Optimization of an Airborne Wind Energy System Using Constrained Gaussian Processes

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

***Abstract*— Wind resources tend to be significantly stronger and more consistent with increasing altitude. This effect creates a potential for power generation that can be reaped by an Airborne Wind Energy system positioned at elevations exceeding the height of conventional wind turbines. A frequent design for such a system includes a flying airfoil tethered to a ground station. The station can be equipped with a power generator or for the application considered here mounted to a sea vessel. We demonstrate a data based method that can maximize the towing force of such a system by optimizing a low level tracking controller at the presence of constraints. We utilise Gaussian Processes to learn the mapping from the set points of the controller to both the objective and the constraint function. We then formulate a chance - constrained optimization problem that takes into consideration uncertainty in the learned functions. The probabilistic objective function is transformed into a deterministic acquisition function which indicates set points with high probability of improving the current optimum and the constraint function is penalized in regions of high uncertainty to ensure feasibility. Simulation studies show that we can find optimal set points for the controller without the use of significant assumptions on model dynamics while respecting the unknown constraint function.**

I. INTRODUCTION

Friction between the atmosphere and the surface of the earth produces a boundary layer effect which is responsible for the low and intermittent winds close to the ground in contrast to the faster and more persistent ones above a certain altitude. The relationship that governs the wind variation at these low heights can be approximated by a power law and mainly depends on the roughness of the ground [1]. Moreover, the power that can be potentially extracted by the wind is proportional to the cube of wind speed [2]. For this reason, among others, modern wind turbines are increasing in height while also wind height variation is taken into account both for the design and operation of wind farms. However, structural and economic limitations of scaling a support mast imply that the higher and more prevalent winds will probably remain outside the reach of conventional wind turbines.

A rising interest both from research and commercial ventures for extracting wind power from higher altitudes has lead to the introduction of new designs and prototypes for Airborne Wind Energy (AWE) systems [3], [4], [5]. Most of the designs include a tethered flying airfoil and are divided between those with airborne generators that transmit electricity to the ground via a power cable and those that exert mechanical line forces which can be used at the ground station. We focus here on a variation of the second design, where the tethered airfoil (quite similar to a kite) exerts a towing force to a unit mounted on a sea vessel. This system allows large ships to take advantage of high altitude winds in order to achieve considerable fuel savings and has been commercialised by the company Skysails [6]. Skysails has developed low level controllers that exhibit stable performance at a variety of different wind conditions and are experimentally validated [7]. However, such a controller cannot automatically guarantee a power optimal trajectory nor that the airfoil will not cross an altitude safety threshold.

We are addressing this problem in the general framework of constrained optimization

$$\begin{aligned} \max_{x} \quad & f(x) \\ \text{s.t.} \quad & g(x) \quad \leq \quad 0 \end{aligned} \tag{1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ and $g : \mathbb{R}^d \to \mathbb{R}^m$ are the objective and constraint functions of a multivariable decision input $x \in \mathbb{R}^d$ (here the set-points of the low level controller). The functions can be nonlinear and/or non-convex. We are more specifically trying to learn the set points that optimize the towing force, while respecting an altitude constraint. The fact that AWE systems of this type operate best at crosswind conditions imply that the constraint will be frequently active for the lower part of the kite trajectory.

Both functions are considered unknown and we are actively learning them using measurements under the framework of Gaussian Processes (GP), a methodology mainly developed and used in Machine Learning [8]. GPs can be used for regression with relatively few assumptions on the form of the unknown function and have been extensively utilised for unconstrained optimization [9], [10].

More challenging situations arise in the presence of non-trivial constraints that are also being learned from data. A first approach to this problem but with known constraints was given in [11] by simply avoiding sampling inside the infeasible regions. Another approach assumes that the learned functions are dependent and requires assumptions on their coupling [12], while [13] allows for sampling in infeasible regions if this is deemed sufficiently informative. Finally, the same authors in [14] use an augmented Lagrangian approach to account for the constraints.

