Point Cloud Models Improve Visual Robustness in Robotic Learners

Skand Peri [1] Iain Lee [2] Chanho Kim [1] Li Fuxin [1] Tucker Hermans [2,3] Stefan Lee [1]

***Abstract*—Visual control policies can encounter significant performance degradation when visual conditions like lighting or camera position differ from those seen during training – often exhibiting sharp declines in capability even for minor differences. In this work, we examine robustness to a suite of these types of visual changes for RGB-D and point cloud based visual control policies. To perform these experiments on both model-free and model-based reinforcement learners, we introduce a novel Point Cloud World Model (PCWM) and point cloud based control policies. Our experiments show that policies that explicitly encode point clouds are significantly more robust than their RGB-D counterparts. Further, we find our proposed PCWM significantly outperforms prior works in terms of sample efficiency during training. Taken together, these results suggest reasoning about the 3D scene through point clouds can improve performance, reduce learning time, and increase robustness for robotic learners.**

This work has also been accepted to ICRA 2024.

***Index Terms*—point cloud world model, model-based RL, vision-based robot control, robustness**

I. Introduction

To broaden the deployment of robot manipulators in the world, we must extend their understanding of and ability to operate in unstructured environments [1]. However, the dynamics of such environments contain significant uncertainty. Furthermore, robots can typically only sense these environments through partial observations. Modeling every aspect of the world "in the wild" is thus intractable. Owing to this, planning under such situations can be prohibitively expensive especially in unseen scenes when novel objects are introduced. Hence, to endow manipulators to act in complex scenarios with only partial view sensing information, recent works have relied on learning based robotic control [2]–[4].

However, learning-based robot control policies that rely on imagery as input can exhibit significant performance degradations when visual conditions like lighting, camera position, or object textures differ from those seen during training [5]. This lack of robustness is a hurdle for in-the-wild deployment and has prompted the extensive study of data augmentation [6]–[8] and pretraining [3], [9] techniques in visual policy learning. In this work, we examine the question of policy robustness from the perspective of visual input representation – finding policies that encode observations as XYZ-RGB point clouds rather than RGB-D images demonstrate greater robustness.

To illustrate this phenomenon, we examine a simple control task in Fig. 1 where a robotic arm must lift a red cube from a table up to a green goal point. We trained state-of-the-art model-based reinforcement learning (RL) policies [10] for this task with RGB-D input (denoted DreamerV3) and point cloud input (denoted PCWM). When tested on novel viewpoints, we observe that the success rate of DreamerV3 drops by half even for a slight variation and fails completely on more significant changes. In contrast, the PCWM's performance decays much more slowly. We find similar trends in our larger suite of experiments later in this paper. At first glance, the reasons for this are unclear. XYZ-RGB point clouds and RGB-D images contain much of the same information. How then can we account for such a difference between these policies?

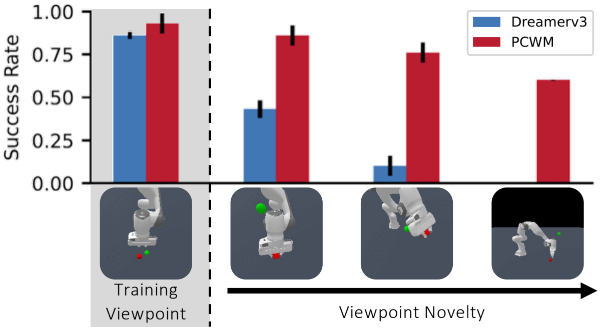

Fig. 1. *Motivating Example.* We compare **DreamerV3**, a state-of-the-art RL model that is trained on RGB-D inputs with our **Point Cloud World Model (PCWM)** on a simple task of lifting a cube. We find the point clouds are significantly more robust to viewpoint changes compared to RGB-D.

We hypothesize this difference comes from *how* these modalities are typically encoded. Generally, RGB-D inputs are encoded with CNNs – simply treating the depth information as a fourth image channel. The convolutional kernels aggregate features based on closeness in the 2D pixel space, even when neighboring pixels may have vastly different. This could lead to features between far away objects being averaged together, leading to worse performance. In contrast, point cloud representations allow the XYZ coordinates to directly serve as features, enabling networks to learn geometric equivariances with respect to rotation and scaling [11].

Contributions. We summarize our main contributions:

- We study robustness to changes in viewpoint, field-of-view, lighting, and distractor objects for RGB-D and point cloud-based visual control policies.
- We propose Point Cloud World Models (PCWMs), a model-based reinforcement learning framework based on partial point clouds. We show gains in sample efficiency and robustness over comparable RGB-D models.
- Beyond increased robustness, we show PCWMs adapt more

[1]Oregon State University [2]University of Utah [3] NVIDIA

quickly when finetuned in new environments with significant differences in visual conditions.

II. Point Cloud World Models

Problem Formulation. We pose our problem as an infinite-horizon Partially Observable Markov Decision Process (POMDP) [12] defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \mathcal{O}, \gamma, \rho_0)$. $\mathcal{S}$ represents the state space with a complete scene point cloud, which is not accessible to the agent. The observation space, $\mathcal{O} \in \mathbb{R}^{N \times 6}$ denotes partial point cloud observations with N points featurized with position (x, y, z) and color (r, g, b). $\mathcal{A} \in \mathbb{R}^m$ is an m-dim continuous action space, $\mathcal{T} : \mathcal{O} \times \mathcal{A} \rightarrow \mathcal{O}$ is the transition function, $\mathcal{R} : \mathcal{O} \rightarrow \mathbb{R}$ is the reward function, $\gamma \in [0, 1)$ is the discount factor and ρ_0 denotes the initial state distribution. The goal of the agent is to learn a policy $\pi : \mathcal{O} \rightarrow \mathcal{A}$ that maximizes the expected sum of discounted rewards; $\max_\pi \mathbb{E}_\pi[\sum_{t=1}^{\infty} \gamma^t \mathcal{R}(s_t)]$.

