Learn to Trace Odors: Autonomous Odor Source Localization via Deep Learning Methods

Lingxiao Wang
Dept. of Electrical Engineering and Computer Science
Embry-Riddle Aeronautical University
Daytona Beach, FL, 32114
lingxiaw@my.erau.edu

Shuo Pang
Dept. of Electrical Engineering and Computer Science
Embry-Riddle Aeronautical University
Daytona Beach, FL, 32114
shuo.pang@erau.edu

Jinlong Li
Marine Design and Research Institute of China
Shanghai, China
leoljl@126.com

***Abstract*—Autonomous odor source localization (OSL) has been viewed as a challenging task due to the nature of turbulent airflows and the resulting odor plume characteristics. Here we present an olfactory-based navigation algorithm via deep learning (DL) methods, which navigates a mobile robot to find an odor source without explicating specific search algorithms. Two types of deep neural networks (DNNs), namely traditional feedforward and convolutional neural networks (FNN and CNN), are proposed to generate robot velocity commands on x and y directions based on onboard sensor measurements. Training data is obtained by applying the traditional olfactory-based navigation algorithms, including moth-inspired and Bayesian-inference methods, in thousands of simulated OSL trials. After the supervised training, DNN models are validated in OSL tests with varying search conditions. Experiment results show that given the same training data, CNN is more effective than FNN, and by training with a fused data set, the proposed CNN achieves a comparable search performance with the Bayesian-inference method while requires less computational time.**

***Index Terms*—Olfactory-based navigation algorithms, odor source localization, end-to-end deep learning.**

I. INTRODUCTION

Olfaction is an important sensing ability widely used by animals to perform life-essential activities, such as homing, foraging, mate-seeking, and evading predators. Inspired by animals' olfactory behaviors, a mobile robot or an autonomous vehicle, equipped with odor-detection sensors, could locate an odor source in an unknown environment. The technology of employing robots to find odor sources is referred to as odor source localization (OSL) [1]. Some practical OSL applications include monitoring air pollution [2], locating chemical gas leaks [3], finding unexploded mines and bombs [4], and marine surveys such as locating underwater hydrothermal vents [5].

An effective olfactory-based navigation algorithm is critical for an OSL problem. Like image-based navigation algorithms, which extract the information from images as the reference to navigate a robot, olfactory-based navigation algorithms detect odor plumes as cues to guide a robot toward an odor source. During the plume tracing process, estimating plume locations is the main challenge since the plume propagation is not only related to the molecular diffusion that takes plumes away from the odor source but also the advection of airflow [6]. In laminar flow environments, plume dispersal is a steady and stable process, which results in a spatially coherent trajectory. In this scenario, the intuitive chemotaxis [7] navigation method, which commands the robot to trace plumes by following the odor concentration gradient, is applicable. However, this method fails in turbulent flow environments, where plumes are stretched and twisted to form an intermittent plume trajectory.

Alternatively, two other categories of olfactory-based navigation methods, namely bio-inspired and engineering-based (i.e., probabilistic) methods, are proposed. Bio-inspired methods direct a plume-tracing robot to mimic animal olfactory behaviors, such as the mate-seeking behaviors of male moths, which could locate female moths over a long distance via tracing emitted pheromones [8]. To complete this task, a male moth adopts a 'surge/casting' behavior pattern: a male moth flies upwind when it detects pheromones and traverses the wind direction when pheromones are absent. The 'surge/casting' model was successfully implemented on an underwater OSL trial [9], which directs an autonomous underwater vehicle to find a chemical source over a large search area. Shigaki *et al.* [10] presented a time-varying moth-inspired algorithm, where the duration of the 'surge' behavior is precisely controlled by an experience formula obtained from biological experiments.

By contrast, an engineering-based method utilizes mathematical and physics approaches to deduce odor plume distributions and predict odor source locations. Constructing a source probability map is a commonly-used approach to indicate the possible odor source locations. Methods that produce a source probability map includes Bayesian-inference method [11], particle filter [12], hidden Markov model (HMM) [13], fuzzy inference theory [14], source term estimation [15], and partially observable Markov decision process (POMDP) [16]. After the source estimations are obtained, a path planning algorithm, such as the artificial potential field (APF) [17] and A-star [18] algorithms, is employed to produce a search trajectory that guides a plume tracing robot moving toward the estimated target. Besides, Vergassola *et al.* [19] presented the 'infotaxis' method, which employs the information entropy to guide a plume tracing robot searching for an odor source.

Considering existing olfactory-based navigation methods,

the limitation of the bio-inspired methods is lacking the ability to estimate odor plume locations. When the robot loses plume contact, it can only conduct a time-consuming 'casting' behavior to re-detect plumes (e.g., [9]). Moreover, this type of methods usually fails in turbulent flow environments since the patchy and rapidly changing plume trajectory impedes the robot to constantly detect plumes. As for an engineering-based method (e.g., [11]), the computational cost grows significantly with respect to the size of the search area and the resolution of the source probability map. The high computational requirement for loop updating source estimations in every time step restrains its application on mobile robots, which have limited computational capacity. Derived from these considerations, a desired olfactory-based navigation algorithm should be effective in different flow environments and light-weighted in computational demands for implementing on robotic platforms.

