2025 IEEE International Conference on Robotics and Automation (ICRA)
May 19-23, 2025. Atlanta, USA

Flying Quadrotors in Tight Formations using Learning-based Model Predictive Control

Kong Yao Chee*, Pei-An Hsieh*, George J. Pappas, M. Ani Hsieh

***Abstract*— Flying quadrotors in tight formations is a challenging problem. It is known that in the near-field airflow of a quadrotor, the aerodynamic effects induced by the propellers are complex and difficult to characterize. Although machine learning tools can potentially be used to derive models that capture these effects, these data-driven approaches can be sample inefficient and the resulting models often do not generalize as well as their first-principles counterparts. In this work, we propose a framework that combines the benefits of first-principles modeling and data-driven approaches to construct an accurate and sample efficient representation of the complex aerodynamic effects resulting from quadrotors flying in formation. The data-driven component within our model is lightweight, making it amenable for optimization-based control design. Through simulations and physical experiments, we show that incorporating the model into a novel learning-based nonlinear model predictive control (MPC) framework results in substantial performance improvements in terms of trajectory tracking and disturbance rejection. In particular, our framework significantly outperforms nominal MPC in physical experiments, achieving a 40.1% improvement in the average trajectory tracking errors and a 57.5% reduction in the maximum vertical separation errors. Our framework also achieves exceptional sample efficiency, using only a total of 46 seconds of flight data for training across both simulations and physical experiments. Furthermore, with our proposed framework, the quadrotors achieve an exceptionally tight formation, flying with an average separation of less than 1.5 body lengths throughout the flight.**

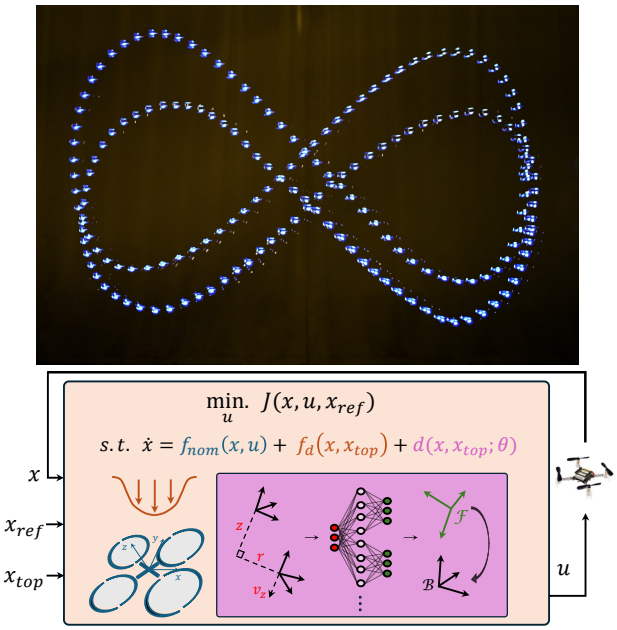


Fig. 1: **KNODE-DW MPC:** *Top*: A composite photo depicting two Crazyflie (CF) quadrotors flying in lemniscate trajectories, under a stacked formation with a commanded separation of 2 body lengths. This is achieved with our proposed framework. The frames used to create this photo are extracted from a video taken during a physical experiment. *Bottom*: Schematic of our framework. *CF image in schematic*: [16].

I. INTRODUCTION

Quadrotor teams have emerged as leading contenders for practical tasks such as surveillance [1], fighting forest fires [2], payload transportation [3], and aerial construction [4]. However, the difficulty of close proximity flights impedes critical subtasks such as docking [5], formation flight [6], and other cooperative maneuvers [7]. In particular, the complex aerodynamic effects that manifest within a quadrotor team [8] makes it challenging for them to fly in tight formations. Hence, the ability to characterize and compensate for these aerodynamic effects would enable them to achieve tighter formations and accomplish these tasks more effectively.

Related works: One possible approach to characterizing these aerodynamic effects is to derive a first-principles model based on physics and empirical data [9], [10], [11]. This model can then be used for controller design to counteract these effects or disturbances. A key limitation of this approach is that these models are often overly simplified and do not provide accurate predictions of the velocity flow field and representations of the aerodynamic effects. A recent study in [12] shows that the flow field near a quadrotor is highly complex and nonlinear, which makes the modeling of these aerodynamic effects using first principles particularly challenging. An alternative approach is to first collect data and extract a data-driven representation [8], [13]. These data-driven models, however, are significantly less sample efficient compared to first-principles models. Moreover, this approach may require an onerous data collection procedure, such as those described in [13], [14]. Instead, we leverage a state-of-the-art deep learning tool, knowledge-based neural ordinary differential equations (KNODE) [15]. This allows us to construct a knowledge-based, data-driven representation that only requires a small amount of data to train. More importantly, this representation captures the complex aerodynamic effects with unprecedented accuracy.

To enable the quadrotors to fly in close proximity, the derived models are often deployed in reactive controllers, *e.g.*, [13], [17]. These controllers consider the aerodynamic effects or disturbances at the current time step, but neglect potential disturbances that may occur within the next few

We gratefully acknowledge the support of Office of Naval Research (ONR) Award No. N00014-22-1-2157 and DSO National Laboratories, 12 Science Park Drive, Singapore 118225.

