Contextual Neural Moving Horizon Estimation for Robust Quadrotor Control in Varying Conditions

Kasra Torshizi*[1], Chak Lam Shek*[1], Khuzema Habib[1], Guangyao Shi[2], Pratap Tokekar[1], Troi Williams[1]

***Abstract*— Adaptive controllers on quadrotors typically rely on estimation of disturbances to ensure robust trajectory tracking. Estimating disturbances across diverse environmental contexts is challenging due to the inherent variability and uncertainty in the real world. Such estimators require extensive fine-tuning for a specific scenario, which makes them inflexible and brittle to changing conditions. Machine-learning approaches, such as training a neural network to tune the estimator's parameters, are promising. However, collecting data across all possible environmental contexts is impossible. It is also inefficient as the same estimator parameters could work for "nearby" contexts. In this paper, we present a sequential decision making strategy that decides which environmental contexts, using Bayesian Optimization with a Gaussian Process, to collect data from in order to ensure robust performance across a wide range of contexts. Our method, Contextual NeuroMHE, eliminates the need for exhaustive training across all environments while maintaining robust performance under different conditions. By enabling the neural network to adapt its parameters dynamically, our method improves both efficiency and generalization. Experimental results in various real-world settings demonstrate that our approach outperforms the prior work by 20.3% in terms of maximum absolute position error and can capture the variations in the environment with a few carefully chosen contexts.**

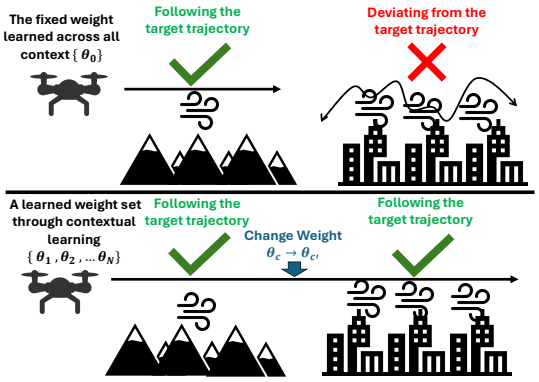


Fig. 1: (top) Without adaptation, a single global model using fixed mapping $\theta_0(\cdot)$ fails to generalize across varying environment contexts c, c', leading to deviation from the target trajectory. (bottom) In contrast, our approach learns a set of context-specific mappings $\{\theta_1(\cdot), \theta_2(\cdot), \ldots, \theta_N(\cdot)\}$ through contextual learning. When the context shifts from c to c', the model dynamically adapts by switching from $\theta_c(\cdot)$ to $\theta_{c'}(\cdot)$, enabling robust trajectory tracking under different environmental conditions.

I. Introduction

Quadrotors have found widespread use in applications such as environmental monitoring [1], [2], precision agriculture [3], [4], and search and rescue operations [5], [6]. However, their deployment in diverse, dynamic environments presents a significant challenge. Quadrotors must adapt to unpredictable disturbances, such as fluctuating wind conditions and environmental interactions [7]–[9]. To ensure reliable trajectory tracking and maintain control performance, it is essential that the controller [10] explicitly accounts for disturbances through accurate disturbance estimation. Developing a universal model to estimate the full spectrum of disturbances encountered in various flight scenarios is impractical due to time and resource constraints [11]. Given the wide range of possible conditions, it is not feasible to train a separate model for each scenario. This underscores the need for adaptive methods capable of estimating disturbances in real-time and adjusting flight control parameters dynamically to handle changing environmental conditions (Figure 1).

Disturbance estimation for quadrotors has been a long-standing area of research due to its critical importance in maintaining stable flight performance. Traditional approaches have largely relied on momentum-based estimators [12]–[14], often implemented with stable low-pass filters. While such filters are effective for static or slowly varying disturbances, they struggle to capture rapidly changing forces. To overcome these limitations, more advanced methods such as Moving Horizon Estimation (MHE) [15], [16] and deep neural networks (DNNs) [17] have been investigated. MHE-based approaches formulate disturbance dynamics within a finite-horizon constrained optimization problem and have been widely adopted for state estimation. Although they demonstrate improved performance, these methods remain highly sensitive to the choice of internal parameters. To mitigate this, NeuroMHE [18] has been introduced, which adaptively tunes estimator parameters based on the observed state. However, NeuroMHE is restricted to operation under a fixed disturbance distribution, limiting its applicability in diverse scenarios. Training multiple specialized models to cover a wide range of conditions is possible but incurs prohibitive data and computational demands.

Alternatively, recent research has increasingly focused on training smaller models tailored to individual tasks, receiving growing attention for their ability to capture context-specific dynamics. This form of contextual learning leverages environmental information to construct models specialized for

* Equal Contribution
[1] University of Maryland, College Park. {ktorsh, cshek1, khabib, tokekar, troiw}@umd.edu.
[2] University of Southern California. shig@usc.edu

narrow operating regimes, where data availability is often limited. For quadrotors, this adaptability is crucial for real-time adjustment to disturbances such as wind. Recent studies have demonstrated the efficacy of contextual learning in robotics and autonomous systems [19]–[22]. Lee et al. [23] introduced a context-aware dynamics model that enhances generalization in model-based reinforcement learning, allowing robots to adapt to new, unseen conditions without retraining the entire system. Similarly, Huang et al. [24] developed a robust adaptation module that generalizes well in both in-distribution and out-of-distribution environments, effectively handling disturbances like wind and payload variations in real-time.

However, many existing methods, including contextual learning methods, for disturbance estimation are tailored to specific scenarios and therefore fail to generalize across diverse environments. Training a single model that can estimate disturbances accurately under all conditions is often inefficient and unreliable—not only because it requires exclusive, environment-specific data, but also because it is fundamentally impossible to collect data covering the full spectrum of possible contexts. Moreover, even with large datasets, it is inevitable that certain contexts will remain underrepresented, leading to gaps that prevent robust generalization. This fundamental limitation underscores the need for approaches that can adaptively leverage informative contexts rather than attempting to exhaustively cover every possible environment.

