Quasistatic state feedback output tracking: PX4 SITL validation

Zifei Jiang[1], Mohamed Al Lawati[1,2], Arash Mohammadhasani[1] and Alan Lynch[1*]

[1*]Department of Electrical and Computer Engineering, University of Alberta, 9211 116 Street NW, Edmonton, T6G 1H9, Alberta, Canada.
[2]Department of Mechanical and Industrial Engineering, Sultan Qaboos University, Al Khoud, 123, Muscat, Oman.

*Corresponding author(s). E-mail(s): alan.lynch@ualberta.ca;
Contributing authors: zifei.jiang@ualberta.ca; maallawa@ualberta.ca;
Arash.Mhasani@ualberta.ca;

Abstract

This paper presents an algorithm for designing a quasi-static state feedback control for differentially flat nonlinear dynamics. This design exactly linearizes the closed-loop in a new state coordinates. The linearizing feedback has an important static dependence as it does not require dynamics in the controller. We apply the algorithm to a Slung Load System (SLS) which is a flat system consisting of a multirotor UAV and a suspended payload. After linearization, a straightforward output tracking control is used to ensure that error dynamics in the design coordinates are linear and exponentially stable. The open-source PX4 autopilot is tested with Software-in-the-Loop (SITL) to simulate the procedure. An automation pipeline is also presented for transforming a theoretical symbolic controller into a real-world excutable on embedded device in C++.

Keywords: keyword1, Keyword2, Keyword3, Keyword4

1 Introduction

Recently, interest has increased in the application of unmanned aerial vehicles (UAVs) to transport loads. A summary of the recent developments can be found in [1]. One method of transporting payloads is a multirotor UAV Slung Load System

(SLS), in which a cable bolted to the UAV's bottom suspends the load. The ability to maneuver the system and maintain a safe distance between the payload and the vehicle are just a few of the advantages of SLSs. SLS also offers a reliable way to move loads that can be used with different types of vehicles. Despite the fact that SLSs are a desirable option for load transportation, creating a high-performance and rigorous motion control system is difficult due to the intricate, underactuated nonlinear dynamics involved. The creation of a UAV SLS fits into a larger recent trend known as "unmanned aerial manipulation" that focuses on interacting with the environment [1–3].

The concept of differential flatness was first presented in [4] and refers to a property of nonlinear systems similar to controllability in a linear system. A survey of results is in [5]. The nonlinear SLS model has been demonstrated in [6] to be a flat system under common modelling assumptions. The so-called flat outputs of a flat system differentially parameterize the input and state. In other words, the state and input can be expressed in terms of the time derivatives of the flat output. Motion planning or open-loop control design frequently employs this parameterization. Every flat system is known to be linearizable by endogenous dynamic feedback [4]. Endogenous dynamic feedback is a type of dynamic state feedback in which controller state and auxiliary input can be described as functions of system state, system input, and their time derivatives. The flatness attribute is kept under this form of feedback [7]. By using an equivalence transformation, any endogenous dynamic feedback controller can be converted into a quasi-static controller [8]. The advantage of quasi-static feedback is that it can achieve linearization without adding controller states to the closed-loop dynamics, unlike dynamic state feedback.

There are two aspects of recent efforts in flatness-based control research. Theoretically, deriving complete necessary and sufficient conditions which can be used to construct a flat output remains an open problem. Only partial results are available and examples of recent work include [9] where the system dynamics is assumed to be a one-fold prolongation flat. Two-input systems and endogenous dynamic feedback are considered in [10]. On the applied flatness based controller side, the concept of flatness is widely used in open-loop trajectory planning [11, 12]. For classical quadrotor control system, flatness based control is developed in [13, 14]. The benefits of flatness based control include accurate aggressive trajectory-tracking with impressive performance in real world experiments and low computation cost compared to model predictive controllers [15]. For the SLS, although flatness was utilized in that study for open-loop trajectory planning [6], there is no previous work on employing flatness for *feedback* control of the SLS. Since tracking error dynamics may be linearized, adopting flatness for closed-loop control has the advantage of making stability analysis easier. This should be compared to the results in [6, 16] where stability analysis is complicated by the multi-loop controller structure.

Beyond flatness based control, there has been a fair amount of motion control research as a result of the value in UAV SLSs. A nonlinear UAV model coupled with pendulum dynamics, which describes the suspended load, is the foundation of the majority of the work. The coupled translational UAV/pendulum dynamics in this case are controlled by an outer loop. Using a reference from the outer loop,

an inner loop regulates the rotational degrees of freedom (DoFs) of the UAV. This method is illustrated by [6, 16–18]. A 3-loop control structure is proposed in [6, 18] with the innermost loop tracking UAV attitude. Load attitude and UAV yaw are controlled by the middle loop. The final loop tracks the position of the load. It has been demonstrated that the entire tracking error dynamics are almost globally exponentially attractive. The nonlinear error dynamics achieve exponential stability (ES) in a difficult-to-describe local region that is dependent on controller gain. Another nested control result is in [16]. In this instance, the outer loop only receives a partial feedback linearization, whereas the inner loop receives a full feedback linearization. The load angle is stabilized while achieving exponential convergence of UAV position error. The possible motion by this design is limited by the controller's inability to track time-varying load position trajectories. [19] presents further development on geometric control,

The contribution of this paper is to describe a novel general algorithm, called the quasi-static feedback algorithm (QSFA), for computing a quasi-static state feedback. Such an algorithm does not appear in the literature to-date. As well, our contribution is to apply the QSFA to the SLS. Similarly to the DEA (Dynamical Extension Algorithm), the QSFA provides a straightforward procedure for testing whether a system is flat relative to a given output. This should be compared to the complicated conditions such as [9, 10] which apply to a restricted system class. Compared to our conference paper [20], the paper extends the model to counter the pendulum mass. Another major extension in this paper is in the implementation side. This paper implemented the proposed quasi controller in PX4 software-in-the-loop system to validate the performance, and this is an important step to achieve real world implementation.

2 SLS Modelling

This section presents the dynamic model for the SLS. The suspended load is modelled as a spherical pendulum attached to the UAV Centre of Mass (CoM). Table 1 and Fig. 1 gives some of the notation and variables used. Two references frames are used: a navigation frame $\mathscr{N}$ and body frame $\mathscr{B}$. Frame $\mathscr{N}$ is assumed inertial and has orthonormal basis vectors n_1, n_2, n_3 oriented north, east, and down, respectively. The origin of $\mathscr{B}$ is the UAV's CoM and its basis vectors b_1, b_2, b_3 oriented (relative to the vehicle) forward, right, and down, respectively. The configuration space of the UAV SLS is $SE(3) \times \mathbb{S}^2$ which describes pendulum position p_L, UAV attitude R, and pendulum attitude q. The unit direction vector q of the load relative to $\mathscr{N}$ is parameterized by two angles α and β. The angle of the pendulum about n_1 is α, and β is the angle about n_2 (see Fig. 1). We have

$$q = R_{n_1}(\alpha) R_{n_2}(\beta) n_3 = [s_\beta, -s_\alpha c_\beta, c_\alpha c_\beta]^T$$

where R_{n_1} and R_{n_2} are elementary rotation matrices about n_1 and n_2 axis, respectively.

The relation between load position p_L and quadrotor position p_Q is

$$p_L = p_Q + Lq = p_Q + L[s_\beta, -s_\alpha c_\beta, c_\alpha c_\beta]^T \tag{1}$$

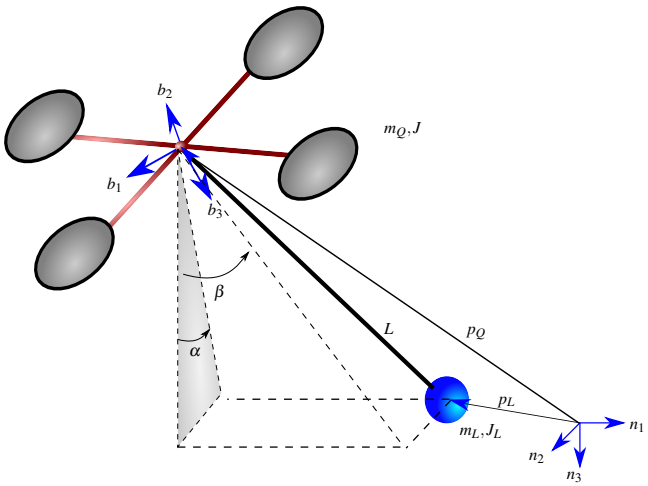

Fig. 1 The UAV with suspended load. The suspended load is modelled as a spherical pendulum attached to the UAV CoM.

