3D Radar Velocity Maps for Uncertain Dynamic Environments

Ransalu Senanayake[*1], Kyle Beltran Hatch[*1], Jason Zheng[2], and Mykel J. Kochenderfer[1]

Abstract—**Future urban transportation concepts include a mixture of ground and air vehicles with varying degrees of autonomy in a congested environment. In such dynamic environments, occupancy maps alone are not sufficient for safe path planning. Safe and efficient transportation requires reasoning about the 3D flow of traffic and properly modeling uncertainty. Several different approaches can be taken for developing 3D velocity maps. This paper explores a Bayesian approach that captures our uncertainty in the map given training data. The approach involves projecting spatial coordinates into a high-dimensional feature space and then applying Bayesian linear regression to make predictions and quantify uncertainty in our estimates. On a collection of air and ground datasets, we demonstrate that this approach is effective and more scalable than several alternative approaches.**

I. INTRODUCTION

Future urban transportation system concepts include a mixture of both autonomous and human-driven vehicles. We anticipate driverless cars on roads as well as delivery drones [1] and urban air mobility (UAM) systems with vertical take-off and landing capabilities [2]. These advances add complexity to our transportation systems (Figure 1a), making decision making for autonomous vehicles operating in such dynamic systems even more challenging.

Safe and efficient transportation in a congested environment requires accurately modeling the flow of traffic in 3D space at both the macroscopic and microscopic levels. At the macroscopic level, the velocity maps estimate the average behavior of vehicles at different locations in the system, as opposed to estimating the current behavior of surrounding targets (Figure 1b). We may have large amounts of data to build such models, and we want to be able to build them efficiently. At the microscopic level, we want to model the surroundings of an individual vehicle from a continuous stream of data to allow it to maneuver safely (Figure 1c). These velocity maps estimate the behavior of vehicles surrounding ego vehicle at the current point in time. We often only have to learn models from very little data.

To provide robust control of these vehicles, it is important to capture the uncertainty associated with our velocity maps. For every point in the 3D space, for example, it would be helpful to quantify both the mean and variance of the various velocity components. While Bayesian nonparametric methods such as Gaussian process-based models [3] can provide uncertainty estimates, they (as well as many of their approximations [4]) are generally not suitable for building large-scale macroscopic models because their computational complexity grows too quickly with the number of data points [3], [5].

This paper applies an approach that has many of the desirable attributes of a Gaussian process-based model, but it is faster and more memory efficient. We adopt an approach that involves projecting the spatial coordinates into a high-dimensional, yet interpretable, feature space to capture information local to a given area. Bayesian linear regression is used to learn the parameters of the model. We can then query arbitrary points in the 3D space to obtain smooth estimates of the mean and variance of the velocity components. The model can be efficiently updated online, making it amenable to changes in the environment. A theoretical analysis shows that this model is robust against input noise. We demonstrate our approach on simulated and real-world datasets.

II. RELATED WORK

Many robotics applications involve building maps of the environment. Occupancy maps is the most common representation. Occupancy maps alone can only be used to navigate through an environment when surrounding obstacles are stationary. However, in real urban environments an autonomous vehicle must be able to safely navigate around both stationary obstacles and moving vehicles. Developing velocity maps are therefore crucial for planning algorithms to plan safe trajectories for autonomous vehicles operating in urban environments.

Techniques such as occupancy grid maps [6] discretize the environment and model if a cell is free or occupied. Such models assume the cells are independent, ignoring neighborhood information. While such methods can model the binary occupancy probability, they do not model epistemic uncertainty. To address these limitations, several 2D Bayesian models have been proposed [5], [7]–[9]. These models can be queried at an arbitrary resolution at run time. There have also been various attempts to model occupancy in dynamic environments [10]–[12]. However, such models do not explicitly represent the velocity as a map. We model the uncertainty of velocity in the 3D space which is computationally challenging compared to conventional 2D occupancy mapping techniques. Although there are similarities with the model we explore in this paper, these models use a binary random variable to model occupancy [13]. In this work, we are interested in modeling the velocity which is not a binary variable.

---
*Equal contribution.

[1]R. Senanayake, K. B. Hatch, and M. J. Kochenderfer are with the Stanford Intelligent Systems Laboratory (SISL) in the Aeronautics and Astronautics Department, Stanford University, 496 Lomita Mall, Stanford, CA 94305, USA. Email: {khatch, ransalu, mykel}@stanford.edu.

[2]J. Zheng is with the Department of Computer Science, Stanford University. Email: jzzheng@stanford.edu.

