Drone reference tracking in a non-inertial frame: control, design and experiment.

Yasmine Marani
dept. Electril and Computer Engineering
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
yasmine.marani@kaust.edu.sa

Kuat Telegenov
dept. Electril and Computer Engineering
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
kuat.telegenov@kaust.edu.sa

Eric Feron
dept. Electril and Computer Engineering
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
eric.feron@kaust.edu.sa

Meriem-Taous Laleg Kirati
dept. Electril and Computer Engineering
King Abdullah University of Science and Technology
Thuwal, Saudi Arabia
National Institute for Research in Digital Science and Technology
Paris-Saclay, France
taousmeriem.laleg@kaust.edu.sa

2022 IEEE/AIAA 41st Digital Avionics Systems Conference (DASC) | 978-1-6654-8607-1/22/$31.00 ©2022 IEEE | DOI: 10.1109/DASC55683.2022.9925849

***Abstract*—Drones and mobile robots in general experience motion sickness when put inside a GPS denied moving environment. This navigation problem is of a novel nature and barely explored in the literature. The objective of this paper is to design a control strategy for drone reference tracking inside the moving environment. First, we provide an initial formulation of the problem where the non-inertial frame is assumed to have only a translation motion relative to the inertial reference frame. Then we derive the dynamic model of the drone in the non-inertial frame using the relative motion principle. After that, we use a combined Sliding mode controller and Extended Kalman Filter with Unknown Inputs (EKF-UI) for trajectory tracking. The EKF-UI enables to jointly estimates the states of the drone and the non-inertial frame accelerations. The sliding mode control laws are computed using the measured and estimated states to ensure the asymptotic convergence of the whole observer-based control strategy. The performance of the proposed observer-based control strategy is tested through simulation and experiment.**

***Index Terms*—Unmanned Aerial Vehicles, reference tracking, non-inertial frame, Extended Kalman Filter with Unknown Inputs, sliding mode.**

I. INTRODUCTION

Unmanned aerial vehicles (UAVs), also known as drones, have gained popularity in recent years as a tool for a variety of applications, including aerial photography and mapping, search and rescue, smart agriculture, tracking and monitoring industrial facilities and wildlife, surveillance, disaster response, and the delivery of healthcare supplies [1]–[18]. An instrumented positioning infrastructure or a Global Navigation Satellite System (GNSS) is needed for these applications. However, UAVs may prove useful in a wide range of unexplored environments in which GNSS cannot be used, including elevators, cars, boats, trucks, trains, ships, and airplanes. Flying in moving environments can be very problematic for several reasons. In the absence of GNSS, a UAV can't obtain its absolute position in the Geocentric coordinate system, which is called an inertial frame for vehicles like drones. The onboard sensors like accelerometer and gyroscope will provide angular rates and acceleration with respect to this inertial frame. On the other hand, the onboard positioning sensors will provide a position with respect to the moving environment, which is non-inertial frame for the drones. Consequently, a UAV's onboard positioning will be out of sync with its inertial measurements. There are several YouTube videos that demonstrate people trying to fly UAVs inside trucks or elevators as an example of this challenge [19]–[23]. Human pilots achieve decent results at low speeds, but their results are inconsistent. It becomes more difficult or nearly impossible to control a UAV as the moving environment acceleration increases. Few studies have been conducted on UAVs operating with non-inertial reference frames. In [24] authors present an autonomous landing control approach for an unmanned aerial vehicle subject to wind disturbance and three-dimensional movements of the landing platform. Another study considers the problem of controlling a UAV attached to a generic and independently moving platform [25]. A similar study addresses tracking issues for a non-inertial frame-referenced UAV that is controlled by a cascaded PID controller [26]. Neither of these works considered flying UAVs in a non-inertial environment or on a platform without access to inertial position data. To the best of the authors' knowledge, autonomous UAVs flying inside a moving environment without inertial positioning have not presented in the literature. A problem formulation is presented in this paper in order to more comprehensively understand the question, followed by simulation and experiment results.

A UAV modeling has been extensively studied in the literature. The most widely used model is the non-linear model was developed using the Newton-Euler formalism in [27]–[29]. A hybrid dynamic model for drones was also presented in [30]. However, all these models describe the drone motion in an inertial reference frame. Therefore, based on the relative motion principle, we derive the drone model in the non-inertial frame where the moving environment acceleration acts as an unknown input.

There have been studies of observers for systems with unknown inputs for nearly half a century. Several designs have been proposed for both linear and non-linear systems to jointly estimate the states and the unknown inputs [31]–[33]. In [34] the authors propose an unknown input Kalman Filter for linear systems, and in [35] the Extended Kalman Filter with Unknown Inputs (EKF-UI) is developed. The EKF-UI estimator was tested in several applications [36]–[38] and demonstrated good performance. Therefore, we will use the EKF-UI in this paper to estimate the drone's states simultaneously with the moving environment accelerations.

Quadrotor control has been widely studied in the literature. Several linear controllers such as PID, LQ, and LQR are proposed to control the UAV by linearizing its dynamics around an operating point [39]–[41]. In [42], [43], a special case is considered where the drone constantly accelerates. The authors used a linear controller consisting of a triple integrator to tackle this problem. In addition, [44] proposes adaptive designs that were tested afterward in-flight. Unlike linear controllers that provide local convergence guarantees, non-linear controllers provide global convergence for a wider flight range. Several non-linear controllers are implemented for quadrotor control and demonstrate good trajectory tracking performance. Examples of these controllers include backstepping [45], feedback linearization [46], and sliding mode [47]. In the present paper, we use the sliding mode controller, which does not require any knowledge about the non-inertial frame acceleration except its bound. In [48], the authors implemented the sliding mode controller experimentally on a drone in presence of wind disturbances where it demonstrated good performance, especially for disturbance rejection. To the best of the authors' knowledge, sliding mode control or any other controller has not been successfully tested yet in experiments for drone control inside moving environments.

