Multi-robot Rendezvous Based on Bearing-aided Hierarchical Tracking of Network Topology

Shaocheng Luo[a], Jonghoek Kim[b,*], Ramviyas Parasuraman[a,c], Jun Han Bae[a], Eric T. Matson[a,d], Byung-Cheol Min[a]

[a] *Computer and Information Technology, Purdue University, West Lafayette, IN 47907, USA*

[b] *Hong Ik University, Sejong, South Korea*

[c] *Department of Computer Science, University of Georgia, Athens, GA 30605, USA*

[d] *Department of Electrical Engineering, Kyung Hee University, Yongin, South Korea*

Abstract

Rendezvous control of multiple robots without losing network connectivity has important implications in multi-robot system including formation control, coordinated task assignments, and cooperative robotic missions. This paper introduces a new coordinate-free, bearing-based algorithm to enable rendezvous of distributed mobile robots at any designated leader robot node using hierarchical tracking of wireless network topology. An assumption is made that the robot can only sense and communicate with their neighbors (i.e., local sensing). The proposed approach preserves connectivity during the rendezvous task, adapts to dynamic changes in the network topology (e.g., losing or re-gaining a communication link), and is tolerant of mobility faults in the robots. We theoretically analyze the proposed algorithm and experimentally demonstrate the approach through simulations and extensive field experiments. The results indicate that the method is effective in a variety of realistic scenarios in which the robots are distributed in a cluttered environment.

Keywords: rendezvous control, networked robots, multi-robot systems, connectivity maintenance, graph theory

*Corresponding author.

Email addresses: luo191@purdue.edu (Shaocheng Luo), jonghoek@hongik.ac.kr (Jonghoek Kim), ramviyas@purdue.edu (Ramviyas Parasuraman), bae21@purdue.edu (Jun Han Bae), ematson@purdue.edu (Eric T. Matson), minb@purdue.edu (Byung-Cheol Min)

1. Introduction

Networked mobile robots have tremendous potential for applications such as monitoring and securing large complex environments, in search and rescue, first response, and multi-purpose mobile sensor networks [1, 2, 3, 4, 5, 6, 7]. Rendezvous control is an important module of a multi-robot system to enable formation control of multiple robots without losing network connectivity. Consider the formation control of multiple robots using a leader robot [8, 9]. The leader robot has a communication module that is superior to those of other robots. Thus, the operator of the multi-robot system can communicate with the leader robot and control the leader robot. All robots, except for the leader, move based on the local interaction with nearby robots. To maintain network connectivity between the robots, we use the rendezvous control occasionally so that all robots move closer to the leader robot while avoiding a collision. In this way, we can control the multi-robot system without losing network connectivity.

Specifically, *rendezvous control* is a strategy to enable rendezvous of all robots at a designated leader robot. Figure 1 shows an example of three mobile robots rendezvousing at a static leader robot (control station).

We consider a realistic scenario in which the robots do not have a global localization[1] or a positioning framework, i.e., the problem of rendezvous without coordinates [10]. A solution can be achieved through bearing-only or range-only control methods [11]. In this scenario, the robots cannot move to a designated position directly but only with a cooperative (or distributed) network control approach.

There has been a significant amount of research done in distributed control methods to achieve rendezvous based on consensus among robots (e.g., [12, 13, 14]). However, instead of rendezvous at an arbitrary point, we focus on rendezvous at a specific (mobile) node that can be dynamically assigned and updated. We call this *any-node* rendezvous, which has various applications such as sensor deployments [15], multi-agent coordination based on leader selection (rendezvous at the leader robot) [8], rendezvous at a des-

[1] A Global Position System (GPS) cannot be used in some cases. For instance, consider the case where robots are deployed in underground or underwater environments and locations where the GPS satellites do not have good reachability.

Figure 1: A representative figure showing three robots arriving at a control station (leader robot).

ignated position for automated multi-agent recharging [16], and collective transport of multiple robots [17].

In this paper, we propose a rendezvous control strategy to achieve *any-node* rendezvous without coordinates. Particularly, we propose a bearing-aided hierarchical tracking method that exploits the dynamic nature of a wireless network topology. To the best of our knowledge, our paper is unique in achieving multi-robot rendezvous by using wireless communication sensing among robots. The proposed control mechanism enables unique advantages such as detection and a fault-tolerance mechanism for mobility or communication failures in the robots. We base our approach on graph theory and local sensing. The proposed bearing-based control algorithm explicitly bounds the control inputs[2] (e.g., upper bound of robot velocity) to realize a platform-independent rendezvous strategy. The main contributions of this paper are as listed below:

- We propose a coordinate-free bearing-aided rendezvous control method and discuss its theoretical properties such as guaranteed convergence

[2]Note the control input bounds are determined by hardware limitations (finite forces or torques).

and connectivity maintenance during the rendezvous task;

- We implement the proposed strategy by extending a wireless signal tracking navigation system based on the one proposed in our previous work [18], in which the bearing (direction of arrival) of neighboring robots is estimated using the RSSI (Received Signal Strength Indicator) of a rotating Wi-Fi directional antenna on the robot.
- We integrate a bearing-aided obstacle avoidance strategy in the rendezvous control algorithm;
- We conduct comprehensive field experiments and simulations to verify and validate the proposed rendezvous method.

