Mapping Urban Wind Fields via Gaussian Processes Regression Models that Consider Building Morphology

Nicholas Kakavitsas*, Mesbah Uddin†, and Artur Wolek‡
University of North Carolina at Charlotte, Charlotte, North Carolina, 28223

This paper presents a framework for training a Gaussian process (GP) to estimate a steady urban wind field from a sparse set of wind measurements by leveraging training data collected from computational fluid dynamics (CFD) simulations. Gaussian process models for spatial estimation often use measurement locations as the input space with proximity-based covariance functions. This work investigates including building morphology features into the GP model that are defined by the signed distance field (SDF) and its gradient evaluated at a pattern of points around each sample location. Augmenting the measurement locations with different subsets of building morphology features leads to unique feature spaces. Several different GP models are trained using various feature spaces and covariance functions, including with a coregion covariance function that allows simultaneous training over multiple CFD datasets for different urban geometries. A framework is developed to generate CFD wind field data for a set of randomized geometries, build various feature spaces, and perform the estimation with the proposed GP models. The framework is evaluated with a simple environment that consists of two buildings with randomized position and geometry in a wind field with constant inflow magnitude and direction. Results are presented comparing the estimation performance across different GP models with an increasing number of optimization iterations. The computation versus accuracy trade-off of using hyperparameters trained over multiple similar prior CFD datasets, rather than hyperparameters that are optimized on-the-fly, over a single dataset is also demonstrated.

I. Nomenclature

$f(\cdot)$	=	random scalar function
$\boldsymbol{x}$	=	vector of n features
F	=	feature space over which $\boldsymbol{x}$ is defined
$\mu(\boldsymbol{x})$	=	mean function of a Gaussian process
$\mathbb{E}[\cdot]$	=	expected value operator
$\kappa(\boldsymbol{x}, \boldsymbol{x}')$	=	covariance/kernel function of a Gaussian process
$\boldsymbol{h}$	=	vector of spatial lag terms for the squared exponential kernel
$\boldsymbol{L}$	=	vector of length scales associated with respective elements in $\boldsymbol{x}$
$\boldsymbol{\theta}_{\text{sq}}$	=	vector of hyperparameters for the squared exponential kernel ; σ , $\boldsymbol{L}$
σ^2	=	variance of Gaussian process
ε	=	zero-mean Gaussian measurement noise with variance σ_{n}^2
$\boldsymbol{y}$	=	noisy wind vector measurement
$\boldsymbol{R}$	=	vector of M points where data is collected
$\boldsymbol{Y}$	=	vector of M observations collected at $\boldsymbol{R}$
$\boldsymbol{X}$	=	vector of M feature vectors at which observations $\boldsymbol{Y}$ are collected
$\boldsymbol{G}$	=	grid of G prediction points
$\hat{\boldsymbol{Y}}(\boldsymbol{G};\ \boldsymbol{X}, \boldsymbol{Y})$	=	estimate of process at locations $\boldsymbol{G}$ given data $\boldsymbol{X}$ and $\boldsymbol{Y}$
$\boldsymbol{P}_{\hat{\boldsymbol{Y}}}(\boldsymbol{G};\ \boldsymbol{X}, \boldsymbol{Y})$	=	covariance matrix of the estimate $\hat{\boldsymbol{Y}}$
$\boldsymbol{K}(\cdot, \cdot)$	=	matrix of covariance values relating observation and/or grid points

*Graduate Student, Department of Mechanical Engineering and Engineering Science, nkakavit@charlotte.edu, AIAA Student Member.
†Professor, Department of Mechanical Engineering and Engineering Science, muddin@charlotte.edu, AIAA Senior Member.
‡Assistant Professor, Department of Mechanical Engineering and Engineering Science, awolek@charlotte.edu, AIAA Senior Member.

$\boldsymbol{I}$	=	identity matrix of given size
Q	=	number of datasets
$\boldsymbol{Z}$	=	inducing points for sparse variational Gaussian process model
$\boldsymbol{r}$	=	point in 3D space
$\rho(\boldsymbol{r})$	=	signed distance field (SDF)
$\nabla\rho(\boldsymbol{r})$	=	signed distance field (SDF) gradient
$\mathcal{I}$	=	inertial reference frame with origin P and orthonormal unit vectors $\{\boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3\}$
D	=	urban environment with length/width l_{env} and height h_{env}
$[h_{\min}, h_{\max}]$	=	range of possible heights for randomized buildings
A_i	=	base footprint of building i
O	=	volume of space occupied by a set of N_O obstacles in D
$\boldsymbol{w}(\boldsymbol{r})$	=	$[u(\boldsymbol{r}), v(\boldsymbol{r})]$ wind field component vector at $\boldsymbol{r}$
z_{des}	=	desired constant altitude for estimation
$\tilde{(\cdot)}$	=	noisy measurement of $(\cdot)$
E_u, E_v	=	RMSE error for the estimated components of $\boldsymbol{w}(\boldsymbol{r})$
ME	=	mapping error across prediction points
r	=	radius of mesh refinement zone

II. Introduction

Uncrewed aerial systems (UAS) and urban air mobility (UAM) vehicles operating within urban environments can experience adverse wind conditions and turbulence that affects flight stability and control [1, 2]. Estimating real-time wind conditions in an urban space can inform path planning to reduce risk; however, urban wind fields are spatially complex and difficult to model or estimate. Computational fluid dynamics (CFD) simulations can provide urban wind flow models for flight simulation [1–10]. However, due to the computational complexity, CFD-based solutions are not well suited for rapid estimation. Machine learning (ML) techniques have been suggested to estimate wind fields using measurements taken by either a vehicle operating in that environment [11, 12], or aided by pre-installed wind sensing infrastructure [13, 14]. Among ML techniques, Gaussian process (GP) based spatial interpolation has been used for UAS in windy urban environments [12, 15–19].

This paper presents preliminary work towards the development of an urban wind field mapping framework that leverages the realism of CFD simulations with the efficiency of GP spatial interpolation. A simplified urban wind field environment that consists of randomly generated buildings in a steady flow-field is studied. For each building geometry, the signed distance function (SDF) is computed to encode the distance to the nearest building. The gradient of the SDF provides information concerning the overall geometry of nearby buildings. The building geometry is then used for CFD simulation using OpenFOAM [20] with a constant inflow velocity, to build a database of training data. Two independent GP models are trained to predict the components of the flow velocity in the plane. Typically, spatial positions of measurements are used to define the input feature space of a GP. This work evaluates whether incorporating SDF data into the feature vector can yield improvements in estimation performance. This evaluation is motivated by the intuition that measurements with similar relative position to nearby buildings (i.e., similar SDF and SDF gradient values) may be more highly correlated than pairs of points with dissimilar SDF values. The GPflow [21] python library is used to train a standard GP regression (GPR) and a variational GP regression (VGP) model by optimizing the hyperparameters of the models to best predict the wind field given the feature input vectors. The models are optimized for an increasing number of training iterations. Various combinations of the standard squared exponential kernel and a coregion kernel are used for comparison. The approach could potentially be used to assimilate measurements from urban wind sensing infrastructure and airborne platforms to produce approximate urban wind field maps for flight path planning and re-routing.

A. Contributions

The contributions of this paper are (1) an efficient framework to train GP models with CFD generated wind field data using open-source tools, and (2) the evaluation of novel kernel formulations that incorporate building morphology data into GP models for prediction of urban wind fields.

B. Paper Organization

The remainder of the paper is organized as follows. Section III presents related work. Section IV presents preliminaries related to GP estimation and the SDF. Section V formulates the problem of sampling an uncertain wind field, sampling building morphology, and describes the optimization of the GP and VGP models and their respective kernels. Section VI presents the training framework. Section VII provides results for training the GP models with various feature spaces. The paper is concluded in Section VIII.

III. Related Work

This section highlights prior work on CFD modeling of urban wind fields and the use of machine learning for wind estimation, focusing on work that supports UAS flight simulation, path planning, or control.

A. CFD for Modeling Building Flows and UAS Simulation

Computational fluid dynamics (CFD) techniques have been widely used for modeling urban flow fields, including individual tall buildings [3–5], non-convex building geometries [6] and dense urban centers, with [1, 7, 8] and without [2, 9, 10] detailed terrain models. Some studies have investigated the effects of winds in the atmospheric boundary layer (ABL) on UAS flight [22–25]. In simulating urban wind fields, many prior works use large eddy simulations (LES) for atmospheric boundary layer studies [22], single building studies [3–6, 26], and for dense urban center studies [1, 8], to examine the high-level effects of wind flow around urban geometries. The effects of building geometry on turbulence in an urban environment was explored in [2], showing that building height and surface roughness can affect the generation of turbulent updrafts and downdrafts. Other work has used Reynolds-Averaged-Navier-Stokes (RANS) models for single building [5, 27] and dense urban center studies [2, 7, 9, 27–29]. In [5] the effects of both LES and RANS methods on the flight of a simulated quadrotor are compared. In [9] RANS results are validated against real-world wind measurements. Reference [30] provides a critical review of wind flow estimation and forecasting techniques used in the UAM community, comparing commonly applied wind flow models from wind engineering and atmospheric science, and offers an overview of urban wind flow conditions. RANS and LES methods are highlighted for wind data validation, with RANS models identified as well suited for synthetic wind database generation. In [31], urban wind field data was generated experimentally using a model of a city in a wind tunnel. The work in [26, 27, 29] shows that $k-\epsilon$ turbulence modeling is an acceptable choice in CFD studies of urban geometries, however, there are counterarguments as well. Some studies [32] have shown that the $k-\omega-$ SST turbulence model developed in [33–35], with calibrated model closure coefficients, performs better in predicting wind flow around isolated and urban buildings. Nevertheless, the $k-\omega$ model's sensitivity to free-stream conditions can affect its accuracy in certain scenarios. While the $k-\epsilon$ model has deficiencies in resolving separated and near-wall flows, it was generally more robust, less sensitive to free-stream conditions, and well suited for predicting free-shear flows. The choice between these two approaches remains subjective and dependent on user preferences and specific application requirements.

