Distributed Spatial Filtering Over Networked Systems

Shinsaku Izumi, *Member, IEEE*, Ryosuke Katayama, Xin Xin, *Senior Member, IEEE*, and Taiga Yamasaki

***Abstract*—This letter concerns distributed spatial filtering over networked systems, i.e., transforming signal values given for nodes to those with a desired spatial frequency characteristic via a distributed computation. An existing filtering algorithm can achieve only low-pass filter characteristics, which limits its range of applications. To address this limitation, we extend the aforementioned filtering algorithm using an additional design parameter. We then present a characterization of all the realizable filter characteristics as a necessary and sufficient condition for achieving distributed spatial filtering. As a result, it is shown that the extended algorithm increases the range of the realizable filter characteristics. The proposed method is verified not only by simulation but also by denoising experiments for a real sensor network. The results show that the proposed method effectively reduces spatial noise and achieves higher performance than an average consensus algorithm and an average filter.**

***Index Terms*—Control of networks, distributed control, filtering, sensor networks.**

I. Introduction

NETWORKED systems, in which subsystems (i.e., nodes) are interconnected through networks, are of great interest in the field of systems and control. The reason for this is that this type of system appears widely in modern applications, such as sensor networks and intelligent transportation systems.

For the networked systems, typical tasks, e.g., consensus [1], [2] and formation [3], [4], have been well studied, whereas here, we consider a different task, namely *distributed spatial filtering (DSF)*. In this task, the nodes obtain signal values with a desired *spatial frequency* characteristic from given ones only through local communications. An example of low-pass filtering is illustrated in Fig. 1, where x_i is the signal value for node i and we assume that the nodes with close indices

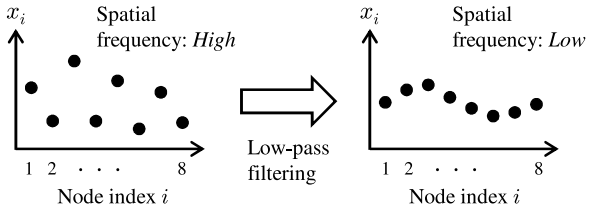


Fig. 1. Example of spatial low-pass filtering.

are located near each other. The nodes obtain the signal values with a low spatial frequency (right) from those with a high spatial frequency (left), in a distributed manner. The main reason for considering this filtering is that the spatial frequencies of some physical quantities have useful properties. For example, temperature generally has similar values at locations within close proximity to each other (i.e., the spatial frequency is low). Thus, when measuring temperature at different locations using a sensor network, spatial low-pass filtering is useful for reducing the measurement noise. Moreover, spatial high-pass filtering can be applied to the detection of the sensor malfunction [5], as the malfunction of a certain sensor will cause unusual differences between the measurements, which increases the high-frequency components.

So far, DSF has never been studied in the field of systems and control because researchers have been mainly interested in control in the time domain (see [1]–[4]), and hence, they have not considered the spatial frequencies. Nevertheless, there are related studies of signal processing on graphs and its applications. For example, Shuman *et al.* [6] gave a distributed graph filtering method using the Chebyshev polynomial approximation. This method may require a large memory for each node, thereby increasing costs. Segarra *et al.* [7] and Yi *et al.* [8] examined the linear transformation between two signals on a graph and average consensus, respectively, but these are different from DSF.

Motivated by this, Izumi *et al.* [9] proposed a method of DSF based on a consensus algorithm. However, the following two issues remain unsolved. First, the achievable filter characteristics are limited to low-pass ones. Overcoming this issue broadens the application range of DSF. Second, the method proposed in [9] was not verified through experiments. DSF requires wireless communications between nodes, but they are

Manuscript received March 16, 2020; revised May 28, 2020; accepted June 12, 2020. Date of publication June 24, 2020; date of current version July 9, 2020. This work was supported in part by the Okayama Foundation for Science and Technology and JSPS KAKENHI under Grant 19K15016. Recommended by Senior Editor J. Daafouz. *(Corresponding author: Shinsaku Izumi.)*

The authors are with the Faculty of Computer Science and Systems Engineering, Okayama Prefectural University, Okayama 719-1197, Japan (e-mail: izumi@cse.oka-pu.ac.jp; xxin@cse.oka-pu.ac.jp; taiga@cse.oka-pu.ac.jp).

Digital Object Identifier 10.1109/LCSYS.2020.3004728

unreliable in practice, which leads to gaps between theory and practice. Thus, examining the magnitude of theses gaps through experiments is crucial for the practical use of DSF.

This letter addresses the above two issues and makes the following two contributions.

