Sim-to-real transfer for fixed-wing uncrewed aerial vehicle: pitch control by high-fidelity modelling and domain randomization

Daichi Wada[1], Sergio Araujo-Estrada[2], and Shane Windsor[2]

***Abstract*—Deep reinforcement learning has great potential to automatically generate flight controllers for uncrewed aerial vehicles (UAVs), however these controllers often fail to perform as expected in real world environments due to differences between the simulation environment and reality. This study experimentally investigated how this reality gap effect could be mitigated, focusing on fixed-wing UAV pitch control in wind tunnel tests. Three different training approaches were conducted: a baseline approach that used simple linear dynamics, a high-fidelity modeling approach, and a domain randomization approach. It was found that the base line controller was susceptible to the reality gap, while the other two approaches successfully transferred to real tests. To further examine the controllers' capabilities to generalize, a variety of configuration changes were experimentally implemented on the UAV, such as increased inertia, extended elevator area, and aileron offset. While the high-fidelity controller failed to cope with these changes, the controller with domain randomization maintained its performance. These results highlight the importance of selecting appropriate sim-to-real transfer approaches and how domain randomization is applicable to fixed-wing UAV control with uncertainty in real environments.**

***Index Terms*—Aerial Systems: Mechanics and Control, Motion Control, Reinforcement Learning**

I. Introduction

FOR nonlinear control, neural-network-based controllers trained using machine learning techniques have the potential to be an alternative approach to custom designed controllers, with many potential applications in a variety of fields such as robotics and aviation. Supervised learning has potential benefits for efficiency and smooth interpolation in applications where a baseline controller already exists [1]. However, when a teaching controller does not exist, or the goal is to create a new controller with improved performance beyond that of existing controllers, deep reinforcement learning is a more suitable approach. By designing reward functions that elicit desired behaviors, the controller is automatically trained to behave in an optimized way, potentially offering time savings and performance benefits over manual design processes [2]–[8].

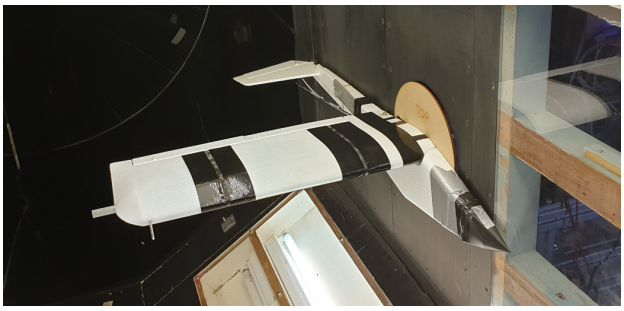

Fig. 1. A fixed-wing UAV in the wind tunnel. The pitch angle is controlled by deflecting the elevator.

Although deep reinforcement learning has produced convincing results in simulation, experimental demonstration is often challenging due to discrepancies between simulated and real environments; this is known as the "reality gap" [9], [10]. Deep reinforcement learning usually optimizes a controller for a single theoretical scenario and the controller's performance can be poor if the target domain is too different from the simulation [11].

To close the reality gap, sim-to-real transfer approaches have been studied. One approach has been to improve the accuracy of the simulation by using higher fidelity modeling, for example by more accurate system identification [12], [13] or by utilizing another neural network to better approximate the systems dynamics [14], [15]. Orthogonal to the high-fidelity modeling, another solution is to generate controllers with robustness. This is a viable approach especially when it is not feasible to fully reproduce the rich dynamics of the real environment. An example is the domain randomization approach, where parameters in the source domain, such as the visual appearance of objects [16]–[18], or dynamic parameters of the system [11], [12], [19], [20], are randomized during training so that the controllers learn robustness to environmental or system uncertainty.

In the aviation field specifically, sim-to-real transfers have

Manuscript received: February, 24, 2022; Revised June, 17, 2022; Accepted July, 26, 2022.

This paper was recommended for publication by Editor Pauline Pounds upon evaluation of the Associate Editor and Reviewers' comments.

A part of this work was funded by JSPS KAKENHI (grant number JP19K04850). This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 679355) and support from the UKRI Trustworthy Autonomous Systems Node in Functionality (EP/V026518/1).

[1] Daichi Wada is with Aeronautical Technology Directorate, Japan Aerospace Exploration Agency, 6-13-1 Osawa, Mitaka, Tokyo, 181-0015, Japan, wada.daichi@jaxa.jp

[2] Sergio Araujo-Estrada* and Shane Windsor are with Department of Aerospace Engineering, University of Bristol, Bristol, BS8 1TR, United Kingdom, *Present address: Aeronautical and Astronautical Engineering Department, University of Southampton, Southampton, SO16 7PX, United Kingdom, s.araujo-estrada@soton.ac.uk; shane.windsor@bristol.ac.uk

Digital Object Identifier (DOI): see top of this page.

been demonstrated for quadrotors [21]–[23], but demonstrations for fixed-wing UAVs remain rare. Fixed-wing UAVs differ from quadrotors in that rather than directly controlling the balance of thrust between rotors to control flight trajectory, control surfaces are deflected to modify the aerodynamic forces produced by the wing and tail. Fixed wing aerodynamics have quite different characteristics from rotor aerodynamics, such as fixed wings stalling at high angles of attack and having different time scales for their dynamic effects. These fundamental differences suggest the need for specific investigation of the sim-to-real transfer considerations for fixed-wing UAV control. A series of previous studies investigated typical relationships between the type of reality gaps and their consequence in control performance [24], [25]. The effect of model errors, namely friction and delay, were investigated experimentally using wind tunnel testing. This study extends this line of previous research on closing reality gaps by introducing a comparative perspective of different sim-to-real transfer approaches. With an integrated understanding of the cause and effect of reality gaps and how they can be overcome, we aim to provide examples of how different sim-to-real approaches influence the performance of reinforcement-learning-based controllers for fixed-wing UAVs.

