Trajectory Tracking with a Quaternion-based Quadcopter Model

Victor Freire

***Abstract*— We propose two linearization methods for the quaternion-based quadcopter model: Feedback linearization leveraging the differential flatness property of the model and equilibrium linearization at the static hover operating point. We analyze the properties of each model, including stability and controllability. We then design two trajectory tracking controllers for each linearized model: A Lyapunov-based stabilizing controller and a LQR-based optimal controller. Finally, we compare the performance of each controller tracking mild and aggressive trajectories in simulation.**

I. Introduction

Quadcopters have gained popularity very rapidly in the last couple of decades thanks to the development of small, cheap, lightweight and powerful computers [1]–[4]. These systems are also attractive to the nonlinear control community because they facilitate the practical verification of many nonlinear control algorithms [5], [6]. Quadcopters are underactuated, highly nonlinear systems consisting of four rotors that generate thrust in the same direction. Most of the nonlinearities enter the system model through the spatial orientation (attitude) parameterization; Euler angles ($\mathbb{R}^3$), quaternions ($\mathbb{R}^4$) and rotation matrices ($\mathbb{R}^{3\times3}$) are the most commonly used attitude descriptions [7]. While Euler angles are arguably the most intuitive, they suffer from singularities (i.e. the map from Euler angles to orientations is not one-to-one). Quaternions suffer from ambiguity because they double-cover SO(3), the group of all possible rotations. This means that any one orientation can always be described by two different quaternions. Finally, rotation matrices result in computationally expensive operations because they require 9 parameters. We conclude that the most efficient, singularity-free representation of quadcopter orientations is the unit quaternion and note that the ambiguity must be handled explicitly. However, the literature rarely considers the quaternion-based model of quadcopters when designing controllers because they are relatively hard to work with as they form a noncommutative ring (such as square matrices) but standard linear algebra (which most numerical software uses) does not apply to them easily. With this work, we aim to demistify the quaternion-based quadcopter model by transforming it to a linear system with different techniques. We then use linear control techniques to design trajectory tracking controllers for the linearized models. Finally, we explore the performance of each controller in the real system in simulation.

The rest of the paper is organized as follows: Section II introduces the nonlinear quadcopter model considered and presents two methods to linearize it: Feedback linearization and linearization around an equilibrium point. Section III analyzes the stability and controllability properties of the linearized systems. Section IV designs controllers for the linearized models. Section V compares the performance of the designed controllers in simulation and Section VI concludes the paper.

Victor Freire is with the Department of Mechanical Engineering at the University of Wisconsin-Madison {freiremelgiz}@wisc.edu

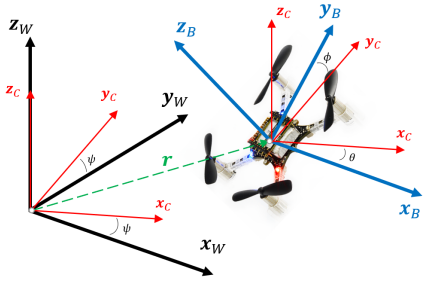

Fig. 1. World (black) coordinate frame W and Body (blue) coordinate frame B. The intermediate (red) coordinate frame C is also shown. Crazyflie figure from [8].

II. System Description & Linear Model

We introduce the considered quadcopter model, which is nonlinear, and present two methods to linearize it.

A. Quadcotper Dynamics

The equations of motion for a quadcopter system are given by [7], [9], [10]:

$$\ddot{\mathbf{r}} = T\mathbf{q}\mathbf{z}_W\mathbf{q}^* - g\mathbf{z}_W, \tag{1a}$$

$$\dot{\mathbf{q}} = \frac{1}{2}\mathbf{q}\boldsymbol{\omega}, \tag{1b}$$

where $\mathbf{r} = (x, y, z)^T$ is the position vector, T is the normalized total thrust, $\mathbf{q}$ is a quaternion (see e.g. [11] for a review of quaternion algebra) describing the orientation of the quadcopter, $\boldsymbol{\omega} = (p, q, r)^T$ is the body angular velocity vector and $\mathbf{z}_W$ is the inertial frame's z-coordinate axis (see Figure 1). Let the state and input vectors be defined, respectively, as:

