Optimization of an Airborne Wind Energy System using Constrained Gaussian Processes with Transient Measurements

Sanket Sanjay Diwale[1] and Ioannis Lymperopoulos[1] and Colin N. Jones[1],

***Abstract*— Airborne wind energy systems are built to exploit the stronger and more consistent wind prevalent at high altitude. This requires a reliable controller design that keeps the airborne system flying while respecting all operational and safety constraints. A frequent design for such a system includes a flying airfoil tethered to a ground station. Here, we demonstrate an on-line data based method that optimizes the average towing force of such a system in the presence of altitude constraints and varying wind. We utilize Gaussian Processes to learn the mapping from the input to the objective, constraint and state dynamic functions of the system. We then formulate a chance - constrained optimization problem that takes into consideration uncertainty in the learned functions and finds feasible directions for improvement. Simulation studies show that we can find near optimal set points for the controller without the use of significant assumptions on model dynamics while respecting the unknown constraint function. The results demonstrate an improved performance over our previous work which was restricted to steady state measurements.**

I. Introduction

The power available in a wind stream is proportional to the cube of its speed [1]. This is a major reason behind the continuous increase in the height of modern wind turbines. However, further scaling the support mast incurs considerable costs, can arouse environmental opposition and has certain structural limitations. To circumvent this obstacle a number of commercial and research efforts have been focusing in Airborne Wind Energy (AWE) systems that operate without the need for structural support [2], [3].

We demonstrate here a data based optimization algorithm on an AWE design actively used by a commercial company (Skysails) in large marine vessels to increase their fuel savings [4]. A tethered flexible airfoil is launched from a mounting station at the front of the ship towards the sky where it performs figure eight loops using a custom made low level controller. In favourable wind conditions the aerodynamic force generated upon the foil is transferred through the tether to the boat and pulls forward, reducing the effort of its onboard motors. However, such a controller can neither guarantee an optimal trajectory nor that the airfoil will keep flying above an altitude safety threshold.

We have previously addressed this problem in the framework of steady state constrained optimization and have demonstrated that a data based approach can be a potential solution [5]. We were however examining stable wind conditions and using training measurements only after reaching steady state (for the system here this implies at least 10 loops with the same inputs applied and fairly constant wind). We now present a more flexible solution where transient measurements (after each loop) are utilised. The general optimization setting is

$$\begin{aligned} \max_{x,u} \quad & P(x,u,w) \\ \text{s.t.} \quad & \dot{x} \;\; = f(x,u,w), \\ & 0 \;\; \leq G(x,u,w) \end{aligned} \tag{1}$$

where $P : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}$ is the objective function, $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}^{n_x}$ describes the system dynamics and $G : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_w} \to \mathbb{R}$ is the constraint function. The variables $x \in \mathbb{R}^{n_x}$ describe the states of the system and can be indirectly manipulated through the inputs u which might be bounded by additional constraints in a set $\mathcal{X} \subseteq \mathbb{R}^{n_u}$. The variables w represent the exogenous signals (here the wind) on which we have no control. The functions can potentially be nonlinear and non-convex, unknown, and are being actively learned through measurements using the framework of Gaussian Processes (GP) [6]. GPs can be used for regression with relatively few assumptions on the structure of the unknown function and have been extensively used in unconstrained optimization [7], [8].

The challenge arises in the presence of constraints that need to be learned as well. A first approach to this problem but with known constraints was given in [9] while [10] assumes that the outputs of the objective and constraint functions are dependent. The authors in [11] use an augmented Lagrangian approach to transform the problem to an unconstrained one. We follow here an approach where we also learn the system dynamics and make predictions about its future states which incur additional reachability type constraints. We use simulation results to demonstrate that the AWE system can be both optimized and remain adaptive to wind variation while respecting the altitude safety thresholds.

Section II describes the AWE system, section III gives an overview to GPs for optimization, section IV introduces the algorithm proposed, section V presents simulation results, while section VI provides a conclusion and discusses future perspectives.

