2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

GDM-Net: Gas Distribution Mapping with a Mobile Robot Using Deep Reinforcement Learning and Gaussian Process Regression

Iliya Kulbaka, Ayan Dutta, O. Patrick

***Abstract*— In a gas distribution mapping (GDM) task, th objective of a mobile robot is to map the gas concentrations c an airborne chemical over a region of interest using onboar sensing. Given the limited battery budget available to the robo covering the entire area to measure gas concentrations a every location might be infeasible. Assuming that the robc only has a budget for b meters of travel, in the rest of th locations, gas concentrations can be inferred using a supervise machine learning technique, namely the Gaussian Process (GP In this paper, we propose a novel technique that combines dee reinforcement learning and GP regression to find an effectiv policy for GDM. We have implemented the proposed techniqu in Python within a 16×16 4-connected plane. We have used si types of Gaussian plumes to validate our presented approacl Compared to two popular baselines, our approach outperforms greedy and random exploration by 62% and 151% in terms of earned rewards, while outperforming them by 47% and 345%, respectively, in terms of the precision of gas distribution modeling in all test cases without obstacles. Our approach also improves the coverage of the exploration while consequently reducing the uncertainty in the prediction.**

I. Introduction

Gas distribution mapping (GDM) is the process of building a spatial map of airborne chemical concentrations. We can use a mobile robot equipped with a gas sensor, e.g., an E-nose to measure gas concentrations at various locations in the environment [1]. Given that mobile robots have limited energy, they can only sample a limited number of points in the environment. In the rest of the environment, a predictive model can be used to create a global gas distribution map. GDM has practical relevance in real-world applications such as environmental monitoring and search-and-rescue [2].

GDM with a mobile robot can become challenging for two main reasons. The gas distribution may be initially unknown, in which case a quality adaptive exploration strategy is likely outperform any pre-computed optimal trajectory. Secondly, although the Gaussian Process Regressor (GPR) is often the predictive model of choice within adaptive exploration, its computational overhead (i.e., time complexity being cubic in the dimension of the multivariate Gaussian [3]) may present a bottleneck for small-scale robots.

To this end, we propose a deep reinforcement learning (DRL) approach toward solving the GDM problem that incorporates GPR only during offline training. Our proposed neural network architecture, named `GDM-Net`, utilizes multiple convolutional and linear layers to map the spatial gas distribution efficiently. During deployment, the robot only follows (i.e., infers from) the learned policy–it does *not* use GPR online. The state of the environment as observed by the robot is provided as the input. The next action (e.g., moving to a neighboring cell) following the policy is received as an output, which is then executed by the robot. The overall framework is illustrated in Fig. 1. Our framework can successfully perform without any prior knowledge of the parameters of the gas distribution instance.

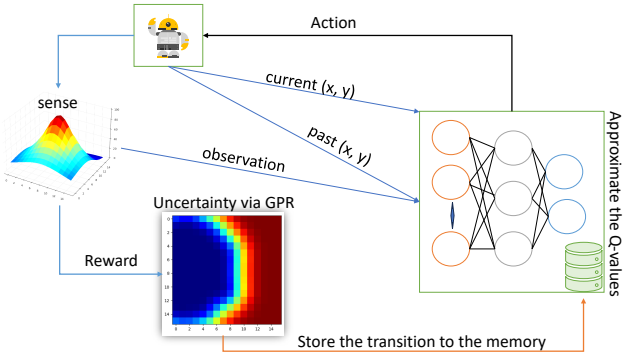


Fig. 1. The proposed GDM-Net framework is illustrated. The variances in the prediction of the gas concentration measurements in the unvisited locations are computed using GP regression, which in turn, are used as rewards. On the other hand, the robot's sensed gas concentrations along with the corresponding past and current visited locations are used as the state information for the neural network.

We have validated the presented technique on six types of Gaussian gas plumes, which vary in the spread of the plume. We observe that following our combined DRL and GPR training technique when used online, the policy generates trajectories for the robot that lead to spatially distributed exploration. This, in turn, leads to better information gathering about the gas concentrations at various parts of the environment. Numerical results show that the policies produced by GDM-Net consistently outperform a random walk and also perform favorably compared to a popular greedy GPR-based strategy [4], the latter requiring the robot to support GPR calculations during deployment.

The main contributions of this paper are as follows.

1) To the best of our knowledge, this is the first work that combines deep reinforcement learning with Gaussian Process regression for gas distribution mapping.
2) Our proposed DRL approach is model-free and does not require the robot to train the GPR model with new

This work is supported by NSF CPS Grants #1932300 and #1931767.

I. Kulbaka, A. Dutta, O.P. Kreidl, and S. Roy are with the University of North Florida, USA. Emails: {n01427009, a.dutta, patrick.kreidl, s.roy}@unf.edu
L. Bölöni is with the University of Central Florida, USA. Email: ladislau.boloni@ucf.edu

gas concentration measurements at deployment.

3) Across all six types of tested gas distribution types, our proposed approach consistently outperforms GPR-based greedy and random walk baselines.

