We thank the reviewers and are glad that they find our work theoretically sound [R2, R3], well-written [R1, R2, R3, R4] and empirically impressive [R1, R3]. R2 says "the paper is theoretically well grounded, and it is especially strong in its integration of deep learning and control theory" and R4 finds it "important for combining high-frequency control with low-frequency inference when using high-dimensional inputs". R1, R4 suggested additional imitation learning experiments and ablations. *We are pleased to report that we have completed all those experiments*. Due to limited space, we can't provide all answers and result plots but promise to include them in the camera-ready which allows 9 pages.

[**R1**, **R4**] ***"lack of imitation learning experiments... in robotic manipulation domains"***: Exactly as R1 advised, we train PPO-based experts on robotic manipulation tasks from our paper and use it to collect data which is then used for learning NDPs and a baseline neural network policy via behavior cloning. Success rates in Table 1 show superior performance for NDPs.

Method	NN	**NDP (ours)**
Throw	0.528 ± 0.262	$\mathbf{0.642 \pm 0.246}$
Push	0.002 ± 0.004	$\mathbf{0.208 \pm 0.049}$
Soccer	0.885 ± 0.016	$\mathbf{0.890 \pm 0.010}$
Faucet	0.532 ± 0.231	$\mathbf{0.790 \pm 0.059}$

Table 1: Imitation learning on robotic tasks. Success rate is out of 1.

[**R1**]***"a fairer comparison could be provided by running VICES at a lower policy frequency"***: As suggested, we ran another version of VICES where the higher level policy runs at similar frequency as NDP. Results indicate that *NDP still outperforms it by approx. 75%*. We can't include the plots here due to space constraint but promise to include them in the camera-ready.

[**R1**] ***"ablation results for... the Throwing task.", "is it useful to also learn alpha and beta in eqn 4?", "Does it suffice to just learn g and set the forcing term to*** 0***?"***: We have finished all these three set of experiments. Firstly, upon running similar ablations as pushing for throw task too, we found that NDPs show similar robustness across all variations. Secondly, we ran ablations with the forcing term set to 0 and found it variant to be significantly less sample efficient than NDPs while converging to a slightly lower asymptotic performance. Finally, we ran ablation where α is also learned by the policy while setting $\beta = \frac{\alpha}{4}$ for critical damping. We found it to be less sample efficient and have a lower performance than NDP. We will include all plots corresponding to all these settings in the final version.

[**R1**, **R2**] ***"The description for PPO... needs to be made more clear", "clarify the way... the proposed approach [is used] for reinforcement learning", "no [explicit loss] term is presented for the policy gradient"***: We will update the final version to include a separate paragraph about our modifications to PPO and it's objective function. We will also clarify in NDP-RL algorithm block that M integration steps are used in each of the k rollout steps. Thank you.

[**R2**] ***"if the output is an end-effector position (eg a waypoint), dynamics can be integrated in a natural way."***: This is a good point. In fact, the approach described in VICES creates waypoints (e.g. a cubic spline) in end-effector or joint space. To get natural or smooth outputs, such methods often require tuning (for every task), or employ high dimensional actions (such as VICES). NDPs are less susceptible to these issues because the final outputs are low level actions.

[**R3**] ***"DMPs have been used... for almost 20 years...Stefan Schaal's group. connect also to the large corpus of work by Aude Billard"***: Indeed, we are inspired by the large body of work on DMPs from Stefan Schaal's, Jan Peter's and Aude Billard's group. We believe our work is complimentary as it bridges the gap between recent advances in end-to-end deep RL and seminal work on dynamical systems. Unlike our end-to-end architecture, most prior works use either a single DMP to represent the whole trajectory or the trajectory is manually segmented to learn different DMPs. In the current version of our manuscript, we already start the introduction by mentioning these seminal papers. To further pay due respect to the two decades of literature, as suggested by R3, we will expand the discussion of these papers and reorganize accordingly.

[**R3**] ***Differences from Pahic et al.***: Pahic et al. only tackle imitation learning from demonstrations. This is because they use only one DMP to represent the whole task trajectory. In contrast, our NDP framework can output a new dynamical system for each timestep to fit diverse trajectory behaviours over time. Hence, NDP can easily be incorporated not just in imitation learning but also end-to-end deep RL setup. We do not claim that DMP formalism itself is our contribution (also agreed by R1,R2,R3) and will further make it absolutely clear in Section 3.2 where derivatives are discussed. We also provided a detailed proof in the appendix by showing a recurrence relationship between partial derivatives.

[**R3**] ***"I would explain [baselines] deeper (with pros and cons)..."***: This is a great suggestion! We will update final version to include a discussion on the action dimensionality & impedance controllers used in VICES, as well as a discussion of DynE's dimensionality reduction, temporally abstract actions and its tendencies to overfit to passive data.

[**R3**] ***"I am inclined to increase my score if the weaknesses are fixed"***: We sincerely hope the concerns are addressed.

[**R4**] ***Imitation learning experiments***: Please see first answer on top.

[**R4**] ***"It would be interesting to see more real robot experiments"***: We are indeed planning to conduct real robot experiments for throwing and other tasks. However, due to COVID-19, we did not have access to the lab and were unable to run real robots prior to the deadline. We hope to have robot results ready before the camera-ready deadline.