Proceedings of Machine Learning Research vol 242:414–427, 2024 6th Annual Conference on Learning for Dynamics and Control

Real-World Fluid Directed Rigid Body Control via Deep Reinforcement Learning

Mohak Bhardwaj[*†] **Thomas Lampe**[*] **Michael Neunert**[*]
Francesco Romano[*] **Abbas Abdolmaleki**[*] **Arunkumar Byravan**[*]
Markus Wulfmeier[*] **Martin Riedmiller**[*] **Jonas Buchli**[*]

Editors: A. Abate, M. Cannon, K. Margellos, A. Papachristodoulou

Abstract

Recent advances in real-world applications of reinforcement learning (RL) have relied on the ability to accurately simulate systems at scale. However, domains such as fluid dynamical systems exhibit complex dynamic phenomena that are hard to simulate at high integration rates, limiting the direct application of modern deep RL algorithms to often expensive or safety critical hardware. In this work, we introduce "Box o' Flows", a novel benchtop experimental control system for systematically evaluating RL algorithms in dynamic real-world scenarios. We describe the key components of the Box o' Flows, and through a series of experiments demonstrate how state-of-the-art model-free RL algorithms can synthesize a variety of complex behaviors via simple reward specifications. Furthermore, we explore the role of offline RL in data-efficient hypothesis testing by reusing past experiences. We believe that the insights gained from this preliminary study and the availability of systems like the Box o' Flows support the way forward for developing systematic RL algorithms that can be generally applied to complex, dynamical systems. Supplementary material and videos of experiments are available at https://sites.google.com/view/box-o-flows/home.

Keywords: Fluid dynamics, reinforcement learning, dynamical systems

1. Introduction

Reinforcement learning promises to deliver a principled, general-purpose framework for generating control policies for complex dynamical systems directly from experiential data, without the need for domain expertise (Sutton and Barto, 2018). Indeed, modern *deep* RL approaches that leverage expressive neural networks for function approximation have led to breakthroughs in a variety of domains, such as game-playing (Mnih et al., 2013; Schrittwieser et al., 2020; Mnih et al., 2015), protein folding (Jumper et al., 2021), control of tokamak plasmas in nuclear fusion reactors (Degrave et al., 2022), and real-world robotics (Tan et al., 2018; Handa et al., 2022).

However, a key ingredient in the success of these applications has been the ability to accurately simulate these systems at scale, and constructing such simulation environments themselves requires significant human effort and knowledge, thus forgoing the original promise of removing the need for domain expertise. For instance, leading approaches for learning-

[*] Google DeepMind
[†] University of Washington

based locomotion and dexterous manipulation (Tan et al., 2018; Kumar et al., 2021; Fu et al., 2021; Handa et al., 2022; Pinto et al., 2017) rely on a *sim-to-real* paradigm to learn robust policies in simulation that can be directly transferred to the real world. Even when policies are learned directly on real hardware, practitioners often rely on simulation to gain intuition about the problem domain, and make critical design decisions such as the choice of algorithm, reward functions and other hyperparameters (Lee et al., 2022; Schwab et al., 2019).

In addition to human expertise involved in simulation design, the high sample complexity of current RL algorithms necessitates fast simulations to achieve reasonable wall clock times for training. While this is possible for domains such as video games and rigid-body systems (Todorov et al., 2012; Liang et al., 2018), for several real-world problems satisfying this need becomes increasingly expensive or outright impossible. Examples include systems involving non-steady fluid dynamics and/or continuum mechanics (e.g. flying, swimming, soft matter based mechatronic systems), and multi-scale problems that occur in biological systems or digital twins of large industrial systems. How can we scale RL to such systems?

This work focuses on one such domain - the control of coupled mechanical-fluid dynamic systems. Here, the fact that one can not assume steady state dynamics hugely increases the complexity of simulations. For example, consider an Unmanned Aerial Vehicle operating in off-nominal regimes such as high angle of attack or ground/obstacle effects. Here, the turbulent air flows that are generated can be difficult to model, and create instabilities that nominal controllers are incapable of handling. While there is a growing literature on learning control policies in the presence of non-steady fluid flows that utilize simulation (Verma et al., 2018), and the dynamics are known in principle, simulating them requires supercomputers which is beyond the resources of most practitioners. The study of such systems raises interesting questions that have several implications for real-world deployment of reinforcement learning.

1. How do we design experiments to characterize the capabilities of a system that is hard to simulate at scale?
2. How do we ensure sample efficient learning given limited data collection rates?
3. How can we efficiently re-use prior experience to test different hypotheses, and aid the learning of new behaviors?

To investigate these questions, we have developed a novel fluid-dynamic control system dubbed "Box o' Flows". This system consists of 9 upward facing nozzles arranged in parallel with a proportional pneumatic valve per nozzle regulating the airflow. The valves can be controlled programmatically to create complex pressure fields between two parallel panels forming a box. The airflow can be used to control the state of rigid objects, such as colored balls, that are placed inside. The setup is also equipped with an RGB camera capturing the box and objects inside it (Fig. 1 provides a detailed overview). The system is intentionally designed to be impossible to simulate accurately at the high integration rates required by deep RL algorithms, and exhibits complex non-steady fluid dynamics which makes (unknowingly) injecting prior human knowledge, or hand-designing control policies hard in practice. In Fig. 2 we demonstrate fluid patterns generated by the air flowing through the nozzles.

