Jurnal Elektronika dan Telekomunikasi (JET), Vol. 22, No. 2, December 2022, pp. 72-77
Accredited by KEMDIKBUDRISTEK, Decree No: **158/E/KPT/2021**
doi: 10.55981/jet.490

Cooperative Line Formation Control of Multi-Agent Systems Based on Least Squares Estimation

Samratul Fuady[a, *], Arumjeni Mitayani[b], Ario Birmiawan Widyoutomo, Arief Suryadi Satyawan[b], Alexander Christantho Budiman[c], Suyoto Suyoto[b], Mochamad Mardi Marta Dinata[b]

[a]Departement of Electrical Engineering
Jambi University
Jl. Jambi - Muara Bulian No. KM. 15, Mendalo Darat, Muaro Jambi
Jambi, Indonesia
[b]Research Center for Telecommunication
National Research and Innovation Agency
Komplek BRIN Jl. Sangkuriang No. 21
Bandung, Indonesia
[c]Research Center for Transportation Technology
National Research and Innovation Agency
Komplek BRIN Jl. Sangkuriang No. 21
Bandung, Indonesia

Abstract

In this paper, we consider the problem of multi-agent systems where each agent aims to establish a line formation in a distributed manner. In constructing an efficient line formation, finding a line with the closest total distance from every agent is essential. We propose a formation control using least squares estimation (LSE) performed by each agent with only the local information that consists of the corresponding agent's and neighbors' positions. Each agent calculates the local cost function, which is the squared distance from the LSE line to the related agent's and its neighbors' positions. Our goal is to minimize the global cost function, which is the sum of these local cost functions. To achieve this, we employ distributed optimization to the global cost function of the overall system using the subgradient method performed by each agent locally. We evaluate our proposed method using numerical simulation, and the result complies with our goal of this paper.

Keywords: LSE, formation control, distributed optimization, multi-agent systems.

I. INTRODUCTION

Lately, the tasks performed by autonomous agents are becoming more complicated. Forcing an individual agent to achieve a complex mission is costly and impractical, and sometimes impossible. For example, in logistic transportation [1], using an individual agent to deliver heavy and numerous objects will require massive and expensive equipment. Meanwhile, if the goods are split and transported by multiple agents, each agent will only need a simple requirement that is easier to implement. Consequently, research on multi-agent systems has received much attention in recent years. Multi-agent systems have been deployed in various applications, such as military [2], air traffic control [3], autonomous vehicle [4], satellite [5], communication [6], and logistic transportation, as mentioned previously. There are multi-agent systems approaches: centralized and distributed [7]. The centralized approach, there is one centralized unit or agent acting as a leader that can oversee and direct all other agents in the system. Meanwhile, in the distributed approach, there is no hierarchical structure amongst the agents, i.e., all agents have the same level of autonomy and make decision based on their local information. In this paper, we use the latter approach because in the centralized approach, the system will potentially collapse if the central agent malfunctions. The multiple agents in our system share the information with the adjacent agents, called neighbors. The agents can share the information containing the parameter of interest, for example, location, sensor reading, and other parameters depending on the application. Multi-agent systems working cooperatively under a particular coordination scheme have many advantages over single-agent systems, such as reducing complexity and availability of redundancy.

There are some issues regarding multi-agent systems: coverage, consensus, navigation, and formation control [8], however, this paper will focus on formation control, and the other issues are out of this paper's scope. Formation control is adapting the formation behavior of nature [9], such as schools of fish, flocks of birds, the swarm of ants, and many other animals staying in formation, as well as pedestrian behavior in humans [10]. In this case, animal formation has many benefits, such as conserving energy, retreating from predators, and keeping connectedness. Similarly,

* Corresponding Author.
Email: sfuady@unja.ac.id
Received: August 14, 2022 ; Revised: September 12, 2022
Accepted: November 15, 2022 ; Published: December 31, 2022

the agents in multi-agent systems also adapt those animals' behavior in making a formation. The benefits as mentioned above are what agents in the systems are expected to achieve. Thus, with formation control, the agents can save their energy, increase systems' robustness and efficiency , and keep their connectedness [11].