B. Machine Learning Methods for Wind Interpolation

Machine learning (ML) methods such as Gaussian process (GP) regression have been used for predicting wind flows using various kernels such as the standard squared exponential [15, 17, 19, 36], and radial basis kernel functions [18]. Long-short-term-memory (LSTM) neural networks have also been proposed to predict the time evolution of a CFD wind field [37] and then train a recurrent neural network (RNN) to predict the flow [4]. In [38], convolutional neural networks (CNNs) were studied to model both large and small scale flows through an urban environment using a cascaded approach of two models tuned for the different spatial scales; the results were comparable to CFD while being over thirty times faster to compute. In [16], two GP regression based models were presented that provide uncertainty predictions for flow fields with low and high variability. GP regression has also been shown to aid in real-time on-board wind estimates in [19], while using an anemometer to sample the wind. A neural-network model was trained on CFD generated models of wind flow over realistic terrains in [23] and generated comparable results in a fraction of the required computational time.

Prior work has used GP regression to provide wind estimates in urban environments [17, 19] and in the atmospheric boundary layer [15, 23] where the spatial regression uses relative distance to correlate samples. In [39], a GP regression model estimated wind magnitude around buildings in a realistic urban environment by training the hyperparameters of a squared exponential kernel using an evolutionary optimization algorithm. In [17], a GP regression model was trained to estimate a wind field based on probe measurements taken by a UAV using a custom kernel that was parameterized

by spatial distance and wind drift. In [19], two GP regression models were used to estimate the wind magnitude and direction respectively, by training the models on 35 sets of CFD data with varied inlet conditions. Reference [19] generated CFD training data using the open-source CFD software OpenFOAM with a $k-\epsilon$ turbulence model. A coregion kernel was used with a GP regression model in [40] to train a model on the correlation between commonly monitored parameters at 58 different pollutant monitoring stations, which was then applied to estimate the pollutant levels at 5 unseen locations on the same day. In [16], a GP regression model was paired with a squared exponential kernel to estimate ocean currents. The hyperparameters used in the model presented in [16] were trained on subsets of training data, then they were averaged together for use in an estimate, making training on a large set of data for a given day more efficient. Reference [39] generated accurate high-fidelity CFD based training data using a $k-\epsilon$ turbulence model for generalized wind conditions in winter and summer seasons for a particular set of buildings. In [41], an estimate of the pedestrian scale wind flow around a single building was generated using a GP regression model with a squared exponential kernel. The hyperparameters of the GP regression model in [41] were optimized using a genetic algorithm for 150 CFD simulations of the building with varied dimensions and orientations.

C. Relation to Prior Work

This work explores using GP regression and variational GP regression models paired with the squared exponential kernel and a coregion kernel to estimate an urban wind field using sparse observations. A novel aspect of our approach is the inclusion of building morphology information via the SDF, in addition to sample location, to define the feature space of the GP regression. Additionally, we demonstrate the use of a coregion kernel to simultaneously train GP models on multiple sets of urban CFD datasets. Without the use of such a kernel the training of a model on multiple different CFD datasets would be more difficult.

IV. Preliminaries

This section describes mathematical preliminaries of standard GP regression (GPR) and variational GP (VGP) models, which are used later in the urban wind estimation framework. The concept of a signed distance function to characterize building morphology is also introduced.

A. Gaussian Process (GP) Models

1. Gaussian Process Regression (GP)

A GP is a random scalar function $f(\boldsymbol{x})$ defined over a feature space $F \subseteq \mathbb{R}^n$ that is characterized by a mean $\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})]$ and covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f(\boldsymbol{x}) - \mu(\boldsymbol{x}))(f(\boldsymbol{x}') - \mu(\boldsymbol{x}'))]$, where $\mathbb{E}[\cdot]$ is the expected value operator, and $\boldsymbol{x}, \boldsymbol{x}' \in F$ are two feature vectors [42]. The covariance function $\kappa(\boldsymbol{x}, \boldsymbol{x}')$ describes the correlation between two feature vectors. In shorthand, a GP is often denoted as

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}')) \ . \tag{1}$$

In many spatial estimation contexts, the feature space (sometimes called the input space) F is the 2D or 3D Euclidean space in which measurements are obtained. This work investigates alternative feature spaces that augment the traditional Euclidean input space with additional variables representing the geometry of an urban environment around each sample. Different feature elements may have different correlations, thus a unique set of hyperparameters is associated with each dimension. The standard squared exponential kernel

$$\kappa_{\text{sq}}(\boldsymbol{h},\ \boldsymbol{\theta}_{\text{sq}}) = \sigma^2 \exp\left(-\frac{1}{2}\sum_{i=1}^{n}\left(\frac{h_i}{L_i}\right)^2\right) \ , \tag{2}$$

is adopted here, where $\boldsymbol{h} = [h_1, \ldots, h_n]$ is a vector of spatial lag terms, i.e., $h_i = ||x_i - x_i'||$ where x_i and x_i' are the components of $\boldsymbol{x}, \boldsymbol{x}' \in F$, $\boldsymbol{L} = [L_1, L_2, \ldots, L_n] \in \mathbb{R}^n$ is a vector describing the approximate length scale of correlation for each feature vector dimension, and σ^2 is an overall variance of the random process. The hyperparameter vector is $\boldsymbol{\theta}_{\text{sq}} = [\sigma, \boldsymbol{L}]$. A sensor measures the value of the random function at points $\boldsymbol{x}$ as

$$y = f(\boldsymbol{x}) + \varepsilon \ , \tag{3}$$

where $\varepsilon \sim \mathcal{N}(0, \sigma_{\text{n}}^2)$ is zero-mean Gaussian measurement noise with variance σ_{n}^2. Collecting measurements at multiple different locations in the feature space allows for GP regression to interpolate and extrapolate the measurements to unsampled regions. Let $\boldsymbol{Y} = [y_1, \ldots, y_M] \in \mathbb{R}^{3\times M}$ be a vector of M observations (3) collected at feature points $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M] \in \mathbb{R}^{n\times M}$ (collectively called the training data). Let $\boldsymbol{G} = [\boldsymbol{g}_1, \ldots, \boldsymbol{g}_G] \in \mathbb{R}^{n\times G}$ be a collection of prediction points over which the process is to be estimated. Then, the GP regression of the function at the prediction points consists of a mean estimate, $\hat{\boldsymbol{Y}}$, and a covariance, $\boldsymbol{P}_{\hat{\boldsymbol{Y}}}$, describing the uncertainty:

$$\hat{\boldsymbol{Y}}(\boldsymbol{G};\ \boldsymbol{X}, \boldsymbol{Y}) = \boldsymbol{K}(\boldsymbol{G}, \boldsymbol{X})(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\text{n}}^2 \boldsymbol{I})^{-1}\boldsymbol{Y} \tag{4}$$

$$\boldsymbol{P}_{\hat{\boldsymbol{Y}}}(\boldsymbol{G};\ \boldsymbol{X}, \boldsymbol{Y}) = \boldsymbol{K}(\boldsymbol{G}, \boldsymbol{G}) - \boldsymbol{K}(\boldsymbol{G}, \boldsymbol{X})(\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_{\text{n}}^2 \boldsymbol{I})^{-1}\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{G}) \tag{5}$$

where $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) \in \mathbb{R}^{M\times M}$ is a matrix relating the covariance of observation points to each other, $\boldsymbol{K}(\boldsymbol{G}, \boldsymbol{G}) \in \mathbb{R}^{G\times G}$ is a matrix relating the covariance between grid points, $\boldsymbol{K}(\boldsymbol{G}, \boldsymbol{X}) \in \mathbb{R}^{G\times M}$ is a matrix relating the covariance of grid points to samples, $\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{G}) = \boldsymbol{K}(\boldsymbol{G}, \boldsymbol{X})^{\text{T}}$, and $\boldsymbol{I} \in \mathbb{R}^{M\times M}$ is the identity matrix. Each covariance matrix takes the form of

$$\boldsymbol{K}(\boldsymbol{s}, \boldsymbol{s}) = \begin{bmatrix} \kappa(s_1, s_1) & \cdots & \kappa(s_1, s_M) \\ \vdots & \ddots & \vdots \\ \kappa(s_M, s_1) & \cdots & \kappa(s_M, s_M) \end{bmatrix}, \tag{6}$$

where $\boldsymbol{s}$ is an input matrix of given size (equal to either $\boldsymbol{X}$ or $\boldsymbol{G}$), and where each entry contains a scalar covariance value from (2) evaluated with the respective ith and jth input elements from the input matrix. The above discussion describes GP regression to predict a scalar function defined over a feature space. If the spatial quantity of interest is instead a vector field then methods for multi-output GP modeling can be used. A simpler approach is to develop multiple independent GP models, one for each component of the vector quantity, and perform the estimation in parallel.

