Recursive Gaussian Process-Based Adaptive Control, With Application to a Lighter-Than-Air Wind Energy System

Joe Deese and Chris Vermillion, *Member, IEEE*

***Abstract*— This brief presents a nonmodel-based adaptive control technique that combines principles from machine learning and iterative design optimization with those of continuous-time, falsification-based adaptive control. At the crux of the proposed control strategy are two core elements. First, the recursive Gaussian Process (RGP) modeling is used to maintain an online characterization of the system at hand without the need to maintain a complete database of previously collected measurements (which is required in traditional GP modeling). Second, an adaptation strategy is employed that *falsifies* candidate controllers from a continuous candidate design space based on desired performance specifications and statistical hypothesis testing. In specific, the control parameter design space is explored by selecting points associated with high uncertainty. Through the use of statistical hypothesis testing, regions of the design space determined to be suboptimal at a user-specified level of confidence are rejected in order to converge to an optimal set of control parameters. The RGP-based adaptation is validated through simulations and laboratory-scale experiments using an airborne wind energy case study. Through these studies, the RGP-based adaptation approach is shown to be effective and is shown to exhibit favorable convergence times when compared with a mature adaptive control technique, extremum seeking (ES).**

***Index Terms*— Adaptive control, Gaussian processes (GPs), machine learning, optimization, wind energy.**

I. Introduction

ADAPTIVE control strategies exist for the purpose of controlling systems with uncertainties and variations in operating conditions/parameters (see [1]). However, many classic adaptive control strategies, such as indirect methods, that rely on gradient-based/recursive least-squares parameter estimations, model reference adaptive control (see [1]–[4]), or nonlinear backstepping techniques (see [5], [6]), depend on significant structural knowledge and assumptions of the system to be controlled. While the aforementioned techniques come with significant performance guarantees, their underlying assumptions limit their application to complex systems.

As the complexity of engineering systems increases, our ability to accurately describe the system dynamics by a small, finite set of equations (and corresponding parameters) diminishes. Many real-world engineering systems rely on complex simulation models or require experimental evaluation. Therefore, adaptive control strategies that do not require significant knowledge of the system are of great interest. One such strategy in the control literature is extremum seeking (ES) [7]. ES is a nonmodel-based adaptive control technique that relies on a periodic perturbation and system performance feedback to converge to the optimal control parameter(s). However, due to a required timescale separation between the plant time constants, filter time constants, and perturbation period, the technique can suffer from slow convergence times.

One candidate approach to adaptive control, adopted from the machine learning community, involves maintaining a Gaussian process (GP) model of the system at hand, then using that GP model to drive a parameter exploration law. GP modeling arose as an iterative technique for developing a model of uncertain systems by treating the system as a black-box function [8]. While traditional GP modeling requires each new measurement to be appended to an ever-growing database of all past data, this challenge can be circumvented through a recursive GP (RGP) modeling technique that updates the statistics of the GP model as observations from the system are collected (see [9] [10]). Since the RGP modeling strategy was developed for iterative applications, two key challenges still exist for *real-time* adaptive control.

1) RGP modeling characterizes an input–output map but does not capture the temporal correlations between successive measurements that result from *system dynamics*.
2) The RGP model alone does not define a strategy for *exploring* the controller parameter space and *identifying* the optimal parameter(s) in the design space.

The proposed RGP-based adaptation, seen in Fig. 1, is a unique, nonmodel-based adaptive control technique that utilizes RGP modeling in *real time*. To do so, the RGP modeling technique from [9] is extended by using Kalman filtering techniques from [11] to account for the temporal correlation between points collected in *rapid succession*. The resulting real-time RGP model is used to explore the candidate design space by visiting areas of high uncertainty.

Airborne wind energy (AWE) systems, used as a case study here, replace the conventional tower of a wind energy system with tethers and a lifting body (e.g., a wing or high-lift aerostat), enabling the systems to harness energy from high-altitude winds. At higher altitudes, winds have been shown to be stronger and more consistent than ground-level winds (see [12]). AWE systems are an interesting case study because the system dynamics are highly nonlinear, and an analytical, closed-form expression of the performance index

Manuscript received June 3, 2020; accepted July 23, 2020. Date of publication August 18, 2020; date of current version June 10, 2021. Manuscript received in final form August 1, 2020. This work was supported by NSF (CAREER: Efficient Experimental Optimization for High Performance Airborne Wind Energy Systems) under Grant 1453912. Recommended by Associate Editor L. Fagiano. *(Corresponding author: Joe Deese.)*

Joe Deese is with the Department of Mechanical Engineering, University of North Carolina at Charlotte, Charlotte, NC 28223 USA (e-mail: jdeese23@uncc.edu).

Chris Vermillion is with the Department of Mechanical and Aerospace Engineering, North Carolina State University, Raleigh, NC 27695 USA, also with Altaeros, Inc., Somerville, MA 02143 USA, and also with Windlift, LLC., Morrisville, NC 27560 USA (e-mail: cvermil@ncsu.edu).

Color versions of one or more of the figures in this article are available online at https://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCST.2020.3014159

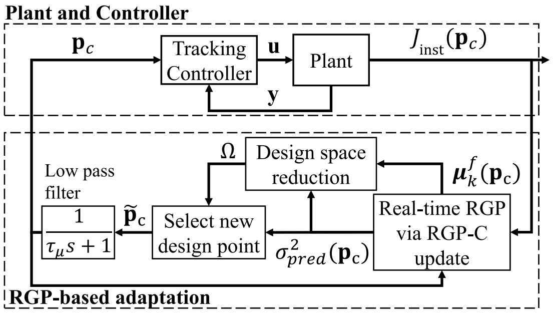

Fig. 1. General process diagram of the RGP-based adaptive control law.

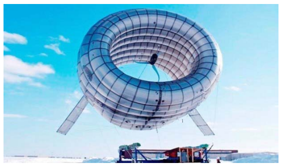

Fig. 2. BAT of Altaeros, Inc., [16].

as a function of the controller variables does not exist. Using both simulation-based and experimental studies of the Buoyant Airborne Turbine (BAT) of Altaeros, Inc., Somerville, MA, USA (shown in Fig. 2), we consider control parameter optimization for crosswind flight. Efficient crosswind flight increases the power output of AWE systems (see [13]–[15]). In this work, the control parameters that describe the geometric characteristics of the crosswind flight path are optimized using the RGP-based adaptation.

