Efficient and Robust Online Trajectory Prediction for Non-Cooperative Unmanned Aerial Vehicles

Guangrui Xie* and Xi Chen†
Virginia Tech, Blacksburg, Virginia 24061

https://doi.org/10.2514/1.I010997

As an important type of dynamic data-driven application system, unmanned aerial vehicles (UAVs) are widely used for civilian, commercial, and military applications across the globe. An increasing research effort has been devoted to trajectory prediction for non-cooperative UAVs to facilitate their collision avoidance and trajectory planning. Existing methods for UAV trajectory prediction typically suffer from two major drawbacks: inadequate uncertainty quantification of the impact of external factors (e.g., wind) and inability to perform online detection of abrupt flying pattern changes. This paper proposes a Gaussian process regression (GPR)-based trajectory prediction framework for UAVs featuring three novel components: 1) GPR with uniform confidence bounds for simultaneous predictive uncertainty quantification, 2) online trajectory change-point detection, and 3) adaptive training data pruning. The paper also demonstrates the superiority of the proposed framework to competing trajectory prediction methods via numerical studies using both simulation and real-world datasets.

I. Introduction

THE advancement of technologies, increased access by the public, and reduced cost have enabled fast-growing applications of unmanned aerial vehicles (UAVs), in both commercial and civilian uses, including wireless communication backup and parcel delivery [1–4]. As the number of UAVs increases, concern has arisen regarding their safe and efficient operation, e.g., the possibility of collisions between UAVs and interference with manned aircraft. Furthermore, a strong need for conflict detection and resolution (CD&R) technologies for UAVs emerged in order to support routine UAV beyond visual line of sight operations [5]. The important role of CD&R technologies in air traffic management, such as the Traffic Alert and Collision Avoidance System, dates as far back as the early 1990s [6]. Supported by a large amount of aircraft performance model data, CD&R technologies for manned aircraft have been extensively studied. Recent years have witnessed an upsurge of interest in developing CD&R technologies for non-cooperative UAVs. Reliable CD&R technologies for UAVs hinge on accurate and timely predictions of threat UAV trajectories, which require efficient data transfer and processing given limited onboard computer memory and battery capacity. While non-cooperative UAVs typically do not share data voluntarily, information technologies (such as radar, lightweight and low-power cameras, and computer vision-based systems) enable UAVs to monitor and collect data from threat UAVs actively [7]. Furthermore, an increasing research effort in building cloud-based databases and flight information management systems has made efficient and effective operational information exchange and voluntary data sharing among non-cooperative UAVs closer to reality [5,8].

There exists a rich class of trajectory prediction methods for manned aerial vehicles (e.g., civil aircraft) in the literature. These methods typically assume the availability of flight data on thrust, mass, position, and other variables associated with point mass models. These variables play an essential role in kinematic particle models for inferring future positions of manned aircraft [9–11]. The trajectory prediction methods developed for manned aircraft may be unsuitable or impractical to apply to a maneuvering UAV. Accurate and timely UAV trajectory prediction is far more challenging to achieve, mainly because UAVs can maneuver more aggressively with much shorter detection horizons. Moreover, the data collected for UAVs are more susceptible to the impact of external factors (such as wind force, precipitation, and temperature). Hence, the data for UAVs can be less accurate than those for manned aircraft. For instance, wind force can push a UAV off its path originally planned [12], inducing extra uncertainty to its trajectory.

Innovations in UAV trajectory prediction methods have been gaining momentum in recent years. Many recently developed model-based approaches for UAV trajectory prediction use position data collected via active sensing using radar and lidar; see, e.g. [13–16]. Specifically, studies [13,14] adopted Kalman filter (KF) for predicting trajectories, whereas studies [15,16] employed a particle filter. Model-free machine learning techniques (e.g., neural networks) have also attracted a lot of attention [17,18]. These methods do not rely on kinetic equations; instead, they try to extract patterns from the historical position and kinetics data for predicting the future positions of a target UAV. These existing methods, although effective on many occasions, suffer from several drawbacks. First and foremost, few previous studies focused on addressing uncertainty quantification of predictions made for the future trajectory of a target UAV. Indeed, there are prediction methods such as KF that can effectively mitigate sensor errors. There also exist research attempts that estimate external disturbances and counteract their impact on trajectory predictions (e.g., [7]). A more thorough solution—adequately quantifying the prediction uncertainty owing to external factors (potentially unknown and of various types)—however, remains elusive. Second, most existing trajectory prediction methods lack the capability of identifying abrupt changes in UAV flying patterns (e.g., a sharp turn). Thus, these methods may fail to make prompt model parameter adjustments to adapt to a new flying pattern and make accurate predictions. This can result in a substantial loss in the predictive accuracy achieved by existing trajectory prediction methods [19].

This paper proposes a robust online trajectory prediction framework for non-cooperative UAVs and demonstrates its practical implementation efficiency and effectiveness. The proposed prediction framework features the following three components:

1) A Gaussian process regression (GPR) model with uniform error bounds is adopted for quantifying the uncertainty of predictions induced by external factors.

2) A change-point detection technique based on sequential generalized likelihood ratio test is applied to detect abrupt changes in the flying pattern of a target UAV, enabling timely GPR model adjustment to adapt to the changes to maintain a high predictive accuracy.

3) A dynamic training data pruning (TDP) technique based on Hellinger distance is adopted for improving the computational

Received 30 March 2021; revision received 17 July 2021; accepted for publication 30 October 2021; published online XX epubMonth XXXX.

*Grado Department of Industrial and Systems Engineering; guanx92@vt.edu.

†Grado Department of Industrial and Systems Engineering; xchen6@vt.edu (Corresponding Author).

efficiency of GPR, which maintains the size of the training dataset at a desirable low level while data streams in, making GPR modeling and prediction sufficiently fast for real-world implementation.

The rest of the paper is organized as follows. Section II provides a review of relevant background knowledge. Section III elaborates on the aforementioned three components of the proposed framework. Section IV presents numerical studies of the proposed framework based on simulation and real-world datasets. Finally, Sec. V concludes the paper.

II. Background Review

In this section, we provide a brief review of relevant background knowledge on GPR modeling, change-point detection, and approaches for improving the computational efficiency of GPR.

A. Gaussian Process Regression

GPR is a popular machine learning method for regression tasks. Compared with other machine learning models such as support vector machine and neural networks, GPR enjoys favorable properties such as being highly flexible to capture various features exhibited by the data at hand and providing an uncertainty measure for its prediction [20]. We provide a brief review of GPR next and refer the interested reader to [20,21] for more details.

Consider a stochastic system whose input point is denoted by $\boldsymbol{x} = (x_1, x_2, \ldots, x_d)^\top \in \mathbb{X} \subset \mathbb{R}^d$. We can observe a noisy system output at $\boldsymbol{x}$, $y(\boldsymbol{x}) \in \mathbb{R}$, which is described by the following model:

$$\begin{aligned} y(\boldsymbol{x}) &= g(\boldsymbol{x})^\top \boldsymbol{\beta} + \mathsf{M}(\boldsymbol{x}) + \varepsilon \\ &= f(\boldsymbol{x}) + \varepsilon \end{aligned} \tag{1}$$

where $f(\boldsymbol{x}) = g(\boldsymbol{x})^\top \boldsymbol{\beta} + \mathsf{M}(\boldsymbol{x})$ denotes the true system response free of observation noise at $\boldsymbol{x}$ that we intend to estimate, $g(\boldsymbol{x})$ is a $p \times 1$ vector of known functions of $\boldsymbol{x}$, and $\boldsymbol{\beta}$ is a vector of unknown coefficients of compatible dimensions. The term ε denotes normally distributed observation noise with mean zero and variance σ^2. GPR modeling further assumes that M is a realization of a mean zero Gaussian process [21]. That is, one can think of M as being randomly sampled from a space of functions mapping from $\mathbb{R}^d$ to $\mathbb{R}$. The functions in this space exhibit spatial correlation; namely, the values of $\mathsf{M}(\boldsymbol{x})$ and $\mathsf{M}(\boldsymbol{x}')$ tend to be similar if $\boldsymbol{x}$ and $\boldsymbol{x}'$ are close to one another in space. This is captured by the covariance function $K(\cdot,\cdot)$, where $K(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}(\mathsf{M}(\boldsymbol{x}), \mathsf{M}(\boldsymbol{x}')) = \zeta^2 R(\boldsymbol{x}, \boldsymbol{x}')$ for any $\boldsymbol{x}, \boldsymbol{x}' \in \mathbb{R}^d$, with ζ^2 and $R(\cdot,\cdot)$ denoting the process variance and the correlation function, respectively. Commonly used correlation functions include the squared exponential and Matérn kernels; for a more detailed discussion, see, e.g., Chapter 4 of [20].

Given a training dataset of sample size N, $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$, comprising the $N \times 1$ vector of observed input points $\boldsymbol{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N)^\top$ and the $N \times 1$ vector of corresponding noisy outputs $\boldsymbol{Y} = (y(\boldsymbol{x}_1), y(\boldsymbol{x}_2), \ldots, y(\boldsymbol{x}_N))^\top$, we are interested in predicting the true system response $f(\boldsymbol{x}^*)$ at any input point $\boldsymbol{x}^* \in \mathbb{X}$. GPR considers linear predictors of the form $\hat{f}(\boldsymbol{x}^*) = \boldsymbol{c}(\boldsymbol{x}^*)^\top \boldsymbol{Y}$, where $\boldsymbol{c}(\boldsymbol{x}^*)$ is an $N \times 1$ vector of weights that depends on the test point $\boldsymbol{x}^*$ and is chosen to give desirable properties such as minimum mean-squared error (MSE) for predicting $f(\boldsymbol{x}^*) = g(\boldsymbol{x}^*)^\top \boldsymbol{\beta} + \mathsf{M}(\boldsymbol{x}^*)$. Specifically, GPR obtains the best linear unbiased predictor (BLUP) by choosing $\boldsymbol{c}(\boldsymbol{x}^*)$ to minimize $\mathrm{MSE}[\hat{f}(\boldsymbol{x}^*)] = \mathrm{E}[(\boldsymbol{c}(\boldsymbol{x}^*)^\top \boldsymbol{Y} - f(\boldsymbol{x}^*))^2]$ subject to the unbiasedness constraint: $\mathrm{E}[\boldsymbol{c}(\boldsymbol{x}^*)^\top \boldsymbol{Y}] = \mathrm{E}[f(\boldsymbol{x}^*)]$. Define the $N \times p$ matrix $\boldsymbol{G}$ of full rank as $\boldsymbol{G} = (g(\boldsymbol{x}_1)^\top, g(\boldsymbol{x}_2)^\top, \ldots, g(\boldsymbol{x}_N)^\top)^\top$. For ease of exposition, we abuse the notation slightly and denote $K(\boldsymbol{X}, \boldsymbol{X})$ as the $N \times N$ matrix that contains the pairwise covariance between each pair of training input points in $\boldsymbol{X}$. Similarly, denote $K(\boldsymbol{X}, \boldsymbol{x}^*)$ as the $N \times 1$ vector that contains the pairwise covariance between each of the N input points in $\boldsymbol{X}$ and $\boldsymbol{x}^*$. Following similar steps as given in Sec. 1.5 of [22] and Appendix A.1 of [23], we introduce a $p \times 1$ Lagrange multiplier vector $\boldsymbol{\lambda}$ and obtain that $\hat{f}(\boldsymbol{x}^*) = \boldsymbol{c}(\boldsymbol{x}^*)^\top \boldsymbol{Y}$ is the BLUP for $f(\boldsymbol{x}^*)$ if the following condition holds:

