Sensors & Actuators: B. Chemical 419 (2024) 136267

Contents lists available at ScienceDirect

Sensors and Actuators: B. Chemical

journal homepage: www.elsevier.com/locate/snb

Super-resolution for Gas Distribution Mapping☆

Nicolas P. Winkler [a,b,*], Oleksandr Kotlyar [b], Erik Schaffernicht [b], Haruka Matsukura [c], Hiroshi Ishida [d], Patrick P. Neumann [a], Achim J. Lilienthal [b,e]

[a] *Bundesanstalt für Materialforschung und -prüfung (BAM), Unter den Eichen 87, Berlin, 12205, Germany*
[b] *Örebro University, Fakultetsgatan 1, Örebro, 70281, Sweden*
[c] *University of Electro-Communications, 1-5-1 Chofugaoka, Chofu, Tokyo, 182-8585, Japan*
[d] *Tokyo University of Agriculture and Technology, 2-24-16 Nakacho, Koganei, Tokyo, 184-8588, Japan*
[e] *Technical University of Munich (TUM), Chair "Perception for Intelligent Systems", Arcisstrasse 21, München, 80333, Germany*

ARTICLE INFO

Dataset link: https://github.com/BAMresearch/SRGDM

Keywords:
Gas Distribution Mapping
Spatiotemporal interpolation
Mobile Robot Olfaction
Sensor network
Deep learning

ABSTRACT

Gas Distribution Mapping (GDM) is a valuable tool for monitoring the distribution of gases in a wide range of applications, including environmental monitoring, emergency response, and industrial safety. While GDM is actively researched in the scope of gas-sensitive mobile robots (Mobile Robot Olfaction), there is a potential for broader applications utilizing sensor networks. This study aims to address the lack of deep learning approaches in GDM and explore their potential for improved mapping of gas distributions. In this paper, we introduce Gas Distribution Decoder (GDD), a learning-based GDM method. GDD is a deep neural network for spatial interpolation between sparsely distributed sensor measurements that was trained on an extensive data set of realistic-shaped synthetic gas plumes based on actual airflow measurements. As access to ground truth representations of gas distributions remains a challenge in GDM research, we make our data sets, along with our models, publicly available. We test and compare GDD with state-of-the-art models on synthetic and real-world data. Our findings demonstrate that GDD significantly outperforms existing models, demonstrating a 35% improvement in accuracy on synthetic data when measured using the Root Mean Squared Error over the entire distribution map. Notably, GDD appears to have superior capabilities in reconstructing the edges and characteristic shapes of gas plumes compared to traditional models. These potentials offer new possibilities for more accurate and efficient environmental monitoring, and we hope to inspire other researchers to explore learning-based GDM.

1. Introduction

Gas Distribution Mapping (GDM) provides information about the spatial distribution of gases in a given environment. This information is valuable for a wide range of applications, including industrial process control, environmental monitoring, and emergency response [1–3].

Environmental monitoring tasks have traditionally been performed using stationary sensor networks. However, mobile robots are becoming increasingly popular for such tasks, as they can actively explore the environment and prioritize different areas of interest. Nevertheless, both stationary sensor networks and mobile robots can only capture data at a limited number of locations, and therefore, interpolation techniques are required to create high-resolution gas distribution maps by filling the gaps between measurement locations in both space and time. Interpolation methods such as Kriging [4] have been traditionally used for spatial interpolation in geographical information systems. However, they may not always provide the required precision in GDM tasks [5], as gas distribution is a highly dynamic process influenced by various factors like obstacles, weather conditions, and sources of gas emission.

Our primary research goal is to obtain improved GDM models that are able to map accurate gas distributions based on sparsely located sensor measurements and compare them with current state-of-the-art GDM models. Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNNs) are able to capture complex spatial and temporal patterns, as demonstrated in related gas sensing tasks such as target discrimination with e-nose systems [6].

In this paper, we use a transposed CNN architecture trained with synthetic gas concentration data based on real wind data. We then evaluate the model on both synthetic and real-world gas distribution

---

☆ Funding: This research was funded by SAF€RA within the project RASEM and JSPS, Japan (KAKENHI Grant Number 22H04952 and 22K12124).

* Corresponding author at: Bundesanstalt für Materialforschung und -prüfung (BAM), Unter den Eichen 87, Berlin, 12205, Germany.
E-mail address: mro@bam.de (N.P. Winkler).

[1] https://github.com/BAMresearch/SRGDM.

https://doi.org/10.1016/j.snb.2024.136267
Received 31 May 2023; Received in revised form 7 March 2024; Accepted 7 July 2024

data. Our models and data are published online on Github,[1] allowing others to reproduce and build upon our work. The main contributions of the paper are:

1. Gas Distribution Decoder (GDD): We introduce a specific deep GDM method for spatiotemporal interpolation of spatially sparse sensor measurements.
2. We collected an extensive data set of synthetic gas distribution maps based on actual airflow measurements and made it publicly available. As generating ground truth maps is nearly impossible, this data set provides a valuable resource for researchers in this field.
3. We carried out and presented a detailed comparative evaluation of GDD with state-of-the-art models on synthetic and real gas distribution data.

In the following, we first give an overview of related work (Section 2) before we describe our synthetic and real-world data sets and model architecture (Section 3). We evaluate our model and compare it with traditional GDM models in Section 4. In Section 5, we discuss the results and potential future work in this area. We hope this work will inspire further research in the field of GDM with employment of modern deep-learning techniques.

2. Related work

Gas distribution mapping involves spatiotemporal interpolation of measurements. One popular method for estimating missing values is through Gaussian Processes. [5,7,8]. These are especially popular in environmental monitoring, where they are known as Kriging [4] and are commonly used for interpolating large-scale areas. Another method used in environmental monitoring is land-use regression, which uses land-use characteristics (e.g., land cover, population density) to estimate levels of air pollution at a fine spatial scale [9–11].

There have been several demonstrated approaches for GDM in Mobile Robot Olfaction (MRO), which are suitable for small to medium-sized areas. The most common ones are Gaussian Markov Random Field (GMRF), Kernel DM+V, and Echo State Maps (ESM). GMRF is a probabilistic framework that models multivariate data and builds upon a graph structure to define the conditional relationships between variables. [12]. This graph structure enables precise control over information transfer between cells, making it possible to create obstructions such as walls that block the flow of air. In the context of MRO, this structure is utilized for joint estimation of wind and gas maps [13], and it has been further improved in terms of computational efficiency [14].

Kernel DM+V is a statistical method that models the distribution of gas sources. It treats distribution mapping as a density estimation problem and models both the distribution mean and variance. This method has been found to be effective in evaluating the quality of the map regarding the data likelihood. Gas measurements are considered as noisy samples from the distribution that needs to be modeled. Kernel DM+V has been extended to three dimensions with wind measurements [15] and to account for temporal dynamics [16].

The Echo State Map (ESM) approach [17] is another model that incorporates temporal input data. ESM connects Gaussian Processes to an Echo State Network (ESN) which is a type of RNN used to learn temporal dependencies between sensor measurements. This makes ESM the first model to utilize a neural network for GDM. Since then, modern neural networks, such as CNN, have been utilized for GDM-related tasks. For example, [18] trained a CNN with synthetic data to act as a surrogate plume model in a simple indoor environment.

Outside of MRO, neural networks were successfully applied to the task of spatiotemporal interpolation, e.g., for precipitation nowcasting [19], air pollution modeling [20–22], or radio environment mapping [23]. One of the latest trends are Physics-Informed Neural Networks (PINNs) which are trained to respect specific physical laws or constraints. Combining the expressiveness of neural networks with physical constraints has been shown to be useful for solving a wide range of partial and ordinary differential equations [24–26].

Inspired by these recent advances in deep learning, this paper focuses on applying modern neural networks for GDM. The term *super-resolution* is commonly used in computer vision to describe the reconstruction of high-resolution images from low-resolution source images [27]. By training a model with high-resolution output images, the model learns to recognize and reconstruct patterns from lower-dimensional input images. Therefore, we interpret the task of spatial interpolation between sparsely distributed measurements as a super-resolution task. Previous results have been published in [28,29]. In [28], we proposed an encoder–decoder architecture, which we now simplify to a decoder architecture. In [29], we addressed the temporal dimension, i.e., sequential input data. Here, we compare the decoder approach with state-of-the-art models and analyze its behavior in different scenarios. Additionally, we make our models and data sets publicly available to facilitate research on this topic.

3. Material and methods

In this chapter, we first describe the synthetic and experimental data. After that, we present the architectures of our neural networks and their implementation. Lastly, we describe the state-of-the-art models we used for comparison and our evaluation metrics.

3.1. Data generation

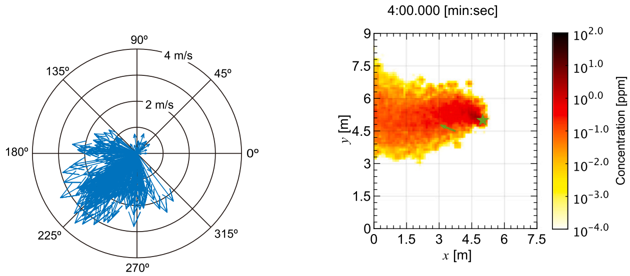
3.1.1. Generation of synthetic data

Traditional machine learning models typically require large amounts of training data that are costly to obtain with real experiments. While techniques like one-shot learning attempt to address this issue [30,31], access to high-resolution ground truth representations of the gas distribution remains a limitation. Therefore, we generated synthetic gas distributions based on actual, single-point wind data collected during outdoor experiments.

