Article

AWEbox: an Optimal Control Framework for Single- and Multi-Aircraft Airborne Wind Energy Systems

Jochem De Schutter[1]*, Rachel Leuthold[1], Thilo Bronnenmeyer[3], Elena Malz[4], Sebastien Gros[5] and Moritz Diehl[1,2]

[1] Systems Control and Optimization Laboratory, Department of Microsystems Engineering, University of Freiburg, Freiburg, Germany; rcleuthold@gmail.com (R.L.); moritz.diehl@imtek.uni-freiburg.de (M.D.)
[2] Department of Mathematics, University of Freiburg, Freiburg, Germany
[3] Kiteswarms GmbH, Freiburg, Germany; t.bronnenmeyer@googlemail.com
[4] Department of Electrical Engineering, Chalmers University of Technology, Göteborg, Sweden; elena@malz.me
[5] Department of Engineering Cybernetics, Norwegian University of Science and Technology, Trondheim, Norway; sebastien.gros@ntnu.no
* Correspondence: jochem.de.schutter@imtek.uni-freiburg.de

Abstract: In this paper we present `AWEbox`, a Python toolbox for modeling and optimal control of multi-aircraft systems for airborne wind energy (AWE). `AWEbox` provides an implementation of optimization-friendly multi-aircraft AWE dynamics for a wide range of system architectures and modeling options. It automatically formulates typical AWE optimal control problems based on these models, and finds a numerical solution in a reliable and efficient fashion. To obtain a high level of reliability and efficiency, the toolbox implements different homotopy methods for initial guess refinement. The first type of methods produces a feasible initial guess from an analytic initial guess based on user-provided parameters. The second type implements a warmstart procedure for parametric sweeps. We investigate the software performance in two different case studies. In the first case study we solve a single-aircraft reference problem for a large number of different initial guesses. The homotopy methods reduce the expected computation time by a factor of 1.7 and and the peak computation time by a factor of 8, compared to when no homotopy is applied. Overall, the CPU timings are competitive to timings reported in the literature. When the user initialization draws on expert a priori knowledge, homotopies do not increase expected performance, but the peak CPU time is still reduced by a factor of 5.5. In the second case study, a power curve for a dual-aircraft lift-mode AWE system is computed using the two different homotopy types for initial guess refinement. On average, the second homotopy type, which is tailored for parametric sweeps, outperforms the first type in terms of CPU time by a factor of 3. In conclusion, `AWEbox` provides an open-source implementation of efficient and reliable optimal control methods that both control experts and non-expert AWE developers can benefit from.

Keywords: airborne wind energy; optimal control; open-source software

1. Introduction

Airborne wind energy (AWE) is a renewable energy technology that aims at harvesting strong and steady high altitudes winds that cannot be reached by conventional wind technology, at a fraction of the material resources [1]. It is based on the principle of one or more tethered autonomous aircraft flying fast crosswind maneuvres. In the majority of AWE concepts, electricity is either produced by on-board turbines on the aircraft and conducted to a ground station through the tether (*drag-mode*), or in a periodic fashion by reeling-out the tether at high tension to drive a winch at the ground station, and reeling back in at low tension, so as to achieve a net positive energy output over one period (*lift-mode*). Although there exist many other interesting AWE concepts, e.g. those based on tethered rotorcrafts [2,3], we will limit the scope of this paper to rigid-wing lift-

 doi:10.20944/preprints202212.0018.v1

and drag-mode systems. The reader is referred to [4,5] for a recent and comprehensive overview of the different technologies.

The principle of AWE was first investigated in 1980 by Miles Loyd, who derived an upper limit for the power that could be produced by a crosswind AWE system [6]. Since then, and in particular in the past two decades, AWE has gained an increasing interest from both academia and industry, leading to significant technological progress and many small- to medium-scale prototypes, the largest of which was based on a 26 m wing span aircraft [7]. While AWE developers are considering a multitude of different designs, most systems are based on a single-aircraft setup. At this moment, AWE technology is still in a pre-commercial stage, with some companies taking first steps towards market-entry [8]. One of the central unresolved challenges for AWE developers is achieving techno-economic performance at utility-scale, i.e. designing systems that produce large amounts of electricity at low cost.

Multi-aircraft systems have been proposed and investigated in the literature as a more efficient and cheap way of producing utility-scale electricity [9–12]. In a multi-aircraft AWE system, two or more tethered aircraft fly tight crosswind maneuvres around a shared main tether, thereby minimizing the latter's crosswind motion and hence also the associated dissipation losses due to aerodynamic drag. These systems can be up to twice as efficient as their single-aircraft counterparts [10], while having superior, modular, upscaling properties [12], intrinsically smooth power output profiles [13] and higher potential power densities in farm configurations [14]. As a consequence of the increased system complexity, this system class has thus far only been investigated in simulation studies.

A crucial condition for the performance of both single- and multi-aircraft systems is finding power-efficient flight paths that satisfy flight envelope constraints and airframe load limits. This is not only necessary for path planning purposes but also for, e.g., offline performance prediction, design optimization and control strategy design. Optimal control is an evidently suitable path planning technique for AWE, given its natural ability to handle unstable, nonlinear, constrained systems with multiple in- and outputs. In the past decade, it has become an established method in the field, leading to various applications ranging from performance assessment studies [15], over model predictive control [16] and system identification [17], to flight path planning for a real-world soft-kite system [18]. We refer the reader to [19] for a complete overview of applications.

Despite its obvious advantages, optimal control comes with its own challenges: the dependence on an accurate model; the computational burden associated with finding a numerical solution; and a rather complex implementation that heavily relies on expert knowledge. In case an accurate model is unavailable, it is possible to resort to model-free, adaptive techniques such as extremum seeking (ES) [20] or iterative learning control (ILC) [21]. However, in [22], a validated reference model was proposed for a lift-mode, rigid-wing single-aircraft system. While identifying the parameters of such a model is a complicated and time-consuming task [17,23], this shows that deriving a physical model that fits the measurements very well is in principle possible. In the following, we will focus on the two other challenges mentioned above.

In order to increase computational efficiency, a general model structure based on non-minimal coordinates was proposed in [24], resulting in smooth dynamic equations of low symbolic complexity. Also, since the system nonlinearity gives rise to highly non-convex optimization problems, a feasible initial guess is typically needed for fast and reliable convergence of Newton-type optimization solvers. Such an initial guess is in many cases not available a priori. Therefore, a homotopy procedure was proposed that produces a close-to-optimal, feasible initial guess based on a generic, naive one [25]. Combined with a direct-collocation based transcription method, this led to reported computation times of below one minute for a representative power cycle of a lift-mode AWE system with a six-degree-of-freedom aircraft model [26]. Another homotopy variant was investigated in [27] to efficiently compute drag-mode power cycles for

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

large-scale wind data. Note that while in this paper, we only consider optimal control problems (OCP) in the time domain, they can also be formulated in the frequency domain as proposed in [28]. Such a formulation could in some cases lead to a more efficient discretization and more easily interpretable solutions.

While many open-source AWE control and simulation frameworks exist for single-aircraft models [29,30], and even for multi-aircraft models [31], there are only few available open-source implementations of optimization methods tailored for AWE. The MATLAB library `MegAWES` [20] provides an implementation of a megawatt-class system model and of a power optimization algorithm based on ES. The Optimal Control Library `OpenOCL` [32] provides a user-friendly MATLAB interface for formulating and solving OCPs, which can be linked to the optimization-friendly lift-mode model implemented in the framework `OpenAWE` [33]. However, this framework does not offer homotopy-based initial guess refinement or multi-aircraft models.

In this paper, we present `AWEbox`, an open-source `Python` framework for modeling and optimal control of single- and multi-aircraft AWE systems. The contributions of the software package can be summarized as follows:

- *Usability:* the user specifies only high-level modeling and optimization parameters. `AWEbox` implements optimization-friendly system dynamics for single- and multi-aircraft systems, for various system architectures and combinations of model options. It automatically formulates typical AWE optimization problems and implements and interfaces the algorithms needed to compute a numerical solution efficiently.
- *Reliability:* `AWEbox` increases reliability by efficiently computing a feasible OCP initial guess via homotopy methods, based on an analytic initial guess defined by a small number of user input parameters. The framework also implements an algorithm to efficiently and reliably perform parameter sweeps.
- *Extensibility:* within the baseline non-minimal-coordinates structure, users can add new or alternative modeling components (e.g. wind model, aerodynamics, etc.) in straightforward fashion. The homotopy procedure for initial guess refinement can be extended in a modular fashion, so that new model components can be introduced without affecting reliability.

`AWEbox` is freely available [34] and open-source under the GNU LGPLv3, which allows use in proprietary software. The toolbox heavily builds on lower level open-source software packages such as `CasADi` [35], a framework for algorithmic differentiation and optimization, and the nonlinear program (NLP) solver `IPOPT` [36].

The remainder of this paper is structured as follows. Section 2 discusses the multi-aircraft modeling procedure, while Section 3 gives an overview of the optimization ingredients used to formulate and numerically solve periodic AWE optimal control problems. Section 4 then outlines the software implementation details. Section 5 presents two case studies that highlight the efficiency and reliability of the implementation, as well as its multi-aircraft capability. Section 6 draws conclusions based on these results and makes suggestions for further research.

2. AWE modeling for optimal control

In this section, we define the multi-aircraft topologies considered in `AWEbox`. We outline the optimization-friendly AWE model structure for six-degrees-of-freedom (6DOF) aircraft dynamics as described in [22] and the extensions made for the multi-aircraft case [12,37].

2.1. Topology

We consider any tree-structured multi-aircraft topology as previously introduced in [12]. Each tree is described by a set of nodes $\mathcal{N}$, where each node $n \in \mathcal{N}$ represents the end-point of a tether. All tethers in the tree are assumed to be rigid and straight, which is a reasonable assumption if tether tension is high compared to gravity and tether drag

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

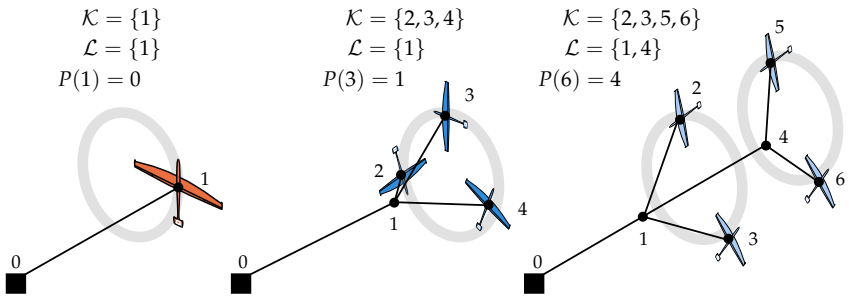

Figure 1. Illustration of the topology of a single-drone (left), triple-drone (middle) and two-layer-dual-drone (right) AWE system.

[22]. Some of the nodes $k \in \mathcal{K}$ correspond to *aircraft* nodes, while other nodes $l \in \mathcal{L}$ are *layer* nodes, with $\mathcal{L} := \mathcal{N} \backslash \mathcal{K}$ if $|\mathcal{N}| > 1$ and $\mathcal{L} := \mathcal{N}$ in the single-aircraft case. The parent map $P(n)$ uniquely defines the interlinkage between nodes, and the children map $C(n) := \{\, \bar{n} \in \mathcal{N} | P(\bar{n}) = n \}$ returns the set of nodes with parent n. Fig. 1 illustrates the proposed notation for some typical examples.

2.2. System dynamics and variables

The considered topologies require a multi-body modeling approach which should exhibit certain optimization-friendly properties. For one, the dynamics should have a low symbolic complexity to allow for fast repeated numerical evaluation, in particular of its sensitivities. Second, model nonlinearity should be kept low in order to enable fast and reliable use of Newton-type optimization techniques. Third, the model should avoid singularities that might be visited by and that might crash the optimization algorithm.

In [24], the efficacy of a non-minimal coordinates modeling approach to describe the translational and rotational dynamics of multiple interlinked aircraft is demonstrated. In this approach, each node is considered as a separate rigid body and linked by algebraic constraints. The aircraft orientation is parametrized in a non-singular fashion by the direction cosine matrix (DCM).

The resulting multi-body models are of reasonable complexity and nonlinearity but result in model equations in the form of index-3 DAEs. This representation does not allow for the deployment of classical integration methods within the optimal control problem [38]. Therefore an index-reduction technique is applied, which involves time-differentiation of the algebraic constraints. The resulting model equations for both lift- and drag-mode AWE systems are summarized by the following index-1 DAE:

$$\mathbf{F}(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t), \boldsymbol{\theta}, \mathbf{p}) = 0\,, \tag{1}$$

with associated consistency conditions $\mathbf{C}(\mathbf{x}(t)) = 0, \forall t \in \mathbb{R}$.

To define the differential state vector $\mathbf{x}$ for both lift- and drag-mode systems, we first define the basic multi-aircraft state

$$\mathbf{x}^{\text{base}} := (\mathbf{q}, \dot{\mathbf{q}}, \mathbf{R}, \omega, \delta)\,. \tag{2}$$

This state vector firstly contains $\mathbf{q}$ and $\dot{\mathbf{q}}$ that are concatenations of the node positions $\mathbf{q}_n \in \mathbb{R}^3$ and velocities $\dot{\mathbf{q}}_n \in \mathbb{R}^3$ respectively, $\forall n \in \mathcal{N}$. These are followed by the states specific to aircraft nodes, namely $\mathbf{R}, \omega, \delta$, which are concatenations of all $\boldsymbol{R}_k, \boldsymbol{\omega}_k, \delta_k, \forall k \in \mathcal{K}$. The DCMs $\mathbf{R}_k := [\hat{\mathbf{e}}_{1,k}, \hat{\mathbf{e}}_{2,k}, \hat{\mathbf{e}}_{3,k}] \in \mathbb{R}^{3\times3}$ contain the chord-wise, span-wise and upwards unit vectors of the aircraft body frames, expressed in the inertial frame $\{\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z\}$. All DCMs should be orthonormal, i.e. they are constrained to evolve on the 3D manifold defined by

$$\mathbf{c}_{\text{R},k} := P_{\text{ut}}(\mathbf{R}_k^\top \mathbf{R}_k - I) = 0\,, \tag{3}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

where the operator P_{ut} is used to select the six upper triangular elements of a matrix. The aircraft angular velocities $\boldsymbol{\omega}_k \in \mathbb{R}^3$ are given in the body frame. The surface deflections $\boldsymbol{\delta}_k = [\delta_{\text{a},k}, \delta_{\text{e},k}, \delta_{\text{r},k}] \in \mathbb{R}^3$ of aileron, elevator and rudder respectively, give control over the aircraft aerodynamics.

The lift- and drag-mode state vector can now be defined as

$$\mathbf{x}^{\text{lift}} := (\mathbf{x}^{\text{base}}, l_{\text{t}}, \dot{l}_{\text{t}}) \quad \text{and} \quad \mathbf{x}^{\text{drag}} := (\mathbf{x}^{\text{base}}, \boldsymbol{\kappa}) \,, \tag{4}$$

where the tether length $l_{\text{t}} \in \mathbb{R}$ and speed $\dot{l}_{\text{t}} \in \mathbb{R}$ describe the main tether reel-in and -out evolution. The variable $\boldsymbol{\kappa}$ is the concatenation of all $\kappa_k \in \mathbb{R}, \forall k \in \mathcal{K}$, which represent the on-board turbine drag coefficients.

The controls

$$\mathbf{u}^{\text{lift}} := (\dot{\boldsymbol{\delta}}, \ddot{l}_{\text{t}}) \quad \text{and} \quad \mathbf{u}^{\text{drag}} := (\dot{\boldsymbol{\delta}}, \dot{\boldsymbol{\kappa}}) \tag{5}$$

are given by the concatenation of all aircraft surface deflection rates $\dot{\boldsymbol{\delta}}_k \in \mathbb{R}^3$ and by either the tether acceleration $\ddot{l}_{\text{t}} \in \mathbb{R}$ or the concatenation of the turbine drag coefficient derivatives $\dot{\kappa}_k \in \mathbb{R}$.