$$\begin{bmatrix} K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N & \boldsymbol{G} \\ \boldsymbol{G}^\top & \boldsymbol{0}_p \end{bmatrix} \begin{bmatrix} \boldsymbol{c}(\boldsymbol{x}^*) \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} K(\boldsymbol{X}, \boldsymbol{x}^*) \\ g(\boldsymbol{x}^*) \end{bmatrix}$$

where $\boldsymbol{I}_N$ denotes the $N \times N$ identity matrix and $\boldsymbol{0}_p$ represents the $p \times p$ matrix of zeros. Solving the system of equations above, we obtain that the optimal weight vector is given by $\boldsymbol{c}(\boldsymbol{x}^*) = (\Sigma^{-1} - \Sigma^{-1} \boldsymbol{G}(\boldsymbol{G}^\top \Sigma^{-1} \boldsymbol{G})^{-1} \boldsymbol{G}^\top \Sigma^{-1}) K(\boldsymbol{X}, \boldsymbol{x}^*) + \Sigma^{-1} \boldsymbol{G}(\boldsymbol{G}^\top \Sigma^{-1} \boldsymbol{G})^{-1} g(\boldsymbol{x}^*)$, where for notational compactness we have used $\Sigma = K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N$. Hence, the corresponding BLUP for $f(\boldsymbol{x}^*)$ follows as

$$\hat{f}(\boldsymbol{x}^*) = g(\boldsymbol{x}^*)^\top \hat{\boldsymbol{\beta}} + K(\boldsymbol{X}, \boldsymbol{x}^*)^\top (K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N)^{-1} (\boldsymbol{Y} - \boldsymbol{G}\hat{\boldsymbol{\beta}}) \tag{2}$$

where $\hat{\boldsymbol{\beta}} = (\boldsymbol{G}^\top (K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N)^{-1} \boldsymbol{G})^{-1} \boldsymbol{G}^\top (K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N)^{-1} \boldsymbol{Y}$ is the generalized least-squares estimator of $\boldsymbol{\beta}$. The variance of the BLUP, which equals its MSE because the predictor is unbiased, follows as

$$\begin{aligned} \mathrm{Var}[\hat{f}(\boldsymbol{x}^*)] = {} & K(\boldsymbol{x}^*, \boldsymbol{x}^*) - K(\boldsymbol{X}, \boldsymbol{x}^*)^\top (K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N)^{-1} K(\boldsymbol{X}, \boldsymbol{x}^*) \\ & + \eta^\top (\boldsymbol{G}^\top (K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N)^{-1} \boldsymbol{G})^{-1} \end{aligned} \tag{3}$$

where $\boldsymbol{\eta} = g(\boldsymbol{x}^*) - \boldsymbol{G}^\top (K(\boldsymbol{X}, \boldsymbol{X}) + \sigma^2 \boldsymbol{I}_N)^{-1} K(\boldsymbol{X}, \boldsymbol{x}^*)$. It is worth noting that, from a Bayesian inference perspective, $\hat{f}(\boldsymbol{x}^*)$ in Eq. (2) and $\mathrm{Var}[\hat{f}(\boldsymbol{x}^*)]$ in Eq. (3) are regarded as the posterior mean and variance of the true system response $f(\boldsymbol{x}^*)$ conditional on the training dataset $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ [20].

In particular, $\hat{f}(\boldsymbol{x}^*)$ in Eq. (2) provides a point prediction for $f(\boldsymbol{x}^*)$ and $\mathrm{Var}[\hat{f}(\boldsymbol{x}^*)]$ in Eq. (3) gives a natural uncertainty measure for the point prediction of GPR. In cases where an interval prediction of $f(\boldsymbol{x}^*)$ for an arbitrary test point $\boldsymbol{x}^*$ is needed, a standard pointwise confidence interval with $100(1-\delta)\%$ confidence level ($\delta \in (0,1)$) can be constructed with the upper and lower confidence limits ($\mathrm{UCL}(f(\boldsymbol{x}^*))$ and ($\mathrm{LCL}(f(\boldsymbol{x}^*))$ respectively given by $\hat{f}(\boldsymbol{x}^*) \pm z_{1-\delta/2} \sqrt{\mathrm{Var}[\hat{f}(\boldsymbol{x}^*)]}$, where $z_{1-\delta/2}$ denotes the $(1-\delta/2)$-quantile of the standard normal distribution.

For control purposes, we are often interested in achieving a prescribed high probability guarantee (say, at level $1-\delta$) that the true system response values $f(\boldsymbol{x}^*)$'s at multiple input points in a test set (say, $\boldsymbol{X}^*$) are simultaneously covered. In such cases, directly obtaining the pointwise $100(1-\delta)\%$ confidence level at each $\boldsymbol{x}^*$ is problematic, as the resulting simultaneous coverage probability will be much lower than the nominal value $1-\delta$. To resolve this issue, one can apply the Sidak correction to construct individual pointwise confidence intervals [24]. The upper and lower confidence limits of Sidak corrected pointwise confidence interval for $f(\boldsymbol{x}^*)$ can be respectively given by

$$\mathrm{UCL}(f(\boldsymbol{x}^*)) = \hat{f}(\boldsymbol{x}^*) + z_{1-\psi/2} \sqrt{\mathrm{Var}[\hat{f}(\boldsymbol{x}^*)]} \tag{4}$$

$$\mathrm{LCL}(f(\boldsymbol{x}^*)) = \hat{f}(\boldsymbol{x}^*) - z_{1-\psi/2} \sqrt{\mathrm{Var}[\hat{f}(\boldsymbol{x}^*)]} \tag{5}$$

where $\psi = 1 - (1-\delta)^{1/N^*}$ and N^* is the number of test points to make predictions simultaneously.

B. Change-Point Detection

Change-point detection (CPD) is the problem of finding abrupt changes in patterns exhibited by time series data [25]. One disadvantage of existing trajectory prediction methods is that they cannot promptly adapt to abrupt changes (or change points) in the flying patterns of the target UAVs. This usually results in a delay in updating the model parameters, rendering the model's trajectory predictions inaccurate. Hence, having an adequate CPD capability is critical to the success of UAV trajectory prediction methods.

Various approaches have been proposed for CPD in the literature, and some are dedicated to the context of GPR modeling. For example, a Bayesian online CPD algorithm was proposed in [26] to obtain a

posterior distribution over change-point times. Another CPD algorithm for GPR was proposed in [27], which incorporates the change-point time as an input variable in the kernel function and estimates the change-point time via optimization. However, these approaches typically require a high computational cost that scales with the number of training points, rendering them infeasible for real-time trajectory prediction.

Most of the remaining CPD methods were proposed for a more general setting other than GPR modeling. These methods can be classified into two classes: supervised and unsupervised learning approaches. Supervised learning approaches treat CPD as a classification problem. Various machine learning models (e.g., decision tree, support vector machine, neural network) in this category can be employed to separate change points from the other data points. However, these approaches are problematic if applied for online trajectory prediction as they are typically computationally demanding and require a large sample of labeled change points. On the other hand, unsupervised learning approaches are more computationally efficient and do not require labeled data. The most commonly used methods of this type are based on generalized likelihood ratio test (GRLT). The idea is to analyze the probability distributions of data before and after a candidate change point, and declare the candidate a change point if the difference between the two distributions is detected to be statistically significant. The likelihood ratio between two consecutive intervals in time series data is monitored for detecting change points. The methods based on GLRT are computationally efficient and provide a statistical guarantee for the false-alarm rate; hence they are more suitable for UAV trajectory prediction. The interested reader is referred to [28,29] for a detailed survey of methods for CPD.

C. Approaches for Improving the Computational Efficiency of GPR

An impediment to online implementation of GPR is the computational complexity, which scales as $\mathcal{O}(N^3)$, where N is the number of training points used for constructing a standard GPR model [20]. As data accumulate over time for a time-dependent system, the computational cost grows rapidly if all data collected are used for GPR modeling.

The efforts to make the training of a GPR model on a large dataset more computationally efficient have been focused on constructing sparse GPR models. The idea behind sparse GPR modeling is to find a set of inducing points from the full training dataset to construct a GPR model and make inference as well as subsequent predictions [30,31]. The resulting computational cost scales as $\mathcal{O}(M^3)$, where M denotes the number of inducing points. The size of the inducing-point set M is much lower than that of the full training dataset N. However, finding inducing points is a challenging optimization problem, which is typically computationally expensive to solve.

Apart from sparse GPR modeling, many previous studies proposed to construct a GPR model based on a moving window that only contains a portion of data received most recently. The rationale behind this idea is that data observations are typically more correlated if they are close in time; hence, the most recently received data are more relevant for prediction at the next time instant than data received in the distant past. The size of the moving window can be dynamically adjusted to achieve satisfactory predictive performance. Various methods have been proposed for dynamically adjusting the training window size; e.g., see [32–36]; however, most of these methods are heuristics in nature and not tailored for GPR modeling. Hence, they cannot provide any rigorous statistical guarantee on the predictive performance of the resulting GPR model constructed on the truncated dataset. A rigorous and efficient data truncation technique that works well in conjunction with GPR modeling is required.

III. GPR-Based Online Trajectory Prediction Framework

In this section, we propose a robust GPR-based online trajectory prediction framework comprising three important components: 1) GPR with uniform confidence bounds to adequately quantify the uncertainty induced by external factors, 2) a change-point detection (CPD) method based on generalized likelihood ratio test to detect abrupt changes in the target UAV's flying pattern, and 3) a TDP technique to improve the computational efficiency of GPR for online implementation. We refer to the proposed framework as GUCT hereinafter, where "GU," "C," and "T" respectively stand for the three components.

We first elaborate on the three components of GUCT in Secs. III.A, III.B, and III.C. Then in Sec. III.D, we discuss how the three components are integrated into a unified framework.

A. GPR Modeling for Trajectory Prediction with Uniform Confidence Bounds

In our problem setting, we aim to predict the future trajectory of a target UAV at each time instant. The position of a UAV at the current time instant t_i is uniquely defined by a tuple using the Global Positioning System (GPS): (latitude, longitude, altitude). GPS sensors can provide accurate position data of the objects to which they are attached, thanks to the satellites. Denote the latitude, longitude, and altitude of a UAV at the current time instant t_i by la_{t_i}, lo_{t_i}, and al_{t_i}, respectively.

