In: AIAA Aviation Forum and Ascend 2025, 21-25 July 2025, Las Vegas, Nevada, USA, Paper number 2025-3548
DOI: 10.2514/6.2025-3548

Real-Time Surrogate for Quadcopter Attitude Control: A Least-Squares Quadratic Imitation Learning Approach

Sonali Rani * and Matthias Gerdts†
Universität der Bundeswehr München, Neubiberg-85579, Munich, Germany

Dmitry Ignatyev ‡
Cranfield University, College Road, Cranfield-MK43 0AL, United Kingdom

Imitation Learning (IL) has shown considerable promise in enabling automated systems to replicate expert behavior with reduced computational overhead. However, many IL techniques often suffer from limited interpretability, generalization, and the need for vast quantities of high-quality expert data. This paper presents Least-Squares based Quadratic Imitation Learning (LS-QIL), a simple yet powerful IL framework designed to overcome these limitations through a transparent and analytically tractable policy structure. LS-QIL learns a quadratic control policy using linear least-squares optimization, resulting in a closed-form mapping between system states and control actions that enables efficient and reliable deployment in real-time environments. Leveraging high-performance Nonlinear Model Predictive Control (NMPC) as the expert, LS-QIL captures complex, optimal behavior while significantly reducing online computational demands. The framework is applied to quaternion-based quadrotor attitude control, leveraging its compact and singularity-free representation to ensure robustness and stability. Through extensive simulation and analysis, the imitation learning-based framework demonstrates strong performance, adaptability, and practical feasibility, making it a promising surrogate for real-time control tasks. This research strengthens the foundations of imitation learning by bridging the gap between model-based control and learning-based methods. It offers a stable, scalable, and interpretable solution for real-time applications, highlighting a path toward scalable autonomy in aerial robotics and beyond.

Nomenclature

$f(x, u)$	:	Nonlinear state dynamics
x	:	State vector
u	:	Control input vector
n, m	:	Dimensions of the state and control input vectors
x_{ref}	:	Reference state trajectory
u_{ref}	:	Reference control trajectory
$\mathcal{D}$	:	Dataset of expert demonstrations
M	:	Number of expert demonstration samples
μ	:	Quadratic state-action mapping function
A, B, C	:	Learned weight matrix A, vector B, and scalar C of the quadratic policy
α	:	Regularization coefficient in least-squares optimization
N	:	Prediction horizon length
$\mathbf{q} = [q_0, q_1, q_2, q_3]^\top$	:	Quaternion vector representing vehicle orientation
$\mathbf{q}_d,\ \mathbf{q}_e$	:	Desired, and error quaternion vectors
$\boldsymbol{\omega} = [p, q, r]^\top$	:	Angular velocity vector in the body frame
$\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]^\top$	:	Control torque vector about body x-, y-, and z-axes
$\mathbf{I} = \text{diag}(I_x, I_y, I_z)$	:	Inertia matrix of the vehicle

*PhD Student, Institute of Applied Mathematics and Scientific Computing, Department of Aerospace Engineering, sonali.rani@unibw.de
†Professor, Institute of Applied Mathematics and Scientific Computing, Department of Aerospace Engineering, matthias.gerdts@unibw.de
‡Senior Lecturer in Flight Dynamics and Control, Faculty of Engineering and Applied Sciences, d.ignatyev@cranfield.ac.uk

Published by AIAA. This is the Author Accepted Manuscript issued with: Creative Commons Attribution License (CC:BY 4.0).
The final published version (version of record) is available online at DOI:10.2514/6.2025-3548. Please refer to any applicable publisher terms of use.

I. Introduction

The rapid evolution of quadcopter technology has paved the way for autonomous flying vehicles capable of executing predefined missions across diverse industries, such as precision agriculture, infrastructure inspection, environmental monitoring, and delivery services [1]. These applications often require drones to perform repetitive, structured tasks that are well-defined in advance. By leveraging advanced control, the vehicles can execute these missions with a high degree of accuracy, efficiency, and repeatability, minimizing human intervention. As these technologies become increasingly integral to industries, the demand for high-precision, adaptable, and real-time control solutions continues to grow, and it becomes crucial to design a robust and efficient quadcopter control.

At the core of quadcopter control lies the accurate representation of its dynamics and the development of effective controllers. The quadcopter's attitude, referring to its orientation in three-dimensional space, is typically represented through either Euler angles or quaternions [2]. Although Euler angles are intuitive, they suffer from singularities like gimbal lock, which can lead to unpredictable behavior during flight [3]. In contrast, quaternions provide a singularity-free and efficient approach to attitude representation, making them better suited for modern control applications [4].

The importance of attitude stabilization controllers is undeniable, as they form the foundation of a quadcopter's ability to maintain stable flight. An effective attitude controller must ensure fast response, precise tracking, robustness to uncertainties, and stability under real-world conditions [5]. Various model-based approaches such as proportional-integral-derivative, linear quadratic regulator, and model predictive control have been widely employed to meet these requirements [6, 7].

Despite their strengths, these control methods are inherently limited by their reliance on accurate modeling and tuning, which can hinder adaptability to real-world applications. In response to these challenges, learning-based controllers have emerged as a promising alternative. Machine learning approaches are being adopted for Unmanned Aerial Vehicles (UAV) navigation, guidance and control, enabling real-time decision-making in dynamic environments [8]. Techniques such as artificial neural networks and Reinforcement Learning (RL) offer advantages including nonlinear system identification, real-time adaptability, and enhanced robustness to external disturbances [9]. Deep RL further enhances UAV capabilities through hierarchical motion planning for obstacle-avoidance tasks [10]. In high-performance applications like autonomous drone racing, Deep Learning (DL) strategies have demonstrated low-latency responses and strong resilience in noisy, fast-changing environments [11].

Imitation Learning (IL) has emerged as an effective approach for UAVs, allowing them to learn complex control behaviors from experts, spanning tasks from model enhancement to end-to-end autonomous control [12]. For instance, Learning-from-Demonstration framework is utilized to capture aerobatic fixed-wing UAV maneuvers, encoding the demonstrated behavior using dynamic motion primitives for stable tracking [13]. Perception-aware planners like Deep-PANTHER and generative adversarial self-imitation frameworks enable UAVs to navigate dynamic environments in real time by learning from expert demonstrations while respecting sensor constraints [14, 15]. Other strategies explore model-free adaptive sliding mode control for quadcopter attitude regulation [16], contrastive self-supervised learning for policy generalization [17], and fuzzy brain emotional learning for robust control under disturbances [18].

Among all model-based high-performance quadrotor controllers, Model Predictive Control (MPC) is widely recognized for its ability to generate optimal control sequences over a receding horizon by explicitly accounting for system dynamics and constraints [4]. However, the practical deployment of MPC in real-time applications faces significant challenges, including computational overhead, sensitivity to model inaccuracies, and the need for frequent re-optimization. The growing interest in combining the model-based control with learning-based paradigms that retain MPC's optimality while enhancing efficiency and adaptability reflects the potential of hybrid strategies to achieve scalable, adaptive, and high-performance control for modern aerial systems. Typical objectives of such hybrid approaches include [12]: (i) control law approximation, (ii) inverse learning of underlying cost structures, (iii) trajectory-based imitation for reference generation, and (iv) data-driven modeling of system dynamics.

For example, [19] formulates MPC as a parameterized controller and applies probabilistic policy search to learn high-level decisions, enabling agile quadrotor maneuvers. In swarm robotics, nonlinear MPC is augmented with neural network-based downwash prediction for improved trajectory tracking in close-proximity flight, supporting safer multi-agent coordination [20]. For high-speed applications, real-time neural MPC integrates DL into MPC's predictive step, considering complex dynamics without compromising real-time feasibility [21]. Approaches like DiffTune-MPC optimize MPC's performance over extended horizons [22], while IL compresses robust MPC policies into deep networks for data-efficient learning and domain transfer [23]. Integration of moving horizon estimation with adaptive cost learning further improves nonlinear MPC's resilience to disturbances [24]. Finally, multimodal nonlinear MPC combined with RL enables real-time adaptation in unstructured environments, facilitating sim-to-real transfer for navigation and obstacle avoidance [25].

The Imitation Learning (IL) pipeline can be broadly understood through four critical phases [26]: expert demonstration, policy representation, policy learning, and policy refinement. A fundamental aspect of IL is the *expert demonstration* phase, where the quality and consistency of expert data directly impact the success of policy learning. Selecting an appropriate expert involves more than choosing a high-performing controller, it also requires evaluating whether the expert's demonstrations are quantitatively sufficient and qualitatively informative for effective learning. The next critical component of IL is *policy representation*. The policy must encode complex, nonlinear mappings between high-dimensional states and control actions. It should be expressive enough to model expert behavior accurately, yet interpretable and compact enough to facilitate generalization.

Once the policy structure is defined, the process of *policy learning* involves training the policy to replicate expert actions using supervised learning techniques. The choice of appropriate loss functions and evaluation metrics is critical for guiding the learning process and ensuring performance alignment with expert behavior. However, even after initial training, *policy refinement* is often necessary to further enhance performance through additional interaction with the environment. This stage allows the policy to adapt to subtle dynamics not captured during training and helps it reach or even surpass the performance level demonstrated by the expert.

Each of these stages plays a pivotal role in translating expert behavior into a deployable control policy, and their effectiveness determines the overall success and reliability of the learned controller.