(a) A future urban environment.   (b) Velocity of drone trajectories.   (c) Radar point cloud velocity of the red car.
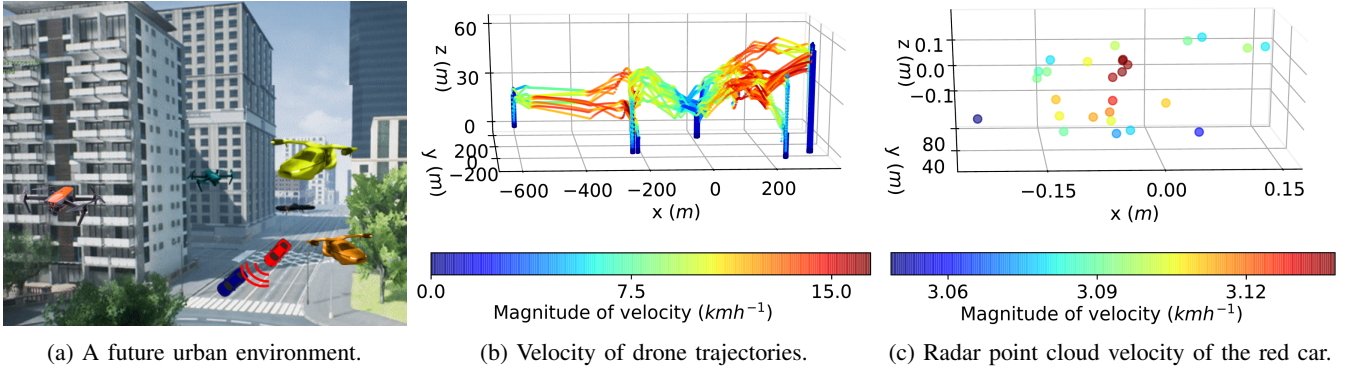
Fig. 1: (a) Since we will have complex urban environments with ground vehicles, delivery drones, and urban air mobility (UAM) in the future, it is important to model the dynamics such as velocity and acceleration of the environment. (b) We need to build *macroscopic* models of both ground and air roads for the whole city from *large* amounts of trajectory data. (c) Vehicles also need to build *microscopic* models of the velocity of objects around them from *small* amounts of sparse data. Being able to learn the velocity *quickly* helps making efficient decisions. Being able to quantify the associated *uncertainty* aids in making safe decisions.

Velocity modeling has previously been studied in various disciplines [14], [15]. However, these models are deterministic. Other models that attempt to estimate quantities that change temporally include modeling the long-term occupancy [4] and directions [16] in 2D. In contrast, the objective of this paper is modeling velocity with their associated epistemic uncertainties in 3D space.

III. BAYESIAN DYNAMIC FIELDS

This section introduces the proposed framework for modeling 3D velocity maps. Since we want to model the spatial field of dynamics such as velocity with its associated uncertainty, we refer to this framework as Bayesian Dynamic Fields (BDF). First, we explain how to generate high dimensional features from data using kernel functions. Then, we discuss how to build a linear model from these features and estimate uncertainty with Bayesian linear regression with these basis functions. These basis functions can be customized for different datasets. This model is applied to build macroscopic and microscopic dynamic models. Finally, a theoretical analysis of the robustness of the proposed framework is presented.

A. High dimensional feature space

Our objective is to model the velocity field and its corresponding uncertainty field in a given 3D space. Velocity variations exhibit nonlinear patterns with respect to spatial location. A common tool for modeling nonlinear patterns is deep neural networks, but they generally require large amounts of training data and can be slow to train. We focus on kernel methods [17], which have been successfully used to model spatial quantities such as occupancy [7], [18], [19] and directions [20] in robotics, soil concentration in geostatistics [21], and disease propagation in epidemiology [22].

Kernels are similarity functions. A kernel, $k(\mathbf{x}_a, \mathbf{x}_b)$, takes two inputs $\mathbf{x}_a$ and $\mathbf{x}_b$ and outputs a measure on how similar

the two inputs are. In this work, we use the squared-exponential kernel because of its simplicity and interpretability:

$$k(\mathbf{x}_a, \mathbf{x}_b) = \exp(-\gamma \|\mathbf{x}_a - \mathbf{x}_b\|_2^2), \tag{1}$$

where γ is the inverse bandwidth hyperparameter. This hyperparameter controls the sensitivity of the similarity. Kernels with smaller values of γ capture correlations over larger areas.

We use this kernel to define a set of M basis functions, $k(\mathbf{x}, \tilde{\mathbf{x}}_1), \ldots, k(\mathbf{x}, \tilde{\mathbf{x}}_M)$, where $\mathbf{x}$ is any point in 3D space and $\tilde{\mathbf{x}}_1, \ldots, \tilde{\mathbf{x}}_M$ are fixed points in that space. We arrange the fixed points in a 3D regular grid. Although kernels can alternatively be computed through random Fourier features [19], [23] or Nystrom approximation, we use a grid for simplicity, interpretability, and accuracy [19]. If desired, these fixed points can be learned alongside other parameters [24].
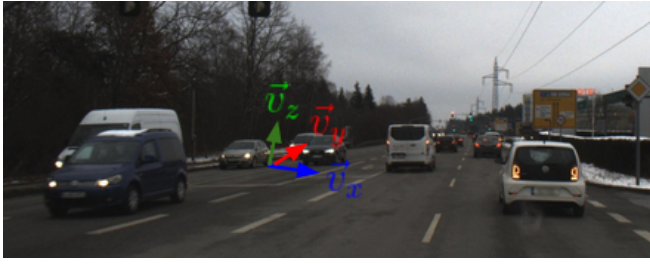
A data point $\mathbf{x} \in \mathbb{R}^3$ may come, for example, from radar (Figure 2a–b) or IMU measurements. For N such data points, $\mathbf{X} = \{\mathbf{x}_n\}_{n=1}^N$, the feature matrix $\Phi(\mathbf{X}) \in \mathbb{R}^{N \times M}$ is defined as,

