# Dynamic Sensor Placement on Graphs Based on Graph Signal Sampling Theory

Saki Nomura*, Junya Hara†, Hiroshi Higashi†, and Yuichi Tanaka†
* Tokyo University of Agriculture and Technology, Tokyo, 184–8588 Japan
† Osaka University, Osaka, 565–0871 Japan
E-mail: s_nomura@msp-lab.org, {j.hara, higashi, ytanaka}@comm.eng.osaka-u.ac.jp

***Abstract*—In this paper, we consider a sensor placement problem where sensors can move within a network over time. Most existing methods assume that sensor positions are static, i.e., they do not move, however, many mobile sensors like drones, robots, and vehicles can change their positions over time. Moreover, underlying measurement conditions could also be changed, which are difficult to cover with statically placed sensors. We tackle the problem by allowing the sensors to change their positions in their neighbors on the network. We dynamically determine the sensor positions based on graph signal sampling theory such that the non-observed signals on the network can be best recovered from the observations. For signal recovery, the dictionary is learned from a pool of observed signals. It is also used for the sensor position selection. In experiments, we validate the effectiveness of the proposed method via the mean squared error of the reconstructed signals. The proposed dynamic sensor placement outperforms the existing static sensor placement in synthetic data.**

## I. Introduction

Sensor placement problem is one of the main research topics in sensor networks [1]–[3]. Its purpose is to find $K$ sensor positions among $N$ candidates $(K < N)$ where sensors can be placed optimally so that the reconstruction error is minimized in some sense. It has been extensively studied in machine learning and signal processing [2], [4]–[6].

Data on a network can be measured by sensors located at its nodes like wireless sensor networks, the Internet, and power grids [7]. Therefore, sensor placement on a network, i.e., graph, has been widely studied [8]. In the graph-based sensor placement problem, nodes of the graph represent possible sensor positions, edges correspond to relationships among the sensors, and the signal value on a node is a measurement.

Many existing works of sensor placement problem assume the static sensor placement, i.e., sensors do not or cannot move [9]–[11]. However, the measurement conditions could be changed over time. In this case, the selected sensors in the static positions do not reflect the current signal statistics. This may lead to the limitation of reconstruction qualities.

To overcome the limitation of the above-mentioned problem, in this paper, we propose a dynamic sensor placement where sensors can flexibly change their positions over time. To realize the dynamic sensor placement, we sequentially learn the dictionary for signal reconstruction from a pool of observed signals on a network (i.e., graph signals) by utilizing sparse coding. We then dynamically determine the sensor positions at every time instance based on graph signal sampling theory [12], such that the non-observed graph signal can be best recovered from the sampled graph signal.

In experiments, we demonstrate that the proposed method outperforms existing sensor placement methods in synthetic datasets.

*Notation:* We consider a weighted undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ and $\mathcal{E}$ represent sets of nodes and edges, respectively. The number of nodes is $N = |\mathcal{V}|$ unless otherwise specified. The adjacency matrix of $\mathcal{G}$ is denoted by $\mathbf{W}$ where its $(m, n)$-element $\mathbf{W}_{mn} \geq 0$ is the edge weight between the $m$th and $n$th nodes; $\mathbf{W}_{mn} = 0$ for unconnected nodes. The degree matrix $\mathbf{D}$ is defined as $\mathbf{D} = \mathrm{diag}\,(d_0, d_1, \ldots, d_{N-1})$, where $d_m = \sum_n \mathbf{W}_{mn}$ is the $m$th diagonal element. We use graph Laplacian $\mathbf{L} \coloneqq \mathbf{D} - \mathbf{W}$ as a graph variation operator. A graph signal $\mathbf{x} \in \mathbb{R}^N$ is defined as a function $x : \mathcal{V} \to \mathbb{R}$. Simply speaking, $x[n]$ is located on the node $n \in \mathcal{V}$ of $\mathcal{G}$.

The graph Fourier transform (GFT) of $\mathbf{x}$ is defined as $\hat{x}(\lambda_i) = \langle \mathbf{u}_i, \mathbf{x} \rangle = \sum_{n=0}^{N-1} u_i[n]x[n]$, where $\mathbf{u}_i$ is the $i$th column of an orthonormal matrix $\mathbf{U}$. It is obtained by the eigendecomposition of the graph Laplacian $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ with the eigenvalue matrix $\mathbf{\Lambda} = \mathrm{diag}\,(\lambda_0, \lambda_1, \ldots, \lambda_{N-1})$. We refer to $\lambda_i$ as a *graph frequency*.

## II. Graph Signal Sampling

In this section, we introduce graph signal sampling. We first review sampling theory on graphs and then describe a sampling set selection on graphs in [13].

### A. Sampling Theory on Graphs

Here, we assume graph signals (i.e., sensor measurements in this paper) inherit the underlying graph information like smooth signals which dominate at low graph frequencies on a network [14]. Suppose that a graph signal is characterized by the following model [12]:

$$\mathbf{x} \coloneqq \mathbf{A}\mathbf{d}, \tag{1}$$

where $\mathbf{A} \in \mathbb{R}^{N \times M}$ $(M \leq N)$ is a known generator matrix and $\mathbf{d} \in \mathbb{R}^M$ is a vector containing expansion coefficients. We can also assume that the small number of atoms (i.e., columns in $\mathbf{A}$) essentially contributes to the signal $\mathbf{x}$. The generator matrix $\mathbf{A}$ specifies the signal subspace $\mathcal{A}$, i.e., $\mathcal{A} = \mathcal{R}(\mathbf{A})$, where $\mathcal{R}(\cdot)$ represents the range space. A piecewise constant (PC) model is well-studied generators [12], [15], [16].

