A Environments and Datasets

We describe the simulated environments and the dataset collection in more detail below.

Environments and Tasks. We evaluate our method in four simulated environments. The first task is the standard walker task from the DeepMind Control suite (Tassa et al., 2020). The observations consist of raw 64×64 images. The second experiment consists of a modified version of the D'Claw screw task from the Robel benchmark (Ahn et al., 2019), where the goal of the robot is to continuously turn the valve as fast as possible. The agent receives a dense negative penalty for positioning the fingers and a sparse reward whenever it turns the valve. The third environment is based on the Adroit pen task (Rajeswaran et al., 2018) with a fixed goal, which requires the agent to flip the pen around and catch it at certain angle. The observation space for both environments consist of robot proprioception and 128×128 raw images. These are challenging environments as the model needs to merge proprioception and visual information in order to estimate a hard contact model with realistic physics under a high-dimensional action space (9 and 26 respectively) and a sparse reward function (for the calw environment). The final simulation experiment is based on a Sawyer manipulation task, which requires opening a door. The agent received a sparse reward when the door is fully opened and no reward otherwise. The goal of this environment is to test learning in a realistic multi-stage robot arm environment with a sparse reward. This is a hard environment, not solvable with online RL. We provide visualizations of all simulated tasks in Figure 2.

Datasets. Thus, we construct new sets of offline datasets with image observations. Similar to the protocol by Fu et al. (2020), we create three types of datasets for each task, which are obtained by training an agent using soft-actor critic (Haarnoja et al., 2018) from the ground-truth state and recording the corresponding image observations, actions, and rewards. The *medium-replay* datasets consist of data from the training replay buffer up to the point where the policy reaches performance of about half the expert level performance. The goal of these datasets is to test learning on incomplete training data. The *medium-expert* datasets consist of the second half of the replay buffer after the agent reaches medium-level performance. The goal of these datasets is to test learning on a mixture of data from sub-optimal policies. The *expert* dataset consist of data sampled from the stochastic SAC expert policy. The goal of these datasets is to test learning on a thin data distribution.

B Ablation Studies on Varying Offline Dataset Size

We test the effect of dataset size, given that the data is sampled from the same distribution. We create a medium-expert dataset of size 1M on the DClaw Screw environment by mixing data from 3 separate policy training runs. We then created two more datasets by sub-sampling by a factor of 25 and 25 respectively. We observe that LMOPO still performs well in the low-data regime, as compared to regular latent model-based RL and Offline SLAC.

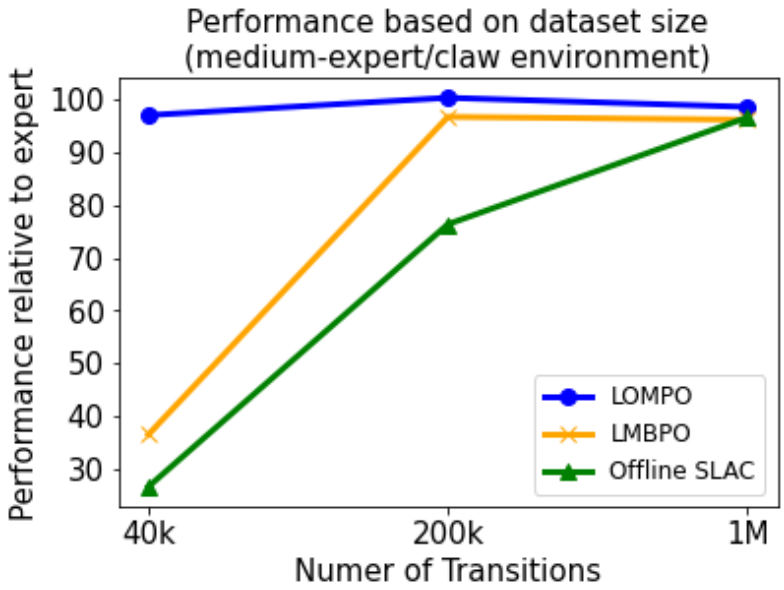

Figure 3: Agent performance based on dataset size

C Variational Latent Model Samples

For samples generated by our variational latent models, see Figure 4.

D Implementation Details

The latent dynamics model and the observation model consist of the following components as in (Hafner et al., 2020):

$$\begin{aligned} &\text{Image encoder:} && h_t = E_\theta(x_t) \\ &\text{Inference model:} && s_t \sim q_\theta(s_t|h_t, s_{t-1}, a_{t-1}) \\ &\text{Latent transition model:} && s_t \sim T_\theta(s_t|s_{t-1}, a_{t-1}) \\ &\text{Reward predictor:} && r_t \sim p_\theta(r_t|s_t) \\ &\text{Image decoder:} && x_t \sim D_\theta(x_t|s_t). \end{aligned} \tag{11}$$