$$\Phi(\mathbf{X}) = \begin{bmatrix} k(\mathbf{x}_1, \tilde{\mathbf{x}}_1) & k(\mathbf{x}_1, \tilde{\mathbf{x}}_2) & \ldots & k(\mathbf{x}_1, \tilde{\mathbf{x}}_M) \\ k(\mathbf{x}_2, \tilde{\mathbf{x}}_1) & k(\mathbf{x}_2, \tilde{\mathbf{x}}_2) & \ldots & k(\mathbf{x}_2, \tilde{\mathbf{x}}_M) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_N, \tilde{\mathbf{x}}_1) & k(\mathbf{x}_N, \tilde{\mathbf{x}}_2) & \ldots & k(\mathbf{x}_N, \tilde{\mathbf{x}}_M) \end{bmatrix}. \tag{2}$$
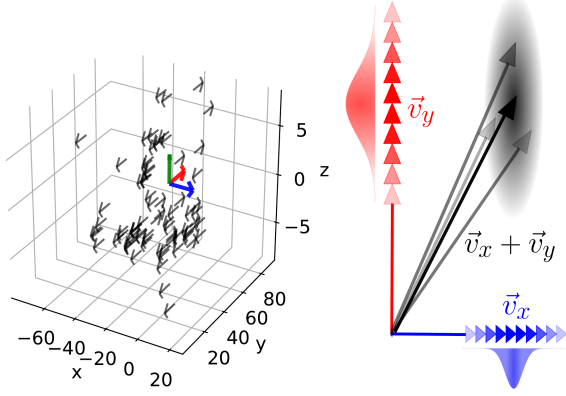
This matrix would be mostly sparse with many values close to zero because the kernel function goes to zero when its two inputs are far apart.

B. Bayesian inference

We want to build a model that can estimate the velocity of a given point in the environment. For instance, as shown in Figure 2, given some sparse velocity measurements, we want to know the velocity at an arbitrary location indicated by the three colored arrows. Because the three directional velocity

(a) Field of view.

(b) Radar measurements.　(c) Uncertainty of velocity.

Fig. 2: Unlike LIDAR, automotive radars provides the velocity associated with each point in the point cloud. We want to estimate the uncertainty of velocity at the arbitrary point indicated by the three colored arrows. (a) The camera image of an area an automotive radar can see. (b) The velocity vectors from pre-processed 3D radar measurements [25] are in black. (c) The estimates from our model are not deterministic vectors but distributions of velocities for each direction. For simplicity, we only show the red and blue directions. A few resulting vectors are shown in black.

components $v^{(x)}$, $v^{(y)}$, and $v^{(z)}$ are independent with each other, three different models are learned in parallel.

If the velocity component labels of each datapoint $\mathbf{x}$ is denoted by v, then the training dataset can be defined as $\mathcal{D} = \{(\mathbf{x}_n, v_n)\}_{n=1}^{N} = (\mathbf{X}, \mathbf{v})$. Since we projected the data into M-dimensional space, we can now create a linear model $v = \mathbf{w}^\top \Phi(\mathbf{x}) + \epsilon$ with noise $\epsilon \sim \mathcal{N}(0, \beta^{-1})$ where β is the noise precision. Our objective is to learn the parameter vector $\mathbf{w} \in \mathbb{R}^M$ from $\mathcal{D}$. The velocities at one of the three directions in a given location are modeled as a Gaussian distribution. Because measurements are *i.i.d.*, the likelihood can be decomposed as $p(\mathbf{v}|\mathbf{w}, \mathbf{X}, \beta) = \prod_{n=1}^{N} \mathcal{N}(v_n|\mathbf{w}^\top \Phi(\mathbf{x}_n), \beta^{-1})$.

As illustrated in Figure 2c, we are not only interested in estimating the velocity but also the associated uncertainty. In order to model the epistemic uncertainty, we consider a prior probability distribution over $\mathbf{w}$. A Gaussian distribution over $\mathbf{w} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$ is a conjugate prior to the likelihood model of our interest. With this prior, the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{v}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$ from Bayesian linear regression

in the feature space can be computed analytically [26]:

$$\boldsymbol{\mu}_1 = \boldsymbol{\Sigma}_1(\boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0 + \beta\Phi^\top(\mathbf{X})\mathbf{y}) \quad (3)$$

$$\boldsymbol{\Sigma}_1 = (\boldsymbol{\Sigma}_0^{-1} + \beta\Phi^\top(\mathbf{X})\Phi(\mathbf{X}))^{-1}. \quad (4)$$

Given that our prior knowledge about an area is minimal, we can set $\boldsymbol{\mu}_0 = \mathbf{0}$ and $\Sigma_0 = \alpha^{-1}\mathbf{I}$, where α is a small parameter indicating the precision (i.e. the inverse of variance) of the prior. This indicates an almost uninformative prior. Furthermore, this prior acts as a natural regularizer for the high-dimensional regression problem.

Having obtained the posterior distribution, the posterior predictive distribution $p(v_*|\mathbf{x}_*, \mathcal{D}, \alpha, \beta) = \mathcal{N}(v_*|\boldsymbol{\mu}_*, \boldsymbol{\sigma}_*)$ for an arbitrary unknown point $\mathbf{x}_* \in \mathbb{R}^3$ can be queried analytically:

$$\boldsymbol{\mu}_* = \boldsymbol{\mu}_1^\top \Phi(\mathbf{x}_*) \quad (5)$$

$$\boldsymbol{\sigma}_*^2 = \beta^{-1} + \Phi^\top(\mathbf{x}_*)\boldsymbol{\Sigma}_1\Phi(\mathbf{x}_*). \quad (6)$$

If we have a batch of query points, Φ can be computed as in (2).

