Adaptive Hierarchical Distributed Control with Cooperative Task Allocation for Robot Swarms

Pham Duy Hung[†,‡], Hung Manh La [§] *Senior Member, IEEE* and Trung Dung Ngo[‡] *Senior Member, IEEE*

***Abstract*— We present an adaptive hierarchical distributed control strategy (AHDC) enabling a small swarm of mobile robots to explore and track very large target clouds. The hierarchical distributed control architecture (HDC) is designed to govern not only robot behaviours but also their neighbourhood connectivities for global network integrity preservation. A mobile robot equipped with the HDC is capable of adaptively pruning neighbourhood connectivities so the robot can deal with complexity and constraints of local connectivity topologies. A cooperative target observation, tracking, and release (COTR) algorithm is incorporated into the HDC to allow a robot to track more than one target in large target clouds. We have demonstrated and evaluated effectiveness of the AHDC through both simulation and real-world experiments.**

I. Introduction

Swarm robotics has been an interesting research topic of robotics over the past decades. A robot swarm can be employed for numerous applications e.g., coverage, surveillance, searching, patrolling, observation. Multi-target tracking is known as a typical application of robot swarms. It is classified into two allocated tasks: *target observation* enabling a robot swarm to search and observe targets in a given environment, and *target tracking* allowing the robots to follow and catch their assigned target [1], [2]. Until now, most distributed control strategies were designed for cooperative target observation and tracking using number of robots greater than or equal to number of targets [3]–[7]. A few studies was conducted with targets outnumbering robots [2]. If number of robots is less than number of targets, the robot swarm is not capable of tracking all the targets all the time. Beyond the current state of the art, we propose a distributed control strategy for a small swarm of mobile robots to observe and track unlimited number of targets over time in sequence. Specifically, each robot is capable of tracking and occupying a target for a short while, and then leaving to track next target while still maintaining the global network for cooperative target observation. Hence, an adaptive hierarchical distributed control strategy with cooperative task allocation enabling a swarm of mobile robots to track very large target clouds is the main objective of our study presented in this paper.

[‡]Pham Duy Hung and Trung Dung Ngo are with the More-Than-One Robotics Laboratory, University of Prince Edward Island, Canada. Ngo acknowledges the support of the Natural Sciences and Engineering Council of Canada (NSERC), DND-IDEaS program. dungnt@ieee.org

[†]Pham Duy Hung is also with the University of Engineering and Technology, Hanoi, Vietnam. hungpd@vnu.edu.vn

[§]Hung Manh La is with the Advanced Robotics and Automation (ARA) Laboratory, University of Nevada, Reno, USA. hla@unr.edu

Using a robot swarm for cooperative target observation and tracking has been intensively studied [3]–[9]. In [3], a seed growing graph partition(SGGP) algoithm was proposed for tracking and observing a moving target. The 3D model for multi-target observation by multiple robots was developed in [4]. In [8], a probability hypothesis density filter was applied for the robot control to estimate number of target and its location and Lloyd's algorithm was used to control robot motion in the multi-target searching and tracking process. Inspired from the predating behaviours of social animals, a coordination control in a competitive manner for target tracking was proposed in [9], in which only winners were allocated to task performance. In [6], the multi-robot task allocation with local communication was investigated to simultaneously assign trajectories and targets to robots. The concurrent region decomposition and allocation algorithm for multi-target tracking and surveillance missions of multi-robot system was studied in [7].

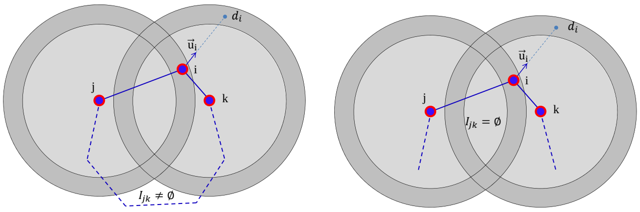
Our primary aim is to design an **a**daptive **h**ierarchical **d**istributed **c**ontrol (AHDC) with cooperative task allocation enabling a *small swarm of mobile robots to track unlimited number of targets*. We integrated the **h**ierarchical **d**istributed **c**ontrol (HDC) in [10] and our newly developed **c**ooperative target **o**bservation, **t**racking, and **r**eleasing algorithm (COTR). The original HDC consists of the distributed node control responsible for preserving the global network integrity of all the robots and the distributed connectivity control dealing with local connectivity topologies to allow the robots to move towards their assigned targets. The COTR is responsible as the coordinator facilitating the robots to track more than one target over time. We validated our methodological approach through simulation and real-world experiments.