The organization of the present paper is as follows: we first formulate the problem in Section II for moving environments with translation motion only. Then, we derive in Section III the drone model in a non-inertial frame. In Section IV and Section V, we present the designing steps of the Extended Kalman Filter with Unknown Input and the Sliding Mode controller respectively. The experimental setup is explained in Section VI. The simulation and experimental results of the combined EKF-UI and Sliding Mode controller are presented in Section VII. Finally, concluding remarks and future work are given in Section VIII.

II. Problem formulation

This paper tackles the drone reference tracking problem inside a GPS-denied moving environment whose position, velocity, and acceleration are not assumed to be measured. The moving environment is associated with a non-inertial frame **R'**(o',x',y',z') has only translation motion with respect to the inertial frame **R**(o,x,y,z). Figure 1 illustrates the different frames considered in this problem.

Since the moving environment translates only, the linear positions and velocities of the drone are the only states affected by the non-inertial frame motion. The angular positions and angular rates remain unchanged with the change of frames. The measured states are the drone's relative linear position in the non-inertial frame, the angular position, and the angular rates. In the experiments conducted as part of this work, the drone's angular positions and rates are measured using MindPX instrumentation unit. [49], while the relative linear positions in the moving environment are measured using onboard localization sensors. More details regarding the instrumentation will be given in section [Experiment Section]. The drone's linear velocities measurements are usually the result of sensor fusion where linear acceleration measurements are combined with position measurements. However, onboard accelerometers provide absolute linear acceleration while the position sensors provide relative position in the non-inertial frame. Therefore, an observer needs to be implemented to recover the relative linear velocities of the drone.

The drone reference tracking problem will be tackled following the steps hereafter:

1) Derive the model describing the drone dynamics in the non-inertial frame.
2) Design and estimator to estimate the non-measurable states (linear velocities) simultaneously with unknown inputs (non-inertial frame accelerations).
3) Design a controller that ensures the convergence of the drone states to their desired reference inside the moving environment.

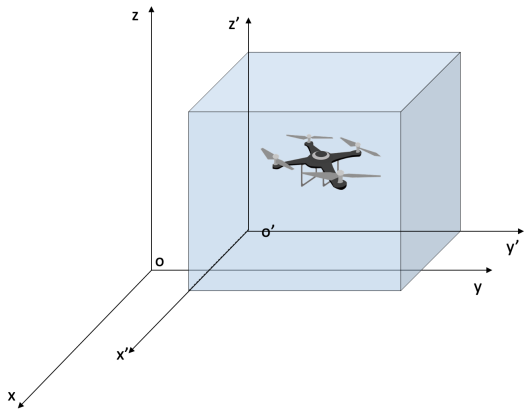

Fig. 1. The different frames related to flying a drone inside a moving environment.

III. Dynamical model

To derive the model of the drone in the non-inertial frame, first, we write the dynamical model of the drone in the inertial frame. Then, using the relative motion principles, we derive the drone dynamic model in the non-inertial frame.

A. Dynamic model of the drone in the inertial frame **R**

The equations of motion of the drone with respect to the inertial frame **R** are given by [28]

$$\begin{array}{l}\ddot{x}_a = u_1(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)\\ \ddot{y}_a = u_1(\cos\phi\sin\theta\sin\psi - \sin\psi\cos\psi)\\ \ddot{z}_a = u_1(\cos\phi\cos\theta) - g\\ \ddot{\phi} = a\dot{\theta}\dot{\psi} + u_3\\ \ddot{\theta} = b\dot{\psi}\dot{\phi} + u_4\\ \ddot{\psi} = c\dot{\theta}\dot{\phi} + u_5.\end{array} \tag{1}$$

where $a = \frac{I_y - I_z}{I_x}, b = \frac{I_z - I_x}{I_y}$ and $c = \frac{I_x - I_y}{I_z}$, and u_1, u_2, u_3, and u_4 are the inputs of the drone given by

$$\begin{array}{l}u_1 = \frac{b}{m}\left(\Omega_1^2 + \Omega_2^2 + \Omega_3^2 + \Omega_4^2\right)\\ u_2 = \frac{b}{I_x}\left(\Omega_4^2 - \Omega_2^2\right)\\ u_3 = \frac{b}{I_y}\left(\Omega_3^2 - \Omega_1^2\right)\\ u_4 = \frac{l}{I_z}\left(-\Omega_1^2 + \Omega_2^2 - \Omega_3^2 + \Omega_4^2\right)\end{array}.$$

ϕ, θ, and ψ represent respectively the pitch, roll, and yaw angles. Ω_i, i=1,2,3,4 represent the angular rates of the four rotors.

The state-space model of the drone motion in the inertial frame is:

$$\begin{cases}\dot{x}_{a1} = x_{a2}\\ \dot{x}_{a2} = [\cos(x_7)\sin(x_9)\cos(x_{11}) + \sin(x_7)\sin(x_{11})]\,u_1\\ \dot{x}_{a3} = x_{a4}\\ \dot{x}_{a4} = [\cos(x_7)\sin(x_9)\sin(x_{11}) - \sin(x_7)\cos(x_{11})]\,u_1\\ \dot{x}_{a5} = x_{a6}\\ \dot{x}_{a6} = [\cos(x_7)\cos(x_9)]\,u_1 - g\\ \dot{x}_7 = x_8\\ \dot{x}_8 = ax_{10}x_{12} + u_2\\ \dot{x}_9 = x_{10}\\ \dot{x}_{10} = bx_8x_{12} + u_3\\ \dot{x}_{11} = x_{12}\\ \dot{x}_{12} = cx_8x_{10} + u_4.\end{cases} \tag{2}$$

The state vector for the drone motion in the inertial frame is given by $(x_{a1}, x_{a2}, x_{a3}, x_{a4}, x_{a5}, x_{a6}, x_7, x_8, x_9, x_{10}, x_{11}, x_{12})^T = (x_a, \dot{x}_a, y_a, \dot{y}_a, z_a, \dot{z}_a, \phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi})^T$.

B. Dynamic model of the drone in the non-inertial frame **R'**

Since the non-inertial frame has translates only in the inertial frame, the drone's linear positions and velocities are the only states affected. The angular positions and velocities remain unchanged.

