The Journal of Supercomputing (2022) 78:14794–14812
https://doi.org/10.1007/s11227-022-04478-4

Enhancing gas detection-based swarming through deep reinforcement learning

Sangmin Lee[1] · **Seongjoon Park**[1] · **Hwangnam Kim**[1]

Accepted: 17 March 2022 / Published online: 9 April 2022

Abstract

Swarm-Intelligence (SI), the collective behavior of decentralized and self-organized system, is used to efficiently carry out practical missions in various environments. To guarantee the performance of swarm, it is highly important that each object operates as an individual system while the devices are organized as simple as possible. This paper proposes an efficient, scalable, and practical swarming system using gas detection device. Each object of the proposed system has multiple sensors and detects gas in real time. To let the objects move toward gas rich spot, we propose two approaches for system design, vector-sum based, and Reinforcement Learning (RL) based. We firstly introduce our deterministic vector-sum-based approach and address the RL-based approach to extend the applicability and flexibility of the system. Through system performance evaluation, we validated that each object with a simple device configuration performs its mission perfectly in various environments.

Keywords Swarm-intelligence · Remote sensing · Reinforcement learning · Multi-robot control

Sangmin Lee and Seongjoon Park contributed equally to this work.

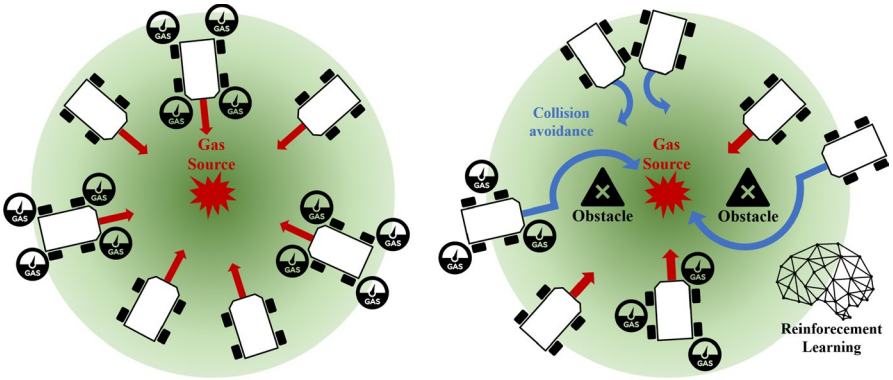
Preliminary version of this paper appeared in the Proceedings of the 6th International Conference on Next Generation Computing 2020 (ICNGC).

✉ Hwangnam Kim
hnkim@korea.ac.kr

Sangmin Lee
lsm5505@korea.ac.kr

Seongjoon Park
psj900918@korea.ac.kr

[1] Korea University, Seoul, Republic of Korea



(a) Vector-sum based system overview. **(b)** RL based system overview.

Fig. 1 Overview of the proposed systems

1 Introduction

Recently, micro-robot control technology has been developed with the aim of processing tasks that are difficult for humans to do in various environments such as production lines [40], disaster environments [30], logistics [31], surveillance [22], and medical systems [7]. Along with this trend, Swarm Intelligence (SI) technology is also in the spotlight as a promising robotic solution. The concept of SI in robotics was introduced in the early 1990s [4], but its commercialization was not accomplished due to the limitations of the technology. However, in recent years, with the development of robotics, research and application of SI technology has been actively carried out [28, 39]. This trend is because in many cases it is much more productive when several simple robots perform a particular job cooperatively. The collective flight of wild geese, the collaboration of ant groups, and the collective action of bees gave SI an original idea [23]. In case of space exploration, multiple robots, rather than a single robot, assigned to each mission enable much more comprehensive and three-dimensional exploration. It is more efficient to fly hundreds of drones and conduct simultaneous searches than to fly one in several missions, such as survivor searching in a forest fire area [1]. It may be much better to use a number of small robots to detect leaks in gas pipes [33]. No matter how many are lost in the process, the remaining objects could continue their global mission.

In the field of robotics, various studies have been conducted to explore algorithms to control these swarming robots. Several studies have been conducted including the systems using infrared and acoustic signal. Studies that constructed various simulation environments [16, 35] were also addressed. However, despite of the explosive attention of deep learning, collaboration with SI concept has not been deeply considered, because of difficulty of satisfying SI concept; in SI system, each object should determine the control decision with local and particular information.

In this paper, we propose a novel swarming system based on Reinforcement Learning (RL) [9] with gas sensing, which is shown in Fig. 1. The system is focused on pinpointing the source point, especially in situations of gas leakages or vapor

distribution in indoor environments. The proposed system includes a process of converting gas sensing data into distance information. Therefore, it is difficult to utilize the system in a space in which gas is dispersed due to a strong air flow or in a space saturated by gas leaked for a long period of time. Our swarming system is specialized in quickly detecting gas leakage points in the early stages of the accident by utilizing multiple robots in an indoor space where the structure is unknown. In terms of security guarantees, the proposed systems are essential. For example, with the systems, it is possible to find source points at gas accident sites to ensure safety quickly, or to find danger points in high-risk areas to prevent accidents. The proposed system is highly flexible in determining the shape of the robot frame, such as the number and location of mounted sensors, and dimensions.

We introduce a robot control system that uses vector sum based on gas sensing data (Fig. 1a). We then validate the performance through simulation and address the advantages and the limitation of the system. To compensate these shortcomings, we further propose an advanced model applying RL technique. The advanced design shows high performance of scalable multi-robot migration, collision prevention between robots, and obstacle avoidance while moving. As a result, each individual does not have high intelligence, but based solely on sensing data, it can accurately find the source point without collision (Fig. 1b).

The remaining part of this paper is organized as follows. We introduce related work in Sect. 2. In Sect. 3, we describe detail design of our proposed system. Section 4 validates the performance of the proposed system in simulation environments. Finally, Sect. 5 concludes the paper.

2 Releated work

2.1 Swarm-robotics

SI is an artificial intelligence based on distributed collective behavior and self-organizing systems. SI is a system made of simple objects that locally interact with other objects. Even without a central control structure that dictates the behavior of each object, it acts according to a simple rule, which creatively leads to an "intelligent-looking" action through a local, somewhat random interaction, without understanding the entire rule [3].

Swarm Robotics (SR) [5] refers to the technology of moving several simple robots at once, and its background is in SI. SR was initially used to support and validate biological research. The ant cluster optimization algorithm [11] and the particle cluster optimization algorithm [13] are typical. Since then, as algorithms for swarm robots have been proposed in robotics, studies to solve real-world problems are actively conducted. Full-fledged research began in the early 21st century. Typical examples are Sentibots [10] supported by DARPA and Swarm-bots project [8] supported by the EU. Seaswarm [41], which removes oil from the sea surface in a disaster situation, is also a representative example of SR. In recent years, SR has been commonly introduced in various logistics lines and military operations. In addition,

Ars Electronica [19], Intel [21], EHang [14] are actively utilizing swarm robot technology by directing various types of drone shows.

