1 challenging configuration. The new Push and Reach experiment shows that LERP significantly outperforms prior methods at generalizing. Specifically, Figure 2 shows that only LERP can solve all initial configurations, while the next-best method (TDM-100) fails to consistently solve any of them. Due to time constraints, we did not have time to run the “different block” configuration requested, and exploring generalization to new environment configurations would be an interesting avenue for future work.

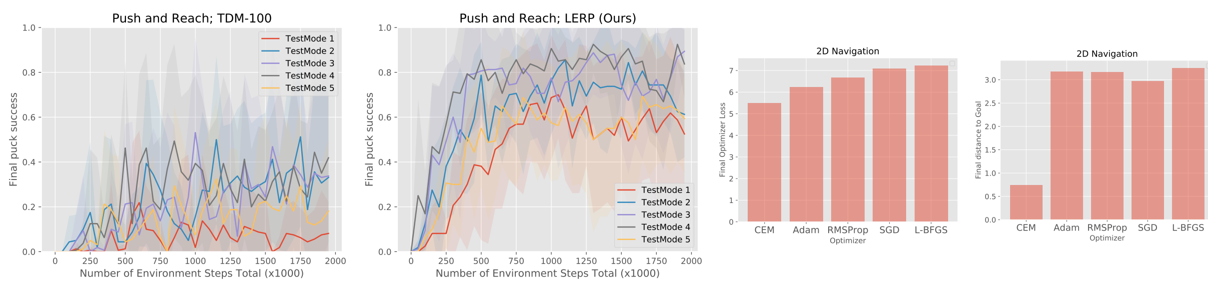

Figure 2: (Left) If we split the next-best method (TDM-100) by test configuration, it fails to generalize to all configurations. By contrast, we see that LERP solves all configurations. (Right) We compare different optimizers for LERP. We found that CEM outperformed L-BFGS, Adam, RMSProp, and gradient descent (SGD) even after tuning the learning rate.

R3 : We compared to gradient-based optimizers after tuning their learning rates. Figure 2 shows that CEM consistently performed the best, likely due to its ability to escape local optima. Using more advanced non-gradient optimizers would be promising future work.

R2 : We increased the number of seeds from 3 to 8, for Push and Reach. The shaded region represents one standard deviation across seeds. We will update figures accordingly and describe the shaded region in Section 5.

R3 : We also compared to HIRO on the 2D Navigation. Due to time constraints, we were only able to run one seed, but the preliminary results in Figure 3 suggest that the existing baselines and our method signficantly outperform HIRO.

Figure 3: Preliminary HIRO results HIRO on 2D Navigation. Compared to Figure 1 (left), HIRO does not appear competitive.

R3 : Thank you for the additional references. We will add a discussion to the Related Works section.

[1] Chua et al. Deep Reinforcement Learning in a Handful of Trials using Probabilistic Dynamics Models. NeurIPS 2018. [2] Andrychowicz et al. Hindsight Experience Replay. NeurIPS. 2017. [3] Florensa et al. Automatic Goal Generation for Reinforcement Learning Agents. ICML 2018. [4] Pong et al. Skew-Fit: State-Covering Self-Supervised Reinforcement Learning. CoRR 2018. [5] Zhao et al. Energy-Based Hindsight Experience Prioritization. CoRL. 2018. [6] Nachum et al. Data-Efficient Hierarchical Reinforcement Learning. NeurIPS. 2018.