We thank the reviewers for their helpful feedback. The reviewers found our "demonstrations on a real robot interesting"(R1); approach "novel and sound" and "backed by solid empirical results"(R2); and "bridging fields"(R3). The reviewers R1 and R3 suggested additional experiments. We are pleased to report that we have completed those experiments. We report those results and address other concerns below.

[R1] "running robots in a modular fashion is the defacto choice... negligible minority of roboticists attempting to learn robotic tasks in an end-to-end... low-level controllers in robotics are always task-agnostic... do not see any novelty in the concept of decoupling what and how": We tackle the generalized setting of learning from *third-person demonstrations from raw sensory data*. At test-time, our agent first observes a video of a human demonstrating the task in front of it, and then it performs the task by itself. To the best of our knowledge, such general setup of manipulating novel objects from raw-sensory data with 3rd person demos (and not kinesthetic trajectories) is not yet defacto for imitation learning.

Although, traditional robotics approaches employ modularity in terms of planning and control, but those controllers are based x,y coordinates or joint angle positions as input in a fully observable environment. Hence, they are difficult to generalize to unseen objects/orientations (as also noted by R3). In contrast, our low-level controller is more like a policy and *learn* from raw high-dimensional images which allows generalization to novel objects/configurations.

The only prior work that tackles third-person imitation from raw sensory observation without any handcrafted features is DAML [24] and we already compared our proposed approach to it. Since, the reviewer has not provided references, it is extremely difficult for us to argue or provide additional comparisons other than already in the paper.

[R1] "...literature on hierarchical reinforcement learning": Current RL approaches from pixels (not state vectors) usually take millions of steps in simulation [Minh et.al. 2016] and are too sample inefficient to be scalable to complex robotics scenarios. In contrast, our supervised learning approach is trained via maximum likelihood, and thus, efficient enough to scale to real robots.

[R1] "Can the low-level controller perform tasks that it has not been trained on?": Yes, indeed. Section 5.3, Table 3 shows that our learned controllers generalize to unseen tasks at test time significantly better than the baselines.

[R1] "emphasis should not be running experiments on a real robot, but performing rigorous experiments, even on a simulator, to validate the model and analyze the sensitivity": We respectfully disagree on the opinion for the emphasis not being real robots. Firstly, manipulation tasks involve intricacies like fine-grained touch, slipping, real objects etc which are still very difficult to simulate and major challenge for imitation learning. Secondly, real robots need the algorithm to be extremely sample efficient to be applicable.

Upon R1's suggestion, we setup a simulation environment where we transfer results from a Baxter robot demonstrations to a Sawyer robot. We inverse model trained on this simulation data. With respect to different object locations, our learned controller achieves mean RMSE of 6.09 with stderr of 2.8, which suggests the robustness of the controller.

[R1] "... predicting images over latent features": Our approach is agnostic to features/images, and our contribution is orthogonal to the design of observation space. Although, VAE-like methods can learn good appearance based embeddings, but learning an embedding that respects fine-grained displacements while capturing the task-oriented cues is an open research problem. One way could be to learn such an embedding via inverse model, but then it would become specific to the training task and prevent the modular decoupling. Hence, we opted for prediction in the image space.

[R1] "... thoughts on using DiscoGAN... and similar generators": DiscoGAN is very similar to CycleGAN baseline shown in Table 1. Upon suggestion, we also tried on flow-based models [Zhou et.al. 2016] which did not perform well and were unable to account for in plane rotations that tasks like pouring require. The performance is (L1: 127.28, SSIM: 0.81) significantly worse than our method in Table 1.

[R2] "... open-source code": We will release our code and data publicly with the paper.

[R2] "whether trained models have to be trained on very tightly coupled data or not; robot same as [22]?": We used a Baxter Robot for our experiments (as also employed in [22]). However, we re-calibrated the robot from scratch and tested with different objects. While similarity in distribution is always helpful, the modular decoupling is also reasonably effective when the data is not tightly coupled.

[R3] "Add comparisons with feature-based models... with methods based on trajectory": Upon reviewer's suggestion, we ran two baseline comparisons. (1) Trajectory-based features: Given a human demo at test time, we find feature-based nearest neighbor human demonstrations from training set and replay their corresponding joint angles. The feature used for matching are state-of-the-art temporal deep visual features trained on video action datasets (Non-local Neural Networks - [Wang et.al. 2018]); performance = [rMSE: 22.20, stderr: 2.14]. (2) Static feature based model: Align human demonstration frames with robot ones in SIFT space and find nearest neighbors from training; performance = [rMSE: 45.32, stderr: 6.12]. Both the baselines perform significantly worse than our results shown in Table 2. In particular, SIFT features didn't perform well in finding correspondences between the human and robot demonstrations because of the large domain gap. We will try ablations with more feature descriptors for the final version.

[R3] "Add discussion about limitations of the approach": This is a great suggestion and we will include a section on this. Some limitations include (a) Temporal continuity is not used in the trajectory prediction; (b) Our inverse model can be trained via self-supervised data, but goal-generator needs demonstration. (c) Goal-generator is not task-agnostic.