The main advantages of the proposed rendezvous method include: (1) operation in cluttered and dynamic environments (no prior knowledge of obstacles), and (2) fault-tolerance (e.g., faults in robot mobility or communication).

2. Related Work

Rendezvous in a multi-robot system has been studied extensively in the literature because of the wide potential applications such as in formation control [19], coordination among robots [20], cooperative control [21], and autonomous recharging [22]. A theoretical graph approach has been the base of many rendezvous control solutions as it provides a theoretical foundation to propose and validate control laws [23, 24, 25, 26].

Notably, in most of the works that addressed the problem of multi-robot rendezvous, rendezvous control is performed by sharing robot state (e.g., position) and achieving a consensus among the robots either in a centralized or in a distributed manner. For instance, in [27], the rendezvous happens at the centroid location of all the robots (geometric consensus). In [28], a rendezvous control solution is proposed using an average of bearings (headings) of the nearest neighbors, so that all robots meet at the same point over some iterations.

Distributed, bounded control laws that prove convergence can enable rendezvous while maintaining connectivity among the robots [29, 30, 13]. In [31, 32], circumcenter-based consensus algorithms are proposed, which until now has predominated the rendezvous literature. However, consensus algorithms cannot work with the presence of faulty robots, link failures, and

unknown disturbances. Therefore, rendezvous strategies that are tolerant to known or unknown failures have been proposed in the literature using a robust PID control method [33] or control protocol applicable to rectangle-like robot distributions [34]. Although these methods are restricted regarding geometry or control implementations, they advance the work toward robust robot rendezvous.

A major advancement in fault-tolerant rendezvous framework was made in [35] in which the authors use Voronoi partitioning of safe regions between robots to assure rendezvous tasks, even when unidentified broken robots exist. However, this method requires either a controllable sensing range or a densely-connected communication graph. Meeting these requirements may not always be feasible in real-life applications. Therefore, we aim to address the problem of fault-tolerant rendezvous that can operate even in sparsely-connected graphs with a limited sensing range, assuming that fault-detection is in place to detect faults in robots.

Similar to the work in [11, 36], we explicitly consider the upper bound of the rendezvous control inputs, especially maximum robot velocity S_m, which is determined by the kinematic constraints of the robot platform. This expands the applicability of our controller in a variety of robot platforms.

Importantly, an additional consideration is that the efficacy of a rendezvous task has to be guaranteed despite the lack of global coordinate systems, which is the motivation behind coordinate-free rendezvous approaches. For example, the authors in [10, 11, 37, 38] proposed bearing-only control laws to achieve consensus among robots. Taking inspiration from these approaches, we consider a coordinate-free rendezvous problem in which only bearing sensors are used for distributed sensing and control in a rendezvous algorithm. However, such works cannot be used for the *any-node* rendezvous task, since they do not assure that all robots meet at a predefined location. Moreover, the works mentioned above are not fault-tolerant regarding link or mobility failure. Therefore, we do not directly compare our work with the state-of-the-art consensus-based rendezvous methods.

In practical applications, robots are deployed in a cluttered environment with many obstacles (e.g., [39]). In such scenarios, it is important to integrate collision avoidance methods in the rendezvous algorithm. In the literature, artificial potential fields have been primarily used to avoid obstacles [40]. However, in our work, we implement a bearing-based obstacle avoidance system as presented in [18, 41] in our rendezvous algorithm, as it can be readily integrated into our bearing-based rendezvous controller.

A communication graph hierarchy (i.e., a tree structure) has been proven useful in the exploration of unknown environments by a team of robots while maintaining links between the robots [42]. Also, in [43], a hierarchical tree structure was used to achieve cyclic-pursuit rendezvous of multiple robots. Therefore, we use a network/communication graph as the basis of our proposed algorithm.

Borrowing several ideas and motivations from these works, we design a coordinate-free bearing-based robot controller that tracks the hierarchy of the wireless network topology to coordinate the *any-node* rendezvous task in a distributed fashion while integrating a collision avoidance system using onboard sensors. We also extend the problem space to enable fault-tolerance in the controller regarding robustness for link or mobility failures. Specifically, the proposed algorithm adapts to dynamic changes in the network topology (e.g., losing or re-gaining a communication link), and is tolerant of identifiable mobility failures.

3. Background

This section reviews background information that is used in the proposed rendezvous control.

3.1. Graph theory

Following graph theory [44], $G = (V, E)$ denotes an undirected weighted graph with node set V and edge set E. Every edge in E has its weight $w(e) :\rightarrow Z^+$. A graph G is *connected* if there exists a path (sequence of connected edges) between every pair of distinct nodes. A tree is a connected graph without cycles. A *rooted tree* has one node which is set as the *root*. Each node in a rooted tree has a parent-child relationship with its neighboring nodes. In a rooted tree, $p(v)$, the *parent* of a node v, is the node adjacent to v on the path to the root. $c(v)$, *child* of a node v, is a node of which v is the parent. A *leaf* in a tree graph is a node having no child. Figure 2 illustrates a connected tree graph, in which $c(v)$ is a leaf.

The *subgraph* of G induced by a node set $S \subset V$ is the graph (S, E_S) in which $E_S = \{\{x, y\} \in E : x, y \in S\}$. In a tree graph, a path between any two nodes is unique. Given a connected, edge-weighted graph G, a *shortest-path tree* $T \subset G$ is a tree, such that a path in T, between any node u and the root, is the shortest-path in G between the two nodes.