II. Related Work

A comprehensive survey of robotic exploration techniques for gas distribution mapping is presented in [2]. One of the earliest studies on model-free gas distribution mapping is due to [5]. Here, in a 2D environment, the robot has to cover most of the environment, which is unlike our approach. We assume the budget available to the robot is only 30% of the entire environment. In many GDM solutions, the robot follows a pre-determined trajectory while sensing gas concentrations along the way [4]. Authors in [4] also present a comparative study between various predictors for the GDM task, namely GPR, neural networks, and interpolation. They report that GPR performed the best in terms of predicted model accuracy among the tested techniques. Most importantly the tests were conducted in an indoor environment with controlled, replicable scenarios. In [6], the authors have utilized smelling nano aerial vehicles for 3D GDM. However, the robot paths were predefined by the authors.

Neumann et al. [7] have solved the GDM problem while utilizing the wind estimation sensor available to the robot. They have used a Gaussian kernel extended from [8]. The authors in [9] proposed a method for methane gas mapping in both outdoor and indoor scenarios – their proposed mapping algorithm was analogous to computer-assisted tomography. Gas distribution modeling including gas release rate and spread generation has been extensively studied. In [10], the authors develop mathematical models for gas generation and simulation for both instantaneous and continuous release. In our formulation, the release is instantaneous. Similarly, Liu et al. [11] have proposed a quantification method for modeling methane gas dispersion behaviors in sub-sea pipelines. Detection of gas and measuring its concentrations at various locations is studied in [12]. The authors have used an ensemble learning-based approach for gas measurement collection and online classification. Although we study a model-free gas distribution mapping problem that does not make any prior assumption about the distribution, there has been a considerable amount of research done on model-based approaches [13], [14]. For example, recently, He et al. [15] proposed a distribution mapping algorithm specifically for GP plume models. On the other hand, Fukazawa and Ishida [13] assume a turbulent diffusion model of the gas distribution.

One of the related problems to this is gas source localization (GSL). In this, the robot follows a trajectory until it reaches the source of the gas and then reports back the source location to the control station [16]. Although not for GDM, DRL has recently been used to solve the GSL problem [17], [18]. Note that for GSL, no GPR is needed as the robot is only concerned about the sensed gas measurements and not the concentrations at the unvisited cells, which makes GSL a significantly different problem from GDM.

III. Problem Setup

Our problem setup starts by jointly discretizing space and time. More specifically, we assume the planar region is uniformly divided into n discrete square cells, each ith cell centered at a 2D position $\mathbf{p_i}$ and associated with an (initially unknown) aggregate concentration value c_i. We further assume the robot navigates the discretized region over 4-connected paths (e.g., each move is either north, south, east, or west), accordingly discretizing time such that each step permits movement to a 4-adjacent cell and observation of the associated concentration.[1] Following the state-of-the-art in gas modeling [19], we consider gas distribution mapping (GDM) tasks assuming that the gas concentrations vary (1) significantly over locations within a planar spatial region yet (2) negligibly over time and due to the robot's movements within the budgeted duration of exploration.

The overarching model of our GDM problem is an instance of a partially observable Markov Decision Process (POMDP). The partial observability stems from the exploration objective given the robot's budget renders it infeasible to access the global state of the environment. A POMDP is formally defined as a tuple $\langle S, A, T, O, R, \Omega \rangle$ with S and A denoting (finite) state and action spaces, respectively, $T : S \times A \times S \to [0, 1]$ and $R : S \times A \to (-\infty, \infty)$ denoting the single-stage transition and reward functions, respectively, Ω denoting the observation space and $O : S \times A \times \Omega \to [0, \infty)$ denoting the probabilistic representation of the single-stage observation process. A reinforcement learning (RL) method typically assumes that functions T and/or O cannot be modeled explicitly (e.g., because the cardinality of S and Ω is prohibitively large), yet it is possible to realize (via simulation or experiential) traces of sequential observation-action pairs and the associated sequence of rewards. The remainder of this section develops the reward function that befits our GDM objective, while Section IV discusses the specific simulation-based RL technique to yield effective exploration policies.

The GDM-Net reward function is based upon first modeling environmental uncertainties using Gaussian Processes (GPs) [20], [21], [22], [23]. Such GPs assume that all collection locations generate information according to the Gaussian random vector $\mathbf{C}$ with known mean vector μ and covariance matrix Σ. At each time step of the exploration, the set of all collection locations can be divided into two disjoint subsets, U and V, corresponding to the robot's unvisited and visited locations, respectively. The subvector $\mathbf{C}_U$, characterizing the unobserved information conditioned on the values $\mathbf{C}_V = \mathbf{c}_V$ observed in the visited locations, has posterior statistics

$$\begin{aligned} \mu_{U|\mathbf{C}_V} &= \mu_U + \Sigma_{UV}\Sigma_{VV}^{-1}(\mathbf{c}_V - \mu_V) \\ \Sigma_{UU|\mathbf{C}_V} &= \Sigma_{UU} - \Sigma_{UV}\Sigma_{VV}^{-1}\Sigma_{VU} \end{aligned} \tag{1}$$

[1] Our problem setup readily generalizes to less restrictive spatiotemporal discretization, but this generality is avoided here in the interest of brevity—our specific assumptions befit the simulation experiments in Section V-B.

with the prior statistics organized into the corresponding block forms concerning U and V i.e.,

$$\mu = \begin{bmatrix} \mu_U \\ \mu_V \end{bmatrix} \quad \text{and} \quad \Sigma = \begin{bmatrix} \Sigma_{UU} & \Sigma_{UV} \\ \Sigma_{VU} & \Sigma_{VV} \end{bmatrix}.$$

It is well known in (1) that the mean vector represents the minimum-mean-square-error prediction of uncollected information $\mathbf{C}_U$, while the posterior covariance matrix characterizes the prediction's uncertainty.