If we do not know the exact $\mathbf{A}$ but have its prior knowledge, it may be learned from the observations. Let $\mathbf{X} \in \mathbb{R}^{N\times M}$ be a collection of $M$ graph signals. Typically, $\mathbf{A}$ has been determined by $\mathbf{A} = \boldsymbol{\mathcal{U}}$, where $\boldsymbol{\mathcal{U}}$ is obtained by the singular value decomposition (SVD) of the collection of observed signals $\mathbf{X} \coloneqq \boldsymbol{\mathcal{U}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{V}}^\top$ [17], and $\boldsymbol{\mathcal{U}}$ and $\boldsymbol{\mathcal{V}}$ are unitary matrices.

Let $\mathbf{S}^\top \in \mathbb{R}^{K\times N}(K \leq N)$ be an arbitrary linear sampling operator where columns of $\mathbf{S}$ are linearly independent. Hereafter, we assume $K = M$ for simplicity, while we can easily extend the case of $K < M$. It specifies the sampling subspace $\mathcal{S}$, i.e., $\mathcal{S} = \mathcal{R}(\mathbf{S})$. A sampled graph signal is represented by $\mathbf{c} = \mathbf{S}^\top\mathbf{x}$. It is best recovered by the following transform [18]–[20]:

$$\tilde{\mathbf{x}} = \mathbf{A}(\mathbf{S}^\top\mathbf{A})^\dagger\mathbf{c}, \tag{2}$$

where $\cdot^\dagger$ is the Moore-Penrose pseudoinverse.

If $\mathcal{A}$ and $\mathcal{S}$ together span $\mathbb{R}^N$ and only intersect at the origin, perfect recovery, i.e., $\tilde{\mathbf{x}} = \mathbf{x}$, is guaranteed. This is referred to as the *direct sum* (DS) condition [12]. Note that the DS condition implies whether $\mathbf{S}^\top\mathbf{A}\mathbf{A}^\top\mathbf{S}$ is invertible, since $(\mathbf{S}^\top\mathbf{A})^\dagger = \mathbf{A}^\top\mathbf{S}(\mathbf{S}^\top\mathbf{A}\mathbf{A}^\top\mathbf{S})^{-1}$ holds if and only if the DS condition is satisfied.

From the perspective of the sensor placement, $\mathbf{S}^\top$ can be considered as a node-wise sampling operator. In the following, we introduce a design method of $\mathbf{S}^\top$ on $\mathcal{G}$ for an arbitrary generator.

### *B. D-optimal Sampling Strategy*

We introduce a graph signal sampling method utilized for our dynamic sensor placement [20]. Here, we define a node-wise sampling operator as follows:

**Definition 1** (Node-wise graph signal sampling)**.** *Let $\mathbf{I}_{\mathcal{M}\mathcal{V}} \in \{0,1\}^{K\times N}$ be the submatrix of the identity matrix indexed by $\mathcal{M} \subset \mathcal{V}$ ($|\mathcal{M}| = K$) and $\mathcal{V}$. The sampling operator is defined as*

$$\mathbf{S}^\top \coloneqq \mathbf{I}_{\mathcal{M}\mathcal{V}}\mathbf{G}, \tag{3}$$

*where $\mathbf{G} \in \mathbb{R}^{N\times N}$ is an arbitrary graph filter.*

While $\mathbf{G}$ is arbitrary, a typical selection of $\mathbf{G}$ is the identity matrix or a graph lowpass filter as that in classical sampling theory. In a sensor placement perspective, $\mathbf{I}_{\mathcal{M}\mathcal{V}}$ specifies the sensor positions on $\mathcal{G}$ and $\mathbf{G}$ can be viewed as a coverage area of sensors. For example, we consider the following polynomial filter

$$[\mathbf{G}]_{nm} = \sum_{i=0}^{N-1}\sum_{p=0}^{P}\alpha_p\lambda^p u_i[n]u_i[m] = \left[\mathbf{U}\left(\sum_{p=0}^{P}\alpha_p\boldsymbol{\Lambda}^p\right)\mathbf{U}^\top\right]_{nm}, \tag{4}$$

where $\{\alpha_p\}$ are arbitrary coefficients. In graph signal processing, (4) is referred to as a $P$-hop localized filter [21], whose nonzero response is limited within $P$-hop neighbors of the target node. This implies that the coverage area of all sensors is limited to $P$-hop.

Sensor placement problem is generally formulated as

$$\mathcal{M}^* = \arg\max_{\mathcal{M}\subset\mathcal{V}} e(\mathcal{M}), \tag{5}$$

where $e$ is a properly designed cost function. Typically, $e$ is designed based on the optimal experimental designs [22], such as A-, E-, and D-optimality. D-optimality is one of the popular experimental designs and is widely used in graph signal sampling. The D-optimality is more appropriate than other optimality designs if we aim to recover graph signals from non-ideally sampled signals [20].

In the D-optimal design, the following problem is considered as selecting an optimal sampling set for graph signals [20]:

$$\mathcal{M}^* = \arg\max_{\mathcal{M}\subset\mathcal{V}} \det(\mathbf{Z}_\mathcal{M}), \tag{6}$$

where $\mathbf{Z} = \mathbf{G}\mathbf{A}\mathbf{A}^\top\mathbf{G}$ and thus $\mathbf{Z}_\mathcal{M} = \mathbf{I}_{\mathcal{M}\mathcal{V}}\mathbf{G}\mathbf{A}\mathbf{A}^\top\mathbf{G}\mathbf{I}_{\mathcal{M}\mathcal{V}}^\top$. Recall that $\mathbf{S}^\top\mathbf{A}\mathbf{A}^\top\mathbf{S}$ is invertible when the DS condition is satisfied. Therefore, the maximization of $\det(\mathbf{Z}_\mathcal{M}) = \det(\mathbf{S}^\top\mathbf{A}\mathbf{A}^\top\mathbf{S}) = |\det(\mathbf{S}^\top\mathbf{A})|^2$ leads to the best recovery of graph signals.