*: Equal contribution. The authors are with the GRASP Laboratory, University of Pennsylvania, Philadelphia, PA 19104, USA. {ckongyao, pahsieh, pappasg, m.hsieh}@seas.upenn.edu

2025 IEEE International Conference on Robotics and Automation (ICRA) | 979-8-3315-4139-2/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICRA55743.2025.11127756

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.

time steps. On the other hand, model predictive control (MPC) offers an opportunity to anticipate future aerodynamic disturbances, through the consideration of a prediction horizon. In [14], a learned model is incorporated into an MPC framework, allowing two quadrotors to fly close to each other. However, the tests considered in [14] are only limited to short-duration traversals between two quadrotors, and are conducted at one particular speed and separation distance. Hence, it is unclear if the framework can generalize to different flight conditions and maneuvers. In contrast, our proposed framework achieves high accuracy across a wide range of test cases. These include a particularly challenging *stacked* formation, where a quadrotor constantly flies in the wake of another quadrotor. The aerodynamic effects in this case are highly transient, making it extremely challenging for the quadrotors to achieve close proximity flight.

Contributions: The main contributions of this work are three-fold. First, we introduce a novel knowledge-based, data-driven model that accurately characterizes the complex aerodynamic disturbances within a quadrotor team. We combine analytical models of the quadrotor dynamics and aerodynamic disturbances, which we refer to as *knowledge* of the system, with a data-driven component. Second, we incorporate the combined model into a nonlinear MPC framework, as depicted in the schematic in Fig 1. We ascertain the closed-loop performance through extensive simulations. Notably, we show that our learning-based MPC framework achieves performance comparable to an omniscient MPC framework, where the model in the controller is identical to the true dynamics. Third, we demonstrate through physical experiments that our framework achieves accurate and consistent closed-loop performance across numerous test cases. In particular, our framework generalizes exceptionally well beyond the training dataset. With our framework, the quadrotors fly in an incredibly tight formation, where they maintain an average separation distance of less than 1.5 body lengths throughout the entire flight. To the best of the authors' knowledge, this is a feat never achieved before for a quadrotor team of this size and weight class.

II. PRELIMINARIES

A. Quadrotor Dynamics and Aerodynamic Disturbances

Consider a quadrotor system with the dynamics [18],

$$\begin{aligned} \dot{p} &= v, & \dot{v} &= -g + (1/m) R\tau, \\ J\dot{\omega} &= \alpha - \omega \times J\omega, & \dot{q} &= \Omega\,(q/2), \end{aligned} \tag{1}$$

where p, v, $\omega := [\omega_x, \omega_y, \omega_z]^\top \in \mathbb{R}^3$ and $q \in \mathbb{R}^4$ are the position, velocity, angular rate and quaternions representing its dynamics and kinematics. The vectors $\tau := [0\ 0\ \eta]^\top \in \mathbb{R}^3$ and $\alpha \in \mathbb{R}^3$ consist of the thrust η and torques generated by the motors of the quadrotor. The matrix $R \in \mathbb{R}^{3\times3}$ is the rotation matrix from the body $(\mathcal{B})$ to the world $(\mathcal{W})$ reference frame. The world frame follows the East-North-Up convention where the z axis is pointing upwards. The matrix $\Omega \in \mathbb{R}^{4\times4}$ is defined as

$$\Omega := \begin{bmatrix} 0 & -\omega_x & -\omega_y & -\omega_z \\ \omega_x & 0 & \omega_z & -\omega_y \\ \omega_y & -\omega_z & 0 & \omega_x \\ \omega_z & \omega_y & -\omega_x & 0 \end{bmatrix}. \tag{2}$$

The mass and inertia matrix of the quadrotor are denoted by m and $J \in \mathbb{R}^{3\times3}$ respectively. By defining the state and control inputs as $x := [p^\top\ v^\top\ q^\top\ \omega^\top]^\top$ and $u := [\eta\ \alpha^\top]^\top$, the equations of motion (1) can be written in a compact form,

$$\dot{x} = f_{nom}(x, u), \tag{3}$$

which we refer to as the nominal dynamics model.

Consider a scenario where two quadrotors fly in a tight formation where they must maintain a fixed vertical separation distance. We refer to them as the bottom and top quadrotors. The positions of the quadrotors are denoted by $p \in \mathbb{R}^3$ and $p_{top} \in \mathbb{R}^3$, and the position difference is given by $\Delta p := p - p_{top}$. By denoting the unit vector of the body frame of the top quadrotor in the z direction as e_3', we define a flow field reference frame $\mathcal{F}$ with the unit vectors $\{e_1, e_2, e_3\}$ given as

$$e_1 = e_2 \times e_3, \quad e_2 = \frac{e_3 \times \Delta p}{||e_3 \times \Delta p||_2}, \quad e_3 = -e_3', \tag{4}$$

and the rotation matrix $R^{\mathcal{F}}_{\mathcal{W}} \in \mathbb{R}^{3\times3}$ rotates a vector from $\mathcal{W}$ to $\mathcal{F}$.

To characterize the disturbances on the bottom quadrotor, we consider a velocity flow model [11] where the relative velocity of the airflow towards a point $p_f := [x_f\ y_f\ z_f]^\top \in \mathbb{R}^3$ on the bottom quadrotor in the frame $\mathcal{F}$ is given as

$$V(z, r) = \frac{U_H \frac{Bd}{\tilde{z} - z_0}}{\left(1 + \left(\sqrt{2} - 1\right)\left(\frac{\tilde{r}}{S(\tilde{z} - z_0)}\right)^2\right)^2} - v_z^{\mathcal{F}}, \tag{5}$$

where $S \in \mathbb{R}$ is the spreading factor and the velocity $v^{\mathcal{F}} = R^{\mathcal{F}}_{\mathcal{W}} v := [v_x^{\mathcal{F}}\ v_y^{\mathcal{F}}\ v_z^{\mathcal{F}}]^\top \in \mathbb{R}^3$. The radial and vertical separations between p_f and the top quadrotor in the frame $\mathcal{F}$ are denoted as r and z respectively, where $r := \sqrt{x_f^2 + y_f^2}$ and $z := z_f$. The normalized separations are given as $\tilde{r} := r/\lambda$ and $\tilde{z} := z/\lambda$. For more details on the flow model, readers are referred to [11].