In practice, the principal challenge of Gaussian prediction is to obtain accurate prior statistics for the environmental information. Such priors are typically derived from training data via statistical learning methods (e.g., maximum-likelihood, Bayesian regression [21]) and, for spatially distributed GPs, usually also leverage domain-specific environmental considerations. A length-n GP has $d = 2n + n(n-1)/2$ degrees-of-freedom, in general, where requirements that the number of training samples exceed d are often formidable in robotics applications. It is thus common to assume a reduced-order structure for the GP. For example, the so-called "homogeneous and isotropic Gaussian Markov random field using a -squared-exponential kernel" defines the covariance matrix using just two hyper-parameters: given a pair of locations i and j at spatial positions $\mathbf{p}_i$ and $\mathbf{p}_j$, respectively, the kernel function defines the associated pairwise covariance by

$$\mathbf{Cov}(C_i, C_j) = \beta^2 \exp\left(-0.5||\mathbf{p}_i - \mathbf{p}_j||^2/\ell^2\right) \qquad (2)$$

where $||\cdot||$ denotes Euclidean distance, $\beta > 0$ is the local standard deviation and $\ell > 0$ is the length scale of diminishing correlation between information at increasingly-distant locations. Such spatial structure often implies computational advantages within standard GP calculations. For example, squared-exponential kernels are known to induce diagonally-dominant covariance matrices and, in turn, motivates information maximization objectives involving the determinant of $\Sigma_{UU|\mathbf{C}_v}$ in (1) to be approximated by minimizing the product of the component variances $\left\{\sigma^2_{u|\mathbf{C}_V}; u \in U\right\}$ along its diagonal. A greedy approach to minimizing this product is for the robot to next observe the unvisited cell with highest posterior variance, subject to respecting path adjacency constraints. That cell, say $u \in U$, produces observation c_u and, after evolving the disjoint sets $V := V \cup \{u\}$ and $U := U - \{u\}$, the GP can be updated via a kernel-based regression against the set of position-concentration pairs $\{(\mathbf{p}_v, c_v); v \in V\}$.

IV. GDM-Net for Gas Distribution Mapping

Neural Network Architecture. We present the proposed neural network architecture in Fig. 2. The input to the network is a 3-channel state information. This is partial information about the environment that is available to the robot. This input tensor consists of three two-dimensional layers of size $l \times l$. One layer holds the binary data representing the robot's current location, the second one represents its past visited locations in a binary format, and finally, the last layer holds the sensed concentration values at the visited locations. The unvisited locations contain zeros in this layer. Note that in Fig. 2, the first layer is the first hidden layer, which is convolutional as well as the second hidden layer. For these, the kernel size is 5 and the stride is set to 1. The rest of the layers are fully connected linear layers with the sizes indicated in Fig. 2. We have used ReLU activation function. The output layer has four neurons estimating the Q-values for the four available actions to the robot. As our test environment is of size 16×16, $l = 16$.

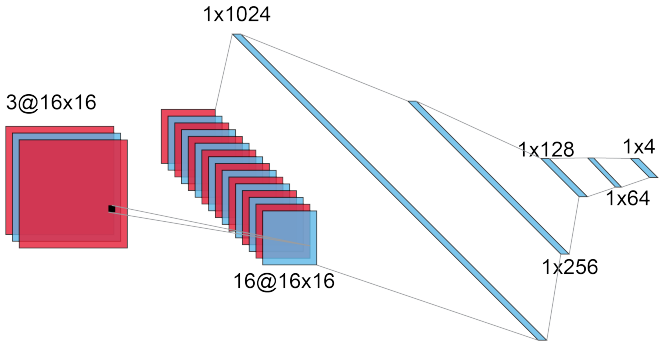


Fig. 2. The proposed neural network architecture.

Reward Function. Our proposed reward function is tied to the GPR technique mentioned in Section III. More specifically, if V denotes all cells already visited and u denotes an as-yet unvisited cell adjacent to the current cell location, the robot receives a positive reward equal to the posterior variance $\sigma^2_{u|\mathbf{C}_V}$ associated with cell u. This encourages the robot to go to cells that have high uncertainties, but we also explicitly penalize the robot for repeat visits to any cell. Formally, employing the notation around (1) and (2), the reward function R is taken as

$$R = \begin{cases} \sigma^2_{u|\mathbf{C}_V}, & u \in U \\ -4\beta^2, & \text{otherwise} \end{cases}, \qquad (3)$$

which we note will vary with time step in correspondence with the evolving sets U and V as exploration progresses.

Training and Testing. Our training technique is adapted from double DQN (DDQN) [24]. Given the fact that $S \times A$ can be astronomically large, it is infeasible for the robot to find stable Q-values for all possible state-action pairs. Therefore, in a deep RL approach such as that used in this paper, neural networks are used to approximate those values. More specifically, we maintain two neural networks denoted by Q_ϕ and Q_{ϕ^-} representing the policy and the target network. Both of these have the same architecture as shown in Fig. 2. However, the target network's parameters are updated less frequently than the policy network for better stabilization.