$$\mathbf{x} = \begin{bmatrix} x & y & z & q_0 & q_1 & q_2 & q_3 & \dot{x} & \dot{y} & \dot{z} \end{bmatrix}^T, \tag{2a}$$

$$\mathbf{u} = \begin{bmatrix} T & p & q & r \end{bmatrix}^T. \tag{2b}$$

In the following, we will assume that the state can be completely measured (i.e., $\mathbf{y}(t) = \mathbf{x}(t)$). Then, the system (1) can be shown to be control affine:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t)) + g(\mathbf{x}(t))\mathbf{u}(t). \tag{3}$$

B. Feedback Linearization

From [12], system (1) is *differentially flat* [13]. Let $\boldsymbol{\sigma}$ be the *flat output* vector:

$$\boldsymbol{\sigma} = \begin{bmatrix} x & y & z & q_3 \end{bmatrix}^T. \tag{4}$$

This means that we can find an algebraic function:

$$(\mathbf{x}, \mathbf{u}) = \Phi(\boldsymbol{\sigma}, \dot{\boldsymbol{\sigma}}, \ddot{\boldsymbol{\sigma}}, \dddot{\boldsymbol{\sigma}}), \tag{5}$$

given as in [9], [14]. This algebraic mapping allows us to define *flat state* and *flat input* vectors:

$$\mathbf{z} = \begin{bmatrix} x & y & z & \dot{x} & \dot{y} & \dot{z} & \ddot{x} & \ddot{y} & \ddot{z} & q_3 \end{bmatrix}^T, \tag{6a}$$

$$\mathbf{v} = \begin{bmatrix} \ddot{x} & \ddot{y} & \ddot{z} & \dddot{x} & \dddot{y} & \dddot{z} & \dot{q}_3 \end{bmatrix}^T, \tag{6b}$$

which follow the linear dynamics:

$$\dot{\mathbf{z}}(t) = A_f \mathbf{z}(t) + B_f \mathbf{v}(t), \tag{F-LTI}$$

where

$$A_f = \begin{bmatrix} \mathbf{0}_{3\times3} & I_3 & \mathbf{0}_{3\times4} \\ \mathbf{0}_{7\times3} & \mathbf{0}_{7\times3} & \mathbf{0}_{7\times4} \end{bmatrix}, \quad B_f = \begin{bmatrix} \mathbf{0}_{3\times7} \\ I_7 \end{bmatrix}.$$

The controller for the original system can then be recovered applying the flat map:

$$(\mathbf{x}, \mathbf{u}) = \Phi(\mathbf{z}, \mathbf{v}) \tag{7}$$

C. Equilibrium Point Linearization

It can be shown that static hovering:

$$\mathbf{x}_e = \begin{bmatrix} x_e & y_e & z_e & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T, \tag{8a}$$

$$\mathbf{u}_e = \begin{bmatrix} g & 0 & 0 & 0 \end{bmatrix}^T, \tag{8b}$$

for any $m_e \in \mathbb{R},\ m \in \{x, y, z\}$, is an *equilibrium point* of system (1). We can then linearize the system around this operating point to arrive at:

$$\delta\dot{\mathbf{x}}(t) = A_e \delta\mathbf{x}(t) + B_e \delta\mathbf{u}(t), \tag{E-LTI}$$

where $\delta\mathbf{x} \triangleq \mathbf{x} - \mathbf{x}_e$, $\delta\mathbf{u} \triangleq \mathbf{u} - \mathbf{u}_e$ and the matrices are given as:

$$A_e = \frac{\partial f}{\partial \mathbf{x}}(\mathbf{x}_e, \mathbf{u}_e), \quad B_e = g(\mathbf{x}_e). \tag{9}$$

Specifically, they have the form:

$$A_e = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times4} & I_3 \\ \mathbf{0}_{4\times3} & \mathbf{0}_{4\times4} & \mathbf{0}_{4\times3} \\ \mathbf{0}_{3\times3} & M_A & \mathbf{0}_{3\times3} \end{bmatrix}, \quad B_e = \begin{bmatrix} \mathbf{0}_{3\times1} & \mathbf{0}_{3\times3} \\ \mathbf{0}_{4\times1} & M_B \\ \mathbf{z}_W & \mathbf{0}_{3\times3} \end{bmatrix},$$

where

$$M_A = 2g \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad M_B = \frac{1}{2} \begin{bmatrix} \mathbf{0}_{1\times3} \\ I_3 \end{bmatrix},$$

and $\mathbf{z}_W = (0, 0, 1)^T$ is the inertial frame's z-axis (Figure 1).

