Scalable Gas Sensing, Mapping, and Path Planning via Decentralized Hilbert Maps

sensors

MDPI

Article

Scalable Gas Sensing, Mapping, and Path Planning via Decentralized Hilbert Maps

Pingping Zhu [1,*], Silvia Ferrari [1], Julian Morelli [1], Richard Linares [2] and Bryce Doerr [3]

[1] Sibley School of Mechanical and Aerospace Engineering, Cornell University, Ithaca, NY 14853, USA; ferrari@cornell.edu (S.F.); jam888@cornell.edu (J.M.)
[2] Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA; linaresr@mit.edu
[3] Department of Aerospace Engineering and Mechanics, University of Minnesota, Minneapolis, MN 55455, USA; doerr024@umn.edu
* Correspondence: pingping.zhu@cornell.edu

Received:18 December 2018; Accepted: 24 March 2019; Published: 28 March 2019

Abstract: This paper develops a decentralized approach to gas distribution mapping (GDM) and information-driven path planning for large-scale distributed sensing systems. Gas mapping is performed using a probabilistic representation known as a Hilbert map, which formulates the mapping problem as a multi-class classification task and uses kernel logistic regression to train a discriminative classifier online. A novel Hilbert map information fusion method is presented for rapidly merging the information from individual robot maps using limited data communication. A communication strategy that implements data fusion among many robots is also presented for the decentralized computation of GDMs. New entropy-based information-driven path-planning methods are developed and compared to existing approaches, such as particle swarm optimization (PSO) and random walks (RW). Numerical experiments conducted in simulated indoor and outdoor environments show that the information-driven approaches proposed in this paper far outperform other approaches, and avoid mutual collisions in real time.

Keywords: gas sensing; multi-agent systems; very large-scale robotic systems; information theory

1. Introduction

Fugitive emissions and the dispersion of pollutants in the atmosphere are significant concerns affecting public health as well as climate change. The accidental release of hazardous gases from both urban and industrial sources is responsible for a variety of respiratory illnesses and environmental concerns [1,2]. Many sensors are currently fabricated and deployed both indoors and outdoors for air quality data collection and communication. However, obtaining a spatial representation of a gas distribution is a challenging problem because existing mapping and fusion algorithms do not scale to networks of potentially hundreds of mobile and stationary sensors. The decentralized mapping and path-planning methods presented in this paper are applicable to very large-scale sensor systems, and as such can potentially be used to assess air quality, classify safe and hazardous areas based on the concentration of the harmful gases, and even localize fugitive emissions [3].

Due to the fundamental mechanisms of atmospheric gas dispersion, auxiliary tools are required to ensure early detection and to respond with appropriate counter measures by planning future measurements efficiently in both space and time. Methods for obtaining gas distribution maps (GDM) have recently been developed along two lines of research [4]. In the first line of research, a stationary sensor network is used to collect and fuse measurements to estimate the position of a source, typically requiring expensive and time-consuming calibration and data-recovery operations [5]. The use of

stationary sensors, however, is not typically effective or sufficiently expedient for gas sensing in response to high source rates of critical emissions. The second line of research, pursued in this paper, involves the use of mobile sensors, such as terrestrial robots equipped with gas sensors that can be controlled to rapidly and efficiently collect and fuse measurements over time. The latter approach provides for flexible sensor configurations that can respond to measurements online and thus can be applied to mapping gas distributions in unknown and complex environments.

Mobile gas sensing can be performed by a single robot [2,3,6–9] or by networks of robots also referred to as multi-agent systems (MAS) [10–14]. Compared to single robots, MAS present several advantages, including increased probability of success and improved overall operational efficiency [14], but also present additional technical challenges. In addition to requiring solving the GDM problem as a dynamical optimization problem, multi-agent path planning, coordination, communication, and fusion can become intractable as the number of robots increases [4,15–20].

MAS path planning and coordination can be achieved via centralized or decentralized methods, depending on the underlying communication infrastructure [21]. Centralized methods require persistent communication between a central station and every agent in the network, such that the central station can process and fuse all sensor measurements and use them to plan, or re-plan, the robot paths in a coordinated and collaborative fashion. Decentralized methods allow each robot to process its own measurements individually and then to communicate and fuse it with the measurements of a subset of collaborative robots, such as its nearest neighbors, with established connectivity. As a result, the performance of centralized methods depends entirely on the reliability of communication protocols under the operating conditions. Because fragile communication links are common in many hazardous scenarios, reliance on persistent communications with the central station and the associated power consumption can hinder or even prevent the applicability of MAS over large regions of interest that require repeated long-distance data transmission. Nevertheless, most of the existing GDM methods to date rely on centralized methods and algorithms [10,12,13].

One of the main challenges in developing decentralized GDM methods lies in solving the data fusion problem for neighboring robots such that each robot can build its own representation of the GDM based on local measurements, but also update it incrementally as new information is obtained from neighboring robots. Considering the limited communication bandwidth and computing resources of most autonomous robots, it is also impractical to expect each sensor to share all the raw measurement data with its neighbors. Therefore, the decentralized approach to gas source localization presented in [11] shares only the largest gas concentration and corresponding source position with its neighbors. However, this decentralized approach cannot be extended to high-performance GDM representations, such as Gaussian process (GP) mixture models [8] or kernel-based models [3,22], and fusing GDMs obtained by different robots would result in redundant and expensive computations that potentially operate on the same raw data repeatedly.

The Hilbert map is an alternate GDM that represents a probability map learned from local concentration measurements [7,23]. The novel GDM representation developed in this paper uses kernel logistic regression (KLR) to express the probability that the gas concentration at a certain position belongs to a predefined range as a Hilbert map function. By this approach, new decentralized fusion algorithms can be developed that present several advantages, including decentralized fusion, agent-level complete GDM representation, and update for decentralized decision making. Additionally, information fusion operations are implemented via simple summations and only the local Hilbert map needs to be shared among neighboring robots at every measurement update. As a result, at the end time of the task, information about the gas concentration distribution over the entire region of interest can be delivered to the client or operator even if only a few robots complete the mission. Two GDM-based path-planning methods, the entropy-based artificial potential field (EAPF) and the entropy-based particle swarm optimization (EPSO) algorithms, are presented in this paper, and the simulation results show that they significantly outperform existing algorithms.

2. Problem Formulation

Consider the problem of optimally planning the trajectory of a distributed sensing system comprised of N cooperative robots engaged in GDM through a large region of interest (ROI), denoted by $\mathcal{W} \subset \mathbb{R}^2$. In general, the gas concentration at a position $\mathbf{x} \in \mathcal{W}$, at time t, is modeled as a random variable $C(\mathbf{x}, t)$, thus the whole gas concentration is a random field. The gas distribution is considered constant over a time interval for simplicity and thus is modeled as a spatial function $c(\mathbf{x})$ defined over $\mathcal{W}$. The approach can be extended to time-varying concentrations by augmenting the dimensionality of the GDM.

Let $\mathcal{U} \subset \mathbb{R}^m$ denote the space of admissible actions or controls. The dynamics of each robot are governed by stochastic differential equations (SDEs),

$$\dot{\mathbf{x}}_n(t) = \mathbf{f}[\mathbf{x}_n(t), \mathbf{u}_n(t), t] + \mathbf{G}\mathbf{w}(t),\ \mathbf{x}_n(T_0) = \mathbf{x}_{n_0} \text{ and } n = 1, \cdots, N \tag{1}$$

where $\mathbf{x}_n(t) \in \mathcal{W}$ denotes the nth robot's position and the velocity at time t, $\mathbf{u}_n(t) \in \mathcal{U}$ denotes the nth robot action or control, and $\mathbf{x}_{n_0}$ denotes the robot initial conditions at initial time T_0. The robot dynamics in (1) are characterized by additive Gaussian white noise, denoted by $\mathbf{w}(t) \in \mathbb{R}^2$, and $\mathbf{G} \in \mathbb{R}^{2\times 2}$ is a known constant matrix. In this paper, we assume that the position of the nth robot is obtained by an on-board GPS and is denoted by $\hat{\mathbf{x}}_n(t)$, where $\hat{(\cdot)}$ denotes the estimated variable's value.

All N robots are equipped with identical metal oxide sensors in order to cooperatively map a time-constant gas distribution, where the quantity and position of gas plumes are unknown *a priori*. Because the reaction surface of a single gas sensor is very small ($\approx$1 cm^2), a single measurement from a gas sensor can only provide information about a very small area. Therefore, to increase the resolution of the GDM, a small metal oxide sensor array is used instead of a single oxide sensor for each robot. The area that is covered by this small metal oxide sensor array is called the field of view (FOV) of the robot, denoted by $\mathcal{S}(\mathbf{x}_n) \in \mathcal{W}$. Because the structure of the metal oxide sensor array is fixed, the positions of each metal oxide sensor can be approximated by the robot position measurements, denoted by $\hat{\mathbf{x}}_{n,m}$, $m = 1 : M$, where M is the number of oxide sensors on-board the nth robot. For example, for a 5×5 metal oxide sensor array, the number of sensors is $M = 25$.

The gas concentration measurement obtained by the mth sensor can be modeled as the sum of the actual concentration and measurement noise,

$$\hat{c}_{n,m}(\mathbf{x}_n) = c(\mathbf{x}_{n,m}) + \hat{v}(\mathbf{x}_{n,m}) \tag{2}$$

where $\hat{v}(\mathbf{x}_{n,m})$ is a realization of the random measurement noise, $V(\mathbf{x}_{n,m})$. Furthermore, the concentration measurement, $\hat{c}_{n,m}(\mathbf{x}_n)$, can be treated as a sample of the random variable $C(\mathbf{x}_{n,m})$, defined as

$$C(\mathbf{x}_{n,m}) \triangleq c(\mathbf{x}_{n,m}) + V(\mathbf{x}_{n,m}) \tag{3}$$

Then, the concentration measurement $\hat{c}_{n,m}(\mathbf{x}_n)$ can be considered as an observation of the random field $C(\cdot)$ at position $\mathbf{x}_{n,m}$.

Because the actual gas concentration, $c(\mathbf{x})$, and the distribution of the measurement noise, $V(\mathbf{x})$, are unknown, our objective is to approximate the random field $C(\cdot)$, and then use this approximated random field to estimate the gas concentration map. The cost function of the nth robot over a fixed time interval $[t_0,\ t_f]$ can then be expressed as an integral of the future measurement values, conditioned on past measurements, and the robot control usage, i.e.,

$$J_n = \int_{\mathcal{W}} H[C_n(\mathbf{x}|\mathcal{M}_n(T_f))]d\mathbf{x} + \int_{t_0}^{t_f} \mathbf{u}_n(t)^T \mathbf{R}\mathbf{u}_n(t)dt \tag{4}$$

where $H[C_n(\mathbf{x}|\mathcal{M}_n(t))]$ is the entropy of the random variable $C_n(\mathbf{x}|\mathcal{M}_n(t))$ approximated by the nth robot given all past measurements, $\mathcal{M}_n$. $\mathbf{R}$ is a positive definite matrix that weighs the importance of the elements of the control input $\mathbf{u}_n(t)$, and the superscript "T" denotes the matrix transpose.

