TITLE:
Leader–Follower Navigation in Obstacle Environments While Preserving Connectivity Without Data Transmission

AUTHOR(S):
Sakai, Daito; Fukushima, Hiroaki; Matsuno, Fumitoshi

CITATION:

ISSUE DATE:
2018-07

URL:
http://hdl.handle.net/2433/232584

RIGHT:
© 2017 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works. This is not the published version. Please cite only the published version.
Abstract—In this paper, we propose a control method for leader-follower navigation in obstacle environments while preserving sensing network connectivity without data transmission between robots. Unlike most connectivity-preserving algorithms, the control input is determined in such a way as to not only guarantee connectivity preservation and collision avoidance, but also to ensure input constraints are not violated at each time step. We also introduce a simple rule for changing network topology depending on environments such that some sensing links are deactivated in order to pass through narrow spaces, while active links are increased in free spaces to keep the group as cohesive as possible. The effectiveness of the proposed method is demonstrated in simulations and experiments.

Index Terms—Leader-follower navigation, connectivity maintenance, collision avoidance, obstacle environment, line of sight.

I. INTRODUCTION

The cooperative control of multiple mobile robots has been intensively studied for potential applications such as exploration, surveillance, mapping of unknown environments, and the transport of large objects (see, e.g., [1], [2], and [3] for an overview). This paper focuses on the fundamental problem of how to move a group of robots as a whole to a target area. Specifically, we assume that only one of the robots, called the leader, knows the path to the target area. Thus, since each robot in the group has a limited sensing and communication range, the connectivity of the sensing/communication network must be preserved in order to avoid leaving some robots behind. Furthermore, we aim to derive an algorithm without relying on data transmission between robots, in order to deal with environments where there is no wireless network available for such information exchange.

Various methods for controlling multi-agent systems while preserving network connectivity have been proposed [4]–[24], as detailed in Section II (see also [25] for an overview). However, most of the previous methods have had at least one of the following limitations.

1) An obstacle-free environment is considered [4]–[20].
2) Data transmission between robots through a wireless network is required to estimate network connectivity [4]–[12], [22].
3) It is difficult to explicitly consider input constraints [4]–[17], [22]–[24].
4) A fixed network topology must be preserved or only links can be added [13]–[17], [23], [24].
5) Inter robot collision is not considered [18]–[21].

In this paper, we propose a control method for leader-follower navigation in obstacle environments while preserving sensing network connectivity without data transmission between robots. Unlike most connectivity-preserving algorithms, the control input is determined so as not only to guarantee connectivity preservation and collision avoidance, but also to ensure that a given input constraint is not violated at each time step. Although an input constraint is considered in the connectivity-preservation algorithm for the leader-follower navigation proposed in [20], it has limitations in that an obstacle-free environment is assumed and that collision avoidance is not guaranteed. In obstacle environments, the proposed method manages control input so as to preserve line-of-sight (LOS) visibility between neighbors as well as a maximum distance constraint. We also derive conditions for collision avoidance not only with robots that are visible at the current sampling step, but also with those that are not visible, e.g., due to an obstacle. Another key issue for leader-follower navigation in obstacle environments is how to move through narrow spaces without getting stuck. We introduce a simple rule to change network topology depending on environments in such a way that some sensing links are deactivated in order to pass through narrow spaces while active links are increased in free spaces to keep the group as cohesive as possible. Furthermore, unlike many other studies including [20], the effectiveness of the algorithm is demonstrated not only with simulations, but also in real robot experiments.

II. RELATED WORKS

In this section, we review previous studies on network connectivity preservation in multi-agent systems. Although many control methods have been developed by assuming network connectivity (see e.g. [26]–[38]), we do not focus on these in this paper.

Many studies use the Fiedler value [39], which is the second smallest eigenvalue of the Laplacian matrix of the graph describing the network, as a metric for overall network
connectivity. If the Fiedler value of a graph is positive, the connectivity of the graph is guaranteed. Some early studies presented centralized algorithms to increase the Fiedler value of graphs. Kim and Mesbahi [4] proposed an iterative semidefinite programming-based approach to maximizing the Fiedler value of the graph, while Zavlanos and Papas [5] reported an artificial potential function-based approach to keeping the Fiedler value positive.

One way to decentralize such algorithms is to make use of a decentralized estimation method of the connectivity [6]–[8], in which each agent estimates the eigenvalues (or eigenvectors) of the Laplacian matrix using information received from neighbors. Based on the connectivity estimation, a gradient controller to maximize the Fiedler value was designed in [6] and [7], while an artificial potential function to keep the Fiedler value positive was used in [9], [10], [22].

Although these algorithms based on the Fiedler value are theoretically sophisticated, it is difficult to apply them to the control problem described in Section I, since they require data transmission between robots (or between a central computer and each robot in the centralized algorithms [4], [5]), through wireless communication to exchange information on the Laplacian matrix. Another limitation is the difficulty in explicitly considering an input constraint. Furthermore, poor estimation of connectivity could lead to violation of network connectivity. To address this issue, Sabattini et al. [9] proved the boundedness of estimation errors of the Fiedler value, and suggested taking an estimation error bound into account in the design of the artificial potential function. However, it is not clear how to obtain such an error bound that is not too conservative to apply to the artificial potential design.

In [11], an auction algorithm was implemented to decide cooperatively whether or not a link could be deactivated without violating network connectivity. Zavlanos et al. [12] proposed a flocking algorithm that achieved velocity synchronization of agents in a network while preserving connectivity using a method similar to that in [11]. However, the auction algorithm required data transmission between robots to share information on bids from other robots.

On the other hand, network connectivity preservation algorithms without data transmission between robots have also been studied. In [13]–[17], [23], artificial potential functions were used to preserve the initial network topology, which was assumed to be connected. In particular, connectivity preservation in obstacle environments was considered in [23]. Since these methods do not allow for the deactivation of any sensing link, it is difficult for robots to pass through narrow spaces if their initial network topology has many redundant links to make the group cohesive. Another limitation of these methods is that it is difficult to explicitly consider an input constraint. On the other hand, in [18]–[21], a given input constraint can be explicitly considered in the control design.

In [18], [19], [21], multi-robot rendezvous algorithms while preserving connectivity are proposed. In particular, obstacle environments are considered in [21]. With these algorithms, each robot computes at each time a convex region, called a constraint set, in which the robot is constrained to move within this region to preserve sensing links with neighbors. The robots then move towards the circumcenter of their constraint set. The amount of movement is determined such that no given input constraint is violated. However, moving towards the circumcenter of the constraint set in order to gather at the same location is not necessarily appropriate in the leader-follower navigation problem considered here. Another difference between our proposed control method and these rendezvous algorithms is that we consider inter-robot collisions. Thus, we need to consider the problem of how to deactivate sensing links in order for robots to pass through narrow spaces without getting stuck; this does not arise if inter-robot collisions are ignored. Furthermore, our method does not need to compute a constrained set to determine control inputs, which results in decreased computation time.

The leader-follower navigation algorithm in [20] decides the direction of movement using an artificial potential function, then the amount of movement is determined taking into account the input constraint and network connectivity. We also use this basic procedure in our proposed method. However, [20] did not consider obstacle environments and inter-robot collisions. To overcome this limitation, we derive additional constraints on the amount of movement so as to achieve LOS visibility preservation, obstacle avoidance and inter robot collision avoidance. Furthermore, as already mentioned, we introduce a link deactivation rule in order for robots to pass through narrow spaces without getting stuck.

Panagou and Kumar [24] proposed a leader-follower navigation method in obstacle environments where the network topology was fixed to a chain formation. As the distance between robots was controlled to a given constant value, their multi-robot system can be regarded as a tractor-trailer system. A limitation of this method is that as the number of robots grows, the turning radius of the leader must be increased, and a wider path is required. Furthermore, tracking error from a target relative position is not guaranteed to converge to zero, and estimation of the error bound is difficult especially in the case of multiple followers. If the tracking error is large, connectivity maintenance and collision avoidance might not be achieved.

III. PROBLEM SETTING

We consider $N$ robots in a two-dimensional work space with obstacles. The movement of the $i$th robot ($i = 1, 2, \ldots, N$) is described as the following discrete-time system

$$x_i(k+1) = x_i(k) + u_i(k), \quad \|u_i(k)\| \leq u_{\max} \quad (1)$$

where $x_i(k)$ and $u_i(k)$ are the position and control input of robot $i$, respectively, at time step $k = 0, 1, \ldots$. The input limit $u_{\max}$ in (1) is given by taking into account the hardware limitations and length of the sampling interval. We assume that a sufficient condition for collision avoidance between robots $i$ and $j$ is given as follows.

$$\|x_i - x_j\| \geq d_c, \quad \forall j \in \mathcal{V} \setminus \{i\} \quad (2)$$

where $\mathcal{V} := \{1, 2, \ldots, N\}$ is the set of indices of all robots. Although a robot is modeled as a point in (1), $d_c$ in (2) should be determined by taking into account the size of the actual
robots. We also assume that a sufficient condition for obstacle avoidance is given as

$$\|x_i - x_o\| \geq d_o, \quad \forall x_o \in \mathcal{O}$$

(3)

where $\mathcal{O}$ is a set of all points on obstacles in the workspace.

In order to describe the sensing model, we first define the line segment joining $p$ and $q$ as

$$\mathcal{L}(p, q) := \{(1 - \lambda)p + \lambda q, \forall \lambda \in [0, 1]\}.$$  

(4)

Furthermore, we define

$$\mathcal{L}_{ij}(k) := \mathcal{L}(x_i(k), x_j(k)).$$

(5)

Then, we assume that robot $i$ is able to sense the relative position of robot $j$ if $j \in \mathcal{V} \setminus \{i\}$

$$x_{ji}(k) := x_j(k) - x_i(k)$$

(6)

if the following conditions are satisfied

$$\|x_j - x_i\| \leq d_s$$

(7)

$$\|q - x_o\| \geq d_l, \quad \forall q \in \mathcal{L}_{ij}, \quad \forall x_o \in \mathcal{O}.$$  

(8)

The condition in (7) implies that the maximum sensing range is given by a positive number $d_s$. The condition in (8) implies that the distance from $\mathcal{L}_{ij}$ to each obstacle is not less than a minimum clearance $d_l$, so that the LOS between robots $i$ and $j$ is not interrupted by obstacles.

