

DATA-DRIVEN SYNTHESIS OF AIRCRAFT FEEDBACK CONTROL LAWS FOR NEW COMMERCIAL FLIGHT SCENARIOS

Szymanski, Michael[1] & Gollnick, Volker[1]

[1]Hamburg University of Technology, Institute of Air Transportation Systems, Germany

Abstract

In the past, the process for designing flight control laws in commercial aviation has mostly relied on using classical methods stemming from linear control theory. However, these methods often lack a more complete understanding of the underlying flight control problem and fail to capture system nonlinearities, account for possible multi-system behaviors, or enable climate-friendly flight. The *University Sustainable Long-Range Commercial Transport Vehicle Study* (UniSELECT) project, which is funded by the German government, aims at developing methods that allow for new forms of commercial air traffic scenarios, namely formation and air-to-air refueling flights. The proposed flight design process shall improve the conventional one by making use of modern numerical optimization and data-driven methods. This shall then enable the design of scenario-driven control architectures for new forms of sustainable air travel.

Keywords: Flight Controls, Trajectory Optimization, System Identification, Data-Driven Control Systems, Nonlinear Control Theory

1. Introduction

The conventional process for designing flight control laws focuses on developing gain-scheduled and cascaded control architectures. Their purpose is to stabilize the aircraft about a set of trimmed operational conditions, generally referred to as *trim points*, gathered within the allowable flight envelope of the aircraft [1].

This process starts by formulating a sufficiently detailed simulation model of the considered aircraft and its environment. The resulting flight simulation model is then evaluated at multiple trim points, which cover a wide range of the aircraft's allowable and tolerable flight envelope. These trim points represent the required values for the state and input vectors of the aircraft's dynamics such that a desired steady state is maintained, where pilot and autopilot control action are driven to a minimum. This is usually achieved by setting the aircraft's body-fixed translational and rotational acceleration terms to zero and solving the equations of motion in regard to the desired trim state. For example, in the design of an altitude-hold controller, this process may include identifying trim points for a horizontal steady state cruise flight based on a set of desired altitudes [2].

Linearized system models can then be derived from the original nonlinear simulation model following a 1^{st}-order Taylor series expansion about all considered trim points. This step enables control designers to assess and analyze the aircraft's system dynamics at all considered points using classical state-space and frequency-domain methods [1, 3, 2, 4]. Most commonly, state-space methods, such as direct pole placement or Linear Quadratic Regulation (LQR), are used for improving inherent system stability and dynamics of the aircraft. These systems are commonly referred to as Stability Augmentation Systems (SAS). Frequency-domain methods focus on establishing stable and robust control over a particular system variable via pilot or autopilot commands. In general, any system assisting in the realization of pilot-issued commands via manual input devices such as sticks, levers,

yokes, or pedals, for example, is referred to as a Control Augmentation System (CAS) [2]. For establishing control over higher-level variables, such as altitude, Mach number, and navigational headings for example, an autopilot system will issue the correct commands to the CAS automatically. In this case, the pilots responsibility shifts from directly controlling the aircraft to configuring the autopilot via a Flight Control Unit (FCU) for short-term changes, or a Flight Management Computer (FMC) for long-term changes to the flight plan.

The control designers may further want to cover the aircraft's entire flight envelope by constructing an adaptive parameter-varying control algorithm that enables smooth switching between multiple trim points during flight. A popular method used for these purposes is known as *Gain-Scheduling* [5, 3]. The aircraft is then able to follow a sequence of points in time (known as *reference* or *trajectory* tracking) or space (for *path following*). For example, this sequence can represent the desired flight path. The control designers may further choose to use existing software tools to develop a robust gain-scheduled control algorithm [6].

Having tested the full-envelope control algorithm in multiple simulations, the next steps include piloted simulations and real flight tests using piloted or autonomous aircraft to further tune and robustify the control algorithm. The conventional flight control design process is depicted in Figure 1.

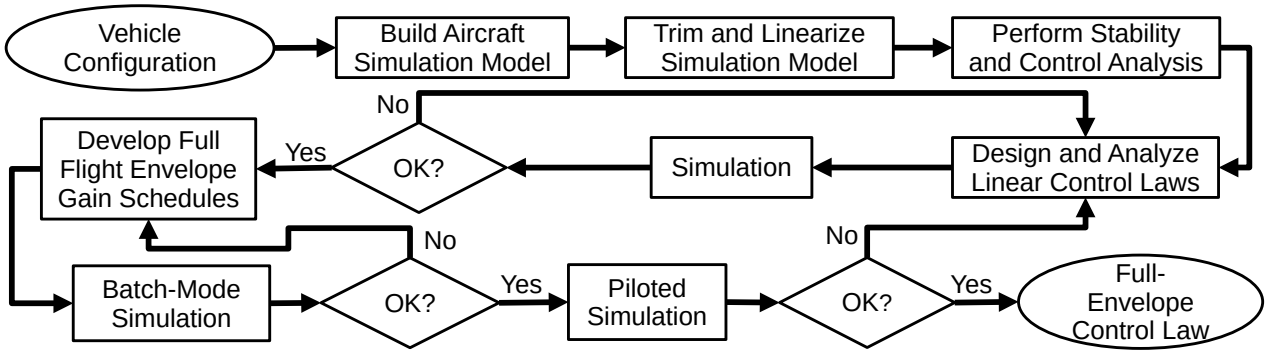


Figure 1 – Conventional Flight Control Design Process (adapted from [1])

However, there seem to exist a few shortcomings regarding this approach. Namely, and in correlation to the UniSELECT project, there does not appear to exist a unified control law design method which considers energy- or emissions-efficient flying strategies, coupled multi-aircraft scenarios, or new missions and scenarios in commercial aviation in an explicit manner. Therefore, we suggest improving and expanding the conventional control design process such that new flight scenarios can be considered more effectively within this process.

2. Proposed Methodology

First, a reference flight control architecture is developed using state-of-the-art methods, such as a robust full-envelope gain-scheduled control law [6, 4]. Simulation-based tuning is then used to iteratively improve system stability, performance, and robustness [7, 8].

If simulations indicate that the reference flight controls still fail to meet the specified robustness, performance, and efficiency goals, the new methodology identifies missing control dynamics using modern system identification techniques from data. For this purpose, goal-oriented trajectory optimizations are performed to generate optimal input-output time-series data. This data is then used to identify a nonlinear feedback controller via modern interpolation and regression methods [3, 9]. The control dynamics can then be iteratively refined until all stability, robustness, and performance requirements are met. This yields a full-envelope, goal-oriented flight control architecture. This method can also be applied when no reference control architecture is initially imposed. Figure 2 shows the proposed enhanced flight control design process.

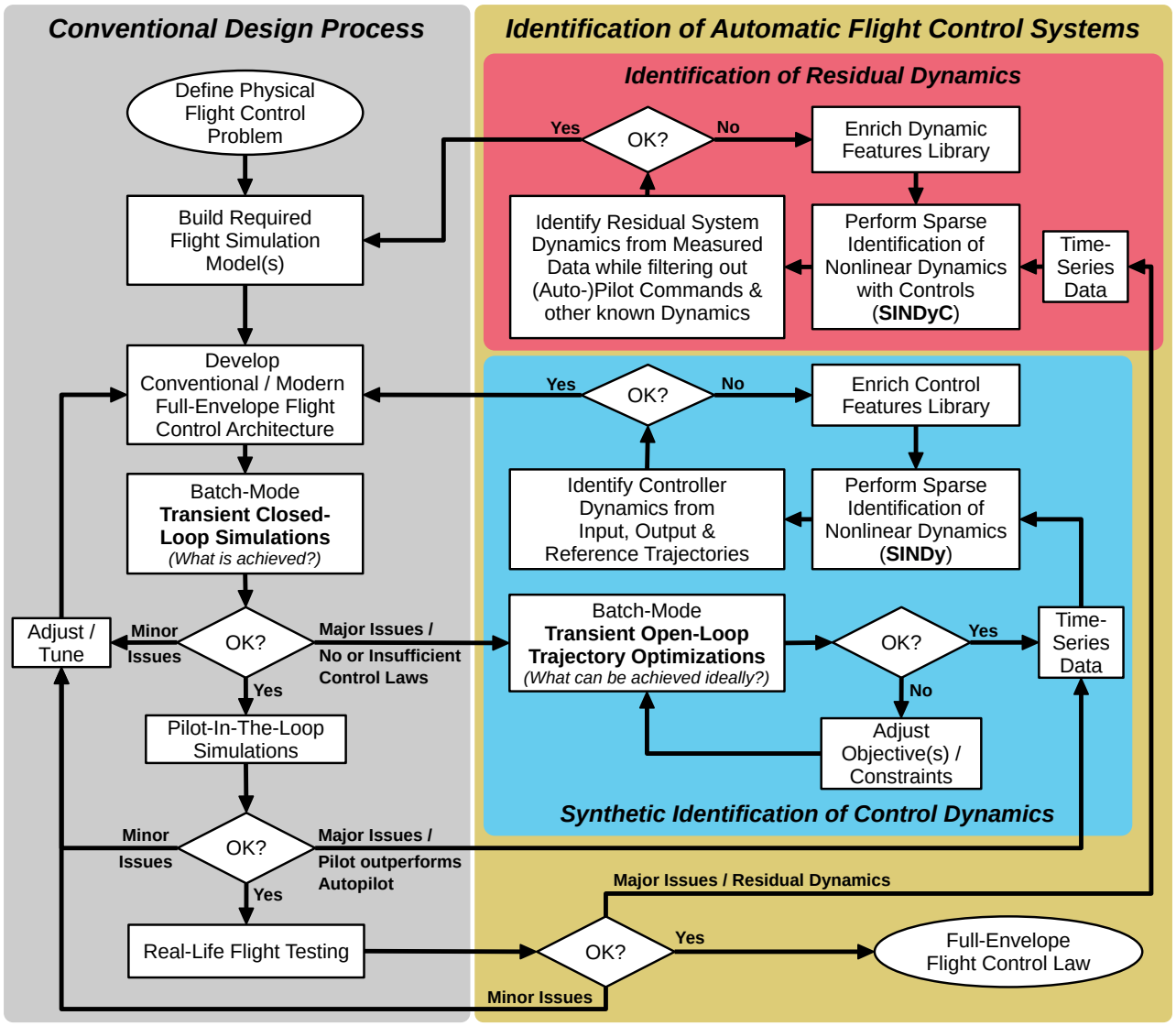


Figure 2 – Enhanced Flight Control Design Process incorporating modern Trajectory Optimization and System Identification Methods for Data-Driven Aircraft Feedback Control Synthesis

As is further suggested in Figure 2, pilot-in-the-loop simulation or real flight test data may be used at later development stages to optimize and augment the control architecture, or to identify residual system dynamics for increased model fidelity.

To provide the necessary context for the proposed methodology, Section 2.1 first examines select aspects of the conventional design process in more detail. Section 2.2 then briefly reviews modern nonlinear flight control methods currently used or under development for commercial aircraft. The proposed extensions to the conventional process, which are shown in the blue box on the right-hand side of Figure 2, are introduced in Section 3.. Finally, Section 4. presents first proof-of-concept results, in which a controller is identified from time-series data generated via multiple trajectory optimizations of a parameter-varying climb scenario.

2.1 A Review of the Conventional Design Process

The first step in both the conventional and new design process is the definition of a precise flight control problem (see left-hand side of Figure 2). This step is crucial in specifying the performance, stability, and robustness requirements of the desired flight control architecture for the evaluated aircraft in exact terms. These can include technical and operational aspects such as the vehicle configuration, the allowable flight envelope, and system and regulatory constraints. A precise problem definition also clarifies which physical systems and sub-systems must be considered, as well as the required level of model fidelity. Understanding the flight control problem that follows from the initial

requirements and the derived physical models is essential for determining *how* the problem should be represented, analyzed, and solved mathematically. Although a complete problem definition is beyond the scope of this paper, it is important to mention that the relationship between the underlying physical system and its mathematical representation is fundamental to the design of suitable flight control laws.

Having defined the flight control problem, an adequate flight simulation model is initially developed. This model incorporates typical flight dynamics equations while being further augmented to depict additional physical phenomena, if necessary. In regard to the UniSELECT project, this is the case when developing flight control laws for close formation flight or air-to-air refueling scenarios. Here, it is necessary to incorporate a mathematical model of the wake vortices produced by a leading aircraft when formation flight is considered [10, 11]. The aerodynamic effects of the leading aircraft's wingtip wake vortices on a trailing aircraft can then be modeled. This is done to evaluate and make optimal use of a physical phenomenon known colloquially as *Wake Vortex Surfing* [12]. Here, the leading aircraft creates an upwash wind field which affects any aircraft flying behind it. This field changes the airflow direction for the trailing aircraft, increasing its effective angle of attack (AoA). Consequently, this results in an increase of lift for this aircraft. The trailing aircraft then needs to decrease its own AoA to maintain level cruise flight and avoid accidental climbing. Most significantly, this decrease in AoA further results in a reduction of drag, leading to more fuel-efficient flying since less thrust is now required to maintain level cruise flight. It has been shown that overall, a V-formation consisting of three aircraft has a shared potential to reduce fuel consumption by up to $13\,\%$ between all aircraft [13]. Further, the average relative change of climate response may be mitigated by up to $24\,\%$ [14].

