Robotics and Autonomous Systems 123 (2020) 103314

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot

Estimating and reacting to forces and torques resulting from common aerodynamic disturbances acting on quadrotors

Christopher D. McKinnon *, Angela P. Schoellig

University of Toronto, Institute for Aerospace Studies, Toronto, Ontario, Canada

ARTICLE INFO

Article history:
Received 23 October 2018
Received in revised form 21 September 2019
Accepted 30 September 2019
Available online 5 October 2019

Keywords:
Force estimation
Machine learning
Quadrotors

ABSTRACT

Quadrotors are increasingly expected to perform a wide variety of tasks that put them in close proximity to other objects and surfaces in the environment (including other quadrotors), where they are often subject to significant external forces and torques resulting from aerodynamic effects. We present an algorithm – based on an Unscented Kalman Filter – that estimates such forces and torques without making assumptions about their source, allowing us to bypass much of the complexity involved in modeling how wind currents interact with quadrotor dynamics. Furthermore, our algorithm does not rely on special sensors, making it suitable for commercial systems where payload and add-on capabilities are limited. Via experiment we show that the estimation algorithm can be used in conjunction with controls and machine learning for detecting and avoiding downwash and walls, and for tracking wind from a fan. We also show that the algorithm is sensitive enough to measure even small changes in force and torque.

Quadrotors often experience significant external forces that affect their dynamic behavior. Yet as advances in sensing and computing have led to the development of exceptionally small and capable quadrotors, the variety of tasks we expect them to perform has increased, and so, too, has the variety and complexity of the aerodynamic disturbances they must cope with. For example, in inspection applications we expect quadrotors to physically interact with and fly in close proximity to surfaces and objects in the environment [1].

1. Introduction

Quadrotors often experience significant external forces that affect their dynamic behavior. Yet as advances in sensing and computing have led to the development of exceptionally small and capable quadrotors, the variety of tasks we expect them to perform has increased, and so, too, has the variety and complexity of the aerodynamic disturbances they must cope with. For example, in inspection applications we expect quadrotors to physically interact with and fly in close proximity to surfaces and objects in the environment [1–5], and in flight formation applications we expect them to fly in close proximity to each other [6–8]. Given this diverse range of applications, it is challenging to model all the force interactions that a quadrotor might encounter; and yet the ability to accurately estimate and react appropriately to external forces is often essential to the safe and effective completion of a given task. This paper presents a force estimation algorithm that does not rely on specialized sensors or models of the task at hand, but rather estimates the aerodynamic forces directly. The algorithm thereby bypasses much of the complexity involved in modeling how air currents interact with quadrotor dynamics. Based on the Unscented Kalman Filter (UKF), the algorithm can be used to map the forces experienced by a quadrotor as it flies near such diverse objects as a fan, a wall, and another quadrotor, for example. We demonstrate via experiment how measurements of this force field can be used in combination with machine learning or admittance control to enable a quadrotor to achieve a desired behavior during each one of these interactions.

In this paper, we use the term ‘interaction’ to describe an external source exerting a force on the quadrotor. In particular, we consider external forces that are repeatable when parametrized by the state of the interaction, that is, for example, the proximity to a wall, the position relative to a fan or to another quadrotor, or the position of a quadrotor relative to the downwash of another quadrotor. We specifically focus on reacting to aerodynamic disturbances in this work, as these are commonly experienced by quadrotors and other unmanned aerial vehicles.

Compared to our previous work [9] we have: *(i)* shown that external forces and torques can be used to detect proximity to the wall, *(ii)* included a more extensive literature review, *(iii)* conducted a comparison of our force estimates to an analytical model for ground effect, and *(iv)* included more detail on our

* Corresponding author.
E-mail addresses: chris.mckinnon@robotics.utias.utoronto.ca (C.D. McKinnon), angela.schoellig@robotics.utias.utoronto.ca (A.P. Schoellig).

https://doi.org/10.1016/j.robot.2019.103314
0921-8890/

experiments and methods for acquiring the necessary parameters of our algorithm.

One well-studied aerodynamic effect is known as *downwash*, which occurs when one quadrotor creates a strong, downward flow of air that causes catastrophic loss of lift in a second quadrotor that is passing below it. Early approaches to dealing with downwash in practice involved measuring the affected region and ensuring that no quadrotor passed into it [6]. To enforce this, they used central coordination which is not always practical. Since then, in the hope of designing a controller that can compensate for the downwash effect, researchers have tried to model how the downward airflows affect the lower quadrotor [7]. Promising results were shown in simulation, assuming the lower quadrotor is equipped to measure air speed and has known aerodynamic properties. In practice, however, it can be difficult to identify these parameters, and mounting wind speed sensors out of the quadrotors own downwash may require an elaborate mount [10], which is not always possible for commercially available quadrotors.

Enabling quadrotors to function in downwash was further studied in [8]. Here, researchers designed a specialized sensor to measure the wind speed around a quadrotor, modeled the airflow induced by the downwash, and then used these in conjunction to localize the lower quadrotor relative to the downwash and plan a safe path around it. Results were promising, but like the previous example, the approach requires a specialized sensor, making it an impractical solution for commercially available quadrotors that do not accommodate the modular addition of sensors.

Instead of directly measuring wind, researchers in [11] estimated wind speed using on-board sensors and a model of how wind affects the quadrotor dynamics (simulation results only). They estimated the quadrotor's position, velocity, attitude, and body angular rate using an Extended Kalman Filter, and then used these quantities to calculate wind speed by inverting a simplified model for the wind effects on a quadrotor. While this eliminates the need for a specialized sensor, it still requires a detailed model of the quadrotor's aerodynamics and an accurate identification of several model parameters. Identifying these parameters can be an intricate and time-consuming process [12]. We will show that such detail is not necessary for tasks such as downwash detection, or even positioning relative to a fan using the forces induced by the wind disturbance. We instead bypass much of the complexity involved in modeling how wind currents affect quadrotor dynamics and estimate the aerodynamic forces directly.

Another aerodynamic often studied aerodynamic disturbance is known as the *ground effect* - the phenomenon that less thrust is required for a quadrotor to hover close to the ground. Researchers in [13] identified this relationship for a specific quadrotor, and showed how it can be used to estimate the height above the ground based on measurements of the motor turn rates required to hover. In their study, the ground effect began to reduce motor turn rates required to hover 20 cm above the ground, and reduced it by 10% once the quadrotor was within 5 cm of the ground. They used this relationship to map the height of several obstacles on the ground ranging in height from 6 to 15 cm. The quadrotors moved at 6 cm/s, 20 cm above the ground in a grid pattern guided by an external motion capture system, and produced a qualitatively accurate height map of the room. The authors noted that data from a test rig consisting of an isolated propeller with adjustable height and a load cell was consistent with analytical models of the ground effect, but not with the data from free flight. As a consequence, we chose to conduct all of our experiments on quadrotors in flight.

While analytical models have been established for identifying the effects of (and controlling for) flying in wind, near the ground, and in close proximity to other quadrotors [7,11,13], *wall effects* are less well understood. To the best of our knowledge, there are no comparable studies of the aerodynamic forces involved when flying near walls, even though researchers have suggested numerous applications where this might be useful [1–5]. For this reason, we study the effects experimentally, and use machine learning techniques to identify when the vehicle is close to the wall.

There is no comparative study of aerodynamic forces close to the wall even though researchers continue to show numerous applications where quadrotors are required to perform tasks close to walls. For the ground effect, and the effects of flying in wind fields and in close proximity to other quadrotors, analytical models have been established that can be used to assist in control and identification. To our knowledge, there does not exist a similar relationship for the wall effect. For this reason, we study the associated aerodynamic effects experimentally and use machine learning techniques to identify when the vehicle is close to the wall. We use this example to show that our algorithm is sensitive enough to measure even small changes in force and torque.

In order to lift the final requirement of an aerodynamic model for sensing forces caused by downwash or flight in proximity to surfaces, we build on work that considers external force estimation for quadrotors. This was first presented in [14] with a focus on human–quadrotor interaction. Authors used a Kalman filter and a model of the closed loop dynamics of the quadrotor (linearized around hover) to estimate force along the x-axis; the force estimate was then used as input to an admittance controller. Their analysis applies to estimating the full 3-D external force vector, but is restricted to small changes in attitude and external force.

Since then, much research has been done using external force and torque as an input to quadrotor algorithms, and force and torque estimators have evolved to better suit the demands of increasingly complex tasks. A non-linear observer was first applied to the task of estimating external forces and torques by [4], and later extended in [15]. Authors in [15] showed, in theory, how an accurate force and torque estimate could be used to reduce the risk of damage in a collision, in a tactile mapping task, for impedance control, for takeoff and landing detection, and for identifying the material of a surface by colliding with it. Experimental results in [15] were only shown for takeoff and landing detection, for collision detection and location, and for impedance control. In practice, the inputs and outputs of the non-linear observer must be carefully filtered (which can be an intricate and time-consuming process) since it does not account for noise in its formulation.

Researchers studying robot manipulation devices face similar challenges when estimating the contact force and torque involved when a manipulator interacts with a surface. Dynamics for robotic manipulators are non-linear, and measurements of any real system are noisy. Key work by [16] shows that non-linear, stochastic estimation techniques, in particular the Unscented Kalman Filter (UKF), are well suited for this task since they explicitly account for the stochastic nature of the real system in their formulation. The Extended Kalman Filter (EKF) has been used successfully to perform state estimation for a small, autonomous helicopter showing that this type of approach is capable of providing the relevant accuracy and update rate required by small, agile UAVs [17]. The purpose of the study by [16] was to compare two different recursive techniques for contact force and torque estimation. Results in simulation showed that the UKF out-performed an EKF for contact force and torque estimation. Experimental results were comparable, suggesting that other sources of error were more significant, however there are other advantages to the UKF, which we will explain later. Since both manipulators and quadrotors have highly non-linear dynamics,

we hypothesized that the UKF might be similarly extended to quadrotors. We hence use this approach to estimate external forces and torques acting on quadrotors in several experimental settings, and compare this algorithm to a non-linear observer in simulation.

There are many algorithms that modify the behavior of a quadrotor based on estimates of external forces and torques. These include: admittance control [14,15], which modifies the reference trajectory based on a force input; impedance control [15], which modifies the force a quadrotor applies to its environment given tracking error; interconnection and damping assignment [4], which modifies the apparent drag and inertial properties of a quadrotor; and tool manipulation [18], which involves applying a tool to a surface with a desired force along a trajectory. These all define a control law directly in terms of the external force and torque. We will show that our estimation algorithm is amenable to this kind of algorithm by demonstrating its performance with an admittance controller in experiments. Another set of algorithms defines a set of discrete state–action pairs, such as take-off and landing detection or collision detection [15]. For these specific examples, a force threshold is often sufficient to distinguish between two states since the forces experienced in each state of the interaction are significantly different. We will demonstrate that this is the case for detecting downwash. Choosing these thresholds manually can be difficult when dealing with smaller forces, especially when there is no model for how the forces vary during the interaction (e.g. during flight close to a wall). In this case, we use machine learning techniques that have been successfully employed in ground robotics for contact-based terrain classification.

Contact-based terrain classification often relies on data from a probe being dragged across the surface in question [19,20] or from actuator and accelerometer measurements from a robot driving over various types of terrain [21,22]. Statistical or spectral properties of these signals are used as features to distinguish between the different terrain types. Machine learning algorithms assign a set of values for these features to each terrain type, which allows these algorithms to distinguish between many different types of terrain without knowing the shape of these sets ahead of time [20]. This property is advantageous for wall detection since we do not know the relationship between forces and torques experienced by a quadrotor and its proximity to the wall.

In [23], a comparison of different machine learning approaches showed that a Support Vector Machine (SVM) was well suited for terrain classification compared to other methods such as a probabilistic Neural Network, k-nearest neighbor searches, decision trees, and Naive Bayes. Other advantages of SVMs are that they result in a convex optimization problem, have only a few parameters, and can capture complex decision boundaries. For these reasons, and because it performed well on our data sets in initial trails, we chose to use an SVM for our application; namely, for wall detection using only measurements of the quadrotors position, attitude, and motor speed.

Our first contribution to the study of force and torque estimation is that our estimator can handle noisy measurements. Based on an Unscented Kalman Filter (UKF), we develop an external force and torque estimator that: *(i)* uses a non-linear model for the quadrotor dynamics; *(ii)* explicitly takes into account sensor noise and imperfections in our quadrotor model; and *(iii)* is lightweight enough to be implemented in applications that require high update rates.

Our second contribution is that we demonstrate via experiment how the estimated values of force and torque can be used to react to a wide variety of aerodynamic disturbances without explicitly modeling them. Specifically: we use force estimates to detect downwash and design a downwash avoidance strategy; we use an admittance controller to enable a quadrotor to track the center of a fan; and we detect proximity to a wall by means of a machine learning algorithm.

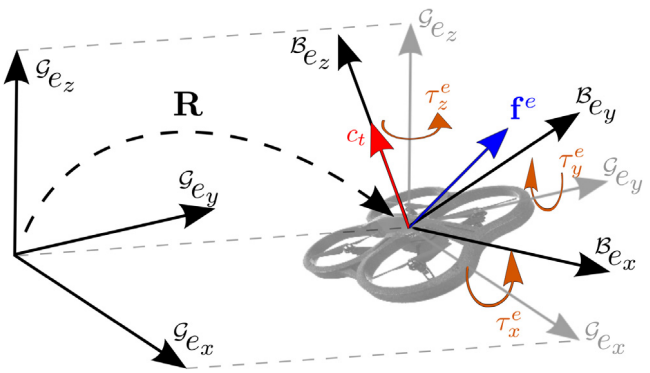


Fig. 1. Coordinate frames and variable definitions for the quadrotor where $\mathcal{B}$ is the body-fixed frame and $\mathcal{G}$ is the global frame. The collective thrust $\mathbf{c}_t$ produced by the four motors is shown in red. External forces $\mathbf{f}^e = (f_x^e, f_y^e, f_z^e)$ and torques $\boldsymbol{\tau}^e = (\tau_x^e, \tau_y^e, \tau_z^e)$ are shown in blue and orange, respectively, and expressed in global coordinates. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

2. Notation

In this paper, vectors are represented by boldface, lowercase characters and matrices by boldface, uppercase characters. Unit vectors are represented by $\check{(\cdot)}$. We denote the $(n \times n)$ identity matrix by $\mathbf{1}_{n\times n}$. The first and second derivative of a quantity with respect to time are denoted by $\dot{(\cdot)}$ and $\ddot{(\cdot)}$, respectively.

