Physics filtering favors the generalization of robot learning

Jindou Jia
Nanyang Technological University
Shixuan Han
Beihang University
Meng Wang
Beihang University
Gen Li
Nanyang Technological University
Zihan Yang
Beihang University
Sicheng Zhou
Nanyang Technological University
Kexin Guo
Beihang University
Jianfei Yang
jianfei.yang@ntu.edu.sg
Nanyang Technological University
Xiang Yu
Beihang University
Wei Wang
Beijing Aerospace Control Instrument Research Institute
Lei Guo
Beihang University

Article

Keywords:

Posted Date: May 6th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9460171/v1

Additional Declarations: No competing interests reported.

Physics filtering favors the generalization of robot learning

Jindou Jia[1,3†], Shixuan Han[3†], Meng Wang[3†], Gen Li[1], Zihan Yang[4], Sicheng Zhou[2], Kexin Guo[4*], Jianfei Yang[1,2*], Xiang Yu[3], Wei Wang[5], Lei Guo[3]

[1]School of Mechanical and Aerospace Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798, Singapore.
[2]School of Electrical and Electronic Engineering, Nanyang Technological University, 50 Nanyang Avenue, 639798, Singapore.
[3]School of Automation Science and Electrical Engineering, Beihang University, Xueyuan Road, Beijing, 100191, China.
[4]School of Aeronautic Science and Engineering, Beihang University, Xueyuan Road, Beijing, 100191, China.
[5]Beijing Aerospace Control Instrument Research Institute, Yongding Road, Beijing, 100854, China.

*Corresponding author(s). E-mail(s): kxguo@buaa.edu.cn; jianfei.yang@ntu.edu.sg;
Contributing authors: jindou.jia@ntu.edu.sg; SY2503104@buaa.edu.cn; mwangbuaa@126.com; gen.li@ntu.edu.sg; snrt_zzhan@buaa.edu.cn; sicheng.zhou@ntu.edu.sg; xiangyu_buaa@buaa.edu.cn; yfwangwei@vip.sina.com; lguo@buaa.edu.cn;
†These authors contributed equally to this work.

Abstract

Living organisms exhibit extraordinary adaptability to unseen environments through their intrinsic physical structures and lifelong feedback-driven learning. Endowing robots with comparable generalization is critical for reliable operation in the real world. While recent approaches attempt to improve generalization by scaling training data, such strategies remain impractical for robotics, where collecting real-world demonstrations at the scale of large language models is prohibitively costly and slow. Contrary to this reliance on massive datasets, we show that robots can generalize effectively even with limited training data

by leveraging a feedback mechanism, namely PhyFilter, that corrects learning outputs with physics-filtered learning residuals. PhyFilter operates as a lightweight, model-agnostic module whose parameters can be automatically optimized through an auto-learning algorithm, eliminating manual tuning and enabling seamless integration with diverse robot policies. We validate PhyFilter across four representative robotic systems, demonstrating that it enables quadruped robots to generalize to unseen terrains, payload variations, and speed ranges; drones to flight under unseen wind disturbances; aerial manipulators to achieve centimeter-level in-air capture despite wind and mass uncertainties; and acceleration differentiators to remain robust with distribution shift. These results reveal a scalable and data-efficient pathway toward generalizable machine intelligence, showing that physics-filtered feedback can serve as a powerful alternative to massive data scaling.

1 Introduction

Over the past decade, large language models (LLMs) [1–3] have attained remarkable advancements, primarily fueled by extensive training datasets, large-scale neural networks, and massive computational power [4–6]. As robots serve as the physical instantiation of intelligence, delineating a feasible pathway toward human-level robotic intelligence has emerged as a pivotal and intuitive goal [7–9]. A core prerequisite for such machine intelligence lies in generalization—robust adaptation to unseen environments—a critical capability for reliable real-world deployment [10]. With network architectures and graphics processing unit hardware now relatively mature, a fundamental question naturally arises: Can scaling data alone enable generalization in robotics?

Unlike the text corpora used to train LLMs, which are abundant and readily available on the internet, robot learning requires massive training data collected in the physical world. Currently, robot data scaling efforts follow two primary paths. The first is to collect data through human teleoperation [11, 12], but obtaining human demonstrations at LLM scale (billions to hundreds of billions of tokens) remains technically difficult, time-consuming, and costly [13, 14]. Even large industrial efforts, such as Tesla's teleoperation-assisted factory data pipeline, face fundamental bottlenecks in throughput and scalability [15]. The second path relies on synthetic data from simulation engines (e.g., Isaac Lab, MuJoCo) [16–18] and video-based generation methods (e.g., OpenAI Sora, Cosmos world model) [19–22]. Although these approaches yield performance improvements on certain robotic tasks [22–24], they remain limited in scaling robot generalization due to the persistent sim-to-real gap, prohibitive computational overhead, and the lack of standardized data formats, software, and hardware interfaces across diverse robotic platforms [10].

From a bionic point of view, the powerful generalization capabilities of living organisms do not rely solely on the scaling law [10, 25, 26]. It has been observed that innate or acquired physical structure emerges to facilitate the adaptation of living organisms to unseen environments [27–30]. For example, cortical neural responses are governed

by the synergistic integration of feedforward signals, feedback loops, and prior drives, which evolve dynamically to minimize a global energy function [28]. Likewise, larval zebrafish can integrate their innate physical integral structure and real-time state feedback with an updated cerebellar internal model to effectively respond to unpredictable visual perturbations [30]. Inspired by these insights, intelligent robots should also utilize readily accessible physics structures or other priors [13, 14, 31], instead of depending exclusively on data scaling. Different from digital LLMs, real-world robots indeed possess distinct, well-established physical structures and real-time feedback resulting from environmental interactions that can be leveraged. This leads to an interesting question: How to harness these physical architectures to improve robots' learning generalization?

We propose a novel **Phy**sics **Filter**ed learning approach, termed PhyFilter, which enhances both generalization and interpretability of robot learning. Concretely, PhyFilter operates by correcting learning outcomes according to readily accessible physical differential structure and real-time state feedback, as shown in Fig. 1 (a and b). By framing the method within a filtering-like paradigm, PhyFilter enhances interpretability and improves convergence. We also present an auto-learning algorithm capable of searching for the optimal parameters of PhyFilter, making manual tuning unnecessary. Unlike prior physics-informed methods [32–35] that typically require a retraining of the modified network, PhyFilter features a plug-and-play design, enabling direct deployment with pretrained models.

PhyFilter applies seamlessly to two prevailing learning paradigms, reinforcement learning (RL) and supervised learning (SL), and has been demonstrated to be effective in four representative robotic scenarios. Experiments include quadruped locomotion [36–42], drone maneuvering flight [43, 44], aerial manipulation [45], and acceleration perception [46], as depicted in Fig. 1c. Specifically, with the assistance of PhyFilter, a policy trained solely on simulated flat terrain can stably generalize to diverse real-world terrains on a quadruped robot. Moreover, an aerial manipulator can perform precise pick-and-place tasks with a maximum manipulation error of 2.5 cm, a 23.99% *avg.* and 55.04% *var.* improvement over baseline, despite 5 m/s wind and 0.3 kg mass uncertainties. Experimental videos and details are provided in Supplementary Information.

2 Results

2.1 Physics filtered learning

We begin by introducing the core concept of PhyFilter. Consider an unknown mapping $\boldsymbol{f}(t)$ appearing in governing physical equations. The learning residual $\boldsymbol{\gamma}(t)$ in unseen scenarios can be formalized as $\boldsymbol{\gamma}(t) = \boldsymbol{f}(t) - \boldsymbol{f}_{\boldsymbol{\theta}}(t)$, with the truth model $\boldsymbol{f}(t)$ and learned model $\boldsymbol{f}_{\boldsymbol{\theta}}(t)$ parameterized by parameter $\boldsymbol{\theta}$.

Intuitively, if real-time knowledge of $\boldsymbol{\gamma}(t)$ is accessible, the model output could be corrected online as $\boldsymbol{f}_{\boldsymbol{\theta}}(t) + \boldsymbol{\gamma}(t)$. In practice, however, the absence of labels during deployment makes this correction infeasible, as $\boldsymbol{\gamma}(t)$ cannot be directly inferred. Even natural organisms are unable to instantly interpret unfamiliar environments, but instead undergo an adaptation process. Previous studies have discovered a low-pass

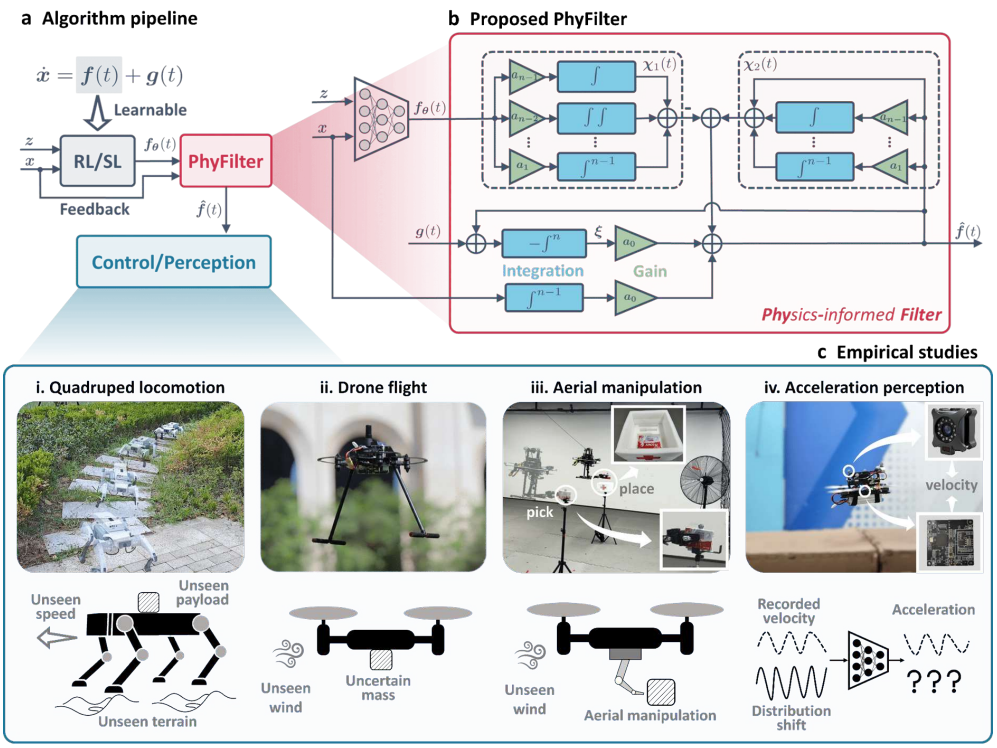

Fig. 1 Overview of our PhyFilter. a, The learning output of RL or SL is enhanced by a physics-informed filter, which helps narrow the large generalization gap. **b**, The implementation details of PhyFilter, which is analytic and convergent. Additional theoretical details are provided in the "Filtering learning residual" section in Methods. **c**, PhyFilter is validated across four representative examples. **i**, Policy learning for quadruped locomotion. A quadruped robot trained in a simplified simulator generalizes to unseen speeds, payload variations, and real-world terrains when equipped with PhyFilter. **ii**, Dynamical learning for drone maneuvering flight. A learning-based model captures mass variations, while PhyFilter helps handle unseen wind disturbance. **iii**, Dynamical learning for aerial manipulation in a pick-and-place task. A dedicated model learns drone-manipulator coupling uncertainties, with PhyFilter aiding in the handling of unseen wind disturbances and mass uncertainties. **iv**, Kinematics learning for acceleration perception. A neural network differentiator maps velocity sequences to real-time acceleration, and PhyFilter improves its robustness to distribution shift.