Given a sufficient and possibly augmented flight simulation model to incorporate additional physical effects, the process shown in Figure 1 can be used to develop an initial stabilizing control architecture for all considered steady state operational conditions. Steady state refers to all operational conditions where the aircraft finds itself in an equilibrium [5], meaning that

$$\dot{\boldsymbol{x}} = \boldsymbol{f}\left(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq}\right) = \mathbf{0}\,. \tag{1}$$

This further constitutes the principle definition of a trim point as mentioned previously. Having obtained a sufficient set of equilibrium points within the flight envelope of the aircraft, it is common practice to obtain linear system models using a 1^{st}-order Taylor series approximation of the nonlinear system about each point following

$$\dot{\boldsymbol{x}} \approx \underbrace{\boldsymbol{f}\left(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq}\right)}_{=\mathbf{0}} + \left.\frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{x}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_{eq} \\ \boldsymbol{u}=\boldsymbol{u}_{eq}}} \overbrace{\left(\boldsymbol{x} - \boldsymbol{x}_{eq}\right)}^{\tilde{\boldsymbol{x}}} + \left.\frac{\partial \boldsymbol{f}\left(\boldsymbol{x}, \boldsymbol{u}\right)}{\partial \boldsymbol{u}}\right|_{\substack{\boldsymbol{x}=\boldsymbol{x}_{eq} \\ \boldsymbol{u}=\boldsymbol{u}_{eq}}} \overbrace{\left(\boldsymbol{u} - \boldsymbol{u}_{eq}\right)}^{\tilde{\boldsymbol{u}}}\,. \tag{2}$$

This linearization naturally leads to a quasi Linear Parameter-Varying (quasi-LPV) system representation in Equation 3 due to being scheduled on endogenous system parameters such as the state and input vector [15].

$$\dot{\boldsymbol{x}} \approx \mathbf{A}\left(\boldsymbol{x}, \boldsymbol{u}\right)\Big|_{\substack{\boldsymbol{x}=\boldsymbol{x}_{eq} \\ \boldsymbol{u}=\boldsymbol{u}_{eq}}} \tilde{\boldsymbol{x}} + \mathbf{B}\left(\boldsymbol{x}, \boldsymbol{u}\right)\Big|_{\substack{\boldsymbol{x}=\boldsymbol{x}_{eq} \\ \boldsymbol{u}=\boldsymbol{u}_{eq}}} \tilde{\boldsymbol{u}}\,. \tag{3}$$

In case only a single equilibrium point is considered, the quasi-LPV system can then be reduced to a Linear Time-Invariant (LTI) system as in

$$\dot{\boldsymbol{x}} \approx \mathbf{A}_{eq}\tilde{\boldsymbol{x}} + \mathbf{B}_{eq}\tilde{\boldsymbol{u}}\,. \tag{4}$$

Provided a wide range of different steady state operational conditions, the conventional flight control design process is capable of developing stabilizing and robust flight control laws within the vicinity of each equilibrium point [3, 2, 4]. Regarding the design of robust full-envelope gain-scheduled flight control laws, this process, known as gain-scheduled H_∞-Loopshaping, has been fully analyzed and automated in [6].

2.2 Nonlinear Control Methods in Modern Flight Control

A major drawback of any type of linear control architecture remains with proving and guaranteeing formal stability and robustness requirements for the nonlinear aircraft dynamics throughout the entire flight envelope. While gain-scheduling appears successful in approximating such controls, especially when there exists a sufficiently dense multidimensional grid of equilibrium points for which robust controls had been developed, such an approach may not always reach specified performance requirements. This mainly considers transient system behavior, where these types of control laws may fail or perform insufficiently if not tested and tuned rigorously. Therefore, one may choose to employ nonlinear control techniques, instead. Considering nonlinear control theory, there exist two major groups. These consist of methods based on *Feedback Linearization* and on the notion of *Lyapunov Stability* [5]. Both are presented and summarized in the following paragraphs.

Feedback Linearization has seen popular traction in the form of model-based *Nonlinear Dynamic Inversion* (NDI) and its sensor-driven *Incremental* variant (INDI) for modern nonlinear flight control [2, 3, 16, 17, 18, 19]. Essentially, the goal of feedback linearization is to find a control law which inverts the nonlinear dynamics [5]. This way, the system's nonlinearities may be fully compensated, provided the exact full nonlinear system dynamics are known. Ideally, the resulting plant then acts as an integrator or a general LTI or LPV system, leading to an overall plant model which can now be controlled using established linear control methods. However, it should be noted that this is a purely theoretical depiction of this approach, since modeling errors are unavoidable and inaccurate inverse models may actually destabilize the real system, instead. Inverse models may not even exist in case the actual mathematical model is not invertible about all considered operational conditions within the flight envelope. To mitigate model inaccuracies and avoid further numerical issues, INDI control has seen recent popularity in multiple flight control applications as it replaces model-dependent terms with measurements, instead. NDI, INDI, and many hybrid methods have since been developed [16, 17, 18, 20]. Currently, these methods are used for modern fighter jet aircraft controls [21]. A variant of INDI has also been tested for a Cessna Citation II in 2018, making this the first application of feedback linearization to a CS-25 certified aircraft [16].

Lyapunov-based nonlinear control methods make use of the notion of *Lyapunov Stability*. Paraphrasing [5], if an autonomous nonlinear system $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x})$ and some equilibrium point $\boldsymbol{x}_{eq} = \boldsymbol{0}$ of the system are provided, and if there exists a scalar function $V(\boldsymbol{x})$ which fulfills the following three criteria:

$$V\left(\boldsymbol{x}_{eq} = \boldsymbol{0}\right) = 0, \tag{5a}$$

$$V(\boldsymbol{x}) > 0 \quad \forall \boldsymbol{x} \neq \boldsymbol{x}_{eq}, \tag{5b}$$

$$\dot{V}(\boldsymbol{x}) = \left(\frac{\partial V(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)^{\mathsf{T}} \dot{\boldsymbol{x}} = \left(\frac{\partial V(\boldsymbol{x})}{\partial \boldsymbol{x}}\right)^{\mathsf{T}} \boldsymbol{f}(\boldsymbol{x}) < 0 \quad \forall \boldsymbol{x} \neq \boldsymbol{x}_{eq}, \tag{5c}$$

then $\boldsymbol{x}_{eq}$ is *locally asymptotic stable* for every initial state vector $\boldsymbol{x}$ within a *region of attraction* of the equilibrium. The *region of attraction* is defined by all initial state vectors for which Criteria 5b and 5c are satisfied. The scalar function $V(\boldsymbol{x})$ can be interpreted as an energy-like potential that is, essentially, being minimized by the autonomous system following from Criterion 5c. Intuitively speaking, the nonlinear system $\boldsymbol{f}(\boldsymbol{x})$ acts as a minimizer to $V(\boldsymbol{x})$ as long Criterion 5c is satisfied. In principle, any positive-definite scalar function can be used following Criteria 5a and 5b, with convex functions being typical candidates. Both imply that, given an initial state vector $\boldsymbol{x}_0 \neq \boldsymbol{x}_{eq}$ within this region, there exists a trajectory in time $\boldsymbol{x}(t)$ along which the system will *strictly* converge to the equilibrium $\boldsymbol{x}_{eq}$ at some point in time. While exponential stability about the equilibrium may not be guaranteed this way, the control designers may then choose to approximate the system to be linear about the reached equilibrium following Equation 2. This way, stabilizing linear controls can then be employed, instead.

This notion of stability is exploited for the synthesis of explicit nonlinear control methods such as *Control Lyapunov Functions*, *Backstepping*, *Sliding Mode Control*, and *Total Energy Control* [5, 22, 23]. A popular implicit control method using this notion of stability is *Model Predictive Control* (MPC) [24, 25]. MPC can be understood as repeatedly solving an updated *Trajectory Optimization* problem using feedback over a finite-temporal prediction horizon in real-time. Here, the optimized cost function is typically convex.

Comparing both nonlinear control techniques, they appear to serve different purposes. Considering feedback linearization methods, their goal is to first compensate system nonlinearities by using model- or sensor-based feedback signals such that the resulting global system appears linear about all steady and transient states. Then, linear control methods can be employed to drive the system between all considered operating points. In contrast, Lyapunov-based controls appear to act on the nonlinear system directly. They drive the system to a desired equilibrium point from any initial state within a region of attraction of this point in an asymptotically stabilizing fashion. This avoids having to first develop feedback linearization laws and then additional linear controls. However, a good system model is generally required.

For the purposes of this paper, the notion of Lyapunov stability is employed by selecting a suitable cost function term for trajectory optimization according to Criteria 5a– 5b. This way, it appears possible to embed performance-based criteria, such as energy- and fuel-efficient flying, directly into the optimization problem while simultaneously considering all system constraints. Also, asymptotically stabilizing solutions may be retrieved. However, no rigorous proofs of Criterion 5c are entailed in this paper.

3. Synthesizing Aircraft Feedback Control Laws from Optimal Trajectory Data

In this section, the proposed extensions to the conventional design method found on the right-hand side of Figure 2 are introduced. Initially, the generation of optimal input trajectories using trajectory optimization is presented in Section 3.1. In order to then obtain continuous representations of surrogate control dynamics from optimal trajectory data, a modified sparse system identification method based on [26] is proposed in Section 3.2.

3.1 Generating Discrete Maps of Optimal System Trajectories between Equilibrium Points using Trajectory Optimization

If simulation results show that the conventional process fails to deliver a suitable control architecture considering the specified flight controller requirements and performance goals, numerical methods and, more specifically, trajectory optimization is employed. Treating the system simulation as an Optimal Control Problem (OCP) and transforming it into a mathematical program using appropriate methods [25, 27, 28] allows for the use of state-of-the-art numerical optimization techniques. This includes the nonlinear Interior Point Optimization method (IPOPT) developed by [29, 30] in combination with the Multifrontal Massively Parallel Sparse direct linear solver (MUMPS) developed by [31], for example. Furthermore, computer algebra programs such as CasADi integrate these methods while aiding the control designers in formulating suitable OCPs [32]. This way, solutions to optimal trajectories for all types of manipulated variables can be obtained without initially imposing any type of control architecture. When compared to the conventional control design approach, this is a crucial difference: we first obtain discrete maps of optimal input, state, and output trajectories which reach equilibrium points without initially specifying any explicit feedback control dynamics. Then, we identify the underlying control dynamics from these discrete maps using the method later described in Section 3.2.

Further, trajectory optimization enables the control designers to include explicit constraints on the aircraft dynamics, states, inputs, and outputs. Also, additional optimization goals can be included. This way, the minimization of fuel consumption over time and, consequently, climate impact can be stated explicitly. From this assertion, a general formulation of a continuous single-stage OCP is first

considered in Equation 6 following [25].

$$\min_{\boldsymbol{u}(t)} \quad J = \int_{t_0}^{t_N} L\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) dt + M\left(\boldsymbol{x}(t_N)\right) \tag{6a}$$

s.t.

$$\boldsymbol{x}(t_0) - \hat{\boldsymbol{x}} = \boldsymbol{0}, \qquad \text{(fixed initial value)}, \tag{6b}$$

$$\dot{\boldsymbol{x}}(t) - \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) = \boldsymbol{0}, \;\; t \in [t_0, t_N], \;\; \text{(system dynamics)}, \tag{6c}$$

$$\boldsymbol{l}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) \leq \boldsymbol{0}, \;\; t \in [t_0, t_N], \;\; \text{(path constraints)}, \tag{6d}$$

$$\boldsymbol{m}\left(\boldsymbol{x}(t_N)\right) \leq \boldsymbol{0}, \qquad \text{(terminal constraints)}, \tag{6e}$$

where $\boldsymbol{x}$ is the state vector, $\hat{\boldsymbol{x}}$ represents the initial state, $\boldsymbol{u}$ the input vector, $\boldsymbol{f}$ the system dynamics vector, $\dot{\boldsymbol{x}}$ the differential state vector, $\boldsymbol{l}$ a vector of inequality constraints, $\boldsymbol{m}$ a vector of terminal inequality constraints, and where L and M represent some running and terminal cost terms, respectively. Here, L and M may express the previously-mentioned optimization goals such as fuel consumption over time. The dependence of some variables and vectors on time t makes them trajectories over the finite time horizon $t \in [t_0, t_N]$, where t_0 indicates the initial and t_N the terminal point in time.

The following paragraphs include further information on performing trajectory optimizations and how certain aspects of the system model can be represented mathematically. Some of these points are later considered for the trajectory optimizations performed in Section 4..