To address these challenges, we propose an active learning method, Contextual NeuroMHE, for disturbance estimation that adapts to varying flight scenarios under resource constraints. Rather than training a separate model for each situation, we formulate the problem as a sequential decision task, where a Gaussian Process [25] using Bayesian Optimization [26] guides the selection of the most informative contexts that improve the estimator. This formulation leverages the property that as the number of training samples grows, the marginal gain in performance diminishes, making it essential to prioritize the most valuable contexts instead of collecting data randomly. By exploiting this principle, our method optimizes disturbance estimation with a limited budget, training a compact set of models that generalizes across diverse scenarios. This provides a scalable solution for real-time disturbance estimation and robust flight control across multiple environments.

The main contributions of this work are threefold:

1) We formulate context selection for disturbance-aware estimation as a sequential decision problem, enabling adaptive selection of informative training contexts.
2) We propose a novel learning framework that integrates GP–based context selection with NeuroMHE, allowing weight matrices to adapt dynamically to varying environmental conditions.
3) Through extensive real-world experiments on a Crazyflie 2.1 with household fans, covering 13 wind contexts across three environments, we demonstrate a **20.3%** improvement in estimation and control performance over baseline approaches.

II. RELATED WORK

Handling disturbances in quadrotor flight has long motivated the use of MHE, since conventional observers, such as Extended Kalman Filter (EKF) [27] often struggle with constraints, strong nonlinearities, and time-varying dynamics. Early works established MHE [15], [16] as a constrained optimization framework for state estimation, later extended to nonlinear systems with stability guarantees [28], [29]. Comparisons [30] showed MHE's advantages over EKF under process noise and constraints, while algorithmic contributions [31], [32] focused on real-time feasibility through efficient optimization and sensitivity updates. Broader surveys [33] positioned MHE within modern estimation alongside particle filtering, and more recent robotics-oriented work [34] demonstrated its utility for constrained robot localization. Building on these foundations, recent research has advanced MHE with data-driven and learning-based methods: differentiable MHE formulations for robust quadrotor flight control [18], [35], trust-region neural MHE that leverages second-order optimization for efficient tuning [36], and Gaussian process-aided MHE to capture aerodynamic effects in agile quadrotor flight [37]. Collectively, the studies demonstrate that MHE is an effective framework for *quadrotor disturbance estimation*, offering robustness to wind and environmental interactions and benefiting from recent advances in machine learning.

While prior approaches have shown success in specific settings, they often lack generality, requiring extensive case-specific training and manual tuning, while deep learning methods remain resource-intensive. To address these limitations, we propose a contextual method that adapts to diverse flight scenarios under limited data and computational budgets, enabling efficient and scalable disturbance estimation for robust quadrotor control.

III. PRELIMINARIES

MHE is an optimization-based state estimation framework that computes the system state by solving a constrained optimization problem over a finite horizon of past measurements and control inputs. Unlike recursive estimators such as the Kalman filter, which update the state estimate based only on the most recent observation, MHE maintains a sliding window of information. This enables it to explicitly account for nonlinear dynamics, state and input constraints, and non-Gaussian disturbances. These properties make MHE particularly attractive for safety-critical robotic systems such as quadrotors, where robustness to disturbances is essential.

At each time step $t \geq N$, the MHE procedure estimates a sequence of states $\{x_k\}_{k=t-N}^{t}$ and process noise vectors $\{w_k\}_{k=t-N}^{t-1}$ by solving a nonlinear optimization problem of

the form

$$\min_{x,w} J = \frac{1}{2}\|x_{t-N} - \hat{x}_{t-N}\|_P^2 + \frac{1}{2}\sum_{k=t-N}^{t}\|y_k - h(x_k)\|_{R_k}^2 + \frac{1}{2}\sum_{k=t-N}^{t-1}\|w_k\|_{Q_k}^2 \tag{1}$$

subject to the discrete-time system dynamics

$$x_{k+1} = f(x_k, u_k, w_k, \Delta t), \tag{2}$$

where $f(\cdot)$ denotes the dynamics model and Δt is the sampling interval. The objective consists of three components: (i) an arrival cost $||x_{t-N} - \hat{x}_{t-N}||_P^2$ that incorporates the prior estimate $\hat{x}_{t-N}$, (ii) a measurement cost penalizing the discrepancy between predicted and observed outputs, and (iii) a process cost penalizing deviations from the nominal dynamics. Here, $||x||_M^2 = x^\top M x$ denotes the weighted norm induced by a positive-definite matrix M.

MHE performance is highly sensitive to these weights. If an improper set of weights is used, the resulting estimates $\hat{x}$ may be biased or inaccurate, compromising closed-loop performance. To mitigate this issue, NeuroMHE [18] employs neural networks to learn weightings directly from data.

The parameter vector θ collects all tunable elements in the NeuroMHE weighting matrices:

$$\theta = \{\text{vec}(P), \{\text{vec}(R_k)\}_{k=t-N}^{t}, \{\text{vec}(Q_k)\}_{t-N}^{t-1}\} \in \mathbb{R}^p, \tag{3}$$

where P, R_k, and Q_k are symmetric positive-definite matrices weighting the initial-state deviation, measurement residuals, and process noise, respectively.

A neural network outputs the weighting matrices or residual models, and analytic gradients of the MHE solution with respect to these weights are derived, enabling efficient embedding of MHE within a differentiable learning framework. NeuroMHE has shown the capability to automatically tune key MHE parameters across varying flight conditions, improving estimation accuracy without requiring ground-truth disturbance data.

In our formulation, a neural network parameterized by ϖ maps input features to the elements of θ. The network is trained end-to-end by minimizing a downstream control-oriented loss, such as trajectory tracking error. We define the context c as a set of latent or observable variables that characterize the environment, such as wind conditions, sensor noise levels, or payload configurations. Importantly, within the same context c, the noise and disturbance statistics are assumed to follow a consistent Gaussian distribution with fixed parameters. This assumption ensures that models trained under a given context capture stable statistical properties, while variations across different contexts reflect shifts in the underlying distribution. By leveraging this structure, the neural network can exploit context-dependent regularities to produce appropriate weightings $\theta(c)$ that remain valid within each regime.