1) We extend the theoretical framework developed in [9]. By introducing a new design parameter, we extend the filtering algorithm given in [9]. We then characterize *all* the achievable filter characteristics for the extended algorithm, unlike in [9], where all the achievable filter characteristics were not guaranteed to be characterized. The result shows that the extended algorithm provides a wide range of filter characteristics compared to the original algorithm, and further reveals its performance limit as the spatial filter.
2) We demonstrate the effectiveness of our DSF method through experiments, using a real sensor network. We investigate denoising via DSF, and perform experiments to verify its performance. The experimental results indicate that our DSF method effectively reduces spatial noise. This demonstrates that the gap between our theory and practice is small and that our DSF method is relevant for real applications.

Notation: Let $\mathbb{R}$ and $\mathbb{R}_+$ be the real number field and the set of positive real numbers, respectively. We use I to represent the identity matrix. For the numbers $x_1, x_2, \ldots, x_n \in \mathbb{R}$, let $\mathrm{diag}(x_1, x_2, \ldots, x_n)$ be the diagonal matrix whose i-th diagonal entry is x_i, and let $[x_i]_{i\in\mathbb{I}} \coloneqq [x_{i_1}\ x_{i_2}\ \cdots\ x_{i_m}]^\top \in \mathbb{R}^m$, where $\mathbb{I} \coloneqq \{i_1, i_2, \ldots, i_m\} \subseteq \{1, 2, \ldots, n\}$. We denote the cardinality of the set $\mathbb{S}$ as $|\mathbb{S}|$.

II. Signal Processing on Graphs

To explain the problem addressed in this letter, we briefly introduce signal processing on graphs [5], [10].

Signals on graphs (or *graph signals*) consist of graphs and signal values on their vertices, where the graphs represent the relation among the signal values. Graph signals are mathematically described as follows. Consider the undirected graph $G = (\mathbb{V}, \mathbb{E})$ with n vertices, where $\mathbb{V} \coloneqq \{1, 2, \ldots, n\}$ and $\mathbb{E}$ are the vertex and edge sets, respectively. Then, the pair (G, s) defines a graph signal, where $s \in \mathbb{R}^n$ is the vector consisting of the signal values on the n vertices.

Next, the Fourier transform of graph signals (or the graph Fourier transform) is introduced. Consider the graph signal (G, s). Let $L \in \mathbb{R}^{n\times n}$ denote the Laplacian matrix of the graph G and λ_p ($p \in \{1, 2, \ldots, n\}$) denote the eigenvalue of L with the p-th smallest modulus. Then, the graph Fourier transform $f \in \mathbb{R}^n$ is given by

$$f(\lambda_1, \lambda_2, \ldots, \lambda_n) \coloneqq V^\top s, \tag{1}$$

where $V \in \mathbb{R}^{n\times n}$ is an orthogonal matrix (i.e., $V^\top = V^{-1}$) satisfying

$$V^\top L V = \Lambda \tag{2}$$

for $\Lambda \coloneqq \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. There always exists such a V because L is a symmetric matrix when G is an undirected graph. From (1) and (2), the graph Fourier transform f is defined as the expansion of s in terms of the eigenvectors of L. This parallels the classical Fourier transform known to be the expansion of a signal in terms of the complex exponentials that are the eigenfunctions of the one-dimensional Laplace operator [5], [10]. The graph Fourier transform f is a spatial frequency-based representation of the graph signal (G, s). More concretely, f indicates the magnitude of the signal value differences among connected vertices on G. The eigenvalue λ_p ($p \in \{1, 2, \ldots, n\}$) in (1) corresponds to the spatial frequency, and the p-th entry of f represents the component of λ_p, where $\lambda_1, \lambda_2, \ldots, \lambda_n$ are nonnegative real numbers because G is an undirected graph. Besides, the relation $V^\top = V^{-1}$ implies that the inverse graph Fourier transform is given by

$$s = Vf(\lambda_1, \lambda_2, \ldots, \lambda_n). \tag{3}$$

Finally, we define the spatial filtering of graph signals as

$$\tilde{s} = V\mathrm{diag}(h(\lambda_1), h(\lambda_2), \ldots, h(\lambda_n))V^\top s, \tag{4}$$

where $\tilde{s} \in \mathbb{R}^n$ is the collective signal value of a filtered graph signal and $h : \mathbb{R}_+ \cup \{0\} \to \mathbb{R}$ is a function determining the filter characteristic. As an example of h, if $|h(\lambda_p)|$ decreases as p increases, the resulting filter is a low-pass one which reduces the amplitudes of high-frequency components. Choosing an appropriate h and using (4), we can obtain $\tilde{s}$ with a desired spatial frequency characteristic.

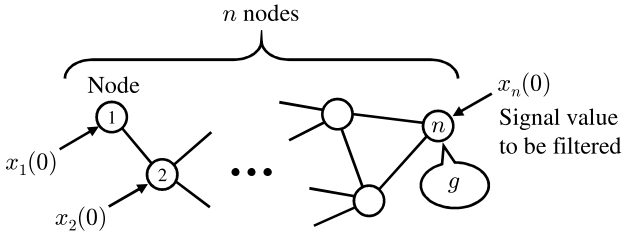


Fig. 2. Networked system Σ.