It is also assumed that robot $i$ is able to detect a point on an obstacle $x_o \in \mathcal{O}$, if

$$\|x_o - x_i\| \leq d_s$$

(9)

$$\|x_o - x_i\| \leq \|q - x_i\|, \quad \forall q \in \mathcal{L}(x_i, x_o) \cap \mathcal{O}$$

(10)

where $\mathcal{L}(x_i, x_o) := \{(1 - \lambda)x_i + \lambda x_o, \forall \lambda \geq 0\}$. While the set $\mathcal{L}(x_i, x_o)$ only includes points between $x_i$ and $x_o$, the set $\mathcal{L}(x_i, x_o)$ includes points behind $x_o$ along the line from $x_i$ to $x_o$, in addition to the points in $\mathcal{L}(x_i, x_o)$. Thus, the condition in (10) implies that there is no other obstacle point closer to $x_i$ than $x_o$ on $\mathcal{L}(x_i, x_o)$. We denote $\mathcal{O}_i(k)$ as the set of points on obstacles detected by robot $i$ at time $k$.

In terms of the sensing mentioned above, we represent the network topology of the multi-robot system using a graph $\mathcal{G}_s(x(k)) = (\mathcal{V}, \mathcal{E}_s(x(k)))$ where $x = [x_1, x_2, \ldots, x_N]$. We denote $\mathcal{V}$ and $\mathcal{E}_s(x(k))$ as the node set and edge set, respectively. The elements of $\mathcal{E}_s(x(k))$ are pairs of robot indices that are able to sense each other’s position at time $k$. The graph $\mathcal{G}_s$ is said to be connected, if for every pair of nodes there exists a path from one node to the other.

We assume that a target path is given to only one of the $N$ robots, called the leader, whose index is set as $N$ without loss of generality. Other robots $i = 1, 2, \ldots, N - 1$ are called followers. It is assumed that each follower is not able to recognize whether or not another robot is the leader. If $\mathcal{G}_s$ is connected, there is a path between the leader and each follower. Thus, since the maximum length of each link of $\mathcal{G}_s$ is kept to no more than a given finite value $d_s$, each follower is forced to follow the leader at a certain distance (at most $(N - 1)d_s$). Furthermore, if the length of the link is kept to less than $d_s$, the distance between the leader and a follower can be decreased. Thus in this paper, we aim to preserve the connectivity of the following subgraph of $\mathcal{G}_s(x(k))$.

$$\mathcal{G}_s(x(k)) := (\mathcal{V}, \mathcal{E}_s(x(k)))$$

(11)

$$\mathcal{E}_s(x(k)) := \{(i, j) \in \mathcal{E}_s(x(k)) \mid \|x_i(k) - x_j(k)\| \leq d_n\}$$

where $d_n < d_s < d_s$. Then, the set of neighbors of robot $i$ is defined as

$$\mathcal{N}_i(x(k)) := \{j \mid (i, j) \in \mathcal{E}_s(x(k))\}.$$  

(12)

From the definition of $\mathcal{G}_s(x(k))$, the connectivity of $\mathcal{G}_s$ is preserved if that of $\mathcal{G}_s(x(k))$ is preserved. As clarified in Theorem 2 in Section IV-C, it is required that $\sqrt{d_s^2 + d_s^2} \leq d_s$ holds to guarantee the preservation of $\mathcal{G}_n$. In other words, if $d_n = d_s$, i.e., $\mathcal{G}_n = \mathcal{G}_s$, it is difficult to guarantee the preservation of $\mathcal{G}_s$, which is another reason why we introduce $\mathcal{G}_n$ in addition to $\mathcal{G}_s$.

We assume that $\mathcal{G}_n$ is connected at the initial time $k = 0$. Thus, the simplest way to preserve connectivity is to control the robots such that the edges of $\mathcal{G}_n$ at $k = 0$ are not lost. However, in obstacle environments, it is often necessary to change the network topology appropriately to navigate through a narrow space. Thus, it is necessary to select edges to be maintained at each time step, such that the connectivity of $\mathcal{G}_n$ is preserved at the next time step. To describe the edges to be preserved, we define the symmetric indicator function $\sigma_{ij}(k) = \sigma_{ji}(k) \in \{0, 1\}$. If $\sigma_{ij} = 1$, there will be an effort to preserve the edge $(i, j)$. In other words, robot $i$ aims to preserve the link to robot $j$ in the following set

$$\mathcal{N}_i^*(x(k)) := \{j \in \mathcal{N}_i(x(k)) \mid \sigma_{ij}(k) = 1\}.$$  

(13)

We also define the following subgraph of $\mathcal{G}_n(x(k))$

$$\mathcal{G}_n(x(k)) := (\mathcal{V}, \mathcal{E}_n(x(k)))$$

(14)

$$\mathcal{E}_n(x(k)) := \{(i, j) \in \mathcal{E}_n(x(k)) \mid \sigma_{ij}(k) = 1\}.$$  

(15)

If $\sigma_{ij}$ is determined such that $\mathcal{G}_n$ is connected, the connectivity of $\mathcal{G}_n$ is preserved by moving the robots in such a way that no edge of $\mathcal{G}_n$ is lost.

In this paper, we propose an algorithm to determine $u_i$ so as to preserve the connectivity of $\mathcal{G}_n$, while satisfying the collision avoidance conditions in (2) and (3). The proposed method aims to preserve these properties not only at $x_i(k) (k = 0, 1, \ldots)$ but also at each point on the line segments $\mathcal{L}(x_i(k), x_i(k + 1))$. It is reasonable to consider a path composed of line segments in the sense that a robot described by a commonly used continuous time model $\dot{x}_i(t) = u_i(t)$ follows such a path when the control input $u_i(t)$ is fixed between discrete time steps. Furthermore, the proposed algorithm aims to determine $\sigma_{ij}$ in order to avoid deadlock in narrow spaces, while increasing active links to keep the group as cohesive as possible if there are no obstacles around the robots.

**Remark 1:** In order to keep the notation as simple as possible, the problem setting in this section and control algorithm in the next section are described by the global coordinates. However, for implementation, each robot determines the control input vector using local coordinates. Global coordinates such as $x_i$, $x_j$ and $x_o$ are replaced by local coordinates $x_i^l$, $x_j^l$ and $x_o^l$ in the control algorithm, where $x_i^l = 0$ since the
origin of robot $i$'s local frame is located at $x_i$. The control input $u_i$ with respect to the local frame is obtained by the control algorithm using the local coordinates. By applying $u_i$ to the robot, the global input vector $u_i = ^0R_iu_i$ is applied in (1), where $^0R_i \in SO(2)$ denotes the orientation of robot $i$'s local frame with respect to the global frame. Therefore, the proposed algorithm can be implemented using only local information.

IV. CONTROL ALGORITHM

The outline of the control algorithm for robot $i$ at time $k$ is described as follows.

Step 1: According to the sensing information on the relative position of robot $j \in N_i$, the indicator function $\sigma_{ij}(k)$ is determined, such that $G_n(x(k))$ is connected.

Step 2: The direction of the control input vector $u_i(k)$ is determined based on an artificial potential function.

Step 3: The magnitude of $u_i(k)$, which guarantees the edge preservation of $G_n(x(k))$ and collision avoidance at each point on $\mathcal{L}(x_i(k), x_i(k+1))$, is determined by taking into account the given input constraint in (1).

In the following subsections, we describe the details of each step.

A. Link Deactivation for Navigation in Narrow Spaces

By link deactivation, we mean deciding the edges of $G_n(x(k))$ that will not be preserved. More precisely, we obtain the set $\mathcal{E}_n$ of preserved edges by removing some edges from $\mathcal{E}_\sigma$. There are two cases where an edge in $\mathcal{E}_n$ is not included in $\mathcal{E}_\sigma$. To describe the first case, the following region is defined for given $x_i, x_j$ and $x_{ji}$ in (6).

$$D_{ij} := \{q \mid (q - x_i)^T x_{ji} > 0, (q - x_j)^T x_{ji} < 0\}$$  (16)

As illustrated in Fig. 1 (left). We also define

$$\varphi(p, q) := \frac{p^T H q}{\|H q\|^2} \quad H := \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$  (17)

which is the projection of a vector $p$ to the line orthogonal to $q$. Thus, as shown in Fig. 1 (left), $\|\varphi(x_{mi}, x_{ji})\|$ is the distance from $x_m$ to the line including $x_i$ and $x_j$. Then, we define $\mathcal{T}$ as the set of triples of robots $(i, j, m)$ that satisfies

$$\|\varphi(x_{mi}, x_{ji})\| \leq d_{del}, \quad x_m \in D_{ij}, \quad \sin \alpha_m > 0$$  (18)

$$(i, j) \in \mathcal{E}_n, \quad (j, m) \in \mathcal{E}_n, \quad (m, i) \in \mathcal{E}_n$$  (19)

where $d_{del}$ is a positive constant satisfying

$$d_{del} < d_c \sin \frac{\pi}{3}$$  (20)

In (18), $\alpha_m \in (-\pi, \pi]$ denotes the angle from the vector $x_{im}$ to $x_{jm}$ measured in the counter-clockwise direction. The condition $\sin \alpha_m > 0$ implies that the triple vertices in $\mathcal{T}$ are ordered in a counter-clockwise direction, as shown in Fig. 1 (left). Then, robot $i$ does not include the edge $(i, j) \in \mathcal{E}_n$ in $\mathcal{E}_\sigma$ if

$$(i, j, m) \in \mathcal{T} \quad \text{or} \quad (j, i, m) \in \mathcal{T}, \quad \exists m \in V \setminus \{i, j\}.$$  (21)

If the condition in (20) and the collision avoidance condition in (2) are satisfied, it is guaranteed that the links $(j, m)$ and $(m, i)$ of the robots $(i, j, m)$ satisfying (21) will not be deactivated, as shown in Lemma 3 in Appendix A. Therefore, the network connectivity is preserved at least if $N = 3$, i.e., if there are no robots other than $(i, j, m)$. Furthermore, Theorem 1 discusses connectivity in the case of $N > 3$. On the other hand, if (20) is violated, the preservation of connectivity is not guaranteed even if $N = 3$. For example, if $d_{del} = d_c \sin \frac{\pi}{3}$, all the links among the robot triple $(i, j, m)$ will be deactivated, when their positions constitute an equilateral triangle with edge length $d_c$.