II. System Description

Skysails has developed a commercial system to assist the propulsion of large maritime vessels. The system consists of a flying foil, a control pod applying deflections to the flexible

[1] Sanket Sanjay Diwale and Dr. Ioannis Lymperopoulos and Prof. Colin N. Jones are with the Automatic Control Laboratory at the Department of Mechanical Engineering in EPFL, Lausanne - 1015, Switzerland sanket.diwale@epfl.ch, ioannis.lymperopoulos@epfl.ch, colin.jones@epfl.ch

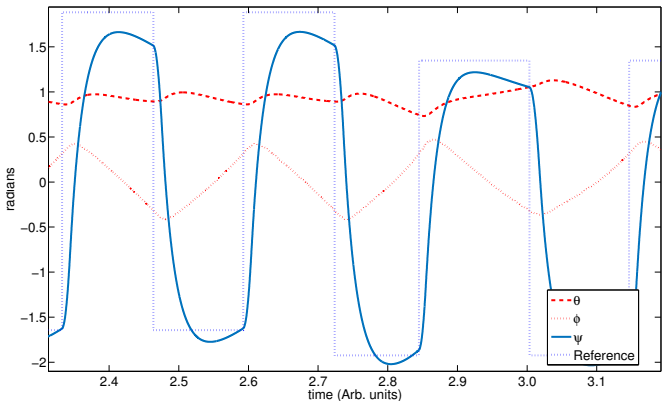

Fig. 1: Low level tracking control for changing set points. The yaw (ψ - kite orientation) controller tracks the level reference signal which changes value when a crossing occurs. The positional states (θ and ϕ) are affected by ψ.

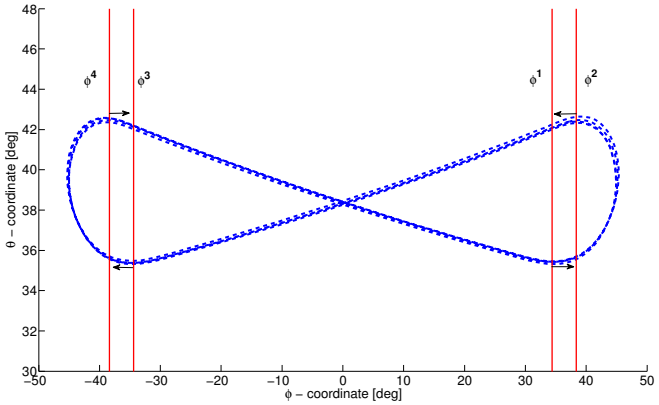

Fig. 2: Example of kite trajectory (blue) and switching surfaces (red) for the Skysails controller in the spherical coordinate system. Arrows denote the direction of crossing for which a switching surface is active.

surface and a mounting unit that transfers the tension from the tether to the boat.

A. System Dynamics

The skysails model neglects the mass in the dynamics (a reasonable assumption for large counteracting aerodynamic forces) and substitutes the sagging tether by a rigid rod (acceptable while the tether remains under substantial tension). The equations of motion describing the system are

$$\dot{\vartheta} = \frac{wE}{L}\cos\vartheta\cos\psi - \frac{w}{L}\sin\vartheta \tag{2}$$

$$\dot{\varphi} = -\frac{wE}{L}\frac{\cos\vartheta}{\sin\vartheta}\sin\psi \tag{3}$$

$$\dot{\psi} = wEg\cos(\vartheta)\delta + \dot{\varphi}\cos\vartheta. \tag{4}$$

The spherical coordinates (ϑ, φ) and orientation (ψ) of the airfoil represent the states of the system. The exogenous signal is the wind speed (w). Note that the system coordinates are defined in such a way that one of the horizontal axes remains always aligned to the wind direction. The main system parameters are the airfoil glide ratio (E), the deflection coefficient (g) and the tether length (L). Finally, δ is the deflection applied to the kite affecting its orientation.

The system exhibits nonlinear behaviour even with considerable simplification of the aerodynamics and it is difficult to find closed form expressions for the functions of interest. Numerical optimization results using model based approaches, while useful, would be challenged in realistic situations where the model mismatch and wind conditions are unknown.

B. Skysails Control

A simple but robust low level controller developed by Skysails is used for tracking "figure eight" loops. The controller uses a cascaded control scheme where a given set-point on the kite orientation (yaw angle, ψ) is tracked by applying deflections to the kite, see Figure 2. This set-point changes during every single loop at predefined switching positions. The set point values at each corresponding switch represent the high level decision variables employed by our optimization algorithms.