One of the most used structures of formation control is the line-based formation which will be referred to as line formation because of its simplicity and applicability in a lot of applications. For example, in military applications, military robots need to form platoon lines to execute missions efficiently for both attacking and defending while keeping communication intact. In this formation, the agent must create and maintain line arrangements during their mission. There are several approaches to forming a formation in multi-agent systems, such as distance-based [12], virtual structure [13], etc. In this paper, to obtain the line for achieving the desirable formation, we propose least squares estimation (LSE) based line formation control as it is intuitively the most efficient way to obtain a line with the shortest total distance from every agent. In application, it will imply the energy and time efficiency of the proposed system. To the best of the authors' knowledge, this paper is the first to propose LSE as the approach for line formation control.

LSE is usually performed when we have all the data points. However, in this case, each agent will need to perform LSE based on its local data, which can generate different results from each agent. Thus, in this paper, we use distributed optimization [14] to have all the agents agree on the optimal line based on the local data.

This paper is organized as follows. We describe the problem formulations in Section II. Then, we explain the proposed algorithm in section III. In section IV, we show the simulation result of the proposed algorithm. We finally conclude our work in Section V.

II. PROBLEM FORMULATION

In this section, we describe the setup of the problem and provide some notations related to the problem set. Furthermore, we also elaborate on the LSE we use in the problem.

A. Graph Notions

Consider a multi-agent system with n agents which interact with each other through a network described as an undirected graph $\mathcal{G} = (\mathcal{V}, \varepsilon)$ where $\mathcal{V} = \{1,2,3, \dots, n\}$ is the node set and $\varepsilon \subset \mathcal{V} \times \mathcal{V}$ is the undirected edge set. The edge $(i,j) \in \varepsilon$ means that agents i and j can exchange information (or i and j are neighbor). For each node i, we denote its neighbor set by $\mathcal{N}_i \subset \mathcal{V}$ and total number of neighbors of agent i is written as $|\mathcal{N}_i|$.

The adjacency matrix $\mathcal{A} = [a_{ij}] \in \mathbb{R}^{n \times n}$, which describes connection between agents of graph $\mathcal{G}$, is defined as $a_{ij} = a_{ji} > 0$ if $(i,j) \in \varepsilon$ where $i \neq j$ and $a_{ij} = 0$ otherwise. The Laplacian matrix $\mathcal{L} = [l_{ij}] \in \mathbb{R}^{n \times n}$, which is the representation of graph $\mathcal{G}$, is defined as $l_{ii} = \sum_{i \neq j} a_{ij}$ and $l_{ij} = -a_{ij}$ where $i \neq j$. A square matrix $\mathcal{P}$ is called a stochastic matrix when its components p_{ij} are nonnegative and each sum of the row of $\mathcal{P}$ is equal to 1, and it is doubly stochastic when both $\mathcal{P}$ and $\mathcal{P}^T$ are stochastic matrices.

The dynamics of each agent are described as discrete-time single-integrator dynamics, which is shown in (1),

$$r_i(k+1) = r_i(k) + u_i(k), \quad i = 1,2, \dots, n. \tag{1}$$

where $r_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \in \mathbb{R}^2$ represent the agents' position in two-dimensional plane, and u_i is the control signal of agent i.

B. Least Squares Estimation (LSE)

We consider the problem when each agent performs LSE based on its local information. Our parameter of interest in this case is the agent's position on the plane, which is given by $r_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$. Each agent can only communicate its position to its neighbors.

Consider each agent has dataset $\widetilde{x_i} \in \mathbb{R}^{|\mathcal{N}_i|+1}$ consisting of x_i and $x_j, j \in \mathcal{N}_i$ and $\widetilde{y_i} \in \mathbb{R}^{|\mathcal{N}_i|+1}$ consisting of y_i and $y_j, j \in \mathcal{N}_i$. We can see that each agent's dataset contains its own position and its neighbors' positions. For LSE problem, we define $\widetilde{X_i} = [\widetilde{x_i} \quad \mathbf{1}]$ where $\mathbf{1} = [1 \quad \dots \quad 1]^T \in \mathbb{R}^{|\mathcal{N}_i|+1}$. Suppose $\widetilde{y_i}$ is an m vector and $\widetilde{X_i}$ an $m \times n$ matrix with linearly independent columns. Then there is a unique n vector $\widehat{\gamma_i}$ which minimizes $\left\| \widetilde{y_i} - \widetilde{X_i} \gamma_i \right\|$ over all γ_i (the norm taken as the Euclidean m-space norm). In our multi-agent system, m is equal to $|\mathcal{N}_i| + 1$ and with n is equal to 2, the LSE will result first degree polynomial, which is a line. Another value of n, for example, 3 or more, will result second and higher degree polynomial, which is not a straight line we are looking for. The analytical γ_i which is $\widehat{\gamma_i}$ can be calculated as (2).

