An Improved Algorithm for Fault-tolerant Rendezvous of Multi-robot Systems with Reasonable Fixed Sensing Range

Jiani Li

Abstract

In this paper, we focus on the Fault-tolerant Rendezvous problem for multi-robot systems. The algorithm presented in [1] explores the problem of fault-tolerant rendezvous for synchronous multi-robot systems. However, the controllable sensing range it provided seems to be impractical. Though the system can achieve connectivity by controlling sensing range, it is unreasonable that the system chooses a small sensing range at first and later increases it if it can choose a larger one at the beginning. In light of this, we seek to improve the algorithm presented in [1] to achieve rendezvous while maintaining the connectivity of the system with a reasonable fixed sensing range and we prove that the system is fault-tolerant with a fixed sensing range larger than a certain value proportional to robots' range of motion.

1. Introduction

Rendezvous problem is a main task for multi-robot systems. To achieve rendezvous, robots must estimate their own configuration, maintain communication across the global network, as well as coordinate to make collective decisions. Algorithms to achieve these requirements need to operate at communication network and distributed computation. One challenge has been proposed for algorithms to achieve rendezvous for multi-robot systems: Through the process of rendezvous, the communication between robots may be lost, robots may not necessarily converge to one point (may converge to multiple points). Therefore, we must maintain connectivity of the multi-robot systems while achieving rendezvous.

The ADRC-Sync algorithm (which stands for approximate distributed robust convergence algorithm for synchronous algorithms) presented in paper [1] explored the problem of fault-tolerant rendezvous for synchronous multi-robot systems. However, the controllable sensing range it applies is considered unreasonable. Some of the previous work of [2-4] has come up with solutions to the coordination problem that preserve connectedness in the presence of limited sensing and communication ranges. However, such algebraic graphtheoretic tools are conflict with ADRC-Sync algorithm. In this paper, we are able to utilize a number of results from the area of connectivity maintenance using geometric approaches, especially in [5] and [6], to maintain connectivity of the system while achieving rendezvous.

In addition, we consider the case where some robots may be faulty. In most of the approaches hitherto, it is implicitly granted that all of the distributed agents will faithfully and precisely execute an agreement. Here, we consider the case in which some of the agents, termed faulty agents, fail to follow the policy. This can be induced by physical failure due to depletion of energy or component malfunction, such as sensor error. More generally, malicious robots should be taken into consideration, which are controlled by adversaries and try to maximally degrade the performance of coordination task.

Previous studies have dived into fields of fault-tolerant consensus and gathering problems [11-15]. However, these studies either assume that the consensus variable is restricted to the real-line or assume each fault-free robot can see all other robots in the workspace.

In this work, we seek to present a distributed algorithm for rendezvous of fault-free robots with limited disk sensing range in the presence of faults, provided that a few assumptions on the interconnection topology are satisfied. Also, we show via simulation results that the fault-free robots executing proposed algorithm converge to a point within a minor error-bound in the presence of stationary faults.

For communication model, each agent is equipped with a communication device that permits reliable broadcasting to all the other agents within fixed communication radius *r*. The service operates in synchronous rounds, and it has access to a positioning device. Finally, the service assumes the existence of a motion planner which is queried at each round for the desired target position, the service produces a trajectory which preserves connectedness and, when possible, gets closer to the target [5].

The paper is organized as follows. Section 2 introduces preliminary definitions, including notations, models and backgrounds of discrete geometry that will be necessary for the rest of the paper. Section 3 states the rendezvous problem that we are going to discuss in the paper. In section 4, we briefly describe the ADRC-Sync algorithm and in section 5, we add the connectivity maintenance property to form the improved ADRC-Sync-Conn algorithm and we also prove the system is fault-tolerant with a fixed sensing range larger than a certain value proportional to robots' range of motion in this section. Simulations are presented in section 6. And finally we conclude the paper in section 7.

