Learned Incremental Nonlinear Dynamic Inversion for Quadrotors with and without Slung Payloads

Eckart Cobo-Briesewitz[1] COBO-BRIESEWITZ@CAMPUS.TU-BERLIN.DE
Khaled Wahba[1] K.WAHBA@TU-BERLIN.DE
Wolfgang Hönig[1,2] HOENIG@TU-BERLIN.DE
[1]*Technical University of Berlin, Berlin, Germany*
[2]*Robotics Institute Germany (RIG)*

Abstract

The increasing complexity of multirotor applications demands flight controllers that can accurately account for all forces acting on the vehicle. Conventional controllers model most aerodynamic and dynamic effects but often neglect higher-order forces, as their accurate estimation is computationally expensive. Incremental Nonlinear Dynamic Inversion (INDI) offers an alternative by estimating residual forces from differences in sensor measurements; however, its reliance on specialized and often noisy sensors limits its applicability. Recent work has demonstrated that residual forces can be predicted using learning-based methods. In this paper, we show that a neural network can generate smooth approximations of INDI outputs without requiring specialized rotor RPM sensor inputs. We further propose a hybrid approach that integrates learning-based predictions with INDI and demonstrate both methods for multirotors and multirotors carrying slung payloads. Experimental results on trajectory tracking errors demonstrate that the specialized sensor measurements required by INDI can be eliminated by replacing the residual computation with a neural network.

Keywords: Learning-based control, Residual force estimation, Quadrotor, Slung payload.

1. Introduction

The rise in complexity of tasks to be executed by multirotors has necessitated the development of more accurate flight controllers that are able to model all forces acting on these robots. Traditional flight controllers (Lee et al., 2010) model a large part of these forces but are unable to model all of them. The remaining forces, called residual forces, can arise from diverse sources such as blade flapping, drag, or ground forces (Shi et al., 2020, 2022; Bauersfeld et al., 2021). Trying to compute residual forces directly can be computationally complex and intractable for real time control, especially on quadrotors with restricted hardware.

Recent work has shown that learning algorithms can lead to significant improvements in flight performance (Bauersfeld et al., 2021), even modeling the interaction forces between multiple quadrotors (Shi et al., 2020). By only learning the residual forces rather than the full dynamics of the quadrotor, the amount of training data needed is greatly reduced. In addition, learning models can help improve flight performance when sensor readings are noisy or lacking.

Incremental Nonlinear Dynamic Inversion (INDI) (Smeur et al., 2016) derives the residual forces from the mismatch between the nominal dynamic model and the real-time sensor feedback, using this residual to incrementally refine the control input. In the context of Unmanned Aerial Vehicles (UAVs), INDI typically relies on rotor RPM measurements to approximate the nominal model. Although INDI can significantly enhance flight performance, its effectiveness depends on the availability of specific sensor data and may still be affected by measurement noise.

One promising application of multirotors is payload transportation, particularly in time-sensitive fields such as medicine and rescue operations. When robots carry a payload attached by a cable (Tang et al., 2014; Wahba and Hönig, 2023), the dynamics of the multirotor become more complex due to the inclusion of the payload, which introduces additional forces on the system. These added dynamics can give rise to new types of unmodeled residual forces, thereby increasing the need for accurate estimators and making residual force prediction even more crucial.

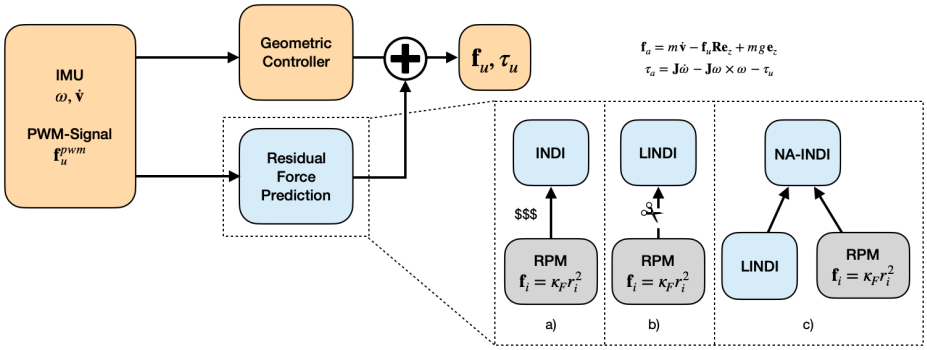


Figure 1: Overview of the three approaches for computing residual forces compared in our paper: a) Standard INDI relies on additional sensor measurements, which can be expensive and prone to noise. b) Learned INDI (LINDI) learns the outputs of INDI, eliminating the need for specialized sensors and yielding smoother predictions. c) Neural Augmented INDI (NA-INDI) fuses LINDI predictions with the original sensor data, resulting in higher-quality residual force estimates. LINDI and NA-INDI are formally defined in Section 4.

We propose Learned Incremental Nonlinear Dynamic Inversion (LINDI) a neural-network-based approach that achieves comparable flight performance to the INDI controller without requiring rotor RPM measurements. By replacing the incremental model with a learned component integrated into two traditional flight controllers (Lee et al., 2010; Yu et al., 2020), our approach removes the need for specialized RPM sensors. Quantitative results show that it matches INDI's performance, and we further analyze a hybrid setup, Neural-Augmented Incremental Nonlinear Dynamic Inversion (NA-INDI), combining both INDI and neural network predictions. Figure 1 summarizes the three evaluated approaches.

Figure 2: Hardware used for experiments. We rely on a Bitcraze Crazyflie 2.1 with standard motors, Micro-SD-card extension, and custom RPM measurement board that uses IR-LEDs. These LEDs can also be tracked by a motion capture system. The payload is equipped with another active IR LED for tracking.