In our version we follow a generic approach where two independent GPs are trained for both functions and an acquisition function based on Expected Improvement (EI)

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

denotes the next point to sample. We do not explicitly allow for function evaluations outside the feasible part of the domain, while we also restrict the augmentation of our search space by penalising learned inputs that exhibit high output variance. Simulation results show that this strategy can optimize the towing force generated without violating the altitude constraints. The result achieved is similar to the performance of a numerical optimal control software (GPOPS - II) [15] where control action is available for manipulation throughout the trajectory.

In Section II we give an introduction to GPs for regression, optimization and constraint handling. Section III provides an overview of the AWE system and the low level controller that is used for tracking. Section IV presents our simulation setup and results, while in Section V we provide a conclusion and discuss future directions.

II. GAUSSIAN PROCESSES FOR CONSTRAINED OPTIMIZATION

Gaussian processes are used in supervised learning for regression or classification from a Bayesian perspective. They are an extension of the multivariate Gaussian distribution to infinite dimensional stochastic processes where any finite combination of outputs is jointly Gaussian. In this sense a GP describes a distribution in the function space. Making predictions in unsampled locations, incorporating new measurements and quantifying uncertainties is straightforward due to the Gaussianity assumptions. In addition, GPs allow for more flexibility in the structure of the unknown function compared to generic parametric models [16].

A. Gaussian Processes - Regression

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

$$y_{1:N} \sim \mathcal{N}(m_{1:N}, K(x_{1:N}, x_{1:N})) \tag{2}$$

where $y_{1:N} = f(x_{1:N})$ and $m_{1:N}$ is also compact notation for $m(x_{1:N})$. The covariance matrix K is calculated elementwise using the kernel $k(.,.)$. For any unobserved point (or collection of points) x^*, we can analytically predict the output of the learned function $y^* = f(x^*)$ as a conditional distribution

$$y^*|\mathcal{D}, x^* \sim \mathcal{N}(\mu(x^*|\mathcal{D}), \sigma(x^*|\mathcal{D})), \tag{3}$$

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 $K_{1:N} = K(x_{1:N}, x_{1:N})$. For most practical cases the prior mean is either assumed zero or is fixed to a constant. It is interesting to note that the covariance does not depend on the mean nor on the function output but just on the location of the training and sample points. The crucial ingredient is the kernel, the element that encodes our assumptions on the learned function. We use here an ARD (Automatic Relevance Determination) squared exponential (SE) kernel that is 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 $\Lambda = diag(\lambda_1, ..., \lambda_d)$, $\delta_{ii'}$ is Kroenecker's delta ($= 1$ iff $i = i'$, 0 otherwise) and the parameters $\theta = \{\sigma_y, \lambda_{1:d}, \sigma_n\}$ (usually called hyperparameters in this context) have to be learned from $\mathcal{D}$. The kernel here includes measurement uncertainty (σ_n). To select appropriate hyperparameters θ, we have to find those that maximize the log marginal likelihood for the observed data

$$\log p(y_{1:N}|x_{1:N}, \theta) = -\frac{1}{2}\log K_{1:N} - \frac{1}{2}y_{1:N}^T K_{1:N}^{-1} y_{1:N} - N/2 \log 2\pi,$$

where $K_{1:N}$ is a function of θ. For many cases this function is nonconvex and presents an optimization challenge. A more accurate solution from the Bayesian perspective, but computationally expensive, would be Monte Carlo sampling in the hyperparameter space with probability proportional to the likelihood and then averaging over the outputs over the different θ values sampled. However, here, we follow the simpler first approach. Once the selection of hyperparameters is finalized the learning phase is over. We can now predict the output at any point of interest using (3).

B. Gaussian Processes - Optimization

To optimize over the learned function we could search for the optimum of the mean of the GP and sample there. However, this method is likely to get trapped into suboptimal solutions. We follow a methodology where an auxiliary acquisition function is used to indicate the next sampling point [17]. This function is called Expected Improvement (EI) and measures the expectation of the magnitude for improvement over the current best sampled value $y_{max} = \max y_{1:N+K}$ (where N are the initial training points and K the additional points that have been sampled). This function can be analytically derived for any given point x in the search space as

$$EI(x) = \sigma(x)[u\Phi(u) + \phi(u)], \tag{4}$$

where $u = (y(x) - y_{max})/\sigma(x)$, $\sigma(x)$ 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/or high mean and this way can account for both exploration and exploitation.