To describe the other case where an edge is removed from $\mathcal{E}_\sigma$, we define $\mathcal{T}$ as the set of robot triples $(i, j, m)$ that satisfy

$$\|x_{ij}\| = \|x_{jm}\| = d_n, \quad \|x_{mi}\| = d_c, \quad \sin \alpha_m > 0$$  (22)

and (19). As illustrated in Fig. 1 (right), a robot triple $(i, j, m) \in \mathcal{T}$ forms an isosceles triangle with two edges of length $d_n$ and one edge of length $d_c$. In the edge deletion rule in this paper, robot $i$ does not include the edge $(i, j) \in \mathcal{E}_n$ in $\mathcal{E}_\sigma$ if

$$(i, j, m) \in \mathcal{T} \quad \text{or} \quad (j, i, m) \in \mathcal{T}, \quad \exists m \in V \setminus \{i, j\}.$$  (23)

In summary, the rule to decide $\sigma_{ij}$ is described as follows.

$$\sigma_{ij}(k) = \begin{cases} 0, & \text{if (21) or (23)}, \\ 1, & \text{otherwise}. \end{cases}$$  (24)

This rule is decentralized and does not require data transmission between robots. We obtain the following result on the connectivity of $G_n$ when multiple links are deleted at the same time by the rule in (24).

Theorem 1: Suppose A1) $G_n(x(k))$ is connected, A2) all robots satisfy the collision avoidance conditions in (2)-(3) at time $k$, A3) the condition in (20) is satisfied, and A4) $\pi$ is not an integer multiple of $\sin^{-1}(d_c/2d_n)$. Then, $G_n(x(k))$ obtained by the rule in (24) is connected.

Proof: See Appendix A.

Remark 2: Since the equalities in (22) do not hold exactly in practice, we instead check if the following inequalities

$$d_n - \epsilon_n \leq \|x_{ij}\| \leq d_n, \quad d_n - \epsilon_n \leq \|x_{jm}\| \leq d_n$$

$$d_c \leq \|x_{mi}\| \leq d_c + \epsilon_c$$

are satisfied for small positive constants $\epsilon_n$ and $\epsilon_c$. 

Fig. 1. Conditions for $(i, j, m) \in \mathcal{T}$ (left) and $(i, j, m) \in \mathcal{T}$ (right).
B. Direction of Control Input Vector

For the leader \((i = N)\), the direction of \(u_i(k)\), which is equivalent to the robot’s direction of movement, is given from the target path in the same way as described in [20]. For the followers, on the other hand, the direction of the movement is decided by

\[
v_i = -\nabla_x \Psi_i(x) , \quad i \in \mathcal{V} \setminus \{N\}
\]  

(25)

where \(\Psi_i(x)\) is the artificial potential function defined below. Note that \(v_i\) in (25) can be computed using only local information, since \(\Psi_i(x)\) depends only on a part of the elements of \(x\), i.e., \(x_i\) and \(x_j\) \((j \in \mathcal{N}_i^n)\).

The artificial potential function \(\Psi_i(x)\) is a weighted sum of \(\Psi_{i}^{\text{col}}, \Psi_{i}^{\text{obs}}, \Psi_{i}^{\text{los}}\) and \(\Psi_{i}^{\text{coh}}\) as follows

\[
\Psi_i := c_1 \Psi_{i}^{\text{col}} + c_2 \Psi_{i}^{\text{obs}} + c_3 \Psi_{i}^{\text{los}} + c_4 \Psi_{i}^{\text{coh}}.
\]

(26)

The first component \(\Psi_{i}^{\text{col}}\) takes into account the desired value and constraints on the relative distances to neighbors \(j \in \mathcal{N}_i^n\).

Artificial potential functions for this purpose are available in the literature. In this paper, we adopt a similar function to that in [17], as follows:

\[
\Psi_{i}^{\text{col}}(x) = \sum_{j \in \mathcal{N}_i^n} \Phi_{i}^{\text{col}}(||x_i - x_j||)
\]

(27)

\[
\Phi_{i}^{\text{col}}(z) := \frac{(z - d_r)^2(d_n - z)}{(d_n - d_r)^2(z - d_r) + (d_r - d_n)^2(d_n - z)/\kappa_1} + \frac{(z - d_r)z - d_i)^2}{(d_n - d_r)^2(d_n - z) + (z - d_r)(d_n - d_r)^2/\kappa_2}
\]

where \(\kappa_1 \) and \(\kappa_2 \) are design parameters whose values are equivalent to \(\Phi_{i}^{\text{col}}(z)\) at \(z = d_r\) and \(z = d_n\), respectively. Fig. 2 illustrates an example of \(\Phi_{i}^{\text{col}}(z)\) for \(d_r = 0.3, d_n = 0.7, d_n = 1.0, \kappa_1 = \kappa_2 = 10\). As shown in this example, \(\Phi_{i}^{\text{col}}(z)\) has the minimum value at the desired relative distance \(d_r\), and monotonically increases as \(z\) goes to the maximum allowable distance \(d_n\), or to the minimum allowable distance \(d_r\).

The second component \(\Psi_{i}^{\text{obs}}(z)\) is introduced for obstacle avoidance. More precisely, \(\Psi_{i}^{\text{obs}}(x)\) is decided in such a way that \(-\nabla_x \Psi_{i}^{\text{obs}}(x)\) is in the direction away from the closest obstacle point detected at time \(k\). Such an obstacle point is defined as

\[
o_{i}^{\text{obs}} = \arg \min_{x_o \in \mathcal{O}_i(k)} ||x_o - x_i(k)||.
\]

(28)

By using \(o_{i}^{\text{obs}}\) above, \(\Psi_{i}^{\text{obs}}(x)\) is described as

\[
\Psi_{i}^{\text{obs}}(x) = \Phi_{i}^{\text{obs}}(||x_i - o_{i}^{\text{obs}}||)
\]

\[
\Phi_{i}^{\text{obs}}(z) = \begin{cases} \frac{1}{2} \left( \frac{z - d_r}{d_r - d_o} + \delta \right)^{-1} - \frac{1}{4 + \delta} \right)^2, & \text{if } z < d_o \\ 0, & \text{otherwise} \end{cases}
\]

where \(d_o > d_r\) is a design parameter, and \(\delta\) is a small number to keep \(\Phi_{i}^{\text{obs}}(z)\) finite. Fig. 3 presents an example of \(\Phi_{i}^{\text{obs}}(z)\) for \(d_o = 0.1, d_or = 0.2\). As shown in this example, \(\Phi_{i}^{\text{obs}}(z)\) monotonically increases as \(z\) goes from \(d_o\) to the minimum allowable distance \(d_o\) to obstacles.

The third component \(\Psi_{i}^{\text{los}}(x)\) is decided in such a way that \(-\nabla_x \Psi_{i}^{\text{los}}(x)\) is in the direction moving away from the closest point to \(\mathcal{L}_{ij}(k)\) among the obstacle points in \(D_{ij}\) detected at time \(k\). Such an obstacle point is defined as

\[
o_{ij}^{\text{los}} = \arg \min_{x_o \in \mathcal{O}_{ij} \cap \mathcal{D}_{ij}} ||\varphi(x_{oi}, x_{ji})||.
\]

(29)

We also define

\[
j^* = \arg \min_{j \in \mathcal{N}_i^n} ||o_{ij}^{\text{los}}||.
\]

(30)

By using \(o_{ij}^{\text{los}}\) and \(j^*\) above, \(\Psi_{i}^{\text{los}}(x)\) can be described as

\[
\Psi_{i}^{\text{los}}(x) = \Phi_{i}^{\text{los}}(||\varphi(x_i - o_{ij}^{\text{los}}, x_i - x_{j^*})||)
\]

\[
\Phi_{i}^{\text{los}}(z) = \begin{cases} \frac{1}{2} \left( \frac{z - d_r}{d_r - d_{lr}} + \delta \right)^{-1} - \frac{1}{4 + \delta} \right)^2, & \text{if } z < d_{lr} \\ 0, & \text{otherwise} \end{cases}
\]

In the same way as \(\Phi_{i}^{\text{obs}}(z)\), the value of \(\Phi_{i}^{\text{los}}(z)\) monotonically increases as \(z\) goes from \(d_{lr}\) to the minimum allowable distance \(d_l\) to obstacles. Thus, the component \(-\nabla \Psi_{i}^{\text{los}}(x)\) of \(v_i\) in (25) has the effect of moving \(\mathcal{L}_{ij}\) away from its closest obstacle point.

The fourth component \(\Psi_{i}^{\text{coh}}(x)\) is introduced to make the group more cohesive. More precisely, \(\Psi_{i}^{\text{coh}}(x)\) is decided such that \(-\nabla_x \Psi_{i}^{\text{coh}}(x)\) is in the direction of moving closer to
we further derive conditions where (33)-(34) are satisfied in order to avoid collision with any robot or obstacle point. Thus, follows.

In [20] without taking into account obstacles and collision avoidance. It should be noted that the leader and followers use the same algorithm to determine $\|u_i(k)\|$, unlike the direction of $u_i(k)$ in Section IV-B.