At the beginning of every training episode, the robot's position is reset and a new gas plume is generated. More details about the gas generation function can be found in Section V. The robot has b maximum actions/steps available to it. Therefore, the goal of the robot would be to visit the cells that maximize the uncertainty reduction following Eq. 3. In each step, the current state is passed as an input to

Q_ϕ, and the Q-values of the available actions are received as outputs. An action a is chosen following the ϵ-greedy strategy. The robot executes this action, moves to a new cell, and a new observation (i.e., gas concentration in the new cell) is made. The robot also receives a reward for a. The transition from s to s' after executing a, i.e., $\langle s, a, R, s', c\rangle$, is stored in a memory buffer $\mathcal{D}$. Next, we optimize the policy network parameters at the end of every step. We randomly select k transitions from $\mathcal{D}$ for training. The target Q-values, Y, are calculated as

$$Y = R + \gamma \cdot Q(s', \arg\max Q(s', a; Q_\phi), Q_{\phi^-}) \quad (4)$$

Following DDQN, we calculate the expected target values from the target network to minimize overestimation [24]. Note that this is unlike DQN [25]. Next, we compute the error between the Q-values from Q_ϕ and Q_{ϕ^-} as follows.

$$\mathcal{L} = \mathbf{E}[(Y - Q_\phi(s)] \quad (5)$$

More specifically, we use the Adam optimizer to regress the policy network towards the target Q-values. This process continues until the maximum number of training episodes are complete. The pseudocode is presented in Algorithm 1.

Algorithm 1: Gas distribution mapping using deep reinforcement learning and GP regression

```
for each episode do
    Place the robot at the starting cell and generate a new gas plume;
    Initialize a new GP model;
    for b steps do
        s ← current state;
        a ← selected action based on the ε-greedy strategy;
        Execute a, transition to state s', and measure the gas concentration c_i;
        Add the observation to the GP model and update it;
        Receive reward R following Eq. 3;
        Store the transition information in D;
        Sample random k experiences from D;
        Minimize the loss L (Eq. 5) between the Q-values from Q_φ and Q_φ⁻;
    Update the Q_φ⁻ parameters every τ episode.
```

During the testing, the robot only has access to the learned policy and no learned GP model. The current state is passed and the policy outputs an action that the robot executes to transition to a new state. Similar to training episodes, the robot has b steps available to it. We record numerous performance metrics in each such test episode for validation.

TABLE I

AVERAGE VALUES OF GAS GENERATION PARAMETERS.

Gas type	σ_y	σ_z
Very Unstable	2.92206	1.38403
Moderately Unstable	2.03447	1.08212
Slightly Unstable	1.32307	0.81559
Neutral	0.93636	0.60371
Moderately Stable	0.96506	0.65967
Very Stable	0.83042	0.56893

TABLE II

LIST OF PARAMETERS USED IN OUR EXPERIMENTS.

Parameters	Values
State	$3 \times 16 \times 16$ tensor
Action	Up, Down, Left, Right
Number of training episodes	5000
Episode length (b)	77 (30% of 16×16)
Priority replay memory size	$20,000$
Mini-batch size	32
Discount factor	0.90
Learning rate	0.00025
Target network update frequency (τ)	100
Epsilon decay type	Exponential
Epsilon decay rate	800
Epsilon start value	0.9
Epsilon end value	0.01
Loss function	Mean Square Error
Optimizer	Adam
Number of testing episodes	1000
Local std. dev. (β)	1

V. EXPERIMENTS AND RESULTS

A. Setup

The gas concentration c_i at $\mathbf{p_i} = (x, y, z)$ can be measured by the following Gaussian plume dispersion formulation.

$$c_i = \frac{G}{2\pi\sigma_y\sigma_z W} e^{\left(-\frac{y^2}{2\sigma_y^2}\right)} \left[e^{\left(-\frac{(z-h)^2}{2\sigma_z^2}\right)} + e^{\left(-\frac{(z+h)^2}{2\sigma_z^2}\right)} \right], \quad (6)$$

where G is the gas release rate (grams/s.), h is the height of the plume center-line, and W is the wind speed. Finally, σ_y and σ_z are the standard deviations representing the spread of the plume in the y and z directions. In stable gas generation models, the gas will travel further before dispersing whereas in unstable models, the gas will disperse quickly. The average parameter values to control the stability are presented in Table I. Given that our proposed approach is limited to 2D exploration, the gas concentrations across the z-axis are summed up. We have tested our proposed technique on six variations of the Gaussian plume. G and W values are fixed to 40 and 5 respectively for all gas types.

We have quantified our proposed GDM-Net framework's performance within a 16×16 grid environment based on four main metrics: 1) episodic reward, 2) number of unique cells visited, 3) mean square error (MSE) between the ground truth and final predicted gas distribution map, and 4) end of episode uncertainty. To benchmark our proposed approach, we have compared it against two popular baselines: 1) Random: the robot chooses the next cell randomly from the four available neighbors, and 2) Adaptive greedy: The robot moves to its neighbor cell u with the highest reward

TABLE III

AVERAGE REWARD GAINED IN 1000 TEST CASES (NO OBSTACLE).