This work extends our work in [17] by comparing our novel RGP-based adaptation to ES. Since ES is one of the most well-known and mature nonmodel-based adaptive control techniques in the literature, having been applied successfully to a number of applications (see [18]–[20]), it provides a key benchmark against which to compare the efficiency of the RGP-based adaptation. From the comparative analysis, it is shown that the RGP-based adaptation has faster convergence times than ES without detriment to performance, in addition to being effective at finding boundary optima and globally exploring the control parameter space.

II. Overall Framework and Notation

The ultimate goal of the proposed adaptation is to converge upon an optimal control parameter vector, $\tilde{\mathbf{p}}_c$, for which

$$\begin{aligned}\tilde{\mathbf{p}}_c &= \arg\max_{\mathbf{p}_c\in\Omega} J(\mathbf{p}_c) = \int_0^{t_f} J_{\text{inst}}(\boldsymbol{\zeta}(t),\mathbf{u}(t);\mathbf{p}_c)dt\\ &\text{given: } \dot{\boldsymbol{\zeta}} = g(\boldsymbol{\zeta},\mathbf{u}).\end{aligned} \tag{1}$$

Here, J_{inst} is an instantaneous performance index that captures desired performance specifications of the dynamic system described by $\dot{\boldsymbol{\zeta}} = g(\boldsymbol{\zeta},\mathbf{u})$ at a given time step.

In this brief, the RGP-based adaptation is used to identify $\tilde{\mathbf{p}}_c$. The RGP-based controller adaptation presented here extends tools available in the GP modeling literature to recursively update predictive mean and variance estimates in *real time*. Our novel RGP-based adaptation tackles two features of traditional GP modeling that are (in their original form) ill-suited for online adaptive control.

1) Traditional GP modeling approaches rely on a database of all previously collected observations, which becomes computationally burdensome for an online adaptation.
2) GP and RGP modeling approaches assume successive observations of the performance to be uncorrelated, but *real-time* applications exhibit dynamics, leading to the temporal correlation between successive observations.

Fig. 1 shows the manner by which RGP-based modeling and subsequent statistically driven design space reduction are used to reduce the control design space and adjust the control parameter vector. The mean function estimate ($\boldsymbol{\mu}_k^f(\mathbf{p}_c)$) from the real-time RGP update is used to approximate the relationship between the controller design parameter(s) ($\mathbf{p}_c$) and instantaneous performance (J_{inst}). Prediction variance ($\sigma^2_{\text{pred}}(\mathbf{p}_c)$) quantifies uncertainty within the controller design space ($\boldsymbol{\Omega}$). The exploration strategy of the RGP-based adaptation in this work selects values of $\mathbf{p}_c$ for which uncertainty in $\boldsymbol{\mu}_k^f(\mathbf{p}_c)$ is high. As points are visited, the uncertainty is reduced. To converge to an optimal $\mathbf{p}_c$, the design space must be reduced, which is achieved based on statistical confidence intervals.

III. Real-Time Recursive Gaussian Process Model

GP modeling is a nonlinear regression technique used to approximate the relationship between an input vector and unknown function (typically called a latent function) based on observed data [8]. The value of the unknown function at a generic input vector location ($f(\mathbf{p}_{c,i})$) is assumed to be a random variable. A GP model is fully defined by a mean ($\mu(\mathbf{p}_c)$) and covariance function ($\varphi(\mathbf{p}_c, \mathbf{p}_c')$)

$$F(\mathbf{p}_c) \sim \text{GP}\big(\mu(\mathbf{p}_c), \varphi(\mathbf{p}_c, \mathbf{p}_c')\big)$$

where

$$\begin{aligned}\mu(\mathbf{p}_c) &= \mathbb{E}[f(\mathbf{p}_c)]\\ \varphi(\mathbf{p}_c, \mathbf{p}_c') &= \mathbb{E}\big[(f(\mathbf{p}_c) - \mu(\mathbf{p}_c))\big(f(\mathbf{p}_c') - \mu(\mathbf{p}_c')\big)\big].\end{aligned} \tag{2}$$

The mean function represents the expected value of the latent function, while the covariance function characterizes how two observations change together. In [8], a number of different covariance functions are provided. Here, we use the squared exponential covariance function.

To generate the predictive distribution, one requirement is to maintain a *complete* database of all of the previously tested points. The database will be denoted by $\mathbb{D} = \{\mathbf{X}_{\mathbb{D}} = \mathbf{p}_c^{1:k-1}, \mathbf{Y}_{\mathbb{D}} = J_{\text{inst}}^{1:k-1}(\mathbf{p}_c)\}$. At each sampling instant, k, the mean function estimate and prediction variance for any candidate point, $\mathbf{p}_c^k$, are calculated by

$$\mu_f(\mathbf{p}_c^k) = \mathbb{E}\big[f(\mathbf{p}_c^k)\big] = \varphi(\mathbf{p}_c^k, \mathbf{X}_{\mathbb{D}})K_{\mathbf{p}}^{-1}\mathbf{Y}_{\mathbb{D}} \tag{3}$$

$$\sigma_f^2(\mathbf{p}_c^k) = \varphi(\mathbf{p}_c^k, \mathbf{p}_c^k) - \varphi(\mathbf{p}_c^k, \mathbf{X}_{\mathbb{D}})K_{\mathbf{p}}^{-1}\varphi(\mathbf{X}_{\mathbb{D}}, \mathbf{p}_c^k) \tag{4}$$

respectively. The matrix $K_{\mathbf{p}}$ represents the covariance between all inputs corresponding to the collected data plus measurement noise (which is assumed to be white), i.e., $K_{\mathbf{p}} = \varphi(\mathbf{X}_{\mathbb{D}}, \mathbf{X}_{\mathbb{D}}) + \sigma^2_{\text{noise}} I$. Clearly, as more observations are added to the database, the matrix K_p grows in size. If a GP model is implemented in a real-time environment with subsecond time steps (i.e., multiple observations are collected every second), calculating the inverse of K_p becomes intractable in a relatively short amount of time. Therefore, applying GP modeling directly to adaptive control is generally not feasible.

The RGP modeling strategy detailed in [9] and [10] updates the mean function and covariance matrix sequentially as new observations are collected, avoiding the need for a database. A finite number of basis vectors are used to discretize a continuous design space. The collection of basis vectors that span a given design space will be denoted by the matrix $\mathbf{X} = [\bar{\mathbf{p}}_{c,1}\ \bar{\mathbf{p}}_{c,2}\ \ldots\ \bar{\mathbf{p}}_{c,s}]^T$. Each element of this matrix, $\bar{\mathbf{p}}_{c,i}$, corresponds to an individual design in the candidate design space. If the candidate design space is 1-D, the basis vectors will correspond to a simple grid of the design space (i.e., $\bar{\mathbf{p}}_{c,i} \in \mathbb{R}^{1\times 1}\ \forall i \in [1, s]$), but higher order design spaces may require more efficient basis vectors. The mean function estimate at each basis vector location is treated as a state within a state–space representation in [9]. To maintain a consistent computation time as time evolves, a Kalman filter is used in [9] to recursively update the GP model as data is collected.

