We thank the reviewers for their insights and suggestions. Questions primarily concerned the utility of γ-model predictions and associated tradeoffs, policy-dependent models, and scalability. We clarify how γ-model predictions are used for decision-making (with further elaboration on value estimation), discuss how the tradeoff induced by the choice of γ is one that must be made in any RL algorithm, and comment on scaling to higher-dimensional environments. Answers below will be included in expanded discussions in future versions of the paper.

(R3) Utility of "jumpy" predictions. One of the insights of this paper is that model-based RL algorithms do not need to know the time at which a state will be encountered as long as states are sampled according to the discounted occupancy. This does mean that decisions should be based on expectations over many samples (Figure R1b) and not based on a single sample. In the case of R3's car example, as long as states from 10 steps into the future are sampled according to a probability of $p(t = 10) = (1 - \gamma)\gamma^9$, crashing at $t = 10$ will be reflected in such an expectation.

(R2, R3) Clarifications on RL experiments. (i) In RL experiments, γ-models were **not** pre-trained; they were trained online. (ii) Though bounds exist, it is difficult to relate control performance to state prediction accuracy in practice [Lambert et al., 2020]. The intermediate quantity of value prediction error (Figure R1a) is more informative in this respect, and shows a small accuracy improvement for $\gamma > 0.95$, consistent with the performance of γ-MVE vs MVE.

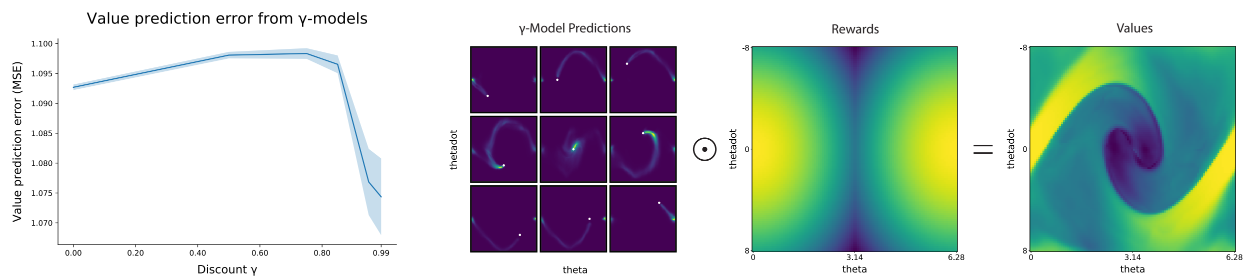

Figure R1: (a) Value prediction error for an occupancy with target discount $\widetilde{\gamma} = 0.99$ as a function of model discount γ. Higher γ's perform slightly better. **(b)** Visual depiction of value estimation (in Pendulum) as an expectation of reward over γ-model predictions.

(R1, R2) Tradeoff with γ. The γ-model bridges the gap between model-based and model-free learning, and as such allows one to trade off between training-time model-free errors (referred to as bootstrap error accumulation; Kumar et al. 2019) and testing-time model-based rollout errors. The tradeoff between these two is not a weakness of the γ-model, nor is it even unique to the γ-model; such compounding errors are unavoidable in sequential decision-making. The advantage of the γ-model is that it allows for more careful interpolation between these extremes.

(R2, R3) Policy-dependence. The γ-model is policy-dependent in the same way that a Q-function is. While the single-step transition distribution is certainly independent of a policy, parametric single-step models are still policy-dependent because the policy determines the training data distribution for statistical learning.

(R1, R4) Scaling to image-based environments. We agree that experiments in more complex domains would improve the submission. Unfortunately, model-based experiments in image domains like Atari games are quite complex to set up, generally requiring specialized image prediction models that can take weeks to train [Kaiser et al., 2019]. We have found that γ-model training scales to state spaces of the MuJoCo gym benchmark suite for discounts up to $\gamma = 0.9$, but leave extensions to image-based environments for future work.

Other questions from Reviewer 2 ❶ *Gradients:* The gradient computation indeed does not pass through the bootstrapped target. This is discussed in L211-L215 in Section 6 "Practical Training of γ-Models".

Other questions from Reviewer 3 ❶ *Prediction visualization:* The first five columns of Figure 3 depict learned γ-model predictions. The only Monte Carlo trajectory estimates are in the final column for comparison. ❷ *Relation to n-step models:* Training an n-step model for n larger than 10 or 20 quickly becomes impractical, and is impossible without on-policy samples. By directly learning the discounted occupancy, we are able to train a model with much larger effective (probabilistic) horizon while only using off-policy single-step transitions. ❸ *Bellman equation:* There is no recursion in Equation 2 because it describes a bootstrapped target distribution (akin to a bootstrapped target value $r(\mathbf{s}_t) + \gamma V(\mathbf{s}_{t+1})$) and is not a Bellman equation by itself. ❹ *Required rollout lengths:* Figure 2b shows the length of a γ-model-based rollout required to recover 95% of the probability mass of an occupancy with discount $\widetilde{\gamma}$. This is more precise than an effective horizon. For example, the effective horizon for $\widetilde{\gamma} = 0.99$ with a single-step model is 100, which recovers only $1 - .99^{101} \approx 64\%$ of the full infinite-horizon occupancy. ❺ *Energy-based model:* We introduce the EBM formulation because it provides the simplest way to explain how temporal difference updates may be used to train generative models. A neural generator reduces training times, but this is more an implementation choice than a fundamental component of infinite-horizon prediction, so we opt to first explain the framework with as few moving pieces as possible. Similarly, we have found that the γ-model may also be instantiated as a normalizing flow, and will add experiments comparing the flow and GAN instantiations of γ-models.

References N Lambert et al, *arXiv:2002.04523*. A Kumar et al, *arxiv:1906.00949*. L Kaiser et al, *arXiv:1903.00374*.