The rest of paper is organized as follows. Swarm model, global network integrity preservation and critical connectivity minimization are addressed in section II. The architecture of AHDC is presented in section III. In section IV, experiment results and discussion are described and discussed. We draw remarks and conclusion in section V.

II. Preliminaries

A. Swarm Model

Consider a swarm of N mobile robots equipped with sensing capacity of measuring and estimating relative localization of its nearest neighbours N_i in a disk-based sensing range $\mathbf{S}_i$ within radius r_c. With an assumption of sensing range shorter than communication range, a robot is capable of peer-to-peer communication within the sensing range. Denote x_i



(a) Polygon topology (b) Chain topology

Fig. 1. Local connectivity topologies

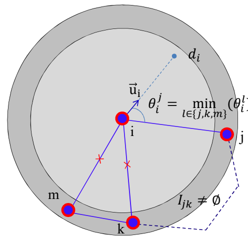


Fig. 2. A group of polygon topologies with $N_i^g = \{j, k, m\}$ is minimized by the rule described in Eq. 11. The connectivity e_{ij} is maintained because of $\theta_i^j = \min_{l \in N_i^g} (\theta_i^l)$ while other connectivities (red crosses) are removed.

and u_i as position and velocity of robot $i \in N$, respectively. Every robot in the swarm is described by a single-integrator kinematic model as follows:

$$\dot{x}_i = u_i, i \in N \tag{1}$$

The sensing range $\mathbf{S}_i$ is partitioned into two areas: *Critical area* $\mathbf{S}_i^c$ inside the annulus circle between two radii r_c and r_n, where $\varepsilon \triangleq r_c - r_n > 0$; and *non-critical area* $\mathbf{S}_i^n$ inside the circle with the radius r_n covering *obstacle avoidance area* $\mathbf{S}_i^a$ with the radius range $r_a < r_n < r_c$.

A network of N mobile robots is represented as an undirected graph $G = (V, E)$ where $V = \{1, ..., N\}$ is a set of N robots and $E = \{e_{ij} \mid i, j \in V, i \neq j\}$ is a set of connectivities formed by pairs of robot i and j. An adjacency matrix $\mathbf{A}$ of the graph G is a symmetric matrix in which each element e_{ij} represents the weight of the connectivity between a pair of robot i and j, determined by the relative distance $r_{ij} \triangleq \|x_i - x_j\|$:

$$e_{ij} = e_{ji} \triangleq \begin{cases} 1 & r_{ij} \leq r_c \\ 0 & r_{ij} > r_c \end{cases} \tag{2}$$

Let I_{ij} be a path inter-connecting immediate robots between two robot i and j, and $\ell_{I_{ij}}$ be a number of connectivities on the shortest path between them. Any two robots $i, j \in N$ can be connected either directly if $e_{ij} = 1$ or indirectly if $\exists I_{ij}$, where $\ell_{I_{ij}} > 1$. Mathematically, the connectivity property of the graph G can be estimated through the second smallest eigenvalue λ_2 of its Laplacian matrix, e.g., all the robots are considered as being connected if $\lambda_2 > 0$, and used for the control design [11], [12]. However, *in this study, λ_2 is not used for controlling connectivity in the distributed control, instead we only used λ_2 to verify our distributed control in preserving global network integrity as shown in Fig. 5.*

B. Global network Integrity Preservation

Based on sensing partition, the nearest neighbour set N_i of robot i is separated into critical robot set N_i^c and noncritical robot set N_i^n as follows:

Definition 1 (Critical Robot): Robot $j \in N_i$ is considered as a *critical robot* of robot i if $x_j \in \mathbf{S}_i^c$ and $\nexists k : x_k \in \mathbf{S}_i^n \cap \mathbf{S}_j^n$. Robot i's critical robot set is presented as follows:

$$N_i^c \triangleq \{j \mid x_j \in \mathbf{S}_i^c, \nexists k : x_k \in \mathbf{S}_i^n \cap \mathbf{S}_j^n\} \tag{3}$$