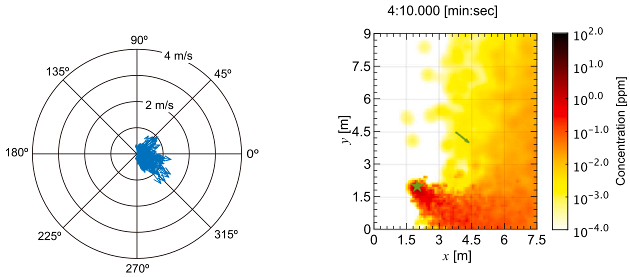
We decided on an experiment area of $9 \times 7.5\,\mathrm{m}^2$, in conjunction with related real-world experiments (Section 3.1.2). This area is divided into 30 grid cells, each measuring $1.5 \times 1.5\,\mathrm{m}^2$, with a virtual sensor placed in the center of each cell. To accommodate potential particle backflow due to variable wind directions, we simulated an overall area of $18 \times 15\,\mathrm{m}^2$.

In separate simulations, we simulated single ethanol vapor sources of size $0.02 \times 0.02\,\mathrm{m}^2$ in each of the 30 grid cells. These sources are placed slightly below and to the left of the corresponding virtual sensor (see Figs. 1(a) and 1(b)). This placement is intentional to avoid direct alignment with the sensor, thereby preventing an immediate maximal sensor response to the gas. The gas distribution was calculated using GADEN [32], which treats a fixed number of gas molecules as a single gas puff called a filament [33] and calculates the transport and diffusion of these gas puffs. By tracking only the center position and width of each filament, the computational cost is greatly reduced compared to using a computational fluid dynamics (CFD) solver to calculate gas diffusion and advection. Each gas source position was simulated with 24 different wind data sets for $300\,\mathrm{s}$. Subsequently, we omitted each simulation's first $30\,\mathrm{s}$ to allow full plume developments.

All simulations were calculated with a time step of $0.1\,\mathrm{s}$ and a grid size of $0.1 \times 0.1\,\mathrm{m}^2$. It is worth noting that the grid size was larger than the size of the gas source. However, since GADEN represents gas concentrations with Gaussian kernels and simulates the transport of gas filaments using a Lagrangian specification of the flow field, it does not rely on the simulation grid for the gas transport simulation. The gas concentration is calculated by summing up all the filaments whose Gaussian concentration distribution has at least some overlap with the cell. Therefore, the calculated gas concentration in each cell in the grid represents the average concentration in the cell and is assumed to be uniform in the cell [33]. The gas concentration in parts per trillion (ppt) of every grid cell was saved every $1\,\mathrm{s}$.



(a) Wind vectors and a sample with gas source in cell {3,4}.



(b) Wind vectors and an example gas source in cell {5,2}.

{1,1}	{1,2}	{1,3}	{1,4}	{1,5}
{2,1}	{2,2}	{2,3}	{2,4}	{2,5}
{3,1}	{3,2}	{3,3}	{3,4}	{3,5}
{4,1}	{4,2}	{4,3}	{4,4}	{4,5}
{5,1}	{5,2}	{5,3}	{5,4}	{5,5}
{6,1}	{6,2}	{6,3}	{6,4}	{6,5}

(c) A gas source is simulated with 24 wind data sets in each cell. The cells are $1.5 \times 1.5\ \mathrm{m}^2$ large, resulting in a total experiment area of $9 \times 7.5\ \mathrm{m}^2$.

Fig. 1. We simulated 30 different gas source positions with 24 real wind data sets. Figs. 1(a) and 1(b) show all wind vectors of two different simulations, along with a sample snapshot during the corresponding simulation (previously published in [28]). Each configuration was simulated for 300 s, using only the last 270 s to account for properly developed plumes. Saved with a frequency of 1 Hz, this results in $30 \times 24 \times 270 = 194,400$ gas distribution samples.

3.1.2. *Real-world experiments*

The wind data sets used to generate the synthetic data were obtained during a series of outdoor experiments in Tokyo, Japan, between April 2019 and October 2021. A single-point anemometer positioned at the center of a sensor grid array was utilized to measure wind speeds and directions. The grid array was comprised of 30 metal oxide gas sensors, which were factory-calibrated Figaro TGS 2620 models. The sensors were placed at equal distances of 1.5 meters from each other, resulting in a grid of $9 \times 7.5\ \mathrm{m}^2$, similar to the simulation setup. These sensors were equally spaced and were responsible for detecting the distribution of ethanol. The sensor responses were obtained through a 12-bit analog-to-digital converter. Multiple experiments were conducted with different source positions. The setup used for these experiments is depicted in Fig. 1(c). Initial results of this series of experiments were previously published in [34,35].

3.2. *Description of data sets*

3.2.1. *Pre-processing*

For simplicity, we will refer to each gas distribution snapshot as an "image". Cropping the simulation results to the experimental area results in 90 × 75 pixels. We downscaled all images to 30 × 25 pixels to allow for more compact network architectures. These images' pixel values correspond to gas concentration in PPT, and some pixels have very high values, in the range of millions, leading to right-skewed data sets. To address this, we logarithmically scaled all data sets, clamped the values between 0 and the maximum value of the training set, and normalized all sets between 0 and 1 using min–max normalization. These images serve as the ground truth y of the gas distribution.

To mimic the spatially sparse sensor network with equally distributed 6 × 5 sensors, we create input X by directly extracting pixel values at each sensor position in the 30 × 25 grid. This approach ensures that the input data is not an averaged, downsized representation of the ground truth but a collection of point measurements taken by the sensor network.

3.2.2. *Synthetic data sets*

We divided our synthetic data into two parts: one for training the model and the other for testing it. As explained in Section 3.1.1, we simulated each gas source position using 24 different wind sets. We assigned simulations of twelve wind data sets each to the training and test data. Although this 50/50 split may not be optimal, we obtained the simulations of test data chronologically after the training data and deliberately avoided re-balancing the data sets to prevent any bias. Training and test data each consist of 97,200 samples generated by simulating 30 source positions with twelve wind sets for 270 s. For evaluation of the model's generalization ability with k-fold cross-validation, the training data set is further divided into training and validation subsets during the training process.

3.2.3. *Real-world "Tokyo data set"*

We took the raw sensor response data from multiple outdoor experiments of the experiment series described in Section 3.1.2 for further evaluation. This data set also contains experiments without an anemometer. While each pixel of the synthetic data represents the gas concentration in ppt, the pixels now represent the voltage sensor output. We log-scaled, and min–max normalized the samples according to the sensor's inherent minimum and maximum output, such that 1 corresponds to the sensor's maximum detection limit to ethanol and 0 corresponds to no detection of ethanol. In total, this data set consists of 342,000 samples. We do not have full ground truth data, only the measurements from the 6 × 5 sensor grid.

3.3. *Network architecture*

3.3.1. *Non-Sequential Gas Distribution Decoder (GDD)*

Our transposed CNN model upsamples the 6 × 5 input image to a higher-resolution 30 × 25 output image. We refer to this model as "non-sequential" as it only takes a single time step or image as input, and as "decoder", as it is inspired by the decoder part of the autoencoder concept [36–38]. We interpret the low-dimensional 6 × 5 image as an encoded vector of size $[1, 6, 5]$ (i.e., [channels, height,

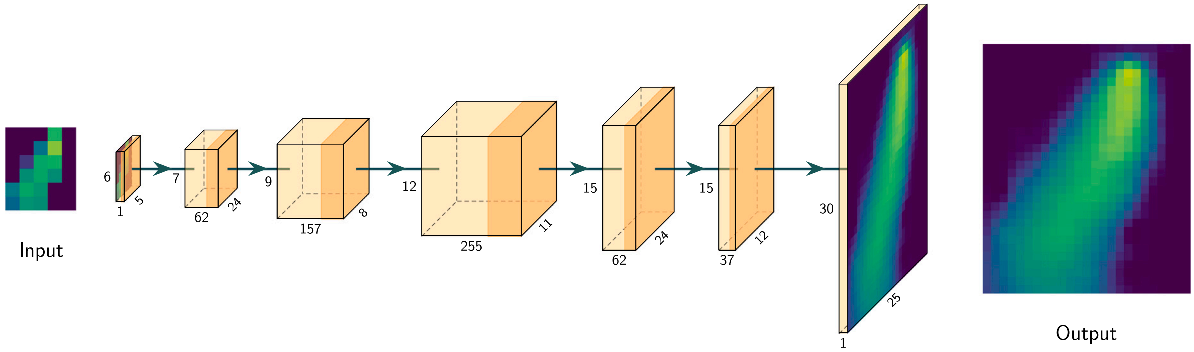


Fig. 2. Architecture of the Gas Distribution Decoder (GDD). The shaded rectangular boxes represent the matrix dimensions of each layer [channels, width, height]. The darker shades on the right half additionally symbolize a transposed convolutional operation, and the arrows symbolize the ReLU activation function. For example, the innermost layer contains 255 matrices of size 12 × 11 and is followed by a transposed convolution and a ReLU function.

width]). The decoder architecture consists of multiple transposed convolutional layers. A transposed convolution is commonly used in CNNs for upsampling input data. It is the reverse of a convolution, which is used to downsample data. The transposed convolutional layer operates similarly to a convolutional layer by learning a set of filters to expand the spatial dimension of the input data [39]. Rectifier linear unit (ReLU) activation functions are used between the transposed convolutions to set negative values to zero.

$$\mathrm{ReLU}(x) = x^{+} = \max(0, x) \tag{1}$$

The dimensions of the height and width of the internal layers in the neural network are determined through manual, heuristic selection. The remaining parameters were determined with tree-structured Parzen estimation [40] during the development process. Our preliminary studies involved the use of neural networks with two to nine layers. While it is generally believed that more layers allow for learning more complex patterns in the data [41], the best model we trained is a six-layer model. This model is displayed in Fig. 2.

