Deploying Neural-Fly in the Field

Michael O’Connell∗,1, Guanya Shi∗,2, Xichen Shi1, Kamyar Azizzadenesheli3, Animashree Anandkumar4,1, Yisong Yue1,4, and Soon-Jo Chung1

Abstract—Neural-Fly is a learning-based control method that allows rapid online learning by incorporating a pre-trained representations of unmodeled aerodynamics with adaptive control. Not is Neural-Fly robust to different tasks, environments, and drones, but also it builds on a standard control architecture, creating a straightforward way to begin integrating learning-based control into safety critical applications. In this workshop, we will present the latest results for Neural-Fly on field robots, discuss implementation considerations, and show case other applications of Neural-Fly, such as fault tolerant control.

I. INTRODUCTION

The proliferation of uninhabited aerial vehicles (UAVs) offers the prospect to revolutionize many areas. Applications range from drone delivery to drone rescue and search, and from urban air mobility to autonomous farming tools. These applications demand precise and agile control methods that can handle the complex aerodynamics while adapting to changing environmental and operating conditions. This creates a need for a control method that can rapidly adapt to new tasks, environments, and drones.

Our recent work, Neural-Fly [1], offers a solution, by pretraining a neural network to enable rapid and robust online learning of wind effects. This enables Neural-Fly to transfer between different tasks, environments, and drones while maintaining high performance. Neural-Fly has been applied to deep-learning-based trajectory tracking control, and it has allowed quick adaptation to rapidly-changing wind conditions with centimeter-level position-error tracking of agile unmanned vehicles.

Neural-Fly outperforms state-of-the-art control methods when implemented in a readily-available UAV control architecture, PX4 [2]. Furthermore, Neural-Fly has demonstrated the ability to transfer control policies from one robot to another, and from limited range of constant wind speeds to a wide range of time-varying wind speeds. At this workshop, we will present the latest results for Neural-Fly deployment on field robots, some important implementation considerations, and show case other applications of Neural-Fly, such as fault tolerant control.

∗ Equal contribution to this work.
1 California Institute of Technology.
2 Robotics Institute, Carnegie Mellon University.
3 NVIDIA Corporation.
4 Latitude AI.

Fig. 1: Neural-Fly design. Neural-Fly learns a model of aerodynamics with linearly separated wind-variant and wind-invariant components. Since only part of the model must be updated in real time, Neural-Fly can quickly learn and adapt to new wind conditions.

II. BRIEF OVERVIEW OF NEURAL-FLY

Our method has two main stages: an offline learning phase and an online adaptive control phase used as real-time online learning. For the offline learning phase, we have developed Domain Adversarially Invariant Meta-Learning (DAIML) that learns a wind-condition-independent deep neural network (DNN) representation of the aerodynamics in a data-efficient manner. The output of the DNN is treated as a set of basis functions that represent the aerodynamic effects. This representation is adapted to different wind conditions by updating a set of linear coefficients that mix the output of the DNN. DAIML is data efficient and uses only 12 total minutes of flight data in just 6 different wind conditions to train the DNN. DAIML uses spectral normalization [3], [4] to control the Lipschitz property of the DNN to improve generalization to unseen data and provide closed-loop stability and robustness guarantees. As seen in Fig. 3, training data generated in different wind conditions can have high correlation between the actual trajectory of the vehicle and the wind condition present. DAIML uses a discriminative network to ensure that the learned representation is wind-invariant and to prevent overfitting to the highly correlated training data. The result is that DAIML trains a concise representation of the aerodynamics that is both data efficient and generalizes well to new wind conditions and even new vehicles.
Fig. 2: Agile flight through narrow gates. Neural-Fly was tested in the Caltech Real Weather Wind Tunnel. These panels show the moment the UAV passed through a narrow gate, only slightly wider than the UAV itself.

For the online adaptive control phase, Neural-Fly uses a robust and fast adaptive control law to update the model for new wind conditions. The adaptation algorithm is built from a Kalman Filter [5, 6] estimator of the linear coefficients, $a(w)$, which provides robustness and regularization properties. The Kalman Filter is augmented with a tracking error term to make the closed loop dynamics stable during rapid adaptation. The combination of the prediction error based Kalman filter and tracking error based adaptation term makes this approach a composite adaptive control law, and effectively guarantees fast and stable adaptation to any wind condition and robustness against imperfect learning. The speed of adaptation is further aided by the concise representation learned from DAIML.

### III. Experimental Validation

Neural-Fly was tested on an agile figure-8 trajectory and compared with several methods that represent the state of art in quadrotor control. Each method was tested in a variety of wind conditions, including wind speeds inside the range of wind speeds seen in training (0 m/s to 4.2 m/s), and wind speeds outside the range of wind speeds seen in training (8.5 m/s to 12.1 m/s), and time varying wind speeds that break the constant wind-speed assumption made during training (8.5 + 2.4 sin(τ) m/s).

When measuring position tracking errors, we observe that our Neural-Fly method outperforms state-of-the-art flight controllers in all wind conditions. Using Neural-Fly, we report an average improvement of 66% over a nonlinear tracking controller [7], 42% over an L1 adaptive controller [8], [9], and 35% over an Incremental Nonlinear Dynamics Inversion (INDI) controller [10].

Neural-Fly can generalize to new conditions, as demonstrated by its performance in wind speeds outside the training range and in time varying wind speeds. Neural-Fly-Transfer was trained on data collected using a different quadrotor without motor tilt, different size, and a different propeller configuration, but Neural-Fly-Transfer maintains nearly as good of performance as Neural-Fly. Thus, Neural-Fly-Transfer demonstrates that Neural-Fly is robust to changes in vehicle configuration and modeling errors.

### IV. Conclusion

Neural-Fly is formulated generally for all robotic systems described by the Euler-Lagrange equation, and should be applicable to a wide range of robotic systems. Neural-Fly demonstrates a new paradigm for designing adaptable controllers that can be trained once and then used to control a wide range of vehicles.

### References