III. DSF Over Networked Systems

A. Problem Formulation

Consider the networked system Σ with n nodes, depicted in Fig. 2.

The behavior of node i ($i \in \mathbb{V}$) is described by

$$x_i(t+1) = g([x_j(t)]_{j\in\mathbb{N}_i}, t), \tag{5}$$

where $x_i(t) \in \mathbb{R}$ is the state (corresponding to a memory), $[x_j(t)]_{j\in\mathbb{N}_i} \in \mathbb{R}^{|\mathbb{N}_i|}$ is the input, and $\mathbb{N}_i \subset \{1, 2, \ldots, n\}$ is the index set of the nodes whose information can be used by node i. The function $g : \mathbb{R}^{|\mathbb{N}_i|} \times \{0, 1, \ldots\} \to \mathbb{R}$ characterizes the system Σ and is handled as a design parameter later. We note that g does not have the subscript i. This imposes the constraint that g is common to all the nodes, ensuring the scalability of the entire system.

The network structure of the system Σ is described by the undirected graph $G = (\mathbb{V}, \mathbb{E})$, which is composed of the vertex set $\mathbb{V}$ representing the node indices and the edge set $\mathbb{E}$ describing the relation among the nodes. Then, the set $\mathbb{N}_i$ is defined as the index set of node i and its neighbors

on G. In addition, the collective state of Σ is denoted by $x(t) \coloneqq [x_1(t)\ x_2(t)\ \cdots\ x_n(t)]^\top$. Using this notation, we can consider $(G, x(t))$ as one of the graph signals explained in Section II, where G and $x(t)$ correspond to the graph and the signal values, respectively.

For the system Σ, we assume that signal values to be filtered are given as the initial state $x(0)$, as shown in Fig. 2. Then, by regarding the final state $x(\infty)$ as the filtered signal values, we can consider Σ as a (spatial) filter, whose filter characteristic is determined by the function g. Based on this, we address the following problem.

Problem 1: For the networked system Σ, design the function g (i.e., a common distributed algorithm for all the nodes) such that $x(\infty)$ has a desired spatial frequency characteristic in the sense of the graph signal $(G, x(\infty))$.

B. Main Result

To solve Problem 1, we use an approach similar to that proposed in [9]. More concretely, we focus on the relation between the behavior of the system Σ and the spatial filtering (4) of graph signals when considering $x(0)$ and $x(\infty)$ as s and $\tilde{s}$, respectively. If the transformation from $x(0)$ to $x(\infty)$ determined by the function g is equivalent to (4) for a filter function h, we say that Σ achieves DSF of the graph signal $(G, x(0))$ for h, and design g to achieve DSF.

However, it is difficult to directly obtain such a g from a given h because Σ is a dynamical system, whereas (4) does not include dynamics. Moreover, g must satisfy the constraint that one of its inputs is $[x_j(t)]_{j\in\mathbb{N}_i}$ (i.e., local information), as shown in (5). Thus, inspired by the results in [9], we assume that g provides a finite-time consensus-type algorithm, i.e.,

$$g([x_j(t)]_{j\in\mathbb{N}_i}, t) \coloneqq k(t)x_i(t) + \ell(t)\sum_{j\in\mathbb{N}_i}(x_j(t) - x_i(t)), \qquad (6)$$

where $k(t) \in \mathbb{R}\setminus\{0\}$ and $\ell(t) \in \mathbb{R}$ are time-varying gains such that $k(t) = 1$ and $\ell(t) = 0$ for $t \geq m$ and a positive integer m. Then, the algorithm for each node converges in m timesteps from (5), (6), and $k(t) = 1$ and $\ell(t) = 0$ for $t \geq m$. In addition, the algorithm is guaranteed to be common and distributed, where the former follows because $k(t)$ and $\ell(t)$ are the same for all the nodes. We now examine if the resulting Σ achieves DSF for some types of h, and if so, what filter characteristics can be obtained.

The result is summarized as the following theorem.

Theorem 1: Consider the networked system Σ incorporating the consensus-type algorithm in m timesteps given by (5) and (6). Suppose that the filter function $h(\lambda)$ of the frequency variable $\lambda \in \mathbb{R}_+ \cup \{0\}$ is given. Then, the system Σ achieves DSF of the graph signal $(G, x(0))$ for $h(\lambda)$ if and only if $h(\lambda)$ is a real polynomial with non-zero real roots, i.e.,

$$h(\lambda) \coloneqq a_m\lambda^m + a_{m-1}\lambda^{m-1} + \cdots + a_1\lambda + a_0 \qquad (7)$$

for some $a_0, a_1, \ldots, a_m \in \mathbb{R}$, whose roots $r_1, r_2, \ldots, r_m$ are non-zero real numbers.