The algebraic variables $\mathbf{z} := \boldsymbol{\lambda}$ describe the concatenation of all Lagrange multipliers $\lambda_n \in \mathbb{R}$ related to the tether constraints that restrict the position of each node $n \in \mathcal{N}$ to evolve on a 2D manifold defined by

$$c_n := \frac{1}{2}\left(\Delta\mathbf{q}(n)^\top \Delta\mathbf{q}(n) - l_n^2\right) = 0 \,, \tag{6}$$

where

$$\Delta\mathbf{q}(n) := \begin{cases} \mathbf{q}_n - \mathbf{q}_{P(n)}, & \text{if } n \notin \mathcal{K}, \\ \mathbf{q}_n + \mathbf{R}_n \mathbf{r}_{\text{t}} - \mathbf{q}_{P(n)}, & \text{if } n \in \mathcal{K}. \end{cases} \tag{7}$$

In these constraints, $\mathbf{r}_{\text{t}}$ is the tether attachment point described in the aircraft frame. The variable l_n describes the tether length associated with node n, and is defined together with the tether diameter d_n as

$$(l_n, d_n) := \begin{cases} (l_{\text{t}}, d_{\text{t}}), & \text{if } n = 1, \\ (l_{\text{s}}, d_{\text{s}}), & \text{if } n \in \mathcal{K}\backslash\{1\}, \\ (l_{\text{i}}, d_{\text{i}}), & \text{if } n \in \mathcal{L}\backslash\{1\}, \end{cases} \tag{8}$$

with l_{s} and d_{s} the length and diameter of the secondary tethers and l_{i} and d_{i} those of the layer-linking tethers in stacked multi-aircraft configurations. The ground station is located at the origin of the inertial frame, such that $\mathbf{q}_0 := \mathbf{0}$.

The variables $\boldsymbol{\theta}$ represent variable system parameters that can be optimized over. In the general stacked multi-aircraft case, they are defined as

$$\boldsymbol{\theta}^{\text{lift}} := (l_{\text{s}}, \boldsymbol{d}) \quad \text{and} \quad \boldsymbol{\theta}^{\text{drag}} := (l_{\text{t}}, l_{\text{s}}, \boldsymbol{d}) \tag{9}$$

where $\boldsymbol{d} := (d_{\text{t}}, d_{\text{s}}, d_{\text{i}})$.

The constant parameters $\mathbf{p}$ allow the dynamics to be evaluated for varying model parameters, such as aircraft wing span, wind model parameters, etc. The system parameters values used in the numerical experiments in this paper are listed in Appendix A, Table A1.

2.3. Lagrangian dynamics

The system dynamics (1) can be derived in accordance with the Lagrangian approach proposed in [24]. The system Lagrangian is defined as

$$L := T - V - \boldsymbol{\lambda}^\top \boldsymbol{c} \,, \tag{10}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

with $\boldsymbol{c}$ the concatenation of all tether constraints c_n for all $n \in \mathcal{N}$ and with the kinetic energy T and potential energy V defined as

$$T \coloneqq \sum_{k\in\mathcal{K}} T_{\mathrm{K},k} + \sum_{n\in\mathcal{N}} T_{\mathrm{t},n} \tag{11}$$

$$V \coloneqq \sum_{k\in\mathcal{K}} m_{\mathrm{K}} g \mathbf{q}_k^\top \mathbf{e}_z + \sum_{n\in\mathcal{N}} \frac{1}{2} m_{\mathrm{t},n} g (\mathbf{q}_n - \mathbf{q}_{P(n)})^\top \mathbf{e}_z \,. \tag{12}$$

Here, m_{K} is the aircraft mass, and the tether mass $m_{\mathrm{t},n} \coloneqq \rho_{\mathrm{t}} l_n \frac{\pi d_n^2}{4}$, with ρ_{t} the tether material density, and g is the gravitational acceleration. The kinetic energy related to the aircraft $T_{\mathrm{K},k}$ and to the tethers $T_{\mathrm{t},n}$ [39] are given by

$$T_{\mathrm{K},k} \coloneqq \frac{1}{2} m_{\mathrm{K}} \dot{\mathbf{q}}_k^\top \dot{\mathbf{q}}_k + \frac{1}{2} \boldsymbol{\omega}_k^\top J_{\mathrm{K}} \boldsymbol{\omega}_k \,, \tag{13}$$

$$T_{\mathrm{t},n} \coloneqq \frac{1}{6} m_{\mathrm{t},n} (\dot{\mathbf{q}}_n^\top \dot{\mathbf{q}}_n + \dot{\mathbf{q}}_{P(n)}^\top \dot{\mathbf{q}}_{P(n)} + \dot{\mathbf{q}}_n^\top \dot{\mathbf{q}}_{P(n)}) \,, \tag{14}$$

with J_{K} the aircraft moment of inertia and with the tether velocity at the ground station given by

$$\dot{\mathbf{q}}_0 \coloneqq \dot{\mathbf{q}}_1^\top \boldsymbol{e}_{\mathrm{t}} \quad \text{with} \quad \boldsymbol{e}_{\mathrm{t}} \coloneqq \frac{\mathbf{q}_1}{l_{\mathrm{t}}} \,. \tag{15}$$

Note that for a drag-mode system with constant tether length, this implies that $\dot{\mathbf{q}}_0 = 0$.

With the system Lagrangian defined, the translational dynamics read:

$$\frac{\mathrm{d}}{\mathrm{d}t} \frac{\partial L}{\partial \dot{\mathbf{q}}} - \frac{\partial L}{\partial \mathbf{q}} = \mathbf{F} + \mathbf{F}_{\mathrm{corr}} \tag{16}$$

with $\mathbf{F}$ the concatenation of the external forces $\mathbf{F}_n$ exerted on each of the nodes. The term $\mathbf{F}_{\mathrm{corr}}$ is a Lagrangian momentum correction term for open systems:

$$\mathbf{F}_{\mathrm{corr}} \coloneqq \sum_{n\in\mathcal{N}} \frac{\mathrm{d}m_n}{\mathrm{d}t} \dot{\mathbf{q}}_n^\top \frac{\partial \dot{\mathbf{q}}_n}{\partial \dot{\mathbf{q}}} \,. \tag{17}$$

This term is non-zero for lift-mode systems, since tether mass and energy are entering and leaving the system due to the reeling motion.

The rotational dynamics are projected on a 3D manifold in the aircraft body frame [24] so as to read:

$$J_{\mathrm{K}} \frac{\mathrm{d}\boldsymbol{\omega}_k}{\mathrm{d}t} + \boldsymbol{\omega}_k \times J_{\mathrm{K}} \boldsymbol{\omega}_k + 2U(\mathbf{R}_k^\top \nabla_{\mathbf{R}_k} \boldsymbol{\lambda}^\top \mathbf{c}) = \mathbf{M}_k \,, \quad \forall k \in \mathcal{K} \,, \tag{18}$$

with $\mathbf{M}_k$ the aerodynamic moment exerted on the aircraft and with U the "unskew" operator, i.e. $U(\mathtt{skew}(\mathbf{a})) = \mathbf{a}$, as defined in [24].

Next to the dynamic equations, also the holonomic constraints $\boldsymbol{c} = 0$ need to be enforced. Since these constraints do not explicitly depend on the generalized accelerations $\ddot{\mathbf{q}}$ or on the algebraic variables $\boldsymbol{\lambda}$, it is not possible to numerically integrate the resulting dynamic equations with standard algorithms. Therefore an index reduction is performed by differentiating $\boldsymbol{c}$ twice with respect to time. Note that $\ddot{\boldsymbol{c}}$ depends on $\ddot{\mathbf{q}}$.

Because of the index reduction, as well as the overparametrization of the rotational degrees-of-freedom, the consistency conditions $\mathbf{C}(\mathbf{x}) \coloneqq (\mathbf{c}, \dot{\mathbf{c}}, \mathbf{c}_{\mathrm{R}}) = 0$ must be enforced at an arbitrary time point in the trajectory. These quantities are called *invariants*, since their value is preserved by the dynamics. System invariants, when not dealt with carefully, can lead to failure of the Linear Independence Constraint Qualification (LICQ) in the context of periodic optimal control. Performing Baumgarte stabilization on the invariants is an effective way to avoid this issue, while simultaneously ensuring that

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

$\mathbf{C}(\mathbf{x}) = 0$ is satisfied over the entire time period [40]. Therefore the tether constraint dynamics are augmented with the following Baumgarte stabilization scheme:

$$\ddot{\boldsymbol{c}} + 2\kappa_t \dot{\boldsymbol{c}} + \kappa_t^2 \boldsymbol{c} = 0\,, \tag{19}$$

with κ_t a Baumgarte tuning parameter [41].

2.4. System Kinematics

The system dynamics also describe the following trivial kinematics. First, as explained in the previous section, the rotational kinematics are augmented with a Baumgarte-type stabilization on the orthogonality conditions [42]:

$$\frac{\mathrm{d}\mathbf{R}_k}{\mathrm{d}t} = \mathbf{R}_k\left(\frac{\kappa_R}{2}(I - \mathbf{R}_k^\top \mathbf{R}_k) + \mathtt{skew}(\boldsymbol{\omega}_k)\right)\,, \tag{20}$$

with κ_R another tuning parameter. The remaining kinematics

$$\frac{\mathrm{d}}{\mathrm{d}t}(\mathbf{q}, \delta) = (\dot{\mathbf{q}}, \dot{\delta})\,, \quad \frac{\mathrm{d}}{\mathrm{d}t}(l_t, \dot{l}_t) = (\dot{l}_t, \ddot{l}_t) \textit{ (lift-mode)}, \quad \frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\kappa} = \dot{\boldsymbol{\kappa}} \textit{ (drag-mode)}, \tag{21}$$

together with (16) - (20) then complete the system dynamics summarized by (1). The remaining modeling effort now focuses on the generalized forces $\mathbf{F}$ and moments $\mathbf{M}$.

2.5. Wind and atmosphere model

AWE systems typically operate at altitudes of several hundreds of meters, and the altitude variation within a typical power cycle is of the same order of magnitude [15]. In particular the multi-aircraft variant, unhindered by the drag losses caused by main tether cross-wind motion, can theoretically operate at arbitrarily high altitudes, wherever the wind power density is highest [10,12]. Therefore a wind model is needed that accounts for the varying wind power availability with altitude. Within the community, it is common to use one of the following approximations:

a) Logarithmic profile: A logarithmic model [43] is typically used as a very simple wind shear approximation. Assuming steady, laminar flow, the logarithmic model provides us with the following expression for the freestream wind velocity $\mathbf{u}_\infty(z)$:

$$\mathbf{u}_\infty(z) := u_{\mathrm{ref}} \frac{\log \frac{z}{z_0}}{\log \frac{z_{\mathrm{ref}}}{z_0}} \mathbf{e}_x\,, \tag{22}$$

which in this model is assumed to be aligned with the x-axis in the inertial frame. Here, u_{ref} is the reference wind speed that is measured at an altitude z_{ref}, whereas z_0 is the surface roughness length, which depends on local terrain characteristics.

b) Power-law profile: Another frequent approximation is given by the power law:

$$\mathbf{u}_\infty(z) := u_{\mathrm{ref}} \left(\frac{z}{z_{\mathrm{ref}}}\right)^{c_f} \mathbf{e}_x\,, \tag{23}$$

where c_f is a ground surface friction coefficient. We will use this model in the numerical experiments in this paper, in accordance with case studies in [15,22].

(c) 3D wind data interpolation: The disadvantage of the logarithmic and power-law models is that they are only useful to represent long-term average wind conditions. Realistic wind profiles come in a wide variety of shapes and they are subject to strong short-term (hourly, diurnal, seasonal) changes. Furthermore, the approximation accuracy typically breaks down at altitudes relevant to AWE systems [44]. Hence, for accurate optimal-control based power curve and capacity factor estimation, it is often necessary to generate a more detailed but still differentiable wind model based on highly spatially resolved wind speed measurements.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

To achieve this, we adopt the approach presented in [27,45]. We assume a wind profile that is represented by discrete 2D wind measurements $\mathbf{u}_{\mathrm{m},1}, \ldots, \mathbf{u}_{\mathrm{m},n_{\mathrm{lag}}} \in \mathbb{R}^2$. These measurements correspond to a set of altitudes $z_1, \ldots, z_{n_{\mathrm{lag}}}$. We can then create a smooth wind model to approximate the measured wind profile, by creating an interpolating function based on Lagrange polynomials:

$$\mathcal{W}(z, \boldsymbol{\zeta}) := \sum_{i=1}^{n_{\mathrm{lag}}} \left(\boldsymbol{\zeta}_i \cdot \prod_{\substack{k=1 \\ k \neq i}}^{n_{\mathrm{lag}}} \frac{z - z_k}{z_i - z_k} \right), \tag{24}$$

with $\boldsymbol{\zeta}$ the concatenation of the polynomial coefficients $\boldsymbol{\zeta}_i \in \mathbb{R}^2$ obtained by solving the following optimization problem

$$\boldsymbol{\zeta}^* := \underset{\boldsymbol{\zeta}}{\arg\min} \quad \frac{1}{2} \sum_{i=1}^{n_{\mathrm{lag}}} \|\mathcal{W}(z_i, \boldsymbol{\zeta}) - \mathbf{u}_{\mathrm{m},i}\|^2 + k \left\| \frac{\mathrm{d}^2 \mathcal{W}}{\mathrm{d}z^2}(z_i, \boldsymbol{\zeta}) \right\|^2 . \tag{25}$$

The cost function is tuned with weight k so that $\boldsymbol{\zeta}_i^* \approx \mathbf{u}_{\mathrm{m},i}$, $\forall i = 1 \ldots n_{\mathrm{lag}}$, while preventing overfitting via the penalization of the second derivative of the interpolating polynomials. The smooth and differentiable wind model is then given by $\mathbf{u}_\infty(z) := (\mathcal{W}(z, \boldsymbol{\zeta}^*), 0)$.

Wind power availability is linear in the air density, and the atmospheric density drop is non-negligible in the altitudes relevant to AWE. Therefore the density variation with altitude $\rho(z)$ is modeled according to the international standard atmosphere model [43]:

$$\rho(z) := \rho_0 \left(\frac{T_0 - T_{\mathrm{L}} z}{T_0} \right)^{\frac{g}{T_{\mathrm{L}} R} - 1}, \tag{26}$$

where R is the universal gas constant. The parameters T_0 and ρ_0 are the temperature and air density at sea level, and T_{L} is the temperature lapse rate.

2.6. Aerodynamic model

The apparent wind at each aircraft node $k \in \mathcal{K}$ is defined as

$$\mathbf{u}_{\mathrm{a},k} := \mathbf{u}_\infty(\mathbf{q}_k^\top \mathbf{e}_z) - \dot{\mathbf{q}}_k . \tag{27}$$

We then define the dynamic pressure as $q_k := \frac{1}{2}\rho(\mathbf{q}_k^\top \mathbf{e}_z)\|\mathbf{u}_{\mathrm{a},k}\|^2$. The aerodynamic forces (in the inertial frame) and moments (in the body frame) on the aircraft wings are then given by

$$\mathbf{F}_{\mathrm{A},k} := q_k S \mathbf{R}_k \mathbf{C}_{\mathrm{F},k} \quad \text{and} \quad \mathbf{M}_{\mathrm{A},k} := q_k S \begin{bmatrix} b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & b \end{bmatrix} \mathbf{C}_{\mathrm{M},k} . \tag{28}$$

with S the aircraft aerodynamic surface and with the aerodynamic coefficients $\mathbf{C}_{\mathrm{F},k} := (C_{\mathrm{X},k}, C_{\mathrm{Y},k}, C_{\mathrm{Z},k})$ and $\mathbf{C}_{\mathrm{M},k} := (C_{\mathrm{l},k}, C_{\mathrm{m},k}, C_{\mathrm{n},k})$, which are a function of the angles of attack α_k and side-slip angles β_k, given by the small-angle approximations

$$\alpha_k := \frac{\hat{\mathbf{e}}_{3,k}^\top \mathbf{u}_{\mathrm{a},k}}{\hat{\mathbf{e}}_{1,k}^\top \mathbf{u}_{\mathrm{a},k}} \quad \text{and} \quad \beta := \frac{\hat{\mathbf{e}}_{2,k}^\top \mathbf{u}_{\mathrm{a},k}}{\hat{\mathbf{e}}_{1,k}^\top \mathbf{u}_{\mathrm{a},k}} . \tag{29}$$

The force and moment coefficients $\mathbf{C}_{\diamond,k}$ (with $\diamond \in \{\mathrm{F},\mathrm{M}\}$) read as

$$\mathbf{C}_{\diamond,k} := \mathbf{C}_{\diamond,0}(\alpha_k) + \mathbf{C}_{\diamond,\beta}(\alpha_k)\beta_k + \mathbf{C}_{\diamond,\boldsymbol{\omega}}(\alpha_k) \begin{bmatrix} b & 0 & 0 \\ 0 & c & 0 \\ 0 & 0 & b \end{bmatrix} \frac{\boldsymbol{\omega}_k}{2\|\mathbf{u}_{\mathrm{a},k}\|} + \mathbf{C}_{\diamond,\delta}(\alpha_k)\delta_k \tag{30}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

The dependence of these coefficients on α_k is approximated by second-order polynomials of the form:

$$C_{\diamond,\circ}(\alpha) := \begin{bmatrix} c_{\diamond,\circ,2} & c_{\diamond,\circ,1} & c_{\diamond,\circ,0} \end{bmatrix} \begin{bmatrix} \alpha^2 \\ \alpha \\ 1 \end{bmatrix} \tag{31}$$

with the values of the coefficients $c_{\diamond,\circ,i}$ used in this study given in Table 2 in [22].