The linear acceleration of the drone in the inertial frame **R** is expressed as the sum of the moving environment acceleration in **R** and the drone relative acceleration in **R'**

$$\begin{pmatrix}\ddot{x}_a\\ \ddot{y}_a\\ \ddot{z}_a\end{pmatrix} = \begin{pmatrix}\ddot{x}_e + \ddot{x}_r\\ \ddot{y}_e + \ddot{y}_r\\ \ddot{z}_e + \ddot{z}_r\end{pmatrix}. \tag{3}$$

where $\ddot{x}_e, \ddot{y}_e, \ddot{z}_e$ represent the accelerations of the moving environment, and $\ddot{x}_r, \ddot{y}_r, \ddot{z}_r$, represent the relative accelerations of the drone in the non-inertial frame. Therefore the drone acceleration w.r.t R' can be expressed as:

$$\begin{pmatrix}\ddot{x}_r\\ \ddot{y}_r\\ \ddot{z}_r\end{pmatrix} = \begin{pmatrix}\ddot{x}_a - \ddot{x}_e\\ \ddot{y}_a - \ddot{y}_e\\ \ddot{z}_a - \ddot{z}_e\end{pmatrix}. \tag{4}$$

For the purpose of clear notation we consider the following notations for the rest of the paper:

$$\begin{pmatrix}\ddot{x}_e\\ \ddot{y}_e\\ \ddot{z}_e\end{pmatrix} = \begin{pmatrix}a_x\\ a_y\\ a_z\end{pmatrix}.$$

$$\mathrm{c}(x) = \cos(x),$$

$$\mathrm{s}(x) = \sin(x).$$

The state-space representation of the drone in the box frame is

$$\begin{cases}\dot{x}_1 = x_2\\ \dot{x}_2 = [\mathrm{c}(x_7)\,\mathrm{s}(x_9)\,\mathrm{c}(x_{11}) + \mathrm{s}(x_7)\,\mathrm{s}(x_{11})]\,u_1 - a_x\\ \dot{x}_3 = x_4\\ \dot{x}_4 = [\mathrm{c}(x_7)\,\mathrm{s}(x_9)\,\mathrm{s}(x_{11}) - \mathrm{s}(x_7)\,\mathrm{c}(x_{11})]\,u_1 - a_y\\ \dot{x}_5 = x_6\\ \dot{x}_6 = [\mathrm{c}(x_7)\,\mathrm{c}(x_9)]\,u_1 - g - a_z\\ \dot{x}_7 = x_8\\ \dot{x}_8 = ax_{10}x_{12} + u_2\\ \dot{x}_9 = x_{10}\\ \dot{x}_{10} = bx_8x_{12} + u_3\\ \dot{x}_{11} = x_{12}\\ \dot{x}_{12} = cx_8x_{10} + u_4,\end{cases}$$
$$y = \begin{pmatrix}x_1 & x_3 & x_5 & x_7 & x_8 & x_9 & x_{10} & x_{11} & x_{12}\end{pmatrix}^T. \tag{5}$$

$x = (x_r, \dot{x}_r, y_r, \dot{y}_r, z_r, \dot{z}_r, \phi, \dot{\phi}, \theta, \dot{\theta}, \psi, \dot{\psi})^T$ is the state vector, and a_x , a_y, a_z represent the acceleration of the box with respect to x , y , and z axis, respectively.

System 5 has the following global structure:

$$\begin{cases}\dot{x} = f_c(x,u) + Ea(t)\\ y = Cx,\end{cases} \tag{6}$$

where $a(t) = [a_x, a_y, a_z]^T$ is the moving environment acceleration. The matrices E and C are given by:

$$E = \begin{pmatrix}0 & 0 & 0\\ -1 & 0 & 0\\ 0 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & 0\\ 0 & 0 & -1\\ \hline \multicolumn{3}{c}{\mathbf{0}_{6\times 3}}\end{pmatrix}$$

$$C = \left(\begin{array}{cccccc|c}1 & 0 & 0 & 0 & 0 & 0 & \\ 0 & 0 & 1 & 0 & 0 & 0 & \mathbf{0}_{3\times 6}\\ 0 & 0 & 0 & 0 & 1 & 0 & \\ \hline \multicolumn{6}{c|}{\mathbf{0}_6} & I_6\end{array}\right)$$

IV. ESTIMATION PROBLEM

Adding process and measurement noise to system 6, the system becomes

$$\begin{cases} \dot{x} = f_c(x,u) + Ea(t) + w(t) \\ y = Cx + v(t), \end{cases} \tag{7}$$

where $w(t)$ is the process noise and $v(t)$ is the measurement noise.

In the above system, the moving environment acceleration $a(t)$ acts as an unknown input on the drone system. Therefore, we will use We use Extended Kalman Filter with Unknown Input (EKF-UI) [35] to jointly estimate the drone's states and the moving environment acceleration. In order to jointly estimate the states of the drone and the moving environment acceleration. Without loss of generality, we assume that $a(t)$ is piece-wise constant.

First, we augment the system considering the above assumption

$$\begin{cases} \dot{\zeta} = \begin{pmatrix} \dot{x} \\ \dot{a} \end{pmatrix} = \begin{pmatrix} f_c(x,u) + Ea(t) \\ 0 \end{pmatrix} + \begin{pmatrix} w(t) \\ w_a(t) \end{pmatrix} \\ y = C_\zeta \zeta + v(t) \end{cases}, \tag{8}$$

where $\zeta = [x \quad a]^T$ is the augmented state vector, $w_a(t)$ is the acceleration's process noise, and $C_\zeta = [C \quad 0_{9\times 3}]$.

Then, we discretize the augmented system using backward Euler integrator:

$$\begin{cases} \zeta[k] = \zeta[k-1] + T_s \begin{pmatrix} f_c(x[k-1], u[k]) + Ea[k-1] \\ 0 \end{pmatrix} \\ \qquad + T_s \begin{pmatrix} w(t) \\ w_a(t) \end{pmatrix} \\ y[k] = C_\zeta \zeta[k] + v[k] \end{cases}. \tag{9}$$

where T_s is the sampling period.