This study investigates deep reinforcement-learning-based pitch control for a fixed-wing UAV, with a focus on sim-to-real transfer. The fixed-wing UAV is tested in a one degree-of-freedom configuration in a wind tunnel, in order to allow assessment with a simple and clear basis (Figure 1). Two sim-to-real transfer approaches, namely high-fidelity modeling and domain randomization, are compared. Furthermore, a variety of experimental modifications were implemented on the model configuration, such as additional inertia, altered control effectiveness and changes to aerodynamic performance. By examining the controllers' behavior, the difference between the two sim-to-real transfer approaches is highlighted in terms of their capabilities under unpredicted dynamic variations.

II. Related Works

A. Neural-network-based control in aviation (simulation studies)

For nonlinear control, supervised learning has been applied to flight control systems [1]. For example, gain schedules for existing control systems [26], nonlinear transformations for feedback linearisation [27], [28], and decision-making lookup tables for collision avoidance [29], have all been approximated by neural networks.

Deep reinforcement learning, on the other hand, generates policies (controllers) without needing a baseline. Deep-reinforcement-learning based controllers have shown a high level of performance for complex tasks, such as trajectory planning and navigation [4], [8], fixed-wing aircraft landing under wind disturbance [5], flocking control for fixed-wing UAVs [6], aerobatic maneuvers [7], and attitude control for fixed-wing UAVS [3] and quadrotors [2].

All above aviation-related applications have been explored in simulation and the challenge remains to transfer these approaches to real environments.

B. Sim-to-real transfer and domain randomization

The basic concept of sim-to-real transfer is to use a policy learned in simulation in a real environment while keeping the theoretical performance, even though differences exist between the simulated and real environments (known as the reality gap). To close the reality gap, previous works have proposed various methods to mitigate the gap effect including Bayesian optimization [30], deep inverse dynamics models [31], progressive networks [32], and domain randomization. In domain randomization, a policy is trained to work across all of the simulated environments with randomized parameters, thus becoming robust to real-world variability.

This idea has been experimentally tested in various applications, such as for control of robot arms [19], quadruped robots [12], and dexterous robot hands [20]. As an extended alternative, more adaptive approaches have been proposed, where controllers identify, implicitly or explicitly, environmental parameters using a limited number of on-site trials, and then tune the outputs correspondingly. For example, latent space adaptation methods identify a latent representation of the dynamics on-site and modulate the systems actions [33]–[35].

Domain randomization approaches do not require on-site tuning when moving from sim-to-real. This is an advantage in applications such as aviation, where flight trials are expensive in terms of both time and cost and safety issues mean lose of control has to be avoided during any trial-and-error tuning. In this context, domain randomization has been successfully applied to quadrotors, such as flight through a narrow gap [22] and flight control of hybrid UAVs with a variety of flight modes [21]. Investigation of trajectory tracking performance has been also carried out from the viewpoint of how to design control outputs (action spaces) [23]. However, studies of sim-to-real for fixed-wing UAVs specifically remain rare. Therefore, inspired by the previous success in quadrotors, this study applies domain randomization to a fixed-wing UAV to explore the reality gap effect and how it can be mitigated.

As the simulated environment with randomized parameters can be regarded as a type of partially observable Markov decision process, its optimal policy in general depends on access to the state's history [36]. In this sense, it becomes essential to use policies with some form of memory-augmentation along with domain randomization. To address this point, models using recurrent neural networks have been proposed, including Gated Recurrent Unit (GRU) [37] and Long Short-Term Memory (LSTM). Another method is to use policies without memory functions and instead input multiple time steps to policies [33], [35]. These successful previous studies suggest that domain randomization should work if the policy captures time histories by either method. This study follows recent success in using LSTM with domain randomization [19], [20].

III. Methods

A. Aircraft model

An off-the-shelf radio control aircraft (WOT4 Foam-E Mk2+, Ripmax) was modified into a span-wise half model, mounted to the side wall of a wind tunnel. The model was free to pitch around the shaft of the wing at its root, as depicted in

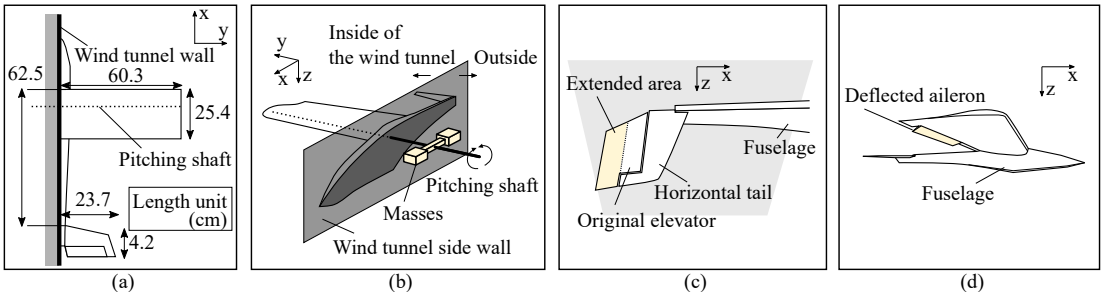

Fig. 2. Experimental configuration variants: (a) baseline with characteristic dimensions (b) added masses, (c) extended elevator, (d) deflected aileron.

Figure 2a. The model was equipped with an air speed sensor (custom-built based on SDP31, Sensirion), an elevator servo and an inertial measurement unit (IMU) (Pixhawk 1, 3DR). See [25] for full details. The signal processing and elevator control was conducted at a rate of 20 Hz. The wind speed was kept constant for each test case.