A. World Model

We base our world model on the Recurrent State-Space Model (RSSM) framework [13] which learns a recurrent world model with a d-dim latent variable z. The RSSM model is derived from an evidence lower bound on the likelihood of an observation sequence $o_{1:T}$ given actions $a_{1:T}$. This results in a loss composed of two components: a reconstruction term measuring how well observations (and rewards) can be predicted from the latent representation and a KL divergence term keeping predicted latent states near corresponding real observation encodings.

For point cloud observations, designing the reconstruction task is non-trivial – irregular point densities may bias the loss function to denser regions and jointly predicting the structure and featurization of a point cloud from a latent vector is challenging. To sidestep these issues, we follow [14], [15] by dropping observation reconstruction and relying only on multi-step reward prediction and the KL term for supervision.

More concretely, our world model shown in Fig. 4 is parameterized by ϕ and consists of the following components:

$$\begin{aligned}
&\text{Representation:} && z_t \sim q_\phi(h_t, o_t) \\
&\text{Recurrent Model:} && h_t = f_\phi(z_{t-1}, h_{t-1}, a_{t-1}) \\
&\text{Dynamics:} && \hat{z}_t \sim p_\phi(\hat{z}_t | h_t) \\
&\text{Reward:} && \hat{r}_t \sim p_\phi(\hat{r}_t | h_t, z_t) \\
&\text{Continuation Predictor:} && \hat{c}_t \sim p_\phi(\hat{c}_t | h_t, z_t)
\end{aligned} \tag{1}$$

where we use $\sim$ to denote the sampling operation. The continuation flag $c_t \in \{0, 1\}$ indicates whether the episode has ended. Except for the input encoder network within q_ϕ, we retain architectural choices from [10] for the other components.

Given a partial point cloud $o_t \in \mathbb{R}^{N \times 6}$ with N points, we encode it using a series of PointConv layers [16] to obtain an embedding $e_t \in \mathbb{R}^{n \times d}$, where n is the number of points in the downsampled point cloud with each point consisting of a d-dim feature. We then aggregate the features of n points in the point cloud latent space to obtain the d-dim feature $\mathtt{agg}(e_t)$, where $\mathtt{agg}$ is an aggregation function that is used to predict the latent z.

Like TD-MPC [14] and VPN [15], we find that supervising rewards by rolling out each latent in the future for H steps helps learn a better world model and leads to better performance. Along with this, we simultaneously train the continuation predictor c_t using binary cross entropy loss. Since z_t is predicted using the input point cloud (o_t) and for dynamics model rollouts we do not have access to o_t, we employ a KL loss term to ensure that posterior prediction, z_t and prior prediction $\hat{z}_t$ are close. Hence, this way, $\hat{z}_t$ can be used for accurate rollouts to train the policy. The overall world model training objective can thus be written as follows

$$\begin{aligned}
L_\phi = \sum_{t=1}^{T} \overbrace{p_\phi(r_t, c_t \mid h_t, z_t)}^{\text{current-step loss}} + \overbrace{\left(\sum_{i=1}^{H} p_\phi(\hat{r}_{t+i} \mid h_{t+i}, \hat{z}_{t+i})\right)}^{\text{multi-step reward loss}} \\
+ \underbrace{\mathrm{KL}\left(\, q_\phi(z_t \mid h_t, o_t) \;||\; p_\phi(\hat{z}_t \mid h_t)\, \right)}_{\text{one-step temporal consistency}}
\end{aligned} \tag{2}$$

B. Policy Learning

For the policy, we adopt the Actor-Critic framework [17] similar to DreamerV3 [10], which consists of a *Critic* network that predicts the value at a given state and an *Actor* that predicts the action distribution given a state.

$$\begin{aligned}
&\text{Actor network:} && a_t \sim \pi_\psi(a_t | z_t) \\
&\text{Critic network:} && v_\psi(z_t) \approx \mathbb{E}_{q(.|z_t)}[\textstyle\sum_{\tau=t}^{t+H}(\gamma^{\tau-t} r_\tau)]
\end{aligned} \tag{3}$$

The critic is learned by discrete regression [10], [18] using generalized λ-targets [19]. We train the actor network to maximize the value function via dynamics backpropagation [20], updating actor parameters using the gradients computed through the world model.

III. Experimental Setup

Environments. We conduct our experiments on the ManiSkill2 [21] benchmark on a simulated 7-DoF Franka Panda robotic arm with a parallel gripper. We consider two representative manipulation tasks – (i) *Pick & Place* and (ii) *Mobile Manipulation*. For details of the environments, please refer to App. VII-A.

Baselines. Beyond PCWMs, we consider representative methods for the other three {model-based, model-free} × {RGB-D, point cloud (PC)} settings. For model-based RGB-D, we modify a stable PyTorch implementation of DreamerV3 [22] to include depth reconstruction and denote this model as **RGBD-WM**. For PC and RGB-D model-free policies, we take policy architectures and representation encoders from our corresponding model-based approaches and train them directly from real environment experience using PPO [23] – denoting these models as **PC-PPO** and **RGBD-PPO** respectively. For implementation details, refer to App. VII-B.

