FlowDrone: Wind Estimation and Gust Rejection on UAVs Using Fast-Response Hot-Wire Flow Sensors

Anonymous Author(s)
Affiliation
Address
email

Abstract: Traditional multirotor UAV platforms do not *directly* sense wind; instead, drones typically observe the effects of wind indirectly through accumulated errors in position or trajectory tracking. In this work, we integrate a novel flow sensor based on micro-electro-mechanical systems (MEMS) hot-wire technology [1] onto a multirotor UAV for wind estimation. In order to achieve superior tracking performance in windy conditions, we train a 'wind-aware' residual-based controller via reinforcement learning using simulated wind gusts and their aerodynamic effects on the drone. In extensive hardware experiments, we demonstrate the wind-aware controller outperforming two strong 'wind-unaware' baseline controllers in challenging windy conditions.

1 Introduction

Autonomous multirotor drones have the potential for transformative impact in domains such as infrastructure inspection and repair, search-and-rescue, and aerial package delivery. Current systems face a major challenge: severe wind conditions in outdoor environments. For example, a typical multirotor (e.g., DJI Phantom [2]) is wind-limited to 20 mph, which corresponds roughly to a windy day at the beach. This challenge is exacerbated by the presence of complex airflow phenomena (e.g., ground and surface effects) when the drone operates in proximity to obstacles or in urban canopies.

Modern multirotor systems rely almost exclusively on *indirect* methods to sense wind, e.g., position error measured by onboard sensors. This approach – as opposed to directly sensing the wind and anticipating its forces – is used in part because existing anemometers (e.g., conventional pitot tubes and hot-wires) are typically too slow, insensitive, or lack the form-factor for deployment on multirotor systems. In this work, we leverage an omnidirectional flow sensor [1] based on micro-electro-mechanical systems (MEMS) hot-wire technology [1, 3, 4] to *directly* sense wind for real-time compensation. We demonstrate the effectiveness of a 'wind-aware' controller that uses turbulent airflow measurements to improve multirotor performance in gusty conditions.

Figure 1: FlowDrone calculates fast and accurate wind estimates with the MAST (inset) to achieve superior flight performance in gusty conditions.

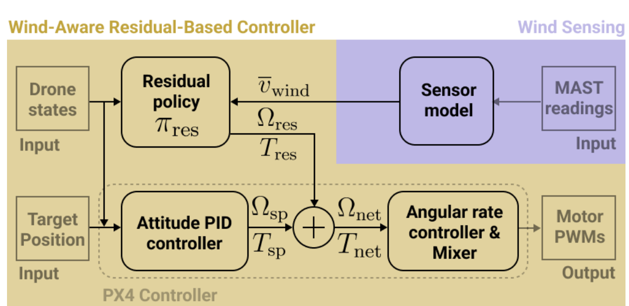

Figure 2: Controller diagram showing the wind sensing and the wind-aware residual-based controller. Taking MAST voltage readings and drone states as input, the controller computes motor PWM commands compensating for wind disturbances.

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

Related work in drone control in wind. Modern drones typically treat wind as an external disturbance and rely on a feedback controller to perform gust rejection [5, 6], e.g., using techniques from robust and adaptive control [7, 8, 9]. Recently, techniques from adaptive control have also been combined with deep neural network representations of aerodynamic effects in order to perform multirotor control in the presence of wind [10]. Alternate approaches to wind estimation include utilizing an extended Kalman filter with measurements from the drone's IMU (e.g., as implemented by the widely used PX4 controller [11]). Such wind estimates can then be utilized by a feedback controller for gust rejection [12]. In this work, we seek to leverage sensors that directly measure wind with adequate accuracy and frequency for improving multirotor drone control.

2 Hardware Platform and Wind Sensing

Drone hardware. The FlowDrone platform is built on the Holybro X500 platform equipped with the Pixhawk 4 autopilot, which additionally uses the PX4FLOW Camera and LIDAR-Lite v3 for state estimation. The state estimate is passed to the Raspberry Pi 4 companion computer over ROS2 at 100 Hz. Onboard sensing and computation allows for deployment outdoors.

Sensors and wind estimation. For wind estimation, we use the MAST (MEMS Anemometry Sensing Tower) – an omnidirectional flow sensor suitable for integration on multirotor UAVs [1]. The MAST consists of five pentagonally-arranged MEMS Hotwire chips (see Fig. 1 inset). Vertical PCBs project the MEMS Hotwires sufficiently above the rotor plane (150 mm) and into the free-stream velocity. The MEMS Hotwires consist of a Wheatstone bridge of platinum wire arrays. One leg of the bridge is exposed to the surrounding flow, and the differential convective cooling is measured as a voltage.