Two theoretical models were developed; an approximated linear model and a high-fidelity model including frictional effects. The dynamics of these models were expressed as:

$$I_{yy}\ddot{\alpha} = \frac{1}{2}\rho V^2 S\, c\, C_M - g_4 \tanh(g_5 q), \tag{1}$$

$$C_M = C_{M_0} + C_{M_\alpha}\alpha + C_{M_q}\frac{c}{2V}q + C_{M_{\delta_e}}\delta_e, \tag{2}$$

where I_{yy} is moment of inertia in pitch, ρ is air density, V is air speed, S is wing area, c is chord length, α is pitch angle, q is pitch rate, δ_e is elevator angle and g_4 and g_5 are the coefficients for the Coulomb friction effect. Each of the C_{M_x} terms represents individual aerodynamic coefficients. These parameter values are given in Table I. For the linear model, the friction effect was set to $g_4 = 0$. Detailed information about the parameter identification and the model fidelity have been discussed in previous research [25].

To assess the generalization performance of the controllers presented here, four experimental model variants were tested. Figure 2b depicts the variant with added masses to increase inertia. Two sets of 200 g masses were mounted to the pitching shaft at a 15 cm radial distance from the center of the shaft. Figure 2c depicts the variant with the extended elevator, which doubled the area of the elevator. This modification changed the equilibrium point and the effectiveness of the elevator deflection. Figure 2d depicts the variant with the deflected aileron. The aileron deflection was set to 40°, which changed the equilibrium pitch angle.

TABLE I
PARAMETERS FOR SIMULATION AND PPO.

Parameter	Value	Randomization range
I_{yy} (kg m^2)	1.90×10^{-1}	-
ρ (kg/m^3)	1.23	-
S (m^2)	3.06×10^{-1}	-
c (m)	2.54×10^{-1}	-
C_{M_0}	-3.00×10^{-3}	additive $\mathcal{U}$(-0.015, 0.015)
C_{M_α}	-2.25×10^{-1}	scaling $\mathcal{U}$(0.6, 1.4)
C_{M_q}	-5.46	scaling $\mathcal{U}$(0.6, 1.4)
$C_{M_{\delta_e}}$	-5.35×10^{-2}	scaling $\mathcal{U}$(0.6, 1.4)
g_4 (N m)	8.53×10^{-1}	-
g_5 (s/rad)	100	-
α noise (°)		additive $\mathcal{N}(0, 0.03^2)$
V noise (m/s)		additive $\mathcal{N}(0, 0.1^2)$
q noise (°/s)		additive $\mathcal{U}$(-3, 3)
PPO clip threshold	0.2	-
PPO batch size	512	-
PPO epochs	1	-
Discount factor	0.99	-
Adam learning rate	1.0×10^{-3}	-

$\mathcal{U}(\min, \max)$ signifies a uniform distribution. $\mathcal{N}(\text{mean}, \text{variance})$ signifies a normal distribution.

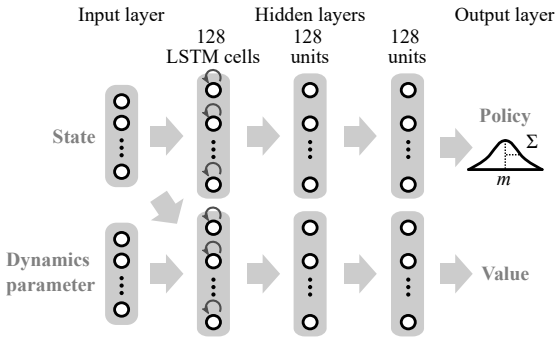

Fig. 3. Schematic of the policy (top) and value (bottom) networks. The policy received the state, which was processed by 128 LSTM cells and two 128 hidden units with *elu* activations. The output layer specified the mean m of the action distribution. The standard deviation Σ of the action distribution was specified by a fixed diagonal matrix. The value function was modelled by a separate network. The state and the dynamics parameter were received and processed by 128 LSTM cells and two 128 hidden units with *elu* activations.

B. Controller training

Deep reinforcement learning was conducted using the Proximal Policy Optimization (PPO) algorithm [39], which ensured learning stability by limiting the divergence between the old and updated policies using the so-called clipped surrogate loss function. As described later in this section, actor-critic algorithms are advantageous for domain randomization in that the critic function can utilize the randomized parameters in training, which are generally not explicitly known to the controller. Although there are other promising actor-critic algorithms including Twin Delayed Deep Deterministic policy gradient (TD3) [40] and Soft Actor-Critic (SAC) [22], [41], this study was inspired by the success of previous studies employing PPO [8], [20], [23]. Table I summarizes the hyperparameter settings for training with PPO. The neural networks and training process were built in a Python environment using

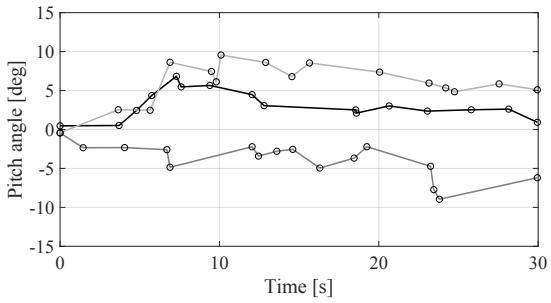

Fig. 4. Examples of the target pitch schedules. Each line shows the schedules generated with different random seeds. The Ornstein–Uhlenbeck process [38], with mean $\mu = 0$, volatility $\sigma = 2$ and reverting rate towards the mean $\theta = 0.1$, was used to calculate 16 discrete points with random intervals, which are indicated with circles. The adjacent points were linearly interpolated. The initial point was randomly set within $\pm 0.5°$, and the target pitch angles varied approximately within $\pm 10°$ during each episode.