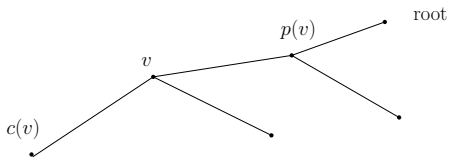


Figure 2: Illustration of a connected tree graph.

3.2. Assumptions and definitions

We introduce assumptions and definitions that drive our control laws. Let $\mathbf{q}_i \in R^2$ denote the position of a robot i. We assume that each robot is equipped with a wireless (ad-hoc) network device and a bearing sensor to measure the relative bearing of neighboring robots. The robots sense neighboring robots and share this information across the network, which we use to build and dynamically update the network topology. The proximity information from local sensors (e.g., sonar, radar, and lidar), if available, can be used to assign weights to the edges in the network topology. If such range sensors are not available, then each edge will have an equal weight in the network graph. Note, we assume that the local proximity sensors and bearing sensors are not faulty.

We say that a robot *senses* another robot in the case where the distance between these two robots is shorter than R_m and the line-of-sight (LOS) is established between the two robots. We also claim that a robot *encounters* another robot in the case where the relative distance between them is shorter than $\epsilon \approx 0 \ll R_m$.The two robots are *neighbors* in the case where they can sense each other while satisfying that the relative distance between these two robots is shorter than $R_m - \epsilon$.

In the case where a robot is equipped with range sensors, such as radar or lidar, they can be used by the robot sensing its neighbors. In the case where two robots sense each other, the line segment connecting the two robots must not intersect an obstacle. This way, as one robot moves towards its neighboring robot, it does not collide with any obstacle.

In our experiments, a robot is not equipped with range sensors. In our experiments, a robot uses a communication module to sense its neighboring robot. In the case where LOS between one robot A and another robot B is blocked, then a signal strength from A to B degrades significantly. Also, as the relative distance between A and B increases, signal strength from A to B decreases. Considering these aspects, we assume that A senses B in the case where signal strength from A to B is bigger than a certain threshold.

We use the network topology to build the adjacency graph of the multi-agent system. Let $G = (V, E)$ represent the connectivity of the multi-agent system. In the graph G, every node in V represents a robot. Every edge, say $\{v_1, v_2\} \in E$, indicates that two robots, corresponding to v_1 and v_2, are neighbors. This further implies that the two robots can sense each other. The weight of an edge $\{v_1, v_2\}$ is the relative distance between two robots associated to v_1 and v_2, respectively. If range sensors cannot be utilized, equal weights are used.

Let $G_0 = (V_0, E_0)$ denote the *initial connectivity/interaction graph (at time step $t = 0$)*. We assume that G_0 is connected. We further assume that one robot, say D, is designated as a rendezvous robot, which can be re-assigned at any time.

3.3. Goal

The goal of an *any-node* rendezvous algorithm is to gather every robot in V at the root (rendezvous) robot D, i.e., when $\lim_{t\to\infty} \|\mathbf{q}_D - \mathbf{q}_i\| = 0$, $\forall i \in V(G) - D$.

4. Proposed Rendezvous Control Approach

To achieve the above goal, we propose a rendezvous controller using the following procedures: first, let every robot, except for D (the root/leader robot), rendezvous at D while maintaining connectivity; once all robots rendezvous at D, we let D head towards the designated rendezvous location, since D can lead the other robots to the final position.

4.1. Rendezvous algorithm

In Algorithm 1, we present the core rendezvous control approach. The algorithm works as follows. The network graph G is available to all robots in the network (see Algorithm 2 for our approach on building the network graph in a distributed fashion). Based on the graph and the assigned rendezvous robot D, we build the shortest path tree $T = (V, E_T)$ from G rooted at node D, using Dijkstra's algorithm [45]. Here, the number of edges $|E_T| = |V| - 1$. Since we assume an undirected and connected graph G with non-negative weights, the shortest-path tree T exists and is guaranteed by the Algorithm 1. In graphs where all edges have equal weights, T becomes the shortest-hop tree generated by breadth-first search (BFS) algorithm.

Algorithm 1 Rendezvous Control Algorithm

D is designated as the rendezvous location
Get the updated network graph G using Algorithm 2
Given G, we build a shortest-path tree T rooted at D
repeat
  $u \leftarrow$ every robot
  if u is associated to a leaf of T **then**
    u begins heading towards $p(u)$
    After encountering $p(u)$, u becomes the child of $p(p(u))$
  else if u is a parent robot with at least one child **then**
    if u encounters all its children **then**
      u becomes a leaf node and heads towards $p(u)$
    end if
  end if
  if there is a change in neighbor set of a robot (e.g., addition or removal of an edge because of a communication interruption) or change in the root node assignment **then**
    Update the network graph G utilizing Algorithm 2
    Update the shortest-path tree T using the updated G and/or D
  end if
until all children of D encounter D

Every node, say u, that does not have any children is called *leaf nodes* in T. We represent the immediate parent of a robot u as $p(u)$. At $t = 0$, all u nodes begin heading towards their parents. These movements initiate the entire rendezvous maneuvers. Since every robot associated with a leaf of T begins moving initially, $p(u)$ encounters all its children as the time goes on.