Using (5), the disturbance forces $\tau_d \in \mathbb{R}^3$ and torques $\alpha_d \in \mathbb{R}^3$ are computed as in [9],

$$\tau_d(x, x_{top}) = \int_{A_b} e_3\, C_D\, \zeta\, dA, \tag{6a}$$

$$\alpha_d(x, x_{top}) = \int_{A_b} \left(p_f - R^{\mathcal{F}}_{\mathcal{W}} \Delta p\right) \times e_3\, C_D\, \zeta\, dA, \tag{6b}$$

where the dynamic pressure is denoted by $\zeta := \frac{1}{2}\rho V(z, r)^2 \in \mathbb{R}$, $C_D \in \mathbb{R}$ is the drag coefficient, and the vector $x_{top} \in \mathbb{R}^{13}$ denotes the state of the top quadrotor. The surface area of the bottom quadrotor is a circle, *i.e.*, $A_b := \pi(\lambda/2)^2$. These forces and torques (6) are succinctly expressed as

$$\begin{aligned} \dot{x}_{dw} &= \begin{bmatrix} 0_{1\times3} & \frac{\left(R^{\mathcal{F}\top}_{\mathcal{W}} \tau_d\right)^\top}{m} & 0_{1\times4} & J^{-1}\left(R^{\mathcal{F}\top}_{\mathcal{B}} \alpha_d\right)^\top \end{bmatrix}^\top \\ &:= f_d(x, x_{top}). \end{aligned} \tag{7}$$

It is important to note that the model (7) is unlikely to be highly accurate in practice, due to the complexity of the near-field airflow of the quadrotor, as discussed in [12]. This is particularly evident when the quadrotors are in a tight formation, *i.e.* when the vertical separation within the

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.

quadrotors is less than 2 body lengths. In these conditions, the flow model (5) is likely to provide accurate predictions only in the far-field airflow below the top quadrotor [11]. Given that it is challenging to derive a model that accurately characterizes the near-field airflow, our approach is to leverage this simplified model (7) as part of our prior knowledge within the overall dynamics model, and use a data-driven component to learn the complex dynamics of the aerodynamic disturbances, described in Section III-A.

B. Learning-based MPC

To control the quadrotor system, we consider the following finite horizon optimal control problem [19],

$$\min_{\{x_i\},\{u_i\}} \quad \sum_{i=0}^{N-1} ||x_i - x_{r,i}(k)||_Q^2 + ||u_i||_R^2 \tag{8a}$$

$$+ ||x_N - x_{r,N}(k)||_P^2 \tag{8b}$$

$$\text{s.t.} \quad x_{i+1} = \hat{f}(x_i, u_i; \theta^\star), \quad i = 0, \dots, N-1, \tag{8c}$$

$$x_i \in \mathcal{X}, \quad u_i \in \mathcal{U}, \quad i = 0, \dots, N-1, \tag{8d}$$

$$x_N \in \mathcal{X}_f, \quad x_0 = x(k), \tag{8e}$$

where $N \in \mathbb{N}_+$ denotes the prediction horizon and the notation $||x||_A$ denotes $x^\top Ax$. The matrices Q, R and P are the cost matrices for the state, control and terminal state costs respectively. The sets $\mathcal{X}$, $\mathcal{X}_f$ and $\mathcal{U}$ define the state, terminal state and control input constraints. The vector $x(k)$ is the state measurement at the k^{th} time step. The sequence $\{x_{r,i}(k)\}_{i=0}^N$ denotes the reference state trajectories to be tracked by the system. At each time step, the problem (8) is solved in a receding horizon manner to obtain a sequence of optimal control inputs $\{u_i^\star\}_{i=0}^{N-1}$, and the first element of the sequence $u_0^\star$ is applied as the control action to the system.

The key difference between (8) and that in a nominal MPC framework lies in the dynamics constraints (8c). In nominal MPC, a nominal model (3) is used as the dynamics constraints. However, due to the presence of unmodeled dynamics, this nominal model may not be sufficiently accurate in representing the true system dynamics, which can cause a degradation in the closed-loop performance. To alleviate this issue, a learning-based framework is applied. A model with trained parameters $\theta^\star$ is used to characterize the full dynamics, which includes the unmodeled dynamics. This model is then applied to the constraints (8c). It has been shown in prior work that a learning-based MPC framework provides significant performance improvements over nominal MPC [20], [21]. In this work, we apply a similar methodology, but with some enhancements to accommodate the aerodynamic disturbance model (7) and the predictions of the state of the top quadrotor x_{top}.