2.2 Gas detection

Gas detector is a device that monitors the presence of gases in an area, often as part of a safety system [27]. This type of equipment is used to detect a gas leak or other emissions and is connected to a control system so that one of its processes can automatically shut down the entire system that leaks the gas. Gas leak detection is the process of identifying potentially hazardous gas leaks by sensors.

Gas sensing is commonly performed with various types of gas sensors. The semiconductor gas sensor [36] detects gas by using the change in density of the surface conduction electrons by chemical interaction between the air component and the semiconductor surface. The catalytic gas sensor [6] uses a catalyst (platinum, palladium, and so on.) sensor. The principle is to measure the increase or decrease of heat generated by catalytic combustion of gas generated on the surface of the catalyst. The thermal conductivity sensor [34] measures the concentration of the gas by using the difference in the heat conduction of the two mixed gases. Non-Dispersive Infrared (NDIR) gas sensor [20] detects gas by using the phenomenon that the radiated infrared rays cause molecular vibration of the target gas, and the infrared rays of a specific wavelength are absorbed. The electrochemical gas sensor [12] detects gas on the principle of converting energy generated by a chemical reaction (redox reaction) into electrical energy.

2.3 Reinforcement learning

Reinforcement learning is one of the learning method based on the Markov Decision Process (MDP) [25]. RL combines the concept of redundancy and the concept of animal psychology [38], as known as trial-and-error. RL constructs a reward function using data derived from the environment and improves it repeatedly to achieve the optimal goal. The whole process of learning is as follows. The agent recognizes the current state based on the data that can be obtained within the defined environment. Then, among the selectable actions, the action or sequence of actions that obtains the greatest reward is selected.

However, the initial reinforcement learning model has a limitation that it is quite difficult to learn about systems with higher complexity compared to simple linear systems [17]. To resolve this problem, Deep Reinforcement Learning (DRL) [29] combining Deep Neural Networks (DNN) [24] have been developed, which enables flexible learning in more diverse situations. Utilizing this DRL algorithm, various algorithms have been developed, such as Deep Q-Networks (DQN) [15], Deep Deterministic Policy Gradient (DDPG) [26], Asynchronous Advantage Actor-Critic (A3C) [2], PPO (Proximal Policy Optimization) [37], and Soft Actor-Critic (SAC) [18]. In this paper, we address the design that includes DNN into swarming system.

3 System design

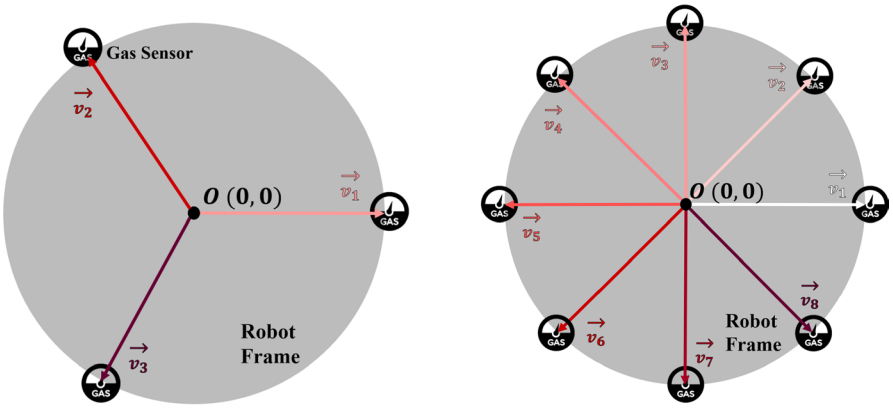
We propose two systems that implement swarm intelligence for gas detection. The first is the system based on vector summation. The swarm control system that moves using the sum of vectors is simple to implement, and each object moves quickly and efficiently, searching the shortest path in real time to its destination, the gas source. The second is the RL-based system. We propose a system in which each object performs more flexible and sensitive movement over episode, by designing proper states and reward that considers a variety of environments. Following subsections describe specific designs for two systems.

3.1 Vector-sum-based control

The vector-sum-based control algorithm allows each robot to select the optimal direction through appropriate sensor placement.

Figure 2 shows the standard models of the system. Figure 2a represents the basic model of vector-sum system. The robot frame forms a circle with a radius of r, and each sensor is attached to the circumference by dividing it into equal parts. The Vector-sum system is not limited to the number of sensors if it has at least three sensors, which is the minimum number required for the system. As such, the rule of the standard model of the vector-sum control system is a model in which N gas sensors divide the frame into N equal parts, and the system shows the maximum performance in standard model. Figure 2b show the models in which the number of sensors is expanded from 3 sensors to 8.

Before the system operates, each robot assumes the center of its frame as a local origin and specifies vectors for its all sensors. These vectors are called the basic vectors. The robot equipped with N sensors has a total of N basic vectors. During system operation, the robot detects gas every cycle and updates sensor values s_1 to



(a) Standard model with 3 sensors. **(b)** Standard model with 8 sensors.

Fig. 2 Sensor configuration and vector selection of vector-sum-based system

s_N. Finally, the robot's period-wise moving vector $\mathbf{M}$ within the vector-sum system can be obtained as follows.

$$\mathbf{M} = \{\Sigma_{k=1}^{N} s_k \cdot \mathbf{v_k} | 3 \leq N\} \tag{1}$$

Sensors in the vector-sum system do not necessarily have to be mounted in specified locations. If only one following requirement is satisfied, the sensors can be freely repositioned. However, in such cases, the performance of the system may be reduced. The vector-sum control system has one essential requirement for optimal path setting. The total sum of the basic vectors must be zero. The placement of the sensors do not necessarily have to be fixed. However, when the sum of the basic vectors is a specific vector, the robot cannot estimate the exact location of the destination and gradually flows in the corresponding direction. The vector-sum-based control algorithm can accurately grasp the shortest route to the gas source destination with at least three gas sensors per robot, and the same result can be obtained even if the number of sensors is increased. In noisy environments, the more sensors mounted, the higher the robot's noise resilience, which results in a route approaching a straight line to the source. Furthermore, it can simply control the speed of robot by multiplying a scalar value by $\mathbf{M}$ derived via Eq. (1).

3.2 RL-based control

Vector-sum-based approach can make simple but accurate decisions for the gas detection robot system, but some limitations are difficult to address. The motivations of introducing RL are listed as follows.