A. Research Challenges and Objectives

While learning from demonstrations provides a structured framework for observing, interpreting, and replicating expert behavior, applying IL to high-dimensional, real-world systems introduces substantial challenges. IL leverages a range of techniques, including probabilistic models, policy gradients, decision trees, neural networks (NNs), and reinforcement learning, to model complex behaviors. Among these, NNs are often favored for their flexibility in capturing nonlinear dynamics. However, they suffer from a lack of interpretability and transparency, making it difficult to understand internal mechanisms or adjust inputs to produce desired outputs [27].

In addition to these limitations, many IL methods face challenges due to the large number of parameters that need to be trained. These include the need for large volumes of high-quality expert data, limited generalization to out-of-distribution scenarios, difficulties in balancing exploration and exploitation, and challenges in inferring reward structures, particularly in complex or unstructured tasks. These problems are amplified in safety-critical, real-time applications that demand fast, reliable decision-making [9, 11].

This research aims to develop an IL framework that enables real-time control while ensuring accuracy, scalability, interpretability, and stability for deployment in dynamic, high-stakes environments:

(a) Design compact and structured policies that model nonlinear behaviors while remaining interpretable and tunable for practical use.
(b) Define learning objectives and evaluation metrics that promote robustness and generalization, minimizing reliance on extensive expert datasets.
(c) Select appropriate expert controllers for training and assess whether the IL framework can replicate their strengths while improving upon their limitations.
(d) Investigate generalization performance to ensure consistent behavior in previously unseen scenarios.

B. Contributions

This work presents a versatile and statistically grounded imitation learning framework designed to replicate expert behavior across diverse systems. The core contributions are as follows:

1) *Quadratic policy representation:* A compact and expressive policy structure that captures nonlinear input-output relationships using a quadratic mapping function, balancing modeling accuracy and computational efficiency.
2) *Least-squares policy learning:* A transparent and scalable learning strategy based on minimizing the sum of squared errors between predicted and expert actions, enabling closed-form policy solutions via regularized linear least-squares optimization.
3) *Cross-platform validation:* Extension of the LS-QIL method, originally proposed for fixed-wing aircraft path following [28], to a quadrotor attitude stabilization task using quaternion-based control, demonstrating the framework's adaptability across vehicle dynamics and mission profiles.
4) *Expert-level and beyond performance with minimal overhead:* Trained on high-fidelity nonlinear MPC demonstrations, the learned policy not only matches but exceeds expert performance in some metrics, while significantly reducing the online computational demand and enabling real-time deployment.

C. Outline of the Paper

The structure of the rest of the paper is as follows:

(a) Imitation Learning from Nonlinear MPC: Control Framework presents a generic Imitation Learning (IL) control framework, followed by the nonlinear MPC expert control architecture, aiming to replicate the expert for real-time execution.

(b) Quaternion-based Quadcopter Dynamic Model describes the quadrotor dynamics using Newton-Euler equations, focusing on its rotational motion. These equations form the foundation for the attitude control strategy.

(c) Attitude Control via NMPC and I-NMPC presents the NMPC design using quaternion-based error metrics for attitude control and introduces the I-NMPC approach, where IL-derived policies approximate the expert for computational efficiency.

(d) Results evaluates the proposed IL-based attitude controller, covering the process from expert NMPC demonstrations to IL training and performance assessment across diverse simulation scenarios.

(e) Discussion analyzes the LS-QIL in terms of control accuracy, computational efficiency, and robustness.

(f) Conclusions summarizes the key findings and contributions, and discusses future directions for advancing real-time IL-based UAV control.

II. Imitation Learning from Nonlinear MPC: Control Framework

IL challenges the traditional paradigm of control design by shifting focus from manually crafted rules to data-driven imitation of expert behavior. At its core, IL requires state-action policies that generalize well to unseen scenarios, depending on well-structured task formulation, high-quality expert data, and the use of suitable policy representations. These algorithmic choices are critical in systems, where robustness, adaptability, and interpretability are essential.

Nonlinear Model Predictive Control (NMPC) offers a principled framework for optimal decision-making in dynamic systems, relying on forward-looking optimization to compute control actions. In this work, NMPC is utilized as the expert to generate high-fidelity, constraint-satisfying demonstrations for complex control tasks. This enables the IL framework to replicate the optimal behavior of NMPC while addressing one of its key limitations: computational complexity. By using imitation learning as a surrogate controller, the framework inherits NMPC's high-performance strategies while overcoming the limitation of real-time execution, ensuring both optimal decision-making and efficient deployment in dynamic, real-world environments.

This section formulates a generic IL framework capable of integrating any expert (human or algorithmic) across a range of complex control tasks, while providing insight into the internal decision-making process. The control architecture of the NMPC expert governing a nonlinear system is then presented.

A. Problem Description

To formalize the control task, the discrete-time system dynamics are described by the following difference equation:

$$x(k+1) = f(x(k), u(k)) \tag{1}$$

where $x(k) \in \mathbb{X}$ and $u(k) \in \mathbb{U}$ denote the state and control input at time step k, respectively. The sets $\mathbb{X} \subseteq \mathbb{R}^n$ and $\mathbb{U} \subseteq \mathbb{R}^m$ define the admissible states and controls.

An expert controller generates a dataset $\mathcal{D} \in (\mathbb{R}^n \times \mathbb{R}^m)^M$ comprising M data points:

$$\mathcal{D} = \{(x^{(i)}, u^{(i)}) \mid i = 1, \ldots, M\} \tag{2}$$

where $x^{(i)}$ and $u^{(i)}$ are the i-th state and corresponding control input.

To stabilize and control the system, a feedback control policy is defined as:

$$u(k) = \mu(x(k)) \tag{3}$$

where $\mu : \mathbb{R}^n \to \mathbb{R}^m$ maps the states $x(k)$ to the control input $u(k)$.

In IL, capturing nonlinear relationships requires a policy representation that is both expressive and computationally tractable. While linear policies are too simplistic and higher-degree polynomials overly complex, quadratic representations strike an effective balance, offering both expressiveness and computational efficiency [29]. Their structure supports closed-form solutions through least-squares optimization [30], making them well-suited for complex control tasks.

This study aims to derive a quadratic state-action mapping function, as defined in Eq. (3), using least-squares optimization to approximate the expert demonstrations in Eq. (2) with a compact and interpretable policy. This mapping is applied to the attitude control task, serving as a surrogate for the expert's decisions in both stabilization and tracking.

B. Imitation Learning Control

Imitation learning aims to approximate the control policy $\mu : \mathbb{X} \to \mathbb{U}$ defined in Eq. (3) by employing a quadratic function of the state vector in the form:

$$\mu(x^{(i)}) = \frac{1}{2} x^{(i)\top} A x^{(i)} + B^\top x^{(i)} + C \tag{4}$$

where $x^{(i)} \in \mathbb{R}^n$. The matrix $A \in \mathbb{R}^{n \times n}$, vector $B \in \mathbb{R}^n$, and scalar $C \in \mathbb{R}$ are the coefficients to be trained.

To learn these policy parameters A, B, and C, a regularized least-squares problem is formulated:

$$\min_{A,B,C} \quad \frac{1}{2} \sum_{i=1}^{M} \|\mu(x^{(i)}) - u^{(i)}\|_2^2 + \frac{\alpha}{2} \left(\|A\|_F^2 + \|B\|_2^2 + \|C\|_2^2 \right) \tag{5}$$

where $\alpha > 0$ is a regularization coefficient. The Frobenius norm $\|A\|_F$, $\|B\|$ and $\|C\|$ penalizes the complexity of the quadratic, linear and bias terms, respectively.

This approach, also known as Tikhonov regularization or ridge regression, balances data fitting and parameter norm minimization. Although the policy in Eq. (4) is nonlinear in x, it is linear in the policy parameters, allowing the reformulation of Eq. (5) as a standard regularized linear least-squares regression.

To express the policy in a linear form, define the parameter vector $\boldsymbol{\theta} \in \mathbb{R}^d$, and associate each state sample $x^{(i)} \in \mathbb{R}^n$ with a corresponding feature vector $\phi(x^{(i)}) \in \mathbb{R}^d$, where the dimension $d = n^2 + n + 1$. These are expressed as follows:

$$\boldsymbol{\theta} = \begin{bmatrix} \mathrm{vec}(A)^\top & B^\top & C \end{bmatrix}^\top, \quad \phi(x^{(i)}) = \begin{bmatrix} \mathrm{vec}(x^{(i)} x^{(i)\top})^\top & x^{(i)\top} & 1 \end{bmatrix}^\top$$

The quadratic policy in Eq. (4) can now be expressed as:

$$\mu(x^{(i)}) = \phi(x^{(i)})^\top \boldsymbol{\theta}$$

which is linear in $\boldsymbol{\theta}$.

For M data points, a design matrix $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{M \times d}$ and a target vector $\boldsymbol{\mathcal{U}} \in \mathbb{R}^M$ are constructed as:

$$\boldsymbol{\mathcal{X}} = \begin{bmatrix} \phi(x^{(1)})^\top & \phi(x^{(2)})^\top & \cdots & \phi(x^{(M)})^\top \end{bmatrix}^\top, \quad \boldsymbol{\mathcal{U}} = \begin{bmatrix} u^{(1)} & u^{(2)} & \cdots & u^{(M)} \end{bmatrix}^\top$$

The optimization problem in Eq. (5) reduces to:

$$\min_{\boldsymbol{\theta}} \quad \frac{1}{2} \|\boldsymbol{\mathcal{X}}\boldsymbol{\theta} - \boldsymbol{\mathcal{U}}\|_2^2 + \frac{\alpha}{2} \|\boldsymbol{\theta}\|_2^2 \tag{6}$$

which corresponds to a standard regularized linear least-squares regression.