The direct maximization of (6) is combinatorial and is practically intractable. Therefore, a greedy method is applied to (6). Suppose that $\mathrm{rank}(\mathbf{Z}) \geq K$. In [23], (6) can be converted to a greedy selection as follows:

$$\begin{aligned} y^* &= \arg\max_{y\in\mathcal{M}^c} \det(\mathbf{Z}_{\mathcal{M}\cup\{y\}}) \\ &= \arg\max_{y\in\mathcal{M}^c} \mathbf{Z}_{yy} - \mathbf{Z}_{y\mathcal{M}}(\mathbf{Z}_\mathcal{M})^{-1}\mathbf{Z}_{\mathcal{M}y}. \end{aligned} \tag{7}$$

The selection is performed so that the best node $y^*$ is appended to the existing sampling set $\mathcal{M}$ one by one.

## III. Dynamic Sensor Placement on Graphs

In this section, we propose a dynamic sensor placement problem based on sampling theory on graphs. First, we derive online dictionary learning based on sparse coding. Second, we consider the control law of sensors. Fig. 1 shows the overview of the proposed method. The flow of the proposed method is outlined as follows:

1) The dictionary is sequentially learned from previous observations (Section III-A).
2) Time-varying graph signals are sampled and reconstructed based on graph sampling theory (Section II).
3) Sensor positions are dynamically determined based on the dictionary (Section III-B).

### *A. Online Dictionary Learning*

In sampling and reconstruction introduced in Section II-A, $\mathbf{A}$ in (1) is assumed to be fixed. Note that (7) is regarded as a static sensor placement problem since the sensor positions are determined only once. In contrast, we consider the time-varying $\mathbf{A}_t$ in this paper where the subscript $t$ denotes the time instance. In this setting, the optimal sensor positions can be changed according to $\mathbf{A}_t$.

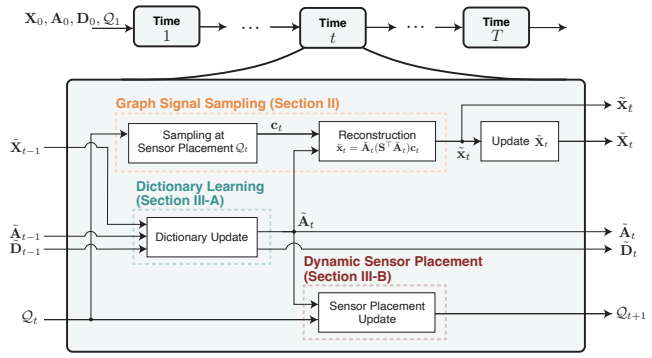


Fig. 1: Overview of the proposed method.

We infer a time-varying generator $\mathbf{A}_t$ from previous $D$ observations

$$\mathbf{X}_{t-1} = [\mathbf{x}_{t-D}, \dots, \mathbf{x}_{t-1}], \tag{8}$$

in every time instance for dynamic sensor placement.

We assume the measurement model as follows:

$$\mathbf{X}_t = f(\mathbf{X}_{t-1}), \tag{9}$$

where $f$ is a mapping from $\mathbf{X}_{t-1}$ to $\mathbf{X}_t$. That is, every observation is generated from signals in the previous time instance. Although the mapping of the time-evolution, $f$, is generally unknown, we suppose that $f$ is Lipschitzian, i.e., there exists some constant $L$ that satisfies the following inequality.

$$\|f(\mathbf{X}_{t-1}) - f(\mathbf{X}_t)\|_F \leq L\|\mathbf{X}_{t-1} - \mathbf{X}_t\|_F. \tag{10}$$

By definition of $f$, (10) can be transformed into

$$\|\mathbf{X}_t - \mathbf{X}_{t+1}\|_F \leq L\|\mathbf{X}_{t-1} - \mathbf{X}_t\|_F. \tag{11}$$

This assumption implies that signals vary smoothly in time[1].

Next, we consider estimating the generator $\mathbf{A}_t$ as a dictionary from (the estimated) $\tilde{\mathbf{X}}_{t-1}$ in (8) based on the assumption in (11), where $\tilde{\cdot}$ corresponds to the estimated vector/matrix.

Since the exact value of the number of atoms, $M$, may not be known in general, herein, we suppose $M \ll D$, and denote a dictionary and expansion coefficient by $\mathbf{A}_t \in \mathbb{R}^{N\times D}$ and $\mathbf{d}_t \in \mathbb{R}^D$, respectively. Therefore, we seek the optimal $\mathbf{A}_t$ such that the number in nonzero values of $\mathbf{d}_t$ is close to $M$, i.e., $\mathbf{d}_t$ is sparse. This setting is similar to well-studied dictionary learning problems [24], [25].

Let $\mathbf{D}_t = [\mathbf{d}_{t-D+1}, \dots, \mathbf{d}_t] \in \mathbb{R}^{D\times D}$ be the collection of expansion coefficients from time $t-D+1$ to $t$. Then, we can express $\mathbf{X}_t = \mathbf{A}_t\mathbf{D}_t$ (see (1)).