IV. Problem Formulation

The objective of this paper is to adaptively tune the weighting matrices in MHE to minimize state estimation errors in downstream control tasks. Specifically, we define a scalar loss function $L(\hat{x}(\theta(c))$ where $\hat{x}$ denotes the estimated state produced by MHE, $c \in \mathcal{C} \subseteq \mathbb{R}^d$ represents a continuous context variable that modulates the noise statistics, and θ denotes the weights of the MHE. This loss function is designed to penalize trajectory tracking errors or other task-relevant quantities, enabling end-to-end optimization of the estimation process with respect to context.

A key challenge is that collecting data across the full spectrum of contexts $\mathcal{C}$ is infeasible. In practice, training can only rely on a finite set of sampled contexts, which may not fully capture rare or extreme operating regimes. This limitation motivates methods that can generalize from limited training data by actively and selectively choosing training contexts that maximize overall performance.

To enable such adaptation, we formulate the problem as a sequential decision problem, where the learner must iteratively select the most informative contexts and update the associated weighting functions. We model this as learning a set of M neural network mappings $\boldsymbol{\theta} = \{\theta_{\varpi_{c_1}}(\cdot), \theta_{\varpi_{c_2}}(\cdot), \ldots, \theta_{\varpi_{c_M}}(\cdot)\}$, where each $\theta_{\varpi_{c_i}}(\cdot)$ takes the historical state associated with context c_i as input and dynamically outputs the optimal weighting parameters for the MHE. Here, ϖ denotes the parameters of the neural networks.

Formally, the sequential nature of the problem can be expressed using recursive evaluations of performance. Based on previous losses $L(\theta_{\varpi_{c_1}}, c), \ldots, L(\theta_{\varpi_{c_{k-1}}}, c)$ for all $c \in \mathcal{C}$, we define

$$L(\theta_{\varpi_{c_k}}, c'; \theta_{\varpi_{c_{1:k-1}}}) = \min\left(L(\theta_{\varpi_{c_k}}, c'), L(\theta_{\varpi_{c_{1:k-1}}}, c')\right), \quad \forall c' \in \mathcal{C}; k > 1. \tag{4}$$

The overall sequential decision problem can then be written

$$\min_{c_k \in \mathcal{C}\setminus c_{1:k-1}} V(c_k; \theta_{\varpi_{c_{1:k-1}}}) = \min \mathbb{E}_{c' \sim U(\mathcal{C})}\left[L(\theta_{\varpi_{c_k}}, c'; \theta_{\varpi_{c_{1:k-1}}})\right]. \tag{5}$$

This formulation ensures that the estimator sequentially selects the most informative context to collect data in order to maximize overall performance across diverse environments.

V. Contextual NeuroMHE

A. Overview

Our approach consists of two main components: context selection and model training (Figure 2). In the first stage, we use Bayesian Optimization with Gaussian Process regression to sequentially select the next most informative context from the remaining candidates, sequentially selecting each new context to train a corresponding model. At the end of this process, the trained models collectively form a representative training set that enables the drone to perform effectively across diverse scenarios. In the second stage, we train a NeuroMHE model for each selected context, learning context-dependent parameters that minimize estimation errors and

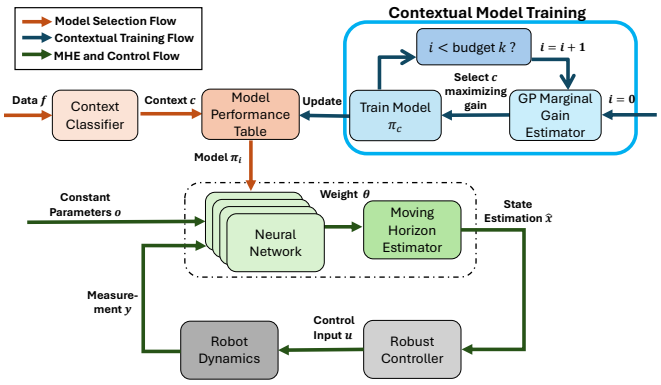


Fig. 2: **System architecture for context-aware control with online model selection and adaptation.** (1) Training context selection via GP-based marginal gain estimation; (2) model selection of context-conditioned models π_c with data $\mathcal{F}$; and (3) deployment using MHE-based state estimation $\hat{x}$ and robust control with context-specific weights θ. The context c is classified online and used to select the model and controller.

improve generalization across the task space. Throughout this process, we maintain a value function V to record and evaluate the performance of the sequential context selection strategy. The details of the algorithm are provided in Alg. 1.

B. Assumption on Generalization Gap

We consider the performance functions $J(\pi, c)$ and $V(c)$ for a given policy π are assumed to vary smoothly over tasks. Our focus is on characterizing how performance degrades when transferring a policy trained on a source task to a different target task. We assume that the generalization gap between a source task c and a target task c' is modeled as a linear function of their distance:

$$\Delta\hat{J}(\pi_c, c') = J(\pi_c, c) - \hat{J}(\pi_c, c') = \alpha \|c - c'\|, \quad (6)$$

where $\alpha > 0$ is a slope parameter, and $\|c - c'\|$ denotes the distance between the task contexts c and c'.

This assumption captures the intuition that the transferability of a policy decreases gradually with task dissimilarity. When two tasks are close in the task space, the generalization gap is small, indicating that the policy trained on the source task performs well on the target. As the distance increases, the gap grows linearly, reflecting a predictable decline in performance.

C. Task Selection

To determine the next context to train the model, we employ a Gaussian Process (GP) to model the performance across the contextual space. At each selection step k, a context c_k is chosen by maximizing the acquisition function [38]:

$$a(c; c_{1:k-1}) = \mathbb{E}_{c' \in \mathcal{X}}\Big[\big[\mu_{k-1}(c) + \beta^{1/2}\sigma_{k-1}(c) - \Delta\hat{J}(\pi_c, c') - \hat{J}(\pi_{1:k-1}, c')\big]_+\Big], \quad (7)$$

where $[\cdot]_+ = \max(\cdot, 0)$, and β is a trade-off parameter between exploitation and exploration [38].