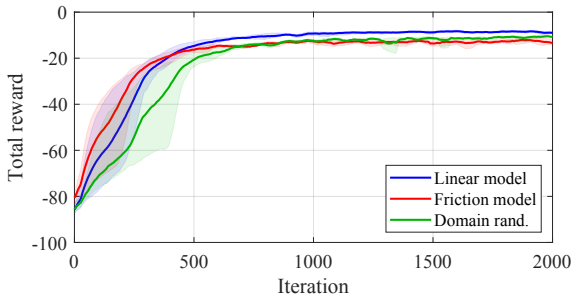

Fig. 5. Learning curves for the three controller types. The blue, red and green lines correspond to the controller trained with linear model, friction model and domain randomization, respectively. For each controller type, five independent training ran with different random seeds. The solid lines and shaded areas represent the mean and the minimum to maximum range, respectively.

the PyTorch framework (Ver. 1.7.1).

We designed a neural network controller which output the elevator angle rate based on the state input. The elevator angle rate rather than the elevator angle itself was chosen as an output so that the output was directly used in a penalty function as described later. The state space consisted of the error between the target and observed pitch angle, observed pitch angle, pitch rate, elevator angle and wind speed. The neural network architecture is illustrated in Figure 3. The policy function observed what was observable in the real application, that is, the dynamic parameters were unknown to the controller. The LSTM layers were employed to find an optimal history-dependent policy that adapted to every randomized environment. In the absence of direct knowledge of the real dynamics, a history of past states and actions was expected to help capture the system dynamics. The value function was designed to observe the full conditions because it was used only during training and the dynamic parameters of the simulator were known [19], [35]. The value function estimates the value of the states in the randomized simulator.

Three types of controller were trained: the first was trained with the linear model (baseline), the second was trained with the friction model (high-fidelity modeling approach), and the third was trained with domain randomization (robust approach). The linear model was used as the basis for the domain randomization. In the domain randomization, the aerodynamic coefficients were randomly changed in each episode as listed in Table I. These randomization factors were fed into the value function.

All controllers used the following common training conditions, each episode had $30\,\mathrm{s}$ duration and 600 time steps, and the control rate was set to $20\,\mathrm{Hz}$. The wind speed was randomly chosen for each episode ranging from 8 to $22\,\mathrm{m/s}$, and set constant over the episode. The target pitch schedules were randomly assigned for each episode. Figure 4 shows three examples of the target pitch schedules with different random seeds. State observation noise was applied at every time step. The noise distributions for each observation parameter are shown in Table I. These values were estimated from calibration tests of the IMU and the air speed sensor. The elevator angle in the state was the target value given in the previous time step. It was the theoretical value, and therefore, the observation noise was not applied.

Delay was randomly sampled and applied at every time step based on the probability density function expressed as

$$f(x) = \frac{1}{(x-t_0)\sigma\sqrt{2\pi}} \exp\left(-\frac{(\log(x-t_0)-\mu)^2}{2\sigma^2}\right) \quad (3)$$
$$\text{for } x > t_0,$$

where x is time (in s), t_0 is offset (in s), σ is the standard deviation ($1.67\,\mathrm{s}$) and μ is the mean ($-5.27\,\mathrm{s}$). This is a log-normal function with the peak at t_0, which was found to best fit the measured delay in the previous experiments [25]. The effective delay was empirically set as $t_0 = 0.100\,\mathrm{s}$.

The reward function was defined as

$$r = -(|e_\alpha| + 0.1|q| + 0.1|a| + 0.1|\Delta\delta_e|), \quad (4)$$

where e_α (rad) is the difference between target and observed pitch angle, q (rad/s) is the pitch rate, a (rad/s) is the action output and $\Delta\delta_e$ (rad) is the deviation of the current elevator angle from the exponential moving average with $5\,\%$ smoothing factor. The pitch rate, action and elevator penalties encouraged smooth and stable pitch maneuvers.

Figure 5 shows the learning curves for the controllers considered here. In the three training types, the total reward in the training episodes reached saturation at around 1000 iterations and converged for all trials, which indicated valid training results.

IV. Results

A. Performance in simulated and real environments

The performance of the three types of trained controllers was assessed through simulation and in experiments. In the experiments, the wind speed range was 10 – $20\,\mathrm{m/s}$ at $2\,\mathrm{m/s}$ intervals. At each wind speed, control trials were repeated three times. For brevity, we present a subset of these cases; these were chosen as representative of the behavior observed for all experimental conditions, with similar trends being observed through wind speeds and repetitive trials.

Figure 6 shows a comparison of the pitch control time histories for both simulation and real-world experiments.