In this work, we attempt to solve the OSL problem via deep learning (DL) approaches. The objective is to obtain a DL model that achieves a desired search performance as the engineering-based methods while requires less computational demands like the bio-inspired methods. The core idea of DL is letting the computer recognize the pattern of performing a task by relying on data sets without providing explicit instructions [20]. Various DL algorithms have demonstrated the powerful ability to model complex dependencies, such as feature extraction [21], medical diagnosis [22], and self-driving vehicles [23]. There are many benefits for applying DL methods on the OSL problem: 1) compared to complex engineering-based navigation methods, the query time of DL models is predictable and unaffected by search environments, which is suitable for implementing on mobile robots; 2) DL models can learn other successful navigation methods from demonstrations without explicating the specific searching algorithm; 3) DL models can continually improve the searching performance by adding more search examples in training data sets. However, this application's challenge is to obtain sufficient training data sets since OSL experiments are expensive to be repeatedly performed in different search conditions.

In the field of DL-based OSL methods, limited work has been carried. Recent developments include employing DL models to predict the gas leaking locations from a stationary sensor network and adopting reinforcement learning algorithms to learn a plume tracing algorithm. Kim *et al.* [24] trained a recurrent neural network (RNN) to predict possible odor source locations via the data obtained from a stationary sensor network, where training data is acquired from a simulation program. Hu *et al.* [25] presented a plume tracing algorithm based on the model-free reinforcement learning algorithms, where the deterministic policy gradient (DPG) algorithm is employed to control an autonomous underwater vehicle to search odor plumes and find the odor source. By summarising these works, it can be discovered that despite the high-level intelligence and potential benefits of DL technologies, using DL methods to solve an OSL problem is still in its infancy and requires further research.

In this paper, the feasibility of implementing DL methods on OSL problems is investigated. Specifically, two types of deep neural networks (DNNs), namely feedforward and convolutional neural networks (FNN and CNN), are employed to guide a plume tracing robot in finding an odor source. During the plume tracing process, DNNs produce suitable robot commands based on onboard sensor measurements. Two paradigms from categories of bio-inspired and engineering-based methods, namely moth-inspired [26] and Bayesian-inference [11] methods, are employed as expert methods to generate training data sets in a realistic simulation program. It should be mentioned that collecting training data from real OSL trials is preferred, but considering the difficulty of repeatedly performing actual OSL tests and the demand for a large quantity of training data, utilizing a simulation program is an acceptable option. After the supervised training, the proposed DNNs are implemented in OSL tests with various search conditions, where the success rate and the averaged search time are calculated. To analyze the generalization error of the trained DNN models, OSL tests are conducted in previously unseen environments. Additionally, trained DNN models are also compared with traditional navigation methods to evaluate the validity of the proposed methods.

II. Methodology

A. Problem Formulation

The main objective of this work is to obtain a DNN model that guides a plume tracing robot to locate an odor source in an unknown environment. To achieve this goal, the DNN model is trained to calculate suitable robot commands $\mathbf{C}$ based on robot states $\mathbf{S}$:

$$\mathbf{C} = \mathcal{F}_\theta(\mathbf{S}). \tag{1}$$

This DNN model is parametrized by a parameter vector θ, and the optimal θ is found during the process of supervised training, which minimizes the difference between outputs of the DNN and the ones demonstrated by expert methods.

B. Generate Training Data Sets

1) Defining Inputs and Outputs of DNNs: As mentioned, two expert methods, namely moth-inspired and Bayesian-inference methods, are employed to generate training data sets. To learn expert methods, DNNs should be offered with similar input information. In the moth-inspired method, odor concentrations (ρ) and wind directions (ϕ) determine whether the robot is in the 'surge' or 'casting' search phase, and for the Bayesian-inference method, robot positions (x and y), wind speeds (u_x and u_y), and algorithm running time (t) are essential to estimate odor source locations. All aforementioned variables should be included in DNN's input state, therefore, the input state vector $\mathbf{S}$ is defined as:

$$\mathbf{S} = (t, u_x, u_y, \rho, x, y, v_x, v_y) \tag{2}$$

where u_x, u_y, v_x and v_y are wind and robot speeds in x and y directions, respectively. To control a mobile robot on a 2-D

TABLE I
DEFINITIONS OF VARIABLES

Symbols of Variables	Definitions of Variables
t (s)	Algorithm running time
u (m/s)	Wind speed at the robot position
ϕ (rad)	Wind direction at the robot position
ρ (mmpv)	Odor concentration at the robot position
x (m)	Robot horizontal position
y (m)	Robot vertical position
ψ (rad)	Robot heading angle
v (m/s)	Robot speed
v_c (m/s)	Robot speed command
ψ_c (rad)	Robot heading command

mmpv: million molecules per cm^3

plane, only speed and yaw angle commands (v_c and ψ_c) are needed. Thus, DNN's outputs are defined as:

$$\mathbf{C} = (v_{c,x}, v_{c,y}) \tag{3}$$

where $v_{c,x}$ and $v_{c,y}$ are robot velocity commands on x and y directions, respectively.