2. Related Work

Several prior studies have highlighted the importance of INDI for achieving high tracking performance, particularly in high-speed flight. Specifically, incorporating the INDI component is essential for both MPC- and geometric-based controllers, with both approaches achieving comparable performance when provided with a valid reference trajectory (Sun et al., 2022). Performance can be further enhanced by modeling drag (Sun et al., 2022; Faessler et al., 2018) or motor delay (Tal and Karaman, 2021), though these factors have a comparatively smaller effect. More recently, INDI has also been shown to play a critical role in agile flight of multirotors carrying payloads (Sun et al., 2025).

An alternative to online estimation, as in INDI, is to learn a function that directly predicts the nonlinear residual dynamics. Bauersfeld et al. (2021) propose augmenting the controller with a neural network that predicts residual forces acting on a quadrotor. Their method adopts a non-Markovian assumption, using previous states as inputs rather than only the current state. The network outputs a six-dimensional vector representing the residual forces and torques along the x-, y-, and z-axes. Experimental results demonstrate that such learned residual models can significantly enhance flight performance. Similarly, Shi et al. (2022) investigate residual forces arising from inter-quadrotor interactions, particularly the downwash effect, where an upper quadrotor induces downward forces on a lower one. Their approach employs DeepSets to aggregate outputs from individual neural networks for each quadrotor, enabling the prediction of residual forces specifically along the z-axis. This method reduces the worst-case z-position error under downwash conditions by a factor of two to four.

Recent studies have explored integrating the INDI framework with learning-based models. Ignatyev and Tsourdos (2022) employ Gaussian Processes to correct sensor measurement inaccuracies in real time; their method updates the Gaussian Process online. In contrast, Zhang and Ran (2023) use meta-learning to continuously adapt residual predictions during flight, updating a neural network to estimate the INDI control effectiveness matrix. Our approach differs from previous work in that the model is trained offline and directly predicts INDI outputs, avoiding the instability that may occur when online models are not fully converged and removing the need for special sensor measurements. Additionally, unlike prior work, we validate our method through extensive real world evaluations on two different quadrotor modalities.

There is also prior work combining learning-based methods with slung-type payload transport in multirotors. Jin et al. (2024), for instance, employ a neural network to predict forces on a multirotor caused by an unknown payload. In contrast, our work assumes a known payload and focuses exclusively on residual dynamics.

The main contribution of our approach relative to previous works is that it eliminates the need for rotor RPM measurements while empirically demonstrating flight performance comparable to INDI, with smoother and less noisy residual force predictions.

3. Problem Description

We first introduce the dynamics that we consider in this work, followed by the control problems that we try to solve.

3.1. Multirotor Dynamics

Consider a multirotor modeled as a rigid floating base with state $\mathbf{x} = [\mathbf{p}, \mathbf{R}, \mathbf{v}, \boldsymbol{\omega}]^\top$. Here, $\mathbf{p}, \mathbf{v} \in \mathbb{R}^3$ represent the position and velocity in the world frame, $\mathbf{R} \in SO(3)$ represents the rotation matrix from body to world, and $\boldsymbol{\omega} \in \mathbb{R}^3$ is the angular velocity expressed in the body frame. The action $\mathbf{u} \in \mathbb{R}^4$ is defined as the angular velocities of the rotors, $\mathbf{u} = [r_1, r_2, r_3, r_4]^\top$. The dynamics are derived from Newton-Euler equations for rigid bodies as

$$\dot{\mathbf{p}} = \mathbf{v}, \qquad m\dot{\mathbf{v}} = f_u \mathbf{R}\mathbf{e}_z - mg\mathbf{e}_z + \mathbf{f}_a, \tag{1}$$

$$\dot{\mathbf{R}} = \mathbf{R}\hat{\boldsymbol{\omega}}, \qquad \mathbf{J}\dot{\boldsymbol{\omega}} = \mathbf{J}\boldsymbol{\omega} \times \boldsymbol{\omega} + \boldsymbol{\tau}_u + \boldsymbol{\tau}_a, \tag{2}$$

where m is the mass, $\mathbf{J}$ is the inertia matrix, g is the gravitational acceleration constant, $\mathbf{e}_z = [0, 0, 1]^\top$, f_u is the collective thrust created by the rotors, $\boldsymbol{\tau}_u$ is the torque vector created by the rotors, $\mathbf{f}_a, \boldsymbol{\tau}_a$ are unknown external forces and torques, respectively, and $\hat{\boldsymbol{\omega}}$ denotes the skew-symmetric matrix of $\boldsymbol{\omega}$ such that $\hat{\boldsymbol{\omega}}\mathbf{v} = \boldsymbol{\omega} \times \mathbf{v}$ for any $\mathbf{v}$.

The relationship between the motor angular velocity and the generated total force and torque on the rigid body is

$$f_i = \kappa_F r_i^2, \tag{3a}$$

$$[f_u, \boldsymbol{\tau}_u]^\top = \mathbf{B}_0 [f_1, f_2, f_3, f_4]^\top, \tag{3b}$$

where κ_F is a propeller constant and $\mathbf{B}_0$ is a fixed and known actuation matrix.

3.2. Multirotor With Cable-Suspended Payload

Consider a slung type payload attached to the center of gravity of the multirotor. The multirotors state is extended to include the payload position $\mathbf{p}_p$ and velocity $\mathbf{v}_p$. The translational dynamics are extended with additional terms:

$$m\dot{\mathbf{v}} = f_u \mathbf{R}\mathbf{e}_z - mg\mathbf{e}_z + T\mathbf{q} + \mathbf{f}_a, \tag{4}$$

$$\dot{\mathbf{p}}_p = \mathbf{v}_p, \quad m_p\dot{\mathbf{v}}_p = -T\mathbf{q} - m_p g\mathbf{e}_z \tag{5}$$

where $\mathbf{q} = \frac{\mathbf{p}_p - \mathbf{p}}{\|\mathbf{p}_p - \mathbf{p}\|}$ indicates the direction of the cable, m_p is the mass of the payload, and T is the cable tension. From (4), the tension can be computed as