For a (3×1) vector, $\mathbf{a} \triangleq [a_x, a_y, a_z]^T$, we define $\mathbf{a}^\times$ as the (3×3) skew-symmetric cross product matrix,

$$a^\times \triangleq \begin{bmatrix} 0 & -a_z & a_y \\ a_z & 0 & -a_x \\ -a_y & a_x & 0 \end{bmatrix}. \tag{1}$$

3. Force and torque estimation

This work focuses on force-based interactions between a quadrotor and its environment. A core component of this work, therefore, is an algorithm that estimates external forces and torques. External forces and torques are quantities that cannot be explained by our first-principles quadrotor model but are exerted by external sources. We present a force/torque estimation scheme based on the Unscented Kalman Filter (UKF), which carefully models the source of process and measurement noise. This greatly simplifies the tuning process because many tuning parameters of the estimation scheme can be derived from the noise properties of the sensors and actuators.

3.1. Quadrotor dynamics

The estimation algorithm is based on a first principles model of the quadrotor augmented with external forces and torques. In this section, we present the continuous-time dynamics model.

3.1.1. Coordinate frames

The global frame $\mathcal{G}$ is defined by axes $\{{}^{\mathcal{G}}\check{\mathbf{e}}_x, {}^{\mathcal{G}}\check{\mathbf{e}}_y, {}^{\mathcal{G}}\check{\mathbf{e}}_z\}$, with ${}^{\mathcal{G}}\check{\mathbf{e}}_z$ pointing in the opposite direction to the gravity vector. The origin of the body frame $\mathcal{B} = \{{}^{\mathcal{B}}\check{\mathbf{e}}_x, {}^{\mathcal{B}}\check{\mathbf{e}}_y, {}^{\mathcal{B}}\check{\mathbf{e}}_z\}$ is located at the center of mass of the quadrotor, with ${}^{\mathcal{B}}\check{\mathbf{e}}_x$ pointing in the preferred forward direction. The ${}^{\mathcal{B}}\check{\mathbf{e}}_z$-axis is perpendicular to the plane of the rotors, pointing vertically up against gravity during hover (see Fig. 1). The rotation matrix $\mathbf{R}$ transforms quantities expressed in $\mathcal{G}$ to $\mathcal{B}$.

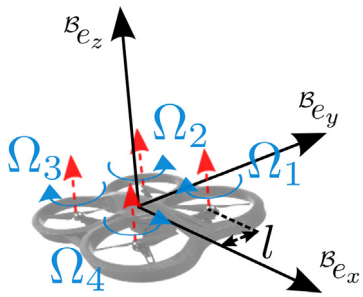


Fig. 2. Each motor is a distance l from the x- (or y)-axis, and produces a thrust c_i shown in red. The direction of rotation for each motor is shown in light blue. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

3.1.2. Translational dynamics

The quadrotor is modeled as a rigid body with mass m and (3×3) inertia matrix $\mathbf{I}$. This model neglects aerodynamic effects but has proven accurate in experiments at low speeds where unmodeled aerodynamic effects are not significant [24].

The quadrotor is actuated by four propellers. Individually, each motor i, $i \in \{1, 2, 3, 4\}$ produces a thrust proportional to the squared motor turn rate [6], $c_i = k_i \Omega_i^2$, where the constant of proportionality k_i may vary depending on the individual propeller efficiency. The individual forces sum up to give the collective thrust

$$c_t = \sum_{i=1}^{4} k_i \Omega_i^2, \tag{2}$$

which acts along ${}^{\mathcal{B}}\check{\mathbf{e}}_z$.

Letting $\mathbf{x} = [x, y, z]^T$ be the position of the center of mass of the quadrotor in the global frame $\mathcal{G}$ and $\ddot{\mathbf{x}}$ its second derivative with respect to time, the translational dynamics read as

$$\ddot{\mathbf{x}} = \mathbf{R}^T \mathbf{c}_t / m - \mathbf{g} + \mathbf{f}^e / m, \tag{3}$$

where $\mathbf{c}_t = [0, 0, c_t]^T$, $\mathbf{g} = [0, 0, g]^T$ is the gravitational force with gravitational constant g, and $\mathbf{f}^e$ is a (3×1) vector of external forces.

3.1.3. Rotational dynamics

We model the quadrotor in the 'X' configuration, referring to Figs. 1 and 2, where each motor is a distance l away from the body x-axis and y-axis. The motors act in pairs to produce a thrust differential that results in a torque, which is conveniently expressed in the body frame, $\boldsymbol{\tau}^m = [\tau_x^m, \tau_y^m, \tau_z^m]^T$. Referring to [6] and Fig. 2, the x- and y-components of $\boldsymbol{\tau}^m$ are calculated using

$$\tau_x^m = l(k_1 \Omega_1^2 + k_2 \Omega_2^2 - k_3 \Omega_3^2 - k_4 \Omega_4^2) \tag{4}$$

and

$$\tau_y^m = l(-k_1 \Omega_1^2 + k_2 \Omega_2^2 + k_3 \Omega_3^2 - k_4 \Omega_4^2). \tag{5}$$

In addition, each motor produces a torque M_i, $i \in \{1, 2, 3, 4\}$ about its own axis of rotation, which is opposite to its direction of rotation (see Fig. 2). This torque is also proportional to the squared motor turn rate by constants p_i, $M_i = p_i \Omega_i^2$ [6]. Motors 2 and 4 rotate in the positive ${}^{\mathcal{B}}\check{\mathbf{e}}_z$ direction opposite to motors 1 and 3. The resulting torque is

$$\tau_z^m = p_1 \Omega_1^2 - p_2 \Omega_2^2 + p_3 \Omega_3^2 - p_4 \Omega_4^2. \tag{6}$$

The external torque, $\boldsymbol{\tau}^e$, which comes from unmodeled external sources, is expressed in the global frame (see Fig. 1).

Letting $\boldsymbol{\omega} = [\omega_x, \omega_y, \omega_z]^T$ represent the angular velocity of the quadrotor expressed in the body frame, the rotational dynamics read as

$$\mathbf{I}\dot{\boldsymbol{\omega}} = \mathbf{R}\boldsymbol{\tau}^e + \boldsymbol{\tau}^m - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}, \tag{7}$$

where $\mathbf{R}$ reflects the quadrotor's orientation.

The orientation of the body frame with respect to the global frame can also be represented by the (4×1), unit quaternion $\mathbf{q} = [q_0, \mathbf{q}_v]^T$,

$$\mathbf{q} \triangleq \begin{bmatrix} q_0 \\ \mathbf{q}_v \end{bmatrix} = \begin{bmatrix} \cos(\theta/2) \\ \check{\mathbf{u}} \sin(\theta/2) \end{bmatrix}. \tag{8}$$

The unit quaternion describes a single rotation by an angle θ about an axis $\check{\mathbf{u}}$, which is a unit vector expressed in $\mathcal{G}$. Quaternions have several important properties: they are smooth, singularity-free, and less susceptible to round-off errors than rotation matrices [25]. However, like rotation matrices, unit quaternions must sometimes be re-normalized due to machine precision and round-off errors. According to [26], unit quaternion $\mathbf{q}$ can be converted to the rotation matrix $\mathbf{R}^T$,

$$\mathbf{R}^T = (2q_0^2 - 1)\mathbf{1}_{3\times 3} + 2\mathbf{q}_v\mathbf{q}_v^T - 2q_0\mathbf{q}_v^\times, \tag{9}$$

where we use the notation introduced in Section 2.

Quaternions evolve through time according to

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Xi}(\boldsymbol{\omega})\mathbf{q}, \tag{10}$$

where

$$\boldsymbol{\Xi}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & -\boldsymbol{\omega}^T \\ \boldsymbol{\omega} & -\boldsymbol{\omega}^\times \end{bmatrix}. \tag{11}$$

Together, these result in the rotational equations of motion,

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Xi}(\boldsymbol{\omega})\mathbf{q}, \tag{12}$$

$$\mathbf{I}\dot{\boldsymbol{\omega}} = \mathbf{R}\boldsymbol{\tau}^e + \boldsymbol{\tau}^m - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}, \tag{13}$$

where $\mathbf{R}$ is obtained from $\mathbf{q}$ using (9).

3.1.4. Summary of the quadrotor dynamics

The continuous-time quadrotor dynamics with state $(\mathbf{q}, \boldsymbol{\omega}, \mathbf{x}, \dot{\mathbf{x}})$ and inputs $\mathbf{c}_t$ and $\boldsymbol{\tau}^m$ are given by:

$$\dot{\mathbf{q}} = \frac{1}{2}\boldsymbol{\Xi}(\boldsymbol{\omega})\mathbf{q}, \tag{14}$$

$$\mathbf{I}\dot{\boldsymbol{\omega}} = \mathbf{R}\boldsymbol{\tau}^e + \boldsymbol{\tau}^m - \boldsymbol{\omega} \times \mathbf{I}\boldsymbol{\omega}, \tag{15}$$

$$\ddot{\mathbf{x}} = \mathbf{R}^T \mathbf{c}_t / m - \mathbf{g} + \mathbf{f}^e / m, \tag{16}$$

with

$$\mathbf{R}^T = (2q_0^2 - 1)\mathbf{1}_{3\times 3} + 2\mathbf{q}_v\mathbf{q}_v^T - 2q_0\mathbf{q}_v^\times, \tag{17}$$

and $\mathbf{c}_t$ and $\boldsymbol{\tau}^m$ being functions of motor turn rate according to (2), (4), (5), and (6).

3.2. Prediction model

The dynamics model of the quadrotor derived above enables us to predict the quadrotor's motion based on the current state of the quadrotor and the applied input. We use this information in the prediction step of the UKF. In this section, we adapt (14)–(16) to accurately represent both the discrete nature of the measurements and inputs, and the uncertainties in the model.

In the following, we use subscript k to denote the discrete-time index (i.e., $\mathbf{x}_k = \mathbf{x}(kT)$ with T being the discrete-time sampling period) and make reasonable assumptions about how quantities vary between time-steps.

3.2.1. Translational dynamics

In order to discretize the continuous-time dynamics, we assume constant acceleration in the global frame between time-steps, that is, constant external force and constant thrust, where we neglect the change in direction of $\mathbf{c}_t$ in $\mathcal{G}$ over one time-step. These are reasonable assumptions for small time-steps (in our work, $T = 5$ ms). Under these assumptions, the time-discretized translational dynamics (3) become

$$\mathbf{x}_k = \mathbf{x}_{k-1} + T\dot{\mathbf{x}}_{k-1} + \frac{1}{2}T^2\ddot{\mathbf{x}}_{k-1}, \tag{18}$$

$$\dot{\mathbf{x}}_k = \dot{\mathbf{x}}_{k-1} + T\ddot{\mathbf{x}}_{k-1}, \tag{19}$$

$$\ddot{\mathbf{x}}_k = \mathbf{R}_k^T(\mathbf{c}_{t,k} + \boldsymbol{\eta}_{c_t,k})/m - \mathbf{g} + \mathbf{f}_k^e/m. \tag{20}$$

We use $\mathbf{R}_k$ as opposed to $\mathbf{R}_{k-1}$ since the predicted motor thrust will act in the direction of $\mathbf{R}_k$ at time-step k, which corresponds to $\ddot{\mathbf{x}}_k$. The rotation matrix $\mathbf{R}_k$ is obtained using the discrete rotational dynamics in the following section.

We have also added process noise in (18)–(20). The primary source of this uncertainty is that the model for the thrust produced by each propeller, (2), is not exact. The thrust mapping is derived close to hover and does not perfectly reflect the quadrotor's behavior when its air speed and attitude are non-zero [27]. Moreover, the measurements of the motor turn rates are quantized to 8-bit, values which adds quantization noise to the system. To account for these effects, we add zero-mean Gaussian noise, $\boldsymbol{\eta}_{c_t,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{c_t})$, to the nominal thrust. The variance of the first and second element of $\boldsymbol{\eta}_{c_t,k}$ are non-zero because the estimate of the orientation of ${}^{\mathcal{B}}\hat{\mathbf{e}}_z$ is not perfect, and its orientation with respect to $\mathcal{G}$ changes over one time-step. The third element primarily accounts for uncertainty in the amount of thrust produced by the propellers.

The external forces $\mathbf{f}^e$ are expressed in the global frame (see Fig. 1). We do not assume any specific underlying dynamics for the external forces. We model its dynamics as a random walk

$$\mathbf{f}_k^e = \mathbf{f}_{k-1}^e + \boldsymbol{\eta}_{\mathbf{f}^e,k}, \tag{21}$$

where $\boldsymbol{\eta}_{\mathbf{f}^e,k}$ is zero-mean Gaussian noise, $\boldsymbol{\eta}_{\mathbf{f}^e,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\mathbf{f}^e})$, and $\mathbf{Q}_{\mathbf{f}^e}$ its diagonal covariance matrix. The expected value of $\mathbf{f}^e$ does not change as a function of time, but its variance increases. Values farther from the mean become more likely as time passes. This choice for the dynamics of $\mathbf{f}^e$ allows the UKF to explain discrepancies between the prediction and measurements by an additional external force acting on the system. The covariance, $\mathbf{Q}_{\mathbf{f}^e}$, becomes a tuning parameter. A smaller covariance indicates that we expect the force to change slowly, and a larger covariance means that we expect it to change quickly. The diagonal noise covariance indicates that components of force vary independently. Modeling force dynamics as a random walk has proven sufficient to estimate unknown, changing forces [14,15,18].