Here, we assume that the graph $\mathcal{G}$, initial generator matrix $\mathbf{A}_0$, and expansion coefficients matrix $\mathbf{D}_0$ are given for training, where $\mathbf{A}_0$ and $\mathbf{D}_0$ are arbitrary matrices. Following from (11), we suppose that $\mathbf{X}_t$ is sufficiently close to $\mathbf{X}_{t-1}$. As a result, we formulate the following problem:

$$\arg\min_{\tilde{\mathbf{A}}_t, \tilde{\mathbf{D}}_t} \|\tilde{\mathbf{X}}_{t-1} - \tilde{\mathbf{A}}_t\tilde{\mathbf{D}}_t\|_F^2 + \mu\|\tilde{\mathbf{D}}_t\|_1 + \eta\|\tilde{\mathbf{A}}_t - \tilde{\mathbf{A}}_{t-1}\|_F^2, \tag{12}$$

[1]Note that (11) is necessary to stably learn the dictionary, while we do not explicitly estimate $f$ in the following part.

where $\tilde{\mathbf{X}}_{t-1} = \tilde{\mathbf{A}}_{t-1}\tilde{\mathbf{D}}_{t-1}$, and $\mu$ and $\eta$ are the parameters. The first term in (12) is the data fidelity term for $\tilde{\mathbf{A}}_t$. The second and third terms are for the regularization controlling the sparsity of $\tilde{\mathbf{D}}_t$ and the temporal variation of $\tilde{\mathbf{A}}_t$, respectively.

Note that we need to solve (12) with respect to two variables $\tilde{\mathbf{A}}_t$ and $\tilde{\mathbf{D}}_t$, which jointly form a nonconvex optimization. In this paper, we divide (12) into two independent subproblems with respect to $\tilde{\mathbf{A}}_t$ and $\tilde{\mathbf{D}}_t$, and solve them alternately, similar to the method in [25].

First, we solve (12) with respect to $\tilde{\mathbf{A}}_t$ by fixing $\tilde{\mathbf{D}}_t$ as $\tilde{\mathbf{D}}_{t-1}$. In this case, we easily obtain the closed-form solution for the dictionary as follows:

$$\begin{aligned}\tilde{\mathbf{A}}_t &= \arg\min_{\tilde{\mathbf{A}}_t} \|\tilde{\mathbf{X}}_{t-1} - \tilde{\mathbf{A}}_t\tilde{\mathbf{D}}_{t-1}\|_F^2 + \eta\|\tilde{\mathbf{A}}_t - \tilde{\mathbf{A}}_{t-1}\|_F^2 \\ &= (\eta\tilde{\mathbf{A}}_{t-1} + \tilde{\mathbf{X}}_{t-1}\tilde{\mathbf{D}}_{t-1}^\top)(\eta\mathbf{I}_D + \tilde{\mathbf{D}}_{t-1}\tilde{\mathbf{D}}_{t-1}^\top)^{-1}. \end{aligned}\tag{13}$$

Second, we update $\tilde{\mathbf{D}}_t$ in (12) by using $\tilde{\mathbf{A}}_t$ in (13), i.e.,

$$\tilde{\mathbf{D}}_t = \arg\min_{\tilde{\mathbf{D}}_t} \|\tilde{\mathbf{X}}_{t-1} - \tilde{\mathbf{A}}_t\tilde{\mathbf{D}}_t\|_F^2 + \mu\|\tilde{\mathbf{D}}_t\|_1. \tag{14}$$

To solve (14), we utilize the proximal gradient method [26]–[28], whose problem is given by the following form.

$$\tilde{\mathbf{D}}_t = \arg\min_{\tilde{\mathbf{D}}_t} g(\tilde{\mathbf{D}}_t) + h(\tilde{\mathbf{D}}_t), \tag{15}$$

where $g$ is a differentiable function and $h$ is a proximable function [22]. Note that (15) is set to the entire problem being convex. As a result, the optimal solution is obtained by the following update rule.

$$\tilde{\mathbf{D}}_{t,n+1} = \mathrm{prox}_{\gamma h}(\tilde{\mathbf{D}}_{t,n} - \gamma\nabla g(\tilde{\mathbf{D}}_{t,n})), \tag{16}$$

where $n+1$ is the number of iterations, $\gamma$ is the step size, $\nabla g$ is the gradient of $g$, and $\mathrm{prox}_{\gamma h}$ is defined as

$$\mathrm{prox}_{\gamma h}(\mathbf{D}) = \arg\min_{\mathbf{C}} \frac{1}{2}\|\mathbf{C} - \mathbf{D}\|_F^2 + \gamma\mu\|\mathbf{C}\|_1 = S_{\gamma\mu}(\mathbf{D}), \tag{17}$$

where $S_{\gamma\mu}$ is the soft thresholding operator defined as follows:

$$[S_{\gamma\mu}(\mathbf{D})]_{ij} = \begin{cases} \mathbf{D}_{ij} - \gamma\mu, & \mathbf{D}_{ij} \geq \gamma\mu \\ 0, & |\mathbf{D}_{ij}| < \gamma\mu \\ \mathbf{D}_{ij} + \gamma\mu, & \mathbf{D}_{ij} \leq -\gamma\mu. \end{cases} \tag{18}$$

By applying (17) to (16), we have the following iteration with ISTA [29].

$$\begin{aligned}\mathbf{D}_{t,n+1} &= S_{\gamma\mu}(\mathbf{D}_{t,n} - \gamma\nabla g(\mathbf{D}_{t,n})) \\ &= S_{\gamma\mu}\{\mathbf{D}_{t,n} - 2\gamma\mathbf{A}_t^\top(\mathbf{A}_t\mathbf{D}_{t,n} - \tilde{\mathbf{X}}_{t-1})\}, \end{aligned}\tag{19}$$

We set the step size as $\gamma \leq 1/\lambda_{\max}(\mathbf{A}_t^\top\mathbf{A}_t)$ according to the convergence condition of ISTA [29]. We iterate (19) until $\|\mathbf{D}_{t,n+1} - \mathbf{D}_{t,n}\|_F^2 < \epsilon$ where $\epsilon$ is a small constant. The optimization of (12) is summarized in Algorithm 1.