Proof: See the Appendix. ■

Theorem 1 presents a necessary and sufficient condition for achieving DSF using (5) and (6), which gives rise to the following two facts. First, if the filter function h is given as the polynomial in (7), the system Σ given by (5) and (6) achieves DSF, where a choice of the gains $k(t)$ and $\ell(t)$ is (11) and (12), as shown in the proof given in the Appendix. Second, the filter characteristics obtained using (5) and (6) are only the ones that can be described by the polynomial in (7), which reveals the performance limit of our algorithm as the spatial filter.

Algorithm 1 Design of Distributed Spatial Filter

Step 1 Design the filter function h with a desired spatial frequency characteristic.

Step 2 Approximate the designed h by the polynomial in (7) with non-zero real roots.

Step 3 Construct the algorithm for the nodes using (5), (6), (11), (12), and the coefficient a_0 and roots of the resulting polynomial h.

Based on the above results, we present a method to solve Problem 1 in Algorithm 1. In Step 2, we obtain a polynomial h satisfying the condition in Theorem 1. In Step 3, we obtain the algorithm for the nodes from the coefficient a_0 and roots of h in addition to (5), (6), (11), and (12).

Remark 1: Theorem 1 does not require the connectivity of the graph G. Meanwhile, potential applications we have in mind, e.g., sensor networks explained in Section I, assume that the spatial frequency on graphs corresponds to that in physical space. Thus, in such applications, G should be a connected graph and its spatially neighboring nodes should be connected.

Remark 2: We now clarify the differences between this letter and [9]. In [9], the case of $k(t) \equiv 1$ was considered. As a result, the constraint $a_0 = 1$ was imposed on the filter function h in (7). This makes controlling low-frequency components more difficult because $h(0) = 1$ always holds. In contrast, we remove that constraint by introducing $k(t)$ and using it appropriately. Besides, we show that the polynomial h is both necessary and sufficient, whereas [9] only showed that the polynomial h is sufficient. Thus, our results not only provide a design method for distributed spatial filters, but they also guarantee that the designed filters use the full performance of the algorithm given by (5) and (6).

C. Example

Consider the case of $n \coloneqq 8$. The network structure G and the initial state $x(0)$ are shown in Fig. 3. In the figure, the circles and the lines connecting them represent the vertices and edges of G, respectively, and the vertical lines represent $x_1(0), x_2(0), \ldots, x_8(0)$. Here, we design a distributed spatial *high-pass* filter. One application of spatial high-pass filters is malfunction detection in sensor networks, as explained in Section I. However, from Remark 2, it is difficult to obtain high-pass filters using the method given in [9].

We choose the desired filter function as $h(\lambda) \coloneqq 1/(1 + e^{-3(\lambda-3.1)})$. This is shown by the thick blue line in Fig. 4, where the green circles show the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_8$, indicating the spatial frequencies of the components of the graph signal $(G, x(0))$. A high-pass characteristic is obtained. Using a least square fitting, we approximate this $h(\lambda)$ by a polynomial of the form (7) with non-zero real roots, which

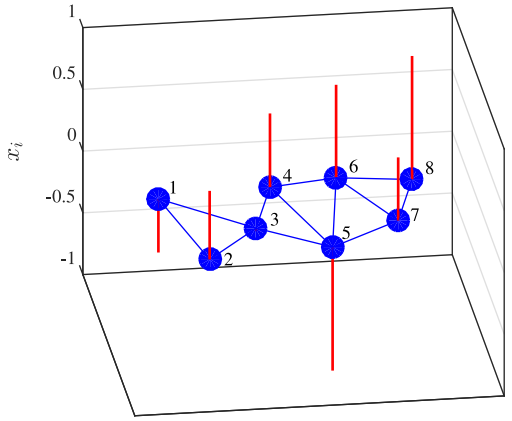


Fig. 3. Initial state $x(0)$.

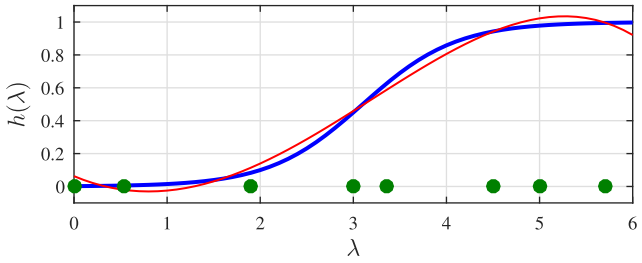


Fig. 4. Filter function $h(\lambda) := 1/(1+e^{-3(\lambda-3.1)})$ (thick blue line) and its polynomial approximation (thin red line), where the green circles denote the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_8$.