To address the first drawback of existing trajectory prediction methods, lack of uncertainty quantification of the impact of external factors, we use GPR to model the impact of external factors directly. Consider predicting the target UAV's position at the next time instant t_{i+1}. Denote the time interval between time instants t_i and t_{i+1} by Δt, which is usually very small in the setting of UAV trajectory prediction; the velocity of the UAV on each axis, v_t^{la}, v_t^{lo}, and v_t^{al}, can be regarded as constants in (t_i, t_{i+1}). In light of a simple kinetics rule and the influence of external factors, we can model the UAV's position at t_{i+1} as follows:

$$la_{t_{i+1}} = la_{t_i} + v_{t_i}^{la}\Delta t + \tilde{w}_{t_{i+1}}^{la} = la_{t_i} + v_{t_i}^{la}\Delta t + w_{t_{i+1}}^{la} + \varepsilon^{la} \quad (6)$$

$$lo_{t_{i+1}} = lo_{t_i} + v_{t_i}^{lo}\Delta t + \tilde{w}_{t_{i+1}}^{lo} = lo_{t_i} + v_{t_i}^{lo}\Delta t + w_{t_{i+1}}^{lo} + \varepsilon^{lo} \quad (7)$$

$$al_{t_{i+1}} = al_{t_i} + v_{t_i}^{al}\Delta t + \tilde{w}_{t_{i+1}}^{al} = al_{t_i} + v_{t_i}^{al}\Delta t + w_{t_{i+1}}^{al} + \varepsilon^{al} \quad (8)$$

where $\tilde{w}_{t_{i+1}}^{la} = w_{t_{i+1}}^{la} + \varepsilon^{la}$ denotes the observed nonlinear change in latitude at time instant t_{i+1}; $\tilde{w}_{t_{i+1}}^{lo}$ and $\tilde{w}_{t_{i+1}}^{al}$ are defined similarly with respect to the longitude and altitude; and ε^{la}, ε^{lo}, and ε^{al} respectively denote the normally distributed mean zero observation noise with respective variances σ_{la}^2, σ_{lo}^2, and σ_{al}^2. The terms $w_{t_{i+1}}^{la}$, $w_{t_{i+1}}^{lo}$, and $w_{t_{i+1}}^{al}$ denote the impact of external forces on each axis of the UAV's trajectory at time instant t_{i+1}. While $v_{t_i}^{la}$, $v_{t_i}^{lo}$, and $v_{t_i}^{al}$ can be easily calculated from the real-time dataset given a sufficiently small Δt, $w_{t_{i+1}}^{la}$, $w_{t_{i+1}}^{lo}$, and $w_{t_{i+1}}^{al}$ are unknown and challenging to capture using sensors. In our framework, we choose to model w^{la}, w^{lo}, and w^{al} using GPR. Assuming that w^{la}, w^{lo}, and w^{al} are independent of each other and have zero mean, we can construct separate GPR models for them by setting $g(\cdot) = 0$ in Eq. (1), which are respectively denoted by GP_{la}, GP_{lo}, and GP_{al}.

We elaborate on the GPR modeling of $w_{t_{i+1}}^{la}$ and the prediction for the future latitude $la_{t_{i+1}}$ next. Consider a training dataset $\mathcal{D}_{t_i}^{la} = \{(\boldsymbol{x}_{t_j}, \tilde{w}_{t_j}^{la})\}_{j=1}^{N}$ for GP_{la}, where $\boldsymbol{x}_{t_j}$ is the training input point at t_j, $\tilde{w}_{t_j}^{la}$ is the corresponding output at t_j, and N denotes the number of training data points accumulated by the current time instant t_i. Notice that the input point $\boldsymbol{x}_{t_j}$ can incorporate any variables that are useful for predicting $w_{t_j}^{la}$, such as position data, pose data, and environmental data (e.g., wind, precipitation, and temperature) if available. The training output $\tilde{w}_{t_j}^{la}$ can be solved from Eq. (6) as $\tilde{w}_{t_j}^{la} = la_{t_j} - la_{t_{j-1}} - v_{t_{j-1}}^{la}\Delta t$ for $j = 1, 2, \ldots, N$. Upon training GP_{la} on $\mathcal{D}_{t_i}^{la}$, we can obtain a point prediction $\widehat{w}_{t_{i+1}}^{la}$ for $w_{t_{i+1}}^{la}$ via Eq. (2) and the corresponding predictive variance $\mathrm{Var}[w_{t_{i+1}}^{la}]$ via Eq. (3) by setting $g(\cdot)$ and $\boldsymbol{G}$ respectively to zeros. The one-step-ahead point prediction for $la_{t_{i+1}}$, denoted by $\widehat{la}_{t_{i+1}}$, can thus be given as $\widehat{la}_{t_{i+1}} = la_{t_i} + v_{t_i}^{la}\Delta t + \widehat{w}_{t_{i+1}}^{la}$. We are also interested in multistep-ahead predictions

within a prespecified prediction horizon H, as UAVs need to take actions in advance to avoid collisions. For n-step ahead point prediction ($1 \le n \le H$), assuming that the velocity v^{la} and the impact of the external factors w^{la} remain unchanged over $(t_i, t_i + n\Delta t)$, we can obtain $\hat{la}_{t_{i+n}} = la_{t_i} + v^{la}_{t_i} n\Delta t + \hat{w}^{la}_{t_{i+1}}$ for $n = 1, 2, \ldots, H$.

Upon obtaining the point predictions, we can build interval predictions for the future latitude of the target UAV as needed. A natural choice for constructing multistep-ahead interval predictions is the Sidak corrected confidence interval as given by Eqs. (4) and (5), because it gives a statistical guarantee to cover all the test points simultaneously in the prediction horizon. However, this correction is theoretically valid with independent system outputs only [37], which is not the case for GPR modeling of trajectory positions. Therefore, a novel type of confidence interval that is uniformly valid across all test points in an arbitrarily defined prediction horizon is required for GPR-based trajectory prediction.

Inspired by [38], we construct uniform error bounds for GP_{la}, GP_{lo}, and GP_{al} to quantify predictive uncertainty simultaneously for all prediction points (potentially infinitely many) in an arbitrarily defined prediction horizon. Take GP_{la} for modeling the latitude as an example. To bound the predictive error achieved by GP_{la}, the idea is to use the Lipschitz continuity of the covariance kernel $K(\cdot,\cdot)$ and the underlying function $w^{la}(\cdot)$ that GP_{la} aims to estimate. Specifically, given a training dataset $\mathcal{D}^{la}_{t_i} = \{\boldsymbol{X}, \boldsymbol{W}\}$, where $\boldsymbol{X} = (\boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_2}, \ldots, \boldsymbol{x}_{t_N})^\top$ and $\boldsymbol{W} = (\tilde{w}^{la}_{t_1}, \tilde{w}^{la}_{t_2}, \ldots, \tilde{w}^{la}_{t_N})^\top$, the Lipschitz constants of the covariance kernel, the posterior mean function, and $w^{la}(\cdot)$ can be respectively given as

$$
\begin{aligned}
L_k &= \max_{\boldsymbol{x},\boldsymbol{x}'\in\mathbb{X}} \left\| \left[\frac{\partial K(\boldsymbol{x},\boldsymbol{x}')}{\partial x_1} \cdots \frac{\partial K(\boldsymbol{x},\boldsymbol{x}')}{\partial x_d} \right]^\top \right\|,\\
L_\mu &= L_k \sqrt{N} \|(K(\boldsymbol{X},\boldsymbol{X}) + \sigma^2_{la}\boldsymbol{I}_N)^{-1}\boldsymbol{W}\|,\\
L_f &= \left\| \begin{bmatrix}
\sqrt{2\log\left(\frac{2d}{\delta_L}\right)} \max_{\boldsymbol{x}\in\mathbb{X}} \sqrt{K^{\partial 1}(\boldsymbol{x},\boldsymbol{x})}\\
+12\sqrt{6d}\max\left\{ \max_{\boldsymbol{x}\in\mathbb{X}} \sqrt{K^{\partial 1}(\boldsymbol{x},\boldsymbol{x})}, \sqrt{rL^{\partial 1}_k} \right\}\\
\vdots\\
\sqrt{2\log\left(\frac{2d}{\delta_L}\right)} \max_{\boldsymbol{x}\in\mathbb{X}} \sqrt{K^{\partial d}(\boldsymbol{x},\boldsymbol{x})}\\
+12\sqrt{6d}\max\left\{ \max_{\boldsymbol{x}\in\mathbb{X}} \sqrt{K^{\partial d}(\boldsymbol{x},\boldsymbol{x})}, \sqrt{rL^{\partial d}_k} \right\}
\end{bmatrix} \right\|
\end{aligned}
$$

where $L^{\partial i}_k$ is the Lipschitz constant of the partial derivative kernel $K^{\partial i}(\cdot,\cdot)(i = 1, 2, \ldots, d)$ on the input space $\mathbb{X} \subset \mathbb{R}^d$ (recall Sec. II.A), $r = \max_{\boldsymbol{x},\boldsymbol{x}'\in\mathbb{X}} \|\boldsymbol{x} - \boldsymbol{x}'\|$ denotes the maximal extension; and $1 - \delta_L$ is the probability for the Lipschitz constant L_f to hold. To economize on space, we refer the reader to [38] for the detailed derivation of the expressions for L_k, L_μ, and L_f. Given a confidence level $100(1-\delta)\%$, we have

$$
\mathrm{P}\left(|w^{la}(\boldsymbol{x}) - \hat{w}^{la}(\boldsymbol{x})| \le \sqrt{\beta(\tau)\mathrm{Var}[\hat{w}^{la}(\boldsymbol{x})]} + \gamma(\tau) \right) \ge 1 - \delta, \quad \forall\, \boldsymbol{x} \in \mathbb{X} \tag{9}
$$

where

$$
\beta(\tau) = 2\log\left(\frac{M(\tau,\mathbb{X})}{\delta}\right), \quad \gamma(\tau) = (L_\mu + L_f)\tau + \sqrt{\beta(\tau)}\eta(\tau),
$$

$$
\eta(\tau) = \left(2\tau L_k\left(1 + N\|(K(\boldsymbol{X},\boldsymbol{X}) + \sigma^2_{la}\boldsymbol{I}_N)^{-1}\| \max_{\boldsymbol{x},\boldsymbol{x}'\in\mathbb{X}} K(\boldsymbol{x},\boldsymbol{x}')\right)\right)^{1/2}
$$

Here, τ is the grid constant of a grid used in the derivation of Eq. (9), and $M(\tau,\mathbb{X})$ is the minimum number of points in a grid over $\mathbb{X}$ with the grid constant τ. Expression (9) indicates that uniform confidence bounds can be constructed such that $w^{la}(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbb{X}$ are covered simultaneously with probability $1 - \delta$. Based on Eq. (9), the upper and lower limits of the uniform confidence interval for $w^{la}_{t_{i+1}}$ can be respectively given by

$$
\mathrm{UCL}(w^{la}_{t_{i+1}}) = \hat{w}^{la}_{t_{i+1}} + \sqrt{\beta(\tau)\mathrm{Var}[\hat{w}^{la}_{t_{i+1}}]} + \gamma(\tau) \tag{10}
$$

$$
\mathrm{LCL}(w^{la}_{t_{i+1}}) = \hat{w}^{la}_{t_{i+1}} - \sqrt{\beta(\tau)\mathrm{Var}[\hat{w}^{la}_{t_{i+1}}]} - \gamma(\tau) \tag{11}
$$

To obtain an interval prediction for the future latitude in the next n time instants, $la_{t_{i+n}} (1 \le n \le H)$, we first make an interval prediction for $w^{la}_{t_{i+1}}$ via Eqs. (10) and (11). Denote the upper and lower confidence limits for $la_{t_{i+n}}$ by $\mathrm{UCL}(la_{t_{i+n}})$ and $\mathrm{LCL}(la_{t_{i+n}})$, respectively; then they can be given as $\mathrm{UCL}(la_{t_{i+n}}) = la_{t_i} + v^{la}_{t_i} n\Delta t + \mathrm{UCL}(w^{la}_{t_{i+1}})$ and $\mathrm{LCL}(la_{t_{i+n}}) = la_{t_i} + v^{la}_{t_i} n\Delta t + \mathrm{LCL}(w^{la}_{t_{i+1}})$.

