Real-Time Fine-Grained Air Quality Sensing Networks in Smart City: Design, Implementation and Optimization

Zhiwen Hu, *Student Member, IEEE*, Zixuan Bai, *Student Member, IEEE*, Kaigui Bian, *Senior Member, IEEE*, Tao Wang, *Senior Member, IEEE*, and Lingyang Song, *Fellow, IEEE*

Abstract—Driven by the increasingly serious air pollution problem, the monitoring of air quality has gained much attention in both theoretical studies and practical implementations. In this paper, we present the architecture, implementation and optimization of our own air quality sensing system, which provides real-time and fine-grained air quality map of the monitored area. As the major component, the optimization problem of our system is studied in detail. Our objective is to minimize the average joint error of the established real-time air quality map, which involves data inference for the unmeasured data values. A deep Q-learning solution has been proposed for the power control problem to reasonably plan the sensing tasks of the power-limited sensing devices online. A genetic algorithm has been designed for the location selection problem to efficiently find the suitable locations to deploy limited number of sensing devices. The performance of the proposed solutions are evaluated by simulations, showing a significant performance gain when adopting both strategies.

Index Terms—Air quality, power efficiency, reinforcement learning, genetic algorithm

I. INTRODUCTION

Based on a recent report of the World Health Organization [1], air pollution has been proved to be one of the greatest threat to human health, which is responsible for one in eight of deaths each year. In addition to the exhaust emission from industrial production procedures, the daily activities of residents also contribute to the accumulation of air pollutants, such as driving fuel automobiles or incinerating garbages [2]. The degree of air pollution is usually quantitatively described by the air quality index (AQI), which is defined according to the concentrations of some typical air pollutants, including the fine Particulate Matters (e.g., $PM_{2.5}$ and PM_{10}) and other basic chemical substances [3]. The value of AQI will be larger if the concentrations of the air pollutants become higher, indicating a higher risk of people suffering from harmful health effects.

To measure the concentration of a specific air pollutant, available approaches could be either the large professional instruments with high precision or the tiny commercial sensors with low cost [4]. For the consideration of accuracy, the government-owned official meteorological bureaus have deployed authoritative monitoring systems across the country with high costs. Despite the high precision they can achieve, these official systems only have limited numbers of observation stations over a large area and provide measurement results with significant latency [5]. However, recent studies show that the concentrations of air pollutants have the intrinsic characteristics to change from meters to meters, especially for the particulate matters in the urban areas with complicated terrain resulted from densely distributed tall buildings [6], [7]. This indicates that the data provided by official measurements lose their accuracy to represent the air quality at remote locations.

Therefore, it is preferred that large number of low-cost tiny sensing devices are deployed to provide air quality sensing for the regions with complicated terrain [8], [9]. Since the deployment of tiny sensing devices can be dense and the data collection can be frequent, the air quality distribution can be updated with low latency and high resolution [10], [11]. Such a solution creates a promising application of Internet-of-Things (IoT) in smart city [12], where massive data can be collected and analyzed [13]. The citizens are able to benefit from the valuable information provided by the air quality sensing system, by following the suggestions like keeping away from the highly polluted area or deciding the best ventilation system for a building [14].

In this paper, we propose the architecture, implementation and optimization of our own air quality sensing system, which provides real-time and fine-grained air quality map of the monitored area. For the system design, a four-layer architecture is established, including the energy-efficient sensing layer, the high-reliable transmission layer, the full-featured processing layer, and the user-friendly presentation layer. For the implementation, we have deployed this system in Peking University (PKU) for six months and have collected over 100 thousand data values from 30 devices. The terrain of our campus is considered complex enough to represent a typical urban terrain of a large smart city, since green areas, tall buildings and vehicle lanes are all included. For the system optimization, we aim to minimize the error of the real-time and fine-grained air quality map, where the limited number of available sensing devices and the limited capacity of their batteries are the challenges.

As the major part of this paper, the optimization of the IoT air quality sensing network is studied in detail, which is rarely

Z. Hu, Z. Bai, K. Bian, and T. Wang are with the National Engineering Laboratory for Big Data Analysis and Applications, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. L. Song is with the State Key Laboratory of Advanced Optical Communication Systems and Networks, School of Electronics Engineering and Computer Science, Peking University, Beijing, China. (Email: {zhiwen.hu, zixuan.bai, bkg, wangtao, lingyang.song}@pku.edu.cn).

taken into account in related works [4], [8], [15]. Specifically, the necessity of performing optimization is essentially due to the fact that, the IoT air quality sensing devices are deployed without external power supply [16], [17] in order to adapt to the complicated measurement area. Therefore, a sensing device can only perform a limited number of power-consuming actions, such as detecting the concentration of an air pollutant, or uploading data back to the server. To recover a real-time and fine-grained air quality map from the sparse data, a procedure of inference and estimation is required, which can be realized by approaches such as machine learning [18], [19]. The accuracy of inferring the data at unmeasured locations and unmeasured times depends on the spatial-temporal structure of the collected data. For instance, inferring the current air quality based on a measured value from long ago would be questionable [20]. In addition, inferring the air quality at a certain location based on the data from a hardly correlated location is also inaccurate [21], [22]. In order to guarantee the accuracy of the established air quality map, it is necessary to consider the problems of where to deploy the limited number of sensing devices (location selection problem) and when to perform sensing actions (power control problem). These two problems are interdependent, e.g., the location selection could influence the correlation of the sensing data values and therefore influences the optimal power control.

In our work, we model the measurement error and the inference error based on the statistical data from our own system. Our objective is to minimize the joint error of the real-time and fine-grained air quality map, by properly designing the power control and location selection strategies. To be specific, the power control problem is solved by the proposed solution based on deep Q-learning by considering the system as a Markov Decision Process (MDP), which can be deployed online to deal with unexpected weather conditions. The location selection problem is solved by the proposed genetic algorithm, which takes the result of k-means clustering as the initial genetic population and iteratively improves the location selection by widely searching the solution space. Both solutions achieve satisfactory suboptimal outcomes, and the combination of our power control and location selection strategies presents a significant superiority to reduce the average joint error. In addition, these solutions are scalable and therefore able to be implemented in a city-wide huge IoT air quality sensing network.

The main contributions of our work are listed as below:

1) We present our energy-efficient real-time and fine-grained air quality sensing system, which has been deployed in PKU for six months by Spet. 2018.
2) We model the measurement error and inference error in the air quality sensing system based on the collected data.
3) We provide a deep Q-learning solution for the power control problem to reasonably plan the sensing tasks of the power-limited sensing devices online.
4) We design a genetic algorithm for the location selection problem to efficiently find the suitable locations to deploy limited number of sensing devices.

5) The performance of the proposed solutions is evaluated by simulations, showing a significant performance gain when deploying both strategies.

The rest of our paper is organized as follows. Section II provides an overview of the design and implementation of our air quality sensing system. Section III formulates the problem of minimizing the joint error. Section IV discusses the parameters that influence the inference error. Section V presents the deep Q-learning solution for power control. Section VI presents the genetic solution for location selection. Section VII shows the simulation results of the proposed solutions. Finally, we conclude our paper in Section VIII.

II. SYSTEM OVERVIEW

In this section, we first provide a brief overview of the design of our air quality sensing system, and then present some of the representative implementation results, and finally describe the collected data set.
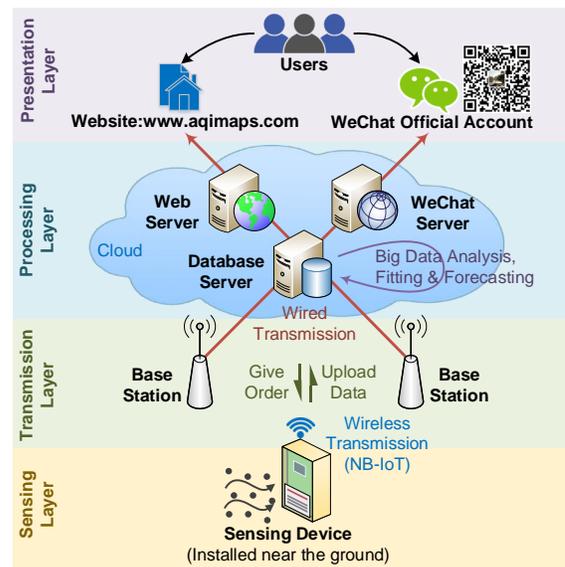

Fig. 1. The architecture of our system, which consists of the sensing layer, the transmission layer, the processing layer and the presentation layer.

A. System Design

As shown in Fig. 1, our air quality sensing system consists of four layers, namely, the sensing layer, the transmission layer, the processing layer and the presentation layer. The sensing layer collects the data of real-time air quality, which is carried out by the sensing devices installed near the ground. The transmission layer enables the bidirectional communications between the sensing layer and the processing layer, which is supported by the infrastructure of the current wireless communication networks. The processing layer is implemented in the cloud server, which is responsible to receive, record and process the data from the sensing layer, and to control the behaviour of the sensing layer. The presentation layer can provide valuable information for the users, which includes our official website and our official WeChat subscription account.
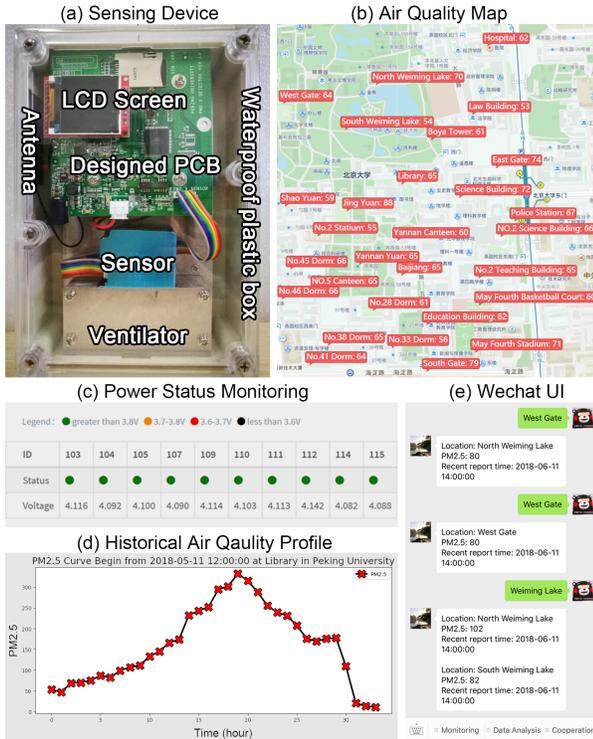
(a) Sensing Device

(b) Air Quality Map

(c) Power Status Monitoring

(d) Historical Air Qaulity Profile

(e) Wechat UI

Fig. 2. A simple exhibition of our system implementation.

B. System Implementation

Fig. 2 shows the implementation of our system, which has been deployed in PKU for 6 months. Most sensing devices are fixed near the ground and powered by batteries. As the data being transmitted back to the server, users can inquire the real-time air quality data on our website [23] or through Wechat official account. The backend of the server also monitors the status of the devices and manage their sensing behaviours to balance between accuracy and battery duration. Spatial inference and short-term prediction can also be supported to guarantee the air quality map to be real-time and fine-grained. More details can be found in [24], which are not presented here, as we focus on the optimization of the air quality sensing network.

C. Data Set Description

During the deployment, we have collected over 100 thousand effective values, mostly for the concentrations of PM2.5. Here we provide the data set collected by 30 on-ground sensing devices [25]. Specifically, it contains the PM2.5 values from two time periods, including the period from March 1st 2018 to May 15th 2018, and the period from June 5th 2018 to Autgest 25th 2018. The provided data set is used to extract some important statistical properties of the monitored area, as given in Section III, in which way we are able to design the corresponding power control and location selection strategies. If the proposed sensing system is expanded to the whole smart city, then the data set of the whole city will be necessary.

III. OPTIMIZATION PROBLEM FORMULATION

In this section, we present the optimization problem in our air quality sensing system. First, we provide the overview of the optimization problem in Section III-A. We then model the measurement error and inference error in Section III-B based on the statistics of our collected data. Finally, we formulate the optimization problem for the air quality sensing system, including power control and location selection.

A. Problem Overview

The air quality sensor and wireless transmission module of each sensing device contribute to most of its power consumption. Therefore, these devices keep themselves in sleep mode during most of the time to save their limited energy supplied by their own batteries. The control server is responsible for planning the sensing tasks for all the devices (i.e., when should each device wake up and collect data), as well as receiving and recording the transmitted data. Since the air quality data from nearby spatial locations and temporal points are not independent, the control server can utilize limited data to establish a real-time air quality map by spatial and temporal inference.