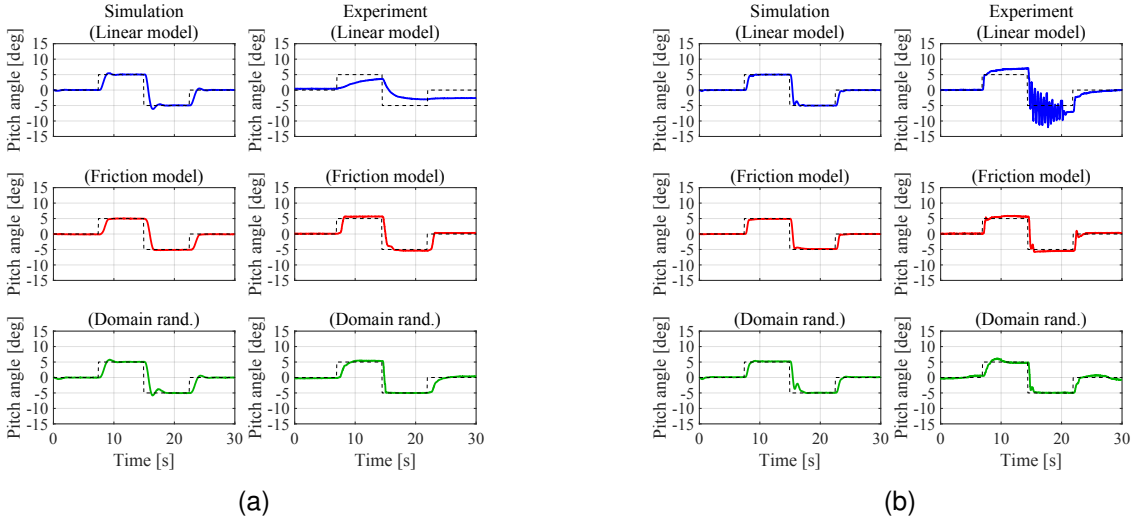

Fig. 6. Pitch control histories when the nominal wind speed was: (a) $10\,\mathrm{m/s}$ and (b) $20\,\mathrm{m/s}$. In each panel, the left and right graphs show simulated and experimental results, respectively. The controllers were trained using the linear model (blue line), the friction model (red line) and domain randomization (green line). In contrast to the baseline performance (linear model), both the high-fidelity modeling approach (friction model) and the robust approach (domain randomization) showed successful control in the experiments.

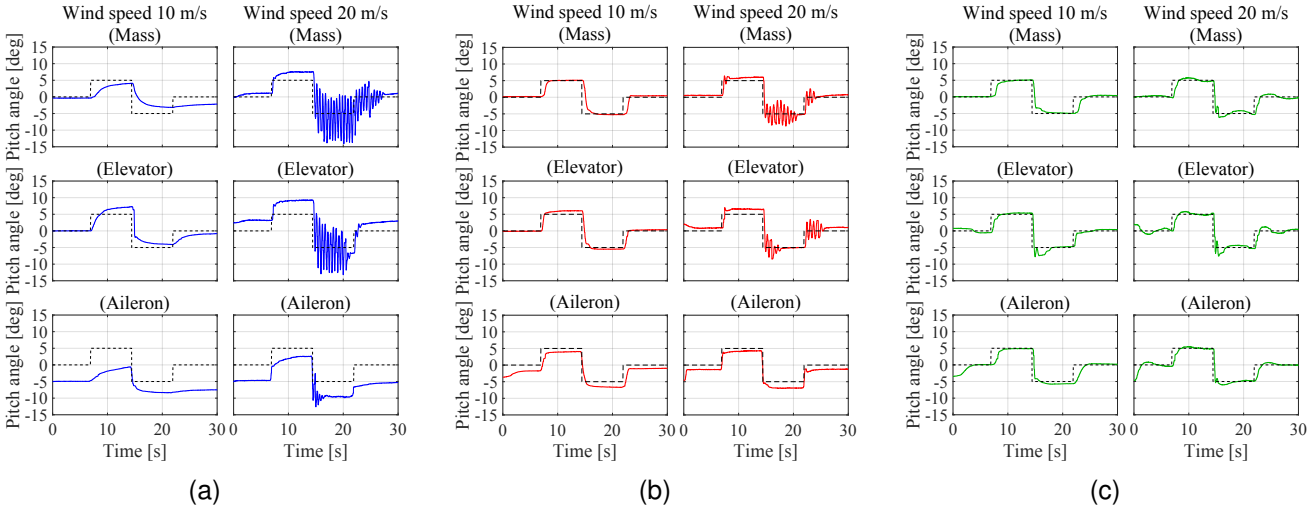

Fig. 7. Experimental results under a variety of model configuration changes and using controllers trained using: (a) the linear model (blue line), (b) the friction model (red line) and (c) domain randomization (green line). Each panel shows results for two nominal wind speeds $10\,\mathrm{m/s}$ (left plots) and $20\,\mathrm{m/s}$ (right plots). Additionally, each panel shows results for the added masses (top plots), the extended elevator (middle plots) and the deflected aileron (bottom plots) variants. Under the model variations, only the domain randomization approach was able to follow the target pitch schedule.

Figure 6a shows results for a wind speed of $10\,\mathrm{m/s}$, while Figure 6b shows results for $20\,\mathrm{m/s}$. In the simulation, the same system dynamics that were used for training were used for cases presented here. To ease comparison, a pitch angle doublet schedule was used as reference for both simulations and experiments.

The three controllers successfully followed the target schedules in both simulation scenarios, indicating valid training. However, the performance of the linear controller was unsatisfactory for both wind speed experimental conditions. Two different failure modes were observed: an offset in pitch at $10\,\mathrm{m/s}$ (Figure 6a); and oscillating behavior at $20\,\mathrm{m/s}$ (Figure 6b). The offset in pitch is thought to be related to friction effects, as at low wind speeds, the elevator deflection did not produce enough moment to overcome static friction and rotate the model. As for the oscillating behavior, this could be explained by a combination of friction and other factors such as system time delay. A previous study [25] investigated the relationship between the simulated delay in training and the effective delay in reality with this experimental setup and concluded that if the delay used in the training simulations was shorter than the actual experimental delay then this induced oscillation at high wind speeds. This behavior, i.e., the control performance not being guaranteed after sim-to-real transfer, is typical of deep-reinforcement-learning-based controllers when compensation measures are not taken.

In contrast, both the friction and domain randomization controllers successfully followed the target pitch schedule in both experimental conditions. The friction controller performance was expected due to it being trained with a high-fidelity

model. The domain randomization controller was effective as it was trained to be robust to uncertainty in the parameters of the dynamics model. It was anticipated that varying the aerodynamic coefficients would express similar dynamics to that of the friction model. It was interesting to note that a larger phase lag was observed, which was particularly evident after the reference transition from -5° to 0° at $10\,\mathrm{m/s}$. The domain randomization encouraged the controller to work across all the simulated environments, and potentially made it conservative.