2. Preliminary Definitions

2.1. Multi-robot systems

We consider a group of n autonomous mobile robots, the motion of which is contained in a bounded range.

Assume V is a set of n robots, where each robot $i \in V$ corresponds to a point $x_i \in \mathbb{R}^d = (x_{i1}, x_{i2}, \dots, x_{id})^T$ in Euclidean space, and $x = (x_1, x_2, \dots, x_n)$ corresponds to the set of robots' positions. We use $\|x_i - x_j\|$ to denote the Euclidean distance between robots, where i, $j \in V$. The topology of communication network of the multi-robot system can be modeled as an undirected graph $G = (V, E)$, where robots are represented by the set of vertices V and the set of edges E between vertices correspond to the communication links.

Given the set of edges, we define the neighborhood for a robot $i \in V$ as $N_i = \{ j \in V \mid \|x_i - x_j\| \leq s_i \}$, where s is the sensing range of robot i.

2.2. Faulty multi-robot systems (F-MRS)

Suppose some of the n robots are faulty. We denote the set of faulty robots as $F \in V$, and the number of faulty robots as n_f, where $n_f = |\mathrm{F}|$. The set of non-faulty or fault-free robots can be denoted by $\bar{V} = V - F$, and the number of fault-free robots is represented by $\bar{n} = n - n_f$. The graph consisted of only fault-free robots can be represented by $\bar{G}$. The number of faulty neighbors for each robot i can be represented by n_{fi} such that $n_{fi} = |N_i \cap F|$.

2.3. Useful results from discrete geometry

In this part, we list several results from discrete geometry regarding partitioning points into a number of intersecting convex hulls that are necessary for the rest of the paper.

Definition 2.1 (r-divisible point set [7]). *A set of n points is r-divisible if it can be partitioned into r pairwise disjoint subsets such that the intersection of the convex hulls of these r subsets is non-empty.*

Theorem 2.1 (Tverberg's Theorem [7]). *Each set of* $[(d+1)(r-1)+1]$ *points in* $\mathbb{R}^d$ *is r-divisible.*

Definition 2.2 ((r, k)-divisible point set [8]). *Each set of n points in* $\mathbb{R}^d$ *is* (r, k) *divisible if it can be partitioned into r pairwise disjoint subsets such that the intersection of the convex hulls of these r subsets is at least k dimensional*$(0 \geq k \geq d)$.

Conjecture 2.1 (Reay's conjecture [8]). *Each set of* $[(d+1)(r-1)+k+1]$ *points in general position in* $\mathbb{R}^d (0 \geq k \geq d)$ *is* (r, k)*-divisible.*

Reay's conjecture is known to be true for k = 0 (Tverberg's theorem [7]) or $\mathrm{d} \in \{2, 3, \dots, 7, 8\}$ [9], etc. For our faulty multi-robot system, it suffices to only consider d = 2, 3.

Corollary 2.1 (Roudneff [9]). *For d = 2, 3, and k = d, each set of* $[r(d+1)]$ *points in general position in* $\mathbb{R}^d$ *is* (r, d)*-divisible.*

Definition 2.3 (Interior Tverberg point). *Suppose X is a set of n points in* $\mathbb{R}^d$ *with index set I, and let* $X' \subset X$ *have index set* I'. *If* X' *is* $(n_f + 1, d)$*-divisible, then each point that is in the relative interior of the d-dimensional intersection of the disjoint subsets of* X' *is called a Interior Tverberg point of* (I', n_f).

In our system, correspond n robots to a set X of n points in $\mathbb{R}^d$. If X is $(r+1)$*-divisible*, then as long as up to r robots are faulty, any partition of X must exist at least one subset only consisting of fault-free robots.

A Lifting Algorithm [10] that runs in linear-time for a fixed d at the cost of decreased depth size is used to calculate Tverberg point. In order to use the such a method, we should make assumptions about the configurations of robots as follows.

Assumption 2.1 Each $\mathrm{x}(t)$ is in general position for all t.