3.2.2. Rotational dynamics

To obtain the discretized rotational dynamics, we assume: *(i)* constant angular velocity in the body frame during each time-step to predict the attitude; and *(ii)* constant motor and external torque during each time-step to predict the angular velocity. Under these assumptions, the rotational dynamics (12), (13) become

$$\mathbf{q}_k = \boldsymbol{\Omega}(\boldsymbol{\omega}_{k-1})\mathbf{q}_{k-1}, \tag{22}$$

$$\boldsymbol{\omega}_k = \boldsymbol{\omega}_{k-1} + T\mathbf{I}^{-1}(\mathbf{R}_{k-1}\boldsymbol{\tau}_{k-1}^e + \boldsymbol{\tau}_{k-1}^m + \boldsymbol{\eta}_{\tau^m,k} - \boldsymbol{\omega}_{k-1} \times \mathbf{I}\boldsymbol{\omega}_{k-1}), \tag{23}$$

where

$$\boldsymbol{\Omega}(\boldsymbol{\omega}_k) = \begin{bmatrix} \cos(0.5\,\|\boldsymbol{\omega}_k\|\,T) & -\boldsymbol{\psi}_k^T \\ \boldsymbol{\psi}_k & \cos(0.5\,\|\boldsymbol{\omega}_k\|\,T)\mathbf{1}_{3\times3} + \boldsymbol{\psi}_k^\times \end{bmatrix}. \tag{24}$$

rotates $\mathbf{q}_{k-1}$ to $\mathbf{q}_k$ with $\boldsymbol{\psi}_k = \sin(0.5\,\|\boldsymbol{\omega}_k\|\,T)\boldsymbol{\omega}_k/\|\boldsymbol{\omega}_k\|$ (see [26] equation (5)). This is equivalent to multiplying $\mathbf{q}_{k-1}$ by the quaternion rotating through angle $\theta = \|\boldsymbol{\omega}\|\,T$ about axis $\breve{\mathbf{u}} = \boldsymbol{\omega}_k/\|\boldsymbol{\omega}_k\|$ in the body frame. The matrix $\mathbf{R}_k$ is obtained from $\mathbf{q}_k$ using (9).

The motor torque $\boldsymbol{\tau}^m$ is uncertain for the same reasons as $\mathbf{c}_t$ stemming from (6), which is derived close to hover. We model this uncertainty as additive zero-mean Gaussian noise $\boldsymbol{\eta}_{\tau^m,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\tau^m})$, with (3×3) diagonal covariance matrix, $\mathbf{Q}_{\tau^m}$.

Similar to the external force $\mathbf{f}^e$ we include an external torque $\boldsymbol{\tau}^e$ in the rotational dynamics equations. We model it as a random walk, where $\boldsymbol{\eta}_{\tau^e,k}$ is zero-mean Gaussian noise, $\boldsymbol{\eta}_{\tau^e,k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\tau^e})$, with $\mathbf{Q}_{\tau^e}$ being the diagonal covariance matrix,

$$\boldsymbol{\tau}_k^e = \boldsymbol{\tau}_{k-1}^e + \boldsymbol{\eta}_{\tau^e,k}. \tag{25}$$

3.2.3. Summary of the prediction model

The discrete-time quadrotor dynamics with state $\mathbf{s}_k = [\mathbf{q}_k, \boldsymbol{\omega}_k, \mathbf{x}_k, \dot{\mathbf{x}}_k, \boldsymbol{\tau}_k^e, \mathbf{f}_k^e]^T$ and inputs $\mathbf{c}_t$ and $\boldsymbol{\tau}_k^m$ are given by:

$$\mathbf{x}_k = \mathbf{x}_{k-1} + T\dot{\mathbf{x}}_{k-1} + \frac{1}{2}T^2\ddot{\mathbf{x}}_{k-1}, \tag{26}$$

$$\dot{\mathbf{x}}_k = \dot{\mathbf{x}}_{k-1} + T\ddot{\mathbf{x}}_{k-1}, \tag{27}$$

$$\mathbf{f}_k^e = \mathbf{f}_{k-1}^e + \boldsymbol{\eta}_{\mathbf{f}^e,k}, \tag{28}$$

$$\mathbf{q}_k = \boldsymbol{\Omega}(\boldsymbol{\omega}_{k-1})\mathbf{q}_{k-1}, \tag{29}$$

$$\boldsymbol{\omega}_k = \boldsymbol{\omega}_{k-1} + T\mathbf{I}^{-1}(\mathbf{R}_{k-1}\boldsymbol{\tau}_{k-1}^e + \boldsymbol{\tau}_{k-1}^m + \boldsymbol{\eta}_{\tau^m,k} - \boldsymbol{\omega}_{k-1} \times \mathbf{I}\boldsymbol{\omega}_{k-1}), \tag{30}$$

$$\boldsymbol{\tau}_k^e = \boldsymbol{\tau}_{k-1}^e + \boldsymbol{\eta}_{\tau^e,k}, \tag{31}$$

with

$$\mathbf{R}_{k-1}^T = (2q_{0,k-1}^2 - 1)\mathbf{1}_{3\times3} + 2\mathbf{q}_{v,k-1}\mathbf{q}_{v,k-1}^T - 2q_{0,k-1}\,\mathbf{q}_{v,k}^\times, \tag{32}$$

$$\ddot{\mathbf{x}}_{k-1} = \mathbf{R}_{k-1}^T(\mathbf{c}_{t,k-1} + \boldsymbol{\eta}_{c_t,k-1})/m - \mathbf{g} + \mathbf{f}_{k-1}^e/m, \tag{33}$$

and $\mathbf{c}_{t,k}$ and $\boldsymbol{\tau}_k^m$ being functions of the motor turn rates at time kT, according to (2), (4), (5), and (6).

3.3. Observation model

Measurements come from an external, high precision, camera based motion capture system, which measures the full six degree-of-freedom pose of the vehicle, $\mathbf{y}_k = (\mathbf{x}_k, \mathbf{q}_k)$, at 200 Hz. We include additive, zero-mean Gaussian measurement noise for $\mathbf{x}_k$, $\boldsymbol{\eta}_{\mathbf{x},k} \sim \mathcal{N}(\mathbf{0}, \mathbf{G}_\mathbf{x})$ and $\mathbf{q}_k$, $\boldsymbol{\eta}_{\mathbf{q},k} \sim \mathcal{N}(\mathbf{0}, \mathbf{G}_\mathbf{q})$. The (3×3) diagonal covariance matrices $\mathbf{G}_\mathbf{x}$ and $\mathbf{G}_\mathbf{q}$ depend on properties of the camera system.

3.4. Unscented filtering

The goal of the UKF is to estimate the full state of the system, $\mathbf{s}_k = (\mathbf{q}_k, \boldsymbol{\omega}_k, \mathbf{x}_k, \dot{\mathbf{x}}_k, \boldsymbol{\tau}_k^e, \mathbf{f}_k^e)$, at each time-step. In this work, we are particularly interested in estimating the external force and torque. We use an Unscented Kalman Filter approach for our analysis since it: *(i)* allows us to use a non-linear model for the quadrotor dynamics, which is accurate for a considerable range of motions including those for inspection and physical interaction; *(ii)* optimally fuses information from the dynamic model with measurements; and *(iii)* is light-weight enough to run at the rate at which we receive measurements (here, 200 Hz). Our choice of the UKF over the Extended Kalman Filter (EKF), a common alternative, is motivated by the superior performance of the UKF on many non-linear problems [25]. The UKF produces an approximation that is accurate to third order for Gaussian random variables, while the EKF is only accurate to first order [25]. In addition, the UKF does not require the derivation of analytic Jacobians of the

dynamics with respect to the state and process noise, which can be a time-consuming and tedious task for high-dimensional state variables. For a good reference on the Unscented Kalman filter, we refer the interested reader to [28, Chapter 4.2].

The UKF is a recursive Gaussian filter. At each time-step, the probability density function of the state is entirely defined by a mean and a covariance. The goal at each time-step is to go from a prior belief of the mean and covariance of the state $\{\hat{\mathbf{s}}_{k-1}, \hat{\mathbf{P}}_{k-1}\}$ to a predicted belief $\{\check{\mathbf{s}}_k, \check{\mathbf{P}}_k\}$ and then correct the prediction using measurements to get the estimate $\{\hat{\mathbf{s}}_k, \hat{\mathbf{P}}_k\}$ for time-step k. We denote predicted values by $\check{(\cdot)}$ and corrected values by $\hat{(\cdot)}$. The corrected value for one time-step is the prior value for the next.

The UKF uses a special set of points called sigma points to represent uncertainty. These points can be transformed exactly through a non-linearity, i.e. the process or observation model, and then re-combined into a mean and covariance to recover the Gaussian probability density of the state. Special care must be taken to ensure that uncertainty in the rotational states is properly accounted for during these steps because the Unscented Transform does not account for the unit-norm constraint on a quaternion. For this purpose, we follow an approach first presented in [29] for spacecraft attitude estimation called the Unscented Quaternion Estimator (USQUE). The USQUE represents the mean of a rotational state using singularity-free unit quaternions, with rotational uncertainty represented as a perturbation to the mean parametrized by a (3×1) vector of Modified Rodrigues Parameters (MRPs). MRPs are singular at $\pm 2\pi$ but do not have any constraint and may therefore be passed through the Unscented Transform directly [25]. Uncertainty greater than $\pm 2\pi$ would mean we have almost no knowledge of the attitude of the system, which is usually never the case.

3.4.1. Preliminaries

The USQUE requires us to frequently convert between local error quaternions and MRPs. The local error quaternion

$$\delta q = [\delta q_0, \delta \mathbf{q}_v^T]^T \tag{34}$$

is used to express a perturbation from the mean attitude estimate. An error quaternion is converted to an MRP

$$\delta \boldsymbol{\rho} = \frac{\delta \mathbf{q}_v}{1 + \delta q_0} \tag{35}$$

to perform operations involving the Unscented Transform. An MRP may be transformed back to an error quaternion using

$$\delta q_0 = \frac{1 - \delta \boldsymbol{\rho}^T \delta \boldsymbol{\rho}}{1 + \delta \boldsymbol{\rho}^T \delta \boldsymbol{\rho}}, \qquad \delta \mathbf{q}_v = \delta \boldsymbol{\rho}(1 + \delta q_0) \tag{36}$$

and can then be added back to the mean rotation.

3.4.2. Prediction step

The first step in each iteration of the UKF is to propagate the prior state estimate to the next time-step using the motion model (26)–(31) and the input $\mathbf{c}_{t,k}$ and $\boldsymbol{\tau}_k^m$. The mean estimate of the system's state at time-step k is denoted by $\hat{\mathbf{s}}_k = (\hat{\mathbf{q}}_k, \hat{\boldsymbol{\omega}}_k, \hat{\mathbf{x}}_k, \hat{\dot{\mathbf{x}}}_k, \hat{\boldsymbol{\tau}}_k^e, \hat{\mathbf{f}}_k^e)$. The prediction step to go from $\hat{\mathbf{s}}_{k-1}$ to the predicted belief at time k, $\check{\mathbf{s}}_k$ is outlined below.

The mean prior state $\hat{\mathbf{s}}_{k-1}$ is converted to the minimal (18×1) representation ${}^{\delta\rho}\hat{\mathbf{s}}_{k-1} = (\delta\hat{\boldsymbol{\rho}}_{k-1}, \hat{\boldsymbol{\omega}}_{k-1}, \hat{\mathbf{x}}_{k-1}, \hat{\dot{\mathbf{x}}}_{k-1}, \hat{\boldsymbol{\tau}}_{k-1}^e, \hat{\mathbf{f}}_{k-1}^e)$, where $\delta\hat{\boldsymbol{\rho}}$ is a (3×1) MRP vector. For the mean, $\delta\hat{\boldsymbol{\rho}} = \mathbf{0}$ since it represents a perturbation from the mean, which for the mean is zero. The state vector is combined with the process noise to form a (30×1) extended state $({}^{\delta\rho}\hat{\mathbf{s}}_{k-1}, \hat{\boldsymbol{\eta}}_{\tau^m}, \hat{\boldsymbol{\eta}}_{\tau^e}, \hat{\boldsymbol{\eta}}_{c_t}, \hat{\boldsymbol{\eta}}_{\mathbf{f}^e}) = ({}^{\delta\rho}\hat{\mathbf{s}}_{k-1}, \hat{\boldsymbol{\eta}})$, where the process noise has the same mean and covariance for all time-steps. This vector contains all uncertain quantities of the prediction step. We assume that we know the physical parameters of the system (such as mass and inertia). The more accurate our estimate of these parameters, the more accurate our estimate of the external forces and torques will be. In general, physical parameters may also be included in the estimated state, cf. [30]. Let $\hat{\mathbf{P}}_{k-1}$ be the (18×18) covariance matrix for the uncertainty in the prior state ${}^{\delta\rho}\hat{\mathbf{s}}_{k-1}$ and $\mathbf{Q}$ be the (12×12) stacked process noise covariance which is constant for all time-steps. The extended mean $\hat{\mathbf{z}}_{k-1}$ and covariance $\hat{\boldsymbol{\Sigma}}_{zz,k-1}$ become

$$\hat{\mathbf{z}}_{k-1} = \begin{bmatrix} \hat{\mathbf{s}}_{k-1} \\ \mathbf{0}_{12\times 1} \end{bmatrix}, \quad \hat{\boldsymbol{\Sigma}}_{zz,k-1} = \begin{bmatrix} \hat{\mathbf{P}}_{k-1} & \mathbf{0}_{18\times 12} \\ \mathbf{0}_{12\times 18} & \mathbf{Q} \end{bmatrix}. \tag{37}$$

With $L = 30$ being the dimension of $\hat{\mathbf{z}}_{k-1}$, we compute a set of $(2L + 1)$ sigma points, $\mathcal{Z}_{k-1,i}$, $i \in 1, \ldots, 2L + 1$, according to

$$\mathbf{S}_{k-1}\mathbf{S}_{k-1}^T = \hat{\boldsymbol{\Sigma}}_{zz,k-1} \tag{38}$$

$$\mathcal{Z}_{k-1,0} = \hat{\mathbf{z}}_{k-1} \tag{39}$$

$$\mathcal{Z}_{k-1,j} = \hat{\mathbf{z}}_{k-1} + \sqrt{L + \kappa}\, col_j \mathbf{S}_{k-1} \tag{40}$$

$$\mathcal{Z}_{k-1,j+L} = \hat{\mathbf{z}}_{k-1} - \sqrt{L + \kappa}\, col_j \mathbf{S}_{k-1}, \qquad j = 1, \ldots, L, \tag{41}$$

where $\mathbf{S}_{k-1}$ is the lower triangular matrix from the Cholesky decomposition of $\hat{\boldsymbol{\Sigma}}_{zz,k-1}$, and κ is a tuning parameter that should be set to two (assuming the state follows a Gaussian distribution [31]).