B. Generalized performance under model variations

To examine the applicability of the sim-to-real transfer approaches in an extended context of large topology and dynamics variations, control experiments were conducted with the wind tunnel model variants as depicted in Figures 2b–2d, using the same controllers presented in Section IV-A (Figure 6). Figure 7 presents a comparison of the pitch control experiments conducted with the model variants and using controllers trained with the linear model (Figure 7a), the friction model (Figure 7b), and domain randomization (Figure 7c).

For the controller trained with the linear model, both failure modes of pitch offset (for both wind speeds) and oscillating behavior (for the higher wind speed) were observed for the added masses and extended elevator variants (top and middle panels in Figure 7a). For the deflected aileron variant, pitch offset errors were observed for both wind speeds (bottom panels in Figure 7a). This is likely explained by the pitching-down moment produced by the aileron deflection.

For the controller trained with the friction model, the controller was stable and showed good performance when applied to the added masses and extended elevator variants for the $10\,\mathrm{m/s}$ wind speed case. However, the oscillation failure mode was observed at the $20\,\mathrm{m/s}$ wind speed case (top and middle-right panels in Figure 7b). When applied to deflected aileron variant, the controller failed to compensate the pitch offset (bottom panels in Figure 7b).

The controller trained with domain randomization successfully followed the target in both wind speeds and when applied to all three variants. A robust response was observed for all the experimental conditions, even at the higher wind speed where the oscillating behavior was observed for the other two controllers, displaying only a small overshoot during the transitory phases of the maneuver. When applied to the deflected aileron variant, the starting pitch offset was rapidly corrected and remained negligible throughout the duration of the tests (bottom panels in Figure 7c).

To gain a statistical perspective, Figure 8 shows the root mean square error (RMSE) for the pitch tracking under model variations. The mean and standard deviation of RMSE were calculated from 18 individual experiments (three controller types, at six wind speeds, from 10 – $20\,\mathrm{m/s}$ with a $2\,\mathrm{m/s}$ interval, and three trials at each condition). Considering the similar RMSE and the small standard deviations, domain randomization was shown to be robust to the model variations conducted. Note that although the friction model might seem to show only slight larger RMSEs than domain randomization,

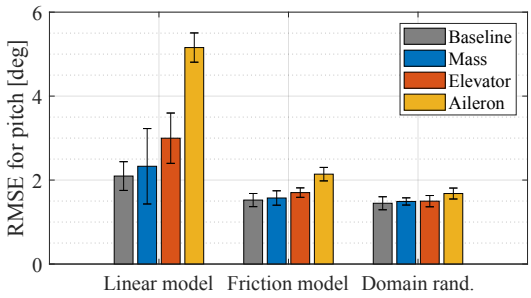

Fig. 8. Root mean square errors for tracking the target pitch angle under model variations. Error bar denotes standard deviation. Note that only the domain randomization approach was able to maintain control performance across the different model variations.

there was a significant difference in the quality of control as seen in Figure 7.

V. Discussion

To highlight the adaptability of each controller type (or lack of) to different conditions, the different control actions for each controller type were studied. Figure 9 shows the elevator deflection time histories for the three controllers considered here. To study the controller behaviors without oscillation, only the $10\,\mathrm{m/s}$ wind speed case is presented. The test cases correspond to the ones shown in Figure 6a and 7.

For the controller trained with the linear model, the four elevator maneuvers were almost identical, even when a variety of errors (such as positive and negative pitch offsets) were present. In the baseline linear dynamics simulation, the elevator and pitch angles were uniquely related for a given wind speed, and training under such dynamics resulted in a controller that simply learned the simulation equilibrium conditions.

On the other hand, for the controller trained with the friction model, the equilibrium condition was path-dependent (i.e., the elevator-pitch angle equilibrium relationship was not unique), which encouraged the controller to react to model variations. This led to different behaviors in the elevator deflection histories and a small degree of adaptability. However, when this controller was applied to the deflected aileron variant, the elevator did not show a significant difference in response, resulting in a large pitch angle error. This is likely related to the effect of the additional pitching moment, produced by the aileron deflection, being larger than the one resulting from friction.

The elevator deflection histories of the controller trained with domain randomization exhibited markedly different responses, which highlighted its robustness to model parameter errors. Concentrating on the extended elevator variant, during the first half of the experiments ($0\,\mathrm{s} \leq t \leq 15\,\mathrm{s}$), the control action sat above the actions corresponding to the other three cases, to compensate for the change in the equilibrium point caused by the extended elevator. Then during the transition from 5° to -5° in pitch reference, overshoot was observed at $t \approx 15\,\mathrm{s}$, transitioning to a smaller value at $15\,\mathrm{s} \leq t \leq 22.5\,\mathrm{s}$. When applied to the deflected aileron variant, the action was

This article has been accepted for publication in IEEE Robotics and Automation Letters. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/LRA.2022.3205442

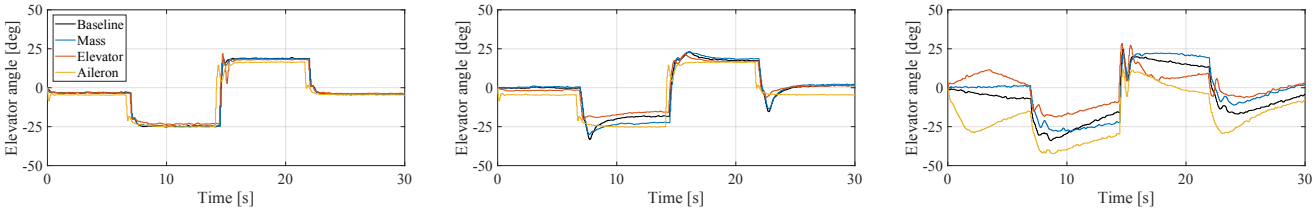

Fig. 9. Experimental elevator deflection time histories for the controllers trained with: (a) the linear model, (b) the friction model and (c) the domain randomization. The control actions when applied to the baseline configuration and three variants have been overlaid, with the baseline shown in black, the added masses shown in blue, the extended elevator shown in orange and the deflected aileron shown in yellow. All the results correspond to the $10\,\mathrm{m/s}$ wind speed case. Note that only the domain randomization controller adapted its output appropriately to changes in the model experimental parameters.

