International Journal of Micro Air Vehicles

Original Research

International Journal of Micro Air Vehicles
Volume 11: 1–14

DOI: 10.1177/1756829319833685
journals.sagepub.com/home/mav
SAGE

In-flight model parameter and state estimation using gradient descent for high-speed flight

S Li, C De Wagter, CC de Visser, QP Chu and GCHE de Croon

Abstract

High-speed flight in GPS-denied environments is currently an important frontier in the research on autonomous flight of micro air vehicles. Autonomous drone races stimulate the advances in this area by representing a very challenging case with tight turns, texture-less floors, and dynamic spectators around the track. These properties hamper the use of standard visual odometry approaches and imply that the micro air vehicles will have to bridge considerable time intervals without position feedback. To this end, we propose an approach to trajectory estimation for drone racing that is computationally efficient and yet able to accurately estimate a micro air vehicle's state (including biases) and parameters based on sparse, noisy observations of racing gates. The key concept of the approach is to optimize unknown and difficult-to-observe state variables so that the observations of the racing gates best fit with the known control inputs, estimated attitudes, and the quadrotor dynamics and aerodynamics during a time window. It is shown that a gradient-descent implementation of the proposed approach converges ∼4 times quicker to (approximately) correct bias values than a state-of-the-art 15-state extended Kalman filter. Moreover, it reaches a higher accuracy, as the predicted end-point of an open-loop turn is on average only ∼20 cm away from the actual end-point, while the extended Kalman filter and the gradient descent method with kinematic model only reach an accuracy of ∼50 cm. Although the approach is applied here to drone racing, it generalizes to other settings in which a micro air vehicle may only have sparse access to velocity and/or position measurements.

Keywords

Autonomous drone race, quadrotor modeling, bias estimation, gradient descent

Received 20 December 2017; accepted 19 January 2019

Introduction

Quadrotors have received considerable attention in recent years, thanks to their mechanical simplicity and good maneuverability combined with hover properties. They have offered new possibilities in a variety of fields like aerial photography, inspection and even transportation. With recent advances in on-board computation and sensor technology, aggressive maneuvering has come within reach of many applications. To further stimulate aggressive and fast flight, autonomous drone racing is gaining interest. The first ever autonomous drone race was held by the International Conference on Intelligent Robots and Systems (IROS) in 2016.[1] A track consisting of gates had to be flown autonomously in a pre-specified order. The robot had to achieve this as fast as possible, while only relying on onboard sensors and processing. Figure 1 illustrates the setup of the 2016 indoor track.

Autonomous indoor drone racing brings many new challenges to the fields of quadrotor navigation and control. One initial challenge is the navigation without any external positioning system like VICON, Optitrack or GPS. Typical approaches to this problem make use of on-board cameras and use Visual Inertial Odometry

Micro Air Vehicle Lab, Delft University of Technology, Delft, the Netherlands

Corresponding author:
GCHE de Croon, Micro Air Vehicle Lab – TUDelft, Kluyverweg 1, 2629HS Delft, the Netherlands.
Email: g.c.h.e.decroon@tudelft.nl

to integrate position. This type of algorithms rely on integrating inertial information, tracking visual features over several frames and solving an optimization problem to retrieve the most likely solution. In autonomous drone racing, on top of this position estimation algorithm, gate detection is often needed when the position of gates is not precisely known, or when gates contain moving parts—as is the case in the IROS competitions. With the limited computational resources of small indoor drones, to achieve the fast speeds needed in drone racing, this paper proposes a navigation solution based solely on gate detection, augmented with inertial measurements and an aerodynamic model. To cope with the sometimes sparse and noisy non-Gaussian visual observations, we formulate the navigation solution as an optimization problem. We then solve it using a gradient descent method. The resulting method provides online estimation of the quadrotor position, velocity and inertial biases using less computational resources than traditional Visual Inertial Odometry. The proposed approach also estimates aerodynamic properties of the quadcopter—which become increasingly important in the case of fast aggressive control. Finally, the approach scales favorably with increasing flight speeds as it keeps performing well even with very few position updates. As a comparison, we use the Kalman filter, which is currently still the default choice for navigation. Since the extended Kalman filter (EKF) is significantly less computationally complex than the unscented Kalman filter (UKF),[2] in this paper we select the EKF as a benchmark. We compare the results with EKF, which is shown to be much more sensitive to visual outliers or other non-Gaussian effects.

In the section Related work, an overview of studies on aerodynamics modeling and state estimation methods is given. The section Quadrotor model will describe the quadrotor model parameters that will be solved. The section State estimation proposes two different approaches for the visual state estimation. First a classic 15-state EKF is developed as benchmark. Then the novel FMINCON-based gradient descent optimization method is proposed to solve the model parameters and states. In the section Experiment setup and result, both algorithms are compared on flight test data and Conclusion summarizes the conclusions.

Related work

Several researchers have already proposed aerodynamics models for quadrotors.[3–7] The main object of their studies is to derive a nonlinear quadrotor aerodynamics model to improve the control performance by compensating for the nonlinear terms. In some studies, a detailed aerodynamic model is analyzed through theory and fitted by experimental data.[3] Simplified aerodynamic models are also established from experiments.[5,6] It should be noted that their models are all obtained off-line using external measurements, such as GPS, VICON and thrust test beds. Aerodynamic models can also be combined with on-board measurements, for instance from computer vision,[8] in order to better estimate the velocity of the drone on-line. In this article, we employ a simplified aerodynamic model in the trajectory estimation exactly for this purpose.

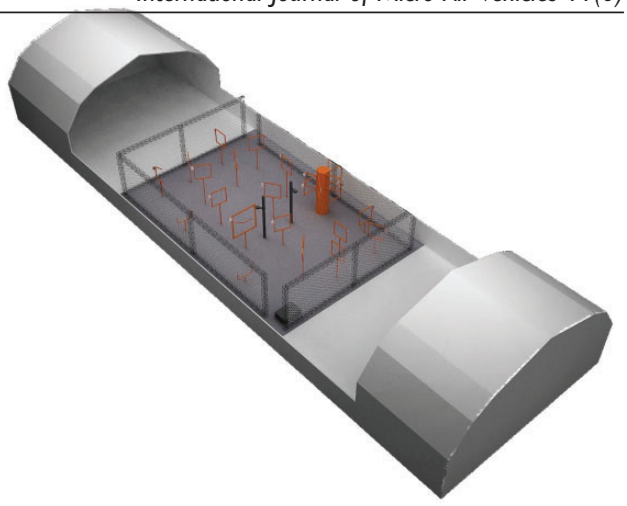

Figure 1. The map of the IROS 2016 drone race. In this drone race, the UAVs have to fly through orange gates in a pre-specified order as fast as possible.

Quadrotor control heavily relies on attitude estimation from an attitude and heading reference system (AHRS). This system is typically based on inertial sensors (accelerometers and gyroscopes), but also relies on orientation sensors (magnetometer) and/or positioning sensors (GPS, VICON) to estimate inertial sensor biases and compensating for long term drift. Sensor biases become increasingly important as the drone will have to fly longer or temporarily perform feedforward control maneuvers in the absence of sensor measurements. Hence, for drone racing, it is important to estimate them accurately. Here we briefly discuss the sensors and then the filtering employed in estimating both attitude and position or velocity on micro air vehicles (MAVs).

Most systems intended for outdoor environments utilize the magnetometer and GPS-measurements.[9–13] The indoor equivalent is the use of a motion tracking system such as VICON or Optitrack.[14] In many applications—like autonomous drone racing, it is required to have accurate state estimation without the help of external systems. The necessary position or velocity measurements can be obtained from multiple sensors. One early option is to use laser scanners.[15,16] But a laser scanner contains sensitive optics and mirrors, which are susceptible to shock and vibration problems.[17] Another choice for on-board navigation is RGB-D

devices.[18–20] The main drawback of these RGB-D devices is that their maximum depth perception range is limited to a few meters.[21,22] This is why light-weight and inexpensive on-board cameras which are more robust to vibration and shock, have attracted interest of researchers for the navigation of drones. Generally, visual odometry (VO) algorithms[23] using a stereo camera or monocular camera are used for estimation of the MAV's translation and rotation between frames.[24–28] However, generic visual odometry approaches necessitate detecting features, matching corresponding features and estimating motions, which leads to a heavy demand for on-board computational resources and low-frequency estimation. In the meantime, aggressive maneuvers may introduce blur into generic visual odometry and seriously affect the accuracy of estimation. Moreover, in complicated environments like drone racing, dynamic spectators may also interfere visual odometry. Less generic but computationally efficient methods are employed in some specific environments, for instance, using detection of known visual markers to determine position.[29,30] However, these methods cannot cope with other generic environments.