Assume that there are totally K suitable locations for sensing deployment in the concerned area, and only L sensing devices are available to be deployed, where $L < K$. We denote the set of locations with sensing devices as $\mathcal{K}_L$, and the set of locations without sensing devices as $\mathcal{K}_U$. Here, we have $\mathcal{K}_L \bigcup \mathcal{K}_U = \mathcal{K}$ and $\mathcal{K}_L \cap \mathcal{K}_U = \varnothing$.

The sensing system is divided into equal-length time slots, and we should decide whether each device is waken up to collect data at each time slot. We provide the 0-1 matrix $\Phi_{K \times T}$ to represent the power control strategy, where T is the expected number of time slots that the whole system should sustain without recharging. As the element of the matrix Φ, $\phi_{k \times t} = 1$ indicates that the device in the kth location is turned on to sense data at the tth time slot, and $\phi_{k \times t} = 0$ indicates that this device still keeps asleep or there is no device at the kth location. The missing data values are inferred by the server, based on the current and previous collected data, according to their spatial-temporal relation.

B. Measurement Error and Inference Error

In this subsection, we model the measurement error and inference error based on our statistical data [25]. The inference error here is modeled independently of any advanced inference algorithms that based on massive historical data (such as neural networks), in which way we can depict the most general situation. The adopted inference method is based on Gaussian model, which accords with our collected long-term data as testified in Section VII-B. Due to its simplicity and analyzability, this inference method can be considered as a benchmark to depict the "worst inference error" in the most general case. By planning the location selection and power control strategy based on such an inference method, we are actually improving the worst-case performance of the system.
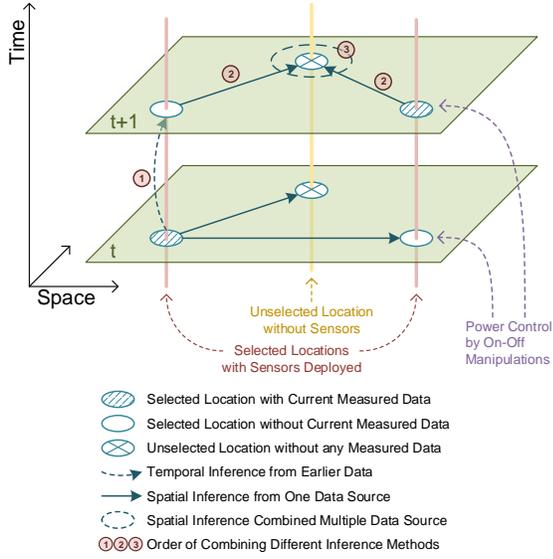
Fig. 3. The spatial-temporal model of the air quality sensing system which contains spatial and temporal inference in order to provide real-time air quality map. Each inference results in different change in the distribution of the estimated data value.

Regardless of whether the data is being directly measured or being inferred from other data, we denote the air quality value at the kth location at the tth time slot as a random variable $X_{k,t}$, where $k \in \mathcal{K}$ and $t = 0, 1, 2, 3, \cdots$. In the following, the deviation of the mean value of $X_{k,t}$ and the uncertainty (variance) of $X_{k,t}$ are considered as the major indicators to represent the error of the measurement or the inference.

Measurement: The measurements of the sensing devices are not perfect, the distribution of the measured value (e.g., PM2.5) at a ceratin location and a certain time approximately complies to Gaussian distribution[1], given by

$$
\begin{cases}
X_{k,t} \sim \mathcal{N}(\mu_{k,t}, \sigma_{k,t}^2), \\
\mu_{k,t} \approx \mu_t, \\
\sigma_{k,t}^2 \approx \mu_{k,t}^2 \times \sigma_0^2,
\end{cases}
\forall t \geq 0, \forall k \in \mathcal{K}, \phi_{k,t} = 1, \quad (1)
$$

where $\mu_{k,t}$ is the precise value of the kth location at the tth time slot[2], μ_t is the average value at the tth time, and σ_0^2 is a constant that reflects the common error of the adopted type of low-cost sensor. And we call σ_0^2 as the normalized measurement variance. We can see that the standard deviation $\sigma_{k,t}$ has a linear correlation with the precise value $\mu_{k,t}$ (or the average value μ_t), which implies that the precision of the measurement decreases as the air quality is getting bad.

Temporal inference: With a measured value $X_{k,t}$ at location k time t, we can infer the possible value at time $t + \tau$ for the same location. As time goes on, the new value of this location deviates from the original one randomly. Such

---

[1]To be more precise, the values complies to truncated Gaussian distribution since the PM2.5 or any other air quality indicators should be ≥ 0, but the small tail below zero can be ignored in most of the cases.

[2]The precise PM2.5 value can be detected by a high-precision calibrating instrument TSI8530, which is expensive and not economical to be massively deployed.

deviation, can be seem as a additive random noise applied on the original measured value. As long as the length of the time slot is fixed, the deviation between two adjacent time slots has a fixed distribution, given as

$$
X_d^{t \to t+1} \sim \mathcal{N}(0, \sigma_d^2), \qquad \forall t \geq 0, \qquad (2)
$$

where σ_d^2 is the constant showing the average change rate of the air quality based on the given length of time slot. We call σ_d^2 as the temporal deviation variance. Therefore the distribution of $X_{k,t+\tau} = X_{k,t} + X_d^{t \to t+1} + \cdots + X_d^{t+\tau-1 \to t+\tau}$ is given by

$$
\begin{cases}
X_{k,t+\tau} \sim \mathcal{N}(\mu_{k,t}, \sigma_{k,t+\tau}^2), \\
\sigma_{k,t+\tau}^2 = \sigma_{k,t}^2 + \tau \sigma_d^2,
\end{cases}
\forall t \geq 0, \forall k \in \mathcal{K}, \phi_{k,t} = 1,
$$
(3)

which implies that the more time span it is, the less accurate the inference will be.

Spacial inference by single source: Based on $X_{k,t}$, no matter it is a directly measured value or the result of a temporal inference, we are able to infer the value at another location at the same time slot, $X_{k',t}$, as shown in Fig. 3. To achieve this, we exploit the relevance among different locations from historical data and find that the deviations among different locations can also be modeled as additive. Specifically, the additive random deviation from location k to location k' is denoted as $X_d^{k \to k'}$, obeying the following distribution:

$$
\begin{cases}
X_d^{k \to k'} \sim \mathcal{N}(\mu_{k,k',t}, \sigma_{k,k',t}^2), \\
\mu_{k,k',t} \approx \mu_t \times \mu_{k,k'}, \\
\sigma_{k,k',t}^2 \approx \mu_t^2 \times \sigma_{k,k'}^2,
\end{cases}
\forall t \geq 0, \forall k, k' \in \mathcal{K} \quad (4)
$$

where μ_t is the average value at time t, $\mu_{k,k'}$ is the constant describing the normalized average deviation from location k to k', and $\sigma_{k,k'}$ is the constant describing the normalized increased variance when using $X_{k,t}$ to infer $X_{k',t}$. Also note that $\sigma_{k,k}^2 = 0$, $\forall k \in \mathcal{K}$. Now we have the distribution of the inferred $X_{k',t} = X_{k,t} + X_d^{k \to k'}$ as:

$$
\begin{cases}
X_{k',t} \sim \mathcal{N}(\mu_{k',t}, \sigma_{k',t}^2), \\
\mu_{k',t} = \mu_{k,t} + \mu_{k,k',t}, \\
\sigma_{k',t}^2 = \sigma_{k,t}^2 + \sigma_{k,k',t}^2,
\end{cases}
\forall t \geq 0, \forall k, k' \in \mathcal{K}. \quad (5)
$$

Note that as μ_t in (4) gets larger (indicating worse air quality), the additional inference variance $\sigma_{k',t}^2$ in (5) gets larger.

Spacial inference by multiple sources: We can further utilize M values from multiple locations to infer an unknown value at a different location at the same time slot t. The utilized values can either be the directly measured values or the inferred values through earlier measurement based on (3). For each of these value, we use (5) to perform a single-source inference for the target location. The mth inference result for the target location k is denoted as $X_{k,t,m} \sim \mathcal{N}(\mu_{k,t,m}, \sigma_{k,t,m}^2)$, where $1 \leq m \leq M$. Then we can multiple all the probability density functions (PDF) of these inference results together to get the PDF of the target location. For simplicity, we assume the distributions of different $X_{k,t,m}$ are independent (since

they can be traced back to different sensors). Therefore the final inference $X_{k,t}$ also has a Gaussian distribution, given as:

$$
\begin{cases}
X_{k,t} \sim \mathcal{N}(\mu_{k,t}, \sigma_{k,t}^2), \\
\mu_{k,t} = \dfrac{\sum_{m=1}^{M} \mu_{k,t,m}/\sigma_{k,t,m}^2}{\sum_{m=1}^{M} 1/\sigma_{k,t,m}^2}, \qquad \forall t \geq 0, \forall k' \in \mathcal{K}. \quad (6) \\
\sigma_{k,t}^2 = \dfrac{1}{\sum_{m=1}^{M} 1/\sigma_{k,t,m}^2},
\end{cases}
$$

where the inference result has a weighted mean based on the mean of these M random variables and has a smaller variance compared with each one of these random variables.

Rule of inference: For a measured value $X_{k,t}$ with $\phi_{k,t} = 1$, no inference is performed. For an unmeasured value $X_{k,t}$ with $\phi_{k,t} = 0$, we consider a three-step inference. The first step is to execute up to L times of temporal inferences for all the selected locations based on their previous measured values, in which way we have L intermediate results for the current time, according to Eqn. (3). The second step is to utilize these intermediate results to perform L times of "single source" spatial inference for the target location, according to Eqn. (5). And the final step is to combine these inferences to form a "multi-source" spatial inference, according to Eqn. (6). Fig. 3 provides a simple illustration of the above inference steps.

C. Environment Model

In the last subsection, we have mentioned μ_t as the average result of the tth time slot. This value can be seen as the air quality for the whole area in a coarse-grained perspective. Without the loss of accuracy, we consider this value is the same as the true average air quality for the whole area. And we aim to establish a statistic model for the change of μ_t.

From our collected data, we find that there is an approximately fixed statistic pattern of μ_t. Specifically, we can calculate how often does a certain level of polluted weather occurs, given by

$$
P[\mu_t = y], \quad t \in [0, T], \quad y \in \mathcal{Y}, \quad (7)
$$

where $\mathcal{Y}$ is the value space of the possible air quality. The values of air quality (such as PM2.5) are usually in the form of integer, thus we consider $\mathcal{Y}$ as a finite discrete value space. In addition, for a fixed length of time slot (such as 10 minutes), the probability of air quality transition between adjacent time slots can also be calculated, given by

$$
P[\mu_t = y \mid \mu_{t-1} = y'], \quad t \in [1, T], \quad y, y' \in \mathcal{Y}, \quad (8)
$$

where the current coarse-grained air quality μ_t has a relation with μ_{t-1}.

It is assumed that μ_t can be roughly known when it comes to the tth time slot. The corresponding approaches could be neural networks [19], or checking the official weather report (which is not our focus in this paper). We focus on how to increase the fine-grained air quality map by power control and location selection, as presenting in the next subsection.

D. Problem of Power Control and Location Selection

The limited capacity of each sensing device confines the number of sensing data it can collect. For simplicity, we assume that the sensing devices have the same battery capacity and each one of them can only perform E times of sensing tasks (including data sensing and an immediate data uploading) before its battery dies, where $E < T$. Therefore, we have $\sum_{t=0}^{T} \phi_{k,t} \leq E, \forall k \in \mathcal{K}_L$, showing the energy budget of the devices. In addition, we expect that each device should not be silent for too long. The maximum number of consecutive time slots that a device can keep asleep is ΔT, which provides $\sum_{t}^{t+\Delta T+1} \phi_{k,t} \geq 1, \forall k \in \mathcal{K}_L, \forall t \in [0, T-\Delta T-1]$. We should guarantee that $\Delta T \cdot E > T$ to avoid contradiction.

Since the server needs to provide a *real-time* distribution of the air quality, the incomplete data at the unmeasured locations should be inferred by the collected data according to the spatial and the temporal inference mentioned in Section III-B. For a given time slot, when the current air quality map is established with the help of inference, we can investigate the accuracy of this map. For $X_{k,t}$, we define its *joint error*, $J_{k,t}$, as the indicator to quantitatively show reliability of the data, which is given as below:

$$
J_{k,t} = \sqrt{\sigma_{k,t}^2 + (\mu_{k,t} - \mu_t)^2}, \quad (9)
$$

which jointly considers the variance of the value and the deviation from the current average value. Specifically, a larger variance or a larger deviation could increase the joint error of the data, i.e., $X_{k,t}$ is less reliable as $J_{k,t}$ gets larger. Note that if $X_{k,t}$ is a measured value, then $\sigma_{k,t}^2 = \mu_t \sigma_0^2$ and $\mu_{t,k} \approx \mu_t$. We consider $\mu_{t,k} - \mu_t = 0$ in this case for simplicity. Otherwise, $\sigma_{k,t}^2$ and $\mu_{t,k}$ should be calculated according to Eqn. (2)~(6) based on the inference rule. Either way, the joint error of each value at the current time slot can be calculated if we have determined the subset of sensing devices being turned on.