- *Sensor position constaint*: In vector-sum-based approach, the sum of center-to-sensor vectors should be a zero vector, which decreases the flexibility of the robot design. This constraint can be relaxed by designing complex weight of each vector, but it is hard to prove that the obtained weight comprehensively works in every case.
- *Rigidity on applications*: The mathematical model applied on the proposed system only solves the proposed problem. For different sensors or objectives, control system should be revised regarding the characteristics of the targeted cases.

From these reasons, we considered to utilize RL-based approach to solve more complexed problem of our system. The following subsections describe the details of our proposed RL system components.

3.2.1 Algorithm

We adopted *REINFORCE* algorithm for learning mechanism. Recently, deterministic policy gradient algorithms such as DDPG [26], SAC [18] are eagerly considered for control systems, due to the requirements on the discrete action space. However, we claim that deterministic algorithms could be inadequate for the problem we face,

primarily the randomness of the problem. In our problem, a robot is firstly spawned at the randomized position and seeks the direction to the gas source from its sensor measurements. In most of our studies, we observed that deterministic algorithm tends to converge to the local optimum which outputs acceptable results in some part of the cases, which does not consider the diversity of the initial position. Exploration by random noise [32] temporally helps to escape this wrong convergence, but controlling the scale of noise would be labor-intensive and case-dependent approach. Thus, we attempted stochastic approach that opens entire case of the action while narrowing the probability distribution from stored experiences.

3.2.2 Gas tracking

In this section, we address how we determined the state vector used in our algorithm. We measured the actual sensor value of sensor device s with respect to the distance to the gas source. Figure 3 shows the result of the experiment. As shown at the sensor value (black dotted line) and its fitting curve (red dotted line) in the figure, sensor value with respect to the distance can be obtained as

$$s = \alpha\sqrt{\beta - |\mathbf{p_s} - \mathbf{p_{dest}}|} + \gamma \tag{2}$$

where $\mathbf{p_s}$, $\mathbf{p_{dest}}$ refer to the position of the sensor and the destination, respectively, and α, β, and γ refer to the estimation parameter of regression. The error of theoretical distance is 0.4170 m in average. Since square root function is inversible in positive domain, each robot can roughly derive the distance to its sensors. Note that Eq. (2) is an example of the sensor value transformation, and the form of equation depends on the HW model, type, sensing material, and so on.

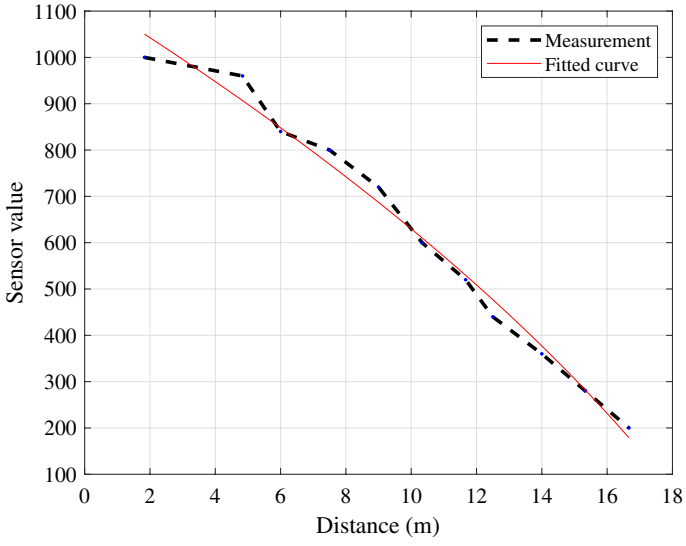


Fig. 3 Gas sensor measurement approximation with respect to distance

It is worthwhile to try to set the state vector as the sensor values, but we found that the algorithm hardly increases the episodic reward in the learning phase. We argue that the cause of the learning failure is a too small differences in the sensor values compared to the sensor values themselves. The sensors output a value approximately between 100 and 1000, and the differences among the sensor values are measured less than 10. Since the differences between the sensor values are the core clues to determine the direction of the robot, we simply normalized the state values by deducting the sensor values from a pre-selected sensor value. By doing so, the state vector has a set of numbers approximately ranged from -10 to 10, where the algorithm can efficiently seize the state changes from the actions.

In summary, we can obtain our desired state value, difference of the distance between the sensors to the source, from the sensor values collected online. In addition, relative positions of the sensors (vector from robot's center to each sensor) are no longer required since the state value implicitly contains this information. The remaining concern about state is the error during transformation. In this case, the noise can be considered during learning, in detail, updating and correcting the probability distribution of the action space. In our evaluation section (Sect. 4), we show that the proposed system learns from the errors and draw out the best decision from noisy measurements.

As aforementioned, we define the discrete action space A as a set of the unit vector, expressed as

$$A = \left\{ \left(\cos\left(\frac{2\pi k}{N_A}\right), \sin\left(\frac{2\pi k}{N_A}\right) \right) | 0 \leq k < N_A, k \in \mathbb{N} \right\} \tag{3}$$

where N_A refers to the dimension of the action space. In our implementation, we set N_A to 36. Then, we model the reward function $r(t)$ as

$$r(t) = \mathbf{a(t)} \cdot u(\mathbf{p_{dest}} - \mathbf{p(t-1)}) \tag{4}$$

where $\mathbf{a(t)} \in A$ refers to the action chosen in the step t, $u(\mathbf{v})$ refers to the unit vector of $\mathbf{v}$, $\mathbf{p_{dest}}$ refers to the position of destination, and $\mathbf{p(t-1)}$ refers to the position of robot at $t-1$. Note that $r(t)$ is close to 1 when $\mathbf{a(t)}$ is close to the direction toward the destination, and close to -1 when $a(t)$ is opposite to the direction toward the destination. We firstly organized the reward formula with the differential of the sensor values s_i, and we found that the sensor values brings inefficient learning curve, which means delayed learning. Since the sensor value increases as it closes to the destination, the experience at the far positions has relatively less effect on the policy network. Thus, we adopted the angular difference of the resulting action for equalized reward. If the robot reaches to the destination, expressed as $|\mathbf{p(t)} - \mathbf{p(dest)}| < \epsilon$, we grant prize reward $r_{success} = 2$ to encourage the positive actions of the network.

3.2.3 Collision avoidance

In addition to the gas tracking, we attempted to add collision avoidance feature for our control system. To decide the movement that avoids collision, the policy network

should secure the related parameter that can warn the risk of collision. Thus, we added ranging sensor to the robot that scans the nearby space and outputs the range to its nearby area, such as Lidar or radar. Note that the sensor can detect whether an obstacle is mobile or stationary, so can be utilized at the swarming and the path finding. Then, we appended two state values: the minimum range value d_{min}, and the degree where the value is obtained θ_{min}. From this, policy network can recognize where the nearest obstacle is and select the alternative direction for avoiding collision.