Concerning filtering, with white-Gaussian position measurement, Kalman filter and its variants are widely used. It is well-known that nonlinearities in the state update or observation equations can be handled by an EKF[11,12,22,31] and that heavy nonlinearities are often handled better by a UKF.[30,32,33] Also, there are factor graph-based smoothing methods which can handle nonlinearity and allows multi-rate, asynchronous, and possibly delayed measurements, which have similar performance with an EKF.[34,35] We hypothesize that when these measurements get sparser, and their noise distribution moves further away from the Gaussian distribution, it will be better to estimate the attitude, heading, and trajectory in general as an optimization problem that uses more data at a time. In particular, we want to optimize the trajectory and parameters such as sensor biases, given a specified time-window with the corresponding sensor measurements, control inputs, and knowledge of the aerodynamic model. Our approach will be explained below, starting with our dynamic quadrotor model.

Quadrotor model

Dynamic model of quadrotor

Before deriving the dynamic model for quadrotor, two reference frames are introduced (Figure 4).

- Earth frame E. The origin of the local tangent earth frame is on the ground, the x-axis x^E points to north, the y-axis y^E points east and the z-axis z^E points down.
- Body frame B. The origin of the body frame is at the center of mass. Its x-axis x^B is in the symmetry plane of the drone and points forward. Its z-axis z^B also lies in the symmetry plane and points downward. The y-axis y^B is directed to the right, perpendicular to the symmetry plane.

The relative relation between two frames can be expressed by three successive rotations along three axes. In this paper, we use z–y–x sequence to rotate one frame to the other. The corresponding angle of rotation is defined by ϕ_E^B, θ_E^B, and ψ_E^B which are also called Euler angles. Given the Euler angles between the two frames, the rotation matrix between two frames can be expressed by

$$\mathfrak{R}_E^B = \begin{bmatrix} C_\theta C_\psi & C_\theta S_\psi & -S_\theta \\ S_\phi S_\theta C_\psi - C_\phi S_\psi & S_\phi S_\theta S_\psi + C_\phi C_\psi & S_\phi C_\theta \\ C_\phi S_\theta C_\psi + S_\phi S_\psi & C_\phi S_\theta S_\psi - S_\phi C_\psi & C_\phi C_\theta \end{bmatrix} \tag{1}$$

where C_X and S_X denote the cosine and sine of X, respectively. The control of the quadrotor is often divided in to two loops which can be independently developed, namely a high level translation loop and a faster low-level attitude loop. For the attitude loop, the inputs of the system are the four rotor speeds and the output consists of the three Euler angles. For the translation loop, the inputs of system are three Euler angles and the output is position. Since quadrotor attitude control is a well-developed topic, in this work we only derive the translational model and have used INDI from Smeur et al.[36] as innerloop.

According to Newton's laws of motion, the motion of quadrotor can be described as

$$m\dot{\mathbf{V}} = m\mathbf{g} + \mathbf{F} \tag{2}$$

where m is mass of the drone, $\mathbf{g}$ is gravity vector and $\mathbf{F}$ is the specific force vector. The change in position can be described by the kinematic equation

$$\dot{\mathbf{X}} = \mathbf{V} \tag{3}$$

In equation (2), the specific force $\mathbf{F}$ can be expressed in Body frame B as

$$\mathbf{F}^B = \begin{bmatrix} F_x^B \\ F_y^B \\ F_z^B \end{bmatrix} \tag{4}$$

Gravity acting on the center of mass and expressed in Earth frame is

$$m\mathbf{g}^E = m\begin{bmatrix}0\\0\\g\end{bmatrix} \tag{5}$$

Combining all forces yields the equations of motion in inertial frame

$$\begin{aligned}\begin{bmatrix}\dot{x}\\\dot{y}\\\dot{z}\end{bmatrix} &= \begin{bmatrix}v_x\\v_y\\v_z\end{bmatrix}\\ \begin{bmatrix}\dot{v}_x\\\dot{v}_y\\\dot{v}_z\end{bmatrix} &= \begin{bmatrix}0\\0\\g\end{bmatrix} + \Re_B^E\begin{bmatrix}a_x^B\\a_y^B\\a_z^B\end{bmatrix}\end{aligned} \tag{6}$$

where

$$\begin{bmatrix}a_x^B\\a_y^B\\a_z^B\end{bmatrix} = \begin{bmatrix}F_x^B\\F_y^B\\F_z^B\end{bmatrix}/m \tag{7}$$

In the system above, we have six states $\mathbf{x} = [x, y, z, v_x, v_y, v_z]^{\mathrm{T}}$ and four inputs $\mathbf{u} = [\phi, \theta, \psi, a_z^B]^{\mathrm{T}}$. In equation (6), the specific force is a nonlinear function of velocity, attitude, angular rates and other factors. It can be expressed as $\mathbf{F} = \mathbf{f}_a(\mathbf{V}, \phi, \theta, \psi, \ldots)$. This system is a multiple input multiple output nonlinear system.

IMU misalignment

Equation (6) reveals that rotation matrix $\Re_E^B$ is an essential part of the model. However, in the real world, many aspects can contribute to attitude estimation errors. A first reason is the misalignment of the IMU (see Figure 2). Assembly inaccuracy can cause the measurements of the IMU to differ from the real states in body frame. Rotor misalignment can also affect the performance of quadrotor. In an ideal quadrotor, the four rotors should be perpendicular to x_BOy_B plane. In practice however, due to installation errors or deformation of rotors or axes, the thrust produced by the rotors is not perfectly perpendicular to the x_BOy_B plane.

Both factors lead to non-zero required attitude during hover: $\phi_E^B \neq 0^\circ$ and $\theta_E^B \neq 0^\circ$. In order to model this misalignment error, we introduce a new frame. The IMU frame I is an orthogonal frame whose three axis coincide with three axes of the accelerometers. The rotation between the IMU frame I and the body frame B can be described by Euler angles $\Phi_I^B = [\phi_I^B, \theta_I^B, \psi_I^B]^{\mathrm{T}}$. The rotation matrix between the IMU frame I and the body frame B is $\Re_I^B(\Phi_I^B)$. Since the IMU frame is physically attached to the body frame, we have the assumption

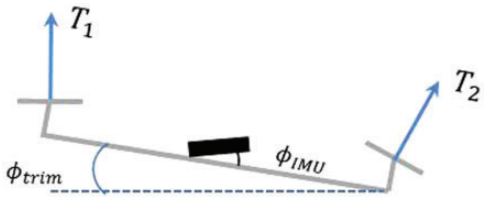

Figure 2. When a quadrotor hovers, usually the average attitude of the quadrotor and reading of the AHRS are not zero. This is caused by the misalignment of both the IMU and the rotors.

$$\begin{cases}\dot{\phi}_I^B(t) = 0\\ \dot{\theta}_I^B(t) = 0\\ \dot{\psi}_I^B(t) = 0\end{cases} \tag{8}$$

Aerodynamic model

There are many factors that can affect the quadrotor's aerodynamics. Some examples are the quadrotor's velocity $\mathbf{V}$, its angle of attack α, the thrust $\mathbf{T}$, the rotor speed ω, the angular velocity $\mathbf{q}$ and so on. Accurate and complete quadrotors models can be complicated and nonlinear.[37,38] Moreover, accurate modeling also requires many more parameters to be estimated and this leads to heavier computations. In the context of autonomous drone racing we opted for a faster approach using a minimal model that covers the most important aerodynamic effects, hereby maximizing the yield for a given computational load. In particular, many drag factors—such as induced drag, translation drag and blade flapping drag—can be approximated as linear functions of body velocity v_x^b and v_y^b with the assumption that wind is still.[6] This results in the following simple lumped parameter model

$$\begin{cases}a_x^B = K_x v_x^B\\ a_y^B = K_y v_y^B\end{cases} \tag{9}$$

where

$$\begin{bmatrix}v_x^B\\v_y^B\end{bmatrix} = \Re_E^B(3;3)\begin{bmatrix}v_x\\v_y\end{bmatrix} \tag{10}$$

a_x^B, a_y^B are the acceleration caused by drag in the body frame. K_x, K_y are first-order drag coefficients in body frame coordinates B and have units 1/s.

AHRS bias model

When positioning information is available, the mainstream approach for estimating attitude is merging information from gyro, accelerometer and the positioning system. For instance, the classic 15-state Kalman filter uses accelerometer and gyro measurements to predict states along with GPS measurement updates. It can provide non-biased optimal attitude by estimating the gyro and accelerometer biases as states.