This convex problem yields a closed-form solution computable via standard linear algebra:

$$\boldsymbol{\theta}^* = \left(\boldsymbol{\mathcal{X}}^\top \boldsymbol{\mathcal{X}} + \alpha I \right)^{-1} \boldsymbol{\mathcal{X}}^\top \boldsymbol{\mathcal{U}} \tag{7}$$

which provides an efficient and numerically stable method for estimating the parameters of the quadratic policy.

The overall Least-Squares based Quadratic Imitation Learning (LS-QIL) procedure is outlined in Algorithm 1.

Algorithm 1 Least-Squares based Quadratic Imitation Learning

1: **Define system dynamics:** Model the system using the nonlinear state equation given in Eq. (1).
2: **Collect expert demonstrations:** Obtain an expert dataset consisting of M state-action pairs as shown in Eq. (2).
3: **Define the policy representation:** Approximate the expert's control policy μ in Eq. (3) using a quadratic function as formulated in Eq. (4).
4: **Formulate the least-squares optimization problem:** Define the regularized least-squares problem to minimize the error function in Eq. (5), ensuring a balance between accuracy and regularization.
5: **Solve for $\boldsymbol{\theta}$:** Transform the regression problem to Eq. (6) and compute the optimal $\boldsymbol{\theta}^*$ using Eq. (7).
6: **Deploy learned policy:** Use the learned parameters (A^*, B^*, C^*) to approximate the expert policy in Eq. (4) as:

$$\mu^*(x) = \frac{1}{2} x^\top A^* x + B^{*\top} x + C^*$$

7: **Evaluate and refine:** Implement and validate the learned policy on test trajectories. For further improving the performance, refine the IL model by tuning the weight of each state parameter and control input.

The training of the IL quadratic policy in Eq. (3) based on the expert demonstrations is conducted in `MATLAB`.

C. Expert: Nonlinear Model Predictive Control

Nonlinear Model Predictive Control (NMPC) is an efficient optimal control technique that predicts the future behavior of a nonlinear model by formulating and solving an optimization problem at each time step. It constructs an optimal control problem (OCP) in continuous time, aiming to minimize the cost function subject to the system dynamics in Eq. (1), initial conditions and constraints over the prediction horizon N to track a given reference trajectory $(x_{\text{ref}}, u_{\text{ref}})$.

Mathematically, the optimal control problem can be expressed as:

$$\min_{x_e(k),u_e(k)} \sum_{k=n}^{n+N-1} \left(\|x_e(k)\|^2_{W_x} + \|u_e(k)\|^2_{W_u} \right) \tag{8}$$

subject to:

$$\begin{aligned} x(k+1) &= f(x(k), u(k)), && k = n, \ldots, n+N-1, \\ x(k) &\in \mathbb{X}, && k = n, \ldots, n+N, \\ u(k) &\in \mathbb{U}, && k = n, \ldots, n+N-1, \\ x(n) &= x_0 \end{aligned}$$

where $x_e(k) = x(k) - x_{\text{ref}}(k)$ and $u_e(k) = u(k) - u_{\text{ref}}(k)$. x_0 is the initial state. W_x and W_u are symmetric, positive semi-definite weight matrices for the state and control deviations, respectively.

When implementing the NMPC framework to design a specific control strategy, the optimal control problem formulated in Eq. (8) is solved using the `OCPID-DAE1` software package (user manual [31]). This package uses a single shooting approach, transforms the OCP into a finite-dimensional nonlinear programming (NLP) problem. The resulting NLP is efficiently solved using sequential quadratic programming techniques. `OCPID-DAE1` is particularly well-suited for systems governed by differential-algebraic equations, which are prevalent in many engineering control applications.

For any given prediction horizon $N > 0$, the solution to the optimal control problem is computed by following the procedure outlined in Algorithm 2. The key steps involve predicting the system's future states, defining a cost function that incorporates trajectory tracking, control effort minimization, constraint satisfaction, and solving the resulting optimization problem to obtain the optimal control sequence.

The advantages of NMPC include its ability to handle complex systems with nonlinear dynamics, making it ideal for applications like robotics, chemical processes, automotive systems etc., where linear models are insufficient. However, its primary limitation lies in the high computational cost due to the iterative nature of solving nonlinear optimization problems, requiring substantial resources and advanced algorithms. To mitigate this, the surrogate model based on least-squares based quadratic imitation learning (LS-QIL), introduced in Subsection II.B, efficiently approximates the expert's control law and significantly reduces the computational overhead while maintaining accuracy.

Algorithm 2 Nonlinear Model Predictive Control (NMPC)

1: **Initialize:** Set $k = 0$ and $x(k) = x_0$.
2: **Measure state:** Obtain the current state $x(k)$.
3: **Formulate OCP:** Define and set up the optimization problem as in Eq. (8), minimizing the cost over a prediction horizon of N steps.
4: **Solve OCP:** Use `OCPID-DAE1` to compute the optimal control sequence:

$$\{u^*(k),\ u^*(k+1),\ \ldots,\ u^*(k+N-1)\}$$

5: **Apply control input:** Apply the first control action $u^*(k)$ to the system.
6: **State update:** System evolves to next state using dynamics:

$$x(k+1) = f(x(k), u^*(k))$$

7: **Increment the step:** Set $k + 1 \rightarrow k$ and return to Step 2.

In the LS-QIL framework, expert demonstrations in Eq. (2) are generated by solving the constrained optimal control problem in Eq. (8) using `OCPID-DAE1` in `Fortran`, as outlined in Algorithm 2.

III. Quaternion-based Quadcopter Dynamic Model

The dynamics of the quadrotor can be described using Newton-Euler equations, which capture its rotational motion in three-dimensional space. The rotational dynamics are governed by externally applied torques, controlling the vehicle's orientation along roll, pitch, and yaw axes. This section presents the quadrotor's rotational equations of motion, which form the foundation for designing control strategies to stabilize the vehicle in practical scenarios.

A. Quaternion-based Attitude Representation

Quaternions provide an optimal solution for representing three-dimensional rotations, addressing the limitations of traditional Euler angles, such as singularities like gimbal lock [3]. They offer a compact, singularity-free, and computationally efficient representation of rotations, which is essential for maintaining stability and robustness in dynamic systems [32].

A quaternion can be expressed as:

$$\mathbf{q} = q_0 + q_1\hat{i} + q_2\hat{j} + q_3\hat{k}$$

where q_0 is the scalar part, and q_1, q_2, q_3 are the vector components.

A unit quaternion represents the orientation of a rigid body relative to a reference frame. It satisfies the condition:

$$|\mathbf{q}| = \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} = 1$$

This ensures that the quaternion represents a pure rotation without scaling.

The quaternion multiplication, performed by Kronecker product, is a non-commutative operation used to combine two quaternions. Given two quaternions, $\mathbf{q}_1 = [q_{1_0}, q_{1_1}, q_{1_2}, q_{1_3}]^\top$ and $\mathbf{q}_2 = [q_{2_0}, q_{2_1}, q_{2_2}, q_{2_3}]^\top$, their product is another quaternion given by:

$$\mathbf{q}_1 \otimes \mathbf{q}_2 = \begin{bmatrix} q_{1_0}q_{2_0} - q_{1_1}q_{2_1} - q_{1_2}q_{2_2} - q_{1_3}q_{2_3} \\ q_{1_0}q_{2_1} + q_{1_1}q_{2_0} + q_{1_2}q_{2_3} - q_{1_3}q_{2_2} \\ q_{1_0}q_{2_2} - q_{1_1}q_{2_3} + q_{1_2}q_{2_0} + q_{1_3}q_{2_1} \\ q_{1_0}q_{2_3} + q_{1_1}q_{2_2} - q_{1_2}q_{2_1} + q_{1_3}q_{2_0} \end{bmatrix}$$

For intuitive interpretation and visualization, the quadcopter's quaternion-based orientation is often converted to Euler angles [2]. Assuming a normalized quaternion and a standard ZYX (yaw–pitch–roll) rotation order, the angles represent rotations about the body-fixed x (roll), y (pitch), and z (yaw) axes. The conversion is given by:

$$\begin{aligned} \phi &= \arctan 2\left(2(q_0q_1 + q_2q_3),\ 1 - 2(q_1^2 + q_2^2)\right) \\ \theta &= \arcsin\ (2(q_0q_2 - q_3q_1)) \\ \psi &= \arctan 2\left(2(q_0q_3 + q_1q_2),\ 1 - 2(q_2^2 + q_3^2)\right) \end{aligned}$$

B. Attitude Dynamics

The attitude dynamics of a quadrotor capture rotational motion in the body frame. This study assumes a rigid body with a diagonal and constant inertia matrix $\mathbf{I}$, neglecting aerodynamic drag, rotor gyroscopic effects, and external disturbances. Under these assumptions, the net torque vector $\boldsymbol{\tau}$ is treated as the control input for attitude regulation.

Rotational kinematics describe how a quadrotor's orientation changes based on its angular velocity. The general equations of motion for a rigid body rotating in space is given by Euler's equations:

$$\frac{d}{dt}\mathbf{L} = \boldsymbol{\tau}$$

where $\mathbf{L}$ is the angular momentum and $\boldsymbol{\tau} = [\tau_x, \tau_y, \tau_z]^\top$ is the net external torque acting on the rigid body.