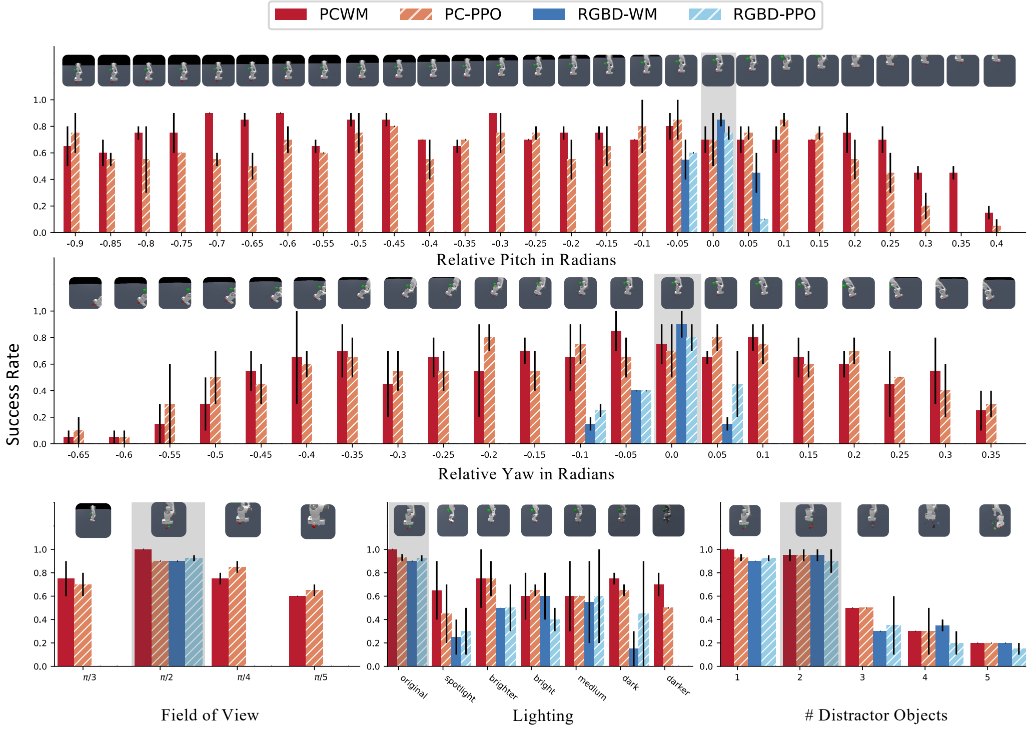

Fig. 2. Fine-grained robustness analysis for the `Clutter Pick` task. Example frames from each condition are shown above policy performance plots. Grey shaded backgrounds indicate the original training environment. We find RGB-D models generalize poorly to new viewpoints or FoV in this setting.

IV. Results

This section is divided into the following claims and supporting evidence from our experimental results.

A. Point Cloud World Models (PCWMs) can be more sample efficient learners than analogous RGB-D models.

We show reward curves over the course of training for our six tasks in Fig. 3. In all tasks, the proposed `PCWM` matches or exceeds the performance of the baseline methods – including the model-based RGBD-WM [24]. Strikingly, `PCWM`s learn considerably faster in the Pick & Place style tasks (top row). For example, on `Clutter Pick` (top middle), `PCWM` achieves task success in under 1 million interactions whereas the other methods fail to do so after 2 million. This trend is more pronounced in the `StackCube` task (top right). We attribute this gain in efficiency to `PCWM`'s ability to reason with explicit 3D representations. For model free methods, we observe that PC-PPO outperforms RGBD-PPO as well.

For `OpenCabinetDoor` and `MoveBucket`, RGBD-WM and `PCWM` achieved similar performances. We hypothesize that the model-based policy training is a dominant factor in these cases as opposed to the input representation. The mobile manipulation tasks tend to be more complex, involving navigation to the target object in all three and bimanual coordination for `MoveBucket`. While all models achieve $> 75\%$ success rates for `OpenCabinetDrawer`, we find they struggle on `OpenCabinetDoor` and `MoveBucket`, indicating the need for more interaction to achieve task success.

B. Point cloud-based policies are more robust to changes in visual conditions than analogous RGB-D policies.

The models in the previous discussion were all trained in *single, fixed imaging conditions* – i.e. with a fixed camera viewpoint, field of view, and scene lighting. Here, we examine their performance when these conditions are systematically varied. Note this set of experiments does not involve any further policy training. Below we describe these variations:

- *Viewpoint.* We alter either camera pitch or yaw by 0.05 radian increments through ranges that keep the task objects and manipulators in frame. We select -0.9 to 0.4 radians for yaw and -0.65 to 0.35 for pitch for a total of 42 conditions.
- *Field of View.* We vary the field of view of the camera at three discrete levels $\{\frac{\pi}{2}, \frac{\pi}{4}, \frac{\pi}{5}\}$ yielding 3 conditions.
- *Lighting.* We consider 6 lighting conditions – varying ambient illumination through 5 stages from bright to dark and adding a yellow spotlight focused on the table.

These conditions are visualized in Fig. 2 for the `Clutter Pick` task and average rewards across these conditions for all tasks are shown in Tab. I for `PCWM` and RGBD-WM. We take the model with best return to compute the results across 3 different seeds. Given their lower overall performance, we do not include the model-free methods in this comparison.