Each sigma point is un-stacked into prior uncertainty and process noise,

$$\mathcal{Z}_{k-1,i} = \begin{bmatrix} \hat{\mathbf{s}}_{k-1,i} \\ \hat{\boldsymbol{\eta}}_{k-1,i} \end{bmatrix}. \tag{42}$$

The MRP vector in each sigma point i is converted to an error quaternion $\delta\hat{\mathbf{q}}_{k-1,i}$ which is multiplied by the prior mean $\hat{\mathbf{q}}_{k-1}$ to get the full orientation quaternion for that sigma point

$$\hat{\mathbf{q}}_{k-1,i} = \delta\hat{\mathbf{q}}_{k-1,i} \otimes \hat{\mathbf{q}}_{k-1}, \qquad i = 0, \ldots, 2L, \tag{43}$$

where $\otimes$ represents the quaternion multiplication and adds the rotation $\delta\hat{\mathbf{q}}_{k-1,i}$ to $\hat{\mathbf{q}}_{k-1}$. Each sigma point can then be passed through the deterministic process model to get the predicted state for each sigma point at time k, $\check{\mathbf{s}}_{k,i}$.

Once propagated through the process model, each quaternion is then converted back into an error quaternion $\delta\check{\mathbf{q}}_{k,i}$ by comparing it to the predicted mean $\check{\mathbf{q}}_{k,0}$ using

$$\delta\check{\mathbf{q}}_{k,i} = \check{\mathbf{q}}_{k,i} \otimes [\check{\mathbf{q}}_{k,0}]^{-1}, \tag{44}$$

and then to MRPs $\delta\check{\boldsymbol{\rho}}_{k,i}$ so that each sigma point is now of the form ${}^{\delta\rho}\check{\mathbf{s}}_{k,i}$. These sigma points are recombined into the predicted mean and covariance using

$${}^{\delta\rho}\check{\mathbf{s}}_k = \sum_{i=0}^{2L} \alpha_i \, {}^{\delta\rho}\check{\mathbf{s}}_{k,i}, \tag{45}$$

$$\check{\mathbf{P}}_k = \sum_{i=0}^{2L} \alpha_i \left({}^{\delta\rho}\check{\mathbf{s}}_{k,i} - {}^{\delta\rho}\check{\mathbf{s}}_k\right)\left({}^{\delta\rho}\check{\mathbf{s}}_{k,i} - {}^{\delta\rho}\check{\mathbf{s}}_k\right)^T, \tag{46}$$

where

$$\alpha_i = \begin{cases} \frac{\kappa}{L+\kappa} & \text{if } i = 0, \\ \frac{1}{2}\frac{\kappa}{L+\kappa} & \text{otherwise.} \end{cases} \tag{47}$$

Finally, the mean perturbation $\delta\check{\boldsymbol{\rho}}_k$ is converted to $\delta\check{\mathbf{q}}_k$, and added to $\check{\mathbf{q}}_{k,0}$ in order to yield the predicted mean state $\check{\mathbf{s}}_k$ for this time-step. The whole process is summarized in Algorithm 1.

3.4.3. Correction step

The second step of the UKF is to correct our prediction of the state using measurements of the six degree-of-freedom pose $\mathbf{y}_k = (\mathbf{x}_k^y, \mathbf{q}_k^y)$ from our motion capture system.

Algorithm 1 Summary of Prediction Step

$\hat{\mathbf{s}}_{k-1}, \hat{\mathbf{P}}_{k-1} \leftarrow$ Mean and cov. for $\mathbf{s}_{k-1}$
$\mathbf{Q} \leftarrow$ Process noise cov.
$\hat{\mathbf{z}}_{k-1}, \hat{\mathcal{Z}}_{zz,k-1} \leftarrow$ Extended prediction state (37)
$\left\{\mathcal{Z}_{k-1,i}\right\}_{i=0}^{2L} \leftarrow$ generate sigma points (38)–(41)
for $i = 0..2L$ **do**
  $\check{\mathbf{s}}_{k,i} \leftarrow$propagate $\mathcal{Z}_{k-1,i}$ through (26)–(31)
  ${}^{\delta\rho}\check{\mathbf{s}}_{k,i} \leftarrow$ perturbation from $\check{\mathbf{s}}_{k,0}$ to $\check{\mathbf{s}}_{k,i}$
end for
${}^{\delta\rho}\check{\mathbf{s}}_k, \check{\mathbf{P}}_k \leftarrow$ combine $\left\{{}^{\delta\rho}\check{\mathbf{s}}_{k,i}\right\}_{i=0}^{2L}$ using (45), (46)
$\check{\mathbf{s}}_k \leftarrow$ add ${}^{\delta\rho}\check{\mathbf{s}}_k$ to $\check{\mathbf{s}}_{k,0}$

The generalized Gaussian correction equations are [31],

$$\mathbf{K}_k = \check{\mathcal{Z}}_{xy,k}\check{\mathcal{Z}}_{yy,k}^{-1}, \tag{48}$$

$$\hat{\mathbf{P}}_k = \check{\mathbf{P}}_k - \mathbf{K}_k\check{\mathcal{Z}}_{xy,k}^{T}, \tag{49}$$

$$\Delta\hat{\mathbf{s}}_k = \mathbf{K}_k(\mathbf{y}_k - \check{\mathbf{y}}_k), \tag{50}$$

where $\Delta\hat{\mathbf{s}}_k$ is the correction applied to the predicted state, $\mathbf{K}_k$ is the Kalman Gain, $\check{\mathcal{Z}}_{xy,k}$ is the predicted state-measurement covariance matrix, and $\check{\mathcal{Z}}_{yy}$ is the predicted measurement covariance matrix.

Below we explain how to obtain the quantities in (48)–(50). The first step to set up the correction is to form an extended measurement state $\check{\mathbf{z}}_k = ({}^{\delta\rho}\check{\mathbf{s}}_k, \boldsymbol{\eta}_x, \boldsymbol{\eta}_\rho)$ which includes the predicted measurement noise. The extended measurement is the predicted mean stacked with a (6×1) vector of zeros representing the mean noise. The extended measurement covariance is a block diagonal matrix including the predicted uncertainty covariance $\check{\mathbf{P}}_k$ and the diagonal measurement noise covariance $\mathbf{G}$ which is constant for all time-steps

$$\check{\mathbf{z}}_k = \begin{bmatrix} {}^{\delta\rho}\check{\mathbf{s}}_k \\ \mathbf{0}_{6\times1} \end{bmatrix}, \qquad \check{\mathcal{Z}}_{zz,k} = \begin{bmatrix} \check{\mathbf{P}}_k & \mathbf{0}_{18\times6} \\ \mathbf{0}_{6\times18} & \mathbf{G} \end{bmatrix}. \tag{51}$$

Note that the elements of $\check{\mathbf{z}}_k$ corresponding to $\delta\check{\boldsymbol{\rho}}_k$ are zero since this is a zero-mean perturbation. The mean and covariance from (51) is converted to a sigma point representation using (38)–(41). This gives us a set of predicted sigma points, which include the predicted uncertainty and the measurement noise. These sigma points are passed through the observation model to give us the predicted measurements

$${}^{\delta\rho}\check{\mathbf{y}}_{k,i} = \begin{bmatrix} \check{\mathbf{x}}_{k,i} + \boldsymbol{\eta}_{x,i} \\ \delta\check{\boldsymbol{\rho}}_{k,i} + \boldsymbol{\eta}_{\rho,i} \end{bmatrix}. \tag{52}$$

These sigma points are re-combined into a mean predicted measurement, ${}^{\delta\rho}\check{\mathbf{y}}_k$, and predicted measurement covariance $\check{\mathcal{Z}}_{yy,k}$ by using (45), (46) and substituting $\check{\mathbf{P}}_k$ with $\check{\mathcal{Z}}_{yy,k}$ and ${}^{\delta\rho}\check{\mathbf{s}}_k$ with ${}^{\delta\rho}\check{\mathbf{y}}_k$. The state-measurement covariance $\check{\mathcal{Z}}_{xy,k}$ is then calculated as

$$\check{\mathcal{Z}}_{xy,k} = \sum_{i=0}^{2L} \alpha_i({}^{\delta\rho}\check{\mathbf{s}}_{k,i} - {}^{\delta\rho}\check{\mathbf{s}}_k)({}^{\delta\rho}\check{\mathbf{y}}_{k,i} - {}^{\delta\rho}\check{\mathbf{y}}_k)^T, \tag{53}$$

where α_i are from (47).

Now that we have the predicted measurement, we compare it to the actual measurement from our motion capture system. The measured attitude $\mathbf{q}_k^y$ of the vehicle is compared to the predicted measurement $\check{\mathbf{q}}_k$ to get a perturbation

$$\delta\mathbf{q}_k^y = \mathbf{q}_k^y \otimes \check{\mathbf{q}}_k^{-1}, \tag{54}$$

which is converted to MRPs denoted by $\delta\boldsymbol{\rho}_k^y$. The Kalman gain $\mathbf{K}_k$ and corrected uncertainty, $\hat{\mathbf{P}}_k$, are computed using (48) and (49).

The correction to the predicted estimate is calculated by comparing the predicted measurement to the actual measurement

$$\delta\mathbf{s}_k^y = \mathbf{K}_k\left(\begin{bmatrix}\delta\boldsymbol{\rho}_k^y \\ \mathbf{x}_k^y\end{bmatrix} - {}^{\delta\rho}\check{\mathbf{y}}_k\right). \tag{55}$$

MRPs from $\delta\mathbf{s}_k^y$ are converted to an error quaternion $\delta\mathbf{q}_k^y$ which is used to update the mean of the predicted attitude. This gives us the corrected attitude for this time-step

$$\hat{\mathbf{q}}_k = \delta\mathbf{q}_k^y \otimes \check{\mathbf{q}}_k. \tag{56}$$

The other components of the prediction are updated by direct addition,

$$\hat{\boldsymbol{\omega}}_k = \delta\boldsymbol{\omega}_k^y + \check{\boldsymbol{\omega}}_k, \tag{57}$$

$$\hat{\mathbf{x}}_k = \delta\mathbf{x}_k^y + \check{\mathbf{x}}_k, \tag{58}$$

$$\hat{\dot{\mathbf{x}}}_k = \delta\dot{\mathbf{x}}_k^y + \check{\dot{\mathbf{x}}}_k, \tag{59}$$

$$\hat{\boldsymbol{\tau}}_k^{e,y} = \delta\boldsymbol{\tau}_k^{e,y} + \check{\boldsymbol{\tau}}_k^e, \tag{60}$$

$$\hat{\mathbf{f}}_k^e = \delta\mathbf{f}_k^{e,y} + \check{\mathbf{f}}_k^e, \tag{61}$$

completing the measurement update.

3.4.4. Useful approximation

The UKF is a recursive filter for which the computational effort required for each step is constant. However, evaluating the non-linear process model in the prediction step for each sigma point can be relatively expensive. Fortunately, a useful approximation exists to reduce the computation required. Details are presented in [31]. Here, we summarize the main steps.

In the case that the non-linear model has the special form

$$\mathbf{s}_k = \mathbf{g}(\mathbf{s}_{k-1}) + \boldsymbol{\eta}_k, \tag{62}$$

where $\mathbf{g}(\mathbf{s})$ is the non-linear process model, (26)–(31), the UKF prediction step can be greatly sped up by reducing the number of times we evaluate the non-linear process model [31].

In our case, the only equation in the process model that violates this assumption is (33) since the noise is multiplied by a rotation matrix that depends on the state. For small roll and pitch, the rotation matrix is roughly the identity matrix, so we can separate the noise

$$\ddot{\mathbf{x}}_k = \mathbf{R}_k^T\mathbf{c}_{t,k} - \mathbf{g} + \mathbf{f}_k^e/m + \boldsymbol{\eta}_{\ddot{\mathbf{x}},k}, \tag{63}$$

where we use $\boldsymbol{\eta}_{\ddot{\mathbf{x}},k} \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}_{\ddot{\mathbf{x}}})$ instead of $\boldsymbol{\eta}_{\mathbf{c}_t,k}$ to represent uncertainty in the mass-normalized thrust, and $\mathbf{Q}_{\ddot{\mathbf{x}}}$ is the diagonal noise covariance matrix of mass-normalized thrust. Now the process model is of the form (62). We can inflate the diagonal elements of $\mathbf{Q}_{\ddot{\mathbf{x}}}$ to account for the fact that ${}^{\mathcal{B}}\check{\mathbf{e}}_z$ is not aligned with ${}^{\mathcal{G}}\check{\mathbf{e}}_z$.

We proceed to separate the sigma points for the prediction step into two categories based on the block-diagonal partitioning of the matrix Σ_{zz}. If $N = 18$ is the dimension of the state, then there are $2N + 1$ sigma points from the dimension of the state, and $2(L - N)$ from the dimension of the noise.

To make this convenient, we re-order the indexing of the sigma points to be

$$\check{\mathbf{s}}_{k,i} = \begin{cases} \mathbf{g}(\check{\mathbf{s}}_{k-1,i}) & \text{if } i = 0, \ldots, 2N, \\ \mathbf{g}(\check{\mathbf{s}}_{k-1}) + \boldsymbol{\eta}_i & \text{if } i = (2N+1), \ldots, (2L) \end{cases} \tag{64}$$

where the first $2N + 1$ account for uncertainty in the state and the remainder account for process noise.

With some manipulation, we can re-write the expression for $\check{\mathbf{s}}_k$ as

$$\check{\mathbf{s}}_k = \sum_{i=0}^{2N} \beta_i\check{\mathbf{s}}_{k,i}, \tag{65}$$

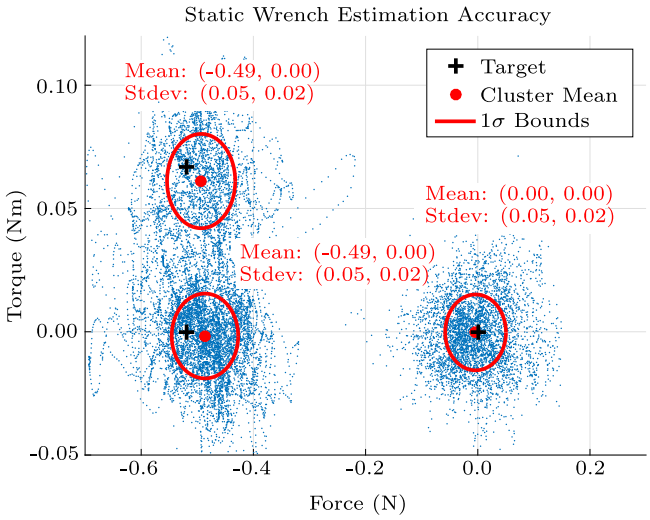


Fig. 3. Force and torque measurements from the calibration experiment. Note that these form distinct clusters corresponding to the three test cases. A mixture of Gaussians was fit to the data to get cluster statistics.