This work serves as a preliminary investigation of how model-free RL can be used to learn a variety of dynamic control tasks on the Box o' Flows directly in the real world, as well as characterize hardware capabilities. We limit the algorithms tested to the state-of-the-art

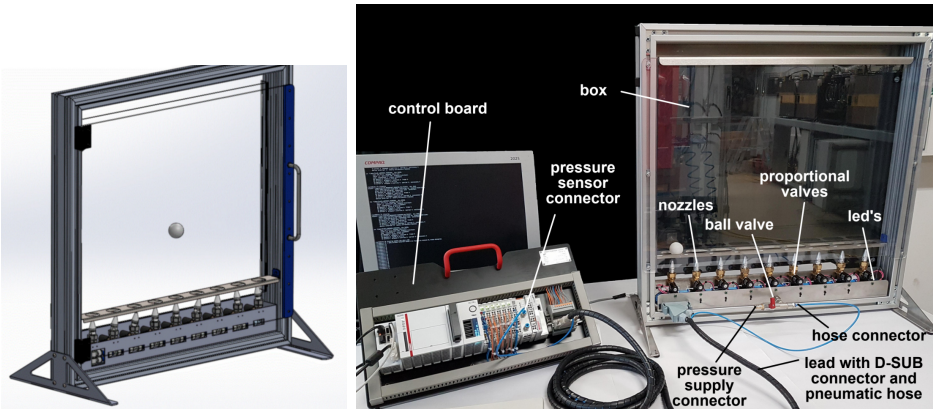


Figure 1: An overview of the different components of bench-top Box o' Flows system.

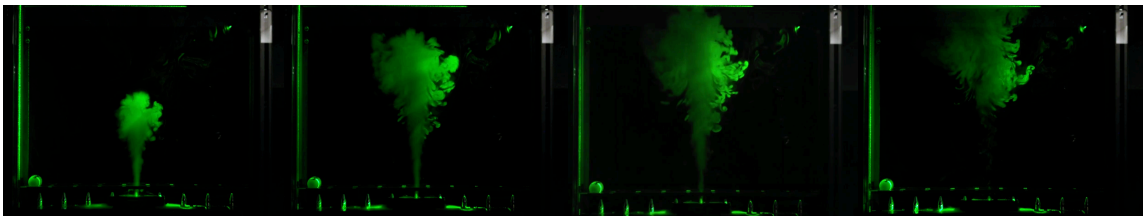


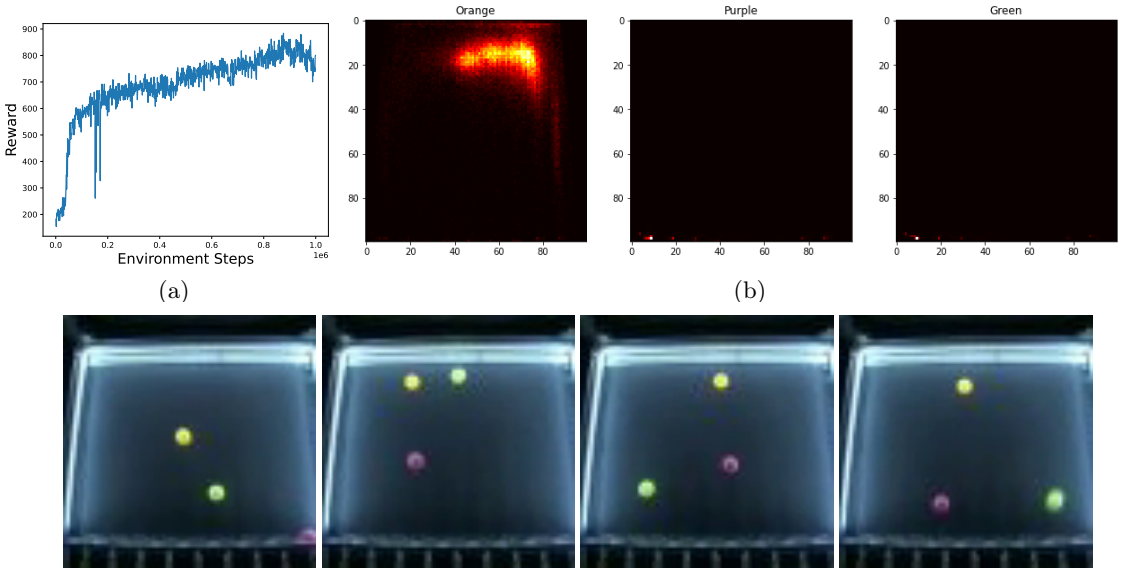
Figure 2: Smoke visualizes the complex flow field that emerges from a single valve with constant flow. This illustrates the complex relationship between actuator and the flow field and ultimately its effects on the balls. This relationship is further complicated when several actuators are acting simultaneously.