Actuator, Control, and Output Dynamics

In case actuator, control, or output dynamics are known and provided, it is possible to extend the trajectory optimization problem to include these effects. For example, the system dynamics constraint in Equation 6c may be augmented by additional actuator and control dynamics when provided in differential form following

$$\dot{\boldsymbol{u}}(t) - \boldsymbol{g}\left(\boldsymbol{x}_{aug}(t), \boldsymbol{r}_{cmd}(t)\right) = \boldsymbol{0}, \;\; t \in [t_0, t_N], \;\; \text{(actuator/control dynamics)}, \tag{7}$$

where $\boldsymbol{x}_{aug} = [\boldsymbol{x}, \;\; \boldsymbol{u}]^\intercal$ is the augmented state variable with additional actuator or control states. Following this augmentation, more detailed actuator dynamics or known control architectures can be included, if required. Note that $\boldsymbol{u}$ is now a differential state, while $\boldsymbol{r}_{cmd}$ may now represent a different set of input vector variables instead of $\boldsymbol{u}$, thus changing the input variables under consideration. Further, Equation 7 can now be used to represent other types of controls. In case gain-scheduled and cascaded controls already exist for an aircraft model, these can then be modeled and added following Equation 7. Due to the resulting change in input variables this causes, one can now obtain optimal trajectories for the controller gains, instead. Alternatively, it is now possible to optimize relevant parameters of an underlying gain scheduling surface representing the gains. These may be, for example, the coefficients to Nth-order polynomial functions which represent such surfaces.

In case the state vector cannot be fully retrieved through measurements or state estimation methods, or in case additional output measurements are considered for various purposes, one may further include measured outputs through output dynamics constraints as in Equation 8.

$$\boldsymbol{y}(t) - \boldsymbol{h}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) = \boldsymbol{0}, \;\; t \in [t_0, t_N], \;\; \text{(output dynamics)}. \tag{8}$$

Equations 7 and 8 demonstrate that there is a level of flexibility within general OCP frameworks. Such frameworks can hence be adapted to best fulfill the needs of the control designers when defining their physical flight control problem in mathematical terms. A key observation is that the OCP can be reduced and expanded depending on the information available regarding the simulation model and flight control requirements.

Selection of a Cost Term

In case system trajectories are optimized following the notion of Lyapunov stability, a suitable function may be included within the Lagrange term L of Equation 6a. As mentioned previously, the goal is to find optimal input and state trajectories which best minimize Equation 6a under a set of system

constraints. For these purposes, the notion of asymptotic stability in an energy-optimal sense is furthered using the total and balanced energy terms described in [23]. This was originally used for deriving their *Total Energy Control System* (TECS). Following [23],

$$J = \int_{t_0}^{t_N} \frac{1}{2}\tilde{e}_{TECS}^2(t)\,dt + \int_{t_0}^{t_N} \boldsymbol{e}_u^{\mathsf{T}}(t)\,\mathbf{R}\,\boldsymbol{e}_u(t)\,dt \tag{9}$$

is considered as a possible cost term, where $\tilde{e}_{TECS}^2$ describes the normalized squared sum of total and balanced system energy terms relative to an equilibrium point's system energies. In more detail,

$$\frac{1}{2}\tilde{e}_{TECS}^2 = \frac{1}{2}\left(\tilde{e}_{tot}^2 + \tilde{e}_{bal}^2\right) = \frac{1}{2}\left(\left(\tilde{e}_{tot,des} - \tilde{e}_{tot}\right)^2 + \left(\tilde{e}_{bal,des} - \tilde{e}_{bal}\right)^2\right), \tag{10}$$

with normalized energy terms

$$\tilde{e}_{tot} = \frac{1}{\mathrm{m}\cdot g}\cdot\left(\frac{\mathrm{w}_{V_A}}{2}\cdot\mathrm{m}\cdot\left(\frac{V_A}{V_{avg}}\right)^2 + \mathrm{w}_h\cdot\mathrm{m}\cdot g\cdot\frac{h}{h_{avg}}\right), \text{ and} \tag{11}$$

$$\tilde{e}_{bal} = \frac{1}{\mathrm{m}\cdot g}\cdot\left(\frac{\mathrm{w}_{V_A}}{2}\cdot\mathrm{m}\cdot\left(\frac{V_A}{V_{avg}}\right)^2 - \mathrm{w}_h\cdot\mathrm{m}\cdot g\cdot\frac{h}{h_{avg}}\right), \tag{12}$$

where w_{V_A} and w_h are optional scaling weights. Further, $\boldsymbol{e}_u = \boldsymbol{u}_{des} - \boldsymbol{u}$ is the input error term, whereas $\mathbf{R}$ is a diagonal positive-definite weighing matrix, penalizing and scaling certain input error terms more strongly than others depending on context. Similarly, parameters V_{avg} and h_{avg} are used to scale their respective variables such that the cost term J later returns solutions around 1.0. The reasons for employing scaling parameters, weights, input penalization, and imposing positive-definiteness on $\mathbf{R}$ are not trivial and discussed in [28, 33, 34, 35, 36, 27]. This is not only related to increasing the penalty on certain cost terms such as fuel burn, time, or similar, for example, but is also related to problem conditioning and whether an optimization problem is well-posed or not.

Numerical Trajectory Optimization

Considering the final goal of obtaining input trajectories for optimal trim point transitions under system constraints, the trajectory optimization problem in Equation 13 is proposed

$$\min_{\boldsymbol{u}(t)} \quad J = \int_{t_0}^{t_N} \frac{1}{2}\tilde{e}_{TECS}^2(t)\,dt + \int_{t_0}^{t_N} \boldsymbol{e}_u^{\mathsf{T}}(t)\,\mathbf{R}\,\boldsymbol{e}_u(t)\,dt \tag{13a}$$

s.t.

$$\boldsymbol{x}(t) - \hat{\boldsymbol{x}} = \mathbf{0}, \quad t = t_0, \qquad \text{(fixed initial value)}, \tag{13b}$$

$$\dot{\boldsymbol{x}}(t) - \boldsymbol{f}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) = \mathbf{0}, \quad t \in [t_0, t_N], \quad \text{(\textit{augmented} system dynamics)}, \tag{13c}$$

$$\boldsymbol{y}(t) - \boldsymbol{h}\left(\boldsymbol{x}(t), \boldsymbol{u}(t)\right) = \mathbf{0}, \quad t \in [t_0, t_N], \quad \text{(output dynamics)}, \tag{13d}$$

$$\boldsymbol{l}\left(\boldsymbol{x}(t), \boldsymbol{u}(t), \boldsymbol{y}(t)\right) \leq \mathbf{0}, \quad t \in [t_0, t_N], \quad \text{(path constraints)}, \tag{13e}$$

$$\boldsymbol{y}(t_N) - \boldsymbol{y}_{des} = \mathbf{0}, \qquad \text{(terminal boundary value constraint)}, \tag{13f}$$

$$\underline{\boldsymbol{x}}(t) \leq \boldsymbol{x}(t) \leq \overline{\boldsymbol{x}}(t), \qquad t \in [t_0, t_N], \quad \text{(state bounds)}, \tag{13g}$$

$$\underline{\boldsymbol{y}}(t) \leq \boldsymbol{y}(t) \leq \overline{\boldsymbol{y}}(t), \qquad t \in [t_0, t_N], \quad \text{(output bounds)}, \tag{13h}$$

$$\underline{\boldsymbol{u}}(t) \leq \boldsymbol{u}(t) \leq \overline{\boldsymbol{u}}(t), \qquad t \in [t_0, t_N], \quad \text{(input bounds)} \tag{13i}$$

where, and in comparison to Equation 6, additional system outputs $\boldsymbol{y}$ are added, $\dot{\boldsymbol{x}}$ already considers a possibly augmented dynamical system, a terminal constraint on select system outputs $\boldsymbol{y}$ is imposed, and where lower and upper bounds on the states, inputs, and outputs are considered. The terminal constraint signifies that at the end of the output trajectory, a set of outputs $\boldsymbol{y}_{des} = \boldsymbol{y}_{eq} = \boldsymbol{h}\left(\boldsymbol{x}_{eq}, \boldsymbol{u}_{eq}\right)$ must be reached. This presents the general approach for obtaining optimal trajectories for the trim point transition problem of the nonlinear aircraft model later presented in Section 4..

3.2 Obtaining Synthetic Control Laws from Optimal Input & Output Trajectories using Sparse Identification of Nonlinear Systems (SINDy)

Once a suitable amount of mission-related simulation results from multiple trajectory optimizations is obtained, it may be possible to detect the underlying control dynamics using modern system identification methods as described in [9, 37]. The field of system identification has seen many new developments over recent years [26, 38, 9, 39]. One such development is the Sparse Identification of Nonlinear System Dynamics (SINDy) method developed by [26]. Given a library of multiple nonlinear candidate functions, the SINDy algorithm is able to identify a system of *best fit* nonlinear functions describing the governing equations. This has the potential to be used not only for identifying the governing system equations of autonomous or externally controlled systems $\boldsymbol{f}(\boldsymbol{x})$ and $\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$, respectively, but also the governing control dynamics given multiple time-series of mission-related discrete trajectory optimization results. This way, it may be possible to synthesize goal-specific explicit controls.

The approach that will be tested in Section 4. considers the discovery of an input map as a differential function of the output error and its incremental integral error

$$\dot{\boldsymbol{u}} = \boldsymbol{g}\left(\boldsymbol{e}_y, \Delta\boldsymbol{e}_{I,y}\right). \tag{14}$$

This can be interpreted as an attempt at discovering a form of Differential Proportional Integral (DPI) control [40, 20]. Following the method described in [26] and adapting it to the discovery of differential control dynamics as postulated in Equation 14, discrete time-series trajectories is first collected. This consists of input, output error, and incremental integral output error data obtained from optimal transient solutions following Equation 13 for different combinations of initial states and equilibrium points $\hat{\boldsymbol{x}} \rightarrow \boldsymbol{y}_{eq}$.

First, we construct a general discrete error vector $\mathbf{e}$. In our case, it consists of the discrete output error and incremental integral output error trajectories at times $t_0, t_1, \ldots, t_N$ with discrete entries

$$\mathbf{e}(t_k) = \left[\boldsymbol{e}_y^{\mathsf{T}}(t_k), \Delta\boldsymbol{e}_{I,y}^{\mathsf{T}}(t_k)\right]^{\mathsf{T}} \quad , \text{ for } k \in [0,N]. \tag{15}$$

The output error data is then structured in a matrix as in

$$\mathbf{E} = \begin{bmatrix} \mathbf{e}^{\mathsf{T}}(t_0) \\ \mathbf{e}^{\mathsf{T}}(t_1) \\ \vdots \\ \mathbf{e}^{\mathsf{T}}(t_N) \end{bmatrix} = \begin{bmatrix} e_1(t_0) & e_2(t_0) & \ldots & e_m(t_0) \\ e_1(t_1) & e_2(t_1) & \ldots & e_m(t_1) \\ \vdots & \vdots & \ddots & \vdots \\ e_1(t_N) & e_2(t_N) & \ldots & e_m(t_N) \end{bmatrix}, \tag{16}$$

and the input rate as in

$$\dot{\mathbf{U}} = \begin{bmatrix} \dot{\mathbf{u}}^{\mathsf{T}}(t_0) \\ \dot{\mathbf{u}}^{\mathsf{T}}(t_1) \\ \vdots \\ \dot{\mathbf{u}}^{\mathsf{T}}(t_N) \end{bmatrix} = \begin{bmatrix} \dot{u}_1(t_0) & \dot{u}_2(t_0) & \ldots & \dot{u}_l(t_0) \\ \dot{u}_1(t_1) & \dot{u}_2(t_1) & \ldots & \dot{u}_l(t_1) \\ \vdots & \vdots & \ddots & \vdots \\ \dot{u}_1(t_N) & \dot{u}_2(t_N) & \ldots & \dot{u}_l(t_N) \end{bmatrix}, \tag{17}$$

where $e_0(t_k), e_1(t_k), \ldots, e_m(t_k)$ and $\dot{u}_0(t_k), \dot{u}_1(t_k), \ldots, \dot{u}_l(t_k)$ for time steps $k \in [0,N]$ are any general scalar error or input rate value found in a single vector $\mathbf{e}(t_k)$ or $\dot{\mathbf{u}}(t_k)$, respectively. Further, the indices m and l indicate the lengths of vectors $\mathbf{e}$ and $\dot{\mathbf{u}}$, respectively. The discrete input rate trajectories $\dot{u}(t_k)$ may be derived numerically from $u(t)$ at times $t = t_k$ using finite differences [26]. They may also be obtained from measurements, if available. This is in contrast to the original purpose of SINDy, which entails the identification of systems more akin to the error dynamics $\dot{\boldsymbol{e}}$ instead of $\dot{\boldsymbol{u}}$. While this may be useful in context to some other applications hinted at in Figure 2, it will, however, not be investigated here. A later discussion considering identifying residual dynamics is held in Section 6..