A robot can decide if it has met its parent or its child using its local sensor information. For example, in our experiments, we use the RSSI from the Wi-Fi receiver as the parameter to decide if it is close to (encounter) its parent or its child by applying a RSSI threshold.

After encountering all its children, $p(u)$ heads toward its parent, say $p(p(u))$. On the other hand, once u encounters its immediate parent $p(u)$, the robot u becomes a child of its next parent in the hierarchy of the tree, say $p(p(u))$, and moves toward $p(p(u))$. This process is repeated until all robots reach the top-most parent, the root robot D, where they should rendezvous. Thus, the above iterative approach is carried out until all robots rendezvous

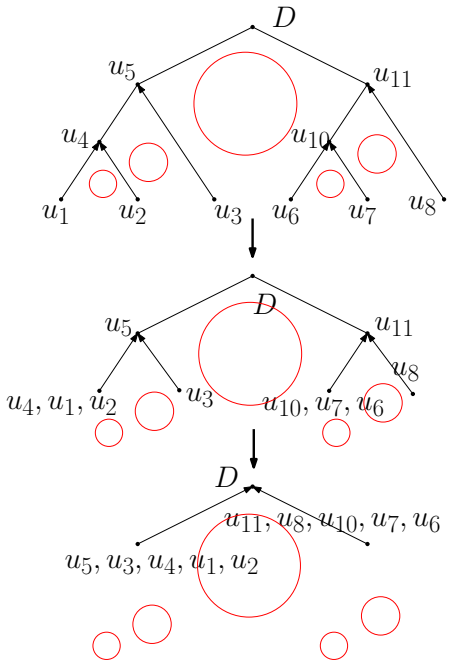


Figure 3: Illustration of the procedure following Algorithm 1. The obstacles in the environment are illustrated as red circles.

at D, which is the stop condition of the algorithm.

Figure 3 illustrates the procedure of the proposed algorithm. At the top of Figure 3, the initial graph T of a randomly-generated graph is illustrated, in which $u_1, u_2, u_3, u_6, u_7, u_8$ are leaves of T. For instance, u_3 is only connected to u_2, u_6, and u_5. Initially, all nodes head toward their corresponding parents following the hierarchy in T. The movement of each robot is depicted as an arrow in this figure. In further iterations, u_4 and u_{10} encounters all their children, and then all of them moves to their parents u_5 and u_{11}, shown in the middle of Figure 3. Finally, after u_5 (or u_{11}) encounters all their children, they begin heading toward D (bottom of the figure). Note that the obstacle boundaries are depicted as red circles.

Thus, the rendezvous is achieved using the iterative tracking movements of the child robots to their parent robots using the hierarchy in the tree T. Using the definition of T, the movement of a child robot towards its parent in T is along the shortest path to D contained in G_0. The velocity control inputs for individual robots are driven by its local bearing sensors. See Sec. 4.2 for more details on the robot controller.

Algorithm 2 Distributed Algorithms for Building a Distance List L

Initialize $L_{i,-1}$ as an empty set
Initialize $L_{i,0}$ as the *local neighbor lists* which are built using robots in N_i
$k = 0$
For every robot $i \in V$
repeat
  Detect the change in the local list L_i, $\bar{L}_{i,k} = L_{i,k} - L_{i,k-1}$
  Send $\bar{L}_{i,k}$ to all $u_j \in N_i$
  Receive $\bar{L}_{j,k}$ from all $u_j \in N_i$
  $L_{i,k+1} = L_{i,k} + \bar{L}_{j,k}$ for all $u_j \in N_i$
  $k = k + 1$
until $k \geq \mathbb{D}(G)$

To initialize Algorithm 1 following a distributed pattern, each robot senses its neighbors utilizing local sensors and generates a set named neighbor list L_i. Every robot thereafter shares its neighbor list utilizing a distributed algorithm (see Algorithm 2) and generates G utilizing the accumulated list L containing all vertices and edges of the networked system. Algorithm 2 was designed taking inspiration from the distributed consensus algorithm in [19]. As an example, let's consider a cycle graph consisting three nodes a, b, and c. Obviously, L of the cycle graph is the aggregation of the following three unordered lists: $L_a = \{b, c, w_{bc}\}$, $L_b = \{a, c, w_{ac}\}$, and $L_c = \{a, b, w_{ab}\}$. Here, w_{ij} for every edge is calculated utilizing available range sensors of robots. Otherwise, equal weights are used.

In Algorithm 2, N_i is the *neighbor set* of a robot i. Also, $\mathbb{D}(\mathbb{G})$ represents the diameter of G (the number of nodes from D to a node which is the farthest from D along the shortest path). In this algorithm, a robot shares the update in the local neighbor list with its neighbors. Hence, the global network graph G can be derived in a distributed manner. The convergence to a global list L (hence the global network graph G) at every robot is achieved in a maximum of $\mathbb{D}(G)$ iterations. Therefore, $L_{i,\mathbb{D}(G)} = L \ \forall i \in V$. This statement can be proved similarly to Proposition 1 of [19]. Therefore, the proof is omitted in this paper.