Angular momentum is related to angular velocity $\boldsymbol{\omega} = [p, q, r]^\top$ by the diagonal inertia matrix $\mathbf{I} = \mathrm{diag}(I_x, I_y, I_z)$, yielding:

$$\begin{aligned} \frac{d}{dt}(\mathbf{I}\boldsymbol{\omega}) &= \boldsymbol{\tau} \\ \mathbf{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times (\mathbf{I}\boldsymbol{\omega}) &= \boldsymbol{\tau} \end{aligned}$$

Expanding the vector equation component-wise for each rotational axis gives the following equations of motion:

$$\begin{aligned} I_x\dot{p} + (I_z - I_y)qr &= \tau_x \\ I_y\dot{q} + (I_x - I_z)rp &= \tau_y \\ I_z\dot{r} + (I_y - I_x)pq &= \tau_z \end{aligned} \tag{9}$$

The rate of change of the quaternion $\dot{\mathbf{q}}$ is given by the following differential equation:

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{K}(\boldsymbol{\omega}) \otimes \mathbf{q}$$

where $\mathbf{K}(\boldsymbol{\omega})$ is the quaternion matrix associated with the angular velocity:

$$\mathbf{K}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -p & -q & -r \\ p & 0 & r & -q \\ q & -r & 0 & p \\ r & q & -p & 0 \end{bmatrix}$$

Finally, the quaternion dynamics can be derived as:

$$\begin{aligned} \dot{q_0} &= \frac{1}{2}(-pq_1 - qq_2 - rq_3) \\ \dot{q_1} &= \frac{1}{2}(pq_0 + rq_2 - qq_3) \\ \dot{q_2} &= \frac{1}{2}(qq_0 - rq_1 + pq_3) \\ \dot{q_3} &= \frac{1}{2}(rq_0 + qq_1 - pq_2) \end{aligned} \tag{10}$$

IV. Attitude Control via NMPC and I-NMPC

Attitude controllers ensure the stability and responsiveness of quadrotor flight. The goal is to minimize quaternion error by adjusting control torques affecting the roll, pitch, and yaw motions. This section first presents the development of expert NMPC for quaternion-based stabilization. Subsequently, the imitation learning-driven NMPC (I-NMPC) approach is presented, where policies learned from expert trajectories are used to approximate the NMPC.

The error ($\mathbf{q}_e$) between the desired and actual quaternion orientations is computed as [32]:

$$\mathbf{q}_e = \mathbf{q}_d \otimes \mathbf{q}$$

where $\mathbf{q}_d$ is the desired quaternion, and $\mathbf{q}$ is the current quaternion.

For the given desired Euler angles of roll (ϕ_d), pitch (θ_d) and yaw (ψ_d), components of desired quaternion $\mathbf{q}_d$ are calculated as:

$$\begin{aligned} q_{d_0} &= C_{\phi_d}\, C_{\theta_d}\, C_{\psi_d} \;+\; S_{\phi_d}\, S_{\theta_d}\, S_{\psi_d} \\ q_{d_1} &= S_{\phi_d}\, C_{\theta_d}\, C_{\psi_d} \;-\; C_{\phi_d}\, S_{\theta_d}\, S_{\psi_d} \\ q_{d_2} &= C_{\phi_d}\, S_{\theta_d}\, C_{\psi_d} \;+\; S_{\phi_d}\, C_{\theta_d}\, S_{\psi_d} \\ q_{d_3} &= C_{\phi_d}\, C_{\theta_d}\, S_{\psi_d} \;-\; S_{\phi_d}\, S_{\theta_d}\, C_{\psi_d} \end{aligned}$$

where:

$$C_{\phi_d} = \cos\left(\frac{\phi_d}{2}\right), \quad C_{\theta_d} = \cos\left(\frac{\theta_d}{2}\right), \quad C_{\psi_d} = \cos\left(\frac{\psi_d}{2}\right)$$

$$S_{\phi_d} = \sin\left(\frac{\phi_d}{2}\right), \quad S_{\theta_d} = \sin\left(\frac{\theta_d}{2}\right), \quad S_{\psi_d} = \sin\left(\frac{\psi_d}{2}\right)$$

The quaternion error $\mathbf{q}_e$ is then computed by performing the Kronecker product:

$$\begin{aligned} q_{e_0} &= \;\; q_{d_0}q_0 + q_{d_1}q_1 + q_{d_2}q_2 + q_{d_3}q_3 \\ q_{e_1} &= -q_{d_0}q_1 + q_{d_1}q_0 - q_{d_2}q_3 + q_{d_3}q_2 \\ q_{e_2} &= -q_{d_0}q_2 + q_{d_1}q_3 + q_{d_2}q_0 - q_{d_3}q_1 \\ q_{e_3} &= -q_{d_0}q_3 - q_{d_1}q_2 + q_{d_2}q_1 + q_{d_3}q_0 \end{aligned} \tag{11}$$

A. Nonlinear MPC

To model the system dynamics, the state vector $\mathbf{X}$ and control input vector $\mathbf{U}$ are defined as follows:

$$\mathbf{X} = [p, q, r, q_0, q_1, q_2, q_3]^\top, \qquad \mathbf{U} = \left[\tau_x, \tau_y, \tau_z\right]^\top$$

The NMPC optimization problem in Eq. (8) is formulated as:

$$\min_{\mathbf{q}_e, \mathbf{U}} \quad \sum_{k=0}^{N-1} \left(\|\omega(k)\|^2_{W_{X_1}} + \|\mathbf{q}_e(k)\|^2_{W_{X_2}} + \|\mathbf{U}(k)\|^2_{W_U} \right) \tag{12}$$

subject to

$$\begin{aligned}
&\text{system dynamics:} && \text{Equations (9) and (10)} \\
&\text{state constraints:} && \|\mathbf{q}\| = 1, \\
&\text{control constraints:} && -1 \le \tau_x \le 1, \quad -1 \le \tau_y \le 1, \quad -1 \le \tau_z \le 1 \quad \text{N-m} \\
&\text{initial conditions:} && \mathbf{X}(0) = \mathbf{X}_0
\end{aligned}$$

Here, W_{X_1}, W_{X_2}, and W_U are weight matrices for the angular velocity, quaternion error and torque, respectively.

The quadrotor model parameters are set as $I_{xx} = I_{yy} = 6.5 \times 10^{-4}\,\text{kg}\cdot\text{m}^2$ and $I_{zz} = 1.2 \times 10^{-3}\,\text{kg}\cdot\text{m}^2$ [32]. NMPC aims to minimize the quaternion error defined in Eq.(11) by regulating control inputs over a prediction horizon N, as formulated in the optimization problem in Eq.(12). It is implemented with the decision variables given in Table 1.

Table 1 NMPC decision variables of quadcopter stability controller

Parameter	W_{X_1}	W_{X_2}	W_U	N	Timestep
Value	$\text{diag}(0.003, 0.003, 0.003)$	$\text{diag}(0.1, 0.1, 0.1, 0.3)$	$\text{diag}(5, 5, 5)$	21	0.6 sec

B. Imitation Learning-Driven NMPC

This subsection outlines the construction of an IL-driven surrogate controller designed to mimic the expert NMPC behavior. Each control policy is designed to control isolated component of angular motion, as depicted in Fig. 1, with parameters identified through Algorithm 1. This approach ensures a practical balance between simplicity, interpretability, and performance. The system architecture in Fig. 2 illustrates the transition from the expert NMPC to the I-NMPC (Imitation-based NMPC) framework, where pre-trained policies are deployed in place of online computation. By embedding these lightweight policies in the control loop, the I-NMPC achieves a stable and efficient attitude regulation, offering a viable alternative to computationally intensive NMPC.

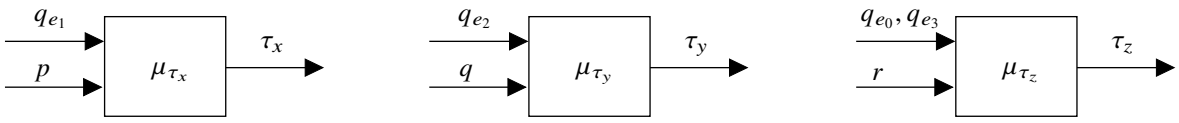


Fig. 1 Input-output mappings for roll, pitch, and yaw torque

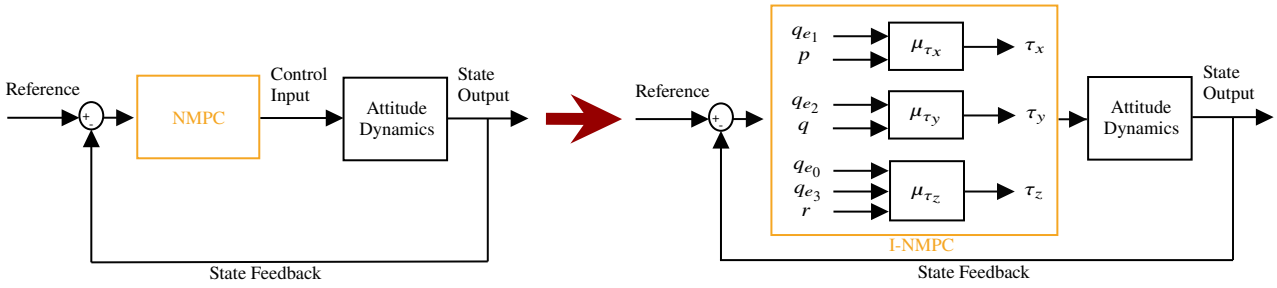


Fig. 2 Block diagram of quadcopter attitude controller using NMPC and I-NMPC

The I-NMPC policies (μ_τ) are trained using `MATLAB`, and the IL controller is developed within `MATLAB Simulink`.