III. MODEL AND CONTROL DESIGN

A. Model Formulation and Training

For the dynamics model, we consider a state-of-the-art deep learning tool, knowledge-based neural ODEs (KNODE) [15], [22], to incorporate knowledge of the quadrotor dynamics and aerodynamic disturbances during training and within the controller. First, we parameterize the residual aerodynamic effects that are not captured by knowledge using a neural ODE $d(x, x_{top}; \theta)$. This neural ODE is constructed as a Υ-branch, L-layer feedforward neural network,

$$h_{0,\upsilon} = \left[z,\ r,\ v_z^{\mathcal{F}}\right]^\top, \quad \upsilon = 1, \dots, \Upsilon, \tag{9a}$$

$$h_{L,\upsilon} = \text{MLP}_\upsilon(h_{0,\upsilon}), \quad \upsilon = 1, \dots, \Upsilon, \tag{9b}$$

$$d(x, x_{top}; \theta) := MH\left[h_{L,1}^\top, \dots, h_{L,\Upsilon}^\top\right]^\top, \tag{9c}$$

where $H \in \mathbb{R}^{13\times(n_L\Upsilon)}$ is a selection matrix and n_i is the number of neurons in the i^{th} layer. The variables in (9a) are defined in Section II-A. The multi-layer perceptron (MLP) for the υ^{th} branch, $\text{MLP}_\upsilon(h)$, is given as

$$\text{MLP}_\upsilon(h) := W_{L\upsilon}\,\phi_{(L-1)\upsilon}\left(\dots \phi_{0\upsilon}(h)\right) + b_{L\upsilon}, \tag{10a}$$

where

$$\phi_{l\upsilon}(h) = \sigma\left(W_{l\upsilon}h + b_{l\upsilon}\right), \quad l = 0, \dots, L-1, \tag{10b}$$

and the weight matrix and bias vector are given as $W_{l\upsilon} \in \mathbb{R}^{n_{l+1}\times n_l}$ and $b_{l\upsilon} \in \mathbb{R}^{n_{l+1}}$. The hyperbolic tangent is used as the activation function σ. Collectively, the parameters of the neural ODE are denoted by $\theta := \{(W_{01}, b_{01}), \dots, (W_{L\Upsilon}, b_{L\Upsilon})\}$.

This architectural choice in (9) imposes structural knowledge about the disturbances experienced by the bottom quadrotor. First, choosing the inputs to be agnostic to any direction in the $x-y$ plane of $\mathcal{F}$ results in a neural ODE that is axially symmetric about the z-axis of $\mathcal{F}$. Second, the MLP outputs in (9b) are multiplied by the matrix H, which selects the state derivatives directly affected by the disturbances. Finally, these outputs are enforced to be in $\mathcal{F}$ by rotating them into the required reference frames, through the multiplication of the matrix $M \in \mathbb{R}^{13\times13}$ given by

$$M := \begin{bmatrix} M_{11} & 0_{10\times3} \\ 0_{3\times10} & \hat{J}^{-1}R^\top R_{\mathcal{W}}^{\mathcal{F}\top} \end{bmatrix}, \tag{11a}$$

where

$$M_{11} = \begin{bmatrix} & 0_{3\times10} & \\ 0_{3\times3} & \frac{R_{\mathcal{W}}^{\mathcal{F}\top}}{m} & 0_{3\times4} \\ & 0_{4\times10} & \end{bmatrix}, \tag{11b}$$

and $\hat{J}^{-1} := J^{-1}/||J^{-1}||_F$ where $||\cdot||_F$ is the Frobenius norm.

Combining the neural ODE with the knowledge described in Section II-A, the overall KNODE-DW model is given as

$$\dot{x}(\theta) = f_{nom}(x, u) + f_d(x, x_{top}) + d\left(x, x_{top}; \theta\right), \tag{12}$$

and we denote a model consisting of only the components $f_{nom}(x, u)$ and $d\left(x, x_{top}; \theta\right)$ as the KNODE model. We highlight that our approach is fundamentally different from those in the existing literature, *e.g.*, [14], [23]. We leverage analytical models and use a data-driven component to capture the residual aerodynamics, while existing works use the data-driven component to capture the full aerodynamics, similar to a KNODE model.

To train the model, we follow a procedure that is similar to those described in [20], [24]. Given a dataset $\mathcal{D} = \{(x_i, x_{top,i}, u_i)\}_{i=1}^M$, we define the loss function to be a weighted mean squared error (MSE) between the one-step state predictions and the true states,

$$\mathcal{L}(\theta) := \frac{1}{M-1}\sum_{i=2}^{M} \left\|\left(\hat{x}_i(\theta) - x_i\right)\right\|_{W_x}^2, \tag{13}$$

where $W_x \in \mathbb{R}^{n\times n}$ is a user-specified weighting matrix. The one-step predictions are given as

$$\hat{x}_i(\theta) := x_{i-1} + \textstyle\int_{t_{i-1}}^{t_i} \dot{x}_{i-1}(\theta)\, dt, \quad i = 2, \dots, M \tag{14}$$

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.

where the integration is done using a given numerical solver. Next, we leverage deep learning tools, *e.g.*, `PyTorch` [25], to find a set of optimal parameters $\theta^\star$. In particular, we compute the gradient of the loss function $\mathcal{L}(\theta)$ through a backpropagation procedure, and incorporate them into an optimization algorithm, *e.g.*, Adam [26], which solves for the parameters $\theta^\star$ in an iterative manner.

One advantage of neural ODEs, as compared to standard networks, is that force and torque measurements are not required. In practice, the acceleration measurements from the inertial measurement unit (IMU) are noisy and may contain non-constant biases. Furthermore, the IMU only provides measurements of the angular velocities. This implies that the moments of inertia have to be sufficiently accurate such that accurate torque estimates can be extracted. On the other hand, velocity and orientation estimates are more easily attainable and are often more accurate.