C. Dimension-adjusted kernels

When defining the kernel in (1), we considered that the norm between a data point $\mathbf{x} \in \mathbb{R}^3$ and a fixed point $\tilde{\mathbf{x}} \in \mathbb{R}^3$ is scaled by the hyperparameter γ. However, if the nonlinear patterns change in different rates in different axes of the 3D space, we can scale them differently to better fit the model:

$$k(\mathbf{x}, \tilde{\mathbf{x}}) = \exp\big(-(\mathbf{x} - \tilde{\mathbf{x}})^\top \Gamma^{-1}(\mathbf{x} - \tilde{\mathbf{x}})\big), \quad (7)$$

where

$$\Gamma = \begin{bmatrix} \gamma_x & 0 & 0 \\ 0 & \gamma_y & 0 \\ 0 & 0 & \gamma_z \end{bmatrix}, \quad (8)$$

is the hyperparameter matrix. Although it is possible to consider the full non-diagonal matrix by considering the hyperparameter-hyperparameter covariances, in this application, we ignore such complex interactions for the sake of simplicity. It is also possible to learn the parameters [24] or learn a completely new kernel function (1) [27].

D. Macroscopic and microscopic velocity maps

As shown in Figure 1b, a macroscopic model represents the velocity of a large area of an urban environment. These models will especially be useful when designing urban air mobility systems [28]. In order to build a global model, trajectory information of vehicles in the environment are collected for a long time period. This information can be obtained from surveillance radar [29] or IMU data. The velocity is represented as $\mathbf{v} = (v_x, v_y, v_z)$ and we perform inference separately for each of the different dimensions. For each of these learning problems, we place a regular grid over the entire space for fixed points.

In macroscopic mapping, we have to learn large environments with lots of data. In such environments, when data arrives sequentially, it is possible to use the posterior distribution from the previous time step as the prior distribution in

the current time step before applying the update rules in (3)–(4). Furthermore, when data is obtained sequentially, we can start with the assumption that the environment is static and populate the 3D environment (training dataset) with quasi-Monte Carlo (QMC) samples of velocity zero to improve the learning efficiency. QMC sampling techniques are known to be sample efficient over Monte Carlo techniques [30]. In particular, Sobol and generalized Halton QMC sequences can populate the 3D free space more evenly [30], [31]. Trajectory data points that are in the neighborhood measured by the Euclidean distance of the populated data are removed from the dataset.

For the microscopic model, we are interested only in modeling the field of view of the ego vehicle (Figures 1c and 2). For such models, velocity information coming from automotive radar is used. Since automotive radar point clouds, unlike LIDAR measurements, are extremely sparse, modeling the epistemic uncertainty is important so that we know which of our velocity estimates are less reliable. Furthermore, automotive radar measurements tend to have higher noise levels and therefore, modeling the aleatoric uncertainty is equally important. Equation (6) is the combined aleatoric-epistemic uncertainty estimation.

E. Wasserstein robustness against input noise

Sensor measurements, especially radar measurements, are typically corrupted by some noise. In this section, we theoretically analyze whether the proposed model can withstand input perturbations [32].

Lemma 1: The squared 2-Wasserstein distance between a normal distribution $\mathcal{N}(\mathbf{x}, \nu^2 I)$ and a point $\mathbf{x}_m$ is $\mathcal{W}_2^2 = \|\mathbf{x} - \mathbf{x}_m\|_2^2 + \nu^2$.

Proof: By computing $\inf \mathbb{E}[\|\mathrm{x} - \mathrm{x}_m\|_2^2]$ between $\mathcal{N}(\mathbf{x}, C^2)$ and $\mathcal{N}(\mathbf{x}_m, C_m^2)$, it can be shown that $\mathcal{W}_2 = \|\mathbf{x} - \mathbf{x}_m\|_2^2 + \mathrm{Tr}\big(C^2 + C_m^2 - 2(C_m C^2 C_m)^{\frac{1}{2}}\big)$ [32]. For diagonal matrices C_m and $C = \nu I$, by reducing the Gaussian to a Dirac delta distribution (by taking the limit of the variance terms to zero), we obtain $\mathcal{W}_2 = \|\mathbf{x} - \mathbf{x}_m\|_2^2 + \nu^2$. ∎

Theorem 1: Expectation of estimations of the linear model $\mathbf{w}^\top \Phi(\mathbf{X})$ with weights $\{w_m\}_{m=1}^M$ where $w_m \sim \mathcal{N}(\mu_m, \sigma_m)$ in Bayesian dynamic fields are unaltered by the input noise $\mathcal{N}(0, \nu^2)$ under the 2-Wasserstein metric.

Proof: Let us rewrite the $\mathbf{w}^\top \Phi(\mathbf{X})$ introduced in Section III-B as a summation,

$$\mathbf{y} \approx \sum_{m=1}^M w_m k(\mathbf{x}, \mathbf{x}_m)$$
$$= \sum_{m=1}^M w_m \exp\big(-\gamma(\|\mathbf{x} - \mathbf{x}_m\|_2^2 + \nu^2)\big) \text{ (Lemma 1)}$$
$$= \sum_{m=1}^M w_m \exp(-\gamma\nu^2) \exp(-\gamma\|\mathbf{x} - \mathbf{x}_m\|_2^2)$$
$$= \sum_{m=1}^M w_m' \exp(-\gamma\|\mathbf{x} - \mathbf{x}_m\|_2^2)$$