It should be mentioned that we convert angle-related variables, including wind directions (ϕ), robot yaw angles (ψ), and yaw angle commands (ψ_c), to vector forms in $\mathbf{S}$ and $\mathbf{C}$:

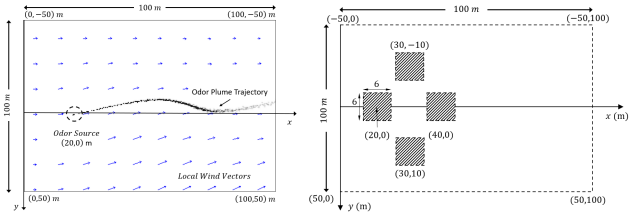
$$\begin{cases} u_x = u\cos\phi, v_x = v\cos\psi, v_{c,x} = v_c\cos\psi_c \\ u_y = u\sin\phi, v_y = v\sin\psi, v_{c,y} = v_c\sin\psi_c \end{cases}. \tag{4}$$

This is because angles do not make a good DL model input: one angle could refer to two different values such as $-\pi$ and π, and angles should not matter if the corresponding speed is zero. For the easy reference, Table I lists variables and corresponding definitions in $\mathbf{S}$ and $\mathbf{C}$.

2) The Simulated Search Environment: In this work, the OSL is considered as a two-dimensional (2-D) problem since the aimed robotic platform is a ground robot. A realistic OSL simulation program designed based on [6] is employed as the platform to produce training data sets. This simulator emulates odor plume trajectories in a time-varying airflow environment and robot motion behaviors in the search area. Some other researchers, such as [27] and [28], also employed this simulator as the environment to evaluate their works.

Fig. 1(a) presents the simulated search area (100×100 m^2), where a coordinate system ($x - y$) is constructed to indicate positions. The plume trajectory is painted with a grey-scale patchy trail, and local wind vectors are represented by arrows in the background. Wind vectors are calculated by a mean flow, $\mathbf{U}_0$, plus Gaussian white noises with zero mean and ς variance. Thus, by adjusting values of $\mathbf{U}_0$ and ς, varying airflow fields and the corresponding odor plume trajectories can be obtained. To reduce the spatial correlation in training data sets, at the beginning of an OSL trial, the odor source location is randomly selected from four regions as indicated in Fig. 1(b).

In the simulation, a two-wheeled mobile robot, as presented in Fig. 2, is employed as the robotic platform to implement olfactory-based navigation algorithms. It is assumed that the robot is equipped with sufficient sensors to measure variables



(a) (b)

Fig. 1. (a) The simulated search environment. (b) Possible odor source locations, where for each OSL trial in training data sets, an odor source is randomly selected from a position inside shadowed regions.

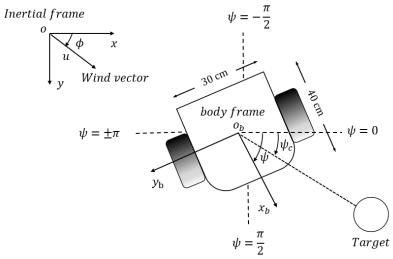


Fig. 2. The two-wheeled mobile robot used in the simulation program. Definition of robot-related parameters are labeled in the diagram.

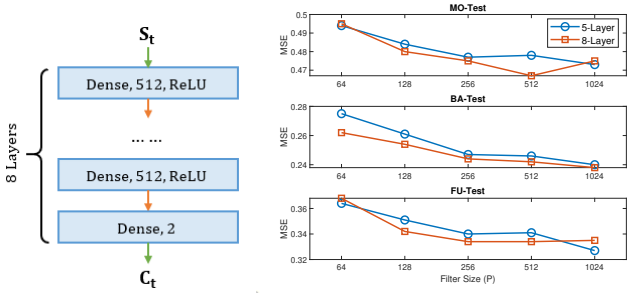
in the state vector $\mathbf{S}$. The onboard sensor suite and the corresponding functionalities are listed below:

- An onboard computer: implement navigation algorithms and measure algorithm running time t;
- A chemical sensor: measure odor concentration ρ;
- An anemometer: measure wind speeds u and directions ϕ in the inertial frame;
- A positioning sensor: measure robot positions (x, y) and speeds v in the inertial frame;
- A compass: measure robot yaw angles ψ.

Comparing to the large scale of the search area, the size of the robot is negligible. Thus, the robot is approximated as a single point in the simulation.