Then, a library $\boldsymbol{\Theta}(\mathbf{E})$ is constructed. This library may consist of constant, polynomial, trigonometric,

or any linear and nonlinear algebraic function as indicated in Equation 18.

$$\boldsymbol{\Theta}(\mathbf{E}) = \begin{bmatrix} | & | & | & | & & | & | & \\ \mathbf{I} & \mathbf{E} & \mathbf{E}^{P_2} & \mathbf{E}^{P_3} & \dots & \sin(\mathbf{E}) & \cos(\mathbf{E}) & \dots \\ | & | & | & | & & | & | & \end{bmatrix}, \tag{18}$$

where, for example,

$$\mathbf{E}^{P_2} = \begin{bmatrix} e_1^2(t_0) & e_1(t_0)\,e_2(t_0) & \dots & e_2^2(t_0) & \dots & e_m^2(t_0) \\ e_1^2(t_1) & e_1(t_1)\,e_2(t_1) & \dots & e_2^2(t_1) & \dots & e_m^2(t_1) \\ \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ e_1^2(t_N) & e_1(t_N)\,e_2(t_N) & \dots & e_2^2(t_N) & \dots & e_m^2(t_N) \end{bmatrix}. \tag{19}$$

Assuming that only a few entries of $\boldsymbol{\Theta}(\mathbf{E})$ are active in the control dynamics, a sparsity coefficient matrix $\boldsymbol{\Xi}$, with vector entries $\boldsymbol{\Xi} = [\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \dots, \boldsymbol{\xi}_l]$, is introduced. This matrix consists of coefficients which determine the active and dominant terms in $\boldsymbol{\Theta}(\mathbf{E})$. The purpose of SINDy is to then identify the values of those coefficients by solving the following least-squares regression problem

$$\underset{\boldsymbol{\Xi}}{\arg\min} \quad \|\dot{\mathbf{U}} - \boldsymbol{\Theta}(\mathbf{E})\,\boldsymbol{\Xi}\|_2^2 \ . \tag{20}$$

The exact method used for this regression problem is the Sequentially Thresholded Least Squares algorithm (STLSQ). Initially, a threshold parameter $\lambda > 0$ is set. Then, Equation 20 is first solved

$$\boldsymbol{\Xi}_0 = \boldsymbol{\Theta}^{\dagger}(\mathbf{E})\,\dot{\mathbf{U}} \tag{21}$$

to obtain an initial coefficient matrix $\boldsymbol{\Xi}_0$. Next, any coefficients $\left|\xi\right| \leq \lambda$ inside $\boldsymbol{\Xi}_0$ are permanently set to zero. Then, $\boldsymbol{\Xi}_0$ is used as an initial guess for $\tilde{\boldsymbol{\Xi}}$ when solving

$$\underset{\tilde{\boldsymbol{\Xi}}}{\arg\min} \quad \|\dot{\mathbf{U}} - \boldsymbol{\Theta}(\mathbf{E})\,\tilde{\boldsymbol{\Xi}}\|_2^2 + a\|\tilde{\boldsymbol{\Xi}}\|_2^2 \ , \tag{22}$$

where $a \geq 0$ is an optional weight on the L2-Regularization term $\|\tilde{\boldsymbol{\Xi}}\|_2^2$. This term is added in order to smooth the regression problem for improved convergence and noise reduction due to actual system noise or numerical approximation errors [26]. Once a solution is obtained, it is checked whether there are any coefficients $\left|\tilde{\xi}\right| \leq \lambda$ inside $\tilde{\boldsymbol{\Xi}}$. As was done previously in the first step, these coefficients are now also permanently set to zero. Then, Equation 22 is again solved with the updated and now sparser guess for $\tilde{\boldsymbol{\Xi}}$, and the entire process repeated until subsequent iterates of $\tilde{\boldsymbol{\Xi}}$ do not change significantly. This implies convergence of the optimal sparse coefficient matrix $\boldsymbol{\Xi}_{opt} = \tilde{\boldsymbol{\Xi}}_{opt}$. Once this occurs, an approximation of the governing equations of the differential input dynamics is obtained using

$$\dot{\boldsymbol{u}} \approx \boldsymbol{g}_{sindy}(\boldsymbol{e}) = \boldsymbol{\Xi}_{opt}^{\mathsf{T}}\left(\boldsymbol{\Theta}(\boldsymbol{e}^{\mathsf{T}})\right)^{\mathsf{T}}, \tag{23}$$

where $\boldsymbol{e} = \left[\boldsymbol{e}_y, \Delta\boldsymbol{e}_{I,y}\right]^{\mathsf{T}}$ is a symbolic error vector instead of a data matrix. This leads to a symbolic expression of Equation 14, retrieving an approximation of the optimal differential control dynamics from synthetic trajectory optimization data.

Using the `Python`-based toolbox PySINDy, this process is fully automated [39]. Provided the required time-series data of output errors, their incremental integral values, and inputs, PySINDy automatically retrieves a symbolic expression of Equation 23. The coefficient of determination R^2 is then also obtained, providing a measure on how well a simulation of Equation 23 performs compared to the actual time-series input data. A value of $R^2 = 1.0$ suggests an ideal fit, while values $R^2 \in [0.8, 0.95]$ suggest good model fits. Values $R^2 < 0.5$ suggest poor fits, in which case it may be required to further enrich the function library $\boldsymbol{\Theta}$ in Equation 18 with additional terms, or filter out noise [41].

This way, it appears possible to synthesize entirely new control architectures for the new commercial flight scenarios considered in UniSELECT. Hypothetically speaking, it may even occur that for a given

simulation scenario, SINDy may identify a system of linear equations to best fit the optimal control dynamics. This way, one may not only be able to synthesize an optimal control architecture, but simultaneously verify that certain control architectures are already satisfactory considering a specific scenario objective and constraints. As a result, this new methodology shall allow for a better informed and integrated control design approach for commercial aircraft considering new flight scenarios in an effort to reduce civil aviation's climate impact.

4. Trajectory Optimization and Control Dynamics Identification Results considering Climb Transitions of a Nonlinear Longitudinal Aircraft Model

In a first attempt at retrieving synthetic feedback control laws from data using the modified SINDy approach postulated in Section 3.2, multiple trajectory optimizations of aircraft climb scenarios between different trim points are performed. For these purposes, a nonlinear longitudinal aircraft model is first derived, and its trim points obtained in Sections 4.1 and 4.2, respectively. Then, optimal input and output time-series data is obtained by solving Equation 13 for varying initial states $\hat{x}$ and equilibrium points x_{eq} considering climb maneuvers in Section 4.3. Finally, synthetic controls are identified from optimal trajectory data and evaluated using the modified SINDy approach in Section 4.4.

4.1 Aircraft Simulation Model

For testing purposes, the nonlinear model of the longitudinal dynamics of the F16 fighter jet aircraft found in [42, 2] is used. This is a scaled high-fidelity nonlinear model obtained from extensive real-life testing campaigns employing identification techniques outlined in [42]. The system dynamics for the longitudinal aircraft model are described as follows:

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{p}_x \\ \dot{h} \\ \dot{u}_b \\ \dot{w}_b \\ \dot{\theta} \\ \dot{q}_b \\ (\dot{\eta})_{lim} \\ \dot{P} \end{bmatrix} = \begin{bmatrix} u_b \cdot \cos(\theta) + w_b \cdot \sin(\theta) \\ u_b \cdot \sin(\theta) - w_b \cdot \cos(\theta) \\ -g \cdot \sin(\theta) + \frac{1}{\mathrm{m}} \left(\overline{q} \cdot S_{ref} \cdot C_X + F \right) \\ g \cdot \cos(\theta) + \frac{1}{\mathrm{m}} \left(\overline{q} \cdot S_{ref} \cdot C_Z \right) \\ q_b \\ \frac{1}{I_Y} \cdot \left(\overline{q} \cdot S_{ref} \cdot \bar{c} \cdot C_M \right) \\ \left(\frac{1}{\tau_\eta} \cdot (\eta_{cmd} - \eta) \right)_{lim} \\ \frac{1}{\tau_P} \cdot (P_{cmd} - P) \end{bmatrix}, \tag{24}$$

where the state vector

$$\boldsymbol{x} = [p_x \;\; h \;\; u_b \;\; w_b \;\; \theta \;\; q_b \;\; \eta \;\; P]^\mathsf{T} \tag{25}$$

includes the x-position in the inertial frame p_x, the altitude h, the x- and z-axis body-fixed velocities u_b and w_b, respectively, the pitch angle θ, the pitch rate q_b, the elevator angle η, and the engine's power level P. Further,

$$\overline{q} = \frac{1}{2} \cdot \rho_A \cdot V_A^2 \tag{26}$$

$$V_A = \sqrt{u_b^2 + w_b^2} \tag{27}$$

are the dynamic pressure and true airspeed, respectively, assuming no wind. Considering gravity g, air density ρ_A, and the ambient temperature T_A when retrieving the Mach number $Ma = \sqrt{\kappa \cdot R_s \cdot \mathrm{T}_A}$ later on, these are nonlinear functions of the altitude h assuming a nominal gravitational and atmospheric model of earth as defined in [43]. Here, the isentropic exponent $\kappa = 1.4$ and the specific gas

constant for air $R_s = 287.06\,\mathrm{J}\cdot\mathrm{K}^{-1}\cdot\mathrm{kg}^{-1}$ are also taken from [43].
Regarding the aerodynamic coefficients in body-fixed coordinates C_X, C_Z, and C_M, they are represented by high-order polynomial terms as functions of the angle of attack $\alpha = \tan^{-1}\left(\frac{w_b}{u_b}\right)$, the elevator angle η, the normalized pitch rate $\tilde{q} = \frac{q_b \cdot \bar{c}}{2 \cdot V_A}$, and the difference $x_{rel} = x_{rel,cg,ref} - x_{rel,cg}$ between the reference body-fixed x-axis position of the aircraft's center of gravity (CoG) and the actual one relative to $\bar{c}$. The exact polynomial representations of the aerodynamic coefficients are found in [44].
An additional elevator rate limit is also integrated. This limit is enforced explicitly using smooth hyperbolic tangent functions, and leads to the following expression for

$$\begin{aligned}(\dot{\eta})_{lim} = \dot{\eta}\cdot\frac{1}{4}\cdot\left(1+\tanh\left(\frac{1}{\varkappa}\cdot\left(\dot{\eta}^{+}_{lim}-\dot{\eta}\right)\right)\right)\cdot\left(1+\tanh\left(\frac{1}{\varkappa}\cdot\left(\dot{\eta}-\dot{\eta}^{-}_{lim}\right)\right)\right)\\ +\dot{\eta}^{+}_{lim}\cdot\frac{1}{2}\cdot\left(1+\tanh\left(\frac{1}{\varkappa}\cdot\left(\dot{\eta}-\dot{\eta}^{+}_{lim}\right)\right)\right)+\dot{\eta}^{-}_{lim}\cdot\frac{1}{2}\cdot\left(1+\tanh\left(\frac{1}{\varkappa}\cdot\left(\dot{\eta}^{-}_{lim}-\dot{\eta}\right)\right)\right)\end{aligned} \tag{28}$$

with $\dot{\eta}^{+}_{lim} = -\dot{\eta}^{-}_{lim} = 60\,\mathrm{deg}\cdot\mathrm{s}^{-1}$ and a scaling variable $\varkappa > 0$, selected to be $\varkappa = 100$. This specific choice of $\varkappa$ starts to drive $\dot{\eta}$, which represents the unconstrained dynamics, to zero at a precision of two digits behind the decimal point once one of the limits is being approached. Simultaneously, that respective limit is then also recovered, smoothly saturating $\dot{\eta}$.
The remaining parameters related to the dynamic, mass, and geometric properties of the F16 model are found throughout [45, 42, 2]. They are compiled in Table 2 and already converted to SI units.

Name	Symbol	Value	Unit
Mass	m	9,298.60	kg
Moment of Inertia about body-fixed y-axis	I_Y	75,673.62	$\mathrm{kg}\cdot\mathrm{m}^2$
Mean Aerodynamic Cord	$\bar{c}$	3.45	m
Reference Wing Area	S_{ref}	27.87	m^2
Wing Span	$\bar{b}$	9.144	m
Relative CoG reference position	$x_{rel,cg,ref}$	.35	–
Relative CoG actual position	$x_{rel,cg}$	.25	–
Elevator Time Constant	τ_η	.0495	s
Power Level Time Constant	τ_P	$.\overline{3}$	s

Table 1 – Mass, Geometric, and Dynamic Properties of the nonlinear F16 aircraft model obtained from [42, 2].

Further, the output vector, where the flight path angle $\gamma = \theta - \alpha$, and the input vector, where the throttle position is denoted by δ_F, are defined as

$$\boldsymbol{y} = [h \;\; V_A \;\; \alpha \;\; \gamma]^\mathsf{T} \;, \tag{29}$$

$$\boldsymbol{u} = [\eta_{cmd} \;\; \delta_F]^\mathsf{T} \;, \tag{30}$$

respectively. All system variables and their respective bounds are found in Table 2.

