Control Engineering Practice 172 (2026) 106915

Contents lists available at ScienceDirect

Control Engineering Practice

journal homepage: www.elsevier.com/locate/conengprac

Generalist AI control: Towards multi-purpose adaptive algorithms

Klinsmann Agyei, Pouria Sarhadi *

School of Physics, Engineering and Computer Science, University of Hertfordshire, Hatfield, UK

ARTICLE INFO

Keywords:
Autonomous vehicles
Generalist control
Adaptive control
Multi-purpose control
LSTM

ABSTRACT

Traditional controllers are designed for specific systems and do not transfer across different system orders and dynamics. We present a Generalist Controller, a learning-based controller capable of controlling systems of varying orders and dynamics. The approach introduces a novel dynamic state-space representation using attention mechanisms with masking, enabling a single neural network, trained in one shot, to handle systems with different dimensions without architectural modifications by assigning a system tag to each system. We generated 314,630 demonstrations from 25 diverse systems, including stable, unstable, minimum-phase, and non-minimum-phase dynamics, spanning linear and nonlinear systems from autonomous underwater and aerospace vehicles to mechanical systems and chemical processes. The model learns cross-system control strategies through multi-scale temporal processing and a mixture-of-experts architecture. Simulation results demonstrate that the proposed generalist controller achieves comparable performance to system-specific LQI controllers across all tested systems, including challenging cases such as non-minimum-phase and unstable dynamics, whilst generalising to unseen operating conditions including actuator saturation, noise, disturbance, and reference trajectories not encountered during training. This work represents a significant step towards generalist control policies within a defined family of dynamical systems, demonstrating effective control across a range of single-input single-output (SISO) systems of varying order and dynamics using a single learned policy without system-specific tuning.

1. Introduction

The control engineering discipline stands at an inflexion point where traditional model-based design paradigms are being complemented by data-driven approaches. Classical control methodologies (Astrom & Wittenmark, 1994; Doyle, 1996; Mayne et al., 2000) have proven effective across countless applications that shape our daily lives.

However, these approaches require accurate system models and separate controller design for each deployment. For many practical systems or systems with poorly characterised dynamics, accurate models are difficult or expensive to obtain and maintain. This motivates learning-based controllers that acquire control behaviour directly from demonstrations rather than analytical models. Whilst such approaches are well established (Argall et al., 2009; Mandlekar et al., 2021; Pomerleau, 1988; Schaal, 1999), they typically train separate policies for each system. We propose a generalist learning-based controller, a single policy trained on demonstrations from diverse systems without per-system tuning. Crucially, the framework is agnostic to the demonstration source, it can learn from model-based experts where accurate models exist, or from human teleoperation and recorded operational data where they do not, unifying both scenarios under a single deployable policy.

The transformative impact of foundation models across artificial intelligence domains has sparked considerable interest in their application to control systems. In natural language processing, models such as (Brown, 2020) have demonstrated remarkable few-shot generalisation capabilities through pre-training on diverse textual corpora, with applications on higher-level decision making in autonomous vehicles (Agyei et al., 2025a). Most pertinently for our domain, the robotics community has witnessed the emergence of foundation models specifically targeting embodied intelligence, suggesting that similar paradigms might revolutionise control theory itself. Octo Model Team (2024), a transformer-based policy trained on 800k robot trajectories across diverse embodiments, has shown that cross-robot generalisation is achievable through large-scale pre-training. Before this, Reed et al. (2022), DeepMind's multi-modal, multi-task, multi-embodiment generalist agent, controls multiple robots whilst simultaneously performing language tasks and playing Atari games with a single set of neural weights. Brohan (2022) and Zitkovich (2023) have achieved zero-shot generalisation to new tasks and environments, bridging the semantic gap between high-level instructions and low-level control through vision-language-action models. NVIDIA's recently announced GR00T N1 (Bjorck, 2025) represents a generalist model for humanoid robots,

* Corresponding author.
E-mail addresses: k.agyei@herts.ac.uk (K. Agyei), p.sarhadi@herts.ac.uk (P. Sarhadi).

https://doi.org/10.1016/j.conengprac.2026.106915
Received 2 November 2025; Received in revised form 11 February 2026; Accepted 9 March 2026
Available online 28 March 2026

featuring a dual-system architecture inspired by human cognition that enables generalised reasoning and manipulation skills across different embodiments.

However, these models fundamentally operate at the perception-action level rather than addressing the underlying control-theoretic challenges. Models like (Octo Model Team, 2024) and (Reed et al., 2022) learn control policies jointly with visual representations, creating entangled representations where control decisions may become linked to visual features. This visual-motor coupling could introduce some fundamental limitations. These models may be less effective without camera inputs, rendering them less suitable for applied control problems.

Parallel to these developments, imitation learning and behavioural cloning (Bratko et al., 1995; Pomerleau, 1988; Zare et al., 2024) have emerged as powerful paradigms for acquiring control policies from demonstrations. These approaches offer several advantages over model-based control design, they can capture complex control strategies that are difficult to derive analytically, they do not require explicit system models at deployment, and they can leverage existing expert controllers or human demonstrations. However, most imitation learning approaches train separate policies for each system, failing to exploit the shared structure underlying diverse dynamical systems.

This paper introduces the Generalist Controller, a learned controller that addresses this fundamental gap by learning control policies that transfer across systems of varying dimensionality and dynamics. Unlike existing approaches that focus on task-level or perception-level generalisation, the proposed generalist controller operates at the control-theoretic level, learning to track references for systems of varying orders. The key insight underpinning this approach is that control at the state-space level represents a highly transferable form of control knowledge across many physical systems. Whilst visual features vary dramatically across robots and environments, the underlying mathematics of dynamical systems remain invariant. A mass-spring-damper system exhibits identical second-order dynamics whether implemented as a vehicle suspension, a building's seismic isolator, or a microscale MEMS device. By learning control policies directly in state-space, we capture these fundamental dynamical patterns that transcend specific embodiments or sensory modalities. This approach introduces several technical innovations to enable cross-order generalisation. We develop a dynamic state-space representation with attention masking that allows a single neural network to process systems of different orders through padding. Through multi-scale temporal processing and mixture-of-experts architecture (Shazeer, 2017), the controller learns to recognise and respond to different dynamical regimes, from stable first-order responses to complex oscillatory behaviours. Trained on 314,630 expert demonstrations from 25 diverse systems, the proposed generalist controller achieves performance comparable to system-specific Linear Quadratic Integral (LQI) controllers. While the broader motivation of this work is inspired by recent foundation model approaches, the scope of the present study is more specific. In this paper, we focus on SISO dynamical systems.

The implications of this work extend beyond technical achievements in conventional control and, more specifically, in imitation learning for control applications. By demonstrating that control knowledge transfers across diverse system dynamics within pre-trained system families, and is not limited to a single category of systems, we enable multi-system control policies that potentially require no per-system re-tuning. This represents a paradigm shift from the traditional approach of designing controllers for specific systems towards learning controllers that embody general principles applicable across entire system classes. As industries face increasing demands for flexible automation, rapid reconfiguration, and robust performance across varying operating conditions, generalist controllers offer a compelling alternative to traditional design methodologies.

Before formalising the problem, we clarify the use of the term *generalist* as employed in this paper. The term *generalist* is used in a pragmatic sense consistent with recent learning-based robotics and foundation-model literature, where a single parameterised policy is trained on data from multiple tasks or systems (Octo Model Team, 2024; Reed et al., 2022). Adapting this notion to control engineering, we define a generalist controller as a single control policy trained on demonstrations from multiple dynamical systems and deployed without per-system retuning. The novelty of this work does not lie in introducing a new learning paradigm, as similar ideas have long been considered in Behavioural Cloning (BC) and Imitation Learning (IL) (Bratko et al., 1995; Pomerleau, 1988; Zare et al., 2024). Rather, leveraging recent advances in deep neural networks, we demonstrate that a single policy can effectively reuse control structure across heterogeneous systems with varying orders and dynamics, without any system-specific tuning. It should be noted that the controller requires a system tag as input to indicate which system is being controlled. It does not infer the system identity from observations alone and cannot generalise to systems not included in the training set. This parallels human adaptive behaviour, which can control systems after training and does not start blindly when encountering different systems.