At the tth time slot, the average joint error of the current generated real-time air quality map is given by $\sum_{k \in \mathcal{K}} J_{k,t}/K$. And for the whole period including T time slots, the average joint error is calculated as

$$
\bar{J} = T^{-1} K^{-1} \sum_{t=1}^{T} \sum_{k \in \mathcal{K}} J_{k,t}, \quad (10)
$$

where we assume all the sensors should perform a sensing at $t = 0$ for a good initialization and the situation at $t = 0$ is not counted. The objective function of minimizing the average joint error of the real-time air quality map is

$$
\min_{\mathcal{K}_L} \min_{\{\phi_{k,t}\}} \quad \bar{J}, \quad (11)
$$

$$
s.t. \sum_{t=0}^{T} \phi_{k,t} \leq E, \quad \forall k \in \mathcal{K}_L, \quad (12)
$$

$$
\sum_{t}^{t+\Delta T+1} \phi_{k,t} \geq 1, \quad \forall k \in \mathcal{K}_L, \forall t \in [0, T-\Delta T-1], \quad (13)
$$

$$
\phi_{k,t} = 0, 1 \quad \forall k \in \mathcal{K}_L, \forall t \in [0, T], \quad (14)
$$

$$\phi_{k,t} = 0, \quad \forall k \in \mathcal{K}_U, \forall t \in [0, T], \tag{15}$$

$$|\mathcal{K}_L| \leq L, \quad |\mathcal{K}_L| \in \mathcal{K}, \tag{16}$$

$$\mathcal{K}_L \bigcup \mathcal{K}_U = \mathcal{K}, \quad \mathcal{K}_L \cap \mathcal{K}_U = \varnothing, \tag{17}$$

where Eqn. (12)$\sim$(15) show the constraints of power control and Eqn. (16)$\sim$(17) show the constraints of location selection.

IV. THEORETICAL ANALYSIS

In this section, we first take a deeper look into the three-step inference rule and obtain some basic properties of the joint inference in Section IV-A. Then we study the influence of the system parameters on the system performance in Section IV-B. Finally, we discuss some intuitions for the optimization problem in Section IV-C, which leads to the solutions in Section V and Section VI.

A. The Mean and The Variance of The Joint Inference

From the three-step inference rule introduced in Section III-B, we know that each unmeasured value is inferred by L values, which are the most current data that collected by the each one of the sensing devices. We provide Fig. 4 as an example to illustrate such procedure. The final inference is a multi-source spatial inference based on L single-source spatial inferences. And each single-source spatial inference is based on a temporal inference if this location has no current measured value.
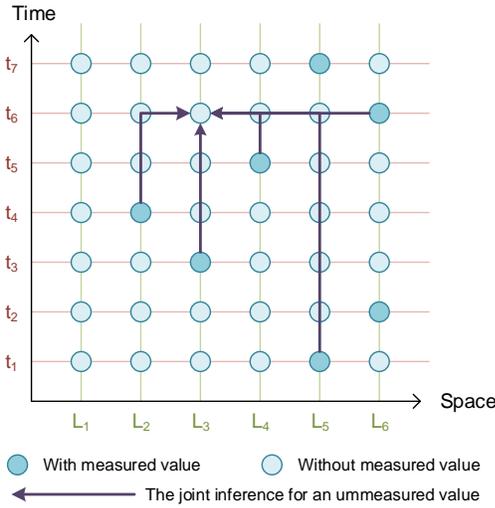

Fig. 4. The joint inference for an unmeasured value in the spatial-temporal graph. The given example, there are six locations and only five sensing device. Each unmeasured value is inferred by the most recent measured values from these five sensing devices.

Now we focus on the inference for a certain location k_0 at a certain time slot t_0. We denote the time span that the kth device has not sense any data until t_0 as $\tau_k \in [0, \Delta T]$. Therefore the kth intermediate inference result after the temporal and the single-source temporal inference for the target location k_0 is given by

$$X_{k_0,t_0,k} \sim \mathcal{N}(\mu_{t_0-\tau_k} + \mu_t\mu_{k,k_0}, \mu_{t_0-\tau_k}^2\sigma_0^2 + \tau_k\sigma_d^2 + \mu_t^2\sigma_{k,k_0}^2), \tag{18}$$

where $\mu_{t_0-\tau_k}^2\sigma_0^2$ is the measurement variance, $\tau_k\sigma_d^2$ is the additional variance of temporal inference, and $\mu_t^2\sigma_{k,k_0}^2$ is the variance of spatial inference based on the relation of k and k_0. Since $\sigma_{k,k_0}^2 = 0$ if the variable $k = k_0$, the above expression is compatible for all situations, such as $(t_3, L_3) \rightarrow (t_6, L_3)$ in Fig. 4.

To combine these L results using a multi-source spatial inference, we use Eqn. (6) to calculate the mean value μ_{k_0,t_0} and the variance σ_{k_0,t_0}^2 of the final result. For the convince of reading, we rewrite the expression of μ_{k_0,t_0} and σ_{k_0,t_0}^2 as below:

$$\mu_{k_0,t_0} = \frac{\sum_{k \in \mathcal{K}_L} \mu_{(k)}/\sigma_{(k)}^2}{\sum_{k \in \mathcal{K}_L} 1/\sigma_{(k)}^2}, \tag{19}$$

$$\sigma_{k_0,t_0}^2 = \frac{1}{\sum_{k \in \mathcal{K}_L} 1/\sigma_{(k)}^2}, \tag{20}$$

where $\mu_{(k)}$ and $\sigma_{(k)}^2$ are short for the mean value and the variance of $X_{k_0,t_0,k}$, respectively, to facilitate reading in the rest of this section.

Remark 1. *From Eqn. (19), we can see that μ_{k_0,t_0} is the weighted sum of $\{\mu_{(k)}|k \in \mathcal{K}_L\}$. The corresponding weight for the kth component is $(\sigma_{(m)}^2)^{-1}$, meaning that a more accurate single-source spatial inference affects more on the final result of the multi-source spatial inference. In addition, we have $\min\{\mu_{(k)}\} < \mu_{k_0,t_0} < \max\{\mu_{(k)}\}$, since*

$$\mu_{min} = \frac{\sum_{k \in \mathcal{K}_L} \frac{\mu_{min}}{\sigma_{(k)}^2}}{\sum_{k \in \mathcal{K}_L} \frac{1}{\sigma_{(k)}^2}} < \frac{\sum_{k \in \mathcal{K}_L} \frac{\mu_{(k)}}{\sigma_{(k)}^2}}{\sum_{k \in \mathcal{K}_L} \frac{1}{\sigma_{(k)}^2}} < \frac{\sum_{k \in \mathcal{K}_L} \frac{\mu_{max}}{\sigma_{(k)}^2}}{\sum_{k \in \mathcal{K}_L} \frac{1}{\sigma_{(k)}^2}} = \mu_{max}, \tag{21}$$

where $\mu_{min} = \min\{\mu_{(k)}\}$ and $\mu_{max} = \max\{\mu_{(k)}\}$.

Remark 2. *From Eqn. (20), we can guarantee that $\sigma_{k_0,t_0}^2 < \min\{\sigma_{(k)}^2\}$, since the following condition holds:*

$$\sigma_{(i)}^2 - \frac{1}{\sum_{k \in \mathcal{K}_L} \frac{1}{\sigma_{(k)}^2}} = \frac{\sum_{k \in \mathcal{K}_L} \frac{\sigma_{(i)}^2}{\sigma_{(k)}^2} - \frac{\sigma_{(i)}^2}{\sigma_{(i)}^2}}{\sum_{k=1}^{L} \frac{1}{\sigma_{(k)}^2}} > 0, \quad \forall i \in \mathcal{K}_L. \tag{22}$$

Remark 3. *The final inference variance σ_{k_0,t_0}^2 is more sensitive to the minimal value of $\{\sigma_{(k)}^2\}$, since we have the following partial derivative:*

$$\partial \left(\sum_{k \in \mathcal{K}_L} \frac{1}{\sigma_{(k)}^2} \right)^{-1} \Big/ \partial(\sigma_{(i)}^2) = \left(\sum_{k \in \mathcal{K}_L} \frac{1}{\sigma_{(k)}^2} \right)^{-2} \cdot (\sigma_{(i)}^2)^{-2}, \tag{23}$$

which means that the same amount of decrease of a smaller $\sigma_{(i)}^2$ will lead to a larger reduce of the final variance.

B. Influence of The System Parameters on The Joint Error

From the expression of the joint error $J_{k,t}$ in Eqn. (9), we can see that both the variance of the inference result $\sigma_{k,t}^2$ and the deviation from the coarse-grained air quality $|\mu_{k,t} - \mu_t|$ contributes to $J_{k,t}$. The increase of $\sigma_{k,t}^2$ and $|\mu_{k,t} - \mu_t|$ could decrease the inference accuracy and lower the confidence level of the established real-time air quality map.

From Eqn. (18) and (20), we can see that the variance of the joint inference depends on the current air quality μ_t, the time span τ_k since the most recent sensing, and the air quality $\mu_{t_0-\tau_k}$ when performing the most recent sensing. This means that the temporal inference from a data long time ago (especially when the value was high back then) is questionable, and the spatial inference on a bad weather condition (high values of air quality) is also inaccurate.

From Eqn. (19) and (20), we can see that the mean of the joint inference is the weighted mean of the corresponding values $\{\mu_{(k)}\}$ from all the L sensing locations. Since $\mu_{(k)}$ is actually the air quality of the $(t-\tau_k)^{th}$ time slot, its difference with the current value μ_t could be large if the air quality changes rapidly in the recent time slots. From our statistical data mentioned in Section III-C, the air quality transition in adjacent time slots presents a greater probability for the similar air quality values, i.e., $P(\mu_t=y|\mu_{t-1}=y')$ is larger if $|y-y'|$ is small. This means that the air quality values in recent time slots is more reliable compared with the values from more previous time slots. Thus $|\mu_{k,t}-\mu_t|$ is expected to be smaller if the sensing devices can turn on more frequently.
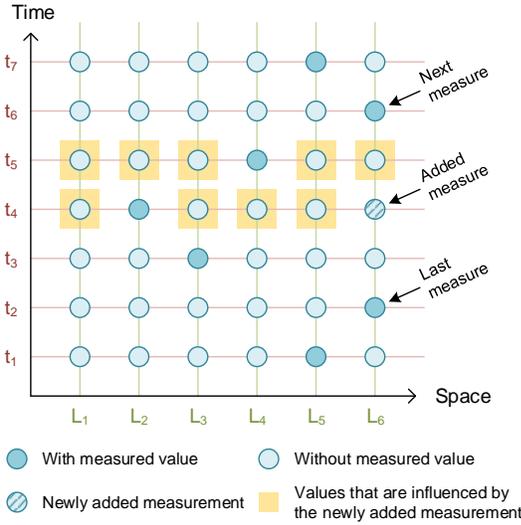

Fig. 5. The influence of adding a new sensing point in the spatial-temporal graph. As we change a node from "unmeasured value" to "measured value", the joint error of some of the nodes are influenced.