The tether drag is modeled as follows. Consider the infinitesimal tether drag force $\mathrm{d}\mathbf{F}_{\mathrm{td},n}(s) := \mathbf{F}_{\mathrm{td},n}(s)\mathrm{d}s$ on an infinitesimal segment $l_n\mathrm{d}s$, for $s \in [0,1]$, with:

$$\mathbf{F}_{\mathrm{td},n}(s) := \frac{1}{2}\rho(\mathbf{q}_{\mathrm{t},n}(s)^\top \mathbf{e}_z) C_{\mathrm{D,t}} d_n l_n \|\mathbf{u}_{\mathrm{t},n}(s)\| \mathbf{u}_{\mathrm{t},n}(s) \,, \tag{32}$$

with $C_{\mathrm{D,t}}$ the tether drag coefficient, and where the segment position and apparent wind speed are given by

$$\mathbf{q}_{\mathrm{t},n}(s) := s\mathbf{q}_{P(n)} + (1-s)\mathbf{q}_n \tag{33}$$

$$\mathbf{u}_{\mathrm{t},n}(s) := \mathbf{u}_\infty(\mathbf{q}_{\mathrm{t},n}(s)^\top \mathbf{e}_z) - \dot{\mathbf{q}}_{\mathrm{t},n}(s) \,. \tag{34}$$

It is shown in [39,46] that the total drag force can be exactly distributed into contributions on node n and on its parent node $P(n)$, so as to read

$$\mathbf{F}^1_{\mathrm{td},n} := \int_0^1 s\mathbf{F}_{\mathrm{td},n}(s)\mathrm{d}s \quad \text{and} \quad \mathbf{F}^0_{\mathrm{td},n} := \int_0^1 (1-s)\mathbf{F}_{\mathrm{td},n}(s)\mathrm{d}s \tag{35}$$

respectively. In order to be able to numerically evaluate the tether drag, the integrals in (35) are discretized using the midpoint rule. Typically, a number of $M_{\mathrm{td}} = 5$ integration intervals is sufficiently accurate.

The generalized forces can now be defined for each node as

$$\mathbf{F}^\star_n := \begin{cases} \mathbf{F}_{\mathrm{A},n} + \mathbf{F}^1_{\mathrm{td},n} & \text{if } n \in \mathcal{K} \wedge (\star = \text{lift}) \\ \mathbf{F}_{\mathrm{A},n} + \mathbf{F}^1_{\mathrm{td},n} + \mathbf{F}_{\mathrm{turb},k} & \text{if } n \in \mathcal{K} \wedge (\star = \text{drag}) \\ \mathbf{F}^1_{\mathrm{td},n} + \sum\limits_{c \in C(n)} \mathbf{F}^0_{\mathrm{t},c} & \text{if } n \in \mathcal{N} \backslash \mathcal{K} \end{cases} \tag{36}$$

and the generalized moments are given by the aerodynamic moments, i.e. $\mathbf{M}_k := \mathbf{M}_{\mathrm{A},k}$, $\forall k \in \mathcal{K}$. In the drag-mode case, also the braking force of the on-board turbines is acting on the aircraft:

$$\mathbf{F}_{\mathrm{turb},k} := \kappa_k \|\mathbf{u}_{\mathrm{a},k}\| \mathbf{u}_{\mathrm{a},k} \,. \tag{37}$$

Note that the tether pulling force and moment exerted on the aircraft are implicitly modeled in the constraint-based dynamics (16) and (18) and should not be considered as part of the generalized forces.

2.7. Power output

For lift-mode systems, the generated power is the product of the main tether force with the tether speed. The pulling force by tether n experienced at node n is given by the expression $\mathbf{F}_{\mathrm{t},n} := -\lambda_n \nabla_{\mathbf{q}_n} c_n$. Note that a positive multiplier corresponds a positive pulling force. The power transferred through tether n is then given by $P_{\mathrm{t},n} := \mathbf{F}^\top_{\mathrm{t},n}\dot{\mathbf{q}}_n$. For the main tether, this expression can be simplified to $P_{\mathrm{t},1} := -\lambda_1 l_{\mathrm{t}} \dot{l}_{\mathrm{t}}$. The mechanical power that arrives at the ground station is given by $P^{\mathrm{lift}} := -P_{\mathrm{t},1}$.

In drag-mode systems, electrical power is generated by the on-board turbines and transferred to the ground station through the tethers. Each aircraft $k \in \mathcal{K}$ generates an amount of electrical power $P_{\mathrm{turb},k} := \eta_{\mathrm{turb}} \kappa_k \|\mathbf{u}_{\mathrm{a},k}\|^3$, with η_{turb} the on-board turbine efficiency. Note that for the case of power consumption, i.e. $\kappa < 0$, the efficiency needs to be inverted. This can be implemented using the logistic function, as proposed in

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

[28]. The total power output generated by the drag-mode system is then given by $P^{\text{drag}} := \sum_{k \in \mathcal{K}} P_{\text{turb},k}$.

3. Optimization ingredients

In this section, we discuss all the necessary ingredients to formulate, discretize and reliably solve power optimization problems for the system model described in the previous section. We state the periodic optimal control problem formulation in continuous-time, and we discuss common system constraints. We explain the transcription method to convert the problem into an NLP and we summarize the interior-point solution strategy used by `IPOPT` to solve it. Then we describe how the initial guess is constructed, and how it can be refined using two different homotopy methods that are tailored for interior-point NLP solvers. Finally we discuss a third homotopy method that is tailored for performing parameter sweeps with interior-point NLP solvers.

3.1. Problem formulation for periodic orbits

The main goal of the toolbox is to facilitate automated computation of dynamically feasible, power-optimal periodic orbits for both lift- and drag-mode systems, while satisfying a set of relevant system contraints. In order to achieve this, we formulate a periodic optimal control problem of a free time period T, which has the distinctive property that the system state at the initial and final time of the OCP time horizon can be chosen freely by the solver, but must be equal. Given that some key system parameters $\boldsymbol{\theta}$, such as the tether diameters and lengths, have a huge impact on the system power output and the optimal flight trajectories, they are included as optimization variables as well.

Let the optimization variables be defined as $\mathbf{w} := (\mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t), \boldsymbol{\theta}, T)$. Then we can compute power-optimal state and control trajectories and a corresponding system design $\boldsymbol{\theta}$ for given parameters $\mathbf{p}$ by solving the following continuous-time optimization problem:

$$\min_{\mathbf{w}} \quad \frac{1}{T} \int_0^T l(\mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t)) \mathrm{d}t \tag{38a}$$

$$\text{s.t.} \quad \mathbf{F}(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t), \boldsymbol{\theta}, \mathbf{p}) = 0, \ \forall t \in [0, T], \tag{38b}$$

$$\mathbf{h}(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t), \boldsymbol{\theta}, \mathbf{p}) \leq 0, \ \forall t \in [0, T], \tag{38c}$$

$$\mathbf{x}(0) - \mathbf{x}(T) = 0, \tag{38d}$$

$$\psi(\mathbf{x}(0)) = 0. \tag{38e}$$

The Lagrange cost term is given by the sum of the negative power output and a penalty on the controls in order to mitigate actuator fatigue, as well as on the side slip angle and the angular accelerations in order to avoid aerodynamic side forces and aggressive maneuvers:

$$l(\mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t)) := -P(t) + \hat{w}(t)^\top \mathbf{W} \hat{w}(t) \,, \tag{39}$$

with $\hat{w}(t) := (\mathbf{u}(t), \boldsymbol{\beta}(t), \dot{\boldsymbol{\omega}}(t))$ and $\mathbf{W}$ a constant diagonal weighting matrix. The variables $\boldsymbol{\beta}$ and $\dot{\boldsymbol{\omega}}$ are the vertical concatenations of the side slip angles β_k and angular accelerations $\dot{\omega}_k$, $\forall k \in \mathcal{K}$. Proper tuning of the weighting matrix $\mathbf{W}$ is necessary to achieve fast convergence of the optimization algorithm as well as to obtain a locally unique solution. We refer the reader to the open-source code for the weighting factors used in the numerical experiments in this study.

The function $\psi : \mathbb{R}^{n_\mathrm{x}} \to \mathbb{R}$ is used to impose a technical constraint that removes the phase invariance inherent to periodic OCPs. For lift- and drag-mode systems, this function is different and reads as either

$$\psi^{\text{lift}}(\mathbf{x}^{\text{lift}}(0)) := \dot{l}_\mathrm{t}(0) \stackrel{!}{=} 0 \quad \text{or} \quad \psi^{\text{drag}}(\mathbf{x}^{\text{drag}}(0)) := \dot{\mathbf{q}}_1(0)^\top \mathbf{e}_\mathrm{y} \stackrel{!}{=} 0 \,. \tag{40}$$

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

The inequality constraints $\mathbf{h}$ are discussed in the following section.

Note that the consistency conditions $\mathbf{C}(\mathbf{x}(t)) = 0$ are not enforced at any given time within the time horizon of the OCP. In combination with the periodicity constraint (38d), this would lead to LICQ deficiency for all feasible trajectories. There exist several technical solutions for this issue [40]. In the dynamic correction approach chosen here, Baumgarte stabilization is applied to the consistency conditions in the system dynamics, as previously mentioned in section 2.3. Therefore the dynamics of $\mathbf{C}$ are exponentially stable and since by value of periodicity it holds that $\mathbf{C}(\mathbf{x}(0)) = \mathbf{C}(\mathbf{x}(T))$, the only feasible periodic state trajectories are those where $\mathbf{C}(\mathbf{x}(t)) = 0, \forall t \in [0, T]$.

3.2. System constraints

A particular feature of OCP (38) is that it has an economic cost function, which is not lower bounded, as opposed to tracking cost functions [47]. OCPs with an economic cost function tend to having extreme solutions in the absence of constraints. In the context of AWE power optimization, it is therefore crucial to impose constraints that avoid a violation of the flight envelope and that preserve the structural integrity of the airframe and the tether.

The flight envelope consists of upper and lower bounds on the angle-of-attack $\boldsymbol{\alpha}$ (to avoid stall) and the side-slip angle $\boldsymbol{\beta}$ (to avoid additional drag and preserve model validity) for all aircraft in the system. Additionally the stress in the tethers should not exceed the yield strength with a certain safety factor f_s:

$$\sigma_n \coloneqq f_\mathrm{s} \frac{4\|\mathbf{F}_{\mathrm{t},n}\|}{\pi d_n^2} \leq \sigma_\mathrm{max} \,, \quad \forall n \in \mathcal{N} \,. \tag{41}$$

Here, the tether force magnitude can be simplified to $\|\mathbf{F}_{\mathrm{t},n}\| = \lambda_n l_n$, following the definition in Section 2.7. The aircraft orientation is also constrained in order to avoid collision of the airframe with the tether, which might occur during sharp turns in transition maneuvers:

$$(\mathbf{q}_k - \mathbf{q}_{P(k)})^\top \hat{\mathbf{e}}_{3,k} \geq \cos(\gamma_\mathrm{max}) l_k \,, \quad \forall k \in \mathcal{K} \,, \tag{42}$$

where γ_max is the maximum angle between the tether vector and the upwards unit vector of the aircraft body frame, which should be set lower than at most $\pi/2$. In the multi-aircraft case, following anti-collision constraints might be included:

$$\|\mathbf{q}_k - \mathbf{q}_m\|_2 \geq f_\mathrm{b} b \,, \quad \forall k, m \in \mathcal{K}, k \neq m \,, \tag{43}$$

where f_b is safety factor in multiples of the wing span b.

Along these nonlinear constraints, variable bounds are typically imposed on variables such as flight altitude, tether length, speed and acceleration, aircraft angular velocity, control surface deflections and their rates, etc. One pair of variable bounds that is crucial in the context of periodic optimal control, are the bounds on the time period T. Since the OCP will be discretized in a discrete number of numerical integration intervals, the integration accuracy is variable along with T. Therefore T should be bounded from above to guarantee an acceptable simulation accuracy. Also, by translating a priori knowledge on the optimal value of T into variable bounds, we narrow the search space and exclude many possible local solutions, which typically increases reliability and speeds up convergence of the NLP solver.

3.3. Problem transcription

The continuous-time OCP (38) has an infinite number of variables and constraints. Hence, we apply direct optimal control to transcribe the OCP to an NLP. We choose transcription by direct collocation, which is a fully simultaneous approach, where the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

numerical simulation variables are treated as variables in the optimization problem [26]. We chose this approach for the following reasons.

First, fully simultaneous optimal control is characterized by faster contraction rates of the Newton-type iterations compared to simultaneous and sequential optimal control, in particular for highly nonlinear and unstable systems [48]. Second, in the fully simultaneous case, the simulation problem is solved directly by the NLP solver, which is typically more robust than the rootfinder used in standard available numerical integrators. Finally, since OCP (38) is highly non-convex, the NLP solver benefits from computing the Newton step using exact Hessian information. The NLP Hessian becomes considerably cheaper to evaluate in the fully simultaneous approach.

Although the resulting direct collocation NLP is comparably large, it is also sparse. In combination with a sparsity-exploiting NLP solver, direct collocation is a highly efficient transcription method for the models presented in this paper.

In direct collocation, the time horizon is divided into N (usually equidistant) intervals described by $[t_i, t_{i+1}]$, where $0 < t_0 < t_1 < \cdots < t_N = T$. The control trajectory is parameterized as a piecewise constant function $\tilde{\mathbf{u}}(t) := \mathbf{u}_i$ if $t \in [t_i, t_{i+1})$. The state trajectory is parametrized by piecewise polynomials of order $M+1$, i.e. $\tilde{\mathbf{x}}(t) := \tilde{\mathbf{x}}_i(t)$ if $t \in [t_i, t_{i+1})$, with

$$\tilde{\mathbf{x}}_i(t) := \sum_{j=0}^{M} \xi_j(\tau)\mathbf{x}'_{i,j}\,, \tag{44}$$

with the normalized time $\tau := \frac{t-t_i}{\Delta t_i}$, $\tau \in [0,1]$, with $\Delta t_i := t_{i+1} - t_i$ and with the variables $\mathbf{x}'_{i,j}$ placed at the time points $(\tau_0, \boldsymbol{\tau})$, with $\boldsymbol{\tau} := (\tau_1, \ldots, \tau_M)$ and with $\tau_0 := 0$. The Lagrange polynomials ξ_j are uniquely defined by the choice of collocation grid points $\boldsymbol{\tau}$:

$$\xi_j(\tau) := \prod_{\substack{k=0 \\ k\neq j}}^{M} \frac{\tau_k - \tau}{\tau_k - \tau_j}\,. \tag{45}$$

Note that it holds that $\tilde{\mathbf{x}}_i(t_i + \Delta t_i \tau_j) = \mathbf{x}'_{i,j}$. The state derivative is given by the derivative of the polynomials, i.e.

$$\dot{\tilde{\mathbf{x}}}_i(t) := \sum_{j=0}^{M} \frac{1}{\Delta t_i} \frac{\mathrm{d}\xi_j}{\mathrm{d}\tau}(\tau)\mathbf{x}'_{i,j}\,. \tag{46}$$

The algebraic variables are also discretized in each i'th time interval as $\mathbf{z}'_{i,j}$, and allocated to the collocation points $\tau_1, \ldots, \tau_M$.