$$\widehat{\gamma_i} = \left(\widetilde{X_i}^T \widetilde{X_i} \right)^{-1} \widetilde{X_i}^T \widetilde{y_i} \tag{2}$$

From (2) we can have the LSE parameter for each agent. For illustration, we have six agents with the positions and network topology shown in Figure 1. The agents' positions are represented in labeled dots and the black lines that connect one agent to another agent represent network topology. The adjacency matrix in this case is given by (3).

$$\mathcal{A} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \tag{3}$$

As we can see, agent A has three neighbors: agent B, agent C, and agent D. Based on its local information, which is the position of its three neighbors and itself, agent A performs LSE, and the result is the red line. Similarly, agent B, which has four neighbors: agent A, agent C, agent D, and agent F, performs LSE and result the yellow line. This process is also performed by other agents in the system simultaneously. The resulting LSE for each agent is shown in corresponding color in Figure 1.

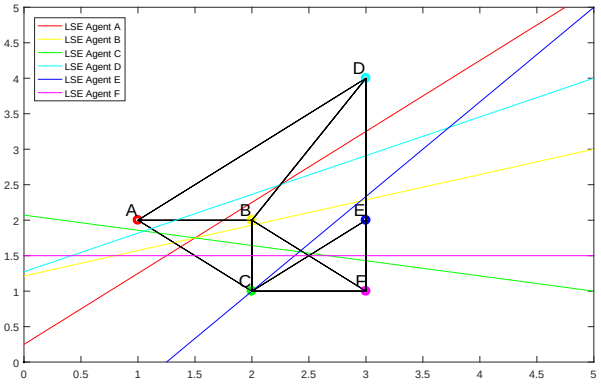

Figure 1. LSE calculated by each agent based on its local information.

Clearly, we can see that the resulting line is different from one agent to the other agents. This is because the total information that is received by each agent is different depending on its neighbors. Our goal is to have all agents agree on the same line, which will be the base line for the formation.

III. Distributed Optimization

In this part, we will describe the method we develop to solve the problem mentioned in the previous section.

A. Distributed Subgradient Method

A distributed algorithm for minimizing the sum of convex functions has been introduced in [14]. In this method, each agent executes the update rule that consists of the consensus step and subgradient step, as given in (4).

$$x_i(k+1) = \sum_{j\in\mathcal{V}} p_{ij}x_j(k) - sd_i(k) \tag{4}$$

where $x_i(k)$ is the position of agent i at time k, p_{ij} is the element of Perron Matrix, a doubly stochastic matrix corresponding to the network topology, s is the stepsize of gradient (or subgradient) descent step, and $d_i(k)$ is the gradient (or subgradient) of the cost function J_i computed at $x_i(k)$. This algorithm is proven to converge to minimizer as long as the graph is connected with the convergence rate given by (5) (see [14] Theorem 1.3).

$$J\left(\widehat{x_i}(k)\right) \le J^* + \frac{sD^2C}{2} + \frac{4nD}{k\beta(1-\beta)}\sum_{j\in\mathcal{V}}\|x_{j0}\| + \frac{n}{2sk}(dist(\alpha, X^*) + sD)^2 \tag{5}$$

where J^* is the optimum cost function, X^* is the optimal solution, $\widehat{x_i}(k) = \frac{1}{k}\sum_{\tau=1}^{k} x_i(\tau)$, D is bound for subgradient, n is number of agents, $C = 1 + 8n(2 + \frac{n}{\beta(1-\beta)})$, $\beta = 1 - \frac{\eta}{4n^2}$, η is lower bound for $p_{ij} > 0$, and $\alpha = \frac{1}{n}\sum_{i=1}^{n} x_i(0)$.

When the number of iterations goes to infinity, the last two terms of (5) become zero. We can see that the accuracy of this algorithm depends on the step size, the upper bound of the cost function gradient, and also the number of agents.