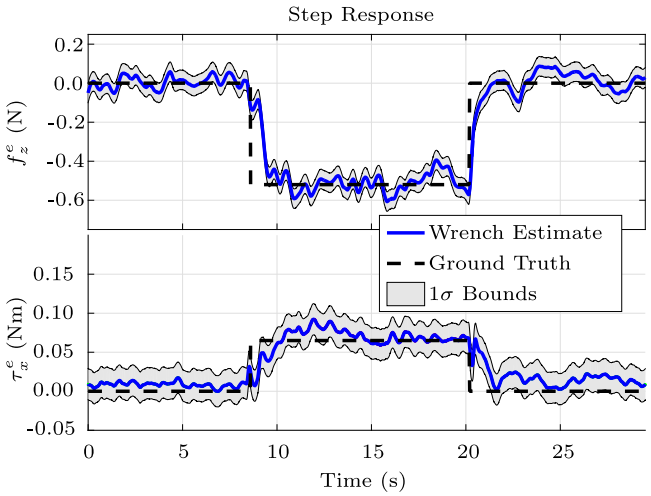


Fig. 4. Time series showing step response of force and torque for a known input. A mass of 53 g is suspended from the quadrotor at about 1 s. The steady state value of the step is shown as a black line. In both cases, the estimator converges to the correct value with a rise time of about 1 s.

where

$$\beta_i = \begin{cases} \alpha_i + \sum_{j=2N+1}^{2L+1} \alpha_j & \text{if } i = 0, \\ \alpha_i & \text{otherwise.} \end{cases} \tag{66}$$

The predicted covariance is then calculated as

$$\check{\mathbf{P}}_k = \sum_{i=0}^{2N} \beta_i (\check{\mathbf{s}}_{k,i} - \check{\mathbf{s}}_k)(\check{\mathbf{s}}_{k,i} - \check{\mathbf{s}}_k)^T + \mathbf{Q}_{\ddot{\mathbf{x}}}. \tag{67}$$

The result is that we need only evaluate the process model, $\mathbf{g}(\mathbf{s})$, $2N + 1 = 37$ times instead of $2L + 1 = 61$ times. In our case, this cuts the number of evaluations by 40%, reducing the cost of the computation significantly. The same trick can be used for the observation model. However, the observation model is fast to compute since we directly observe the state. The reduction in computational effort would be small. If the roll and pitch angles are not small, we can still use (33) which does not make the small roll/pitch angle approximation and use the algorithm as described before this section. This comes at the expense of having to evaluate the process model the full $2L + 1$ times.

3.4.5. Tuning parameters

The main parameters of the algorithm are the covariance matrices $\mathbf{G}_{\mathbf{x}}, \mathbf{G}_{\mathbf{q}}, \mathbf{Q}_{c_t}, \mathbf{Q}_{\tau_m}, \mathbf{Q}_{\mathbf{f}^e}, \mathbf{Q}_{\boldsymbol{\tau}^e}$. To obtain $\mathbf{G}_{\mathbf{x}}$ and $\mathbf{G}_{\mathbf{q}}$, the covariance of the position and attitude measurement noise, place the quadrotor stationary in the environment and take the covariance of the measured position and attitude. The covariance for the attitude is for the MRP parametrization. To obtain $\mathbf{Q}_{c_t}$ and $\mathbf{Q}_{\tau_m}$, the process noise covariance matrices, fly the quadrotor in similar manoeuvres to those that will be conducted for the target experiment making sure that there are no sources of external force or torque. Then solve (20) and (23) for $\boldsymbol{\eta}_{\mathbf{c}_t}$ and $\boldsymbol{\eta}_{\boldsymbol{\tau}_m}$ with all terms related to external force and torque to zero. Finally, $\mathbf{Q}_{\mathbf{f}^e}$, $\mathbf{Q}_{\boldsymbol{\tau}^e}$ are largely tuning parameters that control the smoothness of the estimates of $\mathbf{f}^e$ and $\boldsymbol{\tau}^e$. An initial guess for these values may be obtained by performing a similar procedure to the one for $\mathbf{Q}_{c_t}$ and $\mathbf{Q}_{\tau_m}$ but during a flight where the external force and torque are acting on the quadrotor. These values can then be tuned to obtain the desired responsiveness or smoothness of the external force and torque estimates.

4. Experimental setup and calibration

Our experimental platform was the Parrot AR.Drone 2.0 running firmware version 2.3.3. We interfaced with the AR.Drone through ROS, an open-source robot operating system [32]. More precisely, we used ROS Hydro installed on a 64-bit 12.04 Ubuntu operating system. In addition, we used the ROS *ardrone_autonomy* package [32] version 1.3.1. The ardrone_autonomy package as well as the proposed algorithm and a position controller were run on a laptop. Raw measurements from the quadrotor were received over wifi and from our motion capture system over ethernet, both at 200 Hz. Measurements were time-stamped when they were received on the laptop. The motion capture system consists of 10 Vicon MX F40 cameras. Vehicle parameters such as mass and rotational inertia are given in [33]. A detailed description of the onboard control and navigation embedded in the AR.Drone may be found in [34].

All experiments were conducted with the indoor hull shown in Fig. 1, which protects the vehicle propellers. Any configuration could be used as long as the same configuration is used during training and testing. Some configurations may induce larger forces close to the wall than others, which would make the wall detection task easier, but a comparison is outside of the scope of this paper. The goal of this paper is to show that our method works on a typical quadrotor configuration for indoor flight.

To quantify the accuracy of our system, we compared the force and torque estimates from our system in three configurations during one flight: *(i)* at hover with no external force/torque, *(ii)* with a 53 g mass suspended directly below the quadrotor, and *(iii)* with the mass suspended directly between one pair of propellers at distance l from the body x-axis. This produces three clusters centered at (0,0), (−0.52,0), and (−0.52,0.067) as shown in Fig. 3. The mean value of each cluster is within one standard deviation of the targets, and the standard deviation of measurements suggests we can measure static force and torque to within 0.05 N and 0.02 Nm respectively.

We also tested the dynamic response of our estimate. Fig. 4 shows that the rise time can be made to about 1 s while retaining good noise suppression.

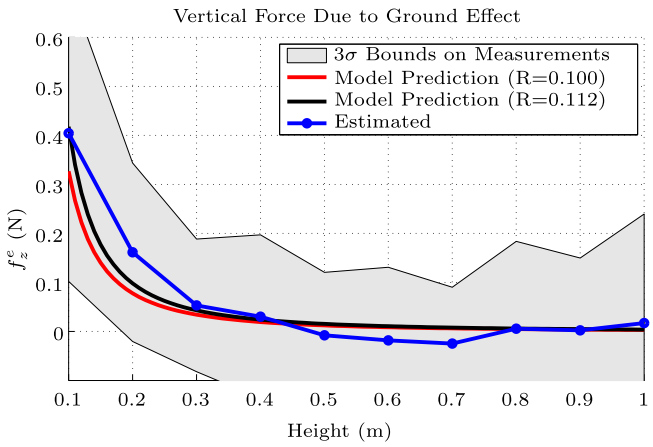
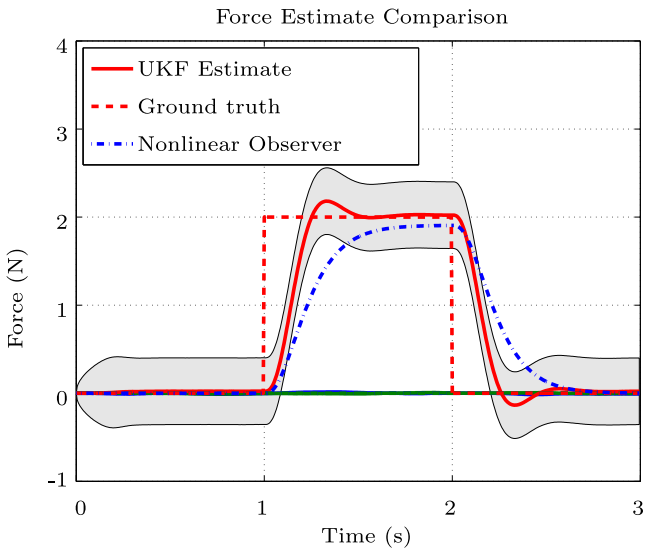
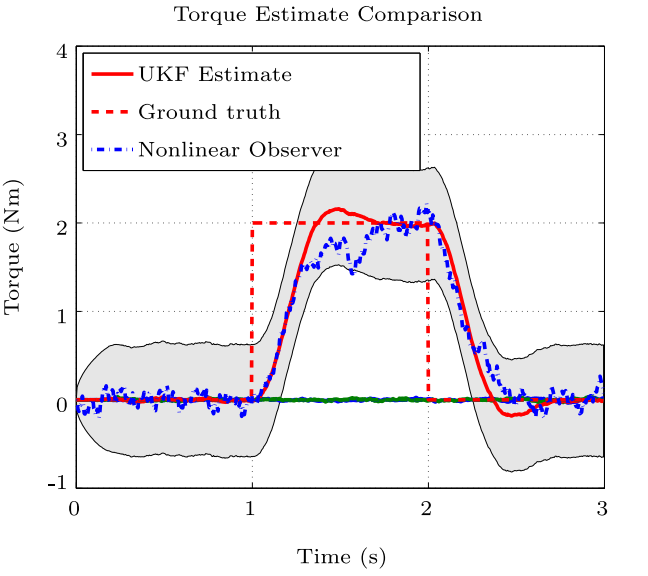


Fig. 5. Predictions of the ground effect using the Method of Images. Note that these predictions (red line) agree well with our data (blue), and consistent with [13], slightly underestimate the external force low to the ground. Comparison of forces measured in experiment to a model for the ground effect. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)



(a) Force estimate comparison in a low noise setting.



(b) Torque estimate comparison in a high noise setting.

Fig. 6. A direct comparison of the proposed algorithm using the correct values for noise covariance against a representative non-linear observer with both inputs and force and torque outputs low-pass filtered. Note that both algorithms perform well when measurements are relatively noise free (a), however the UKF based algorithm is more robust to noise as shown in (b). For the simulation shown above, position measurement noise was set to 0.01 m, and attitude measurement was set to 0.0025 rad.

5. Comparison to existing techniques

5.1. Comparison to models for the ground effect

Before studying other aerodynamic effects, we validated our estimator (Section 3) by comparing its output to established models for the ground effect, cf. [13].

The vehicle was commanded to hover for 10 s at various heights above the ground far from a wall. Measurements of the vertical force f_z^e were binned by height and the results were compared to a model for the ground effect.

The Method of Images [13], which relates T, the thrust required to hover at height z above the ground, to T_{inf}, the thrust required to hover far from the ground, by R, the radius of the propeller:

$$\frac{T}{T_{\text{inf}}} = \frac{1}{1-(\frac{R}{4z})^2}. \tag{68}$$

We plot the additional force due to ground effect ($T_{\text{inf}} - T$) in Fig. 5 for the model (68) and compare it to the output of our force estimator. The Method of Images using the actual value of R for our quadrotor, 10 cm, predicts the ground effect well within 3σ (three times the standard deviation) of our measurements. The model fit with using least squares to find the best R results in a larger R, a phenomenon that was also observed in [13] for small quadrotors. Consequently, the proposed force estimator provides results that are comparable to those found in earlier, analytic models, validating our estimation scheme in the context of estimating aerodynamic forces.

5.2. Comparison to non-linear observer

A popular method for estimating external forces and torques is the non-linear observer, which has been applied to external force and torque estimation on quadrotors in [15,18]. The purpose of this section is to compare our algorithm to a state-of-the-art non-linear observer in simulation, and to show the advantages of our algorithm in the presence of noise.

A key difference between our algorithm, based on the UKF, and a nonlinear observer is that the tuning parameters in our algorithm are the statistical properties of the dynamics model and measurements (almost all of which can be straight-forwardly identified from experiment) whereas the tuning parameters of the nonlinear observer are various low-pass filter gains which must be tuned manually. This makes finding a good set of parameters for our algorithm much easier than for a nonlinear observer which results in better performance as we will show below.

We compare our method to a non-linear observer proposed in [18] with added low pass filtering on the measurements as suggested by [15]. Filtering was essential for the non-linear observer to produce reasonable output in the presence of noise. We found that this method performs well for slowly changing forces with accurate measurements of position and attitude, however it

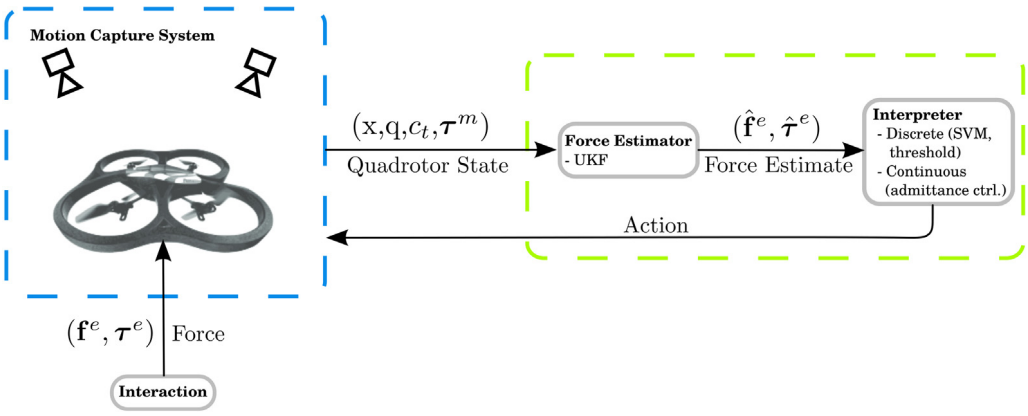


Fig. 7. High level overview of the components required to respond to a general interaction. The system includes a quadrotor and overhead camera system, and the interpretation includes a force estimator and interpreter to select the appropriate action based on the state of the interaction, which corresponds to a specific force estimate.

is difficult to find a compromise between a filter that is robust to noise and one that converges quickly.

Fig. 6 shows how our proposed estimator converges quickly to the true value and remains robust to noise. The non-linear observer can be made to perform similarly with low noise as shown in Fig. 6(a), however filtering becomes difficult when there are many noisy states and measurements in the system. This difficulty results in the poor estimate of $\boldsymbol{\tau}^e$ compared to UKF shown in Fig. 6(b). Theoretically, one might be able to tune the filtering parameters such that the two estimates are comparable, but this is a time consuming process and shows one of the advantages of the UKF-based algorithm since the parameters are in terms of statistical properties of the process model and measurements and can be determined by experiment.