filtering feature in human perception systems, such as the vision [47] and hearing [48]. These observations suggest that filtering may underlie the adaptive mechanisms enabling humans to remain robust under environmental variations. In this work, we try to answer whether a similar scheme can be realized in the robot learning field, i.e.,

$$\hat{\boldsymbol{f}}(t) = \boldsymbol{f}_{\boldsymbol{\theta}}(t) + \mathcal{F}(\boldsymbol{\gamma}(t)) \tag{1}$$

where $\hat{\boldsymbol{f}}(t)$ represents the corrected learning outcome and $\mathcal{F}(\cdot)$ denotes a user-defined low-pass filter. If Eq. (1) can be achieved, the low-frequency characteristic of learning

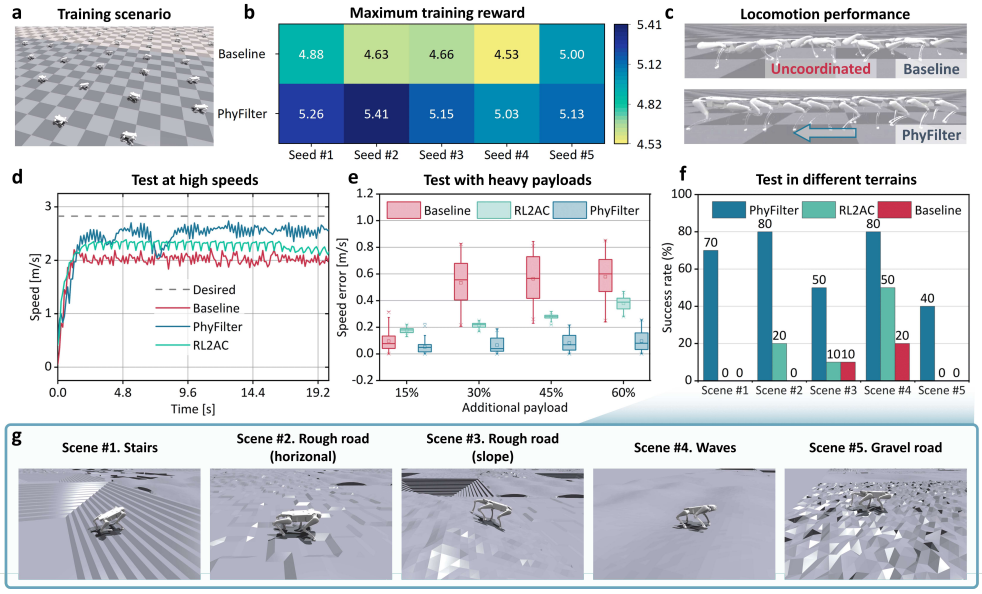

Fig. 2 Simulation results of the quadruped example. a Parallel training setup using only flat terrain. **b**, Maximum training rewards over five random seeds. The number of iteration and training environment are set as 15000 and 256, respectively. **c**, Locomotion performance of the baseline and PhyFilter in the training environment. **d**, Testing performance at an unseen high speed of 2.8 m/s. **e**, Testing performance under unseen payloads ranging from 15% $\sim$ 60% of the robot's 11.86 kg weight. **f**, Testing performance across five unseen scenarios. **g**, Stairs, rough, wave, and gravel roads, unseen in training, are considered during deployment.

residual can be captured convergently and compensated timely, enabling the refinement of learning outcomes. In this sense, the convergence process of the low-pass filter can be regarded as an adaptive procedure in unseen scenarios, like living organisms in nature. In this work, we show that the objective in Eq. (1) can be fully realized by utilizing real-time state feedback and *a priori* structure of the differential equation, such as kinematics or dynamics. Its implementation is detailed in the "Filtering learning residual" section in Methods. We also show that the filter parameters can be auto-learned from an optimal-control perspective ("Learning filter parameter" section in Methods).

2.2 Learning policy for quadruped locomotion

Locomotion control of quadruped robots has garnered substantial interest in recent years. Notably, RL-based methods have achieved impressive robustness on challenging terrains by virtue of parallel training and domain randomization strategies [36–42]. In this paper, we will show that incorporating PhyFilter into the RL training pipeline not only increases maximum training rewards but also significantly enhances generalization to diverse, previously unseen real-world environments, even when trained only in parsimonious scenarios.

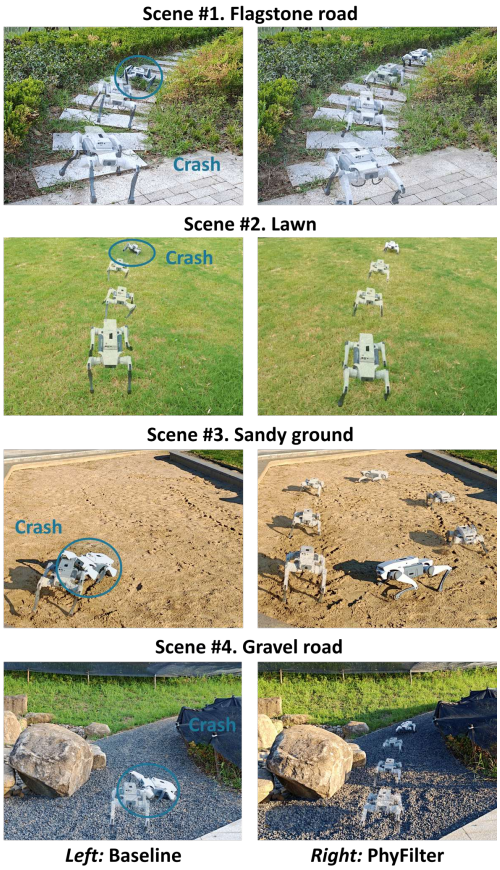

Fig. 3 Experimental results of the quadruped example. Several unseen scenarios including flagstone, lawn, sand, and gravel terrains are evaluated. Notably, the policy is trained solely on simulated flat terrain with no post-deployment tuning. See Supplementary Movie S1 for more visual results.

The parallel deep learning training strategy presented in [42] is adopted as the baseline. Desired joint positions are inferred from proprioceptive observations and body commands, while the proposed PhyFilter is integrated into the joint-space control layer. Additional implementation details are provided in the "Implementation details of quadruped example" section in Methods. To emphasize the generalization ability of the PhyFilter-enhanced RL scheme, the training environment in simulation only consider a flat terrain (Fig. 2a) and narrow parameter randomization ranges (e.g., $0 \sim 1.13$ m/s speed, $-1 \sim 3$ kg payload). During employment, however, the robot encounters broader disturbances, including unseen internal dynamical parameters and diverse unstructured terrains. This setting highlights how the proposed method reduces the randomization burden of conventional RL while maintaining remarkable generalization ability. See Supplementary Movie S2 for simulational details.

Fig. 2b shows the training performance from five random seeds for both the baseline and the proposed method. PhyFilter consistently achieves higher maximum rewards. Fig. 2c further illustrates that PhyFilter exhibits a more coordinated locomotion, whereas baseline displays degraded gait quality. This improvement likely arises from the filtering mechanism, which effectively compensates for unknown uncertainties and facilitates policy training. This observation suggests that incorporating a robust low-level controller can substantially enhance the learning efficiency of high-level RL policies. Supplementary Fig. S3 provides more comparison results.

Subsequently, we further evaluate generalization during deployment under high speed (2.83 m/s, 98.27% beyond the training-set maximum), varying payloads (15% $\sim$ 60% of robot weight 11.86 kg, 137.33% beyond the training-set maximum), and diverse unseen terrains (stairs, rough, wave, and gravel roads, Fig. 2g). Besides the baseline, RL2AC [49], which combines an adaptive controller with RL to mitigate the sim-to-real gap, is also compared. For speed tracking (Fig. 2d), PhyFilter achieves superior accuracy relative to the baseline and RL2AC. With respect to payload tests (Fig. 2e), the tracking performance of the baseline and RL2AC degrades with increasing payload, whereas PhyFilter maintains a smaller tracking error (≤ 0.2 m/s) even under 60% of robot weight payload. For terrain adaptation (Fig. 2f), since only flat terrain is considered during training, the baseline struggles to adapt to new terrains as expected. However, the proposed algorithm exhibits excellent terrain adaptation. Across five different scenarios, the passing success rates of the proposed algorithm are consistently higher than those of the baseline and RL2AC, even though these scenarios have never been seen before.

Finally, the trained policy is directly deployed on the real hardware (Fig. S4) to evaluate the sim-to-real transfer performance, with a particular focus on unseen terrains. As shown in Fig. 3, the quadruped traverses four challenging natural terrains without additional tuning. The baseline exhibits poor adaptability and collapses immediately on sandy and gravel surfaces. In stark contrast, the PhyFilter-enhanced policy enables stable locomotion across all tested terrains, despite being trained exclusively on simulated flat ground. To recap, with limited randomized training data, PhyFilter can not only improve the RL training performance, but also dramatically enhance its generalization ability using simplified physical knowledge ("Implementation details of quadruped example" section in Methods).