By using approaches such as random perturbations, assumption 2.1 could be satisfied.

3. Problem Statement

In this paper, we focus on the problem of fault-tolerant rendezvous of multi-robot systems. The objective is to design a distributed algorithm that achieves rendezvous even in the presence of faulty robots. The system is synchronous. For each cycle, each robot conduct three sequential motions: Look, Compute and Move. In the Look motion, each robot takes a snap of its neighborhood, identifying all its neighbors' as well as its own positions. In the Compute motion, each robot computes its next move using the algorithm it employs to achieve rendezvous based on the positions it collected in the Look state. Each robot then moves to its computed next position in the Move state. Robots are oblivious, anonymous, and dimensionless, i.e., robots only compute its next move based on the information collected in the current cycle, robots cannot identify which of its neighbors are faulty and which are not, and robots can be located in the same point while never collide to each other.

4. ADRC-Syn Algorithm

In this section, we briefly describe the ADRC-Sync algorithm proposed in [1].

ADRC-Sync stands for approximately distributed robust convergence algorithm for synchronous algorithms. We have described the system properties in section 3 that each robot conduct three motions for each cycle. At iteration t, for each robot i, ADRC-Sync detect its neighbors using sensing range $S_i(t)$. Then, the algorithm commands each fault-free robot to compute the estimate faulty neighbors $\widetilde{n_{fi}}$ as

$$\widetilde{n_{fi}}(t) = \left\lceil \frac{|N_i(t)|}{2^d} \right\rceil - 1$$

then, obtains $(N_i(t), \widetilde{n_{fi}}(t))$ -Tverberg point, and assigns to $s_i(t)$.

Here, we briefly explain how to obtain a Tverberg point by using lifting approach. Note that the Tverberg point is not unique. Fig.1 shows the procedure. Given n points distributed in $\mathbb{R}^d$ in general positions. In order to get the Tverberg point of n points, first, project them to a hyperplane H and get the Tverberg point/partition of the projection in Fig.1(a). Then, draw a line l perpendicular to the hyperplane H cross Tverberg point of the hyperplane, i.e., C', and get the intersections with convex hull of lifted Tverberg partition in Fig.1(b). Finally, find the median C as a Tverberg point and corresponding partition for the intersections along l, points are regrouped by the obtained partition in Fig.1(c). The depth of the resulting Tverberg partition is $\lceil n/2^d \rceil$. Complete details can be found in [10].

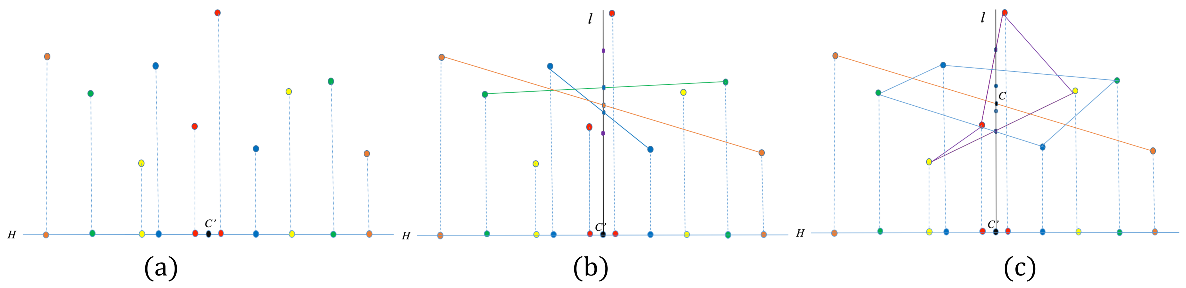

Fig.1. Procedure to obtain a Tverberg point by lifting method

After each fault-free robot calculates a $(N_i(t), \widetilde{n_{fi}}(t))$-Tverberg point, the robot computes its next position $x_i(t+1)$ by