When the kite crosses a particular section the yaw set point is changed to a predefined level such that the kite flies a stable loop. The kite is considered to be crossing the i^{th} switching surface when:

$$\varphi - \varphi_i = 0 \;\&\; k\dot{\varphi} > 0; \quad k = 1 \text{ or } -1 \tag{5}$$

Each surface is defined by a constant φ_i giving its position in the φ space and a constant k taking values 1 or -1 (to define the orientation of the crossing). Each surface considers crossings only in one direction and ignores crossings in the other direction. A depiction can be seen in Figure 2. A detailed analysis can be found in [12], [13].

III. Gaussian Processes for Constrained Optimization

Gaussian processes are an extension of the multivariate Gaussian distribution to infinite dimensional spaces where any finite combination of outputs is jointly Gaussian. They are used in supervised learning mainly for regression from a Bayesian perspective. A GP essentially describes a distribution in the function space. GPs are more flexible than parametric models and due to Gaussianity assumptions can make predictions, incorporate new measurements and quantify uncertainty using mostly closed-form expressions [14].

A. Gaussian Processes - Regression

A GP is fully specified by its mean $m(x)$ and covariance (or kernel) function $k(x, x')$. For a finite set of N training data points $\mathcal{D} = \{x_i, y_i\}_{i=1:N}$ generated by an unknown function $h : \mathbb{R}^d \to \mathbb{R}$, the GP assumes a multivariate distribution

$$y_{1:N} \sim \mathcal{N}(m_{1:N}, K(x_{1:N}, x_{1:N})) \tag{6}$$

where $y_{1:N} = h(x_{1:N})$ and $m_{1:N}$ is compact notation for $m(x_{1:N})$. For any point (or collection of points) x^*, we can

analytically predict the output of the learned function $y^* = h(x^*)$ as a conditional distribution

$$y^*|\mathcal{D}, x^* \sim \mathcal{N}(\mu(x^*|\mathcal{D}), \sigma(x^*|\mathcal{D})), \tag{7}$$

where

$$\mu(x^*|\mathcal{D}) = m^* + k(x^*, x_{1:N})K_{1:N}^{-1}(y_{1:N} - m_{1:N})$$

and

$$\sigma(x^*|\mathcal{D}) = k(x^*, x^*) + k(x^*, x_{1:N})K_{1:N}^{-1}k(x_{1:N}, x^*)$$

with compact notation $K_{1:N} = K(x_{1:N}, x_{1:N})$.

The element that encodes our assumptions on the shape and properties of the learned function is the kernel. We use here an ARD (Automatic Relevance Determination) Squared Exponential (SE) kernel, defined as

$$k_{SE}(x, x') = \sigma_y^2 \exp(-\frac{(x - x')^T \Lambda^{-1}(x - x')}{2}) + \sigma_n^2 \delta_{ii'},$$

where $\Lambda = diag(\lambda_1, ..., \lambda_d)$ and $\delta_{ii'}$ is Kroenecker's delta (= 1 iff $i = i'$, 0 otherwise). The parameters $\eta = \{\sigma_y, \lambda_{1:d}, \sigma_n\}$ (usually called hyperparameters in the GP terminology) are being learned from the measurements $\mathcal{D}$. This kernel includes measurement uncertainty (σ_n). We select as appropriate hyperparameters η those that maximize the log marginal likelihood for the observed data (for many cases this function is nonconvex and presents an optimization challenge). The selection of hyperparameters is essentially the learning phase that allows for the construction of a covariance matrix including any point of interest and the already sampled points. The function output at this point of interest can be easily calculated in a closed form manner using (7).

B. Gaussian Processes - Optimization

Instead of directly optimizing over the mean of the learned function we follow a methodology where an auxiliary acquisition function indicates the next sampling point and gradually progresses towards the optimum while also learning the unknown function [15]. This auxiliary function is called Expected Improvement (EI) and quantifies the expected magnitude of improvement over the current best value $y_{max} = \max y_{1:N+K}$ (where N are the initial training points and K the additional points sampled during the process). This function can be analytically derived for any given point x in the search space as

$$EI(x) = \sigma(x)[v\Phi(v) + \phi(v)], \tag{8}$$

where $v = (y(x) - y_{max})/\sigma(x)$, $\sigma(x)$ is the variance as predicted by the GP and $\Phi(\cdot)$, $\phi(\cdot)$ are the cumulative and probability density functions for the normal distribution. EI promotes points with high variance and large mean and allows for sampling in locations where the mean is low but there exists substantial potential for improvement.