2.3 Learning dynamics for drone maneuvering flight

Beyond RL, SL has also been widely applied across diverse robotic domains. In our second case, we integrate PhyFilter into an SL-based controller on a drone to evaluate its ability to enhance generalization under previously unseen disturbances. Notably, PhyFilter is used in a plug-and-play manner in this case. Mass uncertainties usually appear in payload-transport missions for drones. SEER-I, a learning-based adaptive estimator developed by [43], can effectively estimate mass uncertainty by combining offline basis learning with online adaptive control. However, the SEER-I generalizes poorly to other kinds of unseen disturbances, such as wind. In this part, we augment SEER-I with physics knowledge to assess how PhyFilter handles unseen disturbances.

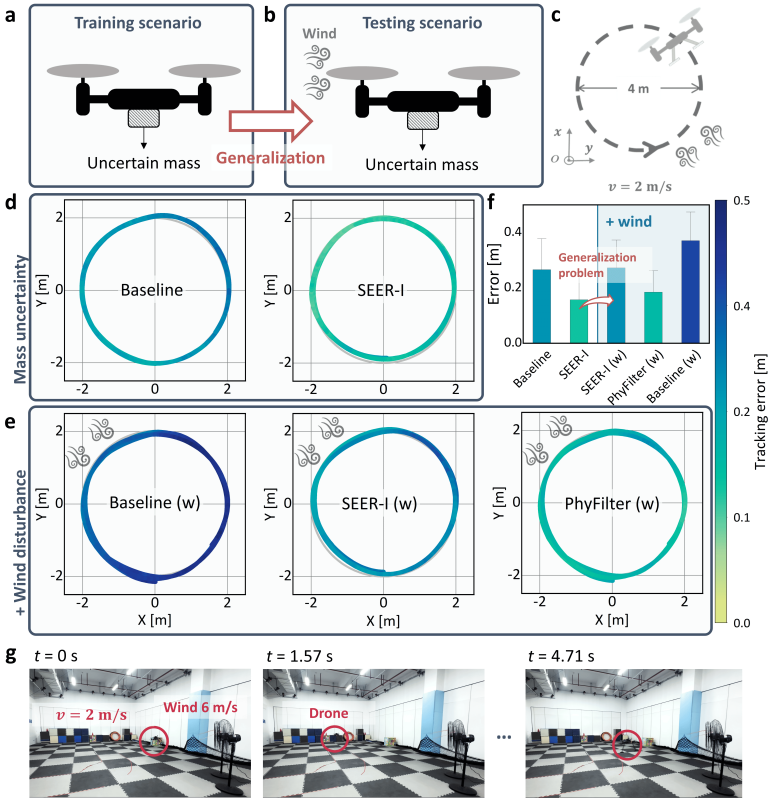

Fig. 4 Experimental results of the drone maneuvering example. a, The employed SEER-I is trained under mass uncertainty, while the testing scenario further includes the wind disturbance, as depicted in **b**. **c**, In tests, the drone is commanded to follow a circular trajectory with 2 m/s. **d**, The tracking performance with only mass uncertainty under the baseline and SEER-I. **e**, The tracking performance with both mass uncertainty and wind disturbance under the baseline, SEER-I, and PhyFilter. **f**, Mean absolute tracking errors across all implemented tests. The error bar indicates the tracking errors presented in (**d**) and (**e**). **g**, Several flight snapshots.

More details about the implementation of SEER-I on the drone can be found in the "Implementation details of drone flight example" section in Methods.

In tests, the drone is commanded to follow a circular trajectory with 2 m/s (Fig. 4c) under different uncertainties (Supplementary Movie S3). We first assess the learned SEER-I model under a single mass uncertainty. As shown in Fig. 4d, the SEER-I can achieve better tracking performance compared with the baseline [44]. However, when the wind disturbance is further included, from Fig. 4 (e and f), the tracking performance of SEER-I is degraded, which indicates its limited ability to generalize to unseen uncertainties. After the presented PhyFilter is employed, the mean absolute tracking error can be reduced by 30.22% relative to SEER-I and 50.17% relative to the baseline. Representative flight snapshots are shown in Fig. 4g. The uncertainty estimation performance of SEER-I and PhyFilter is further provided in Supplementary

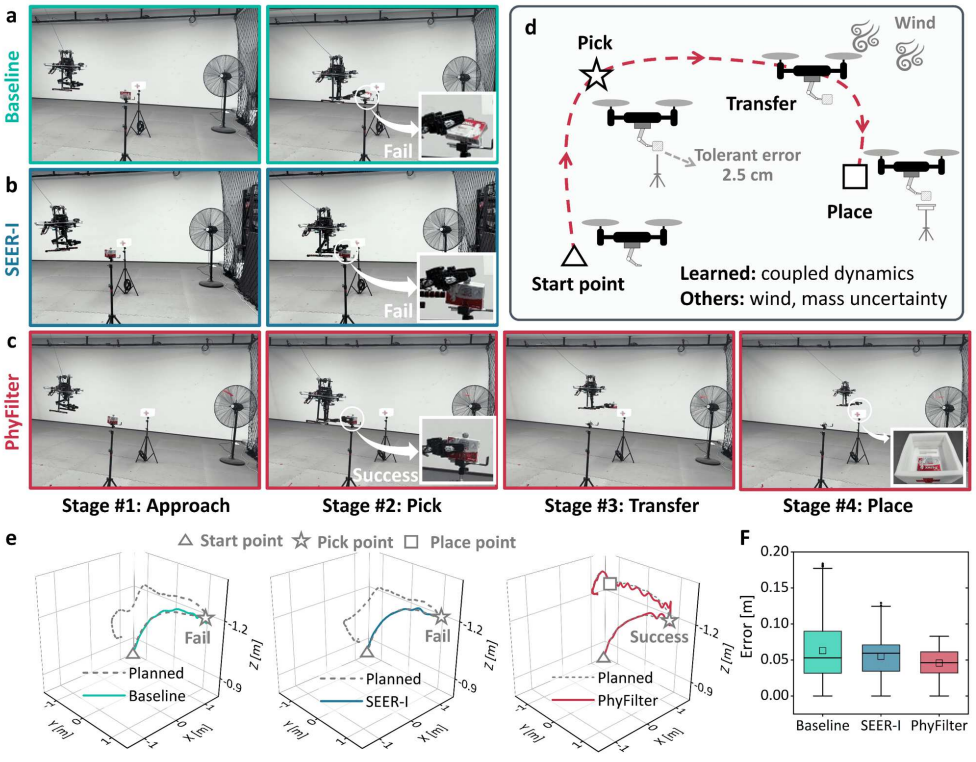

Fig. 5 Experimental results of the aerial manipulation example. a-c, Pick-and-place executions under the baseline, SEER-I, and PhyFilter, respectively. **d**, Illustration of the pick-and-place task involving coupled dynamics, wind disturbance, and mass uncertainty. **e**, Three-dimensional tracking performance of the compared methods. **f**, Tracking errors for each method.

Fig. S1, which shows that SEER-I captures only mass variations, whereas PhyFilter additionally identifies wind-induced and inherent dynamical uncertainties. To sum up, these results suggest that PhyFilter can improve the generalization ability of SL models by leveraging real-time feedback state and dynamical structure.

2.4 Learning dynamics for aerial manipulation

Compared with conventional multirotor platforms, aerial manipulators offer substantially greater flight flexibility and operational dexterity, enabling deployment in complex interactive tasks such as component transport, infrastructure repair, and obstacle removal [45]. High-precision end-effector control is a fundamental prerequisite for the deployment of aerial manipulators. However, achieving such precision is challenging due to the presence of strong coupling dynamics caused by the manipulation movement, along with external disturbances [45, 50, 51].

To evaluate the compatibility of PhyFilter in such precision-critical interaction scenarios, we design a pick-and-place mission. As illustrated in Fig. 5d, the aerial

manipulator is expected to precisely pick a medicine from a tripod and place it into the specified area. The maximum allowable tracking error of the end-effector is 2.5 cm. See Supplementary Movie S4 for experimental scenarios. The pick-and-place reference trajectory is computed in real time using a model predictive control strategy ("Trajectory generation of aerial manipulation" section in Supplementary Information). To mimic realistic environmental disturbances, a 380 W fan is used to generate a maximum wind speed of up to 5 m/s. Additionally, an unmodeled 10% mass uncertainty (0.3 kg) is introduced to replicate real-world payload change. These factors constitute a fundamental active interaction task in the emergency rescue field.

Fig. 5 (a-c) presents representative snapshots corresponding to three distinct control schemes: the baseline controller, SEER-I [43], and the proposed PhyFilter. SEER-I is an offline-trained learning-based model that captures coupled aerial manipulation dynamics using collected states and ground-truth disturbance labels. Implementation details for the baseline and SEER-I are provided in the "Implementation details of aerial manipulation example" section in Methods. PhyFilter is applied on top of SEER-I to enhance generalization to unseen disturbances, including external wind and mass uncertainty. The resulting three-dimensional (3D) tracking performance and tracking errors of these methods are visualized in Fig. 5 (e and f). As observed from the experimental results, both the baseline and SEER-I fail to capture the object due to the involvement of unseen wind and mass uncertainties. Notably, owing to its incorporation of the coupled dynamics model, SEER-I exhibits marginally superior tracking performance compared to the baseline. In stark contrast, PhyFilter markedly improves the generalization capability of SEER-I, enabling the aerial manipulator to maintain centimeter-level manipulation and perform the pick-and-place task.

2.5 Learning kinematics for acceleration perception

Precisely estimating robot acceleration is particularly challenging for resource-constrained platforms such as lightweight drones, which typically lack additional high-fidelity perceptive sensors [43]. The neural network-based differentiators have shown promising performance for state perception [46]. However, as with other neural network-based application scenarios, the generalization problem is an intractable difficulty that cannot be bypassed. In this section, we apply PhyFilter to enhance the robustness and generalization of learned acceleration estimates. Additional theoretical and implementation details are provided in the "Implementation details of perception example" section in Methods.