Across settings, we find `PCWM` policies achieve significantly better performance than those from RGBD-WM. However, in many tasks this difference in performance was also evident in the original unperturbed setting due to `PCWM`'s increased sample efficiency. Focusing on the `LiftCube` setting where both methods achieve similar task performance in the original environment, we still observe significant differences in performance in perturbed conditions. For example, `PCWM` achieves only 6% less average reward across viewpoint changes compared to a **76% reduction** for RGBD-WM.

For a fine grained analysis on robustness of point cloud models, refer to App. VII-C.

V. Conclusion

In this work, we presented a novel model-based RL method for partial point cloud observations `PCWM`. We demonstrated that this model can result in dramatic sample efficiency on certain tasks, and significant robustness gains over analogous RGB-D models in settings such as viewpoint, field of view and lighting changes. We also showed that the choice of the point cloud network significantly impacts sample efficiency.

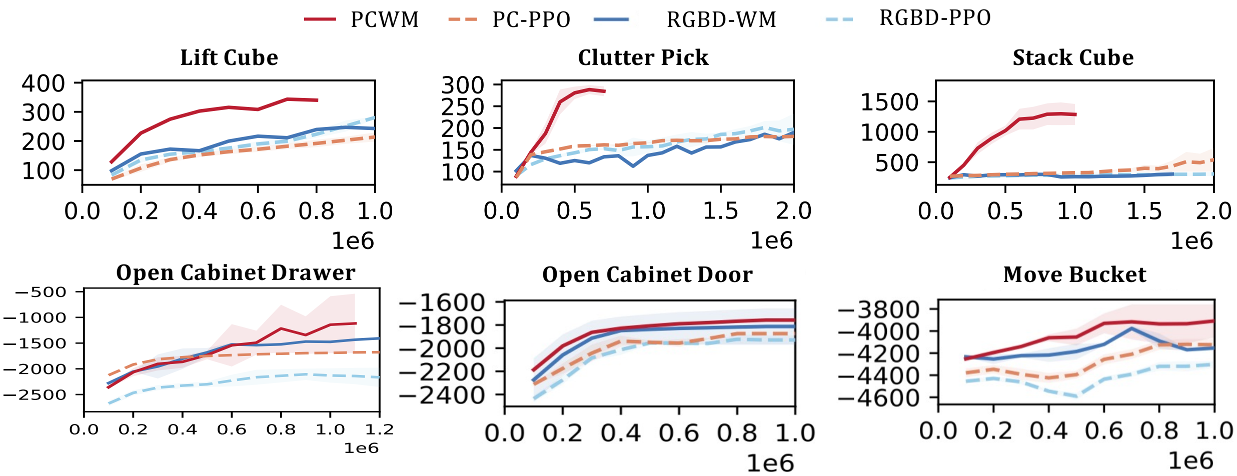

Fig. 3. *Task performance:* We report training curves for six manipulation tasks. Our proposed `PCWM` either matches or outperforms baselines in all settings – demonstrating strong sample efficiency gains in several tasks. `PCWM` is truncated after achieving task success for Pick & Place tasks (top row).

VI. Acknowledgements

SP would like to thank Wesley Khademi for his help with PointConv codebase and DMV & ViRL labmates for providing feedback on an earlier version of the draft. This University of Utah effort is supported by DARPA under grant N66001-19-2-4035. The Oregon State effort is supported in part by the DARPA Machine Common Sense program and ONR awards N00014-22-1-2114, ONR N0014-21-1-2052.

References

[1] T. Bhattacharjee, G. Lee, H. Song, and S. S. Srinivasa, "Towards robotic feeding: Role of haptics in fork-based food manipulation," *IEEE Robotics and Automation Letters*, 2018. 1

[2] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *JMLR*, 2016. 1, 8

[3] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," in *Conference on Robot Learning*, 2022. 1, 7

[4] N. Hansen, Z. Yuan, Y. Ze, T. Mu, A. Rajeswaran, H. Su, H. Xu, and X. Wang, "On pre-training for visuo-motor control: Revisiting a learning-from-scratch baseline," in *International Conference on Machine Learning (ICML)*, 2023. 1

[5] A. Xie, L. Lee, T. Xiao, and C. Finn, "Decomposing the generalization gap in imitation learning for visual robotic manipulation," *ArXiv*, vol. abs/2307.03659, 2023. 1, 7

[6] D. Yarats, I. Kostrikov, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," in *International Conference on Learning Representations*, 2021. 1

[7] D. Yarats, R. Fergus, A. Lazaric, and L. Pinto, "Mastering visual continuous control: Improved data-augmented reinforcement learning," *ArXiv*, vol. abs/2107.09645, 2021. 1, 7

[8] N. Hansen, H. Su, and X. Wang, "Stabilizing deep q-learning with convnets and vision transformers under data augmentation," in *Neural Information Processing Systems*, 2021. 1

[9] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell, "Real-world robot learning with masked visual pre-training," in *Conference on Robot Learning*, pp. 416–426, PMLR, 2023. 1

[10] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse domains through world models," *arXiv preprint arXiv:2301.04104*, 2023. 1, 2, 8

[11] X. Li, W. Wu, X. Z. Fern, and L. Fuxin, "Improving the robustness of point convolution on k-nearest neighbor neighborhoods with a viewpoint-invariant coordinate transform," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1287–1297, 2023. 1

[12] A. R. Cassandra, L. P. Kaelbling, and M. L. Littman, "Acting optimally in partially observable stochastic domains," in *AAAI Conference on Artificial Intelligence*, 1994. 2

[13] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning latent dynamics for planning from pixels," in *International conference on machine learning*, 2019. 2, 8