To grant the feedback of the collision, we applied negative reward $r_{collision} < -1$ in the case of collision. In our implementation, we set $r_{collision} = -2$. Note that too low reward for punishment could lead to the biased weight of the network, which results the wrong or delayed convergence of the policy.

3.2.4 Summary

In summary, we set the state, action, and reward value as following.

$$s(t) = \left(d_{s_1}(t) - d_{s_0}(t), d_{s_2}(t) - d_{s_0}(t), \ldots d_{s_{N_s}}(t) - d_{s_0}(t), d_{min}, \theta_{min} \right), \quad \text{where } d_{s_i} \text{ is obtained by Eq. (2).} \tag{5}$$

$$a(t) \in A, \text{ where A is defined by Eq. (3).} \tag{6}$$

$$r(t) = \begin{cases} r_{success} & |\mathbf{p_{dest}} - \mathbf{p_{robot}}| < \epsilon \\ r_{collision} & \text{Collision occurs} \\ \mathbf{a(t)} \cdot u(\mathbf{p_{dest}} - \mathbf{p(t-1)}) & \text{Otherwise} \end{cases} \tag{7}$$

In Eq. (5), N_s refers to the number of sensors equipped in the robot, and d_{s_i} refers to the distance between the sensor s_i to the destination.

Figure 4 graphically represents the overall structure of the learning system. Since all nodes perform the same action, we use a single policy network applied to all

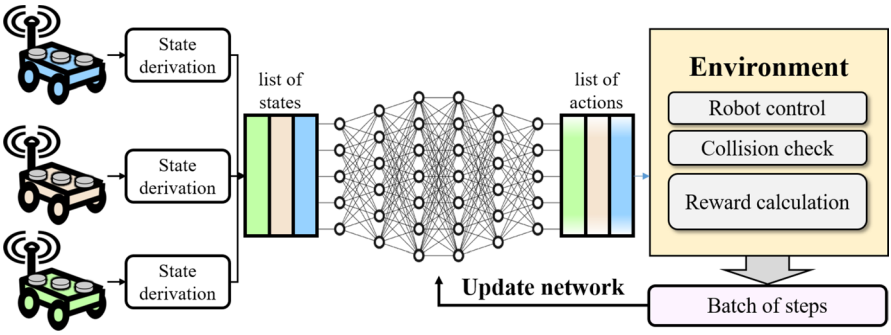


Fig. 4 RL-based gas tracking control design

robots, which matches to the swarming intelligence philosophy. So that, the resulting network with an environment of small number of robots can be applied to the scenario of large number of robots, which is shown in Sect. 4. As shown in the figure, we constructed our learning system based on the pseudo code of REINFORCE algorithm [42]. Algorithm 1 shows schematic pseudo-code of RL-based swarm system. One of main contributions is the formularization of the robot control system which finds the unknown position of the gas source, from one-dimensional sensor values. In addition, we added collision avoidance to the network and made a learning system achieves multiple objectives, with simple state values that the robots can empirically obtain.

Algorithm 1 REINFORCE based Algorithm for Swarming.

Initialize the policy network $\mathcal{P}$
Set learning rate λ
$N_{episode} \leftarrow 0$
Allocate an empty batch
while $N_{episode} < N_{episode,max}$ **do**
  Initialize batch
  Create map and spawn gas source and robots at $\mathbf{p_{dest}}$, $\mathbf{p_{robot,i}}$
  $t \leftarrow 0$
  while True **do**
    Get $a_i(t)$ following policy network
    Advance a step t and get $s_i(t+1)$, $r_i(t)$
    Store $s_i(t), a_i(t), r_i(t)$ to batch
    if Env indicates the end of episode **then**
      break
    end if
    $t \leftarrow t+1$
  end while
  $N_{episode} \leftarrow N_{episode} + 1$
  if mod($N_{episode}$, BATCH_SIZE) $== 0$ **then**
    Calculate policy loss $\mathcal{L} = -\frac{1}{m}\sum_t^T \ln(G_t \pi(a(t)|s(t), w))$
    Update the policy $\mathcal{P} \leftarrow \mathcal{P} + \lambda \nabla \mathcal{L}(w)$
  end if
end while

4 Performance evaluation

In this section, we validate the performance on the two proposed systems by simulation. We first evaluate the impact of the number of sensors mounted on vector-sum systems and subsequently validate RL-based control systems in various environments.

4.1 Number of sensors is vector-sum system

As aforementioned in Sect. 3.1, each robot in vector-sum system must be equipped with at least 3 sensors, and the number of sensors can theoretically increase as much as possible. Therefore, we analyzed the effect of the number of sensors on the system performance in the standard model through simulation. In situations where gas sensor values ideally detect gases without noise, for three or more sensors, the system always presents the robot with the shortest distance from the gas source, that is, movement in a linear direction. However, since this situation is not possible in real-world environments, we added noise to the sensor value and conducted experiments by increasing the number of sensors. Experiments were conducted on 6 standard models of 3, 10, 30, 100, 300, and 500 sensors, and each model has the same initial status except the number of sensors. The experiment environment is as follows.

The coordinates of a gas source are (0 m, 0 m), and each model operates a system at (50 cos(30°) m), 50 sin(30°) m) to explore toward the source. That is, the distance between the gas source and the robot in the initial state is 50 m, and in an ideal case, the robot travels 50 meters in a straight path in one episode and arrives at the gas source accurately. In order to highlight the differences between 6 models, we added a fairly large gaussian noise of $\mathcal{N} \sim (0, 2)$ to each sensor value at every step and calculated the distance the robot has away from the source at the end of one episode. We simulated each of the 6 models for 200 episodes, recording the remaining distance from the gas source.

Figure 5 shows the experimental results. The 3, 10, 30, 100, 300, and 500 sensor models showed an average of 41.48 m, 31.05 m, 4.03 m, 1.09 m, and 0.61 m remaining distances, respectively. Through this, it can be derived that as the number of mounted sensors increases, the exploration to the gas source becomes easier.