When no continuous external positioning information is available, like in our experiment, a compromise is to neglect kinematic accelerations in the attitude filter. In this case, the biases of accelerometers cannot be estimated.

In the case of attitude determination with constant sensor biases and small angles, the Kalman gain in the Kalman filter typically converges to an almost constant value. To avoid the computational overhead of computing the Kalman gain, complementary filters can be used with very similar results. The structure of the complementary attitude determination filter implemented in this work can be found in Figure 3. In Figure 3, $\mathbf{\Omega}_m = [p_m, q_m, r_m]$ are the gyro measurements. $\mathbf{a}_m = [a_x^m, a_y^m, a_z^m]$ contains the accelerometer measurements and

$$\mathbf{R}' = \begin{pmatrix} 1 & \tan\theta\sin\phi & \tan\theta\cos\phi \\ 0 & \cos\phi & -\sin\theta \\ 0 & \dfrac{\sin\phi}{\cos\theta} & \dfrac{\cos\phi}{\cos\theta} \end{pmatrix} \tag{11}$$

Figure 3 shows that the gyroscopes are integrated and the accelerometer is used as feedback to determine attitude. The high-frequency vibrations and centripetal forces which are measured by the accelerometers cancel out on the long term when no constant non-zero accelerations are present. On the long term, the resulting attitude estimation therefore converges to

$$\begin{bmatrix} \hat{\phi}_a(t) \\ \hat{\theta}_a(t) \end{bmatrix} = \begin{bmatrix} \arctan\dfrac{-a_x^m}{-a_z^m} \\ \arctan\dfrac{-\cos\hat{\phi}_a(t)a_x^m}{-a_z^m} \end{bmatrix} \tag{12}$$

where a_x^m, a_y^m, and a_z^m are measurements of the accelerometer in three axes.

The gyroscopes measure angular velocity in the three axes of the body frame. Because they are integrated, even small biases cause drift over time, and in this filter the gyro biases $\mathbf{b}_g = [b_p, b_q, b_r]^T$ are accounted for by the k_I/s term in the filter.

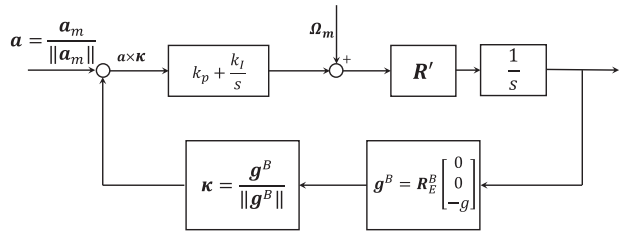

Figure 3. Complementary filter for attitude determination.

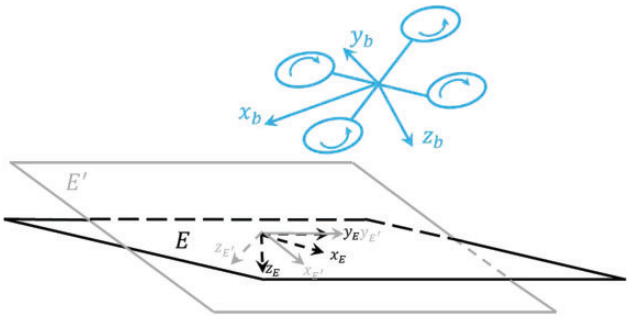

Figure 4. AHRS estimation errors can be represented by an erroneous Earth reference frame E'.

Accelerometers unfortunately also suffer from biases, which is denoted by $\mathbf{b}_a = [b_{a_x}, b_{a_y}, b_{a_z}]^T$, for instance caused by temperature changes. Fortunately, the biases of the accelerometers only change slowly. Everything combined, the AHRS has an erroneous representation of where earth is, which is referred to as coordinate frame E' and is shown in Figure 4. The AHRS attitude is then defined as the rotation between E' and I and is denoted as $\Phi_{E'}^I = [\phi_{E'}^I, \theta_{E'}^I, \psi_{E'}^I]^{\mathrm{T}}$. The corresponding rotation matrix is written as $\Re_{E'}^I(\Phi_{E'}^I)$.

The rotation between the real earth E and E' can be expressed by three Euler angles $\Phi_E^{E'} = [\phi_E^{E'}, \theta_E^{E'}, \psi_E^{E'}]^{\mathrm{T}}$. Based on the assumption that the AHRS error changes slowly, we can assume

$$\begin{cases} \dot{\phi}_E^{E'}(t) \approx 0 \\ \dot{\theta}_E^{E'}(t) \approx 0 \\ \dot{\psi}_E^{E'}(t) \approx 0 \end{cases} \tag{13}$$

With this assumption, on the short term the rotation matrix $\Re_E^{E'}(\Phi_E^{E'})$ is a constant matrix.

Four reference frames have been introduced, namely E, E', I and B. The rotation matrix $\Re_E^B$ in equation (6) can now be expressed as

$$\Re_E^B(\Phi_E^B) = \Re_I^B(\Phi_I^B)\Re_{E'}^I(\Phi_{E'}^I)\Re_E^{E'}(\Phi_E^{E'}) \tag{14}$$

where $\Re_I^B(\Phi_I^B)$ and $\Re_E^{E'}(\Phi_E^{E'})$ are constant matrices and $\Re_{E'}^I(\Phi_{E'}^I)$ represents the attitude as determined by the AHRS.

Full model

Combining equations (6), (9), and (14), we obtain the full model as

$$\dot{\mathbf{x}} = \begin{cases} v_x \\ v_y \\ v_z \\ \begin{bmatrix} 0 \\ 0 \\ g \end{bmatrix} + \Re_B^E \begin{bmatrix} 0 \\ 0 \\ a_z^B \end{bmatrix} + \begin{bmatrix} K_x & 0 & 0 \\ 0 & K_y & 0 \\ 0 & 0 & 0 \end{bmatrix} \Re_E^B \begin{bmatrix} v_x \\ v_y \\ v_z \end{bmatrix} \end{cases}$$

$$\begin{aligned} &\Re_E^B(\Phi_E^B) = \Re_I^B(\Phi_I^B)\Re_{E'}^I(\Phi_{E'}^I)\Re_E^{E'}(\Phi_E^{E'})\Re_B^E = \Re_E^{B\mathrm{T}} \\ &a_z^B = a_z^m - b_{a_z} \end{aligned} \tag{15}$$

The model in equation (15) contains the following parameters, which are assumed to be constant over short periods of time

$$\Theta = \left[K_x, K_y, b_{a_z}, \phi_E^{E'}, \theta_E^{E'}, \psi_E^{E'}, \phi_I^B, \theta_I^B, \psi_I^B\right]^{\mathrm{T}} \tag{16}$$

State estimation

To estimate the states of the model from the section Quadrotor model, two approaches are derived. As a benchmark, an EKF is developed. Secondly, a novel gradient descent based optimization method to estimate the states is proposed.

Vision-based EKF

The attitude determination Kalman filter uses the inertial sensors as inputs to predict the states of the system, then uses different observations to revise the predictions. When the system is linear, observable and the noise is white Gaussian, then it can be mathematically proven that the Kalman filter provides the optimal solution. If the system is nonlinear, it can be linearized at every time step, which is referred to as the EKF. A classic 15-state EKF is implemented as found in Gross's work,[2] the difference being that we use vision measurements instead of GPS as positioning information. The following states are used

$$\begin{aligned} \mathbf{X} &= [x, y, z]^{\mathrm{T}} \\ \mathbf{V} &= [v_x, v_y, v_z]^{\mathrm{T}} \\ \Phi &= [\phi, \theta, \psi]^{\mathrm{T}} \\ \mathbf{b}_a &= [b_{a_x}, b_{a_y}, b_{a_z}]^{\mathrm{T}} \\ \mathbf{b}_g &= [b_p, b_q, b_r]^{\mathrm{T}} \end{aligned} \tag{17}$$

with as inputs

$$\begin{aligned} \Omega_m &= [p^m, q^m, r^m]^{\mathrm{T}} \\ \mathbf{a}_m &= [a_x^m, a_y^m, a_z^m]^{\mathrm{T}} \end{aligned} \tag{18}$$

and as observation

$$\mathbf{y} = \mathbf{h}(\mathbf{x}) = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{19}$$

The process equation is

$$\begin{cases} \dot{\mathbf{X}} = \mathbf{V} \\ \dot{\mathbf{V}} = \mathbf{g} + \Re_B^E(\mathbf{a}_m + \mathbf{b}_a) \\ \dot{\Phi} = \Re'(\Omega_m + \mathbf{b}_g) \\ \dot{\mathbf{b}}_a = 0 \\ \dot{\mathbf{b}}_g = 0 \end{cases} \tag{20}$$

This forms a standard nonlinear system expression

$$\dot{\mathbf{x}}' = \mathbf{f}(\mathbf{x}', \mathbf{u}) \tag{21}$$

where $\mathbf{x}' = [\mathbf{X}, \mathbf{V}, \Phi, \mathbf{b}_a, \mathbf{b}_g]^{\mathrm{T}}$ and

$$\mathbf{f}(\mathbf{x}', \mathbf{u}) = \begin{bmatrix} \mathbf{V} \\ \mathbf{g} + \Re_E^B(\mathbf{a}_m + \mathbf{b}_a) \\ \Re'(\Omega_m + \mathbf{b}_g) \\ 0 \\ 0 \end{bmatrix} \tag{22}$$

The EKF follows five steps:

(1) Predict the states based on equation (20)

$$\hat{\mathbf{X}}_{k|k-1} = \hat{\mathbf{X}}_{k-1} + \mathbf{f}(\hat{\mathbf{X}}_{k-1}, \mathbf{u}_{k-1})\mathrm{T} \tag{23}$$

where T is sampling time.