Sensor model. A sensor model on board the Raspberry Pi estimates the wind vector (direction and magnitude) from MAST voltages using two distinct neural networks trained on data from wind tunnel experiments. The MAST achieves the following performance over 360° and 0-5 m/s: 1.6° expected angle prediction error (with an empirical 95% error upper bound of 5.0°), and 0.14 m/s expected speed prediction error (with an empirical 95% error upper bound of 0.36 m/s). The low-weight MAST and associated sensor model provide low-latency (1.56 ms) wind estimates to the control architecture that are as accurate as any existing method implementable on UAVs.

3 Wind-Aware Control

3.1 Simulated environment with wind

Since it is expensive and time-consuming to learn a residual-based policy in diverse wind conditions directly on hardware, we train the policy in a simulated environment built upon `gym-pybullet-drones` [13], an open-source drone simulation environment based on the PyBullet simulator [14]. We model the bluff-body and induced drag components from wind as in [15]. The simulated wind is generated in the positive X direction in the world frame. We vary the wind speed in a step-like profile (mimicking the measurements taken in real wind conditions as in Fig. 3 (Bottom Left)), which consists of three stages: "Low", "Slope", and "High". The wind speed and duration of the three stages are all randomized to train a robust wind-aware policy. An additional dip of the wind speed is added to the "High" stage to mimic sudden instability, which was frequently observed during real flights as the flow generated by the fan array is unsteady and not spatially uniform.

3.2 Residual policy for wind compensation

In order to perform wind compensation, we train a residual control policy on top of the open-source PX4 attitude controller (see Fig. 2 for a visualization of the overall controller architecture). Using a reinforcement learning approach (instead of a model-based approach) allows us to train a nonlinear policy which can potentially leverage temporal structure in wind gusts. The trained residual policy takes real-time wind estimates and drone states as input and outputs additional body angular rates

Ω_{res} and thrust T_{res}, which are then added to the respective setpoints $\Omega_{\text{sp}}, T_{\text{sp}}$ calculated by the upstream PX4 attitude PID controller. The net setpoints $\Omega_{\text{net}}, T_{\text{net}}$ are then fed into the downstream angular rate controller and mixer. The overall controller runs at 40 Hz. The residual policy π_{res} is parameterized as a multi-layer perceptron (MLP) with hidden layer sizes $[512, 256, 128, 128]$ and ReLU activation. The input to the residual policy

$$[\Omega_{\text{res}}, T_{\text{res}}] = \pi_{\text{res}}(r, \Psi, v, w, \overline{v}_{\text{wind}}) \tag{1}$$

includes the drone's current 3D position ($r = [x, y, z]$), orientation in roll, pitch, and yaw ($\Psi = [\phi, \theta, \psi]$), linear velocity ($v = \dot{r}$), and angular velocity ($w = [w_x, w_y, w_z]$), all relative to the world frame. In addition, the residual policy takes as input the wind measurement at the current timestep t and past four timesteps (only the components in the X direction of the world frame). We skip $t_{\text{s}} = 5$ steps between each wind measurement ($\overline{v}_{\text{wind}} = [v^{t}_{\text{wind}}, v^{t-t_{\text{s}}}_{\text{wind}}, v^{t-2t_{\text{s}}}_{\text{wind}}, v^{t-3t_{\text{s}}}_{\text{wind}}, v^{t-4t_{\text{s}}}_{\text{wind}}]$). Since the control loop is 40 Hz, this roughly covers a time window of 0.5 s. The outputs, Ω_{res} and T_{res}, are normalized between $[-0.3, 0.3]$ rad/s and $[-1, 1]$ N respectively using the Tanh activation function. We also find that large values of body rate setpoints Ω_{sp} from the attitude controller can hinder the training progress of the residual policy. Thus we clip Ω_{sp} to be in $[-0.1, 0.1]$ rad/s.

On the real hardware platform, the wind measurements are obtained by processing the MAST readings through the sensor model described in Sec. 2. In simulation, we treat the wind sensor model as perfect, meaning the wind measurement is the same as the simulated wind. Since the wind profile contains a non-trivial amount of noise, we filter the measurements using a rolling maximum over the past 0.1 s both in simulation and hardware, as shown in Fig. 3 (Bottom Left).