$$x_i(t+1) = x_i(t) + u_i(t)$$

where

$$u_i(t) = \alpha_i(t)(s_i(t) - x_i(t))$$

$\alpha_i(t)$ is dynamically chosen parameter in the range, $0 \le \alpha_{min} \le \alpha_i(t) \le \alpha_{max} \le 1$, such that $u_i(t)$ does not violate constraints, e.g., maximum allowable displacement per stage. Then, each fault-free robot adjusts its sensing range to preserve connectivity with its neighbors by

$$s_i(t+1) = v_{max} + max_{j \in N_i(t)} \|x_j(t) - x_i(t+1)\|$$

To make sure at each stage each fault-free robot can obtain $(N_i(t), \widetilde{n_{fi}}(t))$-Tverberg point, ADRC-Sync algorithm has several assumptions on connectivity.

Assumption 4.1 (Neighborhood size). For a F-MRS, for all $t \in \mathbb{Z} \ge 0$, each $i \in \bar{V}$ has at least $n_{fi}(t)2^d + 1$ number of neighbors.

Assumption 4.2 (Global reachability). For a F-MRS, the initial graph $\bar{G}(0)$ has a globally reachable node.

In ADRC-Sync, every fault-free robot moves toward a Tverberg point that is in the convex-hull of fault-free robots, which does not depend on the position of faulty robots. Thus, regardless of the motions of faulty robots, fault-free robots converge to a point. Therefore, ADRC-Sync algorithm is fault-tolerant.

The ADRC-Sync algorithm uses the controllable sensing range that makes each fault-free robot never lose connectivity with its neighbors. [1] also states first, not a valid connectivity constraint exists for ADRC-Sync algorithm; second, even if the connectivity can be enforced by an algorithm where the sensing range for each robot is invariant, it is not possible to impose the constraint without sacrificing fault-tolerance capability. However, through our work, we find that given the motion range, we can set a fixed sensing range to each fault-free robot executing ADRC-Sync algorithm and enforce a connectivity constraint on each fault-free robots to maintain connectivity of the fault-free graph while preserving fault-tolerant properties. Besides, not constrain connectivity properties on ADRC-Sync algorithm may not necessarily achieve convergence, since the possibility of the case where fault-free robots converge to more than one points, see Fig.2. Therefore, we need a more robust algorithm, which will be introduced in the next section.

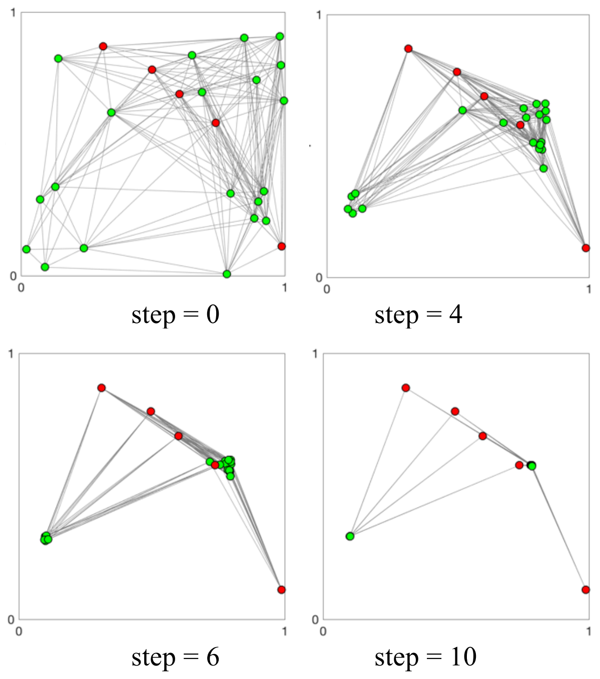

Fig.2. Case of converging to more than one points for ADRC-Sync algorithm (green points—fault-free robtos;

red points— stationary faulty robots; lines—connection between two robots, similarly hereinafter)

5. ADRC-Sync-Conn algorithm

In this section, we present an improved algorithm which achieves rendezvous while maintaining the connectivity of the system with a reasonable fixed sensing range, called ADRC-Sync-Conn algorithm, which stands for ADRC-Sync Connectivity Mainenance algorithm. And we prove that the system is fault-tolerant with a fixed sensing range larger than a certain value proportional to robots' range of motion.