Definition 2 (Noncritical Robot): Robot $j \in N_i$ is considered as noncritical robot of robot i if $j \notin N_i^c$. Robot i's noncritical robot set is presented as follows:

$$N_i^n \triangleq \{j \in N_i \setminus N_i^c\} = N_i^{n_1} \cup N_i^{n_2} \tag{4}$$

where

$$N_i^{n_1} = \{j \mid x_j \in \mathbf{S}_i^n\} \tag{5}$$

$$N_i^{n_2} = \{j \mid x_j \in \mathbf{S}_i^c, \exists k : x_k \in \mathbf{S}_i^n \cap \mathbf{S}_j^n\} \tag{6}$$

As a result, $E_i^c = \{e_{ij} \mid j \in N_i^c\}$ and $E_i^n = \{e_{ij} \mid j \in N_i^n\}$ are defined as robot i's sets of *critical connectivities* and *noncritical connectivities*, respectively.

Proposition 1 (Global network Integrity Preservation): Global network integrity is preserved if every robot i has run-step Δx_i satisfying the following bounded constraint:

$$\Delta x_i \leq \begin{cases} \dfrac{\varepsilon}{2} & N_i^c = \emptyset \\ \dfrac{\varepsilon_i}{2} & N_i^c \neq \emptyset \end{cases}, \varepsilon_i = \min_{j \in N_i^c} (r_c - r_{ij}) \leq \varepsilon \tag{7}$$

Proof: A connectivity e_{ij} between robot i and j, where $i, j \in N$, is maintained if $\Delta x_i + \Delta x_j \leq r_c - r_{ij}$. Because robots i and j play the same role in maintaining their relative localization, robot i's run-step must satisfy the following inequality:

$$\Delta x_i \leq \frac{r_c - r_{ij}}{2} \tag{8}$$

If Eq. 7 is satisfied, the connectivities between robot i and $j \in N_i$, where $N_i = N_i^c \cup N_i^{n_1} \cup N_i^{n_2}$, are considered as follows:

- $N_i^c = \emptyset$: we have

$$\Delta x_i \leq \frac{\varepsilon}{2} \leq \min_{j \in N_i^{n_1}} \frac{(r_c - r_{ij})}{2} \tag{9}$$

Eq. 9 shows that every pair of robot i and $j \in N_i^{n_1}$, satisfy Eq. 8, i.e., all connectivities e_{ij} between robot i and $j \in N_i^{n_1}$ are maintained.

Consider a pair of robot i and $j \in N_i^{n_2}$. The definition 2 shows that there exists robot $k \in N_i^{n_1} \cap N_j^{n_1}$. Because the connectivities e_{ik} and e_{jk} maintained as proven above, robot i and j are connected together indirectly through intermediate robot $k \in N_i^{n_1} \cap N_j^{n_1}$. Hence, robot i is connected with every robot $j \in N_i$ either directly if $j \in N_i^{n_1}$ or indirectly if $j \in N_i^{n_2}$; that is, the global network integrity is preserved.

- $N_i^c \neq \emptyset$: we have

$$\Delta x_i \leq \frac{\varepsilon_i}{2} \leq \frac{\varepsilon}{2}, \text{ where } \varepsilon_i = \min_{j \in N_i^c}(r_c - r_{ij}) \quad (10)$$

Eq. 10 shows that every pair of robot i and $j \in N_i^c$ satisfy Eq. 8, i.e., all connectivities between robot i and $j \in N_i^c$ are maintained.

However, Eq. 10 covers Eq. 9, so robot i is connected with every robot $j \in N_i^{n_1} \cup N_i^{n_2}$ either directly if $j \in N_i^{n_1}$ or indirectly if $j \in N_i^{n_2}$. Hence, robot i is connected with every robot $j \in N_i$ either directly if $j \in N_i^c \cup N_i^{n_1}$ or indirectly if $j \in N_i^{n_2}$; that is, the global network integrity is preserved. ∎

C. Dealing with Complexity and Constraints of Local Connectivity Topologies

The global network integrity of robot i is preserved if run-steps of robot i and its critical robots satisfy Eq. 7. On the other hand, robot i's critical robots may act like "anchors" preventing robot i tracking desired targets, which can be considered as local minima in a network topology.