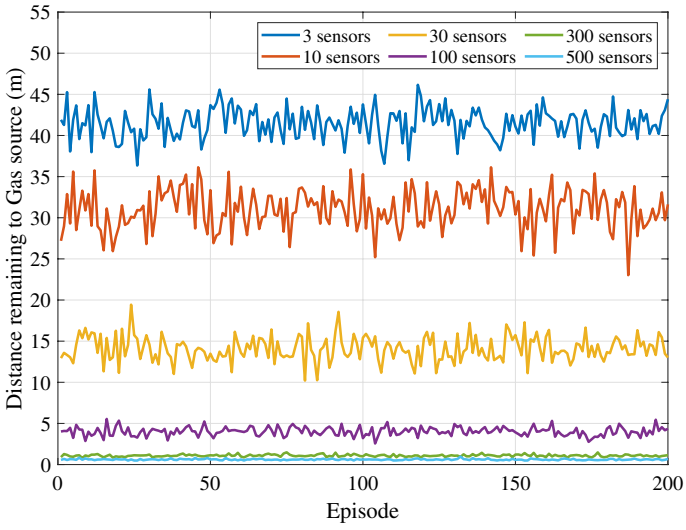


Fig. 5 Distance remaining from gas source according to number of sensors

This is because the more sensing data in the noisy environment, the more accurate results can be inferred. In the standard model, as the number of sensors increases, the angle formed by adjacent sensors decreases, so that the system can provide more precise directions. However, as the number of sensors increases, the model cost also increases, so the user will have to select an appropriate number of sensors that meets their needs.

4.2 RL-based system evaluation

In this section, we decribe how to implement the system and show the evaluation results of our proposed scenario. We conducted the gas tracking scenario in multi-faceted cases, including the aforementioned ones such as noise, sensor formation, and obstacle avoidance. We show the figures to confirm the validity of our system for each case.

4.2.1 Implementation

We implemented each component of our proposed design using Python 3 and `PyTorch` API. The dimension of policy network is $7 \times 16 \times 16 \times 36$. In environment, we created 50m $\times$ 50m two-dimensional map and located a destination point (gas source) at a random position. Then, according to the configuration, we created obstacles and objects into the map. Environment receives the actions derived from the policy network and changes the positions of the objects with designated speed. If the object occurs collision, environment cancels the movement and returns punishment reward for negative feedback. In our system, we set the speed to 1.0 and set the maximum step count to 100 for each episode.

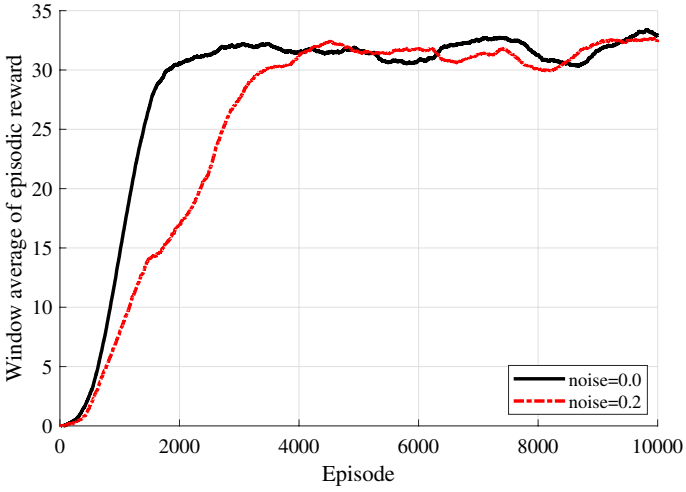


Fig. 6 Episodic reward with respect to the sensor formation

4.2.2 Sensor formation

At first, we varied the formation of the sensors with respect to the object frame. As shown in Fig. 2, vector-sum-based approach forces the symmetric formation of the sensors. However, RL-based system does not require the relative positions of the sensors, so we attempt to run the learning algorithm while randomizing the sensor formation. Figure 6 shows the windowed average of episodic rewards while 10000 episodes. As shown in the figure, it is certain that the uniform sensor formation is advantageous to learn quickly, due to the clear relevance between the sensor distances. However, both average converges to the similar value after about 5000 episodes, which indicates that the randomized sensor formation does not disturb the finest performance of the system, but only does the learning speed.

4.2.3 Sensor noise

Secondly, we added the noise factor considered in Sect. 3.2.2. From the approximation, we found the average noise as 0.4170m. Thus, we added the gaussian noise of $\mathcal{N} \sim (0, \mu_{noise})$, where $\mu_{noise} = \{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$. Figure 7 shows the average reward of each case.

As shown in the figure, as μ_{noise} increases, average reward decreases and less converges in 10000 episodes. However, since the average reward tends to increase during learning, the noise factor is considered in RL system. Because of the randomness of the noise, policy network would reshape the distribution of the action probability to explore multiple choices.