B. Incorporating into Learning-based MPC

After training, we incorporate the learned KNODE-DW model into the optimization problem by replacing (8c) with an updated set of dynamics constraints. These constraints are constructed by discretizing (12) with the parameters $\theta^\star$ using a suitable numerical method, *e.g.*, 4$^{\text{th}}$ order Runge Kutta. At each time step k, the constraints are given as,

$$x_{i+1} := \hat{f}(x_i, u_i, x_{top,i}(k);\, \theta^\star), \quad i = 0, \dots, N-1. \quad (15)$$

To incorporate the states of the top quadrotor, we formulate (8) as a parametric optimization problem. In this formulation, the current state $x(k)$, the reference states $\{x_{r,i}(k)\}_{i=0}^{N}$ and the states of the top quadrotor $\{x_{top,i}(k)\}_{i=0}^{N-1}$ act as parameters to the problem. To obtain the states of the top quadrotor along the prediction horizon, we assume that it travels at a constant velocity along the prediction horizon, and propagate its position using this velocity.

Using an optimization-based control formulation like MPC provides two advantages. First, given a sequence of position estimates of the top quadrotor, the bottom quadrotor is able to anticipate and account for the aerodynamic disturbances in the next few timesteps. In particular, the problem (8) considers the proximity of the top quadrotor when computing the control actions for the bottom quadrotor. This is in stark contrast with some of the current control architectures for close proximity flight, *e.g.*, [13], [17]. These controllers only consider the disturbances caused by the top quadrotor at the current time step. Second, since it is an optimization-based scheme, it is straightforward to include collision avoidance constraints into (8), given the position of the quadrotors. We leave the integration of these constraints to future work.

The incorporation of the KNODE-DW model into the MPC framework is a natural, yet significant, extension of our prior work [20]. The key architectural difference is the addition of the disturbance model (7), during training and deployment. While our proposed approach is conceptually similar, it is shown in our experiments that the framework in [20] does not necessarily lead to accurate closed-loop performance under all test cases. A detailed discussion is given in Section IV.

IV. EXPERIMENTS AND RESULTS

In our experiments, we set out to answer the following questions about our proposed framework: (i) Does the KNODE-DW model provide accurate predictions of the disturbance forces and torques caused by the top quadrotor? (ii) By incorporating the model into learning-based MPC, are there significant improvements in terms of closed-loop trajectory tracking performance, over meaningful baselines? (iii) Do the model and control framework generalize to cases beyond in which training data is collected? (iv) Do the results observed in simulations extend to physical experiments? (v) Can the quadrotors fly in a tight formation with the proposed framework?

We consider a number of baselines to ascertain the efficacy of our framework. First, we consider a nominal MPC framework, which consists of the nominal dynamics f_{nom}. This allows us to examine the improvements brought forth by the disturbance model f_d and the neural ODE $d(\theta^\star)$. Next, we consider a KNODE-MPC framework, which leverages f_{nom} as knowledge, and uses a neural ODE to account for all unmodeled dynamics, including the aerodynamic disturbances. Third, we consider DW-MPC, which only comprises of f_{nom} and f_d and there are no learning components in it. In the simulations, we consider an additional benchmark, an omniscient MPC framework, where the model is identical to the true system dynamics. Finally, we have our proposed framework, KNODE-DW MPC, which not only consists of both f_{nom} and f_d, but also includes a neural ODE $d(\theta^\star)$.

A. Simulations

Setup: A simulator is constructed using (3) and the disturbance model (7). An explicit 5$^{\text{th}}$ order RK method (RK45) with a time step of 5 ms is used for numerical integration to simulate the responses for the system. The disturbances (6) are implemented using a first-order approximation of the 2D trapezoidal rule. To ascertain the efficacy of the neural ODE, we consider a mismatch between the disturbance model in the simulator and that in the controller. The simulated values of C_D and S are 1.3 and 0.0697, and these values in the controller are 1.18 and 0.0997.

The learning-based MPC framework is assumed to have access to the states of both quadrotors. The framework is implemented using `acados` [27] and `CasADi` [28]. It generates thrust and moment commands, which act as inputs to the simulator which generates the closed-loop responses. In simulations, we consider a two-branch, 1-layer neural ODE with 9 neurons for each branch. The elements of the selection matrix H_{ij} are set to be 1 for $(i,j) \in \{(4,4),(5,5),(6,6),(4,17),(5,18),(6,19)\}$ and 0 elsewhere, and the two branches represent the residual disturbance forces and torques.

For two quadrotors moving in a straight path, we observe that the case of crossing trajectories, *i.e.*, when their 2D trajectories overlap momentarily, is significantly more benign, as compared to the cases of a static top quadrotor or when the quadrotors are in a stacked formation. This is because the time period in which the bottom quadrotor experiences the disturbances is shorter. While the case of crossing trajectories

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.

is more commonly considered in the literature, *e.g.*, [13], [14], we are interested in testing the limits of our framework. That motivates us to consider these two latter cases, which we denote as *static top* and *stacked*, in both simulations and physical experiments.

The training data $\mathcal{D}$ consists of four flight segments. Two of the segments are from the *static top* case, while the other two are from the *stacked* case. The dataset consists of 3,532 data points, which corresponds to approximately 18 seconds of flight. They are obtained from closed-loop simulations with nominal MPC, at a speed of 0.4 m/s and separations of 0.3 m and 0.4 m. We set $W_x := \text{diag}\left(\mathbb{I}_{10\times10},\ 0.1\,\mathbb{I}_{3\times3}\right)$ during training. Apart from training cases, additional tests are conducted at two other speeds, 0.3 m/s and 0.5 m/s, and at two other separations of 0.35 m and 0.2 m. This is to evaluate the generalization abilities of the model and controller.