We define two typical local connectivity topologies illustrated in Fig. 1 as follows:

Definition 3 (Polygon Topology): Robot i and its pair of two neighbouring robots (j, k) are formed in a *polygon topology* if $j \in N_i^c$ and/or $k \in N_i^c$, and $\exists I_{jk} \setminus \{j, i, k\} \neq \emptyset$. A polygon topology becomes a triangle topology if $\ell_{I_{jk}} = 1$.

Definition 4 (Chain Topology): Robot i and its neighbouring robot j is formed in a *chain topology* if $\nexists I_{ij} \setminus \{i, j\} \neq \emptyset$.

A polygon topology containing redundant critical connectivities should be minimized to become a chain topology, which must be maintained for the global network integrity. Once redundant critical connectivities of robot i are minimized, robot i escapes from its local minima in order to track its desired targets.

Robot i's consecutive adjacent polygon topologies as shown in Fig. 2 are combined in a group of neighbouring robots N_i^g in which any two robots $j, k \in N_i^g$ are connected together by a path $I_{jk} \setminus \{j, i, k\} \neq \emptyset$. In the group, if robot i only maintains one critical connectivity e_{ij} and removes the other redundant critical links, then robot i is still connected with any robot $k \in N_i^g$ by a path $\{i, j, I_{jk}\}$ formed in a chain topology. Minimization of redundant critical connectivities of local connectivity topologies of robot i is to release its connectivity constraints and allow it to move towards far-reaching targets, thus critical connectivity e_{ij} on the target direction should be maintained and the others should be removed as stated in Eq. 11.

$$\theta_i^j = \min_{k \in N_i^g}(\theta_i^k) \quad (11)$$

where θ_i^k is an angle between robot i's velocity vector towards an assigned target and connectivity e_{ik}.

III. Adaptive Hierarchical Distributed Control

Based on the solid foundation of global network integrity preservation and minimization of redundant critical connectivities of local connectivity topologies in section II, we developed an adaptive hierarchical distributed control strategy (AHDC) enabling a *small swarm of mobile robots to track very large target clouds*. The AHDC is described as follows:

A. Distributed Node Control

Distributed node control based on behavioural control (BC) [13], [14] is responsible for controlling robot's motion. Robot i's velocity vector is synthesized from three parts: cohesion velocity $\vec{\mathbf{u}}_i^c$ controlling the robot to come close to its neighbours; separation velocity $\vec{\mathbf{u}}_i^s$ driving the robot to not collide with other robots, and alignment velocity $\vec{\mathbf{u}}_i^a$ guiding the robot to move closer to the desired target, as follows:

$$\vec{\mathbf{u}}_i = \alpha \vec{\mathbf{u}}_i^c + \beta \vec{\mathbf{u}}_i^s + \gamma \vec{\mathbf{u}}_i^a \quad (12)$$

where α, β, γ are parameters used to adjust weights of cohesion, separation, and alignment, respectively.

Proposition 1 implies that if a run-step of robot i's satisfy Eq. 7, the global network integrity of robots is preserved. Hence, the input control u_i is normalized by $u_{i_{max}}$ for network preservation (NP) as follows:

$$u_{i_{max}} = \begin{cases} \dfrac{\varepsilon}{2\Delta t} & N_i^c = \emptyset \\ \dfrac{\varepsilon_i}{2\Delta t} & N_i^c \neq \emptyset \end{cases} \quad (13)$$

where Δt is a time-step corresponding to run-step Δx_i.

B. Distributed Connectivity Control

The distributed connectivity control is implemented on the top of the distributed node control in order to allow the network to be expandable so robot i is capable of flexibly and adaptably moving towards a far-reaching target by minimizing removable critical connectivities of its local connectivity topologies. In the network expansion (NE) procedure, robot i uses peer-to-peer communication to identify local connectivity topologies and make a consensus decision of minimization of redundant critical connectivities with its neighbouring robots. From Eq. 11, a set of removable critical connectivities in a group of polygon topologies of robot i is identified as follows:

$$E_i^R = \{e_{ij}, j \in N_i^g \mid (\theta_i^j > \min_{k \in N_i^g}(\theta_i^k)) \wedge (\rho_{ij} = 1)\} \quad (14)$$

where $\rho_{ij} = \rho_{ji}$ is a consensus signal between robot i and j with the value 1 if robot j and i mutually agree to remove the connectivity; or 0 otherwise.