The RGP update described in [9] and [10] assumes each successive observation is temporally uncorrelated with the current observation. In real-time control applications, however, the instantaneous performance index of the system depends not only on the instantaneous choice of control parameter ($\mathbf{p}_c$), but also upon the system's state ($\boldsymbol{\zeta}$) at that time. Because of this, two measurements taken in rapid succession will be correlated. Thus, to use an RGP algorithm effectively for adaptive control, temporal correlation must be taken into account. To do this, the RGP model is extended to consider colored noise in the observations by introducing a state model that characterizes the temporal correlation in the noise, resulting in the following set of dynamic equations:

$$\begin{bmatrix} \dot{\boldsymbol{\xi}} \\ \dot{\eta} \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & \dfrac{-1}{\tau} \end{bmatrix} \begin{bmatrix} \boldsymbol{\xi} \\ \eta \end{bmatrix} + \begin{bmatrix} 0 \\ \dfrac{1}{\tau} \end{bmatrix} w$$

$$y = [\mathbf{J}_k \quad 1] \begin{bmatrix} \boldsymbol{\xi} \\ \eta \end{bmatrix}. \tag{5}$$

Here, $\boldsymbol{\xi}$ represents the mean function value at the basis vector locations, η represents the colored noise state, τ represents the colored noise time constant (providing a measure of temporal correlation), w is the Gaussian white noise with zero mean, and $\mathbf{J}_k$ is given by $\mathbf{J}_k = \varphi(\mathbf{p}_c, \mathbf{X})(\varphi(\mathbf{X}, \mathbf{X}) + \epsilon I)^{-1}$. The vector $\mathbf{J}_k$ can be interpreted as a vector used for interpolating between basis vector locations. To avoid singular matrix inversion, a small number, ϵ, is added to the diagonal elements of $\varphi(\mathbf{X}, \mathbf{X})$.

Using the above-mentioned dynamic model formulation, colored noise can be considered within a Kalman filter update using the measurement differencing approach detailed in [11]. In [11], the system dynamics are discretized and reformulated as

$$\mathbf{x}_k = F_{k-1}\mathbf{x}_{k-1} + w_{k-1} \tag{6}$$

$$y_k = H_k\mathbf{x}_k + v_k \tag{7}$$

$$v_k = \psi_{k-1}v_{k-1} + \zeta_{k-1}. \tag{8}$$

Here, F_{k-1} is the state transition matrix in discrete time, H_k is the output vector in discrete time, v_k is the colored noise state, w_{k-1} is white noise, ζ_{k-1} is white noise, $\mathbf{x}_k$ is the unobserved state, and y_t is the observation of the state corrupted by colored measurement noise. An auxiliary signal, y', that contains only white noise instead of colored noise, is defined as follows:

$$y'_{k-1} = y_k - \psi_{k-1}y_{k-1} \tag{9}$$

$$= (H_kF_{k-1} - \psi_{k-1}H_{k-1})\mathbf{x}_{k-1} + (H_kw_{k-1} + \zeta_{k-1})$$

$$= H'_{k-1}\mathbf{x}_{k-1} + v'_{k-1}. \tag{10}$$

By using the new output vector, H'_k, the system dynamics are now represented by the following set of equations:

$$\mathbf{x}_k = F_{k-1}\mathbf{x}_{k-1} + w_{k-1}$$

$$y'_k = H'_k\mathbf{x}_k + v'_k. \tag{11}$$

The dynamic equations presented in (11) are corrupted by random noise with zero mean and finite variance (i.e., $v'_k \sim \mathcal{N}(0, \bar{Q}_k)$ and $w_{k-1} \sim \mathcal{N}(0, Q_k)$). Therefore, the Kalman filter can be directly applied to the system dynamics in (11). By doing so, the RGP update step now changes to the following:

$$\bar{G}_k = C^f_{k-1}H'^T_k\left(H'_kC^f_{k-1}H'^T_k + \sigma^2_{\text{noise}}I\right)^{-1} \tag{12}$$

$$\boldsymbol{\mu}^f_k = \boldsymbol{\mu}^f_{k-1} + \bar{G}_k\left(y'_k - H'_k\boldsymbol{\mu}^f_{k-1}\right) \tag{13}$$

$$C^f_k = C^f_{k-1} - \bar{G}_kH'_kC^f_{k-1}. \tag{14}$$

In summary, the proposed extension of the RGP algorithm, denoted by RGP-C, utilizes a measurement differencing approach to consider colored noise within a Kalman filter update, where the states are the mean function value at each basis vector location. The inference step of the RGP-C algorithm remains the same as that of [9], while the update step for the RGP-C algorithm is given in (12)–(14).

IV. RGP-Based Adaptive Control

A. Real-Time Design Space Exploration Strategy

To form an adaptive control law, the RGP model must be augmented with an exploration strategy. This exploration strategy, inspired by iterative design of experiments, seeks to select the point that maximizes the prediction variance ($\sigma^2_{\text{pred}}(\mathbf{p}_c)$) at each time step. However, it must be acknowledged that large design space moves cannot be realized within a single time step. For physical systems described by slow time constants, large design space moves at each time step will result in large transients that need to settle prior to additional design decisions. Recognizing the physical limitations of large design space moves, the control parameter update law is given by

$$\bar{\mathbf{p}}^k_c = \arg\max_{\bar{\mathbf{p}}_c \in \Omega} \sigma^2_{\text{pred}}(\bar{\mathbf{p}}_c)\exp\left(-K_{\text{sel}}\left(d\left(\bar{\mathbf{p}}^k_c, \bar{\mathbf{p}}^{k-1}_c\right)\right)^2\right). \tag{15}$$

This update law accounts for uncertainty (through prediction variance) and distance between potential control parameters at subsequent time steps [through $d(\bar{\mathbf{p}}_c^k, \bar{\mathbf{p}}_c^{k-1})$]. The Euclidean distance between two points is denoted by $d(\cdot,\cdot)$. The exponential term discourages the selection of design points that are far from the design point at the previous time step.