The performance values $\hat{J}(\pi_c, c)$ for all contexts not selected for training, $c \in \mathcal{X} \setminus c_1, \ldots, c_k$ are modeled using Gaussian Process (GP) regression. Let the selected training data up to iteration k be denoted as $D_k = (c_i, \hat{J}(\pi_{c_i}, c_i))_{i=1}^{k}$. Given this data, the predicted GP posterior at a new context c follows a normal distribution $P(\hat{J} \mid D_k) = \mathcal{N}(\mu_k(c), \sigma_k^2(c))$, where the predictive mean and variance are $\mu_k(c) = \mathbb{E}[\hat{J}(\pi_c, c)] + k^\top (K + \sigma^2 I)^{-1} y$ and $\sigma_k^2(c) = k(c, c) - k^\top (K + \sigma^2 I)^{-1} k$, respectively. Here, $k = [k(c, c_1), \ldots, k(c, c_k)]$ is the vector of covariances between c and each observed context, and $K = [k(c_i, c_j)] 1 \le i, j \le k$ is the covariance matrix over the observed inputs. As new observations are collected, the dataset D_k is updated and the posterior distribution is refined, allowing the GP to progressively improve its estimate of the performance function.

D. NeuroMHE

The NeuroMHE framework [18] is designed to learn context-dependent parameters that minimize state estimation errors while remaining computationally efficient. For each selected training context c, NeuroMHE learns the optimal parameter π_{ϖ_c} by minimizing a context-specific loss function. The loss is chosen to directly penalize trajectory tracking errors and is defined as

$$L(\hat{x}) = \sum_{k=t-N}^{t} \|\hat{x}_{k|t} - x_{k,\text{ref}}\|_W, \quad (8)$$

where $W \succ 0$ is a positive-definite weighting matrix, $\hat{x}_{k|t}$ is the estimated state at time k, and $x_{k,\text{ref}}$ denotes the corresponding reference trajectory. This formulation ensures that the estimator prioritizes accurate state tracking across the horizon.

To optimize the parameters, the gradient of the loss with respect to π is computed via the chain rule:

$$\frac{dL}{d\varpi} = \frac{\partial L}{\partial \hat{x}} \cdot \frac{\partial \hat{x}}{\partial \theta} \cdot \frac{\partial \theta}{\partial \varpi}. \quad (9)$$

Here, the key term $\frac{\partial \hat{x}}{\partial \theta}$ is obtained using analytical gradients derived from a Kalman filter approximation [18]. The state estimates $\{\hat{X}_{k|k}^{KF}\}_{k=t-N}^{t}$ are initialized as

$$P_{t-N} = P^{-1}, C_{t-N} = (I - P_{t-N} S_{t-N})^{-1} P_{t-N},$$
$$\hat{X}_{t-N|t-N}^{KF} = (I + C_{t-N} S_{t-N}) \hat{X}_{t-N} + C_{t-N} T_{t-N}. \quad (10)$$

The recursive estimation then proceeds for $k = t - N + 1, \ldots, t$. At each step, the predicted state is updated by

$$\hat{X}_{k|k-1} = \bar{F}_{k-1} \hat{X}_{k-1|k-1}^{KF} - G_{k-1} (L_{k-1}^{ww})^{-1} L_{k-1}^{w\theta}, \quad (11)$$

while the covariance and correction matrices are updated as

$$P_k = \bar{F}_{k-1} C_{k-1} \bar{F}_{k-1}^\top + G_{k-1} (L_{k-1}^{ww})^{-1} G_{k-1}^\top, \quad (12)$$

$$C_k = (I - P_k S_k)^{-1} P_k, \quad (13)$$

and the filtered state is obtained by

$$\hat{X}_{k|k}^{KF} = (I + C_k S_k) \hat{X}_{k|k-1} + C_k T_k. \quad (14)$$

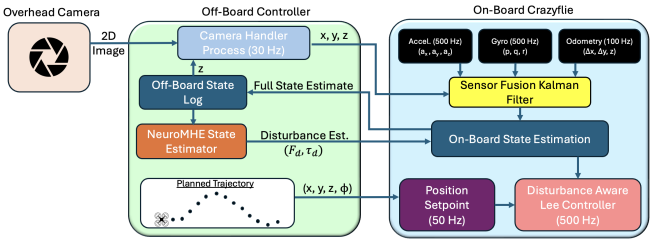


Fig. 3: The system diagram. z-position is computed using the FlowV2 odometry sensor.

Finally, the dual variables are updated backward in time from $k = t$ to $t - N + 1$, with $\Lambda_t^* = 0$, and the collection $\Lambda^* = \{\Lambda_{t-1}^*, \ldots, \Lambda_{t-N+1}^*\}$,

$$\Lambda_{k-1}^* = (I + S_k C_k)\, \bar{F}_k^\top \Lambda_{k+1}^* + S_k \hat{X}_{k|k}^{KF} + T_k. \tag{15}$$

The optimal state estimates are then computed as

$$\hat{X}_{k|t} = \hat{X}_{k|k}^{KF} + C_k \bar{F}_k^\top \Lambda_k^*. \tag{16}$$

$$\frac{\partial \hat{x}}{\partial \theta} = \{\hat{X}_{k|t}\}_{k=t-N}^{t}. \tag{17}$$

This training procedure allows NeuroMHE to leverage analytical gradients for efficient optimization, ensuring stable and accurate state estimation across varying contexts.

E. Weight Usage

This section formalizes the role of the performance function in training and model selection. During training, after each model has been optimized on its assigned context, the performance function V is updated to quantify the generalization capability of the resulting parameters. This update provides an evolving estimate of performance across the contextual space. At test time, no further training is performed; instead, model selection is governed by the maximization of the performance function: $\varpi^\star \in \arg\max_\varpi V(c; \varpi)$, which yields the parameterization that achieves the highest expected performance for a given context c. This corresponds directly to the Model Performance Table illustrated in Figure 2, where V records the performance of each candidate model across contexts and guides the selection of the best-performing network. In this way, the deployed estimator is chosen systematically according to demonstrated generalization performance.