Table 1 Symbol Summary

Symbol	Description
$p_Q \in \mathbb{R}^3$	position of UAV in $\mathscr{N}$
$p_L \in \mathbb{R}^3$	position of load in $\mathscr{N}$
$v_Q = \dot{p}_Q \in \mathbb{R}^3$	linear velocity of UAV in $\mathscr{N}$
$v_L = \dot{p}_L \in \mathbb{R}^3$	linear velocity of load in $\mathscr{N}$
$R \in SO(3)$	rotation matrix from $\mathscr{N}$ to $\mathscr{B}$
$\eta = [\phi, \theta, \psi]^T \in \mathbb{R}^3$	Euler angles parameterizing R
$q \in \mathbb{S}^2$	orientation of the load
$[\alpha, \beta] \in \mathbb{R}^2$	Euler angles parameterizing q
$[\gamma_\alpha = \dot{\alpha}, \gamma_\beta = \dot{\beta}] \in \mathbb{R}^2$	time derivative of $[\alpha, \beta]$
$\omega \in \mathbb{R}^3$	angular velocity of UAV in $\mathscr{B}$
$\omega_L \in \mathbb{R}^3$	angular velocity of load in $\mathscr{N}$
$\bar{u} \in \mathbb{R}_{\geq 0}$	total thrust generated by the actuators
$\tau \in \mathbb{R}^3$	propeller torque in $\mathscr{B}$
$T \in \mathbb{R}_{\geq 0}$	tension in pendulum rod
$J, J_L \in \mathbb{R}^{3\times 3}$	inertia of UAV and load
L	length of the pendulum
m_Q, m_L	mass of UAV and load
g	gravitational acceleration

where L is pendulum length. Differentiating (1), we obtain the velocity and acceleration relations

$$v_L = v_Q + L\dot{q} \tag{2}$$

$$\dot{v}_L = \dot{v}_Q + L\ddot{q} \tag{3}$$

The relation between the pendulum CoM p_p and quadrotor position p_Q is similar to load position expression p_L as

$$p_p = P_Q + \frac{1}{2}Lq \tag{4}$$

$$v_p = v_Q + \frac{1}{2}L\dot{q} \tag{5}$$

$$\dot{v}_p = \dot{v}_Q + \frac{1}{2}L\ddot{q} \tag{6}$$

The UAV dynamics is

$$\dot{p}_Q = v_Q \tag{7a}$$

$$m_Q\dot{v}_Q = m_Q g n_3 - R\bar{u}n_3 + Tq \tag{7b}$$

$$\dot{R} = RS(\omega) \tag{7c}$$

$$J\dot{\omega} = -\omega \times J\omega + \tau \tag{7d}$$

where $J = \mathrm{diag}(J_1, J_2, J_3) \in \mathbb{R}^{3\times 3}$ is the inertia matrix of the UAV, $T \in \mathbb{R}_{\geq 0}$ denotes tension in the pendulum, $\bar{u} \in \mathbb{R}_{\geq 0}$ is the total propeller thrust, $\tau \in \mathbb{R}^3$ is propeller torque expressed in $\mathscr{B}$, g is acceleration due to gravity, and m_Q is UAV mass. The skew operator $S(\cdot) : \mathbb{R}^3 \to so(3)$ in (7c) is given by

$$S(x) = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}, \quad \text{where } x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}.$$

The rotational kinematics in (7c) can be parameterized using the Euler angles $\eta = [\phi, \theta, \psi]^T \in \mathbb{R}^3$:

$$\dot{\eta} = W(\eta)\omega \tag{8}$$

with

$$W(\eta) = \begin{bmatrix} 1 & s_\phi t_\theta & c_\phi t_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi / c_\theta & c_\phi / c_\theta \end{bmatrix}$$

where $t_\theta = \tan\theta, s_\theta = \sin\theta, c_\theta = \cos\theta$.

The pendulum can be modelled as a cylinder. The cylinder has a mass m_C and an inertia matrix w.r.t. its CoM J_C. For the pendulum translational dynamics, we have

$$\dot{p}_L = v_L \tag{9}$$

$$m_L\dot{v}_L = -Tq + (m_L + m_C)gn_3 \tag{10}$$

The rotational dynamics of the pendulum is

$$\dot{q} = \omega_L \times q \tag{11}$$

$$J_L\dot{\omega}_L = -\omega_L \times J_L\omega_L + Lq \times (m_L g n_3 - m_L\dot{v}_Q) + \frac{1}{2}Lq \times (m_C g n_3 - m_C\dot{v}_Q) \tag{12}$$

where J_L is the load inertia matrix about the UAV CoM. Using Steiner's Theorem [21], we can express J_L as a function of the relative position between the load and UAV:

$$J_L = m_L L^2 (I - qq^T) + J_C + m_C \frac{L^2}{4}(I - qq^T) \tag{13}$$

Eliminating the internal force term T in (10) and (7b) we have

$$m_Q \dot{v}_Q + m_L \dot{v}_L = (m_Q + m_L + m_c) g n_3 - R\bar{u} n_3 \tag{14}$$

Substituting for $\dot{v}_Q$ in (14) using (3), we have the translational dynamics for load position

$$\begin{aligned}(m_L + m_Q)\dot{v}_L &= (m_L + m_Q + m_c) g n_3 - R\bar{u} n_3 + m_Q L \ddot{q} \\ &= (m_L + m_Q + m_c) g n_3 - R\bar{u} n_3 \\ &\quad + m_Q L(\dot{\omega}_L \times q + \omega_L \times \dot{q})\end{aligned} \tag{15}$$

where we have substituted $\ddot{q} = \dot{\omega}_L \times q + \omega_L \times \dot{q}$ from (11). Combining (15) and (3), we get the expression for $\dot{v}_Q$ as:

$$\dot{v}_Q = -\frac{m_L}{m_Q + m_L} L \ddot{q} + (1 + \frac{m_c}{m_L + m_Q}) g n_3 - \frac{R\bar{u} n_3}{m_Q + m_L} \tag{16}$$

Substituting (13), and (16) into (12), we are able to obtain the rotational dynamics of the load expressed in $\mathcal{N}$:

$$\begin{aligned}J_L \dot{\omega}_L = &-\omega_L \times J_L \omega_L + Lq \times \big((m_L + \frac{m_C}{2}) \\ &(g n_3 + \frac{m_L}{m_Q + m_L} L \ddot{q} - (1 + \frac{m_c}{m_L + m_Q}) g n_3 + \frac{R\bar{u} n_3}{m_Q + m_L})\big)\end{aligned} \tag{17}$$

In addition, we have the relation between ω_L and $\gamma_\alpha, \gamma_\beta$

$$\omega_L = \begin{bmatrix}1 \\ 0 \\ 0\end{bmatrix} \gamma_\alpha + \begin{bmatrix}0 \\ c_\alpha \\ s_\alpha\end{bmatrix} \gamma_\beta \tag{18}$$

and

$$\begin{aligned}\dot{\omega}_L &= \begin{bmatrix}1 \\ 0 \\ 0\end{bmatrix} \dot{\gamma}_\alpha + \begin{bmatrix}0 \\ c_\alpha \\ s_\alpha\end{bmatrix} \dot{\gamma}_\beta + \begin{bmatrix}0 \\ -s_\alpha \\ c_\alpha\end{bmatrix} \gamma_\alpha \gamma_\beta \\ &= R_{n_1}(\alpha) \begin{bmatrix}\dot{\gamma}_\alpha \\ \dot{\gamma}_\beta \\ \gamma_\alpha \gamma_\beta\end{bmatrix}\end{aligned} \tag{19}$$

Substituting (18) and (19) into (15) and (17), and solving for $\dot{v}_L, \dot{\gamma}_\alpha, \dot{\gamma}_\beta$, we obtain a state space form for the SLS dynamics

$$\dot{x} = \begin{bmatrix} v \\ \gamma_\alpha \\ \gamma_\beta \\ W(\eta)\omega \\ -\mathrm{s}_\beta(\gamma_\alpha^2\mathrm{c}_\beta^2 + \gamma_\beta^2)LM_0 \\ \mathrm{s}_\alpha\mathrm{c}_\beta(\gamma_\alpha^2\mathrm{c}_\beta^2 + \gamma_\beta^2)LM_0 \\ g - \mathrm{c}_\alpha\mathrm{c}_\beta(\gamma_\alpha^2\mathrm{c}_\beta^2 + \gamma_\beta^2)LM_0 \\ 2\gamma_\alpha\gamma_\beta\mathrm{t}_\beta \\ -\gamma_\alpha^2\mathrm{c}_\beta\mathrm{s}_\beta \\ -J^{-1}S(\omega)J\omega \end{bmatrix} + \begin{bmatrix} 0_{8\times1} & 0_{8\times3} \\ \bar{g}(x) & 0_{5\times3} \\ 0_{3\times1} & J^{-1} \end{bmatrix} u \tag{20}$$

where

$$\begin{aligned} x &= [p_L^T, \alpha, \beta, \eta^T, v_L^T, \gamma_\alpha, \gamma_\beta, \omega^T]^T \in \mathbb{R}^{16} \\ u &= [\bar{u}, \tau^T]^T \in \mathbb{R}^4 \\ M_0 &= \frac{2m_Q + m_L}{2(m_Q + m_L + m_C)} \end{aligned}$$

and the expression for $\bar{g}$ is given in the Appendix. We note that (20) has singularities at $\mathrm{c}_\beta = \mathrm{c}_\theta = 0$ due to parametrization of the orientation of the UAV and pendulum. As a result, the domain $\mathscr{M}$ of x is defined as a subset of $\mathbb{R}^{16}$ that contains 0 but excludes these points. Practically, these singularities are unlikely to be of concern because they involve dangerous SLS motion. Furthermore, $\mathrm{c}_\beta = 0$ would require the pendulum to collide with the UAV frame.

3 Quasi-Static Feedback Linearization

We define the general system

$$\dot{x} = f(x) + \sum_{i=1}^{m} g_i(x)u_i \tag{21a}$$

$$y_i = h_i(x), \quad 1 \le i \le m \tag{21b}$$

with vector fields $f, g_i : \mathscr{M} \to \mathbb{R}^n$ and output functions $h_i : \mathscr{M} \to \mathbb{R}$ defined on an open subset $\mathscr{M} \subset \mathbb{R}^n$. The quasi-static feedback is taken as $u = u(x, v, \dot{v}, \ddot{v}, \ldots, v^{(\rho)})$, where v is an auxiliary input which is eventually assigned to a linear function of the output tracking error and its time derivative. Compared with dynamic state feedback, quasi-static feedback is a static function of state, i.e., it requires no state

augmentation. This leads to a simpler control design which is important for onboard implementation. In Section 3.1 we present the Quasi-Static Feedback Algorithm (QSFA) which produces a quasi-static feedback linearizing controller for a flat system of the form (21). The QSFA is then applied to the SLS in Section 3.2.