C. Chance Constrained Optimization

Since both our constraints and objective are unknown functions, approximated by GP_g and GP_f, we should handle them appropriately. This means that we have to restate our initial problem (1) in a chance constrained optimization formulation

$$\begin{aligned} \max_x \quad & EI_f(x) \\ \text{s.t.} \quad & P(GP_g(x) \leq 0) \geq 1 - \beta \\ & \sigma_g(x) \leq \alpha\sigma_n, \end{aligned} \tag{5}$$

where we replace the learned objective function with the respective EI and the constraint function with an auxiliary statement requiring that the probability of satisfying the constraint at a sample point x is larger than $(1-\beta)$. By setting β appropriately we can adjust the conservativeness of our approach (common design values are $0.01-0.1$). Once more, due to the use of GPs we can easily derive this probability at any given point x in the search domain

$$P(GP_g(x) \leq 0) = \Phi_g(x), \tag{6}$$

where Φ_g is the cumulative distribution function for a Gaussian distribution at point $x \in \mathbb{R}^d$, with mean $m_g(x)$ and variance $\sigma_g(x)$, as provided by GP_g.

The second constraint reduces the aggressiveness of our approach to account for the lack of information in the unexplored parts of the learned function. We have experienced that especially in the beginning of the learning process a bad set of hyperparameters might lead to unrealistic anticipations of GP_g in the unsampled parts of the domain. This way large violations might occur (even with very low β, since the GP is essentially wrong on its assumptions). We wish here to eliminate such occurrences by excluding points with high variance, even if their mean implies that they are safe. We observe that this reduces slightly the convergence rate but almost guarantees that no significant violations will occur (within measurement accuracy). The full procedure for learning and optimization can been seen in Algorithm 1. We have to note here that the constraints will not be sufficient to analytically describe the feasible domain which might be non-convex. For optimization over EI_f in the feasible domain we use either MC sampling or SQP.

Algorithm 1 Constrained Optimization with GPs

1: **Initial Samples.** Select N points in the feasible set and insert them in $\mathcal{D}$
2: **Learning.** Train GP_g and GP_f using $\mathcal{D}$
3: **Optimization.** Find f_{max} among the sampled points
4: Calculate EI_f using f_{max} for the objective function
5: Calculate the Chance and Variance Constraints on GP_g
6: **New Data.** Sample in the feasible domain at the point $\tilde{x}$ with the highest EI_f
7: **if** $f(\tilde{x}) \geq f_{max}$ and max $EI_f \leq$ tolerance **then**
8: **Terminate** and use $x^* = \tilde{x}$
9: **else**
10: Add $(\tilde{x}, f(\tilde{x}))$ in $\mathcal{D}$ and Go To 2
11: **end if**

III. CONTROLLER DESIGN

A. System Dynamics Description

Skysails has developed simple but robust low level controllers for tracking "figure 8" loops for large scale autonomous kites used in their naval applications. The results have been experimentally validated and are acceptable for a point mass model of a kite flying with fixed tether length. More details can be found in [7], [18].

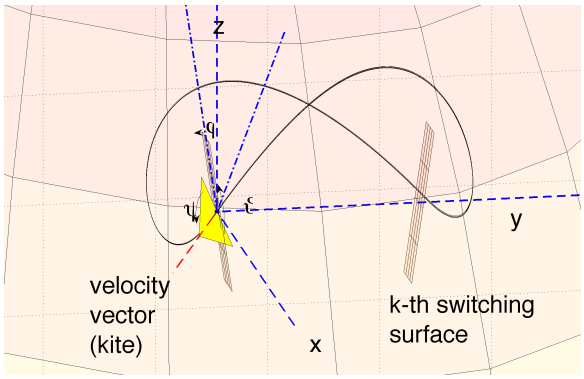


Fig. 1: Coordinates and switching surfaces for the Skysails controller

In order to develop their controller Skysails assumes dynamics where mass is neglected (a reasonable assumption for the large aerodynamic forces created) and where the flexible tether is substituted by a rigid rod (also acceptable in high dynamic regions where the tether is constantly in tension). The equations of motion are

$$\dot{\vartheta} = \frac{u_0 E}{L}\cos\vartheta\cos\psi - \frac{u_0}{L}\sin\vartheta \tag{7}$$

$$\dot{\varphi} = -\frac{u_0 E}{L}\frac{\cos\vartheta}{\sin\vartheta}\sin\psi \tag{8}$$

$$\dot{\psi} = u_0 E g\cos(\vartheta)\delta + \dot{\varphi}\cos\vartheta \tag{9}$$

