Planning to Explore via Self-Supervised World Models

Ramanan Sekar [1 *] **Oleh Rybkin** [1 *] **Kostas Daniilidis** [1] **Pieter Abbeel** [2] **Danijar Hafner** [3 4] **Deepak Pathak** [5 6]

A. Appendix

Results DM Control Suite In Figure 1, we show the performance of our agent on all 20 DM Control Suite tasks from pixels. In addition, we show videos corresponding to all the plots on the project website: https://ramanans1.github.io/plan2explore/

Convention for plots We run every experiment with three different random seeds. The shaded area of the graphs shows the standard deviation in performance. All plot curves are smoothed with a moving mean that takes into account a window of the past 20 data points. Only Figure 5 of the main paper was smoothed with a window of past 5 data points so as to provide cleaner looking plots that indicate the general trend. Low variance in all the curves consistently across all figures suggests that our approach is very reproducible.

Rewards of new tasks To test the generalization performance of the our agent, we define three new tasks in the Cheetah environment:

- **Cheetah Run Backward** Analogous to the forward running task, the reward r is linearly proportional to the backward velocity v_b up to a maximum of $10\mathrm{m/s}$, which means $r(v_b) = \max(0, \min(v_b/10, 1))$, where $v_b = -v$ and v is the forward velocity of the Cheetah.
- **Cheetah Flip Backward** The reward r is linearly proportional to the backward angular velocity ω_b up to a maximum of $5\mathrm{rad/s}$, which means $r(\omega_b) = \max(0, \min(\omega_b/5, 1))$, where $\omega_b = -\omega$ and ω is the angular velocity about the positive Z-axis, as defined in DeepMind Control Suite.
- **Cheetah Flip Forward** The reward r is linearly proportional to the forward angular velocity ω up to a maximum of $5\mathrm{rad/s}$, which means $r(\omega) = \max(0, \min(\omega/5, 1))$.

Environment We use the DeepMind Control Suite (Tassa et al., 2018) tasks, a standard benchmark of tasks for continuous control agents. All experiments are performed with only visual observations. We use RGB visual observations with 64×64 resolution. We have selected a diverse set of 8 tasks that feature sparse rewards, high dimensional action spaces, and environments with unstable equilibria and environments that require a long planning horizon. We use episode length of 1000 steps and a fixed action repeat of $R = 2$ for all the tasks.

Agent implementation For implementing latent disagreement, we use an ensemble of 5 one-step prediction models with a 2 hidden-layer MLP, which takes in the RNN-state of RSSM and the action as inputs, and predicts the encoder features, which have a dimension of 1024. We scale the disagreement of the predictions by $10,000$ for the final intrinsic reward, this was found to increase performance in some environments. We do not normalize the rewards, both extrinsic and intrinsic. This setup for the one-step model was chosen over 3 other variants, in which we tried predicting the deterministic, stochastic, and the combined features of RSSM respectively. The performance benefits of this ensemble over the variants potentially come from the large parametrization that comes with predicting the large encoder features.

Baselines We note that while Curiosity (Pathak et al., 2017) uses L_2 loss to train the model, the RSSM loss is different (see (Hafner et al., 2019)); we use the full RSSM loss as the intrinsic reward for the Curiosity comparison, as we found it produces the best performance. Note that this reward can only be computed when ground truth data is available and needs a separate reward predictor to optimize it in a model-based fashion.

References

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. *ICML*, 2019. 1

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. *ICLR*, 2020. 3

[*]Equal contribution [1]University of Pennsylvania [2]UC Berkeley [3]Google Research, Brain Team [4]University of Toronto [5]Carnegie Mellon University [6]Facebook AI Research. Correspondence to: Oleh Rybkin <oleh@seas.upenn.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Vienna, Austria, PMLR 119, 2020.

Table 1. Zero-shot performance at 3.5 million environment steps (corresponding to 1.75 agent steps times 2 for action repeat). We report the average performance of the last 20 episodes before the 3.5 million steps point. The performance is computed by executing the mode of the actor without action noise. Among the agents that receive no task rewards, the highest performance of each task is highlighted. The corresponding training curves are visualized in Figure 1.