The remainder of the paper is organised as follows. Section 2 reviews related work in adaptive, robust, and generalisation in control. Section 3 formulates the problem and defines the learning objective. Section 4 presents the proposed Generalist Controller architecture. Section 5 describes the training framework and data generation. Section 6 reports simulation and hardware results, including comparisons with classical LQI controllers. Section 7 presents ablation studies analysing, the role of key architectural components. Section 8 concludes the paper and discusses limitations and future directions.

2. Related work

This section reviews prior approaches to achieving control across multiple systems, spanning classical adaptive and robust control, generalisation methods in control theory, and recent learning-based approaches. We highlight the limitations of existing methods in handling systems with varying orders and dynamic characteristics, motivating our approach of learning a single policy from demonstrations across diverse systems.

2.1. Classical adaptive and robust control theory

Classical adaptive control provides theoretical frameworks for handling uncertainty, but focuses on adapting to parameter variations within a fixed system structure. Model Reference Adaptive Control (MRAC) (Ioannou & Fidan, 2006) adjusts controller parameters to match reference model behaviour for a specific plant. Self-tuning regulators (Astrom & Wittenmark, 1994) estimate system parameters online and update control laws accordingly. Neural adaptive control (Selmic & Lewis, 2002) combines neural networks with Lyapunov-based adaptation laws. Whilst effective for parametric uncertainty, these methods assume a known system order and structure, requiring redesign when applied to systems with different dynamics.

Robust control addresses uncertainty through worst-case design. H_∞ control (Zhou et al., 1996) minimises the worst-case gain from disturbances to outputs. μ-synthesis (Doyle, 1982) handles structured uncertainty. Whilst providing strong theoretical guarantees, robust controllers are conservative and require bounds on uncertainty. Recent work on learning-based robust control (Berkenkamp et al., 2017; Richards et al., 2018) combines data-driven methods with robust control theory, but remains limited to specific system classes. Simultaneous stabilisation (Vidyasagar, 2022) represented another promising approach to address the generalisability issues of conventional controllers. The aim was to synthesise a single controller capable of stabilising multiple plants. However, these controllers are limited to plants that share specific structural properties, such as possessing the same number of unstable poles or satisfying parity interlacing conditions. In practice, these theoretical limitations also translate into significant engineering effort, as classical

adaptive and robust controllers typically require careful, system-specific modelling, synthesis, and tuning for each new plant.

This work addresses these limitations by learning a single control policy that operates across systems with different structures and orders.

2.2. Generalisation in control systems

The challenge of generalisation in control extends beyond parameter uncertainty to encompass structural variations, changing objectives, and diverse operating conditions. Koopman operator theory (Brunton & Kutz, 2022) provides a theoretical framework for lifting nonlinear dynamics into infinite-dimensional linear representations, potentially enabling transfer across system classes. However, practical implementations require finite-dimensional approximations that may not preserve important dynamical properties. Behavioural systems theory (Willems, 2007) offers an alternative perspective, focusing on trajectories rather than state-space representations. Whilst this approach provides elegant theoretical insights, practical controller synthesis within this framework remains computationally challenging for complex systems. This work maintains the computational tractability of state-space methods whilst achieving the generalisation benefits sought by behavioural approaches.

Recent advances in reinforcement learning have demonstrated impressive generalisation in game-playing and simulated environments (Schrittwieser, 2020), yet transfer to physical systems remains challenging. The reality gap, encompassing differences in dynamics, sensing, and actuation between simulation and reality, continues to limit practical deployment (Zhao et al., 2020). Sim-to-real transfer techniques, including domain randomisation (Tobin, 2017) and system identification (Torricelli & Pons, 2018), partially address these challenges but require extensive engineering effort for each new system. Our approach sidesteps many of these issues by operating directly on state-space representations, avoiding the complexity of visual sim-to-real transfer whilst maintaining the benefits of learning-based generalisation.

3. Problem formulation

We consider a collection of N dynamical systems, denoted by $\mathcal{S}_i$, where each system may be linear or nonlinear, stable or unstable, and may exhibit minimum or non-minimum-phase behaviour. Each system $\mathcal{S}_i$ is characterised by its state $\mathbf{x}_{pi}(t) \in \mathbb{R}^{n_{x_{p,i}}}$, control input $\mathbf{u}_{pi}(t) \in \mathbb{R}^{n_{u_{c,i}}}$, and output $\mathbf{y}_{pi}(t) \in \mathbb{R}^{n_{y_{p,i}}}$.

For linear time-invariant (LTI) systems, the dynamics are described by

$$\dot{\mathbf{x}}_{p_i}(t) = A_{p_i}\mathbf{x}_{\mathbf{p}i}(t) + B_{p_i}\mathbf{u}_{\mathbf{p}i}(t), \tag{1}$$

$$\mathbf{y}_{p_i}(t) = C_{p_i}\mathbf{x}_{p_i}(t), \tag{2}$$

where $A_{p_i} \in \mathbb{R}^{n_{x_{p,i}} \times n_{x_{p,i}}}$, $B_{p_i} \in \mathbb{R}^{n_{x_{p,i}} \times n_{u_{c,i}}}$, and $C_{p_i} \in \mathbb{R}^{n_{y_{p,i}} \times n_{x_{p,i}}}$ are system-specific matrices.

For nonlinear systems, such as the REMUS Autonomous underwater vehicle (AUV) dynamics considered in this work, the system evolution is given by

$$\dot{\mathbf{x}}_{p_i}(t) = \mathbf{f}_{\mathbf{p}i}(\mathbf{x}_{p_i}(t), \mathbf{u}_{p_i}(t)), \tag{3}$$

$$\mathbf{y}_{p_i}(t) = \mathbf{h}_{\mathbf{p}_i}(\mathbf{x}_{p_i}(t)), \tag{4}$$

where $\mathbf{f}_i : \mathbb{R}^{n_{x_{p,i}}} \times \mathbb{R}^{n_{u_{c,i}}} \to \mathbb{R}^{n_{x_{p,i}}}$ and $\mathbf{h}_{p_i} : \mathbb{R}^{n_{x_{p,i}}} \to \mathbb{R}^{n_{y_{p,i}}}$ denote the nonlinear state transition and output mappings.

The objective is to learn a single shared control policy π_θ that can be deployed across all systems $\mathcal{S}_i$:

$$\pi_\theta : (\mathbf{x}_{\mathbf{p}i,t-T_h+1:t}, \mathbf{r}_{t-T_h+1:t}, \boldsymbol{\phi}_i) \mapsto \mathbf{u}_{\mathbf{c}i,t}, \tag{5}$$

where T_h denotes the history window and θ are the shared parameters of the Generalist Controller. The policy receives a system tag or identifier as input, denoted by ϕ_i, which is simply a labelled number assigned to each system. This label is assumed to be known at deployment. We emphasise that the controller does not perform implicit system identification or zero-shot adaptation to unknown dynamics. The need for this tag is similar to post-training human operation, where control is typically executed with awareness of which system is being controlled. This formulation allows a single policy to be deployed across known heterogeneous linear and nonlinear systems without per-system tuning.

4. The proposed generalist controller

The Generalist Controller employs a hierarchical architecture that progressively transforms raw state information into control actions. Fig. 1 illustrates the complete data flow: feature encoders process states with validity masking, references, and system parameters; temporal processors capture dynamics through parallel LSTMs and attention; and a mixture of experts generates control actions that are applied to the system.

For clarity, we summarise the operation of the proposed controller. At each timestep, the policy receives a history of measured states, the corresponding reference trajectory, and a discrete system tag that tells the controller which system it is controlling so it can select the appropriate control strategy for that specific system. State histories are masked to allow a unified representation across systems of different dimensions. Encoded features are processed by parallel temporal modules and mapped to control actions via a mixture-of-experts module. During training, the policy is optimised by supervised learning of LQI control actions, while at deployment it operates directly on measured states without online optimisation or per-system retuning.

4.1. Feature encoding

The feature encoding stage processes four input streams to create a unified representation through parallel encoding networks.