3.1 Quasi-Static Feedback Algorithm (QSFA)

The Lie derivative of a function $\lambda : \mathscr{M} \to \mathbb{R}$ along the vector field f is defined by $L_f\lambda(x) = \frac{\partial \lambda}{\partial x} f(x)$. When performing QSFA, we make use of a vector of indices $r = [r_1, \dots r_m]$ such that r_i is the largest integer satisfying $L_{g_j} L_f^k h_i(x) = 0, 1 \leq j \leq m, k < r_i - 1$, about some $x_0 \in \mathscr{M}$. The existence of these indices does not imply the system has a well-defined relative degree about x_0 as this requires the decoupling matrix

$$D(x) = \begin{bmatrix} L_{g_1} L_f^{r_1-1} h_1(x) & \dots & L_{g_m} L_f^{r_1-1} h_1(x) \\ \vdots & \dots & \vdots \\ L_{g_1} L_f^{r_m-1} h_m(x) & \dots & L_{g_m} L_f^{r_m-1} h_m(x) \end{bmatrix} \tag{22}$$

is nonsingular at x_0.

Variables have superscript $^{\langle i \rangle}$ to keep track of the algorithm iteration. Superscript $^{(k)}$ denotes the kth order time derivative. We begin with Step 0 and assume the decoupling matrix $D^{\langle 0 \rangle} = D$ given by (22) has constant rank less than m about $x_0 \in \mathscr{M}$ where $r^{\langle 0 \rangle} = [r_1^{\langle 0 \rangle}, \dots, r_m^{\langle 0 \rangle}]$. Define $s^{\langle i \rangle} = \text{rank}(D^{\langle i \rangle}(x_0))$, and $y^{\langle 0 \rangle} = y^{(r^{\langle 0 \rangle})} = [y_1^{(r_1^{\langle 0 \rangle})}, \dots, y_m^{(r_m^{\langle 0 \rangle})}]^T$.

Step 0: According to the definition of $r^{\langle 0 \rangle}$ we can write $y^{(r^{\langle 0 \rangle})}$ as

$$y^{\langle 0 \rangle} = y^{(r^{\langle 0 \rangle})} = a_0(x) + D^{\langle 0 \rangle} u \tag{23}$$

We reorder and decompose $y^{\langle 0 \rangle}$ as

$$y^{\langle 0 \rangle} = \begin{bmatrix} \tilde{y}^{\langle 0 \rangle} \\ \hat{y}^{\langle 0 \rangle} \end{bmatrix} \tag{24}$$

where $\tilde{y}^{\langle 0 \rangle}$ corresponds to the first $s^{\langle 0 \rangle}$ independent rows of $D^{\langle 0 \rangle}$. Introducing auxiliary inputs $v_1 = \tilde{y}^{\langle 0 \rangle}$ and since the last rows of $D^{\langle 0 \rangle}$ are linearly dependent on the first $s^{\langle 0 \rangle}$ rows, we can write

$$\tilde{y}^{\langle 0 \rangle} = \tilde{a}_0(x) + \tilde{b}_0(x) u = v_1 \tag{25}$$

$$\hat{y}^{\langle 0 \rangle} = \hat{y}^{\langle 0 \rangle}(x, v_1) \tag{26}$$

where $\hat{y}^{\langle 0\rangle}$ is affine in v_1 and $\tilde{b}_0(x)$ are the first $s^{\langle 0\rangle}$ independent rows of $D^{\langle 0\rangle}$. Similarly, the index $r^{\langle 1\rangle}$ is calculated by taking the time derivative of $\hat{y}^{\langle 0\rangle}$. We obtain

$$\begin{aligned}\dot{\hat{y}}^{\langle 0\rangle} = \hat{y}^{(r^{\langle 0\rangle}+1)} &= \frac{\partial \hat{y}^{\langle 0\rangle}}{\partial x}[f(x)+\sum_{i=1}^{m} g_i(x)u_i] + \frac{\partial \hat{y}^{\langle 0\rangle}}{\partial v_1}\dot{v}_1 \qquad (27)\\ &= L_f\hat{y}^{\langle 0\rangle} + \frac{\partial \hat{y}^{\langle 0\rangle}}{\partial v_1}\dot{v}_1 + \sum_{i=1}^{m} L_{g_i}\hat{y}^{\langle 0\rangle}u_i\\ &= a_1(x,\dot{v}_1) + b_1(x)u \qquad (28)\end{aligned}$$

where $b_1(x) \in \mathbb{R}^{(m-s^{\langle 0\rangle})\times m}$ is given by

$$b_1(x) = \begin{bmatrix} L_{g_1}\hat{y}_1^{\langle 0\rangle}(x) & \dots & L_{g_m}\hat{y}_1^{\langle 0\rangle}(x) \\ \vdots & \dots & \vdots \\ L_{g_1}\hat{y}_{m-s^{\langle 0\rangle}}^{\langle 0\rangle}(x) & \dots & L_{g_m}\hat{y}_{m-s^{\langle 0\rangle}}^{\langle 0\rangle}(x) \end{bmatrix} \qquad (29)$$

Three different cases may appear. If $b_1(x)$ is identically zero, we should continue to take the time derivative of $\hat{y}^{\langle 0\rangle}$. If any row of $b_1(x)$ is linearly dependent on $\tilde{b}_0(x)$ in (25), it should be expressed by (x, v_1) similar to (26), and then time derivatives of these rows should be taken. If all rows of $b_1(x)$ are linearly independent of $\tilde{b}_0(x)$ in (25). Define $r^{\langle 1\rangle} = [r_1^{\langle 1\rangle}, \dots, r_{m-s^{\langle 0\rangle}}^{\langle 1\rangle}]$ so that we have $y^{\langle 1\rangle} = (\hat{y}^{\langle 0\rangle})^{(r^{\langle 1\rangle})}$. Every row of the corresponding decoupling matrix $D^{\langle 1\rangle}$ is linearly independent of every row of $\tilde{b}_0(x)$. Decompose $y^{\langle 1\rangle}$ as

$$y^{\langle 1\rangle} = \begin{bmatrix} \tilde{y}^{\langle 1\rangle} \\ \hat{y}^{\langle 1\rangle} \end{bmatrix} \qquad (30)$$

where $\tilde{y}^{\langle 1\rangle}$ corresponds to the first $s^{\langle 1\rangle}$ independent rows of $D^{\langle 1\rangle}$. Introducing auxiliary inputs $v_2 = \tilde{y}^{\langle 1\rangle}$. Similar to (25) and (26), $y^{\langle 1\rangle}$ can be written as

$$\tilde{y}^{\langle 1\rangle} = \tilde{a}_1(x, v_1, \dots, v_1^{(r^{\langle 1\rangle})}) + \tilde{b}_1(x, v_1, \dots, v_1^{(r^{\langle 1\rangle}-1)})u = v_2 \qquad (31)$$

$$\hat{y}^{\langle 1\rangle} = \hat{y}^{\langle 1\rangle}(x, v_1, \dots, v_1^{(r^{\langle 1\rangle})}, v_2) \qquad (32)$$

Step $k+1$**.** Suppose that in Steps 0 through k, $y^{\langle 0\rangle}, \dots, y^{\langle k\rangle}$ have been defined so that

$$\begin{aligned}\tilde{y}^{\langle 0\rangle} &= \tilde{a}_0(x) + \tilde{b}_0(x)u\\ &\vdots\\ \tilde{y}^{\langle k\rangle} &= \tilde{a}_k(x, \{v_i^{(j)} : 1 \le i \le k, 1 \le j \le \sum_{l=i}^{k} r^{\langle l\rangle}\})\\ &\quad + \tilde{b}_k(x, \{v_i^{(j)} : 1 \le i \le k, 1 \le j \le \sum_{l=i}^{k} r^{\langle l\rangle} - 1\})u\end{aligned}$$

$$\hat{y}^{\langle k\rangle} = \hat{y}_k^{\langle k\rangle}(x, \{v_i^{(j)} : 1 \le i \le k, 1 \le j \le \sum_{l=i}^{k} r^{\langle l\rangle}\}, v_{k+1})$$

and so that they are rational functions of $v_i^{(j)}$. Define $y^{\langle k+1\rangle} = (\hat{y}^{(r^{\langle k\rangle})})^{(r^{\langle k+1\rangle})}$ which can be decomposed as before:

$$y^{\langle k+1\rangle} = \begin{bmatrix} \tilde{y}^{\langle k+1\rangle} \\ \hat{y}^{\langle k+1\rangle} \end{bmatrix} \tag{33}$$

where $\tilde{y}^{\langle k+1\rangle}$ consist of the first $s^{\langle k+1\rangle}$ independent rows of $D^{\langle k+1\rangle}$. Introducing auxiliary inputs $v_{k+2} = \tilde{y}^{\langle k+1\rangle}$, (33) can be written as

$$\begin{aligned} \tilde{y}^{\langle k+1\rangle} &= \tilde{a}_{k+1}(x, \{v_i^{(j)} : 1 \le i \le k+1, 1 \le j \le \sum_{l=i}^{k+1} r^{\langle l\rangle}\}) \\ &\quad + \tilde{b}_k(x, \{v_i^{(j)} : 1 \le i \le k+1, 1 \le j \le \sum_{l=i}^{k+1} r^{\langle l\rangle} - 1\})u \\ \hat{y}^{\langle k+1\rangle} &= \hat{y}_{k+1}^{\langle k\rangle}(x, \{v_i^{(j)} : 1 \le i \le k+1, 1 \le j \le \sum_{l=i}^{k+1} r^{\langle l\rangle}\}, v_{k+2}) \end{aligned}$$

If $s^{\langle 0\rangle} + s^{\langle 1\rangle} + \cdots + s^{\langle k+1\rangle} = m$, the algorithm terminates. Otherwise, we take the time derivative of $\hat{y}^{\langle k+1\rangle}$ and perform the next iteration.

Defining $[\tilde{y}^{\langle 0\rangle}, \ldots, \tilde{y}^{\langle k+1\rangle}]^T = [v_1, \ldots, v_{k+2}]^T = v \in \mathbb{R}^m$, and when the algorithm terminates, we have an invertible relation between the original input u and auxiliary input v:

$$v = \begin{bmatrix} \tilde{a}_0(x) \\ \vdots \\ \tilde{a}_{k+1}(x, \{v_i^{(j)} : 1 \le i \le k+1, 1 \le j \le \sum_{l=i}^{k+1} r^{\langle l\rangle}\}) \end{bmatrix} + D^o u$$

where D° is the final decoupling matrix with $\operatorname{rank}(D^\circ) = m$.