$$T = -m_p \mathbf{q}^\top (\dot{\mathbf{v}}_p + g\mathbf{e}_z). \tag{6}$$

3.3. Control Problem

Given the nominal models and feasible reference trajectories $\mathbf{x}_r(t)$, we are aiming to find controllers that can minimize the mean tracking error:

$$\underset{\pi}{\operatorname{argmin}} \frac{1}{D} \int_{t=0}^{D} d(\mathbf{x}_\pi(t), \mathbf{x}_r(t)) dt, \tag{7}$$

where $\pi(\mathbf{x}, \mathbf{x}_r) \mapsto \mathbf{u}$ is the control law that influences the state $\mathbf{x}_\pi(t)$, D the duration of the reference trajectory $\mathbf{x}_r(t)$, and d is a distance metric. For the multirotor case, we consider the positional tracking error, i.e., $d(\mathbf{x}_\pi, \mathbf{x}_r) = \|\mathbf{p}_\pi - \mathbf{p}_r\|_2$ and for the payload transport case the positional tracking error of the payload.

In this work, we augment classical controllers that ignore $\mathbf{f}_a$ and $\boldsymbol{\tau}_a$ with measured and/or predicted components to improve the tracking performance (7).

4. Approach

This section introduces our two proposed methods. Learned Incremental Nonlinear Dynamic Inversion (LINDI) is a learning-based approach that eliminates the need for the additional sensor measurements required by INDI, while maintaining — or even improving — performance by learning from smoothed data. Neural-Augmented Incremental Nonlinear Dynamic Inversion (NA-INDI) combines the outputs of a neural network with INDI sensor measurements to achieve performance superior to both INDI and LINDI.

4.1. Geometric Control Laws

4.1.1. Multirotor

An exponentially stable geometric controller for a multirotor computes the desired force and torques as follows (Lee et al., 2010):

$$f_u = (-\mathbf{K}_p\mathbf{e}_p - \mathbf{K}_v\mathbf{e}_v + mg\mathbf{e}_z + m\ddot{\mathbf{p}}_d \boxed{-\mathbf{f}_a}) \cdot \mathbf{R}\mathbf{e}_z, \tag{8a}$$

$$\begin{aligned}\boldsymbol{\tau}_u = &-\mathbf{K}_R\mathbf{e}_R - \mathbf{K}_\omega\mathbf{e}_\omega - \mathbf{J}\boldsymbol{\omega}\times\boldsymbol{\omega} \\ &- \mathbf{J}(\hat{\boldsymbol{\omega}}\mathbf{R}^\top\mathbf{R}_d\boldsymbol{\omega}_d - \mathbf{R}^\top\mathbf{R}_d\dot{\boldsymbol{\omega}}_d) \boxed{-\boldsymbol{\tau}_a},\end{aligned} \tag{8b}$$

where $\mathbf{R}_d$, $\boldsymbol{\omega}_d$ and $\dot{\boldsymbol{\omega}}_d$ are desired references that can be computed using differential flatness; $\mathbf{e}_p$, $\mathbf{e}_v$, $\mathbf{e}_R$, $\mathbf{e}_\omega \in \mathbb{R}^3$ are errors with respect to this reference (mathematically defined in Lee et al. (2010)), and $\mathbf{K}_p$, $\mathbf{K}_v$, $\mathbf{K}_R$, $\mathbf{K}_\omega \in \mathbb{R}^{3\times3}$ are diagonal positive gain matrices.

The highlighted parts $\mathbf{f}_a$, $\boldsymbol{\tau}_a$ are new terms that need to be added to compensate for residual forces and torques.

4.1.2. Multirotor with cable-suspended payload

The previous control law can be extended to track a cable-suspended load (Yu et al., 2020). The controller is a cascaded design, where the first level tracks the position and velocity of the payload, the second level tracks the desired cable direction and its derivative, and the third level tracks the UAVs rotation and angular velocity. Eventually, f_u computes the control force generated by the multirotor to track the desired cable and payload motion, while attenuating the external residuals $\mathbf{f}_a$

$$f_u = (\mathbf{u}_0 \boxed{-\mathbf{f}_a}) \cdot \mathbf{R}\mathbf{e}_z, \tag{9}$$

where $\mathbf{u}_0$ is the nominal control force to track the cable as defined in Yu et al. (2020). Finally the rotational dynamics are identical to (8b), since the cable is assumed to be attached at the center of gravity and the mismatches between the nominal and real model are compensated using $\boldsymbol{\tau}_a$.

4.2. Incremental Nonlinear Dynamic Inversion (INDI)

The key idea of the Incremental Nonlinear Dynamic Inversion (INDI) is to estimate $\mathbf{f}_a$ and $\boldsymbol{\tau}_a$ in real-time using IMU and RPM sensor measurements. The INDI controller is implemented on the multirotor dynamics in Smeur et al. (2016); Tal and Karaman (2021). The dynamics of the multirotor-payload case differ from the standard case by the force applied by the cable. Therefore, re-arranging (2) and (4) yields

$$\mathbf{f}_a = m\dot{\mathbf{v}} - f_u\mathbf{R}\mathbf{e}_z + mg\mathbf{e}_z - T\mathbf{q}, \tag{10a}$$

$$\boldsymbol{\tau}_a = \mathbf{J}\dot{\boldsymbol{\omega}} - \mathbf{J}\boldsymbol{\omega}\times\boldsymbol{\omega} - \boldsymbol{\tau}_u. \tag{10b}$$

In INDI, f_u and $\boldsymbol{\tau}_u$ are computed from RPM measurements by applying (3), $\dot{\mathbf{v}}$ is measured by the accelerometer in body frame and rotated to world frame, and $\boldsymbol{\omega}$ is measured by the gyroscope. Other values like $\dot{\boldsymbol{\omega}}$ and $\dot{\mathbf{v}}_p$ (needed through (6)) are estimated numerically, while $\mathbf{R}$ and $\mathbf{q}$ are computed by the state estimation. For practical purposes, it is important to filter the measured values, e.g., by using a butterworth filter (Tal and Karaman, 2021). The online estimated values can be used as additional feedforward terms in the controllers, as highlighted in (8) and (9).