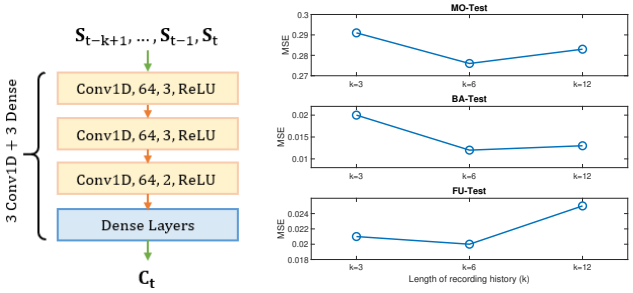
3) Training Data Specifications: To collect training data, around 6000 OSL trials are conducted for each expert method. In an OSL trial, a data tuple $\gamma_t = (\mathbf{S}_t, \mathbf{C}_{exp,t})$ that consists of input states $\mathbf{S}_t$, which are obtained from robot sensor measurements, and expert commands $\mathbf{C}_{exp,t}$, which are produced by the implemented expert method, is recorded at every time t during the plume tracing process. An OSL trial is considered as complete if the robot reaches the odor source location or the algorithm running time is beyond the time limit, i.e., 400 s in this work. It should be mentioned that we consider an odor source has been found if the robot is in vicinity of it. In real-world applications, this step, i.e., source declaration, could be complete with aids of external sensors such as cameras, which could recognize an odor source from a close distance.

Depending on the type of the expert method, two training data sets, namely MO-Train (obtained from the moth-inspired method) and BA-Train (obtained from the Bayesian-inference method), are acquired. Previous experiment data



(a) (b)

Fig. 3. (a) The structure of the proposed FNN. Labels inside a blue layer represent the layer type, filter size, and activation function, respectively. A dense layer indicates a fully-connected neural network; (b) MSEs of FNNs with varying filter sizes and hidden layers on testing data sets



(a) (b)

Fig. 4. (a) The structure of the proposed CNN. Labels inside a yellow layer represent layer type, filter size, kernel size, and activation function, respectively. The Conv1D layer means a convolutional neural network. The last 'Dense Layers' block contain 3 consecutive dense layers with 512, 512, and 2 filters, respectively; (b) MSEs of CNNs with the varying length of recording history on testing data sets

[29] reveals that bio-inspired methods are more efficient (i.e., require less searching time) in laminar flow environments while engineering-based methods outperforms the counterpart in turbulent flow environments in terms of the search success rate. Thus, to learn benefits from two expert methods, two training data sets are combined to form a fused data set, which is named as FU-Train.

Only 80% of training data is used to train DNN models, while 10% of the remaining data, termed testing data set, is used to test DNN models after the training, and the last 10% data, termed validation data set, is used to compute validation errors during the training process: the training process is terminated once the validation error is not improved in 20 episodes. The validation error is defined as mean absolute errors (MAEs) in this work, i.e., $1/n \cdot \sum_{i=1}^{n} |\mathcal{F}_\theta(\mathbf{S}_i) - \mathbf{C}_{i,exp}|$ where n is the size of validation data set.

C. Design DNNs for OSL Problems

Two types of neural networks, i.e., FNN and CNN, are selected for the representation of $\mathcal{F}_\theta$. The motivation for choosing FNN is that we want to use a simple DNN structure to investigate the viability of implementing DL approaches on OSL problems. Besides, the intuitive FNN could also be employed as the baseline to evaluate the performance of other types of DNN models in the OSL problem. Fig. 3(a) presents the structure of a FNN model. To determine the optimal numbers of hidden layers and filter sizes, varying values are investigated. Fig. 3(b) shows the mean square errors (MSEs) of implementing different structure FNNs with varying hidden layers and filter sizes on testing data sets. It can be observed that larger models (i.e., more layers and filters on each layer) achieve better performances (i.e., lower MSEs) but overfit when the model is too complicated (i.e., the MSE increases). Based on plots, the FNN with 8-layer and 512 filters is selected for implementing in OSL tests.

Due to the characteristics of FNN structure, the sensor data history (i.e., $\mathbf{S}_{t-1}$, $\mathbf{S}_{t-2}, ...$) is ignored in calculating FNN outputs. In terms of learning expert methods, the FNN may learn the moth-inspired method well since this expert method does not consider history information as well. However, for the other expert method, i.e., Bayesian-inference method, the sensor data history is the necessary information to estimate the odor source location. To address this issue, a CNN model is employed to process sensor data and produce robot commands. As presented in Fig. 4(a), the proposed CNN not only considers sensor data at the current time but also from previous k time steps. Specifically, three convolution layers are employed to extract features from sensor data history. Then, the extracted features are fed to three dense layers to produce robot commands. To find the optimal value of k, different numbers are investigated. Fig. 4(b) shows MSEs of CNNs with varying k on testing data sets. It can be observed that MSE is lowest on all testing data sets when $k = 6$. Therefore, we choose this CNN structure to implement in later OSL tests.

D. Training DNN Models

In this work, we employ the supervised learning [30] as the training algorithm, which aims to find the optimal parameter vector θ^* that minimizes the loss function J. The loss function J is defined as the mean square error between DNN outputs $\mathcal{F}_\theta(\mathbf{S})$ and expert demonstrations $\mathbf{C}_{exp}$, which can be represented as:

$$J\left(\Gamma_B\right) = \frac{1}{N_B} \cdot \sum_{i=j}^{j+N_B} \left(\mathcal{F}_\theta\left(\mathbf{S}_i\right) - \mathbf{C}_{i,exp}\right)^2, \tag{5}$$

where Γ_B is a mini-batch that contains N_B (32 in our work) samples from a training data set. The gradient of the cost function with respect to model parameters is calculated using the backpropagation algorithm [31], and the optimization algorithm that updates model parameters is the Adam optimizer [32].