State Variables	Symbol	Lower Bound	Upper Bound	Unit
x-position in the inertial frame	p_x	0.0	∞	m
Altitude	h	0.0	$15,240$	m
Body-fixed x-axis velocity	u_b	$-\infty$	∞	$\mathrm{m}\cdot\mathrm{s}^{-1}$
Body-fixed z-axis velocity	w_b	$-\infty$	∞	$\mathrm{m}\cdot\mathrm{s}^{-1}$
Pitch angle	θ	-89.999	89.999	deg
Pitch rate	q_b	$-\infty$	∞	$\mathrm{deg}\cdot\mathrm{s}^{-1}$
Elevator angle	η	-25.0	25.0	deg
Elevator rate limit	$\dot{\eta}_{lim}$	-60.0	60.0	$\mathrm{deg}\cdot\mathrm{s}^{-1}$
Power level	P	0.0	100.0	$\%$
Output Variables	Symbol	Lower Bound	Upper Bound	Unit
Altitude	h	0.0	$15,240$	m
True Airspeed	V_A	0.0	350.0	$\mathrm{m}\cdot\mathrm{s}^{2}$
Angle of Attack	α	-10.0	15.0	deg
Flight Path Angle	γ	-89.999	89.999	deg
Input Variables	Symbol	Lower Bound	Upper Bound	Unit
Elevator Command	η_{cmd}	-25.0	25.0	deg
Throttle Command	δ_F	0.0	1.0	$-$

Table 2 – System variables and bounds of the longitudinal F16 simulation model by [42].

It is important to mentioned that a simplified engine model is used due to the original one being discontinuous [42]. This considers the throttle-to-thrust relation via the power level response $\dot{P}$ in Equation 24. While all non-differentiable but continuous terms found within this relation are approximated using smooth transition functions, such as the hyperbolic tangent function similar to Equation 28, or linearly interpolated using CasADi's internal `interpolant` function, the original power level response model $\dot{P}$ in Equation 24 is a discontinuous function. Here, τ_P is discontinuously and non-smoothly scheduled based off of values of P and P_{cmd}, while P_{cmd} is also a discontinuous and non-smooth function of an input power command $P_{cmd,inp}(\delta_F)$. To clarify the throttle-to-thrust relation:

$$\delta_F, P \rightarrow P_{cmd,inp}(\delta_F) \rightarrow P_{cmd}\left(P_{cmd,inp}\right) \rightarrow \tau_P\left(P_{cmd,inp}, P_{cmd}, P\right) \rightarrow \dot{P}\left(P_{cmd}, P, \tau_P\right) \rightarrow F\left(P, Ma, \mathrm{T}_A\right). \quad (31)$$

While it was possible to obtain a smooth approximation for the original model of $\dot{P}$ using B-Splines within CasADi's `interpolant` method leading to $R^2 \approx 0.995$, this made solving the trajectory optimization problems rather difficult later in Section 4.3. Therefore, a simpler model for $\dot{P}$ is used in Equation 24 based off of [4]. A detailed explanation of the throttle-to-thrust relation is also found in [45], while its implementation in the `Fortran` programming language is provided in [2]. For further improvements to the model in the future, the use of an appropriate surrogate model for the engine dynamics may alleviate the convergence issues encountered during trajectory optimizations [46].

This concludes the description of the nonlinear longitudinal F16 aircraft simulation model used in this paper.

4.2 Obtaining Trim Points via Implicit Numerical Optimization

In general, trim points can be obtained by solving

$$\min_{\boldsymbol{u}} \quad J_{trim} = \boldsymbol{u}^{\mathrm{T}}\tilde{\mathbf{R}}\boldsymbol{u} \quad (32a)$$

$$\text{s.t.}$$

$$\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u}) = \mathbf{0}, \quad (\text{system dynamics}), \quad (32b)$$

$$\boldsymbol{y} - \boldsymbol{h}(\boldsymbol{x},\boldsymbol{u}) = \mathbf{0}, \quad (\text{output dynamics}), \quad (32c)$$

$$\underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}}, \quad (\text{state bounds}), \quad (32d)$$

$$\underline{\boldsymbol{y}} \leq \boldsymbol{y} \leq \overline{\boldsymbol{y}}, \quad (\text{output bounds}), \quad (32e)$$

$$\underline{\boldsymbol{u}} \leq \boldsymbol{u} \leq \overline{\boldsymbol{u}}, \quad (\text{input bounds}). \quad (32f)$$

For the case of steady state horizontal flight, we consider setting the altitude h to be a fixed parameter in order to obtain multiple trim points for a range of different altitudes. For this case, we define

$$\underline{h} = h = \overline{h} \text{ for } h \in [0.0\,\mathrm{m}, 500.0\,\mathrm{m}, \ldots, 5000.0\,\mathrm{m}] \,. \tag{33}$$

Further, the state p_x and its differential term for $\dot{p}_x$ are excluded from the problem in Equation 32 as we do not consider trim points to be located at fixed longitudes or latitudes. Since the model is highly nonlinear, there may exist multiple combinations of elevator and throttle positions leading to steady state horizontal flight. This implies the existence of multiple possible trim points for a single altitude. In order to obtain preferential trim points for energy-efficient flying, we penalize excessive throttle use more heavily than elevator commands by setting $\mathbf{diag}\left(\tilde{\mathbf{R}}\right) = \left[10^{-6}, 10^{0}\right]^{\mathsf{T}}$. This leads to trim points which demand lower engine power and, subsequently, where thrust is minimal. Consequently, low-drag trim points are obtained. $\tilde{\mathbf{R}} > \mathbf{0}$ also regularizes the problem in a numerical sense to avoid ill-posedness of the trim problem [28, 36, 25].

Provided the aircraft model from Section 4.1, the steady state horizontal trim problem in Equation 32 is implemented in CasADi within the `Python` programming language. All resulting optimization problems are then solved in parallel using `Python`'s `multiprocessing` module. All calculations are performed using IPOPT and MUMPS within CasADi on an `AMD Ryzen`$^{\text{TM}}$ `5950X` 16 cores/32 threads desktop processor, with each core clocked up to $4.9\,\mathrm{GHz}$, and $64\,\mathrm{GB}$ of DDR4 random acccess memory (RAM) running at $3{,}600\,\mathrm{MHz}$. This also describes the general setup used for all calculations performed in this paper.

The relevant results for all trim point calculations are compiled in Table 3.

Trim Point (TP)	h [m]	V_A $\left[\mathrm{m\cdot s^{-1}}\right]$	α [deg]	γ [deg]	q_b $\left[\mathrm{deg\cdot s^{-1}}\right]$	P [%]	δ_F [−]	η [deg]
TP_1	0.0	123.24	3.638	0.0	0.0	6.417	0.099	−4.932
TP_2	500.0	125.98	3.662	0.0	0.0	6.732	0.104	−4.943
TP_3	1000.0	128.80	3.689	0.0	0.0	7.056	0.109	−4.956
TP_4	1500.0	131.70	3.714	0.0	0.0	7.387	0.114	−4.970
TP_5	2000.0	137.53	3.505	0.0	0.0	7.670	0.118	−4.866
TP_6	2500.0	140.84	3.521	0.0	0.0	7.941	0.122	−4.874
TP_7	3000.0	144.25	3.539	0.0	0.0	8.219	0.127	−4.883
TP_8	3500.0	147.67	3.566	0.0	0.0	8.639	0.133	−4.896
TP_9	4000.0	151.17	3.356	0.0	0.0	9.076	0.140	−4.911
TP_{10}	4500.0	154.77	3.630	0.0	0.0	9.519	0.147	−4.928
TP_{11}	5000.0	158.46	3.670	0.0	0.0	9.969	0.154	−4.946

Table 3 – Trim Point results from solving Equation 32 for horizontal low-thrust steady state flight at altitudes $h \in [0.0\,\mathrm{m}, 500.0\,\mathrm{m}, \ldots, 5000.0\,\mathrm{m}]$.

Inspecting the pitch rates q_b and flight path angles γ of all trim points reveals that steady state horizontal flight is successfully achieved in all cases, since $\gamma = 0.0\,\mathrm{deg}$ suggests level flight, while $q_b = 0.0\,\mathrm{deg\cdot s^{-1}}$ means that the aircraft is not rotating. This further implies that $\theta = \alpha$. Changing the weights of $\tilde{\mathbf{R}}$ by a factor of 10 or $.1$ did not impact solution outcome in any significant way. These results are used to form the climb transition problems in Section 4.3.

4.3 Obtaining System Trajectories by solving Optimal Control Problems

To obtain optimal system trajectories for the climb transition problem between all trim points listed in Table 3, the two-point boundary value problem in Equation 32 is first discretized using trapezoidal collocation [28]. This formulation yields a sparse optimization problem that is well suited for gradient-based solvers such as IPOPT. Moreover, the numerical solution is always guaranteed to converge to the actual one for decreasing step sizes Δt due to the trapezoidal method being A-stable [28, 25].

In order to perform trim point transitions considering climb trajectories, it follows from Table 3 that

$$\mathrm{TP}_i \rightarrow \mathrm{TP}_j \quad \forall i, j \in [1, 11] \wedge i < j \;\; . \tag{34}$$

Then, the energy-optimal trajectory optimization problem is formulated and solved following

$$\min_{\mathbf{X},\mathbf{U},\mathbf{Y}} \quad J = J_{TECS} + J_{\boldsymbol{u}} \tag{35a}$$

s.t.

$$\mathbf{x}_0 - \hat{\boldsymbol{x}} = \mathbf{0}, \qquad \text{(fixed initial value)}, \tag{35b}$$

$$\mathbf{x}_{k+1} - \mathbf{F}\left(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}, \mathbf{x}_k, \mathbf{u}_k\right) = \mathbf{0}, \;\; k \in [0, N-1], \;\; \text{(\textit{augmented} system dynamics)}, \tag{35c}$$

$$\mathbf{y}_k - \boldsymbol{h}\left(\mathbf{x}_k, \mathbf{u}_k\right) = \mathbf{0}, \;\; k \in [0, N], \;\; \text{(output dynamics)}, \tag{35d}$$

$$\boldsymbol{l}\left(\mathbf{x}_k, \mathbf{u}_k, \mathbf{y}_k\right) \leq \mathbf{0}, \;\; k \in [0, N], \;\; \text{(path constraints)}, \tag{35e}$$

$$\mathbf{y}_N - \boldsymbol{y}_{des} = \mathbf{0}, \qquad \text{(terminal constraint)}, \tag{35f}$$

$$T - T_{max} = 0, \qquad \text{(duration constraint)}, \tag{35g}$$

$$\underline{\mathbf{X}} \leq \mathbf{X} \leq \overline{\mathbf{X}}, \qquad \text{(state bounds)}, \tag{35h}$$

$$\underline{\mathbf{Y}} \leq \mathbf{Y} \leq \overline{\mathbf{Y}}, \qquad \text{(output bounds)}, \tag{35i}$$

$$\underline{\mathbf{U}} \leq \mathbf{U} \leq \overline{\mathbf{U}}, \qquad \text{(input bounds)}, \tag{35j}$$

where

$$J_{\boldsymbol{u}} = \sum_{k=0}^{N-1} \frac{1}{2} \cdot \left(\mathbf{e}_{u,k+1}^{\mathsf{T}} \mathbf{R}\, \mathbf{e}_{u,k+1} + \mathbf{e}_{u,k}^{\mathsf{T}} \mathbf{R}\, \mathbf{e}_{u,k} \right) \cdot \frac{\Delta t}{T}, \tag{36}$$

$$J_{TECS} = \sum_{k=0}^{N-1} \frac{1}{4} \cdot \left(\tilde{\mathbf{e}}_{TECS,k+1}^2 + \tilde{\mathbf{e}}_{TECS,k}^2 \right) \cdot \Delta t \tag{37}$$

and where $\mathbf{x}_k = \boldsymbol{x}(t_k)$, $\mathbf{u}_k = \boldsymbol{u}(t_k)$, $\mathbf{y}_k = \boldsymbol{y}(t_k)$, $\tilde{\mathbf{e}}_{TECS,k} = \tilde{\boldsymbol{e}}_{TECS}(t_k)$, and $\mathbf{e}_{u,k} = \boldsymbol{e}_u(t_k)$ for steps $k \in [0, N]$ when compared to Equation 13. Additionally,

$$\Delta t = \frac{T}{N}, \tag{38}$$

$$\mathbf{F}\left(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}, \mathbf{x}_k, \mathbf{u}_k\right) = \mathbf{x}_k + \frac{\Delta t}{2} \left(\mathbf{f}(\mathbf{x}_{k+1}, \mathbf{u}_{k+1}) - \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k) \right), \tag{39}$$

$$\mathbf{X} = \left[\mathbf{x}_0^{\mathsf{T}}, \mathbf{x}_1^{\mathsf{T}}, \ldots, \mathbf{x}_N^{\mathsf{T}}\right]^{\mathsf{T}}, \tag{40}$$

$$\mathbf{Y} = \left[\mathbf{y}_0^{\mathsf{T}}, \mathbf{y}_1^{\mathsf{T}}, \ldots, \mathbf{y}_N^{\mathsf{T}}\right]^{\mathsf{T}}, \tag{41}$$

$$\mathbf{U} = \left[\mathbf{u}_0^{\mathsf{T}}, \mathbf{u}_1^{\mathsf{T}}, \ldots, \mathbf{u}_N^{\mathsf{T}}\right]^{\mathsf{T}}, \tag{42}$$

and

$$\mathbf{R} = \begin{bmatrix} \mathrm{w}_\eta \cdot \left|\eta_{cmd,tol}\right|^{-2} & 0 \\ 0 & \mathrm{w}_F \cdot \left|\delta_{F,tol}\right|^{-2} \end{bmatrix} \tag{43}$$