Results: To verify the accuracy of the models, we compare the predictions of the disturbance forces and torques with the true disturbances. As seen in Fig. 2, the KNODE-DW model provides the most accurate predictions in both the *stacked* and *static top* cases, attaining a 89.5% improvement over the nominal model, on average. While the KNODE model achieves similar improvements in force predictions under the *stacked* formation, the prediction errors are relatively large under the *static top* scenario. Furthermore, the KNODE model does not yield accurate predictions for the torques, performing worse than the nominal model at times. In contrast, the DW model provides more consistent predictions for the torque predictions, but fares worse for the force predictions. Overall, the KNODE-DW model combines the strengths of both the KNODE and DW models. We highlight that in the current literature, disturbance torques are typically not considered within the controllers, *e.g.*, [13], [14], [17]. In contrast, our framework predicts and compensates these torques in a systematic manner, as shown in Figs. 2 and 3.

Next, we compare the closed-loop tracking performance of the proposed KNODE-DW MPC framework against the baselines. We consider two metrics, the root mean squared errors (RMSEs) between the bottom quadrotor and reference trajectories, and the maximum vertical separation z_{max}, which is computed by taking the longest distance between the bottom quadrotor and reference heights. It is observed in Fig. 3 that all four control frameworks perform better than nominal MPC, since all the percentages are less than 100%. Notably, KNODE-DW MPC achieves performance that is comparable to omniscient MPC in all cases, reducing the average RMSEs and z_{max} by 89% and 96.5% over nominal MPC. It is observed that KNODE-DW MPC and DW MPC perform better than KNODE MPC on average, and DW MPC is consistent than KNODE MPC across the test cases. This illustrates the importance of embedding knowledge into training and within the control framework.

B. Physical Experiments

Setup: Two Crazyflie 2.1 quadrotors [16] are deployed in the experiments. Each of the quadrotors has a body length of 10 cm and weighs approximately 34g. The bottom quadrotor is equipped with the thrust upgrade bundle [16] to increase its control authority. A laptop running on Intel i5 CPU acts as the base station and communication between the Crazyflies are established via Crazyradio PA at an average rate of 400 Hz. A Vicon motion capture system is used to obtain pose measurements and communicates with the base station at an approximate rate of 120Hz. The CrazyROS wrapper [29] is used as part of the software architecture.

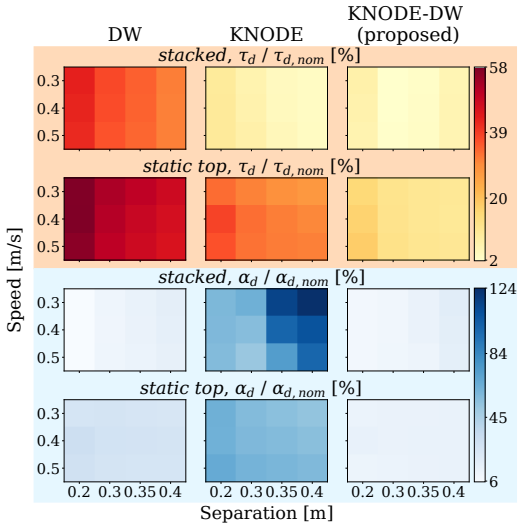


Fig. 2: **Force** (τ_d) **and torque** (α_d) **predictions:** Heatmaps of the prediction root mean squared errors (RMSEs) given by the KNODE, DW and KNODE-DW models, normalized against those given by the nominal model, under different speeds and vertical separations.

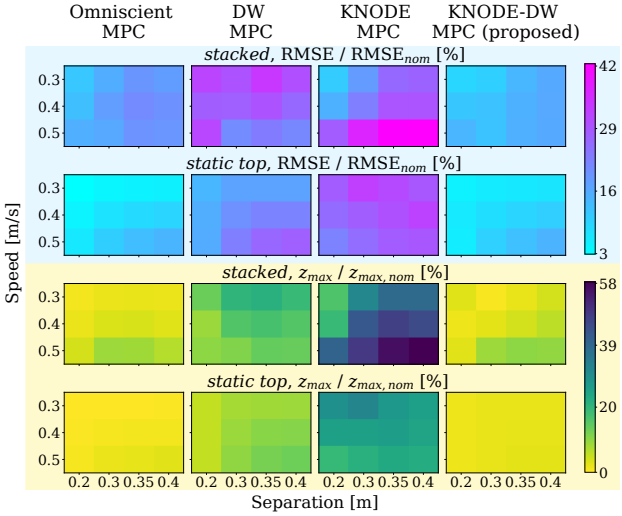


Fig. 3: **Closed-loop performance:** Heatmaps of the tracking RMSEs and maximum vertical separation (z_{max}) of the baseline and proposed MPC frameworks, normalized against those obtained from nominal MPC, under different speeds and vertical separations.

The learning-based MPC framework is implemented using the same software as in simulations and operates on the base station. Instead of generating thrust and moment commands, the framework in physical experiments generates thrust and angular rate commands, which are passed to the PID controllers running in the Crazyflie firmware. With this control architecture, we compensate for the disturbance forces and not the torques, by setting $M := M_{11}$. We define $d(x, x_{top}; \theta)$ as a single branch, single layer neural ODE with

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.