B. Proposed Cost Function

We introduce the local cost function for agent i as the residue from the LSE. The residue is the vertical distance between the agents and the LSE line, which will be referred to as LSE residue as shown in Figure 2 with the example of agent A and its neighbors: agent B, agent C, and agent D, identical to the example we mention in subsection II.B. Local cost function is then calculated as the sum of LSE residue of corresponding agent and its neighbors with respect to the LSE line which can be calculated as (6).

$$J_i(x,y) = \left\|\tilde{y}_i - \tilde{X}_i\hat{y}_i\right\|^2$$
$$J_i(x,y) = \left\|\left[I - \tilde{X}_i\left(\tilde{X}_i^T\tilde{X}_i\right)^{-1}\tilde{X}_i^T\right]\tilde{y}_i\right\|^2 \tag{6}$$

Note that the local cost function will become zero, i.e., $J_i(x,y) = 0$ when the corresponding agent and its neighbors are forming a line, as shown in Figure 3 for the case of agent A.

To optimize the multi-agent system, we need to minimize the cost function of all agents, which is the sum of the local cost function from each agent in (6) as described in (7).

$$J(x,y) = \sum_{i=1}^{n} J_i(x,y) \tag{7}$$

When all agents in (7) are aligned, as shown in Figure 4, we find the total cost function, $J(x,y) = 0$. So, by solving the optimization problem for (7), we will achieve a line formation based on LSE, i.e., LSE-based agreement of all agents in the system. Furthermore, because each agent performs the local calculation not in a centralized manner, we can solve it using distributed optimization method.

To solve this distributed optimization problem, we need to investigate the convexity of our cost function. Let us first consider the local cost function in (6) for an agent having three information (2 from its neighbors and one from itself), which is denoted as $J_1(x,y)$ and can be calculated in (8).

$$J_1(x,y) = \frac{(x_1y_2 - x_2y_1 - x_1y_3 + x_3y_1 + x_2y_3 - x_3y_2)}{(x_1-x_2)^2 + (x_2-x_3)^2 + (x_3-x_1)^2} \tag{8}$$

where (x_1, y_1) denotes the position of the corresponding agent itself, while (x_2, y_2) and (x_3, y_3) represent the positions of its neighbors. The cases for a bigger number of information can be generalized from (8). It is important to note that this is the minimum number of local information because if the agent does only have two pieces of information (from itself and one neighbor), these two points will be a line, i.e., the cost function will be zero at all times, which is not enough to create an agreement with other agents. In the case when

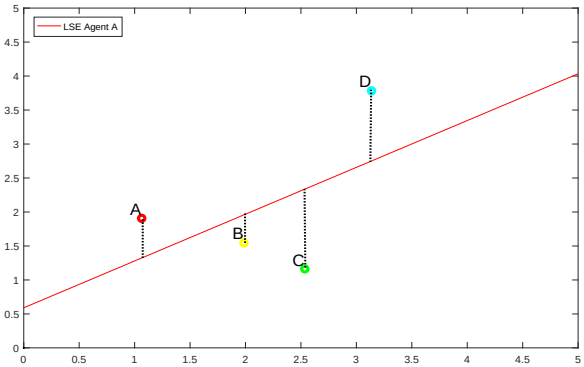

Figure 2. The distance from agent A and its neighbors to LSE line performed by agent A.

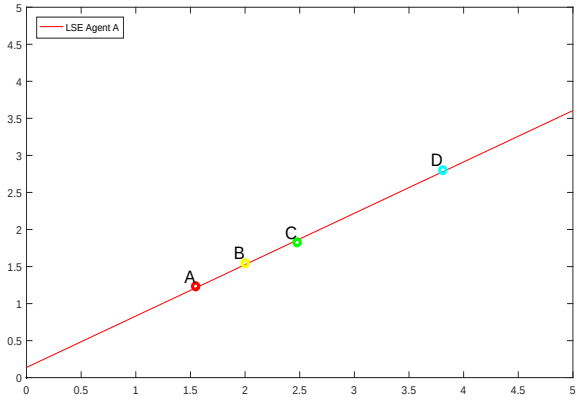

Figure 3. Case when agent A and its neighbors are forming a line.