The optimal planning problem is to find the optimal time history of the robot state $\mathbf{x}_n(t)$ and control $\mathbf{u}_n(t)$, for all $n = 1, \cdots, N$, so that the cost function J_n in (4) is minimized over $[t_0, t_f]$, and subject to (1).

Let the time interval $[t_0,\ t_f]$ be discretized into T_f equal time steps $\Delta t = (t_f - t_0)/T$ and let $t_k = t_0 + k\Delta t$ denote the discrete-time index with $k = 1, \cdots, T_f$. Then the objective function, J_n, can be rewritten as,

$$J_n = \int_{\mathcal{W}} H[C_n(\mathbf{x}|\mathcal{M}_n^{(T_f)})]d\mathbf{x} + \sum_{k=1}^{T_f} \mathbf{u}_{n,k}^T \mathbf{R}\mathbf{u}_{n,k} \tag{5}$$

where the subscript k indicates the kth time step and the superscript "(T_f)" indicates the time until the T_fth step. The term, $\mathcal{M}_n^{(T_f)} = \bigcup_{k=1}^{T_f} \mathcal{M}_{n,k}$, indicates the measurement history until the T_fth step and $\mathcal{M}_{n,k}$ denotes all the measurements obtained by the nth robot up to the kth time step.

3. Representation of GDM

To approximate the probability distribution of the continuous random variable $C_n(\mathbf{x}|\mathcal{M}_n^{(T)})$, there are several methods, including kernel density estimation (KDE) [24,25] and Gaussian process regression (GPR) [26]. In the KDE method, the learned probability density function (PDF) is assumed to be a weighted summation of many parameterized kernel functions, where the weight coefficients and the kernel parameters are critical and learned from the raw measurement data set. In the GPR method, the approximated PDF is assumed to be a Gaussian distribution, where a large matrix, the Gram matrix, is very critical and learned from the raw measurement data. For both of methods, the data fusion of two different measurement data sets and update of the approximated PDF are computationally demanding, because all computations must be implemented from a massive amount of raw data. It is difficult to obtain the updated parameters and coefficients from the previous parameters and coefficients directly.

The method in this paper overcomes this hurdle through an efficient fusion approach developed by approximating the continuous probability distribution by a discrete probability distribution. Denote the range of the concentration in the whole ROI by $\mathcal{R} = [L_c, H_c]$, and then divide the range into L concentration intervals denoted by $\mathcal{R}_1 = [L_1, H_1), \cdots, \mathcal{R}_l = [L_l, H_l), \cdots, \mathcal{R}_L = [L_L, H_L]$, where $H_l = L_{l+1}$ for $l = 1, \cdots, L-1$, and $L_1 = L_c$ and $H_L = H_c$. The cutoff coefficients, $\{(L_l, H_l)\}_{l=1}^L$, specify the concentration intervals of interest. Instead of approximating the PDF, this paper models the probabilities that the gas concentration at every position $\mathbf{x} \in \mathcal{W}$ belongs to the concentration interval $\{p_l(\mathbf{x})\}_{l=1}^L$, or,

$$p_l(\mathbf{x}) \triangleq P(C(\mathbf{x}) \in \mathcal{R}_l),\ l = 1, \cdots, L \tag{6}$$

where $P(\cdot)$ denotes the probability operator. Let $f_c(\mathbf{x})$ denote the PDF of the concentration $C(\mathbf{x})$ and $\Delta L_l = H_l - L_l$ denote the length of the lth concentration interval. The relationship between $p_l(\mathbf{x})$ and $f_c(x)$ can be expressed as,

$$p_l(\mathbf{x}) \triangleq P(C(\mathbf{x}) \in \mathcal{R}_l) = \int_{x \in \mathcal{R}_l} f_c(x)dx \tag{7}$$

where

$$f_c(L_l) = \lim_{\Delta L \to 0} \frac{p_l(\mathbf{x})}{\Delta L} \tag{8}$$

Therefore, the discrete distribution $\{p_l(\mathbf{x})\}_{l=1}^L$ can be used to describe the probability distribution of the continuous random variable $C(\mathbf{x})$ as $L \to \infty$. More importantly, it can be used for decision making, where the concentration intervals correspond to different levels of hazardous gas concentration. The Hilbert map method, developed by Fabio Ramos and Lionel Ott in [27] to model obstacle occupancy, is modified here to approximate the discrete distribution $\{p_l(\mathbf{x})\}_{l=1}^L$ over the entire ROI.

3.1. Mapping with KLR

A Hilbert map is a continuous probability map developed by formulating the mapping problem as a binary classification task. Let $\mathbf{x} \in \mathcal{W}$ be any point in $\mathcal{W}$ and $Y \in \{0,1\}$ be defined as a categorical random variable such that

$$Y = \begin{cases} 1, & \text{if } C(\mathbf{x}) \in \mathcal{R} \\ 0, & \text{if } C(\mathbf{x}) \notin \mathcal{R} \end{cases} \tag{9}$$

where the realization is denoted by y. The Hilbert map describes the probabilities $P(Y=1|\mathbf{x}) = p(\mathbf{x})$ and $P(Y=0|\mathbf{x}) = 1 - p(\mathbf{x})$ at the position $\mathbf{x}$.

Consider the training measurement data set, $\mathcal{M} = \{(\mathbf{x}_m, y_m)\}_{m=1}^{M}$, where $\mathbf{x}_m \in \mathcal{W}$ indicates the measurement position, the Boolean variable $y_m \in \{1,0\}$ indicates whether the concentration measurement, $\hat{c}(\mathbf{x}_m)$, belongs to the range, $\mathcal{R}$ (where $1 = Yes$ and $0 = No$), and M is the number of measurements. The probability $P(Y|\mathbf{x})$ is defined as,

$$p(\mathbf{x}) = P(Y=1|\mathbf{x}) = 1 - \frac{1}{1+\exp\left[f(\mathbf{x})\right]} = \frac{e^{f(\mathbf{x})}}{1+e^{f(\mathbf{x})}} \tag{10}$$

where,

$$1 - p(\mathbf{x}) = P(Y=0|\mathbf{x}) = \frac{1}{1+e^{f(\mathbf{x})}} \tag{11}$$

and $f \in \mathcal{H}$ is a Hilbert function defined on $\mathcal{W}$. $\mathcal{H}$ is a reproducing kernel Hilbert space (RKHS) associated with a kernel $k(\cdot,\cdot)$. The kernel mapping is denoted by $k(\mathbf{x},\cdot) = \varphi(\mathbf{x}), \varphi(\mathbf{x}) \in \mathcal{H}$, and is an injective map from $\mathcal{W}$ to $\mathcal{H}$ [28–32]. According to the kernel trick [28–32], the evaluation of the Hilbert function can be expressed in the form of inner product,

$$f(\mathbf{x}) = \langle f, \varphi(\mathbf{x}) \rangle_{\mathcal{H}} \tag{12}$$

where $\langle \cdot,\cdot \rangle_{\mathcal{H}}$ indicates the inner product in the RKHS. To learn the Hilbert map, the loss function J_H is defined as,

$$\begin{array}{rcl} J_H & = & \sum_{m=1}^{M} \ell_m + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \\ & = & \sum_{m=1}^{M} \left\{ \ln\left[1+e^{f(\mathbf{x}_m)}\right] - y_m f(\mathbf{x}_m) \right\} + \frac{\lambda}{2}\|f\|_{\mathcal{H}}^2 \end{array} \tag{13}$$

where $\ell_m = -\ln\left[P(Y=y_m|\mathbf{x}_m)\right]$ is the negative log-likelihood (NLL) of the data $(\mathbf{x}_m, y_m)$. Here, the term λ is the regularization term, which is a small user-defined positive scalar, $\lambda \ll 1$. Then the gradient of the loss function with respective to f is expressed as,

$$\begin{array}{rcl} g & = & \frac{\partial J_H}{\partial f} = \sum_{m=1}^{M} \left[\frac{e^{f(\mathbf{x}_m)}}{1+e^{f(\mathbf{x}_m)}} \varphi(\mathbf{x}_m) - y_m \varphi(\mathbf{x}_m) \right] + \lambda f \\ & = & \sum_{m=1}^{M} \left[P(Y=1|\mathbf{x}_m) - y_m \right] \varphi(\mathbf{x}_m) + \lambda f \\ & = & \Phi(\mathbf{p}-\mathbf{y}) + \lambda f \end{array} \tag{14}$$

where $\Phi = [\varphi(\mathbf{x}_1), \cdots, \varphi(\mathbf{x}_M)]$, $\mathbf{p} = [p(\mathbf{x}_1), \cdots, p(\mathbf{x}_M)]^T = \left[\frac{e^{f(\mathbf{x}_1)}}{1+e^{f(\mathbf{x}_1)}}, \cdots, \frac{e^{f(\mathbf{x}_M)}}{1+e^{f(\mathbf{x}_M)}}\right]^T$, and $\mathbf{y} = [y_1, \cdots, y_M]^T$. In addition, the Hessian operator is expressed as,

$$\begin{array}{rl} \mathbf{H} & = \frac{\partial g}{\partial f} = \frac{\partial}{\partial f}\left[\frac{e^{f(\mathbf{x}_1)}}{1+e^{f(\mathbf{x}_1)}}, \cdots, \frac{e^{f(\mathbf{x}_M)}}{1+e^{f(\mathbf{x}_M)}} \right] \Phi^T + \lambda \mathbf{I} \\ & = \left[\frac{\partial}{\partial f}\frac{e^{f(\mathbf{x}_1)}}{1+e^{f(\mathbf{x}_1)}}, \cdots, \frac{\partial}{\partial f}\frac{e^{f(\mathbf{x}_M)}}{1+e^{f(\mathbf{x}_M)}} \right] \Phi^T + \lambda \mathbf{I} \\ & = \left[\frac{e^{f(\mathbf{x}_1)}}{1+e^{f(\mathbf{x}_1)}} \frac{1}{1+e^{f(\mathbf{x}_1)}} \varphi(\mathbf{x}_1), \cdots, \frac{e^{f(\mathbf{x}_M)}}{1+e^{f(\mathbf{x}_M)}} \frac{1}{1+e^{f(\mathbf{x}_M)}} \varphi(\mathbf{x}_M) \right] \Phi^T + \lambda \mathbf{I} \\ & = \Phi \Lambda \Phi^T + \lambda \mathbf{I} \end{array} \tag{15}$$

where $\mathbf{I}$ is an identity operator (or matrix) defined in the domain of $\mathcal{H} \times \mathcal{H}$, so that for any function $h \in \mathcal{H}$, and $\mathbf{I}h = h$. Here, Λ is an $M \times M$ diagonal matrix defined as,

$$\Lambda = \mathbf{p}(\mathbf{1}_M - \mathbf{p})^T \tag{16}$$

and $\mathbf{1}_M = [1, \cdots 1]^T$ is an $M \times 1$ vector with all elements equal to one.