The training set ($0 \sim 2$ m/s) and testing set ($0 \sim 3$ m/s) are collected from real flight experiments, as shown in Fig. 6 (a, b, and f). A fully connected neural network with two hidden layers is first trained in a supervised manner to map historical velocities to acceleration. Subsequently, PhyFilter using the knowledge of kinematic structure, is employed to enhance the learning output in the face of the testing set with distribution shift. Several state-of-the-art benchmarks, including robust differentiator (RD) [52] with a predefined convergence time, tracking differentiator (TD) [53], and discrete differentiator (DD) are also implemented as compared methods. All baselines are carefully tuned for their best performance [54]. As shown in Fig. 6 (c and d), the purely learning-based model exhibits generalization failure in the testing set.

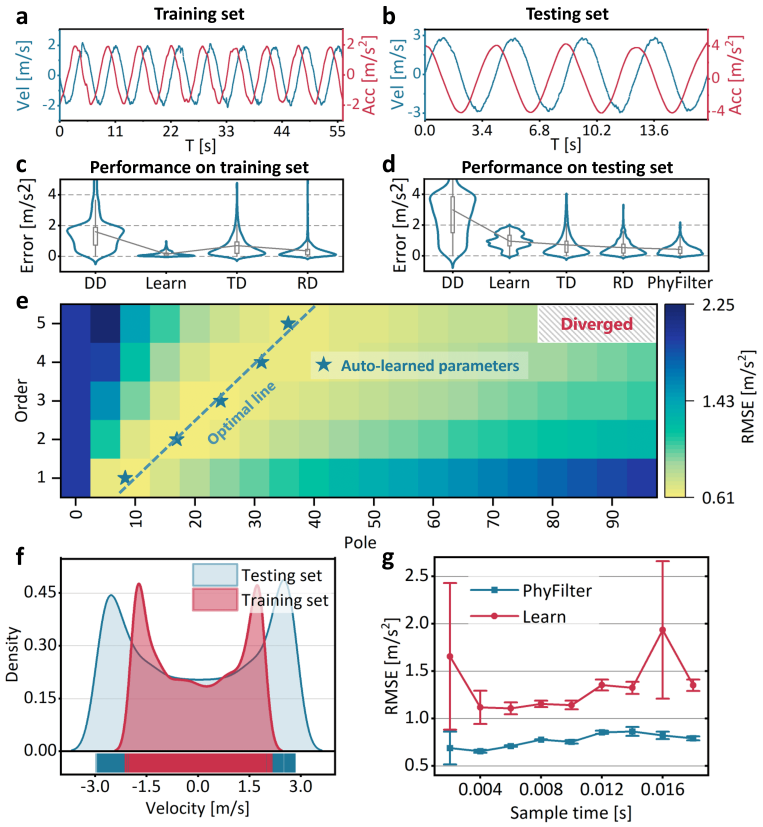

Fig. 6 Experimental results of the accelerator example. a, Training dataset containing velocities from 0 to 2 m/s. **b**, Testing dataset containing velocities from 0 to 3 m/s. **c**, Absolute prediction errors of each method on the training set. **d**, Absolute prediction errors on the testing set. **e**, Ablation study on filtering orders and parameters determined via *pole-placement*. Blue stars denote the auto-learned parameters, which are roughly consistent with manually optimal results. **f**, Distribution of the training and testing datasets. **g**, Ablation study across different sampling intervals. RMSE denotes the root mean square error.

However, it can be seen that PhyFilter achieves the highest accuracy, outperforming both learning-based and classical differentiator baselines.

Ablation experiments on filter order and pole selection ("Filtering learning residual" section in Methods) are summarized in Fig. 6e. The results indicate that higher-order filters benefit from larger pole values. Moreover, an auto-learning procedure ("Learning filter parameter" section in Methods) is further developed to discover the optimal filter parameters. As can be observed from Fig. 6e, the auto-learned optimal parameters are roughly consistent with the manually adjusted ones, showing the effectiveness of the proposed strategy. The convergence performance at different filter orders is provided in Supplementary Fig. S2. Since differentiator performance depends on sampling rate, we additionally examine a range of sampling intervals, as presented

in Fig. 6g. The proposed PhyFilter consistently outperforms the purely learning-based one, ranging from 0.002 s to 0.018 s in the testing set.

3 Disscusion

Stemming from the adaptive mechanisms observed in living organisms, we presented a physics-filtered learning framework aiming at alleviating the long-standing generalization challenge in robot learning. To achieve low-pass filtering of learning residuals (Eq. (1))—mimicking biological systems—we derived an equivalent implementation (Eq. (5)) that leverages real-time state feedback and the intrinsic differential structure of physical models. To eliminate the need for manual parameter tuning, an automated learning algorithm was further developed. Overall, PhyFilter exhibited the advantages of generalization and interpretability.

Four realistic robotic experiments were arranged to validate the effectiveness of PhyFilter. When incorporated with RL strategies for quadrupedal locomotion, PhyFilter not only enhanced training rewards but also substantially improved generalization. Notably, even if the RL policy was trained solely on simulated flat terrain, it could stably generalize to unseen realistic terrains. Additionally, PhyFilter boosted the generalization capability of SL methods. Flight tests demonstrated that, with PhyFilter, drones and aerial manipulators maintained satisfactory control precision while encountering unseen uncertainties. We also demonstrated that PhyFilter also strengthened perception modules, enabling accurate acceleration estimation despite input distribution shifts. Note that PhyFilter is applied in a plug-and-play manner in the last three experiments, underscoring its model-agnostic feature.

3.1 PhyFilter unifies generalization and accuracy

To bridge the simulation-to-reality gap, the prevailing paradigm relies on domain randomization [23, 55], which stochastically randomizes the parameters of the simulator to encompass real-world uncertainties. This approach pursues the average performance among randomized environments, i.e., sacrificing accuracy for robustness [56–58], similar to classical robust control [57, 58]. In contrast, PhyFilter unified generalization and accuracy in one framework via a physics-informed feedback mechanism [56]. Specifically, when encountering scenarios outside the training distribution, PhyFilter was activated to mitigate uncertainty and enhance generalization. Conversely, within familiar training regimes, where learning residuals remain small, its intervention was adaptively attenuated, thereby preserving the intrinsic precision of the original neural network.

3.2 PhyFilter lightens the learning burden

The enhanced learning strategy no longer requires extensive domain randomization in the training phase while still maintaining excellent generalization during deployment. In the quadruped experiment, although integrating PhyFilter increased RL training time due to the computational complexity of evaluating the 12×12 simplified dynamics model, it eliminated the need to account for complex terrains, payloads, and speeds

during training. During deployment, PhyFilter required only a small set of analytic update equations (Fig. 1b). In the cases of drone flight and aerial manipulator, PhyFilter ran in real time on an STM32F765 microprocessor at 500 Hz, demonstrating its suitability for lightweight, resource-constrained platforms.

3.3 Limitation and future work

Two limitations exist in this study that also highlight directions for future development. First, the beneficial knowledge obtained through PhyFilter only enhances the learning output. A more intelligent strategy should involve leveraging the filtered knowledge to refine the parameter or structure of the original neural network. Continual learning [59] may be an integrable approach to achieve this goal. Secondly, the current auto-learning algorithm for tuning the filter parameters is computationally inefficient, as each iteration requires a forward rollout of the entire trajectory. Future work will explore approaches to improve the efficiency of parameter learning. In practice, the manual tuning is found to be intuitive and efficient for anyone with basic *pole-placement* knowledge.

Moreover, while PhyFilter is validated across four diverse robotic scenarios, current experiments mainly focus on limited data and small model capacity. This setting demonstrates PhyFilter's efficacy under challenging, resource-limited conditions; however, it remains unclear how robot behavior might evolve with larger models or more extensive datasets. Understanding how PhyFilter interacts with scaling laws is a critical direction for future work.

4 Methods

4.1 Filtering learning residual

In this part, the theoretical framework of PhyFilter is established. Before proceeding, consider a general differential structure

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{z}) + \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{z}) \tag{2}$$

with internal system state $\boldsymbol{x} \in \mathbb{R}^{n_x}$, external influencing factor $\boldsymbol{z} \in \mathbb{R}^{n_z}$, known nonlinear mapping $\boldsymbol{g}(\cdot) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_x}$, and unknown nonlinear part $\boldsymbol{f}(\cdot) : \mathbb{R}^{n_x} \times \mathbb{R}^{n_z} \to \mathbb{R}^{n_x}$. Note that the above structure can represent unknown dynamics and kinematics in robotic systems. Unless otherwise specified, $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{z})$ and $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{z})$ are simplified as $\boldsymbol{g}(t)$ and $\boldsymbol{f}(t)$, respectively, by considering the time-varying features of $\boldsymbol{x}(t)$ and $\boldsymbol{z}(t)$. It is assumed that $\boldsymbol{f}(t)$ can be learned through neural networks or kernel-based regression methods with parameter $\boldsymbol{\theta} \in \mathbb{R}^{n_\theta}$, denoted as $\boldsymbol{f}_{\boldsymbol{\theta}}(t)$, with learning residual $\boldsymbol{\gamma}(t) \in \mathbb{R}^{n_x}$, i.e., $\boldsymbol{\gamma}(t) = \boldsymbol{f}(t) - \boldsymbol{f}_{\boldsymbol{\theta}}(t)$.

Next, we show that the objective in Eq. (1) can be realized from the filtering perspective. The direct implementation of Eq. (1) is not feasible because $\boldsymbol{\gamma}(t)$ cannot be accessed without additional sensing. However, by leveraging available physical structure and real-time state feedback, we demonstrate that Eq. (1) can be sufficiently achieved. The proposed formulation can also be extended to more general classes of

filters. More details can be found in the "Filtering learning residual with more general form" section in Supplementary Information.