The EKF-UI implementation has two phases:

1) Prediction phase:

We compute the a-priori estimate the augmented state $\hat{\zeta}$ and its covariance matrix $\hat{P}$

$$\hat{\zeta}[k] = f(\hat{\zeta}[k-1], u[k])$$
$$\hat{P}[k] = A[k]\hat{P}[k-1]A[k]^{\mathrm{T}} + Q,$$

where Q represents the covariance matrix of the process noise $w(t)$ and the unknown input noise $w_a(t)$, and A is the Jacobi matrix:

$$A[k] = \begin{pmatrix} T_s A_1[k] + I_{12} & T_s E \\ 0_{3\times 12} & I_3 \end{pmatrix},$$

$$A_1[k] = \begin{pmatrix} \frac{\partial \boldsymbol{f}}{\partial x_1} \\ \frac{\partial \boldsymbol{f}}{\partial x_2} \\ \vdots \\ \frac{\partial \boldsymbol{f}}{\partial x_n} \end{pmatrix}_{x[k-1], u[k]}.$$

2) Update phase

First, we compute the residual error e of the output and its covariance matrix S

$$e[k] = y[k] - C_\zeta \hat{\zeta}[k]$$
$$S[k] = R[k] + C_\zeta \hat{P}[k] C_\zeta^{\mathrm{T}},$$

where R represents the covariance matrix of the measurement noise $v(t)$.

The Kalman gain is given by:

$$K[k] = \hat{P}[k] C_\zeta^{\mathrm{T}} S[k]^{-1},$$

Finally, we obtain the updated state estimate and its covariance matrix:

$$\hat{\zeta}[k] = \hat{\zeta}[k] + K[k]e[k]$$
$$P[k] = (I - K[k]C_\zeta)\hat{P}[k].$$

V. CONTROL PROBLEM

To design a controller for the drone, we consider the following virtual control inputs

$$\begin{cases} u_x = \mathrm{c}(x_7)\,\mathrm{s}(x_9)\,\mathrm{c}(x_{11}) + \mathrm{s}(x_7)\,\mathrm{s}(x_{11}) \\ u_y = \mathrm{c}(x_7)\,\mathrm{s}(x_9)\,\mathrm{s}(x_{11}) - \mathrm{s}(x_7)\,\mathrm{c}(x_{11}), \end{cases} \tag{10}$$

The state space representation of the drone becomes:

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = u_x u_1 - a_x \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = u_y u_1 - a_y \\ \dot{x}_5 = x_6 \\ \dot{x}_6 = [\mathrm{c}(x_7)\,\mathrm{c}(x_9)]\,u_1 - g - a_z \\ \dot{x}_7 = x_8 \\ \dot{x}_8 = a x_{10} x_{12} + u_2 \\ \dot{x}_9 = x_{10} \\ \dot{x}_{10} = b x_8 x_{12} + u_3 \\ \dot{x}_{11} = x_{12} \\ \dot{x}_{12} = c x_8 x_{10} + u_4. \end{cases} \tag{11}$$

In this paper, we will adopt the sliding surface proposed by Slotine and Li propose in [50] of the following general form

$$S(x) = \left(\frac{d}{dt} + \lambda\right)^{r-1} e(x),$$

where $e(x) = x - x_d$ is the tracking error, λ is a positive constant, and r is the relative degree of the system.

Therefore, the sliding surfaces for system 11 are given by

$$S_i = \dot{e}_i + \lambda_i e_i.$$

where $e_i = x_i - x_{id}$, for $i = 1, 3, 5, 7, 9, 11$, are the position tracking errors, and λ_i, $i = 1, 3, 5, 7, 9, 11$, are positive constants.

To derive the control laws, we consider the following:

Assumption1

The the moving environment accelerations are bounded, i.e. $|a_x| \leq a_1$, $|a_y| \leq a_3$, $|a_z| \leq a_5$.

Control input for x :
Consider the following Lyapunov function:

$$V_1 = \frac{1}{2} S_1^T S_1,$$

$$\dot{V}_1 = S_1[\ddot{e}_1 + \lambda_1 \dot{e}_1],$$
$$\dot{V}_1 = S_1[\ddot{x}_1 - \ddot{x}_{1d} + \lambda_1 (\dot{x}_1 - \dot{x}_{1d})],$$
$$\dot{V}_1 = S_1[u_x u_1 - a_x - \ddot{x}_{1d} + \lambda_1 (x_2 - \dot{x}_{1d})],$$

Since x_2 is not measured, we substitute it by its estimate $\hat{x}_2$

$$\dot{V}_1 = S_1 [u_x u_1 - \ddot{x}_{1d} + \lambda_1 (\hat{x}_2 - \dot{x}_{1d})] - S_1 a_x,$$
$$\dot{V}_1 \leq S_1 [u_x u_1 - \ddot{x}_{1d} + \lambda_1 (\hat{x}_2 - \dot{x}_{1d})] + |a_x| \, |S_1| \,,$$

Using **Assumption 1**

$$\dot{V}_1 \leq S_1 (u_x u_1 - \ddot{x}_{1d} + \lambda_1 (\hat{x}_2 - \dot{x}_{1d})) + a_1 |S_1|,$$

For $\dot{V}_1 \leq 0$ it's enough to choose :

$$\dot{V}_1 \leq S_1 \underbrace{[u_x u_1 - \ddot{x}_{1d} + \lambda_1 (\hat{x}_2 - \dot{x}_{1d}) + a_1 \operatorname{sign}(S_1)]}_{-k_1 \operatorname{sign}(S_1)},$$

$$u_x = \frac{1}{u_1} [- (k_1 + a_1) \operatorname{sign}(S_1) + \ddot{x}_{1d} - \lambda_1 (\hat{x}_2 - \dot{x}_{1d})] \,.$$