IV. Learning and Optimization with Transient Measurements

Before proceeding in the main contribution of the paper, where we accommodate for the use of transient measurements, we review our previous approach that is using only steady state measurements.

A. Steady State Optimization

For this setting we assume that the decision variables (u) are being repeatedly applied until the system settles in a stationary orbit. This way the state (x) of the system does not affect the objective (P) or the constraint function (G) which now depend only on the input applied. Thus, we solve the problem in the static optimization setting (no dynamics learned) with a chance constrained formulation

$$\begin{array}{lll} \max\limits_{u} & EI_P(u) & \\ \text{s.t.} & \mathbb{P}(GP_G(u) \leq 0) \geq & 1 - \beta \\ & \sigma_G(u) \leq & \alpha\sigma_n, \end{array} \tag{9}$$

where GP_G is a GP approximating the constraint function and EI_P is the expected improvement for the GP_P (approximating the objective function). The constraint function is replaced with an auxiliary statement requiring that the probability of satisfying the constraint at a sample point u is larger than $(1-\beta)$. By setting β appropriately we can adjust the conservativeness of our approach. Due to the use of GPs we can easily derive this probability at any point u.

The second constraint restricts search only in locations where the variance (σ_G) of GP_G is below a threshold proportional to the measurement uncertainty ($\alpha\sigma_n$). Large variance implies that this part of the domain has not been adequately sampled and usually the estimates are not accurate even in a probabilistic setting (GPs like most regression methods have difficulties in extrapolating). This reduces the convergence rate but does not allow significant constraint violations. The full procedure can been seen in Algorithm 1.

Algorithm 1 Stready State Optimization

1: **Initialization.** Choose N points in the feasible set (forming $\mathcal{D}$)
2: **Training.** Learn the hyper-parameters for GP_G and GP_P
3: **Optimization.** Find P_{max} among the sampled points
4: - Calculate EI_P using P_{max} for the objective function
5: - Calculate the Chance Constraint on GP_G
6: - Calculate the Variance Constraint on GP_G
7: **Next Point.** Select the point $\tilde{u} = \arg\max EI_P$ that satisfies all constraints
8: **if** $P(\tilde{u}) \geq P_{max}$ and $\max EI_P \leq$ tolerance **then**
9: **Terminate** and use $u^* = \tilde{u}$
10: **else**
11: Add $(\tilde{u}, P(\tilde{u}))$ in $\mathcal{D}$ and Go To 2
12: **end if**

B. Optimization with Transient Measurements

We solve now the optimization problem, as described in the general setting (1), that incorporates system dynamics. We define the discrete dynamics as the change in the system state after one complete loop. A loop is complete when the kite crosses a particular switching surface (described in subsection II-B). We assume that the evolution of the state

can be directly observed within some measurement error. The system dynamics can then be represented as

$$x_{n+1} = f(x_n, u_n, w_n) \tag{10}$$

where (x_n, w_n) are the state of the system and the exogeneous signal (here the wind) at the beginning of the n^{th} loop, u_n is the control sequence throughout the loop and x_{n+1} is the state at the end of the loop. We also measure the objective $P(x_n, u_n, w_n)$ and the constraint $G(x_n, u_n, w_n)$ function output over a single loop.

Since we can only observe the outcome of applying u_n at a specific (x_n, w_n) and have no model for $f(\cdot), P(\cdot)$ and $G(\cdot)$ we learn them as GP regression models, as described in subsection III-A, and denote them GP_f, GP_P and GP_G respectively.

For this setting the formulation of the optimization problem becomes

$$\max_{x_s, u_s} \quad EI_P(x_s, u_s, w_n) \tag{11a}$$

s.t.