Train \ Test	Very unstable	Moderately unstable	Slightly unstable	Neutral	Moderately stable	Very stable
Very unstable	**7.41 ± 0.396**	6.04 ± 0.407	0.21 ± 2.447	-1.71 ± 0.905	-1.49 ± 0.990	-1.34 ± 1.142
Moderately unstable	1.97 ± 2.288	**6.27 ± 0.202**	-2.27 ± 0.288	-3.09 ± 0.114	-3.09 ± 0.208	-3.13 ± 0.238
Slightly unstable	-2.12 ± 0.056	-2.39 ± 0.100	**3.12 ± 0.277**	-2.58 ± 0.133	-2.47 ± 0.258	-2.21 ± 0.496
Neutral	0.99 ± 0.491	0.89 ± 0.520	0.79 ± 0.382	**1.09 ± 0.415**	1.40 ± 0.440	1.46 ± 0.509
Moderately stable	2.18 ± 0.260	1.56 ± 0.269	0.51 ± 0.570	0.68 ± 0.430	**0.94 ± 0.467**	1.02 ± 0.637
Very stable	-0.92 ± 1.752	1.02 ± 1.291	0.89 ± 0.646	1.03 ± 0.664	1.04 ± 0.823	**1.20 ± 0.714**

TABLE IV

AVERAGE FINAL UNCERTAINTY IN 1000 TEST CASES (NO OBSTACLE).

Train \ Test	Very unstable	Moderately unstable	Slightly unstable	Neutral	Moderately stable	Very stable
Very unstable	**0.51 ± 0.023**	0.39 ± 0.032	0.52 ± 0.305	0.60 ± 0.124	0.62 ± 0.127	0.61 ± 0.142
Moderately unstable	0.58 ± 0.186	**0.38 ± 0.021**	0.65 ± 0.070	0.65 ± 0.045	0.66 ± 0.064	0.69 ± 0.109
Slightly unstable	0.88 ± 0.004	0.82 ± 0.013	**0.08 ± 0.015**	0.66 ± 0.034	0.69 ± 0.045	0.69 ± 0.068
Neutral	0.13 ± 0.098	0.16 ± 0.090	0.13 ± 0.037	**0.12 ± 0.058**	0.16 ± 0.059	0.19 ± 0.068
Moderately stable	0.29 ± 0.033	0.19 ± 0.028	0.19 ± 0.058	0.15 ± 0.043	**0.18 ± 0.047**	0.21 ± 0.082
Very stable	0.70 ± 0.204	0.29 ± 0.212	0.12 ± 0.094	0.14 ± 0.083	0.18 ± 0.094	**0.20 ± 0.066**

TABLE V

AVERAGE NUMBER OF UNIQUE CELLS VISITED AFTER 77 STEPS IN 1000 TEST CASES (NO OBSTACLE).

Train \ Test	Very unstable	Moderately unstable	Slightly unstable	Neutral	Moderately stable	Very stable
Very unstable	**75.01 ± 0.089**	74.74 ± 2.466	49.04 ± 27.618	34.37 ± 13.275	35.62 ± 13.543	36.37 ± 14.920
Moderately unstable	51.00 ± 18.841	**76.00 ± 0.000**	24.24 ± 3.862	15.94 ± 1.853	15.62 ± 3.329	14.86 ± 3.488
Slightly unstable	18.00 ± 0.000	18.00 ± 0.000	**76.00 ± 0.000**	23.10 ± 1.907	23.60 ± 3.539	26.39 ± 6.590
Neutral	63.54 ± 7.288	65.36 ± 8.537	70.11 ± 5.381	**75.24 ± 3.699**	74.73 ± 3.687	73.90 ± 4.847
Moderately stable	73.69 ± 3.000	73.69 ± 3.450	67.89 ± 7.552	72.67 ± 5.182	**72.62 ± 5.191**	71.12 ± 7.736
Very stable	32.63 ± 19.964	66.87 ± 18.575	70.54 ± 9.109	73.37 ± 7.469	71.44 ± 9.377	**72.19 ± 6.698**

TABLE VI

AVERAGE FINAL MSE IN 1000 TEST CASES (NO OBSTACLE).

Train \ Test	Very unstable	Moderately unstable	Slightly unstable	Neutral	Moderately stable	Very stable
Very unstable	**3.65 ± 0.190**	3.38 ± 0.426	6.34 ± 4.078	9.09 ± 2.756	10.80 ± 3.300	12.36 ± 4.322
Moderately unstable	3.29 ± 1.484	**2.74 ± 0.400**	6.80 ± 1.135	10.37 ± 1.499	12.21 ± 2.171	14.84 ± 4.090
Slightly unstable	6.28 ± 0.228	8.00 ± 0.409	**0.14 ± 0.080**	10.10 ± 0.960	13.10 ± 1.496	15.65 ± 2.359
Neutral	0.25 ± 0.468	0.47 ± 0.370	0.55 ± 0.365	**0.95 ± 1.129**	1.49 ± 1.208	2.18 ± 1.520
Moderately stable	0.51 ± 0.108	0.41 ± 0.108	0.59 ± 0.350	0.79 ± 0.388	**1.39 ± 0.665**	2.49 ± 2.137
Very stable	4.44 ± 2.316	1.30 ± 2.509	0.45 ± 1.210	0.88 ± 1.381	1.49 ± 1.966	**2.02 ± 1.357**

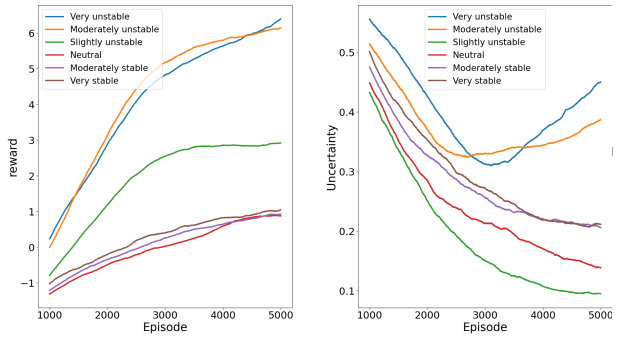
according to (4). The parameters related to the DRL and GPR implementations are listed in Table II.