Let us now define $\mathbf{x}_i := \mathbf{x}'_{i,0}$, $\mathbf{X}_i := [\mathbf{x}'_{i,1}, \ldots, \mathbf{x}'_{i,M}]$ and $\mathbf{Z}_i := [\mathbf{z}'_{i,1}, \ldots, \mathbf{z}'_{i,M}]$. Then, for given state vector $\mathbf{x}_i$ at the start of each interval, the collocation variables $\mathbf{X}_i$ and $\mathbf{Z}_i$ are uniquely determined by enforcing the system dynamics (1) at the grid points $\tau_1, \ldots, \tau_M$:

$$\mathbf{G}_i(\mathbf{x}_i, \mathbf{u}_i, \mathbf{X}_i, \mathbf{Z}_i, \boldsymbol{\theta}, \mathbf{p}, T) := \begin{bmatrix} \mathbf{F}\left(\dot{\tilde{\mathbf{x}}}_i(t_i + \Delta t_i \tau_1), \mathbf{x}'_{i,1}, \mathbf{u}_i, \mathbf{z}'_{i,1}, \boldsymbol{\theta}, \mathbf{p}\right) \\ \mathbf{F}\left(\dot{\tilde{\mathbf{x}}}_i(t_i + \Delta t_i \tau_2), \mathbf{x}'_{i,2}, \mathbf{u}_i, \mathbf{z}'_{i,2}, \boldsymbol{\theta}, \mathbf{p}\right) \\ \vdots \\ \mathbf{F}\left(\dot{\tilde{\mathbf{x}}}_i(t_i + \Delta t_i \tau_M), \mathbf{x}'_{i,M}, \mathbf{u}_i, \mathbf{z}'_{i,M}, \boldsymbol{\theta}, \mathbf{p}\right) \end{bmatrix} = 0\,. \tag{47}$$

The state transition from one interval node to the next is given by the equation

$$\mathbf{x}_{i+1} = \boldsymbol{\phi}(\mathbf{x}_i, \mathbf{X}_i) \quad \text{with} \quad \boldsymbol{\phi}(\mathbf{x}_i, \mathbf{X}_i) := \tilde{\mathbf{x}}_i(t_{i+1}) = \sum_{j=0}^{M} \xi_j(1)\mathbf{x}'_{i,j}\,. \tag{48}$$

The system of equations (47) corresponds to that of an implicit Runge-Kutta integration scheme, where the choice of collocation grid points $\boldsymbol{\tau}$ uniquely defines the Butcher-Tableau of the specific integration method. Here, we choose as collocation grid points

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

the roots of Gauss-Radau polynomials, more specifically those corresponding to the Radau IIa integration scheme because of its high order accuracy and its excellent stability properties (A- and L-stability), which is particularly relevant for stiff DAE systems [49].

Further, the inequality constraints are imposed on the interval nodes and the Lagrange term in the cost function can be computed via a quadrature rule [26]:

$$\int_{t_i}^{t_{i+1}} l(\mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t)\mathrm{d}t \approx \Delta t_i \sum_{j=1}^{M} b'_j \cdot l(\mathbf{x}'_{i,j}, \mathbf{u}_i, \mathbf{z}'_{i,j}) \,, \tag{49}$$

where the quadrature weights are given by

$$\begin{bmatrix} b'_1 & \cdots & b'_M \end{bmatrix} \coloneqq \begin{bmatrix} \xi_1(1) & \cdots & \xi_M(1) \end{bmatrix} \begin{bmatrix} \frac{\mathrm{d}\xi_1}{\mathrm{d}\tau}(\tau_1) & \cdots & \frac{\mathrm{d}\xi_M}{\mathrm{d}\tau}(\tau_1) \\ \vdots & \ddots & \vdots \\ \frac{\mathrm{d}\xi_1}{\mathrm{d}\tau}(\tau_M) & \cdots & \frac{\mathrm{d}\xi_M}{\mathrm{d}\tau}(\tau_M) \end{bmatrix}^{-1} . \tag{50}$$

The NLP resulting from discretizing the OCP (38) using direct collocation is then formulated as

$$\min_{\mathbf{w}} \quad \frac{1}{T} \sum_{i=0}^{N-1} \Delta t_i \sum_{j=1}^{M} b'_j \cdot l(\mathbf{x}'_{i,j}, \mathbf{u}_i, \mathbf{z}'_{i,j}) \tag{51a}$$

$$\text{s.t.} \quad \mathbf{x}_{i+1} - \boldsymbol{\phi}(\mathbf{x}_i, \mathbf{X}_i) = 0, \ \forall i = 0, \dots, N-1, \tag{51b}$$

$$\mathbf{G}_i(\mathbf{x}_i, \mathbf{u}_i, \mathbf{X}_i, \mathbf{Z}_i, \boldsymbol{\theta}, \mathbf{p}, T) = 0, \ \forall i = 0, \dots, N-1, \tag{51c}$$

$$\mathbf{h}(\dot{\mathbf{x}}_i(t_i + \Delta i \tau_M), \mathbf{X}_{i,M}, \mathbf{u}_i, \mathbf{Z}_{i,M}, \boldsymbol{\theta}, \mathbf{p}) \leq 0, \ \forall i = 0, \dots, N-1, \tag{51d}$$

$$\mathbf{x}_0 - \mathbf{x}_N = 0, \tag{51e}$$

$$\psi(\mathbf{x}_0) = 0. \tag{51f}$$

with the decision variables summarized by $\mathbf{w} \coloneqq (\mathbf{x}_0, \mathbf{X}_0, \mathbf{Z}_0, \mathbf{u}_0, \mathbf{x}_1, \dots, \mathbf{u}_{N-1}, \mathbf{x}_N, \boldsymbol{\theta}, T)$. For the remainder of this text, we will write NLP (51) in more compact form as the parametric NLP

$$\mathcal{P}_\mathrm{f}(\mathbf{p}) \coloneqq \min_{\mathbf{w}} \quad \Phi(\mathbf{w}, \mathbf{p}) \tag{52a}$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{w}, \mathbf{p}) = 0, \tag{52b}$$

$$\mathbf{H}(\mathbf{w}, \mathbf{p}) \leq 0. \tag{52c}$$

3.4. Solution strategy

There are two common solution approaches for inequality-constrained nonlinear programs such as (52): sequential quadratic programming (SQP) methods and interior-point (IP) methods [50]. SQP methods are based on iteratively solving a series of convex quadratic programs (QP) that are local approximations of the NLP. IP methods on the other hand perform iterations directly on a relaxed version of the Karush-Kuhn-Tucker (KKT) system corresponding to NLP (52), which read as

$$\begin{aligned} \nabla_{\mathbf{w}} \Phi(\mathbf{w}, \mathbf{p}) + \boldsymbol{\lambda}^\top \nabla_{\mathbf{w}} \mathbf{G}(\mathbf{w}, \mathbf{p}) + \boldsymbol{\mu}^\top \nabla_{\mathbf{w}} \mathbf{H}(\mathbf{w}, \mathbf{p}) &= 0 \\ \mathbf{G}(\mathbf{w}, \mathbf{p}) &= 0 \\ \mathbf{H}(\mathbf{w}, \mathbf{p}) + \mathbf{s} &= 0 \\ \mathrm{diag}(\mathbf{s})\boldsymbol{\mu} &= \tau \mathbf{1} \,, \end{aligned} \tag{53}$$

where $\mathbf{1}$ denotes a vector of ones. Together with the conditions $\boldsymbol{\mu} \geq 0$, $\mathbf{s} \geq 0$, the KKT system (53) for barrier parameter $\tau = 0$ gives the first-order necessary conditions of optimality. However, in this case, the KKT system is non-smooth due to the complementarity condition $\mathrm{diag}(\mathbf{s})\boldsymbol{\mu} = 0$, and therefore difficult to solve with Newton-type methods.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

Therefore, in IP methods, the iterations generally start on a smooth KKT system related to a barrier parameter $\tau_0 > 0$, which is then gradually reduced to a smaller value $\tau_f > 0$, so that the final solution approximates the exact solution of (53) up to sufficient accuracy. It holds that $\|\mathbf{w}_\tau^* - \mathbf{w}^*\|_2 = \mathcal{O}(\tau)$, where $\mathbf{w}_\tau^*$ and $\mathbf{w}^*$ are the solutions to the KKT system for $\tau > 0$ and for $\tau = 0$, respectively.

The advantage of IP methods is that the iterations are computationally cheaper compared those of SQP methods: per iteration only one linear system has to be solved, as opposed to one QP of equal size. Also, because IP methods start iterating on a problem with relaxed inequality constraints, and only gradually tighten these constraints, they are particularly robust in case little or no a priori knowledge on the active set of the optimal solution is available, as is typically the case for AWE systems.

In this work, we use the interior-point NLP solver `IPOPT` [36] in combination with the linear solver `MA57` [51]. `IPOPT` implements a particularly reliable algorithm that implements a filter line search method for globalization [52]. The algorithm also exploits the sparsity of the direct collocation NLP which makes it particularly efficient for this application.

3.5. Circular initial guess construction

In order to efficiently converge to a solution of a highly nonlinear, non-convex NLP, even a robust NLP solver such as `IPOPT` typically requires a good initial guess. Therefore we propose here a circular flight trajectory initialization based on a limited number of user-defined parameters $\boldsymbol{\pi}^0$:

$$\boldsymbol{\pi}^0 := (\dot{q}^0, N_l^0, l_t^0, \theta_e^0, \theta_c^0, \varphi^0, \boldsymbol{\theta}^0)\,, \tag{54}$$

where $\dot{q}^0$ is the aircraft flight speed, N_l^0 the number of loops, l_t^0 the initial tether length, θ_e^0 the (average) elevation angle of the main tether and θ_c^0 the trajectory cone angle with respect to the average main tether vector. The angle φ^0 denotes the phase angle with which the periodic initial guess can be shifted in time. The parameter $\boldsymbol{\theta}^0$ is a direct guess for the system parameters $\boldsymbol{\theta}$.

Building on the parameters $\boldsymbol{\pi}^0$, we then define a stationary tether frame as

$$\mathbf{e}_1' := \cos(\theta_e^0)\cdot\mathbf{e}_x + \sin(\theta_e^0)\cdot\mathbf{e}_z\,,\quad \mathbf{e}_2' := \frac{\mathbf{e}_1'\times\mathbf{e}_x}{\|\mathbf{e}_1'\times\mathbf{e}_x\|}\,,\quad \mathbf{e}_3' := \mathbf{e}_1'\times\mathbf{e}_2'\,, \tag{55}$$

after which we can define for each aircraft k a frame that is rotating about the main tether:

$$\begin{bmatrix}\mathbf{e}_{1,k}''(t) & \mathbf{e}_{2,k}''(t) & \mathbf{e}_{3,k}''(t)\end{bmatrix} := \mathbf{R}_x(\phi_k(t))\begin{bmatrix}\mathbf{e}_1' & \mathbf{e}_2' & \mathbf{e}_3'\end{bmatrix} \tag{56}$$

with the rotation angle $\varphi_k(t)$ for each aircraft defined as

$$\varphi_k(t) := \varphi^0 + \omega^0 t + 2\pi(k - P(k) - 1)/|C(P(k))\backslash\mathcal{L}|\,, \tag{57}$$

and with the rotation radius and speed, and the time period of one loop defined as

$$R^0 := l_k^0\sin(\theta_c^0)\,,\quad \omega^0 := \frac{\dot{q}^0}{R^0}\quad\text{and}\quad T_l^0 := \frac{2\pi}{\omega^0}\,, \tag{58}$$

respectively.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

In the general multi-aircraft case, the node positions and (angular) velocities are then initialized at each time point on the collocation grid t_i by

$$\mathbf{q}^0_{l,i} \leftarrow \mathbf{q}^0_{P(l),i} + l^0_l \cdot \mathbf{e}''_{1,k}(t_i)\,, \qquad \forall l \in \mathcal{L}\,, \tag{59}$$

$$\mathbf{q}^0_{k,i} \leftarrow \mathbf{q}^0_{P(k),i} + \sqrt{l^{0^2}_k - R^{0^2}} \cdot \mathbf{e}''_{1,k}(t_i) + R^0 \cdot \mathbf{e}''_{2,k}(t_i)\,, \qquad \forall k \in \mathcal{K}\,, \tag{60}$$

$$\dot{\mathbf{q}}^0_{k,i} \leftarrow \dot{q}^0 \cdot \mathbf{e}''_{3,k}(t_i)\,, \qquad \forall k \in \mathcal{K}\,, \tag{61}$$

$$\boldsymbol{\omega}^0_{k,i} \leftarrow \omega^0 \cdot \mathbf{e}''_{1,k}(t_i)\,, \qquad \forall k \in \mathcal{K}\,. \tag{62}$$

In the single-aircraft case ($\mathcal{K} = \mathcal{L} = \{1\}$), the aircraft position is initialized using (60). The layer node velocities are set to zero.

The aircraft DCMs are initialized so that the initial guess meets the flight envelope constraints rather than that it exactly satisfies the kinematic relation (20). The apparent wind speed for each drone at time t_i is given by

$$\mathbf{u}^0_{\mathrm{a},k,i} \coloneqq \mathbf{u}_\infty(\mathbf{q}^{0\top}_{k,i}\,\mathbf{e}_\mathrm{z}) - \dot{\mathbf{q}}^0_{k,i}\,, \; \forall k \in \mathcal{K}\,, \tag{63}$$

with $\mathbf{u}_\infty(\cdot)$ the user-defined wind profile. The DCM is then initialized to have zero angle of attack and zero side-slip angle:

$$\hat{\mathbf{e}}^0_{1,k,i} \leftarrow \frac{\mathbf{u}^0_{\mathrm{a},k,i}}{\|\mathbf{u}^0_{\mathrm{a},k,i}\|}\,, \qquad \forall k \in \mathcal{K}\,, \tag{64}$$

$$\hat{\mathbf{e}}^0_{2,k,i} \leftarrow \frac{\mathbf{e}''_{1,k}(t_i) \times \hat{\mathbf{e}}_{1,k,i}}{\|\mathbf{e}''_{1,k}(t_i) \times \hat{\mathbf{e}}_{1,k,i}\|}\,, \qquad \forall k \in \mathcal{K}\,, \tag{65}$$

$$\hat{\mathbf{e}}^0_{3,k,i} \leftarrow \hat{\mathbf{e}}_{1,k,i} \times \hat{\mathbf{e}}_{2,k,i}\,, \qquad \forall k \in \mathcal{K}\,. \tag{66}$$

The tether multipliers are trivially initialized as $\lambda^0_n \leftarrow 1\ \mathrm{Nm}^{-1}, \forall n \in \mathcal{N}$, to ensure a strictly positive tether force. All remaining states and controls are initialized as zero. Finally, the initial overall cycle period is set to $T^0 \leftarrow T^0_1 N^0_1$.

The initial guess is summarized by the vector $\bar{\mathbf{w}}^0$. In the following, we will refer to the method which uses $\bar{\mathbf{w}}^0$ as an initial guess for $\mathcal{P}_\mathrm{f}$ as "NH" (no homotopy).

3.6. Homotopy-based initial guess refinement

Even the educated initial guess defined in the previous section often leads to very slow convergence or even solver failure when solving $\mathcal{P}_\mathrm{f}(p)$. In order to increase computation speed and improve reliability, we propose a refinement procedure based on homotopy methods, which reliably produces a close-to-optimal, feasible initial guess based on the analytic user-defined initialization.

The basic idea is to first solve a trivial version of the intended NLP, and then to repeatedly compute the solution while updating the NLP in a controlled and smooth way to the full nonlinear final problem. Homotopy methods (also known as continuation methods) are widely used in the field of non-convex optimization when little or no a priori knowledge on the location of the optimal solution is available [53,54]. Homotopy methods were originally introduced in the field of AWE optimization in [25]. In this paper, we generalize this approach for multiple homotopy stages and discuss particularities when using interior-point methods.