2. Variational Gaussian Process (VGP) Regression

A shortcoming of GP regression is its $O(n^3)$ computational complexity due to the matrix inverse operation. Sparse GP regression (SGP) can reduce the computational cost to $O(nm^2)$ through the use of m inducing points for estimation (rather than using all available data). The inducing points $\boldsymbol{Z} = [z_1, z_2, \ldots, z_m] \in \mathbb{R}^m$ are the result of an optimization that invokes the k-means algorithm [21], and the resulting regression is [43]:

$$\hat{\boldsymbol{Y}}(\boldsymbol{G};\ \boldsymbol{Y}) = \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{G})^{\text{T}}\boldsymbol{Q}_M^{-1}\boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{X})(\boldsymbol{\Lambda} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{Y} \tag{7}$$

$$\boldsymbol{P}_{\hat{\boldsymbol{Y}}}(\boldsymbol{G};\ \boldsymbol{Y}) = \boldsymbol{K}(\boldsymbol{G}, \boldsymbol{G}) - \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{G})^{\text{T}}(\boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{Z})^{-1} - \boldsymbol{Q}_M^{-1})\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{G}) + \sigma_n^2 \boldsymbol{I}\,, \tag{8}$$

where $\boldsymbol{Q}_M = \boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{Z}) + \boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{X})(\boldsymbol{\Lambda} + \sigma_n^2 \boldsymbol{I})^{-1}\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{Z})$, $\boldsymbol{\Lambda} = \text{diag}(\boldsymbol{\lambda})$, $\boldsymbol{\lambda} = \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) - \boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{X})^{\text{T}}\boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{Z})^{-1}\boldsymbol{K}(\boldsymbol{Z}, \boldsymbol{X})$. See [43] for further details on the SGP model.

Sparse variational GPs (SVGPs) were introduced in [44] and are similar to SGPs in that they both use inducing points as a method of reducing computational complexity of GP regression. However, SVGP selects the inducing points using variational Bayesian learning by minimizing the Kullback-Leibler (KL) divergence between a variational GP and the true GP posterior [44]. Variational GP regression is equivalent to SVGP regression, with $\boldsymbol{Z} = \boldsymbol{X}$ as the input for inducing points, while minimizing the KL divergence between the approximation and the posterior [45], altering the form of (7)-(8).

Training a GP model involves optimizing the hyperparameters of the kernel to minimize a loss function between the predictions and true values of the process of interest (i.e., the training data). However, when the training data consists of multiple processes or realizations over the same input space, then the GP is viewed as multi-output model. In this paper, multiple training datasets are generated using CFD simulations over the same input space, and training is conducted to find a common set of hyperparameters that best describe all training data. To consider all the training datasets simultaneously, the input space can be annotated with an index (by augmenting the feature vector) and a coregion kernel is used to relate the covariance between two training datasets. Refer to [46, 47] for further details.

B. Signed Distance Function (SDF)

The SDF has been widely used in many robotics applications, especially for motion planning to express the distance to the nearest obstacle. Let $O \in \mathbb{R}^3$ represent the volume of three-dimensional space occupied by a set of obstacles in

the environment (buildings, etc.), including the ground plane. Let $\boldsymbol{r} \in \mathbb{R}^3$ represent the position of a robot in space. The signed distance function is a mapping $\rho : \mathbb{R}^3 \to \mathbb{R}$ that represents distance to the nearest obstacle, i.e.,

$$\rho(\boldsymbol{r}) = \inf \|\boldsymbol{r} - \boldsymbol{p}\| \ \forall \ \boldsymbol{p} \in O \ . \tag{9}$$

The gradient of the SDF is denoted $\nabla\rho(\boldsymbol{r}) \in \mathbb{R}^3$ and indicates how the building morphology is changing at a particular location. Many algorithms exist for computing the SDF. In this work an approach is used that accommodates the description of obstacles O as a mesh consisting of vertices and edges.

V. Problem Formulation

This section defines the environment, wind model, and states the problem to be investigated.

A. Urban Environment

Let $\mathcal{I} = \{P, \boldsymbol{i}_1, \boldsymbol{i}_2, \boldsymbol{i}_3\}$ be an inertial reference frame with an origin at point P, and with orthonormal unit vectors oriented along the east-north-up directions, respectively. Consider a network of UAS vehicles and wind sensing infrastructure in an urban environment D described by a rectangular prism of height h_{env} and length/width l_{env}, with an origin P at the center:

$$D = \left\{ (x, y, z) \subset \mathbb{R}^3 \mid 0 \leq z \leq h_{\text{env}}, |x| \leq \frac{l_{\text{env}}}{2}, |y| \leq \frac{l_{\text{env}}}{2} \right\} \ . \tag{10}$$

The environment contains N_O extruded polygons representing buildings or other structures. The volume occupied by each obstacle is denoted $O_i \subset D$ and is defined by a base footprint $A_i \subset \mathbb{R}^2$ and a height $h_i \in [h_{\min}, h_{\max}]$:

$$O_i = \{(x, y, z) \subset D \mid 0 \leq z \leq h_i, (x, y) \in A_i\} \ . \tag{11}$$

Each area A_i has length $d_{\min} \leq l_i \leq d_{\max}$ and width $d_{\min} \leq w_i \leq d_{\max}$ that is constrained by the parameters $d_{\min}$ and $d_{\max}$. The union of all obstacles is $O = \cup_{i=1}^{N_O} O_i$. As a simplifying assumption, the vehicles and sensing infrastructure are assumed to be at a constant altitude, z_{des}. The estimation algorithm to be discussed later assumes knowledge of the operating region D and the location and geometry of all obstacles O.

B. Urban Wind Model and Wind Measurements

A two-dimensional steady wind vector field is defined for each point $\boldsymbol{r} = [x, y]^{\mathrm{T}} \in D \setminus O$, all at a constant altitude z_{des}:

$$\boldsymbol{w}(\boldsymbol{r}) = \begin{bmatrix} u(x, y) \\ v(x, y) \end{bmatrix} \ . \tag{12}$$

Suppose that M locations

$$\boldsymbol{R} = [\boldsymbol{r}_1, \ldots, \boldsymbol{r}_M] \tag{13}$$

are sampled in the environment. A corresponding set of M measurements are obtained

$$\tilde{\boldsymbol{Y}} = [\tilde{\boldsymbol{w}}_1, \ldots, \tilde{\boldsymbol{w}}_M] \ , \tag{14}$$

where $\tilde{\boldsymbol{w}}_i = \boldsymbol{w}(\boldsymbol{r}_i)$. The sensor noise variance σ_{n}^2 is uniform in all three directions. While the proposed approach is developed with real-world applications in mind, this paper evaluates the approach in simulation where the wind vector field $\boldsymbol{w}(\boldsymbol{r})$ is represented by a CFD simulation. The CFD solution outputs the wind vector field at a series of discrete mesh node locations. To apply the approach in simulation this CFD vector-field is linearly interpolated. Similarly, while the notation in Sec. V.A describes an idealized environment, the estimation algorithm will operate over a mesh representation of this urban space.

C. Problem Statement

Let $\boldsymbol{G} \in \mathbb{R}^{2\times G}$ represent a set of planar prediction points in the environment corresponding to an altitude z_{des}. An estimate of of the wind vector-field at locations in $\boldsymbol{G}$ given the data $\tilde{\boldsymbol{Y}}, \boldsymbol{R}$ is generically denoted as $\boldsymbol{Y}(\boldsymbol{G}; \tilde{\boldsymbol{Y}}, \boldsymbol{R})$. Different

estimators are denoted by an appropriate subscript—$\hat{\boldsymbol{Y}}_{\text{GP}}(\boldsymbol{G};\tilde{\boldsymbol{Y}},\boldsymbol{R})$, and $\hat{\boldsymbol{Y}}_{\text{VGP}}(\boldsymbol{G};\tilde{\boldsymbol{Y}},\boldsymbol{R})$, representing the estimators introduced in Sec. IV, which will be used in the proposed algorithm. The goal is to develop an estimation framework that maximizes mapping performance, by minimizing the mapping error (ME). The mapping error is the magnitude of the root-mean-squared error (RMSE) for each estimated wind estimate, where

$$\mathrm{E}_u = \sqrt{\frac{1}{G}\sum_{i=1}^{G}\left(\hat{\boldsymbol{Y}}_u(\boldsymbol{G}_i;\tilde{\boldsymbol{Y}},\boldsymbol{R}) - u(\boldsymbol{G}_i)\right)^2} \tag{15}$$

$$\mathrm{E}_v = \sqrt{\frac{1}{G}\sum_{i=1}^{G}\left(\hat{\boldsymbol{Y}}_v(\boldsymbol{G}_i;\tilde{\boldsymbol{Y}},\boldsymbol{R}) - v(\boldsymbol{G}_i)\right)^2} \tag{16}$$

$$\mathrm{ME} = \sqrt{\mathrm{E}_u^2 + \mathrm{E}_v^2}\,, \tag{17}$$

and $\hat{\boldsymbol{Y}}_u$ and $\hat{\boldsymbol{Y}}_v$ represent two independently trained models to predict the two components of the planar steady wind field. This work uses CFD simulations to optimize the hyperparameters of various GP models such that low ME can be achieved on unseen datasets.