Lamma 1. *Adding a measured value in the existing spatial-temporal graph of the air quality sensing system can averagely decrease the joint error.*

Proof: We assume that the added measurement is at the kth location at the t_2^{th} time slot, given by X_{k,t_2}. And we denote the nearest measurement of location k is at the t_1^{th} and the t_3^{th} time slots, with $t_1 < t_2 < t_3$. As illustrated in Fig. 5, the influenced values are within $[t_2, t_3-1]$, where the earlier unmeasured values are inferred based on X_{k,t_1} and the later unmeasured values are inferred based on X_{k,t_3}. For each of these influence unmeasured values, X_{k,t_2} provides a lower variance in the single-source spatial inference compared with

the original value X_{k,t_1} according to Eqn. (18). This is because the value of τ_k is smaller and the probability distribution of μ_{t_2} is the same as μ_{t_1} in the long term average observations. In addition, $|\mu_t-\mu_{t_2}|$ also has a smaller expectation than $|\mu_t-\mu_{t_1}|$ for all $t \in [t_2, t_3-1]$ since the aforementioned property of statistical air quality transition. ∎

Note that the conclusion of Lemma 1 shows the average outcome of the situations. Based on Lemma 1, we can directly obtain the following propositions:

Proposition 1. *Given a fixed time period T, a fixed number of sensing devices $|\mathcal{K}_L|$, two different settings of energy budget $E_1 < E_2$, the corresponding average joint errors comply to $\bar{J}_1 \geq \bar{J}_2$ in the optimal power control strategy.*

Proof: We assume that the best power control strategy of E_1 is $\{\phi_{k,t}\}$, where $\sum_t \phi_{k,t} < E_1$, $\forall k \in \mathcal{K}$. As we raise the energy budget from E_1 to E_2, more values of $\phi_{k,t}$ can be changed from 0 to 1. Based on Lemma 1, adding a new measured value can averagely reduce the average joint error. Even in the worst case where no newly added measurement increases inference accuracy due to extreme weather condition, we can keep $\phi_{k,t}$ as it is and do not deteriorate the original result. ∎

Proposition 2. *Given a fixed time period T, a fixed energy budget E, two different settings of the number of available sensing devices $L_1 < L_2$, the corresponding average joint errors comply to $\bar{J}_1 \geq \bar{J}_2$ in the optimal power control and location selection strategy.*

Proof: We assume the optimal power control and location selection for L_1 devices are $\{\phi_{k,t}\}$ and $\mathcal{K}_L$, respectively. Assume that we add one more device at location k', then its collected data can be used to infer the values at the unselected locations for $t \in [0, T]$, and the values at its own location only for $\{t \mid \phi_{k',t}=0\}$. From Eqn. (20) we know that the variance of the inference decreases since an additional value participates in the multi-source inference. The remaining problem is to figure out how $|\mu_{k,t} - \mu_t|$ of each inferred value changes. A basic idea is to let the newly added device to copy the power scheduling of one of the existing device. According to Remark 1, this is equivalent to the action of adding the weight of the copied device when calculating Eqn. (19). It is expected that some of the $\{|\mu_{k,t} - \mu_t|\}$ will increase and some will decrease. Find the best existing device to copy its power scheduling can averagely achieve positive effect, which will generally reduce $\bar{J}$. Even in the worst case where the newly added device results in a worse $\bar{J}$ due to some extreme settings, we can eliminate the newly added device and keep the original location selection plan, resulting a same $\bar{J}$. ∎

C. Discussions on The Formulated Optimization Problem

For the location selection, intuitively, the devices need to be deployed in those less correlated locations (with high values of $\sigma_{k,k'}^2$ between each other), acquiring "more diversified" data to help re-establish the fine-grained air quality map.

For the power control, the turning-on frequency of the sensing devices should be properly adjusted. A low frequency sensing plan could reduce the accuracy of the real-time air quality map, and a too frequent sensing plan may lead to the the depletion of their batteries long before the last hour T.

It should be noted that, both the measurement and inference error depends on the average air quality (μ_t). This means that we need to know the air quality in advance to make the perfect strategy, which is not a acceptable assumption. We aim to create a more generalized power control strategy which can dynamically deal with the encountered weather condition as long as the statistics of the air quality ($P[\mu_t = y]$ and $P[\mu_t = y|\mu_{t-1} = y']$) is fixed. Therefore, we only assume the current and the previous air quality (μ_t, $t \in [0, t_{now}]$) is known as the system is establishing the air quality map at t_{now}.

In fact, the joint optimization of power control and location selection is highly intractable even with the help of the statistics of historical data. Therefore, in the following part of this paper, we separate problem into the power control problem and the location selection problem. Specifically, we first study the problem of power control in a stochastic environment based on a fixed location selection in Section V. Next, in Section VI, we study the problem of location selection based on a fixed power control strategy in a given environment. By combining the solutions for these two individual problems together, its is expected that a satisfactory outcome can be acquired.

V. POWER CONTROL STRATEGY

In this section, we provide the power control strategy with a fixed location selection $\mathcal{K}_L$. With the knowledge of the environment statistics (as $P[\mu_t = x]$ and $P[\mu_t = x|\mu_{t-1} = x']$), we aim to provide a best power control strategy that is able to deal with the unknown environment having the same statistics. In our context, the power control strategy is learnt by means of reinforcement learning.

However, before formally studying the problem of multiple devices, we first take a look at a simpler situation where only one device is included. Analyzing and solving this simpler problem can help us deal with the case of multiple devices. Specifically, the problem of power control for a single device can be transformed into a Markov Decision Process (MDP), and solved by a dynamic programming algorithm optimally, as provided in Section V-A. Since the complexity of the optimal dynamic programming algorithm increases exponentially with the number of devices, we provide a deep Q-learning solution with approximated value functions for the problem of multiple devices in Section V-B.

A. Power Control for Single Device

In this subsection, we assume that the number of available device is one, i.e., $L = 1$. This means that all the efforts of the power control is concentrated on this single device. In the following, we establish a MDP model with discrete and finite state space, which describes the state transition during the power control procedure.

A MDP consists of five components, namely, the set of states $\mathcal{S}$, the set of available actions $\mathcal{A}$, the state transition probability matrix $\mathcal{P}$, the reward function $\mathcal{R}$, and the discount factor γ. To be specific, the states in $\mathcal{S}$ should obey the Markov property, where each next state only depends on the current state and the adopted action. Assume that the current state is s, one can choose an action a from the action set $\mathcal{A}$ to make the system change. There could be multiple consequent states $\{s'\} \in \mathcal{S}$ after performing a on s, and the corresponding transition probability is given by $\mathcal{P}_{ss'}^a = [S_{i+1} = s' | S_i = s, A_i = a]$, where S_i and A_i represents the ith state and the ith action in the whole history. In addition, there is an reward $\mathcal{R}_s^a$ of performing a on s, representing the immediate utility/gain. The discount factor $\gamma \in [0, 1]$ indicates the fading utility of the future rewards from the viewpoint the current state.

Definition 1. *In the power control problem with a single sensing device, the ith system state in the whole state transition history is defined in the following form:*

$$S_i = (S_i^t, S_i^p, S_i^d, S_i^r, S_i^e), \quad (24)$$

which has five components. The integer $S_i^t \in [0, T+1]$ represents the time of the system ("t" for "time"). The integer $S_i^p \in [0, E]$ indicates the remaining power of the sensing device ("p" for "power"). The integer $S_i^d \in [0, \Delta T]$ shows the number of time slots since the last time of measurement ("d" for "delay"). The integer $S_i^r \in \mathcal{Y}$ records the average air quality value during the last time of measurement ("u" for "record"). And the integer $S_i^e \in \mathcal{Y}$ shows the current average air quality μ_t ("e" for "environment").

Initial state: The initial state is given by $S_1 = (1, E, 1, \mu_0, \mu_1)$, which means that it is the 1th time slot, and there are E available chances of sensing. Note that since we assume all the devices perform a sensing as soon as being deployed at the 0th time slot (which is not counted in the energy budget), the time span since the last sensing is $S_1^d = 1$, and the recorded air quality is $S_1^r = S_0^e = \mu_0$.

Action set: From any given intermediate state, $S_i = (S_i^t, S_i^p, S_i^d, S_i^r, S_i^e), \forall 1 \le i \le T$, two actions can be performed. Specifically, the action set is given by $\mathcal{A} = \{a_0, a_1\}$, where a_0 is to keep the sensing device asleep and a_1 is to turn on the sensing device. If $S_i^p > 0$ and $S_i^d = \Delta T$, then only a_1 is available since ΔT is the maximum time to keep a device asleep. If $S_i^p = 0$, then only a_0 is available since the power has been depleted[3]. Note that although the subscript of each state, i, equals to S_i^t in the single device problem, we distinguish them as two variables for the preparing for the multi-device problem. Fig. 6 shows an example of the actions performed on each state.

---

[3]If $S_i^p = 0$ and $S_i^d = \Delta T$, we only perform a_0 and force S_{i+1}^d to be ΔT instead of $\Delta T + 1$ to limit the number of states being represented. The occurrence of $S_i^p = 0$ and $S_i^d = \Delta T$ represents the violation of the constraint Eqn. (13), indicating that state S_i involves improper power control at previous steps, which will not be included in the optimal solution.
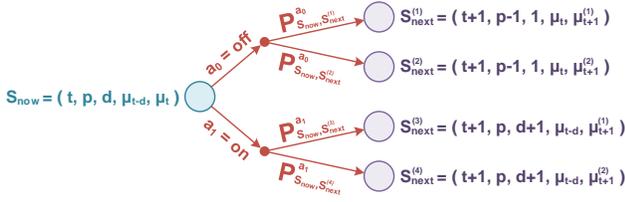
Fig. 6. An illustration of the state transition. Each action may lead to multiple subsequent states since the environment change is random.

Performing "off" action: If we perform a_0 on state S_i, it means that we execute no sensing task at time S_i^t. The overall joint error $\sum_{k \in \mathcal{K}} J_{k,S_i^t}$ at the $(S_i^t)^{th}$ time slot can be calculated according to Eqn. (18) by setting $\mu_t = S_i^e$, $\tau = S_i^d$ and $\mu_{t-\tau} = S_i^r$. We define the reward of taking action a_0 on state S_i as the opposite value of $\sum_{k \in \mathcal{K}} J_{k,S_i^t}$, written as

$$R_{S_i}^{a_0} = -\sum_{k \in \mathcal{K}} J_{k,S_i^t} \Big|_{(S_i^e, S_i^d, S_i^r)}. \tag{25}$$

The following state will be $S_{i+1} = (S_i^t + 1, S_i^p, S_i^d + 1, S_i^r, S_{i+1}^e)$. Note that the first four components are determined, and the last component S_{i+1}^e is generated randomly according to the air quality transition probability. We denote the probability of S_i changing to state S_{i+1} by taking action a_0 as $\mathcal{P}_{S_i, S_{i+1}}^{a_0} = P[\mu_{t+1} = S_{i+1}^e | \mu_t = S_i^e]$.

Performing "on" action: If we perform a_1 on state S_i, it means that we perform a sensing task at time S_i^t. The overall joint error $\sum_{k \in \mathcal{K}} J_{k,S_i^t}$ at the $(S_i^t)^{th}$ time slot can be calculated according to Eqn. (18) by setting $\mu_t = S_i^e$, $\tau = 0$ and $\mu_{t-\tau} = S_i^e$. The corresponding reward is

$$R_{S_i}^{a_1} = -\sum_{k \in \mathcal{K}} J_{k,S_i^t} \Big|_{(S_i^e, 0, S_i^e)}. \tag{26}$$

And the following state will be $S_{i+1} = (S_i^t + 1, S_i^p - 1, 1, S_i^e, S_{i+1}^e)$, where $S_{i+1}^d = 1$ because it has been one time slot since the last time of sensing, $S_{i+1}^r = S_i^e$ indicates the recorded air quality when performing sensing, and S_{i+1}^e also complies to the air quality transition probability based on S_i^e. We denote the probability of S_i changing into S_{i+1} by taking action a_1 as $\mathcal{P}_{S_i, S_{i+1}}^{a_1} = P[\mu_{t+1} = S_{i+1}^e | \mu_t = S_i^e]$.

Termination condition: It can be seen that no matter we use action a_0 or a_1, the component S_i^t increases by one at each time of the state transition. When it comes to $S_i^t = T$, we need to make the last action and the subsequent state will be $(T+1, S_i^p, S_i^d, S_i^r, S_i^e)$, which shows the termination of the state transition.

State-value function: For each state, there is a value function representing the utility of this state, denoted by $V(S)$. Specifically, the termination state $(T+1, S_i^p, S_i^d, S_i^r, S_i^e)$ has zero utility, given by $V(S_{i+1}) = 0$. In each intermediate step, if $S_i \to S_{i+1}$ with reward $R_{S_i}^a$ (with $a = a_0$ or $a = a_1$), then we have

$$V(S_i) = R_{S_i}^a + \gamma V(S_{i+1}), \tag{27}$$

where the discount factor γ is set to 1 in our calculation. It can be seen that the value of S_1 is the sum of the rewards along the path of the experienced states, given by

$$V(S_1) = \sum_{i=1}^{T} R_{S_i}^a = -\sum_{t=1}^{T} \sum_{k \in \mathcal{K}} J_{k,t}, \tag{28}$$

where we can see that maximizing $V(S_1)$ is the same as minimizing the average joint error $\bar{J}$ as the objective function describes.

Action strategy: The problem of maximizing $V(S_1)$ is to find a best path in the state space, which has a size of $T \times E \times \Delta T \times |\mathcal{Y}|^2$. Since the state transition is not fixed due to the random change of air quality, the problem can be interpreted as how to decide the action for each possible state, given by

$$\pi(s) \in \{a_0, a_1\}, \quad \forall s. \tag{29}$$

As proved in [26], there exists an optimal deterministic action strategy for MDP. That is to say, the optimal action strategy $\pi(s)$ for any given state does not need to be a probatilistic one (e.g. with $1/3$ probability choosing a_0 and with $2/3$ probability choosing a_1).