First we construct a homotopy problem $\mathcal{H}_\mathrm{c}(\mathbf{p}, \boldsymbol{\phi})$, with homotopy parameters $\boldsymbol{\phi} \in \mathbb{R}^{n_\phi}$ and $\phi_i \in [0,1], \forall i \in \{1,\dots,n_\phi\}$. Note that $\boldsymbol{\phi}$ can be multidimensional to allow

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

for step-wise introduction of distinct model nonlinearities or couplings. The homotopy problem is defined as

$$\mathcal{H}_c(\mathbf{p}, \boldsymbol{\phi}) := \min_{\mathbf{w}} \quad \Phi_H(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi}) \tag{67a}$$

$$\text{s.t.} \quad \mathbf{G}_H(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi}) = 0, \tag{67b}$$

$$\mathbf{H}_H(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi}) \leq 0 \tag{67c}$$

with the NLP functions $\Phi_H, \mathbf{G}_H$ and $\mathbf{H}_H$ defined such that $\mathcal{H}_c(\mathbf{p}, 1) = \mathcal{P}_0(\mathbf{p})$ and $\mathcal{H}_c(\mathbf{p}, 0) = \mathcal{P}_f(\mathbf{p})$. Here, $\mathcal{P}_0(\mathbf{p})$ is a simplified problem which is trivial to optimize for a large set of initial guesses, and $\mathcal{P}_f(\mathbf{p})$ is the target optimization problem defined in (52). It can be shown that, if $\mathcal{H}_c(\mathbf{p}, \boldsymbol{\phi})$ satisfies the LICQ and second-order sufficient conditions (SOSC) for all $\mathbf{p}$ and $\boldsymbol{\phi}$, there exists a unique and piecewise smooth homotopy path $\mathbf{w}^*(\mathbf{p}, \boldsymbol{\phi})$ between the optimal solutions $\mathbf{w}^*(\mathbf{p}, 0)$ and $\mathbf{w}^*(\mathbf{p}, 1)$ [54].

Algorithm 1 (CIPH) describes a classic procedure to follow the homotopy path $\mathbf{w}^*(\mathbf{p}, \boldsymbol{\phi})$. First we provide an initial guess $\bar{\mathbf{w}}^0$ which is the approximate solution of the initial problem $\mathcal{P}_0(\mathbf{p})$. Then, for each step i in the multi-step homotopy, we reduce the homotopy parameter ϕ_i from one to zero with an increment $\frac{1}{\gamma}$ in a total of γ iterations. At every iteration the homotopy problem $\mathcal{H}_c(\mathbf{p}, \boldsymbol{\phi})$ is solved up to a certain (low) accuracy level, while the NLP solver is warmstarted with the solution of the previous iteration. To improve performance, that maximum number of NLP iterations can be limited in this stage.

The output of the homotopy then is an approximate solution $\bar{\mathbf{w}}_f$ to the intermediate problem $\mathcal{H}_c(\mathbf{p}, 0)$, which can be used as an initial guess for solving $\mathcal{P}_f(\mathbf{p})$ up to high accuracy. If the LICQ and SOSC conditions are fulfilled, there exists a high enough value of γ to guarantee convergence of this algorithm [54, Theorem 5.2].

Algorithm 1 Classic Interior-Point-based Homotopy (CIPH)

Require: $\bar{\mathbf{w}}_0, \mathbf{p}, \gamma > 0$
Output: $\bar{\mathbf{w}}_f$
  $\boldsymbol{\phi} \leftarrow \mathbf{1}_{n_\phi \times 1}$
  $\mathbf{w}^{(0)} \leftarrow \text{NLPSOLVER}(\mathcal{P}_0(\mathbf{p}), \bar{\mathbf{w}}_0)$
  for $i = 1, \ldots, n_\phi$ **do**
    $\mathbf{w}^{(i)} \leftarrow \mathbf{w}^{(i-1)}$
    for $j = 1, \ldots, \gamma$ **do**
      $\phi_i \leftarrow \phi_i - \frac{1}{\gamma}$
      $\mathbf{w}^{(i)} \leftarrow \text{NLPSOLVER}(\mathcal{H}_c(\mathbf{p}, \boldsymbol{\phi}), \mathbf{w}^{(i)})$
    end for
  end for
  $\bar{\mathbf{w}}_f \leftarrow \mathbf{w}^{(n_\phi)}$

3.7. Penalty-based homotopy

The fixed-step continuation approach described in the previous section is simple to implement and works well in practice [27,45,55]. Nevertheless it has two drawbacks. First, the choice of fixed homotopy parameter step renders the algorithm less robust than if an adaptive-step strategy would be used. Second, in terms of computational efficiency, $\gamma \cdot n_\phi$ NLPs need to be solved by default even when larger steps would be feasible.

Of course, adaptive step size strategies for homotopy path following exist and are well-established [53,54]. However, they increase the complexity of the algorithm as well as the amount of hyperparameters to tune. Therefore we propose a simple but effective variation of Algorithm (1), which can be used in particular when the chosen NLP solver is a well-globalised solver. The idea is to use the underlying globalization routines (e.g. line-search) of the NLP solver to choose a suitable homotopy parameter step size.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

The resulting homotopy strategy is *penalty-based* and builds on the reformulation $\mathcal{H}_\text{p}$ of NLP (67) to read:

$$\begin{aligned}
\mathcal{H}_\text{p}(\mathbf{p}, \hat{\boldsymbol{\phi}}) \coloneqq \min_{\mathbf{w}, \boldsymbol{\phi}} \quad & \Phi_\text{H}(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi}) + \mathbf{S}^\top \boldsymbol{\phi} && \text{(68a)}\\
\text{s.t.} \quad & \mathbf{G}_\text{H}(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi}) = 0, && \text{(68b)}\\
& \mathbf{H}_\text{H}(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi}) \leq 0, && \text{(68c)}\\
& \overline{\boldsymbol{\phi}} \geq \boldsymbol{\phi} \geq \underline{\boldsymbol{\phi}}. && \text{(68d)}
\end{aligned}$$

In this formulation, the parameters $\boldsymbol{\phi}$ are treated as a decision variables with a high linear penalty $S \in \mathbb{R}^{n_\phi}_+$. The homotopy path is now parametrized by the bounds on $\boldsymbol{\phi}$, i.e. $\hat{\boldsymbol{\phi}} \coloneqq (\overline{\boldsymbol{\phi}}, \underline{\boldsymbol{\phi}}) \in [0, 1]$.

Algorithm 2 (PIPH) describes the alternative homotopy procedure. The lower bounds $\underline{\boldsymbol{\phi}}$ are successively set to 0 for each homotopy stage, allowing the NLP solver to find a path for the homotopy parameter ϕ_i in stage i, while simultaneously applying correction steps to the decision variables $\mathbf{w}$. Afterwards, the problem is solved again with $\overline{\phi}_i = 0$ to ensure completion of the homotopy stage.

Because of the high linear penalty on $\boldsymbol{\phi}$, the NLP solver will take the largest possible parameter step that is acceptable to the line-search filter, hence providing both robustness and speed. Additionally, only $2 \cdot n_\phi$ NLPs need to be solved instead of the $\gamma \cdot n_\phi$ NLPs in the classic continuation homotopy. This can allow for a significant speed-up even if the number of iterations per NLP solve is naturally higher.

Algorithm 2 Penalty-based Interior-Point-based Homotopy (PIPH)

Require: $\bar{\mathbf{w}}_0, \mathbf{p}$
Output: $\bar{\mathbf{w}}_\text{f}$
  $\overline{\boldsymbol{\phi}}, \underline{\boldsymbol{\phi}}, \boldsymbol{\phi}^{(0)} \leftarrow \mathbf{1}_{n_\phi \times 1}$
  $\mathbf{w}^{(0)} \leftarrow \text{NLPSOLVER}(\mathcal{P}_0(\mathbf{p}), \bar{\mathbf{w}}_0)$
  for $i = 1, \ldots, n_\phi$ **do**
    $\mathbf{w}^{(i)} \leftarrow \mathbf{w}^{(i-1)}$
    $\underline{\phi}_i \leftarrow 0$
    $\mathbf{w}^{(i)} \leftarrow \text{NLPSOLVER}(\mathcal{H}_\text{p}(\mathbf{p}, \hat{\boldsymbol{\phi}}), \mathbf{w}^{(i)})$
    $\overline{\phi}_i \leftarrow 0$
    $\mathbf{w}^{(i)} \leftarrow \text{NLPSOLVER}(\mathcal{H}_\text{p}(\mathbf{p}, \hat{\boldsymbol{\phi}}), \mathbf{w}^{(i)})$
  end for
  $\bar{\mathbf{w}}_\text{f} \leftarrow \mathbf{w}^{(n_\phi)}$

Note that the convergence of Algorithm 2 is only guaranteed for small enough updates of the parameter $\hat{\boldsymbol{\phi}}$. In practice however, convergence is almost always achieved for jumps from 1 to 0.

3.8. Interior-point-based homotopy

The homotopy methods presented above are based on the idea of solving a sequence of closely related problems, where the solution of each problem is used to warmstart the next. However, because an interior-point NLP solver by default starts iterating on the relaxed KKT problem (53) (with a high barrier parameter τ), it is unable to exploit the (active set) information contained in the initial guess, if it is the solution to the non-smooth KKT problem. To circumvent this issue, we apply the following barrier strategy [27,56,57] throughout the homotopy:

1. The initial problem $\mathcal{P}_0$ is solved from an initial barrier parameter τ_0 to an intermediate $\tau_\text{i} < \tau_0$, so that the KKT system remains sufficiently smooth.
2. The homotopy problem $\mathcal{H}_\text{p}$ is repeatedly solved for constant barrier parameter τ_i.
3. The final problem $\mathcal{P}_\text{f}$ is solved from τ_i to a final value $\tau_\text{f} < \tau_\text{i}$.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

Using this strategy, the Newton iterations quickly converge from one intermediate problem to the next during the homotopy stage.

3.9. Homotopy design

In this paper, we propose two homotopy stages ($n_\phi = 2$). The initial problem $\mathcal{P}_0(\mathbf{p})$ thus comprises two alterations with respect to the final problem $\mathcal{P}_\mathrm{f}(\mathbf{p})$. Firstly, the aerodynamic forces and moments in the model are replaced with the direct force controls $\mathbf{F}_{\mathrm{f},k} \in \mathbb{R}^3$ and moment controls $\mathbf{M}_{\mathrm{f},k} \in \mathbb{R}^3$ for all $k \in \mathcal{K}$, which are then added to the control vector $\mathbf{u}$. This step relaxes the nonlinearities and couplings related to the aerodynamics [25]. Secondly the initial problem does not optimize the average power output but rather the tracking error with respect to the user-generated initial guess.

The homotopy problems $\mathcal{H}_\mathrm{c}(\mathbf{p}, \boldsymbol{\phi})$ and $\mathcal{H}_\mathrm{p}(\mathbf{p}, \hat{\boldsymbol{\phi}})$ are then constructed by replacing $\mathbf{F}_{\mathrm{A},k}$ and $\mathbf{M}_{\mathrm{A},k}$ in (36) with

$$\begin{pmatrix} \hat{\mathbf{F}}_{\mathrm{A},k} \\ \hat{\mathbf{M}}_{\mathrm{A},k} \end{pmatrix} := \phi_1 \begin{pmatrix} \mathbf{F}_{\mathrm{A},k} \\ \mathbf{M}_{\mathrm{A},k} \end{pmatrix} + (1-\phi_1) \begin{pmatrix} \mathbf{F}_{\mathrm{f},k} \\ \mathbf{M}_{\mathrm{f},k} \end{pmatrix} \tag{69}$$

as well as by changing the stage cost function to

$$l(\mathbf{x}(t), \mathbf{u}(t), \mathbf{z}(t), \phi) := -\phi_2 P(t) + (1-\phi_2)\|\mathbf{x}(t) - \bar{\mathbf{x}}^0(t)\|_Q^2 + \hat{w}(t)^\top \mathbf{W} \hat{w}(t)\,, \tag{70}$$

with $\bar{\mathbf{x}}^0(t)$ the initial state trajectory guess.

Additionally, in order to reduce the initial degrees of freedom, the system parameters are fixed to their initial values until the second homotopy step. The system parameters are thus only optimized over when the cost function transitions from tracking error to power output:

$$(1-\underline{\phi}_2)\underline{\boldsymbol{\theta}} + \underline{\phi}_2 \boldsymbol{\theta}^0 \leq \boldsymbol{\theta} \leq (1-\underline{\phi}_2)\bar{\boldsymbol{\theta}} + \underline{\phi}_2 \boldsymbol{\theta}^0\,. \tag{71}$$

Substituting equations (69) - (71) into the model, cost function and constraints, we obtain after repeated discretization with direct collocation the functions $\Phi_\mathrm{H}(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi})$, $\mathbf{G}_\mathrm{H}(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi})$ and $\mathbf{H}_\mathrm{H}(\mathbf{w}, \mathbf{p}, \boldsymbol{\phi})$.

3.10. Parametric sweep warmstarting

Once a solution for NLP (52) has been found, it is often interesting to investigate the sensitivity of the optimal solution with respect to one or more of the model parameters $\mathbf{p}$. A typical example is when we compute the NLP solution for different values of u_ref (in the case of a logarithmic or power-law wind profile) to compute a power curve for a particular AWE system. One approach is to apply Algorithms 1 or 2 to compute a solution for all parameter values based on the same initial guess. However, in case the distance between the different parameter values is small, it is more efficient and more reliable to compute an initial guess for one problem from the solution of the previous one.

Algorithm 3 (SIPH) describes how an initial guess for each problem in the set of NLPs $\mathcal{P}_\mathrm{f}(\mathbf{p}_i)$, for $i = 1, \ldots, p$, can be generated efficiently. It starts based on the solution $\bar{\mathbf{w}}_{\mathrm{f},0}$ of the homotopy problem $\mathcal{H}_\mathrm{c}(\mathbf{p}_0, 0)$ for an initial set of parameters $\mathbf{p}_0$. This initial solution can be computed using CIPH or PIPH. We assume that the sequence of parameter vectors $\mathbf{p}_0, \ldots, \mathbf{p}_p$ is ordered so as to minimize the distance from one parameter set to the next, as proposed in [27]. Then, we can compute the initial guess $\bar{\mathbf{w}}_{\mathrm{f},i}$ for problem $\mathcal{P}_\mathrm{f}(\mathbf{p}_i)$ from the guess $\bar{\mathbf{w}}_{\mathrm{f},i-1}$ for the previous problem $\mathcal{P}_\mathrm{f}(\mathbf{p}_{i-1})$, by updating the parameter vector $\mathbf{p}$ from one value to the next via linear interpolation in γ_p steps and by recursively solving the problem $\mathcal{H}_\mathrm{c}(\mathbf{p}, 0)$. We employ the same barrier strategy as in section 3.8 and keep the barrier parameter at a constant value τ_i while solving $\mathcal{H}_\mathrm{c}$, to guarantee a smooth transition from one problem to the next.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

Algorithm 3 Parametric Sweep Interior-Point-based Homotopy (SIPH)

Require: $\bar{\mathbf{w}}_{\mathrm{f},0}, \mathbf{p}_0, \ldots, \mathbf{p}_p$
Output: $\bar{\mathbf{w}}_{\mathrm{f},1}, \ldots, \bar{\mathbf{w}}_{\mathrm{f},p}$
  for $i = 1, \ldots, p$ **do**
    $\bar{\mathbf{w}}_{\mathrm{f},i} \leftarrow \bar{\mathbf{w}}_{\mathrm{f},i-1}$
    for $j = 1, \ldots, \gamma_{\mathrm{p}}$ **do**
      $\mathbf{p} \leftarrow \mathbf{p}_{i-1} + \frac{j}{\gamma_{\mathrm{p}}}(\mathbf{p}_i - \mathbf{p}_{i-1})$
      $\bar{\mathbf{w}}_{\mathrm{f},i} \leftarrow \textsc{NlpSolver}(\mathcal{H}_{\mathrm{c}}(\mathbf{p}, 0), \bar{\mathbf{w}}_{\mathrm{f},i})$
    end for
  end for
  $\bar{\mathbf{w}}_{\mathrm{f}} \leftarrow \mathbf{w}^{(n_\phi)}$

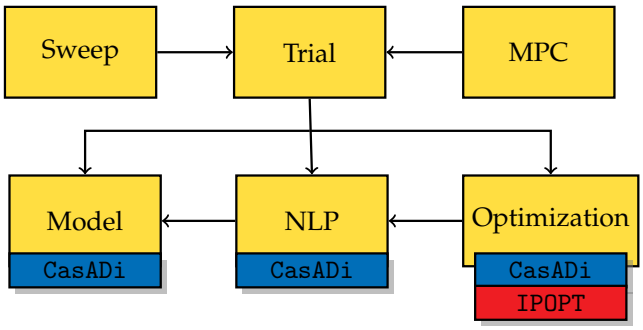

Figure 2. Main `AWEbox` classes (Python) and overall software structure, including dependencies.