In addition, if the localization system provides a changing uncertainty or measurement noise, our algorithm can take direct advantage of this by using the uncertainty from the estimation algorithm to adaptively tune the Kalman gain, which is derived in part from the measurement uncertainty. In our case, the measurement uncertainty from our motion capture system is constant.

6. Practical application

Now that we have shown that our force estimator provides reasonable results and quantified its accuracy, we can use it to measure external forces and torques acting on a quadrotor without any specific model for the mechanism causing these forces and torques. We aim to show how these estimates may be used to guide an interaction between a quadrotor and an unknown source. Our interpretation of an interaction is as follows:

1. An interaction involves an external source exerting a force on the quadrotor.
2. This force is repeatable when parametrized by the state of the interaction (i.e. proximity to the wall, position relative to the fan), which allows us to identify the state, and choose the appropriate action.

We will show three scenarios where the force estimate can be used to estimate the state of an interaction in order to choose the appropriate response. It is not necessary that our force estimate be accurate in each case, rather that it is repeatable as a function of the state of the interaction. Our general framework is shown in Fig. 7.

6.1. The wall effect

6.1.1. Force and torque profiles

Aerodynamic forces close to the wall present a unique challenge since there is no analytical model for the force–proximity relationship, so we cannot use a compact parametric model to estimate distance as was done in [13].

Our first step in studying this interaction was to find way to intuitively view how the force and torque vary in close proximity to a wall. To do this, we measured the force/torque profile experienced by the quadrotor as it approaches the wall at various heights. The quadrotor was commanded to hover for 5–10 s at a series of waypoints spaced 0.25 m apart vertically and horizontally, and move between points at 0.1 m/s to keep the flow as steady as possible.

Distinctive features of the force and torque profiles are shown in Fig. 8. The magnitude of the force in the x–y plane is most significant at 0.5 m and extends 0.5 m from the wall. We show only the magnitude in Fig. 8; however, this value is dominated by the component orthogonal to the wall, which pulls the quadrotor towards the wall with a force of about 0.1 N independent of yaw angle.

The z-component is positive close to the ground due to the ground effect, and negative close to the wall. This indicates that it is energetically unfavorable for quadrotors to fly close to the wall during indoor exploration, since they must overcome this additional negative 'wall force'. Complementary to the x–y component, the z-component decreases close to the wall, but predominantly about 1.0 m above the wall.

The magnitude of the rolling and pitching torque vector, $\|\tau^e_{x,y}\|$, is also largest in this region and together, these form a complementary set of features with which we can detect the wall.

6.1.2. Yaw-invariance of features

All experiments in Section 6.1.1 were conducted with the vehicle facing the wall at zero yaw. We will now show how $\|f^e_{x,y}\|$, $\|\tau^e_{x,y}\|$, and f^e_z vary with yaw, and in particular, how $\|f^e_{x,y}\|$ and f^e_z form two different populations according to their proximity to the wall independent of yaw, and therefore can be used to detect the wall at any yaw angle with reasonable confidence after taking many samples.

We positioned the vehicle at two different positions relative to the wall to measure each feature ($\|f^e_{x,y}\|$, $\|\tau^e_{x,y}\|$, and f^e_z), and vary the yaw angle from 0° to 360° in increments of 45° with the vehicle holding each angle for 5 to 10 s. The quadrotor is positioned 0.25 m and 3.50 m from the wall to show how this property holds even as distance to the wall is varied.

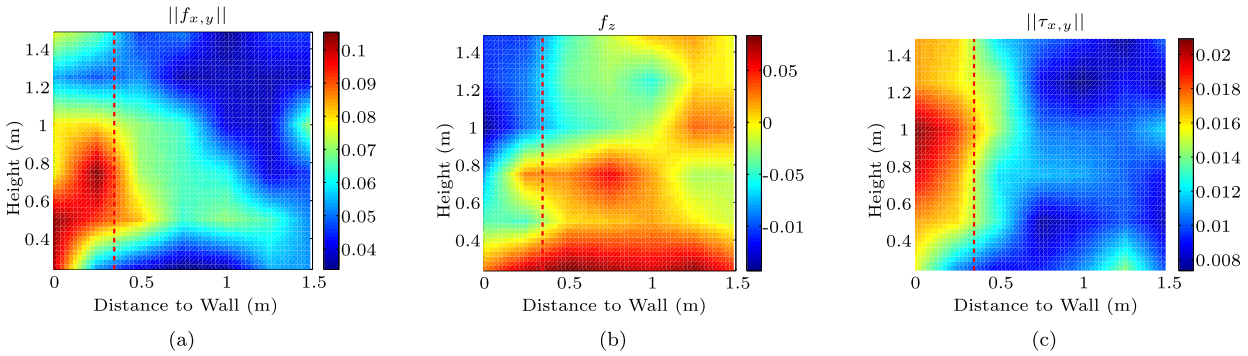


Fig. 8. External force and torque profiles close to the wall. The 2-norm of the external force in the x–y plane $\|f^e_{x,y}\|$ and the vertical force along the z-axis f^e_z both change significantly with distance to the wall compared to their standard deviations: $\|f^e_{x,y}\|$ varies by 0.052 N when approaching the wall at a height of 0.5 m with an average standard deviation of 0.022 N; f^e_z changes up to 0.062 N with an average standard deviation of 0.065 N. The 2-norm of the torque about the x- and y-axis $\|\tau^e_{x,y}\|$ increases by 0.009 Nm with an average standard deviation of 0.006 Nm. The decision boundary for points that are considered close to the wall is 0.35 m (red, dashed line), and aligns with significant changes in all three features.

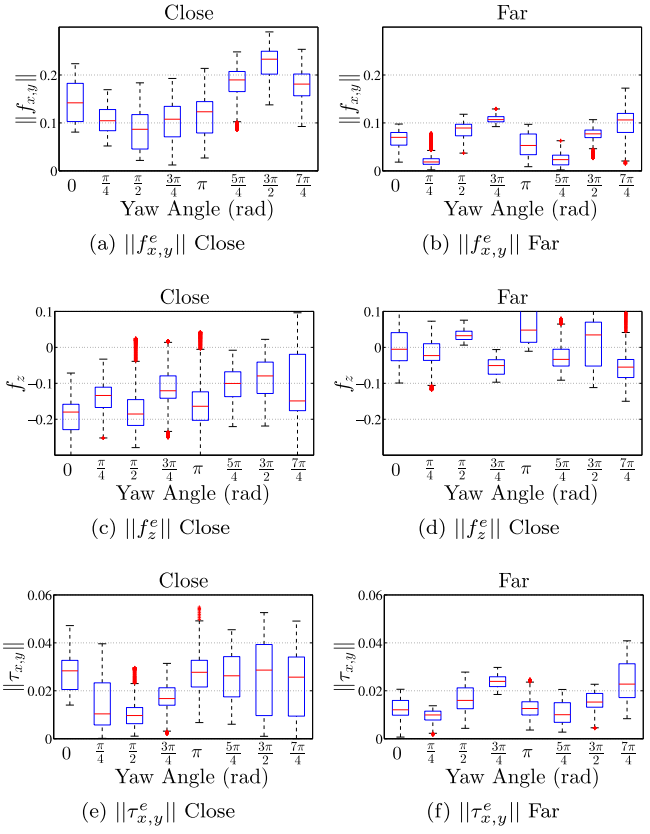


Fig. 9. Yaw dependence of external force and torque close to the wall. The distribution of the norm of external force and torque as yaw is varied. The components of force form distinct populations depending on whether the quadrotor is close to or far from the wall. The values measured for external torque have a significant amount of overlap close to and far from the wall so are not used for detecting when the quadrotor is close to the wall.

In all cases, there is some overlap between measurements taken close to and far from the wall, however $\|f^e_{x,y}\|$ and f^e_z in particular form two distinct populations depending on proximity to the wall. This allows us to obtain a reliable estimate after combining many measurements of these quantities from a single point.

Figs. 9(a)–9(d) show how $\|f^e_{x,y}\|$ and f^e_z stay within one standard deviation as the yaw angle is varied which confirms that those features are yaw-invariant. The magnitude of torque $\|\tau^e_{x,y}\|$ varies more significantly and does not markedly change with distance to the wall as shown in Figs. 9(e) and 9(f). We conclude that classification results using $\|f^e_{x,y}\|$ and f^e_z apply to cases where the vehicle approaches the wall from any angle. The torque $\|\tau^e_{x,y}\|$ is not used in the subsequent classification.

6.2. Fan

The wall effect is an example of an aerodynamic force with a magnitude just above the lower limit of what we can reliably detect with our current experimental set-up. This drove the requirement to measure the force at hover, which is a problem that simpler linear force estimators might be equally well suited for. In this section, we use a fan to generate a large aerodynamic force that disturbs the quadrotor significantly away from equilibrium, and show how the force estimate is still reliable enough to guide the quadrotor to the center of the flow driven by the fan.

Momentum theory (we refer the interested reader to [7]) is commonly used to model the type of steady, inviscid, incompressible flow that is produced by a fan propeller, and predict that there will be a strong wind along the fan's axis of rotation of the fan that generally matches the force profile measured in Fig. 10. In our experiments, the major component of the force field in the flow direction extends about 0.2 m perpendicular from the axis of the fan, which is close to the 0.3 m radius of the fan.

6.3. Downwash

Downwash is another aerodynamic disturbance that exerts a large force on one quadrotor passing below another and is of significant interest in the UAV community. We will show how our force estimator can be used to measure the forces and torques caused by downwash in a realistic experimental setting.

The first step is to build a map of the aerodynamic forces of downwash below another quadrotor. We will make one simplification based on actuator disk theory (see [7]), which tells us that the downwash flow from one rotor will reach a steady state velocity within 1–2 rotor radii. For the AR.Drone, this is 8 m/s within 0.1–0.2 m of the upper quadrotor. After this point, the shape of the flow will be roughly independent of the vertical separation of the quadrotors, which allows us to restrict our experiments to the horizontal plane.

Fig. 11 shows that the strongest downward force is contained within a 0.3 m radius of the upper quadrotor. The strong gradient of vertical force in this region is what causes the torque about the x-axis in Fig. 11(b).

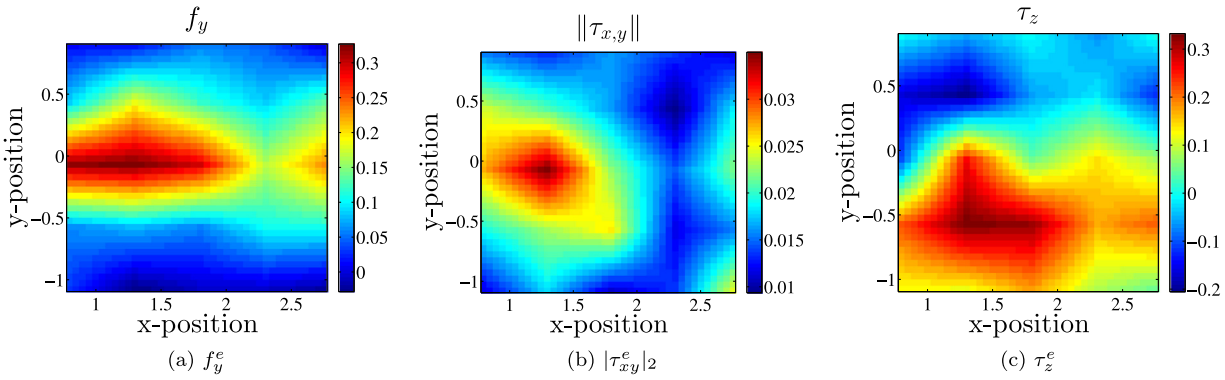


Fig. 10. This figure shows components of the estimated external force and torque when the vehicle flies in front of a fan blowing along the y-axis. The force pushing the quadrotor away from the fan is shown in (a), the norm of the torque in the plane of the floor is shown in (b), and the component of the torque about the z-axis, normal to the floor, is shown in (c). The fan was placed at the origin and the quadrotor sampled the force and torque on a 0.5 m grid. There is a strong axial component in the direction away from the fan (which matches expectations from analytical models) and a strong torque centered on the flow and close to the fan. The anti-symmetric τ_z^e profile makes sense intuitively since the fan would induce a drag force which, when centered on either side of the center of mass, would produce a torque about the body z-axis.

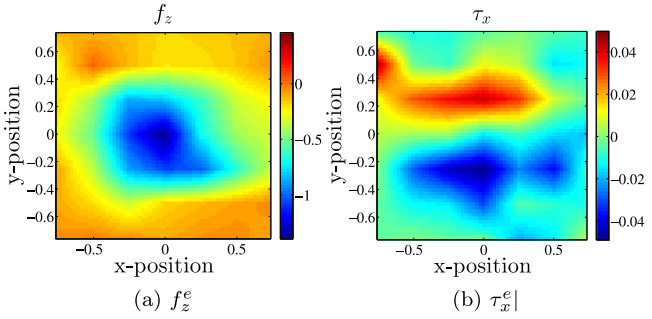


Fig. 11. External force and torque profiles induced by downwash. The force and torque profile measured by a quadrotor flying at 0.75 m/s 1.0 m below another. The upper quadrotor is set to hover at $x, y = 0$. The downwards component of the force is shown in (a). The torque about the x-axis is shown in (b). The sharp increase in downwards force below the upper quadrotor makes it difficult to fly below it slower than 0.75 m/s.

The region where the downward force is strongest is slightly larger than the width of the quadrotor. Similar to what we observed for the fan, this is likely due to the fact that we are using a quadrotor that is 0.5 m wide to take measurements of the force it experiences as opposed to the air speed at a particular point. In the following sections, we will show that this force measurement is sufficient to design a simple but effective means for downwash avoidance.

7. Interaction theory

This section outlines our approach to use the external force and torque estimates to interact each of the sources described in the previous section. For each interaction, an algorithm will be presented to convert the force and torque measurement into a desired position and velocity in x and y that are sent to a position controller for the AR.Drone.

7.1. Wall

7.1.1. Wall detection with a Support Vector Machine

Now that we have a set of features (external forces) that change depending on the quadrotor's proximity to the wall (close or far), we can train a machine learning algorithm to distinguish between these two classes. We build on extensive work in the field of tactile terrain identification where researchers have used machine learning to classify terrain types using experimental data rather than relying on a physics-based model of contact dynamics [19–22]. A comparison by [23] showed that an SVM based algorithm is particularly well suited for this task.