Consequently, robot i's set of critical connectivities E_i^c is updated by $E_i^c \leftarrow E_i^c \setminus E_i^R$. Critical connectivities in E_i^c are formed in chain topologies which must be preserved for the global network integrity by updating N_i^c for the mobility constraint in Eq. 13.

Algorithm 1: Adaptive Hierarchical Distributed Control

```
Goal ⟵ COTR (CallAlgorithm 2)
if Exist Goal then
    f =‖ N_i^c ‖
    switch f do
        case 0 do
            Activate BC
        case 1 do
            Activate BC and NP
        case > 1 do
            Activate BC and NP and NE
else
    Stop
```

Algorithm 2: Cooperative Target Observation, Tracking, and Releasing Policy

```
Input: robots and targets in sensing range
Output: Goal
switch State do
    case Free do
        if Observe targets then
            Assignto nearest target
            State ← Tracking
        else
            Goal: Toward nearest indicator
    case Tracking do
        Goal: Tracking assigned target
        if Observe other targets then
            Become an indicator
        if Reach assigned target then
            State ← Occupied
    case Occupied do
        if Given occupying time then
            Hold occupied target
        else
            Release to become free
```

C. Adaptive Hierarchical Distributed Control with Cooperative Target Observation, Tracking and Releasing

The adaptive hierarchical distributed control (AHDC) is a result of integrating the hierarchical distributed control (HDC) and the cooperative target observation, tracking and release (COTR). The AHDC uses a function $f = \| N_i^c \|$ to enable robot i to perform target observation, tracking and release in cooperation with its swarm. The function $f = \| N_i^c \|$ is used to activate the distributed node control (BC and NP) and the distributed connectivity control (NE) in the CORT process as follows:

- $f = 0$: Robot i and its neighbours are inside non-critical area of each other, thus it can easily move to reach its goal without any constraint by activating BC only.
- $f = 1$: Robot i is in a chain topology so it must activate both BC for motion control and NP for network preservation.
- $f > 1$: Robot i has more than one critical robots located in its critical area. Thus, robot i activates BC for motion control, NP for network preservation and NE for network expansion while it moves to reach its goal.

The operation of the AHDC is described in Algorithm 1. Note that the AHDC allows robots to adaptively deal with allocated tasks in different distribution intensities of targets so the robots are capable of not only tracking a target but also releasing its occupied target in order to track a new target in cooperation. Assume that locations of targets in a large cloud are priorly unknown to the robots due to their limited sensing range and the robots do not re-track the occupied targets. The COTR operates in the following states:

'Occupied': If robot i has successfully occupied a target.

'Tracking': If robot i has decided to track a target. At 'tracking' state, if robot i observes other targets within its sensing range, it becomes an *'indicator'*. An indicator operates as an explorer or a sub-leader that guides free robots to follow it in order to track new targets.

'Free': If robot i has not been decided to track any target.

The COTR operates in three modes: *target searching*, *target tracking*, and *target releasing*.

Target searching: 'Free' robot i is operating in the target searching mode. In this mode, robot i follows the nearest indicator to search for a new target observed by the indicator. If it has observed a number of unoccupied targets, the target closest to robot i is selected and robot i's state is changed from 'free' to 'tracking'.

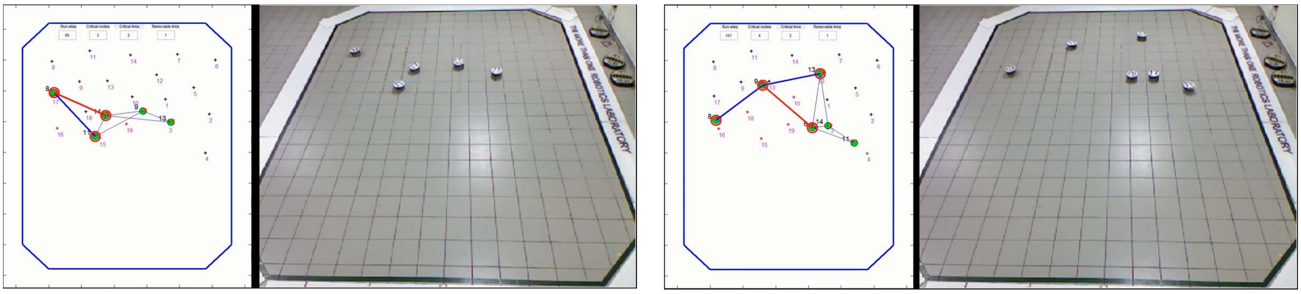
Target tracking: Robot i's moves towards the assigned target. Robot i stops and changes its state from *tracking* to *occupied* when it successfully reached the target. If robot i observes other targets, it becomes an indicator guiding free robots to track such targets.