3.3.2. Sequential Gas Distribution Decoder (sGDD)

The general potential of CNNs for spatial and temporal input has been demonstrated for video processing in previous studies [42,43]. Here, we utilize a much simpler approach and stack a sequence of input images to a three-dimensional input of size $[k, 6, 5]$ to the model, with k being the sequence length of the input. Accordingly, the non-sequential GDD can be interpreted as a variation of the sGDD with $k = 1$. Aside from this, no architectural differences exist between the sGDD and the GDD depicted in Fig. 2.

3.4. Implementation and training scheme

The models were implemented using PyTorch.[2] We utilized PyTorch Lightning[3] and Optuna[4] to facilitate the training process. The training process was carried out using the Adam optimizer on an Nvidia A100 and an Intel Xeon Gold 6342 CPU.

We first combine a hyperparameter study with 10-fold cross-validation to obtain robust hyperparameters. Therefore, we randomly divide the 360 simulations of the training data (30 gas source locations × 12 wind sets = 360 simulations) into ten subsets, train one model on nine subsets, and evaluate it on the remaining subset. We minimize the averaged Mean Squared Error (MSE, see Appendix B) and evaluate the robustness of this model by comparing its variance across all folds. After obtaining the hyperparameters, we train a final baseline model on the whole available training data. This model is then applied to the test data. To prevent overfitting the training data, we shuffle and augment our data sets by randomly flipping each sample vertically and horizontally with a probability of 0.5.

To evaluate the performance of noisy data, we also trained a separate model with random Gaussian noise $\mathcal{N}$ added to the input data. Specifically, we added a noise term ε with mean $\mu = 0$ and standard deviation $\sigma = 0.05$ to the input X to simulate noise in the data. These parameters were chosen heuristically based on our real-world sensor data.

$$X_{noise} = X + \varepsilon \tag{2}$$

$$\varepsilon \sim \mathcal{N}(\mu = 0, \sigma = 0.05) \tag{3}$$

This allows us to assess the robustness of the models to noise in the input data. If not stated otherwise, we refer to the trained model without this noise term.

3.5. State-of-the-art models

We compare our results with three algorithms that approach GDM differently: Kernel DM+V, GMRF, and ESM. In short, Kernel DM+V is a non-parametric approach to density estimation [44], while GMRF takes into account prior knowledge of gas distribution patterns and models the correlation between cells [12,13]. ESM uses an Echo State Network (ESN) to mimic a mobile robot's observations, with the ESN's weights interpolated using Gaussian Processes [17]. Further information on these algorithms and the implementations we used can be found in Appendix A.

4. Results

In this section, we evaluate the generalization ability and present the prediction results of our decoder approach, comparing them to selected state-of-the-art models. We tested our models on a synthetic test set and applied them to real-world data. Our chosen evaluation metrics are described in detail in Appendix B.

4.1. Generalization ability

To evaluate the model's generalization capabilities, we adopted a dual-phase k-fold cross-validation approach, examining the variations in gas source positions and wind conditions separately. Additional experiments assessing the model's robustness to sensor outages, noisy input data, and multiple gas sources are available in Appendix D.

Firstly, we divided the data set into ten folds, each including three unique gas source positions. This ensured that the model was tested on gas source positions that it had not seen before. The analysis revealed that the model performed consistently well across the folds, with a

---

[2] https://pytorch.org/.

[3] https://www.pytorchlightning.ai/.

[4] https://optuna.org/.

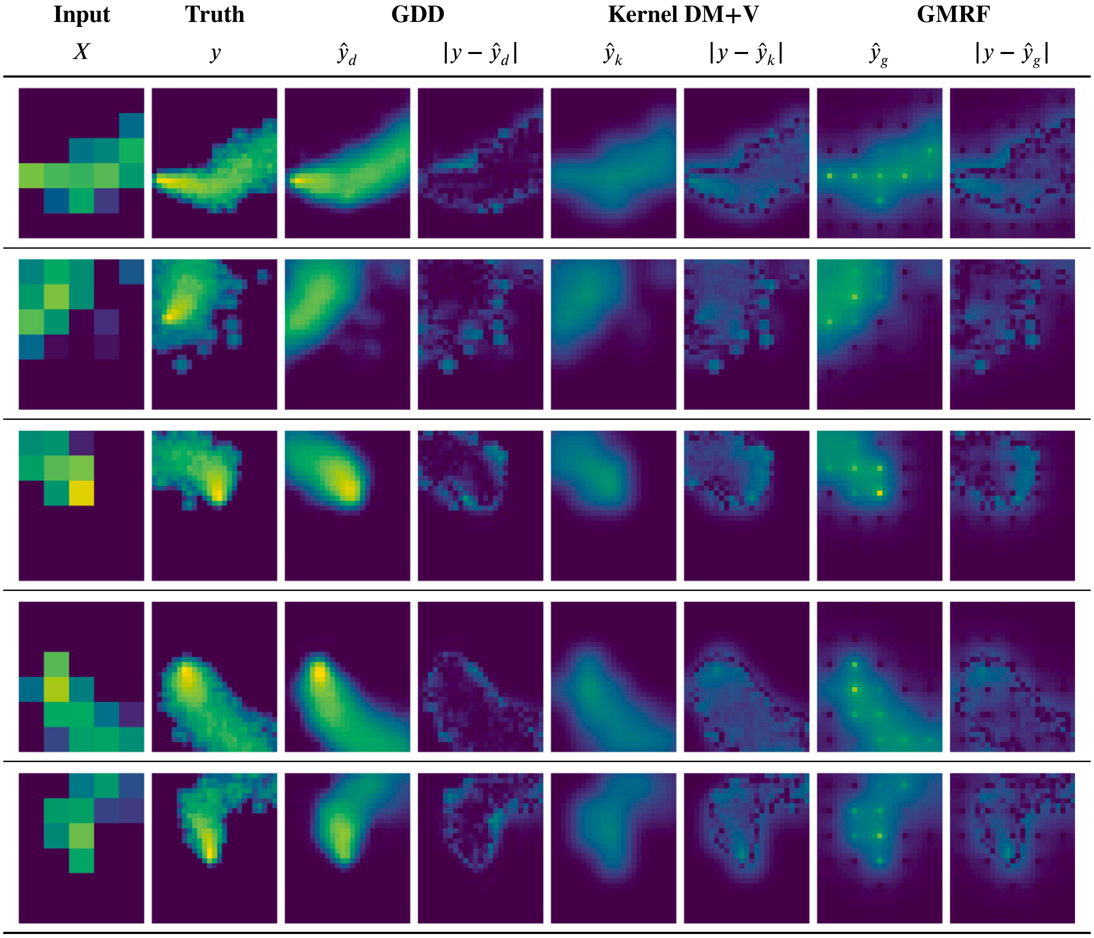


Fig. 3. Predictions $\hat{y}$ of the non-sequential models on input X. The absolute error $|y-\hat{y}|$ to the true distribution y is visualized next to the prediction. The plume edges, especially the upwind edge, are modeled best by GDD. Each row is color-coded based on the minimum and maximum values of the corresponding ground truth.

standard deviation (SD) of 1.23% of the mean MSE ($\text{MSE}_{\text{mean}} = 4.106 \times 10^{-3}$, SD $= 5.058 \times 10^{-5}$).

Secondly, we implemented a 12-fold cross-validation, each fold representing one unique wind set, to validate the model's performance on wind sets not included in its training data. Similar to the first phase, the model demonstrated low variability in performance, with the standard deviation constituting 1.28% of the mean MSE ($\text{MSE}_{\text{mean}} = 4.145\times10^{-3}$, SD $= 5.255 \times 10^{-5}$). These results suggest that the model can generalize well across different gas source positions and wind situations.

4.2. Synthetic test set

In the following, we quantify the prediction accuracy of the best-trained decoder model and compare it to corresponding state-of-the-art algorithms on the synthetic test data set. We compare the GDD against models that do not inherently profit from a sequential input. Although a time-dependent version of Kernel DM+V exists [16], it only models information decay over time and cannot gain new insights from patterns in sequential data. The same applies to GMRF.