Analogously, the aforementioned GPR modeling with uniform error bounds can be applied to obtain point and interval estimates along the longitude and altitude axes. To economize on space, we omit the details. Algorithm 1 summarizes the steps for making predictions for future latitude, longitude, and altitude given the dataset available at each time instant t_i.

B. Change-Point Detection Based on Generalized Likelihood Ratio Test

The second component of our framework aims to detect abrupt changes in a target UAV's flying pattern. Here, we adopt a change-point detection (CPD) method based on generalized likelihood ratio test (GLRT) [39]. The idea is to use a GLR-based hypothesis test to monitor the prediction error obtained by the GPR model currently adopted, in order to check if the model is adequate to capture the trajectory of a target UAV based on the existing data at hand.

Assume that the prediction error e obtained by an adequate prediction model follows a normal distribution with mean 0 and variance σ^2 (i.e., $e \sim \mathcal{N}(0,\sigma^2)$). We conduct the following hypothesis test with H_0: $e \sim \mathcal{N}(0,\sigma^2)$ and H_1: $e \sim \mathcal{N}(\theta,\sigma^2)$, where $\theta \ne 0$. The test statistic $G(e)$ can be given by

$$
\begin{aligned}
G(e) &= 2\log\left(\sup_\theta \exp\left(-\frac{(e-\theta)^2}{2\sigma^2}\right) \Big/ \exp\left(-\frac{e^2}{2\sigma^2}\right) \right)\\
&= (\min_\theta (e-\theta)^2 + e^2)/\sigma^2 = e^2/\sigma^2
\end{aligned} \tag{12}
$$

where we have used $\hat{\theta} = \mathrm{argmin}_\theta (e-\theta)^2 = e$. It can be derived that $a = (\sigma^2 + \hat{\theta}^2)/\sigma^2 = (\sigma^2 + e^2)/\sigma^2$ and $b = (2\sigma^4 + \hat{\theta}^2)/\sigma^4 = (2\sigma^4 + e^2)/\sigma^4$ are the mean and variance of $G(e)$, respectively. Because the prediction error e is assumed to be normally distributed

Algorithm 1: GPR training and prediction for future latitude, longitude, and altitude

1: Update the training datasets using new observations, $\mathcal{D}^{la}_{t_i} = \mathcal{D}^{la}_{t_{i-1}} \cup \{(\boldsymbol{x}_{t_i}, \tilde{w}^{la}_{t_i})\}$, $\mathcal{D}^{lo}_{t_i} = \mathcal{D}^{lo}_{t_{i-1}} \cup \{(\boldsymbol{x}_{t_i}, \tilde{w}^{lo}_{t_i})\}$, $\mathcal{D}^{al}_{t_i} = \mathcal{D}^{al}_{t_{i-1}} \cup \{(\boldsymbol{x}_{t_i}, \tilde{w}^{al}_{t_i})\}$.

2: Maximize the respective log marginal likelihood functions of G_{la}, G_{lo}, and G_{al} to estimate their corresponding hyperparameters.

3: Using the GPR models with the updated hyperparameters to obtain the predicted future impact of external factors on latitude, longitude, and altitude incurred at time instant t_{i+1}, $\hat{w}^{la}_{t_{i+1}}$, $\hat{w}^{lo}_{t_{i+1}}$, and $\hat{w}^{al}_{t_{i+1}}$, and their corresponding interval estimates.

4: Predict the future latitude, longitude, and altitude in the next n time instants via $\hat{la}_{t_{i+n}} = la_{t_i} + v^{la}_{t_i} n\Delta t + \hat{w}^{la}_{t_{i+1}}$, $\hat{lo}_{t_{i+n}} = lo_{t_i} + v^{lo}_{t_i} n\Delta t + \hat{w}^{lo}_{t_{i+1}}$, and $\hat{al}_{t_{i+n}} = al_{t_i} + v^{al}_{t_i} n\Delta t + \hat{w}^{al}_{t_{i+1}}$, for $1 \le n \le H$, and their corresponding interval estimates.

under H_0, the statistic $G(e)$ follows a chi-squared distribution under H_0. The control limit G_α can hence be given as

$$G_\alpha = g\chi^2_{h,\alpha} \tag{13}$$

where $g = b/2a$, $h = 2a^2/b$, and $100(1-\alpha)\%$ is the confidence level of the hypothesis test. We reject H_0 if $G(e) > G_\alpha$, and accept it otherwise.

In our framework, we assume that the errors as a result of the three GPR models, G_{la}, G_{lo}, and G_{al}, are normally distributed with mean 0 and respective noise variances given by σ^2_{la}, σ^2_{lo}, and σ^2_{al} (recall Sec. II.A). When any abrupt change appears in the target UAV's trajectory, e.g., the UAV makes a sharp turn, such a change should be reflected by an unusually large predictive error of the current GPR model, as it becomes inadequate to capture the new flying pattern. Hence, the GLRT method can be applied to check the adequacy of the current GPR model and identify change points in the target UAV's trajectory.

Take GP_{la} (i.e., the GPR model for modeling latitude) for an example. At time instant t_i, the error can be given as $e^{la}_{t_i} = la_{t_i} - \widehat{la}_{t_i}$. Under H_0, we have $e^{la}_{t_i} \sim \mathcal{N}(0, \sigma^2_{la})$, where an estimate of σ^2_{la} can be obtained via maximum likelihood estimation while training the GPR model GP_{la}. $G(e^{la}_{t_i})$ and G^{la}_α at a prespecified confidence level $100(1-\alpha)\%$ can be obtained based on Eqs. (12) and (13), respectively. If $G(e^{la}_{t_i}) \le G^{la}_\alpha$, we claim that GP_{la} can adequately describe the UAV flying pattern along the latitude axis; otherwise, we must refit GP_{la} using the newly arrived data due to the change in the data pattern. Analogously, the same CPD routine can be applied to detect changes in trajectory along the longitude and altitude axes. Algorithm 2 summarizes the steps to implement the CPD routine at time instant t_i.

The benefit of having the CPD component in our framework is twofold. First, it gives a timely notification when the UAV's flying pattern changes abruptly, allowing a prompt update of the GPR model. Second, this CPD component improves the computational efficiency of the proposed GPR-based framework, as the GPR models do not need to be refitted at every time instant, but only when a change point is detected. The CPD component, together with the third component (to be detailed next), can help achieve significant computational savings in practice and make GPR modeling suitable for real-time implementation.

C. Training Data Pruning

As traditional approaches for improving the computational efficiency of GPR (reviewed in Sec. II.C) are not suitable for online implementation, we adopt a novel approach for training data pruning (TDP) in the proposed framework. The idea of this particular TDP method, destructive Hellinger matching pursuit, is to greedily prune the data points in the current training dataset as new data stream in, while ensuring that the discrepancy between the posterior distribution before and after pruning is below a prespecified threshold [40].

Algorithm 2: Implementing the CPD routine at a given time instant t_i

1: Calculate $e^{la}_{t_i} = la_{t_i} - \widehat{la}_{t_i}$, $e^{lo}_{t_i} = lo_{t_i} - \widehat{lo}_{t_i}$, and $e^{al}_{t_i} = al_{t_i} - \widehat{al}_{t_i}$.
2: Obtain the estimates of σ^2_{la}, σ^2_{lo}, and σ^2_{al} while training the GPR models GP_{la}, GP_{lo}, and GP_{al}, respectively.
3: Calculate $G(e^{la}_{t_i})$ and G^{la}_α using $e^{la}_{t_i}$ and σ^2_{la} via Eqs. (12) and (13). Similarly, calculate $G(e^{lo}_{t_i})$, G^{lo}_α, $G(e^{al}_{t_i})$, and G^{al}_α.
4: **if** $G(e^{la}_{t_i}) > G^{la}_\alpha$ **then**
5: Refit G_{la} using $\mathcal{D}^{la}_{t_i}$ following steps 1 and 2 in Algorithm 1.
6: **end if**
7: **if** $G(e^{lo}_{t_i}) > G^{lo}_\alpha$ **than**
8: Refit G_{lo} using $\mathcal{D}^{lo}_{t_i}$ following steps 1 and 2 in Algorithm 1.
9: **end if**
10: **if** $G(e^{al}_{t_i}) > G^{al}_\alpha$ **then**
11: Refit G_{al} using $\mathcal{D}^{al}_{t_i}$ following steps 1 and 2 in Algorithm 1.
12: **end if**

The discrepancy between the two posterior distributions is measured by Hellinger distance. Let $\nu(\boldsymbol{x})$ and $\phi(\boldsymbol{x})$ be the probability density functions of two arbitrary distributions, where $\boldsymbol{x} \in \mathbb{X} \subset \mathbb{R}^d$. The Hellinger distance between these two distributions is defined as

$$d_H(\nu, \phi) = \sqrt{\frac{1}{2}\int_{x\in\mathbb{X}} \left(\sqrt{\nu(\boldsymbol{x})} - \sqrt{\phi(\boldsymbol{x})}\right)^2 \mathrm{d}\boldsymbol{x}} \tag{14}$$

In particular, when both $\nu(\boldsymbol{x})$ and $\phi(\boldsymbol{x})$ are multivariate normal distributions respectively with mean vector $\boldsymbol{\mu}_i$ and variance-covariance matrix Σ_i, $i = 1, 2$, i.e., $\nu(\cdot) \sim \mathcal{N}(\boldsymbol{\mu}_1, \Sigma_1)$ and $\phi(\cdot) \sim \mathcal{N}(\boldsymbol{\mu}_2, \Sigma_2)$, Eq. (14) can be written in the following form:

$$d_H(\nu, \phi) = \sqrt{1 - \frac{|\Sigma_1|^{1/4}|\Sigma_2|^{1/4}}{|\bar{\Sigma}|}\exp\left(-\frac{1}{8}\Lambda\right)}$$

where $\bar{\Sigma} = (\Sigma_1 + \Sigma_2)/2$, $\Lambda = (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^\top \bar{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$, and $|\boldsymbol{A}|$ denotes the determinant of matrix $\boldsymbol{A}$.