Control Input for y and z:
Using the same steps as above, we find the control laws for y and z:

$$u_y = \frac{1}{u_1} [- (k_3 + a_3) \operatorname{sign}(S_3) + \ddot{x}_{3d} - \lambda_3 (\hat{x}_4 - \dot{x}_{3d})] \,.$$

$$u_1 = \frac{1}{\mathrm{c}(x_7)\, \mathrm{c}(x_9)} [-(k_5 + a_5) \operatorname{sign}(S_5) + g + \ddot{x}_{5d} - \lambda_5 (\hat{x}_6 - \dot{x}_{5d})].$$

For x_7 and $x_9 \neq \frac{\pi}{2}$
Control input for ϕ :

$$V_7 = \frac{1}{2} S_7^T S_7,$$

$$\dot{V}_7 = S_7[\ddot{e}_7 + \lambda_7 \dot{e}_7]$$

$$\dot{V}_7 = S_7[a x_{10} x_{12} + u_2 - \ddot{x}_{7d} + \lambda_7 (x_8 - \dot{x}_{7d})],$$

For $\dot{V}_7 \leq 0$ it is enough to choose :

$$u_2 = -k_7 \operatorname{sign}(S_7) - a x_{10} x_{12} + \ddot{x}_{7d} - \lambda_7 (x_8 - \dot{x}_{7d}) \,.$$

Control input for θ and ψ:
Using the same steps as above:

$$u_3 = -k_9 \operatorname{sign}(S_9) - b x_8 x_{12} + \ddot{x}_{9d} - \lambda_9 (x_{10} - \dot{x}_{9d}) \,.$$

$$u4 = -k_{11} \operatorname{sign}(S_{11}) - c x_8 x_{10} + \ddot{x}_{11d} - \lambda_{11} (x_{12} - \dot{x}_{11d}) \,.$$

The desired trajectories x_{1d}, x_{3d}, x_{5d}, and x_{11d} are set a-priori. However, x_{7d} and x_{9d} are obtained from the virtual control laws u_x and u_y as follows:

$$\begin{bmatrix} s(x_{7d}) \\ c(x_{7d})\, s(x_{9d}) \end{bmatrix} = \begin{bmatrix} s(x_{11d}) & -c(x_{11d}) \\ c(x_{11d}) & s(x_{11d}) \end{bmatrix} \begin{bmatrix} u_x \\ u_y \end{bmatrix}.$$

VI. Experimental setup

To support the theoretical part and numerical simulation we conducted the experiment inside the moving elevator. For the moving elevator, the problem can be simplified into a one-dimensional moving environment along the z-axis. Thus, the variables that will be controlled by the sliding mode controller are resumed to x_5, and x_6, The rest of the states variables will be managed by the PX4 pre-built onboard controller, as their desired references will be set to zero. Therefore, the state equations of the reduced control problem are

$$\begin{cases} \dot{x}_5 = x_6 \\ \dot{x}_6 = u_1 - g - a_z. \end{cases} \tag{12}$$

The sliding mode control law for system 13 becomes

$$u_1 = -(k_5 + a_5) \operatorname{sign}(S_5) + g + \ddot{x}_{5d} - \lambda_5 (\hat{x}_6 - \dot{x}_{5d}) \,.$$

The augmented system for the EKF-UI to be implemented onboard of the PX4 is given by

$$\begin{cases} x_5[k] = x_5[k-1] + T_s x_6 \\ x_6[k] = x6[k-1] + T_s(u_1[k-1] - g - a_z[k-1]) \\ a_z[k] = a_z[k-1] \end{cases} \tag{13}$$

The process covariance matrix used in the experiment was chosen as $Q = I_3$, and the measurement covariance matrix was chosen as $R = 1000$. The sampling time used during the experiment is $T_s = 0.02$. The drone parameters are given in table I.

TABLE I
Drone parameters

Name	Symbol	Value	Unit
Mass	m	1.61	Kg
Distance between the rotor and the center of mass	l	0.23	m
Moment of inertia along x	I_x	8.1×10^{-3}	Kg.m^2
Moment of inertia along y	I_y	8.1×10^{-3}	Kg.m^2
Moment of inertia along z	I_z	14.2×10^{-3}	Kg.m^2

To conduct the experiment the following hardware was used. A drone DJI Flame Wheel F450 with MindPX flight controller running PX4 software, an OptiTrack tracking system, a Windows OS computer to host the data for the OptiTrack system, a Linux OS companion computer (Odroid XU4) to run EKF-UI and Sliding mode controller onboard, and a Wi-Fi router. The OptiTrack system consists of six infrared (IR) cameras, Flex 3, mounted on the stage stands inside the elevator as shown in figure 2.
The Motive software on a Windows OS computer is used for visualizing and managing the OptiTrack system as well as processing the data obtained from the IR cameras. The position and orientation of the UAVs with multiple reflective markers can be tracked and continuously streamed by the Motive software to an IP address via the Wi-Fi router. The Motive software defines a remote Virtual-Reality Peripheral Network (VRPN) server to stream the tracking information once the rigid body is created. The VRPN server will transmit

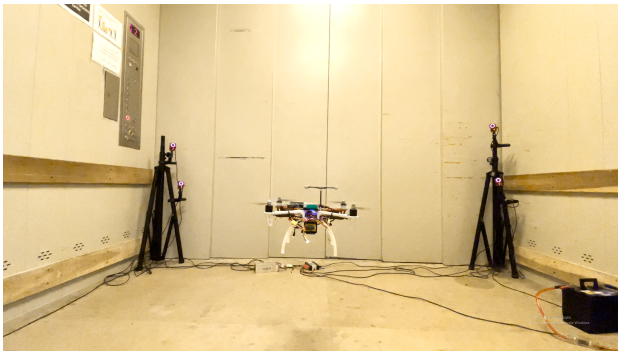

Fig. 2. Experimental setup in the elevator

the position and orientation, of a UAV to the designated remote server.

A companion onboard computer is used to perform more sophisticated and high computations that the flight controller can not. The flight controller is designed for low-level tasks, e.g. attitude control, motor driving, and sensor data acquisition. On the other hand, the companion computer is used for high-level-control, e.g. path planning, trajectory generation, optimization, image processing and etc. The robot operating system (ROS) is a framework with tools and libraries that helps to build robotics applications. The ROS is installed on the drone companion computer. It runs a specific ROS package, VRPN Client ROS, to connect to the remote VRPN server and retrieve the drone position and orientation from the Motive software. The position information is processed by another ROS package, Mavros, which provides a communication driver between the onboard computer and the flight controller via MAVLink protocol. The custom ROS package utilizes the position information for EKF-UI and Sliding mode controller implementation.