(2) Linearize and discretize the system

$$\begin{aligned} \mathbf{F}_{k-1} &= \frac{\partial}{\partial \mathbf{x}} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t))|_{\mathbf{x}(t)=\hat{\mathbf{x}}_{k-1}} \\ \Phi_{k|k-1} &\approx \mathbf{I} + \mathbf{F}_{k-1}\mathrm{T} \\ \mathbf{H}_k &= \frac{\partial}{\partial \mathbf{x}} \mathbf{h}(\mathbf{x}(t))|_{\mathbf{x}(t)=\hat{\mathbf{x}}_{k-1}} \end{aligned} \tag{24}$$

(3) Propagate the covariance matrix $\mathbf{P}_{k|k-1}$

$$\mathbf{P}_{k|k-1} = \Phi_{k|k-1}\mathbf{P}_{k-1}\Phi_{k|k-1}^{\mathrm{T}} + \mathbf{Q}_{k-1} \tag{25}$$

where $\mathbf{Q}_{k-1}$ is system noise covariance matrix.

(4) Calculate the Kalman gain and update the prediction.

$$\begin{aligned} \delta\hat{\mathbf{X}}_k &= \mathbf{K}_k\left\{\mathbf{Z}_k - \mathbf{h}[\hat{\mathbf{X}}_{k|k-1}, k]\right\} \\ \mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}}[\mathbf{H}_k\mathbf{P}_{k|k-1}\mathbf{H}_k^{\mathrm{T}} + \mathbf{R}_k]^{-1} \\ \hat{\mathbf{X}}_k &= \hat{\mathbf{X}}_{k|k-1} + \delta\hat{\mathbf{X}}_k \end{aligned} \tag{26}$$

where $\mathbf{R}_k$ is sensor noise covariance matrix.

(5) Update the covariance matrix of the state estimation error

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_{k/k-1}(\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)^{\mathrm{T}} + \mathbf{K}_k\mathbf{R}_k\mathbf{K}_k^{\mathrm{T}} \tag{27}$$

Vision-based gradient descent method

According to the gate detection algorithm we used in IROS 2016 autonomous drone race, the vision-based position used as observation in the Kalman filter (equation (19)) has very non-Gausian noise, which can significantly affect the estimation accuracy of Kalman filters. The vision measurement model will be discussed later. Therefore the state prediction is rewritten as a parameter optimization problem in the form of a trajectory matching problem.

Unlike the Kalman filter which estimates continuously varying states like pitch and roll for any moment in time, the proposed gradient descent using the model from equation (15) in essence estimates corrections on top of attitude estimates provided by an external complementary attitude filter.

Since most model parameters like drag and AHRS error are integrated twice to arrive at position, observing the trajectory over a period of time allows for extremely fine observations of these parameters. For instance, a sub-degree attitude error is hard to identify in noisy raw accelerometer measurements. However integrating the consequence of this small angle error, which causes a percentage of gravity to be erroneously double-integrated in the lateral position after several seconds, becomes very easily observable.

The observed trajectory is obtained from the vision pipeline and expressed as a list of n noisy measurements. The predicted trajectory is based on integrating the model presented in equation (15) using attitude from the AHRS and given a set of model parameters $\hat{\Theta}$. The resulting trajectory becomes

$$\mathbf{F}(\Theta) = \int_0^t \mathbf{f}(\Theta, \mathbf{u}(t), t)\mathrm{d}t = \begin{bmatrix} \hat{x}(\Theta, \mathbf{u}(t), t) \\ \hat{y}(\Theta, \mathbf{u}(t), t) \\ \hat{z}(\Theta, \mathbf{u}(t), t) \\ \hat{v}_x(\Theta, \mathbf{u}(t), t) \\ \hat{v}_y(\Theta, \mathbf{u}(t), t) \\ \hat{v}_z(\Theta, \mathbf{u}(t), t) \end{bmatrix} \tag{28}$$

The error between the predicted integrated trajectory and the vision measurements is found as

$$J(\Theta) = \sum_{i=i}^{n} \left\| \begin{bmatrix} \hat{x}(\Theta, \mathbf{u}(t_i), t_i) \\ \hat{y}(\Theta, \mathbf{u}(t_i), t_i) \\ \hat{z}(\Theta, \mathbf{u}(t_i), t_i) \end{bmatrix} - \begin{bmatrix} x_i^m \\ y_i^m \\ z_i^m \end{bmatrix} \right\| \tag{29}$$

where x_i^m, y_i^m, z_i^m are position measurements obtained from onboard computer vision. Now the state estimation has become a nonlinear parameter optimization problem that finds a set of optimal parameters Θ^* to minimize the value of $J(\Theta)$ which can be expressed as

$$\begin{aligned} &\min_{\Theta} J(\Theta) \\ &s.t. \quad \dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)) \end{aligned} \tag{30}$$

To solve the problem formulated by equation (30), we can apply many types of nonlinear optimization methods to find the optimal parameters Θ^*. In this paper, we propose the gradient descent method, which is iteratively searching for optimal values in negative gradient direction until it finds the minimum point

$$\Theta_{k+1} = \Theta_k + \alpha\nabla J(\Theta_k) \tag{31}$$

where α is learning rate and

$$\nabla J(\Theta_k) = \left[\frac{\partial}{\partial\Theta_1}J(\Theta) \quad \cdots \quad \frac{\partial}{\partial\Theta_n}J(\Theta)\right]^{\mathrm{T}}\Big|_{\Theta=\Theta_k} \tag{32}$$

is the gradient of $J(\Theta)$.

Figure 5 shows an example of the gradient descent approach. The propagation in time of the model from equation (15) for various parameters Θ is compared to the ground-truth measured by a passive external positioning system. The gradient descent starts with an initial guess of Θ_0, and gradually gets the predicted trajectory closer to the real trajectory until an optimal set Θ^* is found. In this example, we directly use Optitrack data as measurements which better illustrate

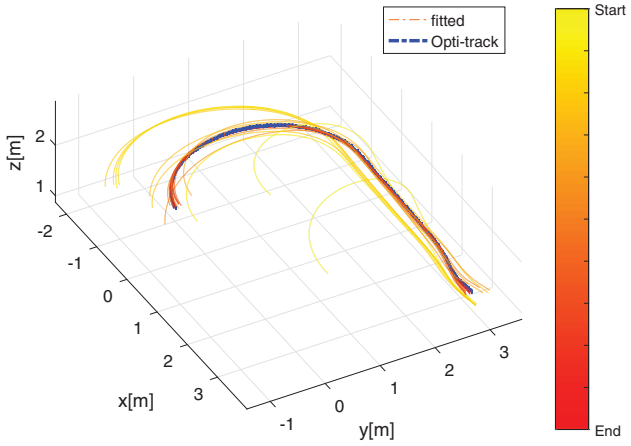

Figure 5. A gradient descent method optimizes a set of parameters Θ to best fit a predicted trajectory through a measured trajectory (blue). During the fitting phase, the gradient descent method converges to the ground-truth trajectory.

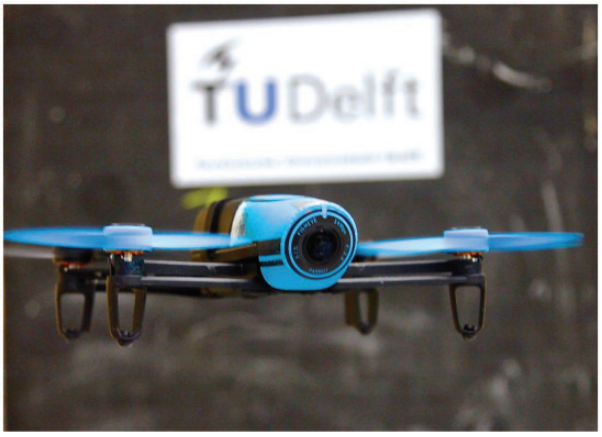

Figure 7. The Parrot Bebop 1 hardware is used as experiment platform. All flight code is replaced with open-source Paparazzi-UAV flight code.