We chose to use a Support Vector Machine (SVM) for wall detection for two main reasons: First, it is a non-parametric classifier, meaning it does not require specialized knowledge of the underlying equations that describe how force and torque vary with position relative to the wall. Rather, the decision boundary is defined in terms of experimental data assembled into a feature vector denoted by $\boldsymbol{\xi}$. This is especially useful since, to our knowledge, a relationship describing how forces and torques vary in close proximity to the wall does not exist in the literature. Second, it has relatively few parameters yet is capable of capturing complicated non-linear decision boundaries.

Now we give a brief overview of the SVM we used in this study. We encourage the interested reader to see [35] for more details.

7.1.2. Support Vector Machine (SVM)

The SVM learns a decision boundary between the two classes as a function of feature vectors obtained during the training phase. Any newly measured feature vector can then be associated with one of the two classes. SVMs are kernel methods, which means they make use of a kernel function $\mathcal{K}(\boldsymbol{\xi})$ to describe the similarity of two feature vectors. We use the Radial Basis Function (RBF),

$$\mathcal{K}_i(\boldsymbol{\xi}) = \exp\left(-\frac{\|\bar{\boldsymbol{\xi}} - \bar{\boldsymbol{\xi}}_{sv_i}\|}{2\sigma}\right), \tag{69}$$

which is capable of capturing non-linear decision surfaces in the original features space. The parameter σ describes the width of the kernel function and is a design parameter, $\bar{\boldsymbol{\xi}}$ denotes the feature vector after whitening (based on mean and variance obtained from the training data set), and $\boldsymbol{\xi}_{sv,i}$ are the so-called support vectors obtained from the training. The training consists in solving an optimization problem, which results in: a subset of relevant feature vectors, the support vectors $\boldsymbol{\xi}_{sv,i}$, a set of weights, α_i, and a bias b (refer to [35] for details). These define the resulting decision function

$$C = \text{sign}\left\{\left(\sum_{i=1}^{n} \mathcal{K}_i(\boldsymbol{\xi})\alpha_i\right) + b\right\}, \tag{70}$$

which decides for a given feature vector $\boldsymbol{\xi}$ which class C it belongs to.

7.2. Fan

Our aim in this section is to design a simple controller to enable the quadrotor to follow the fan.

7.2.1. Admittance controller

A support vector machine can be useful in complex situations where it is difficult to manually define a decision boundary or relation between the force field and a discrete set of desired actions. Another method, called the admittance controller, has been used in [14,15] to react to external forces in a continuous manner. This changes the desired state of the system based on external forces and torques. The τ_z^e profile shown in Fig. 10 has a specific anti-symmetric profile that we can exploit to enable the quadrotor to track a moving fan. The sign of the torque tells us which side of the fan the quadrotor is on, and its magnitude increases to a maximum roughly 0.5 m from the axis of the fan. This leads us to a proportional, admittance controller:

if $\tau_z^e > \epsilon_\tau$ **then**
  $\dot{y} = k_p \tau_z^e$
end if

that can move the quadrotor in front of the fan without any special wind sensors. Here, the detection step is a threshold on τ_z^e of 0.07 Nm, and the action is the proportional controller with $k_p = 1$.

7.3. Downwash

7.3.1. Avoidance strategy

Now we present the downwash avoidance strategy. Fig. 11 shows that the downwash force is restricted to a relatively small area (about 0.5 m around the upper quadrotor) and almost doubles in magnitude over just 0.3 m. This means the opportunity to detect downwash pre-emptively and choose the correct action is very small, so rather than try to avoid it or localize relative to the source, we choose a strategy to detect it using the strong f_z^e and fly through the affected area as quickly as possible. We found this works well in practice. Using torque to estimate which side of the downwash stream the lower quadrotor was on proved difficult, however. This was especially the case when the quadrotor approached the downwash head on because measurements of torque became less indicative of which side the quadrotor was on and quick, decisive action was required. Referring to Fig. 11, measuring $f_z^e < -0.5$ N means there is almost 90% probability the lower quadrotor is within 0.5 m of the upper quadrotor, so that traveling another 1.0 m in a straight line will bring the lower quadrotor across the downwash regardless of its position relative to the upper quadrotor. In other words, we command the lower quadrotor to 'jump' 1 m along its desired path at 1 m/s until it has passed through the downwash stream, at which point normal path following resumes.

Algorithm
if $f_z^e > \epsilon_{f_z^e}$ **then**
  $x_d += 1$
  $\dot{x}_d = 1$
end if

8. Interaction experiments

8.1. Wall detection using a Support Vector Machine

The kernel width, σ, as well as the distance for a point to be considered 'close to the wall' are chosen by random search to optimize classification accuracy on a separate set of training data.

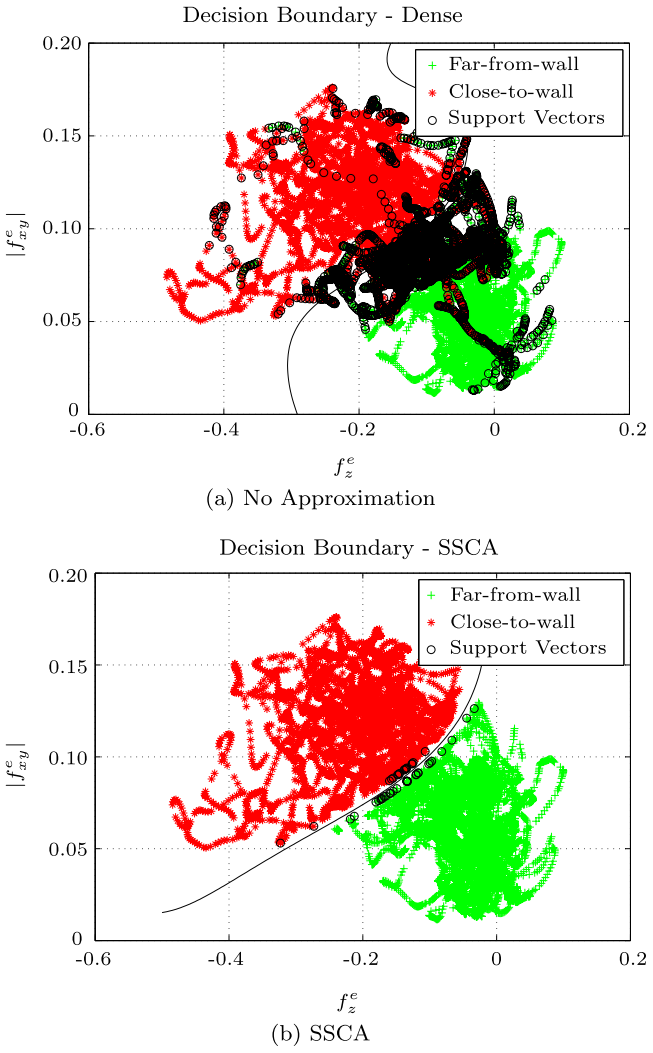


Fig. 12. SVM Decision Boundary for Wall Detection using the Smoothed, Separable Class Approximation (SSCA). The SVM decision boundary for wall detection with and without the SSCA. The force measurements form two different clusters depending on proximity to the wall separated by a decision boundary ($\|f_{x,y}^e\|$ and f_z^e shown). Some overlap between the classes is inevitable since the force varies continuously between the two cases. This overlap results in many support vectors since support vectors are chosen as points close to other points with the opposite class - a criterion very well satisfied by noise. By re-classifying and removing points close to the decision boundary, we can remove most of the redundant support vectors resulting a light weight solution with practically indistinguishable performance.

The hyper-parameter σ was chosen to maximize 10-fold cross validation choosing the best value from 10 random seeds. We found the accuracy was not very sensitive to this parameter.

We used the SVM implementation in the Machine Learning Library of Version 8.1 of Matlab's Statistical toolbox and the Smooth Separable Class Approximation to reduce the number of support vectors and over-fitting, for which we refer the reader to [36]. Fig. 12 shows an example of the original and approximated decision boundary, and how essentially the same decision boundary is obtained as a function of far fewer support vectors (see Table 1).

8.1.1. Blind wall mapping

The method above enables a quadrotor to 'blindly' detect when it is in close proximity to walls. Sampling many points arranged in a grid enables us to produce a map of the boundary of

Table 1
Comparison between a dense SVM and SSCA for wall detection. Classification accuracy over 25 flights. The approximation largely preserves test accuracy of the SVM, and by inspection of Fig. 12, the change in decision boundary lies below the resolution of the force estimator and so should not have a significant impact on the real system. Test data was from a 25% holdout sample.

	Accuracy	
	Wall	Free
Dense SVM	92.6	84.2
SSCA SVM	94.8	79.3

Table 2
Classification accuracy over 25 flights at varied yaw. The proposed classification method is able to achieve above 70% accuracy in all cases. This data is meant to show that the classification accuracy is comparable when the quadrotor approaches the wall from non-zero yaw.

Yaw	Accuracy	
	Free	Wall
0	73.6	92.3
$\pi/2$	78.7	81.8
$\pi/4$	76.7	100

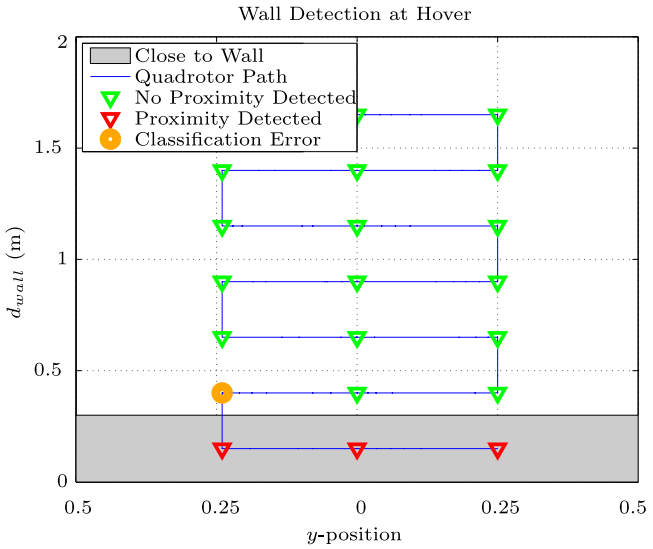


Fig. 13. Wall proximity detection results for a quadrotor navigating a series of waypoints. The vehicle hovers at each waypoint (indicated by markers) for 5 s. Waypoints are labeled with the result of the binned classification. The shaded gray area represents the area considered close-to-the-wall. Classification errors are most common at the boundary of the 'close to wall' area. As a result, the classification can be used for wall detection for a quadrotor slowly exploring the space by hovering at discrete waypoints.

a room without directly measuring distance to the wall. Sampling at hover for 5 s also allows us to bin measurements by location. We can then classify many measurements of force and consider the fraction of measurements classified as close-to-the wall. If this fraction is greater than a threshold, we classify the sampling point as close-to-the-wall. For our experiments, we chose this fraction to be 0.3 by maximizing the classification accuracy of sample points and weighting points close-to and far-from the wall equally.

Fig. 13 illustrates how this mapping might work. Table 2 shows the binned accuracy of the classification scheme over 25 flights. Overall, it shows that the algorithm is capable of detecting proximity with consistently higher accuracy than free flight. This is likely due to the fact that the forces caused by the wall effect are so small that they can be felt during regular flight, and are most exaggerated near the wall. The result is that the quadrotor sometimes experiences random disturbances that are similar to those characteristic of being close to the wall in free flight, but almost always measures these forces close to the wall. This has the practical advantage of being most accurate when the consequences of crashing into the wall are highest.

8.2. Fan tracking

The experimental setup for our fan tracking experiment involved moving a 0.3 m house fan approximately 2.3 m from the quadrotor. Fig. 14(a) shows the quadrotor finding the center of the fan after it is turned on (at about 2 s). Fig. 14(b) shows the quadrotor tracking the fan as it is moved on a cart moved by hand. For this purpose, it is not necessary that our estimator give the true aerodynamic torque, but rather that measurements are repeatable as a function of position relative to the fan. This is the case for all of the interactions we have shown so far.

For the step response in Fig. 14(a) the quadrotor was commanded to start at $y = 0.8$ m and then the fan was turned on to allow the admittance controller to guide the quadrotor to the center of the fan. The fan is turned on just after 2 s, and the vehicle converges to within 0.2 m. This is the region where the magnitude of the torque is less than 0.1 Nm, which is the threshold used for the admittance controller. We thus achieve our objective of tracking the center of the fan.

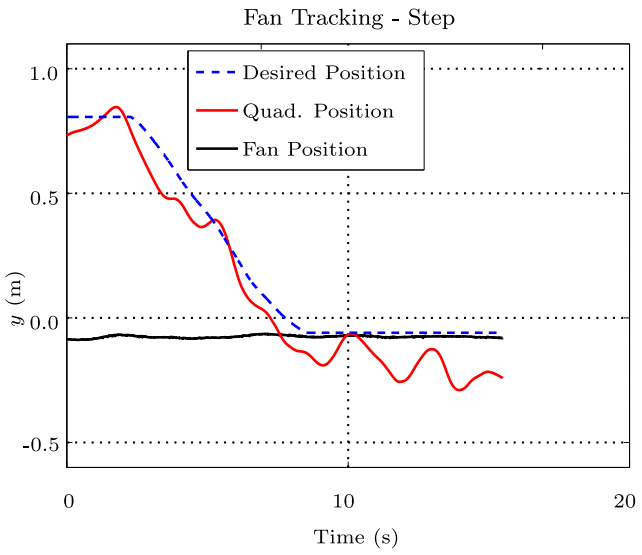
Fig. 14(b) shows the quadrotor tracking a moving fan. This experiment shows the quadrotor lagging about 0.5 m from the center of the fan. Referring to Fig. 10, this corresponds to the maximum value of τ_z^e, where the admittance controller would be most attracted to the center of the fan. This is sufficient to meet our objective of having the quadrotor track the fan since we have already shown that the admittance controller can guide the quadrotor to the fan up to 0.8 m from the fan axis.