Fig. 3 provides a visual comparison of the non-sequential models, and it is evident that GDD outperforms both traditional models for our specific environment. This observation is supported by the results for Root Mean Squared Error (RMSE) and Kullback–Leibler divergence (KLD, see Appendix B), as presented in Table 1a. The GDD accurately models the plume edges, especially the upwind edge, demonstrating its ability to learn the characteristics of gas distributions. Additionally, it can estimate the location of the highest gas concentration, which may not always match the sensor with the highest input concentration. The predictions of Kernel DM+V and GMRF do not match the high variations of gas concentrations, resulting in less pronounced predicted plumes. It is worth noting that GMRF overemphasizes sensor measurements in its predictions. To rule out incorrect execution of the algorithm, we corresponded with the original authors [12,13] and performed a hyperparameter study.

Table 1
RMSE and KLD on the synthetic test set.

(a) Non-sequential models with input sequence length $k = 1$.		
	RMSE	KLD
GDD	6.82×10^{-2}	3.00×10^{-3}
GDD (trained with noise)	7.05×10^{-2}	3.24×10^{-3}
GMRF	1.05×10^{-1}	7.38×10^{-3}
Kernel DM+V	1.12×10^{-1}	8.14×10^{-3}
(b) Sequential models with input sequence length $k = 20$.		
	RMSE	KLD
sGDD	6.48×10^{-2}	2.76×10^{-3}
ESM	1.35×10^{-1}	1.73×10^{-2}

A comparison between sGDD and ESM for sequential inputs is visualized in Fig. 4. Our results, presented in Table 1b, show that the sGDD model performs better regarding RMSE and KLD. The sequential variant also outperforms its non-sequential equivalent, most likely because it receives more input information. ESM, on the other hand, performs worse than Kernel DM+V and GRMF. It appears that ESM assigns excessive weight to the second-to-last input image, especially visible in the bottom row of Fig. 4.

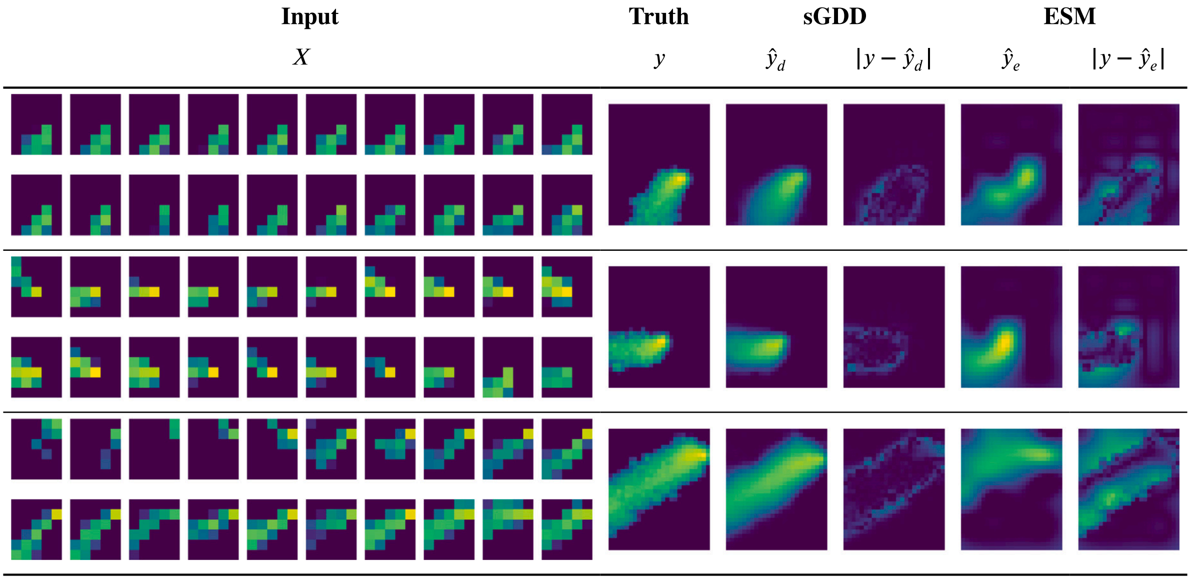


Fig. 4. Predictions of sGDD and ESM on an input sequence of length $k = 20$. The input sequence is displayed in two rows from the earliest sample (upper left) to the most recent sample (lower right). Each of the three samples is color-coded based on the minimum and maximum values of the corresponding ground truth.

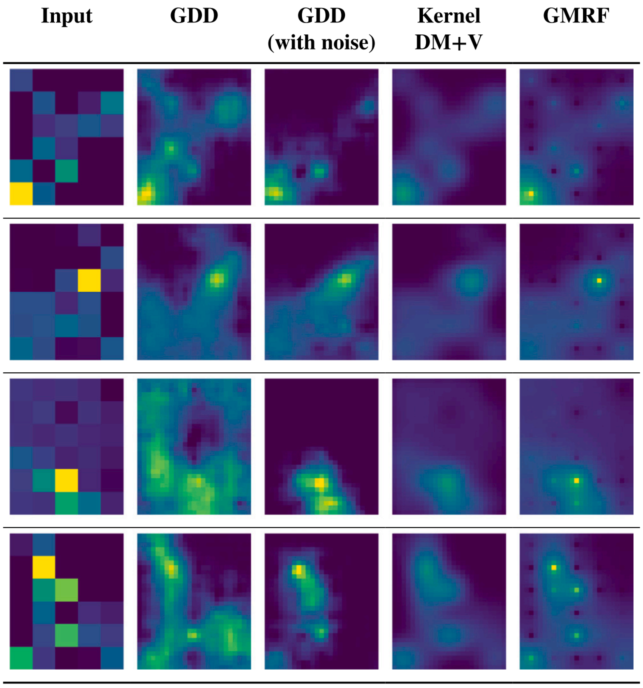


Fig. 5. We applied two different GDD models to the Tokyo data set for which no ground truth data is available. The first model was trained without added noise; the second model was trained with Gaussian noise added to the training data set. The results of this analysis indicate that, while traditional models tend to adhere more closely to the input data, the decoder models interpret the data more freely. It is worth noting that the lack of ground truth data makes it difficult to evaluate the accuracy of the predictions.

Table 2
Results for leave-one-out predictions on the real experimental data.

	RMSE
GDD	2.98×10^{-1}
GDD (trained with noise)	2.71×10^{-1}
GMRF	2.61×10^{-1}
Kernel DM+V	2.49×10^{-1}

4.3. Real-world “Tokyo data set”

For our final demonstration, we tested the decoder approach on the Tokyo data set (Section 3.2.3). Fig. 5 displays some example predictions. Our analysis indicates that the standard GDD struggles when dealing with noisy input data. However, the GDD trained with noise handles it differently. The top row example illustrates the trade-off between the denoising and non-denoising model. In this case, the denoising GDD variant arguably denoises too much and fails to capture the occurring gas dispersion.

We conducted a leave-one-out experiment on the input data to compare the performance of different models. Specifically, we randomly drop a sensor from the input and predict its value based on the other sensors. To adhere to the CNN’s fixed input structure, we interpolated bilinearly between the known positions in the input. Details of this procedure can be found in the Appendix in Algorithm 1. We evaluated the error of each model’s prediction for the dropped sensor’s location based on the remaining 29 sensors. The results of this experiment are presented in Table 2. Interestingly, the traditional models outperformed the GDD model in terms of RMSE, indicating that they can better estimate the dropped value. Among the traditional models, Kernel DM+V and GMRF adhered more strictly to the input data, while the GDD model interpreted the input more freely.

5. Discussion

5.1. Summary of main findings

Our study demonstrates that we were able to train a transposed CNN for a specific GDM scenario. We found that this approach, called GDD, surpasses traditional methods on a synthetic test set. We observed that GDD works well for both sequential and non-sequential input data, as seen in Table 1a and Fig. 3 for non-sequential, and Table 1b and Fig. 4 for sequential data.

We evaluated the reliability and adaptability of the GDD model through dual-phase k-fold cross-validation. In this process, we separately tested its performance on untrained gas source positions and wind conditions. The model demonstrated consistent generalization capabilities across all folds. Moreover, we conducted additional experiments to assess the model’s robustness to sensor outages, which we simulated by removing up to 50% of the input matrix pixels

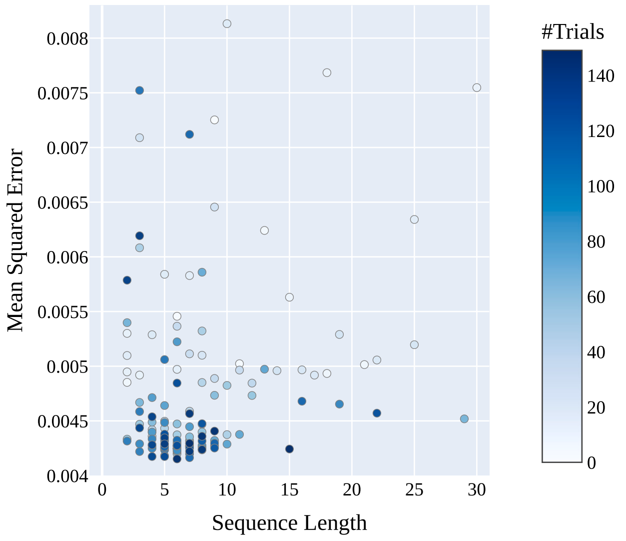


Fig. 6. We conducted a hyperparameter search for sGDD and evaluated the impact of sequence length k on the model's performance. The MSE was calculated on the validation set, and we found that the best results were obtained for sequence lengths ranging from 4 to 7. Apart from tuning the sequence length, we also optimized the dimensions of the inner layers to further improve the model's performance. The color bar represents the number of single trials, with each color shade corresponding to a specific trial number.