B. Real-Time Design Space Reduction

Since the design space exploration strategy focuses on moving to areas with high uncertainty, the design space exploration strategy must be augmented with a design space reduction scheme to exploit areas of improved performance. Portions of the design space are rejected when there is a statistically significant difference between the mean function value at any point within the design space and the mean function value at the perceived optimum point. From the RGP-C update, estimates of the prediction mean and variance are available at each time step to calculate an error bar at each basis vector location. The upper and lower limits of the error bar at a point $\bar{\mathbf{p}}_{c,i}$ are given by

$$\boldsymbol{\mu}^f_{k,\text{upper}}(\bar{\mathbf{p}}_{c,i}) = \boldsymbol{\mu}^f_k(\bar{\mathbf{p}}_{c,i}) + z_{\text{conf}}\sigma_{\text{pred}}(\bar{\mathbf{p}}_{c,i}) \quad (16)$$

$$\boldsymbol{\mu}^f_{k,\text{lower}}(\bar{\mathbf{p}}_{c,i}) = \boldsymbol{\mu}^f_k(\bar{\mathbf{p}}_{c,i}) - z_{\text{conf}}\sigma_{\text{pred}}(\bar{\mathbf{p}}_{c,i}) \quad (17)$$

respectively. Here, z_{conf} is a constant that corresponds to the desired confidence level in the form of a z-score, and σ_{pred} is the prediction standard deviation, which is calculated as the square root of σ^2_{pred}. In a maximization problem, a candidate design point is rejected if the upper limit at the candidate point is less than the lower limit at the perceived optimum, which is mathematically given by the following condition:

$$\boldsymbol{\mu}^f_{k,\text{upper}}(\bar{\mathbf{p}}_{c,i}) \leq \boldsymbol{\mu}^f_{k,\text{lower}}(\bar{\mathbf{p}}^*_c) \quad \forall i \in [1, s]. \quad (18)$$

C. Convergence Detection for RGP-Based Adaptive Control

As the RGP-based adaptive control proceeds, the control parameter design space ($\boldsymbol{\Omega}$) continues to shrink in size. To evaluate the performance of this adaptation, an algorithm for detecting sufficient convergence must be defined. To accomplish this, convergence is detected by calculating a metric that quantifies the "flatness" of $\boldsymbol{\mu}^f_k$ over the reduced design space. The metric used here is the percent difference between the maximum and minimum value of $\boldsymbol{\mu}^f_k$ over the reduced design space. Once the percent difference falls below a user-specified threshold value (e.g., 1%, 5%, and so on), the design space exploration is turned off and the point that optimizes the objective function is selected as the operating point. Quantitatively, convergence is detected when the following inequality is met:

$$\frac{\left|\max\left(\boldsymbol{\mu}^f_k(\bar{\mathbf{p}}_c)\right) - \min(\boldsymbol{\mu}^f_k(\bar{\mathbf{p}}_c))\right|}{0.5\left(\max\left(\boldsymbol{\mu}^f_k(\bar{\mathbf{p}}_c)\right) + \min\left(\boldsymbol{\mu}^f_k(\bar{\mathbf{p}}_c)\right)\right)} \leq \epsilon_R \quad \forall \bar{\mathbf{p}}_c \in \boldsymbol{\Omega}(k). \quad (19)$$

Here, $\boldsymbol{\Omega}(k)$ denotes the reduced design space at the kth time step and ϵ_R is the desired percent difference in performance.

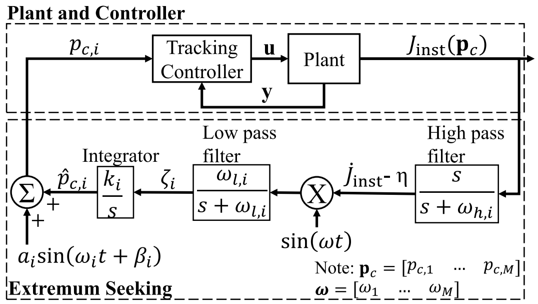
Plant and Controller
$p_{c,i}$
Tracking Controller
u
Plant
$J_{\text{inst}}(\mathbf{p}_c)$
y
Low pass filter
High pass filter
Integrator
$\hat{p}_{c,i}$
$\frac{k_i}{s}$
ζ_i
$\frac{\omega_{l,i}}{s+\omega_{l,i}}$
$\dot{J}_{\text{inst}}$- η
$\frac{s}{s+\omega_{h,i}}$
Σ
+
+
X
$\sin(\omega t)$
$a_i\sin(\omega_i t + \beta_i)$
Note: $\mathbf{p}_c = [p_{c,1} \ \cdots \ p_{c,M}]$
$\boldsymbol{\omega} = [\omega_1 \ \cdots \ \omega_M]$
Extremum Seeking

Fig. 3. Block diagram for the ith element of the control vector in a multiparameter ES algorithm.

V. Comparative Analysis: Extremum Seeking Controller Adaptation

Due to its esteemed place in the adaptive control literature, ES, detailed in [7], is being used as a benchmark to illustrate the efficiency and potential of the contribution in this work, the RGP-based adaptation. Using ES is beneficial in continuous-time optimization when there is limited or no knowledge about the system performance as a function of the control parameters. The limitations of using ES are long convergence times due to the requirement of multiple timescale separations, along with the fact that ES is limited to local performance optimization in an unconstrained setting (i.e., interior optima). The process for ES uses a sequence of a high-pass filter, low-pass filter, and integrator to make adjustments to the controller parameter $\hat{p}_{c,i}$. A sinusoidal perturbation is overlaid on top of $\hat{p}_{c,i}$, which provides the persistent excitation required to converge to the optimal parameter. Fig. 3 shows the ES algorithm for a single-controller parameter within a generic controller design vector $\mathbf{p}_c = [p_{c,1}\ p_{c,2}\ \ldots\ p_{c,M}]^T$. In [7], it is shown that proper selection of ω_i, k_i, $\omega_{h,i}$, $\omega_{l,i}$, and a_i will result in ES converging to a local optimizer.

In several real-world applications, the optimum point lies on the boundary of the "safe" design space (see, for example, [15]). This becomes problematic for ES, as sinusoidal perturbations around the optimal point will temporarily push system parameters into unsafe regions of the design space. While barrier functions can be used to avoid this issue, they limit the convergence of ES by excluding some buffer zone around the edges of the design space.