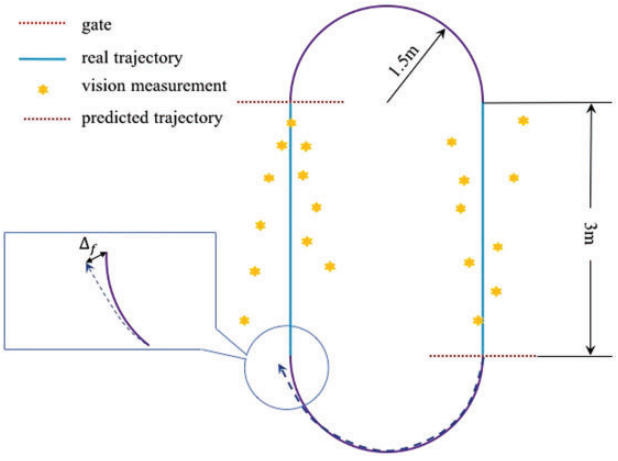

Figure 6. The top view of the experiment track.

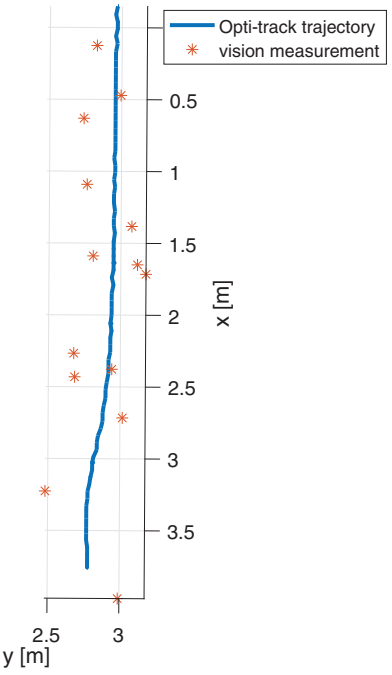

Figure 8. Based on the vision measurement model (equation (33)), simulated vision measurement points (red) are generated around the real trajectory (blue). During the autonomous drone race, only the visual measurement points are available.

how the predicted trajectories converge to the ground-truth trajectory (measured by Optitrack).

Experiment setup and result

Experiment setup

In order to study the performance of state estimation methods, a hippodrome shaped track is used with end circles with radius of 1.5 m and straights of 3 m as shown in Figure 6. Onboard flight data are recorded while flying without computer vision but based on Optitrack position. The data are then analyzed in MATLAB. A Bebop 1 (Figure 7) from Parrot is used as experiment platform. It is equipped with three gyros, three accelerometers, one sonar, one barometer, a front camera and a bottom camera. Only the front camera and IMU are used and the original stock flight-code in the drone is replaced by open-source software from the Paparazzi-UAV project.[39] The AHRS runs on-board and consists of the complementary filter discussed in previous section. The flight time of the test runs is about 100s and the average flight velocity is about

Table 1. Data gathered during the experiment.

Parameter	Symbol	Frequency (Hz)	Source
Acceleration	$\hat{\mathbf{a}}^m$	512	IMU
Angular velocity	$\hat{\mathbf{p}}^m$	512	IMU
Attitude	$\hat{\Phi}^m$	512	AHRS
Position	$\hat{\mathbf{x}}^m$	120	Optitrack
Velocity	$\hat{\mathbf{v}}^m$	120	Optitrack
Altitude	z^m	512	Sonar

1.8m/s, resulting in about 15 circles of the hippodrome. An overview of data gathered is presented in Table 1.

During the IROS 2016 autonomous drone race, we used the bebop 1 onboard camera to detect the gates and provide the position measurements for navigation. In this work, however, noisy vision measurements are generated simulating on-board vision-based gate detections with various levels of accuracy. Along the straight part trajectory, n random points $\mathbf{P}_i$ are randomly sampled ($15 < n < 20, i \in [1, n]$). For each sampled point $\mathbf{P}_i$, we calculate the distance between $\mathbf{P}_i$ and the gate which is denoted by $\hat{x}_i - x_g$. Then, the noise $\Delta\mathbf{P}_i$ is generated depending on the distance to the gate that $\Delta\mathbf{P}_i$ is larger when the gate is further away. Finally, $\Delta\mathbf{P}_i$ is added to $\mathbf{P}_i$ to get the simulated measurements $\mathbf{P}_i^v$. This process can be described by equation (33) (Figure 8)

$$\begin{aligned} \mathbf{P}_i^v &= \mathbf{P}_i^m + \Delta\mathbf{P}_i \\ \Delta\mathbf{P}_i &\sim \mathbf{N}(0, \mathbf{S}_i) \\ \mathbf{S}_i &= \begin{bmatrix} \sigma_i^2 & 0 & 0 \\ 0 & \sigma_i^2 & 0 \\ 0 & 0 & \sigma_i^2 \end{bmatrix} \\ \sigma_i &= 0.1(\hat{x}_i - x_g) \end{aligned} \tag{33}$$

The test flights consist of two distinct phases which are shown in Figure 6.

- During the *straight part* (blue line), the gates are in the field of view of the quadrotor and vision-based position measurements are available. The vision-based EKF (VEKF) can run both prediction and update loops. The vision-based gradient descent method (VGD) searches for parameters Θ that make the prediction best fit the noisy measurements.
- During the arc (purple line), no position measurements are available but an open-loop coordinated turn is performed. The VEKF can only rely on model prediction and the gradient descent method uses the last estimated parameters and on-board inertial data to propagate the states of the quadrotor. This phase must be limited in time as the open-loop integration is diverging as can be seen in Figure 9.

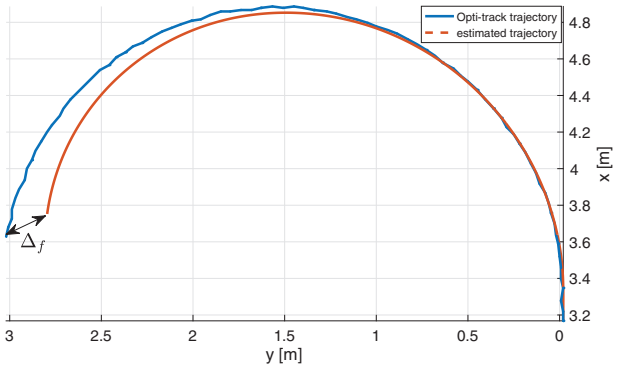

Figure 9. When vision measurements are not available, the quadrotor can only rely on model predictions based on model information and inertial data. This prediction will diverge in time. The better the model prediction is, the smaller the end point prediction error Δ_f becomes.

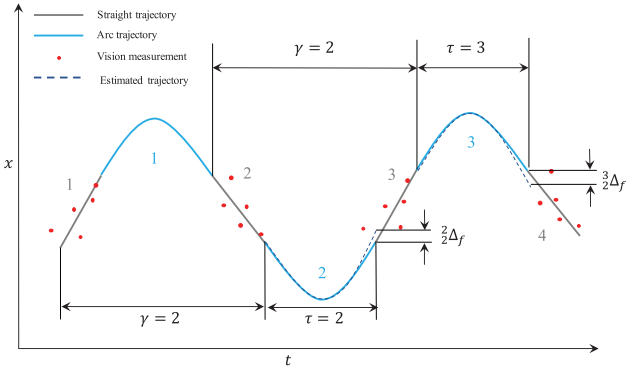

Figure 10. Example test flight data showing the x position in function of time and illustrating the prediction strategy when $\gamma = 2$. First, the data of straight lines 1 and 2 are used to estimate Θ^*. Then the identified model parameters are used to predict the second turn. Finally, the final point error after the second arc ${}_2^2\Delta_f$ is calculated. Here, subscript 2 means the data from 2 straight lines are used and superscript 2 means second arc's trajectory prediction is used. This procedure is repeated by using data of straight lines 2 and 3 and predicting the trajectory of third arc and so forth. (a) Final point error ${}_\gamma^\tau\Delta_f$ in function of γ for various parts of the run τ. (b) Number of FMINCON iterations based on stopping criteria (equation (36)) in function of γ for various parts of the run τ.