In the following, we formulate the dynamic sensor placement based on the learned dictionary. The control law of sensors is also derived based on the sampling strategy introduced in (7).

**Algorithm 1:** Online Dictionary Learning

**Input:** $\tilde{\mathbf{X}}_{t-1}, \tilde{\mathbf{A}}_{t-1}, \tilde{\mathbf{D}}_{t-1}, t$
**if** $t = 1$ **then**
  $\tilde{\mathbf{A}}_1 = \mathbf{A}_0$
**else**
  $\tilde{\mathbf{A}}_t = (\eta\tilde{\mathbf{A}}_{t-1} + \tilde{\mathbf{X}}_{t-1}\tilde{\mathbf{D}}_{t-1}^\top)(\eta\mathbf{I}_D + \tilde{\mathbf{D}}_{t-1}\tilde{\mathbf{D}}_{t-1}^\top)^{-1}$
$\tilde{\mathbf{D}}_t = \tilde{\mathbf{D}}_{t-1}$
**while** $\|\tilde{\mathbf{D}}_{t,n} - \tilde{\mathbf{D}}_{t,n-1}\|_F^2 \geq \epsilon$ **do**
  $\tilde{\mathbf{D}}_{t,n+1} = S_{\gamma\mu}\{\tilde{\mathbf{D}}_{t,n} - 2\gamma\tilde{\mathbf{A}}_t^\top(\tilde{\mathbf{A}}_t\tilde{\mathbf{D}}_{t,n} - \tilde{\mathbf{X}}_{t-1})\}$
  $n \leftarrow n + 1$
**Output:** $\tilde{\mathbf{A}}_t$

### B. Dynamic Sensor Placement

For brevity, we define the following matrix:

$$\mathbf{N}_t \coloneqq \tilde{\mathbf{A}}_t^\top \mathbf{G}. \tag{20}$$

Note that we now assume that $\tilde{\mathbf{A}}_t$ is given. With (20), (7) can be rewritten by a dynamic form as[2]

$$\begin{aligned} y_t^* &= \arg\max_{y\in\mathcal{M}^c} \mathbf{N}_{:y}^\top\mathbf{N}_{:y} - \mathbf{N}_{:y}^\top\mathbf{N}_{:\mathcal{M}}(\mathbf{N}_{:\mathcal{M}}^\top\mathbf{N}_{:\mathcal{M}})^{-1}\mathbf{N}_{:\mathcal{M}}^\top\mathbf{N}_{:y} \\ &= \arg\max_{y\in\mathcal{M}^c} \|\boldsymbol{\nu}_y\|^2 - \|\mathbf{P}_{\mathcal{R}(\mathbf{N}_{:\mathcal{M}})}\boldsymbol{\nu}_y\|^2, \end{aligned} \tag{21}$$

where $\boldsymbol{\nu}_y = \mathbf{N}_{:y}$ and $\mathbf{P}_{\mathcal{R}(\mathbf{Q})}$ is the orthogonal projection onto $\mathcal{R}(\mathbf{Q})$. Since $\mathbf{N}$ can temporally vary, the optimal solution in (21) can also change in every time instance.

Note that (21) is based on a greedy selection introduced in Section II-B. However, in practice, this optimization may not be efficient for large graphs: The control law with (21) implies that we need a large computational burden if all sensor positions are sequentially determined at every time instance. This results in a delay for sensor relocations which should be alleviated.

In this paper, instead, we consider selecting the sensor positions independently, i.e., (21) is solved for a sensor independently of the other ones. In the following, we rewrite (21) as a distributed optimization.

Here, we denote the position of the $i$th sensor at the $t$th time instance by $q_{i,t}$. We define the $i$th Voronoi region in the graph $\mathcal{G}$ as follows:

$$\mathcal{W}_{i,t} = \{v \in \mathcal{V} \mid d(v, q_{i,t}) < d(v, q_{j,t}), \forall q_j \in \mathcal{Q}_t\}, \tag{22}$$

where $d(v, q_{i,t})$ is the shortest distance between $v$ and $q_{i,t}$, and $\mathcal{Q}_t = \{q_{j,t}\}_{j=1,\dots,K}$ denotes the set of all sensor positions at $t$. Note that, in each $\mathcal{W}_{i,t}$, only one sensor is contained: We illustrate its example in Fig. 2.

By utilizing (22), we can rewrite (21) for seeking the best position of a sensor within its Voronoi region $\mathcal{W}_{i,t}$ as follows:

$$q_{i,t+1}^* = \arg\max_{y\in\mathcal{W}_{i,t}} \|\boldsymbol{\nu}_y\|^2 - \|\mathbf{P}_{\mathcal{R}(\mathbf{N}_{:\mathcal{M}_{i,t}})}\boldsymbol{\nu}_y\|^2. \tag{23}$$

[2]For notation simplicity, we omit $\cdot_t$ in $\mathbf{N}_t$ hereafter.

**Algorithm 2:** Dynamic Sensor placement

**Input:** Sensing matrix $\mathbf{G}$, initial positions of sensors $\mathcal{Q}_1$, time duration $T$
**for** $t = 1, \dots, T$ **do**
  Update the dictionary $\tilde{\mathbf{A}}_t$ with Algorithm 1
  Update the data matrix $\tilde{\mathbf{X}}_t$
  Calculate the Voronoi regions $\{\mathcal{W}_{i,t}\}_{i=1,\dots,K}$
  $\mathbf{N}_t \leftarrow \tilde{\mathbf{A}}_t^\top\mathbf{G}$
  **par for** $i = 1, \dots, K$
    Calculate $\mathbf{P}_{\mathcal{R}(\mathbf{N}_{:\mathcal{M}_{i,t}})}$
    Calculate the $P$ hop neighbors $\{\mathcal{N}_{i,t}^P\}_{i=1,\dots,K}$
    $q_{i,t+1}^* = \arg\max_{y\in\mathcal{W}_{i,t}\cap\mathcal{N}_{i,t}^P} \|\boldsymbol{\nu}_y\|^2 - \|\mathbf{P}_{\mathcal{R}(\mathbf{N}_{:\mathcal{M}_{i,t}})}\boldsymbol{\nu}_y\|^2$
  $\mathcal{Q}_{t+1} \leftarrow \{q_{i,t+1}^*\}_{i=1,\dots,K}$
**Output:** Sensor positions $\{\mathcal{Q}_t\}_{t=1,\dots,T+1}$