Within the INDI framework, multirotor–payload dynamics can be modeled either by explicitly including payload and cable dynamics (Sreenath et al., 2013; Yu et al., 2020; Li et al., 2023), or by capturing the coupling via cable tension (Sun et al., 2025; Sreenath and Kumar, 2013). We adopt the latter formulation in (4) due to its practical suitability. The full formulation with explicit cable dynamics and the motivation for this choice are provided in the appendix.

4.3. Learned Incremental Nonlinear Dynamic Inversion (LINDI)

Using a dataset containing example trajectories with IMU data, RPM data, and state estimates, one can also compute $\mathbf{f}_a$ and $\boldsymbol{\tau}_a$ using (10). Here, noisy data such $\dot{\mathbf{v}}$ and $\dot{\boldsymbol{\omega}}$ can be pre-processed with zero-delay filters such as spline fitting. The resulting values of $\mathbf{f}_a$ and $\boldsymbol{\tau}_a$ are the labels of a supervised learning problem.

In the single quadrotor case, we use a multi-layer perceptron (MLP) with 19 inputs, 6 outputs, 2 hidden layers with 24 dimensions each, and Leaky-ReLU activation. The input includes $\mathbf{v}$, $\dot{\mathbf{v}}$, $\boldsymbol{\omega}$, the first two columns of the rotation matrix $\mathbf{R}$, and the motor PWM signal (which is different from r_i as it cannot observe motor delays). The output is the residual force and torque $[\mathbf{f}_a, \boldsymbol{\tau}_a]^\top \in \mathbb{R}^6$. The inputs and outputs of the network are scaled using min-max normalization to the range $[-1, 1]$ to mitigate disparities in value magnitudes and enhance training stability.

For the payload network the input is expanded to include the payload velocity, acceleration and cable direction, resulting in 28 inputs. These quantities are included as they provide relevant information for estimating the residual forces acting on the payload. The architecture remains the same except for the hidden layers which are reduced to a dimension of 16. This is due to the residual forces in the payload scenario being more pronounced and making the training process easier allowing for a smaller network size.

For preprocessing, we fit splines composed of cubic polynomial segments that minimize the L_2 error with respect to the data points, rather than interpolating them exactly. These splines are applied to the collected INDI outputs, and the resulting smooth signals are used as training labels. The impact of training on smoothed data is illustrated in Figure 3. Because this smoothing is performed offline, it is more effective than the Butterworth filter used online in INDI.

4.4. Neural-Augmented Incremental Nonlinear Dynamic Inversion (NA-INDI)

We split the unmodeled dynamics in two parts $\mathbf{f}_a = \mathbf{f}_{a,NN} + \mathbf{f}_{a,INDI}$ and similar for $\boldsymbol{\tau}_a$, where $\mathbf{f}_{a,NN}$ and $\boldsymbol{\tau}_{a,NN}$ are learned functions that were trained using the steps described in 4.3. Then the INDI control law only needs to reason about the remaining mismatch in the dynamics.

5. Experiments

In order to quantify the performance of the different approaches, we perform real world flights on two unseen flight trajectories which are not present in the dataset used to train the neural network.

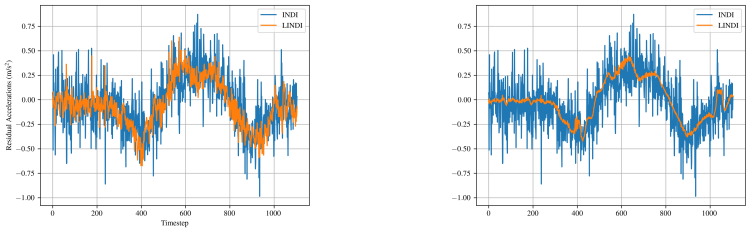
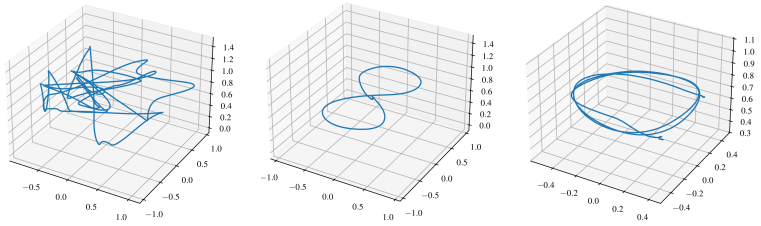


Figure 3: The figure shows the outputs of two different MLPs for the residual accelerations along the y-axis, compared with the original INDI predictions during a Figure8 flight path for a quadrotor carrying a payload. The left plot presents the predictions of an MLP trained on the raw INDI outputs, while the right plot shows the results from an MLP trained on spline-fitted INDI outputs. Although the standard MLP exhibits some inherent de-noising capability, it produces noticeably smoother predictions when trained on less noisy data, especially when using a smaller dataset for training.



(a) Random Points (b) Figure8 (c) Circle

Figure 4: The first image (a) shows a sample trajectory used to generate training data for the neural network. Random points are sampled within a predefined bounding box to define a spline trajectory, which the quadrotor follows at a speed of up to 2 m/s. Images (b) and (c) show the trajectories used for testing the performance of the different controllers.

5.1. Physical Setup

We use the Bitcraze Crazyflie 2.1 quadrotors for our experiments, a small quadrotor with an arm length of 4.6 cm and weight of 34.7 g. It is operated by an STM32 microprocessor (168 MHz, 1 MB of flash, and 192 KB RAM). We equip the standard robot with two additions: a commercially available PCB to log data on a microSD card[1], and a custom PCB to measure RPM as can be seen in Figure 2. In order to accurately predict the position of the quadrotor, we use an OptiTrack motion capture system running at 100 Hz in a 7.5 m × 4 m × 2.75 m flight space. The custom PCB to measure RPMs uses infrared emitter/receivers and can be used as an active marker by the Optitrack motion capture system to estimate the robot's pose. Rotors have reflective markers underneath the blade, which allow the IR receiver to count the rotor rotations. For communicating with the quadrotor we use Crazyswarm2 which is based on Crazyswarm (Preiss et al., 2017) but uses ROS 2 (Macenski et al., 2022).