As mentioned in Section III, the proposed method aims to achieve connectivity preservation and collision avoidance not only at $x_i(k)$ for each discrete time $k = 0, 1, \ldots$ but also at each point on the line segment connecting $x_i(k)$ and $x_i(k+1)$, i.e., $p_i \in P_i(k) := L(x_i(k), x_i(k+1))$. To this end, we first present an upper bound for $\|u_i(k)\|$ to satisfy the following conditions.

i) The maximum distance condition for each $j \in N_i^{\sigma}(k)$

$$\|p_i - p_j\| \leq d_o, \quad \forall p_i \in P_i(k), \quad \forall p_j \in P_j(k). \quad (32)$$

ii) The inter-robot collision avoidance condition for each $j \in S_i(k)$

$$\|p_i - p_j\| \geq d_c, \quad \forall p_i \in P_i(k), \quad \forall p_j \in P_j(k). \quad (33)$$

iii) The obstacle avoidance condition for each $x_o \in O_i(k)$

$$\|p_i - x_o\| \geq d_o, \quad \forall p_i \in P_i(k). \quad (34)$$

iv) The LOS preservation condition for each $j \in N_i^{\tau}(k)$

$$q - x_o \geq d_i, \quad \forall x_o \in O \land \forall q \in L(p_i, p_j) \land \forall p_i \in P_i(k), \quad \forall p_j \in P_j(k). \quad (35)$$

Note that the upper bound of $\|u_i(k)\|$ to satisfy i) is derived in [20] without taking into account obstacles and collisions, while our goal here is to guarantee i)-iv) at the same time in obstacle environments. It also should be noted that conditions (33)-(34) must be satisfied for any $j \in V \setminus \{i\}$ and $x_o \in O$ in order to avoid collision with any robot or obstacle point. Thus, we further derive conditions where (33)-(34) are satisfied for $j \notin S_i$ and $x_o \notin O_i$.

The condition in [20] for $u_i(k)$ to satisfy i) is described as follows.

$$\|u_i(k)\| \leq \frac{1}{2} \left( d_o - \max_{j \in N_{ik}(k)} \|x_{ji}(k)\| \right) =: \tilde{u}_{i}^{\text{cons1}}(k) \quad (36)$$

$$\|u_i(k)\|^2 \leq \min_{j \in N_{ik}(k)} \left\{ u_{i}^T(\kappa) x_{ji}(k) \right\}. \quad (37)$$

In these conditions, $N_{ib}(k)$ denotes the set of robots $j \in N_i^{\sigma}(k)$ from which robot $i$ will move away at time $k + 1$, i.e.,

$$N_{ib}(k) = \{ j \in N_i^{\tau}(k) | \|x_{ji}(k)\| \leq 0 \} \quad (38)$$

and $N_{if}(k)$ denotes the rest of the robots $j \in N_i^{\tau}(k)$, i.e.,

$$N_{if}(k) = \{ j \in N_i^{\tau} | \|x_{ji}(k)\| > 0 \}. \quad (39)$$

C. Magnitude of Control Input Vector

In this section, we determine $\|u_i(k)\|$ ($i = 1, \ldots, N$) so as to achieve the preservation of $E_i(x_k)$ and collision avoidance. It should be noted that the leader and followers use the same algorithm to determine $\|u_i(k)\|$, unlike the direction of $u_i(k)$ in Section IV-B.

Using the vector $v_i(k)$ given in Section IV-B, the condition in (37) can be rewritten as

$$\|u_i(k)\| \leq \frac{1}{\|v_i(k)\|} \min_{j \in N_{if}(k)} \left\{ v_{i}^T(\kappa) x_{ji}(k) \right\} =: \tilde{u}_{i}^{\text{cons2}}(k).$$

In order to describe a condition for ii), we first decompose the set of robots $S_i(k)$ detected by robot $i$ into the following two sets

$$S_{ib}(k) = \{ j \in S_i(k) | v_{i}^T(\kappa) x_{ji}(k) \leq 0 \} \quad (41)$$

$$S_{if}(k) = \{ j \in S_i(k) | v_{i}^T(\kappa) x_{ji}(k) > 0 \}. \quad (42)$$

Using $S_{if}(k)$, an upper bound for $\|u_i(k)\|$ is given as

$$\tilde{u}_{i}^{\text{col}}(k) = \frac{1}{2} \left( \min_{j \in S_{if}(k)} \|x_{ji}(k)\| - d_c \right). \quad (43)$$

An upper bound of $\|u_i(k)\|$ to satisfy iii) is given as

$$\tilde{u}_{i}^{\text{obs}} = \max_{s \geq 0} \{ \|sv_i\| \mid \|x_{oi} - sv_i\| \geq d_o, \forall x_o \in O \}. \quad (44)$$

where $x_{oi} := x_o - x_i$. This upper bound allows robot $i$ to proceed in the direction of $v_i$, unless the distance to the closest obstacle point is less than $d_o$, as indicated in Fig. 4. It should be noted that exact maximization on the right-hand side of (44) is difficult for general obstacle environments. An approximate value of $\tilde{u}_{i}^{\text{obs}}$ can be obtained by discretizing $s$ and $x_o$ in (44). Another way is to approximate obstacles by a simple shape such as circles, for which the maximum in (44) can be analytically obtained (see Appendix D).

In order to describe the input bound $\tilde{u}_{i}^{\text{los}}$ for iv), we define the set of detected obstacle points in $D_{ij}$, toward which robot $i$ will move closer at $k + 1$, as follows.

$$O_{ij}(k) := \{ x_o \in O \cap D_{ij} \mid v_{i}^T(\kappa) \varphi(x_{oi}, x_{ji}) > 0 \}. \quad (45)$$

If $O_{ij}(k) = \emptyset$, no upper bound for $\|u_i(k)\|$ is given, i.e., $\tilde{u}_{i}^{\text{los}} = \infty$. If $O_{ij}(k) \neq \emptyset$, we define the closest obstacle point in $D_{ij}$ to $L_{ij}(k)$ as

$$\tilde{o}_{ij}^{\text{los}} = \arg \min_{x_o \in O_{ij}(k)} \|\varphi(x_{oi}, x_{ji})\| \quad (46)$$
As illustrated in Fig. 5, \( \bar{u}_{ij}^{\text{los}} \) is the maximum allowable distance of movement, such that the distance from robot \( i \) to the obstacles in the direction orthogonal to \( \mathcal{L}_{ij}(k) \) is not less than \( d_l \) at time \( k+1 \). Thus, \( \bar{u}_{ij}^{\text{los}} \) is obtained as

\[
\bar{u}_{ij}^{\text{los}} = \frac{\|\varphi(\theta_{ij}^{\text{los}} - x_i, x_j)\| - d_l}{\sin \theta_{ij}} \quad (48)
\]

where \( \theta_{ij} \in [0, \pi] \) is the angle between \( \mathcal{L}_{ij} \) and \( v_i \). An upper bound for \( \|u_i(k)\| \) to satisfy i)-iv) is now given as

\[
\bar{u}_i = \min \{ \bar{u}_i^{\text{con}1}, \bar{u}_i^{\text{con}2}, \bar{u}_i^{\text{col}}, \bar{u}_i^{\text{los}}, \bar{u}_i^{\text{max}} \}. \quad (49)
\]

Therefore, we determine \( u_i(k) \) as follows

\[
u_i(k) = \begin{cases} \bar{u}_i(k) \frac{\|v_i(k)\|}{\|v_i(k)\|}, & \text{if } \|v_i(k)\| > \bar{u}_i(k) \\ v_i(k), & \text{otherwise} \end{cases} \quad (50)
\]

Note that since \( u_i \) is computed using positions of detected robots and obstacles and since the sensing range of each robot is limited, the computation time for each robot does not grow as the total number of robots and obstacles increases.

**Theorem 2**: Suppose that collision avoidance constraints in (2)-(3) are satisfied for all robots at time \( k \). Then, \( u_i(k) \) in (50) satisfies i)-iv) for each robot \( i \in \mathcal{V} \), if

\[
u_{\text{max}} \leq \min \left\{ \frac{d_s}{2}, \sqrt{d_s^2 - d_l^2} \right\} - \frac{d_c}{2} \quad (52)
\]

Then, the inter-robot collision avoidance condition in (33) is guaranteed for each \( j \in \mathcal{V} \setminus \mathcal{S}_i(k) \). Furthermore, if

\[
u_{\text{max}} \leq d_s - d_o \quad (53)
\]

the obstacle avoidance condition in (34) is guaranteed for each \( x_o \in \mathcal{O} \setminus \mathcal{O}_i(k) \).

**Proof**: See Appendix C.

**Remark 3**: For the leader, \( u_{\text{max}} \) in (49) might be replaced by a smaller value \( u_{\text{max}}^{\text{l}} \), since it improves the cohesion of the group, i.e., more edges are generated in \( \mathcal{E}_n \). In other words, if the leader moves at the maximum speed, i.e., \( \|v_i\| = u_{\text{max}} \), it is difficult for pairs of robots \( (i,j) \notin \mathcal{E}_n \) to decrease the inter-robot distance and to generate an edge in \( \mathcal{E}_n \).

While conditions for connectivity preservation and collision avoidance are derived in Theorems 1–3, it is difficult to clarify conditions to ensure that no robots get stuck in narrow spaces. In the next section, we demonstrate the effectiveness and limitations of the proposed method by simulations under various conditions.
V. SIMULATION

In this section, we run various simulations to demonstrate the effectiveness of the proposed method. The values of the parameters defined in Section III are $N = 10$, $d_n = 2$ m, $d_o = 1$ m, $d_e = 0.3$ m, $d_a = 0.1$ m, $d_l = 0.05$ m, and $u_{max} = 0.01$ m. The coefficients in (26) are set as $c_1 = 0.5$, $c_2 = c_3 = 0.01$, and $c_4 = 5$. The parameters in the artificial potential functions in Section IV-B are set as $\kappa_1 = \kappa_2 = 10$, $d_s = 0.7$, $d_{or} = 0.2$, $d_{ir} = 0.1$, and $\delta = 0.02$. We set $d_{del} = 0.25$, $\epsilon_n = 0.1d_n$, and $\epsilon_o = 0.1d_e$ for link deletion, and $u_{max} = \frac{2}{3}u_{max}$ to improve the cohesion of the group, as mentioned in Remark 3.