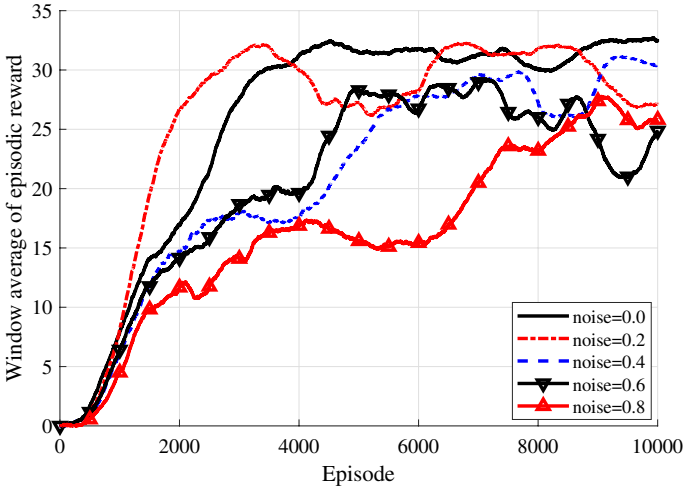


Fig. 7 Episodic reward with respect to the sensor noise

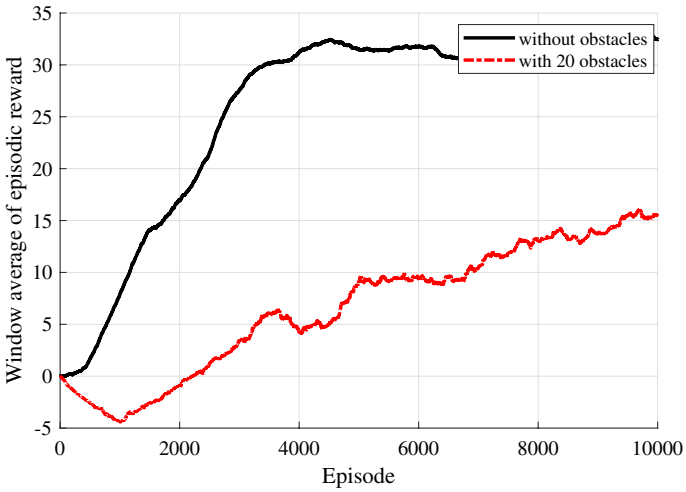
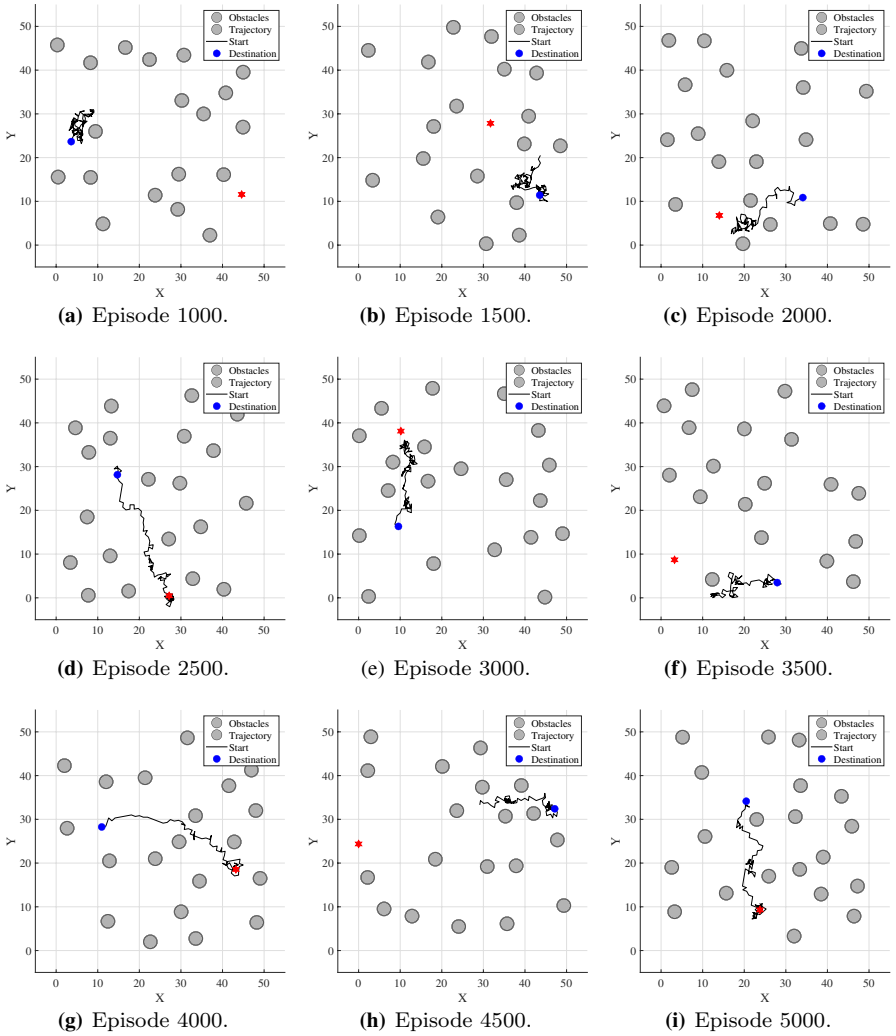


Fig. 8 Episodic reward with respect to the existence of obstacles

4.2.4 Obstacle avoidance

In addition to the gas tracking, we added obstacle avoidance mission by extending the state vector and reward function (Sect. 3.2.3). We located 20 obstacles at the map with radius of 2m, while preventing the total block of the path between the source and the robot starting point. Figure 8 shows the average episodic reward of 20 obstacle case, comparing with the case without obstacle. Since environment returns punishment reward $r_{collision}$ at every collision, overall episodic reward is less than the case without obstacle. However, similar to the case of noise, average episodic reward increases along episodes, which indicates the possibility of the successful obstacle avoidance. To briefly scan the learning trend of this case, we collected the trajectories of the episodes at 0, 2000, 4000, ..., 10000, as shown in Fig. 9. As shown in the figure, the trajectories in earlier episodes show unreached (episode 0,2000) cases, but finds the way after episode 4000. The reason for the increasing episodic rewards is that the robot previously finds the risk of collision by d_{min}, selects the alternative way by θ_{min}, and avoids getting punishment reward for that step.

As discussed in Sect. 3.2.3, robot-to-robot collision avoidance can be also learnt by obstacle avoidance environment. Thus, from the network parameters obtained by the above single robot learning, we ran 20-robot simulation while the robots use the same policy network. Figure 10 shows the overall trajectories and the snapshot of the robots' positions in simulation. As shown in the figure, entire robots gather around the gas source (destination), while keeping the space between the robots. From this experiment, we confirmed that our learning strategy on obstacle avoidance is effective on swarming gas tracking system with existence of obstacles.



(a) Episode 1000. **(b)** Episode 1500. **(c)** Episode 2000.

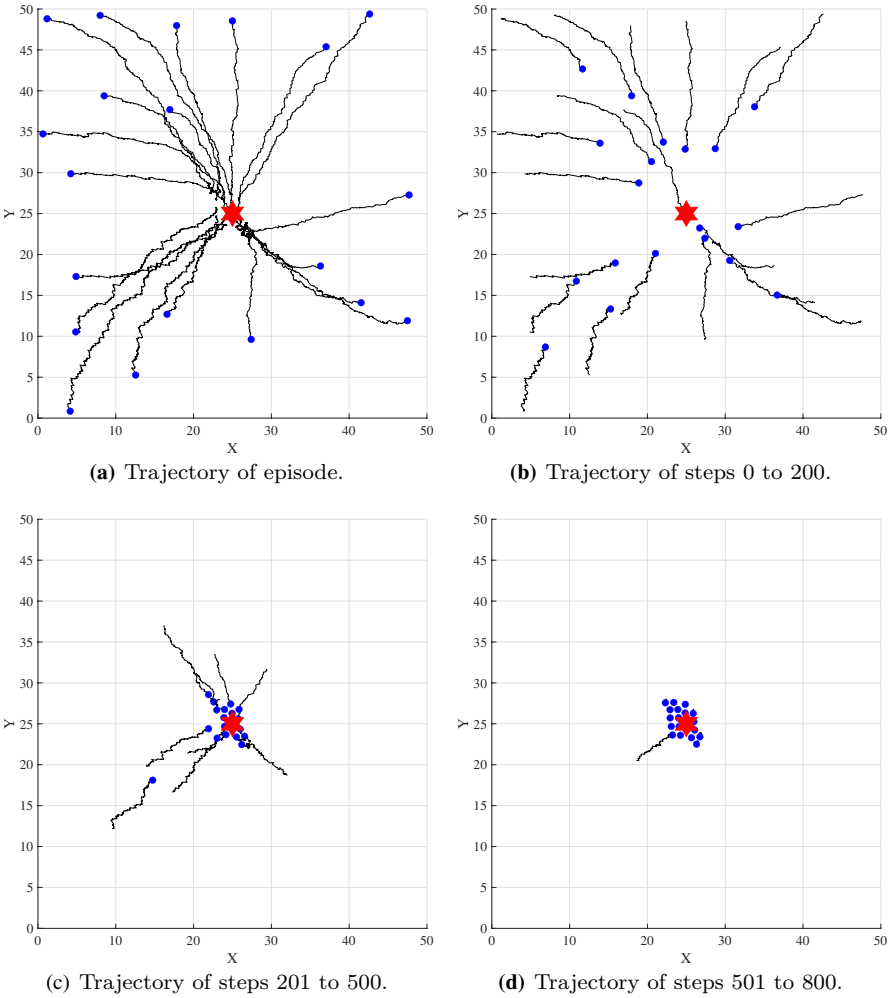
(d) Episode 2500. (e) Episode 3000. **(f)** Episode 3500.