Using the Newton-Raphson method, the Hilbert function, is updated iteratively,

$$\begin{aligned} f_{i+1} &= f_i - \mathbf{H}_i^{-1} g_i = f_i - \left[\Phi \Lambda_i \Phi^T + \lambda \mathbf{I}\right]^{-1} g_i \\ &= f_i - \left[\Phi \Lambda_i \Phi^T + \lambda \mathbf{I}\right]^{-1} \left[\Phi(\mathbf{p}_i - \mathbf{y}) + \lambda f_i\right] \\ &= f_i - \left[\Phi \Lambda_i \Phi^T + \lambda \mathbf{I}\right]^{-1} \Phi(\mathbf{p}_i - \mathbf{y}) - \lambda \left[\Phi \Lambda_i \Phi^T + \lambda \mathbf{I}\right]^{-1} f_i \\ &\approx f_i - \left[\Phi \Lambda_i \Phi^T + \lambda \mathbf{I}\right]^{-1} \Phi(\mathbf{p}_i - \mathbf{y}) \\ &= f_i - \Phi \mathbf{R}_i (\mathbf{p}_i - \mathbf{y}) \end{aligned} \tag{17}$$

where $\mathbf{R}_i = \left[\Lambda_i \Phi^T \Phi + \lambda \mathbf{I}_M\right]^{-1}$, $\mathbf{I}_M$ is an $M \times M$ identity matrix, and the subscript "i" indicates the ith iteration for learning function f. The Hilbert function, $f(\mathbf{x})$, is evaluated iteratively as follows,

$$\begin{aligned} f_{i+1}(\mathbf{x}) &= \langle f_{i+1}, \varphi(\mathbf{x}) \rangle_{\mathcal{H}} \\ &= f_i(\mathbf{x}) - \varphi^T(\mathbf{x}) \Phi \mathbf{R}_i (\mathbf{p}_i - \mathbf{y}) \\ &= f_i(\mathbf{x}) - \mathbf{k}^T \mathbf{R}_i (\mathbf{p}_i - \mathbf{y}) \end{aligned} \tag{18}$$

where $\mathbf{k} = \Phi^T \varphi(\mathbf{x}) = [k(\mathbf{x}_1, \mathbf{x}), \cdots, k(\mathbf{x}_M, \mathbf{x})]^T$.

It can be seen from (18) that the evaluation of $f_{i+1}(\mathbf{x})$ only depends on the evaluations of $f_i(\mathbf{x})$ and $k(\mathbf{x}_m, \mathbf{x})$, $m = 1, \cdots, M$. Therefore, the evaluation $f_i(\mathbf{x})$ for each iteration is not needed if $\mathbf{x} \notin \{\mathbf{x}_1, \cdots, \mathbf{x}_M\}$. Instead, $f_i(\mathbf{x})$ can be calculated at the last iteration directly from the evaluations of $f_i(\mathbf{x}_m)$, such that

$$f_i(\mathbf{x}) = f_0(\mathbf{x}) - \mathbf{k}^T \sum_{i'=1}^{i} \mathbf{R}_{i'} (\mathbf{p}_{i'} - \mathbf{y}) = f_0(\mathbf{x}) - \mathbf{k}^T \mathbf{S}_i \tag{19}$$

where $f_0(\mathbf{x})$ is the initial function evaluation at $\mathbf{x}$, prior to learning the function from $\mathcal{M}$. The following matrix

$$\mathbf{S}_i \triangleq \sum_{i'=1}^{i} \mathbf{R}_{i'} (\mathbf{p}_{i'} - \mathbf{y}) \tag{20}$$

only depends on the measurement data, $\mathcal{M}$, and can be updated iteratively.

Furthermore, consider Q collocation points, $\mathcal{X}^c = \{\mathbf{x}_q^c \in \mathcal{W}\}_{q=1}^{Q}$, in the ROI, characterized by the same spatial interval, labeled by the superscript "c". Let $\Phi^c = \left[\varphi(\mathbf{x}_1^c), \cdots, \varphi(\mathbf{x}_Q^c)\right]$ denote the feature matrix of the collocation points. The evaluations of function $f(\mathbf{x}_q^c)$ at all collocation points can be updated by,

$$\mathbf{f}_{i+1} = \mathbf{f}_i - \Phi^{cT} \Phi \mathbf{R}_i (\mathbf{p}_i - \mathbf{y}) \tag{21}$$

where $\mathbf{f}_i = \Phi^{cT} f_i = \left[f_i(\mathbf{x}_1^c), \cdots, f_i(\mathbf{x}_Q^c)\right]^T$. According to (19), the evaluations, $\mathbf{f}_i = \Phi^{cT} f_i$, can be updated directly by,

$$\mathbf{f}_i = \mathbf{f}_0 - \Phi^{cT} \Phi \mathbf{S}_i \tag{22}$$

where $\mathbf{f}_0$ comprises the initial function evaluations at the collocation points prior to learning the function from the measurement data $\mathcal{M}$.

In summary, instead of learning the function f or its coefficients as in traditional KLR or GPR [26], we update the evaluations of $f(\mathbf{x})$ at the collocation points, $\mathcal{X}^c$. Given the Hilbert function f, the evaluations at the collocation points provide the Hilbert map, $\mathbf{f}$, defined as

$$\mathbf{f} \triangleq \Phi^{c\,T} f = \left[f(\mathbf{x}_1^c), \cdots, f(\mathbf{x}_Q^c) \right]^T \tag{23}$$

3.2. Temporal Update of Hilbert Map

The previous subsection develops a method for learning the Hilbert map from the measurement data set $\mathcal{M}$, obtained by the sensors during one time interval. This subsection considers a Hilbert map updated according to the next data set, $\mathcal{M}_k = \{(\mathbf{x}_{k,m}, y_{k,m})\}_{m=1}^{M_k}$, obtained during the kth time interval. Here, M_k is the number of measurements in the kth time interval, and $\mathcal{X}_k = \{\mathbf{x}_{k,m}\}_{m=1}^{M_k}$ and $\mathcal{Y}_k = \{y_{k,m}\}_{m=1}^{M_k}$ are the measurement positions and the corresponding classification estimates, respectively. To learn the next Hilbert map, the loss function over a period T can be expressed as,

$$J_T = \sum_{k=1}^{T} \left[\gamma(T-k) \sum_{m=1}^{M_k} \ell_{k,m} \right] + \frac{\lambda}{2} \|f\|_{\mathcal{H}}^2 \tag{24}$$

where λ is the regularization term as in (13), $\gamma(T-k)$ is the "forgetting factor", and $\ell_{k,m} = -\ln\left[P(Y = y_{k,m}|\mathbf{x}_{k,m})\right]$ is the NLL of the data $(\mathbf{x}_{k,m}, y_{k,m})$. Similarly to (14), the gradient is expressed as,

$$\begin{aligned} g_T &= \tfrac{\partial J_T}{\partial f} = \textstyle\sum_{k=1}^{T} \left[\gamma(T-k) \sum_{m=1}^{M_k} \tfrac{\partial}{\partial f} \ell_{k,m} \right] + \lambda f \\ &= \left[\Phi_1, \cdots, \Phi_T\right] \Gamma_T \left[(\mathbf{p}_1 - \mathbf{y}_1)^T, \cdots, (\mathbf{p}_T - \mathbf{y}_T)^T \right]^T \\ &= \tilde{\Phi}_T \Gamma_T \left[(\mathbf{p}_1 - \mathbf{y}_1)^T, \cdots, (\mathbf{p}_T - \mathbf{y}_T)^T \right]^T \end{aligned} \tag{25}$$

where $\Phi_k = \left[\varphi(\mathbf{x}_{k,1}), \cdots, \varphi(\mathbf{x}_{k,M_k})\right]$, $\mathbf{p}_k = \left[p(\mathbf{x}_{k,1}), \cdots, p(\mathbf{x}_{k,M_k})\right]^T = \left[\frac{e^{f(\mathbf{x}_{k,1})}}{1+e^{f(\mathbf{x}_{k,1})}}, \cdots, \frac{e^{f(\mathbf{x}_{k,M_k})}}{1+e^{f(\mathbf{x}_{k,M_k})}} \right]^T$, $\mathbf{y}_k = \left[y_{k,1}, \cdots, y_{k,M_k}\right]^T$ for $k = 1, \cdots, T$, and $\tilde{\Phi}_T = \left[\Phi_1, \cdots, \Phi_T\right]$. Furthermore, Γ_T is a diagonal matrix obtained by placing the vector, $\left[\begin{array}{cccccc} \gamma(T-1)\mathbf{1}_{M_1}^T & \cdots & \gamma(T-k)\mathbf{1}_{M_k}^T & \cdots & \gamma(0)\mathbf{1}_{M_T}^T \end{array} \right]^T$ on the diagonal of a zero matrix. In addition, as with (15), the Hessian operator can be expressed as,

$$\begin{aligned} H_T &= \tfrac{\partial g_T}{\partial f} = \left[\Phi_1, \cdots, \Phi_T\right] \Gamma_T \tilde{\Lambda}_T \left[\Phi_1^T, \cdots, \Phi_T^T\right]^T + \lambda \mathbf{I} \\ &= \tilde{\Phi}_T \Gamma_T \tilde{\Lambda}_T \tilde{\Phi}_T^T + \lambda \mathbf{I} \end{aligned} \tag{26}$$

where,

$$\tilde{\Lambda}_T = \begin{bmatrix} \Lambda_1 & \cdots & 0 & \cdots & 0 \\ \vdots & \ddots & & & \vdots \\ 0 & & \Lambda_k & & 0 \\ \vdots & & & \ddots & \vdots \\ 0 & \cdots & 0 & \cdots & \Lambda_T \end{bmatrix} \tag{27}$$

and $\Lambda_k = \mathbf{p}_k(\mathbf{1}_{M_k} - \mathbf{p}_k)^T$.

Then, the update rule for learning the Hilbert function, f, at the ith iteration is given by,

$$\begin{aligned} f_{T,i+1} &= f_{T,i} - \mathbf{H}_{T,i}^{-1} g_{T,i} \\ &\approx f_{T,i} - \left[\tilde{\Phi}_T \Gamma_T \tilde{\Lambda}_{T,i} \tilde{\Phi}_T^T + \lambda \mathbf{I} \right]^{-1} \tilde{\Phi}_T \Gamma_T \left[(\mathbf{p}_{1,i} - \mathbf{y}_1)^T, \cdots, (\mathbf{p}_{T,i} - \mathbf{y}_T)^T \right]^T \\ &= f_{T,i} - \tilde{\Phi}_T \tilde{\mathbf{R}}_i \Gamma_T \left[(\mathbf{p}_{1,i} - \mathbf{y}_1)^T, \cdots, (\mathbf{p}_{T,i} - \mathbf{y}_T)^T \right]^T \end{aligned} \tag{28}$$

where $\tilde{\mathbf{R}}_i = \left[\Gamma_T \tilde{\Lambda}_{T,i} \tilde{\Phi}_T \tilde{\Phi} + \lambda \mathbf{I}_{M_{tol}}\right]^{-1}$ and $M_{tol} = \sum_{k=1}^{T} M_k$. According to the above equation, the function, f, is expressed by using the set of all the measurement positions, $\{\bigcup_{k=1}^{T} \mathcal{X}_k\}$, and the corresponding coefficients.