Zero-shot performance	**Plan2Explore**	**Curiosity**	**Random**	**MAX**	**Retrospective**	**Dreamer**
Task-agnostic experience	3.5M	3.5M	3.5M	3.5M	3.5M	–
Task-specific experience	–	–	–	–	–	3.5M
Acrobot Swingup	**280.23**	219.55	107.38	64.30	110.84	408.27
Cartpole Balance	950.97	917.10	**963.40**	–	–	970.28
Cartpole Balance Sparse	**860.38**	695.83	764.48	–	–	926.9
Cartpole Swingup	**759.65**	747.488	516.04	144.05	700.59	855.55
Cartpole Swingup Sparse	**602.71**	324.5	94.89	9.23	180.85	789.79
Cheetah Run	**784.45**	495.55	0.78	0.76	9.11	888.84
Cup Catch	**962.81**	**963.13**	660.35	–	–	963.4
Finger Spin	655.4	661.96	**676.5**	–	–	333.73
Finger Turn Easy	401.64	266.96	**495.21**	–	–	551.31
Finger Turn Hard	270.83	289.65	**464.01**	–	–	435.56
Hopper Hop	**432.58**	389.64	12.11	17.39	41.32	336.57
Hopper Stand	841.53	**889.87**	180.86	–	–	923.74
Pendulum Swingup	**792.71**	56.80	16.96	748.53	1.383	829.21
Quadruped Run	**223.96**	164.02	139.53	–	–	373.25
Quadruped Walk	182.87	**368.45**	129.73	–	–	921.25
Reacher Easy	**530.56**	416.31	229.23	242.13	230.68	544.15
Reacher Hard	66.76	**123.5**	4.10	–	–	438.34
Walker Run	429.30	**446.45**	318.61	–	–	783.95
Walker Stand	331.20	**459.29**	301.65	–	–	655.80
Walker Walk	**911.04**	889.17	766.41	148.02	538.84	965.51
Task Average	**563.58**	489.26	342.11	–	–	694.77

Table 2. Adaptation performance after 1M task-agnostic environment steps, followed by 150K task-specific environment steps (agent steps are half as much due to the action repeat of 2). We report the average performance of the last 20 episodes before the 1.15M steps point. The performance is computed by executing the mode of the actor without action noise. Among the self-supervised agents, the highest performance of each task is highlighted. The corresponding training curves are visualized in Figure 4 of the main paper.

Adaptation performance	**Plan2Explore**	**Curiosity**	**Random**	**MAX**	**Retrospective**	**Dreamer**
Task-agnostic experience	1M	1M	1M	1M	1M	–
Task-specific experience	150K	150K	150K	150K	150K	1.15M
Acrobot Swingup	**312.03**	163.71	27.54	108.39	76.92	345.51
Cartpole Swingup	**803.53**	747.10	416.82	501.93	725.81	826.07
Cartpole Swingup Sparse	**516.56**	456.8	104.88	82.06	211.81	758.45
Cheetah Run	**697.80**	572.67	18.91	0.76	79.90	852.03
Hopper Hop	**307.16**	159.45	5.21	64.95	29.97	163.32
Pendulum Swingup	**771.51**	377.51	1.45	284.53	21.23	781.36
Reacher Easy	848.65	**894.29**	358.56	611.65	104.03	918.86
Walker Walk	892.63	**932.03**	308.51	29.39	820.54	956.53
Task Average	**643.73**	537.95	155.23	210.46	258.78	700.27