VI. Gaussian Process Training with Building Morphology Data

This section presents the training framework used to optimize the hyperparameters of various GP models. The proposed framework proceeds according to the following steps:

1) A set of Q randomized urban geometries are generated, and the SDF for each geometry is calculated.
2) CFD simulations are conducted on the Q geometries with a constant inflow wind velocity in a fixed direction.
3) One of several feature spaces is chosen, and the computed CFD wind field is sampled to produce training data.
4) The GPR kernel hyperparameters are optimized with the training data using different combinations of the kernels presented in Sec. IV.

A. Geometry Generation

The training framework begins by randomly generating Q geometries of $N_O = 2$ building using a custom MATLAB script. Each building contains eight vertices, packaged in the .STL 3D mesh format to generate an extruded rectangle. The coordinates of the first vertex of each building is randomly selected in the range $\frac{l_{\text{env}}}{2} \leq x, y \leq \frac{l_{\text{env}}}{2}$ centered on an origin P. All other vertices are then based on the initial vertex to maintain the limits introduced in (11). Figure 1 illustrates the generation of six example geometries.

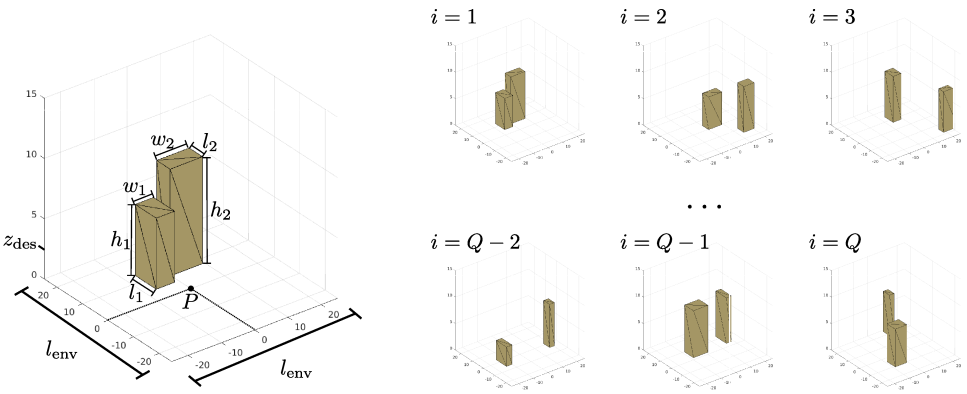

Fig. 1 Generated meshes of urban building geometries, with (left) the parameters defining the urban space and each building, and (right) six example geometries. The parameters used to generate the example geometries are $l_{\text{env}} = 40$ m, $d_{\text{min}} = 3$ m and $d_{\text{max}} = 20$ m.

B. Computational Fluid Dynamics Simulations

Wind field data for a desired urban environment is generated using OpenFOAM [20], an open-source software for computational fluid dynamics (CFD) simulations. First, the .STL urban geometry is imported into the OpenFOAM. Then, a virtual wind tunnel is designed to appropriately accommodate the geometry, see Fig. 2. The virtual wind tunnel has a height of $5h$, where h is the height of the tallest building in the geometry, and the side faces, inlet, and outlet are set to $20h$ to ensure the free-stream conditions of the wind have enough space to properly evolve.

Prior to running the CFD simulation, the wind tunnel environment is first meshed using the OpenFOAM utility `BlockMesh` for a coarse mesh, then refined near the geometries of interest using the `SnappyHexMesh` utility. The background mesh is set to have 40 cells in the x and y directions, and 20 cells in the z direction, leading to cell sizes of ~ 9 m in the x and y directions, and ~ 2 m in the z direction. Two spherical refinement zones are used to obtain finer data near the geometries of interest, with the inner sphere radius set to $r = 30$ m, and the outer sphere radius set to $2.5r$. Each zone is assigned a refinement level, corresponding to how many times the cells in that zone should be divided per direction, according to $\frac{\text{cell size}}{8\times\text{level}}$. The outer zone is set to level 2, and the inner zone is set to level 3 [48].

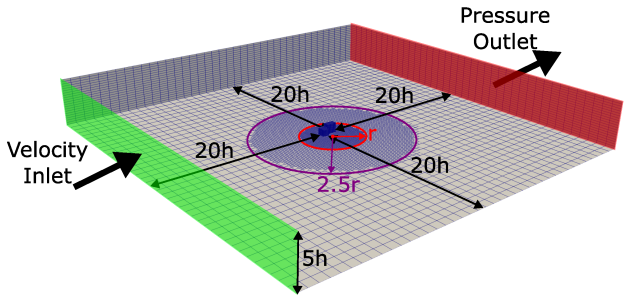

Fig. 2 Example mesh of the wind tunnel environment. Each wall of the environment is $20h$ from the nearest building, where h is the height of the tallest building in the simulation. Two circular mesh refinement regions are defined around the origin, where $r = 30$ m for all environments.

Once the mesh of the geometry is complete, it is used to simulate the wind field for an initial condition of 10 (m/s) in the positive x direction, as defined in Fig. 2. The incompressible CFD solver `simpleFoam` with the $k-\epsilon$ turbulence model are used to solve the Reynolds Averaged Navier-Stokes (RANS) mass and momentum transport equations for the given environments. The simulation is calculated for 400 iterations to reach steady-state conditions in ~ 2.5 hours of CPU time on a laptop computer (11th Gen Intel Core i7-1185G7 processor, with 12MB Cache, up to 3 Ghz CPU, and 16 GB RAM). For further details refer to [49]. The wind magnitude plots of three such simulations is shown in Fig. 3.

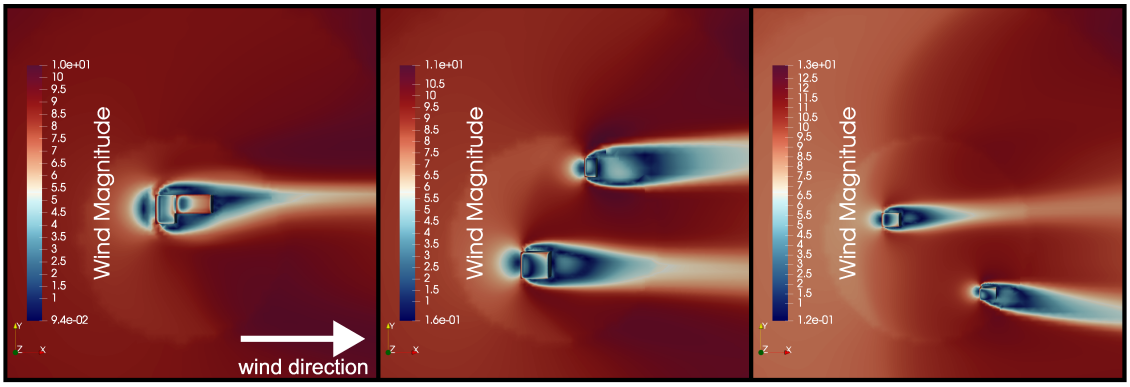

Fig. 3 A set of CFD simulation results for three example geometries. The diverging colorbar represents areas of low wind speed (dark blue) and high wind speed (dark red). The inlet condition for each simulation is 10 (m/s) in the $+x$ direction.

Following the completion of the CFD simulation, Paraview [50] is used to process the data for use in the proposed GPR wind estimation. A slice of the three-dimensional velocity vector at an altitude of z_{des}, as shown in Fig. 1, is recorded to a .CSV (comma-separated value) file for each simulated case. To perform this data extraction process

efficiently, a Paraview macro is used to clip the data to the desired dimensions and then extract a slice at the desired altitude. The dimensions used for the extraction of the data for all geometries in this work were 50 m centered at the origin in the x and y directions, and $z_{\text{des}} = d_{\min} = 3$. A total of $Q = 50$ geometries were generated and processed.

C. Augmenting Feature Vector with Building Morphology Data

Feature vectors augment position data at each measurement location with information extracted from the SDF for the environment. Consider a measurement location $\boldsymbol{r} = [x, y]^{\mathrm{T}}$. The corresponding standard feature space can be defined as

$$\text{Feature space } F_1 \text{ corresponding to feature vector } \boldsymbol{x} = \boldsymbol{r} \in \mathbb{R}^2 \ . \tag{18}$$

To augment this feature space with data representing building morphology, let $B = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_R]$ denote a set of R relative position vectors with $\boldsymbol{b}_i \in \mathbb{R}^3$ for $i = 1, \ldots, R$. Then $\rho(\boldsymbol{r} + \boldsymbol{b}_{i,\ldots,B}) \in \mathbb{R}^R$ and $\nabla\rho(\boldsymbol{r} + \boldsymbol{b}_{i,\ldots,B}) \in \mathbb{R}^{3R}$ are the SDF and SDF gradient samples at these additional R sample points. Table 1 describes the various combinations of these features that are tested.