Dynamic programming algorithm: The MDP of the single device problem is highly structured. Each state with S_i^t can only change to another state with $S_i^t + 1$, indicating an unidirectional dependence of the states. Since all the termination states with $S_i^t = T+1$ have zero value, we can iteratively use the values of the state with S_{i+1}^t to calculate the values of the state with S_i^t. Specifically, we have

$$V(S_i) = \max_{a = a_1, a_0} \left[R_{S_i}^a + \sum_{S_{i+1}} \mathcal{P}_{S_i, S_{i+1}}^a V(S_{i+1}) \right], \tag{30}$$

$$\pi(S_i) = arg \max_{a = a_1, a_0} \left[R_{S_i}^a + \sum_{S_{i+1}} \mathcal{P}_{S_i, S_{i+1}}^a V(S_{i+1}) \right], \tag{31}$$

where we should calculate $V(S_{i+1})$ for all possible S_{i+1} before calculating $V(S_i)$. Since each value of $V(S_i)$ considers all the possible subsequent states, $V(S_i)$ can be maximized and the corresponding $\pi(S_i)$ is the optimal choice for the state S_i. At the end of the iteration procedure, we acquire the final optimal strategy $\{\pi(s)\}$ for all the possible states. Therefore, we can use $\{\pi(s)\}$ to deal with the single-device power control problem in an online mode, where the actions can dynamically adapt to the randomly changed environment (μ_t).

Computation complexity: The value of each state is calculated once. And to calculate the value of each state, no more than $|\mathcal{Y}|$ subsequent states are being considered. Therefore, the final computation complexity is $O(T \cdot E \cdot \Delta T \cdot |\mathcal{Y}|^3)$. If the value space of the air quality $|\mathcal{Y}|$ can be approximated into multiple segments, the complexity can greatly reduce. An overview of this solution is presented in Algorithm 1.

B. Power Control for Multiple Devices

For the problem of L devices, the intuition is to define the MDP states by extending the one in (24). Specifically, we have

$$S_i = (S_i^t, \vec{S_i^p}, \vec{S_i^d}, \vec{S_i^r}, S_i^e), \tag{32}$$

Algorithm 1: Optimal single-device power control.

Input: Measurement variance σ_0^2, temporal inference
    variance σ_d^2, spatial inference variance $\{\sigma_{k,k'}^2\}$,
    and air quality transition matrix $P_{|\mathcal{Y}|\times|\mathcal{Y}|}$.
Output: Optimal state-action strategy $\{\pi(s)\}$.
begin
    Initialize $V(s) = 0$ for $s = (T+1, p, u, r, e)$,
    $\forall p \in [0, E], \forall u \in [1, \Delta T], \forall r, e \in \mathcal{Y}$;
    for t *is from* T *to* 1 **do**
        Calculate $V(s)$ and $\pi(s)$ with $s = (t, p, u, r, e)$,
        $\forall p \in [0, E], \forall u \in [1, \Delta T], \forall r, e \in \mathcal{Y}$ according
        to (30) and (31);
    end
end

---

where $\vec{S_i^p}$, $\vec{S_i^d}$, and $\vec{S_i^r}$ are the row vectors with length L, representing for all the L devices. The possible action for each state is also a L-length vector, given by $(a(1), a(2), \cdots, a(L))$, where $a(l) = a_0$ or a_1, $\forall l \in [1, L]$. It is easy to see that the number of states is $TE^L(\Delta T)^L|\mathcal{Y}|^{L+1}$, and the number of actions is 2^L. Therefore, the optimal dynamic programming algorithm is no longer suitable to solve the multi-device power control problem.

Since both the extremely large state space and value space pose challenge for solving the problem, we first aim to transfer the complexity of the value space to the complexity of the state space. This is done by arranging the sensing devices to take actions in a predefined order. In this way, there are only two possible actions (a_0 and a_1) for each state. And the number of states will be multiply by L after such arrangement.

Definition 2. *In the power control problem with L sensing devices, the ith system state in the whole state transition history is defined in the following form:*

$$S_i = (S_i^t, \vec{S_i^p}, \vec{S_i^d}, \vec{S_i^r}, S_i^e, S_i^l), \qquad (33)$$

which has six components. The integer $S_i^t \in [0, T+1]$ represents the time of the system. The L-length integer vector $\vec{S_i^p}$ indicates the remaining power of each sensing device, with $\vec{S_i^p}(l) \in [0, E]$, $\forall 1 \leq l \leq L$. The L-length integer vector $\vec{S_i^d}$ shows the number of time slots since the last time of measurement for each device, with $\vec{S_i^d}(l) \in [0, \Delta T]$, $\forall 1 \leq l \leq L$. The L-length integer vector $\vec{S_i^r}$ records the average air quality value during the last time of measurement for each device, $\vec{S_i^r}(l) \in \mathcal{Y}$, $\forall 1 \leq l \leq L$. The integer $S_i^e \in \mathcal{Y}$ shows the current average air quality μ_t. And the integer $S_i^l \in [1, L]$ implies who's turn it is to take the action at this state.

Initial state: The initial state is given by $S_1 = (1, \vec{S_1^p}, \vec{S_1^d}, \vec{S_1^r}, \mu_1, 1)$, where $\vec{S_1^p}(l) = E$, $\vec{S_1^d}(l) = 1$, $\vec{S_1^r}(l) = \mu_0$, $\forall 1 \leq l \leq L$. Note the last component of S_1 is 1, indicating that it is the turn of the 1st device to take action.

Alternation rule: The first component S_i^t and the last component S_i^l of each state obey the following rule in the

state transition process, regardless of the exact actions being performed. If we have $S_i^l < L$, then $S_{i+1}^l = S_i^l + 1$ and $S_{i+1}^t = S_i^t$, meaning that it is the turn of the next device to decide sensing or not in the same time slot. Otherwise, $S_{i+1}^l = 1$ and $S_{i+1}^t = S_i^t + 1$, indicating that all the devices have done making decisions in the current time slot and the time moves on.

Action set: For each intermediate state $S_i = (S_i^t, \vec{S_i^p}, \vec{S_i^d}, \vec{S_i^r}, S_i^e, S_i^l)$, two actions can be performed, given by $\mathcal{A} = \{a_0, a_1\}$. If $S_i^p(S_i^l) > 0$ and $S_i^d(S_i^l) = \Delta T$, meaning that the $(S_i^l)^{th}$ device has been asleep long enough and still have power to perform sensing, then only action a_1 can be executed. If $S_i^p = 0$, meaning that the $(S_i^l)^{th}$ device has no power, then only a_0 can be executed. For other cases, both a_0 and a_1 can be chosen for the $(S_i^l)^{th}$ device.

State transition for $S_i^l < L$: Assume that the current state is $S_i = (S_i^t, \vec{S_i^p}, \vec{S_i^d}, \vec{S_i^r}, S_i^e, S_i^l)$. If a_0 is performed, then $S_{i+1}^l = S_i^l + 1$, with other components the same as S_i. If a_1 is performed, then $S_{i+1}^l = S_i^l + 1$, $S_{i+1}^p(S_i^l) = S_i^p(S_i^l) - 1$, $S_{i+1}^d(S_i^l) = 0$, $S_{i+1}^r(S_i^l) = S_i^e$, with other components the same as S_i.

State transition for $S_i^l = L$: Assume that the current state is $S_i = (S_i^t, \vec{S_i^p}, \vec{S_i^d}, \vec{S_i^r}, S_i^e, S_i^l)$. If a_0 is performed, then $S_{i+1}^t = S_i^t + 1$, $S_{i+1}^l = 1$, $\vec{S_{i+1}^d} = \vec{S_i^d} + 1$, $S_{i+1}^e \sim P[\mu_t | \mu_{t-1} = S_i^e]$, with other components the same as S_i. If a_1 is performed, then $S_{i+1}^t = S_i^t + 1$, $S_{i+1}^l = 1$, $S_{i+1}^p(L) = S_i^p(L) - 1$, $S_{i+1}^r(L) = S_i^e$, $S_{i+1}^d(l) = S_i^d(l) + 1$ for $1 \leq l \leq L - 1$, $S_{i+1}^d(L) = 1$, $S_{t+1}^e \sim P[\mu_t | \mu_{t-1} = S_i^e]$, with other components the same as S_i.

Reward: Unlike the case for the single-device problem, there are L states for each time slot as we experience through states. It is not reasonable to still use the sum of the joint error $\sum_{k \in \mathcal{K}} J_{k,S_i^t}$ of the current state-action pair as the corresponding reward. This is because only the state in which all the L devices has taken their actions contributes to the final joint error. Therefore, we define the rewards of the state transition within one time slot as their marginal gain of taking actions. Specifically, for the state with $S_i^l = 1$, the reward is defined as $-\sum_{k \in \mathcal{K}} J_{k,t}(1)$, representing the current joint error after the 1th device take its action. For reading convenience, we do not put the complicated expression here. For each $S_i^l > 1$, the reward is defined as the marginal decrease of the joint error in this time slot, given by $\sum_{k \in \mathcal{K}} J_{k,t}(S_i^l - 1) - \sum_{k \in \mathcal{K}} J_{k,t}(S_i^l)$. When it comes to the last state of this time slot, the sum of these L rewards equals to $-\sum_{k \in \mathcal{K}} J_{k,t}(L)$, which corresponds to the value of the joint error after all the L devices take their actions. Furthermore, the total reward in the whole state transition procedure actually equals to the opposite value of the summation of the joint error for all time slots, just like Eqn. (28).

Proposition 3. *The optimal solution for the MDP in (32) and the optimal solution for the MDP in (33) achieve the same performance (the same $\bar{J}$).*

Proof: Both MDP problems can be optimally solved

by the dynamic programming algorithm (regardless of their complexity). In the optimal solution of MDP, $V(s)$ can be maximized for all possible s. And for the initial state S_1, $V(S_1)$ equals to the sum of the rewards along the path (also equals to the opposite value of the overall joint error). Therefore, the optimal solutions for these two MDPs have the same performance, given as $\bar{J} = -V(S_1)/T/K$. ∎

Proposition 3 indicates that the method of arranging the sensing devices to take actions in a predefined order is feasible. However, we should also cope with problem of the large number of states, since it would take too long to figure out $V(s)$ for each state. One possible solution in the reinforcement learning to deal with such a situation is to use an approximation function to estimate the value of each state, or to estimate the value of each state-action pair [27]. Specifically, the value of performing action a on state s is denoted as $Q(s, a)$. The relation of $V(s)$ and $Q(s, a)$ is shown as follows:

$$V(s) = \max_a \big[Q(s, a) \big], \tag{34}$$

$$Q(s, a) = R_s^a + \sum_{s'} \mathcal{P}_{ss'}^a V(s'). \tag{35}$$

By substituting Eqn. (34) into Eqn. (35) and using the mathematical expectation instead of probability matrix, we have

$$Q(s, a) = \mathbb{E}_{s'} \big[R_s^a + \max_{a'} Q(s', a') \big]. \tag{36}$$

Approximation of Q(s, a): We denote the approximated $Q(s, a)$ as $\tilde{Q}(s, a)$, which should be accurate enough to tell the difference of taking different actions at a certain state. The approximation function could be a linear one, e.g., $\tilde{Q}(s, a) = \sum_i w_i \cdot f_i(s, a)$, where $f_i(s, a)$ is the ith feature of the state-action pair, and all the features constitute a feature vector $\vec{f}(s, a)$. However, the linear form of the approximation may have bad fitting performance due to the complexity of the state-action space. Therefore, we use a deep neural network instead, which uses features as input and calculates the corresponding approximated value $\tilde{Q}(s, a)$, also known as deep Q-learning. Here we denote the neural network model as $\tilde{Q}(s, a) = \mathcal{NN}(\vec{f}(s, a))$.

Feature design: Since these features are used to train the expression of $\tilde{Q}(s, a)$, a careful design is necessary. We first define the power deficiency of the device, given by

$$PD(t, p) = 1/\big(1 + \exp^{T/E - (T-t)/p}\big), \tag{37}$$

where $T - t$ and p are the remaining time and the remaining power, respectively. If $(T - t)/p$ is larger than T/E, then $0.5 < PD < 1$, indicating the power is not sufficient compared with the remaining time. Otherwise $0 < PR < 0.5$, showing the power is relatively sufficient compared with the remaining time. A higher PD indicates the device is less willing to turn on and may relay on the measurement of other devices. In our implementation, the length of the feature vector is $5L + K + 3$, where L is the number of devices and K is the number of all the locations.