We train the residual policy in simulation using Soft Actor Critic [16]. The task of the drone is to hover at the target position $[0, 0, 1]$ m in an inertial East-North-Up frame for a 10-second horizon. The reward function is defined as the negative of the distance of the current drone position to the target. We randomize the initial position and orientation of the drone at each rollout; the ranges of the initial 3D positions, roll and pitch angles, and yaw angle are $[-30, 30]$ cm, $[-0.1, 0.1]$ rad, and $[-0.3, 0.3]$ rad. The model is trained with 10 million total simulation timesteps.

4 Hardware Experiments

We evaluated the wind-aware controller's performance in tracking a hover setpoint in the presence of a wind gust (same task as in simulation) in a real setting. We compare the following three controllers:
Wind-Aware Residual-Based Controller ("wind-aware"): described in Sec. 3 and Fig. 2;
Wind-Unaware Residual-Based Controller ("wind-unaware"): has the same architecture and is trained with the same conditions as wind-aware, except that the residual policy does not have access to the wind estimate $\overline{v}_{wind}$. Differences in performance between this controller and the wind-aware controller thus directly provide evidence for the benefits of utilizing wind measurements for control;
PX4 Attitude Controller ("baseline"): the popular open-source PX4 Autopilot for attitude control. Referencing Fig. 2, this controller is the "PX4 controller" at the bottom half without wind sensing or residual policy.

4.1 Experiment setup

We conducted 10 flights for each of the three controllers in controlled gust conditions. The supplementary video [17] demonstrates representative trials of all three controllers. We used six high velocity (350 cfm) blowers to generate the gusts during hardware evaluation. The blowers were arranged in a 2×3 array (Fig. 1), with the top row blowers inverted, generating a flow volume of 22×86 cm at the blower exit. The peak gust speed at the drone's location was approximately 5 m/s.

For each test, the drone was commanded to take off and hover at $r_{sp} = [0, 0, 1]$ m, $\Psi_{sp} = [0, 0, 0]$ rad, which remained the setpoint for the rest of the flight. From $t = [0, 12)$ s, the drone hovers in zero wind. This delay ensures that the residual policy has not learned an open-loop prediction of when the gust will start. At $t = 12$ s, the fans are turned to their maximum setting for the remaining 18 s of flight. The fans are oriented to blow in the $+X$ direction of the inertial frame.

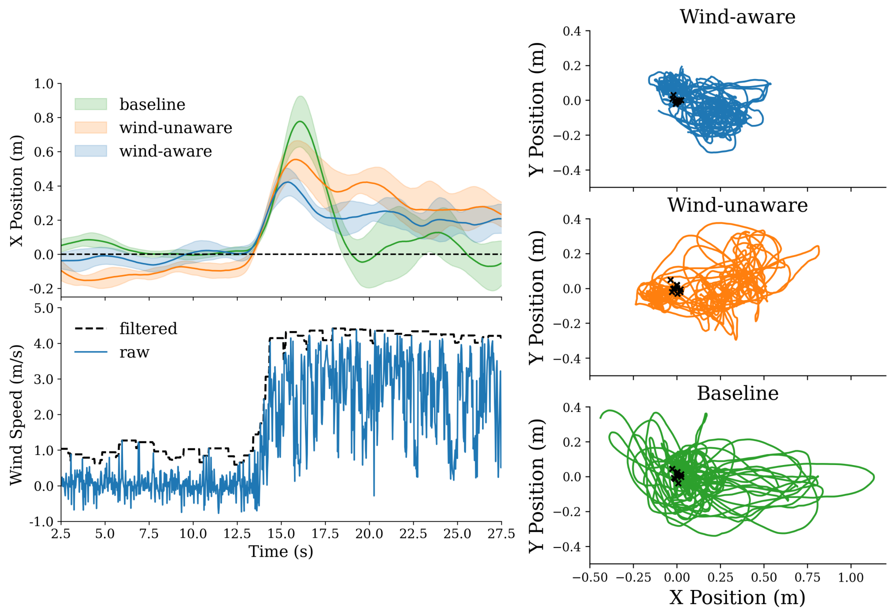

Figure 3: (Top Left) Average X trajectory (line) for each controller with pointwise standard deviation (shaded). (Bottom Left) Raw and filtered wind estimates for a representative gust measured during a wind-aware flight. (Right) Trajectories of all ten trials for each controller. The starting positions for each trial are marked as $\times$s.

4.2 Results and Discussions

The 10 trajectories for each controller are plotted in Fig. 3 (Right). Qualitatively, the wind-aware trajectory is more concentrated in both X and Y. In Fig. 3 (Top Left), the average X-trajectory of each controller is plotted against time, with bands for ± 1 standard deviation. Again, the wind-aware controller deviates the least from the setpoint.