The first two encoders process state vectors with validity masking to handle variable-order systems. Given the zero-padded state history $\mathbf{X} \in \mathbb{R}^{T_h \times n_{\max}}$ and validity mask $\mathbf{M} \in \{0,1\}^{T_h \times n_{\max}}$, the masked state is computed as $\tilde{\mathbf{X}} = \mathbf{X} \odot \mathbf{M}$, where $\odot$ denotes the Hadamard product. The state encoders map this to latent representations:

$$\mathbf{f}_1 = \mathbf{W}_2^{(1)}\,\sigma\Big(\mathbf{W}_1^{(1)}\,\mathrm{vec}(\tilde{\mathbf{X}}) + \mathbf{b}_1^{(1)}\Big) + \mathbf{b}_2^{(1)} \tag{6}$$

$$\mathbf{f}_2 = \mathbf{W}_2^{(2)}\,\sigma\Big(\mathbf{W}_1^{(2)}\,\mathrm{vec}(\tilde{\mathbf{X}}) + \mathbf{b}_1^{(2)}\Big) + \mathbf{b}_2^{(2)} \tag{7}$$

where $\sigma(\cdot)$ denotes the ReLU activation function, $\mathrm{vec}(\cdot)$ vectorises the matrix input, and $\{\mathbf{W}_j^{(i)}, \mathbf{b}_j^{(i)}\}$ are learnable parameters for encoder i and layer j. The third encoder processes the reference trajectory $\mathbf{r} \in \mathbb{R}^{T_h}$, whilst the fourth encoder processes the system parameter vector $\boldsymbol{\phi} \in \mathbb{R}$:

$$\mathbf{f}_3 = \mathbf{W}_2^{(3)}\,\sigma\Big(\mathbf{W}_1^{(3)}\mathbf{r} + \mathbf{b}_1^{(3)}\Big) + \mathbf{b}_2^{(3)} \tag{8}$$

$$\mathbf{f}_4 = \mathbf{W}_2^{(4)}\,\sigma\Big(\mathbf{W}_1^{(4)}\boldsymbol{\phi} + \mathbf{b}_1^{(4)}\Big) + \mathbf{b}_2^{(4)} \tag{9}$$

The encoded features are concatenated to form the composite feature vector serving as input to the temporal processor:

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}_1 \\ \mathbf{f}_2 \\ \mathbf{f}_3 \\ \mathbf{f}_4 \end{bmatrix} \in \mathbb{R}^{d_f} \tag{10}$$

where $d_f = \sum_{i=1}^{4} d_i$ denotes the total feature dimension, with d_i being the output dimension of encoder i.

4.2. Temporal processing

The temporal processor captures dynamics at multiple timescales through two parallel LSTM networks and a multi-head attention mechanism.

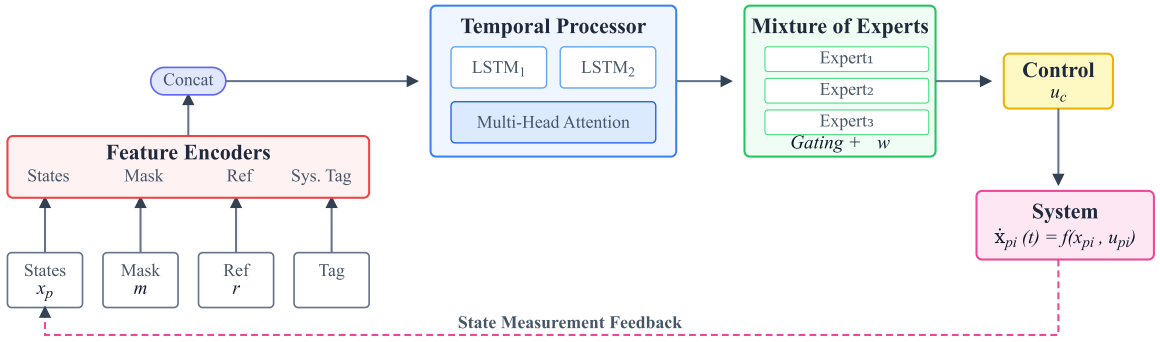


Fig. 1. Architecture of the proposed generalist controller.

The first LSTM processes the complete feature sequence $\{\mathbf{f}_k\}_{k=t-T_h+1}^{t}$ to capture immediate dynamics.

At each timestep k, the LSTM updates its hidden state according to:

$$\mathbf{i}_k^{(1)} = \sigma_g\left(\mathbf{W}_i^{(1)}\mathbf{f}_k + \mathbf{U}_i^{(1)}\mathbf{h}_{k-1}^{(1)} + \mathbf{b}_i^{(1)}\right) \tag{11}$$

$$\mathbf{g}_k^{(1)} = \sigma_g\left(\mathbf{W}_g^{(1)}\mathbf{f}_k + \mathbf{U}_g^{(1)}\mathbf{h}_{k-1}^{(1)} + \mathbf{b}_g^{(1)}\right) \tag{12}$$

$$\mathbf{o}_k^{(1)} = \sigma_g\left(\mathbf{W}_o^{(1)}\mathbf{f}_k + \mathbf{U}_o^{(1)}\mathbf{h}_{k-1}^{(1)} + \mathbf{b}_o^{(1)}\right) \tag{13}$$

$$\tilde{\mathbf{c}}_k^{(1)} = \tanh\left(\mathbf{W}_c^{(1)}\mathbf{f}_k + \mathbf{U}_c^{(1)}\mathbf{h}_{k-1}^{(1)} + \mathbf{b}_c^{(1)}\right) \tag{14}$$

$$\mathbf{c}_k^{(1)} = \mathbf{g}_k^{(1)} \odot \mathbf{c}_{k-1}^{(1)} + \mathbf{i}_k^{(1)} \odot \tilde{\mathbf{c}}_k^{(1)} \tag{15}$$

$$\mathbf{h}_k^{(1)} = \mathbf{o}_k^{(1)} \odot \tanh\left(\mathbf{c}_k^{(1)}\right) \tag{16}$$

where $\sigma_g(\cdot)$ denotes the sigmoid function, $\mathbf{i}_k^{(1)}$, $\mathbf{g}_k^{(1)}$, and $\mathbf{o}_k^{(1)}$ represent the input, forget, and output gates respectively, $\mathbf{c}_k^{(1)}$ is the cell state, and $\mathbf{h}_k^{(1)} \in \mathbb{R}^{d_h}$ is the hidden state.

The second LSTM operates on a downsampled sequence with stride τ to capture slower dynamics and longer-term patterns. Let $\mathcal{T}_\tau = \{t - T_h + 1, t - T_h + 1 + \tau, \ldots, t\}$ denote the downsampled time indices. The second LSTM processes $\{\mathbf{f}_k\}_{k\in\mathcal{T}_\tau}$ using the same gating mechanism with independent parameters $\{\mathbf{W}^{(2)}, \mathbf{U}^{(2)}, \mathbf{b}^{(2)}\}$, yielding hidden state $\mathbf{h}_t^{(2)} \in \mathbb{R}^{d_h}$.

The final hidden states from both LSTMs are concatenated to form the composite temporal feature vector:

$$\mathbf{z} = \begin{bmatrix} \mathbf{h}_t^{(1)} \\ \mathbf{h}_t^{(2)} \end{bmatrix} \in \mathbb{R}^{2d_h} \tag{17}$$

The concatenated features are further processed through a multi-head attention mechanism. We employ $N_h = 3$ attention heads, where each head applies scaled dot-product self-attention to the temporal feature vector $\mathbf{z}$.

For each head $h \in \{1, \ldots, N_h\}$, the query, key, and value projections are computed as:

$$\mathbf{Q}^{(h)} = \mathbf{W}_Q^{(h)}\mathbf{z}, \quad \mathbf{K}^{(h)} = \mathbf{W}_K^{(h)}\mathbf{z}, \quad \mathbf{V}^{(h)} = \mathbf{W}_V^{(h)}\mathbf{z} \tag{18}$$

where $\mathbf{W}_Q^{(h)}, \mathbf{W}_K^{(h)}, \mathbf{W}_V^{(h)} \in \mathbb{R}^{d_k \times 2d_h}$ are learnable projection matrices and $d_k = 2d_h/N_h$ is the dimension per head.