(g) Episode 4000. **(h)** Episode 4500. **(i)** Episode 5000.

Fig. 9 Trajectories of object with 20 obstacles

4.2.5 Comprehensive evaluation

Finally, we dealt with all the aforementioned concerns together and performed a simulation study. We operated 30 robots with 20 obstacles, while equipping randomly attached sensors with $\mu_{noise} = 0.4$. Figure 11 shows the overall trajectories and the step snapshots of the experiment. As shown in the figure, each robot gathers around the gas source while avoiding the collision with obstacles and the other robots. From this evaluation, we showed that our RL-based control system achieves complex objective of multiple agents with noise resilience.



(a) Trajectory of episode.

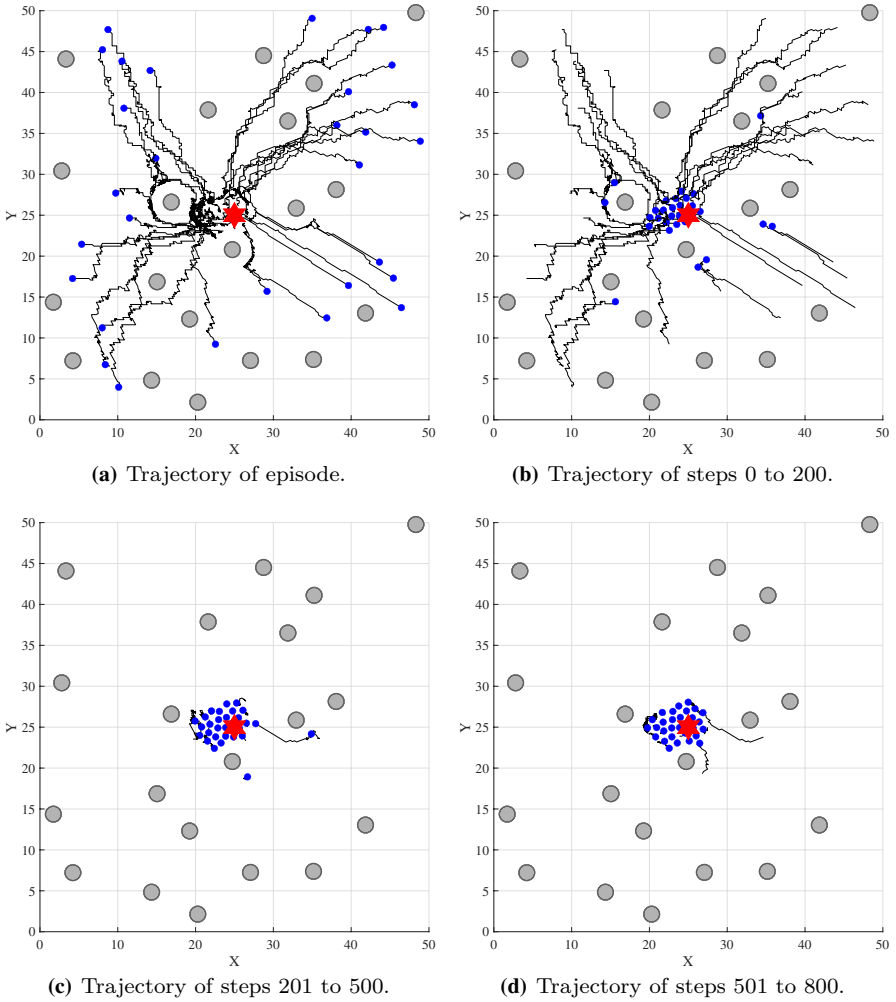
(b) Trajectory of steps 0 to 200.

(c) Trajectory of steps 201 to 500.

(d) Trajectory of steps 501 to 800.

Fig. 10 Trajectories of 20 robots simulation

Additionally, we validate the system performance by constructing various environments up to 30 robots with 40 obstacles. The video shows the position of the robots during the system operation. In all cases, we can see that each robot finds the optimal path and moves appropriately to its destination. It is available on YouTube (https://www.youtube.com/watch?v=p28pxAuExrI).



(a) Trajectory of episode.

(b) Trajectory of steps 0 to 200.

(c) Trajectory of steps 201 to 500.

(d) Trajectory of steps 501 to 800.

Fig. 11 Trajectories of 30 robots simulation with 20 obstacles and $\mu_{noise} = 0.4$

5 Conclusion

In this paper, we proposed RL-based swarming system for gas detection. We approached to implement two swarm robot systems through gas detection in two different ways. First, the vector-sum based control can implement fast movement simply and efficiently. However, this approach is disadvantageous as the number of robots in the swarm increases. This is because the sensor formation is highly constrained, and it is vulnerable to the collisions between robots and obstacles. On the other hand, our proposed RL-based control system is more complicated than the vector-sum approach, but it can adapt to the environment and perform a wider variety of missions. We evaluated the performance of the proposed system

through simulation. We hope our work contribute to the design of the RL-based swarm system.

We have several researches plans as future work. First, we will apply a learning model for multiple gas points. The system will be able to evolve into a model that accurately seeks the highest gas concentration point. Second, if there is no safety issue, we will construct a physical environment similar to the simulation environment to proceed with the empirical experience. By doing so, we will be able to further improve the reliability of the proposed system.

Acknowledgements This research was supported by the Human Resources Program in Energy Technology of the Korea Institute of Energy Technology Evaluation and Planning (KETEP) and the Ministry of Trade, Industry & Energy (MOTIE) of the Republic of Korea (No. 20204010600220) and the National Research Foundation of Korea funded by the Korean Government (grant 2020R1A2C1012389).