The three states of the system represent the spherical coordinates (ϑ, φ) and orientation (ψ) of the airfoil. The rest are constants with E: glide ratio, g: deflection coefficient, u_0: wind speed, L: tether length. Finally, δ is the deflection applied to the kite directly affecting its orientation. We use these dynamics to check against our model-free controller. Please note that E and L are also potential control variables for the kite system. We hope to extend the current formulation to include these in the future.

The system exhibits nonlinear behavior even without considering the more complex effects of aerodynamics that have been significantly approximated. It is thus difficult to find closed form solutions for the response surfaces of interest like the average tether tension or the minimum altitude generated in a periodic loop. Even with numerical optimization the results obtained would not be necessarily valid in the actual setting where the model mismatch and environmental conditions are unknown.

B. Skysails Control

The controller employed is a model free feedback control scheme. A low level controller tracks a given set point on the yaw angle by applying appropriate deflections on the kite. On a higher level the set point is provided based on the position of the kite in its loop and certain predefined switching positions and their corresponding set point values. The scheme is shown below in Figure 2.

When the kite crosses a particular section the set point is changed to a predefined level such that the kite flies a stable

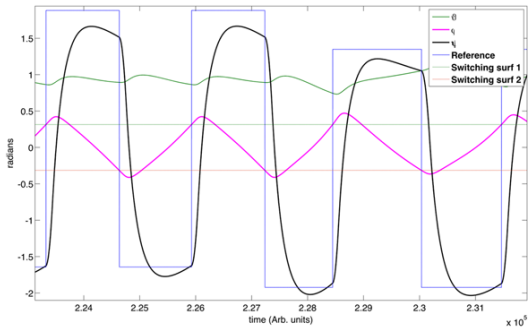


Fig. 2: Behavior of the Skysails control for different set points

loop. While originally the controller uses only two switching surfaces it is not difficult to extend this to a higher dimension, granting a higher degree of control.

The kite is said to cross the i^{th} switching surface when:

$$\phi - \phi_i = 0 \;\&\; k\dot{\phi} > 0; \quad k = 1 \text{ or } -1 \tag{10}$$

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, thus defining its orientation. Each surface considers crossings only in one direction and ignores crossings in the other direction. Moreover, when the kite crosses the i^{th} section, the reference signal to the low level tracking controller is changed to Ψ^i_{ref}. The reference Ψ^i_{ref} is then tracked by a combined feedforward and feedback loop.

IV. RESULTS

We implement our algorithm in Matlab and the Skysails controller in Simulink. For the GP regression we have used the object-oriented toolbox TacoPig [19].

We start our simulations using N=7 or 15 initial training points in the feasible set. This is usually possible from prior experience of the operator. Each pair of set points is used for 10 loops, until the kite reaches a stationary trajectory. Then a measurement is taken (we add noise to emulate sensor errors) both for the power (or tension) produced and the lowest altitude of the loop. Figure 5 shows the evolution of the GPs and the sampling points as Algorithm 1 progresses. The algorithm on average shows quick convergence to the optimum (within 20-30 samples) even with varying initial training sets and decision space (varying number of switching surfaces). Figure 4 shows the measured objective values and the best objective found by the algorithm for different numbers of switching surfaces. This result is summarized in Table I and the algorithm is compared to the optimum calculated using the numerical optimal control solver GPOPS-II, which uses the model and has full control throughout the trajectory and not just at the 2 or 4 switching surfaces.

The true response surfaces were estimated in simulation by taking random i.i.d samples throughout the input space without constraints. Figure 3 shows the response surface of power for reference.