1. https://www.bitcraze.io/products/micro-sd-card-deck/

For the payload scenario we use a 5 g weight attached at the center of the quadrotor by a string with 0.5 m length. The string is connected to the quadrotor by a magnet. The payload is also equipped with an infrared LED for estimating its position as can be seen in Figure 2. For simplicity, we keep the payload size and mass fixed, and refer the reader to prior work on adapting to varying payload parameters Jin et al. (2024).

5.2. Data Collection and Training

We train our neural networks on a set of trajectories generated by sampling random points within a predefined bounding box of dimensions $1.6 \times 1.6 \times 0.4, \mathrm{m}^3$. These points are used to define spline trajectories, which the quadrotor follows at a randomly sampled velocity of up to 2 m/s in the no-payload case and up to 1.75 m/s in the payload case.

We collected a total of 64,996 timestamps, or around 12 minutes of flight data, for the no-payload scenario each one being a pair of state and residual force and torque, and 58,538 timestamps for the payload scenario.

We use the ADAM optimizer (Kingma and Ba, 2015) with an L_1-loss and with an initial learning rate of $3 \cdot 10^{-1}$ while reducing the learning rate by a factor of 0.92 every 10 epochs to enhance training stability and convergence, training for a total of 128 epochs with a batch size of 512. The networks for the single UAV and the UAV with a payload are trained using the same procedure but with datasets corresponding to their respective scenarios. We found similar performance across hyperparameter settings in offline testing, so we omit a detailed ablation study. Our goal is to show that learning-based models can replace specialized sensor measurements, not to optimize a specific architecture.

We run all neural networks locally on the quadrotor hardware. For the MLP described in section 4.3, the inference time is 203 microseconds on the Crazyflie hardware.

5.3. Testing Scenarios

We test the performance of the controllers on two different predefined trajectories which are not present in the training set. The trajectories can be seen in Figure 4. We compute the mean tracking error (7) per trial and repeat over multiple trials (see captions). We do the same experiments for both the payload and no payload scenarios, with the only difference being the speed. For the scenario with no payload the maximum velocity was 2 m/s, while with the payload the maximum velocities were around 1.75 m/s.

5.4. Results

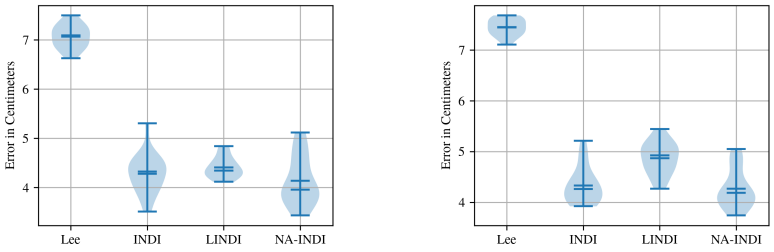
We compare our methods against the standard INDI controller and a geometric controller (Lee et al., 2010) with no residual force predictions.

5.4.1. Importance of RPM Measurements for INDI

In order to prove the reliance on RPM measurements of INDI, we test the performance of an adjusted version of INDI which uses pulse width modulation (PWM) values instead of RPM to measure the total force output of the rotors. For this experiment we train LINDI and NA-INDI on the predictions made by INDI-PWM, with results shown in Table 1. PWM measurements are noisy and introduce delay-induced mismatch in thrust estimation.

Table 1: Comparison of average deviation from desired flight path (centimeters) of the quadrotor with no payload using INDI based on thrust from the PWM signal instead of RPM. The performance of each method was averaged over ten trials.

	Circle	Figure8
Lee	7.52 ± 1.22	7.29 ± 0.31
INDI-PWM	18.27 ± 0.97	18.89 ± 1.82
LINDI-PWM	15.33 ± 1.90	$\mathbf{14.53} \pm 1.75$
NA-INDI-PWM	$\mathbf{14.31} \pm 1.91$	16.71 ± 2.80



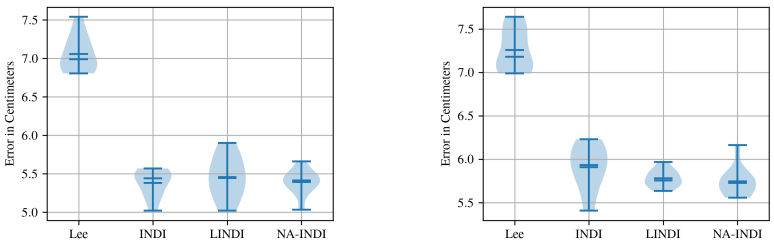
(a) Figure8 (b) Circle

Figure 5: Performance comparison between all four methods on the quadrotor with no payload for (a) the Figure8 flight trajectory and (b) the circle trajectory. Each violin plot has twenty datapoints: the mean tracking error (centimeters) of the multirotor position (7) for each trial. Mean values for Figure8: Lee: 7.07, INDI: 4.28, LINDI: 4.41, NA-INDI: **4.14**. Mean values for Circle: Lee: 7.45, INDI: 4.33, LINDI: 4.87, NA-INDI: **4.26**.

The results show that using LINDI and NA-INDI to learn the residual force calculations from the PWM-based INDI greatly improves flight performance compared to the PWM-based INDI but still underperformed when compared to the basic Lee controller with no residual prediction. This highlights INDI's dependence on accurate sensor measurements. In addition, we observe that the RPM measurements are still required for training our learning methods. The improvement over INDI-PWM with LINDI-PWM and NA-INDI-PWM indicate that the denoising capabilities of LINDI can potentially improve the performance of the standard INDI.