4. The `AWEbox` software package

The goal of the `AWEbox` software package is to provide a user-friendly interface that facilitates the automatic construction of the optimization-friendly dynamics (1). It formulates the the power optimization problem (38) and reliably finds a numerical solution. The toolbox is written in Python 3 and relies heavily on the following software packages: `CasADi`, an open-source symbolic framework for algorithmic differentiation and nonlinear optimization [35]; the interior-point NLP solver `IPOPT` [36]; and (optionally) the linear solver `MA57` [51]. The six main classes and basic structure of the package are shown in Fig. 2, including the dependencies on the external packages.

Starting at the lowest level, the `Model`-class takes the user-provided modeling options and assembles the according state, control and algebraic variable vectors. Then, the dynamics (1), relevant constraints and intermediate model outputs are constructed as `CasADi Function` objects. Table 1 gives an overview of the main modeling options implemented in `AWEbox`. Central here is the use of `CasADi` to compute the partial derivatives of the system Lagrangian in (16). Finally, the `Model` class can also be used in standalone mode, e.g. in case the user is interested in obtaining the dynamics for simulation purposes only.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

Table 1. Main AWE system modeling options and possible variants implemented in `awebox`.

Options	Variants
Topology	single-aircraft, i.e.: $(1,1)$
	multi-aircraft, e.g.: $(1,2)$
	stacked multi-aircraft, e.g.: $(2,3)$
Power generation	lift-mode
	drag-mode
Aircraft DOF	6 DOF
	3 DOF [10]
Wind profile	uniform
	logarithmic
	power-law
	3D-data
Atmosphere	uniform
	International Standard Atmosphere
Induction	constant/zero
	actuator-disk

The `NLP` class receives from a `Model` instance the dynamics and constraints and constructs the NLP functions Φ_{H}, $\mathbf{G}_{\mathrm{H}}$ and $\mathbf{H}_{\mathrm{H}}$ as `CasADi Function` objects, using the direct collocation approach presented in Section 3.3.

From a practical viewpoint, it is essential for the convergence of the NLP solver that all variables, equations and cost terms are properly scaled. Therefore, `AWEbox` implements a heuristics-based scaling procedure based on the system parameters and the user-defined initialization. We refer the reader to the open-source implementation in (cite Zenodo) for the scaling factors obtained in the numerical experiments in this study.

The NLP functions are then passed on to the `Optimization` class, where their first- and second-order derivatives are constructed using `CasADi`, which also provides the interface to `IPOPT`. The `Optimization` class then constructs the initial guess from Section 3.5 and implements both Algorithm 1 and 2 to prepare the homotopy-based initial guess for solving Problem (52).. It is also possible to warmstart of the solver with a user-provided initial guess. Finally, Problem (52) is solved up to high accuracy. The default linear solver for computing the Newton step within `IPOPT` is `MUMPS`, but in general a higher performance in terms of speed and reliability is reached using the solver `MA57`, which has to be installed separately.

On a higher level, the central class with which the user interacts is the `Trial` class, which knits together the functionality of the lower-level classes. To start with, the user can specify modeling options, physical parameters, discretization options, initialization parameters, etc., as in the following (non-exhaustive) example:

```
opts = {}
opts['model.topology'] = {1:0}  # parent map P(n)
opts['model.kite_dof'] = 6
opts['model.system_type'] = 'lift_mode'
opts['model.wind.model'] = 'uniform'
opts['model.wind.u_ref'] = 10. # [m/s]
opts['nlp.N'] = 100
opts['solver.linear_solver'] = 'ma57'
opts['solver.initialization.l_t'] = 400. # [m]
opts['solver.homotopy.phi.0'] = 'penalty'
opts['solver.homotopy.phi.1'] = 'penalty'
```

With these options, the user can create a `Trial` object, and build the system dynamics, constraints and NLP functions, including derivatives. In this example the power optimization is then solved using the penalty-based homotopy. The `Trial` class then performs some quality checks on the numerical accuracy of the solution, e.g. by checking

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

consistency condition satisfaction. The class also contains some basic plotting functionality for visualizing the optimal solution:

```
from awebox import Trial
trial = Trial(opts)
trial.build()
trial.solve()
trial.plot(['states', 'controls'])
```

The high-level class `Sweep`, which builds on the `Trial` class, can be useful for parametric sweeps. This class builds the parametric NLP functions and their derivatives only once, and implements Algorithm 3 for warmstarting the neighboring NLP problems:

```
from awebox import Sweep
sweep_opts = [('model.wind.u_ref', [4,6,8,10,12,14,16])]
sweep = Sweep(opts, sweep_opts)
sweep.build()
sweep.run()
```

The `MPC` class uses the `Trial` class and the lower level classes to construct the tracking MPC problem as defined in [58]. The class takes as an input the optimal solution of Problem (38) to construct a periodic reference on the MPC time grid. It also takes care of correct initialization, and initial guess and periodic reference shifting. The MPC problem can then be recursively solved using `IPOPT` with the warmstarting strategy from [57]. The main goal of this class is not to provide highly efficient numerical solvers aimed at embedded optimization, such as those implemented in the software packages `acados` [59] or `PolyMPC` [60]. Rather, this class provides a reliable controller that conveniently allows for offline closed-loop simulations.

```
from awebox import Pmpc
mpc_opts = {}
mpc_opts['N'] = 20
mpc_opts['terminal_point_constr'] = True
Ts = 0.1
mpc = Pmpc(mpc_opts, Ts, trial)
u0 = mpc.step(x0)
```

Although the focus here is reliability and not computational efficiency, the user can also code-generate and compile the MPC solver functions using `CasADi`, for use in an external codebase or for embedded application.

5. Numerical Results

This section discusses two numerical case studies that highlight the contributions of the `AWEbox` software package. In the first case study we discuss and compare computational performance and robustness of the homotopy algorithms CIPH and PIPH, while solving a single-aircraft lift-mode reference problem. In the second case study we compute a power curve for a dual-aircraft lift-mode system and compare the performance of the algorithms PIPH and SIPH.

5.1. Single-aircraft case study

The first reference problem aims at finding an optimal power cycle for a lift-mode single-aircraft system, with $\mathcal{K} = \{1\}$, $\mathcal{L} = \{1\}$, and $P(1) = 0$. The aircraft parameters are taken from the Ampyx AP2 reference model presented in [22]. We adopt the same wind profile and atmosphere model as presented in [15]. We assume a "reinforced" version of the AP2 airframe, since the real-world airframe load limits lead to an overly pessimistic average power output estimate. Therefore, compared to the OCP in [15], the airspeed limits and tether force limits are omitted and replaced only by a tether stress constraint, while the tether diameter d_t is no longer fixed and is treated as an optimization variable. Table A1 in Appendix A summarizes the model parameter values of this reference problem, while Table 2 lists all variable bounds and path constraints.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

Table 2. System variable bounds and path constraints.

Description	Variable	Min	Max	Units
Side-slip angle	β	-20.0	20.0	deg
Angle-of-attack	α	-6.0	9.0	deg
Tether stress	σ	0.0	3.6	GPa
Rotation angle	γ_r	-40.0	40.0	deg
Tether length	l_t	10.0	700.0	m
Tether speed	$\dot{l}_\mathrm{t}$	-15.0	20.0	m s^{-1}
Tether acceleration	$\ddot{l}_\mathrm{t}$	-2.4	2.4	m s^{-2}
Flight altitude	q_z	100.0	-	m
Time period	T	20.0	70.0	s
Angular velocity	ω	-50.0	50.0	deg s^{-1}
Aileron deflection	δ_a	-20.0	20.0	deg
Rudder deflection	δ_r	-30.0	30.0	deg
Elevator deflection	δ_e	-30.0	30.0	deg
Deflection rates	$\dot{\delta}$	-2.0	2.0	rad s^{-1}

We construct the NLPs (67) and (68) using $N = 100$ intervals, with Radau collocation polynomials of order $M = 4$, and the controls are discretized using a piecewise constant parameterization. The resulting NLPs have 15334 variables, 14323 equality constraints and 600 inequality constraints. We solve the problem on an Intel Core i7 2.5 Ghz, 16GB RAM.

The homotopy meta-parameters are experimentally tuned to minimize the associated CPU time. The intermediate homotopy barrier parameter is chosen as $\tau_\mathrm{i} = 10^{-2}$. For CIPH, the number of parameter update steps per stage are $\gamma_1 = 10$ and $\gamma_2 = 1$. For PIPH, the homotopy parameter penalties are $S_1 = 10^2$ and $S_2 = 1$.

In the following, we wish to investigate the performance and robustness of CIPH and PIPH, compared to the case where the user-provided circular initial guess is applied without refinement ("no homotopy" - NH). For this purpose, the reference problem described above is solved for each method for a set of 100 uniformly sampled initialization parameters $\boldsymbol{\pi}^0$ from the set defined by $\boldsymbol{\pi}^0_\mathrm{lb} \leq \boldsymbol{\pi}^0 \leq \boldsymbol{\pi}^0_\mathrm{ub}$

In the NH-case, performance heavily depends on the a priori knowledge of the user. To account for this fact, we introduce two different users. "User A" is an AWE developer with little a priori knowledge on the location of the optimal solution. Therefore, this user has samples from a wide initialization parameter set. "User B" on the other hand, is a control engineer who is familiar with the system and its optimal behavior for the given conditions. Therefore User B samples from a parameter set that is defined by a range that is a factor 3 smaller than that of User A, centered around the average parameters as evaluated at the solution of interest. Table 3 summarizes the sampling range for all initialization parameters, for both User A and B. The initial number of loops is chosen to be $N^0_\mathrm{l} = 1$.

Table 3. Initialization parameter bounds used for uniform sampling by users A and B.

Description	Variable	Min (A)	Max (A)	Min (B)	Max (B)	Units
Flight speed	$\dot{q}^0$	20.0	60.0	30.6	44.0	m s^{-1}
Tether length	l^0_t	300.0	600.0	300.0	391.8	m
Elevation angle	θ^0_e	30.0	50.0	26.3	32.9	deg
Cone angle	θ^0_c	20.0	30.0	14.9	21.5	deg
Phase angle	φ^0	0.0	360.0	0.0	93.9	deg
Tether diameter	d^0_t	1.0	5.0	1.6	2.9	mm

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

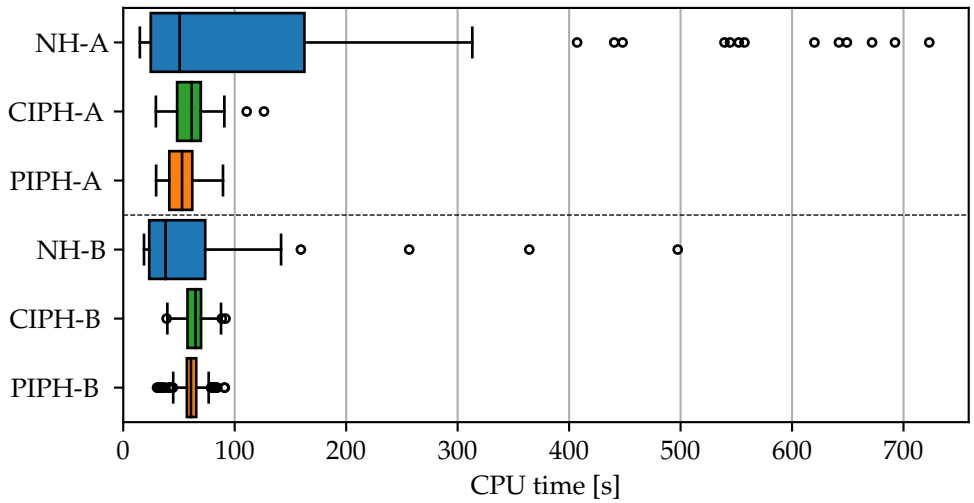

Figure 3. CPU wall time for the NH-, CIPH- and PIPH-method, obtained by initialization parameter sampling by User A and User B.

Figure 3 shows the CPU timing results resulting from the initialization sampling by User A and User B. For User A, NH leads to highly variable CPU timings, ranging from a peak timing of up to 12 minutes down to a minimum of 15 seconds. In two cases, NH does not converge as it exceeds the maximum number of iterations of the NLP solver. The minimum NH-timing is 50% lower than the best timings of the homotopy methods. Hence, it is possible for User A to “get lucky" and converge to a solution very fast without initialization refinement. However, the peak NH-timing is 8 times higher than the worst PIPH-timing and almost six times higher than the worst CIPH-timing. The average NH-timing is a factor 1.7 times higher than in the PIPH case and a factor 1.3 higher than the CIPH case. Therefore, User A benefits significantly from CIPH/PIPH in terms of expected computational performance and in particular in terms of timing consistency. PIPH is on average 13% faster than CIPH, while the peak timing is 30% lower.

For User B, with much better a priori knowledge, the computation times of NH significantly improve compared to user A: average timings are reduced by a factor 2.4, to a value slightly lower compared to CIPH/PIPH for User B. The peak NH-timing is reduced by a factor of 1.5, which is still a factor 5.5 larger than compared to CIPH/PIPH. Thus, while User B has a slightly better expected performance in the NH-case, he or she can still profit from the improved timing consistency provided by CIPH/PIPH. The difference in timings for the CIPH and PIPH methods is almost negligible. The average timings of these methods do not change much compared to the timings obtained for User A. This highlights the property that by pre-structuring the optimization path, the homotopy methods are not able to exploit a priori user knowledge to achieve a better average performance.

Overall the PIPH/CIPH CPU timings range between 30 and 100 seconds. This is comparable to the CPU timing range reported in [26] for similar model complexity and identical collocation grid (but excluding homotopy timings).

The NLP (52) has multiple local solutions and the choice of optimization algorithm influences the frequency with which certain solutions are found by the optimizer. In the experiments for User A, a total of 9 different local solutions were found. Fig. 4 shows the dominant, circular, optimal solution, while Fig. 5 shows as an example the third most frequent optimal solution, which is characterized by the well-known lemniscate flight pattern. Table 4 summarizes for each method the frequency of local solutions.

The homotopy methods almost always converge to the main solution of interest: out of a 100 trials, 100 for PIPH and 98 for CPIH. In the NH-case on the other hand, this

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

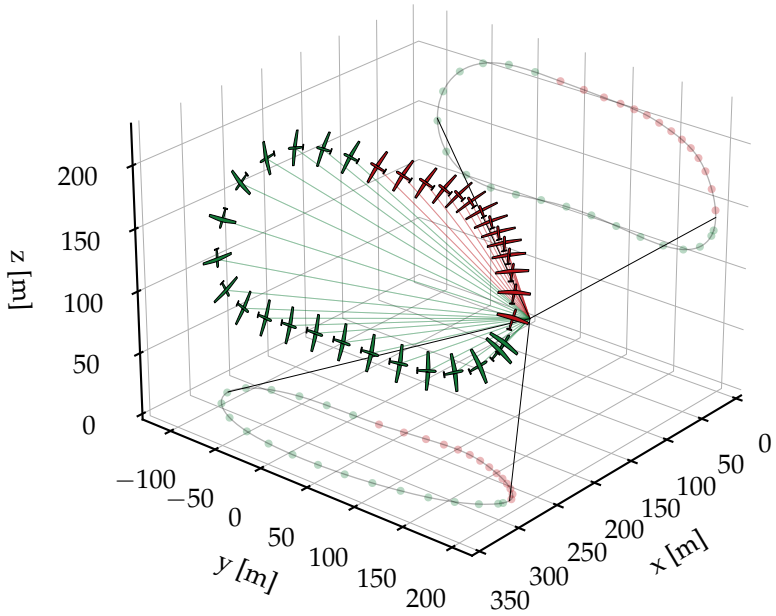

Figure 4. Locally optimal single-aircraft position and orientation trajectory #1 (circular pattern) as found by User A.

is only the case for 71 trials, while failing to converge in 2 cases. Hence, the homotopy methods do not only improve performance and reliability for User A, but they are also more stable in terms of optimization outcome. For User B, all methods always converge to the main solution.

Table 4. Solution frequency in a set of 100 trials, optimal time period T^*, average power output $\bar{P}^*$ and maximum consistency violation of locally optimal solutions found by User A.