Consider an nth-order low-pass filter with the following transfer function

$$\mathcal{L}\left[\mathcal{F}(\boldsymbol{\gamma}(t))\right] = \frac{a_0}{s^n + a_{n-1}s^{n-1} + \cdots + a_1 s + a_0}\Upsilon(s) \tag{3}$$

with *Laplace* transform $\mathcal{L}[\cdot]$, *Laplace* variable s, parameters $a_{n-1}, \cdots, a_1, a_0$, and frequency-domain expression $\Upsilon(s)$ of $\boldsymbol{\gamma}(t)$. By substituting Eq. (3) into Eq. (1) and performing the result in time-domain, it can be rendered that

$$\begin{aligned}&\hat{\boldsymbol{f}}^{(n)}(t) + a_{n-1}\hat{\boldsymbol{f}}^{(n-1)}(t) + \cdots + a_1\hat{\boldsymbol{f}}'(t) + a_0\hat{\boldsymbol{f}}(t)\\ =&\boldsymbol{f}_{\boldsymbol{\theta}}^{(n)}(t) + a_{n-1}\boldsymbol{f}_{\boldsymbol{\theta}}^{(n-1)}(t) + \cdots + a_1\boldsymbol{f}_{\boldsymbol{\theta}}'(t) + a_0\boldsymbol{f}_{\boldsymbol{\theta}}(t) + a_0\boldsymbol{\gamma}(t)\end{aligned}$$

where $(\cdot)^{(n)}$ denotes the nth-order differentiation of a signal. For simplicity, define $\boldsymbol{\chi}_1(t) = a_{n-1}\int\hat{\boldsymbol{f}}(t)dt + \cdots + a_1\int^{n-1}\hat{\boldsymbol{f}}(t)(dt)^{n-1}$ and $\boldsymbol{\chi}_2(t) = \boldsymbol{f}_{\boldsymbol{\theta}}(t) + a_{n-1}\int\boldsymbol{f}_{\boldsymbol{\theta}}(t)dt + \cdots + a_1\int^{n-1}\boldsymbol{f}_{\boldsymbol{\theta}}(t)(dt)^{n-1}$, where $\int^i(\cdot)(dt)^i$ denotes the ith-order integration of a signal. Thus, we have

$$\hat{\boldsymbol{f}}^{(n)}(t) + \boldsymbol{\chi}_1^{(n)} + a_0\hat{\boldsymbol{f}}(t) = \boldsymbol{\chi}_2^{(n)} + a_0\boldsymbol{f}_{\boldsymbol{\theta}}(t) + a_0\boldsymbol{\gamma}(t).$$

Due to that $\boldsymbol{\gamma}(t) = \boldsymbol{f}(t) - \boldsymbol{f}_{\boldsymbol{\theta}}(t)$ and physical structure in Eq. (2), one can imply

$$\hat{\boldsymbol{f}}^{(n)}(t) + \boldsymbol{\chi}_1^{(n)} + a_0\hat{\boldsymbol{f}}(t) = \boldsymbol{\chi}_2^{(n)} + a_0\boldsymbol{f}_{\boldsymbol{\theta}}(t) + a_0(\dot{\boldsymbol{x}} - \boldsymbol{g}(t) - \boldsymbol{f}_{\boldsymbol{\theta}}(t)). \tag{4}$$

It can be seen $\hat{\boldsymbol{f}}^{(n)}(t)$, $\boldsymbol{\chi}_1^{(n)}$, $\boldsymbol{\chi}_2^{(n)}$, and $\dot{\boldsymbol{x}}$ in Eq. (4) are unknown or strenuous to be accurately obtained. Define an auxiliary variable $\boldsymbol{\xi} = (\hat{\boldsymbol{f}}(t) + \boldsymbol{\chi}_1 - \boldsymbol{\chi}_2 - a_0\int^{n-1}\boldsymbol{x}(dt)^{n-1})/a_0$, leading to $a_0\boldsymbol{\xi}^{(n)} = \hat{\boldsymbol{f}}^{(n)}(t) + \boldsymbol{\chi}_1^{(n)} - \boldsymbol{\chi}_2^{(n)} - a_0\dot{\boldsymbol{x}}$. Thus, it can be obtained that

$$\begin{cases}\boldsymbol{\xi}^{(n)} = -\boldsymbol{g}(t) - \hat{\boldsymbol{f}}(t)\\ \hat{\boldsymbol{f}}(t) = a_0\boldsymbol{\xi} - \boldsymbol{\chi}_1 + \boldsymbol{\chi}_2 + a_0\int^{n-1}\boldsymbol{x}(dt)^{n-1}.\end{cases} \tag{5}$$

By introducing a new variable $\boldsymbol{\xi}$, the direct use of $\hat{\boldsymbol{f}}^{(n)}(t)$, $\boldsymbol{\chi}_1^{(n)}$, $\boldsymbol{\chi}_2^{(n)}$, and $\dot{\boldsymbol{x}}$ in Eq. (4) is avoided. The underlying principle is to reconstruct lower-order signal evolution by combining higher-order signal differences with lower-order initial conditions. The resulting analytic implementation of Eq. (5) is portrayed in Fig. 1b.

Up to now, the objective stated in Eq. (1) has been reformulated into the implementable form of Eq. (5), relying solely on accessible measurements. In the case where the low-pass filter is configured as a first-order one, Eq. (5) will roughly equal to EVOLVER [54], SEER-II [43], and feedback neural network [56], in which the *Koopman* operator, *Chebyshev* polynomials, and neural ordinary differential equations (neural ODEs) [60] are respectively utilized to learn $\boldsymbol{f}(t)$ with specific profitable

characteristics ("Several special forms" section in Supplementary Information). Furthermore, if the learned knowledge is additionally disregarded, Eq. (5) ultimately degenerates to a purely feedback-based disturbance observer [61].

4.2 Learning filter parameter

To eliminate the reliance on domain-specific expertise typically required by traditional *pole-placement* theories, we formulate an optimal control problem to determine filter parameters. By restructuring the observer dynamics, the filter parameters are redefined as control inputs, while the system states and their successive higher-order integrals are concatenated into an augmented state space. The learning objective is centered on minimizing a cost functional that quantifies the discrepancy between the model's predicted rollouts and the ground-truth labeled samples. This formulation effectively transforms the task of parameter tuning into the pursuit of an optimal control law subject to the underlying system dynamics.

The optimization is implemented using the *Lagrange* multiplier method to derive the first-order optimality conditions. To ensure computational tractability, particularly for high-dimensional systems, an iterative scheme is employed wherein analytical gradients are computed through a sequence of forward state rollouts followed by backward adjoint solves (*reverse-mode differentiation*) [60]. To further enhance convergence stability and efficiency when processing large-scale experimental datasets, the framework integrates mini-batching and stochastic gradient descent, specifically utilizing the *Adam* optimizer, complemented by an early stopping mechanism. This end-to-end gradient-based architecture enables the autonomous extraction of optimal filter characteristics directly from dynamical data, ensuring robust state estimation. The learning performance is verified in the acceleration example, as shown in Fig. 6e. Theological details can be found in the "Formalization of parameter learning" section in Supplementary Information.

4.3 Implementation details of quadruped example

The fully-actuated *Lagrangian* dynamical model of the quadruped robot is usually described as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) - \boldsymbol{J}^T\boldsymbol{F}_c = \boldsymbol{\tau} + \Delta\boldsymbol{\tau} \tag{6}$$

with joint angle $\boldsymbol{q} \in \mathbb{R}^{12}$, inertial matrix $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{12\times12}$, *Coriolis* force $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) \in \mathbb{R}^{12\times12}$, gravity force $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^{12}$, *Jacobian* matrix $\boldsymbol{J} \in \mathbb{R}^{12\times3}$, contact force $\boldsymbol{F}_c \in \mathbb{R}^3$, joint torque input $\boldsymbol{\tau} \in \mathbb{R}^{12}$, and unknown torque uncertainty $\Delta\boldsymbol{\tau} \in \mathbb{R}^{12}$. $\Delta\boldsymbol{\tau}$ mainly results from internal model mismatches or external disturbances.

One appealing control strategy in RL framework [42, 49] is to infer a policy network from proprioceptive observations $\boldsymbol{o}$ and body commands $\boldsymbol{c}$ to desired joint positions $\boldsymbol{q}_d$, denoted as $\pi(\boldsymbol{q}_{d,t}|\boldsymbol{o}_t,\boldsymbol{c}_t)$. $\boldsymbol{o}_t$ includes the base attitude, base angular velocity, joint angles, joint angular velocities, and their historical sequences. Subsequently, a proportional-derivative (PD) controller is employed to track $\boldsymbol{q}_d$, i.e.,

$$\boldsymbol{\tau} = \boldsymbol{K}_p(\boldsymbol{q}_d - \boldsymbol{q}) - \boldsymbol{K}_d\dot{\boldsymbol{q}}$$

with gains $\boldsymbol{K}_p \in \mathbb{R}^{12\times12}$ and $\boldsymbol{K}_d \in \mathbb{R}^{12\times12}$.

The purpose of PhyFilter is to estimate uncertainties $\Delta\boldsymbol{\tau}$ using the output of RL policy and dynamical structure. By defining $\boldsymbol{x} = \dot{\boldsymbol{q}}$, $\boldsymbol{g}(t) = -\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q}) + \boldsymbol{J}^T\boldsymbol{F}_c + \boldsymbol{\tau}$, and $\boldsymbol{f}(t) = \Delta\boldsymbol{\tau}$, Eq. (6) can be adjusted to

$$\boldsymbol{M}(t)\dot{\boldsymbol{x}} = \boldsymbol{g}(t) + \boldsymbol{f}(t).$$

Compared with Eq. (2), a new term $\boldsymbol{M}(t)$ is further induced. Similar to the derivation process from Eq. (3) to Eq. (5), the PhyFilter algorithm in the locomotion case can be derived as

$$\begin{cases} \boldsymbol{\xi}^{(n)} = -\boldsymbol{g}(t) - \hat{\boldsymbol{f}}(t) \\ \hat{\boldsymbol{f}}(t) = a_0\boldsymbol{\xi} - \boldsymbol{\chi}_1 + \boldsymbol{\chi}_2 + a_0\int^{n-1}\left[\int \boldsymbol{M}(t)d\boldsymbol{x}\right](dt)^{n-1}. \end{cases} \tag{7}$$

where the auxiliary variable $\boldsymbol{\xi}$ is defined as $\boldsymbol{\xi} = (\hat{\boldsymbol{f}}(t) + \boldsymbol{\chi}_1 - \boldsymbol{\chi}_2)/a_0 - \int^{n-1}\left[\int \boldsymbol{M}(t)d\boldsymbol{x}\right](dt)^{n-1}$. Experiments found that 1-st order form of PhyFilter (i.e., $n = 1$ in Eq. (7)) is enough to improve the generalization ability (Figs. 2 and 3).