The test track is designed to resemble an autonomous drone race track, where it is not possible to keep gates in sight at all times. When using fast gate detection as sole means of position information, some maneuvers need to be performed open-loop. But even when gates are in-sight, better model prediction allows the estimation of more accurate trajectories through the noisy visual data. Therefore, as a performance index we selected the prediction error Δ_f at the final point of the open-loop arc to evaluate the performance of both algorithms.

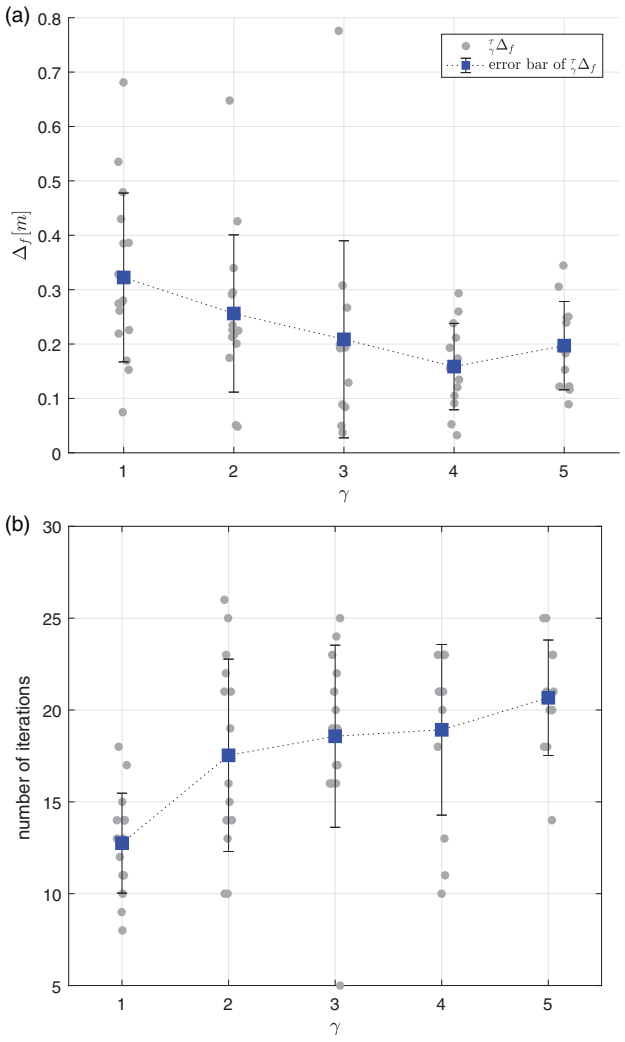

Figure 11. Influence of the history length γ on the prediction accuracy ${}^{\tau}_{\gamma}\Delta_f$ and required number of iterations.

$$\Delta_f = \left\| \begin{bmatrix} x_f \\ y_f \end{bmatrix} - \begin{bmatrix} \hat{x}_f \\ \hat{y}_f \end{bmatrix} \right\| \tag{34}$$

where x_f and y_f, which are from Optitrack, form the ground truth of the end point of the arc, while $\hat{x}_f$ and $\hat{y}_f$ are the filter prediction of the end point.

Analysis of VGD

In this section, we use the on-board flight data and generated vision measurements to analyze the VGD using a MATLAB implementation of gradient descent, FMINCON.

The performance of the gradient descent method is affected by the size of the training data. It is important to investigate how the size of the dataset used to search for Θ^* affects the estimation performance. We use the notation γ ($1 \leq \gamma \leq 5$) to represent the size of the history used by FMINCON. In other words, γ is the number of straight lines whose corresponding vision measurement is used by FMINCON. Too short γ will contain very few visual measurements and the approach is at risk of over-fitting the gate detection noise. Too long γ will violate the constant parameter constraint like for instance equation (13). Figure 10 shows an example where $\gamma = 2$. For each step, we use an array of flight data and vision measurements of size γ in FMINCON to search for Θ^*. Then, Θ^* is used to estimate the trajectory of next arc, which is given by id τ ($1 \leq \tau \leq 15$). Finally the final point error ${}^{\tau}_{\gamma}\Delta_f$ can be calculated using equation (34)

$${}^{\tau}_{\gamma}\Delta_f = \left\| \begin{bmatrix} x_f \\ y_f \end{bmatrix} - \begin{bmatrix} \hat{x}_f \\ \hat{y}_f \end{bmatrix} \right\| \tag{35}$$

The stopping criteria used in the FMINCON optimization is

Table 2. The range of Θ_0 in VGD and VGD-kinematic.

Θ_0	Range	Θ_0	Range
K_*^0	$[-1, 0]$	ϕ_*^{*0}	$[-3^\circ, 3^\circ]$
$\mathbf{b}_a^0$	$[-1m/s^2, 1m/s^2]$	θ_*^{*0}	$[-3^\circ, 3^\circ]$
$\mathbf{b}_g^0$	$[-3^\circ/s, -3^\circ/s]$	ψ_*^{*0}	$[-3^\circ, 3^\circ]$

$$\frac{\|J(\Theta_k) - J(\Theta_{k-1})\|}{\|J(\Theta_k)\|} \leq 10^{-4} \tag{36}$$

With different combinations of τ and γ, a set of 70 ${}^{\tau}_{\gamma}\Delta_f$ is gathered. The prediction accuracy results, ${}^{\tau}_{\gamma}\Delta_f$, and the number of iterations based on the stopping criteria from equation (36) are shown in Figure 11.

Figure 11(a) shows the prediction accuracy ${}^{\tau}_{\gamma}\Delta_f$ as a function of the history length γ. Each gray dot represents an individual arc estimation τ on another part of the data while the blue dots give the average for a given γ. Similarly, in Figure 11(b) the required number of iterations based on the stopping criteria is shown. The figures show that the prediction error ${}^{\tau}_{\gamma}\Delta_f$ keeps decreasing up to $\gamma = 4$. This means that fitting more than one straight part helps improving the accuracy of state estimation. Figure 11(b) shows that the average number of iterations is about 19 and the maximum is only 25, which means this VGD quickly converges and is not very computationally expensive.

Comparison between VEKF, VGD and VGD-kinematic

In this section, in order to show the different performance of the gradient descent between the kinematic model and model from equation (15), we introduce a new method called Vision-based gradient descent method with kinematic model (VGD-kinematic). This method has the same principle as VGD except that it is using a kinematic model 38 as prediction model.