Target releasing: At the occupied state, robot i releases its occupied target after a short while to become 'free' state so it can track other unoccupied targets. When it released the occupied target, its state changes to 'free', 'searching' or 'tracking', depending on its location, neighbours, and targets around it.

IV. Experiment Results and Discussions

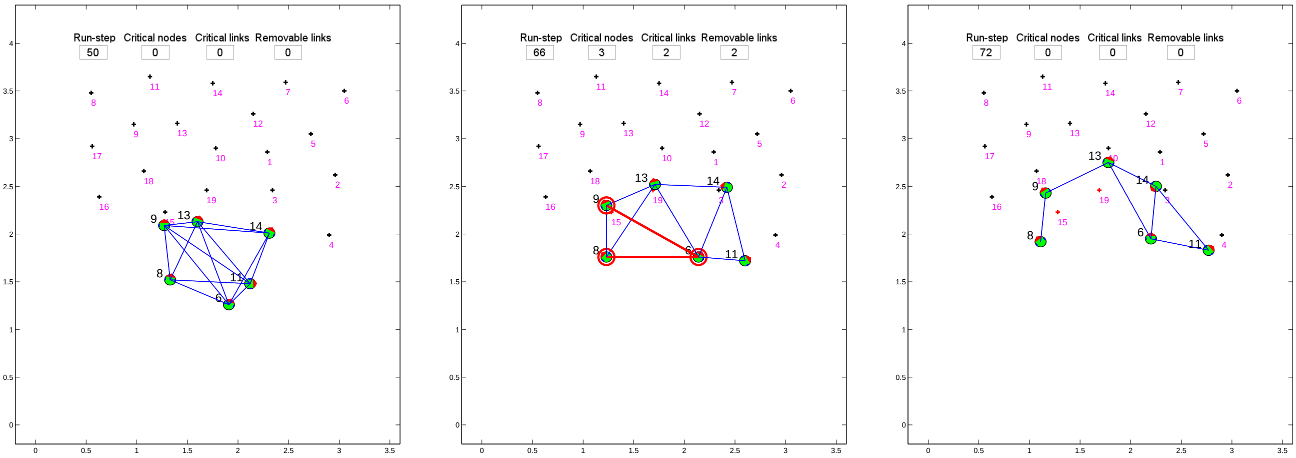
A. Experiment Setup

We used a small swarm of customized 14 cm diameter disc-like differentially driven wheel platforms for real experiments. The actual maximum speed of robots is $1.6m/s$, we set $u_{max} = 0.8m/s$ as their maximum velocity to ensure that the robots well response to motor commands. We used the NaturalPoint motion tracking system to reduce difficulties of representing sensing and communication of the robots.