It consists of a three-phase cycle. For each cycle, each fault-free robot executes three phases — Collection phase, proposal phase, and adjustment phase. The Connectivity maintenance protocol is a simplified vision of that in [5].

In the collection phase, each robot first queries its own position by devices such as GPS, and then broadcasts its position and records the position of neighboring robots discovered within its communication radius. After collecting the neighborhood, each robot calculate its next move target based on ADRC-Sync algorithm using a fixed sensing range. The current and target positions for robot *i* can be represented by s_i and t_i respectively.

In the proposal phase, each robot *i* computes R_i as the intersection of all the disks of sensing range *r* centered at the positions of each of the neighbors of *i*. The proposal target p_i for robot *i* is then computed as the point inside R_i and is closest to its desired target t_i calculated by the last phase. We get the proposal target p_i is optimistic in the sense that if none of its neighboring agents move, then moving from source s_i to the target p_i would not disconnect the network. The proposed target p_i is broadcast and the proposals of other agents are collected.

In the adjustment phase, the algorithm demands every robot *i* to first check whether the distance between its own proposal and the proposals of its neighbors is at most *r*, if yes, then it will return a linear trajectory from s_i to p_i. If not, it will return a linear trajectory from s_i *to* $s_i + \frac{1}{2}(p_i - s_i)$.

The pseudocode of ADRC-Sync-Conn algorithm is provided in the table Algorithm 1.

5.1. Connectivity maintenance proof

In this part, we briefly prove the ADRC-Sync-Conn algorithm preserves network connectivity. The proof refers to [5].

Since R_i is the intersection of a set of disks that contains s_i, by moving through linear trajectory from s_i to p_i, which also located in R_i, the agent *i* will not lose connectivity with all its neighbors given all other agents stay stationary. In the next, we prove that the adjusted proposals of symmetric neighbors are connected.

Lemma 5.1 The adjusted proposals of symmetric neighbors are connected.

Proof The adjusted proposals of symmetric agents *i* and *j* are $p_i{}' = s_i + \frac{1}{2}(p_i - s_i)$ and $p_j{}' = s_j + \frac{1}{2}(p_j - s_j)$. Since $\|s_i - p_j\| \le \mathrm{r}$ and $\|s_j - p_i\| \le \mathrm{r}$, the distance between the adjusted proposals of *i* and *j* are

$$\|p_i{}' - p_j{}'\| = \left\|s_i - s_j + \frac{1}{2}(p_i - s_i) - \frac{1}{2}(p_j - s_j)\right\| \le \frac{1}{2}(\|s_i - p_j\| + \|s_j - p_i\|) \le r$$

thus, the adjusted proposals are connected.

For ADRC-Sync-Conn algorithm, we have the same assumptions about the connectivity as the ADRC-Sync algorithm.

Assumption 5.1 (Neighborhood size). For a F-MRS, for all $\mathrm{t} \in \mathbb{Z} \ge 0$, each $i \in \bar{V}$ has at least $n_{fi}(t)2^d + 1$ number of neighbors.

Assumption 5.2 (Global reachability). For a F-MRS, the initial graph $\bar{G}(0)$ has a globally reachable node.