5.4.2. MULTIROTOR WITH NO PAYLOAD

Figure 5 shows that residual prediction can reduce the tracking error to the desired flight trajectory by around 40%. As can be seen from the results, LINDI leads to similar performance to INDI without the use of the RPM measurements, however, for the circular trajectory, we observe a slight drop in performance, likely because the trajectory is out of distribution relative to the training data. The results also show that the combination of sensor measurements plus learned residuals (NA-INDI) leads to the best performance, although with a small margin. This results from the noise in the RPM measurements being easier to filter, as the INDI component only needs to account for a smaller portion of the residuals, with the remaining effects handled by the LINDI component.



(a) Figure8 (b) Circle

Figure 6: Performance comparison between all four methods on the quadrotor carrying a payload for (a) the Figure8 flight trajectory and (b) the circle trajectory. Each violin plot has ten datapoints: the mean tracking error (centimeters) of the payload position (7) for each trial. Mean values for Figure8: Lee: 7.06, INDI: **5.38**, LINDI: 5.46, NA-INDI: 5.39. Mean values for Circle: Lee: 7.26, INDI: 5.91, LINDI: 5.78, NA-INDI: **5.74**.

5.4.3. MULTIROTOR WITH PAYLOAD

In Figure 6 we can see similar results to the single quadrotor experiments. Once again LINDI is able to match the performance of INDI without the need of RPM measurements, even with the more complex dynamics of the payload. NA-INDI shows the potential to outperform INDI, but still remains similar in performance.

6. Conclusion

We introduce LINDI, a novel approach for learning residual predictions which trains a neural network on smooth data based on spline fitting. The results demonstrate that INDI can be effectively replaced by a neural network, which can output smoother residual force predictions without requiring special sensor measurements. While the proposed method requires a quadrotor equipped with RPM measurement sensors to collect the training data, once the neural network has been trained, it can be deployed on any number of quadrotors with similar specifications. This can significantly reduce the cost of maintaining a large fleet of drones with accurate flight controllers.

This work establishes a foundation for extending learning-based residual modeling toward more realistic scenarios involving external aerodynamic disturbances. Exploring how LINDI performs in the presence of such external effects offers a promising avenue for future research and could enable robust, disturbance-aware residual prediction for real-world operations.

Acknowledgments

The work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) - 448549715.

References

L. Bauersfeld, E. Kaufmann, P. Foehn, S. Sun, and D. Scaramuzza. Neurobem: Hybrid aerodynamic quadrotor model. In *Robotics: Science and Systems XVII*. Robotics: Science and Systems

Foundation, 2021.

Matthias Faessler, Antonio Franchi, and Davide Scaramuzza. Differential flatness of quadrotor dynamics subject to rotor drag for accurate tracking of high-speed trajectories. *IEEE Robotics and Automation Letters*, 3(2):620–626, 2018. doi: 10.1109/LRA.2017.2776353.

Dmitry Ignatyev and Antonios Tsourdos. Incremental nonlinear dynamic inversion with sparse online gaussian processes adaptation for partially unknown systems. In *Mediterranean Conference on Control and Automation (MED)*, pages 233–238, 2022. doi: 10.1109/MED54222.2022.9837175.

Ao Jin, Chenhao Li, Qinyi Wang, Ya Liu, Panfeng Huang, and Fan Zhang. Neural predictor for flight control with payload. *arXiv preprint arXiv:2410.15946*, 2024. URL https://arxiv.org/abs/2410.15946.

D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.

T. Lee, M. Leok, and N. H. McClamroch. Geometric tracking control of a quadrotor uav on se(3). In *IEEE Conference on Decision and Control (CDC)*, 2010.

Guanrui Li, Xinyang Liu, and Giuseppe Loianno. Rotortm: A flexible simulator for aerial transportation and manipulation. *IEEE Transactions on Robotics*, 40:831–850, 2023.

S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall. Robot operating system 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66), 2022.

J. A. Preiss, W. Hönig, G. S. Sukhatme, and N. Ayanian. Crazyswarm: A large nano-quadcopter swarm. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3299–3304, 2017.

Guanya Shi, Wolfgang Hönig, Yisong Yue, and Soon-Jo Chung. Neural-swarm: Decentralized close-proximity multirotor control using learned interactions. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

Guanya Shi, Wolfgang Hönig, Xichen Shi, Yisong Yue, and Soon-Jo Chung. Neural-swarm2: Planning and control of heterogeneous multirotor swarms using learned interactions. *IEEE Transactions on Robotics*, 38(2):1063–1079, 2022. doi: 10.1109/TRO.2021.3098436.

Ewoud J. J. Smeur, Qiping Chu, and Guido C. H. E. Croon. Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *American Institute of Aeronautics and Astronautics*, 39(3):450–461, 2016. doi: 10.2514/1.G001490.

Koushil Sreenath and Vijay R. Kumar. Dynamics, control and planning for cooperative manipulation of payloads suspended by cables from multiple quadrotor robots. In *Robotics: Science and Systems*, 2013.

Koushil Sreenath, Taeyoung Lee, and Vijay Kumar. Geometric control and differential flatness of a quadrotor uav with a cable-suspended load. In *IEEE conference on decision and control*, pages 2269–2274, 2013.

Sihao Sun, Angel Romero, Philipp Foehn, Elia Kaufmann, and Davide Scaramuzza. A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight. *IEEE Transactions on Robotics (T-RO)*, 2022. Preprint available at https://rpg.ifi.uzh.ch/docs/TRO22_Sun.pdf.

Sihao Sun, Xuerui Wang, Dario Sanalitro, Antonio Franchi, Marco Tognon, and Javier Alonso-Mora. Agile and cooperative aerial manipulation of a cable-suspended load. *Science Robotics*, 10(107):eadu8015, 2025.