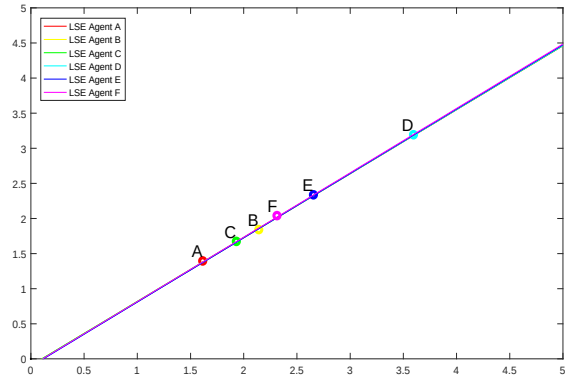

Figure 4. Case when all agents in the system are forming a line.

the agent has only one piece of information (only from itself), it means the agent is disconnected from the group, which also makes agreement impossible. So, the necessary condition for each agent is to have at least two neighbors.

As we analyzed the Hessian matrix of (8), we found that it is not a positive semidefinite matrix for all (x, y), which implies that the cost function (8) is not a convex function. Moreover, due to its denominator, this cost function is also difficult to analyze. So, we propose some modifications to this local cost function without losing the objective of the cost function itself. We can obtain the new cost function in (9) by multiplying the cost function with its denominator.

$$\hat{J}_1(x,y) = (x_1y_2 - x_2y_1 - x_1y_3 + x_3y_1 + x_2y_3 - x_3y_2)^2 \quad (9)$$

We can see that $\hat{J}_1(x,y) = 0$ if and only if the agent with its neighbors forms a line, which suits our objective. As we analyze the Hessian matrix of (9), this new cost function still does not have a positive semidefinite Hessian matrix for all (x, y), which means it is not a convex function. But it has some properties that are very useful for us.

We first look at the gradient of $\hat{J}_1(x,y)$ with respect to x_1 as written in (10).

$$\frac{\partial \hat{J}_1(x,y)}{\partial x_1} = 2(x_1y_2 - x_2y_1 - x_1y_3 + x_3y_1 + x_2y_3 - x_3y_2)(y_2 - y_3) \quad (10)$$

Equation (10) becomes zero in two cases if the agent and its neighbors are on a line or if $y_2 = y_3$. By evaluating the gradient of $\hat{J}_1(x,y)$ with respect to other variables, e.g., x_2, x_3, y_1, y_2, and y_3, we can see that all the gradient becomes zero if and only if the agent and its neighbor form a line or they are in the same point ($x_1 = x_2 = x_3$ and $y_1 = y_2 = y_3$) which can also be considered as forming a line.

According to Fermat's theorem, we can infer that from these findings, for local cost function (9), there is no local minimum except for the case when a line is formed by the agent and its neighbors. Moreover, there always exists a path from one local minimum to another local minimum where agents keep aligned, i.e., stay in the local minimum. This implies the local minimum is next to each other, creating a minimum local area. So, the local cost function (9) is monotonically decreasing until minimum local area and then monotonically increasing afterward, which is close to the convex function.

Following from the case of three information, the cost function for agents with bigger number of neighbors can be generalized as (11).

$$\hat{J}_i(x,y) = \left(\sum_{\substack{p,q \in N_i \cup i \\ p \neq q}} (x_p - x_q)^2 \right) \left\| \left[I - \tilde{X}_i \left(\tilde{X}_i^T \tilde{X}_i \right)^{-1} \tilde{X}_i^T \right] \tilde{y}_i \right\|^2 \quad (11)$$

Consequently, the global cost function to be minimized is (12), which is the sum of local cost function (11).

$$J(x,y) = \sum_{i=1}^{n} \hat{J}_i(x,y) \quad (12)$$

C. Gradient-Based Algorithm

Based on the cost function in (12), we design an algorithm to minimize the global cost function. It is clear that the minimum value of the global cost function is 0 and it happens only when all agents form a line. Because the local cost function is a non-negative

function, this condition will be achieved when all local cost function in (11) is zero, which is when all agents form a line with their own neighbors. With the properties of the local cost function described in subsection III.B, optimization can be done by minimizing each local cost function by each agent using the subgradient descent method as described in subsection III.A.