[14] N. Hansen, X. Wang, and H. Su, "Temporal difference learning for model predictive control," in *International Conference on Machine Learning*, 2022. 2, 8

[15] J. Oh, S. Singh, and H. Lee, "Value prediction network," in *NeurIPS*, 2017. 2

[16] W. Wu, Z. Qi, and L. Fuxin, "Pointconv: Deep convolutional networks on 3d point clouds," in *Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition*, 2019. 2, 6, 7

[17] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. P. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International Conference on Machine Learning*, 2016. 2

[18] E. Imani and M. White, "Improving regression performance with distributional losses," in *International Conference on Machine Learning*, 2018. 2

[19] J. Schulman, P. Moritz, S. Levine, M. I. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *CoRR*, 2015. 2

[20] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," *arXiv preprint arXiv:1912.01603*, 2019. 2, 7, 8

[21] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su, "Maniskill2: A unified benchmark for generalizable manipulation skills," in *International Conference on Learning Representations*, 2023. 2, 6

[22] https://github.com/NM512/dreamerv3 torch, "Dreamerv3-torch," *Github*, 2023. 2

[23] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *ArXiv*, vol. abs/1707.06347, 2017. 2

[24] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering atari with discrete world models," *Internation Conference on Learning Representations*, 2021. 3, 8

[25] T. Mu, Z. Ling, F. Xiang, D. C. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su, "Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. 6

[26] Z. Ling, Y. Yao, X. Li, and H. Su, "On the efficacy of 3d point cloud reinforcement learning," *arXiv preprint arXiv:2306.06799*, 2023. 6, 7

[27] Y. Huang, A. Conkey, and T. Hermans, "Planning for Multi-Object Manipulation with Graph Neural Network Relational Classifiers," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023. 6, 8

[28] S. Elfwing, E. Uchibe, and K. Doya, "Sigmoid-weighted linear units for neural network function approximation in reinforcement learning," *Neural networks : the official journal of the International Neural Network Society*, 2017. 6

[29] J. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *ArXiv*, vol. abs/1607.06450, 2016. 6

[30] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. 6

[31] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2020. 6

[32] C. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 7

[33] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, 2015. 7

[34] H. V. Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *AAAI Conference on Artificial Intelligence*, 2015. 7

[35] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas, and N. Heess, "Reinforcement and imitation learning for diverse visuomotor skills," in *Proceedings of Robotics: Science and Systems*, 2018. 7

[36] Y. Chen, Y. Yang, T. Wu, S. Wang, X. Feng, J. Jiang, Z. Lu, S. M. McAleer, H. Dong, and S.-C. Zhu, "Towards human-level bimanual dexterous manipulation with reinforcement learning," in *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022. 7

[37] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. V. Wyk, A. Zhurkevich, B. Sundaralingam, Y. S. Narang, J.-F. Lafleche, D. Fox, and G. State, "Dextreme: Transfer of agile in-hand manipulation from simulation to reality," *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023. 7

[38] L. Kaiser, M. Babaeizadeh, P. Milos, B. Osinski, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, A. Mohiuddin, R. Sepassi, G. Tucker, and H. Michalewski, "Model-based reinforcement learning for atari," *International Conference on Learning Representations*, 2020. 7

[39] I. Singh, A. Liang, M. Shridhar, and J. Thomason, "Self-supervised 3d representation learning for robotics," in *ICRA2023 Workshop on Pretraining for Robotics (PT4R)*, 2023. 7

[40] J. Thomason, M. Shridhar, Y. Bisk, C. Paxton, and L. Zettlemoyer, "Language grounding with 3d objects," in *Conference on Robot Learning*, 2022. 7

[41] Y. Ze, N. Hansen, Y. Chen, M. Jain, and X. Wang, "Visual reinforcement learning with self-supervised 3d representations," *IEEE Robotics and Automation Letters*, 2022. 7

[42] M. Liu, X. Li, Z. Ling, Y. Li, and H. Su, "Frame Mining: a Free Lunch for Learning Robotic Manipulation from 3D Point Clouds," in *Conference on Robot Learning (CoRL)*, 2022. 7

[43] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang, "Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation," *Conference on Robot Learning (CoRL)*, 2022. 7

[44] Y. Zhu, Z. Jiang, P. Stone, and Y. Zhu, "Learning generalizable manipulation policies with object-centric 3d representations," in *Conference on Robot Learning*, 2023. 7

[45] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman, "Quantifying generalization in reinforcement learning," in *International Conference on Machine Learning*, 2018. 7

[46] X. Song, Y. Jiang, S. Tu, Y. Du, and B. Neyshabur, "Observational overfitting in reinforcement learning," *ArXiv*, vol. abs/1912.02975, 2019. 7

[47] C. Lyle, M. Rowland, W. Dabney, M. Z. Kwiatkowska, and Y. Gal, "Learning dynamics and generalization in reinforcement learning," *ArXiv*, vol. abs/2206.02126, 2022. 7

[48] R. Yang, Y. Lin, X. Ma, H. Hu, C. Zhang, and T. Zhang, "What is essential for unseen goal generalization of offline goal-conditioned rl?," *ICML*, 2023. 7

[49] J. Krantz and S. Lee, "Sim-2-sim transfer for vision-and-language navigation in continuous environments," in *European Conference on Computer Vision (ECCV)*, 2022. 7

[50] J. Krantz, T. Gervet, K. Yadav, A. Wang, C. Paxton, R. Mottaghi, D. Batra, J. Malik, S. Lee, and D. S. Chaplot, "Navigating to objects specified by images," *arXiv preprint arXiv:2304.01192*, 2023. 7