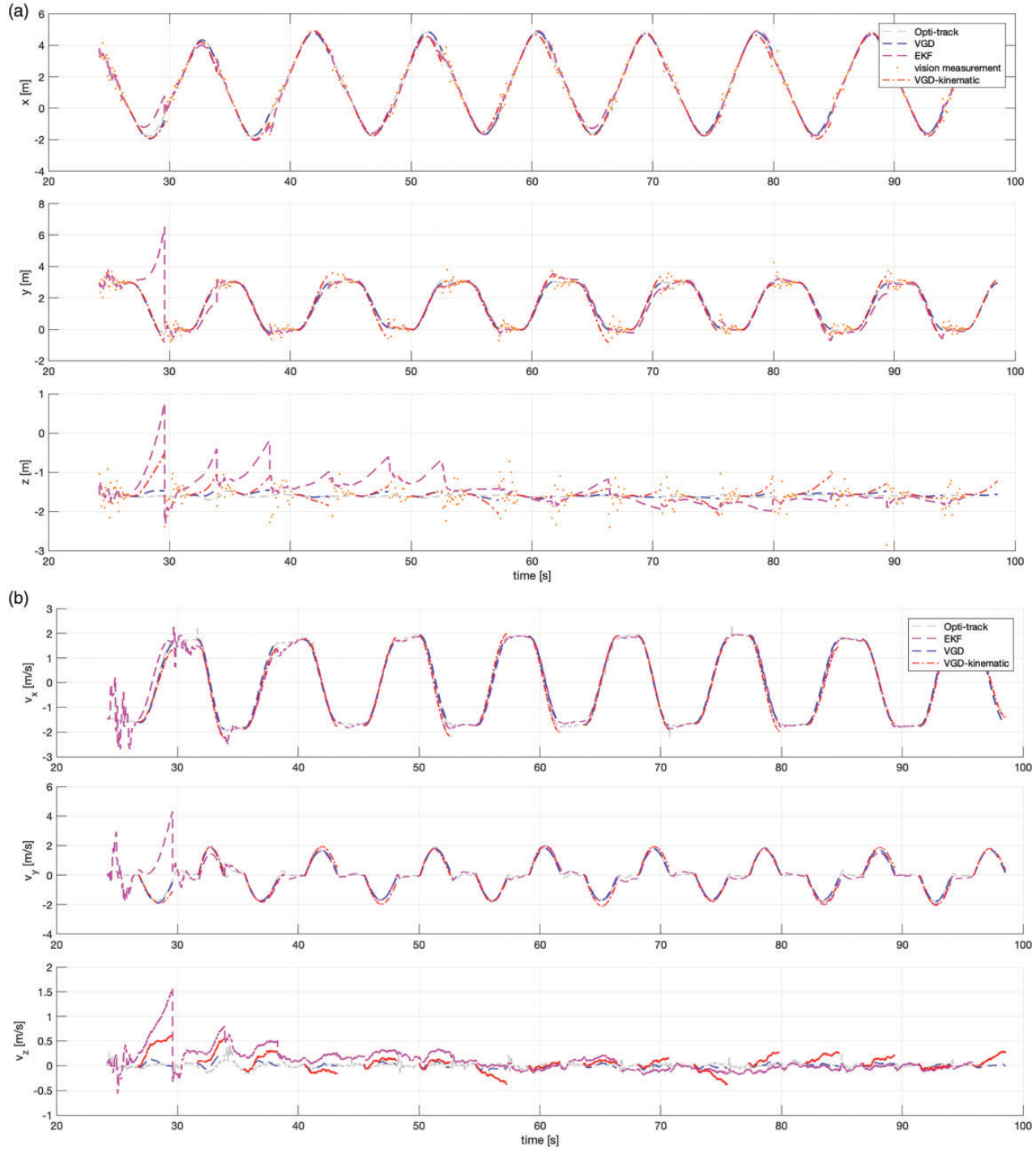

Figure 12. The final point error ${}^{\tau}_{\gamma}\Delta_f$ when using the VEKF, VGD and VGD-kinematic. The VGD has the most stable performance and least ${}^{\tau}_{\gamma}\Delta_f$ compared to the EKF and VGD-kinematic. (a) Position estimation; (b) Velocity estimation; (c) Estimation of accelerometer bias; (d) Estimation of gyro bias.

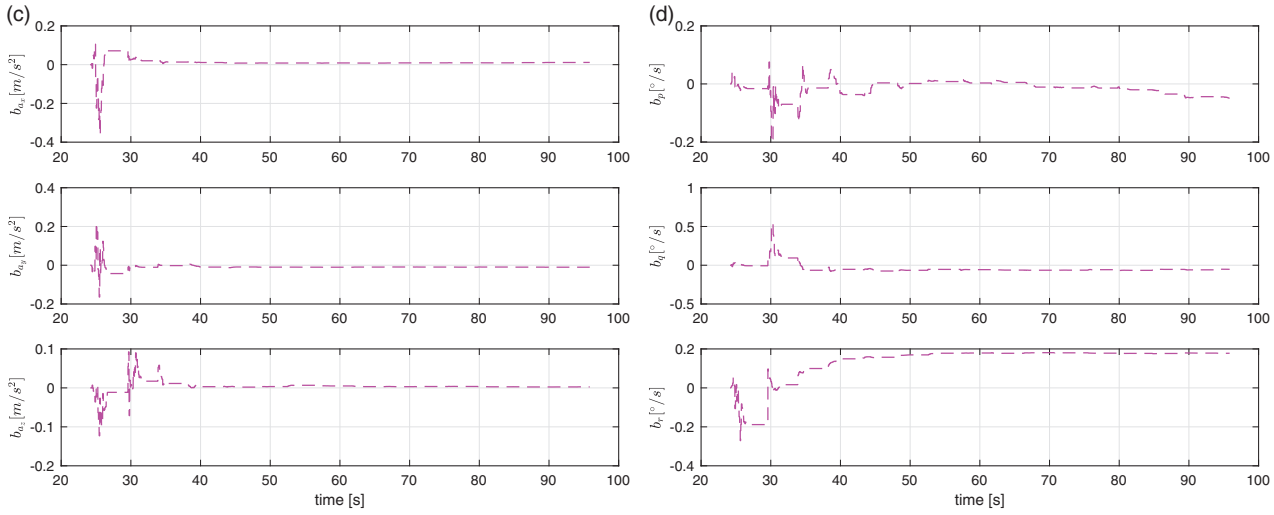

Figure 12. Continued

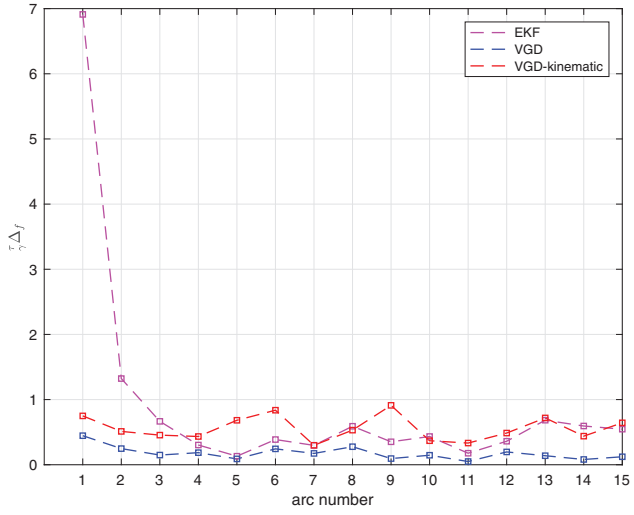

Figure 13. Comparison of the position and velocity estimation results of the EKF and the FMINCON-based gradient descent method using inertial sensors and discrete low frequency noise vision based position measurements from gate detections.

$$\begin{cases} \dot{\mathbf{X}} = \mathbf{V} \\ \dot{\mathbf{V}} = \mathbf{g} + \Re_B^E(\mathbf{a}_m + \mathbf{b}_a) \\ \dot{\Phi} = \Re'(\Omega_m + \mathbf{b}_g) \end{cases} \tag{37}$$

In this case, the parameters to be estimated are the bias of accelerometers and gyros, which can be written as

$$\Theta = [b_{a_x}, b_{a_y}, b_{a_z}, b_p, b_q, b_r]^{\mathrm{T}} \tag{38}$$

To compare the performance of the of three methods, all three methods are tested using the same on-board data and the same generated vision measurements. In both VGD and VGD-kinematic, γ was set to 3, which means that the flight data of the last 3 straights is used in the estimation of Θ^*. Note that during the first two arcs of the flight, there is not yet enough flight data, and γ will be smaller than 3.

The resulting full flight is shown in Figure 13. In Figure 13, the orange dots are the generated vision measurements from the straight parts of the track. The magenta curve is the estimation result of the VEKF. In the VEKF, $\mathbf{R} = diag([2.5^2, 2.5^2, 2.5^2])$, $\mathbf{Q} = diag([(2e-6, 2e-6, 5e-6, e-5, \quad 5e-6, 3e-5, 3e-8, 3e-9, 3e-9, 0, 0, 0, 0, 0, 0])$ and $\mathbf{P}_0 = 10 \times I_{15\times 15}$. The blue curve is the estimation result of the VGD and the red curve is the result of VGD-kinematic. To test the sensitivity of the VGD and the VGD-kinematic algorithm, the initial parameters Θ_0 are selected randomly within some ranges which can be found in Table 2. It can be seen that while the VEKF clearly converges to the measurements. The long prediction horizon combined with few and noisy measurement updates challenges the filter to its limit. On the other hand, the VGD managed to find parameters that fit the model very well through the noisy measurements and is not sensitive to the initial parameters. Even large measurement noise does not affect the prediction too much as the dynamics of the quadrotor cannot explain them.

The final point prediction error ${}^{\tau}_{\gamma}\Delta_f$ after each turn of the three algorithms is shown in Figure 12. The VEKF requires several laps (3rd arc, or about 20 s of flight) to converge to sub-meter prediction accuracy. During the rest of the flight, the EKF can predict the 180° turns with a final point prediction error of around 0.5 m. The VGD-kinematic uses the derived kinematic model as prediction model and utilizes multiple vision

measurements for parameter estimation. It has similar performance when compared with the VEKF. Overall, the VGD, which uses the same measurements as the VGD-kinematic but performs a bias and aerodynamics model estimation, is shown to find the best estimates of all parameters. It even finds good model parameters for the first arc, using only 1 straight line's flight data. During the whole flight, ${}^{\tau}_{\gamma}\Delta_f$ of the VGD is kept around 0.2 m.

Conclusion

Accurate state and parameter estimation is essential for quadrotor control, especially when they perform aggressive maneuver. However, in the environment where only sparse and noisy position measurements are available, a classic Kalman filter can struggle to provide accurate state and model parameter estimation results. In this paper, we presented a novel method that only uses sparse vision measurements to estimate the AHRS error and select aerodynamic parameters of the quadrotor using a gradient descent method. The experiment result shows that our VGD could increase the accuracy of state estimation when compared to a classic Kalman filter in environments where only sparse noisy position measurements are available.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) received no financial support for the research, authorship, and/or publication of this article.