TABLE I: Datasets

Dataset	Source	Description
Chunks	Synthetic	3 closely packed velocity clusters
Blobs	Synthetic	3 separated velocity clusters as blobs
Carla	Simulated	Automotive radar data
Astyx	Real	Automotive radar data
nuScenes	Real	Automotive radar data
AirSim	Simulated	60 drone trajectories to simulate UAM
Airport	Real	100 aircraft trajectories around an airport

TABLE II: Effect of dimension adjustment

$[\gamma_x, \gamma_y, \gamma_z]$	RMSE	MSLL
$[0.1, 0.1, 0.1]$	1.368	-1445
$[100, 0.1, 0.1]$	**0.778**	-1426
$[100, 100, 100]$	1.496	-1413

The measurement noise is absorbed into the distributions $\{w_m'\}_{m=1}^M$ with $w_m' \sim \mathcal{N}(\mu_m', \sigma_m')$ whose parameters are estimated during training. Therefore, the mean estimations are unaffected by noise. ∎

IV. EXPERIMENTS

A. Experimental setup

We studied the effectiveness of BDFs in constructing macroscopic and microscopic models from a variety of datasets summarized in Table I. These datasets were obtained from hi-fidelity simulators and real-world benchmark datasets. Since automotive radar is becoming increasingly popular in driverless cars, we included some of those datasets as well. The models we build using small automotive radar datasets, Carla, Astyx, and nuScenes are microscopic because they only model their surroundings. These datasets have only a few data points per scan (Figure 1c and 2b). AirSim is a dataset that we generated using the AirSim simulator [33]. It contains 66859 data points of drone trajectories (Figure 1b) in a large area $1000{\times}400{\times}60$ m^3. The airport dataset contains 128349 data points of real aircraft tracks within 30 nautical miles of the John F. Kennedy airport [29].

Twenty percent of each dataset is used as the test dataset. For large areas, we normalized data to be in a cube of between -1 and 1 and picked the hyperparameters γ and grid distance using cross validation. The parameters α and

(a) Chunk dataset   (b) $[0.1, 0.1, 0.1]$ (c) $[100, 0.1, 0.1]$
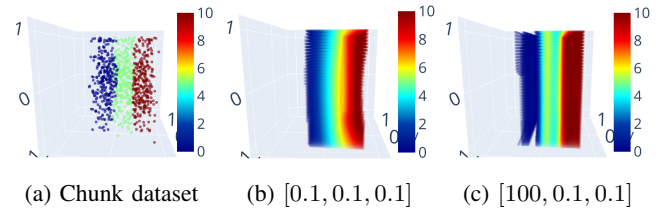
Fig. 3: Effect of dimension-adjusted kernels. Colors indicate velocity and the three values correspond to $[\gamma_x, \gamma_y, \gamma_z]$. In (c), when $\gamma_x > \gamma_y$ and $\gamma_x > \gamma_z$, the two boundaries between the three velocity clusters are much crispier.
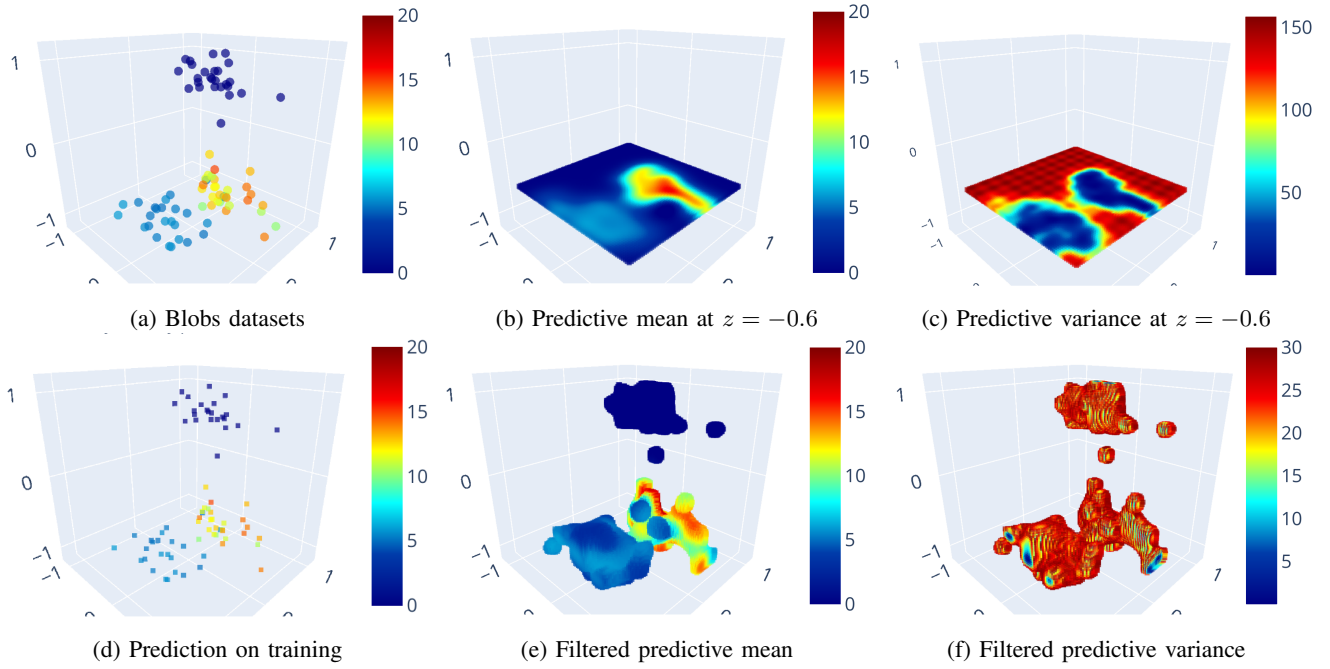
(a) Blobs datasets   (b) Predictive mean at $z = -0.6$ (c) Predictive variance at $z = -0.6$