Sol. #	NH	PIPH	CIPH	T^* [s]	$\bar{P}^*$ [kW]	$\|\mathbf{C}(\mathbf{x}^*(\cdot))\|_\infty$
1	71%	100%	98%	20.2	8.8	$8{\cdot}10^{-4}$
2	15%	-	-	24.0	8.7	$9{\cdot}10^{-4}$
3	7%	-	-	27.9	9.3	$1{\cdot}10^{-3}$
4	1%	-	1 %	32.5	9.0	$3{\cdot}10^{-3}$
5	1%	-	-	41.5	10.7	$7{\cdot}10^{-3}$
6	1%	-	-	41.2	10.6	$2{\cdot}10^{-2}$
7	1%	-	-	47.8	10.0	$4{\cdot}10^{-2}$
8	1%	-	-	37.4	10.3	$6{\cdot}10^{-3}$
9	0%	-	1%	40.4	10.5	$1{\cdot}10^{-2}$
Fail	2%	-	-	-	-	-

When comparing the different local solutions, we notice that average power output increases up to 22% with respect to the main solution for solutions with longer optimal time periods T^*. The solutions with a longer time period typically consist of more than one loop, which leads to a better ratio of reel-out vs. reel-in time, and thus a higher "pumping efficiency". This is in line with the results reported in [26].

Note that for increasing time period T^*, consistency condition satisfaction decreases. This is because the consistency condition trajectory is the periodic solution to the stable uncontrolled dynamics of the invariants. Hence, as simulation accuracy decreases, consistency conditions are moving away from the theoretically optimal solution of a constant zero value. For this reason, `AWEbox` automatically computes the consistency

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

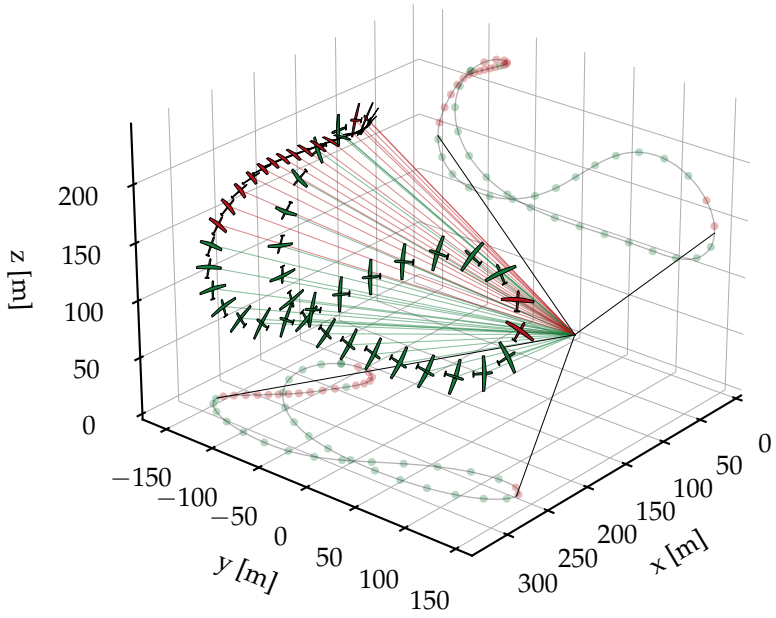

Figure 5. Locally optimal single-aircraft position and orientation trajectory #3 (lemniscate pattern) as found by User A.

conditions for each solution and gives out a user warning once a treshold is reached. The user can then increase the number of collocation intervals, the integration order, or lower the upper bound on T if applicable.

5.2. Dual-aircraft power curve

In the second case study, we compute the power curve for a dual-aircraft lift-mode system, i.e. $\mathcal{K} = \{2,3\}$, $\mathcal{L} = \{1\}$, and $P(1) = 0$, $P(2) = 1$ and $P(3) = 1$. We retain the model parameters and constraints and discretization of the single-aircraft case study, while adding the anti-collision constraint (43).

To give more structure to the problem, we propose the following modification to the OCP. We divide the time horizon in two separate intervals with associated time variables T_1 and T_2 and we define the total time period as $T := T_1 + T_2$. We then impose that the first interval is a single reel-out phase, and the second one a single reel-in phase:

$$\dot{l}_\mathrm{t} \geq 0\,, \quad \forall t \in [0, T_1] \tag{72}$$

$$\dot{l}_\mathrm{t} \leq 0\,, \quad \forall t \in (T_1, T]\,. \tag{73}$$

In the discrete time grid, 70 time intervals are allotted to the reel-out phase, and 30 intervals to the reel-in phase. The resulting NLP has 33464 variables, 31550 equality constraints and 1402 inequality constraints.

The intermediate barrier parameter is tuned manually to be $\tau_\mathrm{i} = 10^{-4}$ for both PIPH and SIPH. The PIPH-tuning is the same as in the single-aircraft case. SIPH performs a homotopy with $\gamma_\mathrm{p} = 10$ steps for every new parameter value. Additionally, the maximum number of NLP solver iterations is limited to 100 for both methods.

We search for solutions with three loops, i.e. $N_\mathrm{l}^0 = 3$. The reason for this is that the resulting trajectories fit well inside the time period bounds defined in Table 2, for all considered wind speeds. The remaining initialization parameters are set to $\dot{q}^0 = 50\frac{\mathrm{m}}{\mathrm{s}}$, $\theta_\mathrm{e}^0 = 25°$, $\theta_\mathrm{c} = 20°$, $\phi^0 = 0°$, $l_\mathrm{t}^0 = 640$ m, $l_\mathrm{s}^0 = 100$ m, $d_\mathrm{t}^0 = 4$ mm and $d_\mathrm{s}^0 = \frac{1}{\sqrt{2}} d_\mathrm{t}^0$. The

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

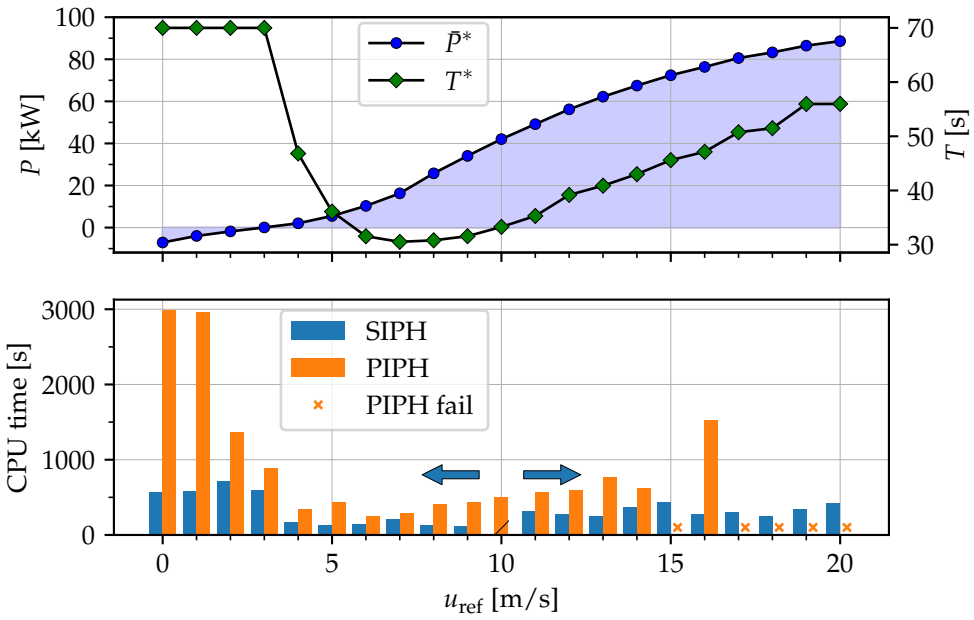

Figure 6. Average power output $\bar{P}^*$ and optimal time period T^* of a dual-aircraft AWE system (top) and CPU wall time for the PIPH and SIPH method (bottom) as a function of the reference wind speed u_{ref}.

secondary tether diameter is initialized under the assumption that the secondary tether force equals the main tether force divided by two.

The power curve for the proposed dual-aircraft system is obtained in the following manner. First the optimal trajectory and design is computed with PIPH for a reference wind speed of $u_{\text{ref}} = 10\ \frac{\text{m}}{\text{s}}$. The resulting optimal design is given by $l_{\text{s}}^* = 142.9$ m, $d_{\text{t}}^* = 4.3$ mm and $d_{\text{s}}^* = 3.2$ mm. The average power output is $\bar{P}^* = 42.0$ kW. Note that this is more than a factor of 4 higher than the single-aircraft solutions in the first case study, while the number of aircraft has only doubled. The power per wing surface area is thus more than doubled as a result of the reduced main tether drag and higher flight altitude. This is in line with the results reported in [10,12].

The optimal design parameters are then fixed, and the NLP is re-solved for u_{ref} ranging from $0\ \frac{\text{m}}{\text{s}}$ to $20\ \frac{\text{m}}{\text{s}}$. This is done once with PIPH, every time starting from the identical user-defined initial guess. Then it is done once using SIPH in two separate sweeps: once downwards and once upwards starting from the solution for $u_{\text{ref}} = 10\frac{\text{m}}{\text{s}}$.

Figure 6 shows the power curve obtained with SIPH, and additionally for each wind speed the optimal time period. Similar to the power curve computed in [37], we identify three operational regions. In the first region of zero wind speed up to the cut-in wind speed $u_{\text{ref}} = 3\ \frac{\text{m}}{\text{s}}$, power is consumed to keep the system airborne. The upper bound on T is active here, as the aircraft glide downwards about an almost vertical rotation axis during the reel-out phase. In the reel-in phase, potential energy is injected back into the system as the aircraft fly slow upwards trajectories. In the second operating region, power grows cubically until the design wind speed is reached. In the third region, power output still grows with the wind speed, but cubic growth is curtailed in order to satisfy the tether stress constraints. The main strategy to limit power output here is to increase the tether reel-out speed so as the decrease the available wind. The optimal time period increases with respect to the design wind speed, as the reel-out speed increases, while the reel-in speed is constrained and cannot grow proportionally. Figures 7 and 8 illustrate the reel-out and reel-in trajectories for $u_{\text{ref}} = 18\ \frac{\text{m}}{\text{s}}$.

Figure 6 also shows for each wind speed the associated CPU time for PIPH and SIPH. The computation times include both the CPU time for the homotopy procedures and the CPU time to solve the final problem $\mathcal{P}_{\text{f}}$. PIPH does not converge for the wind

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 1 December 2022 doi:10.20944/preprints202212.0018.v1

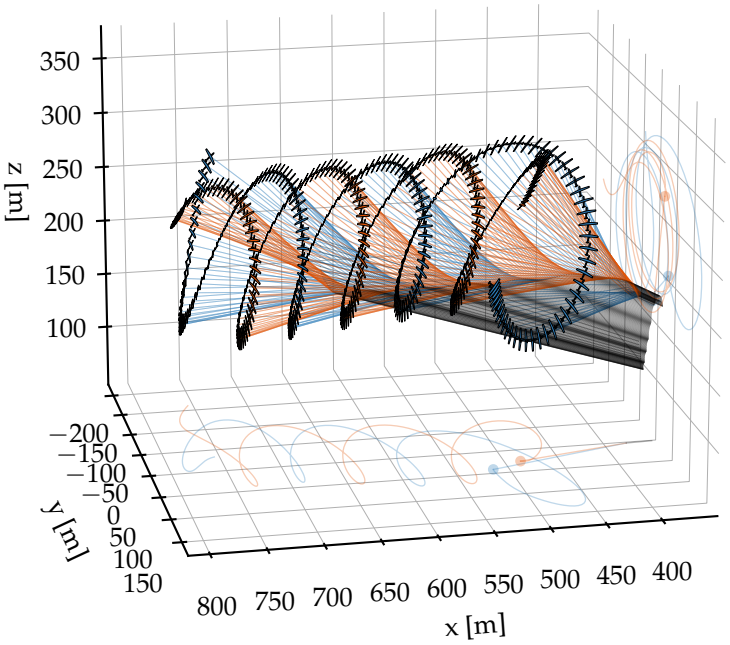

Figure 7. Optimal dual-aircraft flight trajectories for $u_{\text{ref}} = 18\ \frac{\text{m}}{\text{s}}$ (reel-out phase).

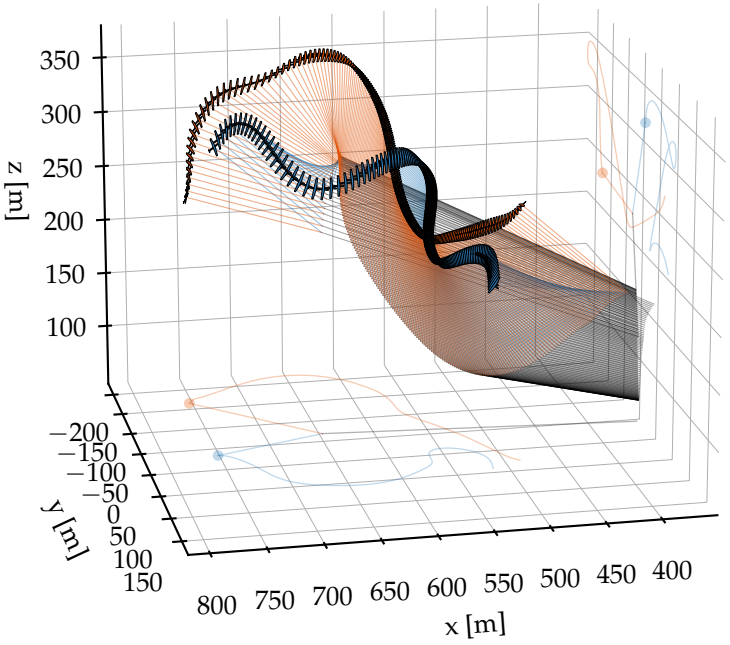

Figure 8. Optimal dual-aircraft flight trajectories for $u_{\text{ref}} = 18\ \frac{\text{m}}{\text{s}}$ (reel-in phase).

 doi:10.20944/preprints202212.0018.v1

speeds of 15 $\frac{\text{m}}{\text{s}}$ and 17 to 20 $\frac{\text{m}}{\text{s}}$. Note that convergence might be recovered for smaller update steps of the homotopy parameter $\bar{\boldsymbol{\phi}}$. However this falls outside the scope of this study.

SIPH outperforms PIPH at every single wind speed (where PIPH converges), but in particular at low wind speeds, when the optimal solution diverges significantly from the user-defined initial guess. Up until the wind speed of 15 $\frac{\text{m}}{\text{s}}$, the average CPU time is 5 minutes and 23 seconds for SIPH and 15 minutes and 20 seconds for PIPH.

6. Discussion

In this work, we presented `AWEbox`, an open-source `Python` toolbox for modeling and optimal control of single- and multi-aircraft AWE systems. We discussed the general multi-aircraft modeling structure, optimization ingredients and implementation details needed to efficiently compute power-optimal orbits for a wide range of system architectures and modeling options. In particular, we proposed and implemented two interior-point based homotopy method variants, in order to increase the performance and reliability of the optimization algorithms. These methods produce a feasible initial guess for the underlying NLP solver, based on an analytic initial guess shaped by the software user. In a numerical experiment, a reference single-aircraft problem was solved for a large set of different initial guesses.

The penalty-based homotopy method reduced the average and peak CPU timing with a factor 1.7 and 8 respectively, compared to the case when no homotopy method was applied by a user with little a priori knowledge. With good a priori knowledge available, the homotopy methods did not improve performance, but still the peak CPU timing was reduced by a factor 5.5. Overall, computation times were in the range of 30 - 100 seconds, which is competitive to those reported in the literature. Additionally, the penalty-based homotopy method consistently led to the same local solution, whereas the no-homotopy case resulted in different local solutions in 29 out of a 100 cases.

In a second case study, we computed a power curve of a dual-aircraft AWE system and compared the performance of the penalty-based homotopy method of the previous case study with that of a third homotopy method tailored for parametric sweeps with interior-point NLP solvers. The penalty-based method was not able to converge to a solution for all wind speeds, while the sweep method succeeded in doing so, while outperforming the penalty-based method on average by a factor 3 in terms of CPU timings. The average CPU timing per NLP solution was about 5 minutes.