The most severe computation burden of Algorithm 2 relies on $\mathbb{D}(G)$ (the graph diameter of G) and the highest number of edges (neighbors) at every node. From [46], we note that $\mathbb{D}(\mathbb{G})$ has an upper bound of $\frac{(N-1)}{K}$. Here, K implies the K-connectedness of the graph. In practical applications, the

number of robots is significantly smaller than the number of edges in the network, and as a consequence, Algorithm 2 is scalable. Moreover, a dense network (a completely connected graph, for example) leads to a smaller diameter and a larger number of edges when compared to a sparse graph such as a path graph. Hence, the workload of both communication and computation can be well balanced concerning graph density. It is worth noting that Algorithm 2 is used only in the case where the network topology changes (for example, a robot detects a new neighbor), after initially building the list L.

4.2. Bearing-aided robot velocity controller

Following the bearing-aided consensus controllers in [38, 11], we devise a velocity controller $\dot{\mathbf{q}}_i = [\dot{x}_i, \dot{y}_i]$ for robot i to head toward its parent robot $p(i)$ as below:

$$\dot{\mathbf{q}}_i \quad = v_{ip} \begin{bmatrix} \cos\alpha_{ip} \\ \sin\alpha_{ip} \end{bmatrix} \tag{1}$$

where $\alpha_{ip} = \tan^{-1}\frac{(y_p - y_i)}{(x_p - x_i)}$ represents the relative bearing of the parent robot $p(i)$ (parent of robot i) with respect to the robot i, in the coordinate frame of robot i. We use the rotation matrix of the robot $R(\theta_i) = \begin{vmatrix} \cos\theta_i & \sin\theta_i \\ -\sin\theta_i & \cos\theta_i \end{vmatrix}$ to compensate the relative bearing from its own orientation θ_i. Thus, we have $\begin{bmatrix} \cos\alpha_{ip} \\ \sin\alpha_{ip} \end{bmatrix} = R(\theta_i)\frac{(\mathbf{q}_p - \mathbf{q}_i)}{\|\mathbf{q}_p - \mathbf{q}_i\|}$. We assume the robots are equipped with a bearing sensor that provides an estimate of the relative bearing α_{ip}.

The velocity control factor $v_{ip} \leq S_m \in \mathbb{R}^+$ is set to a value proportional to the distance between the robot and its parent robot, i.e., $v_{ip} = f(\|\mathbf{q}_{p(i)} - \mathbf{q}_i\|)$, if a range sensor is available. Else, the v_{ip} is set to a positive constant. For instance, in our experiments, although we did not use a range sensor, we used the RSSI from the Wi-Fi measurements, which is a function of the distance between the nodes, as the function to determine the magnitude of the linear velocity. Note, the velocity is bounded by S_m.

According to (1), the robot i moves and converges to its parent $p(i)$, as long as $p(i)$ is stationary, which is guaranteed by Algorithm 1.

In Sec. 5.2.1, we elaborate on how we use a rotating directional antenna as a bearing sensor that provides an estimate of the relative bearings of the neighboring robots based on the direction of arrival (DOA) of wireless signals. We also describe how we integrate the velocity controller with a bearing-based obstacle avoidance strategy.

4.3. Theoretical analysis of the proposed algorithm

We analyze the theoretical properties of the proposed method such as convergence, connectivity maintenance,, and fault detection capability.

4.3.1. Convergence

We propose a theorem (Theorem 1) showing that the robots will eventually encounter D over time to complete the rendezvous process.

Theorem 1. *As $t \to \infty$, every robot's position converges to D using the control law in (1), provided that G_0 is a connected graph.*

Proof. To prove convergence, we need to show that the distance between all the robots with respect to the root robot will converge. We consider the following Lyapunov candidate function,

$$\mathbb{V} = \sum_{i \in V-D} \|\mathbf{q}_{p(i)} - \mathbf{q}_i\| \tag{2}$$

where $p(i)$ is the parent of node i in T. It is clear that $\mathbb{V} = 0$ if and only if all robots encounter D. The time derivative of $\mathbb{V}$ is,

$$\dot{\mathbb{V}} = \sum_{i \in V-D} \frac{(\mathbf{q}_{p(i)} - \mathbf{q}_i)^T}{\|\mathbf{q}_{p(i)} - \mathbf{q}_i\|} (\dot{\mathbf{q}}_{p(i)} - \dot{\mathbf{q}}_i). \tag{3}$$

In Algorithm 1, the parent node does not maneuver until encountering all its children. i.e., $\dot{\mathbf{q}}_{p(i)} = 0$. Hence, (3) leads to

$$\dot{\mathbb{V}} = -\sum_{i \in V-D} \frac{(\mathbf{q}_{p(i)} - \mathbf{q}_i)^T}{\|\mathbf{q}_{p(i)} - \mathbf{q}_i\|} \dot{\mathbf{q}}_i, \tag{4}$$

$$\dot{\mathbb{V}} = -\sum_{i \in V-D} v_{ip} \left(\frac{(\mathbf{q}_{p(i)} - \mathbf{q}_i)}{\|\mathbf{q}_{p(i)} - \mathbf{q}_i\|} \right)^T \left(\frac{(\mathbf{q}_{p(i)} - \mathbf{q}_i)}{\|\mathbf{q}_{p(i)} - \mathbf{q}_i\|} \right) \leq 0. \tag{5}$$

Using LaSalle invariance principle [47], the system converges to the set in which $\dot{\mathbb{V}} = 0$. $\dot{\mathbb{V}} = 0$ if and only if all robots encounter D.