(d) Prediction on training   (e) Filtered predictive mean   (f) Filtered predictive variance

Fig. 4: Modeling the x velocity component of the Blobs training dataset. (a) Each point in the 3D space has its own x velocity. We want to predict the mean and variance of any other point using these data points. (b) Predicted mean velocity for $z = -0.6$. (c) Predicted variance of velocity for $z = -0.6$. Note that in areas where we do not have training data, the variance is high. (d) Predictions on the training dataset. (e)–(f) Velocity is predicted for the entire cubes but only high confidence predictions ($\sigma_* \leq 30$) are shown.

β were set to 10^{-2} and 10^2, respectively. The code and video can be found at https://github.com/RansML/BDF. Experiments were run on a 3.30 GHz CPU. Since Gaussian process-based models are a common choice for modeling epistemic uncertainty in many robotics tasks [5], [34], we base-lined against full Gaussian process (FGP) [3] and its scalable approximations such as subset of data Gaussian process (SGP) [35] and more recent big data Gaussian process (BGP) [4]. GPflow [36] was used for benchmarking.

B. Effect of dimension adjustment

As the first experiment, we verify that dimension-adjusted kernels are useful to maintain sharp velocity transitions. For this purpose, we use the the Chunks dataset as it has sharp velocity transitions. Figure 3 shows that we get much crisper edges when we use a higher γ in the x direction. This is because gamma controls the contribution from each direction. Table II further corroborates that higher γ for the x-axis has the least root mean squared error (RMSE) for a similar mean standardized log loss (MSLL) [3].

C. Runtime and accuracy

Figure 4 shows a detailed example of how we can model the 3D space. Observe that the variance in areas where we do not have data is higher, indicating the epistemic uncertainty. Metrics for each dataset are reported in Table III. For almost all datasets, our model has the least training time to achieve similar accuracy to other methods. For small datasets (points < 1000), SGP is equivalent to FGP as the subset is the

TABLE III: Runtime and accuracy

Dataset	Method	Train time (s)	Query time (s)	RMSE
Chunks	BDF	**6.805**	0.673	0.778
	BGP	1819.469	0.075	1.258
	SGP	9.169	0.163	1.252
	FGP	9.169	0.163	1.252
Blobs	BDF	**0.448**	0.037	1.119
	BGP	89.411	0.038	0.669
	SGP	1.839	0.036	0.688
	FGP	1.839	0.036	0.688
Carla	BDF	**0.726**	0.044	4.581
	BGP	66.618	0.051	5.744
	SGP	1.849	0.032	5.992
	FGP	1.849	0.032	5.992
Astyx	BDF	5.517	0.087	0.329
	BGP	2575.300	0.059	0.275
	SGP	**4.983**	0.110	0.275
	FGP	**4.983**	0.110	0.275
nuScenes	BDF	**0.001**	0.010	0.370
	BGP	45.651	0.019	0.395
	SGP	0.587	0.016	0.396
	FGP	0.587	0.016	0.396
AirSim	BDF	**16.213**	0.533	0.514
	BGP	2979.046	0.125	0.736
	SGP	163.545	1.455	0.154
	FGP	∞ (est.)	∞ (est.)	n/a
Airport	BDF	**16.129**	0.856	1.801
	BGP	2966.236	0.154	2.139
	SGP	214.030	4.232	1.670
	FGP	∞ (est.)	∞ (est.)	n/a

Due to limited scalability of benchmarks, only 3.8% and 4.3% of data in AirSim and JFK, respectively, was used for all four methods.
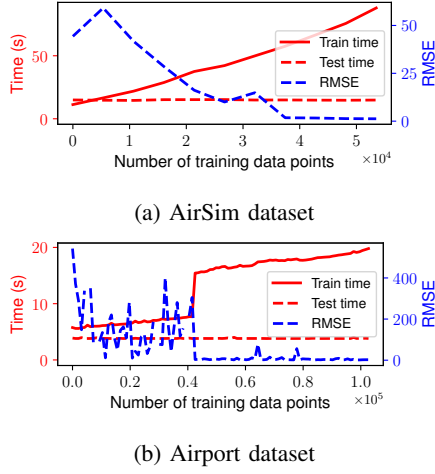
(a) AirSim dataset

(b) Airport dataset

Fig. 5: Effect of the increasing size of the dataset for the macroscopic model.

(a) Ground truth

(b) Mean prediction

Fig. 6: Bird's-eye view of 100 trajectories of the Airport dataset. The mean predictions of vertical velocities are similar to the ground truth.

same as the full dataset. Since BGP is based on a stochastic gradient approach, it is typically harder to optimize compared to other analytical forms and is not extremely useful in small data settings.