Consider the distributed subgradient update rule in (13),

$$u_i(k) = -s\begin{bmatrix}\frac{\Delta\hat{J}_i}{\Delta x_i}\\ \frac{\Delta\hat{J}_i}{\Delta y_i}\end{bmatrix} \tag{13}$$

where $\Delta\hat{J}_i = \hat{J}_i(x(k), y(k)) - \hat{J}_i(x(k-1), y(k-1))$, $\Delta x_i = x_i(k) - x_i(k-1)$, $\Delta y_i = y_i(k) - y_i(k-1)$ and $\hat{J}_i(x(k), y(k))$ can be obtained from (11) with $(x(k), y(k))$ is the instantaneous position at time step k.

The update rule in (13) can be easily applied in each agent locally based on agents' dynamics in (1). Note that the update rule in (13) does not consider the distance between agents. This problem will need further proof and analysis, which is out of the scope of this paper. We provide the simulation results of this in the next section.

IV. Simulation Results

In this section, we evaluate the proposed methods through a simulation. We examine the update rule (13) for the case of six agents with network topology shown in Figure 5. We can see that the graph is connected, and every agent has at least two neighbors. The adjacency matrix for this case is already written in (2). The initial position in the two-dimensional plane is $r(0) = \begin{bmatrix}1 & 2 & 2 & 1 & 3 & 3\\ 2 & 2 & 1 & 4 & 2 & 1\end{bmatrix}$. We set the step size parameter s = 0.05.

From Figure 6(a), we can see that, each agent has different LSE at the beginning as shown in colored lines. After we run the algorithm in each agent, we can see that the LSEs from each agent are getting closer (Figure 6(b)-(c)) and finally coincide into one line, i.e., all agents agree in the same LSE and establish a line formation as shown in Figure 6(d). The local cost function for each agent ($\hat{J}_i$) and the global cost function (J) is shown in Figure 7 and Figure 8, respectively. By applying the update rule in (13), we can see that the cost function is consistently goes down and goes to minimum after 34-time steps.

We also test this algorithm in constructing line formation for source-seeking missions. The agents are placed in the signal field area and need to locate their source. The initial positions of agents are shown in Figure 9.

We combine our LSE-based line formation control with the source seeking and obstacle avoidance control from [15], and the resulting trajectory is shown in Figure 10. As we can see, the multi-agent systems successfully locate the source of signal field and maintain the line formation during the mission.

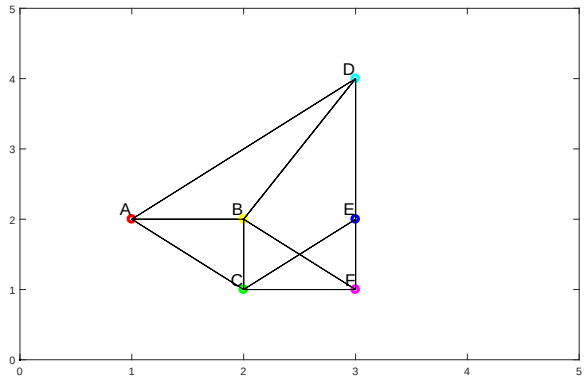

Figure 5. Network topology and positions of the agents.

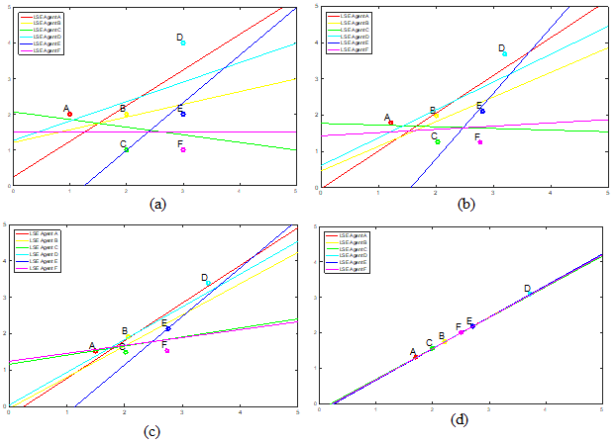

Figure 6. Agents' position and its corresponding LSE at (a) initial condition; (b) 10-time steps; (c) 20-time steps; and (d) 40-time steps.