8.3. Downwash avoidance

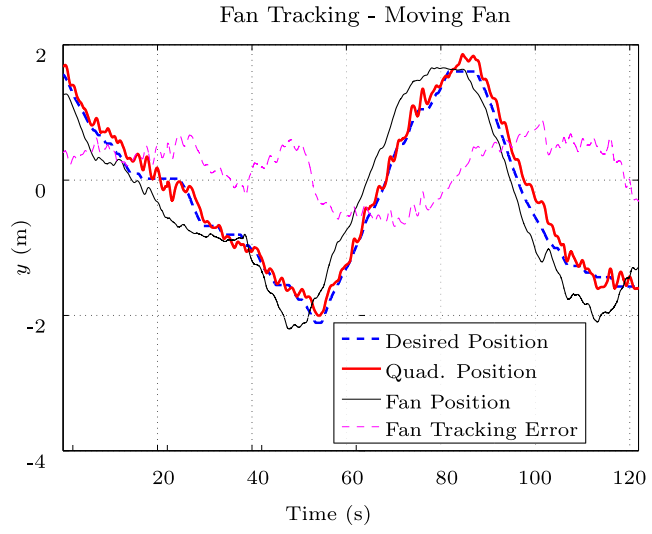
We tested our downwash avoidance strategy in the worst case scenario, namely: when the quadrotor experiences the maximum duration and intensity of downwash forces as it flies through the center of the downwash region. Fig. 15 shows the proposed algorithm acting when the lower quadrotor is set on a path moving at just 0.25 m/s and passing directly below the upper quadrotor. The dotted line shows that the quadrotor is normally incapable of passing through the downwash region under these conditions. It is pushed down to the floor – losing 1.0 m of altitude – and the experiment is terminated. The solid line shows the quadrotor enabled with downwash avoidance as it detects and passes through the downwash, loosing just 0.30 m of altitude. Note that, by comparison, the quadrotor without downwash avoidance saturated its control inputs many times while in the downwash. This suggests that, due to the physical limitations of the platform, flying in the downwash stream would be very challenging for any controller, even one equipped with special knowledge of downwash (such as the one proposed in [7]). We have presented a downwash avoidance that works when payload restrictions and/or lack of modular hardware capabilities prevent the addition of specialized sensors, as is the case with most modestly-priced commercial quadrotors.

9. Conclusion

In conclusion, this paper has presented an algorithm to estimate external forces and torques acting on a quadrotor. We have shown that our algorithm can adequately handle noisy measurements, and serve as a basis for reacting to aerodynamic disturbances without relying on specialized knowledge of the



(a) Step Input



(b) Fan in Motion

Fig. 14. Step response of the admittance controller (a), and the quadrotor tracking a moving fan (b). In both cases, the fan is about 2.3 m away. For the step response, the quadrotor was commanded to start at $y = 0.8$ m and the fan was turned after 2 s. For the second experiment, the fan is moved by hand from one end of the room to the other. For both cases, the fan is pointed along the x-axis.

underlying dynamics of the disturbance, the aerodynamic properties of the quadrotor, or on specialized sensors for measuring wind speed. In addition, we have demonstrated through a set of experiments how the force estimate may be used in conjunction with machine learning or an admittance controller to enable a quadrotor to respond to a variety of tasks including wall detection, holding position relative to a wind source, and avoiding downwash. We encourage the reader to view a video of our experiments available at: https://www.youtube.com/watch?v=x0RL7Jh6F9s.

In general, our algorithm works well in cases where the external forces and torques acting on the quadrotor are large relative to other random disturbances acting on the vehicle. This makes tasks such as tracking a fan and avoiding downwash relatively

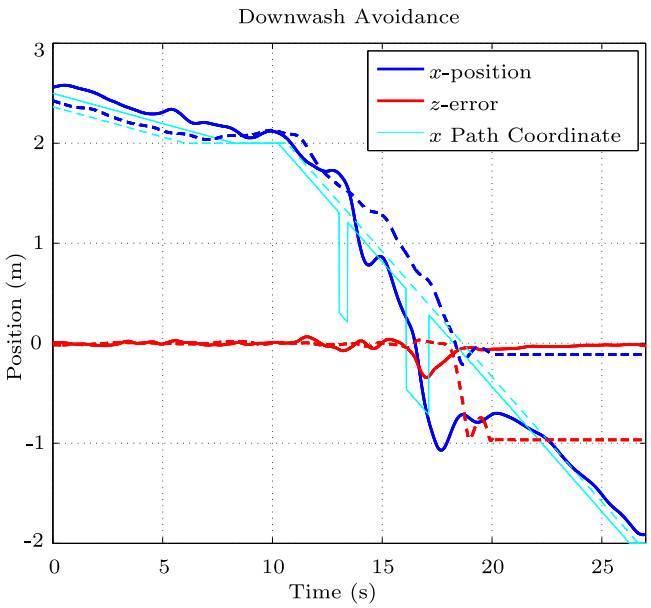


Fig. 15. Force-based downwash detection and avoidance. The colored lines show the x, z, and desired x position of the quadrotor. The solid lines show the quadrotor passing below the other with the downwash avoidance strategy enabled, the dotted lines show these values without it. The upper quadrotor is 2 m above the ground and 1 m above the lower quadrotor. A z-error of -1.0 m means the lower quadrotor is in contact with the ground, which happens at 20 s for the quadrotor without downwash avoidance enabled. The jumps in the desired trajectory correspond to the avoidance strategy being triggered. The algorithm with downwash avoidance enabled recovers from a false trigger and guides the lower quadrotor safely to the other side of the downwash. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

easy compared to detecting proximity to the wall, which is far more sensitive to random disturbances and calibration accuracy.

As future work, we hope to explore the use of external force estimates to localize within a measurable force field. We would also like to examine using this approach as a basis for physically interactive human–robot tasks, and believe our approach may be useful for other types of control and interaction techniques available in the literature.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Canada Foundation for Innovation (CFI) including provincial matching.

References

[1] A. Albers, S. Trautmann, T. Howard, Trong N., M. Frietsch, C. Sauter, Semi-autonomous flying robot for physical interaction with environment, in: Proc. of the Conf. on Robotics Automation and Mechatronics, RAM, 2010, pp. 441–446.

[2] G. Darivianakis, K. Alexis, M. Burri, R. Siegwart, Hybrid predictive control for aerial robotic physical interaction towards inspection operations, in: Proc. of the Intl. Conference on Robotics and Automation, ICRA, 2014.

[3] L. Marconi, R. Naldi, Control of aerial robots: Hybrid force and position feedback for a ducted fan, Control Syst. Mag. 32 (4) (2012) 43–65.

[4] B. Yüksel, C. Secchi, H. Bülthoff, A. Franchi, Reshaping the physical properties of a quadrotor through IDA-PBC and its application to aerial physical interaction, in: Proc. of the Intl. Conf. on Robotics and Automation, ICRA, 2014, pp. 6258–6265.
[5] N. Hai-Nguyen, L. Dongjun, Proc. of the Intelligent Robots and Systems, IROS, 2013, pp. 3458–3464.
[6] N. Michael, D. Mellinger, Q. Lindsey, V. Kumar, The GRASP multiple micro-UAV testbed, Robot. Autom. Mag. 17 (3) (2010) 56–65.
[7] N. Sydney, B. Smyth, D. Paley, Dynamic control of autonomous quadrotor flight in an estimated wind field, in: Proc of the Conf. on Decision and Controls, CDC, 2013, pp. 3609–3616.
[8] D. Yeo, N. Sydney, D. Paley, Onboard flow sensing for downwash detection and avoidance with a small quadrotor helicopter, in: Proc. of the AIAA Guidance, Navigation and Control Conference, 2015.
[9] C. D. McKinnon, A. P. Schoellig, Unscented external force and torque estimation for quadrotors, in: Proc. of the Intl. Conf. on Intelligent Robots and Systems, IROS, 2016, PP. 5651–5657.
[10] B. Arain, F. Kendoul, Real-time wind speed estimation and compensation for improved flight, Trans. Aerosp. Electron. Syst. 50 (2) (2014) 1599–1606.
[11] S. Waslander, C. Wang, Wind disturbance estimation and rejection for quadrotor position control, in: Proc. of the AIAA Infotech@Aerospace Conference, 2009.
[12] F. Schiano, J. Alonso-Mora, K. Rudin, P. Beardsley, R. Siegwart, B. Siciliano, Towards estimation and correction of wind effects on a quadrotor UAV, in: Proc. of the Intl. Micro Air Vehicle Conference and Competition, IMAV, 2014.
[13] C. Powers, D. Mellinger, A. Kushleyev, B. Kothmann, V. Kumar, Influence of aerodynamics and proximity effects in quadrotor flight, in: Experimental Robotics, in: Springer Tracts in Advanced Robotics, vol. 88, 2012, pp. 289–302.
[14] F. Augugliaro, R. D'Andrea, Admittance control for physical human-quadrocopter interaction, in: Proc. of the European Control Conference, ECC, 2013, pp. 1805–1810.
[15] T. Tomic, S. Haddadin, A unified framework for external wrench estimation, interaction control and collision reflexes for flying robots, in: Proc of the Intl. Conf. on Intelligent Robots and Systems, IROS, 2014, pp. 4197–4204.
[16] G. Bätz, B. Weber, M. Scheint, D. Wollherr, M. Buss, Dynamic contact force/torque observer: Sensor fusion for improved interaction control, Int. J. Robot. Res. 32 (4) (2013) 446–457.
[17] P. Bristeau, E. Dorveaux, D. Vissière, N. Petit, Hardware and software architecture for state estimation on an experimental low-cost small-scaled helicopter, Control Eng. Pract. 18 (7) (2010) 733–746.
[18] B. Yuksel, C. Secchi, H.H. Bulthoff, A. Franchi, A nonlinear force observer for quadrotors and application to physical interactive tasks, in: Proc. of the Intl. Conference on Advanced Intelligent Mechatronics, AIM, 2014, pp. 433–440. http://dx.doi.org/10.1109/AIM.2014.6878116.
[19] P. Giguere, G. Dudek, Surface identification using simple contact dynamics for mobile robots, in: Proc of the Intl. Conf. on Robotics and Automation, ICRA, 2009, pp. 3301–3306.
[20] P. Dallaire, D. Emond, P. Giguere, B. Chaib-Draa, Artificial tactile perception for surface identification using a triple axis accelerometer probe, in: Proc. of Intl. Symp. on Robotic and Sensors Environments, ROSE, 2011, pp. 101–106.
[21] C. Weiss, H. Frohlich, A. Zell, Vibration-based terrain classification using support vector machines, in: Proc of the Intl. Conf. on Intelligent Robots and Systems, IROS, 2006, pp. 4429–4434.
[22] P. Giguere, G. Dudek, S. Saunderson, C. Prahacs, Environment identification for a running robot using inertial and actuator cues, in: Proc. of the Robotics and Science and Systems Conference, RSS, Philadelphia, USA, 2006, pp. 271–278.
[23] C. Weiss, N. Fechner, M. Stark, A. Zell, Comparison of different approaches to vibration-based terrain classification, in: Proc. of the European Conference on Mobile Robots, EMCR, 2007.
[24] M. Hehn, Quadrocopter trajectory generation and control, in: IFAC World Congress, 2011, pp. 1485–1491.
[25] J. Kelly, On Temporal and Spatial Calibration for High Accuracy Visual-Inertial Motion Estimation (Ph.D. thesis), University of Southern California, 2011.
[26] J. Kelly, G. Sukhatme, Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration, Int. J. Robot. Res. 30 (1) (2011) 56–79.
[27] G. Hoffmann, S. Waslander, C. Tomlin, Quadrotor helicopter flight dynamics and control: Theory and experiment, in: Proc. of the AIAA Guidance, Navigation, and Control Conference, Vol. 2, 2007.
[28] T.D. Barfoot, State Estimation for Robotics, Cambridge University Press, 2017.
[29] J. Crassidis, F. Landis Markley, Unscented filtering for spacecraft attitude estimation, J. Guid. Control Dyn. 26 (4) (2003) 536–542.
[30] L. Mellinger, M. Shomin, V. Kumar, Design, modeling, estimation and control for aerial grasping and manipulation, in: Proc. of the Intelligent Robots and Systems Conference, IROS, 2011, pp. 2668–2673.
[31] T. Barfoot, State Estimation for Aerospace Vehicles, University of Toronto, 2014, University Lecture.
[32] ardrone_autonomy package, ver. 1.3.1, 2014, Available at www.ros.org. (Accessed).
[33] J. Pestana Puerta, J. Luis Sánchez López, I. Mellado Bataller, C. Fu, P. Campoy Cervera, AR.Drone identification and navigation control at CVG-UPM, Journadas Nac. de Autom. (2012).
[34] P. Bristeau, Fr. Callou, D. Vissiere, N. Petit, The navigation and control technology inside the ar. drone micro uav, Int. Fed. Automat. Control 44 (1) (2011) 1477–1484.
[35] C. Bishop, et al., Pattern recognition and machine learning, Vol. 1, springer, New York, 2006.
[36] D. Geebelen, J. Suykens, J. Vandewalle, Reducing the number of support vectors of SVM classifiers using the smoothed separable case approximation, Trans. Neural Netw. Learn. Syst. 23 (4) (2012) 682–688.

Chris McKinnon is currently pursuing his PhD at the Dynamic Systems Lab at the University of Toronto. He completed his B.A.Sc. in Engineering Science at the University of Toronto in 2013 and his M.A.Sc. at the University of Toronto in 2015. His current research focus is long-term, safe learning control.

Angela Schoellig is an Assistant Professor at the University of Toronto Institute for Aerospace Studies and an Associate Director of the Centre for Aerial Robotics Research and Education at the University of Toronto. She conducts research at the interface of robotics, controls, and machine learning. Her goal is to enhance the performance, safety, and autonomy of robots by enabling them to learn from past experiments and from each other. She is a recipient of a Sloan Research Fellowship, a Ministry of Research, Innovation & Science Early Researcher Award, and a Connaught New Researcher Award. Her team won the 2018 GM/SAE AutoDrive Challenge, a North-America-wide self-driving competition. She is one of MIT Technology Review's Innovators Under 35 (2017), one of Robohub's "25 women in robotics you need to know about (2013)," winner of MIT's 2015 Enabling Society Tech Competition, a 2015 finalist in Dubai's $1 million "Drones for Good" competition, and the youngest member of the 2014 Science Leadership Program, which promotes outstanding scientists in Canada. Angela received her Ph.D. from ETH Zurich, Switzerland, in 2013, and holds both an M.Sc. in Engineering Science and Mechanics from the Georgia Institute of Technology (2007) and a Masters degree in Engineering Cybernetics from the University of Stuttgart, Germany (2008). Her Ph.D. was awarded the ETH Medal and the 2013 Dimitris N. Chorafas Foundation Award (one of 35 worldwide). Further information may be found at http://www.schoellig.name.