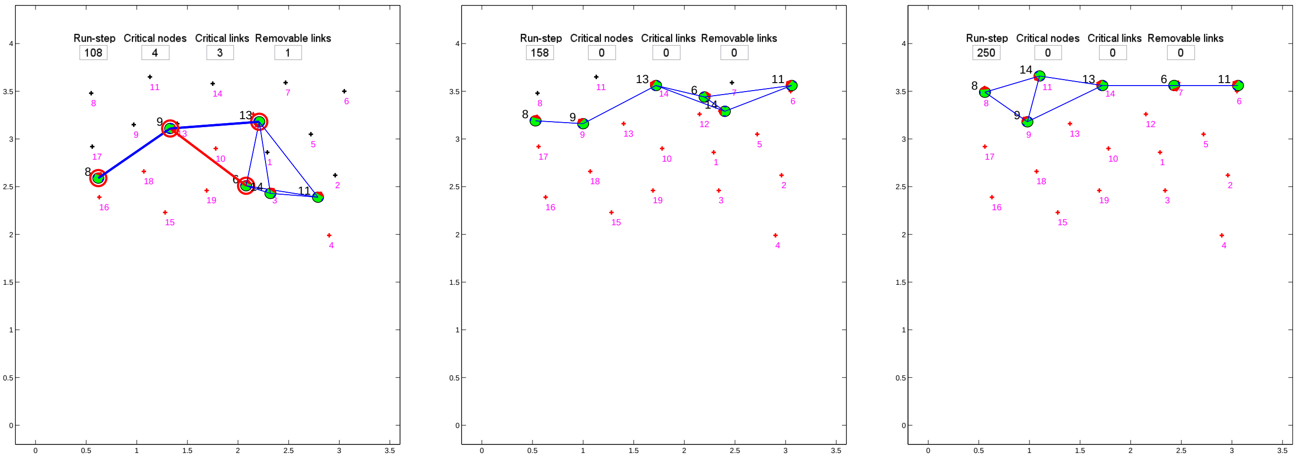


(a) Experiment 1 with 5 robots. (b) Experiment 2 with 6 robots.

Fig. 3. Two real-world experiments for cooperative target tracking of 19 targets.



(a) The swarm reached the first target (b) Critical connectivities removable for swarm dispersing (c) Swarm topology after critical connectivity removed



(d) e_{89} must be preserved but e_{96} is removable (e) e_{96} removed and swarm topology expanded (f) All target successfully occupied

Fig. 4. Snapshots of the swarm of six robots to track 19 targets.

Experiments were implemented in the arena $4m \times 5m$ with system parameters: $r_c = 1m$, $\varepsilon = 0.2m$, $\Delta t = 10ms$, $\alpha = 0.75$, $\beta = 1$, and $\gamma = 1$.

To create randomly generated experiment scenarios, we applied the Gaussian random distribution as a conditional function to drop targets into the experimental arena so that the target distribution graph - target cloud - is well connected under the constraint of radius r_c. With the generated target cloud, a robot can observe a new target when it occupied a target, hence it becomes an indicator to help other robots to occupy the target.

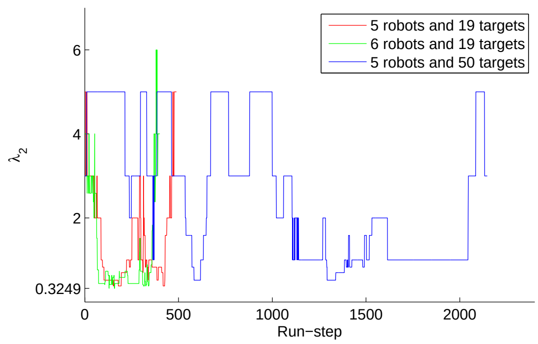


Fig. 5. Connectivity properties of the graph G(V,E) in the experiments over run-steps

B. Experiments and Discussions

We conducted two real experiments with a swarm of five and six robots assigned to track 19 randomly generated target cloud as shown in Fig. 3. The swarm of robots was placed at the opposite corner of the target cloud, thus the robots were not aware of the target locations at the initial stage. To start the experiments, we only pointed the direction of swarm movement toward the target cloud without concerning how the robots approach and track the targets. In the experiments, all the robots were installed with the AHDC as shown in Algorithm 1 to perform the cooperative target tracking.

Looking at the experiment of a swarm of 6 robots named $\{6, 8, 9, 11, 13, 14\}$ tracking 19 targets[1], all the robots were well connected together from the initial stage until reaching the first target as illustrated in Fig. 4(a). They automatically minimized their neighbourhood connectivities when dispersing to track and occupy targets. Zooming in a situation highlighted in Fig. 4(b), we observed that robot 6 dealt with its critical connectivities in the triangle topology, so it decided to remove two critical connectivities (red links) to escape from this local minima. After that, robots 8 and 6, 9 and 6 were formed in chain topologies as shown Fig. 4(c). We can also find a situation where critical connectivity e_{89} must be preserved because it was the unique link between robots 8 and 9 but critical connectivity e_{96} was removed to expand the network topology because robot 6 was able to communicate with robot 8 via robot 13, as depicted in Fig. 4(d), 4(e). We also observed several critical situations of connectivity maintenance and minimization in the real experiment of a swarm of 5 robots tracking 19 targets[2] and simulation of 5 robots tracking 50 targets[3]. In addition, thanks to the CORT depicted in Algorithm 2, each robot occupied a target, hold it for a while, then moved to track another target while still maintaining connectivity with its neighbours for the global network integrity and communication as shown in Fig. 5. Thanks to the HDC, the robot escaped from the local minima to reach its target. Our simulation and real experiment results show that the AHDC is capable of enabling a small swarm of robots to track and occupy large target clouds while preserving the global network integrity for cooperative task allocation.