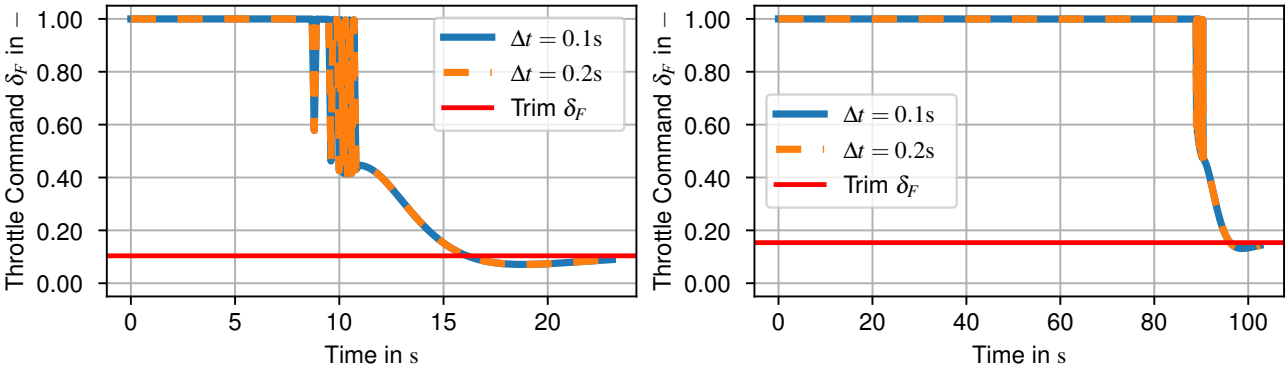
for improved problem scaling [35]. Here, $\eta_{cmd,tol}$ and $\delta_{F,tol}$ represent the tolerance that shall be achieved for the difference between desired (meaning trimmed) and actual input values. Furthermore, normalizing the costs in Equation 36 over time is found to obtain solutions independent of the duration. This means that regularization occurs evenly along the numerical grid independent of time since $\frac{\Delta t}{T} = N^{-1}$. For the energy costs in Equation 37, this is not the goal since the system's total and balanced energy shall be minimized relatively quickly, hence no time normalization occurs. Adding weights such as w_η, w_F, and those found in Equations 11, for example, allows prioritizing certain optimization goals over others. Similarly, certain elements within the total energy cost term J_{TECS} may also need to be penalized more heavily. This is necessary due to having observed slow responses considering V_A. Here, it is necessary to normalize all gravity terms g, effectively canceling g out by setting $\mathrm{w}_{V_A} = 9.80$. The values for h_{avg} and V_{avg} are averages of their respective trim points found in Table 3. The simulation duration is considered a fixed parameter set to $T = 160\,\mathrm{s}$ for all trajectory optimizations. Results will then be truncated to the earliest point in time t_k^* where $\boldsymbol{y}\left(t_k^*\right) \approx \boldsymbol{y}_{des}$ in a post-processing step. To provide initial guesses for all state, output, and input trajectories, linear interpolation is performed between all initial and terminal points over time assuming $T = T_{max}$ [47]. This leads to a total of 55 parallel and scaled trajectory optimization problems.

For demonstration purposes, we consider the edge cases $\mathrm{TP}_1 \rightarrow \mathrm{TP}_2$ and $\mathrm{TP}_1 \rightarrow \mathrm{TP}_{11}$. Having selected appropriate scaling weights able to obtain satisfactory optimal solutions, initial results for the

settings found in Table 4 reveal numerical issues considering the throttle command signal δ_F as visualized in Figures 3a and 3b.

h_{avg} [m]	V_{avg} $\left[\mathrm{m}\cdot\mathrm{s}^{-1}\right]$	w_h [−]	w_{V_A} [−]	$\eta_{cmd,tol}$ [deg]	$\delta_{F,tol}$ [−]	w_η [−]	w_F [s]
2500.0	140.0	1.0	10.0	1.0	0.1	1.0^{-2}	1.0^{-3}

Table 4 – Initial trajectory optimization settings for energy-optimal climb.



(a) $\mathrm{TP}_1 \rightarrow \mathrm{TP}_2$ (b) $\mathrm{TP}_1 \rightarrow \mathrm{TP}_{11}$

Figure 3 – Throttle results for cases $\mathrm{TP}_1 \rightarrow \mathrm{TP}_2$ (a) and $\mathrm{TP}_1 \rightarrow \mathrm{TP}_{11}$ (b) using two different step sizes $\Delta t \in [0.1\,\mathrm{s}, 0.2\,\mathrm{s}]$.

As observed in Figure 3, δ_F appears to chatter during the switch from 1.0 (full throttle) to each cases' respective trim value $\delta_{F,trim}$. This is caused by the fact that δ_F has, effectively, unlimited bandwidth. Looking at both Figures 3a and 3b, this is quite evident: in both cases and for all step sizes considered, δ_F switches arbitrarily between maximum and minimum values before settling to its respective trim value. It is therefore necessary to investigate this type of behavior more thoroughly. As suggested in [48], rate limit constraints can be introduced to mitigate this type of behavior. Further, such constraints can be motivated physically because all physical systems experience inertia [49], meaning that instantaneous system responses are, in principle, not possible. Therefore, it appears necessary to improve the model found in [42]. This is done by augmenting the system dynamics with an additional integrator representing the throttle command's rate of change to better capture realistic signal behavior [48].

Following the suggestions in [48], a 1^{st}-order lag is applied to both the elevator command η_{cmd} and throttle input δ_F signals. This actively limits signal bandwidth and can be understood as a low-order human pilot model inputting commanded values to the aircraft [4, 3]. The system dynamics in Equation 24 are hence augmented by a human pilot model following

$$\dot{\boldsymbol{x}}_{Pilot} = \begin{bmatrix} \dot{\eta}_{cmd} \\ \dot{\delta}_F \end{bmatrix} = \begin{bmatrix} \frac{1}{\tau_{Pilot}} \left(\eta_{cmd} - \eta_{Pilot}\right) \\ \frac{1}{\tau_{Pilot}} \left(\delta_F - \delta_{F,Pilot}\right) \end{bmatrix}, \tag{44}$$

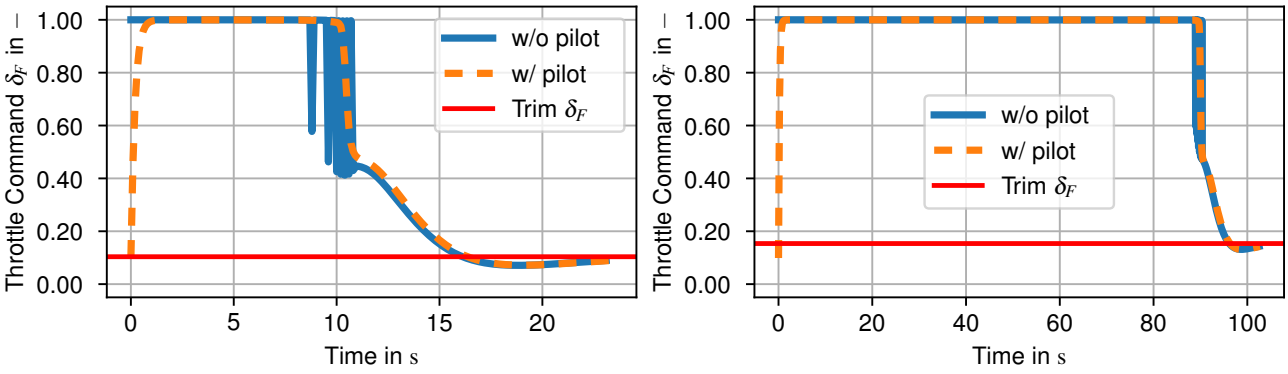
with $\tau_{Pilot} = 0.2\mathrm{s}$. This augments the system state vector to

$$\boldsymbol{x}_{aug} = [p_x \;\; h \;\; u_b \;\; w_b \;\; \theta \;\; q_b \;\; \eta \;\; P \;\; \eta_{cmd} \;\; \delta_F]^\mathsf{T} \tag{45}$$

and changes the input vector to

$$\boldsymbol{u}_{aug} = \left[\eta_{Pilot} \;\; \delta_{F,Pilot}\right]^\mathsf{T}. \tag{46}$$

Comparing the results found in Figure 3 to the augmented system $\dot{\boldsymbol{x}}$, no numerical chattering is observed when adding the human pilot dynamics in Figure 4. Simultaneously, similar trajectories for δ_F are maintained outside the chattering regions.



(a) $TP_1 \rightarrow TP_2$ (b) $TP_1 \rightarrow TP_{11}$

Figure 4 – Throttle results for cases $TP_1 \rightarrow TP_2$ (a) and $TP_1 \rightarrow TP_{11}$ (b) with and without human pilot dynamics for $\Delta t = 0.1\,\mathrm{s}$.

Having reconfigured the system model and adjusted the final trajectory optimization settings for energy-optimal climb in Table 5 for improved performance, the final results for both edge cases $TP_1 \rightarrow TP_2$ and $TP_1 \rightarrow TP_{11}$ are presented in Figure 5 and 6, respectively. These now include the human pilot model described in Equation 44.

h_{avg} [m]	V_{avg} $\left[\mathrm{m \cdot s^{-1}}\right]$	w_h [–]	w_{V_A} [–]	$\eta_{cmd,tol}$ [deg]	$\delta_{F,tol}$ [–]	w_η [–]	w_F [s]	τ_{Pilot} [s]
2500.0	140.0	1.0	20.0	0.5	0.05	1.0^{-2}	1.0^{-3}	0.2

Table 5 – Final trajectory optimization settings for energy-optimal climb.

For the purposes of this paper, we can conclude that both edge cases converge to meaningful solutions for step sizes $\Delta t \leq 0.2\mathrm{s}$ with the human pilot model employed. Since numerical chattering in the input signals is no longer observed, the input signals η_{cmd} and δ_F can now be used for nonlinear control dynamics identification in Section 4.4. These signals are taken instead of the pilot-issued commands η_{Pilot} and $\delta_{F,Pilot}$ since it is assumed that the autopilot stemming from the identification process will issue commands at much faster sampling times than the pilot. Consequently, these are expected to be much closer to η_{cmd} and δ_F instead of η_{Pilot} or $\delta_{F,Pilot}$, respectively.

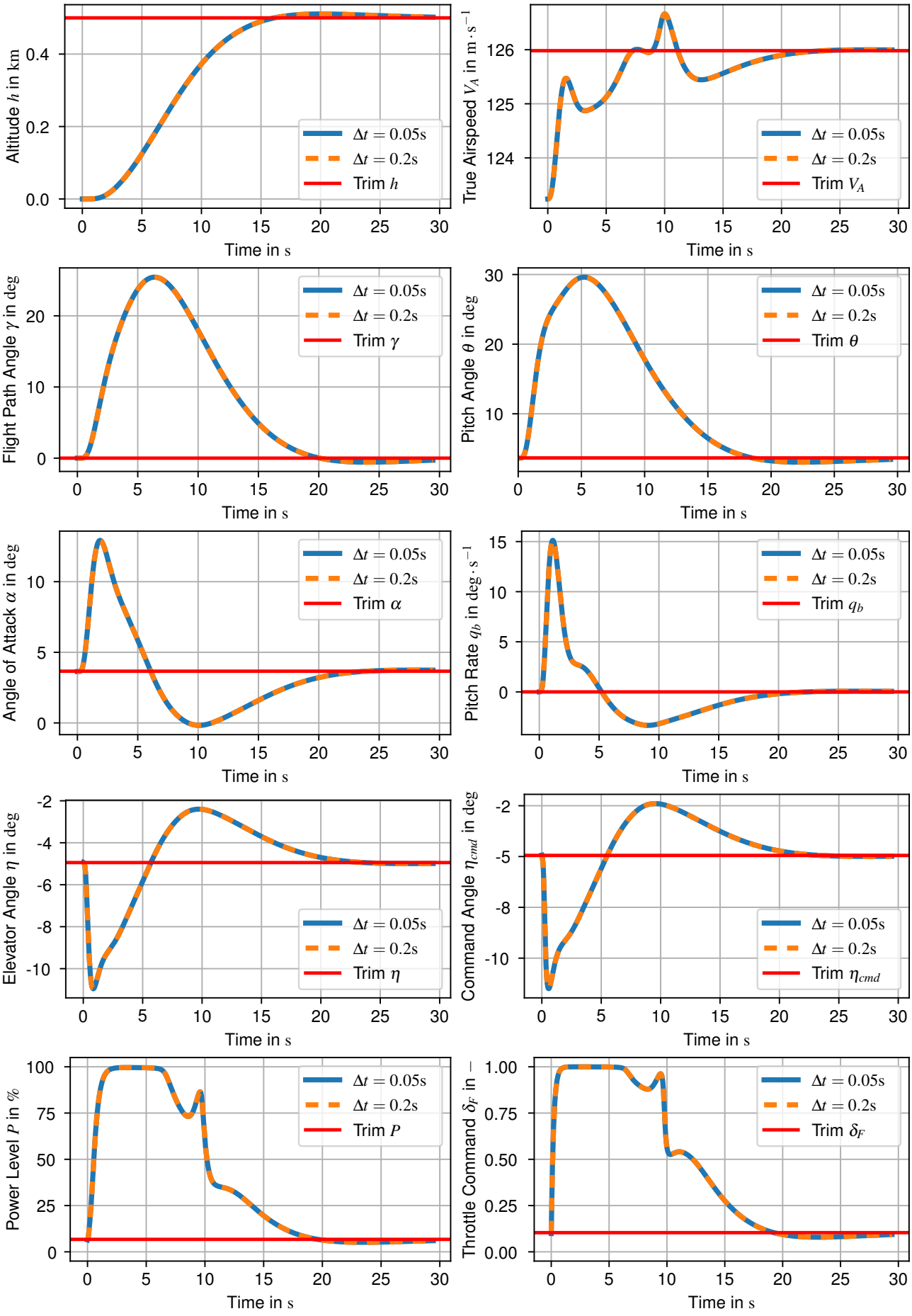


Figure 5 – Trajectory optimization results for $\mathrm{TP}_1 \rightarrow \mathrm{TP}_2$ for step sizes $\Delta t \in [0.05\,\mathrm{s}, 0.2\,\mathrm{s}]$ including the human pilot model from Equation 44.