The attention output for each head is:

$$\mathbf{a}^{(h)} = \text{softmax}\left(\frac{\mathbf{Q}^{(h)}\mathbf{K}^{(h)\top}}{\sqrt{d_k}}\right)\mathbf{V}^{(h)} \tag{19}$$

The outputs from all heads are concatenated and projected to form the final attended representation:

$$\mathbf{z}_{\text{att}} = \mathbf{W}_O\begin{bmatrix} \mathbf{a}^{(1)} \\ \vdots \\ \mathbf{a}^{(N_h)} \end{bmatrix} + \mathbf{z} \tag{20}$$

where $\mathbf{W}_O \in \mathbb{R}^{2d_h \times 2d_h}$ is the output projection matrix. The attended feature vector $\mathbf{z}_{\text{att}} \in \mathbb{R}^{2d_h}$ serves as input to the mixture of experts control stage.

4.3. Mixture of experts control

The final stage employs M expert networks, each specialising in different control strategies. Each expert i is a three-layer feedforward network that maps the attended temporal feature vector $\mathbf{z}_{\text{att}}$ to a control action:

$$\mathbf{u}_c^{(i)} = \mathbf{W}_3^{(e_i)}\,\sigma\left(\mathbf{W}_2^{(e_i)}\,\sigma\left(\mathbf{W}_1^{(e_i)}\mathbf{z}_{\text{att}} + \mathbf{b}_1^{(e_i)}\right) + \mathbf{b}_2^{(e_i)}\right) + \mathbf{b}_3^{(e_i)} \tag{21}$$

where $\mathbf{u}_c^{(i)} \in \mathbb{R}$ is the control action predicted by expert i, $\sigma(\cdot)$ denotes the ReLU activation function, and $\{\mathbf{W}_j^{(e_i)}, \mathbf{b}_j^{(e_i)}\}$ are learnable parameters for expert i and layer j. Layer normalisation is applied after each hidden layer.

A gating network computes soft weights for expert combination based on the attended temporal features:

$$\boldsymbol{\gamma} = \mathbf{W}_g\mathbf{z}_{\text{att}} + \mathbf{b}_g \tag{22}$$

$$\alpha_i = \frac{\exp(\gamma_i)}{\sum_{j=1}^{M}\exp(\gamma_j)}, \quad i \in \{1, \ldots, M\} \tag{23}$$

where $\boldsymbol{\gamma} \in \mathbb{R}^M$ are the gate logits and α_i are the resulting mixture weights satisfying $\sum_{i=1}^{M}\alpha_i = 1$.

The final control sequence is computed as the weighted combination of expert outputs:

$$\hat{\mathbf{u}}_c = \sum_{i=1}^{M}\alpha_i\mathbf{u}_c^{(i)} \in \mathbb{R} \tag{24}$$

The system evolves according to its dynamics in Eq. (1), producing new state measurements that are fed back to the controller, and the process repeats.

5. Training framework

We train the Generalist Controller on expert demonstrations from 25 dynamical systems spanning orders $n \in \{2, 3, 4\}$.

5.1. Training data generation

Training data is generated by simulating each system under LQI control with step reference signals. Step amplitudes are sampled from $[-20, 20]$ using non-uniform discrete intervals, coarse spacing (intervals of 5 or 10) in some regions and finer resolution (intervals of 0.2 or 1) in others to ensure coverage across both small and large reference changes. For each system-amplitude pair, trajectories are recorded over a fixed simulation horizon, yielding state, reference, and control tuples. This procedure produces approximately 314,630 training demonstrations across all 25 systems.

5.2. Training configuration

Training sequences employ sliding windows with history $T_h = 8$. States $\mathbf{x}_{t-T_h:t} \in \mathbb{R}^{T_h \times n}$ are zero-padded to maximum dimension $n_{\max} = 4$, with validity mask $\mathbf{m} \in \{0,1\}^{T_h \times n_{\max}}$ indicating real versus padded dimensions. The history length $T_h = 8$ was selected as a trade-off between capturing sufficient temporal information to infer system dynamics and maintaining low computational cost. These hyperparameters were fixed across all systems to ensure consistency and to avoid system-specific tuning. No claim is made that these values are optimal, rather, they represent pragmatic choices that were found to work reliably across the considered benchmark systems.

The hierarchical architecture comprises: (i) encoders (state encoders) with masked attention processing state trajectories, (ii) parameter and reference encoders mapping to a common latent space, (iii) parallel multi-scale LSTMs capturing temporal dependencies at different resolutions, (iv) multi-head attention for temporal aggregation, and (v) a mixture-of-experts controller with $M = 3$ experts:

$$\mathbf{u}_{c,t} = \sum_{i=1}^{M} \alpha_i(\mathbf{z}_{att}) \cdot C_i(\mathbf{z}_{att}) \tag{25}$$

where α_i are learned gating weights and C_i represents expert i.

Training minimises mean squared error with AdamW optimiser ($\eta = 10^{-3}$, cosine annealing), gradient clipping ($\|\nabla\|_2 \leq 1.0$), and weight decay ($\lambda = 10^{-5}$). The model contains approximately 60,000 parameters and trains in six hours on a single GPU.

6. Simulation and experimental results

This section evaluates the proposed Generalist Controller across diverse dynamical systems. We first present the benchmark systems, then report simulation results on six representative systems spanning different orders and dynamic behaviours. We then validate the approach through hardware experiments on a Crazyflie 2.1+ nano-quadrotor, demonstrating successful sim-to-real transfer. Finally, we assess the controller's robustness under perturbed conditions not encountered during training, including actuator saturation, disturbances, and measurement noise.

6.1. Benchmark systems

The 25 benchmark systems encompass diverse characteristics and applications, fundamental mechanical systems (Mass-Spring-Damper, Damped Oscillator, Two-Mass Spring), electrical circuits (RLC Circuit), aerospace systems (Crazyflie quadrotor, Aircraft Longitudinal dynamics), robotic platforms (Differential Drive, Servo System), process control systems (Two-Tank System, Coupled Tanks, HVAC Zone), and systems with challenging dynamics (Non-Minimum Phase, Negative Stiffness, unstable integrators). Several of these systems were previously analysed in Agyei et al. (2025b), including the CSTR Chemical Reactor, nonlinear AUV yaw dynamics, and Two-Mass Spring system, providing established baselines for comparison.

6.2. Simulation results

In the simulation section, we further assess and present results for six representative systems selected from the benchmark systems: the CSTR Chemical Process (non-minimum phase), an Unstable System, Boeing 747 Aircraft (longitudinal dynamics), Hydraulic Actuator, Nonlinear AUV (REMUS yaw dynamics), and Two-Mass Spring. Table 2 presents these results. In the first two columns, system descriptions and state-space equations are provided. The third column shows controller performance on step response tracking with unit step references, comparing the proposed Generalist Controller against the LQI method. Furthermore, in the fourth column, we present simulation results for sinusoidal signal tracking from non-zero initial states not encountered during training, to assess the generalisability and resilience of the algorithm when initial conditions change. The controller demonstrates similar successful performance on ramp following and composite reference signals, with none of these complex references encountered by the model during training. Due to space constraints, the paper presents representative step-response and sinusoidal tracking results, while additional evaluations under alternative reference trajectories were conducted and yielded consistent qualitative behaviour.

Table 1
Performance metrics comparison between LQI and the proposed method.