$$\hat{x}_{n+1} \sim GP_f(x_n, u_s, w_n) \tag{11b}$$
$$\hat{G} \sim GP_G(x_n, u_s, w_n) \tag{11c}$$
$$\mathbb{P}(\, \|x_s - \hat{x}_{n+1}\| < \epsilon_s \,|\, x_n = x_s \,) > 1 - \beta_s \tag{11d}$$
$$\mathbb{P}(\, \|x_s - \hat{x}_{n+1}\| < \epsilon_r) > 1 - \beta_r \tag{11e}$$
$$\mathbb{P}(\, \hat{G} > 0 \,|\, x_n = x_s \,) > 1 - \beta_{G_s} \tag{11f}$$
$$\mathbb{P}(\, \hat{G} > 0) > 1 - \beta_{G_t} \tag{11g}$$
$$\sigma_f(x_0, u_s, w_n) < \alpha\sigma_{n_f} \text{ for } x_0 = x_s, x_n \tag{11h}$$
$$\sigma_G(x_0, u_s, w_n) < \alpha\sigma_{n_G} \text{ for } x_0 = x_s, x_n \tag{11i}$$

$$P_{best} = \max_{x_s, u_s} \mu_P(x_s, u_s, w_n) \tag{12a}$$

s.t. (11b), (11c), (11d), (11e), (11f), (11g), (11h), (11i)

The objective function, in (11a), is the EI for the GP_P (here P is the average force for a single loop). We call the optimum Expected Improvement obtained from (11) as EI_P^* and the corresponding arguments, x_s^*, u_s^*. The subscript (s) denotes steady state conditions. It is important to note here that we are searching for a steady state solution (x_s, u_s) while the system might be residing in a transient state x_n. Furthermore, we can only directly manipulate u_s, while x_s is the steady outcome of that input sequence.

The dynamics of the system are described in a probabilistic setting, through GP_f as in (11b). The probabilistic output of the constraint function is given in (11c).

The constraint (11d) accepts only the (x_s, u_s) which approximate a stationary orbit (tolerance ϵ_s) with high probability. The chance constraint (11e) restricts to steady states x_s close to states that are reachable in one loop starting from the current state x_n. This is important to maintain consistency in the decisions taken by the algorithm from one iteration to another and prevent the steady states chosen to continuously keep jumping in inconsistent directions. The constraints (11f) and (11g) restrict the choice of x_s, u_s to pairs which do not violate the altitude constraint in steady state and transient respectively.

The bounds on the uncertainty of the GP output is described in constraints (11h) and (11i) which act as a trust region constraint. This way we avoid selecting points for which a GP exhibits very large variance. Here, σ_{n_f} and σ_{n_G} are the measurement noise of the state (angles ϑ, ψ) and the altitude respectively. The rest of the variables ($\epsilon_s, \epsilon_r, \beta_s, \beta_r, \beta_{G_s}, \beta_{G_t}, \alpha$, with common values $\epsilon_\cdot, \beta_\cdot = 0.1$ and $\alpha = 3$) act as tuning parameters and govern the aggressiveness of the search.

A fundamental difficulty with the transient approach is that we do not have direct measurements of P for any steady state (x_s, u_s). To evaluate EI_p however, we need a P_{best} over which all the possible next sampling points are to be compared. We choose as P_{best}, using (12), the point that maximises over the mean prediction of average force by GP_P. With this formulation, P_{best} plays the role of y_{max} in (8). The corresponding input is called u_{best}.

The complete method can be seen in Algorithm 2. Note, that here there is no terminating condition, since the exogenous signal might be constantly changing and the system seldomly resides in steady state.

Algorithm 2 Transient Optimization

1: **Initialization.** Choose N set-points (u) in the feasible set and observe the transients (for Objective, Constraint, Dynamics); (forming an initial $\mathcal{D}$)
2: **Training.** Learn hyper-parameters for GP_G, GP_P, GP_f
3: **Optimization.** Find P_{best}, u_{best} from (12) for current conditions of (x_n, w_n)
- Calculate EI_P^* and u_s^* using P_{best} for the objective function by solving (11)
4: **Next Point.** Select the next set-point $\tilde{u}_n$ as follows:
5: **if** $EI_P^*(x_s, u_s, w_n) \leq EI_{threshold}$ and u_{best} is feasible **then**
6: Use $\tilde{u} = u_{best}$
7: **else**
8: Use $\tilde{u} = u_s^*$
9: **end if**
10: Add measured data to $\mathcal{D}$ and Go To 2

V. Results

We implement our algorithm in Matlab and the Skysails controller in Simulink. For the GP regression we use the object-oriented Matlab toolbox TacoPig [16]. We use the skysails model described in II-A (with parameters E = 10, L = 520, g = 1/7) to generate data corrupted with noise. No algorithm is provided with any direct knowledge of this model.