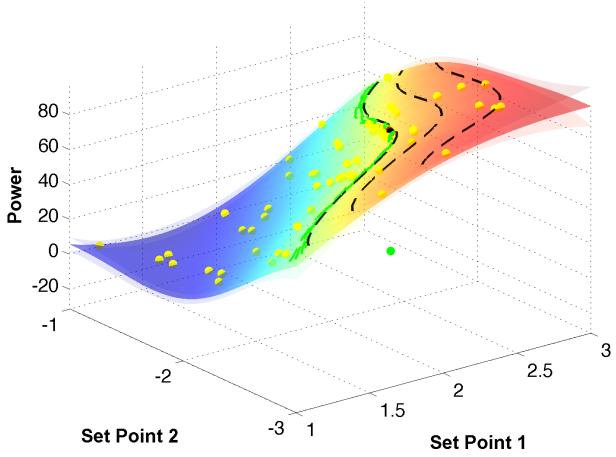


Fig. 3: GP for the power function computed at a large number of sample points. Yellow circles represent measurement points. Green lines represent the chance constraints for different β and black dashed lines the actual altitude constraint (for a different altitude threshold, here 200, 150 and 100 m).

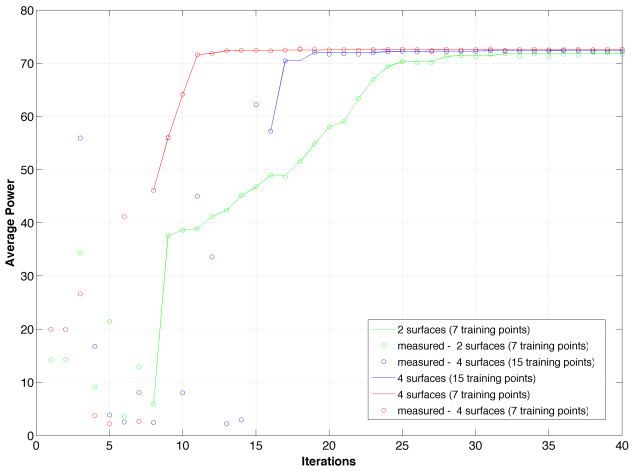
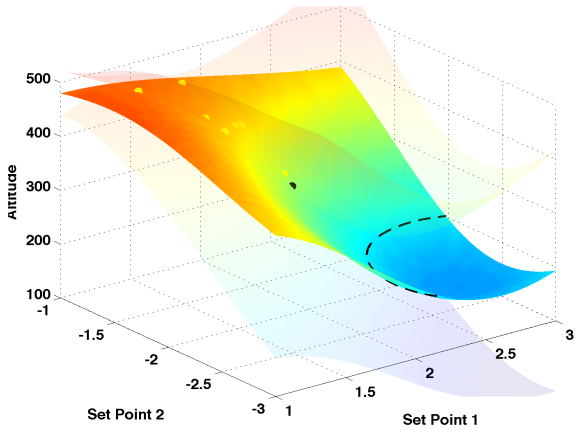


Fig. 4: Convergence of the algorithm for different number of switching surfaces and initial training points.

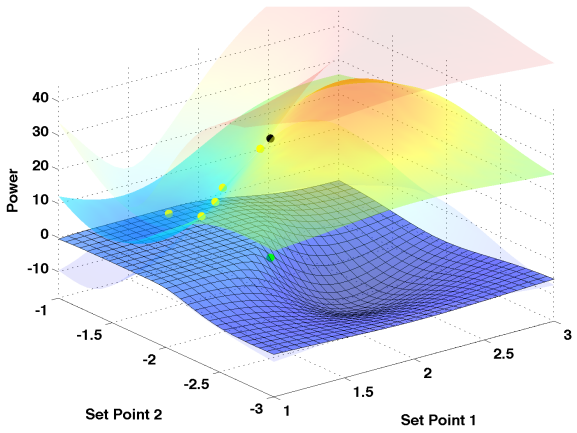
Algorithm 1 can be used in varying wind conditions while also satisfying altitude constraints without any substantial modifications. A sinusoidal wind disturbance with a 200 second time period and 3 m/s amplitude was added to the base wind speed of 10 m/s and the response surfaces were obtained as before with an additional input measurement of wind. Figures 6 and 7 show that the algorithm adapts the set points as the wind changes between 7 m/s to 13 m/s periodically to remain close to the optimum power with only marginal violation of the constraints.