V. Results

This section presents the evaluation of the proposed IL-driven attitude controller for a quadcopter. It begins with expert demonstrations using NMPC to stabilize the quadcopter from given initial attitudes. An IL policy is then trained on this expert data to serve as a surrogate controller. Finally, the IL controller is evaluated through simulation under various test conditions.

A. Expert Demonstrations

The expert demonstrations are generated under controlled hover conditions as a trajectory optimizer. This setup simplifies the training process while capturing the core state-action behavior necessary for stable flight. To generate expert data, NMPC stabilizes the quadcopter from an initial orientation of $[-30°,\ 30°,\ -30°]$ in roll, pitch, and yaw. The goal is to return the system to the zero-attitude equilibrium, demonstrating the controller's performance. After the training, Figures 4–6 present the expert NMPC performance alongside the learned I-NMPC policy, enabling direct comparison for evaluating imitation accuracy and control effectiveness.

B. Training of Imitation NMPC

The training process derives quadratic functions for the policy to best fit the expert demonstration data. This subsection details the training procedure for each control variable, emphasizing the state dependencies essential for accurate policy learning.

1) **Roll torque control** (τ_x) regulates the roll dynamics and its IL policy uses the quaternion error (q_{e_1}) and roll rate (p) via a quadratic mapping. The expression for τ_x is computed as follows:

$$\tau_x = \frac{1}{2}\begin{bmatrix} q_{e_1} & p \end{bmatrix}\begin{bmatrix} 0.00062 & 5.214e^{-5} \\ 5.214e^{-5} & -6.712e^{-6} \end{bmatrix}\begin{bmatrix} q_{e_1} \\ p \end{bmatrix} + \begin{bmatrix} 0.0076 \\ -0.0057 \end{bmatrix}^\top \begin{bmatrix} q_{e_1} \\ p \end{bmatrix} + 3.7048e^{-5} \tag{13}$$

2) **Pitch torque control** (τ_y) regulates the pitch dynamics and its IL policy uses the quaternion error (q_{e_2}) and pitch rate (q) via a quadratic mapping. The expression for τ_y is computed as follows:

$$\tau_y = \frac{1}{2}\begin{bmatrix} q_{e_2} & q \end{bmatrix}\begin{bmatrix} -0.02507 & -0.00352 \\ -0.00352 & 1.78e^{-5} \end{bmatrix}\begin{bmatrix} q_{e_2} \\ q \end{bmatrix} + \begin{bmatrix} 0.1693 \\ -0.0074 \end{bmatrix}^\top \begin{bmatrix} q_{e_2} \\ q \end{bmatrix} - 7.07e^{-5} \tag{14}$$

3) **Yaw torque control** (τ_z) regulates the yaw dynamics and its IL policy uses the combined effects of the quaternion components (q_{e_0}, q_{e_3}), and the yaw rate (r) via a quadratic mapping. The expression for τ_z is computed as:

$$\tau_z = \frac{1}{2}\begin{bmatrix} q_{e_0} \\ q_{e_3} \\ p \end{bmatrix}^\top \begin{bmatrix} -7.38e^{-8} & 0.00012 & -1.38e^{-5} \\ 0.00012 & 0.0473 & -0.0022 \\ -1.38e^{-5} & -0.0022 & 4.94e^{-5} \end{bmatrix}\begin{bmatrix} q_{e_0} \\ q_{e_3} \\ p \end{bmatrix} + \begin{bmatrix} -3.78e^{-5} \\ 0.1388 \\ -0.021 \end{bmatrix}^\top \begin{bmatrix} q_{e_0} \\ q_{e_3} \\ p \end{bmatrix} + 9.6e^{-6} \tag{15}$$

Figure 3 compares NMPC's demonstration data with the LS-QIL fitting, showing close alignment with the expert. The mean absolute residuals between the LS-QIL and expert NMPC policies for each control axis are:

$$\text{Residuals:} \quad \tau_x = 3.123e^{-4}, \quad \tau_y = 0.00872, \quad \tau_z = 0.00381. \tag{16}$$

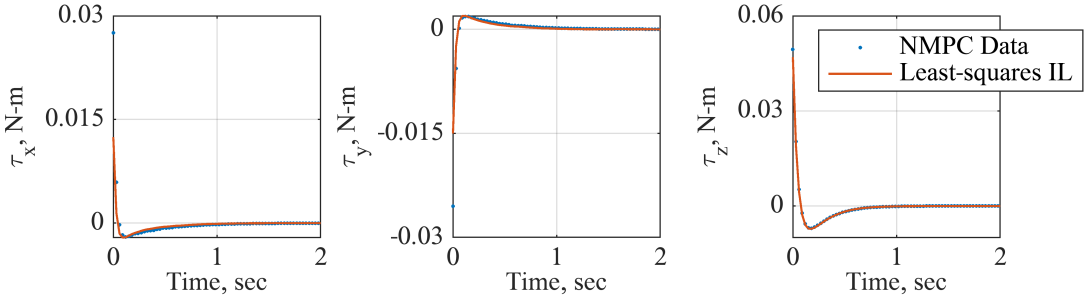


Fig. 3 Curve fitting for roll, pitch and yaw torque

C. Simulation Results

This subsection presents the performance evaluation of the surrogate controller, I-NMPC (Imitation-based NMPC), which leverages quadratic policies (13)–(15) learned from expert NMPC demonstrations. The results are compared both qualitatively and quantitatively with the expert NMPC to assess tracking accuracy and computational efficiency. The I-NMPC is evaluated across a series of representative test cases designed to examine tracking accuracy, responsiveness to dynamic inputs, and robustness under high-amplitude and high-frequency reference commands. Tracking scenarios include predefined reference profiles such as 3-2-1-1 step inputs and sinusoidal trajectories applied to each Euler angle.

1. Case 1: Stabilization Task (Comparison with Expert NMPC)

To assess imitation accuracy, the I-NMPC is initialized from the same off-nominal conditions as the expert controller, enabling a direct performance comparison under identical conditions. Figure 4 shows smooth convergence of Euler angles and body rates, demonstrating effective attitude stabilization by the I-NMPC. The quaternion trajectories are shown in Figure 5, which remain continuous and normalized, ensuring stable orientation tracking. Figure 6 illustrates well-bounded and coordinated control torques, confirming the surrogate model's ability to mimic expert NMPC behavior.

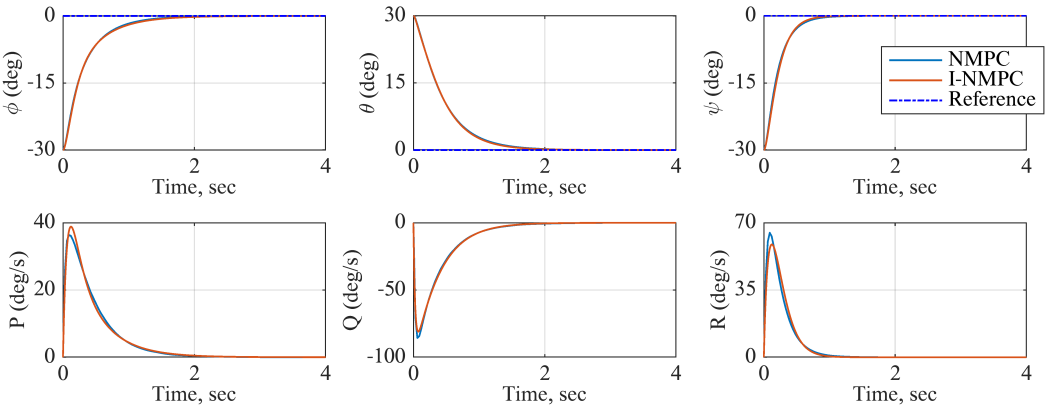


Fig. 4 Angular positions and velocities for Case 1

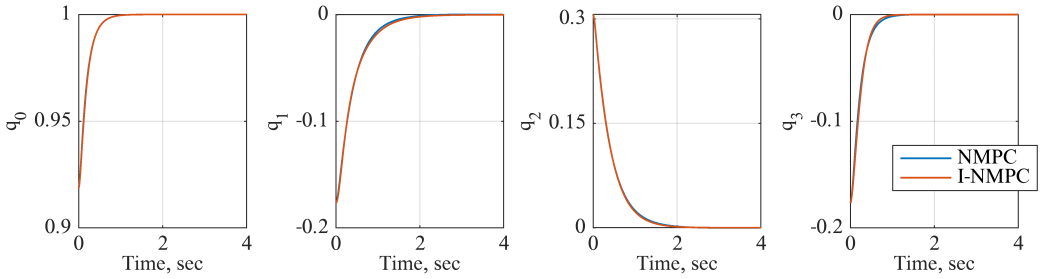


Fig. 5 Quaternion for Case 1

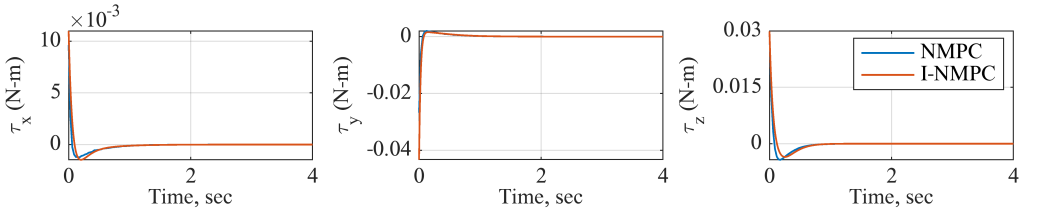


Fig. 6 Control torques for Case 1

2. Tracking Performance with Varying Reference Signals

To further evaluate the generalization capability of the I-NMPC, it is simulated with two types of attitude reference signals: 3-2-1-1 step input and smooth sinusoidal trajectory. These benchmark test the responsiveness and transient behavior of I-NMPC under sudden changes in attitude commands.