In order to measure the connectivity of the graph $G_o$, we compute the Fiedler value [39], which is the second smallest eigenvalue of the Laplacian matrix $L$. The Laplacian matrix $L$ is defined as

$$L = \Delta - A \quad (54)$$

by using the adjacency matrix $A$, whose elements $A_{ij}$ are 1 if $(i, j) \in E_o$ and 0 otherwise, and a diagonal matrix $\Delta$, whose diagonal elements are $\Delta_i = \sum_{j=1}^{N} A_{ij} (i = 1, \ldots, N)$.

Obstacle environments for simulations are illustrated in Fig. 6. The target path of the leader is a series of connected line segments starting at the origin, as shown in the thick solid lines. Obstacles are placed on both sides of the path. The relative angle of the $i$th line segment from $i-1$th line segment is defined as $\phi_i \neq 0$ for $i = 1, \ldots, M$, if $M \geq 1$. Thus, $M$ represents the number of corners of the target path. The first and last line segments can be divided into parts inside and outside the obstacle area. We define $\ell_s$ and $\ell_l$ as the lengths of the parts outside the obstacle area on the first and last line segments, respectively. Furthermore, the length of the $i$th line segment inside the obstacle area is defined as $\ell_i$ for $i = 0, 1, \ldots, M$.

The minimum distance between an obstacle and the leader's target path is defined as $\eta$, which implies that obstacles should not be in the area between two dashed lines. Subject to this minimum distance constraint, we place circular obstacles so that obstacles are as dense as possible to make the problem challenging. To this end, each obstacle is placed so that the distance from an obstacle to the leader's target path is equal to the allowable value $\eta$. In other words, the centers of circular obstacles with radius $r_o$ are placed on a dash-dotted line in Fig. 6, which is at a distance of $\eta + r_o$ from the target path. We also set distance between the centers of neighboring obstacles, $\rho$, as a small value so that the density of obstacles is high. Furthermore, we located additional obstacles with a radius smaller than $r_o$ at the inside corners, making the problem even more difficult. Specifically, we set $r_o = 0.5$ m and $\rho = 0.2$ m, while the radii of small obstacles at the inside corners were 0.1 m and 0.01 m.

We first tested the proposed method in the case where the target path of the leader was a straight line, i.e., $M = 0$. This was to examine the effectiveness of the link deactivation rule in Section IV-A. If the target path has corners, it is possible for robots to get stuck for a reason not necessarily related to the proposed link deactivation rule, as will be shown later. Thus, simulations with a straight path are suitable for investigating the performance of the proposed link deactivation rule. We performed simulations for $\ell_0 = 1$ m, $\ell_s = 1$ m, $\ell_f = 500$ m and various values of $\eta$, which is equivalent to various widths of passages, $2\eta$. Since the allowable minimum distance between a robot and an obstacle is $d_o = 0.1$ m, we could not choose an $\eta$ less than 0.1 m. We therefore tested the proposed method in cases where $\eta$ ranged from 0.105 m to 0.3 m. For each value of $\eta$, we performed 30 simulations for randomly selected initial robot positions. The solid line in Fig. 7 shows the percentage of cases in which all robots passed through the obstacle area without getting stuck using the proposed link deactivation rule. As may be seen in the figure, the proposed link-deactivation rule was effective in most cases, in contrast to the results without link deactivation shown by the dashed-dotted line. The minimum Fiedler value of $G_o$ in all cases was $9.8 \times 10^{-2}$, which implied that connectivity was preserved in all cases, since the Fiedler value was positive. Furthermore, the conditions for inter-robot collision avoidance and obstacle avoidance were satisfied in all simulations, where minimum inter-robot distance and minimum robot-obstacle distance were almost same as their allowable minimum value, $d_e$ and $d_o$, respectively.

We next performed simulations in the case where the target path of the leader had a corner, i.e., $M = 1$. In this case,
Fig. 10. Snapshots of a simulation.

even if sensing links were properly deactivated to go through a path, robots could get stuck at the corner. To illustrate this problem, we show simulation results in the case of $\ell_0 = 10$ m, where the first straight path in the obstacle area is long. The dashed lines in Fig. 8 show that the percentage of cases where all robots passed through the obstacle area for $\ell_1 = 1.5$, $\ell_4 = 1$ m, and $\ell_5 = 500$ m. It can be seen from this result that robots got stuck in many cases of $2\eta = 0.25$ and $2\eta \leq 0.4$ for $\phi_1 = \pi/6$ rad and $\phi_1 = \pi/2$ rad, respectively. This is in contrast that all robots passed through the obstacle area in most cases of $2.1 \leq 2\eta \leq 0.4$ for a straight path, as shown in Fig. 7. Note that since we set $N = 10$ and $d_n = 1$ m, the distance between the first and last robots is no more than 9 m. Thus, when the first robot reaches the corner in the case of $\ell_0 = 10$, all the robots are in the narrow path after the link deactivation has been successfully completed at the entrance of the narrow space. Therefore, this difference of results for $2\eta \leq 0.4$ in Fig. 7 compared with Fig. 8 illustrates that robots got stuck at a corner although the link deactivation was successfully completed. A reason why this problem arises is that the distance between two neighboring robots is too long to make a turn while preserving the LOS. Furthermore, the solid lines in Fig. 8 show that the possibility of getting stuck is increased for $\ell_0 = 1$ m, where the link deactivation has not been completed when some robots reach the corner. One reason for this is that the velocity of robots around an entrance of the narrow space is decreased during link deactivation. This makes it difficult for the robots in front to move forward. As a result, the distance between robots at the corner becomes longer than that in the case of $\ell_0 = 10$. Similarly, the rate of getting stuck is increased as the number of corners increases.

Fig. 9 compares the cases of $M = 1$ and $M = 2$ where we set $\phi_1 = \pi/2$, $\phi_2 = -\pi/2$, $\ell_1 = \ell_2 = 1.5$ m. As shown in this figure, the rate of getting stuck is higher for $M = 2$. The limitation in making turns at narrow corners might be improved by modifying the direction of movement in Section IV-B so that the distance between robots is decreased at corners, although that is beyond the focus of this paper. Despite this limitation, the simulation results show that the robots can successfully go through paths in most cases of $2\eta \geq 0.5$ where it is still difficult to navigate without link deactivation. It should be also noted that the conditions for connectivity preservation, inter-robot collision avoidance, and obstacle avoidance were satisfied in all examples shown in this section, regardless of whether the paths have corners or not. Fig. 10 shows snap shots of the case where $M = 2$, which illustrate that the robots are able to pass through narrow spaces by decreasing the number of active links, and then can increase the active links in free space. However, there is a limitation that regrouping into a cohesive formation takes quite a long time since it is difficult to start regrouping before the last robot clears the obstacle area. One reason for this is that the leader (the first robot) does not consider group cohesion in the control law. Thus, in order to improve the cohesion of the group, followers need to catch up with the leader. However, the followers other than the last robot cannot easily get closer to the leader, since they need to keep the maximum allowable distance $d_n$ with the neighbor behind as well as in front. Therefore, regrouping is not triggered until the last robot gets closer to the neighbor in front, after it clears the obstacle area.

VI. EXPERIMENT

The proposed method was applied to a group of 7 robots. For each robot, we used a mobile robot platform (Kobuki, Yujin Robot). Since the robots were not omni-directional, their orientation was controlled in the direction of the control input $u_i$ before moving forward. The position and orientation of each robot were measured by a motion capture system (OptiTrack...
s250e) in a centralized way. Obstacle avoidance was simulated using virtual obstacles whose positions were known to the robots. However, only local information that could be obtained in the sensing model in Section III was used to compute the control input in order to simulate the decentralized algorithm. Experimental validation using on-board sensors is planned for future research.

The control algorithm was implemented every 0.5 sec. The values of the parameters in Section III were $d_s = 1.6$ m, $d_n = 0.8$ m, $d_i = 0.4$ m, $d_o = 0.3$ m, $d_l = 0.2$ m, and $u_{\text{max}} = 0.04$ m. The coefficients in (26) were set to $c_1 = 0.6$, $c_2 = 0.1$, and $c_3 = c_4 = 1$. Parameters in the artificial potential functions were set to $\kappa_1 = \kappa_2 = 100$, $d_r = 0.7$, $d_{\text{or}} = d_{\text{lr}} = 0.4$, and $\delta = 0.02$. We set $d_{\text{det}} = 0.3$ for link deletion in (21) and $u_{\text{max}}^l = \frac{3}{4} u_{\text{max}}$ to improve the cohesion of the group.

As shown in Fig. 11, the virtual obstacles formed an L-shaped path. The initial positions of the leader and followers are indicated by blue and red circles, respectively. As shown in Fig. 12, the Fiedler value of $G_\sigma$ was always positive, which implies that connectivity was preserved. Fig. 13 shows that inter-robot distances (solid line) did not violate the minimum allowable value $d_c$ (dashed line). Furthermore, as shown in Fig. 14, the distances to obstacles did not violate the minimum allowable value $d_o$. Snap shots of the experiment are shown in Fig. 15, where the red lines in each photo represent the L-shaped path formed by the virtual obstacles. This figure shows that the robots traversed the L-shaped path by deactivating links to be preserved.

VII. CONCLUSIONS

This paper has presented a network connectivity preservation method for leader-follower navigation in obstacle environments that explicitly takes an input constraint into account. We derived conditions for the proposed method to guarantee connectivity preservation and collision avoidance in the presence of obstacles. A deactivation rule of sensing links, which uses only local sensing information to preserve global network connectivity, was introduced to allow the robots to navigate narrow spaces without getting stuck. The effectiveness of the proposed method was demonstrated by simulations and in experiments. Future research will address the problem that robots may get stuck in a corner even if sensing links are properly deactivated depending on the width of the path. The algorithm should also be improved so as to regroup robots into a cohesive formation more rapidly in free spaces.

APPENDIX A

PROOF OF THEOREM 1

To prove Theorem 1 by contradiction, suppose that $G_\sigma$ is not connected, under Assumptions A1)-A4) in Theorem 1. From Assumption A1), $G_n$ is connected. Thus, there exists at least one pair of robots, $i$ and $j$ of $G_\sigma$, such that all the paths connecting them are lost by applying the rule in (24),
Fig. 15. Snapshots of an experiment.