where $x_t \in \mathcal{X}, s_t \in \mathcal{S}, a_t \in \mathcal{A}$. We use θ to denote the concatenation of all the parameters involved in the latent space dynamics model. The latent dynamics model is represented by RSSM. Specifically, we adopt the latent space representation $s_t = [d_t, z_t]$, which consists of a deterministic d_t and a sampled stochastic representation z_t. With such a latent space representation, we use the following components:

$$\begin{aligned} &\text{Deterministic State Model:} && d_t = f_\theta(d_{t-1}, z_{t-1}, a_{t-1}) \\ &\text{Stochastic Inference Model:} && z_t \sim q_\theta(z_t|h_t, d_t) \\ &\text{Ensemble of Transition Models:} && z_t \sim p_{\theta_k}(z_t|d_t, z_{t-1}, a_{t-1}) \end{aligned} \tag{12}$$

where $h_t = E_\theta(x_t)$ are observation features as defined in Eq. 11. The deterministic representation f_θ is implemented as a single GRU cell and is shared between the forward and inference models. All the forward models $T_{\theta_k}, k = 1, \ldots, K$ in the ensemble share the same deterministic model f_θ but separate stochastic transition models $p_{\theta_k}, k = 1, \ldots, K$, which are implemented as MLPs. Finally the stochastic inference model is also implemented as an MLP network.

The encoder network E_θ is modeled as a convolutional neural network. For the DeepMind Control walker task the network has 4 layers with [32, 64, 128, 256] channels respectively. For the D'Claw screw environment the convolutional model has 5 layers with [32, 64, 128, 256, 256] channels. All

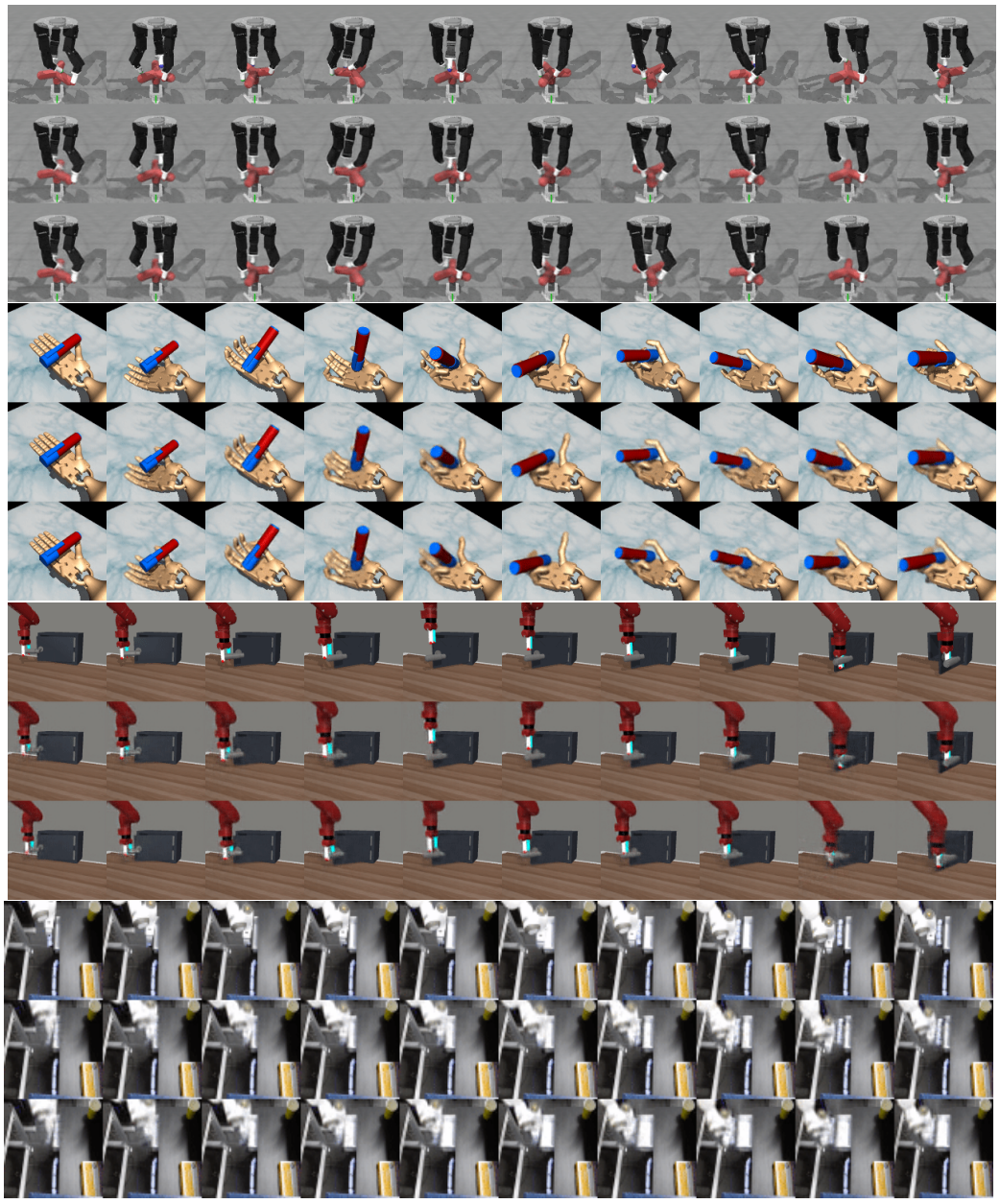

Figure 4: Samples from the learned variational model. Fist row: ground truth sequence; second row: posterior model samples; third row: ensemble latent model rollout.