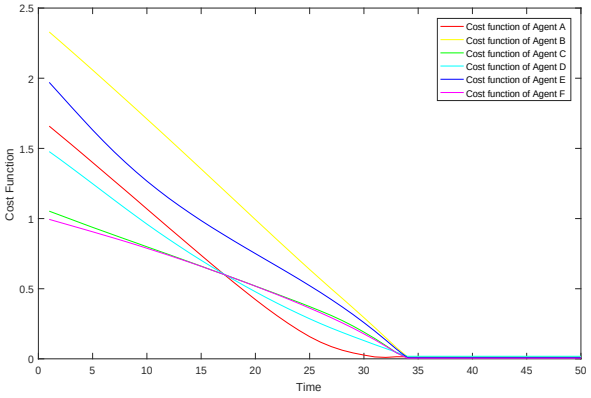

Figure 7. Local cost function of each agent.

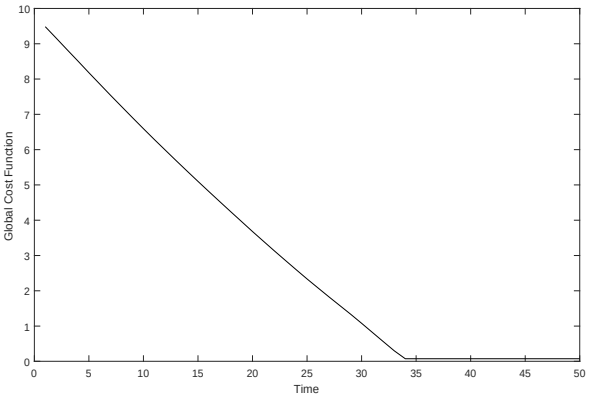

Figure 8. Global cost function.

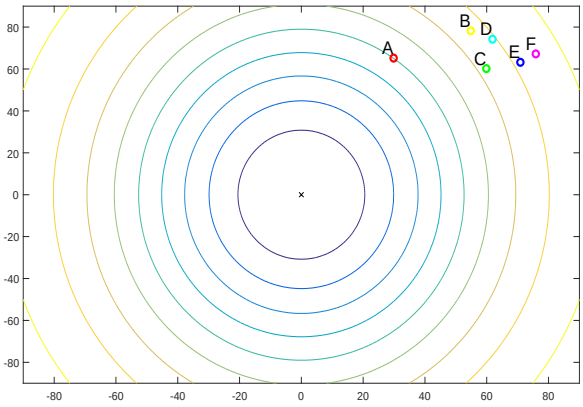

Figure 9. Initial position of agents in source-seeking

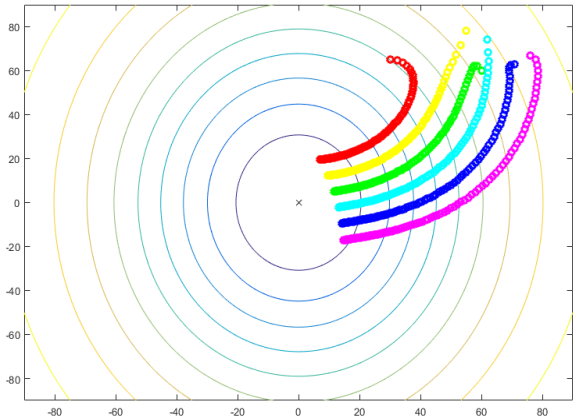

Figure 10. Trajectory of agents doing source seeking in line formation

V. CONCLUSION

In this paper, we have described the problem of multi-agent formation control. We constructed local cost function based on the residue of least squares estimation, which each agent performs. The global cost function, which is the sum of all agents' local cost functions, is then used in distributed optimization. We proposed a subgradient algorithm to minimize the local cost function of each agent. We validated our approach using numerical simulation. It is shown that the algorithm successfully created line formation given the graph is connected and each agent has at least two neighbors. We also tested this algorithm in a multi-agent source-seeking missions, and it kept the agents in line formation accordingly. We believe our result is a promising approach to be applied in various multi-agent missions requiring line formation.

DECLARATIONS

Conflict of Interest

The authors have declared that no competing interests exist.

CRediT Authorship Contribution

Samratul Fuady and Arumjeni Mitayani are the main contributors of this paper. Samratul Fuady: Conceptualization, Visualization, Writing - Original Draft, Writing - Review & Editing; Arumjeni Mitayani: Conceptualization, Visualization, Writing - Original Draft, Writing - Review & Editing; Ario Birmiawan Widyoutomo, Arief Suryadi Satyawan, Alexander Christantho Budiman, Suyoto, and Mochamad Mardi Marta Dinata: Conceptualization, Writing - Review & Editing.