Fig. 16. Example of the case where all paths between two robots \(A_{I(1)}\) and \(B_{I(1)}\) are lost by deactivation rule in (24).

while \(i\) and \(j\) have an edge in \(G_m\), i.e., \((i, j) \in E_m\). This implies that robots \(i, j\) and another one \(m\) satisfy (21) or (23), so that \((i, j)\) is deactivated by the rule (24). We denote this robot triple by \((A_{h_1}, B_{h_1}, C_{h_1})\), where \(h_1 = 1\) in the case of \((A_{h_1}, B_{h_1}, C_{h_1}) \in \mathcal{T}\) as in (23) while \(h_1 = 2\) in the case of \((A_{h_1}, B_{h_1}, C_{h_1}) \in \mathcal{T}\) as in (21). Then, the edge \((A_{h_1}, B_{h_1})\) is deactivated by the rule in (24), regardless of \(h_1 = 1\) or \(h_1 = 2\). Therefore, if neither the edge \((A_{h_1}, C_{h_1})\) nor \((B_{h_1}, C_{h_1})\) is deactivated, there still exists a path \(A_{h_1}B_{h_1}A_{h_1}\) between \(A_{h_1}\) and \(B_{h_1}\), i.e., between robots \(i\) and \(j\). From Lemma 1, there are three cases where \((A_{h_1}, C_{h_1})\) or \((B_{h_1}, C_{h_1})\) is deactivated, under Assumptions A2)–A3).

We denote \((A_{h_1}, B_{h_2}, C_{h_2})\) by the robot triple that causes such a deactivation, where \(h_2 = 1, 2, 3\) corresponding to 1)–3) in Lemma 1 for \((A, B, C) = (A_{h_1}, B_{h_1}, C_{h_1})\) and \((A', B', C') = (A_{h_1}, B_{h_2}, C_{h_2})\). We also define \(I(k) := (h_1, h_2, \ldots, h_k)\) in order to make the notation simple, which implies that

\[
(A_{h_1, h_2}, B_{h_1, h_2}, C_{h_1, h_2}) = (A_{I(2)}, B_{I(2)}, C_{I(2)}). \tag{55}
\]

Although the edge \((A_{I(2)}, B_{I(2)})\), which is equivalent to \((A_{h_1}, C_{h_1})\) or \((B_{h_1}, C_{h_1})\), is deactivated in each case of \(h_2 = 1, 2, 3\), there still exists a path \(A_{I(2)}B_{I(2)}A_{I(2)}\) between \(A_{I(2)}\) and \(B_{I(2)}\), if the edges \((A_{I(2)}, C_{I(2)})\) and \((B_{I(2)}, C_{I(2)})\) remain. However, these edges can be deactivated, if there exists a robot triple \((A_{I(3)}, B_{I(3)}, C_{I(3)})\) that satisfies 1)–3) in Lemma 1 for \((A, B, C) = (A_{I(2)}, B_{I(2)}, C_{I(2)})\) and \((A', B', C') = (A_{I(3)}, B_{I(3)}, C_{I(3)})\), where \(h_3 = 1, 2, 3\) corresponding to 1)–3) in Lemma 1. Similarly, \((A_{I(\mu)}, B_{I(\mu)}, C_{I(\mu)})\) \((h_\mu = 1, 2, 3)\) is defined for \(\mu \geq 4\) as the robot triple that deactivates the edge \((A_{I(\mu-1)}, B_{I(\mu-1)})\) or \((B_{I(\mu-1)}, C_{I(\mu-1)})\). Since the number of robots is finite, the length of such a sequence of deactivations is finite. Thus, the path between \(A_{h_1}\) and \(C_{h_1}\) is not lost, unless a sequence of robot triples for deactivation form a loop; i.e., unless there exist two positive integers \(\mu_1\) and \(\mu_2\) \((\mu_1 < \mu_2)\) such that \((A_{I(\mu_2)}, C_{I(\mu_2)})\) or \((B_{I(\mu_2)}, C_{I(\mu_2)})\) is equivalent to \((A_{I(\mu_1)}, B_{I(\mu_1)})\). As shown in Lemma 2, such a loop is formed only if \((A_{I(\mu)}, B_{I(\mu)})\) \((\mu = 1, 2, \ldots, \mu_2)\) are each deactivated due to \((A_{I(\mu)}, B_{I(\mu)}, C_{I(\mu)})\) \(\in \mathcal{T}\), i.e., \(h_\mu = 1, 2, \ldots, \mu_2\) until a loop is formed, as illustrated in Fig. 16. To form such a loop without collision, it is therefore necessary for 2\(\pi\) to be an integer multiple of \(2\sin^{-1}(d_c/2d_n)\), which contradicts Assumption A4).

A. Lemmas to Prove Theorem 1

In this section, we denote the distance between robots \(A\) and \(B\) by \(|AB|\), in order to simplify the notation.

**Lemma 1:** In addition to A2)–A3) in Theorem 1, we assume that three out of \(N\) robots satisfy \((A, B, C) \in \mathcal{T}\) so that the edge \((A, B)\) is deactivated by the rule in (24). Then, the edge \((B, C)\) or \((A, C)\) is also deactivated, if and only if at least one of the following three conditions is satisfied, as illustrated in Fig. 17.

1) There exists a robot triple \((A', B', C') \in \mathcal{T}\) that deactivates the edge \((B, C)\), which is equivalent to \((A', B')\).
2) There exists a robot triple \((A', B', C') \in \mathcal{T}\) that deactivates the edge \((B, C)\), which is equivalent to \((A', B')\).
3) There exists a robot triple \((A', B', C') \in \mathcal{T}\) that deactivates the edge \((A, C)\), which is equivalent to \((A', B')\).

**Proof:** We only prove the necessity, since the sufficiency is obvious from the deactivation rule in (24).

From Lemma 3, we have \((C, A, B) \notin \mathcal{T}\) and \((B, C, A) \notin \mathcal{T}\), if \((A, B, C) \in \mathcal{T}\). Also, it is obvious from the definition of \(\mathcal{T}\) that \((A, C, B) \notin \mathcal{T}\) and \((B, A, C) \notin \mathcal{T}\) if \((A, B, C) \in \mathcal{T}\).

Thus, the three robots \((A, B, C)\) do not deactivate the edge \((A, C)\) or \((B, C)\) by themselves. In other words, \((A, C)\) or \((B, C)\) is deactivated, only if a robot other than \((A, B, C)\) constitutes a triple \((A', B', C') \in \mathcal{T}\) or \(\mathcal{T}'\), such that \((A', B')\) is equivalent to \((A, C)\) or \((B, C)\).

We first consider the case of \((A, B, C) \in \mathcal{T}\). Then, from Lemma 4, there is no robot triple \((A', B', C') \in \mathcal{T}\) that deactivates the edge \((B, C)\) or \((A, C)\), which is equivalent to \((A', B')\). In other words, \((B, C)\) or \((A, C)\) is deactivated only if \((A', B', C') \in \mathcal{T}\), as illustrated in 2b) and 3) of Fig. 17.

In the case of \((A, B, C) \in \mathcal{T}'\), the edge \((A, C)\) cannot be deactivated as shown in Lemma 5. In other words, only \((B, C)\) can be deactivated due to \((A', B', C') \in \mathcal{T}\) as in 1) or \((A', B', C') \in \mathcal{T}\) as in 2a) in Fig. 17.

**Lemma 2:** Under Assumptions A2–A3 in Theorem 1, there exist two positive integers \(\mu_1\) and \(\mu_2\) \((\mu_1 < \mu_2)\) such that \(A_{(I_{(\mu_2)}, I_{(\mu_2)})}(B_{(I_{(\mu_2)}, I_{(\mu_2)})})\) is equivalent to \((A_{(I_{(\mu_1)}, I_{(\mu_1)})}, B_{(I_{(\mu_1)}, I_{(\mu_1)})})\) only if \(h_{\mu} = 1$$ (\mu = 1, 2, \ldots, \mu_2)$$.

**Proof:** To prove the contrapositive of the lemma, suppose that \(h_{\mu} \neq 1\) for an integer \(\mu\) \((1 < \mu < \mu_2)\). We define \(\tilde{\mu}\) \((\mu \leq \tilde{\mu} \leq \mu_2)\) as the smallest positive integer \(\mu\) satisfying \(h_{\tilde{\mu}} \neq 1\). Then, it follows from Lemma 4 that \(h_{\tilde{\mu}} \neq 1\), i.e., \((A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}, B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}) \in \mathcal{T}\), for \(\tilde{\mu} \leq \mu_2 < \mu_2\). Thus, from Lemma 6, we have

\[
\frac{A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}, C_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}}{B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}} < \frac{A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}}{B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}}.
\]

Since the edge \((A_{(I_{(\tilde{\mu}_2)-1)}, I_{(\tilde{\mu}_2)-1})}, C_{(I_{(\tilde{\mu}_2)-1)}, I_{(\tilde{\mu}_2)-1})}\) is equivalent to \((A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)}), B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})})}\), it follows from (56) and Lemma 6 that

\[
\frac{A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}}{B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}} < \frac{A_{(I_{(\tilde{\mu}_2), I_{(\tilde{\mu}_2)-1})}}}{B_{(I_{(\tilde{\mu}_2), I_{(\tilde{\mu}_2)-1})}}}, \forall \mu_1 < \mu_2.
\]

By repeating the same process, we have

\[
\frac{A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}}{B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}} < \frac{A_{(I_{(\tilde{\mu}_2), I_{(\tilde{\mu}_2)-1})}}}{B_{(I_{(\tilde{\mu}_2), I_{(\tilde{\mu}_2)-1})}}}
\]

for \(\tilde{\mu} \leq \mu < \mu_2\). Thus, in the case of \(\tilde{\mu} \leq \mu_1\), we have

\[
\frac{A_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}}{B_{(I_{(\tilde{\mu}_2)}, I_{(\tilde{\mu}_2)})}} < \frac{A_{(I_{(\tilde{\mu}_2), I_{(\tilde{\mu}_2)-1})}}}{B_{(I_{(\tilde{\mu}_2), I_{(\tilde{\mu}_2)-1})}}}, \forall \mu_1 < \mu_2.
\]

We next consider the case of \(\mu_1 < \tilde{\mu}\). Since \(\tilde{\mu} > 1\) in this case, we have \(h_{\mu} = 1\), i.e., \((A_{(I_{(\mu)}, I_{(\mu)}), C_{(I_{(\mu)}, I_{(\mu)})}}) \in \mathcal{T}\), for \(1 \leq \mu < \tilde{\mu}\). This implies from the definition of \(\mathcal{T}\) that

\[
\frac{A_{(I_{(\mu)}, I_{(\mu)})}}{B_{(I_{(\mu)}, I_{(\mu)})}} = d_n.
\]
Then, since $AB^2 = d_n$ from the definition of $\mathcal{T}$, we have $AC = d_n > d_c$ which contradicts $(A, B, C) \in \mathcal{T}$. □

Lemma 6: In addition to A2–A3) in Theorem 1, we assume that $(A, B, C) \in \mathcal{T}$. Then, we have

$$AC < AB, \quad BC < AB.$$  \hspace{1cm} (67)

Proof: To prove by contradiction, suppose that $BC \geq AB$ and $BC \geq AC$, without loss of generality.