To reduce the computational load, the forgetting factor, $\gamma(T-k)$, is modeled by,

$$\gamma(T-k) = \begin{cases} 1 & \text{if } T-k < \tau \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

such that each robot stores only the measurements obtained during the past τ time steps. By setting $\tau = 1$, the update rule (28) for learning the function f at the ith iteration is expressed as,

$$\begin{aligned} f_{T,i+1} &= f_{T,i} - \left[\Phi_T \Lambda_T \Phi_T^T + \lambda \mathbf{I}\right]^{-1} \Phi_T (\mathbf{p}_T - \mathbf{y}_T) \\ &= f_{T,i} - \Phi_T \mathbf{R}_{T,i} (\mathbf{p}_{T,i} - \mathbf{y}_T) \end{aligned} \tag{30}$$

where $\mathbf{R}_{T,i} = \left[\Lambda_i \Phi_T^T \Phi_T + \lambda \mathbf{I}_{M_T}\right]^{-1}$.

3.3. Approximation of GDM

The previous subsections present a method for learning the Hilbert function, f, from the available measurement data, comprising the measurement position data set $\mathcal{X}_k = \{\mathbf{x}_{k,m}\}_{m=1}^{M_k}$ and corresponding classification estimates $\mathcal{Y}_k = \{y_{k,m}\}_{m=1}^{M_k}$ for $k = 1, \cdots, T$ that indicate whether the concentration measurement at the measurement position belongs to the range, $\mathcal{R}$, defined in (9). If L classification estimates are obtained from the same concentration measurements, indicating that the concentration measurement belongs to the L concentration ranges, such as $\mathcal{R}_1, \cdots, \mathcal{R}_L$, respectively, using the learning method described in Section 3.2, one can approximate L Hilbert functions, $f_1, \cdots, f_L$. Furthermore, the probabilities in (6) can be evaluated at all the collocation points, $\mathcal{X}^c = \{\mathbf{x}_q\}_{q=1}^{Q}$, such that

$$p_l(\mathbf{x}_q^c) = 1 - \frac{1}{e^{f_l(\mathbf{x}_q^c)}},\ q = 1, \cdots, Q \text{ and } l = 1, \cdots, L \tag{31}$$

Now, consider Hilbert functions, $\{f_l\}_{l=1}^{L}$, approximated locally by each robot. Although all the concentration intervals are mutually exclusive and complete, i.e.,

$$\mathcal{R}_i \cap \mathcal{R}_j = \emptyset \tag{32}$$

$$\bigcup_{i=1}^{L} \mathcal{R}_i = [L_c, H_c],\ i, j = 1, \cdots, L \text{ and } i \neq j \tag{33}$$

it is not guaranteed that the sum of all the learned probabilities, $\{p_l(\mathbf{x}_q^c)\}_{l=1}^{L}$, is equal to one, or $\sum_{l=1}^{L} p_l(\mathbf{x}_q^c) = 1$. Therefore, the learned probabilities must be normalized to obtain the discrete distribution, $\boldsymbol{\pi}(\mathbf{x}_q^c) = [\pi_1(\mathbf{x}_q^c), \cdots, \pi_L(\mathbf{x}_q^c)]$. Each component of the discrete distribution is calculated by,

$$\pi_l(\mathbf{x}_q^c) = \frac{p_l(\mathbf{x}_q^c) \prod_{1 \leq l' \neq l \leq L} \left[1 - p_{l'}(\mathbf{x}_q^c)\right]}{p_0(\mathbf{x}_q^c)} \propto \frac{p_l(\mathbf{x}_q^c)}{1 - p_l(\mathbf{x}_q^c)} \tag{34}$$

where $p_0(\mathbf{x}_q^c)$ is a normalization term. Let $\bar{\mathbf{c}} = [\bar{c}_1, \cdots, \bar{c}_L]$ denote the medians of these intervals, where $\bar{c}_l = (H_l - L_l)/2$. Then, the expectation of the random variable $C(\mathbf{x}_q^c)$ can be expressed as,

$$\mathbb{E}[C(\mathbf{x}_q^c)] = \boldsymbol{\pi}(\mathbf{x}_q^c)^T \bar{\mathbf{c}} \tag{35}$$

where $\mathbb{E}(\cdot)$ is the expectation operator.

4. Information Fusion

In this section, the problem of information fusion between neighboring robots is considered, where each robot builds its own Hilbert function locally using a decentralized approach. Assume that all the robots use the same collocation points, $\mathcal{X}^c$. To limit the amount of communicated data, the evaluations of the Hilbert functions at the collocation points are shared among the robots, instead of the coefficients and parameters of the Hilbert functions.

4.1. Hilbert Map Fusion

Consider two robots that have each learned their respective Hilbert functions, f_1 and f_2, from two different sets of measurement data, $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. Then, the fused Hilbert function, f_F, can be obtained based on the following theorem.

Theorem 1. *Let $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ be two Hilbert functions defined on a workspace $\mathcal{W}$, and approximated by two robots based on their own measurement data sets, $\mathcal{M}_1$ and $\mathcal{M}_2$, respectively. These two Hilbert functions can be applied to calculate the conditional probability that the concentration is in the range $\mathcal{R}$ given the corresponding measurement data sets, as follows:*

$$p_1(\mathbf{x}|\mathcal{M}_1) = 1 - \frac{1}{1 + e^{f_1(\mathbf{x})}} \tag{36}$$

$$p_2(\mathbf{x}|\mathcal{M}_2) = 1 - \frac{1}{1 + e^{f_2(\mathbf{x})}} \tag{37}$$

Then, the fused conditional probability $p_F(\mathbf{x}|\mathcal{M}_1, \mathcal{M}_2)$ can be expressed as

$$p_F(\mathbf{x}|\mathcal{M}_1, \mathcal{M}_2) = 1 - \frac{1}{1 + e^{f_F(\mathbf{x})}} \tag{38}$$

where $f_F(\mathbf{x})$ is the fused Hilbert function. In addition, the fused Hilbert function, $f_F(\mathbf{x})$, can be calculated from the Hilbert functions, $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$, as follows,

$$f_F(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) - \ln \varepsilon \tag{39}$$

where $\varepsilon = \frac{p(\mathbf{x})}{1-p(\mathbf{x})}$ is the ratio between the prior probabilities, $P(C(\mathbf{x}) \in \mathcal{R}) = p(\mathbf{x})$ and $P(C(\mathbf{x}) \notin \mathcal{R}) = 1 - p(\mathbf{x})$, at $\mathbf{x} \in \mathcal{W}$.

The proof of Theorem 1 is provided in the Appendix A.1. According to Theorem 1, the following corollary can be obtained.

Corollary 1. *Assume that the prior probability is even, such as $p(\mathbf{x}) = 1/2$. Then, the ratio between the prior probabilities can be calculated by,*

$$\varepsilon = \frac{p(\mathbf{x})}{1 - p(\mathbf{x})} = 1 \tag{40}$$

and the fusion function can be rewritten as

$$f_F(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) \tag{41}$$

Because the prior concentration distribution is unknown, the Hilbert functions are fused according to (41) in Corollary 1, unless otherwise stated. Furthermore, assume that all the robots have the same collocation points, $\mathcal{X}^c$. The information fusion can be implemented by fusing the Hilbert maps among neighboring robots,

$$\mathbf{f}_{H,F} = \mathbf{f}_{H,1} + \mathbf{f}_{H,2} \tag{42}$$

where $\mathbf{f}_{H,1}$ and $\mathbf{f}_{H,2}$, and $\mathbf{f}_{H,F}$ are the Hilbert maps associated with the Hilbert functions $f_1(\mathbf{x})$, $f_2(\mathbf{x})$, and $f_F(\mathbf{x})$, respectively.

4.2. Communication Strategy

Any pair of robots in the network can share and update their Hilbert maps efficiently according to (42). To implement data fusion for a very large number of robots, however, a communication strategy is also required to determine how to share data between robots characterized by active communication links. In this paper, gas measurement data are communicated at every time step T_c. The communication protocol requires four steps at every time k, as follows. In the first, the N robots are partitioned into smaller communication networks according to their positions and communication range r_c, and $\mathcal{G}_\iota$ denotes the index set of the robot in the ιth communication network. Then, for any robot, a_n, with $n \in \mathcal{G}_\iota$, there exists another robot, $a_{n'}$, such that

$$d_{n,n'} = \|\mathbf{x}_n - \mathbf{x}_{n'}\| \leq r_c,\ n, n' \in \mathcal{G}_\iota \tag{43}$$

where $d_{n,n'}$ is the distance between robots a_n and $a_{n'}$, and $\|\cdot\|$ indicates the Euclidean norm.

As a second step, one robot in every communication network is selected as a temporary data center (TDC) denoted by $a_{n_\iota^*}$, $n_\iota^* \in \mathcal{G}_\iota$. The other robots in $\mathcal{G}_\iota$ send the Hilbert map change $\Delta\mathbf{f}_{n,k}$ to robot $a_{n_\iota^*}$. The Hilbert map change, $\Delta\mathbf{f}_{n,k}$, is defined as the change between the nth robot's Hilbert map at the current time step k and its Hilbert map at the previous communication time step, $k - T_c$, such that

$$\Delta\mathbf{f}_{n,k} = \mathbf{f}_{n,k} - \mathbf{f}_{n,k-T_c},\ n \in \mathcal{G}_\iota \tag{44}$$

where $\mathbf{f}_{n,k}$ and $\mathbf{f}_{n,k-T_c}$ denote the Hilbert maps of the nth robot at times kth and $(k - T_c)$th, respectively.

In the third step, the sum of the Hilbert map changes obtained from all robots in $\mathcal{G}_\iota$, defined as,

$$\Delta\mathbf{f}_{G_\iota,k} \triangleq \sum_{n\in\mathcal{G}_\iota} \Delta\mathbf{f}_{n,k} \tag{45}$$

is communicated back to the other robots, except the TDC robot, $a_{n_\iota^*}$. Finally, in the fourth step, all the robots update their own Hilbert maps by adding the received total Hilbert map changes, $\Delta\mathbf{f}_{G_\iota,k}$, to the current Hilbert maps, such that

$$\mathbf{f}_{n,k^+} = \mathbf{f}_{n,k} + \Delta\mathbf{f}_{G_\iota,k},\ n \in \mathcal{G}_\iota \tag{46}$$

where $\mathbf{f}_{n,k^+}$ represents the Hilbert map after the data fusion process.

5. Path-Planning Algorithms

In the previous sections, the Hilbert maps $\mathbf{f}_{l,n,k}$, $l = 1,..,L$, $n = 1,\cdots,N$ and $k = 1,\cdots,T_f$, are approximated and updated by the robots. The corresponding binary probabilities $\mathbf{p}_{l,n,k}$ can be calculated efficiently as follows

$$\begin{aligned} \mathbf{p}_{l,n,k} &= \left[p_{l,n,k}(\mathbf{x}_1^c), \cdots, p_{l,n,k}(\mathbf{x}_Q^c)\right]^T \\ &= \left[\frac{e^{f_{l,n,k}(\mathbf{x}_1^c)}}{1+e^{f_{l,n,k}(\mathbf{x}_1^c)}}, \cdots, \frac{e^{f_{l,n,k}(\mathbf{x}_1^c)}}{1+e^{f_{l,n,k}(\mathbf{x}_1^c)}}\right]^T \end{aligned} \tag{47}$$

and the entropy at each collocation point $\mathbf{x}_q^c \in \mathcal{X}^c$, $q = 1,\cdots,Q$, is obtained by

$$h_{l,n,k}(\mathbf{x}_q^c) = -p_{l,n,k}(\mathbf{x}_q^c)\log[p_{l,n,k}(\mathbf{x}_q^c)] - [1 - p_{l,n,k}(\mathbf{x}_q^c)]\log[1 - p_{l,n,k}(\mathbf{x}_q^c)] \tag{48}$$

Then, an entropy map, $\mathbf{h}_{n,k}$ is obtained from the vector,

$$\mathbf{h}_{n,k} = \left[h_{n,k}(\mathbf{x}_1^c), \cdots, h_{n,k}(\mathbf{x}_Q^c)\right]^T \tag{49}$$

where $h_{n,k}(\mathbf{x}_q^c) = \sum_{l=1}^{L} h_{l,n,k}(\mathbf{x}_q^c)$ denotes the sum of the entropy at the collocation point $\mathbf{x}_q^c$.