The experiment started with a drone taking off autonomously and hovering at an altitude of 0.6 meters. At this stage drone used a PX4 position controller. After a specific time, the code changed the setpoint to accept u_1 as an acceleration setpoint in the z-axis only. Concurrently EKF-UI algorithm is running within the same loop for drone velocity and elevator acceleration estimation. At this time, the elevator started moving upward and downward in the building with five floors. The video from the experiment can be accessed via the following YouTube link: https://youtu.be/OwYMFWLVwtM. It is evident that the drone ascended when the elevator accelerated and then descended when the elevator decelerated, while the elevator is moving downward.

VII. Simulation and Experimental Results

In this section the simulation results will be compared with the results from the experiment conducted in the elevator. The recorded elevator acceleration will be fed to Matlab simulation environment and the corresponding altitude x_5 is generated using the implemented EKF-UI and Sliding Mode controller code. We also record the estimated velocity along the z-axis $\hat{x}_6$

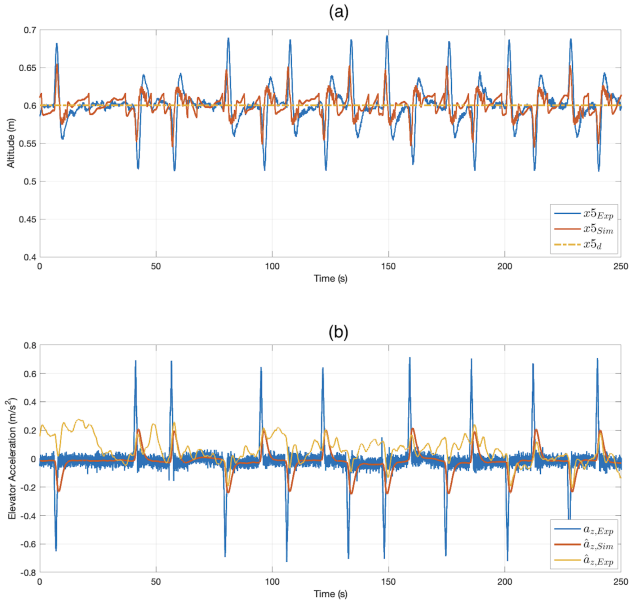

Fig. 3. Altitude reference tracking vs the elevator Acceleration. (a): Comparison between the experimentally measured altitude (in blue) inside the elevator and the simulated altitude (in orange) where the dashed yellow line represents the desired altitude. (b): comparison between the measured elevator acceleration (in blue) and the estimated acceleration using the implement EKF-UI in Matlab (orange) and the estimated acceleration using the onboard EKF-UI (yellow).

and the estimated elevator acceleration $\hat{a}_z$ using the EKF_UI. For the sake of notation clarity, we consider the following:

- x_{Exp} : experimentally measured variable
- x_{Sim} : simulated variable using Matlab.
- $\hat{x}_{Exp}$: estimated variable using the on-board EKF-UI.
- $\hat{x}_{Sim}$: estimated variable using the implemented EKF-UI on Matlab.

The simulation was conducted considering the following:

- Process and unknown input noise covariance matrix $Q = I_{15}$
- Gaussian measurement noise $v(t)$, with a mean of 0.01
- Gaussian Process noise $w(t)$, with a mean of 0.001
- Gaussian noise for the non-inertial frame acceleration $w_a(t)$, with a mean of 0.0001
- ζ_0 =[0 0 0 0 0.6 0 0 0 0 0 0 0 0 0 a_z]^T
- $\hat{\zeta}_0$=$0_{15\times1}$

From figure 3(a) we can see that the drone was able to track the reference of 0.6 m inside the elevator, with the apparition of small peeks when the elevator accelerates or decelerates. In addition, we notice that the simulated altitude $x5_{Sim}$ is close to the experimentally measured one $x5_{Exp}$. The mismatch is due to mobilisation errors of the model implemented in Matlab. Figure 3 (b) compares the measured elevator acceleration $a_{z,Exp}$ with the estimated acceleration from the onboard EKF_UI $\hat{a}_{z,Exp}$ and the estimated acceleration using the implemented EKF-UI on Matlab $\hat{a}_{z,Sim}$. The estimated acceleration using the implemented EKF-UI is better than the one estimated using the onboard EKF-UI. However, we notice

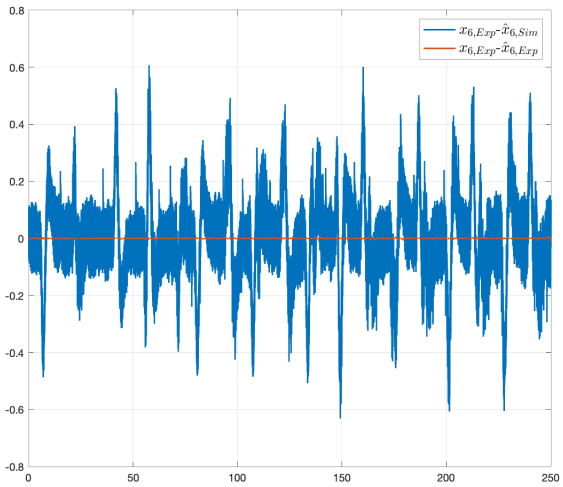

Fig. 4. Comparison between the velocity estimation error of the simulated EKF-UI using Matlab (in blue) and the implement onboard EKF-UI (in orange)

that in the last 50s the estimated acceleration using the onboard EKF-UI $\hat{a}_{z,Exp}$ is very close to the estimated acceleration using the implemented EKF-UI on Matlab $\hat{a}_{z,Sim}$. Despite the estimation error of the elevator acceleration, the closed-loop system converges to the desired reference inside the elevator. This is due to the fact that the sliding mode control law uses the a-priori set upper bound of the elevator acceleration instead of using its estimate.