To train FNNs, the order of training samples is randomized to reduce the temporal correlation in training data sets. This procedure is skipped in the process of training CNNs since the proposed CNN produces robot commands based on time series data. The training process is considered as complete if one of the following two conditions is satisfied: 1) the training

epoch reaches the limit (i.e., 500 in implementations); 2) the validation error does not improve in 20 consecutive epochs. Google® TensorFlow [33] is employed as the framework to construct and train DNN models. Training the proposed FNN and CNN with the FU-Train data set (1.6 million data tuples) on an Intel® i7-8750 CPU with the Nvidia® GeForce GTX 1070 GPU acceleration takes around 4 hours and 2 hours, respectively.

III. Experiments

A. Sample OSL Trials

To exam the effectiveness of DNN models after training, we first implement the proposed FNN and CNN, trained with the fused training data set (i.e., FU-Train), in sample OSL trials, where the mean flow velocity of the environment is $\mathbf{U}_0 = (1, 0)$ m/s and the variance of Gaussian white noises is $\varsigma = 3$. The robot starts at $(60, -40)$ m, and the odor source is located at $(20, 0)$ m.

Fig. 5 shows search trajectories of two expert methods (i.e., moth-inspired and Bayesian-inference methods) and the proposed DNN models (i.e., FNN and CNN) in the sample OSL trial. In these plume tracing methods, the robot first adopts a 'zigzag' search strategy [26] to find the existence of plumes, and after the first plume detection, the corresponding navigation method is activated, which guides the robot to trace plumes. In this laminar flow environment, the moth-inspired method achieves a shorter search time compared to the Bayesian-inference method (93 s vs 126 s). This is because in a laminar flow environment, odor plumes form a stable and continuous trajectory, in which the 'surge' behavior of the moth-inspired method is more effective to quickly trace up-wind and locate the odor source compared to engineering-based methods.

It can be observed in Fig. 5(c) and Fig. 5(d) that both FNN and CNN can correctly locate the odor source in the sample OSL trial. In addition, both trajectories is similar to the one produced by the moth-inspired method, which is a preferred navigation method in this type of environment (i.e., laminar flow environments). Specifically, the FNN search trajectory shows a 'casting' alike behavior to traverse plumes when plumes are absent, while the proposed CNN acts analogously to the 'surge' behavior that controls the robot consistently detecting plumes by moving up-wind. Comparing FNN and CNN search trajectories, the CNN generates a smoother trajectory and finds the odor source within a shorter search time (97 s vs 93 s). Besides, it can be seen that the robot can consistently detect plumes by following the trajectory produced by CNN, which is beneficial for the plume-tracing robot to acquire adequate odor source information.

B. Varying Search Conditions

To investigate the generalization of the proposed DNN models, they are implemented in OSL tests with different search conditions, including varying robot initial positions, odor source locations, and environmental settings.

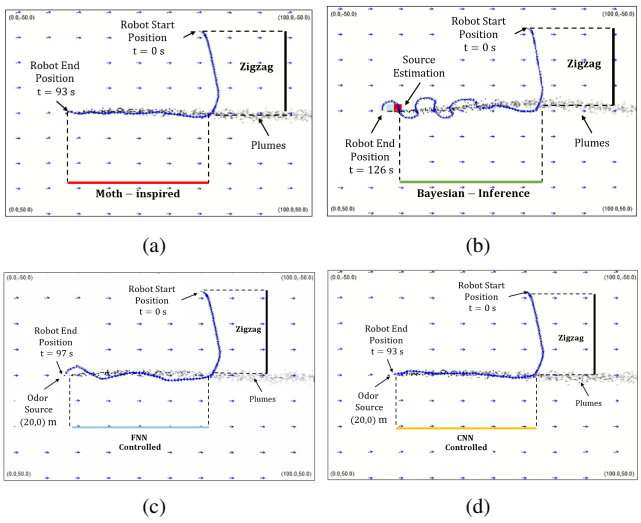


Fig. 5. Search trajectories of expert methods and DNNs in the sample OSL trial, where (a) Moth-inspired method, (b) Bayesian-inference method, (c) FNN, and (d) CNN. The robot trajectory is presented by the blue curve, where the robot initial and end positions are indicated in diagrams. The duration of each navigation method is labeled by different color bars, where black represents zigzag; red represents moth-inspired; green represents Bayesian-inference; light blue represents FNN; yellow represents CNN.