According to the cost function in (5), the objective of the nth robot is to minimize the uncertainty of the concentration distribution, which can be implemented by minimizing the entropy at all the collocation points, such that

$$\tilde{J}_n(k) = \sum_{q=1}^{Q} h_{n,k}(\mathbf{x}_q^c) \tag{50}$$

Therefore, $\tilde{J}_n(T_f)$ can be treated as an approximation of the term $\int_{\mathcal{W}} H[C_n(\mathbf{x}|\mathcal{M}_n^{(T_f)})]d\mathbf{x}$ in (5). In the rest of paper, $\tilde{J}_n(T_f)$ is applied as the approximation of the final cost function in (5) unless otherwise stated. In other words, the nth robot should visit the area around the collocation point $\mathbf{x}_q^c$, which has the higher value of $h_{n,k}(\mathbf{x}_q^c)$ at the kth time step. Based on this idea, two entropy-based path-planning algorithms are proposed in the following section to control the robots such that the value of the concentration measurements obtained by the robots in the ROI can be optimized over time.

5.1. Entropy-Based Artificial Potential Field

An information-driven approach is developed by planning the path of the robots such that they move towards collocation points with higher entropy. The collocation points are treated as goal positions characterized by attractive artificial potential fields defined as,

$$U^{att}(\mathbf{x}_n, \mathbf{x}_q^c) = -\frac{h_{n,k}(\mathbf{x}_q^c)}{\|\mathbf{x}_n - \mathbf{x}_q^c\|^2},\ q = 1, \cdots, Q \text{ and } n = 1, \cdots, N \tag{51}$$

where the superscript "att" indicates the attractive field. The corresponding attractive gradient is expressed as

$$\mathbf{g}^{att}(\mathbf{x}_n, \mathbf{x}_q^c) = \frac{\partial U_q^{att}(\mathbf{x}_n)}{\partial \mathbf{x}_n} = \frac{U_q^{att}(\mathbf{x}_n)}{\|\mathbf{x}_n - \mathbf{x}_q^c\|^4}(\mathbf{x}_n - \mathbf{x}_q^c),\ q = 1, \cdots, Q \tag{52}$$

Similarly to classic artificial potential field methods, the repulsive potential functions U^{rep} generated from the other robots are also considered, such that

$$U^{rep}(\mathbf{x}_n, \mathbf{x}_{n'}) = \begin{cases} \frac{1}{2}\left(\frac{1}{\|\mathbf{x}_n - \mathbf{x}_{n'}\|} - \frac{1}{r_{rep}}\right)^2, & \|\mathbf{x}_n - \mathbf{x}_{n'}\| \le r_{rep} \\ 0, & \|\mathbf{x}_n - \mathbf{x}_{n'}\| > r_{rep} \end{cases},\ 1 \le n \ne n' \le N \tag{53}$$

where r_{rep} is the distance threshold to create a repulsion effect on the robot. The repulsive gradient is expressed as

$$\mathbf{g}^{rep}(\mathbf{x}_n, \mathbf{x}_{n'}) = \begin{cases} -\left(\frac{1}{\|\mathbf{x}_n - \mathbf{x}_{n'}\|} - \frac{1}{r_{rep}}\right)\frac{\mathbf{x}_n - \mathbf{x}_{n'}}{\|\mathbf{x}_n - \mathbf{x}_{n'}\|^3}, & \|\mathbf{x}_n - \mathbf{x}_{n'}\| \le r_{rep} \\ 0, & \|\mathbf{x}_n - \mathbf{x}_{n'}\| > r_{rep} \end{cases},\ 1 \le n \ne n' \le N \tag{54}$$

Using (52) and (54), the total potential gradient for the nth robot is expressed as,

$$\mathbf{g}^{tol}(\mathbf{x}_n) = \xi \sum_{q=1}^{Q} \mathbf{g}^{att}(\mathbf{x}_n, \mathbf{x}_q^c) + \eta \sum_{1 \le n' \ne n \le N} \mathbf{g}^{rep}(\mathbf{x}_n, \mathbf{x}_{n'}) \tag{55}$$

where ξ and η are user-defined coefficients.

Based on the total gradient, $\mathbf{g}^{tol}(\mathbf{x}_n)$, the nth robot can be controlled to visit the collocation points with higher uncertainty and avoid collisions with other robots. The algorithm is developed based on the entropy attraction, which is named as EAPF algorithm.

5.2. Entropy-Based Particle Swarm Optimization

The particle swarm optimization (PSO) algorithm proposed by Clerc and Kennedy in [33] and its variants use a "constriction coefficient" to prevent the "explosion behavior" of the particles, and have been successfully applied to GDM and gas source localization problems [12,13]. In the original PSO algorithm and its variants, the concentration measurements are used to update the robot controls. Considering the different objective function in (5), an entropy-based PSO (EPSO) is proposed.

At the kth time step, the update of the nth robot position, $\mathbf{x}_{n,k}$, can be described as

$$\begin{aligned} \boldsymbol{\nu}_{n,k} &= \chi\left[\boldsymbol{\nu}_{n,k-1} + \psi_1 \mathbf{g}^{att}(\mathbf{x}_{n,k}, \mathbf{b}_{n,k}^{neig}) + \psi_2 \mathbf{g}^{att}(\mathbf{x}_{n,k}, \mathbf{b}_{n,k}^{glob})\right] + \eta \sum_{1\le n'\neq n\le N} \mathbf{g}^{rep}(\mathbf{x}_{n,k}, \mathbf{x}_{n',k}) \\ \mathbf{x}_{n,k+1} &= \mathbf{x}_{n,k} + \boldsymbol{\nu}_{n,k} \end{aligned} \tag{56}$$

where $\boldsymbol{\nu}_{n,k}$ represents the velocity of the nth robot at the kth time step, $\mathbf{b}_{n,k}^{neig}$ and $\mathbf{b}_{n,k}^{glob}$ are the best neighboring and global collocation points, respectively. The best neighboring collocation point, $\mathbf{b}_{n,k}^{neig}$, is determined by,

$$\mathbf{b}_{n,k}^{neig} = \underset{\mathbf{x}\in\mathcal{X}^c,\ \|\mathbf{x}_{n,k}-\mathbf{x}\|\le r_{neig}}{\arg\max} h_{n,k}(\mathbf{x}) \tag{57}$$

where r_{neig} is a coefficient which specifies the neighbor area from $\mathbf{x}_{n,k}$. The best global collocation point, $\mathbf{b}_{n,k}^{glob}$, is determined by

$$\mathbf{b}_{n,k}^{glob} = \underset{\mathbf{x}\in\mathcal{X}^c}{\arg\max} h_{n,k}(\mathbf{x}) \tag{58}$$

The learning coefficients, $\psi_1 \in [0, \bar{\psi}_1]$ and $\psi_2 \in [0, \bar{\psi}_2]$, are two uniform random variables. The constant parameter $\chi > 0$ prevents the explosion behavior. For efficient performance and prevention of the explosion behavior in (56), the parameter settings of the learning coefficients proposed in [12,13,33] is applied. The constriction parameter $\chi > 0$ is calculated by (refer to [33]),

$$\chi = \begin{cases} \frac{2\kappa}{\psi - 2 + \sqrt{\psi^2 - 4\psi}}, & \text{if } \psi > 4 \\ \kappa, & otherwise \end{cases} \tag{59}$$

where $\psi = \bar{\psi}_1 + \bar{\psi}_2$ and $\kappa \in [0, 1]$.

6. Simulations and Results

The performance of the decentralized GDM methods presented in this paper is demonstrated on a gas sensing application, where information about gas concentration obtained by a large network of robots is fused and used for information-driven path planning in a decentralized approach. Two indoor and outdoor environments are simulated and used to test the proposed methods. The new entropy-based EAPF and EPSO path-planning algorithms are compared to the existing algorithms known as random walk (RW) and classical particle swarm optimization (CPSO) [12,13,33].

6.1. Indoor GDM Sensing

The decentralized sensing system consists of a network of $N = 100$ robots characterized by single integrator dynamics,

$$\dot{\mathbf{x}}_i = \mathbf{u}_i,\ i = 1, \ldots, N \tag{60}$$

where $\mathbf{x}_i = [x, y]^T$ is the robot state vector, x, y are the inertial coordinates, and $\mathbf{u}_i = [u_1, u_2]^T$ is the robot control vector comprised of the x- and y-velocity components. The robot state/position is assumed to be observable by a built-in GPS with zero-mean measure noise w, where w is Gaussian white noise $\mathcal{N}(\mathbf{0}, \Sigma)$ with $\Sigma = 0.05 \times \mathbf{I}_2$.. The above distributed sensing system is tasked with mapping a gas distribution in an ROI $\mathcal{W} = [0, L_x] \times [0, L_y]$ where $L_x = 200$ m and $L_y = 160$ m, with an unknown gas distribution shown in Figure 1, and over a fixed time interval $[0,\ T_f]$, where $T_f = 500$ min. The normalized gas concentration range $\mathcal{R} = [0, 100]$ (Figure 1) is divided into $L = 3$ intervals:

$\mathcal{R}_1 = [0, 30)$, $\mathcal{R}_2 = [30, 70)$, and $\mathcal{R}_3 = [70, 100]$, representing low-hazard, medium-hazard, and high-hazard concentrations, respectively.

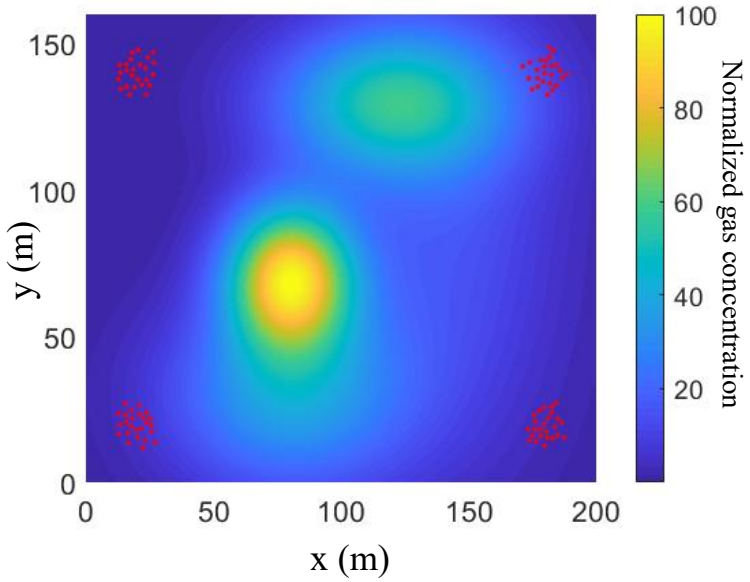

Figure 1. Gas concentration distribution in an indoor environment and initial robot deployment in ROI (plotted by red points).