TABLE II
IDENTIFIED CHANGE IN DYNAMICS PARAMETERS FOR THREE VARIANTS.

Parameter	Change with respect to baseline value		
	Added masses	Extended elevator	Deflected aileron
I_{yy}	$\times$ 1.32	$\times$ 1.00	$\times$ 1.00
C_{M_0}	+ -0.0014	+ -0.0037	+ 0.0074
C_{M_α}	$\times$ 1.49	$\times$ 1.28	$\times$ 1.25
C_{M_q}	$\times$ 1.18	$\times$ 0.98	$\times$ 0.96
$C_{M_{\delta_e}}$	$\times$ 1.48	$\times$ 1.41	$\times$ 1.15

significantly shifted towards the negative elevator deflection side, which was necessary to compensate for the pitching moment induced by the aileron.

The theoretical elevator deflection equilibrium conditions can be computed by solving Eqs. 1 and 2 for δ_e, and by considering both $\ddot{\alpha} = 0$ and $q = 0$. This is given by

$$\bar{\delta}_e = -\frac{C_{M_0} + C_{M_\alpha}\bar{\alpha}}{C_{M_{\delta_e}}} \tag{5}$$

where $\bar{\delta}_e$ is the elevator deflection in equilibrium and $\bar{\alpha}$ is the equilibrium pitch angle. Using Eq. 5 and the variant-specific aerodynamic parameters (Table II), the elevator deflection equilibrium conditions for $5°$ pitch angle were $-22.4°$, $-18.6°$ and $-32.6°$ for the added mass, extended elevator and deflected aileron configurations, respectively. When compared to the controller actions shown in Figure 9, only the controller actions of the domain randomization controller (Figure 9c) were close to these values.

The analysis presented here shows that the domain randomization approach produced a controller with a better degree of adaptability. Table II gives the experimentally estimated change in value for each of the parameters corresponding to the configuration variants and with respect to the baseline values (Table I). When compared with the randomization range as listed in Table I, some parameters exceeded the values assumed in training. Considering the successful control results in experiments, the domain randomization was able to encourage robustness that was applicable to dynamics variation that were more severe than assumed. Note that the true "working range" should always be carefully examined as any case-specific evaluation may not generalize for any combination of dynamics parameter variations.

It is important to note that the wind speed was kept constant in the experiment, which is a simplification of real world flight conditions. This difference is likely to be significant and possibly requires additional wind speed sensing and/or more robust control capability against external disturbances. Future work on sim-to-real approaches should examine varying wind speed conditions.

VI. CONCLUSION

Here we studied sim-to-real transfer of pitch controllers for a fixed-wing UAV. Three different approaches were studied through controllers trained with a simple linear dynamics model, a high-fidelity model and domain randomization. The performance of these controllers was assessed via simulations and experiments with a variety of configuration changes in the wind tunnel. We found that only domain randomization was able to perform consistently with variation in dynamic parameters. This study highlights the importance of carrying out an experimental investigation to evaluate the capabilities of controllers in a real environment, and comparing different sim-to-real transfer approaches to understand their respective applicability and limitations.

To successfully apply deep reinforcement learning to fixed-wing UAV flight control in real scenarios, future research could look into extending the analysis presented here to multi-degree-of-freedom situations under varying wind speeds, including responsiveness and operation efficiency goals into the reward function (aiming for both optimality and robustness), and studying its applicability to inherently unstable aircraft or aircraft with the ability to change stability (e.g. by means of geometry morphing).

ACKNOWLEDGMENTS

A part of this work was funded by JSPS KAKENHI (grant number JP19K04850). This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 679355) and support from the UKRI Trustworthy Autonomous Systems Node in Functionality (EP/V026518/1). The authors would like to thank Mr. Lee Winter from the University of Bristol Wind Tunnel Laboratory, for his invaluable support and work during the assembly of the experimental platform used to carry out the tests presented in this paper.

REFERENCES

[1] W. Gu, K. P. Valavanis, M. J. Rutherford, and A. Rizzo, "A survey of artificial neural networks with model-based control techniques for flight control of unmanned aerial vehicles," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019, pp. 362–371.

[2] W. Koch, R. Mancuso, R. West, and A. Bestavros, "Reinforcement learning for uav attitude control," *arXiv preprint arXiv:1804.04154*, 2018.

[3] E. Bøhn, E. M. Coates, S. Moe, and T. A. Johansen, "Deep reinforcement learning attitude control of fixed-wing uavs using proximal policy optimization," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2019, pp. 523–533.

[4] C. Xi and X. Liu, "Unmanned aerial vehicle trajectory planning via staged reinforcement learning," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020, pp. 246–255.

[5] C. Tang and Y.-C. Lai, "Deep reinforcement learning automatic landing control of fixed-wing aircraft using deep deterministic policy gradient," in *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, 2020, pp. 1–9.

[6] C. Yan, X. Xiang, and C. Wang, "Fixed-wing uavs flocking in continuous spaces: A deep reinforcement learning approach," *Robotics and Autonomous Systems*, vol. 131, p. 103594, 2020.

[7] S. G. Clarke and I. Hwang, "Deep reinforcement learning control for aerobatic maneuvering of agile fixed-wing aircraft," in *AIAA SciTech 2020 Forum*, 2020.

[8] V. J. Hodge, R. Hawkins, and R. Alexander, "Deep reinforcement learning for drone navigation using sensor data," *Neural Computing and Applications*, vol. 33, no. 6, pp. 2015–2033, 2021.