Table 1 Feature spaces that are investigated in this work

No.	$\boldsymbol{r}$	$\rho(\boldsymbol{r})$	$\nabla\rho(\boldsymbol{r})$	$\rho(\boldsymbol{r} + \boldsymbol{b}_{i,\ldots,B})$	$\nabla\rho(\boldsymbol{r} + \boldsymbol{b}_{i,\ldots,B})$	$\mathbb{R}^n$ for $\boldsymbol{r} = [x, y]^{\mathrm{T}}$
F_1	✓					$\mathbb{R}^2$
F_2	✓	✓	✓			$\mathbb{R}^5$
F_3	✓	✓	✓	✓	✓	$\mathbb{R}^{29}$
F_4	✓	✓	✓	✓		$\mathbb{R}^{13}$
F_5	✓	✓		✓		$\mathbb{R}^{11}$
F_6	✓	✓				$\mathbb{R}^3$
F_7	✓	✓	✓		✓	$\mathbb{R}^{21}$
F_8	✓		✓		✓	$\mathbb{R}^{20}$

In practice, the building geometry is provided as a mesh and the SDF function and SDF gradient must be computed numerically to define the above feature vectors. The current framework is limited to two-dimensions, and the SDF and related data are adjusted accordingly, leading to feature spaces of sizes corresponding to the final column in Table 1. To numerically compute the SDF and gradient, the building geometry mesh is sliced at z_{des} to obtain polygonal data for each obstacle using [51]. The operating area is discretized into a uniform grid and the polygon data is used to generate a binary occupancy map over this grid. From the binary occupancy map, the SDF is calculated by using the distance function of the `scikit-fmm` python extension module [52], with a spacing input of 1 m. The resultant SDF data is then interpolated back to a desired resolution compatible with the GP regression and the gradient is calculated using [53]. The SDF and its gradient are shown for geometry 1 in Fig. 4, with a sample location $\boldsymbol{r}$ and a corresponding set of $R = 8$ additional sample points relative to $\boldsymbol{r}$. The relative sample locations can be set to any pattern, but in this work a star pattern was chosen, where for $B = [\boldsymbol{b}_1, \ldots, \boldsymbol{b}_8]$, the relative position is offset from $\boldsymbol{r}$ by 5 m and 10 m at equally spaced radial locations. For each set of CFD data, a set of randomly sampled measurement locations were chosen, where $\boldsymbol{R} = [\boldsymbol{r}_1, \ldots, \boldsymbol{r}_M] \notin O$. At each sample point in $\boldsymbol{R}$, the u and v components of the wind field are taken by interpolating from the .CSV of extracted CFD data,to define the respective sets of observations $\boldsymbol{Y}$.

D. Kernel Selection and Hyperparameter Optimization

Once the CFD data is generated and sampled, it is formatted for a particular choice of feature vector to be amenable with GPflow [54] for GP training and regression. GPflow is a flexible software package that allows assigning a particular feature its own hyperparameters. Three kernel formulations are tested with varying combinations of the squared exponential and coregion kernels. The hyperparameters associated with each kernel are optimized using the TensorFlow Adam optimizer [55], to minimize the negative log-likelihood, as is standard practice in GP training architectures. Up to 25k iterations of optimization are performed for each individual kernel.

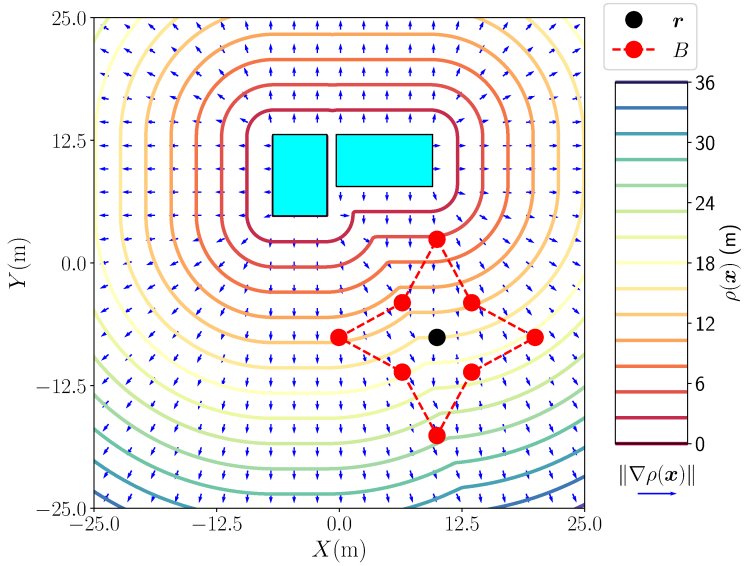

Fig. 4 Calculated SDF data for geometry 1. The cyan rectangles are buildings, with radial contour lines representing $\rho(\boldsymbol{r})$, and blue arrows representing $\nabla\rho(\boldsymbol{r})$. The black and red markers represent an example $\boldsymbol{r}$ and B sampling pattern respectively.

1. Grouped Squared Exponential Kernel

An alternative formulation of the standard squared exponential is investigated, where each grouping of similar parameters in feature space F_3 is assigned a squared exponential kernel with independent hyperparameters. Based on (2), the proposed kernel is then

$$\kappa_1 = \kappa_{\text{sq}}(\boldsymbol{h}, \underbrace{\boldsymbol{\theta}_{\text{sq},\boldsymbol{r}}}_{\mathbb{R}^3}) + \kappa_{\text{sq}}(\boldsymbol{h}, \underbrace{\boldsymbol{\theta}_{\text{sq},\rho(\boldsymbol{r}),\nabla\rho(\boldsymbol{r})}}_{\mathbb{R}^4}) + \sum_{i=1}^{8} \kappa_{\text{sq}}(\boldsymbol{h}, \underbrace{\boldsymbol{\theta}_{\text{sq},\rho(\boldsymbol{r}+\boldsymbol{b}_i),\nabla\rho(\boldsymbol{r}+\boldsymbol{b}_i)}}_{\mathbb{R}^4}) \,, \tag{19}$$

where each vector of hyperparameters has a corresponding variance σ and a number of length scales corresponding to the number of features n per combination. This kernel was initialized with a GPR model using GPflow, and its hyperparameters were optimized for 25k iterations. For each set of M feature vectors, a corresponding set of M 2D wind component measurements $\boldsymbol{Y}_u$ and $\boldsymbol{Y}_v$ are taken. The combination of $\boldsymbol{X}$ and the sets of corresponding wind measurements enable the creation of GPR models for the u and v components of wind individually. The estimates of the components are combined to estimate the wind magnitude.

2. Squared Exponential Kernel Modified with a Coregion Kernel

A second kernel was investigated that combines the squared exponential and coregion kernel (referred to as κ_2) to train a VGP model on the features in F_1 for all simulated geometries. The indices of data from different CFD runs are used to define the `active dimension` for the GPflow coregion kernel, allowing it to train on the correlation between wind from Q different urban geometries. Similar to the GPR models described in Sec. VI.D.1, a set of all u and v wind samples are collected to create a VGP model using κ_2 for the individual 2D wind components.

3. Standard Squared Exponential Kernel

The third kernel tested (referred to as κ_3) is the standard squared exponential kernel formulation (2). For each set of CFD data, and the corresponding $\boldsymbol{R}$ sample locations for each geometry, feature spaces $(F_1, \ldots, F_8)$ are constructed. For each feature space, the length scale vector ($\boldsymbol{L} \in \mathbb{R}^n$) of the corresponding hyperparameters varies in length according to the last column of Table 1, while a single spatial variance (σ) is used per feature space. Each of these models was initialized with a GPR model using GPflow, and to test the accuracy associated with different levels of optimization, the associated hyperparameters were optimized for 5k, 15k and 25k iterations.

VII. Simulation Results and Discussion

This section discusses the simulation results for the framework presented in Sec. VI. Each presented plot shows the true and estimated wind magnitudes calculated for geometry 1 for brevity. Aggregated plots show the mean and standard deviation of (17) for each tested feature space over all $Q = 50$ geometries.

A. Aggregated Feature Space Results

The presented framework estimated the wind field for $Q = 50$ simulated geometries, using $M = 50$ samples each, with the feature spaces defined in Table 1 paired with the proposed kernels presented in Sec. VI.D. The squared exponential hyperparameters for all tested kernels were initialized with $L = \frac{l_{\text{env}}}{10}$ for all length scales, $\sigma_u = \bar{u}$ for the kernels used to initialize the wind field estimator for the u direction, and similarly $\sigma_v = \bar{v}$ for the v direction, where $\bar{\cdot}$ represents the mean of that wind component over each environment. Figure 5 shows the aggregated magnitude of the RMSE mean and standard deviation for all tested kernels. The models are presented in the order of the kernels presented in Sec. VI.D, where the first model uses GPR to pair F_3 with κ_1, the second model uses VGP to pair F_1 with κ_2, and the third is a group of GPR models pairing each feature space with κ_3, respectively, optimized for 5k, 15k, and 25k iterations.