Note that only simplified $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ are employed in tests to reduce computational complexity. Specifically, the inertial matrix $\boldsymbol{M}(\boldsymbol{q})$ is subject to a diagonalization assumption, where inertial coupling between joints is neglected, and only the diagonal terms and local inertia are retained. The *Coriolis* force $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is approximated as linear damping, with the complex velocity-coupled quadratic terms ignored. Experimental results show that despite these simplifications, the proposed PhyFilter can still significantly improve generalization performance.

Training details

The parallel deep learning training strategy presented in [42] is adopted here. The key training parameters and domain randomization ranges are listed in Supplementary Table S1. The proximal policy optimization (PPO) algorithm [62] is used, and the policy with the best training reward is finally adopted to deploy. Note that the training scenario only includes flat terrain with the purpose of highlighting its generalization ability. Moreover, the algorithm is run on a laptop with Intel(R) Core(TM) Ultra and RTX 4060 GPU. For a fair comparison, except for the backend filter mechanism, both the training and testing processes of the proposed algorithm are consistent with those of the baseline.

4.4 Implementation details of drone flight example

SEER-I [43], a supervised machine learning technique, is employed to learn the mass uncertainty of the drone, which can be formalized as

$$\Delta m\boldsymbol{a} = -\boldsymbol{\Xi}\boldsymbol{\Phi}(\boldsymbol{v}_h)\boldsymbol{\varsigma}(\Delta m) \tag{8}$$

with unknown changed mass $\Delta m \in \mathbb{R}^3$, acceleration $\boldsymbol{a} \in \mathbb{R}^3$, constant parameter matrix $\boldsymbol{\Xi} \in \mathbb{R}^{3\times(p+1)^4}$, state-related *Chebyshev* polynomial matrix $\boldsymbol{\Phi}(\cdot) \in$

$\mathbb{R}^{(p+1)^4\times(p+1)}$, historical recorded velocity $\boldsymbol{v}_h = \{\boldsymbol{v}, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_{n_h}\}$, and mass-related *Chebyshev* polynomial vector $\boldsymbol{\varsigma}(\cdot) \in \mathbb{R}^{p+1}$. The model accuracy depends on p. $\boldsymbol{\Xi}$ can be obtained offline via the least squares (LS) algorithm with labeled data. When applied to an unknown mass scenario, $\boldsymbol{\varsigma}(\Delta m)$ can be estimated via a converged adaptive law with known portion $\boldsymbol{\Xi\Phi}(\boldsymbol{v}_h)$. The decomposed structure in Eq. (8) can improve the generalization ability subject to unseen mass uncertainty.

In the presence of other kinds of uncertainties (e.g., wind), the above decomposed structure will no longer be suitable, leading to a poor estimation performance (Fig. 4 (e and f)). Here, we further filter the learning output with physics knowledge, i.e., the dynamics structure. Define $\boldsymbol{f}(t) = \Delta m\boldsymbol{a}$ with other uncertainties. Similar to the derivation procedure of Eq. (5), one can obtain

$$\begin{cases} \boldsymbol{\xi}^{(n)} = -mg\boldsymbol{e}_z - \hat{\boldsymbol{f}}(t) \\ \hat{\boldsymbol{f}}(t) = a_0\boldsymbol{\xi} - \boldsymbol{\chi}_1 + \boldsymbol{\chi}_2 + a_0\int^{n-1}\boldsymbol{x}(dt)^{n-1} \end{cases} \tag{9}$$

with nomimal mass $m \in \mathbb{R}$, acceleration of gravity $g \in \mathbb{R}$, and its direction $\boldsymbol{e}_z \in \mathbb{R}^3$. Moreover, $\boldsymbol{f_\theta}$ in $\boldsymbol{\chi}_2$ comes from the learning output of SEER-I. The training details are consistent with [43]. In experiments, it is found that a 1-st order form of PhyFilter (i.e., $n = 1$ in Eq. (9)) is enough to improve the generalization ability.

4.5 Implementation details of aerial manipulation example

The dynamics of the aerial platform, considering the strong coupling effects with a serial manipulator, can be established as:

$$\begin{aligned} m_b\dot{\boldsymbol{v}}_b &= m_b g\boldsymbol{e}_z - f\boldsymbol{b}_3 - \boldsymbol{\Delta}_F \\ \boldsymbol{J}\dot{\boldsymbol{\omega}}_b &= -\boldsymbol{\omega}_b^\times \boldsymbol{J}\boldsymbol{\omega}_b + \boldsymbol{\tau} + \boldsymbol{\Delta}_\tau \end{aligned}$$

where $g \in \mathbb{R}^+$ is the gravity and $m_b \in \mathbb{R}^+$ represents the mass of the aerial platform, $\boldsymbol{J} \in \mathbb{R}^{3\times3}$ denotes the inertia matrix of the aerial platform, $f \in \mathbb{R}$ represents the generated force of eight rotors, and $\tau \in \mathbb{R}^3$ expresses the generated torque vector. Moreover, $\boldsymbol{\Delta}_F \in \mathbb{R}^3$ and $\boldsymbol{\Delta}_\tau \in \mathbb{R}^3$ are the coupling disturbance force and torque, respectively. $\boldsymbol{\omega}_b^\times$ represents the skew-symmetric operator of $\boldsymbol{\omega}_b$. According to the centroid principle [63], the coupling disturbances can be analytically formulated as:

$$\begin{aligned} \boldsymbol{\Delta}_F &= -m_m\dot{\boldsymbol{v}}_b + m_m g\boldsymbol{e}_z - m_m\boldsymbol{R_b}\left(\boldsymbol{\omega}_b^\times\left(\boldsymbol{\omega}_b^\times\boldsymbol{P}_{bm}^B\right) + \dot{\boldsymbol{\omega}}_b^\times\boldsymbol{P}_{bm}^B + 2\boldsymbol{\omega}_b^\times\dot{\boldsymbol{P}}_{bm}^B + \ddot{\boldsymbol{P}}_{bm}^B\right) \\ \boldsymbol{\Delta}_\tau &= -\boldsymbol{J}_{\boldsymbol{mb}}^{\boldsymbol{B}}\dot{\boldsymbol{\omega}}_b - \boldsymbol{\omega}_b^\times\boldsymbol{J}_{\boldsymbol{mb}}^{\boldsymbol{B}}\boldsymbol{\omega}_b + m_m P_{bm}^B\boldsymbol{R}_{\boldsymbol{b}}^\top\left(g\boldsymbol{e}_z - \dot{\boldsymbol{v}}_b\right) - \dot{\boldsymbol{J}}_{\boldsymbol{mb}}^{\boldsymbol{B}}\dot{\boldsymbol{\omega}}_b - \boldsymbol{\omega}_b^\times\boldsymbol{L}_m^B - \dot{\boldsymbol{L}}_m^B \end{aligned} \tag{10}$$

where $m_m \in \mathbb{R}^+$ is the mass of the manipulator, $\boldsymbol{P}_{bm}^B \in \mathbb{R}^3$ is the center of mass of the manipulator, $\boldsymbol{J}_{\boldsymbol{mb}}^{\boldsymbol{B}} \in \mathbb{R}^{3\times3}$ is the inertia tensor of the manipulator along the body frame, and $\boldsymbol{L}_m^B \in \mathbb{R}^3$ describes the angular momentum of the manipulator. It can be proven that the unknown high-order term is an analytic function regarding the state and inertial parameters of the aerial manipulator. The detailed modelling analysis of the coupling disturbances can be found in [64].

Let $\eta \in \mathbb{R}^3$ be the system state and $\varphi \in \mathbb{R}^{n_\varphi}$ be the unknown parameter vector. On the basis of decomposition theorem [65], $\Delta_F(\eta, \varphi)$ can be decomposed with arbitrary accuracy as:

$$\boldsymbol{\Delta}_F(\eta, \varphi) = \boldsymbol{\Xi}_{\boldsymbol{p}} \boldsymbol{\Phi}_{\boldsymbol{p}}(\eta) \boldsymbol{\xi}_p(\varphi) + \mathcal{O}(n)$$

where $\boldsymbol{\Xi}_{\boldsymbol{p}} \in \mathbb{R}^{3\times(p+1)^{3+n_\varphi}}$ is an unknown parameter weight matrix to be learned. $\boldsymbol{\Phi}_{\boldsymbol{p}}(\eta)$ and $\boldsymbol{\xi}_p(\varphi)$ are mappings with corresponding dimensions. $p \in \mathbb{R}$ is the hyperparameter determining the decomposition accuracy. In the presence of additional uncertainties (e.g., model uncertainties and external wind), the aforementioned decomposed structure is no longer suitable, resulting in a poor estimation performance. Hence, the learning output is further filtered via physics knowledge, i.e., the dynamic structure. The filter framework can be expressed in the same form as Eq. (9) by treating $\boldsymbol{f}(t) = \boldsymbol{\Delta}_F(\eta, \varphi)$ with other uncertainties.

Similarly, the coupling torque can be decomposed as:

$$\boldsymbol{\Delta}_\tau(\eta, \vartheta) = \boldsymbol{\Xi}_{\boldsymbol{\tau}} \boldsymbol{\Phi}_{\boldsymbol{\tau}}(\eta) \boldsymbol{\xi}_\tau(\vartheta) + \mathcal{O}(n)$$

where $\boldsymbol{\Xi}_{\boldsymbol{\tau}} \in \mathbb{R}^{3\times(p+1)^{3+n_\vartheta}}$ is the learned parameter weighting matrix. $\boldsymbol{\Phi}_{\boldsymbol{\tau}}(\eta)$ and $\boldsymbol{\xi}_\tau(\vartheta)$ are mappings with corresponding dimensions, which can consist of the known Chebyshev polynomials. The filter framework for the rotational loop can be derived similarly to Eq. (9).