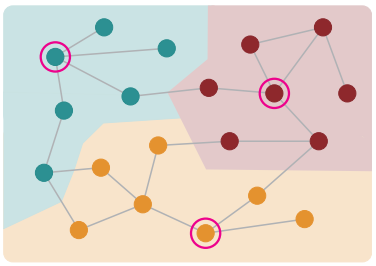

Fig. 2: Voronoi diagram on a graph. Nodes with circles represent the current sensor positions. Each colored area represents the Voronoi region corresponding to each sensor.

Since (23) depends only on the Voronoi region of a sensor, all sensor selections in (23) are independent of each other. Therefore, we can select sensors by (23) in a distributed fashion.

When we assume mobile sensors, their movable areas could be restricted to their neighbors. To reflect this, we can further impose a constraint on the Voronoi region where the movable nodes are restricted to the $P$-hop neighbors of the target node. The $P$-hop neighbor of the $i$th sensor (node) is defined as

$$\mathcal{N}_{i,t}^P = \{v \in \mathcal{V} \mid d(v, q_{i,t}) \leq P\}. \tag{24}$$

As a result, the proposed control law of sensors in (23) is further modified as follows:

$$q_{i,t+1}^* = \arg\max_{y\in\mathcal{W}_{i,t}\cap\mathcal{N}_{i,t}^P} \|\boldsymbol{\nu}_y\|^2 - \|\mathbf{P}_{\mathcal{R}(\mathbf{N}_{:\mathcal{M}_{i,t}})}\boldsymbol{\nu}_y\|^2. \tag{25}$$

Algorithm 2 summarizes the proposed dynamic sensor placement method[3]. Note that our main contribution is to formulate dynamic sensor placement as graph signal sampling. We may be able to use other methods, such as those described in [30] and [31], as internal algorithms for dictionary learning.

[3]We utilize the `MATLAB` notation where we denote the parallel processing in Algorithm 2 as **par for**.

## IV. Experiments

In this section, we compare reconstruction errors of time-varying graph signals measured by selected positions of sensors. We perform recovery experiments for synthetic datasets.

We suppose that the graph $\mathcal{G}$ is given a priori. The nodes represent the candidates of sensor positions, and the edges indicate the paths where sensors can move. Sampling is performed at every time instance. After sampling, the reconstructed signal at $t$, i.e., $\tilde{\mathbf{x}}_t$, is stored as the latest signal collection $\tilde{\mathbf{X}}_t$, which is used for the dictionary/sensor positions update. Simultaneously, $\tilde{\mathbf{x}}_{t-D}$ is discarded from $\tilde{\mathbf{X}}_t$.

### A. Setup

We compare the signal reconstruction accuracy of the proposed method with those of the following existing methods.

1) *Static1*: Static sensor placement with the fixed dictionary learned by the SVD only once at $t = 0$ [17]. This method does not change the dictionary over time.
2) *Static2*: Static sensor placement (similar to Static1) with the non-fixed dictionary learned by the SVD at every time instance.

Since the SVD-based dictionary learning is still widely-used in many sensor placement applications [32], Static1 and Static2 are possible baseline methods. We also compare the proposed method to the extreme case of the proposed method, i.e., $P = \infty$. In this case, sensors can move anywhere in Voronoi region regardless of the current sensor positions.

The recovery experiment is performed for signals on a random sensor graph with $N = 256$. The number of sensors is set to $K = 8$.

We consider the following the piecewise-constant (PC) model:

$$\mathbf{x} = \sum_{i=0}^{M-1} d[i]\mathbf{1}_{\mathcal{T}_i} = [\mathbf{1}_{\mathcal{T}_1}, \ldots, \mathbf{1}_{\mathcal{T}_M}]\mathbf{d}, \tag{26}$$

where $\mathcal{T}_i (i = 1, \ldots, M)$ is the nonoverlapping subset of nodes for the $i$th cluster: $[\mathbf{1}_{\mathcal{T}_i}]_n = 1$ if the node $n$ is in $\mathcal{T}_i$ and 0 otherwise [33]. In this case, the generator matrix is $\mathbf{A} = [\mathbf{1}_{\mathcal{T}_1}, \ldots, \mathbf{1}_{\mathcal{T}_M}]$ where $M$ is the number of clusters. The number of clusters is set to $M = 3$. We randomly separate nodes into $M$ clusters so that nodes in a cluster are connected. We generate $\mathbf{d}_t$ as follows:

$$\mathbf{d}_t = \begin{bmatrix} 3\{\exp(-\frac{t}{25})\sin(t + \frac{\pi}{3}) + 1\} \\ 3\{\exp(-\frac{t}{25})\sin(2t) + 1\} \\ 3\{\exp(-\frac{t}{25})\sin(3t - \frac{\pi}{3}) + 1\} \end{bmatrix}. \tag{27}$$

In this model, signal values in a cluster change simultaneously (but the temporal frequencies are different for different clusters). It can be a possible model for clustered sensor networks.