Maximum A-posteriori Policy Optimization (MPO) (Abdolmaleki et al., 2018b), with fixed hyperparameters across different experiments. Desired behaviors are described via minimally specified rewards functions, which gives the RL agent the freedom to find interesting control strategies. Furthermore, we test how offline RL can be used as a means for hypotheses testing by training new policies on logged data from past experiments, and intermittently evaluating them on the real system. Our framework can generate diverse dynamic behaviors to control the state of multiple rigid objects (table tennis balls) such as hovering, rearrangement, stacking and goal-reaching (detailed in Sec. 4). In summary, our main contributions are:

- We present a novel benchtop fluid-dynamic control system geared towards real-world RL research.
- We demonstrate the application of sample-efficient, model-free RL to learning dynamic behaviors and analyzing hardware capabilities.
- We explore how offline RL with past data can be used to test various hypotheses when simulation is not available.

2. Box o' Flows - System Overview

In this section we describe the Box o' Flows system as shown in Fig. 1. The system comprises of a 70cmX70cm square aluminum frame on which a black opaque back panel



(a) (b)

(c)

Figure 3: Reinforcement learning applied to the task of maximizing the height of orange ball in presence of distractors (purple and green). The non-steady fluid dynamics of interacting objects and complex actuator coupling makes it hard to hand-design controllers. (a) Reward curve (b) Heatmap visualization of states visited by learned policy (averaged over 100 episodes) (c) Filmstrip of an episode (More details in Sec. 4)

and a transparent front panel are mounted, creating a shallow box of roughly 60mm depth. Mounted at the bottom edge of this box is a blade consisting of 9 proportional flow control valves (SMC PVQ 30), each attached to a nozzle facing upwards. An LED strip is mounted on the remaining three sides to evenly illuminate the interior of the box. Objects, such as the colored table tennis balls used in this work, can be placed within the space inside the box, so that their state can be controlled via the airflow.

All valves share a common air supply that is hooked up to an air pump and fed via the proportional control valves at 6 bar. By connecting all the nozzles to a single pump, the supply pressure and consequently the flow across the nozzles drops when multiple valves are opened simultaneously. This cross coupling has been added intentionally, to increase the complexity of the system behaviour. Further, the system can only measure the overall supply pressure and not the pressure or flow at each valve.

Communication with the valves and sensors is realized through EtherCAT, a realtime ethernet protocol providing synchronization between the individual nozzles. The control system runs on an intel-i7 based Beckhoff industrial PC running Linux and the EtherLab EtherCAT master (Ingenieurgemeinschaft IgH GmbH, 2024). A machine vision camera (BASLER Ace acA1920-40gc) is attached via GigE Ethernet and captures RGB images of the interior of the box. While the underlying Ethercat bus runs at higher rates, for the experiments described here a control rate of 20 Hz has been used [1] .

1. Further details of system design can be found at https://sites.google.com/view/box-o-flows/home.

2.1. What Makes Box o' Flows a Hard Problem?

The Box o' Flows brings to light several key challenges in controlling real-world systems with complex dynamics. As a motivating example, consider a simple setting with three colored balls placed inside the box, and one must design a control policy to maximize the height of one of the balls, with the others being distractors, i.e their motion is not constrained. (For reference, Fig. 3(c) shows behavior learned by our framework). While intuitively it may seem straightforward to hand-design a controller (eg. maximally open all valves), the nature of the Box o' Flows makes it hard in practice.

First, the cross coupling between actuators due to shared air supply means that maximally opening all valves will not work for this task since the pressure per valve will drop. This relation is also hard to model and changes unpredictably over time due to practical issues such as oil accumulation. Second, in the Box o' Flows there is a less direct relationship from the actuator space to the state space than a standard robotic system. The non-steady dynamics of the emerging flow given an actuation input is highly complex and stochastic, especially as the objects interact with each other, and the controller must account for this. Moreover, current methods for accurately simulating non-steady flows require large amounts of compute which precludes techniques like *sim-to-real* RL that rely on cheap simulated data.

Third, the system is highly under-observed as we can not directly measure the flow field inside the box, but only the supply pressure. One can only attempt to recover this information from a history of images of object motion from the camera stream. Finally, real-world data collection is a limiting factor. The current setup can collect approximately 1M environment steps per day, thus, experiments must be designed carefully for efficient data use.

From the above, it is clear that hand-designing controllers is non-trivial even in simple settings, and model-based techniques that rely on accurate system identification or simulation can be prohibitively expensive. It is therefore more promising to consider efficient data-driven approaches that can overcome these constraints.