For constant wind conditions, we start both Algorithms 1 and 2 with N = 15 initial training points in the feasible set. This is possible from prior experience of the operator. For Algorithm 1, set points are repeated for 10 loops, until the kite reaches a steady trajectory. On the other hand,

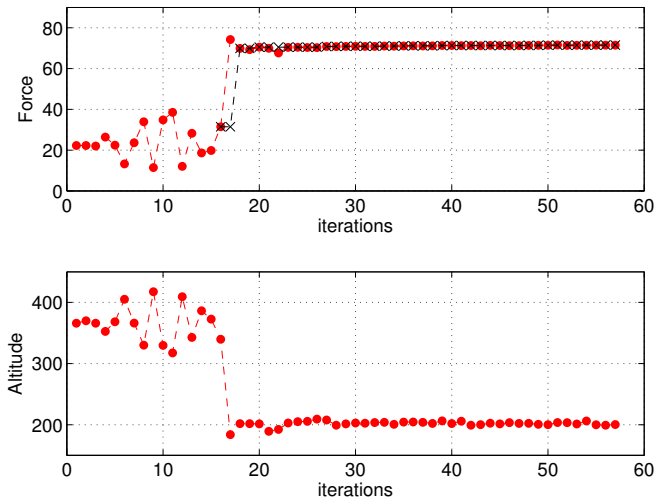

Fig. 3: Convergence of Algorithm 1 in constant wind condition and 200m altitude constraint. Red circles represent measured values and black crosses represent P_{max} (predicted)

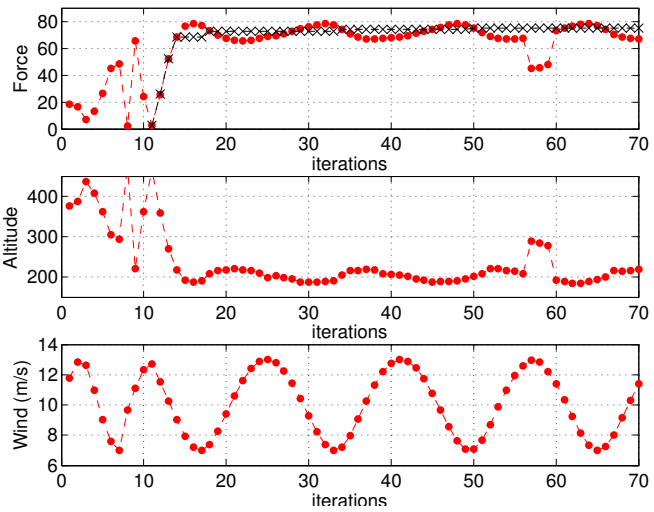

Fig. 4: Maximum objective tracking with Algorithm 1 under varying wind conditions.

for Algorithm 2 we take measurements after every loop. Both algorithms show convergence (within 20-30 samples) to within 5% of the optimum calculated using a numerical optimal control solver GPOPS-II, see [17], which makes use of the model and has full control throughout the trajectory and not just at the switching surfaces. The wind dependence of the objective value has been normalised so that the results are comparable across different wind conditions.

Figure 3 shows the performance of Algorithm 1 in constant wind. Table I compares with Algorithm 2 for the same conditions. Both algorithms perform well in finding the optimum, however Algorithm 2 using 1-step ahead predictions can avoid large constraint violations and converges much faster to the optimum. Also Algorithm 2 guarantees constraint satisfaction both in predicted stationary orbit and during transients.

TABLE I: Performance comparisons in constant wind

Method	**Final value**	**Max Violation (m)**	**Loops**
Algorithm 1 (2 surfaces)	71.17	3.2	320
Algorithm 1 (4 surfaces)	72.60	4.77	300
Algorithm 2 (2 surfaces)	71.05	0.00	65
GPOPS-II	74.61	0.00	-

Algorithm 1 was mainly developed under the assumption of constant wind and is not well suited for handling varying wind conditions. This is due to the large difference between the time at which the decision is made and the time at which the system reaches the corresponding steady state. Algorithm 2 overcomes this limitation by sampling in transients and thus increasing the frequency at which decisions are revised.

Table II summarizes performance results in varying wind conditions. Algorithm 2 can track the optimum without significant violations and is able to learn control policies as the exogenous conditions vary. Figure 5 shows the progress of the transient algorithm under varying wind conditions (and Figure 4 for Algorithm 1), while Figure 7 shows the tracked trajectory under these conditions. Figure 6 shows the evolution of the system states and set points with time and wind. Algorithm 1 performs slightly better because it is oblivious to the changed wind condition at the time of execution and tends to move closer to the constraint for which the average towing force is more favorable.