[51] I. Kostrikov, D. Yarats, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," *ArXiv*, vol. abs/2004.13649, 2020. 7

[52] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, "Improving sample efficiency in model-free reinforcement learning from images," in *AAAI Conference on Artificial Intelligence*, 2019. 7

[53] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik, "Masked visual pretraining for motor control," *ArXiv*, vol. abs/2203.06173, 2022. 7

[54] S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. K. Gupta, "The unsurprising effectiveness of pre-trained vision models for control," in *International Conference on Machine Learning*, 2022. 7

[55] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn, "Robonet: Large-scale multi-robot learning," in *Proceedings of the Conference on Robot Learning*, Proceedings of Machine Learning Research, 2020. 7

[56] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1048–1055, IEEE, 2019. 7

[57] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," *RSS*, 2022. 7

[58] R. S. Sutton, "Dyna, an integrated architecture for learning, planning, and reacting," *ACM Sigart Bulletin*, 1991. 7

[59] T. Walsh, S. Goschin, and M. Littman, "Integrating sample-based planning and model-based reinforcement learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2010. 8

[60] J. Pineau, G. Gordon, S. Thrun, *et al.*, "Point-based value iteration: An anytime algorithm for pomdps," in *Ijcai*, 2003. 8

[61] K. Chua, R. Calandra, R. McAllister, and S. Levine, "Deep reinforcement learning in a handful of trials using probabilistic dynamics models," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, 2018. 8

[62] A. Shrestha, S. Lee, P. Tadepalli, and A. Fern, "Deepaveragers: offline reinforcement learning by solving derived non-parametric mdps," *ICLR*, 2021. 8

[63] S. Rajeswar, P. Mazzaglia, T. Verbelen, A. Piché, B. Dhoedt, A. Courville, and A. Lacoste, "Mastering the unsupervised reinforcement learning benchmark from pixels," in *40th International Conference on Machine Learning*, 2023. 8

[64] N. Hansen, Y. Lin, H. Su, X. Wang, V. Kumar, and A. Rajeswaran, "Modem: Accelerating visual model-based reinforcement learning with demonstrations," in *International Conference on Learning Representations (ICLR)*, 2023. 8

[65] R. Veerapaneni, J. D. Co-Reyes, M. Chang, M. Janner, C. Finn, J. Wu, J. Tenenbaum, and S. Levine, "Entity abstraction in visual model-based reinforcement learning," in *Conference on Robot Learning*, 2020. 8

[66] D. Ha and J. Schmidhuber, "World models," *arXiv preprint arXiv:1803.10122*, 2018. 8

[67] P. Battaglia, J. B. C. Hamrick, V. Bapst, A. Sanchez, V. Zambaldi, M. Malinowski, A. Tacchetti, D. Raposo, A. Santoro, R. Faulkner, C. Gulcehre, F. Song, A. Ballard, J. Gilmer, G. E. Dahl, A. Vaswani, K. Allen, C. Nash, V. J. Langston, C. Dyer, N. Heess, D. Wierstra, P. Kohli, M. Botvinick, O. Vinyals, Y. Li, and R. Pascanu, "Relational inductive biases, deep learning, and graph networks," *arXiv*, 2018. 8

[68] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," in *ICLR*, 2019. 8

[69] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," in *ICML*, 2020. 8

[70] H. Shi, H. Xu, S. Clarke, Y. Li, and J. Wu, "Robocook: Long-horizon elasto-plastic object manipulation with diverse tools," *arXiv preprint arXiv:2306.14447*, 2023. 8

[71] H. Shi, H. Xu, Z. Huang, Y. Li, and J. Wu, "Robocraft: Learning to see, simulate, and shape elasto-plastic objects with graph networks," *arXiv preprint arXiv:2205.02909*, 2022. 8

VII. Appendix

A. Environments

For Pick & Place, we consider `LiftCube` (lifting a single cube) and `StackCube` (stacking one cube on top of another). Additionally, we add a `ClutteredLiftCube` task, where the goal of the agent is to pick the (unique) red cube among a number of distractor cubes in the scene. For these tasks, we consider an 8-d action space consisting of delta position of arm joints (7) and gripper distance (1). For Mobile Manipulation,

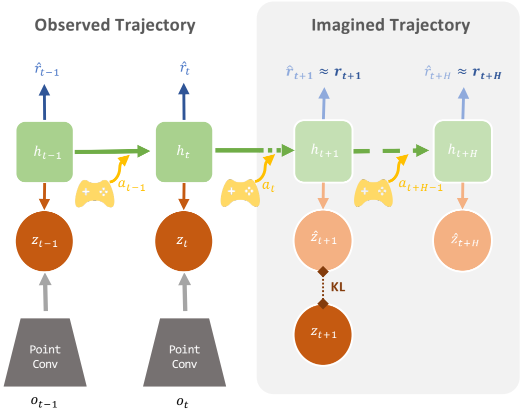

Fig. 4. **PCWM training**: Given a sequence of T partial point cloud observations $o_{1:T}$, we encode them using a PointConv encoder. For each timestep t, we compute a posterior stochastic latent z_t using an encoding of o_t and hidden state h_t that encodes the history. The hidden state is further used to compute the prior latent $\hat{z}_t$ which is used to predict multi-step rewards over a horizon H providing supervision for the world model alongside a KL-loss for temporal consistency See Sec. II-A.

we consider OpenCabinetDrawer (opening drawer of a cabinet), OpenCabinetDoor (opening a cabinet door) and MoveBucket (moving a bucket from ground to a platform situated at a certain height). For the first two tasks, the agent has a 12-d action space involving manipulation (8) and navigation (4). The latter task is bimanual, adding 8 more dimensions to control a second arm. For more details, we refer the readers to the original papers [21], [25].