To compare the efficiency and performance of ES against the proposed RGP-based adaptation, there needs to be a mechanism to determine that the ES algorithm has converged. Once convergence has been detected, the amplitude of the perturbation can then be reduced to zero to evaluate the performance of the system while operating at the perceived optimum. Evaluating the performance of the controller design with the perturbation still engaged would result in unnecessary exploration that would unfairly penalize the performance at the identified optimum. Convergence to the optimal controller parameter is detected by comparing the filtered derivative of $\hat{p}_{c,i}$, denoted by $\dot{\hat{p}}_{c,i}$, to a user-defined threshold. Mathematically, convergence is detected if

$$||\dot{\hat{p}}_{c,i}|| \leq \epsilon_{\text{ES}} \quad \forall t \in [t_c, t_c - \Delta t]. \quad (20)$$

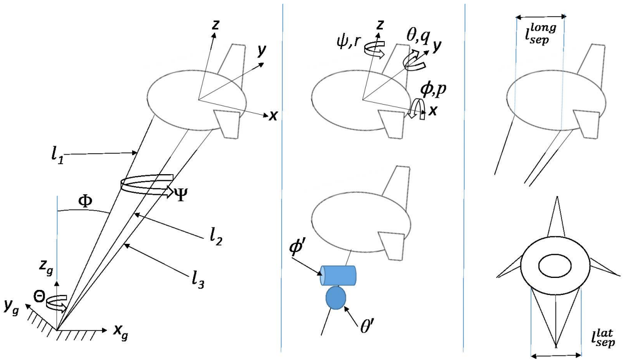

Fig. 4. Axis system for the BAT dynamic model. This diagram shows generalized coordinates (left and bottom middle), Euler angles (top middle), and tether attachment points (right).

After (20) is satisfied for all parameters, the instantaneous performance index value is integrated over a specific time window to compare the performance at the optimal control parameter with that of the RGP-based approach.

VI. Case Study: Buoyant Airborne Turbine

A. AWE Plant Dynamic Model

The AWE system dynamic model used here was derived using an Euler–Lagrange formulation detailed in [21]. The dynamic model approximates all three tethers as a single tether of length L_t with a bridle joint at the airborne shroud. At the bridle joint, there are two controlled degrees of freedom, namely, induced pitch (θ') and induced roll (ϕ'). As discussed in [21], modeling the tether in this manner allows the system dynamics to be characterized entirely by ordinary differential equations, without requiring the incorporation of differential-algebraic equations into the model. The dynamic model is described by the following generalized coordinates: $Q = \begin{bmatrix}\Theta & \Phi & \Psi & L_t & \theta' & \phi'\end{bmatrix}$. The generalized coordinates, along with the ground-fixed and body-fixed coordinate systems, are shown in Fig. 4. The first three coordinates describe the angular orientation of the tether. The tether length (L_t) and induced angles (θ' and ϕ') are the controlled coordinates of the dynamic model. Individual tether release speeds are the control inputs to the system. The controlled coordinates, as a function of the individual tether lengths, are detailed in [21].

B. Lab-Scale Experimental Platform

The water channel-based lab-scale experimental platform from [15], which replicates full-scale flight of AWE systems, was used for the experimental studies in this work. The configuration, shown in Fig. 5, is composed of three dc motors for tether control, three high-speed cameras for image acquisition, and a high-performance target computer for execution of real-time image processing and closed-loop control. The real-time image processing algorithm computes the location and orientation of the scaled model using images from the three cameras. A side view camera is placed perpendicular to the side of the water channel. Meanwhile, two of the cameras are located underneath the water channel; one is perpendicular to the water channel, whereas the other of which is located at a 45° angle. The gray-scale cameras track centroids of the white dots placed on the scaled models. Based on the locations of the dot centroids relative to the body-frame coordinate system, the orientation and position of BAT are determined. The orientation of the scaled BAT is described by the roll (ϕ), pitch (θ), and yaw (ψ) Euler angles. Technical details regarding the image processing algorithm and scaling laws are provided in [15].

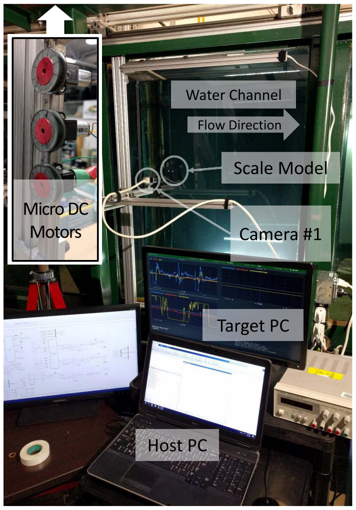

Fig. 5. Laboratory-scale experimental platform at University of North Carolina at Charlotte, Charlotte, NC, USA.

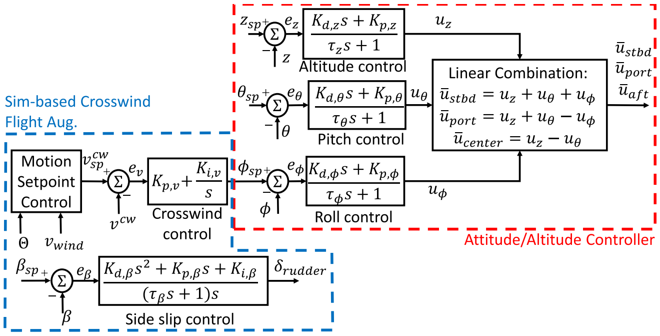
z_{sp}
e_z
$\frac{K_{d,z}s + K_{p,z}}{\tau_z s + 1}$
u_z
z
Altitude control
θ_{sp}
e_θ
$\frac{K_{d,\theta}s + K_{p,\theta}}{\tau_\theta s + 1}$
u_θ
θ
Pitch control
Linear Combination:
$\bar{u}_{stbd} = u_z + u_\theta + u_\phi$
$\bar{u}_{port} = u_z + u_\theta - u_\phi$
$\bar{u}_{center} = u_z - u_\theta$
$\bar{u}_{stbd}$
$\bar{u}_{port}$
$\bar{u}_{aft}$
Sim-based Crosswind Flight Aug.
Motion Setpoint Control
v_{sp}^{cw}
e_v
$K_{p,v} + \frac{K_{i,v}}{s}$
v^{cw}
Crosswind control
ϕ_{sp}
e_ϕ
$\frac{K_{d,\phi}s + K_{p,\phi}}{\tau_\phi s + 1}$
u_ϕ
ϕ
Roll control
Θ
v_{wind}
Attitude/Altitude Controller
β_{sp}
e_β
$\frac{K_{d,\beta}s^2 + K_{p,\beta}s + K_{i,\beta}}{(\tau_\beta s + 1)s}$
δ_{rudder}
β
Side slip control

Fig. 6. Simulation-based crosswind flight controller structure for the BAT.

C. AWE Controller Structure

The BAT system is designed for completely autonomous flight. The lower-level attitude/altitude controller, shown in the block diagram of Fig. 6, consists of three lead filters to control the altitude, pitch angle, and roll angle to desired setpoints. During crosswind flight, the pitch angle setpoint (θ_{sp}) and altitude setpoint (z_{sp}) are constants. Each output from the lower level controllers is mapped to individual tether release speeds by the linear combination shown in Fig. 6.