(Appendix D.1). We also examined its capacity to perform when exposed to noisy input data (Appendix D.2) and multiple gas sources (Appendix D.3)

5.2. Interpretation of results

While we quantified the differences between all models regarding RMSE and KLD, the exact numbers should be taken with caution, as the performances of all models crucially depend on the selected model parameters and their implementations. For example, ESM scores the worst RMSE, albeit having more input information than GMRF and Kernel DM+V. We attribute parts of the observed differences between the ESM and sGDD models to the compromise we had to make when selecting their respective parameters. In our analysis, we compared the performance of both models using a sequence length of $k = 20$. However, we found that the performance of ESM was highly sensitive to the values of kernel bandwidth, spectral radius, and regression delay. No universal set of parameters worked optimally across the entire test set. Increasing the ESM's dependence on previous inputs may result in it generating false plume shapes. Making it more reliant on recent inputs could lead to losing important information in some samples.

Since GDD's performance likewise depends on the chosen parameters, parameter tuning was implemented to find an optimal neural network configuration for our scenario. An earlier hyperparameter search, displayed in Fig. 6, revealed that the best results were obtained for models with $4 \leq k \leq 7$. For the comparison with ESM, however, we set $k = 20$ and kept the same hidden dimensions as found in Table 3.

The results on the real-world data (Section 4.3) probably best unveil the crucial differences between the GDD approach and the traditional models. While traditional approaches strictly extrapolate from available measurements, the predictive component of the neural network model interprets the input data more freely. This, however, is by design, as the GDD was trained to learn connections and identify patterns that may not be immediately visible in the input data. It is worth noting that the GDD that was trained with noisy data produces coherent maps, but their accuracy cannot be fully evaluated due to the lack of ground truth data. As the traditional models emphasize the given data, they can estimate the value of the dropped sensor more accurately than the denoising GDD. From another perspective, this highlights the advantages and potential of a trained model like the GDD, as enhancing and more flexible interpretation of the input can produce more sophisticated gas distribution maps than traditional models. However, developing a model that performs best regarding leave-one-out predictions on actual data was not the main objective of this paper. Additional regularization techniques or architectural changes may lead to improved models in this regard.

Table 3
GDD settings.

Parameter	Value
Layer sizes	[k,6,5] [62,7,6] [157,9,8] [255,12,11] [62,15,12] [37,15,12] [1,30,25]
Batch size	30
Training epochs	50
Optimizer	Adam
Learning rate	1.54×10^{-3}

5.3. Limitations and future research

A significant limitation of our study is its reliance on synthetic data for the training process. Since we did not have access to an extensive high-resolution real-world data set, we had to train and test GDD solely on synthetic data. Although we made efforts to generate realistic plume shapes, the results obtained on real-world data reveal that our synthetic data sets are still not realistic enough. Future work should be based on more sophisticated synthetic data (e.g., modeling the sensor response time) or build upon a more flexible model architecture that allows for training on sparse data. By adding depth and complexity to the simulation and model, it will be able to obtain three-dimensional and physically aware gas mapping models, as the works of [24–26] demonstrate on related fields. Further investigation should also explore alternative loss functions and evaluation metrics that minimize errors in highly dynamic areas like plume edges.

Our current model is tailored to a specific application with a specific scale and is not easily transferable to different scenarios, such as mapping gas distribution in environments with obstacles. A potential avenue for future research could be the development of a generic pre-trained GDM model that could be fine-tuned to new, real-world scenarios and environments. This approach of model transfer has been implemented successfully in various deep learning tasks, including computer vision and natural language processing. [45,46].

6. Conclusions

We have two main takeaways from our work. Firstly, we have shown how deep neural networks can be used for gas distribution mapping (GDM). By training our models on realistic synthetic gas distribution data and applying them to real-world data, we have demonstrated their effectiveness in this context. Our models outperformed state-of-the-art models by accurately modeling the edges of gas plumes, thanks to their ability to learn the intricate behavior and shape of gas plumes.

Secondly, we hope to encourage and expedite research on GDM through our work. We have made all the models and data sets we used in our study available online to inspire other researchers to explore deep learning and GDM. Our data set provides a rich source of ground truth distributions that may significantly reduce the need for time-consuming experiments and enable comparative evaluations of models.

CRediT authorship contribution statement

Nicolas P. Winkler: Conceptualization, Data curation, Methodology, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Oleksandr Kotlyar:** Conceptualization, Methodology, Supervision, Validation, Writing – original draft, Writing – review & editing. **Erik Schaffernicht:** Conceptualization, Methodology, Validation, Writing – review & editing, Supervision. **Haruka Matsukura:** Conceptualization, Data curation, Funding acquisition, Validation, Writing – review & editing. **Hiroshi Ishida:** Conceptualization, Funding acquisition, Validation, Writing – review & editing. **Patrick P. Neumann:** Conceptualization, Funding acquisition, Methodology, Supervision, Validation, Writing – review & editing, Project administration. **Achim J. Lilienthal:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Validation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data is publicly available at https://github.com/BAMresearch/SRGDM.

Appendix A. State-of-the-art models

The Kernel DM+V algorithm treats GDM as a density estimation problem and uses a non-parametric approach, making no assumptions about the functional form of the gas distribution. Sensor measurements are interpreted as noisy samples from the distribution to be estimated, and a Gaussian weighting function is used to assign importance to each measurement. The algorithm then uses this weighting function to compute the gas distribution map and a confidence map, which indicates the reliability of the estimates at each grid cell [44]. For further calculations, we relied on an existing Python implementation of Kernel DM+V[5] with settings stated in Table 4.

The GMRF algorithm considers measurements from sensors and prior knowledge about how gases typically distribute. It models the correlation between cells by penalizing the differences in gas concentrations in adjacent cells. Whenever an obstacle, such as a wall, is defined between two adjacent cells, the correlation between both cells is zero. However, as we did not add obstacles in our simulation of the synthetic data, GMRF does not take advantage of its capability to model obstacles here. The measurements are treated with a time-decreasing weighting factor, determining their relevance when observations at close locations are combined. We relied on a Python implementation,[6] which is provided by the original authors of [12,13]. Our settings can be found in Table 5.

In order to compare our algorithm to ESM, we had to adapt the original algorithm found in [17]. In the first part of ESM, a ESN is trained with inputs from a stationary sensor network and ground truth data from a mobile robot. We decided against simulating a robot, as it would have meant that the ESN sees more ground truth information, resulting in an unfair comparison with the sGDD. Instead, we decided that the ESN shall mimic the input sequence to potentially identify and amplify temporal patterns, similarly to an autoencoder. Apart from this, we kept the main principles of the original ESM algorithm. This means that for each prediction on the test set, we trained an ESN on the input and interpolated the ESN's weights with Gaussian Processes, enabling upscaling of the input to the target resolution. We decided to use a sequence of 20 images presented in their natural order defined by their original time sequence. All settings are stated in Table 6.

Table 4
Kernel DM+V settings.

Parameter	Value
Cell size	1
Kernel size	2.5
Evaluation radius	5

Table 5
GMRF settings.

Parameter	Value
Cell size	1
σ_{gz} (sensor noise for gas observations)	1
σ_{gr} (controls strength of gas regularization)	11.28
σ_{gb} (controls gas concentration's default value)	800

Table 6
ESM settings.

Parameter	Value
Sparsity (percentage of zero weights in the echo layer)	0.15
Echo layer size (number of neurons per ESN layer)	50
Spectral radius (controls temporal decay of ESN)	0.4
Win scale (scaling of ESN input weights)	1000
Regression delay (for weight computation)	19
Noise (applied to input data)	0
Bandwidth (of Gaussian Processes kernel)	3.5

Appendix B. Evaluation metrics

For evaluating our results, we used the Root Mean Squared Error (RMSE) and Kullback–Leibler divergence (KLD), as these are commonly used metrics for assessing the performance of regression models.