3.2 SLS Controller Design Using QSFA

In this section, the QSFA is applied to the 16-dimensional SLS model (20). It is shown that the SLS can be quasi-static state feedback linearized using QSFA and the pendulum position and yaw as outputs y,

$$y = [p_L^T, \psi]^T \tag{34}$$

Step 0**.** According to (22), we have

$$D^{\langle 0\rangle} = \begin{bmatrix} d_{11}^{\langle 0\rangle} & 0 & 0 & 0 \\ d_{21}^{\langle 0\rangle} & 0 & 0 & 0 \\ d_{31}^{\langle 0\rangle} & 0 & 0 & 0 \\ 0 & 0 & \frac{\mathrm{s}_\phi}{J_2 \mathrm{c}_\theta} & \frac{\mathrm{c}_\phi}{J_3 \mathrm{c}_\theta} \end{bmatrix} \tag{35}$$

where d_{11}, d_{21}, d_{31} are functions of state and the index $r^{\langle 0\rangle} = [2,2,2,2]^T$ and $\mathrm{rank}(D^{\langle 0\rangle}) = 2$ on a subset of $\mathscr{M}$ where

$$d_{31}^{\langle 0\rangle}(x) = -\frac{[\mathrm{s}_\beta, -\mathrm{s}_\alpha \mathrm{c}_\beta, \mathrm{c}_\alpha \mathrm{c}_\beta] \cdot Rn_3}{m_Q + m_L + m_C} \neq 0 \tag{36}$$

Referring to (23), we have

$$y^{\langle 0\rangle} = a_0(x) + D^{\langle 0\rangle} u \tag{37}$$

Thus $y^{\langle 0\rangle}$ is decoupled as $\tilde{y}^{\langle 0\rangle} = [\ddot{y}_3, \ddot{y}_4]^T$, and $\hat{y}^{\langle 0\rangle} = [\ddot{y}_1, \ddot{y}_2]^T$. Here, we introduce auxiliary input $v_1 = [\ddot{y}_3, \ddot{y}_4]^T$, we have

$$v_1 = \tilde{y}^{\langle 0\rangle} = \tilde{a}_0(x) + \tilde{b}_0(x) u \tag{38}$$

Then, $\hat{y}^{\langle 0\rangle}$ in (26) can be written as

$$\hat{y}^{\langle 0\rangle} = \begin{bmatrix} -\left(g - [1,0]^T v_1\right) \mathrm{t}_\beta / \mathrm{c}_\alpha \\ \left(g - [1,0]^T v_1\right) \mathrm{t}_\alpha \end{bmatrix} \tag{39}$$

Step 1**.** Taking a time derivative of $\hat{y}^{\langle 0\rangle}$ we obtain

$$\dot{\hat{y}}^{\langle 0\rangle} = \begin{bmatrix} \frac{\dot{v}_1 \mathrm{s}_\beta}{\mathrm{c}_\beta \mathrm{c}_\alpha^2} - \frac{\gamma_\beta (g - v_1)}{\mathrm{c}_\beta^2 \mathrm{c}_\alpha} - \frac{\gamma_\alpha \mathrm{s}_\alpha \mathrm{s}_\beta (g - v_1)}{\mathrm{c}_\beta \mathrm{c}_\alpha} \\ -\dot{v}_1 \mathrm{t}_\alpha + \frac{\gamma_\alpha (g - v_1)}{\mathrm{c}_\alpha^2} \end{bmatrix} \tag{40}$$

Taking the second time derivative of $\hat{y}^{\langle 0\rangle}$ gives

$$\ddot{\hat{y}}^{\langle 0\rangle} = a_1(x, v_1, \dot{v}_1, \ddot{v}_1) + b_1(x, v_1, \dot{v}_1) u \tag{41}$$

where $a_1(x, v_1, \dot{v}_1, \ddot{v}_1) \in \mathbb{R}^{2\times 1}$, $b_1(x, v_1, \dot{v}_1) \in \mathbb{R}^{2\times 4}$ are functions of x, auxiliary input v, and its time derivative. The matrix b_1 has the structure

$$b_1(x, v_1, \dot{v}_1) = \begin{bmatrix} b_{11} & 0 & 0 & 0 \\ b_{21} & 0 & 0 & 0 \end{bmatrix} \tag{42}$$

We observe that both rows of (42) are linearly dependent on the third row of $D^{\langle 0\rangle}$, thus $\ddot{\hat{y}}^{\langle 0\rangle}$ can be expressed using v_1. As a result, (41) can be written as $\ddot{\hat{y}}^{\langle 0\rangle} =$

$\ddot{\hat{y}}^{\langle 0\rangle}(x, v_1, \dot{v}_1, \ddot{v}_1)$ with input u eliminated. The same procedure is applied for $\left(\hat{y}^{\langle 0\rangle}\right)^{(3)}$. When we calculate $\left(\hat{y}^{\langle 0\rangle}\right)^{(4)}$, the corresponding decoupling matrix $D^{\langle 1\rangle}$ is

$$D^{\langle 1\rangle} = \begin{bmatrix} d_{11}^{\langle 1\rangle} & d_{12}^{\langle 1\rangle} & d_{13}^{\langle 1\rangle} & 0 \\ d_{21}^{\langle 1\rangle} & d_{22}^{\langle 1\rangle} & d_{23}^{\langle 1\rangle} & 0 \end{bmatrix} \tag{43}$$

where $d_{ij}^{\langle 1\rangle}$ are functions of $(x, \dot{v}_1, \ldots, v_1^{(3)})$ with $\text{rank}(D^{\langle 1\rangle}) = 2$ and its rows are linear independent of any row in $D^{\langle 0\rangle}$. Thus, we get $r^{\langle 1\rangle} = [4, 4]$. Introducing the auxiliary input $v_2 = \left(\hat{y}^{\langle 0\rangle}\right)^{(r^{\langle 1\rangle})} = y^{\langle 1\rangle}$, we have

$$v_2 = y^{\langle 1\rangle} = \tilde{a}_1(x, \dot{v}_1, \ldots, v^{(4)}) + D^{\langle 1\rangle} u \tag{44}$$

Combining v_1 and v_2, we have a invertible relation between the original input u and auxiliary inputs v_1 and v_2.

$$\begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \begin{bmatrix} \tilde{a}_0(x) \\ \tilde{a}_1(x, \dot{v}_1, \ldots, v^{(4)}) \end{bmatrix} + D^o u \tag{45}$$

where D° is the final decoupling matrix with $\text{rank}\,(D^\circ) = 4$ given below

$$D^\circ = \begin{bmatrix} d_{31}^{\langle 0\rangle} & 0 & 0 & 0 \\ 0 & 0 & \frac{\mathrm{s}_\phi}{J_2 \mathrm{c}_\theta} & \frac{\mathrm{c}_\phi}{J_3 \mathrm{c}_\theta} \\ d_{11}^{\langle 1\rangle} & d_{12}^{\langle 1\rangle} & d_{13}^{\langle 1\rangle} & 0 \\ d_{21}^{\langle 1\rangle} & d_{22}^{\langle 1\rangle} & d_{23}^{\langle 1\rangle} & 0 \end{bmatrix} \tag{46}$$

3.3 Output Tracking

By setting $[v_1, v_2]^T = [y_3^{(2)}, y_4^{(2)}, y_1^{(6)}, y_2^{(6)}]^T$ as in (45), a quasi-static feedback linearizing controller law is obtained. The tracking error is defined as

$$\begin{aligned} \tilde{z} = [&y_1 - y_{d1}, \dot{y}_1 - \dot{y}_{d1}, \ldots, y_1^{(5)} - y_{d1}^{(5)}, \\ &y_2 - y_{d2}, \dot{y}_2 - \dot{y}_{d2}, \ldots, y_2^{(5)} - y_{d2}^{(5)}, \\ &y_3 - y_{d3}, \dot{y}_3 - \dot{y}_{d3}, y_4 - y_{d4}, \dot{y}_4 - \dot{y}_{d4}]^T \end{aligned} \tag{47}$$

where $y_{di}, 1 \le i \le 4$ are desired outputs. The variables $y_1^{(3)}, \ldots, y_1^{(5)}$, and $y_2^{(3)}, \ldots, y_2^{(5)}$ are functions of $x, v_1, \dot{v}_1, \ldots, v_1^{(4)}$, while the remaining outputs and their derivatives in (47) can be expressed as a function of x.

We can express the dynamics in the $\tilde{z}$-coordinates as

$$\dot{\tilde{z}} = A_c \tilde{z} + B_c(\tilde{a} + D^\circ u) \tag{48}$$

where

$$\begin{aligned} A_c &= \text{diag}(A_1, A_2, A_3, A_4) \\ B_c &= \begin{bmatrix} e_6 & e_{12} & e_{14} & e_{16} \end{bmatrix} \\ \tilde{a} &= [\tilde{a}_0(x), \tilde{a}_1(x, \dot{v}_1, \ldots, \dot{v}^{(4)})]^T \end{aligned}$$

$e_i \in \mathbb{R}^{16}$ denotes the unit vector in the ith direction, and

$$A_1 = A_2 = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \text{ and } A_j = \begin{bmatrix} 0_{5\times 1} & I_5 \\ 0 & 0_{1\times 5} \end{bmatrix} \in \mathbb{R}^{6\times 6}$$

with $j = 3, 4$. Applying the linearizing control $u = D^{\circ -1}(K\tilde{z} - \tilde{a} + y_d^{(\bar{r})})$ with $y_d^{(\bar{r})} = [y_{d1}^{(6)}, y_{d2}^{(6)}, y_{d3}^{(2)}, y_{d4}^{(2)}]^T$ to (48) gives

$$\dot{\tilde{z}} = (A_c + B_c K)\tilde{z} \tag{49}$$

where $K \in \mathbb{R}^{4\times 16}$ is the control gain chosen so that $A_c + B_c K$ is Hurwitz and the tracking error transient performance is satisfactory. Because $\dot{v}_1, \ldots, v_1^{(4)}$ are calculated using $y_3 - y_{d3}, \dot{y}_3 - \dot{y}_{d3}, y_4 - y_{d4}, \dot{y}_4 - \dot{y}_{d4}$ and their time derivatives, the controller depends only on x and the reference trajectory. Hence, it is a static state feedback.