Consider a training dataset $\mathcal{D} = \{\boldsymbol{X}, \boldsymbol{Y}\}$ with $\boldsymbol{X} = (\boldsymbol{x}_{t_1}, \boldsymbol{x}_{t_2}, \ldots, \boldsymbol{x}_{t_N})^\top$ and $\boldsymbol{Y} = (y_{t_1}, y_{t_2}, \ldots, y_{t_N})^\top$ for constructing a GPR model to make predictions at test input points in $\boldsymbol{X}^*$. The implementation of the TDP component of the proposed framework is summarized in Algorithm 3. Because Algorithm 3 only involves evaluations of closed-form expressions, the computational cost of the TDP component is very low, and it is suitable for real-time implementation.

D. Integration of the Three Components

A diagram of the proposed online trajectory prediction framework with the three components integrated is provided in Fig. 1. At the initial time instant t_0, we first make predictions following Sec. III.A using the input point $\boldsymbol{x}_{t_0}$ and the initial datasets $\mathcal{D}^{la}_{t_0}$, $\mathcal{D}^{lo}_{t_0}$, and $\mathcal{D}^{al}_{t_0}$, and set the time instant index $i = 1$. At time instant t_i, we first observe the newly arrived position data (i.e., la_{t_i}, lo_{t_i}, and al_{t_i}) and input point $\boldsymbol{x}_{t_i}$. We subsequently implement the CPD routine as detailed in Algorithm 2, the model fitting and prediction steps given in Algorithm 1 (step 2 is skipped if refitting is unnecessary), and the TDP routine as detailed in Algorithm 3. Finally, we advance the time index by 1 and step into the next time instant. The entire procedure proceeds continually, and the three components work collectively to deliver accurate and robust trajectory predictions in an online manner.

Algorithm 3: Implementing the TDP routine for GPR training data pruning

1: Compute the predictive mean at test input points in $\boldsymbol{X}^*$ conditional on $\mathcal{D}$, denoted by $\boldsymbol{\mu}_{\mathcal{D}}$; compute the predictive covariance matrix for $\boldsymbol{X}^*$, denoted by $\Sigma_{\mathcal{D}}$.
2: **while** $\mathcal{D} \neq \varnothing$ **do**
3: **for** $i = 1, 2, \ldots, N$ **do**
4: $\mathcal{D}_{-i} = \mathcal{D} \setminus \{\boldsymbol{x}_{t_i}, y_{t_i}\}$.
5: Compute the predictive mean at test input points in $\boldsymbol{X}^*$ conditional on $\mathcal{D}_{-i}$, denoted by $\boldsymbol{\mu}_{\mathcal{D}-i}$; compute the predictive covariance matrix for $\boldsymbol{X}^*$, denoted by $\Sigma_{\mathcal{D}_{-i}}$.
6: Compute the Hellinger distance between distributions $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{D}}, \Sigma_{\mathcal{D}})$ and $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{D}-i}, \Sigma_{\mathcal{D}_{-i}})$, denoted by d^i_H.
7: **end for**
8: Find the index of the training point that has the lowest d^i_H: $i^* = \mathrm{argmin}_{1\le i\le N} d^i_H$.
9: **if** $d^{i^*}_H > h$ **then**
10: Stop.
11: **else**
12: Prune the training point: $\mathcal{D} \leftarrow \mathcal{D}_{-i^*}$.
13: Update the training sample size: $N \leftarrow N - 1$.
14: **end if**
15: **end while**

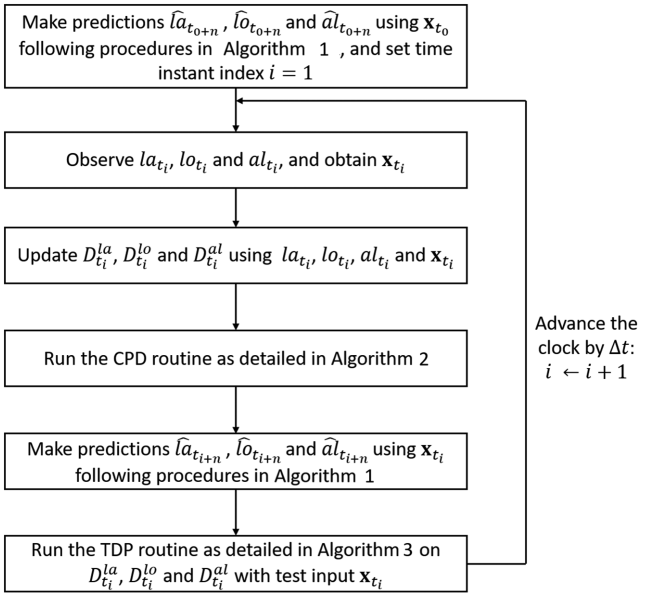

Fig. 1 A diagram of the GPR-based online trajectory prediction framework for UAVs.

IV. Numerical Experiments

In this section, we numerically evaluate the performance of the proposed GPR-based online trajectory prediction framework using a simulation dataset and a real-world dataset. Section IV.A elaborates on the experiment setup. Sections IV.B and IV.C respectively provide the corresponding results obtained from studies based on the simulation and real-world datasets.

A. Experiment Setup

1. Competing Methods

We aim to examine the effects of the three components (i.e., GPR with uniform confidence bounds, CPD, TDP) on the predictive performance and the computational efficiency achieved by the proposed framework via the numerical experiments.

For point prediction, three methods are compared: i) the proposed framework with three components: GPR with uniform confidence bounds, CPD and TDP (referred to as GUCT); ii) GPR with uniform confidence bounds only (referred to as GU), and iii) a benchmarking KF-based method for position tracking (referred to as KF). The comparison between GUCT and KF helps reveal whether the proposed framework outperforms the state-of-the-art trajectory prediction method. The comparison between GUCT and GU helps manifest whether CPD and TDP improve the computational efficiency without compromising the predictive accuracy.

For interval prediction, four methods are considered: i) GUCT; ii) GU; iii) GPR with Sidak corrected pointwise confidence bounds, CPD, and TDP (referred to as GPCT); and (iv) KF. Similar to point prediction, the comparison between GUCT and KF investigates whether the proposed framework outperforms the state-of-the-art trajectory prediction method in terms of interval prediction. The comparison between GUCT and GU helps examine the effects achieved by CPD and TDP. Moreover, the comparison between GUCT and GPCT helps compare the effects of the uniform confidence bound and the Sidak corrected pointwise confidence interval.

2. Evaluation Metrics

We focus on evaluating the predictive performance of the proposed framework GUCT in comparison with competing methods. The predictive performance is evaluated with respect to both point predictive accuracy and coverage ability of the interval estimate. Specifically, to measure the pointwise predictive accuracy achieved at time instant t, we calculate the root-mean-squared error (RMSE) defined as follows:

$$\mathrm{RMSE}_t = \sqrt{\frac{1}{H}\sum_{h=t+1}^{t+H} \|\hat{\boldsymbol{s}}_h - \boldsymbol{s}_h\|^2} \tag{15}$$

where $\|\hat{\boldsymbol{s}}_h - \boldsymbol{s}_h\|$ represents the Euclidean distance between $\hat{\boldsymbol{s}}_h$ and $\boldsymbol{s}_h$, with $\boldsymbol{s}_h = (la_h, lo_h, al_h)^\top$ denoting the true position at time h and $\hat{\boldsymbol{s}}_h$ denoting its estimate given by a candidate method. H denotes the prediction horizon, i.e., the number of future time instants to predict at time instant t. In the studies based on simulation and real-world datasets, two types of RMSEs are calculated, namely, RMSE for multistep-ahead prediction and one-step-ahead prediction, respectively corresponding to $H > 1$ and $H = 1$ in Eq. (15).

To evaluate the performance of the interval estimate given by a candidate method, we use the multistep coverage probability defined as

$$\mathrm{CP}_t = \frac{1}{H}\sum_{h=t+1}^{t+H} \mathbf{1}\{A_h\} \tag{16}$$

where A_h denotes the event that the true trajectory position at time h is covered by the $100(1-\delta)\%$ confidence interval, and $\mathbf{1}\{A_h\}$ is 1 if A_h is true and 0 otherwise. The confidence level $1-\delta$ is set to 0.95, and multistep coverage probability is calculated [i.e., $H > 1$ in Eq. (16) is adopted] in all experiments conducted. Furthermore, to evaluate the computational efficiency of a given candidate method, we record its computational time taken to make one multistep-ahead prediction.

3. Model Configurations

Recall that each component of GUCT (i.e., GPR with the uniform confidence interval, CPD, and TDP) involves specific parameters that affect its performance: the correlation function of GPR, the confidence level $1-\alpha$ of CPD, and the distance threshold h of TDP. Regarding GPR modeling, we adopt the commonly used squared exponential correlation function. Specifically, $R(\boldsymbol{x}, \boldsymbol{x}') = \exp\{-\|\boldsymbol{x} - \boldsymbol{x}'\|^2/(2l^2)\}$, where l denotes the length-scale parameter. The confidence level of CPD is set to 0.95, a common choice for hypothesis testing. The Hellinger distance between two arbitrary distributions ranges from 0 to 1, with 0 indicating most similar and 1 otherwise; we set $h = 0.2$ to ensure that the pruning does not lead to excessive loss in accuracy while improving the computational efficiency of the proposed framework.

4. Summary of Test Scenarios

We provide a summary of test scenarios considered in the two numerical studies in Table 1 to facilitate the reader's understanding. This table shows the detailed information about the respective candidate aircraft considered in the simulation and real-world studies, the characteristics of the datasets, the major settings for trajectory modeling and prediction, and the respective numerical results obtained.

B. Study Based on Simulation Data

The dataset used in this study is generated by a simulation model for a 3DR IRIS+ quadcopter unmanned aircraft with wingspan

Table 1 Summary of test scenarios in numerical studies

Parameter	Simulation study	Real-world study
UAV type	3DR IRIS + quadcopter	Fixed-wing Hobby King Bixler
Wingspan, m	0.55	1.55
Length, m	0.55	0.95
Mass, kg	1.28	1.2
Time lag, s	2	1
Input variables	Latitude, longitude, altitude	Latitude, longitude, altitude & pitch, roll, and yaw rates
Initial sample size	45	15
No. of prediction time instants	202	174
Results	Figs. 4–6, Table 2	Figs. 9–11, Table 3

0.55 m, length 0.55 m, and mass 1.28 kg [41], which is illustrated in Fig. 2.