III. System Analysis

We analyze the properties (such as stability and controllability) of the linearized models (F-LTI) and (E-LTI).

A. Stability

The eigenvalues of A_f and A_e are all 0. This means the systems (F-LTI) and (E-LTI) are *unstable* [15] because their $A_m, m \in \{f, e\}$ matrices have a Jordan block $J \in \mathbb{R}^{10\times10}$.

B. Controllability

Let the controllability matrix for systems (F-LTI) and (E-LTI) defined as:

$$C_m = \begin{bmatrix} B_m & A_m B_m & A_m^2 B_m & \cdots & A_m^9 B_m \end{bmatrix}, \tag{10}$$

with $m \in \{f, e\}$. It can be shown that $\text{rank}(C_f) = 10$. Therefore, the system (F-LTI) is *controllable* [15]. However, $\text{rank}(C_e) = 9$. To gain more insight into which modes of (E-LTI) are uncontrollable, we put the system in its *controllable form*. Let T_e be an orthogonal matrix such that matrices:

$$\overline{A}_e = T_e A_e T_e^T, \quad \overline{B}_e = T_e B_e, \tag{11}$$

have the form:

$$\overline{A}_e = \begin{bmatrix} A_u^e & \mathbf{0}_{1\times9} \\ A_{cu}^e & A_c^e \end{bmatrix}, \quad \overline{B}_e = \begin{bmatrix} \mathbf{0}_{1\times4} \\ B_c^e \end{bmatrix},$$

where the pair (A_c^e, B_c^e) is *controllable*. Then, the transformed state is $\delta\overline{\mathbf{x}} \triangleq T_e \delta\mathbf{x}$ and it evolves according to:

$$\delta\dot{\overline{\mathbf{x}}}(t) = \overline{A}_e \delta\overline{\mathbf{x}}(t) + \overline{B}_e \delta\mathbf{u}(t). \tag{12}$$

The first entry of $\delta\overline{\mathbf{x}}$ represents the uncontrollable mode of the system. In general, this will be a linear combination of the original states. However, from the structure of the obtained T_e, we have that the uncontrollable mode is exactly $\delta q_0 \triangleq q_0 - 1$ by the choice of the equilibrium point $\mathbf{x}_e$.

C. Stabilizability

Since the system (F-LTI) is *controllable*, it follows that it is also *stabilizable* [15]. We now examine the uncontrollable mode of (E-LTI). Recover from (11) and the found, orthogonal T_e, that $A_u^e = 0$ is a scalar. Therefore, the uncontrollable mode is *stable in the sense of Lyapunov* [15]. However, the system (E-LTI) is not *stabilizable*.

D. Algebraic Constraint

By considering only *unit quaternions* in system (1), we can add the algebraic constraint:

$$\|\mathbf{q}\|_2 \triangleq \sqrt{q_0^2 + q_1^2 + q_2^2 + q_3^2} = 1, \tag{13}$$

effectively reducing the state space dimension from 10 to 9 states. With this algebraic constraint, the uncontrollable mode δq_0 is completely determined. Therefore, we will restrict ourselves to designing a controller for

$$\delta\dot{\overline{\mathbf{x}}}_c(t) = A_c^e \delta\overline{\mathbf{x}}_c(t) + B_c^e \delta\mathbf{u}(t), \tag{14}$$

noting that

$$\delta\overline{\mathbf{x}}_c = S_c T_e (\mathbf{x} - \mathbf{x}_e), \quad S_c = \begin{bmatrix} \mathbf{0}_{9\times1} & I_9 \end{bmatrix}.$$

Controlling the above system is equivalent to controlling (E-LTI). Therefore, in the following, we use the label (E-LTI) to refer to either system.