Algorithm 1: ADRC-Sync-Conn

```
Collection phase:
$s_i$ ← query_positioning_device()
broadcast $s_i$ to all neighbors
$N_i$ ← $\{s_j \mid$ for each $s_j$ received$\}$
$t_i$ ← query_motion_planner()
Proposal phase:
$R_i$ ← $\cap_{s_j \in N_i} \mathrm{disk}_r(s_j)$
$p_i$ ← $\mathrm{argmin}_{p \in R_i} \|p - t_i\|$
broadcast $p_i$ to all neighbors
$P_i$ ← $\{p_j \mid$ for each $p_j$ received$\}$
Adjustment phase:
If $\forall\ s_j \in N_i \Rightarrow \|p_j - p_i\| \le \mathrm{r}$ then
    return next move from $s_i$ to $p_i$
else
    return next move from $s_i$ to $s_i + \frac{1}{2}(p_i - s_i)$
end if
```

5.2. Invariant Sensing Range

It is not possible to impose the connectivity constraint without sacrificing fault-tolerance capability. For instance, suppose a faulty robot moves far away from all the other robots, fault-free robots initially connected to the faulty robot can do nothing but to follow the faulty robot in order to maintain connectivity. In such cases, fault-free robots may not be able to achieve rendezvous. However, if satisfying the constraints on the magnitude of the sensing range for fault-free robots, the ADRC-Sync-Conn algorithm can be fault-tolerant.

In this part, we prove that the system is fault-tolerant with a fixed sensing range larger than a certain value proportional to robots' range of motion, i.e., half of the largest distance of two randomly chose points within the motion range of robots.

Proposition 5.1 Given the motion range of robots, the ADRC-Sync-Conn algorithm is fault-tolerant with an invariant sensing range equal to or greater than half of the largest distance of two randomly chose points within the motion range of robots.

Proof Here we only consider the case of stationary faults, because of the complexity of dynamic faults, it is not proved that there does not exist one malicious protocal that can disturbe the convergence consensus of fault-free robots. First, suppose only one stationary faulty agent exists within the motion range. Since robots employing ADRC-Sync-Conn algorithm always seek to move closer to their neighbors, and in each of the following cases, converge will be achieved: 1. The faulty robot is connected to the communication graph in the initial state; 2. The faulty robot is not connected to the communication graph. In the first case, since the faulty robot is stationary, and at least one of the fault-free robot is connected to it, because of the connectivity maintenance protocol, this connection will never lose. Therefore, robots will converge to one point and this point need to be within the sensing range (geographically, not functionally) of the faulty robot. In the second case, if this faulty robot never be connected to the communication graph through stages, then the case can be simolified as fault-free-robot-only case. Therefore, rendezvous will always be achieved. However, if through stages, the faulty robot become connected to the communication graph, then it is the same as case 1. For only one stationary faulty agent, no sensing range constraint is imposed.

Now, consider the case of two stationary faults. Similar to the above examples, fault-free robots also seek to converge as well as maintaining connectivity to its neighbors. The difference is that in the case of two faults exist, the two faults can be allocated far from each other so that when two faults are both connected to the communication graph, in order to maintain connectivity with faults, fault-free robots connected to them will always follow them. Therefore, we could expect the situation where fault-free robots are divided to two groups, each group is connected to one of the two faulty robots, i.e., converge to two points. The case is shown in fig.3.

Therefore, in order to avoid the case of Fig.3, we can set the fixed sensing range equal to or greater than half of the largest distance of two randomly chose points within the motion range of robots. Under this constraint, robots will always converge to one point, and in the limiting case, when the two faults are allocated at the farthermost two points withn the motion range, fault-free robots can also be connected to both of the faulty robots and converge to the intermediate point of two faults.

When we introduce more faults, the case is the same. Since the sensing range satisfys that the final convergence point will always be connected to the farthermost two points, it will always be connected to other faults within this motion range.