B. Results and Discussion

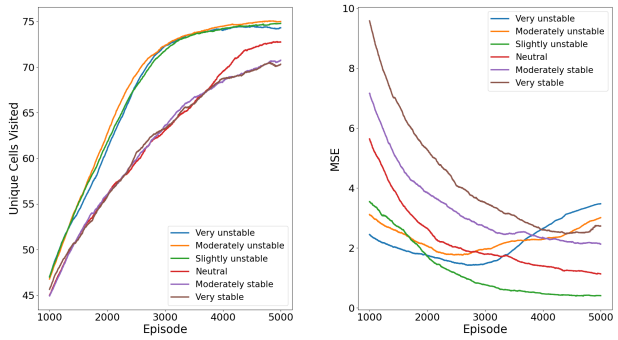
The training follows the logic of the Algorithm. 1 with the robot starting location being the same throughout. Fig. 3 presents the statistics of the key metrics during training. The unique number of cells visited is an important factor. Due to the limited budget, more unique cells visited by the robot lead to higher uncertainty reduction, higher reward, and more importantly lower MSEs at the end. We see in Fig. 3 that the robot learns to visit unique cells with more training episodes, which in turn helps the other metrics to be optimized[2].

We also employed our proposed GDM-Net technique in an environment with a 3×2 block obstacle (shown in cyan in Fig. 5) in a set position. The obstacle impedes the gas dispersion. When wind is applied from top to bottom, the cells behind the obstacle in the direction of the wind get lower-intensity gas values, which alters the gas dispersion map. This model is adopted from [26]. The robot avoids the obstacle by detecting it from one cell away using a laser range sensor, and, therefore, it instead stays in its current cell without colliding with it even if it is prescribed by the Q-network. This results in a negative reward (Eq. 3). Despite the heightened difficulty posed by obstacle-laden environments, the fundamental trends observed in the original (Fig. 3) and obstacle scenarios (Fig. 6) remain consistent. Both environments rely on GPR's capacity to predict based on past observations; however, adjusted value distributions and obstacles contribute to increased overall MSE and uncertainty values. The altered gas distribution in obstacle environments can make it non-Gaussian and that challenges the GPR's prediction quality, reflecting in

[2]Video: `https://youtu.be/Zqi7q4j94Xw`



(a) Reward (b) Uncertainty



(c) Unique cells visited (d) MSE

Fig. 3. Training statistics of key performance metrics in environments with no obstacle (1000-episode rolling average is shown).

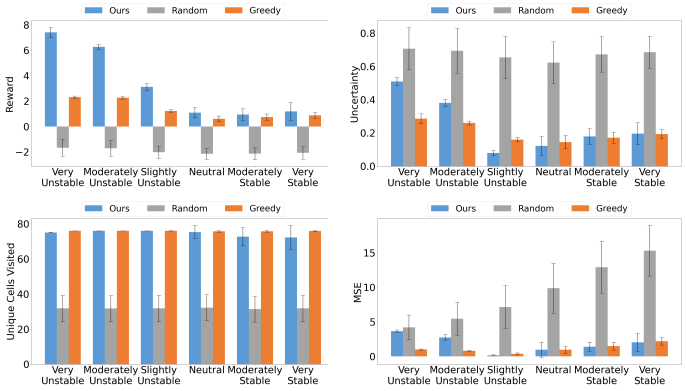
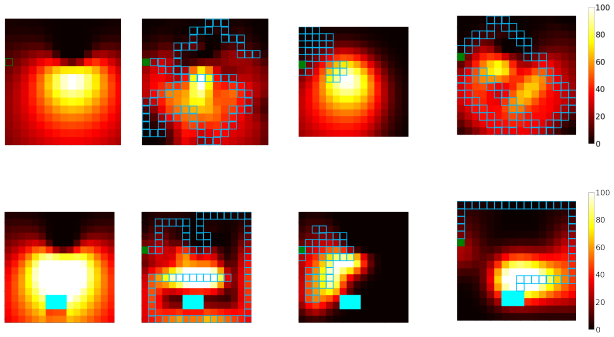


Fig. 4. Average testing performances of our model compared to the baselines (random and greedy) across 1000 test cases.



a) Truth b) Ours c) Random d) Greedy

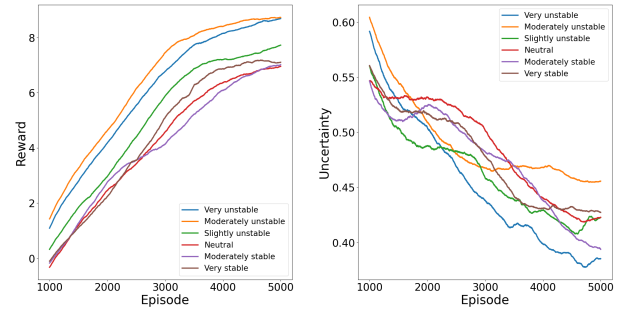
Fig. 5. The top and bottom rows show environments without and with (cyan rectangle) obstacles. (a) Ground truth gas distribution. An example of paths taken by (b) Ours, (c) Random, and (d) Greedy implementations during the testing phase with their respective gas distribution predictions superimposed on their paths. The gas type is 'very unstable'. The green box is the starting cell in each case. The path of the robot is indicated with blue-bordered cells.

higher MSE values. Nonetheless, overarching trends persist, with uncertainty declining more uniformly but less sharply, and MSE exhibiting a gentler initial decline in obstacle environments compared to the original setup, culminating in convergence (Fig. 6).