Case 2: 3-2-1-1 Step Reference

To evaluate I-NMPC under abrupt attitude transitions, a 3-2-1-1 step input sequence is applied separately to roll (ϕ), pitch (θ), and yaw (ψ), each over a 10-sec window. Each reference signal alternates between $\pm 10^{\circ}$ before returning to zero, challenging the controller's ability to respond quickly and maintain stability across all axes.

Figures 7–9 provide the results of this case. Figure 7 shows that the controller accurately follows each step in the dashed reference signal with minimal overshoot and fast settling across roll, pitch, and yaw. Figure 8 confirms stable quaternion evolution throughout the maneuver, demonstrating reliable orientation tracking without drift or discontinuities. Finally, the corresponding control torques in Figure 9 remain smooth and bounded, indicating efficient control effort without saturation.

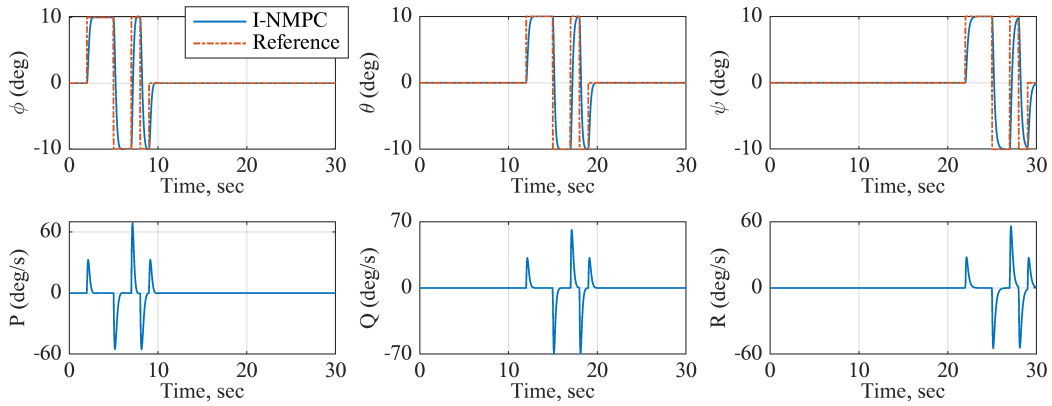


Fig. 7 Angular positions and velocities for Case 2

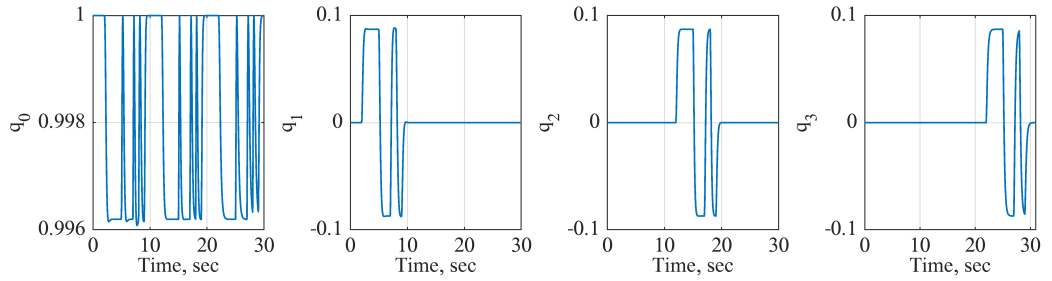


Fig. 8 Quaternion for Case 2

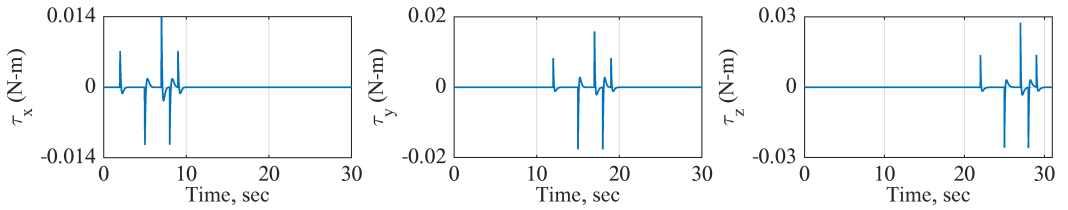


Fig. 9 Control torques for Case 2

Case 3: Sinusoidal Reference with Varying Amplitude

This scenario tests the controller's ability to follow smooth, time-varying commands with increasing levels of difficulty. A sinusoidal reference signal with a fixed frequency of 30°/s is simultaneously applied to the roll, pitch, and yaw angles, with the amplitude varied across trials from 1° to 50°, shown as dashed lines in Figure 10.

Figures 10–12 provide the results of this case. Figure 10 shows the angular positions and velocities across all amplitudes, highlighting accurate tracking for lower amplitudes and visible deviations at 50°. Figure 11 presents the quaternion evolution, confirming the system's smooth response except during high-attitude deviations. Finally, the corresponding control inputs in Figure 12 exhibit increased effort and greater fluctuations with rising amplitude.

This setup evaluates the I-NMPC's performance during continuous, coordinated maneuvers of increasing amplitude. For lower amplitudes, I-NMPC maintains excellent tracking accuracy, producing smooth angular trajectories with negligible phase lag and well-regulated control torques. However, at 50°, the performance begins to degrade. This extreme configuration represents an aggressive maneuver and goes beyond the scope of this paper. Notable discontinuities emerge in several variables, particularly around 10 sec where all three angles simultaneously reach −50°.

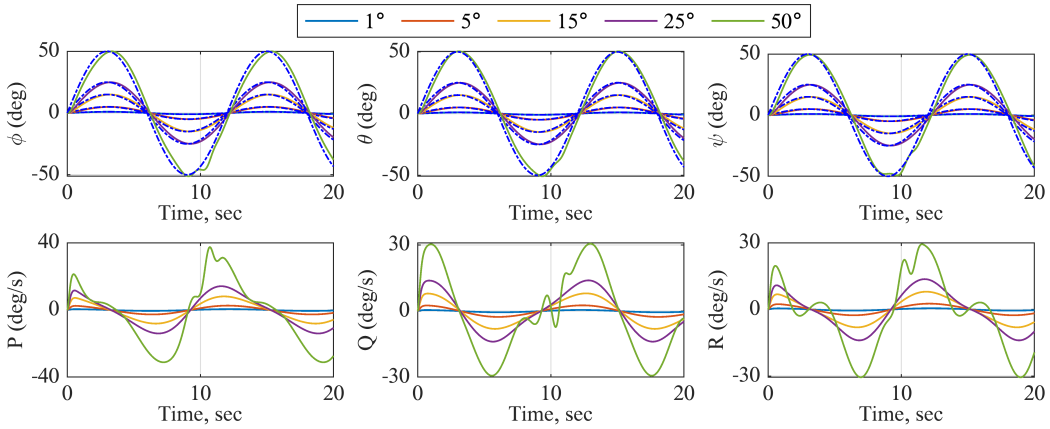


Fig. 10 Angular positions and velocities for Case 3

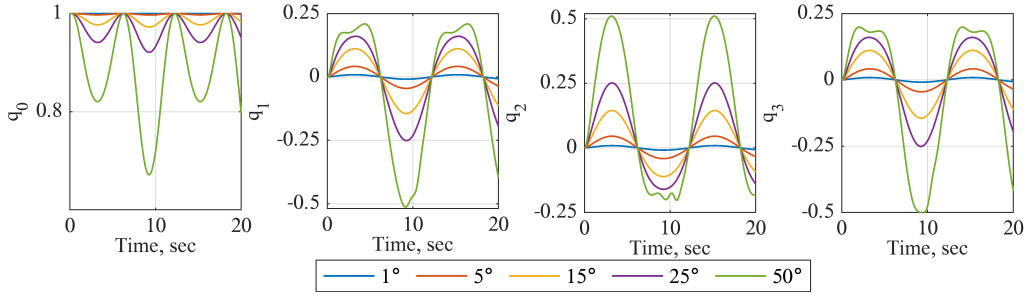


Fig. 11 Quaternion for Case 3

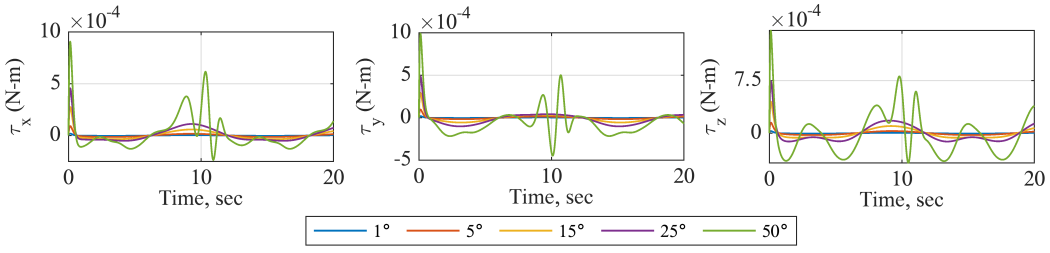


Fig. 12 Control torques for Case 3

Case 4: Sinusoidal Reference with Varying Frequency

This case evaluates the controller's frequency response by applying sinusoidal reference signals with a fixed amplitude of 10° and varying frequencies from 10°/s to 150°/s, simultaneously across all attitude axes, shown as dashed lines in Figure 13. The I-NMPC consistently maintains stable and accurate tracking throughout the frequency range.