To achieve crosswind flight, a periodic roll angle setpoint is commanded via an upper level controller. This upper level controller can be as simple as a square wave, as was implemented in the experimental platform, or can include all of the more sophisticated elements shown in Fig. 6 (and first presented in [22]), which present the potential for enhanced flight performance and have been considered in our simulations.

TABLE I

SUMMARY OF CASE-STUDY SPECIFIC VARIABLES

Description	Generic variables	Case study-specific variables
Inst. Performance Index	J_{inst}	$\left(\frac{\lvert v_{cw}\rvert}{\lvert v_{wind}\rvert}\right)^3$
Sim.-based control parameters	$\mathbf{p}_c$	$[\Theta_{lim} \quad \beta_{sq}]$
Experimental control parameters	$\mathbf{p}_c$	$[a_{sq} \quad T_{sq}]$

When a simple square wave roll angle setpoint is used, which serves to redirect lift and, therefore, induce crosswind flight, and the roll angle setpoint is parameterized according to an amplitude (a_{sq}) and period (T_{sq}), as shown here

$$\phi_{\text{sp}} = \begin{cases} a_{\text{sq}}(t), & \displaystyle\int_{t_i}^{t} \frac{1}{T_{\text{sq}}(\tau)} d\tau \le 0.5 \\ -a_{\text{sq}}(t), & 0.5 < \displaystyle\int_{t_i}^{t} \frac{1}{T_{\text{sq}}(\tau)} d\tau \le 1. \end{cases} \tag{21}$$

The variable t_i denotes the time at which the present square wave started. Once the integral the reaches a value of 1, it is reset to zero.

D. Control Objective and Constraints

The overall goal of crosswind flight is to augment the apparent wind presented to the turbine by moving the AWE system in rapid crosswind motions. Since the power produced by a turbine is proportional to the cube of apparent wind velocity, even slight increases in apparent wind velocity can greatly increase the power produced by the turbine. For the crosswind controller adaptation problem, we seek to optimize the instantaneous power augmentation factor, given by

$$J_{\text{inst}}(\mathbf{p}_c) = \left(\frac{|v_{\text{cw}}|}{|v_{\text{wind}}|}\right)^3. \tag{22}$$

The RGP-based adaptation is used to identify the geometric characteristics of the crosswind flight path that optimize the instantaneous power factor. For the experimental crosswind controller, the geometric characteristics of the crosswind flight path are described by the square wave amplitude and period. Thus, the design variables to be optimized are a_{sq} and T_{sq}. For the crosswind controller in simulation, two of the most influential control parameters on crosswind flight performance are the azimuth angle limit, Θ_{lim}, which dictates the width of the crosswind flight pattern, and the sideslip angle setpoint, β_{sp}, which is known to significantly influence the aerodynamic efficiency of crosswind flight. Thus, the design variables in the simulation are Θ_{lim} and β_{sq}. Table I summarizes the connection between the generic parameters discussed in Sections III and IV and the case-study specific parameters discussed here.

VII. CASE STUDY RESULTS

The results presented here show that the RGP-based adaptive control law is effective in converging to a small space of control parameters that contain the optimal parameters. Furthermore, comparisons to ES demonstrate favorable convergence times without any sacrifice in performance.

TABLE II

SUMMARY OF EXPERIMENTAL RGP-BASED ADAPTATION RESULTS FOR MULTIPLE CONTROLLER PARAMETER INITIAL CONDITIONS

Initial Condition		Opt. Control Design		Conv. Time(s)	Avg. PF
$a_{sq,0}(^\circ)$	$T_{sq,0}(s)$	$a^*_{sq}(^\circ)$	$T^*_{sq}(s)$		
7	5	15	5.588	1717.75	1.67
10	8	15	5.588	1752.125	1.60
10	10	15	5.882	1590.35	1.60

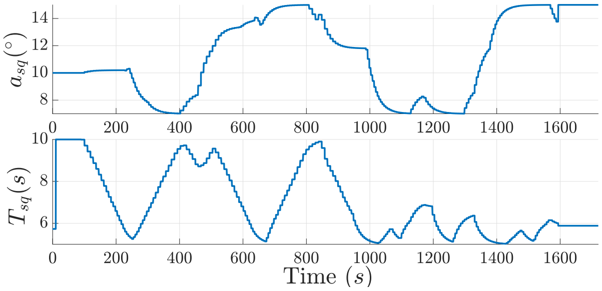

Fig. 7. Experimental design space exploration for initial condition setting with $a_{\text{sq},0} = 10^\circ$ and $T_{\text{sq},0} = 10$ s. Note the first 100 s is used to initialize the experiment.

A. Experimental Crosswind Flight Optimization

To demonstrate the efficient operation of RGP-based adaptive control, multiple experiments were conducted with various initial conditions for the controller design parameters. Table II provides a summary of the results for multiple initial conditions. It is clear that the adaptation converges to nearly the same design point, where small observed differences may be attributed to experimental noise. Fig. 7 displays the control parameter design space exploration from the RGP-based adaptation. Approximately the first 100 s are used to initialize the experiment, which includes initializing the motion capture algorithm and crosswind flight at the prescribed initial condition. Once convergence is detected, around 1600 s in Fig. 7, the adaptation is turned off to calculate the average power factor over a prescribed period, for the identified optimal control parameters. Fig. 8 displays the mean function estimate from the RGP model over the reduced design space at multiple times throughout the experiment. By the time convergence is detected, the design space has been reduced to 12% of its original size.

B. Simulation-Based Crosswind Flight Optimization

The simulation-based analysis was performed at full scale. Fig. 9 displays the design space exploration achieved for the simulation-based case study. Fig. 10 displays the mean function estimate from the RGP-C model across the reduced design space at multiple times during the simulation. After 3000 s of the RGP-based adaptation, the design space has been reduced to 5% of the original design space and consists of control parameters that result in power augmentation of at least 40% through efficient crosswind motion.

C. Comparison of RGP-Based and ES Adaptations in 2D

To evaluate the performance of the RGP-based adaptation, the convergence rate and performance of the point to which the ES- and RGP-based adaptations converged to were compared.