VI. Real-World Evaluation

We evaluated our method against multiple baselines in real-world experiments using a Crazyflie2.1 and multiple common, household fans. We only performed real-world experiments because airflow simulators often fail to capture near-field turbulence, ground effect, and prop–airframe interactions [39] [40]. The following subsections detail our setup.

Algorithm 1 Contextual Learning for Task Training with Model-based Policy Gradient and MHE

Input: Task (context) set X, Training budget K
Initialization: $J, V = 0 \,\forall x \in X$, $\boldsymbol{\varpi} = \emptyset$, $k = 1$

while $k \leq K$ **do**
  % Estimate training performance
  $\mu, \sigma \leftarrow \mathrm{GP}(E[J(\theta(\varpi), c)], k(c, \tilde{c}))$
  % Calculate marginal generalized performance and acquisition function
  Calculate $a(c; x_{1:c-1})$ with Eq. 7
  % Select the next training task
  $c_k = \arg\max_c a(c; x_{1:c-1})$
  % Train using Policy Gradient with MHE
  while L_{mean} does not converged **do**
    for $t = 0$ to T_{episode} **do**
      Forward Pass:
      Compute adaptive weights θ^t using the neural network parameters ϖ_k^t.
      Estimate $\hat{x}$ by solving MHE in Eq. (4).
      Compute control input u^t.
      Apply u^t to update the quadrotor state x^t.
      Backward Pass:
      Compute $\frac{\partial \hat{x}}{\partial \theta}$ using the Kalman filter approximation Eqs. (10–16).
      Compute $\frac{\partial L}{\partial \varpi}$ using Eq. (9).
      Update network parameters ϖ_k^t.
    end for
    Calculate L_{mean} for the next episode
  end while
  $\boldsymbol{\varpi} \leftarrow \boldsymbol{\varpi} \cup \{\varpi_k^t\}$
  $k \leftarrow k + 1$
end while
% calculate generalization performance $V(\varpi_1, \ldots, \varpi_K)$
Output: Set of policies $\boldsymbol{\varpi}$ and performance V

A. Training Data

We collected 13 contexts of varying wind speeds and directions. One context had *No Wind*, while the remaining 12 contexts were combinations of six wind directions ({*Left Crosswind, Right Crosswind, Headwind, Tailwind, Updraft, Downdraft*}) and two wind speeds ({*Low, High*}). For each context, we collected five samples of the drone traveling in a straight line through the wind flow.

B. Disturbance-Aware Lee Controller

Since NeuroMHE outputs estimates of F_{dist} and τ_{dist}, we enhanced the Lee Controller [10] to take into account these two additional values when calculating desired thrust f and the moment vector M:

$$f = (k_x \mathbf{e}_x + k_v \mathbf{e}_v + mg\,\mathbf{e}_3 - m\ddot{\mathbf{x}}_d - F_{dist}) \cdot R\mathbf{e}_3 \tag{18}$$

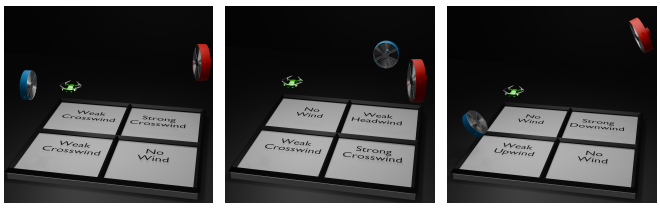
$$\begin{aligned} M = &-k_R e_R - k_\Omega e_\Omega + \Omega \times (J\,\Omega) \\ &-J\left(\hat{\Omega}\, R^\top R_c\, \Omega_c - R^\top R_c\, \dot{\Omega}_c\right) - \tau_{dist}. \end{aligned} \tag{19}$$

Here, m is the drone's mass, $J \in \mathbb{R}^{3x3}$ is the inertia matrix with respect to the body-fixed frame, $R \in SO(3)$ is the rotation matrix with respect to the body frame, Ω is the angular velocity respect to the body-frame, $\ddot{x}_d$ is the linear acceleration with respect to the world frame, the e terms are error terms, and k terms are tuned positive constants.

Then, we obtain the thrusts for each individual motor, f_1, f_2, f_3, f_4 through:

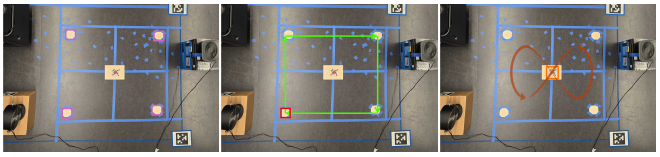
$$\begin{bmatrix} f_1 \\ f_2 \\ f_3 \\ f_4 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 0 & -d & 0 & d \\ d & 0 & -d & 0 \\ -c_\tau & c_\tau & -c_\tau & c_\tau \end{bmatrix}^{-1} \begin{bmatrix} f \\ M_1 \\ M_2 \\ M_3 \end{bmatrix} \tag{20}$$

where d is the distance from each of the motors to the center of the drone, and c_τ is a fixed constant.



(a) Environment One (b) Environment Two (c) Environment Three

Fig. 4: 3D illustration of our three test environments. Each environment has a $1.5m$ x $1.5m$ x $1m$ designated flight space. The red fan represents the *strong* fan, while the blue fan represents the *weak* fan. The specific context is written on the ground of the four quadrants.



(a) Hover Trajectories (b) Square Trajectory (c) Figure-8 Trajectory

Fig. 5: Bird's eye view of our experimental trajectories overlaid atop the actual physical setup. We placed two fans (the grey fan is the *strong* fan and the black one is the *weak* fan) on top of $0.5m$ high boxes for the first two environments. In the third environment, the *weak* fan is placed on the ground and the *strong* fan is placed on top of a taller box, each angled at $45°$.