V. CONCLUSIONS

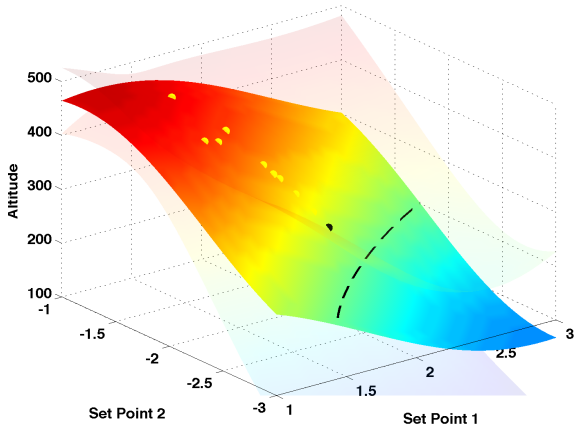
We present here an algorithm for optimizing the towing force produced by an AWE system while also respecting alti-



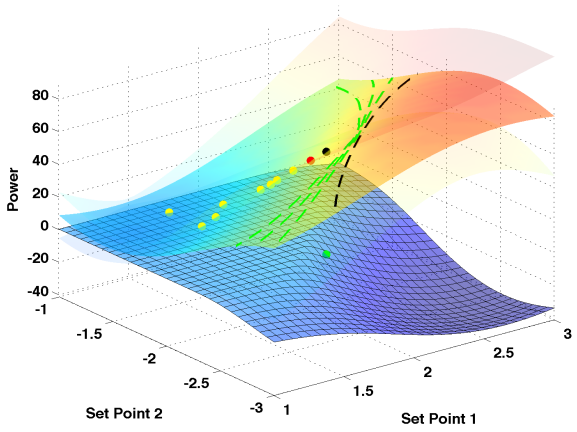
(a) GP_g with initial training points



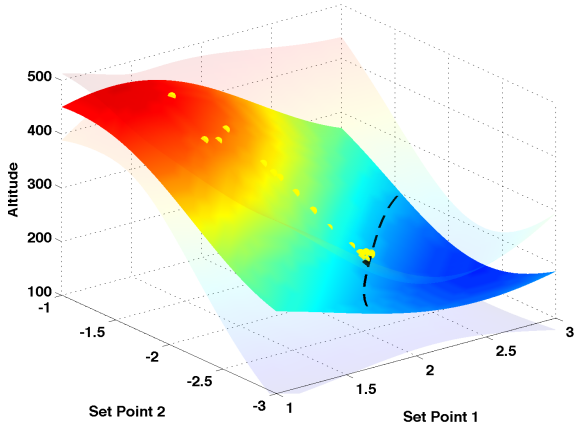
(b) GP_f with initial training points



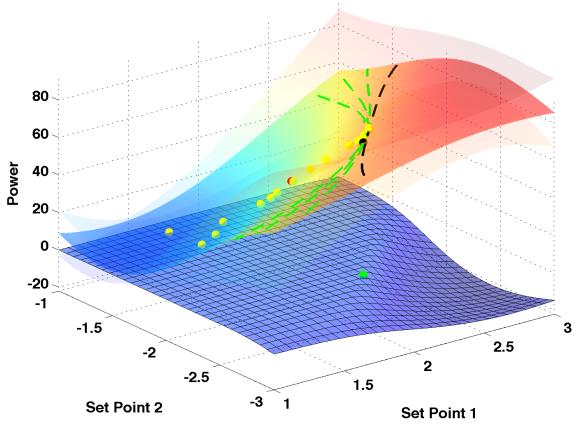
(c) GP_g after incorporating 3 sample points



(d) GP_f after incorporating 3 sample points



(e) GP_g after convergence to the optimal solution



(f) GP_f after convergence to the optimal solution

Fig. 5: Evolution of the GP learning and optimization procedure at the initial (a,b) intermediate (c,d) and final stage (e,f). The figures to the left concern the learning process for the altitude constraint function and the figures to the right for the power function. Yellow circles represent measurements, a black circle indicates the next sampling point and a green circle the projection of this point on the EI surface. The opaque coloured surface is the mean of the learned GP and the transparent ones represent a variation of 2σ around it. The black dashed lines represent the actual altitude constraint (200 m), and the green lines the chance constraints for different β. The blue gridded surface is the negative EI for visualisation reasons, so here we look for its minimum.