Metric	Unst.	NMP	H.Act	Yaw	B-747	TMS	Crazyflie
t_r (LQI)	1.75	1.05	4.35	2.50	1.20	1.70	2.15
t_r (GC)	1.95	1.05	3.95	2.45	1.40	1.75	1.85
M_p (LQI)	5.5	0.4	0.0	0.8	2.3	11.5	1.60
M_p (GC)	5.4	0.7	0.2	0.4	2.0	11.1	2.00
t_s (LQI)	5.05	2.20	8.15	4.25	2.85	7.25	4.05
t_s (GC)	6.35	2.25	7.80	5.10	2.35	9.50	5.55
e_{ss} (LQI)	0.00	0.00	0.00	0.00	0.00	0.00	0.00
e_{ss} (GC)	0.00	0.00	0.00	0.00	0.00	0.00	0.00
ISE (LQI)	1.21	1.09	1.40	1.14	0.82	1.88	0.87
ISE (GC)	1.28	1.12	1.53	1.17	0.92	1.99	0.79
ITAE (LQI)	1.8	0.8	4.8	2.2	0.7	3.9	1.4
ITAE (GC)	2.9	1.0	5.0	4.2	1.1	5.1	2.50
$u_{\max}$ (LQI)	0.73	1.97	0.14	1.10	1.74	1.36	0.08
$u_{\max}$ (GC)	0.65	1.96	0.13	1.12	1.58	1.34	0.10

Table 1 reports a direct quantitative comparison between the proposed Generalist Controller and the system-specific LQI controllers using several commonly applied control performance metrics. These standard measures include the rise time t_r, overshoot percentage M_p, settling time t_s, steady-state error e_{ss}, Integral of Squared Error (ISE), Integral of Time multiplied by Absolute Error (ITAE), and the maximum control effort $u_{\max}$. Detailed definitions of these metrics can be found in (Sarhadi, 2025) and for brevity, they are not repeated here. Across all simulated benchmark systems, the proposed Generalist Controller (GC) achieves performance comparable to LQI in terms of rise time, overshoot, and steady-state error, despite being deployed without per-system tuning. In several cases, including the hydraulic actuator and Boeing 747 longitudinal dynamics, the GC exhibits similar or slightly improved settling times relative to LQI, while maintaining comparable control effort. For more challenging systems, such as the non-minimum-phase and two-mass spring benchmarks, the GC demonstrates slightly increased settling times. Overall, the results indicate that the GC preserves the essential closed-loop behaviour of classical LQI controllers while providing a single deployable policy across heterogeneous systems.

6.3. Experimental results

We validate the proposed controller on hardware using a Crazyflie 2.1+ nano-quadrotor. Fig. 2 shows the experimental setup. The Generalist Controller runs on a Linux computer with Python implementation, communicating with the quadcopter via the Crazyradio 2.0 USB dongle operating at 2.4GHz. The controller receives state feedback at 100Hz and computes control commands in real-time, demonstrating successful deployment of the learned policy on resource-constrained embedded systems.

Fig. 3 presents a direct comparison between simulation and hardware performance for altitude control. The physical system tracks the reference with similar transient behaviour and steady-state accuracy to simulation, confirming that the controller generalises effectively to real hardware without fine-tuning, despite unmodelled aerodynamic effects, sensor noise, and actuator delays.

6.4. Further evaluation under perturbations

To assess the robustness and generalisation capability of the proposed Generalist Controller beyond nominal operating conditions, we

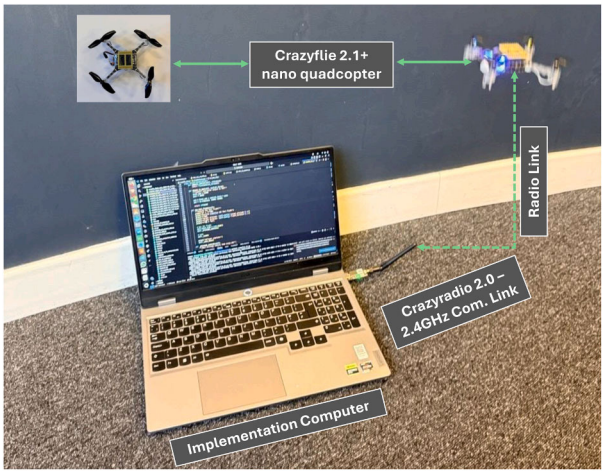


Fig. 2. Hardware validation: Crazyflie in flight with the proposed algorithm.

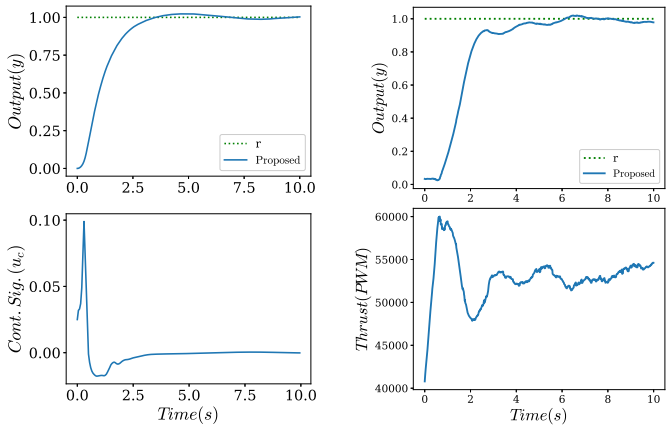


Fig. 3. Sim-to-real transfer: Crazyflie 2.1 + altitude control in (Left) simulation and (Right) hardware.

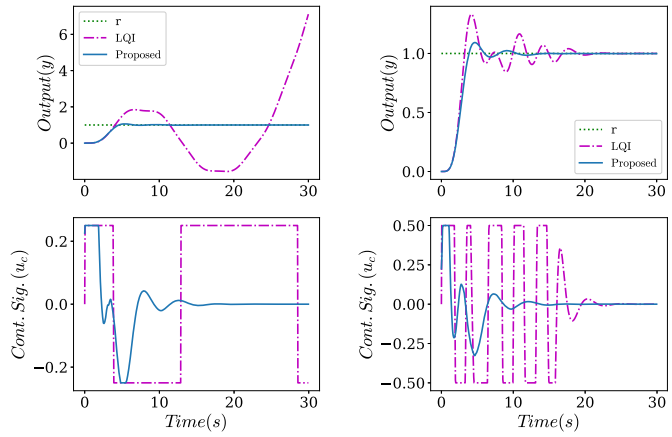


Fig. 4. TMS system under actuator amplitude saturation. Left: $|u_c| \leq 0.25$, Right: $|u_c| \leq 0.5$.

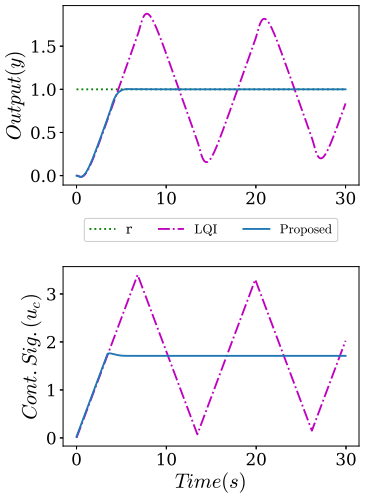


Fig. 5. CSTR chemical process system under input rate saturation ($|\dot{u}_c| = 0.5$).

conduct a series of experiments under perturbed test scenarios not encountered during training.