In our assumption (Sec. 3.2), two robots encounter each other in the case where $\|\mathbf{q}_i - \mathbf{q}_j\| \leq \epsilon$. Once a robot encounters its parent, it re-assigns its associate parent to the parent with a higher-order in the tree. This can generate a jump in $\mathbb{V}$ (a discontinuous fall in $\mathbb{V}$). Nevertheless, the amount of

such jumps is finite and upper bounded by the number of robots. Therefore, the number of such discontinuities does not influence the system convergence.

This concludes the proof that all robots $\mathbf{q}_i, \forall i \in V$, converge (rendezvous) at $\mathbf{q}_D$. □

4.3.2. Connectivity maintenance

In the next proposition, we show that the proposed algorithm preserves connectivity while performing the rendezvous.

Proposition 1. *Using Algorithm 1, a robot is connected to at least one other robot in the graph G in time of rendezvous.*

Proof. To prove this proposition, we first show that the robots do not lose a connection with their parents and that the robots are within the maximum sensing range R_m of at least one other robot in the tree T. Since the movement of a robot is heading toward its immediate parent, we first show that every robot i is always detected by its parent $p(i)$ before they encounter. As per the Algorithm 1, a parent robot $p(i)$ does not move until all its children satisfy the *merge* condition. Therefore, $\|\dot{\mathbf{q}}_p\| = 0$ if for any $i \in \mathbb{C}_p$, $\|\mathbf{q}_p - \mathbf{q}_i\| > \epsilon$. Here, $\mathbb{C}_p$ is the children set of a robot p. Note that the function $\|\mathbf{q}_p - \mathbf{q}_i\|$ is always non-increasing as per the algorithm. This way, the movement of the robot toward its parent is restricted along the line segment where the robot always stays connected. Thus, all children robots are sensed by their parent (and vice versa) until they encounter their parents. After a robot i encounters its parent $p(i)$, both the robot i and its next successive parent $p(p(i))$ can sense each other because $\|\mathbf{q}_{p(p(i))} - \mathbf{q}_{p(i)}\| \leq R_m - \epsilon$[3]; *and* $\|\mathbf{q}_{p(i)} - \mathbf{q}_i\| \leq \epsilon \implies \|\mathbf{q}_{p(p(i))} - \mathbf{q}_i\| \leq R_m$. Hence, every robot will be sensed by at least one other robot in the tree, and the graph stays connected. This concludes the proof.

□

4.3.3. Fault-tolerance in the controller

We claim that Algorithm 1 is designed to handle cases of intermittent communications (disconnection or reconnection of a robot). In such cases,

[3]Remember that two robots are neighbors if and only if they sense each other while satisfying the relative distance between them being less than or equal to $R_m - \epsilon$.

the shortest-path tree T is regenerated by triggering the Algorithm 2, and the Algorithm 1 is updated accordingly.

We also say that a robot is *faulty* if the robot's movement does not follow the Algorithm 1. Here, we refer to a faulty robot regarding mobility issues. We introduce a proposition (Proposition 2) in which we claim that a faulty robot can be detected by its child.

Proposition 2. *Assume that all robots have the same updated graph G using Algorithm 2, and that they move according to Algorithm 1. A faulty robot can be detected by its children during the rendezvous process.*

Proof. Using Algorithm 1, a parent robot $p(i)$ does not move until encountering all its children. In the case where a robot i, a child of $p(i)$, detects that the parent robot $p(i)$ is not stationary (by tracking the relative bearing or the relative range of the parent robot using local sensors), then robot i can presume that $p(i)$ is faulty. However, this converse is not true. That is, a parent robot cannot detect a fault in the child robot if an obstacle avoidance algorithm (which influences the control velocities) is used. Nevertheless, this simple rationale can be extended to detect other types of faults in the system by checking the robots whether they conform to Algorithm 1. □

Once a faulty robot is detected, then the shortest-path tree T is regenerated by triggering Algorithm 2 without the faulty robot, and Algorithm 1 is updated accordingly.

Consider the case where there are too many faulty robots. In this case, we may not be able to generate a connected tree T without the faulty robots. Thus, this approach works as long as a connected T can be built without the faulty robots.

5. Field Experiments

In this section, we first present the experimental setup. Next, we discuss the implementation of the above rendezvous control algorithm integrated with an obstacle avoidance strategy. Then, we present the experiment design and the scenarios tested.

5.1. Experiment setup and distance estimation

Figure 4 presents the configuration of the each robot in detail. All the three robots have the same configuration. The base platform is a commercial Pioneer P3AT mobile robot platform. For communication, we install a

state-of-the-art, low-cost, and small wireless AP, *PicoStation* M2-HP, manufactured by *Ubiquiti Networks Inc.* This AP is equipped with a 5-dBi omnidirectional antenna and supports passive Power over Ethernet (PoE), so it does not require an additional power cord. Also, it runs with IEEE 802.11g protocol having an operating frequency of 2.4 GHz and produces up to 28 dBm output power. In addition to this *PicoStation* AP, we installed a small and light Yagi directional antenna (manufactured by *PCTEL*), connected to a Wi-Fi USB adapter. This antenna is used for measuring the RSSI from different directions (through a rotation tracking system), which is then used for DOA estimation as detailed in Section 5.2.1. The beamwidth of this antenna is 60° at 1/2 power for horizontal and vertical planes. An Asus Eee laptop running Linux mounted on the P3AT robot is used to achieve high-level motion planning. This device have shown strong adaptability for outdoor point-to-point connections [41, 48].