For the PC model, we generate the data matrix $\mathbf{X}_t$ with $T = 40$. The number of time slots is set to $D = 20$. The initial expansion coefficients and dictionary can be arbitrary as mentioned in Section III-A. In the experiment, however, we used an SVD-based dictionary for a fair comparison to alternative methods. Therefore, $\mathbf{A}_0 = \boldsymbol{\mathcal{U}}$ at $t = 0$, where $\boldsymbol{\mathcal{U}}$ is obtained by the SVD of the known measurement data, i.e., $\mathbf{X}_0 = \boldsymbol{\mathcal{U}}\boldsymbol{\Sigma}\boldsymbol{\mathcal{V}}^\top$. The initial value of $\mathbf{D}_t$ is also set to $\mathbf{D}_0 = \mathbf{1}\mathbf{1}^\top$. We set the temporal sampling period $T_s = \frac{\pi}{30}$ to avoid aliasing.

The parameters of the dictionary learning are experimentally set to $(\gamma, \eta, \mu) = (10^{-4}, 3, 1)$, and $P = 1$ is used for the sensor movable area where we assume a sensor can only be moved to its one-hop neighbor based on reasonable sensor mobility.

Recovery accuracy is measured in noisy scenarios where white Gaussian noise $w[i] \sim \mathcal{N}(0, 0.1)$ is added to the signals. For all methods, the initial position of sensors is determined by a state-of-the-art sampling set selection method on graphs [2]. We calculate the averaged mean squared errors (MSEs) for 50 independent runs.

### B. Results

Fig. 3 shows the experimental results for the noisy case. We also show examples of the original and reconstructed graph signals of PC models in Fig. 4. Fig. 3 indicates that the proposed method shows consistently (and significantly) smaller MSEs than the other methods in all time instances, and the MSE of the proposed method gradually decreases over time. This validates that the proposed method can adapt to the change in measurement conditions, while the static methods fail to do so. In this experiment, $P = 1$ and $\infty$ do not results in a large difference. However, as shown in Fig. 4 at $t = 13$, we observe that $P = \infty$ produces a larger error than $P = 1$ because several sensor locations are very close to each other. A possible reason for this is that the sensor positions were biased toward the right side.

## V. Conclusion

In this paper, we propose a dynamic sensor placement method based on graph sampling theory. We sequentially learn the dictionary from a time series of observed graph signals by utilizing sparse coding. Using the dictionary, we dynamically determine the sensor placement at every time instance such that the non-observed graph signal values can be best recovered from those of the observed (selected) nodes. In experiments, we demonstrate that the proposed method outperforms existing static sensor placement methods in synthetic datasets.

More details on theoretical aspects and comprehensive experimental results with respect to this paper can be found in [34].

## Acknowledgment

This work is supported in part by JSPS KAKENHI under Grant 23K26110 and 23K17461, and JST AdCORP under Grant JPMJKB2307.

## References

[1] F. Y. Lin and P.-L. Chiu, "A near-optimal sensor placement algorithm to achieve complete coverage-discrimination in sensor networks," *IEEE Commun. Lett.*, vol. 9, no. 1, pp. 43–45, 2005.

[2] A. Sakiyama, Y. Tanaka, T. Tanaka, and A. Ortega, "Eigendecomposition-free sampling set selection for graph signals," *IEEE Trans. Signal Process.*, vol. 67, no. 10, pp. 2679–2692, 2019.

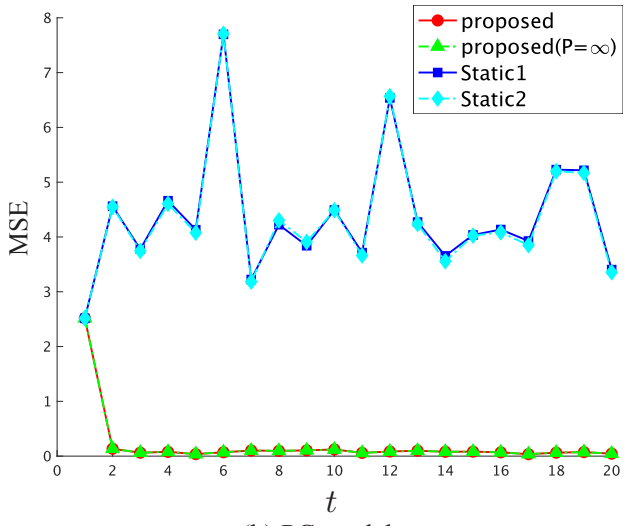


Fig. 3: MSE comparison of the reconstructed synthetic data.

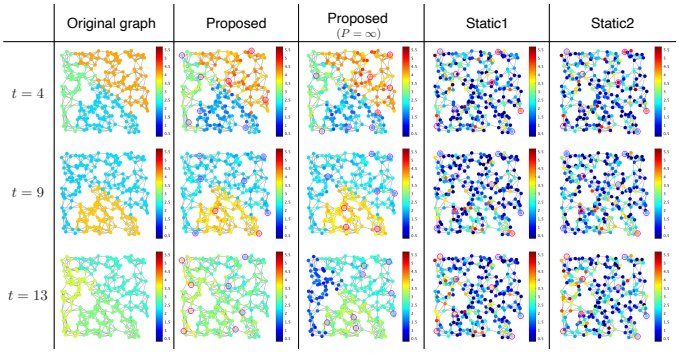


Fig. 4: Reconstructed graph signals for PC signals at time $t = 4, 9, 13$. Circled nodes represent selected sensor positions.


[3] Y. Jiang, J. Bigot, and S. Maabout, *Sensor Selection on Graphs via Data-driven Node Sub-sampling in Network Time Series*, 2020. arXiv: 2004.11815.

[4] A. Downey, C. Hu, and S. Laflamme, "Optimal sensor placement within a hybrid dense sensor network using an adaptive genetic algorithm with learning gene pool," *Struct. Health Monit.*, vol. 17, no. 3, pp. 450–460, 2018.