[1]https://youtu.be/ECq06voeNhI
[2]https://youtu.be/FJCLHGn5fKs
[3]https://youtu.be/qT-gl2pfclY

V. Conclusion

We have presented the new adaptive hierarchical distributed control enabling a small swarm of mobile robot to track large target clouds. The distributed control was designed by integrating distributed node control and distributed connectivity control to allow a robot to adaptively deal with mobility constraints caused by local minima of local network topologies. The cooperative task observation, tracking, and releasing algorithm was incorporated on the distributed control to make it adaptable to track more than one target in large target clouds while preserving the global network integrity for cooperative task allocation. We demonstrated and validated our control method in both simulation and real-word experiments.

References

[1] C. Robin and S. Lacroix, “Multi-robot target detection and tracking: taxonomy and survey,” *Autonomous Robots*, vol. 40, no. 4, pp. 729–760, 2016.

[2] M. Senanayake, I. Senthooran, J. C. Barca, H. Chung, J. Kamruzzaman, and M. Murshed, “Search and tracking algorithms for swarms of robots: A survey,” *Robotics and Autonomous Systems*, vol. 75, pp. 422–434, 2016.

[3] H. M. La and W. Sheng, “Dynamic target tracking and observing in a mobile sensor network.” *Robotics and Autonomous Systems*, vol. 60, no. 7, pp. 996–1009, 2012.

[4] T. Nestmeyer, P. R. Giordano, H. H. Bülthoff, and A. Franchi, “Decentralized simultaneous multi-target exploration using a connected network of multiple robots,” *Autonomous Robots*, pp. 1–23, 2016.

[5] P. D. Hung, T. Q. Vinh, and T. D. Ngo, “A scalable, decentralised large-scale network of mobile robots for multi-target tracking,” in *Intelligent Autonomous Systems 13*. Springer, 2016, pp. 621–637.

[6] Y. Sung, A. K. Budhiraja, R. K. Williams, and P. Tokekar, “Distributed simultaneous action and target assignment for multi-robot multi-target tracking,” in *2018 IEEE International conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 1–9.

[7] E. Adamey, A. E. Oğuz, and Ü. Özgüner, “Collaborative multi-msa multi-target tracking and surveillance: a divide & conquer method using region allocation trees,” *Journal of Intelligent & Robotic Systems*, vol. 87, no. 3-4, pp. 471–485, 2017.

[8] P. Dames, “Distributed multi-target search and tracking using the phd filter,” in *2017 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. IEEE, 2017, pp. 1–8.

[9] L. Jin, S. Li, H. M. La, X. Zhang, and B. Hu, “Dynamic task allocation in multi-robot coordination for moving target tracking: A distributed approach,” *Automatica*, vol. 100, pp. 75–81, 2019.

[10] P. D. Hung, T. Q. Vinh, and T. D. Ngo, “Hierarchical distributed control for global network integrity preservation in multirobot systems,” *IEEE transactions on cybernetics*, 2019.

[11] L. Sabattini, N. Chopra, and C. Secchi, “Decentralized connectivity maintenance for cooperative control of mobile robotic systems.” *The International Journal of Robotics Research*, vol. 32, no. 12, pp. 1411–1423, 2013.

[12] A. Gasparri, L. Sabattini, and G. Ulivi, “Bounded control law for global connectivity maintenance in cooperative multirobot systems,” *IEEE Transactions on Robotics*, vol. 33, no. 3, pp. 700–717, 2017.

[13] C. W. Reynolds, “Flocks, herds and schools: A distributed behavioral model,” *SIGGRAPH Comput. Graph.*, vol. 21, no. 4, pp. 25–34, Aug. 1987.

[14] M. J. Mataric and M. J. Mataric, “Designing and understanding adaptive group behavior,” *Adaptive Behvior*, vol. 4, pp. 51–80, 1995.