3.4 Domain of the Dynamic State Feedback Linearization

In this section, we determine the domain on which the quasi-static control law is well-defined. The control is singular at $\theta = \pm 90^\circ$ and $\beta = \pm 90^\circ$ as these are singularities occur in the SLS model due to Euler angles. At points where the Jacobian matrix of the $\tilde{z}$-coordinates is singular, the control is also singular. These are the same points as those where the distribution rank condition [22] does not hold. Alternatively, we can find these points from singularities in the decoupling matrix D° in (46). We have,

$$\phi = \pm 90^\circ \tag{50a}$$

$$[\mathrm{s}_\beta, -\mathrm{s}_\alpha \mathrm{c}_\beta, \mathrm{c}_\alpha \mathrm{c}_\beta] \cdot R n_3 = 0 \tag{50b}$$

$$\ddot{p}_3 = g - \frac{\mathrm{c}_\alpha \mathrm{c}_\beta (\gamma_\alpha^2 \mathrm{c}_\beta^2 + \gamma_\beta^2) L m}{m_Q + m_L} \tag{50c}$$

$$\ddot{p}_3 = g \tag{50d}$$

We note that the geometric interpretation of the left-hand-side (LHS) of (50b) (which is a scaling of a_{31} from (36)) is the inner product of $q = [\mathrm{s}_\beta, -\mathrm{s}_\alpha \mathrm{c}_\beta, \mathrm{c}_\alpha \mathrm{c}_\beta]$ and the direction of the thrust vector Rn_3. Thus, when the direction of the pendulum is perpendicular to the thrust vector, a physical singularity occurs. The condition (50c) corresponds to $\bar{u} = 0$. When the pendulum's downward linear acceleration $\ddot{p}_3 = g$, we obtain (50d). This is another physical singularity that often appears in a UAV motion control, e.g., [23]. Hence, we conclude that the controller's domain is a reasonable

subset of $\mathscr{M}$ that excludes the aforementioned points which are not typical of normal operation.

4 Matlab Simulation

In this section, the quasi-static control law is validated using simulation. We consider simulations for output stabilization and tracking. Table 2 contains the system parameters used in the model and controller.

The quasi-static controller from the QSFA is compared with the dynamic state feedback linearizing controller obtained from the DEA [24, 25]. As well, we compare the design with a traditional linear LQR controller based on an approximate linearization at the origin. Both QSFA and DEA controllers have linear error dynamics. The advantage of quasi-static feedback is that it avoids dynamics in the controller. This makes quasi-static feedback easier to implement as no discretization of dynamics is required.

Table 2 System parameters.

m	1.6 kg
m_p	0.16 kg
L	1 m
J_1	$0.03\,\mathrm{kg}\cdot\mathrm{m}^2$
J_2	$0.03\,\mathrm{kg}\cdot\mathrm{m}^2$
J_3	$0.03\,\mathrm{kg}\cdot\mathrm{m}^2$

Position Stabilization This section considers stabilization of the SLS at $x = 0$. The system is initialised with $p_L(0) = [2,2,2]^T$ m and the remaining states are set to zero (i.e., the SLS is at rest). The control gain for the LQR control is based on $Q = 100 \cdot I_{16}, R = 0.1 \cdot I_4$, where I_n is the $n \times n$ identity matrix. Given that the error dynamics (49) is linear time invariant (LTI), designing the transient performance for y is straightforward. Figs. 3, and 2 show the control input and configuration variables. The LQR controller exhibits similar transient error performance to the quasi-static control. However, LQR control results in large control input transients near $t = 0$. The dynamic state feedback linearization results in similar performance to the quasi-static control.

Trajectory Tracking: The proposed QSFA controller is capable of tracking complex reference trajectories. We consider the "figure-8" reference

$$y_d(t) = \begin{bmatrix} 3\sin(\pi t/4) + 1 \\ 1.5\sin(\pi t/2) \\ 4\sin(\pi t/4) - 4 \\ 0.02t \end{bmatrix} \tag{51}$$

The tracking error is shown in Fig. 4 and the configuration variables are shown in Fig. 5. The input trajectories are given in Fig. 6. We observe that the tracking error converges to 0 with a fast transient and small overshoot. Input trajectories remain within a practical range. The gain for the QSFA and DEA controllers were obtained

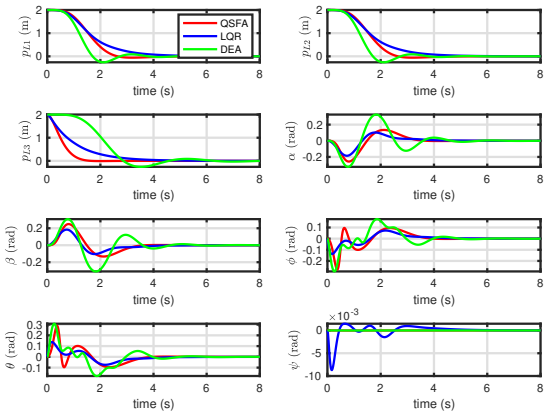

Fig. 2 System states p_L, α, β, η.

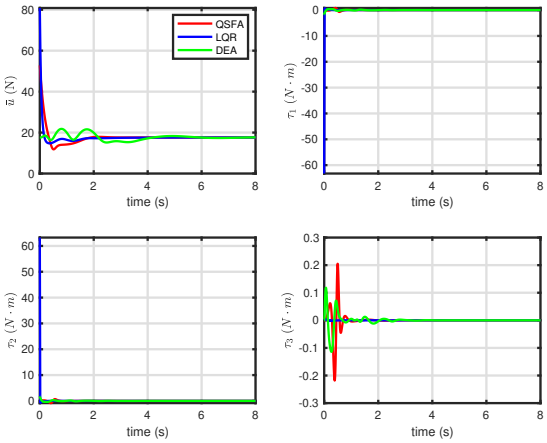

Fig. 3 Inputs $\bar{u}, \tau$.

using LQR. The QSFA and DEA controls have similar performance. However, since the DEA controller is dynamic, it requires more computation.

5 PX4:SITL Simulation

This section presents the Software-In-The-Loop (SITL) simulation of the proposed controller. SITL simulates most of what is running during a real-world experiment. For instance, it guarantees that the suggested architecture can be used in a typical autopilot framework where unmodeled phenomena (such as controller saturation, various sampling rates, computational delays, etc.) affect performance or even render the controller inoperable (e.g., due to limited on-board processing power or memory). We choose PX4 autopilot as the SITL framework [26]. Gazebo is chosen as the simulator with ODE as the multibody dynamics engine [27].

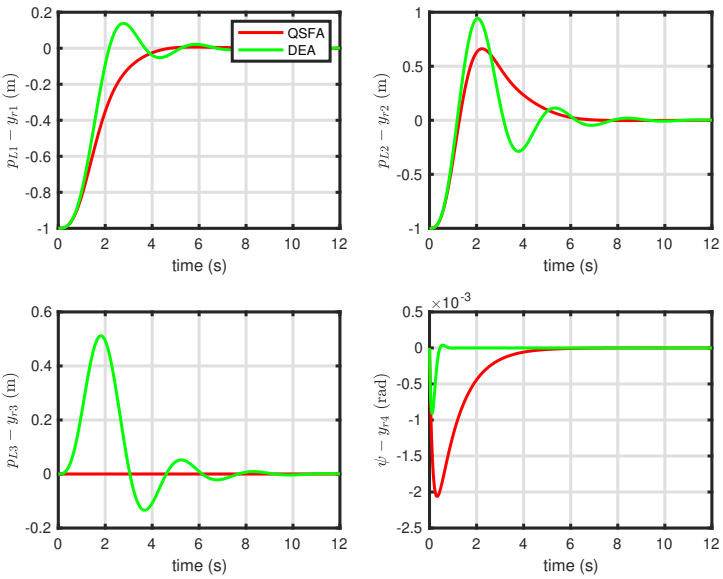

Fig. 4 Output tracking error $y - y_d$.

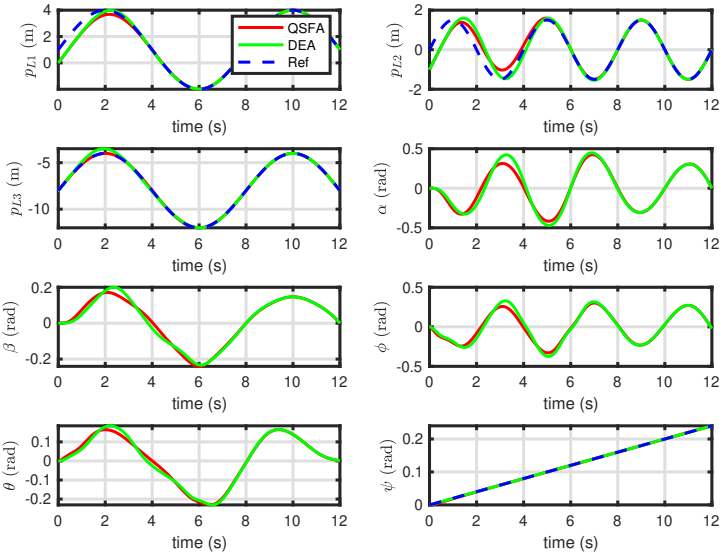

Fig. 5 System states p, α, β, η.