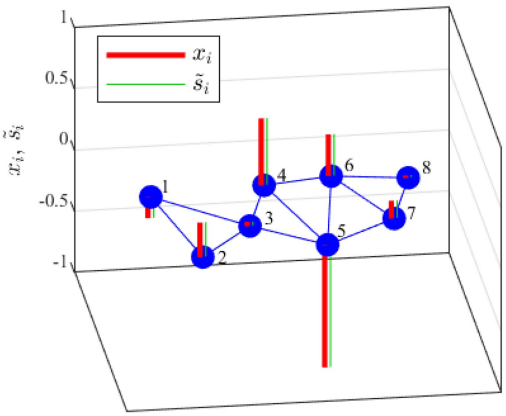


Fig. 5. Final state $x(\infty)$ obtained using our DSF method and the collective signal value $\tilde{s}$ of the graph signal filtered using (4).

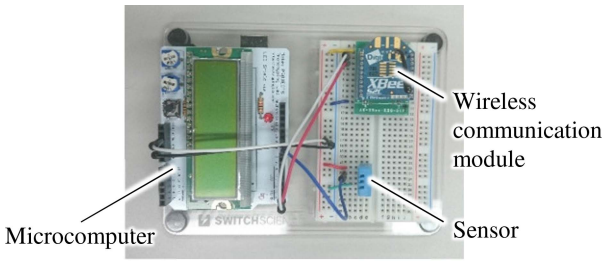


Fig. 6. Sensor node.

yields $h(\lambda) := -0.00124\lambda^4 - 0.00877\lambda^3 + 0.161\lambda^2 - 0.239\lambda + 0.0628$ with $m := 4$. This approximation is depicted by the thin red line in Fig. 4, which indicates that the approximation is close to the original $h(\lambda)$. The above polynomial has the coefficient $a_0 = 0.0628$ and the roots $r_1 = -15.9$, $r_2 = 7.26$, $r_3 = 1.29$, and $r_4 = 0.339$. Hence, these values and (5), (6), (11), and (12) give the algorithm for each node i.

The thick red lines in Fig. 5 show the resulting final state $x(\infty)$ (which is the same as $x(t)$ for $t \geq m$). The high-frequency components, where the state differences among neighboring nodes are large, are extracted. Moreover, the thin green lines show the collective signal value $\tilde{s}$ of the graph signal filtered using (4) for $s := x(0)$. Even though our filter is distributed and (4) is not, the same result is obtained in either case. This demonstrates the validity of Theorem 1.

Remark 3: In the proposed design method, the choice of the degree m of the polynomial in (7) is important. The detailed discussion on that choice was given in [9].

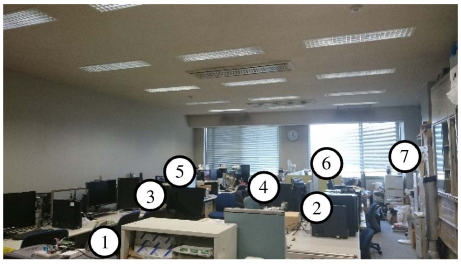
Remark 4: Although this letter examines the consensus-type algorithm given by (5) and (6), we can consider a more generalized algorithm, in order to broaden the range of the achievable filter characteristics. This will be an important direction for future research. Meanwhile, our purpose in this letter is to develop a filtering algorithm providing a wide range of filter characteristics compared to that given in [9]. The proposed algorithm can handle various filter characteristics described as polynomials, which includes high-pass ones that could not be achieved in [9]. Hence, the proposed algorithm provides sufficient performance to achieve our purpose.

IV. Experimental Verification

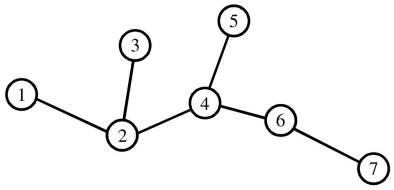
To verify the performance of our DSF method in a real setting, we developed a sensor network with seven sensor nodes. One sensor node is depicted in Fig. 6. This consists of a sensor, a microcomputer, and a wireless communication module. The sensor is the DHT11 humidity and temperature sensor [11], which has an accuracy of ± 5 %RH for humidity and ± 2 °C for temperature. The microcomputer is the Arduino Uno, in which the designed filter is installed. The wireless communication module is XBee S2C, which allows the sensor nodes to communicate with each other.

We placed these sensor nodes in a room of size 7×13 m, as shown in Fig. 7 (a), where the numbers in the figure indicate the positions of the sensor nodes. We then constructed a sensor network with the structure depicted in Fig. 7 (b) by connecting sensor nodes at spatially close locations so that the spatial frequency on graphs corresponds to that in physical space. The numbers in Fig. 7 (b) correspond to those in the photo of the setup shown in Fig. 7 (a). Using this sensor network, we measured the temperature at the locations at which the sensor nodes were placed. In such a setup, each sensor measurement was affected by factors such as sensor biases and disturbances. We call these effects spatial noise, and we attempted to reduce it using DSF.