The robots are initially deployed at four corners in the ROI by sampling from a given Gaussian Mixed Model with 4 components where $\mu_1 = [20, 20]^T$, $\mu_2 = [20, 140]^T$, $\mu_3 = [180, 20]^T$ and $\mu_4 = [180, 140]^T$, and identical covariance matrices, $\Sigma = \begin{bmatrix} 10 & 0 \\ 0 & 10 \end{bmatrix}$. The initial robot positions are shown as red points in Figure 1. Each robot is equipped with a small metal oxide sensor array comprised by $M = 5 \times 5 = 25$ gas sensors, where the spatial intervals between two sensors are all 10 cm. The FOV of each sensor array covers an area of 40×40 cm^2 in the ROI, which is very small relative to the whole ROI. Assume that the measurement noise of the gas concentration $V(\mathbf{x})$ is also characterized by Gaussian white noise, $\mathcal{N}(0, \Sigma_c(\mathbf{x}))$ with the covariance matrix $\Sigma_c(\mathbf{x}) = 0.5 \times \mathbf{I}_2$ everywhere in $\mathcal{W}$. All of robots communicate with each other at the same time with a communication period $T_c = 10$ min. The communication range is $r_c = 30$ m. In addition, $Q = 100 \times 100$ virtual collocation points are evenly deployed in the ROI to generate the Hilbert maps, where the Gaussian kernel is used for Hilbert function learning with a kernel size of $\sigma = 2$ m, and the parameter τ is set to 3.

The EPSO and the CPSO neighbor range coefficient, $r_{neig} = 5$ m, is applied to determine the best neighboring collocation points. The maximum velocity of each robot is 2 m/min for all simulations. The four path-planning algorithms, referred to as EAPF, EPSO, CPSO, and RW, are tested for mapping the gas distribution. The approximated cost function $\tilde{J}_n(k)$ defined in (50) is calculated by each robot at every time step as shown in Figure 2, where the solid line represents the mean of the approximated cost values over all of the robots at the kth time step, and the dashed line indicates one standard deviation above and below the mean. These cost histories reflect how effective the robots are at mapping the gas distribution. A lower value of $\tilde{J}_n(k)$ indicates that more information about the gas distribution is obtained by the robot at the kth step. It can be observed that the EAPF and EPSO algorithms achieve significantly better performance than the others. They can map the gas distribution more rapidly and completely, while the CPSO and RW algorithms perform poorly because they do not used prior measurement data for planning.

The means and standard deviations of the approximated cost function values of the all robots at the final time $k = T_f = 500$ for all the algorithms are tabulated in Table 1. It can be seen that the

approximated cost function obtained by the EPSO algorithm outperforms the other algorithms in the indoor environment. Let n^* denote the index of the robot who gets the lowest value of $\tilde{J}_n(T_f)$ at the final step. Then, the value of $\tilde{J}_{n^*}(k)$ represents the best performance among all the robots. The estimates of the gas concentration in the ROI are calculated based on these learned Hilbert maps according to (35), where $\bar{\mathbf{c}} = [15, 50, 85]$ is calculated from the cutoff coefficients. For each path-planning algorithm, three snapshots of the estimated GDMs generated by the n^*th robot in each simulation are presented in Figure 3, where the snapshots are taken at the $k = 20$th, $k = 100$th, $k = 500$th time steps, respectively. It can be seen that the robots controlled by the EAPF and EPSO algorithms obtain clean GDMs at the final time step, while the robots controlled by the other two existing algorithms cannot complete the mapping task in the given time period.

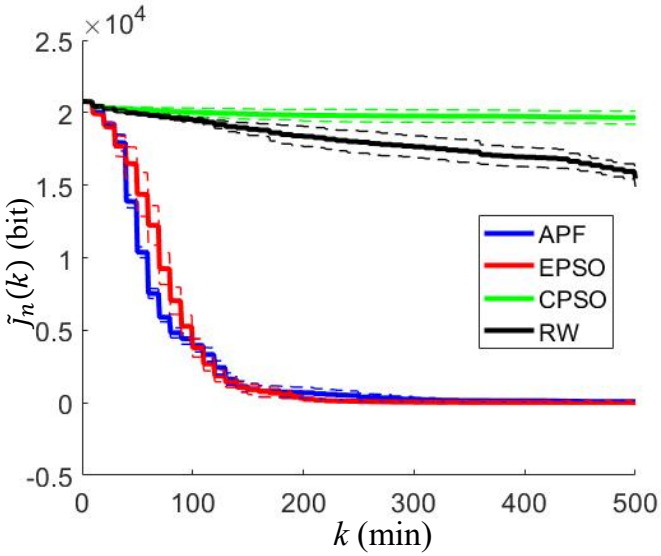

Figure 2. Approximated cost functions for different path-planning algorithms in the indoor environment, where the solid lines represent the mean of the approximated cost values over all the robots at the kth time step and the dashed lines indicate one standard deviation above and below the solid lines.

Table 1. Statistical results of the approximated cost function at the final time step in the indoor environment.

Algorithm	Mean of $\tilde{J}_n(T_f)$	Std. of $\tilde{J}_n(T_f)$
EAPF	72.01	66.64
EPSO	28.003	36.56
CPSO	19650.11	459.56
RW	15442.17	555.88

The normalized mean square errors (NMSE) between the estimated gas distribution and the actual gas distribution are calculated for each robot. The means and the corresponding standard deviations of the NMSE over the all robots for the different planning algorithms are reported in Table 2, which obviously shows that the EPSO algorithm outperforms the other algorithms to estimate the GDMs.

Table 2. Statistical results of the NMSE between the estimated GDMs and the actual GDMs at the final time step in the indoor environment.

Algorithm	Mean of NMSE	Std. of NMSE
EAPF	0.17521	0.00934
EPSO	0.17022	0.00432
CPSO	1.72230	0.03225
RW	1.20700	0.12516

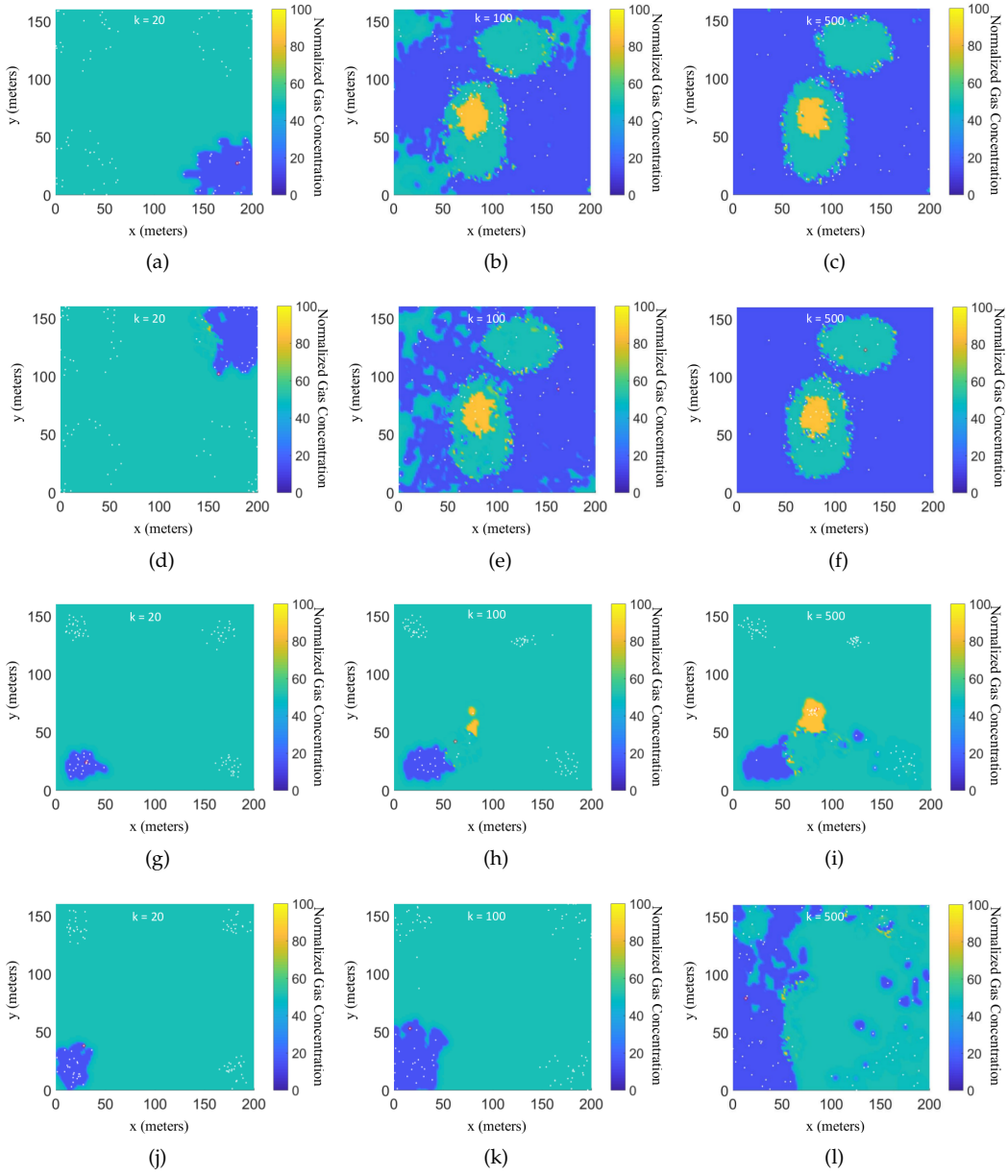

Figure 3. Evolution of the estimated gas distribution map in the indoor environment generated by the n^*th robot in each simulation at three instants in time, where the red point indicates the position of the n^*th robot and white points indicate the other robots, where the sub-figures in the first row (**a**–**c**) show the evolution of the estimated gas distribution obtained by the EAPF algorithm; the sub-figures in the second row (**d**–**f**) show the evolution of the estimated gas distribution obtained by the EPSO algorithm; the sub-figures in the third row (**g**–**i**) show the evolution of the estimated gas distribution obtained by the CPSO algorithm; and the sub-figures in the fourth row (**j**–**l**) show the evolution of the estimated gas distribution obtained by the RW algorithm.

6.2. Outdoor GDM

To verify the robustness and versatility of the proposed approaches, a GDM shown in Figure 4, originally presented in [34], is used to represent the gas distribution in an outdoor environment. The gas concentrations are normalized to the range $\mathcal{R} = [0, 100]$ for comparison like the indoor simulations. In this case, the intervals are chosen as $\mathcal{R}_1 = [0, 60)$, $\mathcal{R}_2 = [60, 80)$, and $\mathcal{R}_3 = [80, 100]$, to represent the plume shapes. All other parameters, including the initial robot positions, are the same as those used in the previous subsection.

The approximated cost function $\tilde{J}_n(k)$ obtained by all four algorithms is plotted in Figure 5. The approximated cost function at the final step are also tabulated in Table 3. Similarly to the indoor simulations, the gas concentration is estimated according to (35), where $\bar{\mathbf{c}} = [30, 70, 90]$ is calculated based on the cutoff coefficients. The evolution of the estimated GDM obtained by different algorithms is presented in Figure 6. Furthermore, the statistical results of the NMSE between the estimated GDM and the actual GDM are reported in Table 4.