Future work might entail model accuracy improvements, in particular concerning tether and induction modeling. Efficient problem formulations and implementations that include stability and robustness considerations would be a useful contribution, in particular for multi-aircraft systems. Finally, efficient algorithms that enable simultaneous trajectory and design optimization with expensive models (e.g. aero-elastic models) could lead to faster and more accurate system design loops.

Author Contributions: Conceptualization, J.D.S., R.L. and M.D.; methodology, J.D.S, R.L., T.B., E.M., S.G. and M.D.; software, J.D.S, R.L., T.B. and E.M.; validation, J.D.S.; investigation, J.D.S.; writing—original draft preparation, J.D.S.; writing—review and editing, R.L., E.M., S.G. and M.D.; visualization, J.D.S.; supervision, S.G. and M.D.; project administration, J.D.S.; funding acquisition, M.D.; All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by the EU via H2020-ITN-AWESCO (642 682) and via ELO-X 953348, by DFG via Research Unit FOR 2401 and project 424107692, and by an industrial project with the company Kiteswarms GmbH

Data Availability Statement: The software package as well as the code for the numerical experiments performed in this paper are accessible online at https://github.com/awebox/awebox.

Acknowledgments: The authors would like to thank Greg Horn, Mario Zanon and Joris Gillis for fruitful discussions on AWE optimization, in particular on penalty-based homotopy methods. Thomas Haas and Markus Sommerfeld were beta-users of the `AWEbox` software package and pro-

 doi:10.20944/preprints202212.0018.v1

vided useful user feedback. The authors would like to thank Reinhart Paelinck for his enthusiasm for and inspiring discussions on multi-aircraft AWE systems.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

AWE	Airborne wind energy
CIPH	Classic Interior-Point Homotopy
DCM	Direction Cosine Matrix
IP	Interior-Point
KKT	Karush-Kuhn-Tucker
LICQ	Linear Independence Constraint Qualification (LICQ)
MPC	Model Predictive Control
NH	No-Homotopy
NLP	Nonlinear Program
OCP	Optimal control problem
PIPH	Penalty-based Interior-Point Homotopy
QP	Quadratic Program
SOSC	Second-Order Sufficient Condition
SQP	Sequential Quadratic Programming
SIPH	Sweep Interior-Point Homotopy

Appendix A

Table A1. System parameters.

Description	Parameter	Value	Units
Aircraft mass	m_{K}	3.6800e+01	kg
Moment of inertia	$J_{\mathrm{K},x}$	2.5000e+01	$\mathrm{kg{\cdot}m^2}$
Moment of inertia	$J_{\mathrm{K},y}$	3.2000e+01	$\mathrm{kg{\cdot}m^2}$
Moment of inertia	$J_{\mathrm{K},z}$	5.6000e+01	$\mathrm{kg{\cdot}m^2}$
Moment of inertia	$J_{\mathrm{Kz},xz}$	4.7000e-01	$\mathrm{kg{\cdot}m^2}$
Wing span	b	5.5000e+00	m
Wing chord	c	5.5000e-01	m
Wing area	S	3.0000e+00	$\mathrm{m^2}$
Tether drag coefficient	$C_{\mathrm{D,t}}$	1.2000e+00	-
Tether density	ρ_{t}	1.4642e+03	$\mathrm{kg{\cdot}m^{-3}}$
Tether Baumgarte constant	κ_{t}	1.0000e+00	-
Tether attachment point	$r_{\mathrm{t,x}}, r_{\mathrm{t,y}}, r_{\mathrm{t,z}}$	0.0000e+00	m
DCM Baumgarte constant	κ_{R}	1.0000e+00	-
Wind friction coefficient	c_{f}	1.5000e-01	-
Reference height	z_{ref}	1.0000e+02	m
Sea level temperature	T_0	2.8820e+02	K
Temperature lapse rate	T_{L}	6.5000e-03	$\mathrm{K{\cdot}m^{-1}}$
Sea level air density	ρ_0	1.2250e+00	$\mathrm{kg{\cdot}m^{-3}}$
Tether safety factor	f_{s}	3.0000e+00	-
Anticollision safety factor	f_{b}	4.0000e+00	-

References

1. Diehl, M. Airborne Wind Energy: Basic Concepts and Physical Foundations. In *Airborne Wind Energy*; Springer Berlin Heidelberg, 2013; pp. 3–22.
2. Read, R. Kite Networks for Harvesting Wind Energy. In *Airborne Wind Energy: Advances in Technology Development and Research*; Springer Singapore, 2018.

3. De Schutter, J.; Leuthold, R.; Diehl, M. Optimal Control of a Rigid-Wing Rotary Kite System for Airborne Wind Energy. Proceedings of the European Control Conference (ECC), 2018.
4. Cherubini, A.; Papini, A.; Vertechy, R.; Fontana, M. Airborne Wind Energy Systems: A review of the technologies. *Renewable and Sustainable Energy Reviews* **2015**, *51*, 1461—-1476.
5. Fagiano, L.; Quack, M.; Bauer, F.; Carnel, L.; Oland, E. Autonomous Airborne Wind Energy Systems: Accomplishments and Challenges. *Annual Review of Control, Robotics, and Autonomous Systems* **2022**, *5*, 603–631. doi:10.1146/annurev-control-042820-124658.
6. Loyd, M. Crosswind Kite Power. *Journal of Energy* **1980**, *4*, 106–111.
7. Makani Power Inc.. Response to the federal aviation authority. Technical report, 2017.
8. Weber, J.; Marquis, M.; Lemke, A.; Cooperman, A.; Draxl, C.; Lopez, A.; Roberts, O.; Shields, M. Proceedings of the 2021 Airborne Wind Energy Workshop. Technical report, Golden, CO: National Renewable Energy Laboratory. NREL/TP-5000-80017, 2021.
9. Houska, B.; Diehl, M. Optimal Control for Power Generating Kites. Proceedings of the European Control Conference (ECC); , 2007; pp. 3560–3567.
10. Zanon, M.; Gros, S.; Andersson, J.; Diehl, M. Airborne Wind Energy Based on Dual Airfoils. *IEEE Transactions on Control Systems Technology* **2013**, *21*, 1215–1222.
11. Cherubini, A. Advances in Airborne Wind Energy and Wind Drones. PhD thesis, 2017.
12. De Schutter, J.; Leuthold, R.; Bronnenmeyer, T.; Paelinck, R.; Diehl, M. Optimal Control of Stacked Multi-Kite Systems for Utility-Scale Airborne Wind Energy. Proceedings of the IEEE Conference on Decision and Control (CDC), 2019.
13. De Schutter, J.; Leuthold, R.; Diehl, M. Power Smoothing in Utility-Scale Airborne Wind Energy Trajectory Optimization. Book of Abstracts of the Airborne Wind Energy Conference 2021, 2022.
14. De Schutter, J.; Harzer, J.; Diehl, M. Vertical Airborne Wind Energy Farms with High Power Density per Ground Area based on Multi-Aircraft Systems. arXiv preprint, 2022.
15. Licitra, G.; Koenemann, J.; Bürger, A.; Williams, P.; Ruiterkamp, R.; Diehl, M. Performance assessment of a rigid wing Airborne Wind Energy pumping system. *Energy* **2019**, *173*, 569–585. doi:10.1016/j.energy.2019.02.064.
16. Zanon, M.; Gros, S.; Diehl, M. Model Predictive Control of Rigid-Airfoil Airborne Wind Energy Systems. In *Airborne Wind Energy*; Ahrens, U.; Diehl, M.; Schmehl, R., Eds.; Springer, 2013.
17. Licitra, G.; Bürger, A.; Williams, P.; Ruiterkamp, R.; Diehl, M. Aerodynamic model identification of an autonomous aircraft for airborne wind energy. *Optimal Control Applications and Methods* **2019**, pp. 1–26. doi:10.1002/oca.2485.
18. Erhard, M.; Horn, G.; Diehl, M. A quaternion-based model for optimal control of the SkySails airborne wind energy system. *Zeitschrift für Angewandte Mathematik und Mechanik* **2017**, *97*, 7–24. doi:10.1002/zamm.201500180.
19. Vermillion, C.; Cobb, M.; Fagiano, L.; Leuthold, R.; Diehl, M.; Smith, R.S.; Wood, T.A.; Rapp, S.; Schmehl, R.; Olinger, D.; Demetriou, M. Electricity in the air: Insights from two decades of advanced control research and experimental flight testing of airborne wind energy systems. *Annual Reviews in Control* **2021**, *52*, 330–357. doi:https://doi.org/10.1016/j.arcontrol.2021.03.002.
20. Eijkelhof, D.; Schmehl, R. Six-degrees-of-freedom simulation model for future multi-megawatt airborne wind energy systems. *Renewable Energy* **2022**, *196*, 137–150. doi:https://doi.org/10.1016/j.renene.2022.06.094.
21. Cobb, M.K.; Barton, K.; Fathy, H.; Vermillion, C. Iterative Learning-Based Path Optimization for Repetitive Path Planning, With Application to 3-D Crosswind Flight of Airborne Wind Energy Systems. *IEEE Transactions on Control Systems Technology* **2020**, *28*, 1447–1459. doi:10.1109/TCST.2019.2912345.
22. Malz, E.C.; Koenemann, J.; Sieberling, S.; Gros, S. A Reference Model for Airborne Wind Energy Systems for Optimization and Control. *Renewable Energy* **2019**, *140*, 1004–1011.
23. Licitra, G.; Bürger, A.; Williams, P.; Ruiterkamp, R.; Diehl, M. Optimal Input Design for Autonomous Aircraft. *Control Engineering Practice* **2018**.
24. Gros, S.; Diehl, M. Modeling of Airborne Wind Energy Systems in Natural Coordinates. In *Airborne Wind Energy*; Springer-Verlag Berlin Heidelberg, 2013.
25. Gros, S.; Zanon, M.; Diehl, M. A Relaxation Strategy for the Optimization of Airborne Wind Energy Systems. Proceedings of the European Control Conference (ECC), 2013, pp. 1011–1016.
26. Horn, G.; Gros, S.; Diehl, M. Numerical Trajectory Optimization for Airborne Wind Energy Systems Described by High Fidelity Aircraft Models. In *Airborne Wind Energy*; Springer-Verlag Berlin Heidelberg, 2013.
27. Malz, E.; Verendel, V.; Gros, S. Computing the Power Profiles for an Airborne Wind Energy System based on Large-Scale Wind Data. *Renewable Energy* **2020**.
28. Trevisi, F.; Castro-Fernández, I.; Pasquinelli, G.; Riboldi, C.E.D.; Croce, A. Flight trajectory optimization of Fly-Gen airborne wind energy systems through a harmonic balance method. *Wind Energy Science* **2022**, *7*, 2039–2058.
29. Jonkman, J.; Hayman, G.; Mudafort, R.; Damiani, R.; Wendt, F.; USDOE.; Inc., G. KiteFAST. https://www.osti.gov//servlets/purl/1786962, 2018.
30. Fechner, U. Julia Kite Power Tools. Book of Abstracts of the Airborne Wind Energy Conference 2021, 2022.
31. Sánchez-Arriaga, G. LAgrangian Kite SimulAtors. https://github.com/apastor3/laksa, 2022.
32. Koenemann, J.; De Schutter, J.; Leuthold, R.; Licitra, G.; Diehl, M. OpenOCL - the open optimal control library. Book of Abstracts of the Airborne Wind Energy Conference 2019, 2019.
33. OpenAWE - Airborne Wind Energy Library. https://github.com/OpenAWE/.

34. https://github.com/awebox, 2022.
35. Andersson, J.; Gillis, J.; Horn, G.; Rawlings, J.; Diehl, M. CasADi – A software framework for nonlinear optimization and optimal control. *Mathematical Programming Computation* **2019**, *11*, 1–36.
36. Wächter, A.; Biegler, L.T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* **2006**, *106*, 25–57.
37. Leuthold, R.; De Schutter, J.; Malz, E.C.; Licitra, G.; Gros, S.; Diehl, M. Operational Regions of a Multi-Kite AWE System. European Control Conference (ECC), 2018.
38. Ascher, U.; Petzold, L. *Computer Methods for Ordinary Differential Equations and Differential–Algebraic Equations*; SIAM: Philadelphia, 1998.
39. Houska, B. Robustness and Stability Optimization of Open-Loop Controlled Power Generating Kites. Master's thesis, University of Heidelberg, 2007.
40. Gros, S.; Zanon, M. Numerical Optimal Control with Periodicity Constraints in the Presence of Invariants. *IEEE Transactions on Automatic Control* **2018**, *63*, 2818–2832.
41. Baumgarte, J. Stabilization of Constraints and Integrals of Motion in Dynamical Systems. *Computer Methods in Applied Mechanics and Engineering* **1972**, *1*, 1–16.
42. Gros, S.; Zanon, M.; Diehl, M. Baumgarte Stabilisation over the SO(3) Rotation Group for Control. Proceedings of the IEEE Conference on Decision and Control (CDC), 2015, pp. 620–625.
43. Archer, C. An Introduction to Meteorology for Airborne Wind Energy. In *Airborne Wind Energy*; Springer Berlin / Heidelberg, 2013.
44. Archer, C.; Delle Monache, L.; Rife, D.L. Airborne wind energy: Optimal locations and variability. *Renewable Energy* **2014**, *64*, 180–186.
45. Malz, E.C.; Hedenus, F.; Göransson, L.; Verendel, V.; Gros, S. Drag-mode airborne wind energy vs. wind turbines: An analysis of power production,variability and geography. *Energy* **2020**, *139*.
46. Zanon, M.; Horn, G.; Gros, S.; Diehl, M. Control of Dual-Airfoil Airborne Wind Energy Systems Based on Nonlinear MPC and MHE. Proceedings of the European Control Conference (ECC), 2014, pp. 1801–1806.
47. Faulwasser, T.; Grüne, L.; Müller, M. Economic Nonlinear Model Predictive Control. *Foundations and Trends® in Systems and Control* **2018**, *5*, 1–98.
48. Rawlings, J.B.; Mayne, D.Q.; Diehl, M.M. *Model Predictive Control: Theory, Computation, and Design*, 2nd ed.; Nob Hill, 2017.
49. Hairer, E.; Nørsett, S.; Wanner, G. *Solving Ordinary Differential Equations II – Stiff and Differential-Algebraic Problems*, 2nd ed.; Springer Series in Computational Mathematics, Springer: Berlin, 1996.
50. Nocedal, J.; Wright, S.J. *Numerical Optimization*, 2 ed.; Springer Series in Operations Research and Financial Engineering, Springer, 2006.
51. HSL. A collection of Fortran codes for large scale scientific computation., 2011.
52. Wächter, A.; Biegler, L.T. Line Search Filter Methods for Nonlinear Programming: Motivation and Global Convergence. *SIAM Journal on Optimization* **2006**, *16*, 1–31.
53. Allgower, E.L.; Georg, K. *Introduction to Numerical Continuation Methods*; Colorado State University Press, 1990.
54. Deuflhard, P. *Newton methods for nonlinear problems: affine invariance and adaptive algorithms*; Vol. 35, Springer, 2011.
55. Malz, E.C.; Zanon, M.; Gros, S. A Quantification of the Performance Loss of Power Averaging in Airborne Wind Energy Farms. European Control Conference (ECC), 2018. doi:10.23919/ECC.2018.8550357.
56. Haverbeke, N.; Diehl, M.; Moor, B.D. A structure exploiting interior-point method for moving horizon estimation. Proceedings of the IEEE Conference on Decision and Control (CDC); , 2009. doi:10.1109/CDC.2009.5400804.
57. Zanelli, A.; Quirynen, R.; Jerez, J.; Diehl, M. A homotopy-based nonlinear interior-point method for NMPC. Proceedings of the IFAC World Congress; , 2017.
58. Haas, T.; De Schutter, J.; Diehl, M.; Meyers, J. Wake characteristics of pumping mode airborne wind energy systems. *Journal of Physics: Conference Series* **2019**.
59. Verschueren, R.; Frison, G.; Kouzoupis, D.; Frey, J.; van Duijkeren, N.; Zanelli, A.; Novoselnik, B.; Albin, T.; Quirynen, R.; Diehl, M. acados – a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation* **2021**, pp. 147–183. doi:10.1007/s12532-021-00208-8.
60. Listov, P.; Jones, C. PolyMPC: An efficient and extensible tool for real-time nonlinear model predictive tracking and path following for fast mechatronic systems. *Optimal Control Applications and Methods* **2020**, *41*, 709–727. doi:10.1002/oca.2566.