Figures 13–15 provide the results of this case. Figure 13 shows consistent tracking across all frequencies, with minor tracking lag at 150°/s. Figure 14 confirms the stability of the quaternion evolution, even under high-frequency inputs. Finally, Figure 15 presents the associated control torques, which remain smooth and well-behaved.

At lower frequencies, the controller closely follows the desired signals with minimal error and phase delay. As the frequency increases, the tracking performance remains stable but begins to exhibit slight attenuation at the highest tested frequency of 150°/s. This minor reduction in amplitude results from the fast-changing nature of the signal; by the time the controller manages to reach the peak value, the reference shifts direction rapidly before a full response is achieved. Despite this, the I-NMPC remains stable and responsive, showcasing its competence in high-speed attitude control scenarios.

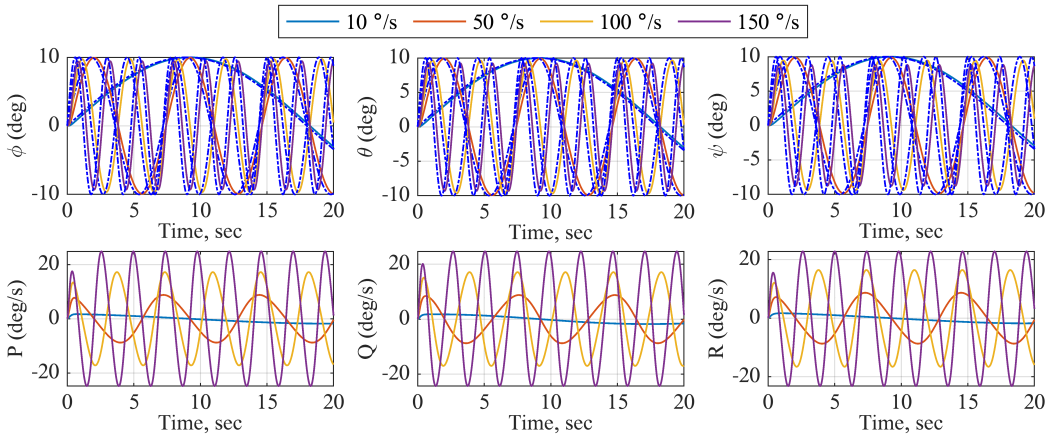


Fig. 13 Angular positions and velocities for Case 4

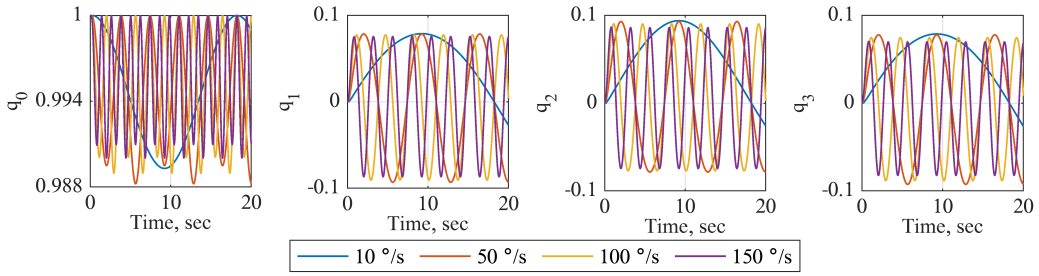


Fig. 14 Quaternion for Case 4

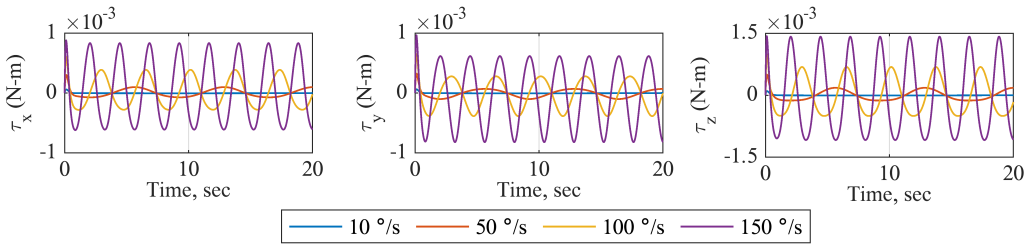


Fig. 15 Control torques for Case 4

3. RMSE Analysis

Following the qualitative evaluation of simulation results, a quantitative performance assessment of the I-NMPC is presented. Root Mean Square Error (RMSE) metrics are used to compare the tracking accuracy of Euler angles across previously presented test cases. These include comparisons with the expert NMPC, and responses to signals with varying amplitudes and frequencies, summarized in Table 2. In Case 1, the comparison shows that I-NMPC effectively replicates the expert with minimal performance loss. Case 2 exhibits notably better performance, despite the presence of structured step changes. In Cases 3 and 4, although RMSE increases with higher input amplitudes and frequencies, I-NMPC maintains stable tracking, demonstrating strong generalization under more challenging conditions.

Table 2 Tracking RMSE across different controller setups and scenarios

	Comparison		**I-NMPC**								
	Case 1		**Case 2**	**Case 3: Different Amplitude** (deg)					**Case 4: Different Frequency** (deg/s)			
Euler Angles	NMPC	I-NMPC	3-2-1-1	1	5	15	25	50	10	50	100	150
Roll (deg)	6.55	6.36	2.49	0.085	0.42	1.29	2.21	5.05	0.29	1.38	2.73	3.98
Pitch (deg)	5.75	5.59	2.36	0.082	0.41	1.24	2.06	4.16	0.28	1.33	2.63	3.81
Yaw (deg)	7.79	7.53	2.56	0.102	0.51	1.49	2.42	4.35	0.34	1.63	3.13	4.42

VI. Discussion

This section analyzes the performance, generalization capability, and computational efficiency of the proposed I-NMPC framework across a variety of control scenarios. The remarkably low residuals in Eq. (16), which measure the difference between the LS-QIL policy output and expert NMPC data, indicate that the learned policy closely replicates expert behavior with high fidelity. This strong alignment ensures that the surrogate controller achieves accurate and reliable attitude stabilization and tracking, while benefiting from a compact and computationally efficient representation.

The following discussion further examines its tracking precision and responsiveness across diverse test cases:

- **Case 1 – Stabilization task:** I-NMPC effectively stabilizes the quadcopter, demonstrating its capability to replicate expert behavior in maintaining a steady hover. When compared to the expert NMPC, the tracking accuracy of the imitation controller was within approximately 3% across all attitude axes, indicating near-identical performance with substantially reduced computational effort.

  Following the initial evaluation, I-NMPC policies were refined to reduce tracking delays and enhance accuracy.
- **Case 2 – 3-2-1-1 Step reference:** I-NMPC follows a structured sequence of sharp attitude transitions. It maintains accurate tracking with minimal overshoot and fast settling times. This case reinforces the controller's robustness in responding to abrupt, high-gradient reference changes.
- **Case 3 – Sinusoidal reference with varying amplitude:** The controller is evaluated under smooth, time-varying signals with increasing amplitude. At lower amplitudes, I-NMPC tracks the references accurately and smoothly. Some performance degradation is observed at the highest amplitude (50°), where large and simultaneous attitude changes introduce minor discontinuities. Nonetheless, the IL controller remains stable throughout.
- **Case 4 – Sinusoidal reference with varying frequency:** This scenario examines the controller's responsiveness to increasingly rapid reference changes. The I-NMPC maintains accurate tracking up to a frequency of 150°/s, with only slight lag or amplitude attenuation at higher frequencies, indicating that accuracy decreases as input dynamics become faster.

  Despite this, the IL controller shows strong competency in dynamic environments that require rapid maneuvering.

Computational performance: Both NMPC and I-NMPC were evaluated with identical simulation duration and timestep for Case 1. NMPC, implemented in `FORTRAN` via `OCPID-DAE1`, requires an average of 3.407 ms per control update. In contrast, I-NMPC running in `MATLAB Simulink` executes in just 0.015 ms per update. This corresponds to a dramatic reduction in per-cycle computation time, making I-NMPC highly suitable for real-time applications.

Implications: The quadratic policy structure not only enables efficient implementation but also enhances interpretability, allowing for straightforward tuning of state and control weighting terms to improve tracking performance. The ability of the I-NMPC to generalize from minimal expert data, captured in a simple hover task, to a range of unseen, dynamically complex trajectories, highlights the data efficiency and model-agnostic nature of proposed IL controller. This characteristic is particularly valuable for platforms where generating rich expert demonstrations is costly, time-consuming, or impractical. The ability to replicate expert-level control behavior across distinct platforms, ranging from fixed-wing [28] to multirotor systems, demonstrates the cross-platform versatility, data-efficiency, and computational feasibility of the imitation learning-driven framework.

VII. Conclusions

This research presented a simple, scalable and effective imitation learning (IL) driven framework designed to address some of the most critical challenges in control systems: interpretability, efficiency, generalization, and adaptability. The core contribution lies in integrating quadratic policy representations with linear least-squares optimization to form a data-driven controller that successfully replicates expert behavior.