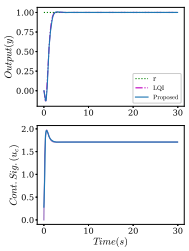
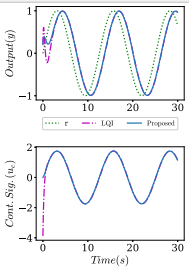
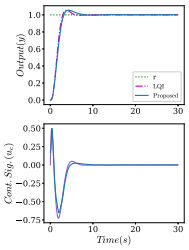
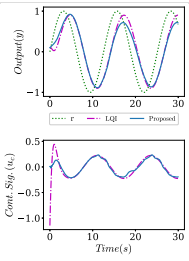
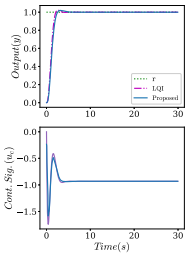
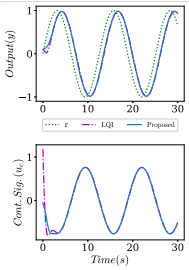
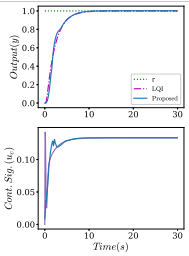
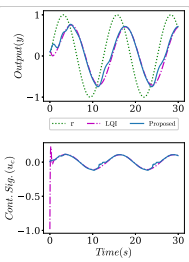
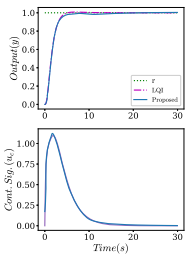
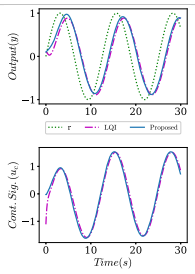
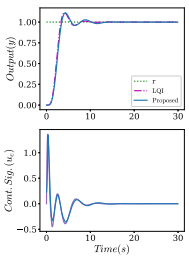
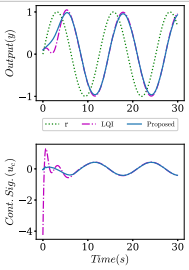
We first evaluate the controller under actuator amplitude saturation constraints applied to the challenging Two-Mass Spring (TMS) system. Fig. 4 presents results for two saturation limits: $|u_c| \leq 0.5$ and $|u_c| \leq 0.25$. Under moderate saturation ($|u_c| \leq 0.5$), the proposed method successfully tracks the reference with minimal overshoot, whilst the LQI controller exhibits significant oscillatory behaviour before eventually converging. When the saturation constraint is tightened to $|u_c| \leq 0.25$, the LQI controller becomes unstable and diverges, whereas the proposed method maintains stable tracking despite the limited control authority. These results demonstrate that the baseline LQI controller fails to maintain performance under actuator constraints, whereas the learned policy sustains stable tracking despite these unseen limitations.

We further evaluate performance under input rate saturation applied to the CSTR chemical process system. Fig. 5 shows the response when the rate of change of the control signal is constrained. The LQI controller, unable to adapt to the rate limitation, produces large oscillations in both output and control signal as it attempts corrections that are subsequently clipped, encountering the windup phenomenon. In contrast, the proposed method, though never exposed to rate constraints during training, maintains smooth reference tracking with a well-behaved control signal.

Finally, we assess performance under combined measurement noise and external disturbances for both TMS and CSTR systems. Fig. 6 presents the results with noise injected into state measurements after $t = 25$ s. For the CSTR system, the proposed method demonstrates superior performance compared to LQI, maintaining tighter tracking with reduced output variance despite the noisy measurements. The LQI controller exhibits highly aggressive control behaviour with large-amplitude oscillations in the control signal, which propagate to the output. For the TMS system, both controllers maintain stability under noise, though the proposed method shows slightly more variation in the output. However, examining the control signals reveals that the LQI controller demands significantly more aggressive actuation with peak-to-peak amplitudes substantially larger than those of the proposed method. This aggressive behaviour, whilst achieving marginally tighter tracking in the TMS case, would be problematic in practical applications where actuator wear, energy consumption, and excitation of unmodelled high-frequency dynamics are concerns.

These experiments demonstrate that the proposed controller generalises to operating conditions outside the training distribution, including actuator constraints and measurement corruption not seen during training. The successful performance under amplitude saturation, rate saturation, disturbance and noise injection provides evidence that the learned policy captures robust control principles rather than merely memorising input-output mappings from the training data.

Table 2
Results of the proposed generalist controller on multiple benchmark systems. (Inanc et al. (2025) is cited in this table).

System	State-Space Representation	Step Response	Sine Response
CSTR Chemical Process (Non-Minimum Phase) Agyei et al. (2025b)	$\dot{\mathbf{x}}_\mathbf{p} = \begin{bmatrix} 0 & 1 \\ -5.47 & -4.719 \end{bmatrix} \mathbf{x_p} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_p$ $y_p = \begin{bmatrix} 3.199 & -1.135 \end{bmatrix} \mathbf{x_p}$		
Unstable System ($G_p = \frac{1}{s(s-1)}$)	$\dot{\mathbf{x}}_\mathbf{p} = \begin{bmatrix} 0 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{x_p} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u_p$ $y_p = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x_p}$		
Boeing 747 Aircraft (Longitudinal Dynamics) Inanc et al. (2025)	$\dot{\mathbf{x}}_\mathbf{p} = \begin{bmatrix} -0.32 & 0.86 \\ -0.93 & -0.43 \end{bmatrix} \mathbf{x_p} + \begin{bmatrix} -0.02 \\ -1.16 \end{bmatrix} u_p$ $y_p = \begin{bmatrix} 1 & 0 \end{bmatrix} \mathbf{x_p}$		
Hydraulic Actuator (3D Servo Valve)	$\dot{\mathbf{x}}_\mathbf{p} = \begin{bmatrix} 0 & 1 & 0 \\ -10 & -1.167 & 25 \\ 0 & 0 & -0.8 \end{bmatrix} \mathbf{x_p} + \begin{bmatrix} 0 \\ 0 \\ 2.4 \end{bmatrix} u_p$ $y_p = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \mathbf{x_p}$		
Nonlinear AUV (REMUS Yaw Dynamics) Agyei et al. (2025b)	$(m - I_1)\dot{v} + (m\nu_g - I_4)\dot{r} = (Y_{uv}\lvert u\rvert + Y_{uuu}u^2)\nu$ $+(Y_{ur}\lvert u\rvert + Y_{uur}u^2)r - m\nu_0 u\dot{\psi}_0^2$ $(m\nu_g - V_1)\dot{v} + (I_z - N_r)\dot{r} = (N_{uv}\lvert u\rvert +$ $N_{uuu}u^2)\nu + (N_{ur}\lvert u\rvert + N_{uur}u^2)r - m\nu_g u\dot{\psi}_0 +$ $N_{uur}\delta\delta_0^2$ $\dot{\phi} = r$		
Two-Mass Spring (ACC Benchmark Problem) Agyei et al. (2025b)	$\dot{\mathbf{x}}_\mathbf{p} = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 1 & 0 & 0 \\ 1 & -1 & 0 & 0 \end{bmatrix} \mathbf{x_p} + \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} u_p$ $y_p = \begin{bmatrix} 0 & 1 & 0 & 0 \end{bmatrix} \mathbf{x_p}$		

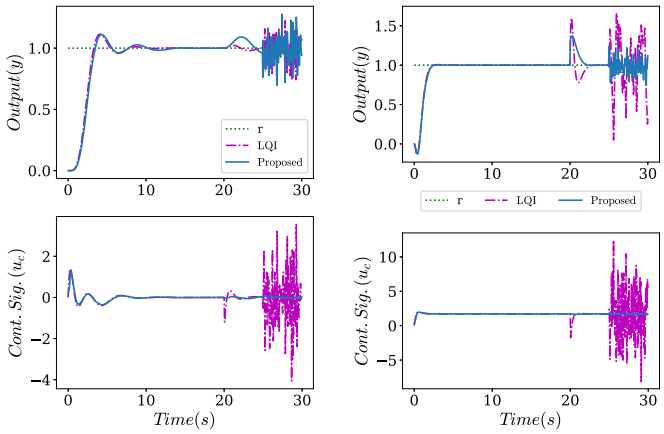


Fig. 6. Performance evaluation under combined measurement noise and external disturbance. Left: TMS. Right: CSTR.

7. Ablation studies

To validate the design choices of the proposed generalist controller architecture, we conduct an ablation study, examining the impact of key hyperparameters on control performance. We evaluate the model across five diverse dynamical systems: a double integrator representing a point-mass motion control, Crazyflie quadrotor for autonomous aerial vehicle control, a non-minimum phase chemical reactor exhibiting inverse response, a mass-spring-damper system, and a Building HVAC Zone (Two zones, thermal interaction). Each ablation systematically varies a single architectural component while maintaining all other parameters constant.