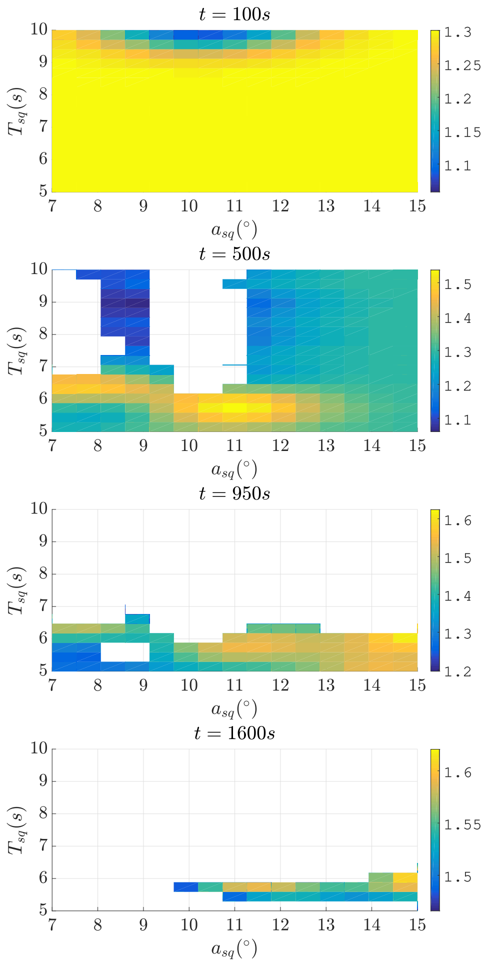

Fig. 8. Design space and response surface characterization (contours) at $t = 100$, $t = 500$, $t = 950$, and $t = 1600$ s for experimental case study.

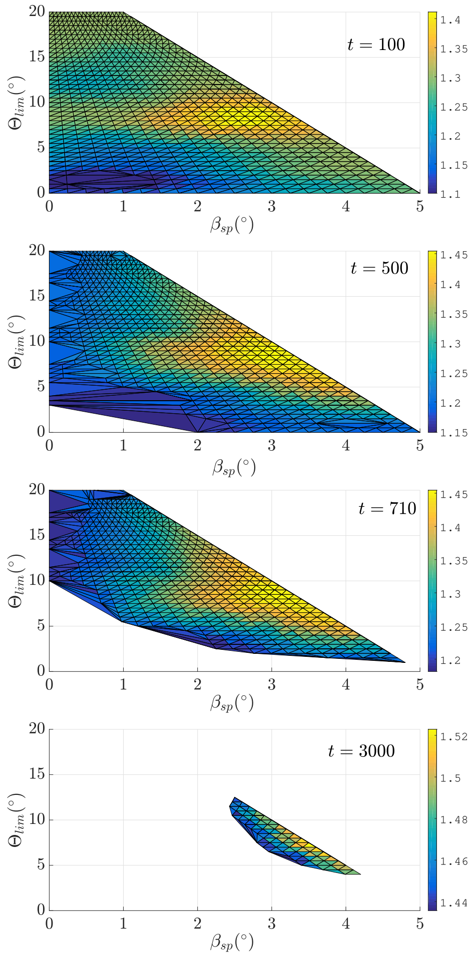

Fig. 10. Design space and response surface characterization (contours) at $t = 100$, $t = 500$, $t = 710$, and $t = 3000$ s for simulation-based case study.

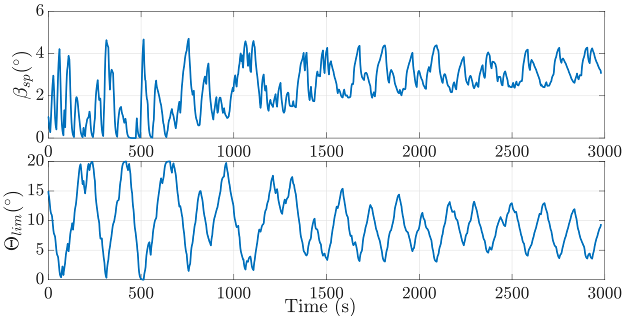

Fig. 9. Simulation-based design space exploration for RGP-based adaptation.

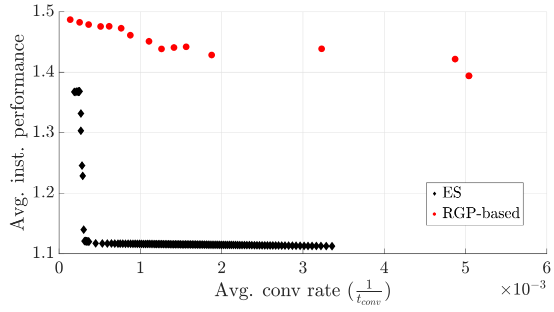

Fig. 11. Pareto front for comparing performance of ES and RGP-based adaptation in 2-D.

A Pareto front for each algorithm was generated by varying the convergence criteria for each algorithm. For ES, this corresponds to adjusting the threshold on the filtered derivative [i.e., ϵ_{ES} in (20)]. Meanwhile, for the RGP-based adaptation, the percent difference threshold was adjusted [i.e., ϵ_R in (19)]. For each convergence criterion, multiple simulations with different plant designs were executed. The average convergence rate (i.e., the inverse of the convergence time) and the resulting integral performance value for each convergence criterion was calculated. Each point in Fig. 11 corresponds to the average convergence rate and performance across multiple plant designs for a single-convergence criterion. As the criterion is made smaller, the convergence rate decreases, but the resulting performance is improved, as shown in Fig. 11. When the criteria are loosened, convergence rates increase, and the performance of the ES controller degrades, whereas the RGP-based adaptive control maintains consistent performance under these looser convergence criteria.

Fig. 12 displays the instantaneous performance versus time for a nominal set of controller parameters, the controller parameters converged to by the RGP-based adaptation, and the controller parameters converged to by ES. In this case study, the RGP-based controller adaptation significantly outperforms

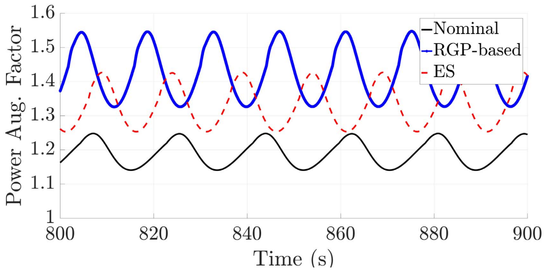

Fig. 12. Comparison of instantaneous performance index values of controllers from the RGP-based and ES adaptations.

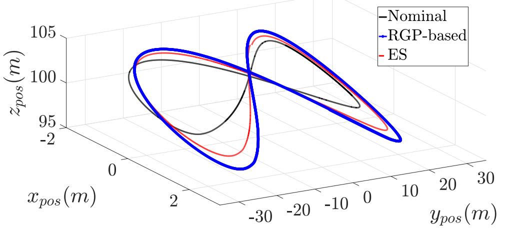

Fig. 13. Comparison of crosswind flight paths between controller parameters from the nominal case, ES adaptation, and RGP-based adaptation.