3. Methods

We focus on sample-efficient, model-free RL algorithms that can facilitate learning control policies from limited real-world experience, both via online interactions and offline datasets. To this end, we leverage a high performance off policy actor-critic algorithm, Maximum Aposteriori Policy Optimization (MPO) (Abdolmaleki et al., 2018a,b). At iteration k, MPO updates the parameters ϕ and θ of the critic $Q_\phi^{\pi^k}$ and policy $\pi_\theta^k(\cdot|s)$ respectively by optimizing

$$\min_\phi \left(r_t + \gamma Q_{\phi'}^{\pi^{k-1}}(s_{t+1}, a_{t+1} \sim \pi^{k-1}) - Q_\phi^{\pi^k}(s_t, a_t) \right) \tag{1}$$

$$\pi_\theta^{k+1} = \arg\min \mathrm{E}_\mu \left[KL(q(a|s)||\pi_\theta((a|s))) \right] \tag{2}$$

where $q(a|s) \propto \exp(Q_\phi^k(s,a)\mu/\beta))$ is a non-parametric estimate of the optimal policy given a temperature β, and $KL\left(q(\cdot|s)||\pi(\cdot|s)\right)$ is the KL divergence, and μ is the distribution of states stored in a replay buffer. The efficient off-policy updates enable MPO to demonstrate sample-efficient learning in high dimensional continuous control tasks. We refer the reader to Abdolmaleki et al. (2018a) for a detailed derivation of the update rules.

Offline RL: Since Box o' Flows is distinct from existing robotic setups, it can be a priori unknown what reward functions can lead to desired behaviors with online RL. This problem is aggravated by the lack of simulation and constrained data collection rates. Thus, it is vital to be able to to re-use prior experience to test hypotheses about new rewards. To this end, we focus on the offline RL paradigm that enables learning effective policies from logged datasets without further exploration (Levine et al., 2020). To deal with limited data coverage, modern offline RL algorithms (Kumar et al., 2020; Cheng et al., 2022) rely on a concept of pessimism under uncertainty by optimizing performance lower bounds, such that the agent is penalized for choosing actions outside the data support.

The actor update of MPO can be easily adapted to the offline setting. Given a dataset of transitions $\mathcal{D} = \{(s_i, a_i r_i, s_{i+1})\}_{i=1}^{N}$ collected by a behavior policy μ_B, we can modify the distribution of states in Eq. 2 from μ to μ_B (state distribution in $\mathcal{D}$) and non-parametric optimal policy to $q(a|s) \propto \exp(Q_\phi^k(s,a)\mu_B/\beta)$. The actor update thus encourages reward maximization while staying close to μ_B. This is a special case of Critic Regularized Regression (CRR) (Wang et al., 2020), a state-of-the-art offline RL algorithm, and can be implemented it in a common framework with MPO. In our setting, we re-label data from prior online RL experiments with new rewards (in line with (Davchev et al., 2021; Yarats et al., 2022; Lambert et al., 2022; Tirumala et al., 2023)), and train a CRR agent offline that is tested intermittently on the real system to validate policy performance. The minimal use of hardware enables us to test multiple policies instead of just one that continuously trains online. We now present our main empirical results.

4. Experiments

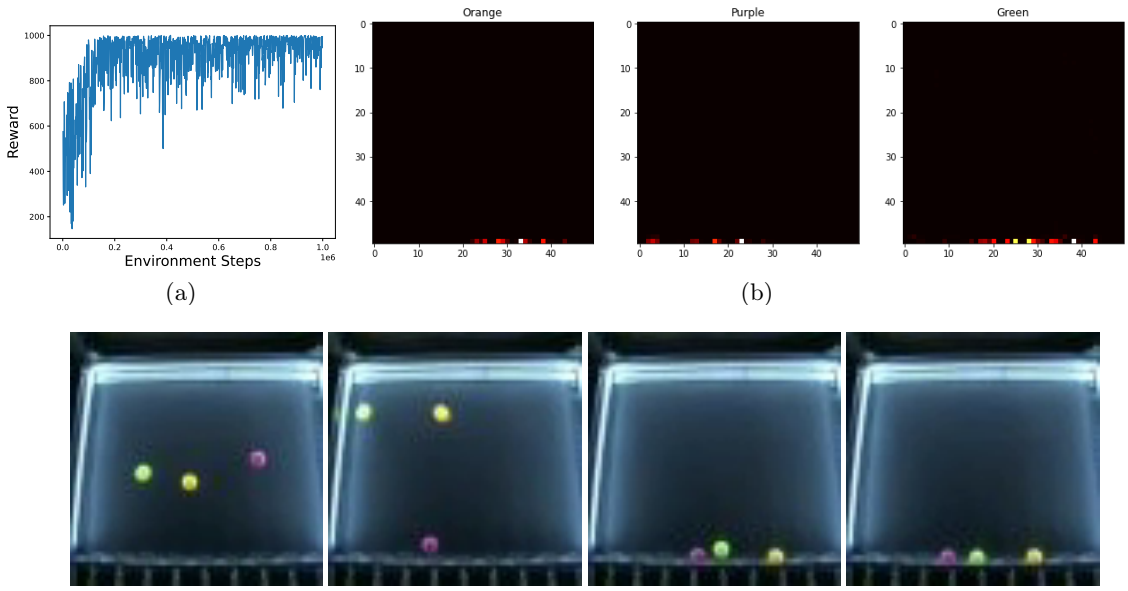
We use a suite of dynamic control tasks to test the efficacy of our RL framework and study the physical capabilities of the Box o' Flows system.