PX4 is an open-source autopilot system that combines Pixhawk flight controller hardware and PX4 autopilot software [26]. Compared to Matlab simulation software-in-the-loop simulation includes the PX4-Autopilot in the process. This gives us a very realistic simulation of the controller's performance in a real-world experiment and the controller implemented in SITL can be moved to a real-world experiment

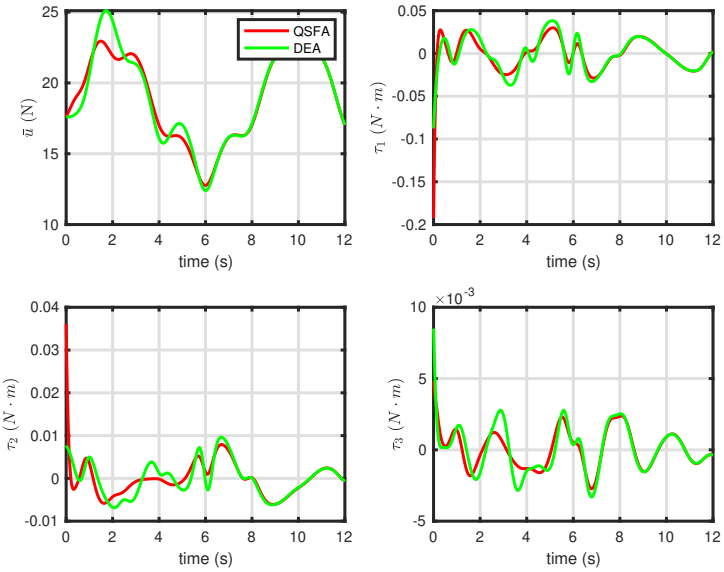

Fig. 6 Inputs $\bar{u}, \tau$.

without any changes. Our lab quadrotor hardware is always based on the open-source PX4 project. Thus, getting the PX4-SITL simulation working is a very useful and necessary step in implementing a controller from theory formulas to real-world experiments. We chose Gazebo as the simulator because of its flexibility, open source project, and because we can use the same code for real flight testing. The PX4-SILT-Gazebo simulation is based on RotorS simulation [28]. RotorS is built on ROS and a Gazebo framework[28]. To explain how each component works in the PX4:SITL simulation and how the info flows, a flow chart is given in Fig 1. A contribution of this paper is to provide simulation source code which can be validated or extended by the research community. As shown in Fig 7, there are mainly three parts in the pipeline, Gazebo simulator, PX4-Autopilot and ROS offboard controller. The Gazebo simulator contains multibody dynamics engine, a 3D graphics interface, Iris quadrotor SLS model description file, external disturbance, and a batch of plugins adopted from RotorS that simulates onboard sensors e.g. Global Positioning System, Inertia Measurement Unit and Magnetometer.

5.1 Theoretical Controller to PX4-SITL Pipeline

Model-based controller design benefits from theoretical stability, high performance, and ease of analysis and improvement. However, it suffers from complex symbolic calculations. With help of symbolic calculation software e.g. MapleSoft, Mathematic and Matlab, we are able to perform lie derivative in high dimensional space. Transferring the resulting controller from Maplesoft to C++ in SITL requires an automation process. We adopted a Maple-Matlab-SITL pipeline for developing controllers. First, we do all symbolic calculations including dynamical system modelling and quasi-static feedback algorithm in MapleSoft. Secondly, the resulting symbolic expression

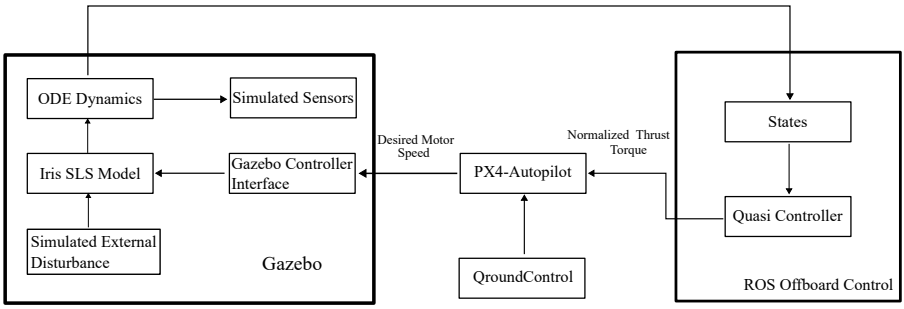

Fig. 7 PX4-SITL System.

for the dynamical system and controller are exported to Matlab to do numerical simulations to verify the correctness of the controller in theory. Lastly, the controller in Matlab will be exported as C++ by Matlab code generation toolbox to SITL system.

5.2 Gazebo Simulator for Drones

The output of the quasi-controller is the total thrust and torque, which are physically meaningful. However, PX4 autopilot expected a normalized version of thrust and torque in the range of $[-1,1]$. Then, these normalized thrusts and torques are sent to the mixer of PX4. The mixer will receive the normalized commands and convert them to rotor speed commands based on the quadrotor frame parameters. In this paper, we choose 3DR Iris as the quadrotor frame. The mixer has the following form:

$$\begin{bmatrix} \tilde{T} \\ \tilde{\tau} \end{bmatrix} = \begin{bmatrix} k_u & k_u & k_u & k_u \\ -k_u\ell_1 & k_u\ell_3 & k_u\ell_1 & -k_u\ell_3 \\ k_u\ell_2 & -k_u\ell_4 & k_u\ell_2 & -k_u\ell_4 \\ k_\tau & k_\tau & -k_\tau & -k_\tau \end{bmatrix} \begin{bmatrix} \tilde{\Omega}_1^2 \\ \tilde{\Omega}_2^2 \\ \tilde{\Omega}_3^2 \\ \tilde{\Omega}_4^2 \end{bmatrix} \tag{52}$$

where $\tilde{T}, \tilde{\tau}$ are normalized thrust and torques and $[\tilde{\Omega}_1, \tilde{\Omega}_2, \tilde{\Omega}_3, \tilde{\Omega}_4]$ are normalized rotor speed ranging from $[0,1]$, k_u is the thrust coefficient and k_τ is the torque coefficient. In the PX4, $k_u = 1$ and $k_\tau = 0.05$ is adopted. We exact the normalized mixer from the px4 which is given by matrix (53).

$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ -0.495383 & 0.495383 & 0.495383 & -0.495383 \\ 0.707107 & -0.707107 & 0.707107 & -0.707107 \\ 0.765306 & 1.000000 & -0.765306 & -1.000000 \end{bmatrix} \tag{53}$$

The gazebo simulator will receive this normalized rotor speed command and do a scaling to the normalized rotor speed command as

$$[\Omega_1, \Omega_2, \Omega_3, \Omega_4] = C_\Omega [\tilde{\Omega}_1, \tilde{\Omega}_2, \tilde{\Omega}_3, \tilde{\Omega}_4] \tag{54}$$

where C_Ω is as 1000 to be the scalling constant. The last step is for Gazebo rotor model to generate thrust from the rotor speed input. The rotor model inside Gazebo

is described by the following equations:

$$F_T = \Omega^2 C_T \tag{55}$$

$$M_D = -C_M \cdot F_T \tag{56}$$

where C_T is the rotor thrust constant, C_D is the rotor drag constant. In the case here, we choose $C_T = 5.84 \times 10^{-6}$ and $C_M = 0.06$.

5.3 Stabilization

For quasi-static controller stabilization, we test the controller in no disturbance condition, constant disturbance and variable disturbance.

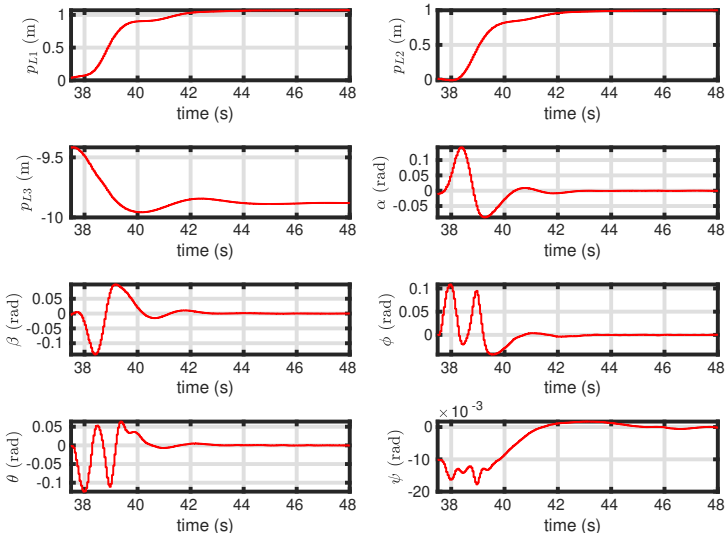

Fig. 8 System states p, α, β, η.

5.4 Trajectory Tracking

6 Conclusions

This paper presents a novel algorithm for deriving a quasi-static feedback controller for a general flat system. The algorithm is applied to an SLS. The design leads to LTI exponentially stable pendulum error dynamics on a well-defined and practical region of state space. Although the QSFA is investigated in [29–31], the design procedure is only demonstrated by example. This paper is the first to provide an algorithm for quasi-static control. The procedure for applying the QSFA is described in detail for the SLS. Simulations illustrate improved performance relative to linear state feedback. Similar performance is obtained relative to a dynamic feedback linearization.

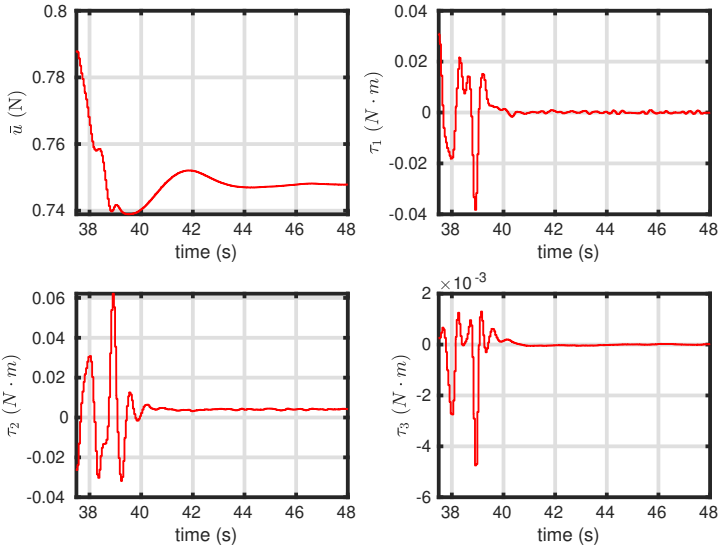

Fig. 9 Inputs $\bar{u}, \tau$.