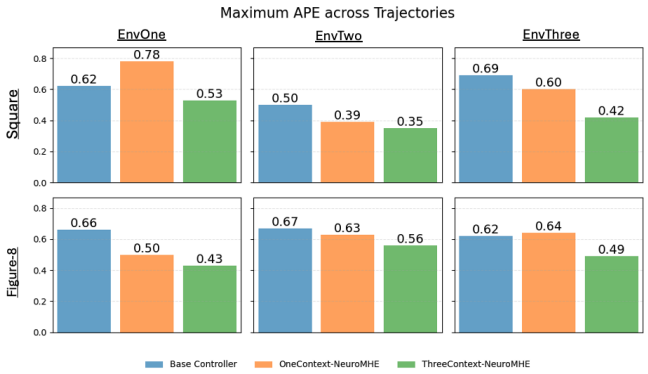


Fig. 6: The maximum APE, for the square and figure-8 trajectories, which is defined as the maximum offset of the recorded state from the desired setpoint across all steps.

C. System Parameters

The neural networks are implemented as fully connected feedforward models with two hidden layers of 30 units each and ReLU activations. Each network takes an input of size six and produces a 25-dimensional output corresponding to the flattened parameterization of the MHE weighting matrices. The notion of context $c \in \mathbb{R}^2$ (wind direction and wind speed provided to GP) is used to define training scenarios but is not directly provided as input to the network. Training is performed using the Adam optimizer with a learning rate of 10^{-4}, and each model is trained until the loss difference between successive epochs falls below 10^{-3}. For GP-based context selection, we employ an RBF kernel with length scale 1.0 and set the trade-off parameter to $\beta = 1.0$ with slope parameter 10^{-3} to balance exploration and exploitation.

D. Physical Setup

Our physical setup featured an off-board computer, a Crazyflie drone, and an overhead camera (Figure 3). Due to limited computation on the Crazyflie, we ran the GP and NeuroMHE models off-board on a laptop equipped with a 12-core Intel i9-8950HK CPU and RTX 2080 Mobile. The drone computed its state using four sensors: an accelerometer, a gyroscope, an odometry sensor, and an overhead camera. Path following was accomplished through updating the position setpoint (x, y, z, ϕ) at 50Hz.

E. Testing Trajectories

We defined three different trajectories for experimentation.

Hover: Starting in any of the four corners (magenta squares in Figure 5a), the drone rose $0.5m$, held that position for five seconds, then landed back in the starting position.

Square: Starting in the bottom left corner (red square in Figure 5b), the drone rose $0.5m$, then visited each of the four corners clockwise. The drone traveled at $0.3m/s$ throughout the trajectory.

Figure-8: Starting from the center of the environment (orange square in Figure 5c), the drone rose $0.5m$ and did a figure-8 trajectory on the xy plane at $0.3m/s$.

F. Evaluated Controllers

Base Controller: A standard Extended Kalman Filter for state estimation and a standard Lee Controller (Disturbance-Unaware) for low-level control.

OneContext-NeuroMHE: A single NeuroMHE model used for all contexts coupled with a Disturbance-Aware Lee Controller for low-level control. We evaluated the performance of all 13 models (one model for each context) and picked the "No-Wind" model since it had the highest general

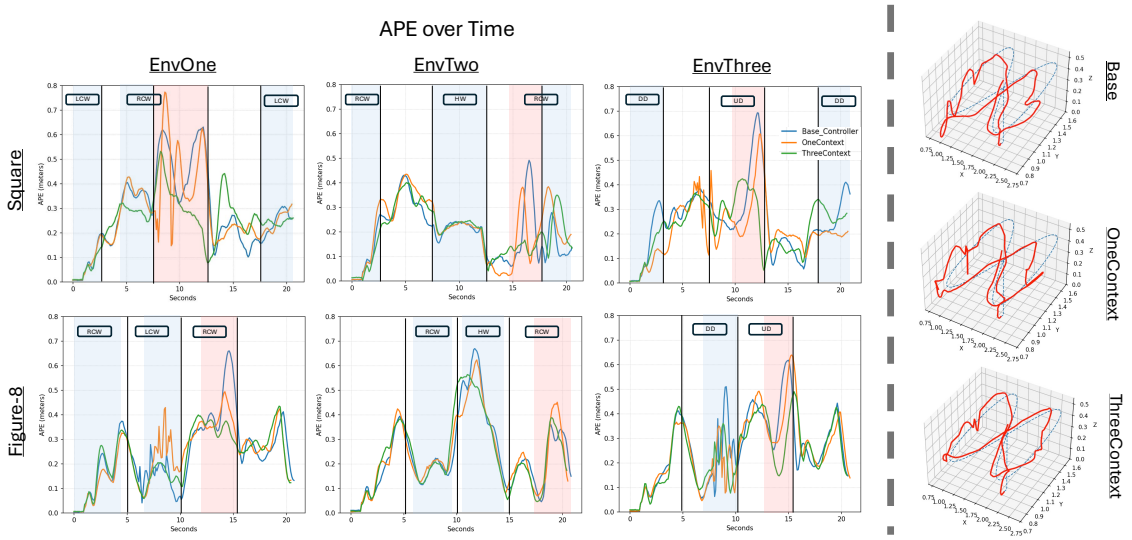


Fig. 7: (Left) Absolute Position Error (APE) for the square and figure-8 trajectories in three environments. The blue shaded regions indicate a low-wind flow, the red-shaded regions indicate a high-wind flow, and the black vertical lines indicate a context switch. The wind flows may not cover the entire context, causing the occasional discrepancy between the context switch and the indicated wind flow. (Right) 3D Plots of the Figure-8 Trajectory flown in EnvThree (solid red is the actual path and dotted blue is the reference). LCW/RCW - Left/Right Crosswind, HW - Headwind, UD/DD - Up/Down Draft.

TABLE I: RMSE Absolute Position Error (APE) across environments and trajectories. The different hover trajectories are aggregated together. With just three contexts, we perform comparably with the oracle and much better than the baseline.