B. Implementation Details

In this section, we discuss several design choices and hyperparameters of our model.

Point Cloud Encoding. Using known camera intrinsics and extrinsics, we first transform the point clouds to world coordinates. Then, we use 4 PointConv [16] layers with a downsampling factor of 2 to encode the input point cloud into $e_t \in \mathbb{R}^{n \times d}$ with n=64 and d=256. We aggregate (agg) the point cloud latent using mean pooling to obtain a 256-d representation, which we found to work well across all the tasks. See Sec. VII-E for discussion of encoder choice.

Point Pruning. Similar to [26], we found it helpful to prune distant or task-irrelevant (e.g. floor) points in the scene. This pruning could be realized in practice by depth-based clipping, background removal, or object segmentation-based filtering [27]. After pruning, we apply farthest point sampling to generate 1024 points for Pick & Place tasks and 2048 points for Mobile Manipulation tasks. The higher point resolution for the latter set of tasks is to ensure that key parts of the scene such as the door or drawer handle are represented.

World Model and Policy Training. We pretrain the world model for 1000 steps on 10 random trajectories before starting policy training. For world model training, we uniformly sample sequences of length 64 with a batch size of 8 from the replay buffer. The deterministic state h is 256 dimensional and the continuous stochastic state z is 32 dimensional. The reward and continuation prediction heads are 2 layer MLPs with sigmoid linear unit (SiLU) [28] activation and layer normalization [29]. We jointly train the multi-step reward and continuation prediction losses with $H = 5$ and the dynamics consistency (KL) loss with an Adam optimizer [30] with a learning rate of 0.0001. For policy training, we rollout using the world model for 15 timesteps from each of the 64 states of the sampled trajectory. The actor and the value heads are also 2 layer MLPs with SiLU activation and LayerNorm trained with Adam optimizer with a learning rate of $3\text{e}{-5}$.

C. Robustness results

Finer-grained Analysis. The above analysis aggregates over a range of conditions to provide a general sense of policy robustness. To examine this more closely, we take Clutter Pick as an exemplar task and examine policy robustness in each condition separately. Further, we extend the analysis to the model-free methods – PC-PPO and RGBD-PPO. To ensure all models have similar baseline competency in the original setting, we continue to train all methods beyond the steps shown in Fig. 3 until convergence. All achieve >90% task success rate. We also consider including additional distraction objects as another perturbation. Results are shown in Fig. 2. We denote point cloud methods in red and RGB-D in blue.

For viewpoint changes, we see that both RGB-D policies (RGBD-WM and RGBD-PPO) rapidly drop in success rate for minor changes. This effect results in **0% success rate** when pitch or yaw change by more than ±0.1 radians (or about 5.7°). In contrast, the point cloud-based models are robust even to extreme changes, provided the arm remains clearly in view. Changes in viewpoint do not distort object shapes or relative distance between points. We hypothesize point cloud-based policies remain performant in these instances as they learn to rely on these features rather than absolute positions or object scales.

For field of view (FoV) changes, we observe that point cloud policies suffer a minor penalty under different conditions, yet RGBD policies achieve a **0% success rate** for all perturbed settings. Changes to the FoV greatly affect the captured image – dramatically scaling the image contents as in the example frames. For RGB-D models, this results in significantly out-of-distribution inputs. For point clouds, the geometric relationships between points and their positions relative to the camera do not shift with the FoV changes.

For lighting changes, we find RGB-D methods to be impacted by irregular lighting (spotlight) or darker scenes, but respond similarly to point cloud models for other conditions. RGB networks are known to be somewhat robust to lighting changes [31], but point cloud models meet or exceed them. For additional distractor objects, we see no significant difference between point cloud and RGB-D models – suggesting the robustness exhibited by point clouds may not extend to settings where changes require that the policy performs higher-order relational reasoning with an increased number of objects.

TABLE I

AVERAGE REWARD IN ORIGINAL (GREY) AND VISUALLY PERTURBED SETTINGS COMPUTED FROM 25 EPISODES IN EACH PERTURBED CONDITION (SEE SEC. IV-B) AND STANDARD DEVIATIONS ACROSS 3 RANDOM SEEDS. PCWMS ACHIEVE HIGHER REWARD AND ARE MORE ROBUST TO VISUAL CHANGES.

	PCWM (Ours)				RGBD-WM			
Task	Original	Viewpoint	Field of View	Lighting	Original	Viewpoint	Field of View	Lighting
Lift Cube	325 ± 20	280 ± 50	257 ± 33	259 ± 31	305 ± 13	73 ± 98	112 ± 137	126 ± 11
Clutter Pick	358 ± 29	286 ± 71	246 ± 64	349 ± 27	329 ± 47	85 ± 39	30 ± 13	242 ± 98
Stack Cube	1721 ± 283	1269 ± 412	1006 ± 343	1465 ± 143	251 ± 12	193 ± 59	202 ± 23	213 ± 29
Open Cabinet Drawer	-500 ± 32	-647 ± 59	-631 ± 48	-549 ± 43	-1410 ± 29	-2460 ± 132	-1782 ± 238	-1638 ± 126
Open Cabinet Door	-1726 ± 48	-1972 ± 177	-1983 ± 56	-1794 ± 53	-1925 ± 17	-2303 ± 396	-2120 ± 194	-2120 ± 194
Move Bucket	-3632 ± 85	-3901 ± 135	-3881 ± 47	-3681 ± 29	-4168 ± 89	-4572 ± 21	-4419 ± 39	-4276 ± 55

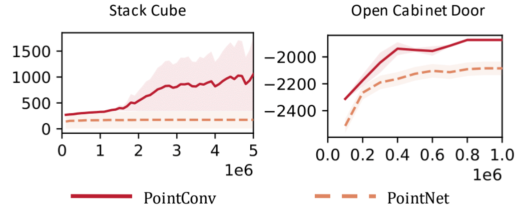

Fig. 5. *Comparison of point cloud encoders*: PointConv [16] consistently shows greater sample efficiency as compared to PointNet [32] in the StackCube and OpenCabinetDoor tasks.