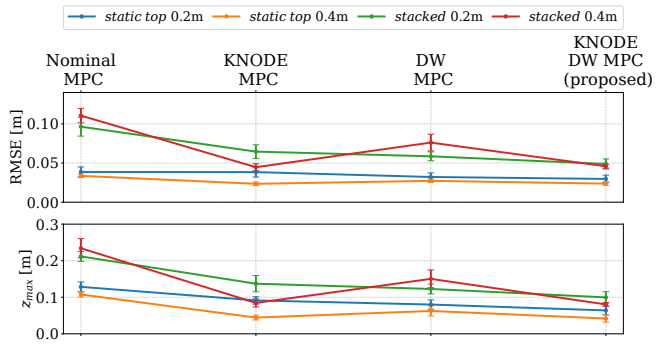


Fig. 4: **Experiment statistics:** Statistics of the runs for the baselines and the proposed framework. The top subplot depicts the RMSEs, and the maximum vertical separation (z_{max}) is shown in the bottom subplot. The markers and the error bars indicate the mean and standard deviation of the runs. The values behind the test cases in the legend, *e.g.*, 0.2m for *stacked* 0.2m, denote the commanded vertical separation between the two quadrotors.

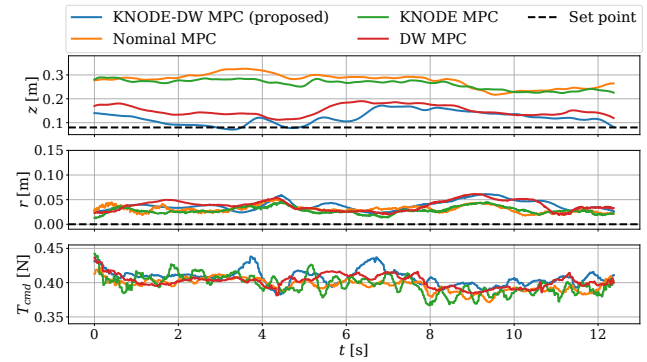


Fig. 5: **Experiment results:** Time histories of the vertical (z) and radial (r) separations, and the commanded thrust (T_{cmd}) of the bottom quadrotor, under the baselines and KNODE-DW MPC. This is during a more challenging test case, where the vertical separation between the two quadrotors is set at 0.8 body lengths throughout the flight.

4 hidden neurons. The elements of the selection matrix H_{ij} are 1 for $(i,j) \in \{(4,4),(5,5),(6,6)\}$ and 0 elsewhere. To obtain an approximate model for the disturbance forces, we conduct a system identification procedure where we adjust the model parameters so that DW MPC attains reasonable performance under training conditions, setting C_D as 0.236.

Similar to the simulations, the training data is collected under the *static top* and *stacked* cases at a speed of 0.4m/s and a separation of 0.4m. The dataset consists of 6,720 data points, corresponding to 28 seconds of flight. This is done in closed-loop by flying the quadrotors under nominal MPC. This is in stark contrast with [14], where the model is trained using 570 seconds of data. To ascertain generalization, another test case at a speed of 0.4m/s and at a smaller separation of 0.2 m is considered.

Results: Fig. 4 depicts the statistics of the runs under different baselines. For each test case, 5 runs are conducted. First, we observe that all 3 baselines, KNODE MPC, DW MPC and KNODE-DW MPC, perform better than nominal MPC, which demonstrate the importance of having an accurate model in the controller. Moreover, it is observed that the KNODE-DW MPC framework not only achieves the lowest RMSEs, but also maintains the smallest z_{max} across all test cases. In particular, our proposed framework achieves an average of 40.1% improvement in RMSE and 57.5% reduction in z_{max} over nominal MPC across the test cases. While KNODE MPC provides comparable performance in cases where training data is collected, it fails to provide consistent performance for the other two test cases, unlike KNODE-DW MPC. The improvement of KNODE-DW MPC over DW MPC indicates that the neural ODE is able to compensate for the mismatch between the knowledge component and the true system dynamics.

To further evaluate the performance of our framework, we first consider a complex flight formation, as shown in the composite photo in Fig. 1. The two CFs are flying in vertical lemniscate trajectories, while maintaining a *stacked* formation. Although the model has not seen this trajectory during training, it performs remarkably well, highlighting the generalizability of our framework. Next, we consider a more challenging *stacked* formation where the two quadrotors fly in a straight line, under a *very* tight formation. They are commanded to be 0.8 body lengths (0.08m) vertically apart, throughout the flight. The time histories of the vertical separation z is shown in Fig. 5. Under KNODE-DW MPC, the average vertical separation between the drones is 0.12m, which is 57.1% smaller than that when nominal MPC is applied, which is at 0.28m. We highlight that this vertical separation of 1.2 body lengths is significantly smaller than what has been reported in the literature, which is of at least 2 body lengths, *e.g.*, [5], [17]. We also note that the average radial separation r in our framework is at 0.03m, which implies that the bottom quadrotor is well in the wake of the top quadrotor. The difficulty of flying in such a tight formation stems from the complex aerodynamic disturbances experienced by the bottom quadrotor, when it is within 2 body lengths of the top quadrotor [11], [12]. We demonstrate with this demanding test case that our framework is able to overcome these complex aerodynamic disturbances, enabling tight formation flight for the first time. Interested readers are referred to the accompanying video for more details on the physical experiments.

V. CONCLUSION

In this work, we present a framework to model and learn the aerodynamic disturbances caused by another quadrotor in an accurate and sample efficient manner. It is demonstrated through simulations and physical experiments that the proposed model and control framework are highly accurate in terms of disturbance predictions, as well as closed-loop trajectory tracking performance. One possible direction of future work is to consider contemporary uncertainty quantification methods such as those in [30], to improve the uncertainty awareness in these tight formations.

VI. ACKNOWLEDGMENTS

The authors thank Anthony Cowley for inspiring the photo in Fig. 1, and Pratik Kunapuli for providing CF components during part of the development.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.

REFERENCES

[1] J. J. Acevedo, B. C. Arrue, I. Maza, and A. Ollero, "Cooperative large area surveillance with a team of aerial mobile robots for long endurance missions," *Journal of Intelligent & Robotic Systems*, vol. 70, pp. 329–345, 2013.