TABLE II: Performance comparisons for varying wind

Decision space	**Final value**	**Max Violation (m)**	**Loops**
Algorithm 1 (2 surfaces)	75.09	15.35	320
Algorithm 2 (2 surfaces)	73.84	2.90	47

VI. Conclusions

We present here two algorithms for optimizing the towing force produced by an AWE system that take into account altitude constraints. Both methods are model free and use Gaussian Processes to progressively learn the system dynamics, objective and constraint functions. The first method uses only steady state measurements and converges slowly towards the optimum. The second method utilizes measurements from transient state and converges an order of magnitude faster towards the optimum. Moreover it better adapts to changing wind conditions without significant constraint violations (below measurement error). The results are then compared to an off-line optimal control numerical solver (with full knowledge of the model and extensive input control, not simply controller set points) and the performance in terms of the objective is found similar. Future plans include testing the

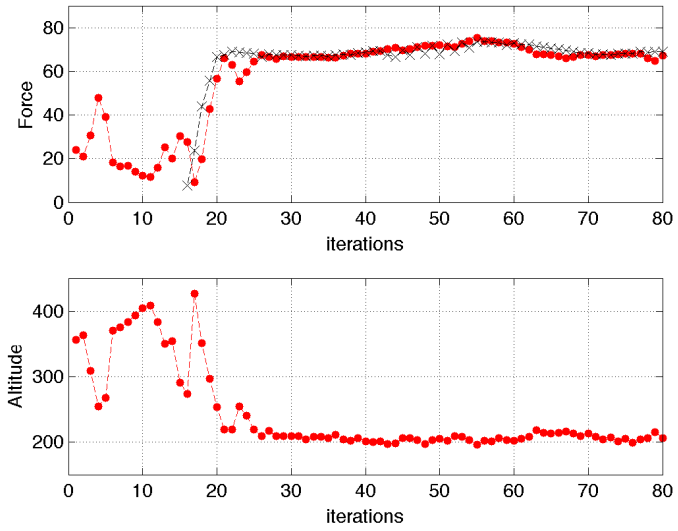

Fig. 5: Towing force optimization for a 200m altitude constraint and varying wind condition using Algorithm 2.

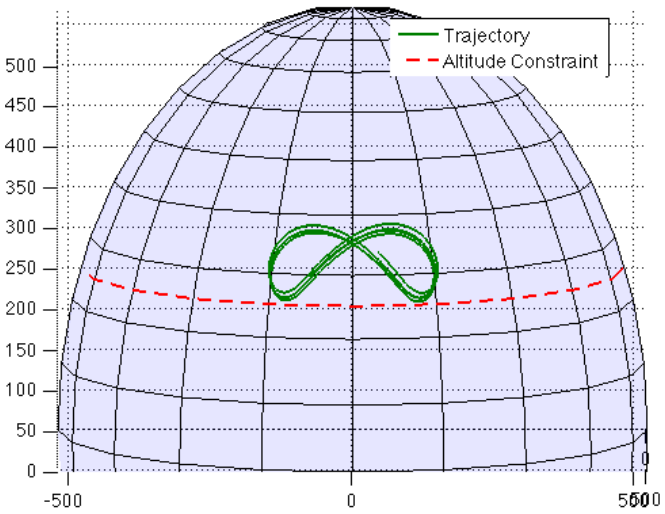

Fig. 7: Kite trajectory after applying Algorithm 2 in varying wind conditions.