In Fig. 3 (Bottom Left), we plot a representative gust measured during a wind-aware flight. The raw wind estimate is shown, as well as the input to the residual policy, to which we applied a moving maximum filter over the previous 100 wind estimates (0.1 seconds).

The performance of each controller is evaluated by three metrics on the X trajectory: max error, mean-squared error, and total range. The results are shown in Table 1. Max error penalizes the gust onset effect, while mean squared error (MSE) penalizes error over the entire trajectory. The range metric additionally penalizes under- or over-shoot. By each metric, the wind-aware controller outperforms the others; this illustrates the wind-aware controller's ability to reduce both maximal and overall error in the presence of wind. In terms of max error, the wind-aware controller improves on average by 44% over the baseline controller and by 24% over the wind-unaware controller.

	Wind-aware	Wind-unaware	Baseline
Max Error (m)	$\mathbf{0.441}$ (0.064)	0.582 (0.094)	0.780 (0.142)
MSE (m^2)	$\mathbf{0.035}$ (0.006)	0.079 (0.013)	0.057 (0.016)
Range (m)	$\mathbf{0.538}$ (0.072)	0.773 (0.100)	0.962 (0.222)

Table 1: Hardware performance of each controller along several metrics, with the standard deviation in parentheses and the minimum entry of each row in bold.

5 Conclusions

We have presented the FlowDrone: a multirotor UAV platform that integrates fast-response hot-wire sensors for real-time wind estimation. We implemented a reinforcement learning pipeline for active gust rejection. We demonstrated significant improvements in tracking a hover setpoint with the wind-aware controller in gusty conditions and the importance of direct wind measurements.

References

[1] N. Simon, A. Piqué, D. Snyder, K. Ikuma, A. Majumdar, and M. Hultmark. Fast-response hot-wire flow sensors for wind and gust estimation on UAVs, Sept. 2022. URL http://arxiv.org/abs/2209.06643.

[2] Phantom 4 pro v2.0. https://www.dji.com/phantom-4-pro-v2/specs.

[3] Y. Fan, G. Arwatz, T. Van Buren, D. Hoffman, and M. Hultmark. Nanoscale sensing devices for turbulence measurements. *Experiments in Fluids*, 56(7):1–13, 2015.

[4] M. Fu, Y. Fan, C. Byers, T. Chen, C. B. Arnold, and M. Hultmark. Elastic filament velocimetry (EFV). *Measurement Science and Technology*, 28(2):025301, 2016.

[5] S. Tang and V. Kumar. Autonomous flight. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:29–52, 2018.

[6] G. Hoffmann, H. Huang, S. Waslander, and C. Tomlin. Quadrotor helicopter flight dynamics and control: Theory and experiment. In *AIAA guidance, navigation and control conference and exhibit*, page 6461, 2007.

[7] J. H. Gillula, H. Huang, M. P. Vitus, and C. J. Tomlin. Design of guaranteed safe maneuvers using reachable sets: Autonomous quadrotor aerobatics in theory and practice. In *2010 IEEE international conference on robotics and automation*, pages 1649–1654, 2010.

[8] S. Mallikarjunan, B. Nesbitt, E. Kharisov, E. Xargay, N. Hovakimyan, and C. Cao. L1 adaptive controller for attitude control of multirotors. In *AIAA guidance, navigation, and control conference*, page 4831, 2012.

[9] 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*, 7 (2):690–697, 2021.

[10] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural-fly enables rapid learning for agile flight in strong winds. *Science Robotics*, 7(66), 2022.

[11] Using the ECL EKF. URL https://docs.px4.io/v1.9.0/en/advanced_config/tuning_the_ecl_ekf.html#mc_wind_estimation_using_drag.

[12] F. Schiano, J. Alonso-Mora, K. Rudin, P. Beardsley, R. Siegwart, and B. Sicilianok. Towards estimation and correction of wind effects on a quadrotor UAV. In *IMAV 2014: International Micro Air Vehicle Conference and Competition 2014*, pages 134–141, 2014.

[13] J. Panerati, H. Zheng, S. Zhou, J. Xu, A. Prorok, and A. P. Schoellig. Learning to fly—a gym environment with pybullet physics for reinforcement learning of multi-agent quadcopter control. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[14] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2018.

[15] W. Craig, D. Yeo, and D. A. Paley. Geometric attitude and position control of a quadrotor in wind. *Journal of Guidance, Control, and Dynamics*, 43(5):870–883, 2020.

[16] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2018.

[17] Anonymized supplementary video. https://drive.google.com/file/d/1v_--Wy9gO2fFOLVoUE2wnNTqM-Ha18N7/view?usp=sharing.