ORCID iD

S Li http://orcid.org/0000-0002-8656-4661
C De Wagter http://orcid.org/0000-0002-6795-8454

References

1. Moon H, Sun Y, Baltes J, et al. The IROS 2016 Competitions [Competitions]. IEEE Robot Autom Mag 2017; 24: 20–29. DOI:10.1109/MRA.2016.2646090.
2. Gross JN, Gu Y, Rhudy MB, et al. Flight-test evaluation of sensor fusion algorithms for attitude estimation. IEEE Trans Aerosp Electron Syst 2012; 48: 2128–2139.
3. Hoffmann G, Huang H, Waslander S, et al. Quadrotor helicopter flight dynamics and control: theory and experiment. In: *AIAA guidance, navigation and control conference and exhibit*. Reston: American Institute of Aeronautics and Astronautics, Aug, 2007, pp. 1–20
4. Huang H, Hoffmann G, Waslander S, et al. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In: *2009 IEEE international conference on robotics and automation*. Piscataway: IEEE, pp. 3277–3282.
5. Bangura M and Mahony R. Nonlinear dynamic modeling for high performance control of a quadrotor. In: 2012 *Proceedings Australasian conference on robotics and automation,* pp.1–10.
6. Svacha J, Mohta K and Kumar V. Improving quadrotor trajectory tracking by compensating for aerodynamic effects. In: 2017 International *conference on unmanned aircraft systems* (ICUAS). Piscataway: IEEE, pp. 860–866.
7. Kai JM, Allibert G, Hua MD, et al. Nonlinear feedback control of quadrotors exploiting first-order drag effects. In: *IFAC World Congress*, Toulouse, France, 2017, pp. 8189–8195
8. Bristeau PJ, Callou F, Vissiere D, et al. The navigation and control technology inside the AR.Drone micro UAV. IFAC Proc Vol 2011; 44: 1477–1484.
9. Yoo CSYCS and Ahn IKAIK. Low cost GPS/INS sensor fusion system for UAV navigation. In: *Digital avionics systems conference, 2003. DASC'03. The 22nd*, vol. 2. Piscataway: IEEE, pp. 8–A.
10. Brown AK. GPS/INS uses low-cost MEMS IMU. IEEE Aerosp Electron Syst Mag 2005; 20: 3–10.
11. Shi E. An improved real-time adaptive Kalman filter for low-cost integrated GPS/INS navigation. In: *2012 International conference on measurement, information and control (MIC)*, vol. 2. Piscataway: IEEE, pp. 1093–1098.
12. Lopes H, Kampen E and Chu Q. Attitude determination of highly dynamic fixed-wing UAVs with GPS/MEMS-AHRS integration. In: 2012 *AIAA guidance, navigation, and control conference*. Reston: American Institute of Aeronautics and Astronautics, p. 4460.
13. Kingston DB and Beard RW. Real-time attitude and position estimation for small UAVS using low-cost sensors. In: *AIAA 3rd unmanned unlimited technical conference, workshop and exhibit*. Reston: American Institute of Aeronautics and Astronautics, pp. 2004–6488.
14. Mellinger D and Kumar V. Minimum snap trajectory generation and control for quadrotors. In: *2011 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE, pp. 2520–2525.
15. Bry A, Richter C, Bachrach A, et al. Aggressive flight of fixed-wing and quadrotor aircraft in dense indoor environments. Int J Rob Res 2015; 34: 969–1002.
16. Tomic T, Schmid K, Lutz P, et al. Toward a fully autonomous UAV: research platform for indoor and outdoor urban search and rescue. IEEE Robot Autom Mag 2012; 19: 46–56.
17. Hrabar S. An evaluation of stereo and laser-based range sensing for rotorcraft unmanned aerial vehicle obstacle avoidance. J Field Robot 2012; 29: 215–239.
18. Valenti RG, Dryanovski I, Jaramillo C, et al. Autonomous quadrotor flight using onboard RGB-D visual odometry. In: *2014 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE, pp. 5233–5238.
19. Bachrach A, Prentice S, He R, et al. Estimation, planning, and mapping for autonomous flight using an RGB-

D camera in GPS-denied environments. Int J Rob Res 2012; 31: 1320–1343.
20. Sampedro C, Bavle H, Rodríguez-Ramos A, et al. A fully-autonomous aerial robotic solution for the 2016 international micro air vehicle competition. In: *2017 International conference on unmanned aircraft systems (ICUAS)*. Piscataway: IEEE, pp. 989–998.
21. Mostegel C, Wendel A and Bischof H. Active monocular localization: towards autonomous monocular exploration for multirotor MAVs. In: *2014 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE, pp. 3848–3855.
22. Huh S, Shim DH and Kim J. Integrated navigation system using camera and gimbaled laser scanner for indoor and outdoor autonomous flight of UAVS. In: *2013 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. Piscataway: IEEE, pp. 3158–3163.
23. Nistér D, Naroditsky O and Bergen J. Visual odometry. In: 2004 *Proceedings of the 2004 IEEE computer society conference on computer vision and pattern recognition. CVPR 2004*, vol. 1. Piscataway: IEEE, pp. I–I.
24. Andert F, Ammann N, Puschel J, et al. On the safe navigation problem for unmanned aircraft: visual odometry and alignment optimizations for UAV positioning. In: *2014 International conference on unmanned aircraft systems (ICUAS)*. Piscataway: IEEE, pp. 734–743.
25. Strydom R, Thurrowgood S and Srinivasan MV. Visual odometry: autonomous UAV navigation using optic flow and stereo. In: *Australasian conference on robotics and automation (ACRA)*. Australian Robotics and Automation Association, pp. 1–10.
26. Mondragón IF, Olivares-Méndez MA, Campoy P, et al. Unmanned aerial vehicles UAVs attitude, height, motion estimation and control using visual systems. Auton Robot 2010; 29: 17–34.
27. Rodolfo García Carrillo L, Enrique Dzul López A, Lozano R, et al. Combining stereo vision and inertial navigation system for a quad-rotor UAV. J Intell Robot Syst 2012; 65: 373–387.
28. Martínez-Carranza J and Calway A. Efficient visual odometry using a structure-driven temporal map. In: *2012 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE, pp. 5210–5215.
29. Falanga D, Mueggler E, Faessler M, et al. Aggressive quadrotor flight through narrow gaps with onboard sensing and computing using active vision. In: *2017 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE, pp. 5774–5781.
30. Loianno G, Brunner C, McGrath G, et al. Estimation, control, and planning for aggressive flight with a small quadrotor with a single camera and IMU. IEEE Robot Autom Lett 2017; 2: 404–411.
31. Jang JS and Liccardo D. Small UAV automation using MEMS. IEEE Aerosp Electron Syst Mag 2007; 22: 30–34.
32. Crassidis JL. Sigma-point Kalman filtering for integrated GPS and inertial navigation. IEEE Trans Aerosp Electron Syst 2006; 42: 750–756.
33. Zhang P, Gu J, Milios EE, et al. Navigation with IMU/GPS/digital compass with unscented Kalman filter. In: *2005 IEEE international conference on mechatronics and automation*, vol. 3. Piscataway: IEEE, pp. 1497–1502.
34. Lange S, Sünderhauf N and Protzel P. Incremental smoothing vs. filtering for sensor fusion on an indoor UAV. In: *2013 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE, pp. 1773–1778.
35. Indelman V, Williams S, Kaess M, et al. Factor graph based incremental smoothing in inertial navigation systems. In: *2012 15th International conference on information fusion (FUSION)*. Piscataway: IEEE, pp. 2154–2161.
36. Smeur EJ, Chu Q and de Croon GC. Adaptive incremental nonlinear dynamic inversion for attitude control of micro air vehicles. *J Guid Control Dyn* 2015; 38(12): 450–461.
37. Hoffmann GM, Huang H, Waslander SL, et al. Quadrotor helicopter flight dynamics and control: theory and experiment. In: *Proceedings of the AIAA guidance, navigation, and control conference*, vol. 2. Reston: American Institute of Aeronautics and Astronautics, p. 4.
38. Bangura M, Mahony R, et al. Nonlinear dynamic modeling for high performance control of a quadrotor. In: *Australasian conference on robotics and automation*. pp. 1–10.
39. Gati B. Open source autopilot for academic research – the paparazzi system. In: *American control conference (ACC), 2013*. Washington, DC: IEEE, pp. 1478–1481.