Ezra Tal and Sertac Karaman. Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness. *IEEE Transactions on Control Systems Technology*, 29(3):1203–1218, 2021. doi: 10.1109/TCST.2020.3001117.

Sarah Tang, Koushil Sreenath, and Vijay Kumar. Aggressive maneuvering of a quadrotor with a cable-suspended payload. In *Robotics: Science and Systems, Workshop on Women in Robotics*, pages 1–20, 2014.

Khaled Wahba and Wolfgang Hönig. Efficient optimization-based cable force allocation for geometric control of a multirotor team transporting a payload. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1234–1240, 2023. doi: 10.1109/ICRA48506.2023.1234567.

Beomyeol Yu, Kanishke Gamagedara, Seungkeun Kim, Taeyoung Lee, and Jinyoung Suk. Geometric control and experimental validation for a quadrotor uav transporting a payload. In *IEEE Conference on Decision and Control (CDC)*, pages 201–207, 2020.

Xinyue Zhang and Maopeng Ran. Meta-learning-based incremental nonlinear dynamic inversion control for quadrotors with disturbances. *Applied Sciences*, 13(21):11844, 2023. doi: 10.3390/app132111844. Special Issue: Intelligent Unmanned System Technology and Application.

Appendix A. Cable-Suspended Payload Dynamics

The dynamics of a single quadrotor transporting a point-mass payload via a cable modeled as a rigid rod are governed by the following equations. According to Tang et al. (2014); Sreenath et al. (2013); Yu et al. (2020); Li et al. (2023), the coupled quadrotor–payload dynamics can be expressed from Hamilton's principle as

$$\begin{aligned}
&m_T \mathbf{a}_p = \mathbf{u}^{\parallel} - ml\|\boldsymbol{\Omega}\|^2 \mathbf{q} - m_T g \mathbf{e}_z + \mathbf{f}_{a_p}, \\
&\dot{\mathbf{q}} = \boldsymbol{\Omega} \times \mathbf{q}, \\
&\dot{\boldsymbol{\Omega}} = -\frac{1}{ml} \hat{\mathbf{q}}_1 \mathbf{u}^{\perp} + \mathbf{f}_{a_c}, \\
&\dot{\mathbf{R}} = \mathbf{R}\hat{\boldsymbol{\omega}}, \\
&\mathbf{J}\dot{\boldsymbol{\omega}} = \mathbf{J}\boldsymbol{\omega} \times \boldsymbol{\omega} + \boldsymbol{\tau}_u + \boldsymbol{\tau}_a,
\end{aligned} \tag{11}$$

where $m_T = m_p + m$ is the total mass of the payload–quadrotor system. The vectors $\mathbf{p}_p, \mathbf{v}_p, \mathbf{a}_p \in \mathbb{R}^3$ denote the position, velocity, and acceleration of the payload, respectively. The unit vector $\mathbf{q}$ points from the quadrotor position $\mathbf{x} \in \mathbb{R}^3$ toward the payload position $\mathbf{x}_p$, and $\boldsymbol{\Omega} \in \mathbb{R}^3$ represents the angular velocity of the cable.

The total control force applied by the quadrotor is decomposed as

$$\mathbf{u}_0 = \mathbf{u}^{\parallel} + \mathbf{u}^{\perp}, \tag{12}$$

where the parallel and perpendicular components are defined as

$$\mathbf{u}^{\parallel} = \mathbf{q}\mathbf{q}^{\top} f_u \mathbf{R}\mathbf{e}_z, \qquad \mathbf{u}^{\perp} = -\hat{\mathbf{q}}^2 f_u \mathbf{R}\mathbf{e}_z, \tag{13}$$

and f_u is the collective thrust generated by the quadrotor.

The residual aerodynamic forces acting on the system are represented by $\mathbf{f}_{a_p}$ and $\mathbf{f}_{a_c}$, applied to the payload and cable dynamics, respectively. The last two equations in (11) describe the rotational dynamics of the quadrotor.

Appendix B. Cable-Suspended Payload Geometric-INDI Controller

This section presents the complete mathematical derivation of an alternative geometric controller in Yu et al. (2020) and INDI formulation (our contribution) that explicitly includes the cable dynamics. The purpose is to provide the theoretical foundation complementing the INDI variant used in the main paper.

We adopt a geometric INDI formulation in which the payload tracking is regulated by a virtual force controller $\mathbf{F}_d$ along the cable direction, while the cable attitude is controlled by a perpendicular component. We consider a smooth reference trajectory for the payload, defined by its desired position, velocity, acceleration, and higher-order derivatives up to the n^{th} order $(\mathbf{p}_{p_d}, \mathbf{v}_{p_d}, \mathbf{a}_{p_d}, \mathbf{j}_{p_d}, \mathbf{x}_{p_d}^{(n)})$. Let

$$\mathbf{e}_p = \mathbf{p}_{p_d} - \mathbf{p}_p, \qquad \mathbf{e}_v = \mathbf{v}_{p_d} - \mathbf{v}_p, \tag{14}$$

and let the feedback gains be $k_p, k_v > 0$ (scalars) or, more generally, $\mathbf{K}_p, \mathbf{K}_v \in \mathbb{R}^{3\times3}$ positive definite (diagonal) matrices. The desired payload virtual force is

$$\mathbf{F}_d = m_T(\mathbf{a}_{p_d} + \mathbf{K}_p \mathbf{e}_p + \mathbf{K}_v \mathbf{e}_v + g\,\mathbf{e}_z). \tag{15}$$

The parallel control component is then chosen as

$$\mathbf{u}^{\parallel} = \mathbf{q}\mathbf{q}^{\top}\mathbf{F}_d + ml\|\boldsymbol{\Omega}\|^2\mathbf{q} - \mathbf{f}_{a_p}, \tag{16}$$