IV. Control Design

Suppose we have trajectory $\mathbf{x}_r(t)$, $\mathbf{u}_r(t)$ (obtained, for example, as in [10]) for system (1). In this section, we design trajectory tracking controllers for (F-LTI) and (E-LTI).

A. Lyapunov-based Control

We design static gain for full-state feedback according to the following result:

Lemma 1 [15] *For any $\mu > 0$, if the pair (A, B) is* controllable*, then one can find state feedback $\mathbf{u} = -K\mathbf{x}$ such that the eigenvalues λ_i of the closed-loop system $\dot{\mathbf{x}} = (A - BK)\mathbf{x}$ have real part $Re\{\lambda_i(A - BK)\} \leq -\mu$, $\forall i$.*

The proof of Lemma 1 provides a constructive method for designing exponentially stabilizing feedback controllers. The reader is referred to [15] for more details. Since the pairs (A_f, B_f) and (A_c^e, B_c^e) are *controllable*, we can use Lemma 1 to design controllers that render the respective closed-loop systems *exponentially stable*. Furthermore, we can stabilize the system at the current point in the state trajectory $\mathbf{x}_r(t)$ instead of the origin to obtain a trajectory tracking controller. We first design a controller for (F-LTI):

$$\begin{aligned} \mathbf{v}(t) &= -K_f\big(\mathbf{z}(t) - \mathbf{z}_r(t)\big), \qquad \text{(LYAP-F)} \\ K_f &= \frac{1}{2}B_f^T P_f, \end{aligned}$$

where $P_f \succ 0$ is the solution to the Lyapunov equation:

$$P_f A_f + A_f^T P_f - P_f B_f B_f^T P_f = -2\mu P_f, \qquad (15)$$

for some $\mu > 0$. Note that since the trajectory $\mathbf{x}_r(t)$, $\mathbf{u}_r(t)$ is dynamically feasible [10], we can exactly extract $\ddot{\mathbf{r}}_r(t)$ from (1a), which is needed to extract the system input with

$$\mathbf{u}(t) = \Phi(\mathbf{z}(t), \mathbf{v}(t)). \qquad (16)$$

Next, we design a controller for (E-LTI):

$$\begin{aligned} \mathbf{u}(t) &= \mathbf{u}_e - K_e S_c T_e\big(\mathbf{x}(t) - \mathbf{x}_r(t)\big), \qquad \text{(LYAP-E)} \\ K_e &= \frac{1}{2}(B_c^e)^T P_e, \end{aligned}$$

where $P_e \succ 0$ is the solution to the Lyapunov equation:

$$P_e A_c^e + (A_c^e)^T P_e - P_e B_c^e (B_c^e)^T P_e = -2\mu P_e. \qquad (17)$$

B. Optimal Control

Consider the well-known infinite-horizon LQR problem [15]:

$$\begin{aligned} \min_{\mathbf{x}(t),\mathbf{u}(t)} \quad & \int_0^\infty \mathbf{x}^T(t) Q \mathbf{x}(t) + \mathbf{u}^T(t) R \mathbf{u}(t)\, \mathrm{d}t \qquad \text{(LQR)} \\ \text{s.t.} \quad & \dot{\mathbf{x}}(t) = A\mathbf{x}(t) + B\mathbf{u}(t), \end{aligned}$$

with weight matrices $Q \succeq 0$ and $R \succ 0$. The solution of (LQR) optimally and exponentially stabilizes a system to the origin. It is known [15] that the optimal control input solving (LQR) has the form:

$$\mathbf{u}^*(t) = R^{-1} B^T P \mathbf{x}(t), \qquad (18)$$

where $P \succ 0$ is the unique solution of the Continuous-time Algebraic Riccati Equation:

$$A^T P + PA + Q - PBR^{-1}BP = 0. \qquad \text{(CARE)}$$

While the above framework stabilizes the system to the origin, we are interested in tracking a trajectory. The general approach will involve a change of coordinates:

$$\tilde{\mathbf{x}}(t) \triangleq \mathbf{x}_r(t) - \mathbf{x}(t). \qquad (19)$$