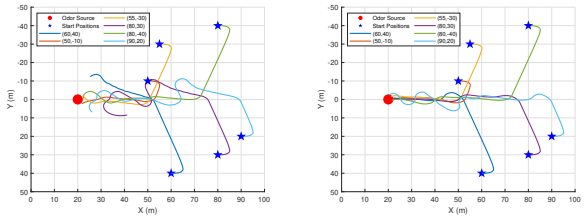


Fig. 6. Search trajectories generated by (a) FNN and (b) CNN in OSL tests with varying robot initial positions

1) Varying Robot and Source Positions: In this group of tests, the proposed DNN models are evaluated in environments with different robot initial positions and odor source locations. Fig. 6 presents the search trajectories of the proposed FNN and CNN with six different robot initial positions that are unseen for DNN models in the training process. It can be seen that without considering the sensor data history, the proposed FNN can barely adapt to new search conditions, where only three out of six trajectories correctly find the odor source. On the other hand, given the same training data sets, the CNN is more effective than FNN in this group of tests, where all trajectories terminate at the odor source location, i.e., the robot correctly finds the odor source in six tests under the direction of the proposed CNN model.

Then, the DNN models are implemented in environments with different the odor source locations. It was discovered in previous test results that if the odor source location is fixed in the training data sets, a DNN model can memorize the odor source location and skip the plume tracing process, i.e., the robot controlled by the DNNs will proceed directly to the fixed

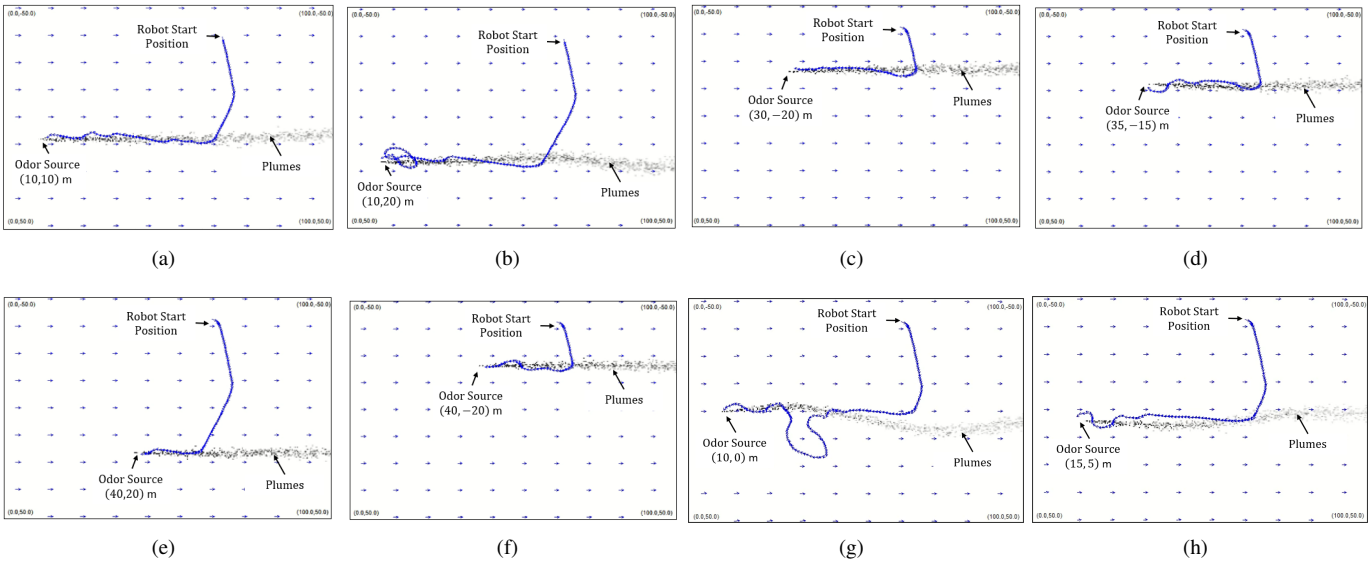


Fig. 7. Search trajectories of the proposed CNN in environments with varying odor source locations, where (a) (10,10) m, (b) (10,20) m, (c) (30,-20) m, (d) (35,-15) m, (e) (40,20) m, (f) (40,-20) m, (g) (10,0) m, and (h) (15,5) m.

TABLE II
ENVIRONMENTAL SETTINGS AND SEARCH TIME OF DIFFERENT NAVIGATION METHODS

Environment Index	Mean Wind Velocity $\mathbf{U}_0$ (m/s)	Gaussian Noise Variance ς	Test Index	Moth-inspired Method (s)	Bayesian-inference Method (s)	The Proposed FNN (s)	The Proposed CNN (s)
Env. 1	$(1, 0)$	3	Test 1	93	123	97	93
			Test 2	91	116	98	103
			Test 3	96	139	123	102
Env. 2	$(1, 0)$	10	Test 1	93	151	-	129
			Test 2	128	125	-	95
			Test 3	94	128	-	110
Env. 3	$(1, -0.4)$	8	Test 1	-	129	-	138
			Test 2	-	116	-	140
			Test 3	-	139	-	137
Env. 4	$(2.5, 0)$	10	Test 1	90	110	-	106
			Test 2	94	176	-	104
			Test 3	95	129	-	145
Env. 5	$(2.5, 0.4)$	12	Test 1	239	95	-	159
			Test 2	-	100	-	160
			Test 3	-	133	-	108

-: Fail to locate the odor source within 500 s.

odor source location instead of tracing plumes. To address this problem, the odor source location is varying in each OSL trial in the training data sets.

