We thank the reviewers for valuable and insightful feedback. The reviewers note that the method is novel, interesting, and relevant to the field. To address the reviewers' concerns about the training data, we provide additional experiments with completely random data collection policy and find that our method improves performance over the data collection policy. We also perform preliminary experiments with robotic manipulation. We will update the manuscript with these experiments, as well as other suggestions, as detailed below.

R4: *"Requires examples of successful trajectories", Can the method "find shorter paths than the demonstrations?"*
To reach a particular goal, our method requires training trajectories that reach goals from the same distribution. Note that we still test on *unseen* goals. This is a common requirement for visual planning and control (Ebert'18 [10], Pathak'18 [44]). These trajectories are collected with a suitable exploration policy that need not be optimal but should cover a wide enough trajectory distribution. To test whether our method can work with very suboptimal training data, we conducted a new experiment with completely random exploration data, and observe that our method still successfully solves navigation tasks in the 9-room environment (see Fig 1). This leads us to believe that the proposed method is scalable even to situations where no good planners exist that can be used for data collection. We will include a full evaluation of training with random action data in the final version. In Tab. 1, we compare the average trajectory length of training data and our method on both, the dataset from the original submission and the random action data. We find that planning with our method leads to substantially shorter trajectories.

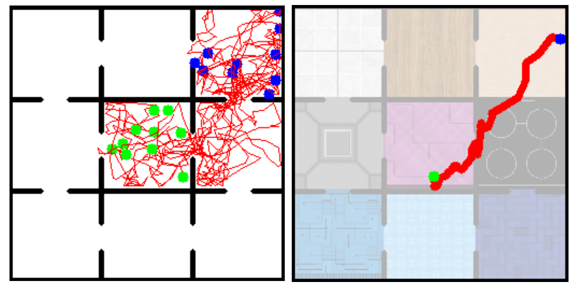

Figure 1: **Left**: random exploration data. **Right**: execution of our method trained on random data.

R3: *What are the differences with Sub-goal trees [26]?*
[26] employs a stochastic dynamic programming approach for learning to predict trajectories with low cost using a hierarchical predictive model. In contrast, we employ a sampling-based planning approach that hierarchically optimizes the latent variables of our stochastic prediction model *at decision time*. Such decision-time planning allows for greater flexibility, e.g. by changing the cost function post hoc, after training the model. Crucially, the use of *latent variables* allows our model to scale to modeling image sequences and we demonstrate its applicability to long-horizon *visual* control tasks from raw pixel inputs, while [26] only apply their method to low-dimensional state-based tasks. Finally, we propose an adaptive binding scheme for non-balanced subgoal splits that can discover bottleneck states.

Table 1: Average Trajectory Length. Planning with GCP finds shorter paths than the training distribution.

	Original Data	Random Data
Training Data	31.4	62.6
GCP-Tree (ours)	**20.7**	**42.6**

R3, R4: *Application to new environment / Pick & Place task*
We have now performed preliminary evaluations of GCP-tree in a state-based robotic pick & place environment. Our approach performs long-horizon object manipulations like lifting blocks over a barrier and stacking them (see Fig 2). We will add a full quantitative evaluation with comparisons in the final paper.

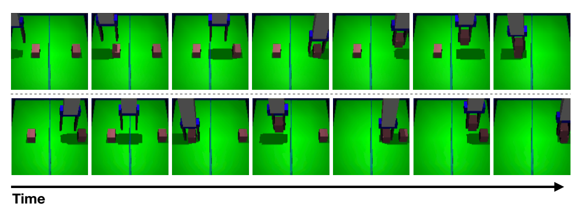

Figure 2: Executions of GCP-Tree on a pick & place task with wall separator (subsampled for visibility).

R1: *"in larger problems [...] optimisation over z infeasible."*
Planning over latent states has similar properties to planning over images, but is more scalable as the latent states are compact. We show that optimizing over z substantially improves the plans over the training data (see Fig. 1, Tab. 1, response the R4 on top). The optimization is indeed harder with longer sequences, but our goal-conditioned prediction and hierarchical planning enable us to optimize well even where prior work fails (e.g. over 200 steps in Fig. 1).

R4: *"The algorithm [...] commits itself to a bad subgoal and has no way of recovering from this choice."*
It is quite possible to maintain multiple potential waypoints in parallel, analogously to a beam search. We found this to not be necessary for our tasks, and our method attains substantially better results than prior methods without a beam search, but we will discuss this as a promising topic for future work.

R4: *"the agent is tasked to reach the goal on the shortest path. Is this reflected in the success rate?"*
No, but it is reflected in the trajectory cost, which we also report in Table 4 (see L283).

R3: *"How does the adaptive binding affect the performance"*
Adaptive binding usually performs comparable or slightly worse due to harder optimization. We expect the benefits of adaptive binding to become more clear with better optimization or where semantic bottlenecks are important.

R5: *"Is there always a set of T observations for each dataset"*
Our datasets contain variable-length sequences. In order to determine where to stop hierarchical generation, we use a learned termination classifier at each node. We will add this explanation to the supplement.