We first design an optimal controller for (F-LTI):

$$\begin{aligned} \mathbf{v}(t) &= -K_f(\mathbf{z}(t) - \mathbf{z}_r(t)), \qquad \text{(LQR-F)} \\ K_f &= R_f^{-1} B_f^T P_f, \end{aligned}$$

where $P_f \succ 0$ is the solution to the CARE:

$$A_f^T P_f + P_f A_f + Q_f - P_f B_f R_f^{-1} B_f P_f = 0, \qquad (20)$$

for given matrices $Q_f \succeq 0$ and $R_f \succ 0$ encoding performance specifications. Next, we design an optimal controller for (E-LTI):

$$\begin{aligned} \mathbf{u}(t) &= \mathbf{u}_e - K_e S_c T_e\big(\mathbf{x}(t) - \mathbf{x}_r(t)\big), \qquad \text{(LQR-E)} \\ K_e &= R_e^{-1}(B_c^e)^T P_e, \end{aligned}$$

where $P_e \succ 0$ is the solution to the CARE:

$$(A_c^e)^T P_e + P_e A_c^e + Q_e - P_e B_c^e R_e^{-1} B_c^e P_e = 0, \qquad (21)$$

for given matrices $Q_e \succeq 0$ and $R_e \succ 0$ encoding performance specifications.

V. Simulation Experiments

We compare the performance of the presented controllers when applied to the real nonlinear dynamics (1) by means of MATLAB simulation. We consider two different trajectories obtained as in [10]. The first trajectory is a mild flight lasting 30 seconds and is exactly the same trajectory as in Example 1 of [10]. The second trajectory is an aggressive trace of the letter "W" in space lasting a total of 12 seconds. We will quantitatively describe performance by the Integrated Absolute Error (IAE) metric for the position vector $\mathbf{r}(t)$. The IAE metric is defined as:

$$\text{IAE} \triangleq \int_0^{t_f} \|\mathbf{r}(t) - \mathbf{r}_r(t)\|_2 \, \mathrm{d}t, \qquad (22)$$

where $\mathbf{r}_r(t)$ is the reference trajectory (desired) position vector. In practice, we approximate this quantity with a discrete sum at a sampling time of $T_s = 0.01$ seconds:

$$\text{IAE} \approx \sum_{k=0}^{\lfloor t_f/T_s \rfloor} \|\mathbf{r}(kT_s) - \mathbf{r}_r(kT_s)\|_2. \qquad (23)$$

We choose parameter $\mu = 2$ for controllers (LYAP-F) and (LYAP-E). The choice of weight matrices in the LQR-based controllers is:

$$\begin{aligned} Q_f &= \text{diag}\big(200, 200, 200, 50, 50, 50, 10, 10, 10, 20\big), \\ R_f &= \text{diag}\big(0.1, 0.1, 0.1, 0.01, 0.01, 0.01, 0.05\big), \\ Q_e &= \text{diag}\big(100, 100, 100, 50, 50, 10, 10, 10, 50\big), \\ R_e &= \text{diag}\big(0.01, 0.1, 0.1, 0.1\big). \end{aligned}$$

Figure 2 shows the performance of the Lyapunov-based controllers (LYAP-F) and (LYAP-E) tracking the mild trajectory. We observe that the equilibrium point linearization outperforms the feedback linearization performance. One possible cause is that the mild trajectory remains near the linearization equilibrium point. Therefore, the linearization remains valid for the entire duration of the trajectory.