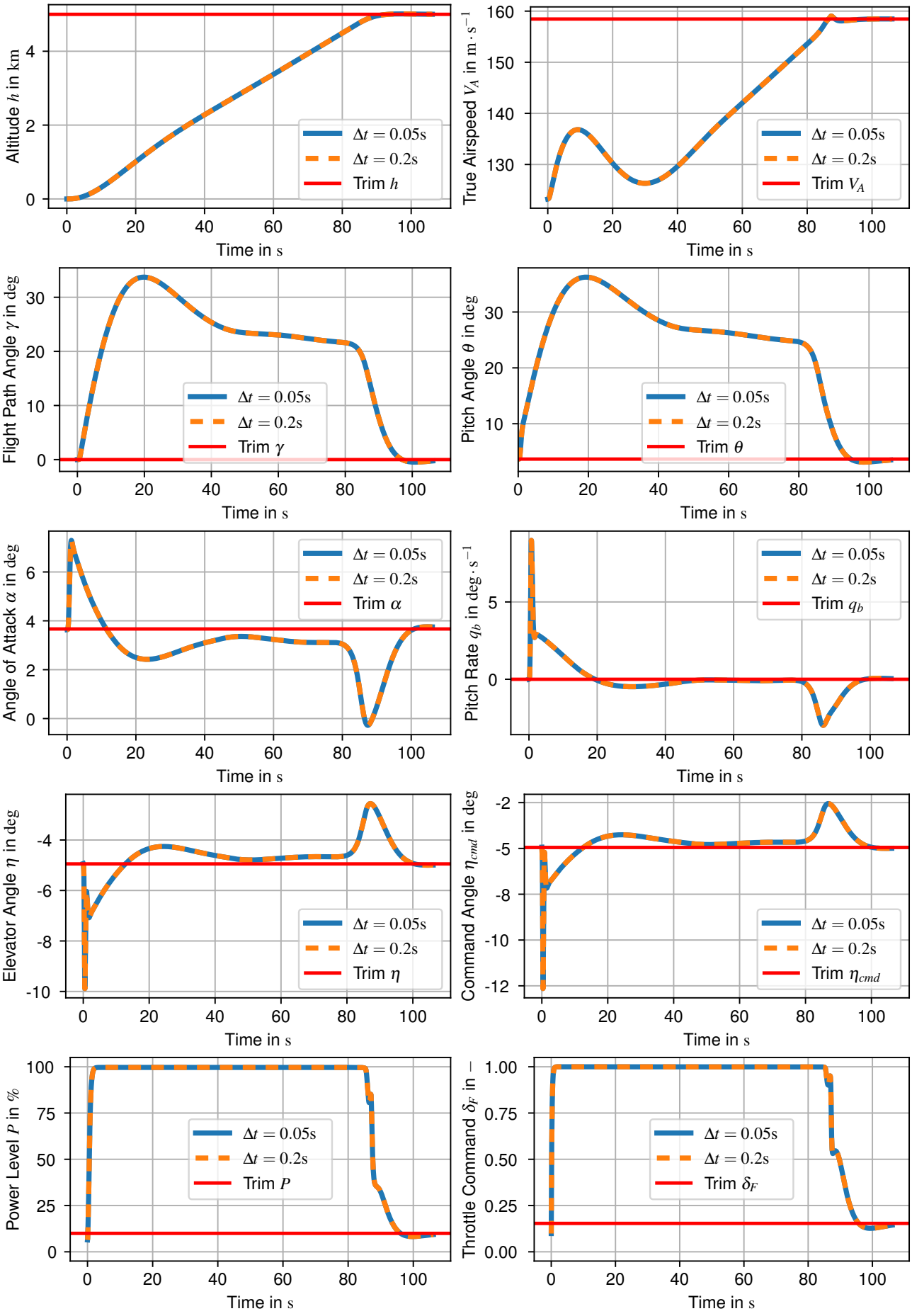


Figure 6 – Trajectory optimization results for $\mathrm{TP}_1 \rightarrow \mathrm{TP}_{11}$ for step sizes $\Delta t \in [0.05\,\mathrm{s}, 0.2\,\mathrm{s}]$ including the human pilot model from Equation 44.

4.4 Identifying Control Dynamics from Optimal Trajectories

Having obtained optimal solutions to multiple variations of the climb problem for all considered trim point combinations, the resulting time-series data is now collected and inputted to the modified SINDy algorithm as presented in Section 3.2. First, the discrete trajectories for the output errors and input signal rates are obtained following Equations 16 and 17, respectively. Considering the input rates of change, these are automatically approximated in PySINDy when providing the discrete input trajectories and time grid using finite-differences [39].

Using STLSQ, initial control dynamics are identified. For these purposes, the altitude, true airspeed, and AoA errors are considered. All three error trajectories each include the regular error $\boldsymbol{e}_y$ and their respective incremental integral error $\Delta\boldsymbol{e}_{I,y}$ obtained through numerical integration using the trapezoidal method. Considering the altitude and true airspeed error terms, these are exclusively fitted onto N^{th}-order polynomial terms. Regarding the AoA angular error terms, these are only fitted onto N^{th}-order sine and cosine terms following a Fourier-series expansion. Further, the effects of cross-coupled terms via multiplication will also be investigated. The STLSQ algorithm settings are compiled in Table 6.

Symbol	λ	a	Finite-Diff. Degree	Max. Iterations	Column Normalization	Unbias
Value	0.1	0.0	6	50	True	True

Table 6 – STLSQ settings for control dynamics identification.

Finally, possible mappings of $\dot{\eta}_{cmd}$ and $\dot{\delta}_F$ are obtained using a library $\boldsymbol{\Theta}(\mathbf{E})$ consisting of the aforementioned nonlinear functions. Through this library, a surrogate model is then fitted against the input rate data as a function of the error data (see Section 3.2). Initial results are compiled in Tables 7 and 8 for $\dot{\eta}_{cmd}$ and $\dot{\delta}_F$, respectively.

Max. Poly. order	Max. Fourier order	Term Coupling?	R^2 for $TP_1 \rightarrow TP_2$	R^2 for $TP_1 \rightarrow TP_{11}$	R^2 for $TP_1 \rightarrow TP_j$	R^2 for $TP_i \rightarrow TP_j$
1	1	False	.328	.090	.059	.252
2	1	False	.716	.187	.063	.328
3	1	False	.772	.292	.065	.341
4	1	False	.835	.364	.076	.352
1	2	False	.429	.229	.091	.282
2	2	False	.814	.507	.099	.366
3	2	False	.841	.550	.10	.379
4	2	False	.923	.576	.115	.385
1	1	True	.973	.707	.241	.364
2	1	True	.999	.977	.776	.747
3	1	True	.999	.997	.952	.847
4	1	True	.999	.999	.976	.890
1	2	True	.994	.843	.280	.414
2	2	True	.999	.988	.850	.798
3	2	True	.999	.999	.971	.877
4	2	True	.999	.999	.985	.907

Table 7 – Control dynamics identification results for $\dot{\eta}_{cmd}\left(\boldsymbol{e}_y, \Delta\boldsymbol{e}_{I,y}\right)$ considering multiple variations of possible surrogate models for different climb problem datasets ($\Delta t = 0.05\text{s}$).

Max. Poly. order	Max. Fourier order	Term Coupling?	R^2 for $TP_1 \rightarrow TP_2$	R^2 for $TP_1 \rightarrow TP_{11}$	R^2 for $TP_1 \rightarrow TP_j$	R^2 for $TP_i \rightarrow TP_j$
1	1	False	.631	.396	.305	.289
2	1	False	.807	.658	.351	.346
3	1	False	.950	.823	.373	.370
4	1	False	.959	.849	.386	.378
1	2	False	.730	.40	.317	.301
2	2	False	.861	.842	.360	.355
3	2	False	.966	.866	.382	.381
4	2	False	.986	.922	.392	.390
1	1	True	.954	.724	.404	.336
2	1	True	.999	.994	.840	.716
3	1	True	.999	.999	.943	.860
4	1	True	.999	.999	.969	.907
1	2	True	.985	.945	.439	.381
2	2	True	.999	.997	.880	.764
3	2	True	.999	.999	.961	.885
4	2	True	.999	.999	.981	.924

Table 8 – Control dynamics identification results for $\dot{\delta}_F\left(e_y, \Delta e_{I,y}\right)$ considering multiple variations of possible surrogate models for different climb problem datasets ($\Delta t = 0.05\text{s}$).

As seen in Tables 7 and 8, adding higher-order polynomial and Fourier-series terms while also allowing term coupling via multiplication (cf. Equation 19) results in better approximation for $\dot{\eta}_{cmd}\left(e_y, \Delta e_{I,y}\right)$ and $\dot{\delta}_F\left(e_y, \Delta e_{I,y}\right)$, respectively. This trend is evidenced by the coefficient of determination R^2 tending closer to 1.0. Therefore, it can be concluded that by enriching the feature library in Equation 18, it seems possible to reproduce not only input trajectories for a single case, but also for multiple variations of the same problem.

However, one persistent issue found throughout the results in Tables 7 and 8 is the notion of overfitting. It appears that for the singular cases $TP_1 \rightarrow TP_2$ and $TP_1 \rightarrow TP_{11}$, the optimal control dynamics are depicted sufficiently well using low-order models. While this problem is not further considered in this paper, it can be addressed and mitigated using regularization methods by setting $a > 0.0$, and cross-validation techniques to evaluate how well an identified model will generalize to independent datasets [9].

Ideally speaking, if it is found that an identified control system modeled by lower-order models (e. g., 1^{st}-order polynomials) produces satisfactory results in simulations, it can then be concluded that for this specific task, linear types of control appear sufficient. This notion may even be exploited to see whether certain types of Lyapunov-based controllers can be extracted or reproduced when using specific Lyapunov functions (e. g., TECS by [23]) in the cost term of the trajectory optimization problem.

5. Conclusion

The final result of this paper sees an enhanced control design process aided by modern numerical optimization and system identification methods found in Figure 2. The proposed methodology in Section 2. enables the identification of optimal control policies from multiple trajectory optimizations. It utilizes state-of-the-art nonlinear programming and optimization techniques described in Section 3.1 to optimize simulations of a highly nonlinear aircraft model found in Section 4.1. The results are then processed, and optimal control maps identified following a modified sparse system identification method derived in Section 3.2.

Considering the results obtained in Section 4., it can be concluded that the new trim and trajectory optimization programs based on CasADi, which includes IPOPT and MUMPS, appears to work well with the longitudinal F16 aircraft model found in [42]. While numerical chattering was observed for some trim point climb transitions in the throttle input signal, possible error sources regarding the numerical approximation and aircraft model used have been identified and mitigated. Having obtained

multiple trim points in Section 4.2 and optimal trajectories for the climb problem in Section 4.3, the SINDy algorithm is used to obtain possible mappings of the control dynamics in Section 4.4. By increasing the order of the polynomials and trigonometric functions while also allowing for term coupling via multiplication, improved solutions for approximate control mappings are observed using the R^2 metric.

In conclusion, it is successfully shown that task-specific nonlinear control dynamics can be synthesized from discrete time-series input-output data following the proposed methodology in this paper. Using the methods described in Section 3., nonlinear controls can hence be synthesized from optimal trajectory data when employing the modified SINDy algorithm formulated in Section 3.2.

6. Outlook & Research Proposal

The results of this paper provide new grounds for synthesizing aircraft flight controls from data. They conclude the initial application of the innermost loop found on the right-hand-side of Figure 2 in the blue box. By providing an adequate flight simulation model, numerical trajectory optimizations can provide synthetic data to the modified SINDy algorithm to identify optimal control dynamics. This process can be extended by first designing a baseline control architecture and then identifying, for example, optimal gain surfaces for gain-scheduled PI controls. Nominal control architectures can therefore be improved by adding new possible types of control terms via the feature library, provided an analytic representation of these controls exists. Convergence is ultimately achieved once flight simulations of the aircraft models using the enhanced control architecture suggest similar results as obtained via trajectory optimizations.