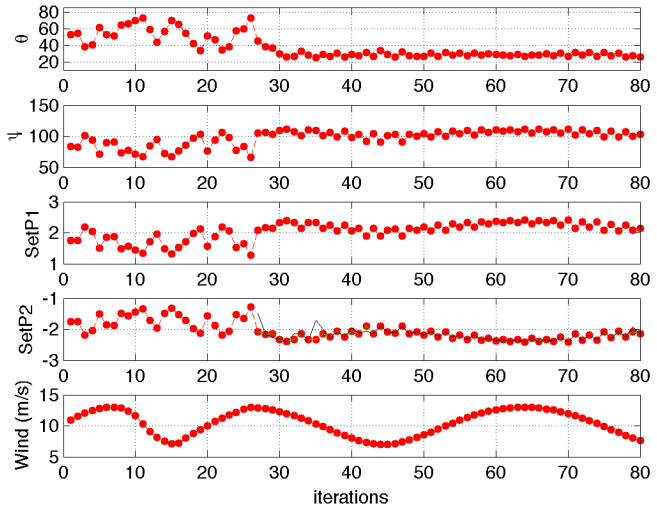

Fig. 6: System states, set-points and wind measurements during the application of Algorithm 2.

algorithm in an experimental testbed against realistic wind conditions and expanding the second algorithm for multiple steps ahead prediction horizon.

ACKNOWLEDGMENT

This work is funded by the Sinergia project "Autonomous Airborne Wind Energy" (A^2WE) of the Swiss National Science Foundation (SNSF).

REFERENCES

[1] I. Argatov, P. Rautakorpi, and R. Silvennoinen, "Estimation of the mechanical energy output of the kite wind generator," *Renewable Energy*, vol. 34, pp. 1525–1532, Jun. 2009.

[2] A. Uwe, D. Moritz, and S. Roland, *Airborne Wind Energy*. Dordrecht, Netherlands: Springer, 2013.

[3] L. Fagiano and M. Milanese, "Airborne wind energy: An overview," in *American Control Conference (ACC 2012)*, Montreal, QC, Jun. 2012, pp. 3132–3143.

[4] Skysails propulsion for cargo ships. [Online]. Available: http://www.skysails.info/english/skysails-marine/

[5] S. S. Diwale, I. Lymperopoulos, and C. N. Jones, "Optimization of an airborne wind energy system using constrained gaussian processes," in *IEEE Multi-Conference on Systems and Control (to appear)*, Antibes / Nice, France, Oct. 2014.

[6] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA: MIT Press, 2006.

[7] D. Jones, "A taxonomy of global optimization methods based on response surfaces," *Journal of global optimization*, vol. 21, no. 4, pp. 345–383, 2001.

[8] E. Brochu, V. M. Cora, and N. de Freitas, "A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning," arXiv.org, eprint arXiv:1012.2599, December 2010.

[9] M. Schonlau, W. J. Welch, and D. R. Jones, "Global versus local search in constrained optimization of computer models," in *New Developments and Applications in Experimental Design: Selected Proceedings of a 1997 Joint AMS-IMS-SIAM Summer Conference*, vol. 34, 1998, p. 11.

[10] B. J. Williams, T. J. Santner, W. I. Notz, and J. S. Lehman, "Sequential design of computer experiments for constrained optimization," in *Statistical Modelling and Regression Structures*, T. Kneib and L. Fahrmeir, Eds. Springer - Verlag, 2010, pp. 449–472.

[11] R. B. Gramacy, G. A. Gray, S. L. Digabel, H. K. Lee, P. Ranjan, G. Wells, and S. M. Wild, "Modeling an augmented lagrangian for improved blackbox constrained optimization," Sandia National Laboratories, Livermore, CA, Tech. Rep. arXiv:1403.4890, Mar 2014.

[12] M. Erhard and H. Strauch, "Control of towing kites for seagoing vessels," *IEEE Trans. Contr. Syst. Technol.*, vol. 21, pp. 1629–1640, Nov. 2012.

[13] ——, "Sensors and navigation algorithms for flight control of tethered kites," in *Control Conference (ECC), 2013 European*. IEEE, 2013, pp. 998–1003.

[14] C. E. Rasmussen, "Gaussian processes in machine learning," in *Advanced Lectures on Machine Learning*. Springer, 2004, pp. 63–71.

[15] D. Jones, M. Schonlau, and W. J. Welch, "Efficient global optimization of expensive black-box functions," *Journal of Global optimization*, vol. 13, no. 4, pp. 455–492, 1998.

[16] A. Reid, S. O'Callaghan, and L. McCalman. TacoPig: Gaussian Process Matlab Toolbox. National ICT Australia (NICTA). [Online]. Available: https://github.com/NICTA/TacoPig

[17] M. A. Patterson and A. V. Rao, "GPOPS- II: A matlab software for solving multiple-phase optimal control problems using hp–adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Transactions on Mathematical Software*, vol. 39, no. 3, 2013.