Cross-testing of the previously trained models was subjected to analysis of the same key four metrics, which are presented in Tables III - VI. As can be seen from the tables, the trained models tend to perform best in tests involving the gas type on which they were trained in the first place. However, in the majority of the cases, even when subjected to a new gas type during testing, our trained models performed relatively well illustrating the fact that our GDM-Net framework has the potential to be successful in adapting to unknown gas distributions.

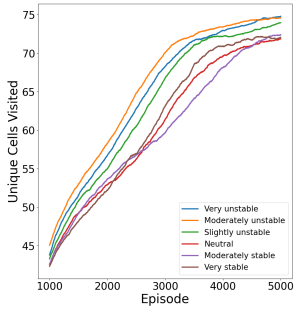
Fig. 4 presents the comparison of results among GDM-Net, random, and adaptive greedy techniques. It is clear from the plots that the random walk approach performed the worst across all key metrics. In terms of the uncertainty metric, the greedy always achieved lower final uncertainty compared to GDM-Net. We believe the main reason for that is only the greedy technique can adapt the posterior variance estimates online to each specific instance of gas concentrations; the GDM-Net technique merely follows the policy learned offline from the ensemble of gas concentrations in the training set. Yet our proposed GDM-Net was able to outperform the greedy technique in five out of six tested gas types in terms of reward, and in four gas types in terms of the final MSE. Interestingly, GDM-Net performed the worst compared to the greedy approach on the neutral gas type. Unlike the other types, neutral gas distribution has a lower gradient. That might be the reason behind such numbers. In general, our proposed GDM-Net framework outperforms adaptive greedy and random exploration by 62% and 151% in terms of rewards and by 47% and 345% in terms of final MSE in all gas types tested, suggesting superiority over popular baselines. The same trend can be noticed in testing with obstacle environments (Fig. 7). Due to the aforementioned reasons, all the implemented techniques saw upticks in MSE values, i.e., deterioration in prediction quality. However, our proposed GDM-Net framework still performed significantly better in terms of reward (117% and 29%), uncertainty (61% and 8%), and MSE (171%, 14%) than random and greedy respectively in the obstacle environments.

Finally, we present an example set of paths followed by the robot under the GDM-Net policy and random, greedy techniques in Fig. 5. Both the adaptive greedy technique

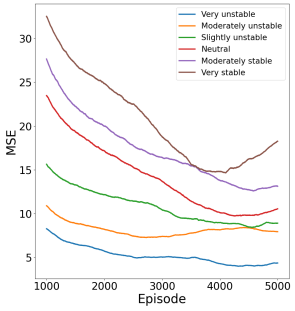


(a) Reward

(b) Uncertainty



(c) Unique cells visited



(d) MSE

Fig. 6. Training statistics of key performance metrics in an indoor environment with obstacles (1000-episode rolling average is shown).

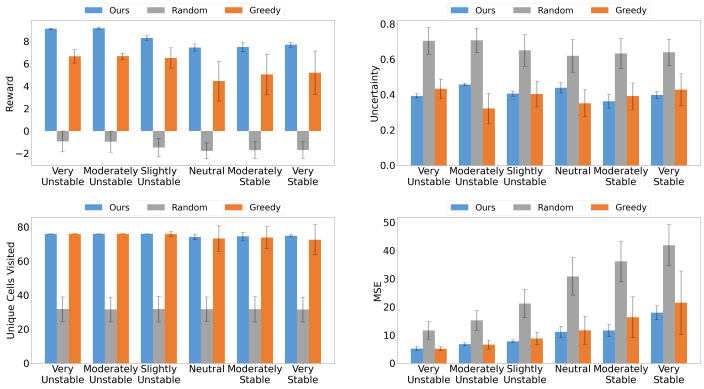


Fig. 7. Average testing performances of our model compared to the baselines (random and greedy) across 1000 test cases (indoor environment with obstacles).

and our GDM-Net policy let the robot scatter out the exploration to create a more accurate predicted model of the gas distribution, unlike the random approach that was more concentrated in one corner of the environment. The average MSEs per cell for our predicted, random, and greedy models are 2.90, 5.40, and 3.52 for no obstacle case and 8.49, 23.01, and 9.71 for the obstacle environment in Fig. 5. Thus, the MSE for the greedy approach in this example is 21% and 14% higher than our MSE in environments without and with an obstacle, respectively.

VI. Conclusion and Future Work

In this work, we investigated a deep reinforcement learning method for the gas distribution mapping problem that uses Gaussian Process regression exclusively during offline training. This is the first study that applies Gaussian Process regression and deep reinforcement learning together for mapping gas distribution. Our presented method is model-free and does not call for the robot to train the GPR model while in deployment with unknown gas distributions. Our proposed GDM-Net framework demonstrates a strong improvement over baselines based on GPR-based adaptive greedy and random walks across all six types of investigated gas distribution types. The results also show that using our GDM-Net framework, the robot could yield respectable performance metrics on gas types that it was not trained on. Furthermore, this opens an interesting avenue for researchers to explore for lightweight deployment of small-scale robots that follow trained policies without needing to build compute-intensive prediction models online. In the future, we plan to explore the effect of using a continuous action space instead of a discreet one used here following the existing studies in the literature. Finally, we are interested in extending the proposed GDM-Net framework to 3D scenarios.