The dataset comprises two sub-datasets. The first sub-dataset describes flight mission 1 for the target UAV that involves takeoff, cruising (flying at a fixed altitude), hovering (flying at a fixed latitude, longitude, and altitude), and landing; Fig. 3 illustrates the trajectory for flight mission 1. For ease of visualization, the trajectories of UAVs are shown in the XYZ coordinate system, where X denotes the north direction, Y denotes the east direction, and Z denotes the up direction. The second sub-dataset corresponds to flight mission 2 for the target UAV, which is similar to flight mission 1. The only difference is that flight mission 1 involves a hovering period of approximately 60 s, which is absent in flight mission 2. Hence, predicting the trajectory for flight mission 1 is arguably more challenging. The average cruising speed of the UAV in flight mission 1 and flight mission 2 is 2.73 m/s. In both sub-datasets, the input vectors used by the GPR models are in the form of $\boldsymbol{x}_{t_i} = (la_{t_i}, lo_{t_i}, al_{t_i})$. Both sub-datasets contain data collected at 146 consecutive time instants, with the time lag between adjacent time instants $\Delta t = 2$ s. Prediction takes place at the 46th time instant and onward in each sub-dataset, and the size of the initial training datasets $\mathcal{D}_{t_0}^{la}$, $\mathcal{D}_{t_0}^{lo}$, and $\mathcal{D}_{t_0}^{al}$ is 45. In total, predictions at 202 future time instants are conducted in this study. The prediction horizon for each multistep-ahead prediction is set to $H = 15$.

Figure 4 shows the boxplots of the multistep ($H = 15$) and one-step ($H = 1$) RMSEs for the 202 point predictions made by GUCT, GU, and KF; recall the definitions of the three competing methods from Sec. IV.A.1. We have the following observations:

1) GUCT and GU outperform KF with respect to both multistep-ahead and one-step-ahead predictions. This can be explained by the robustness of GUCT and GU to the impact of external forces on the target UAV's trajectory, which is absent in KF.

2) GUCT has similar multistep RMSEs and slightly higher one-step RMSEs than GU. This indicates that, if implemented correctly, incorporating CPD and TDP components into the GPR-based framework does not lead to deteriorated predictive performance.

Fig. 2 The 3DR IRIS+ quadcopter.

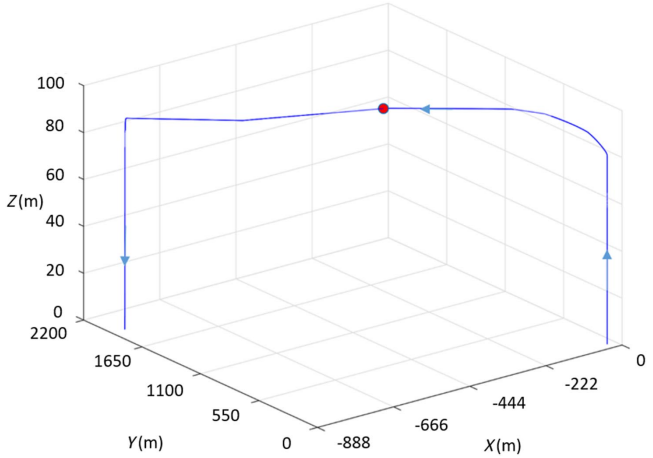

Fig. 3 An illustration of the trajectory for flight mission 1 in the study based on a simulation dataset. The solid dot indicates the position of the target UAV during the hovering period.

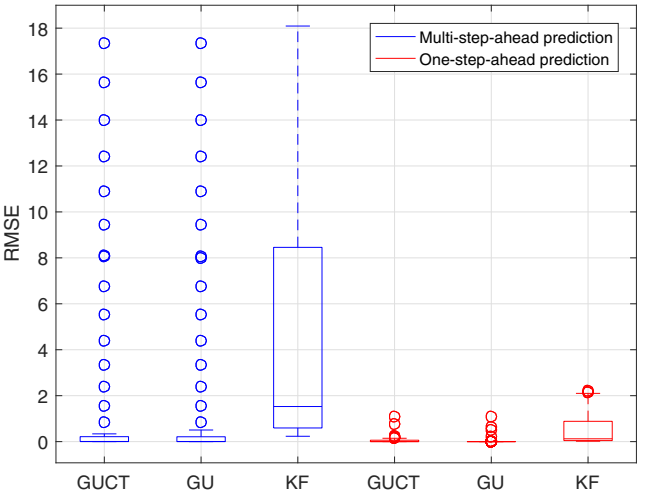

Fig. 4 Boxplots of RMSEs (unit: m) obtained respectively by GUCT, GU, and KF in the study based on the simulation dataset.

Table 2 summarizes the average multistep coverage probabilities (CPs) obtained by GUCT, GU, GPCT, and KF for making the 202 predictions; recall the definitions of the four competing methods from Sec. IV.A.1. We have the following observations. First and foremost, KF has the highest coverage ability while GPCT has the lowest one; the performance of GUCT and GU is similar and comparable to that of KF but with a slightly higher variability. This implies that the uniform confidence interval adopted by GUCT and GU is superior to the Sidak corrected pointwise confidence interval adopted by GPCT. Second, the similar performance of GUCT and GU implies that, if implemented correctly, incorporating CPD and TDP into the GPR-based framework does not lead to deteriorated interval predictions. Last but not least, we closely examine the confidence intervals given by GUCT and KF. It is observed that the confidence intervals given by KF are often unnecessarily wide, which explains their high coverage probabilities shown in Table 2. To economize on space, we only show Fig. 5 that presents the confidence intervals given by GUCT and KF associated with one-step-ahead predictions along the Z axis for flight mission 1; similar observations are made along the X and Y axes. Notice that the wide confidence intervals given by KF are much less informative than those given by GUCT and can cause problems in practice. Such wide confidence intervals given by KF may provide a false sense of security when adopted for trajectory planning purposes.

Moreover, we evaluate the effectiveness of the CPD component of the proposed framework GUCT for detecting abrupt changes in the flying pattern of the target UAV. Figure 6 illustrates the change points detected in the UAV trajectory for flight mission 1 along the Z axis. Figures obtained for the X and Y axes are similar and hence are omitted for the sake of brevity. In summary, the CPD component helps the proposed framework identify the changes taken place along each axis promptly and facilitate adaptive model refitting.

Finally, we compare the computational efficiencies of the competing methods. Specifically, the average computational times taken for making the 202 multistep-ahead predictions (with standard errors given in parentheses) by GUCT, GU, and KF are respectively $0.07(3.08 \times 10^{-4})$ s, 1.34 (0.01) s, and $3.35 \times 10^{-4}(1.35 \times 10^{-4})$ s. Thanks to the CPD and TDP components, GUCT achieves significant computational savings as compared with GU and becomes sufficiently

Table 2 Summary of the average multistep CPs with corresponding standard errors (in parentheses) obtained by GUCT, GU, GPCT, and KF in the study based on the simulation dataset

GUCT	GU	GPCT	KF
$1.000(3.30 \times 10^{-4})$	$1.000(3.30 \times 10^{-4})$	0.9(0.01)	1(0)

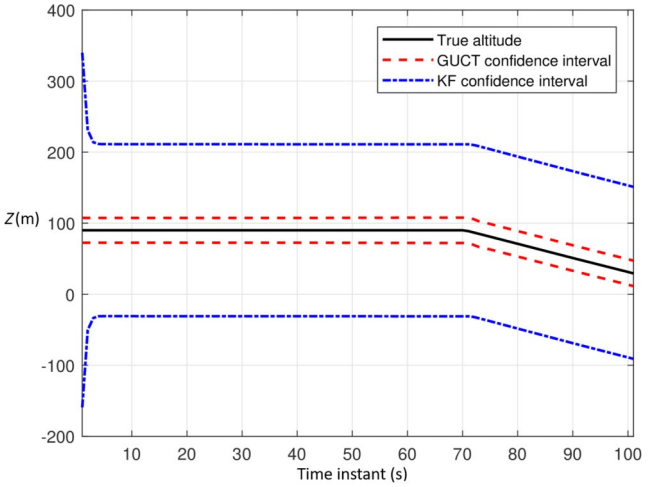

Fig. 5 Confidence intervals respectively given by GUCT and KF associated with one-step-ahead predictions along the Z axis for flight mission 1 in the study based on the simulation dataset.

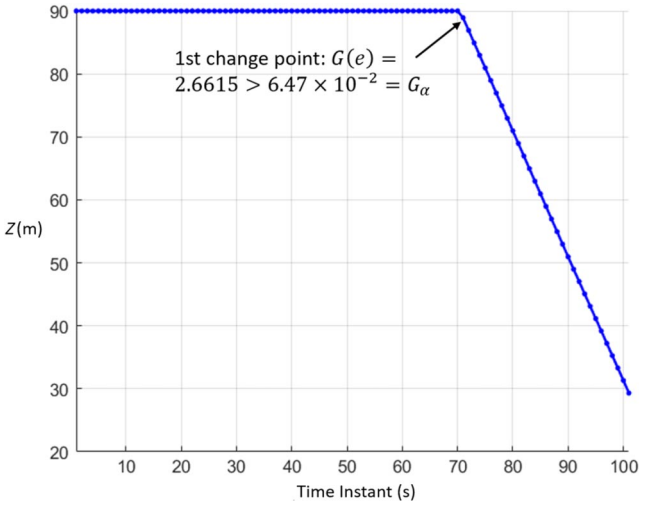

Fig. 6 Change points in the target UAV's trajectory along the Z axis identified by the CPD component of GUCT for flight mission 1 in the study based on the simulation dataset.

fast for real-world implementation. KF is the most computationally efficient, nevertheless at the cost of predictive performance as evidenced by Figs. 4 and 5.

C. Study Based on Real-World Data

The real-world dataset used in this study contains position data sampled by GPS sensors and pose data sampled by inertial measurement units (IMU) [42]. The dataset contains 14 individual sub-datasets; the data in each sub-dataset were collected for a host UAV and a target UAV during the course of a flight mission. Both UAVs are fixed-wing HobbyKing Bixler unmanned aircraft with wingspan 1.55 m, length 0.95 m, and mass 1.2 kg, which are illustrated in Fig. 7.