Training details

During the process of collecting training data, the manipulator is commanded to follow a circular trajectory designed to excite the nonlinear coupling dynamics continuously. The reference trajectory is defined as $\left[0.28 + 0.1\sin(\omega t)/\sqrt{5}, 0.1\sin(\omega t)/\sqrt{5/4}, 0.105 + 0.1\cos(\omega t)\right]$ m in its base frame, which forms a three-dimensional periodic path covering both vertical and horizontal planes. To further enrich the dynamic excitation, the angular frequency of the trajectory is gradually increased from 2.5 to 4.5 rad/s. It is ensured that all joint dynamics of the manipulator are sufficiently excited. Moreover, Eq. (10) implies that the nonlinear coupling effects are associated with these signals $\dot{\boldsymbol{v}}_b$, $\dot{\boldsymbol{\omega}}_b$, $\dot{\boldsymbol{P}}^b_{bm}$, and $\ddot{\boldsymbol{P}}^b_{bm}$ that cannot be accurately measured by onboard sensing. Therefore, the time-delay embedding theorem [66, 67] is used in training, that is, the historical records of signal can reflect its differentiation. Concretely, the collected dataset includes linear velocity, attitude, angular velocity, joint position, and joint velocity, along with three historical values of each signal to encode their derivative information. Other training settings follow those used in the preceding drone example.

4.6 Implementation details of perception example

In the absence of a high-performance accelerometer for drones, the acceleration can be obtained through historical velocities, i.e.,

$$\boldsymbol{a} = \boldsymbol{f}(\boldsymbol{v}_h)$$

with acceleration $\boldsymbol{a} \in \mathbb{R}^3$, historical recorded velocity $\boldsymbol{v}_h = \{\boldsymbol{v}, \boldsymbol{v}_1, \cdots, \boldsymbol{v}_{n_h}\} \in \mathbb{R}^{3n_h}$, and a mapping $\boldsymbol{f}(\cdot)$. For example, $\boldsymbol{f}(\cdot)$ can be designed as the discrete differentiator (DD), tracking differentiator (TD) [53], robust differentiator (RD) [52] with a predefined convergence time, or neural network-based accelerator [46].

To further enhance the generalization of the neural network-based accelerator, we filter its output using the underlying kinematic structure, i.e.,

$$\begin{cases} \boldsymbol{\xi}^{(n)} = -\hat{\boldsymbol{f}}(t) \\ \hat{\boldsymbol{f}}(t) = a_0\boldsymbol{\xi} - \boldsymbol{\chi}_1 + \boldsymbol{\chi}_2 + a_0 \int^{n-1} \boldsymbol{x}(dt)^{n-1} \end{cases}$$

with $\boldsymbol{f_\theta}$ in $\boldsymbol{\chi}_2$ comes from the learning output of the learned neural network. The ablation study on different orders n and parameters $\boldsymbol{a}_f$ is conducted, as shown in Fig. 6e. The parameters $\boldsymbol{a}_f$ can be set according to the *pole-placement* theory or the proposed auto-learned algorithm (Algorithm 2).

Training details

The employed neural network has two hidden layers with 20 hidden units each. The *trainscg* optimizer is used. In Fig. 6 (c-e), the training datasets consist of 28000 samples within $0 \sim 2\ m/s$, while the testing datasets consist of 8400 samples within $0 \sim 3\ m/s$. All compared methods are tuned to their best performance [54].

Data availability. All data needed to evaluate the conclusions in the paper are present in the paper and/or the Supplementary Information.

Code availability. The codes that are used in this study are available in our project site: https://scoardyy.github.io/PhyFilter/.

Acknowledgements. This work was supported in part by the National Natural Science Foundation of China under Grant 62425302, and in part by the Ministry of Education, Singapore, under Tier 1 Grant RG83/25 and Grant RS36/24, and a Start-up Grant from Nanyang Technological University.

Author contributions. J.J. raised the main idea of PhyFilter and contributed to experiments and manuscript writing. Quadruped experiments were conducted with the contributions from S.H.. M.W. designed and implemented the aerial manipulation test. K.G. and J.Y. also gave advisory suggestions for the experimental arrangements. G.L., Z.Y., S.Z., X.Y., W.W., and L.G. revised the manuscript and offered some key suggestions about the manuscript presentation. L.G. directed the research.

Competing interests. The authors declare no competing interests.

References

[1] OpenAI: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)

[2] Gemini Team, G.: Gemini: A family of highly capable multimodal models. arXiv preprint arXiv:2312.11805 (2023)

[3] Guo, D., Yang, D., Zhang, H., Song, J., Wang, P., Zhu, Q., Xu, R., Zhang, R., Ma, S., Bi, X., *et al.*: Deepseek-R1 incentivizes reasoning in LLMs through reinforcement learning. Nature **645**(8081), 633–638 (2025)

[4] Bahri, Y., Dyer, E., Kaplan, J., Lee, J., Sharma, U.: Explaining neural scaling laws. Proceedings of the National Academy of Sciences **121**(27), 2311878121 (2024)

[5] LeCun, Y., Bengio, Y., Hinton, G.: Deep learning. Nature **521**(7553), 436–444 (2015)

[6] Wang, H., Fu, T., Du, Y., Gao, W., Huang, K., Liu, Z., Chandak, P., Liu, S., Van Katwyk, P., Deac, A., *et al.*: Scientific discovery in the age of artificial intelligence. Nature **620**(7972), 47–60 (2023)

[7] Kaufmann, E., Bauersfeld, L., Loquercio, A., Müller, M., Koltun, V., Scaramuzza, D.: Champion-level drone racing using deep reinforcement learning. Nature **620**(7976), 982–987 (2023)

[8] Choi, S., Ji, G., Park, J., Kim, H., Mun, J., Lee, J.H., Hwangbo, J.: Learning quadrupedal locomotion on deformable terrain. Science Robotics **8**(74), 2256 (2023)

[9] Radosavovic, I., Xiao, T., Zhang, B., Darrell, T., Malik, J., Sreenath, K.: Real-world humanoid locomotion with reinforcement learning. Science Robotics **9**(89), 9579 (2024)

[10] Zador, A., Escola, S., Richards, B., Ölveczky, B., Bengio, Y., Boahen, K., Botvinick, M., Chklovskii, D., Churchland, A., Clopath, C., DiCarlo, J., Ganguli, S., Hawkins, J., Körding, K., Koulakov, A., LeCun, Y., Lillicrap, T., Marblestone, A., Olshausen, B., Pouget, A., Savin, C., Sejnowski, T., Simoncelli, E., Solla, S., Sussillo, D., Tolias, A., Tsao, D.: Toward next-generation artificial intelligence: Catalyzing the neuroai revolution. Nature Communications **14**(1), 1597 (2023)

[11] Zhao, T.Z., Kumar, V., Levine, S., Finn, C.: Learning fine-grained bimanual manipulation with low-cost hardware. In: Proceedings of Robotics: Science and Systems (2023)

[12] Lin, F., Hu, Y., Sheng, P., Wen, C., You, J., Gao, Y.: Data scaling laws in imitation learning for robotic manipulation. In: Proceedings of International

Conference on Learning Representations (2025)

[13] Kaelbling, L.P.: The foundation of efficient robot learning. Science **369**(6506), 915–916 (2020)

[14] Dreczkowski, K., Vitiello, P., Vosylius, V., Johns, E.: Learning a thousand tasks in a day. Science Robotics **10**(108), 7594 (2025)

[15] Insider, B.: Tesla shifted its Optimus training strategy — and it's using a familiar playbook. https://www.businessinsider.com/tesla-musk-optimus-humanoid-robot-training-motion-capture-cameras-2025-8 (2025)

[16] Todorov, E., Erez, T., Tassa, Y.: Mujoco: A physics engine for model-based control. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5026–5033 (2012)

[17] Makoviychuk, V., Wawrzyniak, L., Guo, Y.R., Lu, M., Storey, K., Macklin, M., Hoeller, D., Rudin, N., Allshire, A., Handa, A., State, G.: Isaac gym: high performance gpu-based physics simulation for robot learning. arXiv preprint arXiv:2108.10470 (2021)

[18] Geng, H., Wang, F., Wei, S., Li, Y., Wang, B., An, B., Cheng, C.T., Lou, H., Li, P., Wang, Y.-J., Liang, Y., Goetting, D., Xu, C., Chen, H., Qian, Y., Geng, Y., Mao, J., Wan, W., Zhang, M., Lyu, J., Zhao, S., Zhang, J., Zhang, J., Zhao, C., Lu, H., Ding, Y., Gong, R., Wang, Y., Kuang, Y., Wu, R., Jia, B., Sferrazza, C., Dong, H., Huang, S., Wang, Y., Malik, J., Abbeel, P.: Roboverse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning. In: Proceedings of Robotics: Science and Systems (2025)

[19] Peng, X., Zheng, Z., Shen, C., Young, T., Guo, X., Wang, B., Xu, H., Liu, H., Jiang, M., Li, W., Wang, Y., Ye, A., Ren, G., Ma, Q., Liang, W., Lian, X., Wu, X., Zhong, Y., Li, Z., Gong, C., Lei, G., Cheng, L., Zhang, L., Li, M., Zhang, R., Hu, S., Huang, S., Wang, X., Zhao, Y., Wang, Y., Wei, Z., You, Y.: Open-sora 2.0: Training a commercial-level video generation model in 200k. arXiv preprint arXiv:2503.09642 (2025)

[20] NVIDIA, :, Agarwal, N., Ali, A., Bala, M., Balaji, Y., Barker, E., Cai, T., Chattopadhyay, P., Chen, Y., Cui, Y., Ding, Y., Dworakowski, D., Fan, J., Fenzi, M., Ferroni, F., Fidler, S., Fox, D., Ge, S., Ge, Y., Gu, J., Gururani, S., He, E., Huang, J., Huffman, J., Jannaty, P., Jin, J., Kim, S.W., Klár, G., Lam, G., Lan, S., Leal-Taixe, L., Li, A., Li, Z., Lin, C.-H., Lin, T.-Y., Ling, H., Liu, M.-Y., Liu, X., Luo, A., Ma, Q., Mao, H., Mo, K., Mousavian, A., Nah, S., Niverty, S., Page, D., Paschalidou, D., Patel, Z., Pavao, L., Ramezanali, M., Reda, F., Ren, X., Sabavat, V.R.N., Schmerling, E., Shi, S., Stefaniak, B., Tang, S., Tchapmi, L., Tredak, P., Tseng, W.-C., Varghese, J., Wang, H., Wang, H., Wang, H., Wang, T.-C., Wei, F., Wei, X., Wu, J.Z., Xu, J., Yang, W., Yen-Chen, L., Zeng, X., Zeng, Y., Zhang, J.,