TABLE I: Performance comparisons

Decision space	Final value	Max Violation (m)	Iterations
2 surfaces	71.80	3.23	32
4 surfaces	72.60	4.77	30
GPOPS-II	74.61	0.00	-

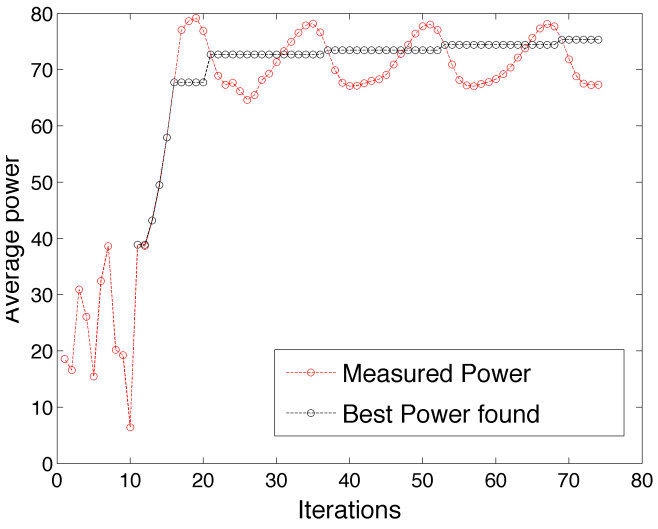


Fig. 6: Maximum Power tracking under varying wind conditions. The red circles represent the current set point selected and the black ones the best found. When the wind changes the set point follows.

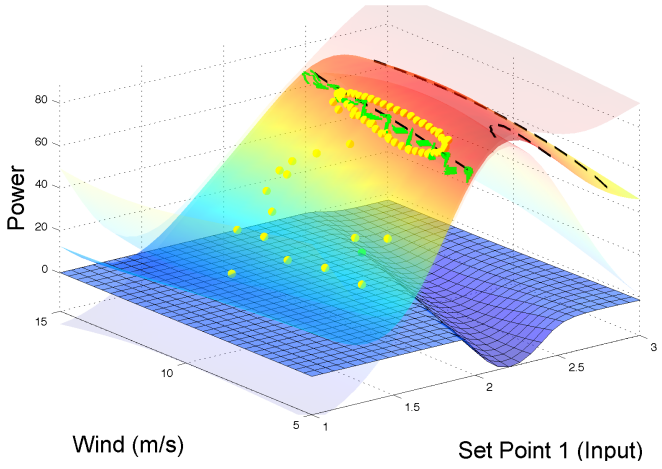


Fig. 7: GP learned for different wind conditions and a common set point for the two switching surfaces. Slight violations might occur since for different wind conditions the last set point does not satisfy the constraints temporarily

tude constraints. We base our method on Gaussian Processes and we assume no prior knowledge of the model dynamics. The algorithm utilises an underlying model-free controller for which it learns adaptively the appropriate set points. The results are comparable to the ones obtained by a numerical optimal control solver (which has knowledge of the model and full input control). Finally, the algorithm is able to adapt into varying wind conditions with no further modifications.

For future work we plan in extending our method in higher dimensions and more control inputs (such as variable tether length and glide ratio).

ACKNOWLEDGMENT

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

REFERENCES

[1] M. Clobes, A. Willecke, and U. Peil, "Classification of measured wind profiles using neural networks," in *The Fifth International Symposium on Computational Wind Engineering (CWE2010)*, Chapel Hill, North Carolina, USA, May 2010.

[2] 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.

[3] M. L. Loyd, "Crosswind kite power (for large-scale wind power production)," *Journal of Energy*, vol. 4, no. 3, pp. 106–111, 1980.

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

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

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

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

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

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

[10] 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.

[11] 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.

[12] 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.

[13] R. B. Gramacy and H. K. H. Lee, "Optimization under unknown constraints," in *Bayesian Statistics 9*. Oxford University Press, 2010, pp. 229–256.

[14] 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.

[15] 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.

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

[17] 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.

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

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