[5] J. Li, "Exploring the potential of utilizing unsupervised machine learning for urban drainage sensor placement under future rainfall uncertainty," *J. Environ. Manage.*, vol. 296, p. 113 191, 2021.

[6] S. Joshi and S. Boyd, "Sensor selection via convex optimization," *IEEE Trans. Signal Process.*, vol. 57, no. 2, pp. 451–462, 2008.

[7] L. M. Oliveira and J. J. Rodrigues, "Wireless Sensor Networks: A Survey on Environmental Monitoring.," *J. Commun.*, vol. 6, no. 2, pp. 143–151, 2011.

[8] A. Sakiyama, Y. Tanaka, T. Tanaka, and A. Ortega, "Efficient sensor position selection using graph signal sampling theory," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, pp. 6225–6229.

[9] B. Li, H. Liu, and R. Wang, "Efficient sensor placement for signal reconstruction based on recursive methods," *IEEE Trans. Signal Process.*, vol. 69, pp. 1885–1898, 2021.

[10] H. Zhou, X. Li, C.-Y. Cher, E. Kursun, H. Qian, and S.-C. Yao, "An information-theoretic framework for optimal temperature sensor allocation and full-chip thermal monitoring," in *DAC Design Automation Conference 2012*, IEEE, 2012, pp. 642–647.

[11] K. Visalini, B. Subathra, S. Srinivasan, G. Palmieri, K. Bekiroglu, and S. Thiyaku, "Sensor placement algorithm with range constraints for precision agriculture," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 34, no. 6, pp. 4–15, 2019.

[12] Y. Tanaka, Y. C. Eldar, A. Ortega, and G. Cheung, "Sampling signals on graphs: From theory to applications," *IEEE Signal Process. Mag.*, vol. 37, no. 6, pp. 14–30, 2020.

[13] J. Hara and Y. Tanaka, "Sampling set selection for graph signals under arbitrary signal priors," in *2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2022, pp. 5732–5736.

[14] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, "The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains," *IEEE Signal Process. Mag.*, vol. 30, no. 3, pp. 83–98, 2013.

[15] S. Chen, R. Varma, A. Sandryhaila, and J. Kovačević, "Discrete signal processing on graphs: Sampling theory," *IEEE Trans. Signal Process.*, vol. 63, no. 24, pp. 6510–6523, 2015.

[16] M. Tsitsvero, S. Barbarossa, and P. Di Lorenzo, "Signals on graphs: Uncertainty principle and sampling," *IEEE Trans. Signal Process.*, vol. 64, no. 18, pp. 4845–4860, 2016.

[17] B. Jayaraman and S. M. Mamun, "On data-driven sparse sensing and linear estimation of fluid flows," *Sensors*, vol. 20, no. 13, p. 3752, 2020.

[18] Y. C. Eldar, *Sampling Theory: Beyond Bandlimited Systems*. Cambridge University Press, 2015.

[19] Y. Tanaka and Y. C. Eldar, "Generalized sampling on graphs with subspace and smoothness priors," *IEEE Trans. Signal Process.*, vol. 68, pp. 2272–2286, 2020.

[20] J. Hara, Y. Tanaka, and Y. C. Eldar, "Graph signal sampling under stochastic priors," *IEEE Trans. Signal Process.*, 2023.

[21] D. I. Shuman, B. Ricaud, and P. Vandergheynst, "Vertex-frequency analysis on graphs," *Appl. Comput. Harmon. Anal.*, vol. 40, no. 2, pp. 260–291, 2016.

[22] L. Condat, "A primal–dual splitting method for convex optimization involving Lipschitzian, proximable and linear composite terms," *J. Optim. Theory Appl.*, vol. 158, no. 2, pp. 460–479, 2013.

[23] F. Zhang, *The Schur Complement and Its Applications*. Springer Science & Business Media, 2006, vol. 4.

[24] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, 2006.

[25] K. Engan, S. O. Aase, and J. H. Husoy, "Method of optimal directions for frame design," in *1999 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 1999, pp. 2443–2446.

[26] G. B. Passty, "Ergodic convergence to a zero of the sum of monotone operators in Hilbert space," *J. Math. Anal. Appl.*, vol. 72, no. 2, pp. 383–390, 1979.

[27] P. Tseng, "Applications of a splitting algorithm to decomposition in convex programming and variational inequalities," *SIAM J. Control Optim.*, vol. 29, no. 1, pp. 119–138, 1991.

[28] P. L. Combettes and V. R. Wajs, "Signal recovery by proximal forward-backward splitting," *Multiscale Model. Simul.*, vol. 4, no. 4, pp. 1168–1200, 2005.

[29] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," en, *SIAM J. Imaging Sci.*, vol. 2, no. 1, pp. 183–202, Jan. 2009.

[30] K. Zhang, M. Coutino, and E. Isufi, "Sampling graph signals with sparse dictionary representation," in *2021 29th European Signal Processing Conference (EUSIPCO)*, IEEE, 2021, pp. 1815–1819.

[31] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 689–696.

[32] K. Manohar, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Data-driven sparse sensor placement for reconstruction: Demonstrating the benefits of exploiting known patterns," *IEEE Control Syst. Mag.*, vol. 38, no. 3, pp. 63–86, 2018.

[33] S. Chen, R. Varma, A. Singh, and J. Kovačević, "Representations of piecewise smooth signals on graphs," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, IEEE, 2016, pp. 6370–6374.

[34] S. Nomura, J. Hara, H. Higashi, and Y. Tanaka, "Dynamic sensor placement based on sampling theory for graph signals," *IEEE Open Journal of Signal Processing*, 2024, in press.