Setup: To delineate the interplay between hardware capabilities and algorithm performance, we keep our RL agent (Sec. 3) fixed across all tasks. We use a distributed learning framework akin to Hoffman et al. (2020), and select hyperparameters using a candidate task where optimal behavior is qualitatively known (see below). The actor and critic are represented by feedforward neural networks, and object state by a history of pixel xy coordinates measured from the vision system via a blob detector. The 9-dim action space represents degree of valve opening in the range $[0, 1]$. Object locations are reset using random air bursts at the beginning of every episode (1000 steps long at 20Hz). We describe desired behaviors via simple rewards based on desired object configurations, which gives the RL agent the freedom to find interesting control strategies. Next, we describe the tasks in detail.[2]

4.1. Learning Dynamic Behaviors with Online RL

Hovering with Distractors: We first consider the task of maximizing the height of a target ball (orange) in the presence of distractors (purple and green), and use it to select relevant hyperparameters. Intuitively, a near-optimal strategy is to place the distractors near a bottom corner and use other valves to hover the target ball. However, as described in

2. A complete description of rewards and hyperparameters can be found in the supplementary material at https://sites.google.com/view/box-o-flows/home



(a) (b)

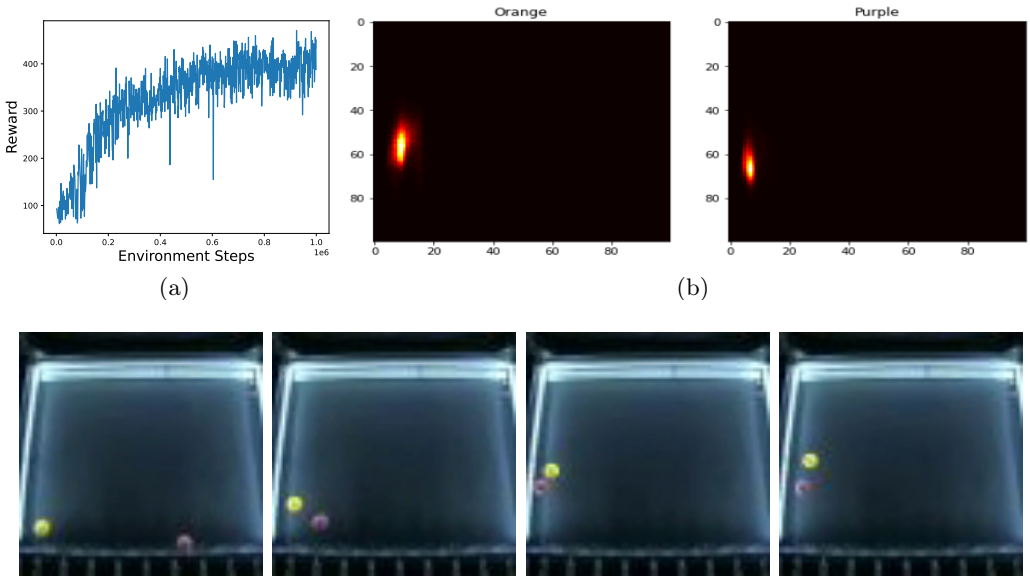
(c)

Figure 4: Task: Orange in right, purple in left (a) Reward curve and (b) Heatmap visualization of states visited by learned policy (averaged over 100 episodes) (c) Filmstrip of an episode.

Sec. 2.1, complex actuator coupling and non-steady flow patterns make it hard to hand-design such a controller. We test whether our MPO agent can recover this intuitive policy, by training it using a reward proportional to the pixel y coordinate of only the target ball, normalized to $[0.0, 1.0]$ (based on maximum and minimum coordinate values). Fig. 3(a) presents the reward obtained over environment steps during training that shows the agent is able to obtain near-optimal reward in about 1M steps. In Fig. 3(b), we visualize the learned behavior via coarsely discretized heatmaps of ball locations over the last 100 training episodes, which show that the agent successfully learns the intuitive policy of maintaining the target ball near the top while pushing the distactors near the bottom left.

Object Rearrangement: Next, we consider a harder task where the agent must place two target balls (orange and purple) anywhere in the right and left halves of the box respectively, with the green ball being a distractor. Here, it is hard to even intuitively reason about optimal behavior as it depends on the initial object locations which are randomized. We provide our agent a sparse reward equal to the product of the horizontal distances from the respective goal regions, which forces it to accomplish both tasks. As shown in Fig. 4, we observe that this task is much easier for RL, and our agent is able to achieve near-optimal reward within approximately 200k environment steps. Interestingly, the agent also learns a stable strategy of switching off controls once the balls are in the target halves as can be seen in the heatmap visualizations in Fig. 4(b) and filmstrip Fig. 4(c).

Stacking: To test if our agent can exploit the airflow at a finer level, we consider a more challenging task of stacking two balls on top of each other. We again provide the agent a product of two simple rewards: keep the y-coordinate of the orange over purple by a fixed value and align x-coordinates. We observe that the agent not only learns to successfully stack



(a) (b)

(c)

Figure 5: Task: Stack orange ball over purple (a) Reward curve. (b) Heatmap visualization of states visited by learned policy (averaged over 100 episodes). (c) Filmstrip of an episode.