B.0.1. Mean Squared Error (MSE) and Root Mean Squared Error (RMSE)

The MSE measures the differences between the predicted distribution $\hat{y}$ and true gas distribution y:

$$\mathrm{MSE}(y, \hat{y}) = \frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i)^2 \tag{4}$$

where N is the number of data points of the sample. The RMSE is the square root of the MSE and penalizes large errors more heavily than small errors, through which it is sensitive to outliers:

$$\mathrm{RMSE}(y, \hat{y}) = \sqrt{\frac{1}{N}\sum_{i=0}^{N}(y_i - \hat{y}_i)^2} \tag{5}$$

B.0.2. Kullback–Leibler Divergence (KLD)

The KLD quantifies the similarity between two probability distributions. A lower KLD is considered better as it indicates a closer alignment between the model's predictions and the actual distribution. We define the pointwise KLD as:

$$\mathrm{KLD}(y_{dist}, \hat{y}_{dist}) = y \cdot \log\frac{y_{dist}}{\hat{y}_{dist}} \tag{6}$$

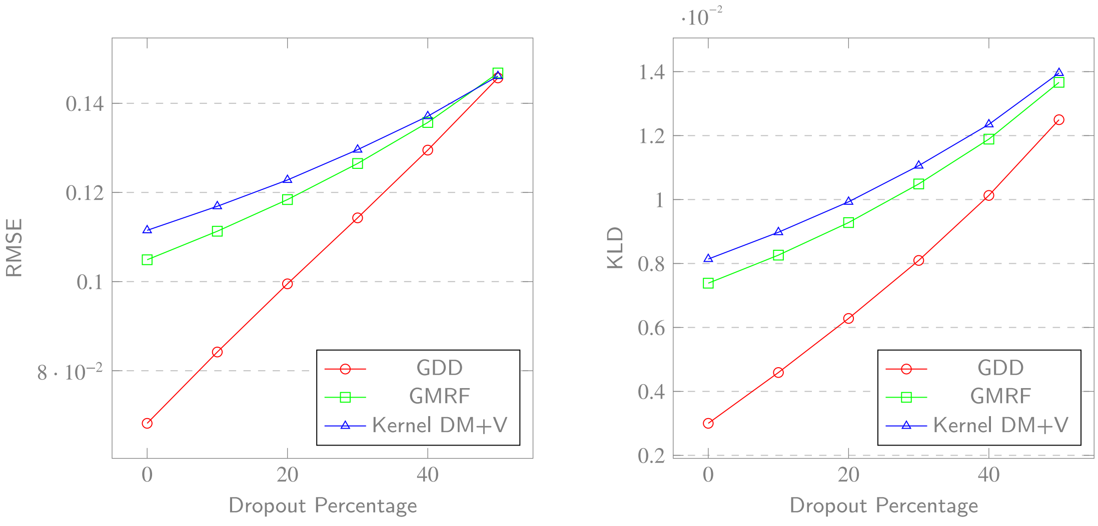
where y_{dist} and $\hat{y}_{dist}$ are the targeted and predicted distribution. To transform the matrices y and $\hat{y}$ to distributions suitable for Eq. (6), we reshape them into one-dimensional vectors and apply the Softmax function. Softmax rescales an input z so that all of its elements lie in the range [0,1] and sum to 1:

$$\mathrm{Softmax}(z_i) = \frac{\exp(z_i)}{\sum_j \exp(z_j)} \tag{7}$$

---

[5] https://gitlab.com/smueller18/TDKernelDMVW.

[6] https://github.com/MAPIRlab/gdm.



(a) RMSE (b) KLD

Fig. 7. Model results for reduced sensor inputs. Each model is tested on the test set, with random sensors of the 6 × 5 sensor network dropping out. Dropped-out sensors were interpolated linearly based on the remaining sensors.

Appendix C. Model parameters

Parameters of the GDD model are displayed in Table 3. Likewise, parameters of our implementations of Kernel DM+V, GMRF, and ESM are displayed in Table 4, Table 5, and Table 6.

Appendix D. Robustness analysis

D.1. Sensor dropout

To assess the performance of the GDD model with fewer sensors, we simulate sensor dropout by randomly removing pixels from a 6 × 5 grid matrix. However, due to the structure of our CNN, we must input a matrix of the same size into the model. Therefore, we cannot directly feed the CNN fewer pixels. To address this issue, we interpolate bilinearly, using the values of the four nearest known pixels to estimate the value of the missing pixel to generate the input data for the CNN. We evaluated dropout rates ranging from 0% to 50% in increments of 10%. The pseudocode is described in Algorithm 1. We also input the interpolated data into the Kernel DM+V and GMRF models to ensure a fair comparison between models.

The results, which are shown in Fig. 7, indicate that as the number of dropped sensors increases, the performance of the various models becomes more similar, as expected due to the bilinear interpolation of the missing values. In this case, the error is increasingly governed by the accuracy of the interpolation.

Algorithm 1 Dropping out and bilinear interpolation of input matrix

```
input: matrix $x$ of size 6x5, dropout rate r
output: matrix $x$ with dropped out values interpolated

for row in 1 to 6 do
    for column in 1 to 5 do
        $p \sim U(0,1)$        ▷ generate random number between 0 and 1
        if $p \leq r$ then
            $x$[row, column] = NaN
        end if
    end for
end for
$x$ = interpolate($x$)        ▷ interpolate with radial basis function[1]
```

Table 7
Results on the test set with Gaussian white noise applied to it. The GDD was trained once without and once with Gaussian white noise added to the training data set.

	RMSE	KLD
GDD (trained without noise)	1.22×10^{-1}	5.96×10^{-3}
GDD (trained with noise)	7.34×10^{-2}	3.49×10^{-3}
GMRF	1.10×10^{-1}	7.80×10^{-3}
Kernel DM+V	1.14×10^{-1}	8.61×10^{-3}

D.2. Noise robustness

We added Gaussian white noise on every test set sample to test the noise robustness and predicted the gas distribution with all models. This procedure was previously described in Section 3.4, where we also describe training a GDD variant with noisy data. As shown in Table 7 and Fig. 8, this GDD variant scores better regarding RMSE and KLD than Kernel DM+V and GMRF, whereas the original GDD performs slightly worse than the state-of-the-art models in terms of RMSE. Loosely speaking, the original GDD may be trying to find patterns in the data that are not there. We link this to the specific structure of our model (see Fig. 2).

Tested on the standard test set, the denoising variant achieves slightly worse results than the original model (Table 1a). As training a denoising model was not the main objective of this research work, we did not focus on implementing regularization techniques like dropout, max pooling, or batch normalization on the architectural level.

D.3. Multiple gas sources

All previous scenarios have assumed a single gas source, which was an assumption made before training the neural network model. However, we are interested in examining how the model performs when faced with multiple gas sources. To investigate this, we did not generate new synthetic data but rather combined the plumes from different gas source positions under the assumption of neutral buoyancy. Our original synthetic data contains only one source at a time, with

---

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.Rbf.html.

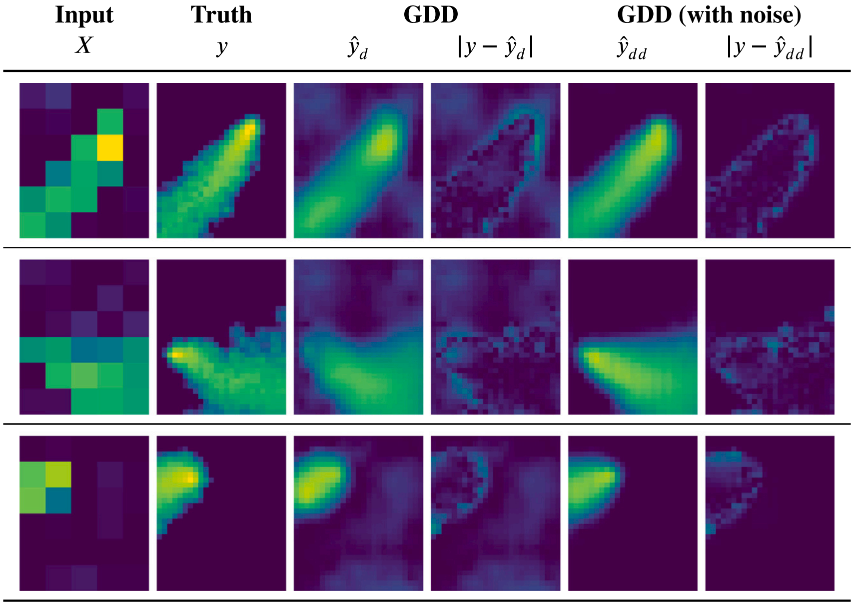


Fig. 8. Results of the GDD with Gaussian white noise applied to the input. The model in the center was trained without noise, while the model to the right columns was trained with random Gaussian white noise applied to each sample.