	Environment One			Environment Two			Environment Three		
Method	Hover	Square	Figure-8	Hover	Square	Figure-8	Hover	Square	Figure-8
Base Controller	0.4524	0.2743	0.4119	0.1612	0.2256	0.3560	**0.1249**	0.2887	0.3446
OneContext-NeuroMHE	0.1679	0.3174	0.3966	0.1650	0.2435	**0.3363**	0.1483	**0.2313**	0.3553
ThreeContext-NeuroMHE	**0.1379**	**0.2693**	**0.3896**	**0.1528**	**0.2218**	0.3397	0.1479	0.2561	**0.3305**
FullContext-NeuroMHE	**0.1240**	**0.2622**	**0.3641**	**0.1344**	**0.2036**	**0.3344**	**0.1205**	0.2467	**0.3207**

performance. This model is essentially equivalent to the original NeuroMHE [18].

ThreeContext-NeuroMHE: Our Contextual NeuroMHE trained with a budget of three.

FullContext-NeuroMHE: We trained a NeuroMHE model for each of the 13 contexts and switched to the matching model whenever that context was active, yielding a full oracle-style controller that served as our lower bound.

G. Results

From our experiments, the ThreeContext-NeuroMHE had a 14.9% and a 4.9% reduction in absolute tracking error on average over the Base Controller and OneContext-NeuroMHE, respectively. (See Table I) The major strength of context switching occurs when traveling through abrupt changes in wind disturbances.

Since the drone deviates most during these disturbance shifts, we examine the maximum APE of the square and figure-8 trajectories. Referring to the six evaluated cases in Figure 6, the Base Controller, OneContext-NeuroMHE, and ThreeContext-NeuroMHE had average Maximum APE scores of 0.62, 0.59, and 0.47, respectively. Therefore, the ThreeContext-NeuroMHE achieves a 24.2% and 20.3% decrease in Maximum APE over the Base Controller and OneContext-NeuroMHE, respectively. This decrease in tracking error can be attributed to the context-specific models having a much stronger prior in those specific wind-conditions, needing less time for the MHE to provide accurate state and disturbance estimations, further showcasing the benefit of enabling context switching.

However, due to the increased communication overhead necessary for Contextual NeuroMHE, there is occasional packet loss, causing occasional drifts around tight turns. A more capable drone would be able to run all processes on board, which would circumvent this issue.

VII. Discussion and Conclusion

Contextual NeuroMHE adapts to abrupt wind disturbances much more effectively than NeuroMHE and our EKF baseline. We highlight that even with a highly constrained Gaussian Process training budget, our method achieves a 20.3% decrease in absolute tracking error, which can be attributed to context-specific models having stronger priors in these new settings, requiring fewer control steps to converge.

While our proposed approach demonstrates promising results in real-world experiments, there are several directions for improvement. First, the current framework primarily relies on offline learning, which makes it susceptible to out-of-distribution disturbances not captured during training. Extending the method toward online or continual learning would enable the quadrotor to adapt more effectively to novel and rapidly changing conditions. Second, the backward pass computation, which leverages Kalman filter approximations for estimating derivatives, remains computationally intensive. Future research could explore simplified or approximate formulations that reduce computational overhead while preserving accuracy.

REFERENCES

[1] B. Charrow, G. Kahn, S. Patil, S. Liu, K. Goldberg, P. Abbeel, N. Michael, and V. Kumar, "Information-theoretic planning with trajectory optimization for dense 3d mapping." in *Robotics: Science and Systems*, vol. 11. Rome, 2015, pp. 3–12.

[2] H. Zhu, J. J. Chung, N. R. Lawrance, R. Siegwart, and J. Alonso-Mora, "Online informative path planning for active information gathering of a 3d surface," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1488–1494.

[3] D. Albani, D. Nardi, and V. Trianni, "Field coverage and weed mapping by uav swarms," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Ieee, 2017, pp. 4319–4325.

[4] M. Popović, G. Hitz, J. Nieto, I. Sa, R. Siegwart, and E. Galceran, "Online informative path planning for active classification using uavs," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5753–5758.

[5] G. Best, R. Garg, J. Keller, G. A. Hollinger, and S. Scherer, "Resilient multi-sensor exploration of multifarious environments with a team of aerial robots," in *Robotics: Science and Systems (RSS)*, 2022.

[6] X. Xiao, J. Dufek, T. Woodbury, and R. Murphy, "Uav assisted usv visual navigation for marine mass casualty incident response," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 6105–6110.

[7] L. N. C. Sikkel, G. C. de Croon, C. de Wagter, and Q. Chu, "A novel online model-based wind estimation approach for quadrotor micro air vehicles using low cost mems imus," *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2141–2146, 2016. [Online]. Available: https://api.semanticscholar.org/CorpusID:4963597

[8] G. Cioffi, L. Bauersfeld, and D. Scaramuzza, "Hdvio: Improving localization and disturbance estimation with hybrid dynamics vio," 2023. [Online]. Available: https://arxiv.org/abs/2306.11429

[9] K. Huang, R. Rana, A. Spitzer, G. Shi, and B. Boots, "Datt: Deep adaptive trajectory tracking for quadrotor control," in *Conference on Robot Learning*, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:264127921

[10] T. Lee, M. Leok, and N. H. McClamroch, "Control of complex maneuvers for a quadrotor uav using geometric methods on se(3)," 2011. [Online]. Available: https://arxiv.org/abs/1003.2005

[11] D. Hanover, P. Foehn, S. Sun, E. Kaufmann, and D. Scaramuzza, "Performance, precision, and payloads: Adaptive nonlinear mpc for quadrotors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 690–697, 2021.

[12] T. Tomić and S. Haddadin, "A unified framework for external wrench estimation, interaction control and collision reflexes for flying robots," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 4197–4204.

[13] B. Yüksel, C. Secchi, H. H. Bülthoff, and A. Franchi, "A nonlinear force observer for quadrotors and application to physical interactive tasks," in *2014 IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 2014, pp. 433–440.