[2] K. A. Ghamry and Y. Zhang, "Cooperative control of multiple uavs for forest fire monitoring and detection," in *2016 12th IEEE/ASME International Conference on Mechatronic and Embedded Systems and Applications (MESA)*. IEEE, 2016, pp. 1–6.

[3] X. Jin and Z. Hu, "Adaptive cooperative load transportation by a team of quadrotors with multiple constraint requirements," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 1, pp. 801–814, 2022.

[4] F. Augugliaro, E. Zarfati, A. Mirjan, and R. D'Andrea, "Knot-tying with flying machines for aerial construction," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 5917–5922.

[5] A. Shankar, H. Woo, and A. Prorok, "Docking multirotors in close proximity using learnt downwash models," in *International Symposium on Experimental Robotics*. Springer, 2023, pp. 427–437.

[6] W. Jasim and D. Gu, "Robust team formation control for quadrotors," *IEEE Transactions on Control Systems Technology*, vol. 26, no. 4, pp. 1516–1523, 2017.

[7] G. Li and G. Loianno, "Nonlinear model predictive control for cooperative transportation and manipulation of cable suspended payloads with multiple quadrotors," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 5034–5041.

[8] J. Gielis, A. Shankar, R. Kortvelesy, and A. Prorok, "Modeling aggregate downwash forces for dense multirotor flight," in *International Symposium on Experimental Robotics*. Springer, 2023, pp. 393–404.

[9] K. P. Jain, T. Fortmuller, J. Byun, S. A. Mäkiharju, and M. W. Mueller, "Modeling of aerodynamic disturbances for proximity flight of multirotors," in *2019 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2019, pp. 1261–1269.

[10] B. Xu, F. Gao, C. Yu, R. Zhang, Y. Wu, and Y. Wang, "Omnidrones: An efficient and flexible platform for reinforcement learning in drone control," *IEEE Robotics and Automation Letters*, 2024.

[11] L. Bauersfeld, K. Muller, D. Ziegler, F. Coletti, and D. Scaramuzza, "Robotics meets fluid dynamics: A characterization of the induced airflow around a quadrotor," *arXiv preprint arXiv:2403.13321*, 2024.

[12] A. Kiran, N. Harutyunyan, N. Ayanian, and K. Breuer, "Downwash dynamics: Impact of quadrotor separation on forces, moments, and velocities for dense formation flight," in *AIAA AVIATION FORUM AND ASCEND 2024*, 2024, p. 4145.

[13] H. Smith, A. Shankar, J. Gielis, J. Blumenkamp, and A. Prorok, "So (2)-equivariant downwash models for close proximity flight," *IEEE Robotics and Automation Letters*, 2023.

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

[15] T. Z. Jiahao, M. A. Hsieh, and E. Forgoston, "Knowledge-based learning of nonlinear dynamics and chaos," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 31, no. 11, p. 111101, 2021.

[16] Bitcraze. Bitcraze store. [Online]. Available: https://store.bitcraze.io/

[17] G. Shi, W. Hönig, Y. Yue, and S.-J. Chung, "Neural-swarm: Decentralized close-proximity multirotor control using learned interactions," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 3241–3247.

[18] K. Y. Chee, T. C. Silva, M. A. Hsieh, and G. J. Pappas, "Enhancing sample efficiency and uncertainty compensation in learning-based model predictive control for aerial robots," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 9435–9441.

[19] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid Syst.* Cambridge University Press, 2017.

[20] K. Y. Chee, T. Z. Jiahao, and M. A. Hsieh, "KNODE-MPC: A Knowledge-Based Data-Driven Predictive Control Framework for Aerial Robots," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2819–2826, 2022.

[21] K. Y. Chee, M. A. Hsieh, and N. Matni, "Learning-enhanced nonlinear MPC using KNODEs and deep ensembles," in *Proceedings of The 5th Annual Learning for Dynamics and Controls*, 2023, pp. 1125–1137.

[22] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," *Advances in Neural Information Processing Systems*, 2018.

[23] G. Shi, W. Hönig, X. Shi, Y. Yue, and S.-J. Chung, "Neural-swarm2: Planning and control of heterogeneous multirotor swarms using learned interactions," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 1063–1079, 2021.

[24] T. Z. Jiahao, K. Y. Chee, and M. A. Hsieh, "Online dynamics learning for predictive control with an application to aerial robots," *Conf. On Robot Learning (CoRL)*, 2022.

[25] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," 2017.

[26] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2015.

[27] R. Verschueren, G. Frison, D. Kouzoupis, J. Frey, N. van Duijkeren, A. Zanelli, B. Novoselnik, T. Albin, R. Quirynen, and M. Diehl, "acados – a modular open-source framework for fast embedded optimal control," *Mathematical Programming Computation*, Oct 2021. [Online]. Available: https://doi.org/10.1007/s12532-021-00208-8

[28] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Math. Program. Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[29] W. Hönig and N. Ayanian, *Flying Multiple UAVs Using ROS*. Springer Int. Publishing, 2017, pp. 83–118.

[30] K. Y. Chee, T. C. Silva, M. A. Hsieh, and G. J. Pappas, "Uncertainty quantification and robustification of model-based controllers using conformal prediction," in *6th Annual Learning for Dynamics & Control Conference*. PMLR, 2024, pp. 528–540.

Authorized licensed use limited to: University of Pennsylvania. Downloaded on October 29,2026 at 13:06:22 UTC from IEEE Xplore. Restrictions apply.