While a proof-of-concept is achieved, the identified control dynamics must now be cross-validated [9]. Initially, this includes generating additional training data by running further trajectory optimizations which consider more trim points. This is done such that the control dynamics can cover a wider spectrum of possible climb problem variations and altitude ranges. Then, validation data is generated by also running additional trajectory optimizations. However, this data now considers slightly perturbed initial and terminal trim points when compared to the training data. Further, the step size is also perturbed to try and remove any bias introduced by discretization. The trained model can then be simulated using the perturbed initial conditions and validated for generalizability [9].

As hinted at in the outermost loop on the right-hand-side of Figure 2, SINDy can be used in combination with real flight test data in order to identify residual dynamics to better represent unmodeled aircraft dynamics or environmental effects. Provided the data is first filtered through an existing model, which also includes the control dynamics, the remaining error data can be fitted onto a model using SINDy with Controls (SINDyC) [50]. The resulting residual models can then be used to further augment the current aircraft system model in an effort to increase model fidelity.

Alternatively, it seems possible to extend the proposed methodology with methods based on Koopman Operator Theory [37, 51]. In general, Koopman Operator Theory seeks to identify possible coordinate transforms for a nonlinear dynamical system such that the resulting global system appears linear. This is achieved by lifting the system into a higher dimension and introducing additional observables to the system. This way, a compact nonlinear system may be decomposed to a sparse linear one. Assuming a lifted linear representation for the nonlinear model exists, Extended Dynamic Mode Decomposition (EDMD) can be used to identify the parameters for such a model from simulation data of the original nonlinear system using SINDy. Once an approximate lifted linear model is identified and validated, linear control methods can be applied provided the observables can be measured in the real system.

Following [18, 19], there exist initial attempts at combining hybrid-based INDI controls with H_∞ gain-scheduling methods as in [6]. It may be interesting to see whether this process can be further improved using Koopman Operator Theory and EDMD. If a lifted linear system model of the nonlinear aircraft dynamics can be identified and the additional observables measured in the real system, sensor-based INDI may then be applied in such a way that specific nonlinearities are better addressed and compensated. Most easily though, designing a single robust control for a lifted linear system model may compensate for the need to develop or simplify gain-scheduling algorithms. Hypothetically speaking, using only a single robust control parameterization without any interpolation

schemes reduces the risk of driving the system through unstable state-space regions. Therefore, if a lifted linear model of the nonlinear process is found, robust stability for each point inside the flight envelope may be guaranteed.

7. Contact Author Email Address

Michael Szymanski: michael.szymanski@tuhh.de
Prof. Dr.-Ing. Volker Gollnick: volker.gollnick@tuhh.de
Mailing Address:
Institute of Air Transportation Systems
Harburger Schlossstr. 22a
21079 Hamburg, Germany

8. Funding Acknowledgment

The *University Sustainable Long-Range Commercial Transport Vehicle Study* (UniSELECT) project is part of the aviation research program "LuFo Climate VI, 3rd call" supported by the German Federal Ministry for Economic Affairs and Energy (20E2228D). The authors gratefully appreciate the support of this research.

Supported by:
Federal Ministry for Economic Affairs and Energy
on the basis of a decision by the German Bundestag

9. Copyright Statement

The authors confirm that they, and/or their company or organization, hold copyright on all of the original material included in this paper. The authors also confirm that they have obtained permission, from the copyright holder of any third party material included in this paper, to publish it as part of their paper. The authors confirm that they give permission, or have obtained permission from the copyright holder of this paper, for the publication and distribution of this paper as part of the ICAS proceedings or as individual off-prints from the proceedings.

References

[1] Blight J, Dailey R, and Gangsaas D. Practical control law design for aircraft using multivariable techniques. *Int. J. Control*, 59(1):93–137, 1994.

[2] Stevens B, Lewis F, and Johnson E. *Aircraft Control and Simulation*. Wiley, 3rd edition, 2016.

[3] Stengel R. *Flight Dynamics*. Princeton University Press, 2nd edition, 2022.

[4] Brockhaus R, Alles W, and Luckner R. *Flugregelung*. Springer, 3rd edition, 2011.

[5] Adamy J. *Nonlinear Systems and Controls*. Springer, 2nd edition, 2024.

[6] Apkarian P and Noll D. The H_∞ Control Problem is Solved, 2017.

[7] Soltesz K, Grimholt C, and Skogestad S. Simultaneous design of proportional–integral–derivative controller and measurement filter by optimisation. *IET Control Theory Appl.*, 11(3):341–348, 2016.

[8] Moritz N and Osterhuber R. Three–Stage Gradient–Based optimization scheme in design of feedback gains within Eurofighter Primary Control Laws. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 6 2006.

[9] Brunton S and Kutz J. *Data–Driven Science and Engineering*. Cambridge University Press, 2nd edition, 2022.

[10] Kaden A and Luckner B. Modeling Wake Vortex Roll–Up and Vortex–Induced Forces and Moments for Tight Formation Flight. In *AIAA Modeling and Simulation Technologies (MST) Conference*, Boston, MA, USA, 2013. American Institute of Aeronautics and Astronautics.

[11] Dogan A, Venkataramanan S, and Blake W. Modeling of aerodynamic coupling between aircraft in close proximity. *Journal of Aircraft*, 42(4):941–955, 2005.

[12] Chan J, Wang P, and Hesse H. Aerodynamic interactions in formation flight for wake vortex surfing. In *AIAA AVIATION 2023 Forum*. American Institute of Aeronautics and Astronautics, 2023.

[13] Ning A, Flanzer T, and Kroo I. Aerodynamic performance of extended formation flight. *Journal of Aircraft*, 48(3):855–865, 2011.

[14] Marks T, Dahlmann K, Grewe V, Gollnick V, Linke F, Matthes S, Stumpf E, Swaid M, Unterstrasser S, Yamashita H, and Zumegen C. Climate impact mitigation potential of formation flight. *Aerospace*, 8(1):14, 2021.

[15] Cisneros P, Voss S, and Werner H. Efficient nonlinear model predictive control via quasi–LPV representation. In *2016 IEEE 55th Conference on Decision and Control*, pages 3216–3221. Institute of Electrical and Electronics Engineers, 2016.

[16] Steinert A, Raab S, Hafner S, Holzapfel F, and Hong H. From fundamentals to applications of incremental nonlinear dynamic inversion: A survey on INDI Part I. *Chin. J. Aeronaut.*, 38(11), 2025.

[17] Steinert A, Raab S, Hafner S, Holzapfel F, and Hong H. Advancements in incremental nonlinear dynamic inversion and its components: A survey on INDI Part II. *Chin. J. Aeronaut.*, 38(11), 2025.

[18] Encarnação L, Pollack T, Looye G, and Theodoulis S. Multi–Objective synthesis of hybrid incremental dynamic inversion control laws using H_∞ Loop–Shaping. In *AIAA SCITECH 2026 Forum*. American Institute of Aeronautics and Astronautics, 2026.

[19] Pollack T, Theodoulis S, and Wang X. Quasi–LPV transformations for robust gain scheduling of incremental nonlinear dynamic inversion–based controllers. *IFAC–PapersOnLine*, 59(15):97–102, 2025.

[20] Steffensen R, Steinert A, and Holzapfel F. Incremental control as an enhanced and robust implementation of gain scheduled controllers avoiding hidden coupling terms. *Aerosp. Sci. Technol.*, pages 1167–1174, 2023.

[21] Hamstra J. *The F–35 Lightning II: From concept to cockpit*. American Institute of Aeronautics and Astronautics, 1st edition, 2018.

[22] Faleiro L and Lambregts A. Analysis and tuning of a Total Energy Control System control law using eigenstructure assignment. *Aerosp. Sci. Technol.*, 3(3):127–140, 4 1999.

[23] Argyle M and Beard R. Nonlinear total energy control for the longitudinal dynamics of an aircraft. *2016 American Control Conference*, pages 6741–6746, 7 2016.

[24] Raković S and Levine W. *Handbook of Model Predictive Control*. Springer, 1st edition, 2018.

[25] Rawlings J, Mayne D, and Diehl M. *Model Predictive Control*. Nob Hill Publishing, 2nd edition, 2024.

[26] Brunton S, Kutz J, and Proctor J. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proc. Natl. Acad. Sci. U.S.A.*, 113(15):3932–3937, 2016.

[27] Biegler L. *Nonlinear Programming*. Society for Industrial and Applied Mathematics, 3rd edition, 2010.

[28] Betts J. *Practical Methods for Optimal Control using Nonlinear Programming*. Society for Industrial and Applied Mathematics, 3rd edition, 2020.

[29] Wächter A. Short Tutorial: Getting Started With Ipopt in 90 Minutes. In *Combinatorial Scientific Computing*, volume 9061 of *Dagstuhl Seminar Proceedings*, pages 1–17, Dagstuhl, Wadern, Germany, 2009. Schloss Dagstuhl – Leibniz Center for Informatics.

[30] Biegler L and Zavala V. Large–scale nonlinear programming using IPOPT: An integrating framework for enterprise–wide dynamic optimization. *Comput. Chem. Eng.*, 11(33):575–582, 2008.

[31] Amestoy P, Duff I, Koster J, and L'Excellent JY. A Fully Asynchronous Multifrontal Solver Using Distributed Dynamic Scheduling. *SIAM J. Matrix Anal. Appl.*, 23(1):15–41, 2001.

[32] Andersson J, Gillis J, Horn G, Rawlings J, and Diehl M. CasADi – A software framework for nonlinear optimization and optimal control. *Math. Program. Comput.*, 11(1):1–36, 2019.

[33] Higham N. *Accuracy and stability of numerical algorithms*. Society for Industrial and Applied Mathematics, 1st edition, 2002.

[34] Nocedal J and Wright S. *Numerical Optimization*. Springer, 1st edition, 2006.

[35] Bryson A and Ho YC. *Applied Optimal Control*. Taylor & Francis, 1st edition, 1975.

[36] Hofmann B and Plato R. On ill–posedness concepts, stable solvability and saturation. *J. Inverse Ill–Posed Probl.*, 26(2):287–297, 2017.

[37] Bramburger J. *Data–Driven Methods for Dynamic Systems*. Society for Industrial and Applied Mathematics, 1st edition, 2024.

[38] Champion K, Zheng P, Aravkin A, Brunton S, and Kutz J. A unified sparse optimization framework to learn parsimonious Physics–Informed models from data. *IEEE Access*, 8:169259–169271, 2020.

[39] Kaptanoglu A, De Silva B, Fasel U, Kaheman K, Goldschmidt A, Callaham J, Delahunt C, Nicolaou Z, Champion K, Loiseau JC, Kutz J, and Brunton S. Pysindy: A comprehensive python package for robust sparse system identifications. *Proc. Natl. Acad. Sci. U.S.A.*, 7(69), 2022.

[40] Osterhuber R, Hanel M, and Hammon R. Realization of the Eurofighter 2000 Primary Lateral/Directional Flight Control Laws with Differential PI–Algorithm. *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2004.

[41] Messenger D and Bortz D. Weak SINDY: Galerkin–Based Data–Driven Model Selection. *Multiscale Model. Simul.*, 19(3):1474–1497, 2021.

[42] Klein V and Morelli E. *Aircraft System Identification*. American Institute of Aeronautics and Astronautics, 1st edition, 2006.

[43] DIN ISO 2533:1979–12, Normatmosphäre, 1979.

[44] Morelli E. Global nonlinear parametric modelling with application to F–16 aerodynamics. In *Proceed-*

ings of the 1998 American Control Conference, volume 21, pages 997–1001. Institute of Electrical and Electronics Engineers, 1998.

[45] Garza F and Morelli E. A Collection of Nonlinear Aircraft Simulations in MATLAB. Technical Report NASA/TM–2003–212145, National Aeronautics and Space Administration, 2021.

[46] Martins J and Ning A. *Engineering Design Optimization*. Cambridge University Press, 1st edition, 2021.

[47] Kelly M. An Introduction to Trajectory Optimization: How to do your own direct collocation. *SIAM Rev.*, 59(4):849–904, 2017.

[48] Ghezzi A, Nurkanović A, Weiss A, Diehl M, and Di Cairano S. A comparative study of MINLP and MPVC formulations for solving complex nonlinear DecisionMaking problems in aerospace applications. *Optim. Control Appl. Methods.*, 47(1):4–23, 11 2025.

[49] Newton I. *Philosophiae Naturalis Principia Mathematica*. Jussu Societatis Regiae ac Typis Josephi Streater. Prostat apud plures bibliopolas, 1 1687.

[50] Fasel U, Kaiser E, Kutz J, Brunton B, and Brunton S. SINDy with Control: A Tutorial. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 16–21, 2021.

[51] Brunton S, Budišić M, Kaiser E, and Kutz J. Modern koopman theory for dynamical systems. *SIAM Rev.*, 64(2):229–340, 2022.