Funding

Research reported in this publication was supported by DIPA BRIN and LPDP.

REFERENCES

[1] N. Jabeur, T. Al-Belushi, M. Mbarki, and H. Gharrad, "Toward leveraging smart logistics collaboration with a multi-agent system based solution," *Procedia Comput. Sci.*, vol. 109, no. 2016, pp. 672–679, 2017, doi: 10.1016/j.procs.2017.05.374.

[2] N. R. Gans and J. G. Rogers, "Cooperative multirobot systems for military applications," *Curr. Robot. Reports*, vol. 2, no. 1, pp. 105–111, 2021, doi: 10.1007/s43154-020-00039-w.

[3] A. Degas, E. Kaddoum, M. P. Gleizes, F. Adreit, and A. Rantrua, "Cooperative multi-agent model for collision avoidance applied to air traffic management," *Eng. Appl. Artif. Intell.*, vol. 102, no. May, p. 104286, 2021, doi: 10.1016/j.engappai.2021.104286.

[4] S. Chen, Y. Leng, and S. Labi, "A deep learning algorithm for simulating autonomous driving considering prior knowledge and temporal information," *Comput. Civ. Infrastruct. Eng.*, vol. 35, no. 4, pp. 305–321, 2020, doi: 10.1111/mice.12495.

[5] S. He, T. Wang, and S. Wang, "Load-aware satellite handover strategy based on multi-agent reinforcement learning," *2020 IEEE Glob. Commun. Conf. GLOBECOM 2020 - Proc.*, 2020, doi: 10.1109/GLOBECOM42002.2020.9322449.

[6] L. Wang, K. Wang, C. Pan, W. Xu, N. Aslam, and L. Hanzo, "Multi-agent deep reinforcement learning-based trajectory planning for multi-uav assisted mobile edge computing," *IEEE Trans. Cogn. Commun. Netw.*, vol. 7, no. 1, pp. 73–84, 2021, doi: 10.1109/TCCN.2020.3027695.

[7] F. Chen and W. Ren, "On the control of multi-agent systems: a survey," *Found. Trends Syst. Control*, vol. 6, no. 4, pp. 1–164, 2019, doi: 10.1561/260-000-0019.

[8] M. M. Gulzar, S. T. H. Rizvi, M. Y. Javed, U. Munir, and H. Asif, "Multi-agent cooperative control consensus: a comparative review," *Electron.*, vol. 7, no. 2, 2018, doi: 10.3390/electronics7020022.

[9] H.-S. Ahn, *Formation Control: Approaches for Distributed Agents*. 2019. doi: 10.1007/978-3-030-15187-4.

[10] Y. P. Pane, S. Fuady, and K. Mutijarsa, "Overtaking in centralized multi robot formation control based on pedestrian behavior," *Proc. - UKSim 15th Int. Conf. Comput. Model. Simulation, UKSim 2013*, pp. 271–276, 2013, doi: 10.1109/UKSim.2013.146.

[11] H. Chu, J. Chen, D. Yue, and C. Dou, "Observer-based consensus of nonlinear multiagent systems with relative state estimate constraints," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 7, pp. 2456–2465, 2020, doi: 10.1109/TSMC.2018.2818172.

[12] Y. H. Choi and D. Kim, "Distance-based formation control with goal assignment for global asymptotic stability of multi-robot systems," *IEEE Robot. Autom. Lett.*, vol. 6, no. 2, pp. 2020–2027, 2021, doi: 10.1109/LRA.2021.3061071.

[13] S. Fuady, A. R. Ibrahim, and B. R. Trilaksono, "Comparative experimental study of formation control of mobile robots," *Procedia Technol.*, vol. 11, no. Iceei, pp. 689–695, 2013, doi: 10.1016/j.protcy.2013.12.246.

[14] A. Nedić and A. Ozdaglar, *Cooperative distributed multi-agent optimization*, vol. 9780521762. 2009. doi: 10.1017/CBO9780511804458.011.

[15] W. Wu, I. D. Couzin, and F. Zhang, "Bio-inspired source seeking with no explicit gradient estimation," in *Proc. 3rd IFAC Work. Distrib. Estim. Control Networked Syst.*, 2012, vol. 45, no. 26, pp. 240–245. doi: 10.3182/20120914-2-US-4030.00024.