To verify the generalization of the proposed DNN models on varying odor source locations, eight different odor source locations that are unseen to DNN models, i.e., the ones that are not included in training data sets, are tested. Fig. 7 presents search trajectories of implementing the proposed CNN in these tests. We can observe that the robot successfully finds the odor source in these eight trials, which demonstrates the validity of the proposed CNN model for finding varying odor source locations. The proposed FNN fails to find the odor source in this group of tests. One possible approach to improve the FNN's search performance is to enlarge the training data set, i.e., make training data set cover more search examples.

2) Varying Environmental Settings: In this group of tests, environmental settings, i.e., mean flow speed $\mathbf{U}_0$ and Gaussian noise variance ς, are varying to produce different airflow environments. To evaluate the search performance of the proposed DNNs, two expert methods are also implemented and compared in this group of tests. Table II presents five different environments and the corresponding search time of four navigation methods. Each navigation method is repeatedly performed three times in every environment. As a remainder, environmental settings in Env. 2-5 are unseen for DNN models during the training process.

As presented in Table II, the proposed FNN can only find the odor source in a laminar flow environment, i.e., Env. 1, while the proposed CNN succeeds in all tests. This result

TABLE III
STATISTICAL RESULTS OF REPEATED TESTS AND THE COMPARISON OF DIFFERENT NAVIGATION METHODS

	Total Tests	Successful Tests	Success Rate	Averaged Search Time (s)
Moth-inspired Method	15	10	67%	240.9
Bayesian-inference Method	15	15	100%	127.3
The Proposed FNN	15	3	20%	421.2
The Proposed CNN	15	15	**100%**	**121.9**

confirms that without considering the sensor data history as inputs, the FNN structure can hardly learn an effective plume tracing method, which is not suitable for an OSL problem. By contrast, search results of CNN in this group of tests indicate that the proposed CNN can learn an effective navigation strategy and apply the learned knowledge on new environments.

Table III presents the statistical results of different navigation methods on varying environment tests. Compared to expert methods, the proposed FNN barely succeeds in these tests, As for the proposed CNN model, it outperforms the moth-inspired method in terms of the success rate (100% vs 67%) and achieves a shorter averaged search time than the Bayesian-inference method (121.9 s vs 127.3 s). Additionally, compared to complex Bayesian-inference method, the CNN method is preferred to be implemented on real-world applications due to the low computational complexity. As mentioned, the computational time of the Bayesian-inference method grows significantly with the increase of the size of the search area [11]. When the search area becomes complex and large, the Bayesian-inference method is not suitable for implementing on robotic platforms due to the long querying time. On the other hand, the computational cost of the proposed CNN is fixed, which is independent with the size of the search area. Thus, the proposed CNN is preferable for real-world applications.

C. Discussions

From experiment results, we observe that the DNN structure is a critical factor that affects the DNN search performance in OSL tests. Given the same training data, the FNN structure is not as effective as CNN in experiments (e.g., Section III-B). Because plume-tracing is a continuous process, the search context is important for the robot to make decisions. The CNN structure, which generates robot commands based on previous sensor data, is suitable for the plum-tracing process. Another essential factor is the quality and quantity of training data sets. To achieve satisfying search results in unseen environments, a DNN model should be trained with data set that covers sufficient search examples in both laminar and turbulent airflow environments. To improve this work, other expert methods could also be considered to generate training data sets, and more types of DNN models can be investigated and implemented in OSL problems.

IV. CONCLUSION

In this paper, we present a new design of the olfactory-based navigation method via deep learning approaches. Two types of DNN models are evaluated, namely FNN and CNN, which control a plume-tracing robot to locate an odor source based on the robot sensor data. After the supervised training with traditional moth-inspired and Bayesian-inference methods, the proposed FNN and CNN are validated in OSL tests with varying search conditions. Simulation results show that given the same training data, CNN performs better than FNN on unseen search environments. Compared to traditional methods, experiment results show that the proposed CNN is more desired for autonomous OSL problems since it achieves a comparable search performance with an engineering-based method but is more stable and requires less computational time.

REFERENCES

[1] 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.

[2] M. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 24–39, 2012.

[3] S. Soldan, G. Bonow, and A. Kroll, "Robogasinspector-a mobile robotic system for remote leak sensing and localization in large industrial environments: Overview and first results," *IFAC Proceedings Volumes*, vol. 45, no. 8, pp. 33–38, 2012.

[4] R. A. Russell, "Robotic location of underground chemical sources," *Robotica*, vol. 22, no. 1, pp. 109–115, 2004.

[5] G. Ferri, M. V. Jakuba, and D. R. Yoerger, "A novel method for hydrothermal vents prospecting using an autonomous underwater robot," in *2008 IEEE International Conference on Robotics and Automation*. IEEE, 2008, pp. 1055–1060.