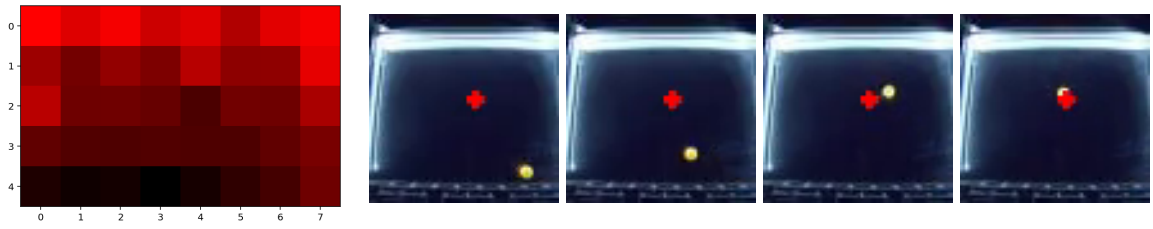
the balls Fig. 5(a), but also discovers an interesting strategy to always align them against the left wall of box as it is easier to control airflow near the walls (Fig. 5(b)).

4.2. Learning Goal-conditioned Policies to Analyze Reachability

We wish to characterize what parts of the Box o' Flows are reachable given the actuator configuration and limits. Since, it is not possible analytically, we leverage our RL agent by designing a goal reaching task where the agent must position a ball to randomly chosen pixel targets. We add the goal location to the observation, and train MPO for 1.2M environment steps (1200 episodes). We visually analyze reachability by plotting a coarsely discretized heatmap of reaching errors for different target regions (Fig. 6). The intensity of each bin is proportional to the cumulative reaching error for every training episode during which the target was in that bin (normalized such that black is minimum error and red is maximum). This accounts for noise due to policy training and exploration, target height and inherent system stochasticity. The analysis clearly shows that target locations closer to the bottom and center are easier to reach in general. Also, targets near the bottom right are harder than bottom-left and bottom-center, which reveals an imbalance in the airflow through different nozzles. Interestingly, targets closer to the walls are also easily reachable since the agent can better exploit the airflow. These findings also align with the behavior learned in the stacking task. The hardest regions to reach are at the top, especially top-left and top-right corners.

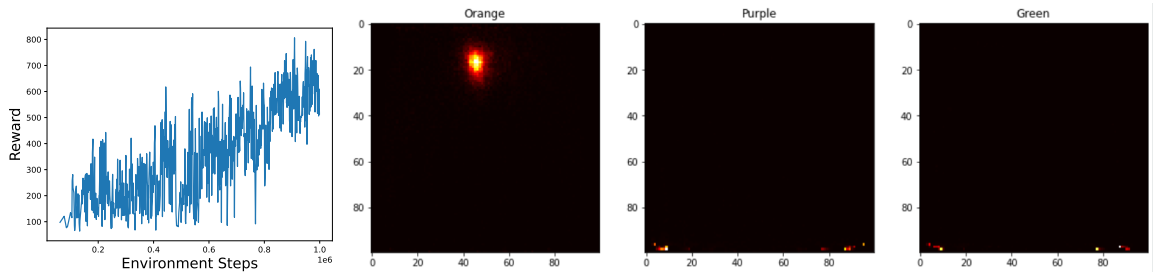
4.3. Re-using Past Experience via Offline RL

As discussed in Sec. 3, we perform a preliminary experiment to study how offline RL from logged datasets obtained from online RL experiments can be used to test new reward



(a) (b)

Figure 6: (a) Pixel intensity is proportional to cumulative error for episodes when the target was in that pixel's bin. Error is the average distance between the ball and target in the last 200 episode steps. (b) Filmstrip of an episode.



(a) (b)

(c)

Figure 7: Task: Maximize the height of orange ball while aligning along the vertical center line in presence of distractors (a) Reward curve and (b) Heatmap visualization of states visited by learned policy (averaged over 100 episodes)(c) Filmstrip of an episode.

functions. If the logged data has sufficient coverage (i.e the target task is close enough) one can expect the learned policy from offline RL to be representative of what we can obtain by running online RL from scratch. Specifically, we use data from the task of hovering with distractors and re-label the rewards to additionally constrain the ball to remain close to the vertical center line. We then train CRR (Sec. 3) and evaluate the current learner's policy intermittently on the real system. We show the learning curve in Fig. 7(a) and a heatmap of the states visited by the learned policy in Fig 7(b). A stark difference is observed compared to the heatmap in Fig. 3(b) as the states concentrate entirely near the center as desired, while distractors are at different bottom corners. This experiment provides a promising first result for applying offline RL to study complex dynamical systems like Box o' Flows.

5. Related Work

Deep RL for Complex Physical Systems: In addition to real-world robotics discussed in Sec. 1, RL is also applied to control other complex systems, such as data center cooling systems (Lazic et al., 2018). Degrave et al. (2022) apply deep RL to control Tokamak plasmas in nuclear fusion reactors. This is a high dimensional dynamic control problem, however, they rely on simulation in a constrained regime to learn policies that transfer to the real system.