Feature specifics: To facilitate explanation, we consider the feature vector as five segments. The first segment has L components as 0-1 values, with the $(S_i^l)^{th}$ location set as 1 and all other locations set as 0. This is to indicate the specific device that is responsible to make an action in this state. The length of the second segment is L, which contains the power state vector $\vec{S}_i^p$, showing the remaining power of each device. The third segment has $L + K + 1$ components, they are set to be zeros if $a = a_0$, otherwise, they show the marginal utility of taking action a_1. Specifically, in the third segment, we have K components showing the decreases of the joint error of all these K location if performing a_1, we have additional L components showing the weighted (multiplied by PD) decrease of the joint error for these L devices, and another constant component 1 just to indicate the action of a_1. The fourth segment has $2L + 1$ components, they are set to be zeros if $a = a_1$, otherwise, they show the potential utility of keeping asleep and waiting for the subsequent devices ($l > S_i^l$) turning on. Specifically, in the forth segment, we have L components showing the decrease of joint error if the subsequent devices turn on at this time slot, we have additional L components showing the weighted (multiplied by PD) decrease of joint error if the subsequent devices turn on, and another constant component 1 just to indicate the action of a_0. At last, the fifth segment only includes the remaining time of the current state, $T - S_i^t$. Each value is normalized to one before training $\mathcal{NN}$.

Algorithm 2: Deep Q-learning for value approximation in multi-device power control.

Input: Feature vector for any state-action pair $\vec{f}(s, a)$.
Output: Action strategy $\{\pi(s) = \arg\max_a \tilde{Q}(s, a)\}$.
begin
  Use a random strategy to generate a training set $\{\langle \vec{f}(s, a), Q(s, a) \rangle\}$ to generate an initial $\mathcal{NN}$;
  Initialize a training data set $\mathcal{D}$;
  for *episode* = 1 *to* N **do**
    Randomly generate an initial state S_1;
    while *State transition is not complete* **do**
      The current state is S_i;
      With probability ϵ choose a random action;
      With $1 - \epsilon$ select $a = \arg\max_a \tilde{Q}(S_i, a)$;
      Perform the action, take the reward $R_{S_i}^a$, and the new state is S_{i+1};
      if S_{i+1} *is the termination state* **then**
        Add $\langle \vec{f}(S_i, a), R_{S_i}^a + \max_{a'} \tilde{Q}(S_{i+1}, a') \rangle$ into $\mathcal{D}$;
      else
        Add $\langle \vec{f}(S_i, a), R_{S_i}^a \rangle$ into $\mathcal{D}$;
      end
    end
    Randomly select a batch of B training data from $\mathcal{D}$ to perform gradient descent to improve the $\mathcal{NN}$.
  end
end

Training of the NN: Massive amount of training data

should be provided to obtain an accurate approximation. We first use a random power control strategy to experience through the state transition procedure. In this way, a set of $\langle \vec{f}(s,a), Q(s,a) \rangle$ values can be collected. Then we perform a training process to minimize $\sum \left[\mathcal{NN}(\vec{f}(s,a)) - Q(s,a) \right]^2$ and get the initial training $\mathcal{NN}$ model. Next, we use the $\mathcal{NN}$ model to perform action decision in a new system, with probability $0 < 1 - \epsilon < 1$ to select $a = \arg\max \tilde{Q}(s,a)$ for each state (and with ϵ to randomly select). In this way, we can create a new episode of the experience and record the tuples of $< \vec{f}(s,a), R_s^a, \tilde{Q}(s',a') >$ along the experienced states. As shown in Eqn. (36), $\sum [\tilde{Q}(s,a) - R_s^a - \tilde{Q}(s',a')]^2$ can also be considered as a new minimizing target to improve the current $\mathcal{NN}$. Therefore, the recorded tuples form a new training set $\{\langle \vec{f}(s,a), R_s^a + \tilde{Q}(s',a') \rangle\}$. We can repeatedly use the $\mathcal{NN}$ to perform action decision in new systems and create multiple sets of training data. To avoid the situation where the $\mathcal{NN}$ may overfit with the newest training data set, we randomly choose a batch of the training data from all the previous training data set and perform limited numbers of gradient descent, just like the way in [28], which was called as deep Q-learning with experience replay. An overview of the training procedure is given in Algorithm 2.

VI. LOCATION SELECTION STRATEGY

In this section, we study the problem of location selection based on a fixed power control strategy for L available sensing devices. Note that the power control $\{\phi_{k,t}\}$ might dynamically change if the environment $\{\mu_t \,|\, t = 0, 1, \cdots T\}$ was not fixed. Therefore, the environment is also considered as a given condition, in which way we can analyze the performance of different location selection schemes. To ensure that the location selection is not affected by the specific environment setting, T should be set large enough to integrate all possible stochastic situations based on Eqn. (8).

The rest of this section describes the solution of selecting L sensing locations from K locations. To be specific, we first model the correlation of different locations and use a clustering algorithm to create initial sets of L locations. Then a genetic algorithm is designed to widely search the solution space in order to acquire the best location selection.

A. Correlation of Locations

As introduced in Section III-B, μ_{k_1,k_2} and σ_{k_1,k_2}^2 are used to quantitatively describe the statistic relation of the locations k_1 and k_2. If we use the value at the location k_1 to infer the value at the location k_2 when the average air quality is μ_t, then the deviation of the mean value, $\mu_t \cdot \mu_{k_1,k_2}$, and the additional variance, $\mu_t^2 \cdot \sigma_{k_1,k_2}^2$, should be counted. A higher value of $|\mu_{k_1,k_2}|$ or a higher value of σ_{k_1,k_2}^2 indicates that the air quality at k_1 and the air quality at k_2 have lower similarity, i.e., greater difference.

Calculation of μ_{k_1,k_2} and σ_{k_1,k_2}^2: These two parameters are calculated from the collected data set. We denoted the

collected data from location k_1 and location k_2 as $y(k_1,t)$ and $y(k_2,t)$, with $t = 1, 2, \cdots, T$. Therefore, we have:

$$\mu_{k_1,k_2} = \frac{1}{T} \sum_{t=1}^{T} \frac{y(k_2,t) - y(k_2,t)}{\mu_t}, \tag{38}$$

$$\sigma_{k_1,k_2}^2 = \frac{1}{T} \sum_{t=1}^{T} \frac{\left[y(k_1,t) + \mu_t \cdot \mu_{k_1,k_2} - y(k_2,t) \right]^2}{\mu_t^2}, \tag{39}$$

where μ_t, the average air quality value, is acquired based on the data collected from all the locations.

Difference Matrix: For any two locations, $k_1, k_2 \in \mathcal{K}$, we defined θ_{k_1,k_2} as their difference, which is given by

$$\theta_{k_1,k_2} = \sqrt{\mu_{k_1,k_2}^2 + \sigma_{k_1,k_2}^2}, \quad k_1, k_2 \in \mathcal{K}, \tag{40}$$

which has a similar form with the definition of the joint error in Eqn. (9). We put θ_{k_1,k_2} in the matrix form, given as below:

$$\Theta_{K \times K} = \begin{bmatrix} \theta_{1,1} & \theta_{1,2} & \cdots & \theta_{1,K} \\ \theta_{2,1} & \theta_{2,2} & \cdots & \theta_{2,K} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{K,1} & \theta_{K,2} & \cdots & \theta_{K,K} \end{bmatrix}. \tag{41}$$

Remark 4. *The difference matrix defined by Eqn. (41) is a symmetric matrix with zeros diagonal elements. The diagonal elements are zeros because $\mu_{k,k} = 0$ and $\sigma_{k,k}^2 = 0$. The matrix is symmetric because $\mu_{k_1,k_2} = -\mu_{k_2,k_1}$ and $\sigma_{k_1,k_2}^2 = \sigma_{k_2,k_1}^2$, which can be directly deduced from Eqn. (38) and Eqn. (39).*

High dimensional feature space: The difference of the locations k_1 and k_2, θ_{k_1,k_2}, can be considered as the distance of them in a high dimensional space. This space depicts the implicit features of each location $k \in \mathcal{K}$. A greater difference (or distance) indicates that the corresponding two locations have less similarity. Note that to avoid the case like $\theta_{1,3} > \theta_{1,2} + \theta_{2,3}$ (which is not acceptable in Euclidean space), we can add a same small amount value $\delta = \theta_{1,3} - \theta_{1,2} - \theta_{2,3}$ to all the values of $\theta_{k_1,k_2}, \forall k_1 \neq k_2$. In the following, we omit the expression of δ and consider $\Theta_{K \times K}$ as a valid distance matrix. We then need to decide the coordinate of each $k \in \mathcal{K}$, denoted by $\vec{x}_k = (x_k^{(1)}, x_k^{(2)}, \cdots, x_k^{(D)})$, where D is the dimension of the feature space. Since the difference matrix $\theta_{K \times K}$ does not explicitly express these coordinates, we provide the following method to acquire them.

Calculating the coordinates: First, the dimension D is set to be $D = K - 1$. The coordinate of the first location is set to be zero, given by $\vec{x}_1 = (0, 0, \cdots, 0)$. The coordinate of the second location is $\vec{x}_2 = (\theta_{1,2}, 0, \cdots, 0)$. For the kth coordinate $\vec{x}_k$, we need to calculate the solution of a $(k-1)$-variable quadratic equation set, given by

$$\begin{cases} \left(x_1^{(1)} - x_k^{(1)} \right)^2 + \cdots + \left(x_1^{(k-1)} - x_k^{(k-1)} \right)^2 = \theta_{1,k}^2 \\ \cdots \quad\quad\quad\quad\quad\quad \cdots \\ \cdots \quad\quad\quad\quad\quad\quad \cdots \\ \left(x_{k-1}^{(1)} - x_k^{(1)} \right)^2 + \cdots + \left(x_{k-1}^{(k-1)} - x_k^{(k-1)} \right)^2 = \theta_{k-1,k}^2 \end{cases} \tag{42}$$

where $x_k^{(1)}, \cdots, x_k^{(k-1)}$ are the variables to be solved. In case of multiple solutions, any one can be acceptable since we only need to satisfy the difference matrix $\Theta_{K \times K}$ as the constraint.

Clustering: For any two of the locations, $k_1, k_2 \in \mathcal{K}$, if they have short distance in the feature space, then the joint error of using one to infer another will be low. Otherwise, the joint error will be relatively greater. From this intuition, the L locations that we aim to select from $\mathcal{K}$ should be "as separated as possible". Therefore, we use the classical k-means algorithm [29] to cluster the K locations in the feature space into L clusters. The major weakness of the k-means algorithm is the necessity of predefining the value of k, which in fact becomes an advantage in our usage since this parameter happens to be fixed as L.

Initial sets: With L clusters, we can randomly choose one location from each cluster to create a location set set $\mathcal{L}$, with $|\mathcal{L}| = L$. Since there could be many possible $\mathcal{L}$, we denote the collection of the location sets as

$$
\begin{cases}
\mathcal{C} = \{\mathcal{L}_c,\}, & \forall c = 1, 2, \cdots C, \\
\mathcal{L}_c = \{k_l\}, & \forall k_l \in \mathcal{K}, \forall l = 1, 2, \cdots L,
\end{cases}
\tag{43}
$$

where we confine the size of collection as C. The collection $\mathcal{C}$ therefore becomes the initial choices of the location selection, which can be used in the following genetic algorithm.

B. Exploring the Solution Space by Genetic Algorithm

The reason we use genetic algorithm is because the solution space of this 0-1 integer optimization problem is highly complicated. Any approximated solution could fall into a minimum value with poor performance. The genetic algorithm can widely search the solution space and provide a satisfying outcome [30]. In the following, we introduce our implemented genetic algorithm, including genetic coding, genetic recombination, genetic mutation, genetic selection and the termination criteria.

Genetic coding: For each location set $\mathcal{L}$, we use a K-length vector to indicates the corresponding coded gene. A gene of the location set $\mathcal{L}$ is denoted as

$$
\vec{G}(\mathcal{L}) = \left(G^{(1)}(\mathcal{L}), G^{(2)}(\mathcal{L}), \cdots, G^{(K)}(\mathcal{L})\right),
\tag{44}
$$

where $G^{(k)}(\mathcal{L})$ is a boolean function, indicating whether the kth location is included in the location set $\mathcal{L}$. For each valid location set (i.e., up to L selected locations), the number of "1"s in its coded gene is no more than L. For the initial collection of location sets $\mathcal{C}$ introduced in Section VI-A, we encode them and add their genes to a gene pool, denoted by $\mathcal{G}$.

Genetic mutation: The mutation of a gene is to create the possibility to randomly try another solution. Each gene creates M copies of itself, where each bit of the gene has a fixed possibility p_m to mutate during the copy ($0 \rightarrow 1$, or $1 \rightarrow 0$). Therefore, different genes (solutions) are generated and added to the gene pool $\mathcal{G}$. Note that the number of 1 may not satisfy the constraint, which will be considered in the following genetic selection part.