In the real-world environment, the UAVs fly under the wind disturbances. The actual wind speed in the north direction is 1.68 m/s and that in the east direction is -1.49 m/s. The average cruising speed of the UAVs is 10 m/s. Because the logging frequencies of the GPS and IMU sensors are different, to synchronize the data, we only retain the position data and pose data (i.e., pitch rate, roll rate, and yaw rate) at the common time instants for evaluation purposes. As a result, each sub-dataset contains observations at approximately 30 time instants, with $\Delta t = 1$ s between adjacent time instants. Prediction takes place for the target UAV at the 16th time instant and onward

Fig. 7 Fixed-wing HobbyKing Bixler.

in each sub-dataset, and the size of the initial training datasets $\mathcal{D}_{t_0}^{la}$, $\mathcal{D}_{t_0}^{lo}$, and $\mathcal{D}_{t_0}^{al}$ is 15. In total, predictions at 174 future time instants are to be made for the target UAV. The prediction horizon for multistep-ahead prediction is set to $H = 5$. In all sub-datasets, the input vectors are in the form of $\boldsymbol{x}_{t_i} = (la_{t_i}, lo_{t_i}, al_{t_i}, p_{t_i}, r_{t_i}, y_{t_i})$, where p_{t_i}, r_{t_i}, and y_{t_i} denote the pitch rate, the roll rate, and the yaw rate at time t_i, respectively. Figure 8 shows the target UAV's trajectory in one arbitrarily chosen flight mission (referred to as flight mission 1 hereinafter) in the XYZ coordinate system. We note that compared with the simulation dataset considered in Sec. IV.B, the trajectories studied here show highly complex flying patterns, which involve more abrupt changes.

Figure 9 shows the boxplots of the multistep ($H = 5$) and one-step ($H = 1$) RMSEs obtained by GUCT, GU, and KF for 174 prediction time instants. We have similar observations as those made in Sec. IV.B: 1) GUCT and GU outperform KF with respect to both multistep-ahead and one-step-ahead predictive accuracies achieved; 2) GUCT has comparable multistep and one-step RMSEs as those of GU. This indicates that, if implemented correctly, the CPD and TDP components do not compromise the point predictive accuracy achieved by the proposed framework.

Table 3 summarizes the average multistep coverage probabilities obtained by GUCT, GU, GPCT, and KF for making the 174 predictions. We have the following observations. First, KF has the highest coverage ability followed by GUCT and GU, whereas GPCT has the lowest coverage ability. This implies that the uniform confidence interval adopted by GUCT and GU is superior to the Sidak corrected pointwise

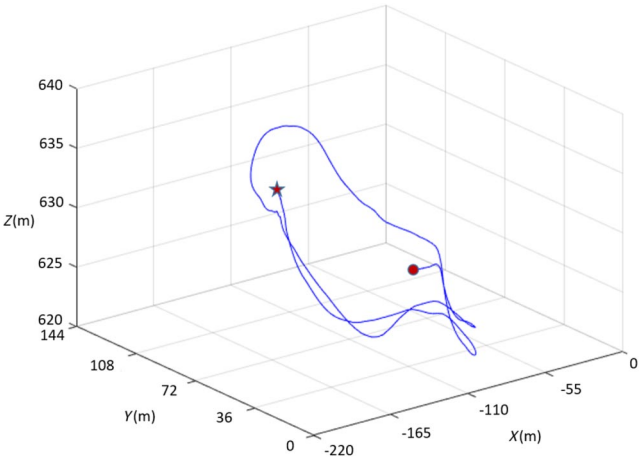

Fig. 8 An illustration of the trajectory of the target UAV for flight mission 1 in the study based on a real-world dataset. The solid dot shows the starting position, and the star shows the ending position of the trajectory.

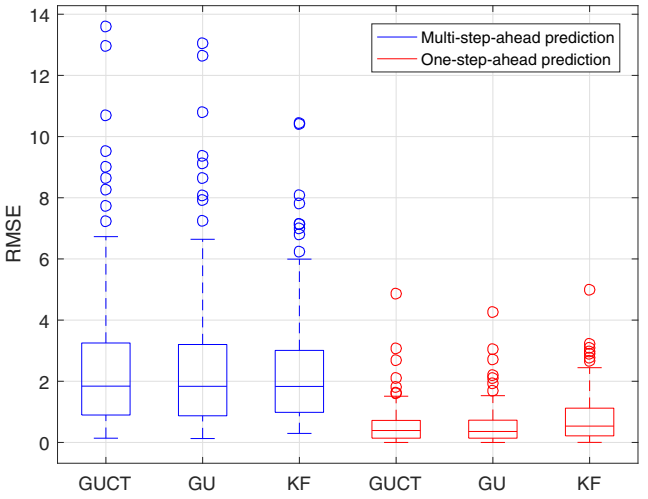

Fig. 9 Boxplots of RMSEs (unit: m) obtained respectively by GUCT, GU, and KF in the study based on the real-world dataset.

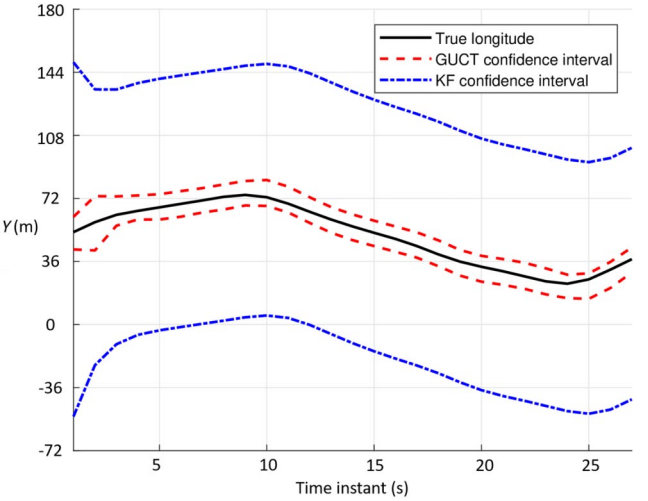

Fig. 10 Confidence intervals respectively given by GUCT and KF associated with one-step-ahead predictions along the Y axis for flight mission 1 in the study based on the real-world dataset.

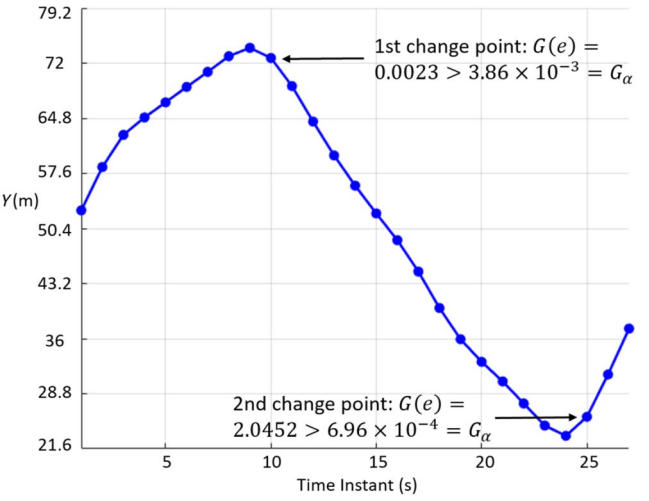

Fig. 11 Change points in the target UAV's trajectory along the Y axis identified by the CPD component of GUCT for flight mission 1 in the study based on the real-world dataset.

confidence interval adopted by GPCT. Second, GUCT has similar coverage probabilities as GU. This indicates that, if implemented correctly, CPD and TDP do not lead to deteriorated interval predictions. Lastly, we compare the performance of GUCT and KF in terms of their respective confidence intervals associated with one-step-ahead predictions along the X, Y, and Z axes for flight mission 1. We find again that the confidence intervals given by KF are often much wider and hence less informative than those given by GUCT, despite the high coverage probabilities achieved by the former. Such wide confidence intervals may cause problems in practice as noted in Sec. IV.B. For the sake of brevity, we only show Fig. 10 that presents the confidence intervals given by GUCT and KF along the Y axis for flight mission 1; similar observations are made regarding confidence intervals obtained along the X and Z axes.

We examine the effectiveness of the CPD component of GUCT in detecting abrupt changes in the target UAV's flying patterns. Figure 11 shows the trajectory of the target UAV along the Y axis and the change points detected in flight mission 1. Figures obtained for the X and Z axes are similar and hence are omitted for the sake of brevity. Once again, we see that the CPD component can accurately identify the change points that occurred in the target UAV's trajectory.

We make some final comments regarding the computational efficiencies of the competing methods. The average computational times taken for making the 174 multistep-ahead predictions (with standard errors given in parentheses) by GUCT, GU, and KF are respectively 0.08(0.03) s, 0.92 (0.01) s, and $3.94 \times 10^{-4}(1.16 \times 10^{-4})$ s. We observe again that GUCT achieves significant computational savings as compared with GU thanks to its CPD and TDP components. KF is the most computationally efficient, however, at the expense of predictive performance as evidenced by Figs. 9 and 10.

V. Conclusions

This paper proposed a GPR-based trajectory prediction framework for online implementation, i.e., GUCT. GUCT features three novel components, respectively: uniform confidence bounds for simultaneous predictive uncertainty quantification, change-point detection for monitoring abrupt changes in flying patterns, and TDP for enhancing the computational efficiency of GPR. Interested in the challenging setting of trajectory prediction for non-cooperative UAVs, the authors tested the performance of GUCT with both simulation and real-world datasets. The proposed framework was found to not only outperform competing methods in terms of point and interval prediction performance but also achieve a desirable computational efficiency in practice. Moreover, GUCT can effectively identify abrupt changes in the flight trajectory of the target UAV and adapt to new flying patterns promptly. The proposed framework GUCT is expected to be useful for online analysis and control of other types of dynamic data-driven application systems where robust and fast predictions are critical.

Table 3 Summary of the average multistep CPs with corresponding standard errors (in parentheses) obtained by GUCT, GU, GPCT, and KF in the study based on the real-world dataset

GUCT	GU	GPCT	KF
$1.00(10^{-3})$	$1.00(10^{-3})$	0.16(0.012)	1(0)

Acknowledgments

The authors thank Craig Woolsey and Andrea L'Afflitto at Virginia Tech for discussions and comments on the paper. The work of the authors was supported by the National Science Foundation (grant numbers IIS-1849300 and CMMI-1846663).