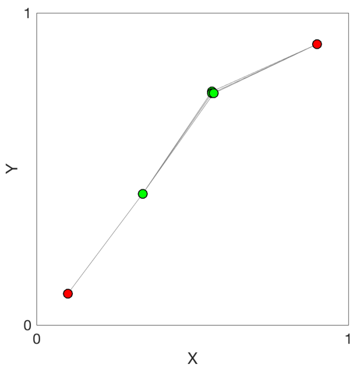

Fig. 3. The case of fault-free robots converging to two points when two faulty robots allocated far from each other

6. Numerical Simulation

Suppose the motion range for robots, including fault-free and faulty ones, is a plane space W=$[0,1]\times[0,1]\in\mathbb{R}^2$. For interconnection topology for fault-free robots, a disk graph is employed. First, we consider the case where only fault-free robots exist. In this case, we do not have constraints on the magnitude of the sensing range as long as it satisfys the assumption that the initial graph is connected. Intuitively, given more robots, the magnitude of the sensing range can set to be smaller, since more robots means greater density, and stronger connectivity. Fig.4 shows the configuration through stages in the case of 20 fault-free robots, the sensing range is set to be 0.5. Next, let's introduce faulty robots. We have proved insection 5.2 that the algorithm is fault-tolerant with the sensing range equal to or greater than half of the largest distance of two randomly chose points within the motion range of robots, in this case,$\sqrt{2}/2$. We choose to set 0.71 as the sensing range here. Fig.5 shows the configuration through stages in the case of 20 fault-free robots and 10 faulty robots.

we show via simulation results that the fault-free robots executing proposed algorithm converge to a point within a minor error-bound with or without stationary faults.

7. Conclusions

In this paper, we show the insufficients of ADRC-Sync algorithm and propose an improved one called ADRC-Sync-Conn algorithm that introduces connectivity maintenance protocol on the base of ADRC-Sync algorithm. We also proved that given the motion range of robots, the ADRC-Sync-Conn algorithm is fault-tolerant with an invariant sensing range equal to or greater than half of the largest distance of two randomly chose points within the motion range of robots. Through numerical simulation, the robustness of the algorithm is proved—that is, with or without fault robots, the convergence of fault-free robots can always be achieved by the given sensing range.

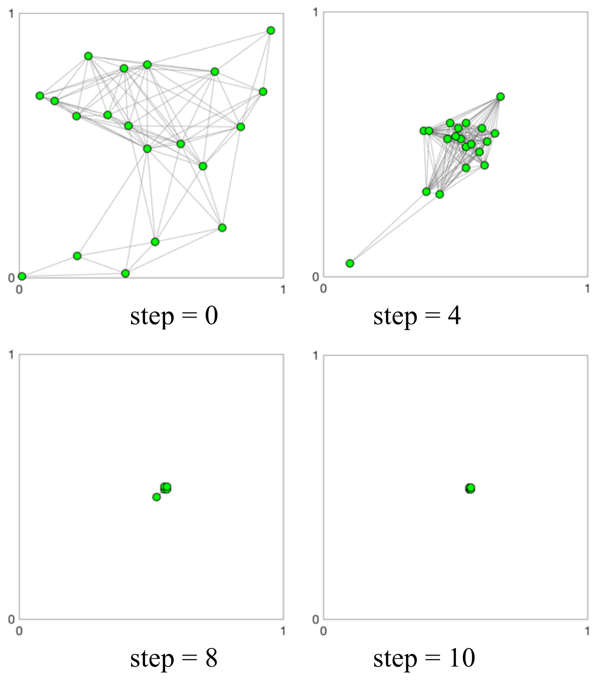

Fig.4. Configuration through stages in the case of 20 fault-free robots, sensing range 0.5

