We thank all reviewers for their insightful comments. Please see the responses below.

To Review 1:

Q1: The connection between the policy and the Hindsight Inverse Dynamics(HID). Instead of mapping $(s_1, g_1) \times (s_2, g_2) \to a$ in vanilla inverse dynamics, the HID encodes (s_2, g_2) to replace g_1, i.e. $g_1' = m(s_2)$. So that both HID and the policy maps $(S \times G) \to a$. (e.g. $(s_1, g_1') \to a$ for an 1-step example). g_2 is omitted as the original goals are not concerned in HID.

Q2: Why is it important to relabel data to learn HID? First, the relabel process enables the HID and policy to be parameterized in the same form, as depicted in **Q1**. Second, with such a relabeling process, multistep HID can be introduced, on the contrary the vanilla ID can only deal with adjacent transitions.

Q3: The combination of PCHID with PPO. Yes, in principle the PCHID should work better when it is combined with off-policy algorithms where the trained samples are collected with off-policy hindsight experiences. The encouraging result of combing PCHID with PPO in our paper illustrates that PCHID is a promising approach to introduce hindsight experience knowledge into the prevailing on-policy algorithms like PPO, enabling on-policy algorithms to learn from failures (resolved a challenge proposed in [3]). As PCHID needs the policy network to fit HID, such procedure can narrow the gap between the on-policy experiences generated by the policy and the HID.

Q4: k-step solvability in continuous action domains. Ideally the k-step solvability means the number of steps it should take from s to g, given the maximum permitted action value. In practice the k-step solvability is a evolving concept that can gradually change during the learning process, thus is defined as "whether it can be solve with π_{k-1} within k steps after the convergence of π_{k-1} trained on (k-1)-step HIDs".

To Review 2:

Q1: The choice of maximum K. We attribute the success of 1-step HID to the flatness of the state space, and under this circumstance extrapolation of 1-step policy works well in multistep situations. Fig.1(b) in the paper shows an analogy of how such a flatness benefits extrapolation: an agent in a grid map is asked to reach the goal g_3 starting from s_0: if the agent has already known how to reach s_1 in the east, intuitively, it is not difficult for it to extrapolate the policy to reach g_3 in the farther east. On the other hand, when the goal is at g_1, the barrier makes the extrapolation of 1-step policy pointing to the north fails to reach the goal. And multistep HIDs help such extrapolations in non-trivial cases. In principle, with larger K, the successful rate of extrapolation will increase. In the GridWorld environment, our ablation study (Fig.1(a) below) shows $K = 4$ is able to achieve good performance. And Fig.1(b) below shows similar results in the FetchPush environment.

To Review 3:

Q1: Dynamic Programming(DP) perspective of PCHID. PCHID can be formulated as a solver from the perspective of DP. For most goal-oriented tasks, the learning objective is to find a policy to reach the goal as soon as possible. In such circumstances, $L^\pi(s_t, g) = L^\pi(s_{t+1}, g) + 1$, where $L^\pi(s, g)$ is defined as the number of steps needed from s to g with policy π and 1 is the additional step. PC can be interpreted as a learning procedure for a solver and HIDs are the corresponding data generated to train the solver. We will detail the DP perspective in the revision. As VIN can be seen as a neural-network approximation of dynamic programming(VI) [30]. The DQN based on VIN can be regarded as a simple DP baseline (Fig.3(a) in the paper).

Q2: Discussion about the False Negatives in the TEST process. In principle, if sufficient exploration and the robustness of neural network approximators are guaranteed, we will have an optimal (k-1)-step sub policy before we use it to TEST on k-step transitions, hence false negatives do not exist. In practice, false negatives do provide some useful information on what the agent has not yet mastered. Although the agent is not guaranteed to learn optimal policy from HIDs with false negatives, it can still learn to find a feasible path. In sparse reward settings, learning a feasible policy is crucial and several previous work like learning from demonstrations [4], self-imitation learning [6] can be absorbed for further improvement.

Q3: Order of article arrangement. Thank you for the advice. We will rearrange Sec.4.3 in the revision.

Q4: More diverse experiments. We evaluate PCHID on the FetchPush environment. The results are shown in Fig.1(b) below. We also test on the choice of maximum K. The multistep PCHID performs much better than 1-step PCHID as the FetchPush is a multistep task, i.e. an agent needs to first move the gripper to the block and then push it to the target position. We will evaluate PCHID on more benchmarks in the future work.

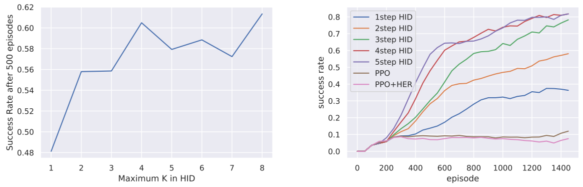

Figure 1: (a): on the selection of maximum K. The curve shows averaged results in 5 experiments with different random seeds. (b): experiments on FetchPush environment. The curve shows averaged results in 3 experiments with different random seeds