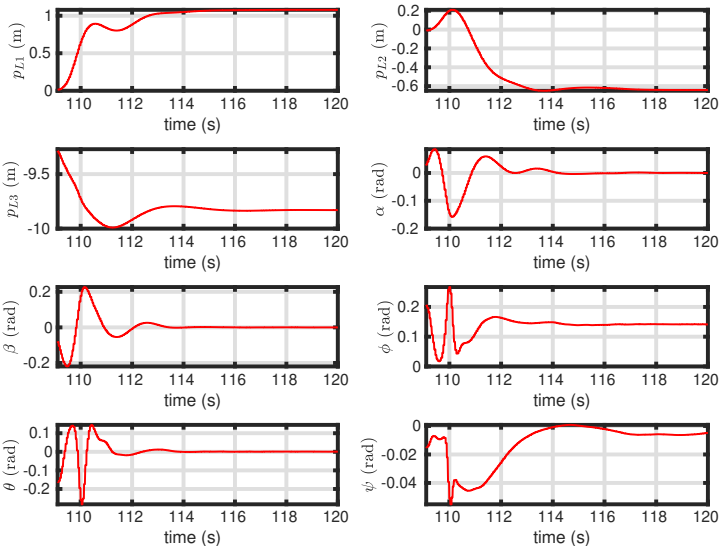

Fig. 10 System states p, α, β, η.

However, the QSFA control requires less computation and no time-discretization, making it more suitable for onboard implementation.

Appendix A

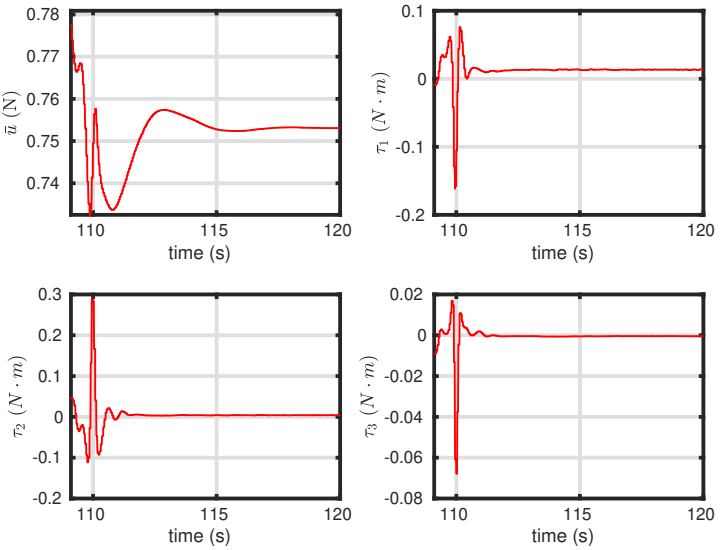

Fig. 11 Inputs $\bar{u}, \tau$.

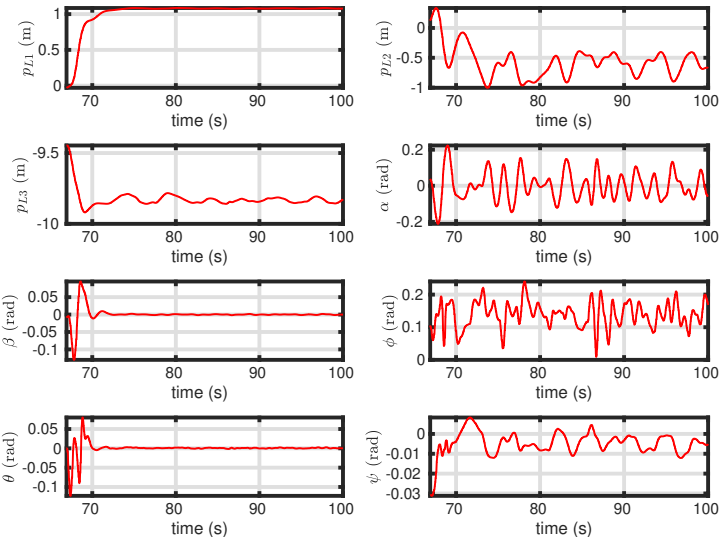

Fig. 12 System states p, α, β, η.

The expression for $\bar{g}(x)$ is

$$\bar{g}(x) = M^{-1}B = \begin{bmatrix} (\xi_1\xi_{11}\xi_{12} + \xi_2)\xi_{10} \\ ((\xi_3 + \xi_4)c_\beta\xi_{12}\xi_{10} + 2m_C\xi_5)\xi_{11} \\ ((\xi_6 + \xi_7)c_\alpha c_\beta\xi_{12}\xi_{10} - 2m_C c_\theta c_\phi)\xi_{11} \\ \frac{(12m_L+6m_C)\xi_8\xi_{11}}{c_\beta L} \\ \frac{(12m_L+6m_C)\xi_9\xi_{11}}{L} \end{bmatrix}$$

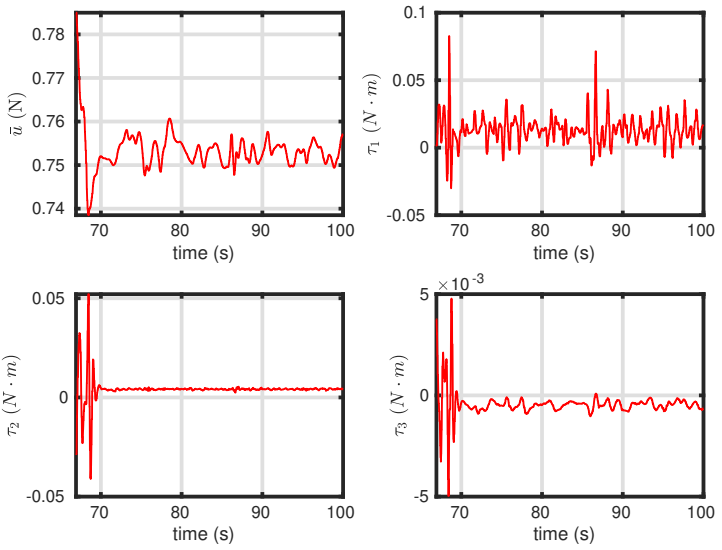

Fig. 13 Inputs $\bar{u}, \tau$.

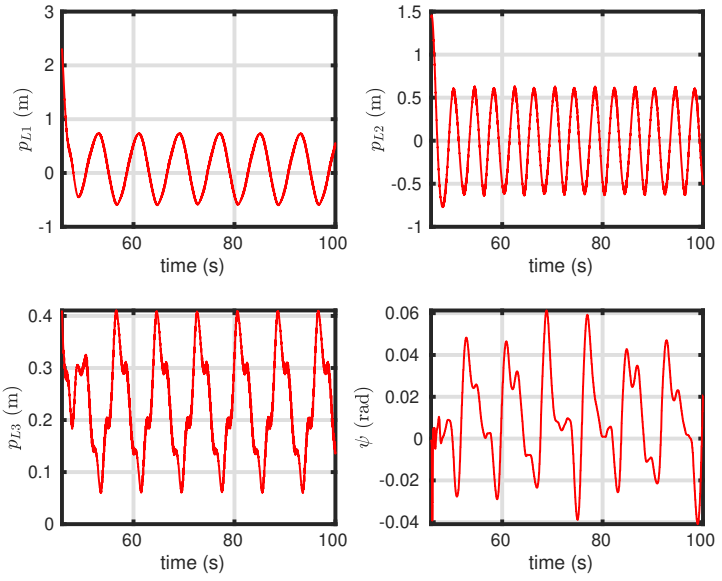

Fig. 14 Output tracking error $y - y_d$.

where

$$
\begin{aligned}
\xi_1 &= c_\beta \left(s_\beta \left(\left(c_\alpha c_\theta - s_\alpha s_\theta s_\psi \right) c_\phi + c_\psi s_\alpha s_\phi \right) - c_\beta \xi_2 \right) \\
\xi_2 &= c_\phi s_\theta c_\psi + s_\phi s_\psi \\
\xi_3 &= c_\beta \left(\left(c_\alpha^2 s_\theta s_\psi + c_\alpha c_\theta s_\alpha - s_\psi s_\theta \right) c_\phi + c_\psi s_\phi s_\alpha^2 \right) \\
\xi_4 &= s_\beta s_\alpha \left(c_\phi s_\theta c_\psi + s_\phi s_\psi \right) \\
\xi_5 &= -s_\psi s_\theta c_\phi + c_\psi s_\phi \\
\xi_6 &= c_\beta \left(\left(-s_\alpha s_\theta s_\psi + c_\alpha c_\theta \right) c_\phi + c_\psi s_\alpha s_\phi \right) \\
\xi_7 &= s_\beta \left(c_\phi s_\theta c_\psi + s_\phi s_\psi \right)
\end{aligned}
$$

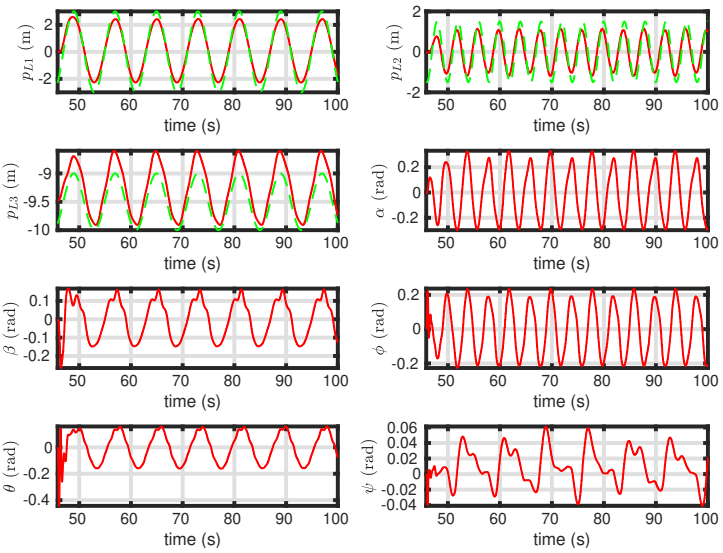

Fig. 15 System states p, α, β, η.