Zhang, Q., Zhang, Y., Zhao, Q., Zolkowski, A.: Cosmos world foundation model platform for physical AI (2025). https://arxiv.org/abs/2501.03575

[21] Jain, V., Attarian, M., Joshi, N.J., Wahid, A., Driess, D., Vuong, Q., Sanketi, P.R., Sermanet, P., Welker, S., Chan, C., Gilitschenski, I., Bisk, Y., Dwibedi, D.: Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers. In: Proceedings of Robotics: Science and Systems (2024)

[22] Chen, H., Sun, B., Zhang, A., Pollefeys, M., Leutenegger, S.: Vidbot: Learning generalizable 3D actions from in-the-wild 2D human videos for zero-shot robotic manipulation. In: Proceedings of the Computer Vision and Pattern Recognition Conference, pp. 27661–27672 (2025)

[23] Tobin, J., Biewald, L., Duan, R., Andrychowicz, M., Handa, A., Kumar, V., McGrew, B., Ray, A., Schneider, J., Welinder, P., *et al.*: Domain randomization and generative models for robotic grasping. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 3482–3489 (2018)

[24] Ai, B., Tian, S., Shi, H., Wang, Y., Pfaff, T., Tan, C., Christensen, H.I., Su, H., Wu, J., Li, Y.: A review of learning-based dynamics models for robotic manipulation. Science Robotics **10**(106), 1497 (2025)

[25] Meister, M.: Learning, fast and slow. Current opinion in neurobiology **75**, 102555 (2022)

[26] Somogyi, E., Ara, C., Gianni, E., Rat-Fischer, L., Fattori, P., O'regan, J.K., Fagard, J.: The roles of observation and manipulation in learning to use a tool. Cognitive Development **35**, 186–200 (2015)

[27] Thuerey, N., Holl, P., Mueller, M., Schnell, P., Trost, F., Um, K.: Physics-based deep learning. arXiv preprint arXiv:2109.05237 (2021)

[28] Heeger, D.J.: Theory of cortical function. Proceedings of the National Academy of Sciences **114**(8), 1773–1782 (2017)

[29] Mejias, J.F., Murray, J.D., Kennedy, H., Wang, X.-J.: Feedforward and feedback frequency-dependent interactions in a large-scale laminar network of the primate cortex. Science advances **2**(11), 1601335 (2016)

[30] Markov, D.A., Petrucco, L., Kist, A.M., Portugues, R.: A cerebellar internal model calibrates a feedback controller involved in sensorimotor control. Nature communications **12**(1), 6694 (2021)

[31] Marshall, J.A., Barron, A.B.: Are transformers truly foundational for robotics? npj Robotics **3**(1), 1–6 (2025)

[32] Wang, C., Ji, K., Geng, J., Ren, Z., Fu, T., Yang, F., Guo, Y., He, H., Chen, X., Zhan, Z., Du, Q., Su, S., Li, B., Qiu, Y., Du, Y., Li, Q., Yang, Y., Lin, X., Zhao, Z.: Imperative learning: A self-supervised neural-symbolic learning framework for robot autonomy. The International Journal of Robotics Research, 02783649251353181 (2024)

[33] Greydanus, S., Dzamba, M., Yosinski, J.: Hamiltonian neural networks. Advances in Neural Information Processing Systems **32** (2019)

[34] Cranmer, M., Greydanus, S., Hoyer, S., Battaglia, P., Spergel, D., Ho, S.: Lagrangian neural networks. arXiv preprint arXiv:2003.04630 (2020)

[35] Romero, A., Song, Y., Scaramuzza, D.: Actor-critic model predictive control. In: Proceedings of IEEE International Conference on Robotics and Automation, pp. 14777–14784 (2024)

[36] Han, L., Zhu, Q., Sheng, J., Zhang, C., Li, T., Zhang, Y., Zhang, H., Liu, Y., Zhou, C., Zhao, R., *et al.*: Lifelike agility and play in quadrupedal robots using reinforcement learning and generative pre-trained models. Nature Machine Intelligence **6**(7), 787–798 (2024)

[37] Yang, C., Yuan, K., Zhu, Q., Yu, W., Li, Z.: Multi-expert learning of adaptive legged locomotion. Science Robotics **5**(49), 2174 (2020)

[38] Kumar, A., Fu, Z., Pathak, D., Malik, J.: RMA: Rapid motor adaptation for legged robots. In: Proceedings of the Robotics: Science and Systems (2021)

[39] Lee, J., Bjelonic, M., Reske, A., Wellhausen, L., Miki, T., Hutter, M.: Learning robust autonomous navigation and locomotion for wheeled-legged robots. Science Robotics **9**(89), 9641 (2024)

[40] Kim, H., Oh, H., Park, J., Kim, Y., Youm, D., Jung, M., Lee, M., Hwangbo, J.: High-speed control and navigation for quadrupedal robots on complex and discrete terrain. Science Robotics **10**(102), 6192 (2025)

[41] Shi, F., Zhang, C., Miki, T., Lee, J., Hutter, M., Coros, S.: Rethinking robustness assessment: Adversarial attacks on learning-based quadrupedal locomotion controllers. In: Proceedings of Robotics: Science and Systems (2024)

[42] Rudin, N., Hoeller, D., Reist, P., Hutter, M.: Learning to walk in minutes using massively parallel deep reinforcement learning. In: Proceedings of Conference on Robot Learning, pp. 91–100 (2022)

[43] Jia, J., Guo, K., Wang, Y., Zhou, S., Zhang, J., Liu, Y., Yu, X., Shi, Y., Guo, L.: FORESEER: Recognize and utilize uncertainties by integrating data-based learning and symbolic feedback. The International Journal of Robotics Research, 02783649251364000 (2025)

[44] Jia, J., Guo, K., Yu, X., Zhao, W., Guo, L.: Accurate high-maneuvering trajectory tracking for quadrotors: A drag utilization method. IEEE Robotics and Automation Letters **7**(3), 6966–6973 (2022)

[45] Ollero, A., Tognon, M., Suarez, A., Lee, D., Franchi, A.: Past, present, and future of aerial robotic manipulators. IEEE Transactions on Robotics **38**(1), 626–645 (2022)

[46] Yu, H., Guo, D., Yin, H., Chen, A., Xu, K., Chen, Z., Wang, M., Tan, Q., Wang, Y., Xiong, R.: Neural motion prediction for in-flight uneven object catching. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 4662–4669 (2021)

[47] Braje, W.L., Tjan, B.S., Legge, G.E.: Human efficiency for recognizing and detecting low-pass filtered objects. Vision Research **35**(21), 2955–2966 (1995)

[48] Peterson, A.J., Heil, P.: Phase locking of auditory nerve fibers: The role of lowpass filtering by hair cells. Journal of Neuroscience **40**(24), 4700–4714 (2020)

[49] Lyu, S., Lang, X., Zhao, H., Zhang, H., Ding, P., Wang, D.: RL2AC: Reinforcement learning-based rapid online adaptive control for legged robot robust locomotion. In: Proceedings of the Robotics: Science and Systems (2024)

[50] Zhang, W., Liu, Q., Wang, M., Jia, J., *et al.*: Design of an aerial manipulator system applied to capture missions. In: Proceedings of International Conference on Unmanned Aircraft Systems, pp. 1063–1069 (2021)

[51] Aucone, E., Mintchev, S.: Embodied aerial physical interaction: Combining body and brain for robust interaction with unstructured environments. Science Robotics **10**(102), 0200 (2025)

[52] Seeber, R., Haimovich, H., Horn, M., Fridman, L.M., De Battista, H.: Robust exact differentiators with predefined convergence time. Automatica **134**, 109858 (2021)

[53] Han, J.: From PID to active disturbance rejection control. IEEE Transactions on Industrial Electronics **56**(3), 900–906 (2009)

[54] Jia, J., Zhang, W., Guo, K., Wang, J., Yu, X., Shi, Y., Guo, L.: EVOLVER: Online learning and prediction of disturbances for robot control. IEEE Transactions on Robotics **40**, 382–402 (2023)

[55] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., Abbeel, P.: Domain randomization for transferring deep neural networks from simulation to the real world. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 23–30 (2017)

[56] Jia, J., Yang, Z., Wang, M., Guo, K., Yang, J., Yu, X., Guo, L.: Feedback favors the generalization of neural ODEs. In: Proceedings of International Conference on Learning Representations (2025)

[57] Ha, S., Lee, J., Panne, M., Xie, Z., Yu, W., Khadiv, M.: Learning-based legged locomotion: State of the art and future perspectives. The International Journal of Robotics Research **44**(8), 1396–1427 (2025)

[58] Green, M., Limebeer, D.J.: Linear Robust Control. Courier Corporation, Chelsea, MA (2012)

[59] Parisi, G.I., Kemker, R., Part, J.L., Kanan, C., Wermter, S.: Continual lifelong learning with neural networks: A review. Neural Networks **113**, 54–71 (2019)

[60] Chen, R.T., Rubanova, Y., Bettencourt, J., Duvenaud, D.K.: Neural ordinary differential equations. Advances in Neural Information Processing Systems **31** (2018)

[61] Chen, W., Yang, J., Guo, L., Li, S.: Disturbance-observer-based control and related methods—An overview. IEEE Transactions on Industrial Electronics **63**(2), 1083–1095 (2015)

[62] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)

[63] Shabana, A.A.: Dynamics of Multibody Systems. Cambridge University Press, Cambridge, UK (2020)

[64] Wang, M., Chen, Z., Guo, K., Yu, X., Zhang, Y., Guo, L., Wang, W.: Millimeter-level pick and peg-in-hole task achieved by aerial manipulator. IEEE Transactions on Robotics **40**, 1242–1260 (2023)

[65] Jia, J., Wang, M., Liu, Y., Guo, K., Yu, X., Xie, L., Guo, L.: Disturbance observer for estimating coupled disturbances. arXiv preprint arXiv:2407.13229 (2024)

[66] Takens, F.: Detecting strange attractors in turbulence. In: Dynamical Systems and Turbulence, Warwick 1980: Proceedings of a Symposium Held at the University of Warwick 1979/80, pp. 366–381 (2006)

[67] Sauer, T., Yorke, J.A., Casdagli, M.: Embedology. Journal of Statistical Physics **65**(3), 579–616 (1991)

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- SupplementaryInformation.pdf
- SupplementaryMovies.zip