Figure 4 illustrates the velocity estimation error from both the onboard EKF-UI and implement EKF-UI on Matlab. The estimation error provided by the onboard EKF-UI is almost equal to zero unlike the estimation provided by the implemented EKF-UI. This is explained by the fact that in the simulation the EKF-UI uses the output of the model which contains model uncertainties. On the other hand, the onboard EKF-UI directly uses the measured altitude. Therefore, velocity estimation is more accurate.

VIII. Conclusion

The present paper proposes a formulation for drone reference tracking in a GPS-denied moving environment. The moving environment is associated with a non-inertial frame and is assumed to have a transnational motion in the geostationary reference frame. Based on the relative motion principles, the drone motion in the non-inertial frame is derived. An Extended Kalman Filter with Unknown Inputs (EKF-UI) is then used to simultaneously estimate the states of the drone and the moving environment accelerations. A sliding mode controller is then designed and the control laws are computed to ensure the overall convergence of the closed loop system. The combined EKF-UI and Sliding Mode controller showed good performance in making the drone track the desired reference inside the moving environment in both simulation and experiment. Our future research in this area will focus on a general case with translational and rotational motions in the inertial frame of reference.

Acknowledgment

Research reported in this publication was supported by King Abdullah University of Science and Technology (KAUST) with the Base Research Funds (BAS/1/1627-01-01, BAS/1/1682-01-01), and KAUST AI-Initiative. The authors would also like to thank Mr. Olivier Toupet from the Jet Propulsion Laboratory for suggesting the idea for this research.

References

[1] Steven Rasmussen, Krishnamoorthy Kalyanam, Satyanarayana Manyam, David Casbeer, and Christopher Olsen. Practical considerations for implementing an autonomous, persistent, intelligence, surveillance, and reconnaissance system. In *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pages 1847–1854, 2017.

[2] Joshua Shaffer, Estefany Carrillo, and Huan Xu. Receding horizon synthesis and dynamic allocation of UAVs to fight fires. In *2018 IEEE Conference on Control Technology and Applications (CCTA)*, pages 138–145, 2018.

[3] Eric Cheng. *Aerial photography and videography using drones*. Peachpit Press, 2015.

[4] Daniel Câmara. Cavalry to the rescue: Drones fleet to help rescuers operations over disasters scenarios. In *2014 IEEE Conference on Antenna Measurements & Applications (CAMA)*, pages 1–4. IEEE, 2014.

[5] Yunus Karaca, Mustafa Cicek, Ozgur Tatli, Aynur Sahin, Sinan Pasli, Muhammed Fatih Beser, and Suleyman Turedi. The potential use of unmanned aircraft systems (drones) in mountain search and rescue operations. *The American journal of emergency medicine*, 36(4):583–588, 2018.

[6] Pere Molina, Ismael Colomina, Pedro Victoria, Jan Skaloud, Wolfgang Kornus, Rafael Prades, and Carmen Aguilera. Drones to the rescue! Technical report, 2012.

[7] UM Rao Mogili and BBVL Deepak. Review on application of drone systems in precision agriculture. *Procedia computer science*, 133:502–509, 2018.

[8] Pratap Tokekar, Joshua Vander Hook, David Mulla, and Volkan Isler. Sensor planning for a symbiotic UAV and UGV system for precision agriculture. *IEEE Transactions on Robotics*, 32(6):1498–1511, 2016.

[9] Shreeram Marathe. Leveraging drone based imaging technology for pipeline and RoU monitoring survey. In *SPE Symposium: Asia Pacific Health, Safety, Security, Environment and Social Responsibility*. OnePetro, 2019.

[10] Oswaldo Menéndez, Marcelo Pérez, and Fernando Auat Cheein. Visual-based positioning of aerial maintenance platforms on overhead transmission lines. *Applied Sciences*, 9(1):165, 2019.

[11] Jarrod C Hodgson, Rowan Mott, Shane M Baylis, Trung T Pham, Simon Wotherspoon, Adam D Kilpatrick, Ramesh Raja Segaran, Ian Reid, Aleks Terauds, and Lian Pin Koh. Drones count wildlife more accurately and precisely than humans. *Methods in Ecology and Evolution*, 9(5):1160–1167, 2018.

[12] Julie Linchant, Jonathan Lisein, Jean Semeki, Philippe Lejeune, and Cédric Vermeulen. Are unmanned aircraft systems (UAS s) the future of wildlife monitoring? a review of accomplishments and challenges. *Mammal Review*, 45(4):239–252, 2015.

[13] Qiuhong Wang, Jingjing Gu, Haitao Huang, and Yanchao Zhao. Online drone-based moving target detection system in dense-obstructer environment. In *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*, pages 860–867. IEEE, 2018.

[14] Tyler Wall and Torin Monahan. Surveillance and violence from afar: The politics of drones and liminal security-scapes. *Theoretical criminology*, 15(3):239–254, 2011.

[15] Gregory S McNeal. Drones and the future of aerial surveillance. *Geo. Wash. L. Rev.*, 84:354, 2016.

[16] Milan Erdelj, Enrico Natalizio, Kaushik R Chowdhury, and Ian F Akyildiz. Help from the sky: Leveraging UAVs for disaster management. *IEEE Pervasive Computing*, 16(1):24–32, 2017.

[17] Geoffrey Ling and Nicole Draghic. Aerial drones for blood delivery. *Transfusion*, 59(S2):1608–1611, 2019.

[18] Evan Ackerman and Eliza Strickland. Medical delivery drones take flight in East Africa. *IEEE Spectrum*, 55(1):34–35, 2018.

[19] The Action Lab. What Happens If You Fly a Drone In An Elevator? Real Experiment!, Youtube, 2019. [Video file]. Available: https://www.youtube.com/watch?v=DUGwdcgi2L8] [Accessed 16-Feb-2021].

[20] Ken Heron. Hovering Drone in a Van - Will it move with it? - KEN HERON, Youtube, 2018. [Video file]. Available: https://www.youtube.com/watch?v=LEgJBDxr3uU] [Accessed 16-Feb-2021].