To regulate the cable direction, define the desired direction

$$\mathbf{q}_d = -\frac{\mathbf{F}_d}{\|\mathbf{F}_d\|}, \tag{17}$$

and let $\boldsymbol{\Omega}_d$ satisfy the kinematics $\boldsymbol{\Omega}_d = \dot{\mathbf{q}}_d \times \mathbf{q}_d$, where $\dot{\mathbf{q}}_d$ is computed with numerical differentiation from $\mathbf{q}_d$ or ideally from the differential flatness computation in Tang et al. (2014) as

$$\dot{\mathbf{q}}_d = \frac{-m_p\mathbf{j}_{p_d} + \dot{T}\mathbf{q}_d}{\|m_p\mathbf{a}_{p_d} + g\mathbf{e}_z\|}, \quad \dot{T} = -m_p(\mathbf{j}_{p_d} \cdot \mathbf{q}_d). \tag{18}$$

Define the cable attitude and angular-rate errors (consistent with Sreenath et al. (2013); Yu et al. (2020)) as

$$\mathbf{e}_q = \mathbf{q}_d \times \mathbf{q}, \quad \mathbf{e}_{\boldsymbol{\Omega}} = \boldsymbol{\Omega} + \hat{\mathbf{q}}^2\,\boldsymbol{\Omega}_d, \tag{19}$$

and choose positive gains $k_{\mathbf{q}}, k_{\boldsymbol{\Omega}} > 0$ (or $\mathbf{K}_{\mathbf{q}}, \mathbf{K}_{\boldsymbol{\Omega}}$ positive definite diagonal matrices). Using the cable dynamics in (11), a perpendicular control that stabilizes $(\mathbf{q}, \boldsymbol{\Omega}) \to (\mathbf{q}_d, \boldsymbol{\Omega}_d)$ is

$$\mathbf{u}^{\perp} = ml\,\hat{\mathbf{q}}_1(-\mathbf{K}_{\mathbf{q}}\,\mathbf{e}_q - \mathbf{K}_{\boldsymbol{\Omega}}\,\mathbf{e}_{\boldsymbol{\Omega}} - (\mathbf{q}\cdot\boldsymbol{\Omega}_d)\,\dot{\mathbf{q}} - \hat{\mathbf{q}}^2\dot{\boldsymbol{\Omega}}_d - \mathbf{f}_{a_c}), \tag{20}$$

Finally, The collective thrust of the quadrotor is defined as $f = \mathbf{u}_0 \cdot \mathbf{R}\mathbf{e}_z$, and the attitude is tracked using the geometric controller proposed in Lee et al. (2010).

B.1. INDI Formulation

In the geometric INDI framework, the residual aerodynamic forces $\mathbf{f}_{a_p}$ and $\mathbf{f}_{a_c}$ correspond to the unmodeled disturbances acting on the payload and the cable, respectively, as defined in the dynamics equations in (11). The term $\mathbf{f}_{a_p}$ captures the discrepancy between the measured acceleration of the payload $\ddot{\mathbf{x}}_p$ (left hand side) and the nominal acceleration predicted by the model (right hand side). Similarly, $\mathbf{f}_{a_c}$ is derived analogously from the measured angular acceleration of the cable $\dot{\boldsymbol{\Omega}}$. These residuals are used by the INDI controller to incrementally adjust the control inputs based on the difference between the measured and model-predicted dynamics as shown in Eqs. (16) and (20).

B.1.1. PRACTICAL IMPLEMENTATION

While the previous section provides the theoretical derivation, the practical realization of this formulation presents significant challenges. We outline below the sensing and numerical differentiation requirements, along with the sources of estimation noise that make the real-time implementation difficult on lightweight multirotors hardware. In our experimental setup, the payload position $\mathbf{p}_p$ is measured directly from the motion capture (MoCap) system. The payload acceleration $\mathbf{a}_p$ is estimated by twice numerically differentiating the measured position data, with a Butterworth filter applied after each differentiation step to obtain smooth velocity and acceleration estimates.

Similarly, the cable direction $\mathbf{q}$ is computed from the measured positions of the quadrotor and the payload. The cable angular velocity $\boldsymbol{\Omega}$ and angular acceleration $\dot{\boldsymbol{\Omega}}$ are estimated through numerical differentiation of $\mathbf{q}$ and filtered using the same Butterworth approach (applied after each

step). In practice, this introduces a trade-off between signal noise and measurement delay: a higher filter cutoff frequency yields a noisier but more responsive signal, while a lower cutoff produces smoother estimates at the cost of additional delay.

Note: In principle, one might implement an Extended Kalman Filter (EKF) for the payload and the cable state as in Sun et al. (2025). However as the payload dynamics are tightly coupled with those of the quadrotor and then the estimation leads to large matrix inversions, which are computationally demanding on the STM32 firmware of the Crazyflie platform.

B.1.2. Discussion

Building on the practical implementation discussed above, we now reflect on the limitations and potential directions for improving the INDI formulation. In principle, the proposed INDI formulation was not experimentally validated in this work, as the estimated states were affected either by significant noise introduced through numerical differentiation or by the delays introduced when using low Butterworth filter cut-off frequencies. In particular, the estimation of the cable angular acceleration $\dot{\boldsymbol{\Omega}}$ proved highly sensitive to measurement noise, even after filtering, which led to residual force estimates that were too noisy to yield reliable experimental results. Nevertheless, we include this formulation as it represents the theoretically consistent approach derived directly from the system dynamics.

In contrast, the INDI formulation used in the main paper represents a trade-off between modeling fidelity and practical feasibility. By dropping the explicit cable dynamics and relying on IMU-measured accelerations of the multirotors (which already capture the coupling effects through the cable tension) the controller assumes perfect knowledge of these effects while gaining smoother residual estimates and improved real-time stability.

A promising direction to address these challenges is to learn the residual dynamics directly using a data-driven model, where temporal input histories implicitly capture higher-order state information, thereby reducing the reliance on explicit state estimation. This analysis provides the foundation for future work on integrating learned residual models and state estimation methods for reliable INDI-based control of cable-suspended systems.