The efficiency of our model is more pronounced in large datasets such as Airport (Figure 6) and AirSim (Figure 7). BDF is at least 10 times faster than the second best performing model. This is because its asymptotic computational complexity is $\mathcal{O}(M^3)$ for M kernels. That is, the speed depends only on the number of kernels but not on the number of data points. In contrast, FGPs have a $\mathcal{O}(N^3)$ memory complexity for both training and query for N data points. In their sparse approximations, P *inducing points* are used to represent the key points in the dataset [38]. For $P \ll N$, the computational complexities of BGP and SGP are $\mathcal{O}(P^2 N)$ and $\mathcal{O}(P^3)$, respectively. Although SGP, in theory, has a similar asymptotic computational complexity, the major drawback of SGP is that it discards a large amount of data to achieve this speedup. In our method, every data point has an equal contribution when training the model.

Although BDFs have $\mathcal{O}(M^3)$ complexity, in practice, when M is finite (around 1000 in most of our experiments), we observe a slight increase with the number of data points due to the matrix product in (3). This can be observed in Figure 5 in which we separately train the model for an increasing number of data points. This slight increase of time is negligible compared to BGP and FGP in which the training time is 50 minutes before failing.

Since BDFs are parametric models that use kernels, we can update parts of the model or combine various models. This is because kernels are similarity functions, and therefore the weight parameters of a kernel located at $\tilde{x}$ are not affected by data far away from them. Similarly, a large-scale model can be easily decomposed to create light-weight models that cater to only a designated area of the environment.
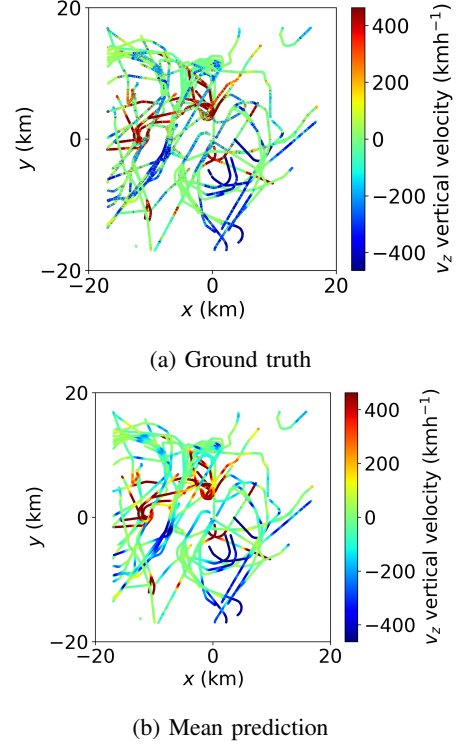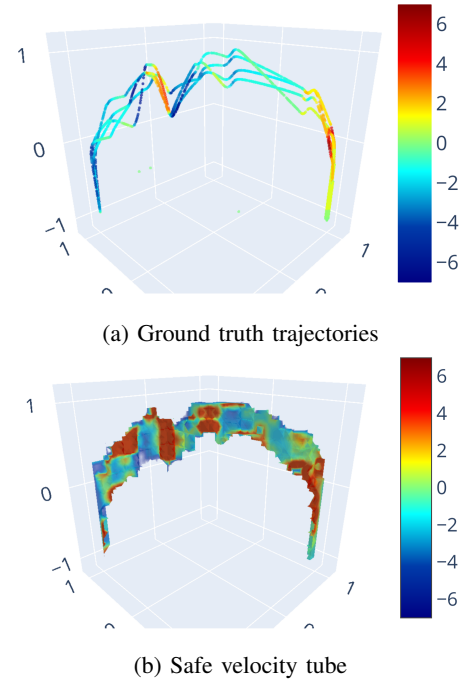

(a) Ground truth trajectories

(b) Safe velocity tube

Fig. 7: One of the "air roads" simulated in AirSim. Color indicates vertical velocity. A "3D tube" of velocity is obtained by filtering mean predictions below a given variance threshold. These tubes can be used for risk-aware control [37].

V. Conclusions

This paper presented Bayesian Dynamic Fields to model velocity in the 3D space. The velocity of the environment is represented as a continuous function that can be queried at arbitrary resolutions. The training procedure is equally suitable for both small and big data regimes, making it suitable to build microscopic and macroscopic transportation models. The model captures both velocity estimates as well as the uncertainty associated with those estimates. In conjunction with common environment representations such as occupancy maps in robotics, in the future, we will use the uncertainty estimates of velocity for decision-making under uncertainty and safety analysis [37].

Acknowledgment

References

[1] S. Choudhury, K. Solovey, M. J. Kochenderfer, and M. Pavone, "Efficient large-scale multi-drone delivery using transit networks," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 4543–4550.

[2] J. Holden and N. Goel, "Fast-forwarding to a future of on-demand urban air transportation," Uber, Tech. Rep., Oct. 2016.

[3] C. E. Rasmussen, "Gaussian processes in machine learning," in *Summer School on Machine Learning*, Springer, 2003, pp. 63–71.

[4] R. Senanayake, S. O'Callaghan, and F. Ramos, "Learning highly dynamic environments with stochastic variational inference," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2017, pp. 2532–2539.

[5] S. T. O'Callaghan and F. T. Ramos, "Gaussian process occupancy maps," *International Journal of Robotics Research (IJRR)*, vol. 31, no. 1, pp. 42–62, 2012.

[6] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[7] R. Senanayake and F. Ramos, "Bayesian hilbert maps for dynamic continuous occupancy mapping," in *Conference on Robot Learning (CoRL)*, 2017, pp. 458–471.

[8] S. McLeod and J. Xiao, "Navigating dynamically unknown environments leveraging past experience," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2019, pp. 29–35.

[9] T. Duong, M. Yip, and N. Atanasov, "Autonomous navigation in unknown environments with sparse bayesian kernel-based occupancy mapping," *arXiv preprint arXiv:2009.07207*, 2020.