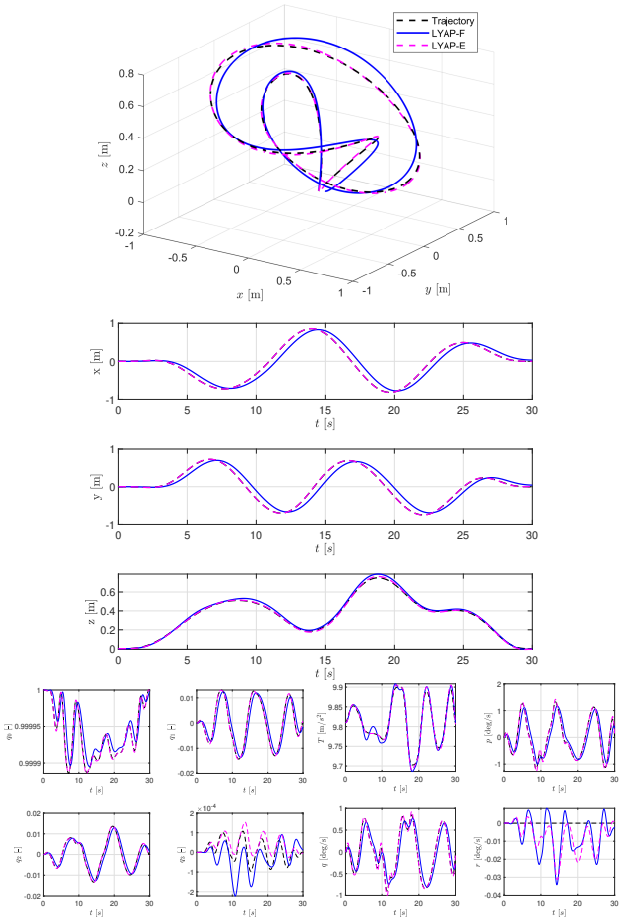

Fig. 2. Simulation results for the Lyapunov-based controllers with $\mu = 2$ tracking a mild trajectory.

Figure 3 shows the performance of the same controllers tracking instead the aggressive trajectory. While the performance of (LYAP-E) degrades significantly because the aggressive trajectory steers the system far from the linearization point, the feedback linearization controller (LYAP-F) also fails to achieve satisfactory tracking. However, notice that the z-axis position tracking for (LYAP-E) is lower in general than that of (LYAP-F). This shows the inability of (LYAP-E) to completely decouple height control from attitude control. A particularly interesting case is that when this trajectory is tracked with $\mu = 1$ (simulation not shown in figures), the controller (LYAP-E) results in the system reaching negative heights ($z \leq 0$) which, in most cases, indicates crashing with the floor. (LYAP-F) does not suffer from this.

Figure 4 shows the performance of the optimal controllers (LQR-F) and (LQR-E) tracking the mild trajectory. In this

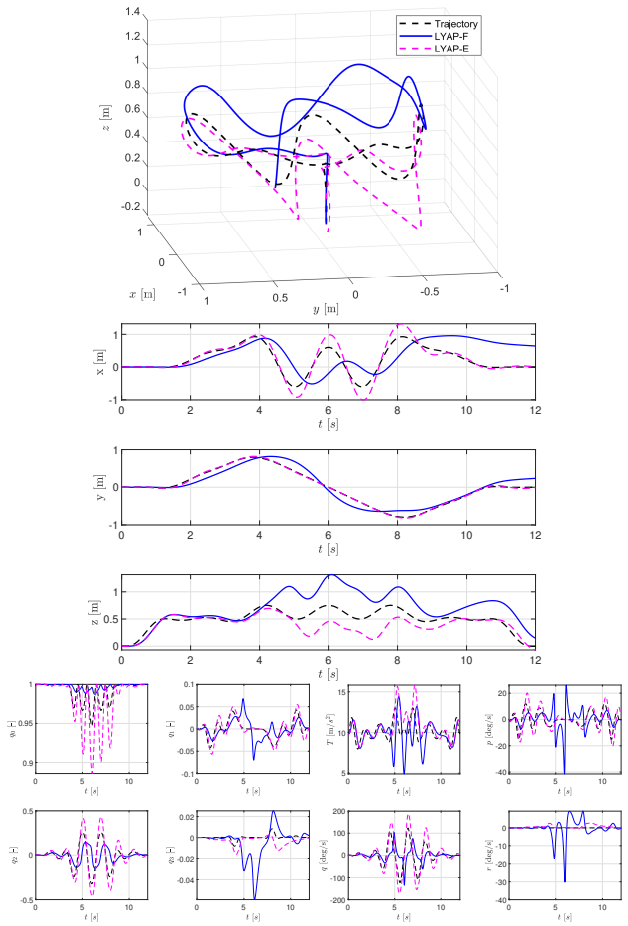

Fig. 3. Simulation results for the Lyapunov-based controllers with $\mu = 2$ tracking an aggressive trajectory.

case, both controllers achieve nearly perfect tracking. Figure 5 shows the performance of the same optimal controllers tracking the aggressive trajectory instead. Surprisingly, the equilibrium linearization controller (LQR-E) still achieves nearly perfect tracking despite the trajectory demanding pitch angles of up to 40 degrees (very far from the linearization point). While the performance of the feedback linearization controller (LQR-F) is acceptable, it is outclassed by (LQR-E).