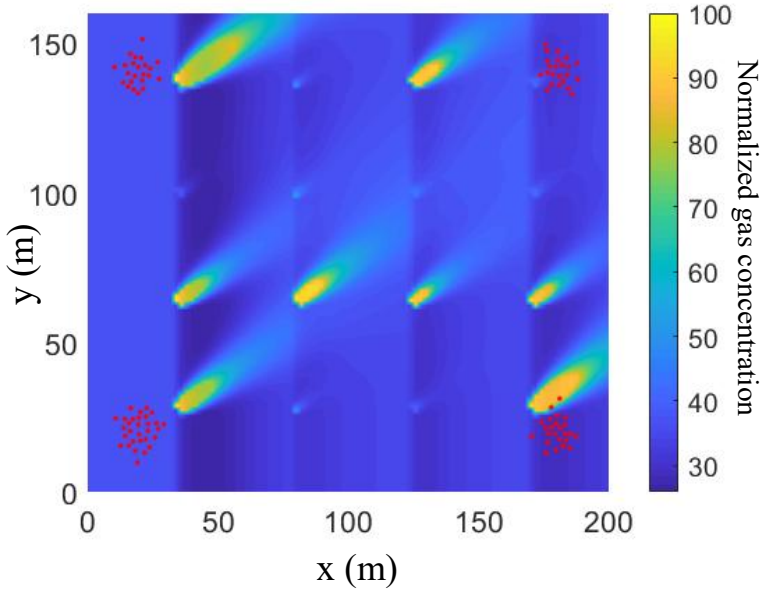

Figure 4. ROI and Gas concentration distribution in an outdoor environment and initial robot deployment in the ROI, where the red points represent the robots.

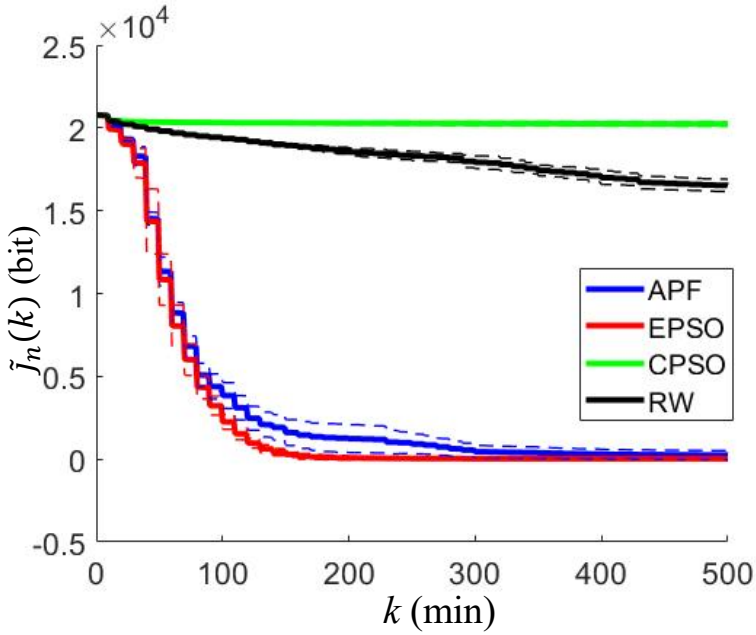

Figure 5. Approximated cost functions for different path-planning algorithms in the outdoor environment, where the solid lines represent the mean of the approximated cost values over all the robots at the kth time step and the dashed lines indicate one standard deviation above and below the solid lines.

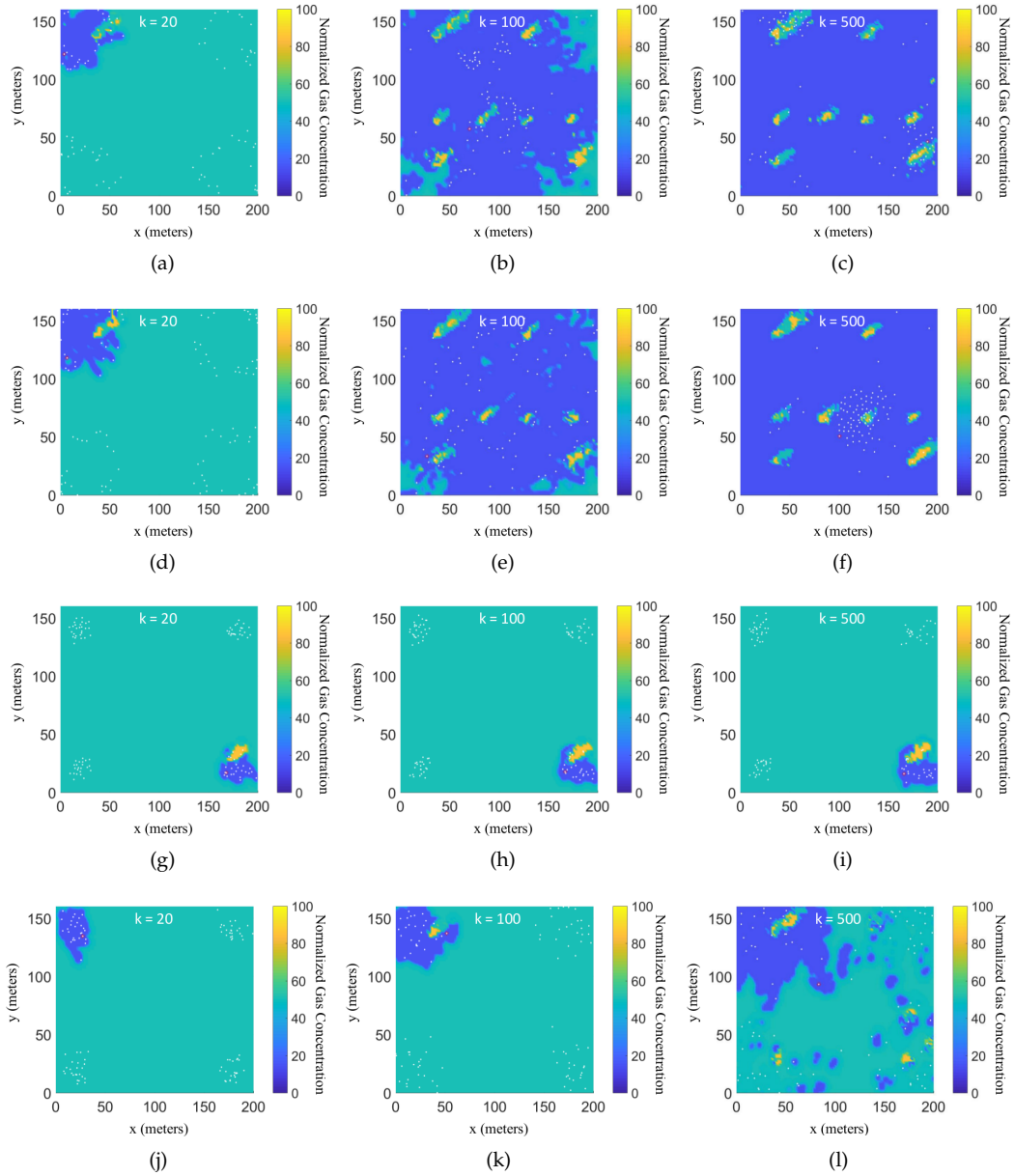

Figure 6. Evolution of the estimated gas distribution map in the outdoor environment generated by the n^*th robot in each simulation at three instants in time, where the red point indicates the position of the n^*th robot and white points indicate the other robots, where the sub-figures in the first row (**a–c**) show the evolution of the estimated gas distribution obtained by the EAPF algorithm; the sub-figures in the second row (**d–f**) show the evolution of the estimated gas distribution obtained by the EPSO algorithm; the sub-figures in the third row (**g–i**) show the evolution of the estimated gas distribution obtained by the CPSO algorithm; and the sub-figures in the fourth row (**j–l**) show the evolution of the estimated gas distribution obtained by the RW algorithm.

Table 3. Statistic results of the approximated cost function at the final time step in the outdoor environment.

Algorithm	Mean of $\tilde{J}_n(T_f)$	Std. of $\tilde{J}_n(T_f)$
EAPF	243.9842	263.3285
EPSO	19.7781	55.7657
CPSO	20232.4784	147.2573
RW	16510.9489	380.798

Table 4. Statistic results of the estimated gas distribution maps at final time step in the outdoor environment.

Algorithm	Mean of NMSE	Std. of NMSE
EAPF	0.05368	0.00506
EPSO	0.04272	0.00123
CPSO	0.51741	0.00402
RW	0.41734	0.01094

As expected, the results presented in Figures 5 and 6 and Tables 3 and 4 all show that the proposed EAPF and EPSO algorithms work well in the outdoor GDM problem, while the CPSO and RW algorithms cannot map the gas concentration in the entire workspace in the given time period. In addition, the EPSO algorithm significantly outperforms the other algorithms in all simulations.

7. Conclusions

This paper presents a decentralized framework for GDM and information-driven path planning in distributed sensing systems. GDM is performed using a probabilistic representation known as a Hilbert map and a novel Hilbert map fusion method is presented that quickly and efficiently combines information from many neighboring robots. In addition, two entropy-based path-planning algorithms, namely the EAPF and EPSO algorithms, are proposed to efficiently control all the robots to obtain the gas concentration measurements in the ROI. The proposed approaches are demonstrated on a system with hundreds of robots that must map a gas distribution collaboratively over a large ROI using on-board iron-oxide arrays and no prior information. The results show that through fusion and decentralized processing, the entropy of the gas map decreases over time, the robot paths remain safe (avoiding mutual collisions), and the entropy-based methods far outperform both traditional and random approaches.

Author Contributions: Conceptualization, P.Z., J.M. and S.F.; Formal analysis, P.Z. and J.M.; Funding acquisition, S.F.; Methodology, P.Z.; Software, P.Z.; Project administration, S.F.; Validation, B.D., R.L. and S.F.; Writing—original draft, J.M. and P.Z.; Writing—review and editing, all authors.

Funding: This research was partially funded by National Science Foundation grant ECCS-1556900 and the Office of Naval Research, Code 321.