[10] R. Senanayake, L. Ott, S. O'Callaghan, and F. T. Ramos, "Spatiotemporal hilbert maps for continuous occupancy representation in dynamic environments," in *Advances in Neural Information Processing Systems (NIPS)*, 2016, pp. 3925–3933.

[11] M. Itkina, K. Driggs-Campbell, and M. J. Kochenderfer, "Dynamic environment prediction in urban scenes using recurrent representation learning," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2019, pp. 2052–2059.

[12] M. Toyungyernsub, M. Itkina, R. Senanayake, and M. Kochenderfer, "Double-prong convlstm for spatiotemporal occupancy prediction in dynamic environments," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2021.

[13] R. Senanayake and F. Ramos, "Building continuous occupancy maps with moving robots," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2018, pp. 105–124.

[14] N. R. Lawrance and S. Sukkarieh, "Path planning for autonomous soaring flight in dynamic wind fields," in *IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2011, pp. 2499–2505.

[15] G. F. Homicz, "Three-dimensional wind field modeling: A review," *Sandia National Laboratories, SAND Report*, vol. 2597, 2002.

[16] R. Senanayake, M. Toyungyernsub, M. Wang, M. J. Kochenderfer, and M. Schwager, "Directional primitives for uncertainty-aware motion estimation in urban environments," in *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2020.

[17] J. Kivinen, A. J. Smola, and R. C. Williamson, "Online learning with kernels," in *Advances in Neural Information Processing Systems (NIPS)*, 2002, pp. 785–792.

[18] G. Vallicrosa and P. Ridao, "H-slam: Rao-Blackwellized particle filter SLAM using Hilbert maps," *Sensors*, vol. 18, no. 5, p. 1386, 2018.

[19] F. Ramos and L. Ott, "Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent," *International Journal of Robotics Research (IJRR)*, vol. 35, no. 14, pp. 1717–1730, 2016.

[20] W. Zhi, R. Senanayake, L. Ott, and F. Ramos, "Spatiotemporal learning of directional uncertainty in urban environments with kernel recurrent mixture density networks," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4306–4313, 2019.

[21] N. Cressie and C. K. Wikle, *Statistics for spatio-temporal data*. John Wiley & Sons, 2015.

[22] R. Senanayake, S. O'Callaghan, and F. Ramos, "Predicting spatiotemporal propagation of seasonal influenza using variational Gaussian process regression," in *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[23] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," in *European Conference on Computer Vision (ECCV)*, Springer, 2020, pp. 405–421.

[24] R. Senanayake, A. Tompkins, and F. Ramos, "Automorphing kernels for nonstationarity in mapping unstructured environments," in *Conference on Robot Learning (CoRL)*, 2018, pp. 443–455.

[25] M. Meyer and G. Kuschk, "Automotive radar dataset for deep learning based 3d object detection," in *2019 16th European Radar Conference (EuRAD)*, 2019, pp. 129–132.

[26] C. M. Bishop, *Pattern recognition and machine learning*. Springer, 2006.

[27] M. Gönen and E. Alpaydın, "Multiple kernel learning algorithms," *Journal of Machine Learning Research*, vol. 12, pp. 2211–2268, 2011.

[28] D. P. Thipphavong, R. Apaza, B. Barmore, V. Battiste, B. Burian, Q. Dao, M. Feary, S. Go, K. H. Goodrich, J. Homola, *et al.*, "Urban air mobility airspace integration concepts and considerations," in *Aviation Technology, Integration, and Operations Conference*, 2018.

[29] S. Jung and M. J. Kochenderfer, "Learning terminal airspace traffic models from flight tracks and procedures," in *Digital Avionics Systems Conference (DASC)*, 2019.

[30] C. Lemieux, *Monte Carlo and Quasi-Monte Carlo sampling*. Springer, 2009.

[31] A. Tompkins, R. Senanayake, P. Morere, and F. Ramos, "Black box quantiles for kernel learning," in *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019, pp. 1427–1437.

[32] D. Kuhn, P. M. Esfahani, V. A. Nguyen, and S. Shafieezadeh-Abadeh, "Wasserstein distributionally robust optimization: Theory and applications in machine learning," in *Operations Research & Management Science in the Age of Analytics*, INFORMS, 2019, pp. 130–166.

[33] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and Service Robotics*, 2017.

[34] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 408–423, 2013.

[35] R. Herbrich, N. D. Lawrence, and M. Seeger, "Fast sparse Gaussian process methods: The informative vector machine," in *Advances in Neural Information Processing Systems (NIPS)*, 2003, pp. 625–632.

[36] A. G. d. G. Matthews, M. van der Wilk, T. Nickson, K. Fujii, A. Boukouvalas, P. León-Villagrá, Z. Ghahramani, and J. Hensman, "GPflow: A Gaussian process library using TensorFlow," *Journal of Machine Learning Research*, vol. 18, no. 40, pp. 1–6, 2017.

[37] M. Cannon, Q. Cheng, B. Kouvaritakis, and S. V. Raković, "Stochastic tube mpc with state estimation," *Automatica*, vol. 48, no. 3, pp. 536–541, 2012.

[38] J. Quiñonero-Candela and C. E. Rasmussen, "A unifying view of sparse approximate Gaussian process regression," *Journal of Machine Learning Research*, vol. 6, no. Dec, pp. 1939–1959, 2005.