kernels have size 4 and stride 2. The reconstruction model D_θ for DeepMind Control walker task has 4 layers with [128, 64, 32, 3] channels with kernel size [5, 5, 6, 6] and stride 2. For the D'Claw screw environment the reconstruction network has 5 layers with [128, 64, 32, 32, 3] with kernel size [5,5,5,4,4] and stride 2. The reward reconstruction network is a two-layer fully connected network with 200 units. The deterministic path f_θ of the RSSM is modeled as a GRU cell with 200 units. All the forward $T_{\theta_i}, i = 1, \ldots, K$ and inference q_θ models are 3-layer fully-connected networks with 200 units. The variational model is trained with the Adam optimizer with $lr = 6e - 4$.

Both the actor and the critic are modeled as fully-connected networks with 3 layers and 200 units. We use the Adam optimizer with $lr = 3e - 4$.

E Main Algorithm

Algorithm 1 LOMPO

Input: model train steps, initial real steps, initial latent steps, number of epochs, epoch real batch, epoch latent batch, number of models K

for *model train steps* **do**
  Sample a batch of sequences of raw observations $(o_{1:T}, a_{1:T-1}, r_{1:T-1})$ from batch dataset $\mathcal{D}_{\text{env}}$ and train variational ensemble model using equation 9 with K models

while *size* $\mathcal{B}_{real} <$ *initial real steps* **do**
  Sample a batch of sequences $(o_{1:T}, a_{1:T-1}, r_{1:T-1})$ from batch dataset D_{env}
  Sample latent states $s_{1:T} \sim q_\theta(s_{1:T}|o_{1:T}, a_{1:T-1})$ from the trained inference model
  Add batches s_t, a_t, s_{t+1}, r_t to real replay buffer $\mathcal{B}_{real}$

while *size* $\mathcal{B}_{latent} <$ *initial latent steps* **do**
  Sample a batch of sequences $(o_{1:T}, a_{1:T-1}, r_{1:T-1})$ from batch dataset $\mathcal{D}_{\text{env}}$
  Sample a set of latent states $\mathbf{S} \sim q_\theta(s_{1:T}|o_{1:T}, a_{1:T-1})$ from the trained inference model **for** $s_0 \in \mathbf{S}$ **do**
    for $h \in \{1 : H\}$ **do**
      Sample a latent transition model from p_{θ_j} from $i = 1, \ldots, K$
      Sample a random action a_{h-1} and next state $s_h \sim T_{\theta_j}(s|s_{h-1}, a_{h-1})$
      Compute reward $\tilde{r}_{h-1}$ using equation 10 and add $(s_{h-1}, a_{h-1}, s_h, \tilde{r}_{h-1})$ to $\mathcal{B}_{latent}$

for *number of epochs* **do**
  for *number actor-critic steps* **do**
    Equally sample batch $(\mathbf{s_t}, \mathbf{a_t}, \mathbf{r_t}, \mathbf{s_{t+1}})$ from $\mathcal{B}_{real} \cup \mathcal{B}_{latent}$
    Update $Q_\phi(s, a), \pi_\phi(a|s)$ using any off-policy algorithm
  for *epoch real batch* **do**
    Sample a batch of sequences $(o_{1:T}, a_{1:T-1}, r_{1:T-1})$ from batch dataset D_{env}
    Sample latent states $s_{1:T} \sim q_\theta(s_{1:T}|o_{1:T}, a_{1:T-1})$ from the trained inference model
    Add batches s_t, a_t, s_{t+1}, r_t to real replay buffer $\mathcal{B}_{real}$
  for *epoch latent batch* **do**
    Sample a batch of sequences $(o_{1:T}, a_{1:T-1}, r_{1:T-1})$ from batch dataset $\mathcal{D}$
    Sample a set of latent states $\mathbf{S} \sim q_\theta(s_{1:T}|o_{1:T}, a_{1:T-1})$ from the trained inference model **for** $s_0 \in \mathbf{S}$ **do**
      for $h \in \{1 : H\}$ **do**
        Sample a latent transition model from T_{θ_j} from $i = 1, \ldots, K$
        Sample an action $a_{h-1} \sim \pi_\phi(a|s_{h-1})$ and next state $s_h \sim T_{\theta_j}(s|s_{h-1}, a_{h-1})$
        Compute reward $\tilde{r}_{h-1}$ using equation 10 and add transition to $\mathcal{B}_{latent}$