[6] J. A. Farrell, J. Murlis, X. Long, W. Li, and R. T. Cardé, "Filament-based atmospheric dispersion model to achieve short time-scale structure of odor plumes," *Environmental fluid mechanics*, vol. 2, no. 1-2, pp. 143–169, 2002.

[7] H. Ishida, K.-i. Suetsugu, T. Nakamoto, and T. Moriizumi, "Study of autonomous mobile sensing system for localization of odor source using gas sensors and anemometric sensors," *Sensors and Actuators A: Physical*, vol. 45, no. 2, pp. 153–157, 1994.

[8] R. T. Cardé and A. Mafra-Neto, "Mechanisms of flight of male moths to pheromone," in *Insect pheromone research*. Springer, 1997, pp. 275–290.

[9] J. A. Farrell, S. Pang, and W. Li, "Chemical plume tracing via an autonomous underwater vehicle," *IEEE Journal of Oceanic Engineering*, vol. 30, no. 2, pp. 428–442, 2005.

[10] S. Shigaki, T. Sakurai, N. Ando, D. Kurabayashi, and R. Kanzaki, "Time-varying moth-inspired algorithm for chemical plume tracing in turbulent environment," *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 76–83, 2017.

[11] S. Pang and J. A. Farrell, "Chemical plume source localization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 5, pp. 1068–1080, 2006.

[12] J. Li, Q. Meng, Y. Wang, and M. Zeng, "Odor source localization using a mobile robot in outdoor airflow environments with a particle filter algorithm," *Autonomous Robots*, vol. 30, no. 3, pp. 281–292, 2011.

[13] J. A. Farrell, S. Pang, and W. Li, "Plume mapping via hidden Markov methods," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 850–863, 2003.

[14] L. Wang, S. Pang, and J. Li, "Olfactory-based navigation via model-based reinforcement learning and fuzzy inference methods," *IEEE Transactions on Fuzzy Systems*, 2020.

[15] F. Rahbar, A. Marjovi, and A. Martinoli, "An algorithm for odor source localization based on source term estimation," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 973–979.

[16] H. Jiu, Y. Chen, W. Deng, and S. Pang, "Underwater chemical plume tracing based on partially observable markov decision process," *International Journal of Advanced Robotic Systems*, vol. 16, no. 2, p. 1729881419831874, 2019.

[17] S. Pang and F. Zhu, "Reactive planning for olfactory-based mobile robots," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 4375–4380.

[18] L. Wang and S. Pang, "Chemical plume tracing using an auv based on pomdp source mapping and a-star path planning," in *OCEANS 2019 MTS/IEEE SEATTLE*. IEEE, 2019, pp. 1–7.

[19] M. Vergassola, E. Villermaux, and B. I. Shraiman, "'infotaxis' as a strategy for searching without gradients," *Nature*, vol. 445, no. 7126, p. 406, 2007.

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

[21] G. Cheng and J. Han, "A survey on object detection in optical remote sensing images," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 117, pp. 11–28, 2016.

[22] K. Kourou, T. P. Exarchos, K. P. Exarchos, M. V. Karamouzis, and D. I. Fotiadis, "Machine learning applications in cancer prognosis and prediction," *Computational and structural biotechnology journal*, vol. 13, pp. 8–17, 2015.

[23] B. Paden, M. Čáp, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Transactions on intelligent vehicles*, vol. 1, no. 1, pp. 33–55, 2016.

[24] H. Kim, M. Park, C. W. Kim, and D. Shin, "Source localization for hazardous material release in an outdoor chemical plant via a combination of lstm-rnn and cfd simulation," *Computers & Chemical Engineering*, vol. 125, pp. 476–489, 2019.

[25] H. Hu, S. Song, and C. P. Chen, "Plume tracing via model-free reinforcement learning method," *IEEE transactions on neural networks and learning systems*, 2019.

[26] W. Li, J. A. Farrell, S. Pang, and R. M. Arrieta, "Moth-inspired chemical plume tracing on an autonomous underwater vehicle," *IEEE Transactions on Robotics*, vol. 22, no. 2, pp. 292–307, 2006.

[27] Y. Tian and A. Zhang, "Simulation environment and guidance system for auv tracing chemical plume in 3-dimensions," in *2010 2nd International Asia Conference on Informatics in Control, Automation and Robotics (CAR 2010)*, vol. 1. IEEE, 2010, pp. 407–411.

[28] Q. Lu, Q.-L. Han, and S. Liu, "A cooperative control framework for a collective decision on movement behaviors of particles," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 859–873, 2016.

[29] L. Wang and S. Pang, "An implementation of the adaptive neuro-fuzzy inference system (anfis) for odor source localization," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2021.

[30] E. Alpaydin, *Introduction to machine learning*. MIT press, 2020.

[31] H. J. Kelley, "Gradient theory of optimal flight paths," *Ars Journal*, vol. 30, no. 10, pp. 947–954, 1960.

[32] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[33] S. S. Girija, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *Software available from tensorflow. org*, vol. 39, no. 9, 2016.