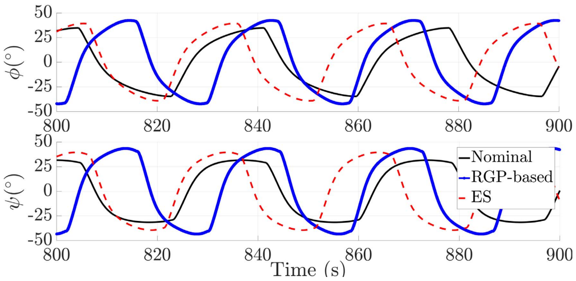

Fig. 14. Comparison of roll and heading angle between controllers from the nominal case, ES adaptation, and RGP-based adaptation.

the ES algorithm. This arises from the fact that the optimal control parameters lie on the boundary of the stable design space, which required a barrier function for the ES algorithm to ensure the unstable region was not explored. The use of a barrier function resulted in significantly inferior performance for the ES algorithm. The flight path of the BAT while operating at the controller parameters from the nominal design, ES adaptation, and the RGP-based adaptation are displayed in Fig. 13. In addition, the behavior of the roll and heading angle for each controller can be seen in Fig. 14.

VIII. Conclusion

A unique, nonmodel-based adaptive control technique that fuses GP modeling with falsification-based adaptive control was presented here. Global design space exploration was achieved by making design decisions based on the uncertainty characterization from a real-time GP model. At each time step, regions of the design space were rejected with confidence using statistical error bars. The adaptation was validated using the crosswind flight optimization of an AWE system as a case study. The efficacy of the RGP-based adaptation was shown by comparing the results to a mature adaptive control technique, namely, ES.

References

[1] P. A. Ioannou and J. Sun, *Robust Adaptive Control*, vol. 1. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

[2] I. Landau, *Adaptive Control: The Model Reference Approach*, vol. 1. Boca Raton, FL, USA: CRC Press, 1979.

[3] S. Nicosia and P. Tomei, "Model reference adaptive control algorithms for industrial robots," *Automatica*, vol. 20, no. 5, pp. 635–644, Sep. 1984.

[4] L. Praly, "Robust model reference adaptive controllers, part I: Stability analysis," in *Proc. 23rd IEEE Conf. Decis. Control*, Dec. 1984, pp. 1009–1014.

[5] M. Krstic, I. Kanellakopoulos, and P. V. Kokotovic, "A new generation of adaptive controllers for linear systems," in *Proc. 31st IEEE Conf. Decis. Control*, Dec. 1992, pp. 3644–3650.

[6] M. Krstic, I. Kanellakopoulos, and P. V. Kokotovic, "Nonlinear design of adaptive controllers for linear systems," *IEEE Trans. Autom. Control*, vol. 39, no. 4, pp. 738–752, Apr. 1994.

[7] K. Ariyur and M. Krstic, *Real-Time Optimization by Extremum-Seeking Control*. Hoboken, NJ, USA: Wiley, 2003.

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

[9] M. F. Huber, "Recursive Gaussian process regression," in *Proc. IEEE Int. Conf. Acoust., Speech Signal Process.*, May 2013, pp. 3362–3366.

[10] M. F. Huber, "Recursive Gaussian process: On-line regression and learning," *Pattern Recognit. Lett.*, vol. 45, pp. 85–91, Aug. 2014.

[11] A. Bryson, Jr., and L. Henrikson, "Estimation using sampled data containing sequentially correlated noise," *J. Spacecraft Rockets*, vol. 5, no. 6, pp. 662–665, 1968.

[12] C. L. Archer, L. Delle Monache, and D. L. Rife, "Airborne wind energy: Optimal locations and variability," *Renew. Energy*, vol. 64, pp. 180–186, Apr. 2014.

[13] A. U. Zgraggen, L. Fagiano, and M. Morari, "On real-time optimization of airborne wind energy generators," in *Proc. 52nd IEEE Conf. Decis. Control*, Dec. 2013, pp. 385–390.

[14] A. U. Zgraggen, L. Fagiano, and M. Morari, "Real-time optimization and adaptation of the crosswind flight of tethered wings for airborne wind energy," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 2, pp. 434–448, Mar. 2015.

[15] M. Cobb, N. Deodhar, and C. Vermillion, "Lab-scale experimental characterization and dynamic scaling assessment for closed-loop crosswind flight of airborne wind energy systems," *J. Dyn. Syst., Meas., Control*, vol. 140, no. 7, pp. 071005-1–071005-12, Jul. 2018.

[16] *The Next Transformation in Rural Infrastructure Efficiency*, Altaeros, Inc., Somerville, MA, USA, 2019.

[17] J. Deese and C. Vermillion, "Real-time experimental optimization of closed-loop crosswind flight of airborne wind energy systems via recursive Gaussian process-based adaptive control," in *Proc. IEEE Conf. Control Technol. Appl. (CCTA)*, 2020.

[18] A. Bafandeh and C. Vermillion, "Real-time altitude optimization of airborne wind energy systems using Lyapunov-based switched extremum seeking control," in *Proc. Amer. Control Conf. (ACC)*, Jul. 2016, pp. 4990–4995.

[19] R. Leyva, C. Alonso, I. Queinnec, A. Cid-Pastor, D. Lagrange, and L. Martinez-Salamero, "MPPT of photovoltaic systems using extremum–seeking control," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, no. 1, pp. 249–258, Jan. 2006.

[20] K. S. Peterson and A. G. Stefanopoulou, "Extremum seeking control for soft landing of an electromechanical valve actuator," *Automatica*, vol. 40, no. 6, pp. 1063–1069, Jun. 2004.

[21] C. Vermillion, B. Glass, and B. Szalai, "Development and full-scale experimental validation of a rapid prototyping environment for plant and control design of airborne wind energy systems," in *Proc. ASME Dyn. Syst. Control Conf.*, 2014, pp. V002T18A001–V002T18A010. [Online]. Available: https://asmedigitalcollection.asme.org/DSCC/proceedings-abstract/DSCC2014/46193/V002T18A001/228878

[22] C. Vermillion, "Altitude and crosswind motion control for optimal power-point tracking in tethered wind energy systems with airborne power generation," in *Proc. ASME Dyn. Syst. Control Conf.*, 2013, pp. V003T49A001–V003T49A009. [Online]. Available: https://asmedigitalcollection.asme.org/DSCC/proceedings-abstract/DSCC2013/56147/V003T49A001/228926