We thank reviewers for their thoughtful feedback! We are encouraged that they find our problem setup well-done (R1) and significant (R3). We are glad they think our method is novel (R1), sound (R3, R4) and potentially promising (R2). We are pleased that they find our method learns explainable (R1) and interpretable (R2) energy that allows for control (R2, R4). We address a few specific concerns below and will incorporate all feedback in the camera-ready version.

@R2 - **Motivation:** Humans can learn to predict the trajectories of mechanical systems, e.g., basketballs or drones, from high-dimensional visual inputs, and learn to control the system, e.g., catch a ball or hover a drone, after a small number of interactions with those systems. We hypothesize that humans use domain-specific knowledge, e.g., physics laws, to achieve efficient learning. Motivated by this hypothesis, we incorporate the Lagrangian prior to learn and control dynamics from image data, hoping to gain interpretability and data efficiency (Please see L25-27 below.)

@R1, R4 - **Quantitative results:** We pointed out (L240) that pixel MSE is not a good metric for *long term* prediction. Please see [10] (Sec. 1) for more discussion. Instead, in the table below, we report the *short term* ($T_{\text{pred}} = 4$) average pixel MSE. Our proposed model has significantly better performance on the CartPole and Acrobot tasks than model variants and has similar performance to model variants on the simpler pendulum task. To R4, we have reported on training data generation including the sizes (SM Sec. S2.1). We generated testing data of equal size. We will include details on model architecture in camera-ready. We will also open source our code for all the experiments.

Average pixel MSE	Pendulum (train | test)		CartPole (train | test)		Acrobot (train | test)	
Lagrangian + caVAE	1.82	1.83	**2.78**	**2.81**	**3.06**	**3.14**
Lagrangian + VAE	**1.52**	**1.56**	9.41	10.30	10.48	10.78
MLPdyn + caVAE	1.92	1.92	14.18	14.97	12.12	12.14

@R2, R3 - **Compare with prior work HNN [6], HGN [Toth'20]:** The main contribution of HNN is learning Hamiltonian dynamics from low dimensional data. Although they proposed PixelHNN to learn from pendulum images, the angle of the pendulum in the training data is constrained to be in $[-\pi/6, \pi/6]$. From our experiments, PixelHNN does not work for pendulum with large angles and does not generalize to CartPole and Acrobot. [Toth'20] also has a similar observation about PixelHNN. We thank R2 and R3 for pointing to HGN [Toth'20], but we'd like to point out three differences. 1) Control. Both HGN and HNN learn energy-conserved dynamics and do not learn control. It is not clear how to incorporate control into HGN. 2) Interpretability. In HGN's pendulum task, the dimension of q is $4 \times 4 \times 16 = 256$. With such a high dimension (for various tasks), HGN does not assume the degree of freedom is given, but it might not be easy to interpret the learned q, while ours are interpretable. 3) Data efficiency. HGN uses $30 \times 50K = 1.5M$ training images for the pendulum task, while we use $20 \times 256 = 5120$ training images (with zero control). We may have a huge advantage on data efficiency. Thus, we disagree with R2 that our work is *"solving ... problem for Lagrangian rather than Hamiltonian ..."*. However, we agree with R2 and R3 that we should acknowledge [Toth'20] and we will update inaccurate statements about prior works. We are working on implementing HGN and we will report the comparison with HGN in terms of pixel MSE, interpretability and data efficiency in the camera-ready.

@R4 - **Compare with Kalman VAE (KVAE):** Thanks for pointing to KVAE. We are not convinced about *"(KVAE)... performs well in pendulum control"*(R4). Although KVAE models control inputs, there is neither controller design nor control results reported in the paper. No pendulum data or pendulum prediction sequences are provided in their Github repo and project website. However, we will investigate KVAE trained with our own pendulum data in camera-ready.

@R4 - **Discuss neural motion prediction and control from images:** We will add the discussion in camera-ready.

@R1 - **Visual difference in Fig. 3:** Our dynamical model enforces energy conservation (Thm 1 in SM), so the learned energy will not drift away from the real constant energy but will oscillate around it. This oscillation explains the visual difference. This oscillation also shows up in prior works. Please see the oscillations in [6] (Fig. 2) and [7] (Fig. 6).

@R2 - **Clarity and exposition:** We presented self-contained preliminary concepts and methods that have been used in the prior works in Sec. 2 to prevent a hard-to-follow Sec. 3, which already takes four pages. R3 and R4 say our paper is *"a pleasure to read"* and *"easy to follow"*, respectively. We provided a short overview of Sec. 3 in L112-120.

@R2 - **Training with constant control:** We didn't elaborate since it has been explained in a prior work [7] (Sec. 3.1.)

@R2 - **Time-varying mass matrix and input matrix:** In general, the *mass matrix* is not constant even if each rigid body has a constant mass, e.g., an Acrobot. Please see the appendix of Sutton'96 for the dependence of mass matrix on the angle of the Acrobot. In general, the input matrix is also non-constant. The dependency on coordinates is necessary. [Sutton'96] Generalization in Reinforcement Learning: Successful Examples Using Sparse Coarse Coding.

@R3 - **Assumptions in one place:** We will summarize and discuss all assumptions in the appendix in camera-ready.

@R3 - **Pros in applied settings:** We imagine this new technique would benefit a robot equipped with camera sensors to learn to predict and control other systems (robots), because of the interpretability and data efficiency.