Considering that the spatial frequencies of the noise are higher than those of the true values to be estimated, we design the filter function as $h(\lambda) := -1/(1+e^{-6(\lambda-1)}) + 1$ to reduce the amplitudes of the high-frequency components. This is depicted by the thick blue line in Fig. 8, which indicates that a low-pass characteristic is obtained. A polynomial



(a) Physical arrangement of nodes.



(b) Network structure.

Fig. 7. Experimental setup.

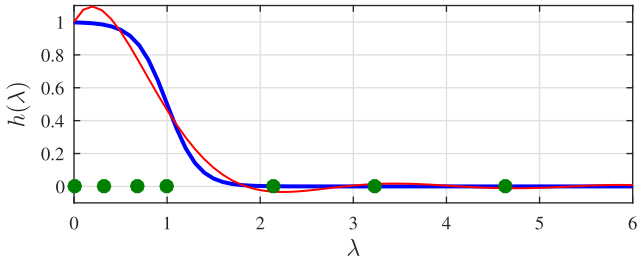


Fig. 8. Filter function $h(\lambda) := -1/(1+e^{-6(\lambda-1)})+1$ (thick blue line) and its polynomial approximation (thin red line), where the green circles denote the eigenvalues $\lambda_1, \lambda_2, \ldots, \lambda_7$.

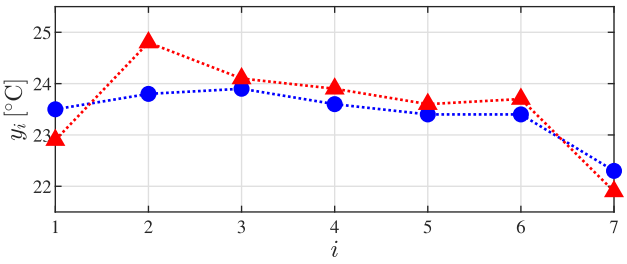


Fig. 9. Temperature measurements from our sensor network (red triangles) and a high-accuracy meter (blue circles).

approximation of $h(\lambda)$ with non-zero real roots is given by $h(\lambda) := -0.0000404\lambda^8 + 0.00163\lambda^7 - 0.0264\lambda^6 + 0.223\lambda^5 - 1.06\lambda^4 + 2.78\lambda^3 - 3.53\lambda^2 + 1.07\lambda + 1$, where $m := 8$. This approximation is depicted by the thin red line in Fig. 8, which indicates that the approximation is close to the original $h(\lambda)$. The above polynomial has the coefficient $a_0 = 1$ and the roots $r_1 = 13.3$, $r_2 = 6.90$, $r_3 = 6.31$, $r_4 = 5.34$, $r_5 = 4.15$, $r_6 = 2.90$, $r_7 = 1.85$, and $r_8 = -0.362$. Thus, using these values and (5), (6), (11), and (12), we construct the algorithm for each node i.

Fig. 9 shows an example of experimental data measured by the above-mentioned sensor network, where $y_i \in \mathbb{R}$ $(i \in \mathbb{V})$ is the temperature measurement from sensor node i. The blue circles denote the measurements from the HN-EHSP temperature/humidity meter [12] which has an accuracy of

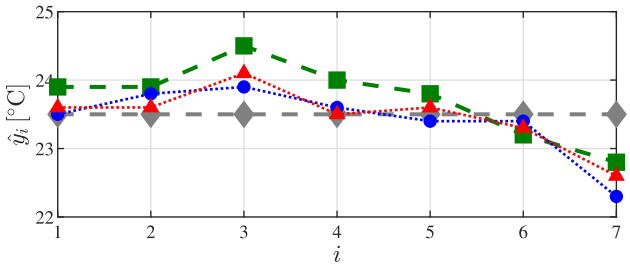


Fig. 10. Experimental results of noise reduction using our DSF method (red triangles), an average consensus algorithm (gray diamonds), and an average filter (green squares), where the true values are shown as the blue circles.

$\pm 0.5\,^{\circ}\mathrm{C}$ for temperature (much higher than that of the sensor nodes), and we assume that these are the true values to be estimated. Fig. 9 indicates that the measurements from the sensor network were different from the true values, which demonstrates the presence of the spatial noise.

For the measurements in Fig. 9, we performed an experiment to reduce the noise by applying our distributed spatial filter to $x_i(0) := y_i$ for every $i \in \mathbb{V}$. The result is shown in Fig. 10, where $\hat{y}_i \in \mathbb{R}$ $(i \in \mathbb{V})$ is the temperature estimated by sensor node i. Figs. 9 and 10 indicate that the estimates obtained using our distributed spatial filter are closer to the true values than the original measurements. Therefore, we conclude that our DSF method is applicable to a real sensor network with some constraints and uncertainties caused by hardware and wireless communications.