7.1. Hidden dimension

Network capacity plays a crucial role in the model's ability to learn complex temporal dynamics and generalise across systems. We examine three hidden dimension configurations: $d \in \{16, 32, 64\}$, where $d = 32$ is the baseline. Fig. 7 illustrates the impact of hidden dimension on both test performance and training dynamics. The left panel reveals that reducing the hidden dimension to $d = 16$ significantly impairs performance across all systems, with training loss increasing by more than an order of magnitude (from 9.43×10^{-4} to 1.02×10^{-2}). This degradation is particularly pronounced for the Crazyflie system, where MSE increases from 5.9×10^{-2} to 1.35, indicating that the reduced capacity network struggles to capture the intricate dynamics of underactuated aerial vehicles. Conversely, increasing the hidden dimension to $d = 64$ consistently improves performance across all test systems and achieves the lowest training loss (4.01×10^{-4}). The training convergence curves in the right panel demonstrate that the larger network not only reaches lower final loss values but also exhibits smoother convergence dynamics. The Crazyflie system benefits most from increased capacity, with MSE reducing from 5.9×10^{-2} to 1.44×10^{-2}. However, this performance gain comes at the cost of approximately 4× increase in parameters (from 60K to 232K).

The baseline configuration with $d = 32$ represents a pragmatic choice that balances model expressiveness with computational constraints. While $d = 64$ offers superior performance, the baseline achieves competitive results across all systems with significantly fewer parameters, making it more suitable for resource-constrained deployment scenarios such as embedded control applications.

7.2. Mixture of experts

The mixture of experts (MoE) mechanism is a critical component for enabling the proposed generalist controller to handle heterogeneous dynamical systems. We investigate the effect of varying the number of expert networks $M \in \{1, 3, 5\}$ on control performance, where $M = 3$ represents the baseline configuration. Fig. 8 presents the ablation results for the number of experts. The left panel shows output mean squared error (MSE) across all test systems for different expert configurations. The single expert configuration ($M = 1$) demonstrates degraded performance compared to the baseline, particularly evident for the Crazyflie quadrotor where MSE increases by over two orders of magnitude (from 5.9×10^{-2} to 3.98). This substantial performance drop confirms that a single expert network lacks sufficient capacity to capture the diverse dynamics across multiple systems.

Increasing the number of experts to $M = 5$ yields marginal improvements in training loss (reducing from 9.43×10^{-4} to 5.87×10^{-4}) and provides the best performance on the HVAC system (5.18×10^{-4} MSE). However, the improvement comes at the cost of increased model complexity, as shown in the complexity-performance trade-off in the right panel of Fig. 8. The baseline configuration with $M = 3$ experts achieves 59,180 parameters and strikes an optimal balance between model capacity and computational efficiency.

It is important to emphasise that the choice of $M = 3$ experts is not claimed to be universally optimal. Rather, it represents a trade-off between model capacity and computational efficiency for the class of systems considered in this study. As the complexity and diversity of target systems increase, a larger number of experts may be required to capture additional dynamical regimes. The ablation study is therefore intended to demonstrate the effect of expert multiplicity, rather than to identify a globally optimal value of M. Odd numbers of experts ($M = \{1, 3, 5\}$) were considered to enable a direct comparison against the single-expert baseline while progressively increasing model capacity, without expanding the ablation study beyond the available page and computational budget.

To further examine whether the mixture-of-experts architecture exhibits meaningful specialisation, we consider a configuration with five experts, which provides finer granularity for interpreting gating behaviour compared to the three-expert baseline. Fig. 9 presents the gate-weight heatmap, revealing non-uniform expert activation patterns. Expert 1 dominates the non-minimum-phase (45%) and damped oscillator (46.0%) systems, while Expert 4 is strongly activated for the Boeing 747 (98%) and Crazyflie (95%). The remaining experts receive consistently low gate weights across all systems. This pattern indicates that expert differentiation emerges implicitly through optimisation, despite joint training under a single global loss.

7.3. Key findings

The ablation studies yielded several important insights regarding the architectural components. The mixture of experts proves essential for effective generalisation across diverse systems. When reduced to a single expert configuration (M = 1), the controller fails to adapt to different dynamical regimes, with particularly severe degradation observed on the Crazyflie quadrotor system. Three experts (M = 3) provide sufficient specialisation to handle stable, unstable, and oscillatory dynamics whilst maintaining parameter efficiency. Increasing to five experts yields negligible performance gains whilst substantially increasing computational complexity. Network capacity significantly impacts controller performance across the benchmark systems. A hidden dimension of d = 16 proves insufficient for capturing the complex dynamics of multi-system control, resulting in order-of-magnitude performance degradation particularly evident in higher-dimensional systems. Whilst d = 64 offers optimal performance with the lowest tracking errors, the intermediate configuration of d = 32 achieves the best trade-off between accuracy and computational efficiency, maintaining performance within 8% of the larger model whilst requiring a quater of the parameters. The analysis reveals diminishing returns when scaling beyond baseline configurations. Increasing model capacity beyond M = 3 experts and d = 32 hidden dimensions yields marginal improvements at substantial computational

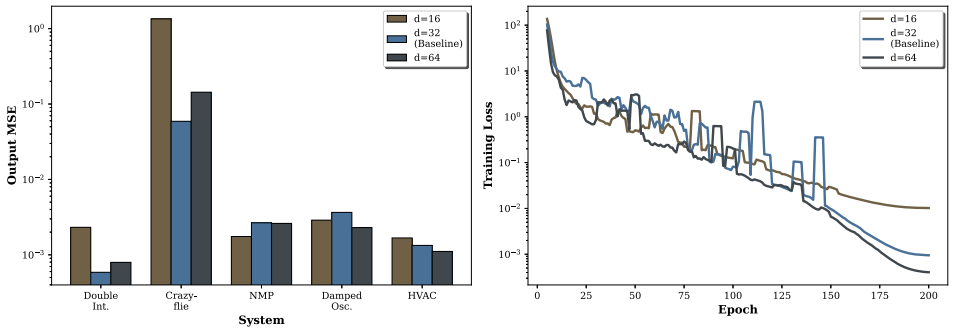


Fig. 7. Ablation study results. Left: Output MSE across systems. Right: Training loss convergence.

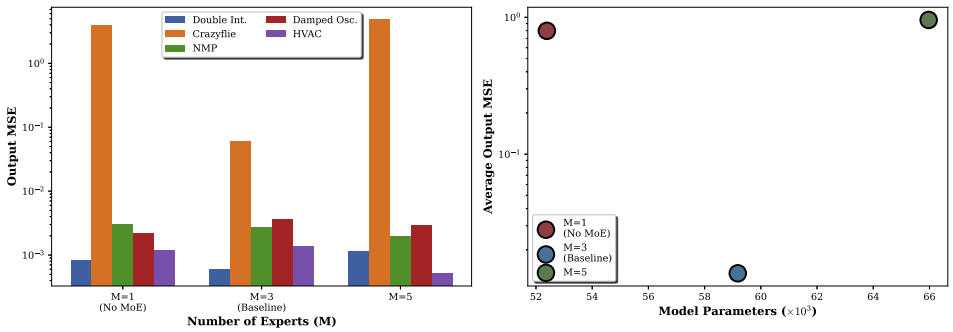


Fig. 8. Ablation study on the MOE module. Left: Output MSE comparison. Right: Trade-off between model complexity (parameters) and average output MSE.

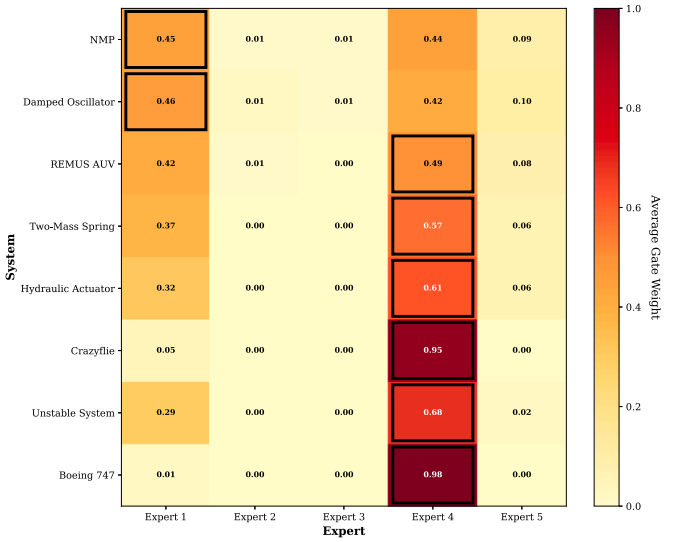


Fig. 9. Average gating weights of the mixture-of-experts controller across representative dynamical systems.

cost. For practical deployment scenarios, the baseline configuration provides near-optimal performance with four times fewer parameters than the large model variant. These findings validate the architectural design choices and demonstrate that the baseline configuration with three experts, hidden dimension of 32, and history window of 8 timesteps effectively balances performance, efficiency, and generalisation across heterogeneous dynamical systems.