TABLE I
IAE VALUES FOR EACH CONTROLLER.

Controller	(LYAP-F)	(LYAP-E)	(LQR-F)	(LQR-E)
Mild Traj.	4.68	0.22	0.31	0.018
Aggressive Traj.	5.69	2.14	0.83	0.11

Table I provides the calculated IAE values for each simulation performed. As expected, the controllers achieve lower IAE in the mild trajectory. Also noteworthy is that the controllers based on (E-LTI) always outperformed those based on (F-LTI).

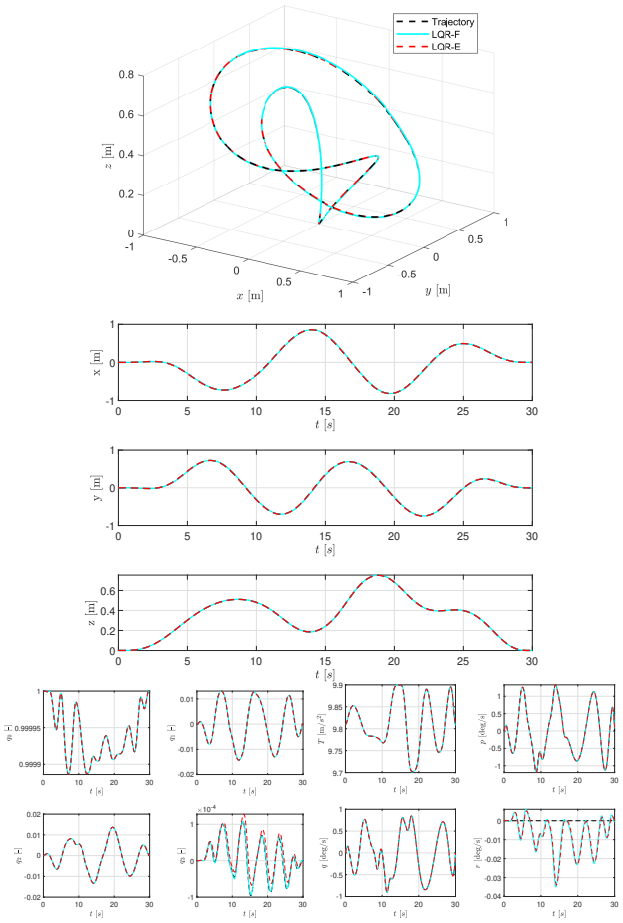

Fig. 4. Simulation using the LQR-based controllers tracking a mild trajectory.