[9] S. Koos, J. B. Mouret, and S. Doncieux, "Crossing the reality gap in evolutionary robotics by promoting transferable controllers," in *Proceedings of the 12th Annual Genetic and Evolutionary Computation Conference, GECCO '10*, 2010, pp. 119–126.

[10] A. Boeing and T. Bräunl, "Leveraging multiple simulators for crossing the reality gap," in *12th International Conference on Control, Automation, Robotics and Vision, ICARCV 2012*, vol. 2012, 2012, pp. 1113–1119.

[11] F. Muratore, M. Gienger, and J. Peters, "Assessing transferability from simulation to reality for reinforcement learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 4, pp. 1172–1183, 2021.

[12] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," in *Robotics: Science and Systems*, 2018.

[13] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne, "Learning locomotion skills for cassie: Iterative design and sim-to-real," in *Proceedings of the Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 100, 2020, pp. 317–329.

[14] S. Bansal, A. K. Akametalu, F. J. Jiang, F. Laine, and C. J. Tomlin, "Learning quadrotor dynamics using neural network for flight control," in *2016 IEEE 55th Conference on Decision and Control, CDC 2016*, 2016, pp. 4653–4660.

[15] J. Fu, S. Levine, and P. Abbeel, "One-shot learning of manipulation skills with online dynamics adaptation and neural network priors," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, p. 4019–4026.

[16] F. Sadeghi and S. Levine, "Cad2rl: Real single-image flight without a single real image," *arXiv preprint arXiv:1611.04201*, 2017.

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

[18] A. Pashevich, R. Strudel, I. Kalevatykh, I. Laptev, and C. Schmid, "Learning to augment synthetic images for sim2real policy transfer," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2651–2657, 2019.

[19] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 3803–3810.

[20] O. A. M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[21] J. Xu, T. Du, M. Foshey, B. Li, B. Zhu, A. Schulz, and W. Matusik, "Learning to fly: Computational controller design for hybrid uavs with reinforcement learning," *ACM Transactions on Graphics*, vol. 38, no. 4, 2019.

[22] C. Xiao, P. Lu, and Q. He, "Flying through a narrow gap using end-to-end deep reinforcement learning augmented with curriculum learning and sim2real," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–8, 2021.

[23] E. Kaufmann, L. Bauersfeld, and D. Scaramuzza, "A benchmark comparison of learned control policies for agile quadrotor flight," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 10 504–10 510.

[24] D. Wada, S. A. Araujo-Estrada, and S. Windsor, "Unmanned aerial vehicle pitch control using deep reinforcement learning with discrete actions in wind tunnel test," *Aerospace*, vol. 8, no. 1, 2021.

[25] ——, "Unmanned aerial vehicle pitch control under delay using deep reinforcement learning with continuous action in wind tunnel test," *Aerospace*, vol. 8, no. 9, 2021.

[26] S. Ferrari and R. F. Stengel, "Classical/neural synthesis of nonlinear control systems," *Journal of Guidance, Control, and Dynamics*, vol. 25, no. 3, pp. 442–448, 2002.

[27] B. S. Kim, A. J. Calise, and M. Kam, "Nonlinear flight control using neural networks and feedback linearization," in *The First IEEE Regional Conference on Aerospace Control Systems*, Westlake Village, CA, USA, 1993, pp. 176–181.

[28] O. Dadian, S. Bhandari, and A. Raheja, "A recurrent neural network for nonlinear control of a fixed-wing uav," in *2016 American Control Conference (ACC)*, Boston, MA, USA, 2016, pp. 1341–1346.

[29] K. D. Julian and M. J. Kochenderfer, "Deep neural network compression for aircraft collision avoidance systems," *Journal of Guidance, Control, and Dynamics*, vol. 42, no. 3, pp. 598–608, 2019.

[30] R. Pautrat, K. Chatzilygeroudis, and J.-B. Mouret, "Bayesian optimization with automatic prior selection for data-efficient direct policy search," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 7571–7578.

[31] P. Christiano, Z. Shah, I. Mordatch, J. Schneider, T. Blackwell, J. Tobin, P. Abbeel, and W. Zaremba, "Transfer from simulation to real world through learning deep inverse dynamics model," *arXiv preprint arXiv:1610.03518*, 2016.

[32] A. A. Rusu, M. Vecerík, T. Rothörl, N. Heess, R. Pascanu, and R. Hadsell, "Sim-to-real robot learning from pixels with progressive nets," in *1st Annual Conference on Robot Learning*, vol. 78, 2017, pp. 262–270.

[33] W. Yu, V. C. Kumar, G. Turk, and C. K. Liu, "Sim-to-real transfer for biped locomotion," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019, pp. 3503–3510.

[34] W. Yu, J. Tan, Y. Bai, E. Coumans, and S. Ha, "Learning fast adaptation with meta strategy optimization," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2950–2957, 2020.

[35] X. B. Peng, E. Coumans, T. Zhang, T.-W. E. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *arXiv preprint arXiv:2004.00784*, 2020.

[36] X. Chen, J. Hu, C. Jin, L. Li, and L. Wang, "Understanding domain randomization for sim-to-real transfer," in *International Conference on Learning Representations*, 2022.

[37] R. Dey and F. M. Salem, "Gate-variants of gated recurrent unit (gru) neural networks," in *IEEE 60th International Midwest Symposium on Circuits and Systems*, 2017, pp. 1597–1600.

[38] G. E. Uhlenbeck and L. S. Ornstein, "On the theory of the brownian motion," *Physical Review*, vol. 36, no. 5, pp. 823–841, 1930.

[39] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[40] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," *arXiv preprint arXiv:1802.09477*, 2018.

[41] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *arXiv preprint arXiv:1801.01290*, 2018.