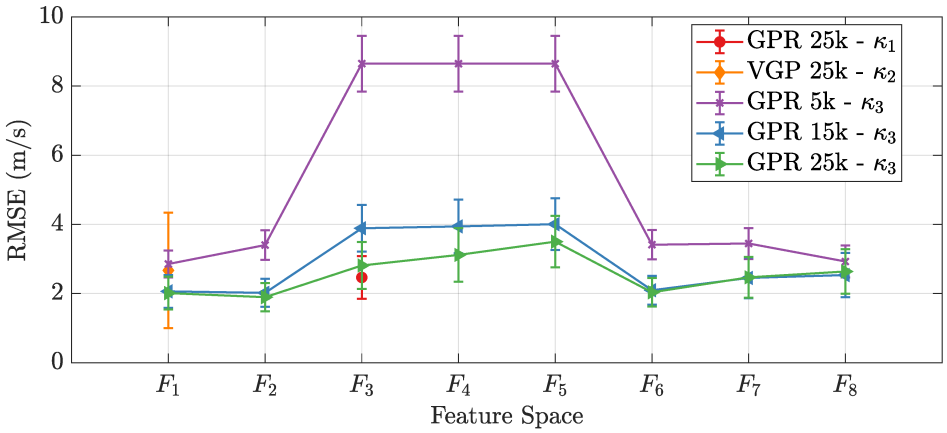

Fig. 5 Aggregated RMSE for $Q = 50$ geometries over all tested feature spaces. The central dots represent the mean of the RMSE for that model, and the error bars represent the standard deviation.

The GPR model which uses κ_1 with feature space F_3, slightly outperformed the VGP model trained on all simulated data using κ_2 with feature space F_1, for 25k optimization iterations. While the estimates for some individual geometries using the VGP model were lower than that of the GPR model, the latter more accurately estimated the wind field overall. Figure 6 shows the estimates generated from both models as compared to the true wind field for geometry 1.

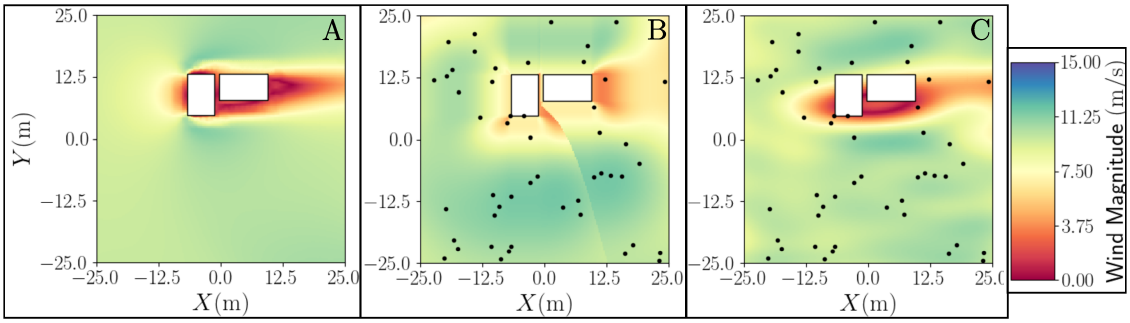

Fig. 6 A) Actual wind field, B) estimated wind field using κ_1 in GPR with F_3, and C) estimated wind field using κ_2 in VGP with F_1. Sample points are shown as black dots. Additional SDF sample points for B are suppressed.

While the estimate generated with the VGP model appears to be closer to the general shape and magnitude of the

true wind field, the estimate is slightly offset from the geometry. Conversely, the GPR model trained on feature space F_3 appears to have utilized the building morphology data to better align its estimate with the buildings, but the overall magnitude and shape of the estimated wind field does not reflect the true wind field. Moreover, an artifact appears in all of the estimates generated using feature spaces containing SDF gradient data. This artifact bisects the estimate along the gradient of the SDF, and can be seen in the center of Fig. 6B. Comparing the geometry of the artifact to the gradient shown by the blue arrows in Fig. 4 shows a correspondence. Using the SDF in the feature space can therefore lead to undesirable results in the estimated wind field.

Comparing the standard kernel implementation (κ_3) for GPR models using feature spaces F_1 through F_8 for various amounts of hyperparameter optimization iterations shows the expected trend of the total error decreasing with more optimized models. Initially, the errors are very high (similar to inflow conditions) but with further optimization they reduce to about 2 m/s RMSE. The models which underwent 5k optimization iterations took an average of $\sim$ 230 seconds to run, per geometry. This processing time scaled based on the number of optimization iterations, with the models optimized 15k taking an average of $\sim$ 640 seconds, and those optimized 25k taking an average of $\sim$ 1055 seconds. Comparing the results of the various feature spaces, F_2 performs slightly better than the standard feature space F_1 for more optimized models. When the models are optimized the least, F_1 and F_8 perform comparably, however this trend does not persist with further hyperparameter optimization. Comparing κ_1 and κ_3 for feature space F_3 indicates that the modified kernel improves the estimate over the standard, although κ_1 required an average of $\sim$ 3800 seconds to optimize for 25k iterations. Overall, the results did not support the hypothesis that including SDF data into the feature space will lead to significant reduction in mapping error.

B. Effects of Additional Samples

To highlight the effects of an increasing number of samples, the GPR model with feature space F_1 is compared for two cases: (1) when its parameters are repeatedly optimized for 10k iterations for each number of samples using a generic initial hyperparameter guess, and (2) when the hyperparameters are kept constant but initialized using $\boldsymbol{\theta}_{\text{sq}}$ hyperparameters determined through training with the coregion kernel κ_2 for all $Q = 50$ datasets. In the second case the standard squared exponential kernel is used for GPR. Figure 7 shows the RMSE for $M = [5, \ldots, 100]$. As expected, the error decreases for both models with an increasing number of samples, and the model which optimizes its hyperparameters outperforms the model that uses constant hyperparameters. The latter requires $\sim 23\times$ less time to compute an estimate, averaging $\sim$ 20 seconds per estimate with no optimization. Given the comparable performance, the model initialized with a trained set of prior hyperparameters would be more ideal for a live estimation application, whereas a model that optimizes its hyperparameters as additional data is collected would be better suited for a post-processed estimate.

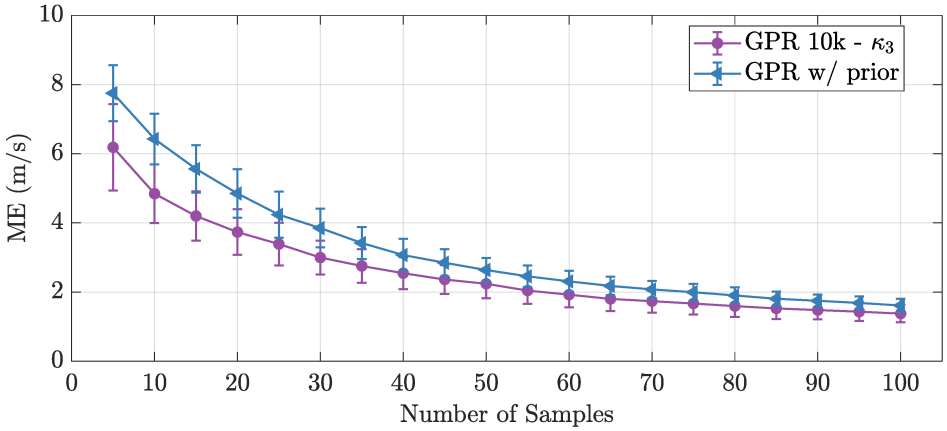

Fig. 7 RMSE for the GPR model with 10k iterations (purple), versus the same model initialized with the squared exponential hyperparameters from κ_2

Figure 8 shows the estimates for both cases evolving for three choices of number of samples M. As more data is collected, the model with previously trained hyperparameters (lower row in Fig. 8) appears to have a stronger correlation in the horizontal direction of the wind flow. However, it under-predicts the wind field at the left and right boundaries of

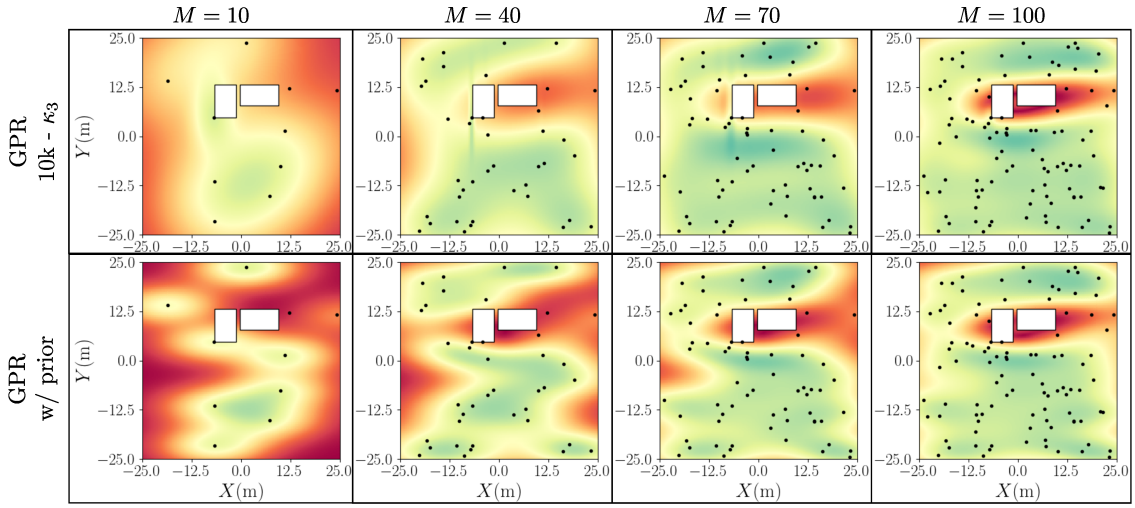

Fig. 8 Estimates of wind field magnitude (m/s) generated with various numbers of samples for (top row) a GPR model with 10k hyperparameter optimization iterations, and (bottom row) a GPR model initialized with prior hyperparameters, without model optimization, both using feature space F_1.

the domain. This example highlights the variability of estimation performance that might be expected when using a feature space exclusively based on the position of observations. Moreover, the GP regression implemented in this paper does not appear to accurately predict the mean value of the wind field in unsampled regions, especially with only a few measurements.