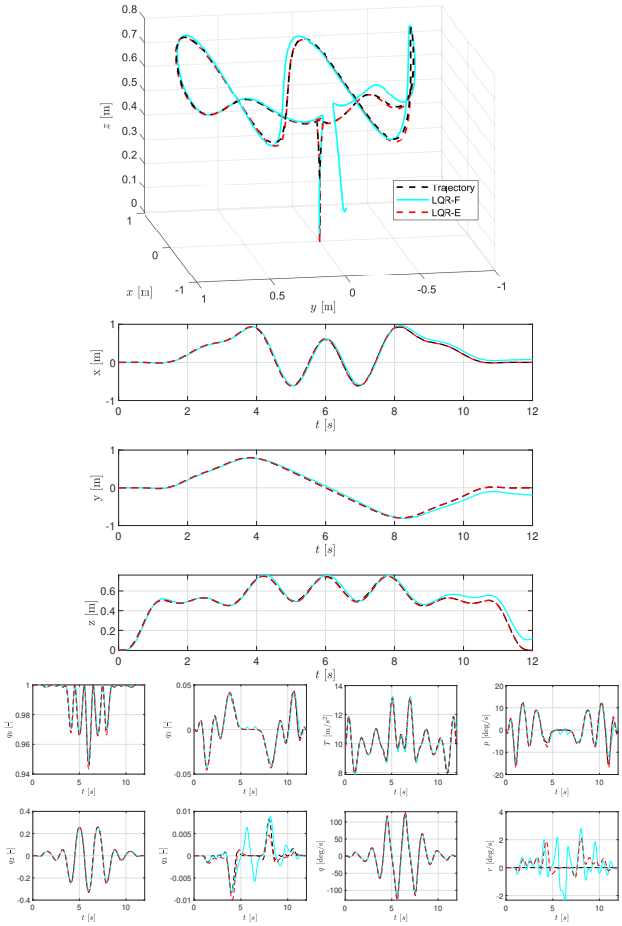

Fig. 5. Simulation using the LQR-based controllers tracking an aggressive trajectory.

VI. Conclusion

We presented two methods for linearizing the quaternion-based quadcopter dynamic model. We also designed trajectory tracking controllers for the linearized models and compared their performance while tracking different trajectories through simulation.

References

[1] G. Rousseau, C. S. Maniu, S. Tebbani, M. Babel, and N. Martin, "Minimum-time b-spline trajectories with corridor constraints. application to cinematographic quadrotor flight plans," *Control Engineering Practice*, vol. 89, pp. 190–203, 2019.

[2] L. Apvrille, T. Tanzi, and J.-L. Dugelay, "Autonomous drones for assisting rescue services within the context of natural disasters," in *2014 XXXIth URSI General Assembly and Scientific Symposium (URSI GASS)*, 2014, pp. 1–4.

[3] J. Navia, I. Mondragon, D. Patino, and J. Colorado, "Multispectral mapping in agriculture: Terrain mosaic using an autonomous quadcopter uav," in *2016 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, 2016, pp. 1351–1358.

[4] S. Tang, V. Wüest, and V. Kumar, "Aggressive flight with suspended payloads using vision-based control," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1152–1159, 2018.

[5] P. Foehn and D. Scaramuzza, "Onboard state dependent lqr for agile quadrotors," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6566–6572.

[6] D. Lee, H. J. Kim, and S. Sastry, "Feedback linearization vs. adaptive sliding mode control for a quadrotor helicopter," *International Journal of control, Automation and systems*, vol. 7, no. 3, pp. 419–428, 2009.

[7] D. Brescianini, M. Hehn, and R. D'Andrea, "Nonlinear Quadrocopter Attitude Control: Technical Report," ETH Zurich, Tech. Rep., 2013, medium: application/pdf,Online-Ressource. [Online]. Available: http://hdl.handle.net/20.500.11850/154099

[8] "Px4 open source project. image of crazyflie." [Online]. Available: https://docs.px4.io/v1.12/en/complete_vehicles/crazyflie21.html

[9] D. Mellinger and V. Kumar, "Minimum snap trajectory generation and control for quadrotors," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2520–2525.

[10] V. Freire and X. Xu, "Flatness-based quadcopter trajectory planning and tracking with continuous-time safety guarantees," *arXiv preprint arXiv:2111.00951*, 2021.

[11] F. Zhang, "Quaternions and matrices of quaternions," *Linear algebra and its applications*, vol. 251, pp. 21–57, 1997.

[12] H. T. Nguyen, T. Nguyen, I. Prodan, and F. Pereira, "Trajectory tracking for a multicopter under a quaternion representation," *IFAC Papers Online*, vol. 53, pp. 5731–5736, 04 2021.

[13] M. Fliess, J. Lévine, P. Martin, and P. Rouchon, "Flatness and defect of non-linear systems: introductory theory and examples," *International journal of control*, vol. 61, no. 6, pp. 1327–1361, 1995.

[14] D. Zhou and M. Schwager, "Vector field following for quadrotors using differential flatness," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 6567–6572.

[15] J. P. Hespanha, *Linear systems theory*. Princeton university press, 2018.