The results demonstrated the effectiveness of Least-Squares based Quadratic Imitation Learning (LS-QIL) as a promising strategy for overcoming the computational challenges inherent in Nonlinear Model Predictive Control (NMPC) while maintaining high-performance control for attitude tracking. The evaluation results highlight the robustness and adaptability of the imitation-based NMPC (I-NMPC) across a range of test scenarios. From stabilization tasks to dynamic reference tracking, I-NMPC consistently demonstrated high performance. The findings suggest that LS-QIL, when combined with NMPC, offers a scalable solution for control systems in safety-critical applications, where both performance and computational efficiency are paramount. Moreover, the ability of I-NMPC to generalize from minimal expert data opens the door to new opportunities for real-world implementation in systems where generating large-scale expert demonstrations may not be feasible or practical.

The ability of LS-QIL to provide both high-level performance and a transparent decision-making process has profound implications for future research and practical deployment of autonomous systems. Future work will focus on extending the LS-QIL approach to plan and track an obstacle avoiding trajectories for a six degree-of-freedom quadcopter model, incorporating both attitude and position control, and ensuring stability through Lyapunov theory.

Acknowledgments

This research is supported by *Munich Aerospace* within the research group "Robust and Efficient Real-Time Flight Path Optimization." A sincere gratitude to Prof. Luis Rodrigues at Concordia University, Canada, for his guidance.

Declaration

During the preparation of this manuscript, the authors used ChatGPT to assist with writing and improve clarity. The authors have reviewed and edited the content and take full responsibility for the publication.

References

[1] Idrissi, M., Salami, M., and Annaz, F., "A review of quadrotor unmanned aerial vehicles: applications, architectural design and control algorithms," *Journal of Intelligent & Robotic Systems*, Vol. 104, No. 2, 2022, p. 22.

[2] Diebel, J., et al., "Representing attitude: Euler angles, unit quaternions, and rotation vectors," *Matrix*, Vol. 58, No. 15-16, 2006, pp. 1–35.

[3] Kuipers, J. B., *Quaternions and rotation sequences: a primer with applications to orbits, aerospace, and virtual reality*, Princeton university press, 1999.

[4] Zeng, W., and Chen, M., "A Comparative Study of PD, LQR and MPC on Quadrotor Using Quaternion Approach," *ScienceDirect Robotics and Autonomous Systems*, 2023.

[5] Tsourdos, A., Ignatyev, D., and Mondal, S., "Uncrewed Aerial Systems: Guidance and Control," *Reference Module in Materials Science and Materials Engineering*, Elsevier, 2024.

[6] Hossein, R., and Majid, H., "Robust Hybrid Nonlinear Control Systems for the Dynamics of a Quadcopter Drone," *Journal of Intelligent & Robotic Systems*, 2021.

[7] Huang, L., and Zhao, Y., "A Model Predictive Control (MPC) Approach on Unit Quaternion Orientation Based Quadrotor for Trajectory Tracking," *SpringerLink Autonomous Robots*, 2020.

[8] Fagundes-Junior, L. A., de Carvalho, K. B., Ferreira, R. S., and Brandão, A. S., "Machine learning for unmanned aerial vehicles navigation: An overview," *SN Computer Science*, Vol. 5, No. 2, 2024, p. 256.

[9] Nguyen, M., and Wang, T., "UAV Model-Based Flight Control with Artificial Neural Networks: A Survey," *Elsevier Aerospace Science and Technology*, 2022.

[10] Hua, H., Wang, Y., Zhong, H., Zhang, H., and Fang, Y., "Deep Reinforcement Learning-Based Hierarchical Motion Planning Strategy for Multirotors," *IEEE Transactions on Industrial Informatics*, 2025.

[11] Hanover, D., Loquercio, A., Bauersfeld, L., Romero, A., Penicka, R., Song, Y., Cioffi, G., Kaufmann, E., and Scaramuzza, D., "Autonomous drone racing: A survey," *IEEE Transactions on Robotics*, 2024.

[12] Mahmoudi, S., Davar, A., Sohrabipour, P., Bist, R. B., Tao, Y., and Wang, D., "Leveraging imitation learning in agricultural robotics: a comprehensive survey and comparative analysis," *Frontiers in Robotics and AI*, Vol. 11, 2024, p. 1441312.

[13] Cao, S., Wang, X., Zhang, R., Yu, H., and Shen, L., "From demonstration to flight: realization of autonomous aerobatic maneuvers for fast, miniature fixed-wing UAVs," *IEEE Robotics and Automation Letters*, Vol. 7, No. 2, 2022, pp. 5771–5778.

[14] Tordesillas, J., and How, J. P., "Deep-panther: Learning-based perception-aware trajectory planner in dynamic environments," *IEEE Robotics and Automation Letters*, Vol. 8, No. 3, 2023, pp. 1399–1406.

[15] Zhang, H., Huo, J., Huang, Y., Cheng, J., and Li, X., "Perception-Aware Based UAV Trajectory Planner via Generative Adversarial Self-Imitation Learning From Demonstrations," *IEEE Internet of Things Journal*, 2024.

[16] Yongjun, D., Jianhong, W., Jinlong, Z., and Xi, L., "Design of quadcopter attitude controller based on data-driven model-free adaptive sliding mode control," *International Journal of Dynamics and Control*, Vol. 12, No. 5, 2024, pp. 1404–1414.

[17] Zhao, J., Wang, Y., Cai, Z., Liu, N., Wu, K., and Wang, Y., "Learning visual representation for autonomous drone navigation via a contrastive world model," *IEEE Transactions on Artificial Intelligence*, Vol. 5, No. 3, 2023, pp. 1263–1276.

[18] Muthusamy, P. K., Suthar, B., Muthusamy, R., Garratt, M., Pota, H., Seneviratne, L., and Zweiri, Y., "Self-organizing BFBEL control system for a UAV under wind disturbance," *IEEE Transactions on Industrial Electronics*, Vol. 71, No. 5, 2023, pp. 5021–5033.

[19] Sekar, R., and Agarwal, R., "Policy Search for Model Predictive Control with Application to Agile Drone Flight," *IEEE Transactions on Robotics*, 2023.

[20] Li, J., Han, L., Yu, H., Lin, Y., Li, Q., and Ren, Z., "Nonlinear mpc for quadrotors in close-proximity flight with neural network downwash prediction," *2023 62nd IEEE Conference on Decision and Control (CDC)*, IEEE, 2023, pp. 2122–2128.

[21] Salzmann, T., Kaufmann, E., Arrizabalaga, J., Pavone, M., Scaramuzza, D., and Ryll, M., "Real-time neural MPC: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robotics and Automation Letters*, Vol. 8, No. 4, 2023, pp. 2397–2404.

[22] Tao, R., Cheng, S., Wang, X., Wang, S., and Hovakimyan, N., "Difftune-mpc: Closed-loop learning for model predictive control," *IEEE Robotics and Automation Letters*, 2024.

[23] Tagliabue, A., and How, J. P., "Efficient deep learning of robust policies from mpc using imitation and tube-guided data augmentation," *IEEE Transactions on Robotics*, 2024.

[24] Meng, C., and Li, H., "Learning-Based Nonlinear MPC With Integrated Moving Horizon Estimation of Quadrotors," *IEEE Transactions on Industrial Electronics*, 2025.

[25] Doukhi, O., and Lee, D. J., "Sim-to-Real Learning-based Nonlinear MPC for UAV Navigation and Collision Avoidance in Unknown Cluttered Environments," *IEEE Access*, 2025.

[26] Hussein, A., Gaber, M. M., Elyan, E., and Jayne, C., "Imitation learning: A survey of learning methods," *ACM Computing Surveys*, Vol. 50, No. 2, 2017, pp. 1–35.

[27] Ahmed, S. F., Alam, M. S. B., Hassan, M., Rozbu, M. R., Ishtiak, T., Rafa, N., Mofijur, M., Shawkat Ali, A., and Gandomi, A. H., "Deep learning modelling techniques: current progress, applications, advantages, and challenges," *Artificial Intelligence Review*, Vol. 56, No. 11, 2023, pp. 13521–13617.

[28] Rani, S., and Gerdts, M., "Real-Time Imitation NMPC Using Least-Squares Optimization for Aircraft Trajectory Tracking," *2024 AIAA DATC/IEEE 43rd Digital Avionics Systems Conference (DASC)*, IEEE, 2024, pp. 1–6.

[29] Rodrigues, L., and Givigi, S., "System Identification and Control Using Quadratic Neural Networks," *IEEE Control Systems Letters*, 2023.

[30] Boyd, S. P., and Vandenberghe, L., *Convex optimization*, Cambridge university press, 2004.

[31] Gerdts, M., "OCPID-DAE1: Optimal Control and Parameter Identification with Differential Algebraic Equations of Index 1," Tech. rep., Technical report, Institut für Mathematik und Rechneranwendung, Universität, 2011.

[32] Fresk, E., and Nikolakopoulos, G., "Full Quaternion Based Attitude Control for a Quadrotor," *2013 European Control Conference (ECC)*, IEEE, 2013, p. 3864.



https://dspace.org/

Real-time surrogate for quadcopter attitude control: a least-squares quadratic imitation learning approach

Rani, Sonali; Gerdts, Matthias; Ignatyev, Dmitry

2025-07-25
American Institute of Aeronautics and Astronautics (AIAA)
https://doi.org/10.2514/6.2025-3548
Rani S, Gerdts M, Ignatyev D. (2025) Real-time surrogate for quadcopter attitude control: a least-squares quadratic imitation learning approach. In: AIAA Aviation Forum and Ascend 2025, 21-25 July 2025, Las Vegas, Nevada, USA, Paper number 2025-3548

Downloaded from a DSpace repository. https://dspace.org/