References

1. Abraham L, Biju S, Biju F, Jose J, Kalantri R, Rajguru S (2019) Swarm robotics in disaster management. In: 2019 International Conference on Innovative Sustainable Computational Technologies (CISCT). IEEE, pp 1–5
2. Babaeizadeh M, Frosio I, Tyree S, Clemons J, Kautz J (2016) Reinforcement learning through asynchronous advantage actor-critic on a gpu. arXiv preprintarXiv:1611.06256
3. Beni G (1988) The concept of cellular robotic system. In: Proceedings IEEE International Symposium on Intelligent Control 1988. IEEE, pp 57–62
4. Beni G, Wang J (1993) Swarm intelligence in cellular robotic systems. In: Robots and Biological Systems: Towards a New Bionics? Springer, pp 703–712
5. Brambilla M, Ferrante E, Birattari M, Dorigo M (2013) Swarm robotics: a review from the swarm engineering perspective. Swarm Intell 7(1):1–41
6. Cabot A, Dieguez A, Romano-Rodrıguez A, Morante J, Barsan N (2001) Influence of the catalytic introduction procedure on the nano-sno2 gas sensor performances: where and how stay the catalytic atoms? Sensors Actuators B: Chem 79(2–3):98–106
7. Ceylan H, Yasa IC, Kilic U, Hu W, Sitti M (2019) Translational prospects of untethered medical microrobots. Progr Biomed Eng 1(1):012002
8. Clark D (1988) The design philosophy of the darpa internet protocols. In: Symposium Proceedings on Communications Architectures and Protocols, pp 106–114
9. Dayan P (2002) Reinforcement learning. Stevens' Handbook of Experimental Psychology
10. Dickerson JP, Kagan V, Subrahmanian V (2014) Using sentiment to detect bots on twitter: Are humans more opinionated than bots? In: 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014). IEEE, pp 620–627
11. Dorigo M, Maniezzo V, Colorni A (1996) Ant system: optimization by a colony of cooperating agents. IEEE Trans Syst Man Cybern Part B (Cybern) 26(1):29–41
12. Dossi N, Toniolo R, Pizzariello A, Carrilho E, Piccin E, Battiston S, Bontempelli G (2012) An electrochemical gas sensor based on paper supported room temperature ionic liquids. Lab Chip 12(1):153–158
13. Eberhart R, Kennedy J (1995) A new optimizer using particle swarm theory. In: MHS'95. Proceedings of the Sixth International Symposium on Micro Machine and Human Science. IEEE, pp 39–43
14. Ehang egret's 1374 drones dancing over the city wall of xi'an, achieving a guinness world records title. http://www.ehang.com/news/365.html. Accessed 24 May 2019
15. Fan J, Wang Z, Xie Y, Yang Z (2020) A theoretical analysis of deep q-learning. In: Learning for Dynamics and Control. PMLR, pp 486–489
16. Gilpin K, Knaian A, Rus D (2010) Robot pebbles: one centimeter modules for programmable matter through self-disassembly. In: 2010 IEEE international Conference on Robotics and Automation. IEEE, pp 2485–2492
17. Gu S, Holly E, Lillicrap T, Levine S (2016) Deep reinforcement learning for robotic manipulation. arXiv preprintarXiv:1610.00633, 1

18. Haarnoja T, Zhou A, Abbeel P, Levine S (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: International Conference on Machine Learning. PMLR, pp 1861–1870
19. Hörtner H, Gardiner M, Haring R, Lindinger C, Berger F (2012) Spaxels, pixels in space. In: Proceedings of the International Conference on Signal Processing and Multimedia Applications and Wireless Information Networks and Systems. pp 19–24
20. Hwang W-J, Shin K-S, Roh J-H, Lee D-S, Choa S-H (2011) Development of micro-heaters with optimized temperature compensation design for gas sensors. Sensors 11(3):2580–2591
21. Intel drone light shows. https://inteldronelightshows.com/. Accessed 11 July 2020
22. Jung J, Yoo S, La WG, Lee DR, Bae M, Kim H (2018) Avss: airborne video surveillance system. Sensors 18(6):1939
23. Kennedy J (2006) Swarm intelligence. In: Handbook of Nature-Inspired and Innovative Computing. Springer, pp 187–219
24. Larochelle H, Bengio Y, Louradour J, Lamblin P (2009) Exploring strategies for training deep neural networks. J Mach Learn Res 10(1)
25. Levin E, Pieraccini R, Eckert W (1998) Using markov decision process for learning dialogue strategies. In: Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP'98 (Cat. No. 98CH36181), vol 1. IEEE, pp 201–204
26. Lillicrap TP, Hunt JJ, Pritzel A, Heess N, Erez T, Tassa Y, Silver D, Wierstra D (2015) Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971
27. Liu X, Cheng S, Liu H, Hu S, Zhang D, Ning H (2012) A survey on gas sensing technology. Sensors 12(7):9635–9665
28. Mavrovouniotis M, Li C, Yang S (2017) A survey of swarm intelligence for dynamic optimization: algorithms and applications. Swarm Evol Comput 33:1–17
29. Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D, Riedmiller M (2013) Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602
30. Park S, Oh Y, Hong D (2017) Disaster response and recovery from the perspective of robotics. Int J Precis Eng Manuf 18(10):1475–1482
31. Park S, Kim HT, Kim H (2020) Vmcs: elaborating apf-based swarm intelligence for mission-oriented multi-uv control. IEEE Access
32. Plappert M, Houthooft R, Dhariwal P, Sidor S, Chen RY, Chen X, Asfour T, Abbeel P, Andrychowicz M (2017) Parameter space noise for exploration. arXiv preprint arXiv:1706.01905
33. Qin C, Yan Q, He G (2019) Integrated energy systems planning with electricity, heat and gas using particle swarm optimization. Energy 188:116044
34. Ricco A, Martin S, Zipperian T (1985) Surface acoustic wave gas sensor based on film conductivity changes. Sensors Actuators 8(4):319–333
35. Rubenstein M, Shen W-M (2010) Automatic scalable size selection for the shape of a distributed robotic collective. In: 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, pp 508–513
36. Sakai G, Matsunaga N, Shimanoe K, Yamazoe N (2001) Theory of gas-diffusion controlled sensitivity for thin film semiconductor gas sensor. Sensors Actuators B: Chem 80(2):125–131
37. Schulman J, Wolski F, Dhariwal P, Radford A, Klimov O (2017) Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347
38. Sutton RS, Barto AG (1999) Reinforcement learning. J Cogn Neurosci 11(1):126–134
39. Thrun MC, Ultsch A (2021) Swarm intelligence for self-organized clustering. Artif Intell 290:103237
40. Tilley J (2017) Automation, robotics, and the factory of the future. McKinsey. https://www.mckinsey.com/business-functions/operations/our-insights/automation-robotics-and-the-factory-of-the-future
41. Vieira LFM, Lee U, Gerla M (2010) Phero-trail: a bio-inspired location service for mobile underwater sensor networks. IEEE J Selected Areas Commun 28(4):553–563
42. Williams RJ (1992) Simple statistical gradient-following algorithms for connectionist reinforcement learning. Mach learn 8(3–4):229–256