Genetic recombination: For any two existing genes, a recombination procedure can be performed. To be specific, for $\vec{G}_1$ and $\vec{G}_2$, we randomly choose a position $g = 1, 2, \cdots K-1$ and exchange the bits of these two genes behind the position g. The new genes are $\vec{G}_1' = \left(G_1^{(1)}, \cdots, G_1^{(g)}, G_2^{(g+1)}, \cdots, G_2^{(K)}\right)$ and $\vec{G}_2' = \left(G_2^{(1)}, \cdots, G_2^{(g)}, G_1^{(g+1)}, \cdots, G_1^{(K)}\right)$. These new genes are also added into the gene pool $\mathcal{G}$.

Coding: Each bit represents whether the location is selected

Mutation: Each bit mutates independently

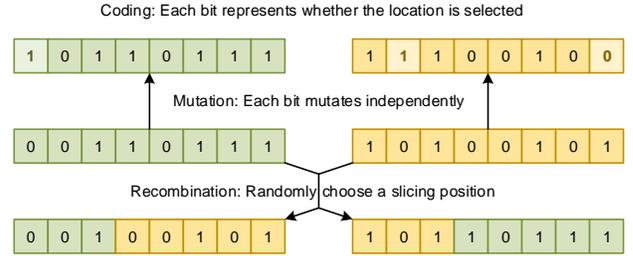Recombination: Randomly choose a slicing position

Fig. 7. An example of genetic coding, mutation and recombination.

Genetic selection: The size of the gene pool $\mathcal{G}$ is not unlimited. As the new genes generated by mutation and recombination are added into the gene pool, there could be too many genes. A selection procedure should be performed to choose the good ones and eliminate the bad ones. First, the duplicated genes and the genes with more than L "1"s are eliminated. Then we check the disadvantage of each existing gene, which is defined as $\bar{J}$ of the corresponding location set based on the given power control and environment. Note that the aforementioned power control matrix $\Phi_{K \times T}$ has $L < K$ non-zero vectors, which indicates the power control for the given L locations. Here, we need to interpret $\Phi_{K \times T}$ as $\Phi_{L \times T}$, with L power control schemes for each location in the location set. If the number of "1"s in a gene is fewer than $L' < L$, then we only choose first L' vectors as the corresponding power control schemes. With the disadvantage of each existing gene, we keep the best H_1 ones in the gene pool, and then randomly choose H_2 genes by using the disadvantage of each gene as its weighted probability. Such probability is given by

$$
p_g \propto \left[\left(\max_g \bar{J}\right) - \bar{J}_g\right], \qquad \forall g \in \mathcal{G}.
\tag{45}
$$

Therefore, the gene pool only keeps $H = H_1 + H_2$ genes after each round of genetic selection.

Evolution and termination: For each round of genetic evolution, based on the current gene pool, we perform the genetic mutation for each existing gene and the genetic recombination between randomly chosen gene pairs. As new genes are added, we then perform genetic selection to keep only H genes and record the lowest value of disadvantage. If the best performance hasn't been improved for 6 rounds, or it has been more than W rounds, then the evolution process terminates. Otherwise, we repeat the mutation and recombination to continue the genetic evolution. An overview of the genetic algorithm is provided in Algorithm 3.

VII. EMULATION

In this section, we evaluate the performance of the proposed power control strategy and the proposed location selection

Algorithm 3: Genetic Algorithm for Location Selection.

Input: The power control strategy $\{\phi_{k,t}\}$ and the environment condition $\{\mu_t\}$.

Output: Optimal location set $\{L\}$.

begin

    Calculate the coordinates of the locations in the feature space according to (42);

    Use L-means algorithm to perform clustering and acquire L clusters;

    Randomly choose one location in each one of the cluster to create a location set;

    Generate C location sets and put their coded genes into the gene pool $\mathcal{G}$;

    for *Evolution count $w = 1$ to W* **do**

        Perform genetic mutation and recombination;

        Eliminate the duplicated genes and the genes with too many "1"s;

        Calculate the disadvantage of each existing gene;

        Select the best H_1 genes and randomly select H_2 genes according to Eqn. (45);

        if *no improvement for 6 iterations* **then**

            Break the evolution;

        end

    end

    Choose the gene with the lowest disadvantage;

end

---

input side to the output side). Each round of gradient descend of the neural network is realized by running the "trainscg" algorithm provided in MATLAB for one epoch on the training data set.

A more detailed parameter setting is listed in Table I.

Normalized measurement variance, σ_0^2	0.0037
Temporal deviation variance, σ_d^2	10.89
Normalized mean relation, $\{\mu_{k,k'}\|k\neq k'\}$	between -0.15 and 0.15
Normalized variance relation, $\{\sigma_{k,k'}^2\|k\neq k'\}$	between 0.001 and 0.1
Air quality values, $\{\mu_t\}$	between 1 and 508
Total number of time slots, T	between 500 and 10000
Total number of locations, K	30
Number of available devices, L	between 5 and 25
Energy budget of each device, E	between 100 and 1000
Maximum allowable sleep time slots, ΔT	between 10 and 12
Episode number to train $\mathcal{NN}$, N	between 1 and 100
Batch size to improve $\mathcal{NN}$ in each round, B	100000
Random action probability, ϵ	from 0.1 to 0
Maximum number of rounds of evolution, W	25
Size of gene pool, H and C $(H = C)$	between 40 and 100
Two types of genetic selection, H_1 and H_2,	$0.1H$ and $0.9H$
Copy number during genetic mutation, M	3
Mutation probability for each bit, p_m	0.1

strategy. Simulation setups are given in Section VII-A, simulation results and corresponding discussions are provided in Section VII-B.

A. Data Set Usage and Parameters setup

Some of the parameters are extracted from our collected data, which is based on 30 air quality sensing devices deployed in Peking University for around seven months [25]. The sensing interval (the length of the time slot) is 10 minutes, and each collected data is an integer representing the detected PM2.5 value at the given location and time.

We denote the measured value at the kth location at time t as $y(k,t)$. Therefore, the normalized measurement variance σ_0^2 and the temporal deviation variance σ_d^2 is given by:

$$\sigma_0^2 = \frac{1}{T}\sum_{t=1}^{T}\frac{1}{L}\sum_{k\in\mathcal{K}_L}\frac{[y(k,t)-\mu_t]^2}{\mu_t^2}, \tag{46}$$

$$\sigma_d^2 = \frac{1}{T-1}\sum_{t=1}^{T-1}\frac{1}{L}\sum_{k\in\mathcal{K}_L}[y(k,t+1)-y(k,t)]^2. \tag{47}$$

For the parameters describing the relation of the nodes, $\{\mu_{k_1,k_2}\}$ and $\{\sigma_{k_1,k_2}^2\}$, we have already provided them in Eqn. (38) and Eqn. (39).

The deep neural network in Section V-B is designed to have 5 hidden layers. Given the location number K and device number L, each hidden layer is designed to have $4K+L, 4K, 3K, 2K,$ and K neurons, respectively (from the

B. Simulation Results and Discussions

In this subsection, we first testify the feasibility of the proposed Gaussian reference model based on statistical validation. Then we take a look at the result of the single device power control plan. After that, we observe the performance of the multi-device power control and the outcome of the location selection strategy. And finally, the combination of multi-device power control and location selection is presented to show the joint advantage.
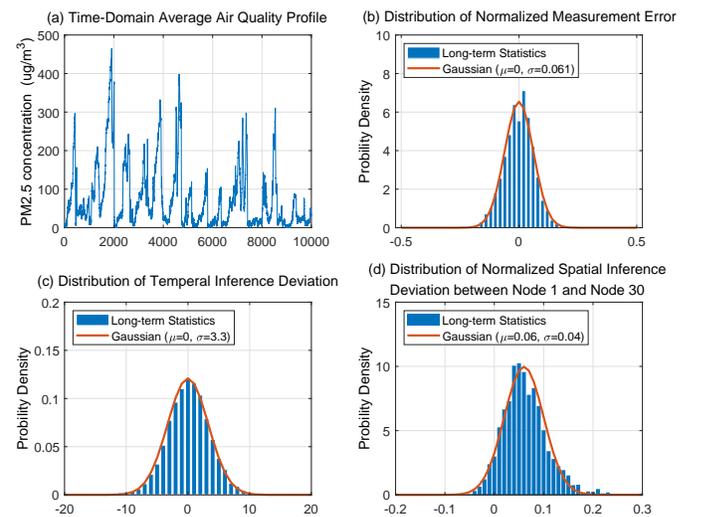

Fig. 8. The testification results of the comparison between the collected data and the Gaussian distribution.

Model Validation: We first use the collected data to testify the feasibility of using Gaussian model to perform inferences. As shown in Fig. 8, we provide four subplots to show the statistics of our data. The subplot (a) illustrates a time-domain profile of the average air quality (of 30 devices) with 10000 continuous time slots (from March 1st to May 15th). A couple of waves of "haze weather" with different peak values can be observed. In the subplot (b), we provide the distribution of the normalized measurement error of the sensors. A zero-mean Gaussian distribution profile with $\sigma = 0.061$ is used to fit the statistics. The subplot (c) shows the distribution of temporal inference deviation. It can be observed that it is quite accurate to use Gaussian distribution to model it. Finally, in subplot (d), the distribution of normalized spatial inference deviation is provided[4]. Without the loss of generality, we only provide the difference between of Node 1 and Node 30 for illustration. The corresponding similarity with Gaussian distribution is not as good as (b) and (c). However, it is not a vital problem since we only aim to provide a rough and simple inference method but not an accurate but unanalyzable method as discussed in Section III-B.
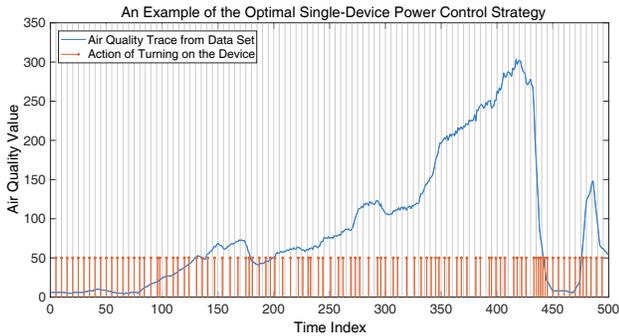

Fig. 9. An example of the optimal single-device control strategy, with $T = 500$, $E = 100$, and $\Delta T = 12$. The sensing actions are performed on the times of those orange sticks.

Single-device power control: We then illustrate the characteristic of the optimal solution for the single-device power control. The result is presented in Fig. 9, where we set the number of time slots $T = 500$, the energy budget $E = 100$, and the maximum allowable sleep time $\Delta T = 10$. The grey vertical lines represent the uniform sensing strategy, which has $T/E = 5$ time slots between adjacent sensing actions. The optimal sensing actions are presented in the form of those short orange lines, which are not uniformly distributed. It can be observed that the sensing actions are more frequent if the air quality changes rapidly. This is because the value of $|\mu - \mu_t|$ will significantly increase if we rely on temporal inference. In this figure, the average joint error $\bar{J}$ of the uniform sensing is

[4] Here we have to exclude the time slots with small values of μ_t due to the following reason: The values of the collected data are an integers (such as PM2.5 values), which are not continuous. If μ_t is low, the difference of two locations mostly resides in $\{-1, 0, 1\}$. The resolution of its distribution is low and the final distribution will have a peak at the location of $x = 0$. This is actually the problem of quantization accuracy, which is not the focus of our paper. Therefore, we only calculate the time slots with $\mu_t > 30$ to get the distribution in the subplot (d).

11.05, while the optimal power control only leads to $\bar{J} = 10.43$ indicating an around 6% performance gain.
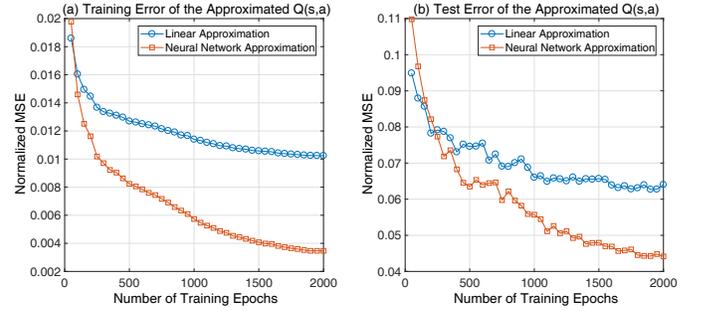

Fig. 10. The training error and test error of the approximated Q(s,a) function. The subplot (a) shows the normalized MSE during training. The subplot (b) shows the normalized MSE when perform a test on another data sets.