References

[1] Yang, D., Wu, Q., Zeng, Y., and Zhang, R., "Energy Tradeoff in Ground-to-UAV Communication via Trajectory Design," *IEEE Transactions on*

Vehicular Technology, Vol. 67, No. 7, 2018, pp. 6721–6726.
https://doi.org/10.1109/TVT.2018.2816244

[2] Zhou, X., Wu, Q., Yan, S., Shu, F., and Li, J., "UAV-Enabled Secure Communications: Joint Trajectory and Transmit Power Optimization," *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 4, 2019, pp. 4069–4073.
https://doi.org/10.1109/TVT.2019.2900157

[3] Yoon, J., Lee, A., and Lee, H., "Rendezvous: Opportunistic Data Delivery to Mobile Users by UAVs through Target Trajectory Prediction," *IEEE Transactions on Vehicular Technology*, Vol. 69, No. 2, 2020, pp. 2230–2245.
https://doi.org/10.1109/TVT.2019.2962391

[4] Aurambout, J.-P., Gkoumas, K., and Ciuffo, B., "Last Mile Delivery by Drones: An Estimation of Viable Market Potential and Access to Citizens across European Cities," *European Transport Research Review*, Vol. 11, No. 30, 2019, pp. 1–21.
https://doi.org/10.1186/s12544-019-0368-2

[5] Ren, L., Castillo-Effen, M., Yu, H., Johnson, E., Nakamura, T., Yoon, Y., and Ippolito, C. A., "Small Unmanned Aircraft System (sUAS) Trajectory Modeling in Support of UAS Traffic Management (UTM)," *17th AIAA Aviation Technology, Integration, and Operations Conference*, AIAA Paper 2017-4268, 2017, pp. 1–17.
https://doi.org/10.2514/6.2017-4268

[6] Lin, C. E., and Lai, Y. H., "UAV Path Prediction for CD&R to Manned Aircraft in a Confined Airspace for Cooperative Mission," *International Journal of Aerospace Engineering*, Vol. 1, March 2018, pp. 1–9.
https://doi.org/10.1155/2018/8759836

[7] Kang, C., and Woolsey, C. A., "Model-Based Path Prediction for Fixed-Wing Unmanned Aircraft Using Pose Estimates," *Aerospace Science and Technology*, Vol. 105, Oct. 2020, pp. 1–11.
https://doi.org/10.1016/j.ast.2020.106030

[8] Tyagi, A., and Nanda, J., "ATLAS: Big Data Storage and Analytics Tool for ATM Researchers," *AIAA Infotech @ Aerospace, AIAA SciTech Forum*, AIAA Paper 2016-0577, 2016, pp. 1–14.
https://doi.org/10.2514/6.2016-0577

[9] Slattery, R., and Zhao, Y., "Trajectory Synthesis for Air Traffic Automation," *Journal of Guidance, Control, and Dynamics*, Vol. 20, No. 2, 1997, pp. 232–238.
https://doi.org/10.2514/2.4056

[10] Vilardaga, S., and Prats, X., "Mass Estimation for an Adaptive Trajectory Predictor Using Optimal Control," *Proceedings of the 5th International Conference on Application and Theory of Automation in Command and Control Systems*, Assoc. for Computing Machinery, New York, 2015, pp. 75–84.
https://doi.org/10.1145/2899361.2899369

[11] Ayhan, S., and Samet, H., "Aircraft Trajectory Prediction Made Easy with Predictive Analytics," *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Assoc. for Computing Machinery, New York, 2016, pp. 21–30.
https://doi.org/10.1145/2939672.2939694

[12] Conde, R., Cobano, J. A., and Ollero, A., "Method Based on A Particle Filter for UAV Trajectory Prediction Under Uncertainties," *Proceedings of the 40th International Symposium on Robotics*, 2009, pp. 245–250.

[13] Hammer, M., Hebel, M., Laurenzis, M., and Arens, M., "Lidar-Based Detection and Tracking of Small UAVs," *Emerging Imaging and Sensing Technologies for Security and Defence III; and Unmanned Sensors, Systems, and Countermeasures*, SPIE Digital Library, Berlin, Germany, 2018, pp. 107990S-1–107990S-9.
https://doi.org/10.1117/12.2325702

[14] Wu, Z., Li, J., Zuo, J., and Li, S., "Path Planning of UAVs Based on Collision Probability and Kalman Filter," *IEEE Access*, Vol. 6, July 2018, pp. 34,237–34,245.
https://doi.org/10.1109/ACCESS.2018.2817648

[15] Sakthivel, C., Sureshkumar, B., Ramkumar, R., and Yokeswaran, R., "Detection and Path Prediction of Aircraft Based on Acoustics and Vibration," *Materials Today: Proceedings*, Vol. 21, Sept. 2020, pp. 588–591.
https://doi.org/10.1016/j.matpr.2019.06.720

[16] Zhang, J., Liu, R. H. J., and Zhu, H., "Online Four Dimensional Trajectory Prediction Method Based on Aircraft Intent Updating," *Aerospace Science and Technology*, Vol. 77, June 2018, pp. 774–787.
https://doi.org/10.1016/j.ast.2018.03.037

[17] Xiao, K., Zhao, J., He, Y., and Yu, S., "Trajectory Prediction of UAV in Smart City using Recurrent Neural Networks," *Proceedings of the 2019 IEEE International Conference on Communications*, IEEE, New York, 2019, pp. 1–6.
https://doi.org/10.1109/ICC.2019.8761110

[18] Shi, Z., Xu, M., Pan, Q., Yan, B., and Zhang, H., "LSTM-Based Flight Trajectory Prediction," *Proceedings of the 2018 International Joint Conference on Neural Networks*, IEEE, New York, 2018, pp. 1–8.
https://doi.org/10.1109/IJCNN.2018.8489734

[19] Corbetta, M., Banerjee, P., Okolo, W., Gorospe, G., and Luchinsky, D. G., "Real-Time UAV Trajectory Prediction for Safety Monitoring in Low-Altitude Airspace," *AIAA Aviation 2019 Forum*, AIAA Paper 2019-3514, 2019.
https://doi.org/10.2514/6.2019-3514

[20] Rasmussen, C. E., and Williams, C. K. I., *Gaussian Processes for Machine Learning*, MIT Press, Cambridge, MA, 2006, Chap. 2.

[21] Santner, T. J., Williams, B. J., and Notz, W. I., *The Design and Analysis of Computer Experiments*, 2nd ed., Springer–Verlag, New York, 2018, Chap. 2.
https://doi.org/10.1007/978-1-4939-8847-1

[22] Stein, M. L., *Interpolation of Spatial Data: Some Theory for Kriging*, Springer–Verlag, New York, 1999, Chap. 1.5.
https://doi.org/10.1007/978-1-4612-1494-6

[23] Chen, X., ankenman, B. E., and Nelson, B. L., "The Effects of Common Random Numbers on Stochastic Kriging Metamodels," *ACM Transactions on Modeling and Computer Simulation*, Vol. 2, No. 2, pp. 7-1–7-20.
https://doi.org/10.1145/2133390.2133391

[24] Brabanter, K. D., Brabanter, J. D., and Suykens, J. A., "Approximate Confidence and Prediction Intervals for Least Squares Support Vector Regression," *IEEE Transactions on Neural Networks*, Vol. 22, No. 1, 2011, pp. 110–120.
https://doi.org/10.1109/TNN.2010.2087769

[25] Kawahara, Y., and Sugiyama, M., "Sequential Change-Point Detection Based on Direct Density-Ratio Estimation," *SIAM International Conference on Data Mining*, John Wiley & Sons, Inc., New York, 2009, pp. 389–400.
https://doi.org/10.1002/sam.10124

[26] Saatci, Y., Turner, R. D., and Rasmussen, C. E., "Gaussian Process Change Point Models," *Proceedings of the 27th International Conference on Machine Learning*, Omnipress, Madison, WI, 2010, pp. 927–934.

[27] Garnett, R., Osborne, M. A., Reece, S., Rogers, A., and Roberts, S. J., "Sequential Bayesian Prediction in the Presence of Changepoints and Faults," *Computer Journal*, Vol. 53, No. 9, 2010, pp. 1430–1446.
https://doi.org/10.1093/comjnl/bxq003

[28] Aminikhanghahi, S., and Cook, D. J., "A Survey of Methods for Time Series Change Point Detection," *Knowledge and Information Systems*, Vol. 51, No. 2, 2017, pp. 339–367.
https://doi.org/10.1007/s10115-016-0987-z

[29] Basseville, M., and Nikiforov, I. V., *Detection of Abrupt Change Theory and Application*, Prentice–Hall, Upper Saddle River, NJ, 1993, Chap. 2.

[30] Quiñonero-Candela, J., and Rasmussen, C. E., "A Unifying View of Sparse Approximate Gaussian Process Regression," *Journal of Machine Learning Research*, Vol. 6, No. 65, 2005, pp. 1939–1959.

[31] Snelson, E., and Ghahramani, Z., "Local and Global Sparse Gaussian Process Approximations," *Proceedings of the 11th International Conference on Artificial Intelligence and Statistics*, Omnipress, Madison, WI, 2007, pp. 1–8.

[32] Wang, Y., and Li, H., "A Novel Intelligent Modeling Framework Integrating Convolutional Neural Network with an Adaptive Time-Series Window and Its Application to Industrial Process Operational Optimization," *Chemometrics and Intelligent Laboratory Systems*, Vol. 179, Aug. 2018, pp. 64–72.
https://doi.org/10.1016/j.chemolab.2018.06.008

[33] Noor, M. H. M., Salcic, Z., and Wang, K. I.-K., "Adaptive Sliding Window Segmentation for Physical Activity Recognition Using a Single Tri-Axial Accelerometer," *Pervasive and Mobile Computing*, Vol. 38, July 2017, pp. 41–59.
https://doi.org/10.1016/j.pmcj.2016.09.009

[34] Dalmazo, B. L., Vilela, J. P., and Curado, M., "Online Traffic Prediction in the Cloud," *International Journal of Network Management*, Vol. 26, No. 4, 2016, pp. 269–285.
https://doi.org/10.1002/nem.1934

[35] Khan, I. A., Akber, A., and Xu, Y., "Sliding Window Regression Based Short-Term Load Forecasting of a Multi-Area Power System," *Proceedings of the 2019 American Control Conference*, IEEE, New York, 2019, pp. 1–5.
https://doi.org/10.1109/CCECE.2019.8861915

[36] Baig, S., Iqbal, W., Berral, J. L., and Carrera, D., "Adaptive Sliding Windows for Improved Estimation of Data Center Resource Utilization," *Future Generation Computer Systems*, Vol. 104, March 2020, pp. 212–224.
https://doi.org/10.1016/j.future.2019.10.026

[37] Blakesley, R. E., Mazumdar, S., Dew, M. A., Houck, P. R., Tang, G., Reynolds, C. F., and Butters, M. A., "Comparisons of Methods for Multiple Hypothesis Testing in Neuropsychological Research,"

Neuropsychology, Vol. 23, No. 2, 2009, pp. 255–264. https://doi.org/10.1037/a0012850

[38] Lederer, A., and Umlauft, J., "Uniform Error Bounds for Gaussian Process Regression with Application to Safe Control," *Proceedings of the 33rd Conference*, Neural Information Processing Systems Foundation, Inc., La Jolla, CA, 2019, pp. 1–17.

[39] Fazai, R., Mansouri, M., Abodayeh, K., Puig, V., Raouf, M.-I. N., Nounou, H., and Nounou, M., "Multiscale Gaussian Process Regression-Based Generalized Likelihood Ratio Test for Fault Detection in Water Distribution Networks," *Engineering Applications of Artificial Intelligence*, Vol. 85, Oct. 2019, pp. 474–491. https://doi.org/10.1016/j.engappai.2019.07.007

[40] Koppel, A., "Consistent Online Gaussian Process Regression Without the Sample Complexity Bottleneck," *Proceedings of the 2019 American Control Conference*, IEEE, New York, 2019, pp. 3512–3518. https://doi.org/10.23919/ACC.2019.8815206

[41] "Simulation Dataset," Intelligent Automation, Inc., 2019, https://iainorthamerica.com/.

[42] Kang, C., and Woolsey, C., "Real-World Datasets," 2017, https://sites.google.com/vt.edu/safe-repository/.

D. Allaire
Associate Editor