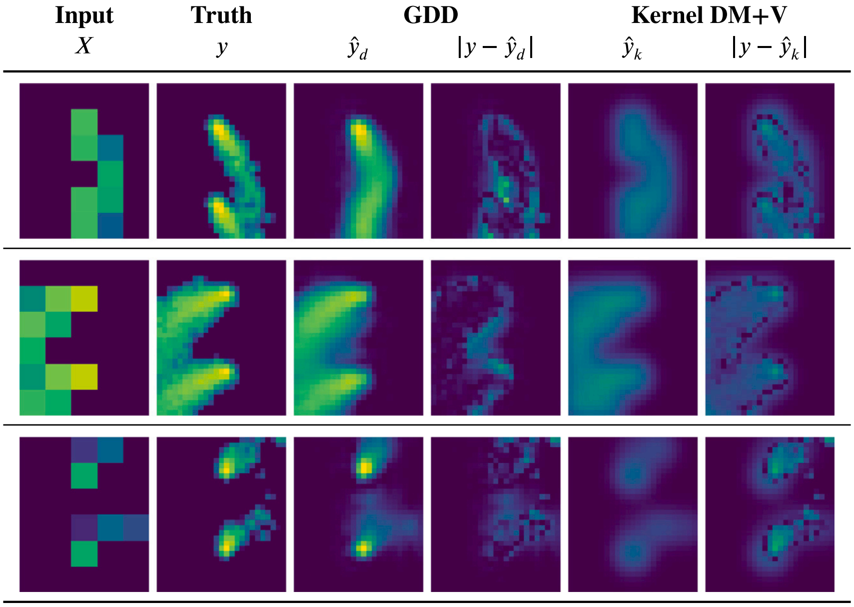


Fig. 9. Exemplary GDD and Kernel DM+V predictions on samples with two gas sources. Although GDD was only trained on data containing single gas sources, it can recognize and reconstruct two plumes in a single image.

one of three wind sets being used to simulate each source position (see Fig. 1(c)). We added two simulation results with the same wind data to obtain gas distributions with two sources. It is worth noting that this approach does not account for random components handled by the simulation. Accordingly, this experiment is just a preliminary demonstration of the model's capabilities to handle multiple sources after training with single source examples.

Results are displayed in Fig. 9. We see that the GDD can handle the input quite well, even though it was only trained with data from single gas sources.

References

[1] C. Zuidema, S. Sousan, L.V. Stebounova, A. Gray, X. Liu, M. Tatum, O. Stroh, G. Thomas, T. Peters, K. Koehler, Mapping occupational hazards with a multi-sensor network in a heavy-vehicle manufacturing facility, Ann. Work Expo. Health 63 (3) (2019) 280–293, http://dx.doi.org/10.1093/annweh/wxy111.

[2] Z. Zhu, B. Chen, Y. Zhao, Y. Ji, Multi-sensing paradigm based urban air quality monitoring and hazardous gas source analyzing: a review, J. Saf. Sci. Resil. (2021) http://dx.doi.org/10.1016/j.jnlssr.2021.08.004.

[3] H. Fan, V. Hernandez Bennetts, E. Schaffernicht, A.J. Lilienthal, Towards gas discrimination and mapping in emergency response scenarios using a mobile robot with an electronic nose, Sensors 19 (3) (2019) 685, http://dx.doi.org/10.3390/s19030685.

[4] M.A. Oliver, R. Webster, Kriging: a method of interpolation for geographical information systems, Int. J. Geogr. Inf. Syst. 4 (3) (1990) 313–332, http://dx.doi.org/10.1080/02693799008941549.

[5] C. Stachniss, C. Plagemann, A.J. Lilienthal, Learning gas distribution models using sparse Gaussian process mixtures, Auton. Robots 26 (2–3) (2009) 187–202, http://dx.doi.org/10.1007/s10514-009-9111-5.

[6] T. Wang, H. Zhang, Y. Wu, W. Jiang, X. Chen, M. Zeng, J. Yang, Y. Su, N. Hu, Z. Yang, Target discrimination, concentration prediction, and status judgment of electronic nose system based on large-scale measurement and multi-task deep

learning, Sensors Actuators B 351 (2022) 130915, http://dx.doi.org/10.1016/j.snb.2021.130915.
[7] C.J. Paciore, Nonstationary Gaussian Processes For Regression And Spatial Modelling (Ph.D. thesis), Carnegie Mellon University, 2003, URL: https://www.stat.berkeley.edu/~paciorek/diss/chapters/frontmatter.pdf.
[8] A.E. Gelfand, E.M. Schliep, Spatial statistics and Gaussian processes: A beautiful marriage, Spatial Statistics Avignon: Emerging Patterns, Spatial Stat. 18 (2016) 86–104, http://dx.doi.org/10.1016/j.spasta.2016.03.006.
[9] C.-Y. Hsu, C.-D. Wu, Y.-P. Hsiao, Y.-C. Chen, M.-J. Chen, S.-C.C. Lung, Developing land-use regression models to estimate PM2.5-bound compound concentrations, Remote Sens. 10 (12) (2018) 1971, http://dx.doi.org/10.3390/rs10121971.
[10] K.P. Messier, S.E. Chambliss, S. Gani, R. Alvarez, M. Brauer, J.J. Choi, S.P. Hamburg, J. Kerckhoffs, B. LaFranchi, M.M. Lunden, J.D. Marshall, C.J. Portier, A. Roy, A.A. Szpiro, R.C.H. Vermeulen, J.S. Apte, Mapping air pollution with google street view cars: Efficient approaches with mobile monitoring and land use regression, Environ. Sci. Technol. 52 (21) (2018) 12563–12572, http://dx.doi.org/10.1021/acs.est.8b03395.
[11] A. Montazeri, A.J. Lilienthal, J.D. Albertson, A spatial land use clustering framework for investigating the role of land use in mediating the effect of meteorology on urban air quality, Atmos. Environ.: X 12 (2021) 100126, http://dx.doi.org/10.1016/j.aeaoa.2021.100126.
[12] J. G. Monroy, J.-L. Blanco, J. Gonzalez-Jimenez, Time-variant gas distribution mapping with obstacle information, Auton. Robots 40 (1) (2016) 1–16, http://dx.doi.org/10.1007/s10514-015-9437-0.
[13] A. Gongora, J. Monroy, J. Gonzalez-Jimenez, Joint estimation of gas and wind maps for fast-response applications, Appl. Math. Model. 87 (2020) 655–674, http://dx.doi.org/10.1016/j.apm.2020.06.026.
[14] C. Rhodes, C. Liu, W.-H. Chen, Scalable probabilistic gas distribution mapping using Gaussian belief propagation, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 9459–9466, http://dx.doi.org/10.1109/IROS47612.2022.9981901.
[15] M. Reggente, A.J. Lilienthal, The 3D-kernel DM+v/w algorithm: Using wind information in three dimensional gas distribution modelling with a mobile robot, in: 2010 IEEE Sensors, IEEE, Kona, HI, 2010, pp. 999–1004, http://dx.doi.org/10.1109/ICSENS.2010.5690924.
[16] S. Asadi, H. Fan, V.H. Bennetts, A.J. Lilienthal, Time-dependent gas distribution modelling, Robot. Auton. Syst. 96 (2017) 157–170, http://dx.doi.org/10.1016/j.robot.2017.05.012.
[17] E. Schaffernicht, V.H. Bennetts, A.J. Lilienthal, Mobile robots for learning spatio-temporal interpolation models in sensor networks — The echo state map approach, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Singapore, Singapore, 2017, pp. 2659–2665, http://dx.doi.org/10.1109/ICRA.2017.7989310.
[18] F. Rahbar, Source Term Estimation Algorithms for Gas Sensing Mobile Robots (Ph.D. thesis), EPFL, Lausanne, 2021, http://dx.doi.org/10.5075/epfl-thesis-7606.
[19] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, W.-c. Woo, Convolutional LSTM network: A machine learning approach for precipitation nowcasting, in: Advances in Neural Information Processing Systems, Vol. 28, Curran Associates, Inc., 2015, URL: https://proceedings.neurips.cc/paper_files/paper/2015/hash/07563a3fe3bbe7e3ba84431ad9d055af-Abstract.html.
[20] V.-D. Le, T.-C. Bui, S.-K. Cha, Spatiotemporal deep learning model for citywide air pollution interpolation and prediction, in: 2020 IEEE International Conference on Big Data and Smart Computing (BigComp), 2020, pp. 55–62, http://dx.doi.org/10.1109/BigComp48618.2020.00-99.
[21] Q. Zhang, Y. Han, V.O.K. Li, J.C.K. Lam, Deep-AIR: A hybrid CNN-LSTM framework for fine-grained air pollution estimation and forecast in metropolitan cities, IEEE Access 10 (2022) 55818–55841, http://dx.doi.org/10.1109/ACCESS.2022.3174853.
[22] J. Ma, Y. Ding, J.C.P. Cheng, F. Jiang, Z. Wan, A temporal-spatial interpolation and extrapolation method based on geographic long short-term memory neural network for PM2.5, J. Clean. Prod. 237 (2019) 117729, http://dx.doi.org/10.1016/j.jclepro.2019.117729.
[23] R. Hashimoto, K. Suto, SICNN: Spatial interpolation with convolutional neural networks for radio environment mapping, in: 2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), 2020, pp. 167–170, http://dx.doi.org/10.1109/ICAIIC48513.2020.9065029.
[24] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, P. Battaglia, Learning to simulate complex physics with graph networks, in: Proceedings of the 37th International Conference on Machine Learning, PMLR, 2020, pp. 8459–8468, URL: https://proceedings.mlr.press/v119/sanchez-gonzalez20a.html.
[25] N. Thuerey, K. Weissenow, L. Prantl, X. Hu, Deep learning methods for Reynolds-averaged Navier-Stokes simulations of airfoil flows, AIAA J. 58 (1) (2020) 25–36, http://dx.doi.org/10.2514/1.j058291.
[26] X. Zhao, Z. Gong, Y. Zhang, W. Yao, X. Chen, Physics-informed convolutional neural networks for temperature field prediction of heat source layout without labeled data, Eng. Appl. Artif. Intell. 117 (2023) 105516, http://dx.doi.org/10.1016/j.engappai.2022.105516.
[27] R. Dahl, M. Norouzi, J. Shlens, Pixel recursive super resolution, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 5449–5458, http://dx.doi.org/10.1109/ICCV.2017.581.
[28] N.P. Winkler, H. Matsukura, P.P. Neumann, E. Schaffernicht, H. Ishida, A.J. Lilienthal, Super-resolution for gas distribution mapping: Convolutional encoder-decoder network, IEEE, 2022, pp. 1–3, http://dx.doi.org/10.1109/isoen54820.2022.9789555.
[29] N.P. Winkler, O. Kotlyar, E. Schaffernicht, H. Fan, H. Matsukura, H. Ishida, P.P. Neumann, A.J. Lilienthal, Learning from the past: Sequential deep learning for gas distribution mapping, in: D. Tardioli, V. Matellán, G. Heredia, M.F. Silva, L. Marques (Eds.), ROBOT2022: Fifth Iberian Robotics Conference, in: Lecture Notes in Networks and Systems, Springer International Publishing, Cham, 2023, pp. 178–188, http://dx.doi.org/10.1007/978-3-031-21062-4_15.
[30] O. Vinyals, C. Blundell, T. Lillicrap, k. kavukcuoglu, D. Wierstra, Matching networks for one shot learning, in: Advances in Neural Information Processing Systems, Vol. 29, Curran Associates, Inc., 2016, URL: https://proceedings.neurips.cc/paper_files/paper/2016/hash/90e1357833654983612fb05e3ec9148c-Abstract.html.
[31] J. Snell, K. Swersky, R.S. Zemel, Prototypical networks for few-shot learning, 2017, http://dx.doi.org/10.48550/arXiv.1703.05175.
[32] J. Monroy, V. Hernandez-Bennetts, H. Fan, A. Lilienthal, J. Gonzalez-Jimenez, GADEN: A 3D gas dispersion simulator for mobile robot olfaction in realistic environments, Sensors 17 (7) (2017) 1479, http://dx.doi.org/10.3390/s17071479.
[33] J.A. Farrell, J. Murlis, X. Long, W. Li, R. Carde, Filament-Based Atmospheric Dispersion Model to Achieve Short Time-Scale Structure of Odor Plumes, Technical Report, University of California, 2002, URL: https://apps.dtic.mil/docs/citations/ADA399832.
[34] C. Bilgera, A. Yamamoto, M. Sawano, H. Matsukura, H. Ishida, Application of convolutional long short-term memory neural networks to signals collected from a sensor network for autonomous gas source localization in outdoor environments, Sensors 18 (12) (2018) 4484, http://dx.doi.org/10.3390/s18124484.
[35] A. Yamamoto, C. Bilgera, M. Sawano, H. Matsukura, N. Sawada, C.-S. Leow, H. Nishizaki, H. Ishida, Application of sequence input and output long short-term memory neural networks for autonomous gas source localization in an outdoor environment, in: 2019 IEEE International Symposium on Olfaction and Electronic Nose (ISOEN), 2019, pp. 1–3, http://dx.doi.org/10.1109/ISOEN.2019.8823160.
[36] G.E. Hinton, R.R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507, http://dx.doi.org/10.1126/science.1127647.
[37] Z. Shao, L. Wang, Z. Wang, J. Deng, Remote sensing image super-resolution using sparse representation and coupled sparse autoencoder, IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. 12 (8) (2019) 2663–2674, http://dx.doi.org/10.1109/JSTARS.2019.2925456.
[38] L. Gondara, Medical image denoising using convolutional denoising autoencoders, in: 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), 2016, pp. 241–246, http://dx.doi.org/10.1109/ICDMW.2016.0041.
[39] V. Dumoulin, F. Visin, A guide to convolution arithmetic for deep learning, 2018, http://dx.doi.org/10.48550/arXiv.1603.07285.
[40] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyper-parameter optimization, Adv. Neural Inf. Process. Syst. 24 (2011).
[41] C. Olah, A. Mordvintsev, L. Schubert, Feature visualization, Distill 2 (11) (2017) e7, http://dx.doi.org/10.23915/distill.00007.
[42] K. Simonyan, A. Zisserman, Two-stream convolutional networks for action recognition in videos, in: Advances in Neural Information Processing Systems, Vol. 27, Curran Associates, Inc., 2014, URL: https://proceedings.neurips.cc/paper/2014/hash/00ec53c4682d36f5c4359f4ae7bd7ba1-Abstract.html.
[43] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, L. Fei-Fei, Large-scale video classification with convolutional neural networks, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1725–1732, http://dx.doi.org/10.1109/CVPR.2014.223.
[44] A.J. Lilienthal, M. Reggente, M. Trincavelli, J.L. Blanco, J. Gonzalez, A statistical approach to gas distribution modelling with mobile robots - the kernel DM+v algorithm, in: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, St. Louis, MO, USA, 2009, pp. 570–576, http://dx.doi.org/10.1109/IROS.2009.5354304.
[45] J. Yosinski, J. Clune, Y. Bengio, H. Lipson, How transferable are features in deep neural networks? 2014, arXiv:1411.1792 [cs].
[46] S.J. Pan, Q. Yang, A survey on transfer learning, IEEE Trans. Knowl. Data Eng. 22 (10) (2010) 1345–1359, http://dx.doi.org/10.1109/TKDE.2009.191.