[14] F. Ruggiero, J. Cacace, H. Sadeghian, and V. Lippiello, "Impedance control of vtol uavs with a momentum-based external generalized forces estimator," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 2014, pp. 2093–2099.

[15] F. Girrbach, J. D. Hol, R. Zandbergen, R. Verschueren, G. Bellusci, and M. Diehl, "On the effect of stabilization methods for quaternion invariants on the uncertainty in optimization-based estimation∗∗this work is supported by the project awesco (h2020-itn-642682) funded by the european union's horizon 2020 research and innovation program under the marie sklodowska-curie grant agreement no. 642682. jeroen d. hol contributed to this work during his former employment at xsens technologies b.v.." *IFAC-PapersOnLine*, vol. 51, no. 25, pp. 116–121, 2018, 9th IFAC Symposium on Robust Control Design ROCOND 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896318327526

[16] D. G. Robertson, J. H. Lee, and J. B. Rawlings, "A moving horizon-based approach for least-squares estimation," *AIChE Journal*, vol. 42, no. 8, pp. 2209–2224, 1996.

[17] P. Simplício, J. Benevides, R. Inoue, and M. Terra, "Robust and intelligent control of quadrotors subject to wind gusts," *IET Control Theory & Applications*, vol. 18, pp. 2594–2611, 01 2024.

[18] B. Wang, Z. Ma, S. Lai, and L. Zhao, "Neural moving horizon estimation for robust flight control," *IEEE Transactions on Robotics*, vol. 40, pp. 639–659, 2023.

[19] Z. Hu, T. Xu, X. Xiao, and X. Wang, "Carol: Context-aware adaptation for robot learning," *arXiv preprint*, 2025. [Online]. Available: https://arxiv.org/abs/2506.07006

[20] B. Evans, A. Thankaraj, and L. Pinto, "Context is everything: Implicit identification for dynamics adaptation," *arXiv preprint*, 2022. [Online]. Available: https://arxiv.org/abs/2203.05549

[21] A. Nagabandi and et al., "Learning to adapt in dynamic, real-world environments through meta-reinforcement learning," *arXiv preprint*, 2018. [Online]. Available: https://arxiv.org/abs/1803.11347

[22] C. L. Shek, X. Wu, W. A. Suttle, C. Busart, E. Zaroukian, D. Manocha, P. Tokekar, and A. S. Bedi, "Lancar: Leveraging language for context-aware robot locomotion in unstructured environments," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2024, pp. 9612–9619.

[23] K. Lee, Y. Seo, S. Lee, J. Shin, and H. Lee, "Context-aware dynamics model for generalization in model-based reinforcement learning," *arXiv preprint*, 2020. [Online]. Available: https://arxiv.org/abs/2005.06800

[24] Y. Huang and et al., "Generalization in deep rl with a robust adaptation module," *arXiv preprint*, 2025. [Online]. Available: https://arxiv.org/abs/2412.04323v2

[25] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.

[26] J. Mockus, V. Tiesis, and A. Zilinskas, "Bayesian methods for seeking the extremum," in *Towards Global Optimization*, L. C. W. Dixon and G. P. Szegö, Eds., vol. 2. North-Holland, 1978, pp. 117–129.

[27] S. J. Julier and J. K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[28] C. V. Rao, J. B. Rawlings, and J. H. Lee, "Constrained linear state estimation—a moving horizon approach," *Automatica*, vol. 37, no. 10, pp. 1619–1628, 2001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109801001157

[29] A. Alessandri, M. Baglietto, and G. Battistelli, "Moving-horizon state estimation for nonlinear discrete-time systems: New stability results and approximation schemes," *Automatica*, vol. 44, no. 7, pp. 1753–1765, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109807004888

[30] E. L. Haseltine and J. B. Rawlings, "Critical evaluation of extended kalman filtering and moving-horizon estimation," *Industrial & Engineering Chemistry Research*, vol. 44, no. 8, pp. 2451–2460, 2005. [Online]. Available: https://doi.org/10.1021/ie0343081

[31] M. J. Tenny and J. B. Rawlings, "Efficient moving horizon estimation and nonlinear model predictive control," *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, vol. 6, pp. 4475–4480 vol.6, 2002. [Online]. Available: https://api.semanticscholar.org/CorpusID:62499110

[32] V. M. Zavala, C. D. Laird, and L. T. Biegler, "A fast moving horizon estimation algorithm based on nonlinear programming sensitivity," *Journal of Process Control*, vol. 18, no. 9, pp. 876–884, 2008.

[33] J. B. Rawlings and B. R. Bakshi, "Particle filtering and moving horizon estimation," *Computers & Chemical Engineering*, vol. 30, no. 10, pp. 1529–1541, 2006, papers form Chemical Process Control VII. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0098135406001566

[34] G. Pillonetto, A. Aravkin, and S. Carpin, "The unconstrained and inequality constrained moving horizon approach to robot localization," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 3830–3835.

[35] B. Wang, Z. Ma, S. Lai, L. Zhao, and T. H. Lee, "Differentiable moving horizon estimation for robust flight control," in *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, 2021, pp. 3563–3568.

[36] B. Wang, X. Chen, and L. Zhao, "Trust-region neural moving horizon estimation for robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 2059–2065.

[37] W. Choo and E. Kayacan, "Data-based mhe for agile quadrotor flight," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2023, pp. 4307–4314.

[38] J.-H. Cho, V. Jayawardana, S. Li, and C. Wu, "Model-based transfer learning for contextual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 37, pp. 88 279–88 319, 2024.

[39] N. L. Oo, D. Zhao, M. Sellier, and X. Liu, "Experimental investigation on turbulence effects on unsteady aerodynamics performances of two horizontally placed small-size uav rotors," *Aerospace Science and Technology*, vol. 141, p. 108535, 2023.

[40] A. Coursey, M. Quiñones-Grueiro, and G. Biswas, "Quantifying the sim-to-real gap in uav disturbance rejection," in *OpenAccess Series in Informatics (OASIcs), DX 2024*, vol. 125, 2024, pp. 16:1–16:14.