D. Adaptation

PCWM adapt more quickly than RGB-D counterparts when trained further in viewpoint perturbed environments. In case of task failures on novel settings, it is desirable to have a model that can be fine-tuned as quickly as possible and not have to learn about the task from scratch. To test if PCWM can adapt well in such situations, we select 2 conditions from viewpoint (Rel. Pitch (0.4) and Rel. Yaw (-0.6) in Figure 2) and one from lighting (Medium in Figure 2) where both PCWM and RGBD-WM performed nearly equally. Starting from the pretrained models, we trained in the new environments until convergence. For viewpoint changes, we observed a significant difference in sample efficiency with PCWM requiring 32 and 25 episodic interactions compared to 94 and 70 for RGB-D models (each episode consists of 200 timesteps). However, for the lighting perturbation, we found both RGBD-WM and PCWM took about 100 episodes to reach 100% task success – suggesting that PCWM adapts quickly in geometrically perturbed situations such as viewpoint.

E. Discussions & Limitations

Our initial experiments suggest the choice of point cloud encoder is important. Fig. 5 shows a comparison between model-free point cloud-based methods (i.e. PC-PPO variants) using encoders based on PointNet [32] and PointConv [16] on two tasks. We find significant gains using PointConv – attributed to PointConv's ability to reason about local points for feature extraction that PointNet lacks.

While we have shown that PCWM can be more sample-efficient and robust to visual perturbations, they can be slow to train (wall clock time) when compared to their RGB-D counterparts owing to the point cloud processing. We find PCWM to be ∼2-3.5x slower depending on the number of points in the input. Additionally, we note that the pruning of points needs to be performed on the novel static viewpoint and our system would likely fail with a moving camera. We hope that the community furthers the research on point cloud models for policy learning and mitigates these limitations.

VIII. RELATED WORK

Point Clouds in RL. Visual policy learning has seen significant progress in game playing [33], [34], robotic and dexterous manipulation [35]–[37], and locomotion tasks [7], [20], [38]. Most of this work leverages RGB(-D) imagery, hence explicit consideration for 3D representation learning has been limited [39]–[41]. Several recent works have proposed *model-free* policies that learn from partial point clouds [26], [42], [43] – demonstrating that the rich 3D information in point clouds can improve sample efficiency in interactive robotic tasks. We extend this body of work by (1) introducing a novel *model-based* RL framework for point clouds (PCWM) and (2) demonstrating that point clouds offer increased visual robustness for both model-based and model-free policies. Recently, GROOT [44] showed how point clouds can be robust to environment changes in the context of imitation learning, however, we focus on agents trained with RL policies.

Robustness in RL. Prior work has demonstrated that vision-based policies learned from RGB(-D) input can have poor generalization to new visual conditions [45]–[50]. These include changes due to new task instances, differences in object textures or lighting, novel viewpoints, or a combination of these induced by sim-to-sim or sim-to-real transfer. Inspired by work in computer vision, data augmentation [7], [51], [52] and representation pretraining [3], [53], [54] techniques have been employed to ameliorate this lack of robustness. These methods require careful design of image augmentations or laborious curation of diverse pretraining datasets to improve generalization [55]–[57]. While these techniques have demonstrated positive impacts, visual control policies for robotics can still exhibit a significant generalization gap [5]. In this work, we study the role of input representation in policy robustness. Our findings suggest that point cloud-based policies can be robust to viewpoints, lighting conditions and addition of new objects in the scene even *without* any of the above techniques.

Model-based RL. One technique in sequential decision making is to learn a model of the environment [58] and use it for planning [59]–[62] or policy learning [10], [20], [24], [63]. In the case of high dimensional inputs such as images, a popular approach is to learn the environment dynamics in a compact latent space that is supervised using rewards [14], [64] and image reconstruction [2], [13], [65], [66]. Such model-based RL agents [10], [20], [24], [63] have showcased higher sample efficiency compared to analogous model-free policies. However, these works have focused on settings where observations are RGB images or privileged state information such as the location of scene objects. We propose the first point cloud world model and investigate its sample efficiency and robustness.

Point Cloud Dynamics. Prior work has proposed variants of graph neural networks [67] to learn dynamics with point clouds [68], [69]. While these approaches can model realistic collision dynamics, they require point-to-point correspondences between frames. When deploying these models as part of a planning system in the real-world, prior work has applied mesh reconstruction on point clouds obtained by either multiple cameras [70] or a single RGB-D camera [71], which could be prone to errors for novel objects. While a dynamics model that takes partial point clouds as input was proposed [27], it requires 6-DoF object poses for its supervision and was not tested within an RL framework. In this work, we propose the first point cloud dynamics model that enables world model training in RL by directly operating on partial point clouds and using only the reward signal for its supervision.