[21] The Action Lab. If You Fly a Drone in a Car, Does it Move With It? (Dangerous In-Car Flight Challenge), Youtube, 2017. [Video file]. Available: https://www.youtube.com/watch?v=XjTj-tGPSWE] [Accessed 16-Feb-2021].

[22] Sinatra314. Flying a drone in an elevator. What happens?, Youtube, 2020. [Video file]. Available: https://www.youtube.com/watch?v=kvazr0W8-Bc] [Accessed 16-Feb-2021].

[23] DroningON. DroningON — Drone In An Elevator/Lift - Ryze/DJI Tello Experiment, Youtube, 2018. [Video file]. Available: https://www.youtube.com/watch?v=vbgHPV4G0Sc] [Accessed 16-Feb-2021].

[24] Qi Lu, Beibei Ren, and Siva Parameswaran. Shipboard landing control enabled by an uncertainty and disturbance estimator. *Journal of Guidance, Control, and Dynamics*, 41(7):1502–1520, 2018.

[25] Marco Tognon, Sanket S. Dash, and Antonio Franchi. Observer-based control of position and tension for an aerial robot tethered to a moving platform. *IEEE Robotics and Automation Letters*, 1(2):732–737, 2016.

[26] Fady Alami, Abdulrahman Hussian, and Naim Ajlouni. Design and implementation of a multi-stage PID controller for non-inertial referenced UAV. *Electrica*, 20(2):199–207, 2020.

[27] Hakim Bouadi, M. Bouchoucha, and M. Tadjine. Modelling and stabilizing control laws design based on backstepping for an uav type-quadrotor. *IFAC Proceedings Volumes*, 40:245–250, 12 2007.

[28] Samir Bouabdallah and Roland Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. volume 2005, pages 2247–2252, 01 2005.

[29] Abdelhamid Chriette. Contribution à la commande et à la modélisation des hélicoptères : asservissement visuel et commande adaptative. 2001.

[30] Matko Orsag and Stjepan Bogdan. Hybrid control of quadrotor. In *2009 17th Mediterranean Conference on Control and Automation*, pages 1239–1244, 2009.

[31] Ron J. Patton, Paul M. Frank, and Robert N. Clarke. *Fault Diagnosis in Dynamic Systems: Theory and Application*. Prentice-Hall, Inc., USA, 1989.

[32] Ron Patton, P. Frank, and R. Clark. *Issue of Fault Diagnosis for Dynamic Systems*. 01 2000.

[33] A.M. Pertew, H.J. Marquez, and Q. Zhao. Design of unknown input observers for Lipschitz nonlinear systems. In *Proceedings of the 2005, American Control Conference, 2005.*, pages 4198–4203 vol. 6, 2005.

[34] Shuwen Pan. *System identification and damage detection of structures with unknown excitations*. PhD thesis, 2006.

[35] Daniel Simon. Optimal state estimation: Kalman, H-infinity, and nonlinear approaches. 01 2006.

[36] Sarah Mechhoud, Emmanuel Witrant, Luc Dugard, and Didier Moreau. Estimation of heat source term and thermal diffusion in Tokamak plasmas using a Kalman filtering method in the early lumping approach. *IEEE Transactions on Control Systems Technology*, 23(2):449–463, 2015.

[37] Junn Loo, Ze Yang Ding, Evan Davies, Surya Nurzaman, and Chee Pin (Edwin) Tan. Curvature and force estimation for a soft finger using an EKF with Unknown Input Optimization. 07 2020.

[38] Zehor Belkhatir, S. Mechhoud, and Taous-Meriem Laleg-Kirati. Kalman filter based estimation algorithm for the characterization of the spatiotemporal hemodynamic response in the brain. *Control Engineering Practice*, 89:180–189, 08 2019.

[39] S. Bouabdallah, A. Noth, and R. Siegwart. PID vs LQ control techniques applied to an indoor micro quadrotor. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2451–2456 vol.3, 2004.

[40] Abdelhamid Tayebi and S. McGilvray. Attitude stabilization of a four-rotor aerial robot. volume 2, pages 1216 – 1221 Vol.2, 01 2005.

[41] Ian D. Cowling, Oleg A. Yakimenko, James F. Whidborne, and Alastair K. Cooke. A prototype of an autonomous controller for a quadrotor UAV. In *2007 European Control Conference (ECC)*, pages 4001–4008, 2007.

[42] Juan-Pablo Afman, Eric Feron, and John Hauser. Nonlinear maneuver regulation for reduced-G atmospheric flight. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 731–736, 2018.

[43] Juan-Pablo Afman, Eric Feron, and John Hauser. Triple-integral control for reduced-G atmospheric flight. In *2018 Annual American Control Conference (ACC)*, pages 392–397, 2018.

[44] Siddhardha Kedarisetty and Joel George Manathara. Acceleration control of a multi-rotor UAV towards achieving microgravity. *Aerospace Systems*, 2(2):175–188, 2019.

[45] Samir Bouabdallah and Roland Siegwart. Backstepping and sliding-mode techniques applied to an indoor micro quadrotor. volume 2005, pages 2247–2252, 01 2005.

[46] Saif A. Al-Hiddabi. Quadrotor control using feedback linearization with dynamic extension. In *2009 6th International Symposium on Mechatronics and its Applications*, pages 1–3, 2009.

[47] T. Madani and A. Benallegue. Backstepping sliding mode control applied to a miniature quadrotor flying robot. In *IECON 2006 - 32nd Annual Conference on IEEE Industrial Electronics*, pages 700–705, 2006.

[48] Tao Jiang, Tao Song, and Defu Lin. Integral sliding mode based control for quadrotors with disturbances: Simulations and experiments. *International Journal of Control, Automation and Systems*, 17, 05 2019.

[49] PX4 autopilot. Px4 system architecture, March 2021. Available: https://docs.px4.io/master/en/concept/px4_systems_architecture.html [Accessed 18-Feb-2022].

[50] Jean-Jacques E Slotine and Weiping Li. *Applied nonlinear control*. 1991.