To enable robot movements, we adopted the leader-follower robotic system introduced in [18], which is composed of a bearing estimation using a rotational directional antenna and an obstacle avoidance algorithm using an ultrasound sensor array. In the field experiments, the network topology updates at the frequency of 1 Hz to synchronize collected data and update the shortest path tree T in Algorithm 1 and the distance list L in Algorithm 2. The selection of this update frequency is determined with careful consideration of the moving velocity of the robots and the time required to detect and react to a failed node.

5.2. Robot control and obstacle avoidance

In this section, we briefly summarize our method, which we adopted from [18], to estimate the direction of arrival (relative bearing) from the RSSI of the rotating directional antenna, and use it to control the robot.

5.2.1. Estimation of relative bearings

The directional antenna is mounted on the robot's pan servo system and uses a USB Wi-Fi adapter to measure the RSSI of the parent robot's AP continuously. Then, with this measured RSSI, a Weighted Centered Algorithm (WCA) is used to calculate the DOA with the AP of the parent robot. The pan servo system has a scanning range of 180°. With a resolution of 10°, the robot measures the RSSI and records the corresponding angle.

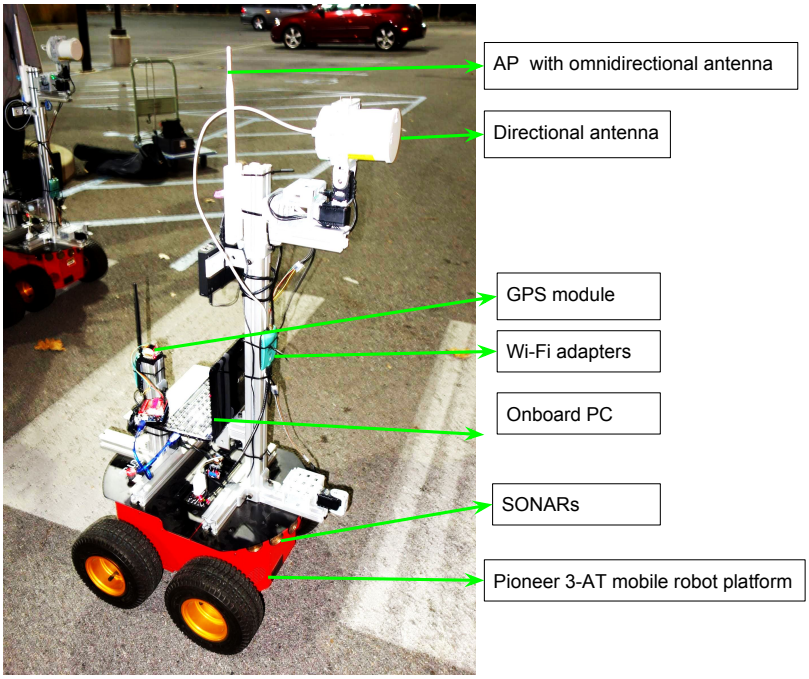


Figure 4: The mobile robot platform used in the experiments.

First, a weight is calculated at robot i based on the RSSI measurements from robot j at any instant k.

$$w_k^j = 10^{\left(\frac{RSSI_k^j}{\gamma_1}\right)} \tag{6}$$

where γ_1 is a positive constant (empirically determined) that can be adjusted to the nature of the environment. The relative bearing of robot j from robot i is estimated using:

$$\widetilde{\alpha}_{ij} = \frac{\sum_{k=1}^{M} w_k^j \theta_k^i}{\sum_{k=1}^{M} w_k^j}, \tag{7}$$

where θ_k^i is the orientation of the antenna at which $RSSI_k$ is measured, and M is the number of measurements per rotation scan. This DOA is then used to aid the children robots track and move toward their parent robots.

This method of estimating the relative bearings has shown to be reasonably accurate and robust to various environments based on our field experiments in the past [18, 48]. For more details on the bearing estimation and control, readers are referred to [48].

5.2.2. Obstacle avoidance

In the case where LOS between a robot and its neighbor is blocked by a (big) obstacle, the path connecting the two robots is not traversed using our algorithms. However, there may be a case where there is a small obstacle between two robots, and the obstacle does not block the LOS between the two robots. In this case, the two robots become neighbor to each other, and the path that connects the two robots may be traversed using our algorithms. As a robot moves along the path, it may collide with the small obstacle. Thus, we require an additional controller to avoid collision using local sensors.

We assume that unknown obstacles in the working space are convex in horizontal contour. To avoid such obstacles (static or dynamic) that are present between a robot moving towards its parent robot, we use the strategy described below.

The robots are equipped with an array of eight ultrasound sensors on the front side of the robot, as mentioned in Sec. 5.1, covering an angular range of 180°. In one scan that happens at 10 Hz, the sonar sensors provide the obstacle distance information over the entire angular range with a resolution of up to 1°. In every 10°, we obtain a distance d_k and compute a weight w_k defined as

$$w_k = 10^{\left(\frac{-d_k}{\gamma_2}\right)} \tag{8}$$

where γ_2 is a positive gain depending on the quality of the sonar signal. The direction guiding the robot to a safe region can be estimated by means of weighted centroid approaches as follows:

$$\widetilde{\alpha}_{obs} = \frac{\sum_{k=1}^{N_s} w_k \theta_k}{\sum_{k=1}^{N_s} w_k} \tag{9}$$

where N_s is the number of distance measurements per scan, and θ_k is the angle at instant k where the distance value d_k is measured.