Multi-device power control: The multi-device power control involves the deep Q-learning which approximates the $Q(s, a)$ function. Before providing its performance, we first observe the accuracy of such approximation, as given in Fig. 10, where we set $T = 10000$, $E = 2000$, $\Delta T = 12$, $K = 30$, and $L = 20$. The subplot (a) shows the change of Mean Square Error (MSE) when we generate the initial neural network. The accuracy of the neural network is compared with a linear approximation function using the same feature vector as discussed in Section V-B. It can be seen that the neural network has a better fitting result since the minimum value of MSE is be much lower than that of the linear approximation. The subplot (b) shows the test error of the trained neural network by using an additional data set, which is not included in the training data set. It can be seen that the test error is much greater than the training error for both linear approximation and the neural network. However, there is still a diminishing trend of the test error as we keep training the neural network, which indicates that it is able to use a limited set of training data to approximate other potential data set.
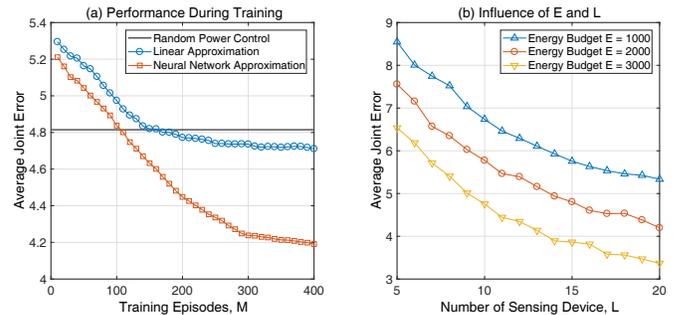

Fig. 11. The performance of the multi-device power control algorithm based on reinforcement learning. The subplot (a) shows performance improvement during the training procedure. The subplot (b) shows the influence of the energy budget and the number of available devices.

Fig. 11 provides the performance of the proposed multi-device power control strategy based on Q-learning. The subplot (a) shows the performance improvement during the iterative training process, where we set $T = 10000$, $E = 2000$,

$\Delta T = 12$, $K = 30$ and $L = 20$. It can be seen that the linear approximation only has a minor advantage over the random power control scheme, while the proposed deep Q-learning scheme shows a much promising performance. The subplot (b) presents the influence of the number of sensing devices and the energy budget of the sensing devices, where $T = 10000$, $\Delta T = 12$, and $K = 30$. There is a negative correlation of the average joint error $\bar{J}$ and the number of devices L, which testifies Proposition 2. And there is a negative correlation of the average joint error $\bar{J}$ and the energy budget E, which testifies Proposition 1. It can also be observed that a greater number of devices would leads to a smaller marginal utility to decrease the joint error.

Fig. 13. The performance showing the result of the combination of location selection and power control.
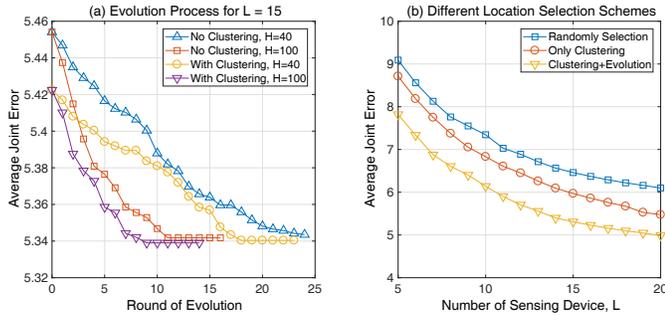
Fig. 12. The outcome of the proposed genetic evolution scheme for location selection. The subplot (a) shows the results of the genetic algorithm with/without initial clustering and different size of gene pool H. The subplot (b) shows the influence of the number of available devices L on different schemes.

Location selection: In Fig. 12, we present the location selection strategy based on the proposed genetic algorithm, where we set the mutation probability $p_m = 0.1$, copy number $M = 3$ and the maximum number of evolution rounds $W = 25$. The subplot (a) shows the improvement of performance during the evolution. Specifically, we have four different settings for comparison by considering the size limit of the gene pool and whether use the clustering to initialize the gene pool. It can be seen that by using the clustering algorithm, the starting point of the system has a better performance and therefore leads to a shorter convergence time. In addition, the 100-sized gene pool has an apparently quicker evolution speed compared with the 40-sized gene pool, at the cost of a higher memory occupation and a greater amount of computation in each round. The four of these lines do not converge to the same value of $\bar{J}$, implying that the solution is still sub-optimal.

Combination of power control and location selection: Finally, we jointly consider the location selection and the power control strategies, as presented in Fig. 13, with $T = 10000$, $\Delta T = 12$, $E = 2000$, $H = 100$, and $K = 30$. The results of four different combinations of the schemes are included, by considering using the two different power control schemes and two different location selection schemes. The upmost curve that represents the random location selection and random power control has the highest average joint error $\bar{J}$. For a spe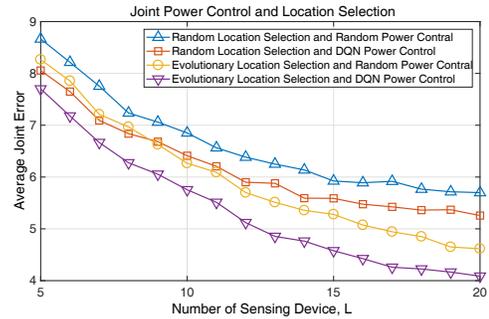cific power control strategy, the evolutionary location selection outperforms the random location selection. And for a specific location selection strategy, the deep Q-learning power control outperforms the random power control. It is also noticeable that the evolutionary location selection has a more obvious gain compared to the deep Q-learning power control when the value of L is high, which causes the crossover of the second and the third curve in Fig. 13.

VIII. Conclusion

In this paper, we propose the architecture, implementation and optimization of our own air quality sensing system, which provides real-time and fine-grained air quality map of the monitored area. Specifically for the optimization, we studied the problem of power control and location selection of the air quality sensing system in a smart city. Our objective was to minimize the joint error of the real-time and fine-grained air quality map, which involved inaccurate data inferences. We first studied the problem of power control in a stochastic environment based on a fixed location selection and then we studied the problem of location selection based on a fixed power control strategy in a given environment. The proposed power control strategy was based on deep Q-learning by re-modeling the problem as a MDP. And the proposed location selection strategy was based on genetic evolutionary algorithm which widely search the solution space. To evaluate the proposed solution, we extracted the properties from our data set based on our own air quality sensing system deployed in Peking University. The simulation result showed that the proposed deep Q-learning power control strategy provided a satisfying performance after learning 200 episodes. And the proposed genetic evolutionary location selection could quickly achieve a suboptimal solution only by using a small gene pool with the size of 100.

References

[1] World Health Organization, "7 Million Premature Deaths Annually Linked to Air Pollution," *Air Quality Climate Change*, vol. 22, no. 1, pp. 53-59, Mar. 2014.

[2] Q. Di *et al.*, "Air Pollution and Mortality in the Medicare Population," *New England J. of Medicine*, vol. 376, no. 26, pp. 2513-2522. Jul. 2017.

[3] G. Kyrkilis *et al.*, "Development of an Aggregate Air Quality Index for an Urban Mediterranean Agglomeration: Relation to Potential Health Effects", *Environment International*, vol. 33, no. 5, pp. 670-676, 2007.

[4] Y. Gao *et al.*, "Mosaic: A Low-Cost Mobile Sensing System for Urban Air Quality Monitoring," *IEEE International Conference on Computer Communications*, San Francisco, CA, Jul. 2016.

[5] Beijing MEMC. (Jul. 2018). *Beijing Municipal Environmental Monitoring Center*. [Online]. Available: http://www.bjmemc.com.cn/

[6] T. Quang *et al.*, "Vertical Particle Concentration Profiles Around Urban Office Buildings," *Atmospheric Chemistry and Physics*, vol. 12, no. 11, pp. 5017-5030, May 2012.

[7] T. N. Quang *et al.*, "Vertical Particle Concentration Profiles around Urban Office Buildings," *Atmos. Chem. Phys.*, vol. 12, no. 11, pp. 5017-5030, May 2012.

[8] Y. Hu *et al.*, "BlueAer: a Fine-Grained Urban PM2.5 3D Monitoring System using Mobile Sensing," in *Proc. IEEE INFOCOM*, San Francisco, CA, Jul. 2016.

[9] Y. Zheng *et al.*, "U-Air: When Urban Air Quality Inference Meets Big Data," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min. (KDD)*, Chicago, IL, USA, Aug. 2013.

[10] Y. Yang *et al.*, "Real-Time Profiling of Fine-Grained Air Quality Index Distribution Using UAV Sensing," *IEEE Internet of Things*, vol. 5, no. 1, pp. 186-198, Feb. 2018.

[11] Y. Yang *et al.*, "Arms: A Fine-Grained 3D AQI Realtime Monitoring System by UAV," in *Proc. IEEE Global Communications Conference*, Singapore, Dec. 2017.

[12] W. Fuertes *et al.*, "Distributed System as Internet of Things for a New Low-Cost, Air Pollution Wireless Monitoring on Real Time," in *Proc. IEEE/ACM 19th International Symposium on Distributed Simulation and Real Time Applications*, Chengdu, China, Oct. 2015.

[13] R. Ranjan *et al.*, "The Next Grand Challenges: Integrating the Internet of Things and Data Science," *IEEE Cloud Computing*, vol. 5, no. 3, pp. 12-26, june 2018.

[14] C. Borrego *et al.*, "How Urban Structure can Affect City Sustainability from an Air Quality Perspective," *Environmental modelling & software*, vol. 21, no. 4, pp. 461-467, Apr. 2006.

[15] Y. Cheng *et al.*, "Aircloud: a Cloud-Based Air-Quality Monitoring System for Everyone," in *Proc. ACM SenSys*, New York, NY, Nov. 2014.

[16] J. Gubbi *et al.*, "Internet of Things (IoT): A Vision, Architectural Elements, and Future Directions," *Future generation computer systems*, vol. 29, no. 7, pp. 1645-1660, 2013.

[17] G. B. Fioccola *et al.*, "Polluino: An Efficient Cloud-Based Management of IoT Devices for Air Quality Monitoring," in *Proc. IEEE Research and Technologies for Society and Industry Leveraging a better tomorrow*, pp. 1-6, Spet. 2016.

[18] Y. Zheng *et al.*, "Forecasting Fine-Grained Air Quality Based on Big Data," in *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining*, Sydney, Australia, Aug. 2015.

[19] Y. Li *et al.*, "Prediction of High Resolution Spatial-Temporal Air Pollutant Map from Big Data Sources," in *Proc. Int. Conf. Big Data Comput. Commun.*, Taiyuan, China, pp. 273C282, Jul. 2015.

[20] Y. Yang *et al.*, "AQNet: Fine-Grained 3D Spatio-Temporal Air Quality Monitoring by Aerial-Ground WSN", in *Proc. IEEE International Conference on Computer Communications*, Honolulu, USA, Apr. 2018.

[21] H. Hsieh *et al.*, "Inferring air quality for station location recommendation based on urban big data," in *Proc. ACM Int. Conf. on Knowledge Discovery and Data Mining*, Sydney, Australia, Aug. 2015.

[22] Y. Yang *et al.*, "Sensor Deployment Recommendation for 3D Fine-Grained Air Quality Monitoring using Semi-Supervised Learning," in *Proc. IEEE International Conference on Communications*, Kansas, USA, May 2018.

[23] Our Website-based GUI. [Online]. Available: http://www.aqimaps.com.

[24] Z. Hu *et al.*, "Aerial-Ground Air Quality Sensing: Architecture, Technologies and Implementation, *IEEE Network Magazine*, accepted, available on https://arxiv.org/abs/1809.03746.

[25] Dataset Collected from Peking University. [Online]. Available: https://github.com/pku-hzw/PKU-Air-IoTJournal-2018.

[26] R. Sutton *et al.*, *Reinforcement Learning: An Introduction*, MIT Press, 1998.

[27] C. Szepesvari, "*Algorithms for Reinforcement Learning*, Morgan & Claypool Publishers, June, 2009.

[28] V. Mnih *et al.*, "Playing Atari with Deep Reinforcement Learning", *arXiv preprint*, arXiv:1312.5602, Dec. 2013.

[29] K. Wagstaff *et al.*, "Constrained k-Means Clustering with Background Knowledge," in *Proc. International Conference on Machine Learning*, pp. 577-584, June, 2001.

[30] D. E. Goldberg *et al.*, "Genetic Algorithm in Search Optimization and Machine Learning," *Machine learning*, vol. 3, no. 2, pp. 95-99, 1988.