Nicolas P. Winkler is a research assistant at BAM, Germany, and a Ph.D. student in computer science at Örebro University, Sweden. Before joining BAM, he received his master's degree in aerospace engineering (2018) from TU Berlin, Germany, and gained experience at Airbus and Rolls-Royce. His current research interests include environmental monitoring with heterogeneous sensor networks and deep neural networks.

Oleksandr Kotlyar obtained his Ph.D. degree in theoretical physics from Low Temperature Physics and Engineering Institute, Kharkiv, Ukraine. He was a Research Fellow at Institute for Low Temperature Physics and Engineering from 2010 to 2018. From 2018 to 2020 he was Marie S.-Curie Research Fellow at Aston Institute of Photonic Technologies, Birmingham, UK. In 2020 he joined Örebro University, Sweden. His current interests lie in the application of machine and deep learning to various scientific and engineering tasks.

Erik Schaffernicht is a senior lecturer in AI and robotics at Örebro University in Sweden. He received his Diploma degree in computer science (2006) and his Doctoral degree in neuroinformatics (2012) both from Ilmenau University of Technology, Germany. He is co-author of more than 70 peer-reviewed publications and a patent. His research interests range from mobile robot olfaction to feature selection methods and reinforcement learning.

Haruka Matsukura received the B.E., M.E., and D.E. degrees in mechanical engineering from the Tokyo University of Agriculture and Technology in 2008, 2009, and 2013, respectively. She was an Assistant Professor with Tokyo University of Agriculture and Technology from 2013 to 2017 and with Osaka University from 2017 to 2021, respectively. She is currently an Assistant Professor with the University of Electro-Communications. Her research interests include olfactory display systems and chemical sensing systems.

Hiroshi Ishida received the Ph.D. degree from Tokyo Institute of Technology in 1997. He was a Research Associate with the Department of Physical Electronics, Tokyo Institute of Technology, from 1997 to 2004. From 1998 to 2000, he was a Post-doctoral Fellow with the School of Chemistry and Biochemistry, Georgia Institute of Technology. In 2004, he joined Tokyo University of Agriculture and Technology, where he is currently a Professor with the Graduate School of Bio-Applications and Systems Engineering. His current research interests include mobile robot olfaction and olfactory displays.

Patrick P. Neumann is senior scientist at BAM in division 8.1 "Sensors, Measurement and Testing Methods" since 2008. He has been serving as the deputy head of division 8.1 since 2020. He holds a diploma degree (Dipl.-Inf.) in Computer Science from the Freie Universität Berlin and graduated in the field of mobile/aerial robot olfaction. His Ph.D. thesis addresses gas source localization and gas distribution mapping with an aerial robot. His responsibilities include the development and coordination of "autonomous and smart sensor systems". Currently, he coordinates BAM-wide all research activities related to UAS, including the supervision of (Ph.D.) students.

Achim J. Lilienthal is professor of Computer Science at TU Munich, Germany, where he leads the chair "Perception for Intelligent Systems". He is also manager of the "KI.Fabrik" at the German Museum in Munich and guest professor at the University of Örebro, Sweden, where he established the Mobile Robotics and Olfaction lab and two spinoff companies: Retenua and QTPIE.

His core research interest is in perception systems in unconstrained, dynamic environments. Typically based on approaches that leverage domain knowledge and AI, his research addresses mobile robot olfaction, rich 3D perception, navigation of autonomous transport robots, human–robot interaction, and mathematics education research. Achim J. Lilienthal obtained his Ph.D. in computer science from Tübingen University with a thesis about gas distribution mapping and gas source localization with mobile robots. He has published more than 300 refereed conference papers and journal articles, has co-edited several special issues, is senior member of IEEE and evaluator for several national funding agencies and the EU.