8. Conclusion

This work presented the Generalist Controller, demonstrating that a single learned policy can effectively regulate diverse dynamical systems without requiring adaptation or fine-tuning after training. The controller, trained on 25 benchmark systems spanning 2D to 4D state spaces with varying dynamic characteristics, successfully controls stable, unstable, minimum-phase, and non-minimum-phase systems, including nonlinear platforms such as autonomous underwater vehicles.

The architecture integrates history-based state embeddings, reference tracking, and system tag into a unified framework that adapts control actions to system-specific dynamics. Once trained, the controller operates directly on state measurements, reference signals, and a system tag, generating appropriate control actions without requiring online model identification or iterative optimisation. Experimental validation across challenging systems confirms cross-system control with 1.05 to 3.95 s rise time, minimal overshoot (0.2 to 11.1%), and negligible steady-state error. The controller demonstrates consistent performance across different reference trajectories and setpoints beyond the unit step responses and sine responses used for evaluation. Hardware validation on a Crazyflie 2.1 + nano-quadcopter reveals successful sim-to-real transfer despite unmodelled aerodynamic effects, sensor noise, and actuator dynamics. Furthermore, evaluation under perturbed conditions not encountered during training, including actuator amplitude saturation, input rate saturation, external disturbances and measurement noise, demonstrates that the learned policy maintains stable performance whilst baseline LQI controllers exhibit degraded or unstable behaviour. These results provide evidence that the controller captures transferable control principles rather than merely memorising training data. This result demonstrates that control policies can be learned in a task-agnostic manner across system families, analogous to recent advances in natural language and computer vision.

Whilst these results represent a significant step towards generalised control, several limitations merit discussion. The controller requires a system tag, ϕ_i, at deployment to indicate which system is being controlled. The level of generalisability to unseen scenarios showed promising results, as presented in Section 6.4. However, applying the controller to completely new systems requires including those systems in the training set; thus, generalisation to entirely unseen systems is not supported. The extent of intra-system generalisability will require fur-

ther investigation. The current implementation is restricted to single-input single-output systems; extending to MIMO configurations introduces additional challenges in state-action coordination and output coupling that require further investigation. Additionally, whilst the controller demonstrates empirical robustness under perturbations, formal stability guarantees remain an open challenge. Existing stability analysis methods for neural network controllers are typically designed for single-system settings and cannot be readily extended to multi-system generalisation frameworks. Developing appropriate theoretical tools for stability certification in this context is an important direction for future work. By showing that a single neural policy can effectively control systems of varying order, stability, and dynamic characteristics whilst exhibiting robustness beyond training conditions, this work demonstrates a viable path towards generalist control policies capable of operating across diverse dynamical systems from a single learned representation.

CRediT authorship contribution statement

Klinsmann Agyei: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization; **Pouria Sarhadi:** Writing – review & editing, Visualization, Validation, Supervision, Methodology, Investigation, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Agyei, K., Sarhadi, P., & Naeem, W. (2025a). Large language model-based decision-making for COLREGs and the control of autonomous surface vehicles. In *European control conference 2025*. IEEE.

Agyei, K., Sarhadi, P., & Polani, D. (2025b). Deep reinforcement learning in applied control: challenges, analysis, and insights. arXiv preprint arXiv:2507.08196.

Argall, B. D., Chernova, S., Veloso, M., & Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, *57*(5), 469–483.

Astrom, K. J., & Wittenmark, B. (1994). Adaptive control. Addison-Wesley Longman Publishing Co., Inc.

Berkenkamp, F., Turchetta, M., Schoellig, A. P., & Krause, A. (2017). Safe model-based reinforcement learning with stability guarantees. In *Neural information processing systems*.

Bjorck, J., et al. (2025). GR00T N1: An open foundation model for generalist humanoid robots. arXiv preprint arXiv:2503.14734.

Bratko, I., Urbančič, T., & Sammut, C. (1995). Behavioural cloning: Phenomena, results and problems. *IFAC Proceedings Volumes*, *28*(21), 143–149.

Brohan, A., et al. (2022). RT-1: Robotics transformer for real-world control at scale. arXiv preprint arXiv:2212.06817.

Brown, T., et al. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, *33*, 1877–1901.

Brunton, S. L., & Kutz, J. N. (2022). Data-driven science and engineering: Machine learning, dynamical systems, and control. Cambridge University Press.

Doyle, J. (1996). Robust and optimal control. In *Proceedings of 35th IEEE conference on decision and control* (pp. 1595–1598). IEEE (*vol. 2*).

Doyle, J. C. (1982). Analysis of feedback systems with structured uncertainties. *IEE Proceedings D - Control Theory and Applications*, *129*(6), 242–250.

Inanc, E., Habboush, A., Gurses, Y., Yildiz, Y., & Annaswamy, A. M. (2025). Neural network adaptive control with long short-term memory. *International Journal of Adaptive Control and Signal Processing*, *39*(9), 1870–1885.

Ioannou, P. A., & Fidan, B. (2006). Adaptive control tutorial. (1st ed.). Philadelphia: Society for Industrial and Applied Mathematics.

Mandlekar, A., Xu, D., Wong, J., Nasiriany, S., Wang, C., Kulkarni, R., Fei-Fei, L., Savarese, S., Zhu, Y., & Martín-Martín, R. (2021). What matters in learning from offline human demonstrations for robot manipulation. arXiv preprint arXiv:2108.03298.

Mayne, D. Q., Rawlings, J. B., Rao, C. V., & Scokaert, P. O. M. (2000). Constrained model predictive control: Stability and optimality. *Automatica*, *36*(6), 789–814.

Octo Model Team (2024). Octo: An open-source generalist robot policy. Proceedings of Robotics: Science and Systems (RSS), Delft, Netherlands. Available at arXiv:2405.12213.

Pomerleau, D. A. (1988). ALVINN: An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems*, *1*, 305–313.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T. et al. (2022). A generalist agent. arXiv preprint arXiv:2205.06175.

Richards, S. M., Berkenkamp, F., & Krause, A. (2018). The Lyapunov neural network: Adaptive stability certification for safe learning of dynamical systems. In *Conference on robot learning*.

Sarhadi, P. (2025). On the standard performance criteria for applied control design: PID, MPC or machine learning controller? arXiv preprint arXiv:2503.14379.

Schaal, S. (1999). Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, *3*(6), 233–242.

Schrittwieser, J., et al. (2020). Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, *588*(7839), 604–609.

Selmic, R. R., & Lewis, F. L. (2002). Neural network approximation of piecewise continuous functions: Application to friction compensation. *IEEE Transactions on Neural Networks*, *13*(1), 64–73.

Shazeer, N., et al. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538.

Tobin, J., et al. (2017). Domain randomization for transferring deep neural networks from simulation to the real world. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 23–30. IEEE.

Torricelli, D., & Pons, J. L. (2018). EUROBENCH: Preparing robots for the real world. In *International symposium on wearable robotics* (pp. 375–378). Springer.

Vidyasagar, M. (2022). Control systems synthesis: A factorization approach. Springer Nature.

Willems, J. C. (2007). The behavioral approach to open and interconnected systems. *IEEE Control Systems Magazine*, *27*(6), 46–99.

Zare, M., Kebria, P. M., Khosravi, A., & Nahavandi, S. (2024). A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, *54*(12), 7173–7186.

Zhao, W., Queralta, J. P., & Westerlund, T. (2020). Sim-to-real transfer in deep reinforcement learning for robotics: A survey. In *2020 IEEE symposium series on computational intelligence (SSCI)* (pp. 737–744). IEEE.

Zhou, K., Doyle, J. C., & Glover, K. (1996). Robust and optimal control. Englewood Cliffs, NJ: Prentice Hall.

Zitkovich, B., et al. (2023). RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on robot learning* (pp. 2165–2183). PMLR.