We thank all reviewers for their time and appreciate the thoughtful feedback. And we are happy that all reviewers agree that the topic of our paper is both interesting and important. We only have space to address the main concerns below, but will take into account all feedback for the camera-ready version.

R2: "I don't think this paper will be very impactful if it only shows results on a toy domain." We respectfully disagree; in fact, we argue the opposite: the fact that we can show the limitations of MuZero on a simple domain like Mountain Car strenghtens our claims rather than weakens it. If MuZero is not able to get close to 0 LoCA regret on a trivial task like Mountain Car, it definitely won't be able to achieve this in more complex domains.

R3: There are multiple issues with the top-terminal fraction. 1) It only measures optimality. 2) Optimal is defined as reaching an end-point, but optimal also depends on how fast this happens. 3) in complex tasks, it can be hard to define the top-terminal fraction. Great points, but there is a small, crucial part in our definition of the top-terminal fraction to prevent precisely issues 1) and 2) mentioned here. We define the top-terminal fraction as the number of times the agent end up in terminal T2 *within a certain time limit*. We mention this on line 136, but admit it is somewhat hidden and will highlight this better in our next iteration of the paper. In our experiments, we have set the time limit at approximately 90% of the average time an optimal policy requires, starting from the evaluation initial-state distribution. Regarding point 3), as long as a meaningful variation with two terminal states can be constructed, a well-defined top-terminal fraction exists. Furthermore, see our relevant remarks at the end of this page.

R4: "the authors haven't shown how to use this metric to further analyze and improve the model" First, we'd like to push back on the implicit notion that identifying a problem is not a valuable contribution in and of itself; many influential papers do just that. Furthermore, note that we *do* perform analysis using the LoCA regret (w.r.t. planning hyperparameters), which leads us to the important observation that on-policy elements hurt the ability to quickly adapt. This provides guidance/clarity to the research community that new techniques should be investigated to get model-based methods that achieve both good performance in long-horizon tasks as well as fast adaptation.

R1: "I can imagine model-based methods that adapt slowly and model-free methods that adapt fast." You bring up a great point: sophisticated model-free methods can behave very similar to model-based methods. That's why the primary goal of the LoCA regret is not to try to identify the internal process a method uses, but to identify *useful behavior* that, according to neuroscience, is associated with model-based learning.

R1: "The paper would be strengthened by a clear example where LoCA is able to distinguish between sample complexity improvement due to confounding factors vs effective planning". Also, **R2: "Where do the authors evaluate examples of a great representation?"** and **R3: "I am not sure whether your representation learning experiments really illustrate anything.** These shared concerns have made us realize that the experiments from Section 3.3 should be better explained. We do believe these experiments are the right ones to show, but will add further explanation as to why these are relevant in the context of representation learning. In particular, we want to clarify the following: if method A uses a state-space with additional random features and method B uses a state-space without such features, then method B can be viewed as having a representation-learning module, compared to method A. Because if method B was given the same state-space as method A, but would also have a representation-learning module that learns to ignore the irrelevant random features during pretraining, the LoCA regret would be the same. So the comparison between, for example, regular Sarsa(λ) (without random features) and MB-VI, $S_{mult} = 5$ (which has random features) can be viewed as two methods operating on the same state space, where one method uses no planning but has a representation-learning module, while the other has no representation-learning module, but uses planning. We hope this clarifies things.

R3: "In the tabular setting, to overcome extra noise features in the state-space, you just need more data and training iterations. [...] If pretraining phase 2 is long enough, you always will correct for the noise." Under the condition that a method can find a near-optimal policy in the limit (which holds for all our experiments), the difference between a poor and a good representation expresses itself *only* through data efficiency, also in non-tabular settings. Our LoCA pretraining is designed among others to remove the effect of the representation, as it is a confounding factor. Also, the effect is only removed for model-based methods; if a method is model-free (Sarsa(λ)) or uses limited planning (MB-SU), the representation does effect the LoCA regret (see Table 1). This effect does *not* go away by having a long pretraining phase 2, because only a restricted part of the state-space is visited during this phase.

R3: "I think it is not trivial to design new LoCA tasks in more complex problems, especially because the agent needs to set a restricted region [...]." Even if designing new LoCA tasks would take some engineering effort, this does not substantially reduce the importance of this paper. Ultimately, there is no need to design a LoCA task for every possible domain; only a small set of representative domains is needed. Besides this, as long as a task-implementation gives a user the ability to set the state of the task, implementing a restricted region is straightforward: a wrapper around the task can be implemented that resets an agent to its previous state, as soon as an action moves the agent outside the restricted region, effectively giving such actions a 'no-op' effect.