Moreover, in Fig. 10, we compare our DSF method with two other distributed algorithms. One is a standard average consensus algorithm [1], where the noise effects are mitigated by using the average of the measurements computed in a distributed manner. The other is an average filter [13], which is a typical technique for spatial low-pass filtering in the field of image processing. The estimate $\hat{y}_i$ via the average filter is defined as $\hat{y}_i := (y_i + \sum_{j\in\mathbb{N}_i} y_j)/(|\mathbb{N}_i|+1)$ for every $i \in \mathbb{V}$. For the vector y^* composed of the true values and $\hat{y} := [\hat{y}_1\ \hat{y}_2\ \cdots\ \hat{y}_7]^\top$, the estimation error $\|y^* - \hat{y}\|_\infty$ [°C] is 0.30 for our method, 1.20 for the average consensus algorithm, and 0.60 for the average filter. Based on this result, we conclude that our DSF method achieves higher accuracy than the other two methods.

V. Conclusion

In this letter, we developed a DSF method for networked systems to overcome the problem that an existing method can handle only limited filter characteristics. By adding a new design parameter, we improved the filtering algorithm used in the aforementioned method. For the improved algorithm, we characterized all the achievable filter characteristics. The characterization shows that our algorithm extends the range of the available filter characteristics and that its performance can be fully utilized when considering filters described as polynomials. Moreover, through denoising experiments for a real sensor network, we demonstrated that our DSF method effectively reduces spatial noise. These results are useful for efficiently

performing filtering based on the spatial structures of signals, using a distributed computation.

APPENDIX
PROOF OF THEOREM 1

First, the "only if" part is proven. We assume that the system Σ achieves DSF of the graph signal $(G, x(0))$ for a filter function h. Using (5) and (6), we can represent the collective dynamics of Σ by

$$x(t+1) = (k(t)I - \ell(t)L)x(t). \tag{8}$$

This yields

$$\begin{aligned} x(m) &= \left(\prod_{t=0}^{m-1}(k(m-1-t)I - \ell(m-1-t)L)\right)x(0) \\ &= \left(\prod_{t=0}^{m-1}\left(k(m-1-t)VV^\top - \ell(m-1-t)V\Lambda V^\top\right)\right)x(0) \\ &= \left(\prod_{t=0}^{m-1} V(k(m-1-t)I - \ell(m-1-t)\Lambda)V^\top\right)x(0) \\ &= V\left(\prod_{t=0}^{m-1}(k(m-1-t)I - \ell(m-1-t)\Lambda)\right)V^\top x(0), \end{aligned} \tag{9}$$

where the first equality is obtained from (8), the second one is given by $V^\top = V^{-1}$ and (2), the third one holds because $k(t)$ and $\ell(t)$ are scalars, and the last one is given by $V^\top V = I$. Because $k(m-1-t)I - \ell(m-1-t)\Lambda$ in (9) is a diagonal matrix for every $t \in \{0, 1, \ldots, m-1\}$, it follows that

$$\begin{aligned} x(m) = V\mathrm{diag}\Bigg(&\prod_{t=0}^{m-1}(k(m-1-t) - \ell(m-1-t)\lambda_1), \\ &\prod_{t=0}^{m-1}(k(m-1-t) - \ell(m-1-t)\lambda_2), \ldots, \\ &\prod_{t=0}^{m-1}(k(m-1-t) - \ell(m-1-t)\lambda_n)\Bigg)V^\top x(0). \end{aligned} \tag{10}$$

In addition, the algorithm for each node i converges in m timesteps, as mentioned in Section III-B, and thus $x(m) = x(\infty)$ holds. Applying this to (10) and regarding $x(0)$ and $x(\infty)$ as s and $\tilde{s}$ in (4), respectively, we obtain $h(\lambda) = \prod_{t=0}^{m-1}(k(m-1-t) - \ell(m-1-t)\lambda)$ because Σ achieves DSF of the graph signal $(G, x(0))$ by our assumption. From $k(t) \in \mathbb{R}\setminus\{0\}$ and $\ell(t) \in \mathbb{R}$ for every $t \in \{0, 1, \ldots, m-1\}$, this $h(\lambda)$ is a polynomial of the form (7) and its roots are non-zero real numbers. This proves the "only if" part.