Conflicts of Interest: The authors declare no conflict of interest and the funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A

Appendix A.1. Proof of Theorem 1

Given two Hilbert functions $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$, the conditional probabilities that the concentration at the position $\mathbf{x}$ is belong the range $\mathcal{R}$ are expressed as

$$p_1(\mathbf{x}|\mathcal{M}_1) = P(C(\mathbf{x}) \in \mathcal{R}|\mathcal{M}_1) \tag{A1}$$

$$p_2(\mathbf{x}|\mathcal{M}_2) = P(C(\mathbf{x}) \in \mathcal{R}|\mathcal{M}_2) \tag{A2}$$

For convenience, the event $\mathcal{C}(\mathbf{x}) \in \mathcal{R}$ and its complement even $\mathcal{C}(\mathbf{x}) \notin \mathcal{R}$ are denoted by e and e^C, respectively. Then, the fused conditional probability, $p_F(\mathbf{x}|\mathcal{M}_1, \mathcal{M}_2)$, can be expressed as,

$$
\begin{aligned}
p_F(\mathbf{x}|\mathcal{M}_1,\mathcal{M}_2) &= P(e|\mathcal{M}_1,\mathcal{M}_2) \\
&= \tfrac{P(e,\mathcal{M}_1,\mathcal{M}_2)}{P(\mathcal{M}_1,\mathcal{M}_2)} \\
&= \tfrac{P(e)P(\mathcal{M}_1|e)P(\mathcal{M}_2|e)}{P(e)P(\mathcal{M}_1|e)P(\mathcal{M}_2|e)+P(e^C)P(\mathcal{M}_1|e^C)P(\mathcal{M}_2|e^C)} \\
&= \tfrac{P(e)P(\mathcal{M}_1|e)P(e)P(\mathcal{M}_2|e)}{P(e)^2P(\mathcal{M}_1|e)P(\mathcal{M}_2|e)+\varepsilon P(e^C)^2P(\mathcal{M}_1|e^C)P(\mathcal{M}_2|e^C)} \\
&= \tfrac{P(e,\mathcal{M}_1)P(e,\mathcal{M}_2)}{P(\mathcal{M}_1)P(e|\mathcal{M}_1)P(\mathcal{M}_2)P(e|\mathcal{M}_2)+\varepsilon P(\mathcal{M}_1)P(e^C|\mathcal{M}_1)P(\mathcal{M}_2)P(e^C|\mathcal{M}_2)} \\
&= \tfrac{P(e|\mathcal{M}_1)P(e|\mathcal{M}_2)}{P(e|\mathcal{M}_1)P(e|\mathcal{M}_2)+\varepsilon P(e^C|\mathcal{M}_1)P(e^C|\mathcal{M}_2)} \\
&= \tfrac{p_1(\mathbf{x}|\mathcal{M}_1)p_2(\mathbf{x}|\mathcal{M}_2)}{p_1(\mathbf{x}|\mathcal{M}_1)p_2(\mathbf{x}|\mathcal{M}_2)+\varepsilon[1-p_1(\mathbf{x}|\mathcal{M}_1)][1-p_2(\mathbf{x}|\mathcal{M}_2)]} \\
&= \tfrac{\frac{e^{f_1(\mathbf{x})}}{1+e^{f_1(\mathbf{x})}}\frac{e^{f_2(\mathbf{x})}}{1+e^{f_2(\mathbf{x})}}}{\frac{e^{f_1(\mathbf{x})}}{1+e^{f_1(\mathbf{x})}}\frac{e^{f_2(\mathbf{x})}}{1+e^{f_2(\mathbf{x})}}+\varepsilon\frac{1}{1+e^{f_1(\mathbf{x})}}\frac{1}{1+e^{f_2(\mathbf{x})}}} \\
&= \tfrac{e^{f_1(\mathbf{x})+f_2(\mathbf{x})}}{e^{f_1(\mathbf{x})+f_2(\mathbf{x})}+\varepsilon} \\
&= \tfrac{e^{f_1(\mathbf{x})+f_2(\mathbf{x})-\ln\varepsilon}}{e^{f_1(\mathbf{x})+f_2(\mathbf{x})-\ln\varepsilon}+1} \\
&= \tfrac{e^{f_F}(\mathbf{x})}{e^{f_F}(\mathbf{x})+1}
\end{aligned}
\tag{A3}
$$

where,

$$
f_F(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) - \ln\varepsilon \tag{A4}
$$

□

References

1. Neumann, P.P.; Asadi, S.; Lilienthal, A.J.; Bartholmai, M.; Schiller, J.H. Autonomous gas-sensitive microdrone: Wind vector estimation and gas distribution mapping. *IEEE Robot. Autom. Mag.* **2012**, *19*, 50–61.
2. Rossi, M.; Brunelli, D. Autonomous gas detection and mapping with unmanned aerial vehicles. *IEEE Trans. Instrum. Meas.* **2016**, *65*, 765–775.
3. Lilienthal, A.; Loutfi, A.; Blanco, J.L.; Galindo, C.; Gonzalez, J. Integrating SLAM into gas distribution mapping. In Proceedings of the ICRA Workshop on Robotic Olfaction–Towards Real Applications, Freiburg, Germany, 19–21 September 2007.
4. Bayat, B.; Crasta, N.; Crespi, A.; Pascoal, A.M.; Ijspeert, A. Environmental monitoring using autonomous vehicles: A survey of recent searching techniques. *Curr. Opin. Biotechnol.* **2017**, *45*, 76–84.
5. Jelicic, V.; Magno, M.; Brunelli, D.; Paci, G.; Benini, L. Context-adaptive multimodal wireless sensor network for energy-efficient gas monitoring. *IEEE Sens. J.* **2013**, *13*, 328–338.
6. Ishida, H.; Nakamoto, T.; Moriizumi, T. Remote sensing of gas/odor source location and concentration distribution using mobile system. *Sens. Actuators B Chem.* **1998**, *49*, 52–57.
7. Lilienthal, A.; Duckett, T. Building gas concentration gridmaps with a mobile robot. *Robot. Auton. Syst.* **2004**, *48*, 3–16.
8. Stachniss, C.; Plagemann, C.; Lilienthal, A.J.; Burgard, W. Gas distribution modeling using sparse Gaussian process mixture models. In Proceedings of the Robotics: Science and Systems Conference 2008, Zürich, Switzerland, 25–28 June 2008; pp. 310–317.
9. Albertson, J.D.; Harvey, T.; Foderaro, G.; Zhu, P.; Zhou, X.; Ferrari, S.; Amin, M.S.; Modrak, M.; Brantley, H.; Thoma, E.D. A mobile sensing approach for regional surveillance of fugitive methane emissions in oil and gas production. *Environ. Sci. Technol.* **2016**, *50*, 2487–2497.
10. Hayes, A.T.; Martinoli, A.; Goodman, R.M. Distributed odor source localization. *IEEE Sens. J.* **2002**, *2*, 260–271.
11. Jatmiko, W.; Ikemoto, Y.; Matsuno, T.; Fukuda, T.; Sekiyama, K. Distributed odor source localization in dynamic environment. In Proceedings of the 2005 IEEE Sensors, Irvine, CA, USA, 30 October–3 November 2005; doi:10.1109/ICSENS.2005.1597684.

12. Akat, S.B.; Gazi, V.; Marques, L. Asynchronous particle swarm optimization-based search with a multi-robot system: Simulation and implementation on a real robotic system. *Turk. J. Electr. Eng. Comput. Sci.* **2010**, *18*, 749–764.
13. Turduev, M.; Cabrita, G.; Kırtay, M.; Gazi, V.; Marques, L. Experimental studies on chemical concentration map building by a multi-robot system using bio-inspired algorithms. *Auton. Agents Multi-Agent Syst.* **2014**, *28*, 72–100.
14. Sinha, A.; Kaur, R.; Kumar, R.; Bhondekar, A.P. Cooperative control of multi-agent systems to locate source of an odor. *arXiv* **2017**, arXiv:1711.03819.
15. Rudd, K.; Foderaro, G.; Ferrari, S. A generalized reduced gradient method for the optimal control of multiscale dynamical systems. In Proceedings of the 52nd IEEE Conference on Decision and Control, Florence, Italy, 10–13 December 2013; pp. 3857–3863.
16. Foderaro, G.; Ferrari, S.; Wettergren, T.A. Distributed optimal control for multi-agent trajectory optimization. *Automatica* **2014**, *50*, 149–154.
17. Ferrari, S.; Foderaro, G.; Zhu, P.; Wettergren, T.A. Distributed optimal control of multiscale dynamical systems: A tutorial. *IEEE Control Syst. Mag.* **2016**, *36*, 102–116.
18. Rudd, K.; Foderaro, G.; Zhu, P.; Ferrari, S. A Generalized Reduced Gradient Method for the Optimal Control of Very-Large-Scale Robotic Systems. *IEEE Trans. Robot.* **2017**, *33*, 1226–1232.
19. Foderaro, G.; Zhu, P.; Wei, H.; Wettergren, T.A.; Ferrari, S. Distributed optimal control of sensor networks for dynamic target tracking. *IEEE Trans. Control Netw. Syst.* **2018**, *5*, 142–153.
20. Doerr, B.; Linares, R.; Zhu, P.; Ferrari, S. Random Finite Set Theory and Optimal Control for Large Spacecraft Swarms. *arXiv* **2018**, arXiv:1810.00696.
21. Jiménez, A.; García-Díaz, V.; Bolaños, S. A decentralized framework for multi-agent robotic systems. *Sensors* **2018**, *18*, 417.
22. Lilienthal, A.; Duckett, T. Creating gas concentration gridmaps with a mobile robot. In Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No. 03CH37453), Las Vegas, NV, USA, 27–31 October 2003; Volume 1, pp. 118–123.
23. Moravec, H.; Elfes, A. High resolution maps from wide angle sonar. In Proceedings of the 1985 IEEE International Conference on Robotics and Automation, St. Louis, MO, USA, 25–28 March 1985; Volume 2, pp. 116–121.
24. Rosenblatt, M. Remarks on some nonparametric estimates of a density function. *Ann. Math. Stat.* **1956**, *27*, 832–837.
25. Parzen, E. On estimation of a probability density function and mode. *Ann. Math. Stat.* **1962**, *33*, 1065–1076.
26. Williams, C.K.; Rasmussen, C.E. *Gaussian Processes for Machine Learning*; MIT Press Cambridge, MA, USA, 2006; Volume 2.
27. Ramos, F.; Ott, L. Hilbert maps: Scalable continuous occupancy mapping with stochastic gradient descent. *Int. J. Robot. Res.* **2015**, *35*, 1717–1730.
28. Zhu, P.; Chen, B.; Príncipe, J.C. Extended Kalman filter using a kernel recursive least squares observer. In Proceedings of the 2011 International Joint Conference on Neural Networks, San Jose, CA, USA, 31 July–5 August 2011; pp. 1402–1408.
29. Zhu, P.; Chen, B.; Príncipe, J.C. A novel extended kernel recursive least squares algorithm. *Neural Netw.* **2012**, *32*, 349–357.
30. Zhu, P. Kalman Filtering in Reproducing Kernel Hilbert Spaces. Ph.D. Thesis, University of Florida, Gainesville, FL, USA, 2013.
31. Zhu, P.; Chen, B.; Príncipe, J. Learning nonlinear generative models of time series with a Kalman filter in RKHS. *IEEE Trans. Signal Process.* **2014**, *62*, 141–155.
32. Zhu, P.; Wei, H.; Lu, W.; Ferrari, S. Multi-kernel probability distribution regressions. In Proceedings of the 2015 International Joint Conference on Neural Networks (IJCNN), Killarney, Ireland, 12–17 July 2015; pp. 1–7.

33. Clerc, M.; Kennedy, J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evolut. Comput.* **2002**, *6*, 58–73.
34. Gemerek, J.R.; Ferrari, S.; Albertson, J.D. Fugitive gas emission rate estimation using multiple heterogeneous mobile sensors. In Proceedings of the 2017 ISOCS/IEEE International Symposium on Olfaction and Electronic Nose (ISOEN), Montreal, QC, Canada, 28–31 May 2017; pp. 1–3.