We first show that $\cos \alpha_A \leq \frac{1}{2}$ in each case of $AC \leq AB$ and $AC > AB$. In the case of $AC \leq AB$, it holds from $BC \geq AB$ that

$$\cos \alpha_A = \frac{AB^2 + AC^2 - BC^2}{2AB \cdot AC} \leq \frac{AC}{2AB} \leq \frac{1}{2}.$$  \hspace{1cm} (68)

In the case of $AC > AB$, it holds from $BC \geq AC$ that

$$\cos \alpha_A = \frac{AB^2 + AC^2 - BC^2}{2AB \cdot AC} \leq \frac{AB}{2AC} < \frac{1}{2}.$$  \hspace{1cm} (69)

Thus, we have $\cos \alpha_A \leq \frac{1}{2}$, which implies $|\alpha_A| \geq \frac{\pi}{3}$. Further, since $(A, B, C) \in \mathcal{T}$, it follows from (18) that $x_C \in D_{AB}$, which implies $|\alpha_A| < \frac{\pi}{3}$. Therefore, since $CA \geq d_c$ due to the collision avoidance condition A2),

$$\varphi(x_{CA}, x_{BA}) = \frac{TA \sin|\alpha_A|}{d_c} \geq \frac{d_c \sin \frac{\pi}{3}}{d_c}$$  \hspace{1cm} (70)

which contradicts $(A, B, C) \in \mathcal{T}$, under Assumption A3) that (20) is satisfied. □

APPENDIX B

PROOF OF THEOREM 2

If conditions ii)–iv) are guaranteed, i) can be proved in the same way as in [20]. Thus, in this section, we prove ii)–iv).

In order to show that ii) is satisfied, it suffices to provide proof for the worst case scenario in which two robots move closer to each other. Thus, we show that ii) is satisfied for any pair of robots $(i, j)$ that satisfies $j \in \mathcal{S}_j$, $i \in \mathcal{S}_f$. Since $\|u_i\| \leq u_i^\text{col}$ for $u_i^\text{col}$ in (43), the following constraints are satisfied.

$$\min_{m \in \mathcal{S}_f} \|x_{mi}(k)\| - 2\|u_i(k)\| \geq d_c$$  \hspace{1cm} (71)

$$\min_{m \in \mathcal{S}_f} \|x_{mj}(k)\| - 2\|u_j(k)\| \geq d_c.$$  \hspace{1cm} (72)

Therefore, from

$$\min_{m \in \mathcal{S}_f} \|x_{mi}(k)\| \leq \|x_{ji}(k)\|$$  \hspace{1cm} (73)

$$\min_{m \in \mathcal{S}_f} \|x_{mj}(k)\| \leq \|x_{ij}(k)\|$$  \hspace{1cm} (74)

and $\|x_{ji}(k)\| = \|x_{ij}(k)\|$, we have

$$\|x_{ji}(k)\| - 2\|u_i(k)\| \geq d_c$$  \hspace{1cm} (75)

$$\|x_{ji}(k)\| - 2\|u_j(k)\| \geq d_c.$$  \hspace{1cm} (76)

By summing (75) and (76), we obtain

$$\|x_{ji}(k)\| - \|u_i(k)\| - \|u_j(k)\| \geq d_c.$$  \hspace{1cm} (77)

for each $(i, j)$ that satisfies $j \in \mathcal{S}_j$, $i \in \mathcal{S}_f$. Thus, it holds for all $\lambda_i, \lambda_j \in [0, 1]$ that

$$\|x_{ji}(k)\| - \|\lambda_i u_i(k)\| - \|\lambda_j u_j(k)\| \geq d_c.$$  \hspace{1cm} (78)

which implies

$$\|\lambda_i u_i(k) - x_{ji}(k) - \lambda_j u_j(k)\| \geq d_c.$$  \hspace{1cm} (79)

Therefore, it holds from (1) that

$$\|\lambda_i u_i(k) + (1 - \lambda_i) x_{ji}(k) - \lambda_j x_j(k) + (1 - \lambda_j) x_j(k)\| \geq d_c$$  \hspace{1cm} (80)

which implies that ii) is satisfied.

We next prove iii). It holds from the definition of $\bar{u}_i^\text{obs}$ in (44) that

$$\|x_{oi} - s v_i\| \geq \bar{d}_o, \quad \forall x_o \in \mathcal{O}_i$$  \hspace{1cm} (81)

for each $s$ such that $\|s v_i\| \leq \bar{u}_i^\text{obs}$. Since $v_i$ and $u_i$ have the same direction, it follows from $\|u_i\| \leq \bar{u}_i^\text{obs}$ that

$$\|x_{oi} - \lambda_i u_i\| \geq \bar{d}_o, \quad \forall x_o \in \mathcal{O}_i, \quad \forall \lambda_i \in [0, 1].$$  \hspace{1cm} (82)

Thus, it holds from (1) that

$$\|x_{o} - (1 - \lambda_i) x_{i}(k) - \lambda_i x_{i}(k + 1)\| \geq \bar{d}_o, \quad \forall x_o \in \mathcal{O}_i$$  \hspace{1cm} (83)

for all $\lambda_i \in [0, 1]$, which implies that iii) is satisfied.

To prove iv), we first define $B(p, r)$ to be a closed ball centered at $p$ of radius $r$. Since $\|u_i(k)\| = \|x_{i}(k + 1) - x_{i}(k)\| \leq u_{\max}$ from (1), the line segments $p_i(k)$ and $p_j(k)$ are included in $B(x_i(k), u_{\max})$ and $B(x_j(k), u_{\max})$, respectively, as shown by the colored circles in Fig. 18. This implies that $L_{ij}(k + 1)$ is included in $co(B(x_i(k), u_{\max}) \cup B(x_j(k), u_{\max}))$ where $co(X)$ denotes the convex hull of a set $X$. Thus, since $d_o - u_{\max} \geq d_l$ from the assumption in (51), the obstacle points outside $co(B(x_i(k), d_o) \cup B(x_j(k), d_o))$ have a distance of more than $d_l$ from any point in $co(B(x_i(k), u_{\max}) \cup B(x_j(k), u_{\max}))$ including $L_{ij}(k + 1)$. Therefore, it is sufficient to consider the obstacle points in $co(B(x_i(k), d_o) \cup B(x_j(k), d_o))$.

Furthermore, from the assumption that (3) is satisfied at time $k$, there is no obstacle point in $B(x_i(k), d_o) \cup B(x_j(k), d_o)$. Thus, it is sufficient to consider the obstacle points in

$$A_{ij}(k) := D_{ij} \cap (co(B(x_i(k), d_o) \cup B(x_j(k), d_o)).$$  \hspace{1cm} (84)

Therefore, in order to prove iv), it suffices to show

$$\|\varphi(x_o - q, x_{ji}(k))\| \geq d_l, \quad \forall x_o \in A_{ij}(k)$$

$$\forall q \in L_{ij}(k), \quad \forall p_i \in p_i(k), \quad \forall p_j \in p_j(k)$$  \hspace{1cm} (85)

for each $j \in N^r_{\lambda}(k)$, since $\|x_o - q\| \geq \|\varphi(x_o - q, x_{ji}(k))\|$. 
From the definition of $\varphi$ in (17), we have
\[
\|\varphi(x_o - p_i, x_j(k))\| = \frac{\|x_o - p_i\|^T H x_j(k)}{\|H x_j(k)\|} = \frac{\|(x_o(k) - p_i + x_j(k))\|^T H x_j(k)}{\|H x_j(k)\|}.
\] (86)

For $j \in \mathcal{N}_i^c(k)$, we have $\|\varphi(x_o(k), x_j(k))\| \geq d_i > 0$ for any $x_o \in \mathcal{O}_i$, since the constraint in (8) is satisfied at time $k$. Thus, any obstacle point $x_o \in \mathcal{O}_i \cap D_{ij}$ belongs to one of the following two sets
\[
\mathcal{B}_{ij}(k) := \{x_o \in \mathcal{O}_i \cap D_{ij} \mid x_o^T(k) H x_j(k) > 0\} \\
\mathcal{B}_{ij}(k) := \{x_o \in \mathcal{O}_i \cap D_{ij} \mid x_o^T(k) H x_j(k) < 0\}.
\]

Without loss of generality, we assume $v_i^T H x_j > 0$, which implies $\mathcal{O}_{ij}^{\text{los}}(k) = \mathcal{B}_{ij}(k)$ in (45) from
\[
v_i^T \varphi(x_o(k), x_j(k)) = \frac{x_o^T(k) H x_j(k)}{\|H x_j(k)\|^2} v_i^T H x_j > 0.\] (87)