Next, the "if" part is proven. We assume that the filter function h in (7) with non-zero real roots is given. Let

$$k(t) := \begin{cases} a_0 & \text{if } t = 0, \\ 1 & \text{otherwise}, \end{cases} \tag{11}$$

$$\ell(t) := \begin{cases} \dfrac{a_0}{r_1} & \text{if } t = 0, \\ \dfrac{1}{r_{t+1}} & \text{if } t \in \{1, 2, \ldots, m-1\}, \\ 0 & \text{otherwise}. \end{cases} \tag{12}$$

These gains satisfy the condition for the convergence of the algorithm given by (5) and (6) in m timesteps, i.e., $k(t) = 1$ and $\ell(t) = 0$ for $t \geq m$. Substituting (11), (12), and $x(m) = x(\infty)$ for (10) yields

$$\begin{aligned} x(\infty) = V\mathrm{diag}\Bigg(&\left(\prod_{t=0}^{m-2}\left(1 - \frac{\lambda_1}{r_{m-t}}\right)\right)\left(a_0 - \frac{a_0\lambda_1}{r_1}\right), \\ &\left(\prod_{t=0}^{m-2}\left(1 - \frac{\lambda_2}{r_{m-t}}\right)\right)\left(a_0 - \frac{a_0\lambda_2}{r_1}\right), \ldots, \\ &\left(\prod_{t=0}^{m-2}\left(1 - \frac{\lambda_n}{r_{m-t}}\right)\right)\left(a_0 - \frac{a_0\lambda_n}{r_1}\right)\Bigg)V^\top x(0) \\ = V\mathrm{diag}\Bigg(&a_0\prod_{t=0}^{m-1}\left(1 - \frac{\lambda_1}{r_{m-t}}\right), a_0\prod_{t=0}^{m-1}\left(1 - \frac{\lambda_2}{r_{m-t}}\right), \\ &\ldots, a_0\prod_{t=0}^{m-1}\left(1 - \frac{\lambda_n}{r_{m-t}}\right)\Bigg)V^\top x(0). \end{aligned} \tag{13}$$

Hence, in a way similar to the discussion above, the system Σ achieves DSF of the graph signal $(G, x(0))$ for h in (7) if

$$h(\lambda_p) = a_0\prod_{t=0}^{m-1}\left(1 - \frac{\lambda_p}{r_{m-t}}\right) \quad \forall p \in \{1, 2, \ldots, n\}. \tag{14}$$

The relation (14) resembles that shown in [9, Appendix A], and thus the proof is omitted. This completes the proof of the "if" part, which proves Theorem 1.

REFERENCES

[1] R. Olfati-Saber, J. A. Fax, and R. M. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[2] Y. Wang and H. Ishii, "Resilient consensus through event-based communication," *IEEE Trans. Control Netw. Syst.*, vol. 7, no. 1, pp. 471–482, Mar. 2020.

[3] J. A. Fax and R. M. Murray, "Information flow and cooperative control of vehicle formations," *IEEE Trans. Autom. Control*, vol. 49, no. 9, pp. 1465–1476, Sep. 2004.

[4] H. Liu, T. Ma, F. L. Lewis, and Y. Wan, "Robust formation control for multiple quadrotors with nonlinearities and disturbances," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1362–1371, Apr. 2020.

[5] A. Sandryhaila and J. M. F. Moura, "Discrete signal processing on graphs: Frequency analysis," *IEEE Trans. Signal Process.*, vol. 62, no. 12, pp. 3042–3054, Jun. 2014.

[6] D. I. Shuman, P. Vandergheynst, and P. Frossard, "Chebyshev polynomial approximation for distributed signal processing," in *Proc. Int. Conf. Distrib. Comput. Sensor Syst. Workshops*, 2011, pp. 1–8.

[7] S. Segarra, A. G. Marques, and A. Ribeiro, "Optimal graph-filter design and applications to distributed linear network operators," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4117–4131, Apr. 2017.

[8] J.-W. Yi, L. Chai, and J. Zhang, "Average consensus by graph filtering: New approach, explicit convergence rate and optimal design," *IEEE Trans. Autom. Control*, vol. 65, no. 1, pp. 191–206, Jan. 2020.

[9] S. Izumi, S.-I. Azuma, and T. Sugie, "Analysis and design of multi-agent systems in spatial frequency domain: Application to distributed spatial filtering in sensor networks," *IEEE Access*, vol. 8, pp. 34909–34918, Feb. 2020.

[10] 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, May 2013.

[11] *DHT11 Humidity & Temperature Sensor, DHT11, Mouser Electronics*, Bengaluru, India, Accessed: Mar. 16, 2020. [Online]. Available: https://www.mouser.com/datasheet/2/758/DHT11-Technical-Data-Sheet-Translated-Version-1143054.pdf

[12] CHINO. (Nov. 18, 2013). *HN-EH Series Palm-Sized Temperature/Humidity Meter, HN-EHSP*. Accessed: Mar. 16, 2020. [Online]. Available: https://www.chino.co.jp/english/products/humidity/hn-eh-series-palm-sized-temperature-humidity-meter/

[13] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2001.