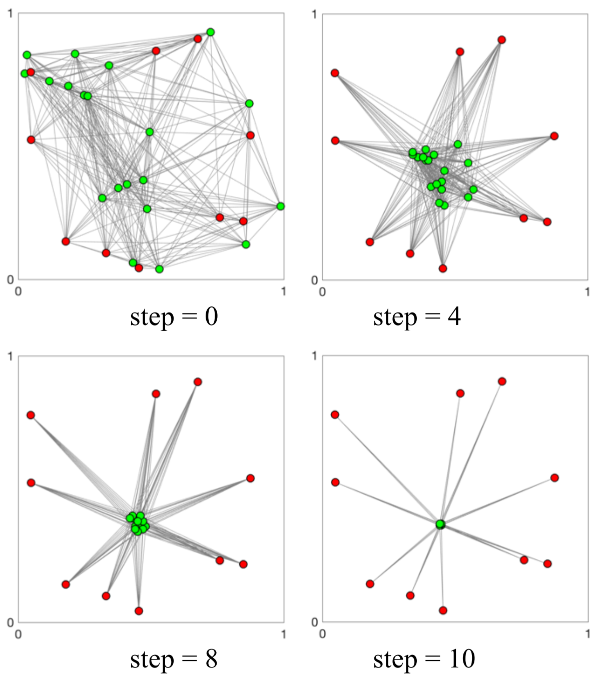

Fig.5. Configuration through stages in the case of 20 fault-free robots and 10 faulty robots, sensing range 0.71

References

[1] Hyongju Park and Seth Hutchinson, *An Efficient Algorithm for Fault-tolerant Rendezvous of Multi-robot Systems with Controllable Sensing Range*. 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, May 16-21, 2016.

[2] M. Ji and M. B. Egerstedt, *Distributed coordination control of multiagent systems while preserving connectedness*. IEEE TRANSACTIONS ON ROBOTICS, vol. 23, No. 4, August 2007.

[3] Michael M. Zavlanos, Magnus B. Egerstedt, George J. Pappas, *Graph-Theoretic Connectivity Control of Mobile Robot Networks*. Proceedings of the IEEE, Vol.99, No. 9, September 2011.

[4] Zavlanos, M.M., Pappas, G.J., *Distributed Connectivity Control of Mobile Networks*. Robotics, IEEE Transactions on, vol.24, no.6, pp.1416-1428, Dec. 2008

[5] Alejandro Cornejo, Fabian Kuhn, Ruy Ley-Wild, and Nancy Lynch, *Keeping Mobile Robot Swarms Connected*, DISC 2009, LNCS 5805, pp. 496–511, 2009.

[6] A. Cornejo and N. Lynch, *Connectivity Service for Mobile Ad-Hoc Networks*. Spatial Computing Workshop, 2008.

[7] H. Tverberg, *A generalization of radon's theorem*. J. London Math.

[8] J. R. Reay et al., *An extension of radon's theorem*. Illinois Journal of Mathematics, vol. 12, no. 2, pp. 184–189, 1968.

[9] J.-P. Roudneff, *Partitions of points into intersecting tetrahedral*. Discrete Mathematics, vol. 81, no. 1, pp. 81–86, 1990.

[10] W. Mulzer and D. Werner, *Approximating tverberg points in linear time for any fixed dimension*. Discrete & Computational Geometry, vol. 50, no. 2, pp. 520–535, 2013.

[11] M. Zhu and S. Mart´ınez, "On distributed constrained formation control in operator–vehicle adversarial networks," Automatica, vol. 49, no. 12, pp. 3571–3582, 2013.

[12] A. Gusrialdi, Z. Qu, and M. A. Simaan, "Robust design of cooperative systems against attacks," in American Control Conference (ACC), 2014. IEEE, 2014, pp. 1456–1462.

[13] S. Sundaram and C. N. Hadjicostis, "Distributed function calculation via linear iterations in the presence of malicious agents-part I: Attacking the network," in American Control Conference, 2008. IEEE, 2008, pp. 1350–1355.

[14] X. D´efago, M. Gradinariu, S. Messika, and P. Raipin-Parv´edy, Faulttolerant and self-stabilizing mobile robots gathering. Springer, 2006.

[15] Z. Bouzid, S. Das, and S. Tixeuil, "Gathering of mobile robots tolerating multiple crash faults," in Distributed Computing Systems (ICDCS), 2013 IEEE 33rd International Conference on. IEEE, 2013, pp. 337–346.