References

[1] J. G Monroy, J.-L. Blanco, and J. Gonzalez-Jimenez, "Time-variant gas distribution mapping with obstacle information," *Autonomous Robots*, vol. 40, no. 1, pp. 1–16, 2016.

[2] A. Francis, S. Li, C. Griffiths, and J. Sienz, "Gas source localization and mapping with mobile robots: A review," *Journal of Field Robotics*, vol. 39, no. 8, pp. 1341–1373, 2022.

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

[4] M. Hutchinson, P. Ladosz, C. Liu, and W.-H. Chen, "Experimental assessment of plume mapping using point measurements from unmanned vehicles," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 7720–7726, IEEE, 2019.

[5] A. Lilienthal and T. Duckett, "Building gas concentration gridmaps with a mobile robot," *Robotics and Autonomous Systems*, vol. 48, no. 1, pp. 3–16, 2004.

[6] J. Burgués, V. Hernández, A. J. Lilienthal, and S. Marco, "Smelling nano aerial vehicle for gas source localization and mapping," *Sensors*, vol. 19, no. 3, p. 478, 2019.

[7] P. P. Neumann, S. Asadi, A. J. Lilienthal, M. Bartholmai, and J. H. Schiller, "Autonomous gas-sensitive microdrone: Wind vector estimation and gas distribution mapping," *IEEE robotics & automation magazine*, vol. 19, no. 1, pp. 50–61, 2012.

[8] A. J. Lilienthal, M. Reggente, M. Trincavelli, J. L. Blanco, and 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*, pp. 570–576, IEEE, 2009.

[9] V. H. Bennetts, A. J. Lilienthal, A. A. Khaliq, V. P. Sesé, and M. Trincavelli, "Gasbot: A mobile robotic platform for methane leak detection and emission monitoring," in *Workshop on Robotics for Environmental Monitoring at the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Citeseer, 2012.

[10] M. Mohan, T. Panwar, and M. Singh, "Development of dense gas dispersion model for emergency preparedness," *Atmospheric Environment*, vol. 29, no. 16, pp. 2075–2087, 1995.

[11] C. Liu, Y. Liao, S. Wang, and Y. Li, "Quantifying leakage and dispersion behaviors for sub-sea natural gas pipelines," *Ocean Engineering*, vol. 216, p. 108107, 2020.

[12] H. Fan, E. Schaffernicht, and A. J. Lilienthal, "Ensemble learning-based approach for gas detection using an electronic nose in robotic applications," *Frontiers in chemistry*, vol. 10, p. 863838, 2022.

[13] Y. Fukazawa and H. Ishida, "Estimating gas-source location in outdoor environment using mobile robot equipped with gas sensors and anemometer," in *SENSORS, 2009 IEEE*, pp. 1721–1724, IEEE, 2009.

[14] P. Ojeda, J. Monroy, and J. Gonzalez-Jimenez, "Information-driven gas source localization exploiting gas and wind local measurements for autonomous mobile robots," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1320–1326, 2021.

[15] X. He, J. A. Steiner, J. R. Bourne, and K. K. Leang, "Gaussian-based kernel for multi-agent aerial chemical-plume mapping," in *Dynamic Systems and Control Conference*, vol. 59162, p. V003T21A004, American Society of Mechanical Engineers, 2019.

[16] R. Rozas, J. Morales, and D. Vega, "Artificial smell detection for robotic navigation," in *Fifth International Conference on Advanced Robotics' Robots in Unstructured Environments*, pp. 1730–1733, IEEE, 1991.

[17] X.-x. Chen and J. Huang, "Odor source localization algorithms on mobile robots: A review and future outlook," *Robotics and Autonomous Systems*, vol. 112, pp. 123–136, 2019.

[18] I. Kulbaka, A. Dutta, L. Bölöni, O. P. Kreidl, and S. Roy, "Cnn-lstm-based deep recurrent q-learning for robotic gas source localization," in *2023 International Conference on Machine Learning and Applications (ICMLA)*, pp. 1060–1065, IEEE, 2023.

[19] T. Wiedemann, C. Vlaicu, J. Josifovski, and A. Viseras, "Robotic information gathering with reinforcement learning assisted by domain knowledge: an application to gas source localization," *IEEE Access*, vol. 9, pp. 13159–13172, 2021.

[20] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies," *Journal of Machine Learning Research*, vol. 9, no. Feb, pp. 235–284, 2008.

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

[22] A. Singh, A. Krause, C. Guestrin, and W. J. Kaiser, "Efficient informative sensing using multiple robots," *Journal of Artificial Intelligence Research*, vol. 34, pp. 707–755, 2009.

[23] T. Said, J. Wolbert, S. Khodadadeh, A. Dutta, O. P. Kreidl, L. Bölöni, and S. Roy, "Multi-robot information sampling using deep mean field reinforcement learning," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 1215–1220, IEEE, 2021.

[24] H. van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA* (D. Schuurmans and M. P. Wellman, eds.), pp. 2094–2100, AAAI Press, 2016.

[25] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[26] X. Chen, C. Fu, and J. Huang, "A deep q-network for robotic odor/gas source localization: Modeling, measurement and comparative study," *Measurement*, vol. 183, p. 109725, 2021.