Machine Learning for Fluid Dynamics: Machine learning and deep RL are being extensively used for the modelling and control of fluid dynamical systems. We provide an overview here and refer the reader to the review papers by Brunton et al. (2020) and Larcher and Hachem (2022) for a comprehensive treatment:

- **Flow Modelling & Control**: Machine learning is leveraged to accelerate high-fidelity numerical simulations of fluid dynamics (Kochkov et al., 2021) and automatic turbulence modelling (Novati et al., 2021). Deep RL is also applied to active flow control (Fan et al., 2020; Bieker et al., 2020) and deformable object manipulation (Xu et al., 2022). The work by Ma et al. (2018) on rigid body manipulation via directed fluid flow is the closest to ours, however, they are limited to simulation with several approximations for computational efficiency.

- **Modelling Biological Systems:** Deep RL can aid the understanding of physical mechanisms and decision-making processes underlying animal behavior. Verma et al. (2018) combine RL with high-fidelity fluid simulation to study how schooling helps fish reduce energy expenditure. However, running such simulations requires computational resources which are prohibitive for most practitioners. The flight behavior of birds is also studied to design agile UAVs. Tedrake et al. design a glider that demonstrates perching under high angle of attack and Reddy et al. (2016) learn energy efficient soaring behaviors by creating numerical models of turbulent thermal convective flows based on bird flight.

Offline RL: Offline RL aims to learn competitive policies using logged data without further exploration, and consists of both model-free (Kumar et al., 2020; Cheng et al., 2022; Kostrikov et al., 2021), and model-based (Yu et al., 2021; Bhardwaj et al., 2023; Kidambi et al., 2020) variants. A key challenge is offline policy evaluation under limited data coverage (Levine et al., 2020) which is generally solved by importance sampling based approaches (Precup, 2000). We tackle this via intermittent evaluations of the learner's policy on the real system.

6. Discussion

We presented Box o' Flows, a novel benchtop fluid-dynamic control system geared towards real-world RL research. We empirically demonstrated how model-free RL can be used to learn diverse dynamic behaviors directly on hardware, and the applicability of offline RL for efficient re-use of past experience. However, the capabilities of the learning agent can be further enhanced. First, model-based RL methods can be utilized to enhance the understanding of system dynamics and share data among tasks. Second, while our preliminary experiment with offline RL offers promising results, we expect we can improve performance by leveraging methods such as Cheng et al. (2022) that provide robust policy improvement guarantees. Last but not least, there are many variants of such table top systems that can be realized fairly straightforwardly to vary the difficulty and scope of the experiment.

Acknowledgments

The authors would like to thank IgH for their contribution to the design and engineering of the Box o'Flows and the Google DeepMind London Robotics Lab team for engineering and operational support.

References

Abbas Abdolmaleki, Jost Tobias Springenberg, Jonas Degrave, Steven Bohez, Yuval Tassa, Dan Belov, Nicolas Heess, and Martin Riedmiller. Relative entropy regularized policy iteration. *arXiv preprint arXiv:1812.02256*, 2018a.

Abbas Abdolmaleki, Jost Tobias Springenberg, Yuval Tassa, Remi Munos, Nicolas Heess, and Martin Riedmiller. Maximum a posteriori policy optimisation. *arXiv preprint arXiv:1806.06920*, 2018b.

Mohak Bhardwaj, Tengyang Xie, Byron Boots, Nan Jiang, and Ching-An Cheng. Adversarial model for offline reinforcement learning. *arXiv preprint arXiv:2302.11048*, 2023.

Katharina Bieker, Sebastian Peitz, Steven L Brunton, J Nathan Kutz, and Michael Dellnitz. Deep model predictive flow control with limited sensor data and online learning. *Theoretical and computational fluid dynamics*, 34(4):577–591, 2020.

Steven L Brunton, Bernd R Noack, and Petros Koumoutsakos. Machine learning for fluid mechanics. *Annual review of fluid mechanics*, 52:477–508, 2020.

Ching-An Cheng, Tengyang Xie, Nan Jiang, and Alekh Agarwal. Adversarially trained actor critic for offline reinforcement learning. *International Conference on Machine Learning*, 2022.

Todor Davchev, Oleg O. Sushkov, Jean-Baptiste Regli, Stefan Schaal, Yusuf Aytar, Markus Wulfmeier, and Jonathan Scholz. Wish you were here: Hindsight goal selection for long-horizon dexterous manipulation. *ArXiv*, abs/2112.00597, 2021.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602 (7897):414–419, 2022.

Dixia Fan, Liu Yang, Zhicheng Wang, Michael S. Triantafyllou, and George Em Karniadakis. Reinforcement learning for bluff body active flow control in experiments and simulations. *Proceedings of the National Academy of Sciences*, 117(42):26091–26098, 2020. doi: 10.1073/pnas.2004939117.

Zipeng Fu, Ashish Kumar, Jitendra Malik, and Deepak Pathak. Minimizing energy consumption leads to the emergence of gaits in legged robots. *arXiv preprint arXiv:2111.01674*, 2021.

Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. *arXiv preprint arXiv:2210.13702*, 2022.

Matthew W Hoffman, Bobak Shahriari, John Aslanides, Gabriel Barth-Maron, Nikola Momchev, Danila Sinopalnikov, Piotr Stańczyk, Sabela Ramos, Anton Raichuk, Damien Vincent, et al. Acme: A research framework for distributed reinforcement learning. *arXiv preprint arXiv:2006.00979*, 2020.

Ingenieurgemeinschaft IgH GmbH. EtherLab EtherCat Master, 2024. URL https://etherlab.org/ethercat.

John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Žídek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In *NeurIPS*, 2020.

Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21):e2101784118, 2021. doi: 10.1073/pnas.2101784118.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33: 1179–1191, 2020.

Nathan Lambert, Markus Wulfmeier, William F. Whitney, Arunkumar Byravan, Michael Bloesch, Vibhavari Dasagi, Tim Hertweck, and Martin A. Riedmiller. The challenges of exploration for offline reinforcement learning. *ArXiv*, abs/2201.11861, 2022.

A Larcher and E Hachem. A review on deep reinforcement learning for fluid mechanics: an update. *Physics of Fluids*, 34(11):111301, 2022.

Nevena Lazic, Craig Boutilier, Tyler Lu, Eehern Wong, Binz Roy, MK Ryu, and Greg Imwalle. Data center cooling using model-predictive control. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

Alex X. Lee, Coline Manon Devin, Yuxiang Zhou, Thomas Lampe, Konstantinos Bousmalis, Jost Tobias Springenberg, Arunkumar Byravan, Abbas Abdolmaleki, Nimrod Gileadi,

David Khosid, Claudio Fantacci, Jose Enrique Chen, Akhil Raju, Rae Jeong, Michael Neunert, Antoine Laurens, Stefano Saliceti, Federico Casarini, Martin Riedmiller, raia hadsell, and Francesco Nori. Beyond pick-and-place: Tackling robotic stacking of diverse shapes. In Aleksandra Faust, David Hsu, and Gerhard Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 1089–1131. PMLR, 08–11 Nov 2022.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Jacky Liang, Viktor Makoviychuk, Ankur Handa, Nuttapong Chentanez, Miles Macklin, and Dieter Fox. Gpu-accelerated robotic simulation for distributed reinforcement learning, 2018.

Pingchuan Ma, Yunsheng Tian, Zherong Pan, Bo Ren, and Dinesh Manocha. Fluid directed rigid body control using deep reinforcement learning. *ACM Transactions on Graphics (TOG)*, 37(4):1–11, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

Guido Novati, Hugues Lascombes de Laroussilhe, and Petros Koumoutsakos. Automating turbulence modelling by multi-agent reinforcement learning. *Nature Machine Intelligence*, 3(1):99–99, 2021.

Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017.

Doina Precup. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, page 80, 2000.

Gautam Reddy, Antonio Celani, Terrence J Sejnowski, and Massimo Vergassola. Learning to soar in turbulent environments. *Proceedings of the National Academy of Sciences*, 113(33): E4877–E4884, 2016.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839): 604–609, 2020.

Devin Schwab, Jost Tobias Springenberg, Murilo Fernandes Martins, Michael Neunert, Thomas Lampe, Abbas Abdolmaleki, Tim Hertweck, Roland Hafner, Francesco Nori, and Martin A. Riedmiller. Simultaneously learning vision and feature-based control policies for real-world ball-in-a-cup. In Antonio Bicchi, Hadas Kress-Gazit, and Seth Hutchinson, editors, *Robotics: Science and Systems XV, University of Freiburg, Freiburg im Breisgau, Germany, June 22-26, 2019*, 2019. doi: 10.15607/RSS.2019.XV.027.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

Russ Tedrake, Zack Jackowski, Rick Cory, John William Roberts, and Warren Hoburg. Learning to fly like a bird. Citeseer.

Dhruva Tirumala, Thomas Lampe, José Enrique Chen, Tuomas Haarnoja, Sandy Huang, Guy Lever, Ben Moran, Tim Hertweck, Leonard Hasenclever, Martin Riedmiller, Nicolas Manfred Otto Heess, and Markus Wulfmeier. Replay across experiments: A natural extension of off-policy rl. 2023.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

Siddhartha Verma, Guido Novati, and Petros Koumoutsakos. Efficient collective swimming by harnessing vortices through deep reinforcement learning. *Proceedings of the National Academy of Sciences*, 115(23):5849–5854, 2018.

Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

Zhenjia Xu, Cheng Chi, Benjamin Burchfiel, Eric Cousineau, Siyuan Feng, and Shuran Song. Dextairity: Deformable manipulation can be a breeze, 2022.

Denis Yarats, David Brandfonbrener, Hao Liu, Michael Laskin, P. Abbeel, Alessandro Lazaric, and Lerrel Pinto. Don't change the algorithm, change the data: Exploratory data for offline reinforcement learning. *ArXiv*, abs/2201.13425, 2022.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.