VIII. Conclusions

This paper explored methods for urban wind field estimation that incorporate knowledge of building morphology and physics-based CFD simulations into Gaussian process regression models. Wind field CFD data was collected for randomly generated geometries of an urban environment consisting of two buildings. A series of GP regressions were then evaluated that include various kernels and feature spaces. The different kernels tested include squared exponential kernels with different hyperparameter structure and a coregion kernel that allows simultaneous training over multiple CFD datasets. The feature spaces tested include augmenting the position of each measurement with various degrees of additional building morphology features. The building morphology features consisted of the value of the signed distance field and its gradient at a pattern of points around each sample. The GP regression models were trained and evaluated using 50 simulated environments using the GPflow software package. Increasing number of samples are shown to improve the estimate for both optimized and non-optimized GPR models. The addition of SDF data to feature spaces using the standard squared exponential kernel does not appear to appreciably improve the wind magnitude estimate, and may produce undesirable artifacts in the estimation results. Future work may explore using other machine-learning based methods for urban wind field estimation that can leverage knowledge of building morphology and that can generalized to efficiently train over more complex urban geometries with greater variability in inflow wind conditions.

Acknowledgments

This work was supported by NSF Grant No. 2301475.

References

[1] Jeong, S., You, K., and Seok, D., "Hazardous Flight Region Prediction for a Small UAV Operated In An Urban Area Using a Deep Neural Network," *Aerospace Science and Technology*, Vol. 118, No. 11, 2021, p. 107060. https://doi.org/10.1016/j.ast.2021.107060.

[2] Giersch, S., El Guernaoui, O., Raasch, S., Sauer, M., and Palomar, M., "Atmospheric Flow Simulation Strategies to Assess

Turbulent Wind Conditions for Safe Drone Operations in Urban Environments," *Journal of Wind Engineering and Industrial Aerodynamics*, Vol. 229, No. 9, 2022, pp. 105136: 1–20. https://doi.org/10.1016/j.jweia.2022.105136.

[3] Raza, S. A., Sutherland, M., Etele, J., and Fusina, G., "Experimental Validation of Quadrotor Simulation Tool for Flight Within Building Wakes," *Aerospace Science and Technology*, Vol. 67, No. 8, 2017, pp. 169–180. https://doi.org/10.1016/j.ast.2017.03.043.

[4] Vuppala, R. K. S. S., and Kara, K., "A Non-Intrusive Reduced Order Model Using Deep Learning for Realistic Wind Data Generation for Small Unmanned Aerial Systems in Urban Spaces," *American Institute of Physics Advances*, Vol. 12, No. 8, 2022, pp. 085020: 1–14. https://doi.org/10.1063/5.0098835.

[5] Sutherland, M., Etele, J., and Fusina, G., "Urban Wake-Field Generation Using Large-Eddy Simulation for Application To Quadrotor Flight," *Journal of Aircraft*, Vol. 53, No. 5, 2016, pp. 1224–1236. https://doi.org/10.2514/1.C033624.

[6] Vuppala, R. K. S. S., Krawczyk, Z., Paul, R., and Kara, K., "Modeling Advanced Air Mobility Aircraft in Data-Driven Reduced Order Realistic Urban Winds," *Scientific Reports*, Vol. 14, No. 1, 2024, pp. 383: 1–16. https://doi.org/10.1038/s41598-023-50719-8.

[7] Nathanael, J. C., Wang, C. H. J., and Low, K. H., "Simulation of Wind Field in a Building Complex for Evaluation of the Wind Effect Along UAS Flight Path," *Proceedings of the 2023 AIAA AVIATION Forum*, American Institute of Aeronautics and Astronautics, 2023, pp. 1–15. https://doi.org/10.2514/6.2023-4096.

[8] Cybyk, B. Z., McGrath, B. E., Frey, T. M., Drewry, D. G., Keane, J. F., and Patnaik, G., "Unsteady Airflows and Their Impact On Small Unmanned Air Systems in Urban Environments," *Journal of Aerospace Information Systems*, Vol. 11, No. 4, 2014, pp. 178–194. https://doi.org/10.2514/1.I010000.

[9] Ware, J., and Roy, N., "An Analysis of Wind Field Estimation and Exploitation for Quadrotor Flight in the Urban Canopy Layer," *Proceedings of the 2016 IEEE International Conference on Robotics and Automation*, IEEE, Stockholm, Sweden, 2016, pp. 1507–1514. https://doi.org/10.1109/ICRA.2016.7487287.

[10] Meyer-Oehme, T., Ginnell, K., Lampl, D. E., and Armanini, S. F., "Landing Framework and Control for EVTOL Aircraft in Urban Environments," *Proceedings of the 2023 AIAA SCITECH Forum*, American Institute of Aeronautics and Astronautics, 2023, p. 1331. https://doi.org/10.2514/6.2023-1331.

[11] Patrikar, J., Dugar, V., Arcot, V., and Scherer, S., "Real-time Motion Planning of Curvature Continuous Trajectories for Urban UAV Operations in Wind," *Proceedings of the 2020 International Conference on Unmanned Aircraft Systems*, IEEE, Athens, Greece, 2020, pp. 854–861. https://doi.org/10.1109/ICUAS48674.2020.9213837.

[12] Cho, D.-H., Ha, J.-S., Lee, S., Moon, S., and Choi, H.-L., "Informative Path Planning and Mapping With Multiple UAVs in Wind Fields," *Proceedings of the Distributed Autonomous Robotic Systems: The 13th International Symposium*, Springer, 2018, pp. 269–283. https://doi.org/10.1007/978-3-319-73008-0_19.

[13] Zhang, B., Ooka, R., Kikumoto, H., Hu, C., and Tse, T. K., "Towards Real-Time Prediction of Velocity Field Around a Building Using Generative Adversarial Networks Based on the Surface Pressure from Sparse Sensor Networks," *Journal of Wind Engineering and Industrial Aerodynamics*, Vol. 231, 2022, pp. 105243: 1–14. https://doi.org/10.1016/j.jweia.2022.105243.

[14] Kakavitsas, N. P., and Wolek, A., "Quadrotor Takeoff Trajectory Planning in a One-Dimensional Uncertain Wind-field Aided by Wind-Sensing Infrastructure," *Proceedings of the 2024 AIAA SCITECH Forum*, 2024, pp. 1–18. https://doi.org/10.2514/6.2024-0987.

[15] Yang, S., Wei, N., Jeon, S., Bencatel, R., and Girard, A., "Real-Time Optimal Path Planning and Wind Estimation Using Gaussian Process Regression for Precision Airdrop," *Proceedings of the 2017 American Control Conference*, IEEE, 2017, pp. 2582–2587. https://doi.org/10.23919/ACC.2017.7963341.

[16] Hollinger, G. A., Pereira, A. A., Binney, J., Somers, T., and Sukhatme, G. S., "Learning Uncertainty in Ocean Current Predictions for Safe and Reliable Navigation of Underwater Vehicles," *Journal of Field Robotics*, Vol. 33, No. 1, 2015, pp. 47–66. https://doi.org/10.1002/rob.21613.

[17] Lawrance, N. R., and Sukkarieh, S., "Path Planning for Autonomous Soaring Flight in Dynamic Wind Fields," *Proceedings of the 2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 2499–2505. https://doi.org/10.1109/ICRA.2011.5979966.

[18] Zheng, L., Yang, R., Pan, J., and Cheng, H., "Safe Learning-Based Tracking Control for Quadrotors Under Wind Disturbances," *Proceedings of the 2021 American Control Conference*, IEEE, New Orleans, LA, USA, 2021, pp. 3638–3643. https://doi.org/10.23919/ACC50511.2021.9482929.

[19] Patrikar, J., Moon, B. G., and Scherer, S., "Wind and the City: Utilizing UAV-Based In-Situ Measurements for Estimating Urban Wind Fields," *Proceedings of the 2020 International Conference on Intelligent Robots and Systems*, IEEE, Las Vegas, NV, USA, 2020, pp. 1254–1260. https://doi.org/10.1109/IROS45743.2020.9340812.

[20] Jasak, H., "OpenFOAM: Open Source CFD in Research and Industry," *International Journal of Naval Architecture and Ocean Engineering*, Vol. 1, No. 2, 2009, pp. 89–94. https://doi.org/10.2478/IJNAOE-2013-0011.

[21] Matthews, A. G. d. G., van der Wilk, M., Nickson, T., Fujii, K., Boukouvalas, A., León-Villagrá, P., Ghahramani, Z., and Hensman, J., "GPflow: A Gaussian Process Library Using TensorFlow," *Journal of Machine Learning Research*, Vol. 18, No. 40, 2017, pp. 1–6. https://doi.org/10.48550/arXiv.1610.08733, URL http://jmlr.org/papers/v18/16-537.html.