Thus, $d_{ij}^{\text{los}}$ in (46) can be described as
\[
d_{ij}^{\text{los}} = \arg \min_{x_o \in \mathcal{B}_{ij}} x_o^T H x_j.\] (88)

Therefore, it holds from (47) and (88) that
\[
\frac{(x_o - sv_i)^T H x_j}{\|H x_j\|} \geq d_i, \quad \forall x_o \in \mathcal{B}_{ij},\] (89)
\[
\frac{(x_o - sv_i)^T H x_j}{\|H x_j\|} \leq -d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\] (90)

for each $s \geq 0$ such that $\|sv_i\| \leq \tilde{u}_{ij}^{\text{los}}$. This implies that
\[
\frac{(x_o - p_i + x_j)^T H x_j}{\|H x_j\|} \geq d_i, \quad \forall x_o \in \mathcal{B}_{ij}
\] (91)
\[
\frac{(x_o - p_i + x_j)^T H x_j}{\|H x_j\|} \leq -d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\]

for each $p_i \in \mathcal{P}_i$, since $p_i = x_i + \lambda_i u_i$ and $\|\lambda_i u_i\| \leq \tilde{u}_{ij}^{\text{los}}$. Under the assumption that $\sqrt{d^2_i + d^2_i} \leq d_i$ in (51), any obstacle point in $\mathcal{A}_{ij}(k)$ is in the maximum sensor range of robot $i$ at time $k$, which implies $\mathcal{A}_{ij} \subset \mathcal{O}_i \cap D_{ij}$. Thus, it follows from (91) that
\[
\frac{(x_o - p_i + x_j)^T H x_j}{\|H x_j\|} \geq d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\] (92)
\[
\frac{(x_o - p_i + x_j)^T H x_j}{\|H x_j\|} \leq -d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\]

where
\[
\mathcal{B}_{ij}^A(k) := \{x_o \in \mathcal{A}_{ij} \mid x_o^T(k) H x_j(k) > 0\} \\
\mathcal{B}_{ij}^A(k) := \{x_o \in \mathcal{A}_{ij} \mid x_o^T(k) H x_j(k) < 0\}.
\]

For $p_j$, as in (92), it can be proved that
\[
\frac{(x_o - p_j + x_j)^T H x_j}{\|H x_j\|} \geq d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\] (93)
\[
\frac{(x_o - p_j + x_j)^T H x_j}{\|H x_j\|} \leq -d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\]

for each $p_j \in \mathcal{P}_j$, using $x_j^T H x_j = -x_i^T H x_j$. Thus, it holds from (92) and (93) that
\[
\frac{(x_o - q + x_j)^T H x_j}{\|H x_j\|} \geq d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\] (94)
\[
\frac{(x_o - q + x_j)^T H x_j}{\|H x_j\|} \leq -d_i, \quad \forall x_o \in \mathcal{B}_{ij}^A
\]

for each $q \in \mathcal{L}(p_i, p_j)$. Therefore, we have (85), which completes the proof of iv).

**APPENDIX C
PROOF OF THEOREM 3**

We first prove (33) for each robot $j \in \mathcal{V} \setminus \mathcal{S}_i$. From the definition of $\mathcal{S}_i$, robot $i$ is not able to detect robot $j$ (i.e., $j \in \mathcal{V} \setminus \mathcal{S}_i$), when at least one of the conditions in (7) and (8) is violated. If (7) is not satisfied at time $k$, we have $\|x_j(k)\| > d_s$. Since $p_i \in \mathcal{P}_i$ and $p_j \in \mathcal{P}_j$ are described as
\[
p_i = x_i(k) + \lambda_i u_i(k), \quad p_j = x_j(k) + \lambda_j u_j(k) \]

for $\lambda_i \in [0, 1]$ and $\lambda_j \in [0, 1]$, it holds that
\[
\|p_j - p_i\| = \|x_j(k) + \lambda_j u_j(k) - \lambda_i u_i(k)\| \\
\geq \|x_j(k)\| - 2u_{\max} > d_s - 2u_{\max}
\]

from $\lambda_j u_j(k) \leq u_{\max}$ and $\|\lambda_i u_i(k)\| \leq u_{\max}$. Thus, since $d_s - 2u_{\max} \geq d_i$, under the assumption in (52), the condition in (33) is satisfied.

In the case where (8) is violated, a robot is behind an obstacle. Since it is assumed that the obstacle avoidance condition in (3) is satisfied at $k$, each robot is located on the boundary or the outside of $\mathcal{B}(x_o, d_o)$ for each obstacle point $x_o$. Thus, as illustrated in Fig. 19, robot $i$ is not able to detect robot $j$ behind an obstacle point $x_o$, if the angle between $x_j$ and $x_o$ is less than $\beta = \tan^{-1}(d_j, \sqrt{x_o^2 - d^2_i})$. In this case, the distance between robots $i$ and $j$ satisfies
\[
\|x_j(k)\| > \sqrt{\|x_o(k)\|^2 - d^2_i + \sqrt{d^2_i - d^2_i}} \]

where the lower bound on the right-hand side is equal to the distance from $x_i$ to point $P_1$ or $P_2$ in Fig. 19. Furthermore, it follows from (97) and $\|x_o(k)\| \geq d_o$, that we obtain
\[
\|x_j(k)\| > 2\sqrt{d^2_o - d^2_i}.
\] (98)
Since $2u_{\text{max}} \leq 2\sqrt{d_o^2 - d_i^2} - d_c$, under the assumption in (52), the inputs of robots $i$ and $j$ are constrained as follows

$$
2\sqrt{d_o^2 - d_i^2} - 2\|u_i(k)\| \geq d_c \quad (99)
$$

$$
2\sqrt{d_o^2 - d_i^2} - 2\|u_j(k)\| \geq d_c. \quad (100)
$$

Thus, from (98), we have

$$
\|x_{ji}(k)\| - 2\|u_i(k)\| \geq d_c \quad (101)
$$

$$
\|x_{ji}(k)\| - 2\|u_j(k)\| \geq d_c \quad (102)
$$

which are the same inequalities as (75) and (76). The rest of the proof is then the same as the proof of ii) in Theorem 2.

We next prove (34) for each obstacle point $x_o \in \mathcal{O} \setminus \mathcal{O}_i(k)$. From the definition of $\mathcal{O}_i$, robot $i$ is not able to detect an obstacle point $x_o$ (i.e., $x_o \in \mathcal{O} \setminus \mathcal{O}_i$) when at least one of the conditions in (9) and (10) is violated. If (9) is not satisfied at time $k$, we have $\|x_o(k)\| > d_s$. Then, in the same way as in (96), we have

$$
\|x_o - p_i\| = \|x_o - \lambda_i u_i\| > d_s - u_{\text{max}} \quad (103)
$$

for $p_i$ in (95), since $|\lambda_i u_i| \leq u_{\text{max}}$ for $\lambda_i \in [0, 1]$. Thus, since $d_s - u_{\text{max}} \geq d_o$ under the assumption in (53), the condition in (34) is satisfied for each $x_o \in \mathcal{O} \setminus \mathcal{O}_i(k)$.

If an obstacle point $x_o$ does not satisfy (10), there is a detected obstacle point $x'_{o}$ which is located on the line segment connecting $x_i$ and $x_o$. Since the obstacle point $x_o$ behind $x'_{o}$ obviously does not move unlike the case where a robot is behind an obstacle point, (34) is satisfied for $x_o$ if it is satisfied for $x'_{o}$. Thus, since Theorem 2 guarantees that (34) is satisfied for any detected obstacle point including $x'_{o}$, (34) is satisfied for any obstacle point $x_o$ violating (10).

APPENDIX D
COMPUTATION OF $\tilde{u}_{i_{\text{obs}}}^C$ FOR CIRCULAR OBSTACLES

In this section, we describe how to compute $\tilde{u}_{i_{\text{obs}}}^C$ in (44) when obstacles detected by robot $i$ are approximated by circles. In other words, $\mathcal{O}_i$ in (44) is replaced by $\bigcup_{m=1}^{N_C} C_m$, where $N_C$ is the number of circles, and $C_m$ is the circumference of the circle centered at $x_{C_m}$ of radius $r_{C_m}$. In this case, $\tilde{u}_{i_{\text{obs}}}^C$ is described as

$$
\tilde{u}_{i_{\text{obs}}}^C = \min_{1 \leq m \leq N_C} \tilde{u}_{i_{\text{obs}}}^{C_m} \quad (104)
$$

$$
\tilde{u}_{i_{\text{obs}}}^{C_m} = \max_{s \geq 0} \{ \|sv_i\| + \|x_{o_m} - sv_i\| \geq d_o, \forall x_o \in C_m \}. \quad (105)
$$

In order to describe how to obtain $\tilde{u}_{i_{\text{obs}}}^{C_m}$, we define

$$
l_c := \|\varphi(x_{C_m} - x_i, v_i)\| \quad (106)
$$

which is the distance from $x_{C_m}$ to the line including $x_i$ and $x_i + v_i$, as shown in Fig. 20. Then, a necessary and sufficient condition for $\tilde{u}_{i_{\text{obs}}}^{C_m}$ in (105) to be finite is that the following inequalities are satisfied.

$$
(x_{C_m} - x_i)^T v_i > 0, \quad l_c - r_{C_m} < d_o. \quad (107)
$$

The first inequality in (107) implies that the robot moves closer to $C_m$, while the second one implies that the distance from $C_m$ to the line including $x_i$ and $x_i + v_i$ is less than $d_o$. Using the condition in (107), $\tilde{u}_{i_{\text{obs}}}^{C_m}$ is obtained as

$$
\tilde{u}_{i_{\text{obs}}}^{C_m} = \left\{ l_v - \sqrt{(d_o + r_{C_m})^2 - l_c^2}, \quad \text{if (107)} \right\}
$$

$$
\infty, \quad \text{otherwise} \quad (108)
$$

where $l_v := \sqrt{\|x_{C_m} - x_i\|^2 - l_c^2}$ as shown in Fig. 20.

REFERENCES