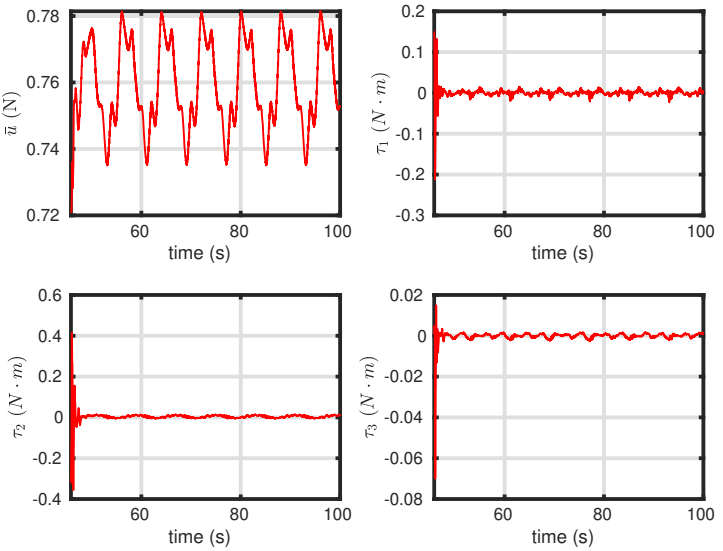

Fig. 16 Inputs $\bar{u}, \tau$.

References

[1] Villa, D.K.D., Brandão, A.S., Sarcinelli-Filho, M.: A survey on load transportation using multirotor UAVs. Journal of Intelligent & Robotic Systems **98**(2), 267–296 (2020). https://doi.org/10.1007/s10846-019-01088-w

[2] Khamseh, H.B., Janabi-Sharifi, F., Abdessameud, A.: Aerial manipulation—a literature survey. Robotics and Autonomous Systems **107**, 221–235 (2018). https://doi.org/10.1016/j.robot.2018.06.012

[3] Ruggiero, F., Lippiello, V., Ollero, A.: Aerial manipulation: A literature review. IEEE Robotics and Automation Letters **3**(3), 1957–1964 (2018). https://doi.org/10.1109/LRA.2018.2808541

[4] Fliess, M., Lévine, J., Martin, P., Rouchon, P.: Flatness and defect of non-linear systems: introductory theory and examples. International Journal of Control **61**(6), 1327–1361 (1995). https://doi.org/10.1080/00207179508921959

[5] Martin, P., Murray, R., Rouchon, P.: Flat systems. Plenary Lectures and Mini-Courses 4th European Control Conference, 1–55 (1997)

[6] Sreenath, K., Lee, T., Kumar, V.: Geometric control and differential flatness of a quadrotor UAV with a cable-suspended load. In: Proceedings of the 52nd IEEE Conference on Decision and Control, Firenze, Italy (2013). https://doi.org/10.1109/cdc.2013.6760219

[7] Rudolph, J.: Flatness-based Control: an Introduction. Shaker Verlag, ??? (2021)

[8] Delaleau, E., Rudolph, J.: Control of flat systems by quasi-static feedback of generalized states. International Journal of Control **71**(5), 745–765 (1998). https://doi.org/10.1080/002071798221551

[9] Nicolau, F., Respondek, W.: Flatness of multi-input control-affine systems linearizable via one-fold prolongation. SIAM Journal on Control and Optimization **55**(5), 3171–3203 (2017). https://doi.org/10.1137/140999463

[10] Gstöttner, C., Kolar, B., Schöberl, M.: Necessary and sufficient conditions for the linearisability of two-input systems by a two-dimensional endogenous dynamic feedback. International Journal of Control, 1–22 (2021) https://arxiv.org/abs/2106.14722 [math.DS]. https://doi.org/10.1080/00207179.2021.2015542. online access

[11] Zhou, X., Wen, X., Wang, Z., Gao, Y., Li, H., Wang, Q., Yang, T., Lu, H., Cao, Y., Xu, C., *et al.*: Swarm of micro flying robots in the wild. Science Robotics **7**(66), 5954 (2022)

[12] LaValle, S.M.: Planning Algorithms. Cambridge university press, ??? (2006)

[13] Tal, E., Karaman, S.: Accurate tracking of aggressive quadrotor trajectories using incremental nonlinear dynamic inversion and differential flatness. IEEE Transactions on Control Systems Technology **29**(3), 1203–1218 (2020)

[14] Faessler, M., Franchi, A., Scaramuzza, D.: Differential flatness of quadrotor

dynamics subject to rotor drag for accurate tracking of high-speed trajectories. IEEE Robotics and Automation Letters **3**(2), 620–626 (2017)

[15] Sun, S., Romero, A., Foehn, P., Kaufmann, E., Scaramuzza, D.: A comparative study of nonlinear mpc and differential-flatness-based control for quadrotor agile flight. IEEE Transactions on Robotics (2022)

[16] Yang, S., Xian, B.: Exponential regulation control of a quadrotor unmanned aerial vehicle with a suspended payload. IEEE Transactions on Control System Technology **28**(6), 2762–2769 (2020). https://doi.org/10.1109/tcst.2019.2952826

[17] Klausen, K., Fossen, T.I., Johansen, T.A.: Nonlinear control with swing damping of a multirotor UAV with suspended load. Journal of Intelligent & Robotic Systems **88**(2-4), 379–394 (2017). https://doi.org/10.1007/s10846-017-0509-6

[18] Sreenath, K., Michael, N., Kumar, V.: Trajectory generation and control of a quadrotor with a cable-suspended load - a differentially-flat hybrid system. In: Proceedings of the IEEE International Conference on Robotics and Automation, Karlsruhe, Germany (2013). https://doi.org/10.1109/icra.2013.6631275

[19] Zeng, J., Sreenath, K.: Geometric control of a quadrotor with a load suspended from an offset. In: 2019 American Control Conference (ACC). IEEE, ??? (2019). https://doi.org/10.23919/acc.2019.8814939

[20] Jiang, Z., Al Lawati, M., Mohammadhasani, A., Lynch, A.F.: Flatness-based motion control of a uav slung load system using quasi-static feedback linearization. In: 2022 International Conference on Unmanned Aircraft Systems (ICUAS), pp. 361–368 (2022). IEEE

[21] Lynch, K.M., Park, F.C.: Modern Robotics. Cambridge University Press, ??? (2017)

[22] Nijmeijer, H., Schumacher, J.: The regular local noninteracting control problem for nonlinear control systems. SIAM Journal on Control and Optimization **24**(6), 1232–1245 (1986)

[23] Moeini, A., Lynch, A.F., Zhao, Q.: A backstepping disturbance observer control for multirotor uavs: theory and experiment. International Journal of Control, 1–15 (2021). https://doi.org/10.1080/00207179.2021.1912393. online access

[24] Nijmeijer, H., Respondek, W.: Dynamic input-output decoupling of nonlinear control systems. IEEE Transactions on Automatic Control **33**(11), 1065–1070 (1988)

[25] Mohammadhasani, A., Al Lawati, M., Jiang, Z., Lynch, A.F.: Dynamic feedback linearization of a UAV suspended load system. In: Proceedings of the

International Conference on Unmanned Aircraft Systems, Dubrovnik, Croatia (2022)

[26] Meier, L., Agar, D., Küng, B., Oes, J., Gubler, T., Riseborough, P., Grob, M., Babushkin, A., Bapst, R., Sidrane, D., px4dev, Charlebois, M., Goppert, J., Bresciani, M., Marques, N., Antener, A.D., Tridgell, A., Mannhart, D., kritz, Bot, P.B., Fuhrer, S., Whitehorn, M., Mohammed, K., Wilks, S., Kirienko, P., Sauder, M., Smeets, S., Rivizzigno, M., JaeyoungLim, Olsson, C.: PX4/PX4-Autopilot: Stable Release V1.13.0. https://doi.org/10.5281/zenodo.6682275. https://doi.org/10.5281/zenodo.6682275

[27] Koenig, N., Howard, A.: Design and use paradigms for gazebo, an open-source multi-robot simulator. In: 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566), vol. 3, pp. 2149–2154. IEEE

[28] Furrer, F., Burri, M., Achtelik, M., Siegwart, R.: RotorS—A Modular Gazebo MAV Simulator Framework. In: Koubaa, A. (ed.) Robot Operating System (ROS): The Complete Reference (Volume 1), pp. 595–625. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-26054-9_23. http://dx.doi.org/10.1007/978-3-319-26054-9_23

[29] Rudolph, J., Delaleau, E.: Some remarks on quasi-static feedback of generalized states. In: New Trends in Design of Control Systems, pp. 51–56. Elsevier, ??? (1994). https://doi.org/10.1016/S1474-6670(17)47621-X

[30] Delaleau, E., Rudolph, J.: Some examples and remarks on quasi-static feedback of generalized states. Automatica **34**(8), 993–999 (1998). https://doi.org/10.1016/s0005-1098(98)00047-8

[31] Fritsch, O., Monte, P.D., Buhl, M., Lohmann, B.: Quasi-static feedback linearization for the translational dynamics of a quadrotor helicopter. In: Proceedings of the American Control Conference, Montreal, QC, Canada (2012). https://doi.org/10.1109/acc.2012.6314682