[22] Davoudi, B., Taheri, E., Duraisamy, K., Jayaraman, B., and Kolmanovsky, I., "Quad-Rotor Flight Simulation in Realistic Atmospheric Conditions," *AIAA Journal*, Vol. 58, No. 5, 2019, pp. 1992–2004. https://doi.org/10.2514/1.J058327.

[23] Achermann, F., Stastny, T., Danciu, B., Kolobov, A., Chung, J. J., Siegwart, R., and Lawrance, N., "Windseer: Real-Time Volumetric Wind Prediction Over Complex Terrain Aboard a Small Uncrewed Aerial Vehicle," *Nature Communications*, Vol. 15, No. 1, 2024, pp. 3507: 1–17. https://doi.org/10.1038/s41467-024-47778-4.

[24] Duan, Y., Achermann, F., Lim, J., and Siegwart, R., "Energy-Optimized Planning in Non-Uniform Wind Fields with Fixed-Wing Aerial Vehicles," *arXiv preprint arXiv:2404.02077*, 2024. https://doi.org/10.48550/arXiv.2404.02077.

[25] Oettershagen, P., Achermann, F., Müller, B., Schneider, D., and Siegwart, R., "Towards Fully Environment-Aware UAVs: Real-Time Path Planning With Online 3D Wind Field Prediction in Complex Terrain," *arXiv preprint arXiv:1712.03608*, 2017.

[26] Galway, D., Etele, J., and Fusina, G., "Modeling of Urban Wind Field Effects on Unmanned Rotorcraft Flight," *Journal of Aircraft*, Vol. 48, No. 5, 2011, pp. 1613–1620. https://doi.org/10.2514/1.C031325.

[27] Hosseinzadeh, A., and Keshmiri, A., "Computational Simulation of Wind Microclimate in Complex Urban Models and Mitigation Using Trees," *Journal of Buildings*, Vol. 11, No. 3, 2021, pp. 112: 1–19. https://doi.org/10.3390/buildings11030112.

[28] Baik, J. J., Park, S. B., and Kim, J. J., "Urban Flow and Dispersion Simulation Using a CFD Model Coupled to a Mesoscale Model," *Journal of Applied Meteorology and Climatology*, Vol. 48, No. 8, 2009, pp. 1667–1681. https://doi.org/10.1175/2009JAMC2066.1.

[29] Toja-Silva, F., Kono, T., Peralta, C., Lopez-Garcia, O., and Chen, J., "A Review of Computational Fluid Dynamics (CFD) Simulations of the Wind Flow Around Buildings for Urban Wind Energy Exploitation," *Journal of Wind Engineering and Industrial Aerodynamics*, Vol. 180, No. June, 2018, pp. 66–87. https://doi.org/10.1016/j.jweia.2018.07.010.

[30] Nithya, D., Quaranta, G., Muscarello, V., and Liang, M., "Review of Wind Flow Modelling in Urban Environments to Support the Development of Urban Air Mobility," *Drones*, Vol. 8, No. 4, 2024, pp. 147: 1–24. https://doi.org/10.3390/drones8040147.

[31] Al Labbad, M., Wall, A., Larose, G. L., Khouli, F., and Barber, H., "Experimental Investigations into the Effect of Urban Airflow Characteristics on Urban Air Mobility Applications," *Journal of Wind Engineering and Industrial Aerodynamics*, Vol. 229, 2022, pp. 105126: 1–21. https://doi.org/10.1016/j.jweia.2022.105126.

[32] Gimenez, J. M., and Bre, F., "An Enhanced k-ω Sst Model to Predict Airflows Around Isolated and Urban Buildings," *Building and Environment*, Vol. 237, 2023, p. 110321. https://doi.org/0.1016/j.buildenv.2023.110321.

[33] Menter, F. R., "Zonal Two Equation k-ω Turbulence Models for Aerodynamic Flows," *Proceedings of the 23rd Fluid Dynamics, Plasmadynamics, and Lasers Conference*, 1993, pp. 1–22. https://doi.org/10.2514/6.1993-2906, aIAA 93-2906.

[34] Menter, F. R., "Two-Equation Eddy-Viscosity Turbulence Models for Engineering Applications," *AIAA Journal*, Vol. 32, No. 8, 1994, pp. 1598–1605. https://doi.org/10.2514/3.12149.

[35] Menter, F. R., Kuntz, M., and Langtry, R., "Ten Years of Industrial Experience with the SST Turbulence Model," *Turbulence, Heat and Mass Transfer*, Vol. 4, No. 1, 2003, pp. 625–632. URL https://api.semanticscholar.org/CorpusID:38105247.

[36] Lee, K. M. B., Yoo, C., Hollings, B., Anstee, S., Huang, S., and Fitch, R., "Online Estimation of Ocean Current from Sparse GPS Data for Underwater Vehicles," *Proceedings of the 2019 International Conference on Robotics and Automation*, 2019, pp. 3443–3449. https://doi.org/10.1109/ICRA.2019.8794308.

[37] Vuppala, R. K. S. S., and Kara, K., "Wind Field Prediction in Urban Spaces for Small Unmanned Aerial Systems Using Convolutional Autoencoders," *Proceedings of the 2022 AIAA AVIATION Forum*, 2022, pp. 1–13. https://doi.org/10.2514/6.2022-3605.

[38] Yasuda, Y., and Onishi, R., "Two-Stage Super-Resolution Simulation Method for Three-Dimensional Flow Fields Around Buildings for Real-Time Prediction of Urban Micrometeorology," *arXiv preprint arXiv:2404.02631*, 2024, pp. 1–51. https://doi.org/10.48550/arXiv.2404.02631.

[39] Wu, Y., Zhan, Q., and Quan, S. J., "Improving Local Pedestrian-Level Wind Environment Based on Probabilistic Assessment Using Gaussian Process Regression," *Building and Environment*, Vol. 205, 2021, p. 108172. https://doi.org/10.1016/j.buildenv.2021.108172.

[40] Schmidt, A. M., and Gelfand, A. E., "A Bayesian Coregionalization Approach for Multivariate Pollutant Data," *Journal of Geophysical Research: Atmospheres*, Vol. 108, No. D24, 2003, pp. 1–9. https://doi.org/10.1029/2002JD002905.

[41] Weerasuriya, A., Zhang, X., Lu, B., Tse, K., and Liu, C., "A Gaussian Process-Based Emulator for Modeling Pedestrian-Level Wind Field," *Building and Environment*, Vol. 188, 2021, p. 107500. https://doi.org/10.1016/j.buildenv.2020.107500.

[42] Rasmussen, C., and Williams, C., *Gaussian Processes for Machine Learning*, MIT Press, Boston, 2006.

[43] Snelson, E., and Ghahramani, Z., "Sparse Gaussian Processes using Pseudo-Inputs," *Advances in Neural Information Processing Systems*, Vol. 18, MIT Press, 2005, pp. 1–8.

[44] Titsias, M., "Variational Learning of Inducing Variables in Sparse Gaussian Processes," *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, Vol. 5, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 2009, pp. 567–574.

[45] Titsias, M. K., "Variational Inference for Gaussian and Determinantal Point Processes," , Dec 2014. URL http://www2.aueb.gr/users/mtitsias/papers/titsiasNipsVar14.pdf, Workshop on Advances in Variational Inference - Proceedings of the 2014 Neural Information Processing Systems Conference.

[46] Bonilla, E. V., Chai, K. M. A., and Williams, C. K. I., "Multi-task Gaussian Process Prediction," *Advances in Neural Information Processing Systems 20*, NIPS Foundation, 2008, pp. 153–160.

[47] Tipping, M. E., and Bishop, C. M., "Probabilistic Principal Component Analysis," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, Vol. 61, No. 3, 1999, pp. 611–622. https://doi.org/10.1111/1467-9868.00196.

[48] Greenshields, C., *OpenFOAM v12 User Guide*, The OpenFOAM Foundation, London, UK, 2024. URL https://doc.cfd.direct/openfoam/user-guide-v12.

[49] Kakavitsas, N. P., Willis, A., Jacobik, R., Uddin, M., and Wolek, A., "Quadrotor Flight Simulation in a CFD-generated Urban Wind Field," *2024 IEEE Aerospace Conference*, 2024, pp. 1–8. https://doi.org/10.1109/AERO58975.2024.10521032.

[50] Ahrens, J., Geveci, B., and Law, C., *ParaView: An End-User Tool for Large Data Visualization*, Elesvier, 2005. ISBN 978-0123875822.

[51] Hague, C., "Horizontal-Mesh-Slicing," https://github.com/c-hague/Horizontal-Mesh-Slicing, 2024. Accessed: 2024-04-05.

[52] scikit-fmm Team, "scikit-fmm documentation," https://pythonhosted.org/scikit-fmm/, 2015. Accessed: 2024-04-05.

[53] Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E., "Array programming with NumPy," , September 2020. https://doi.org/10.1038/s41586-020-2649-2.

[54] GPflow Contributors, "GPflow 2.9.0 documentation - User Guide," https://gpflow.github.io/GPflow/develop/user_guide.html, 2023. Accessed: 2024-11-25.

[55] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," , 2015. URL https://www.tensorflow.org/, software available from tensorflow.org.