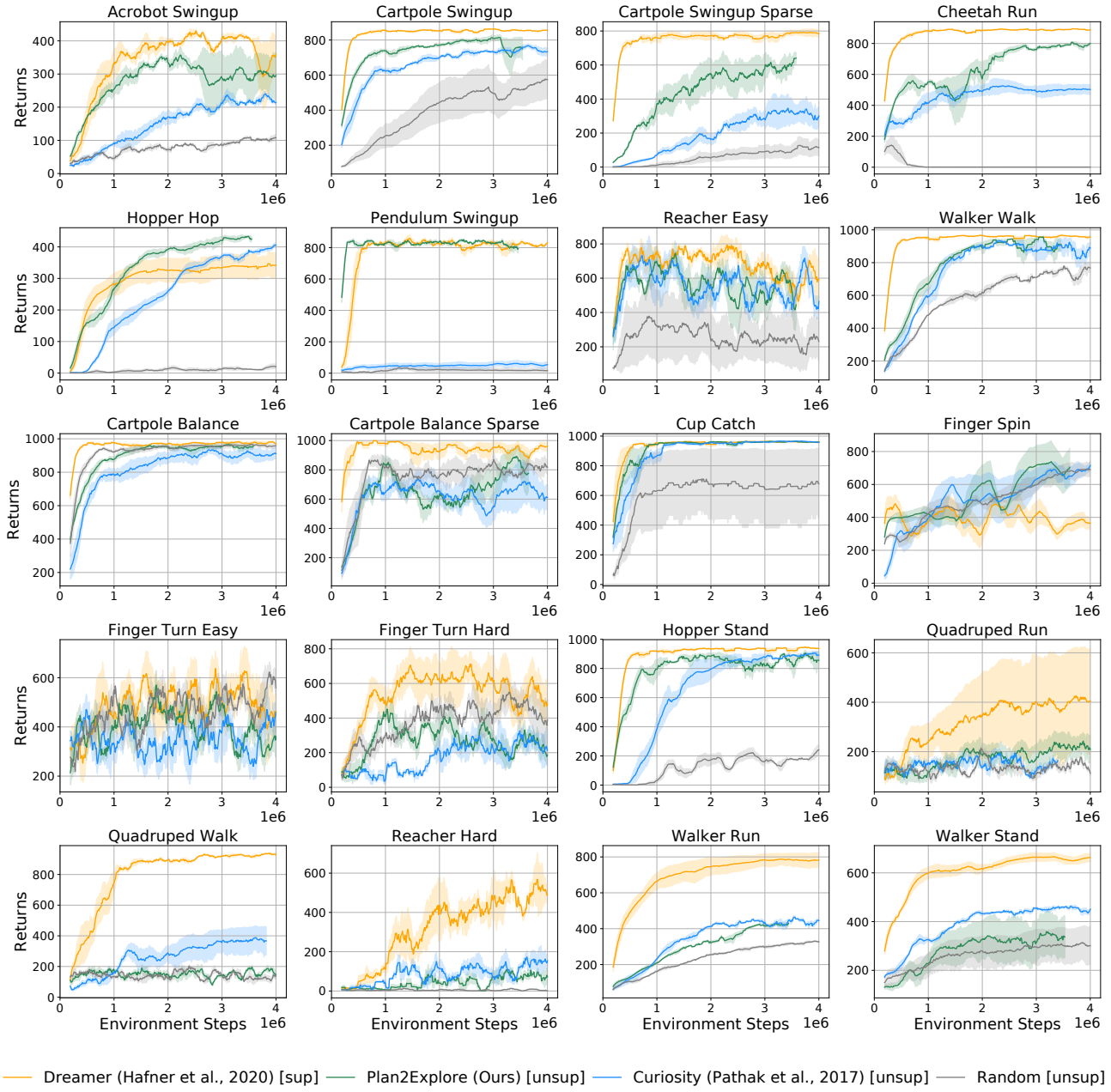

Figure 1. We evaluate the zero-shot performance of the self-supervised agents as well as supervised performance of Dreamer on all tasks from the DM control suite. All agents operate from raw pixels. The experimental protocol is the same as in Figure 3 of the main paper. To produce this plot, we take snapshots of the agent throughout exploration to train a task policy on the downstream task and plot its zero-shot performance. We use the same hyperparameters for all environments. We see that Plan2Explore achieves state-of-the-art zero-shot task performance on a range of tasks. Moreover, even though Plan2Explore is a self-supervised agent, it demonstrates competitive performance to Dreamer (Hafner et al., 2020), a state-of-the-art supervised reinforcement learning agent. This shows that self-supervised exploration is competitive to task-specific approaches in these continuous control tasks.

Pathak, D., Agrawal, P., Efros, A. A., and Darrell, T. Curiosity-driven exploration by self-supervised prediction. In *ICML*, 2017. 1

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., de Las Casas, D., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., Lillicrap, T., and Riedmiller, M. DeepMind control suite. Technical report, DeepMind, January 2018. URL https://arxiv.org/abs/1801.00690. 1