5.2.3. Velocity control

To control the robots on a Cartesian plane, we apply the control law in (1) with a switching controller that selects the α_{ip} as follows:

$$\alpha_{ip} = \begin{cases} \widetilde{\alpha}_{ij}, & \text{if the obstacles are in a safe region,} \\ \widetilde{\alpha}_{obs}, & \text{otherwise.} \end{cases} \tag{10}$$

Thus, we integrate both bearings of the parent robot and the obstacles and navigate the robot to a safe region when it should avoid the obstacles first.

The dynamics are then transferred to the internal kinematics of the robots, where a PID motion controller is applied to the left and right motors as it is a differential-drive vehicle.

Also, the velocity of the child robot is set to be proportional to the RSSI (which is a log-normal function of the distance) from the parent robot. The linear velocity control factor $v_{ip} = \omega_1 - RSSI - \omega_2 RSSI$. Here, the $RSSI$ represents the RSSI value at the relative bearing angle of the parent, ω_1 and ω_2 are positive values such that v_{ip} is non-negative. This function is chosen such that when the robot is far from the parent robot, it moves at a higher speed and vice versa.

5.3. Experiment design

To test our theory and demonstrate its feasibility, we designed a multi-agent rendezvous experiment involving three mobile robots and a static workstation as shown in Figure 1. A large and obstacle-filled parking lot (approx 5000 m^2) is used as the experiment site. For convenience, we name three robots as R_1, R_2, and R_3. We name the static workstation as R_0, which is the designated rendezvous point in the experiments. Therefore, all the robots are supposed to gather at R_0 after triggering a rendezvous signal. We assume that there is no communication between R_0 and R_2, and R_1 and R_3 due to their communication range limits and the non-direct sight of view.

In our experiments, we set equal edge weights to all edges in the tree, which is not an ideal implementation. However, due to the low density of the network graph (number of edges = 3) in the field experiments, the weights do not play a key role in algorithm performance[4]. When we assume equal weights for all edges, T becomes the shortest-hop tree generated by breadth-first search (BFS) algorithm. Thus, each robot moves along the determined shortest-hop path to reach the root node. In this case, the traversal distance of a robot may be long since the shortest-hop path does not assure the shortest distance path. Therefore it is desirable to use the range information (as the weighs in G) obtained from a reliable range sensor, if available.

[4]Note, in Section 7, we simulate the case where every robot is equipped with both range and bearing sensors. The simulation experiments evaluate the algorithm for scalability and make use of the range sensors to determine the weights of the edges.

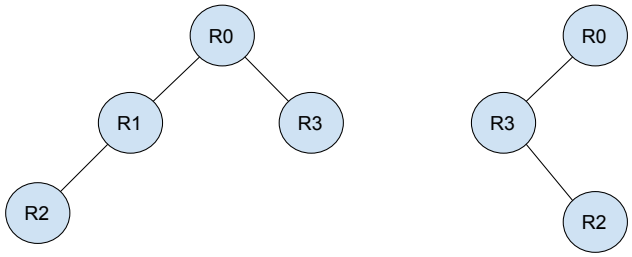


Figure 5: Initial hierarchical connected tree for all the three scenarios (left figure). Here, the diameter (depth) of the tree is $\mathbb{D}(T) = 2$. The updated hierarchical connected tree in Scenario 3 is shown on right figure.

The Algorithm 2 resulted in the same graph G at all the robots, which is then used in the main algorithm (Algorithm 1). Through our experiments, we verified that the proposed rendezvous algorithm works without any range sensors. The maximum sensing (communication) range R_m is set to 40 m (by applying a minimum RSSI threshold $RSSI_{min} = -80\,\text{dBm}$ at which the robot can reliably communicate with another robot), and the maximum velocity is set to be 0.2 m/s. The maximum RSSI threshold that is used to decide if the robots encounter each other is set to -22 dBm, which approximately corresponds to a threshold distance of $\epsilon = 2$ m.

5.4. Experiment scenarios

We experimented with three different scenarios. In all the scenarios, the initial communication hierarchy is the same as in Figure 5 (left).

5.4.1. Scenario 1

In this scenario, a typical rendezvous behavior is considered. The node R_0 is the rendezvous point, with children R_1 and R_3. The node R_2 has R_1 as its parent. Thus, on initiating the rendezvous task, the node R_3 starts moving toward R_0, while R_2 moves toward R_1. After R_2 reaches R_1, both R_1 and R_2 move toward R_0, finishing the rendezvous objective.

5.4.2. Scenario 2

This scenario is similar to the previous one, while a node movement fault during the rendezvous task is simulated. This scenario is designed to demonstrate the strategy discussed in Sec. 4.3.3. That is, R_1 node fails to move properly, but still able to communicate with other nodes. Therefore, as soon

as the node R_2 reaches R_1, R_2 switches to R_0 as its parent and moves toward R_0. On the other hand, R_3 starts moving toward R_0 from the beginning as R_0 is its direct parent. Thus, at the end of rendezvous, only R_3 and R_2 arrive at R_0, while R_1 is broken and does some random movements around its original position due to its mobility fault.

5.4.3. Scenario 3

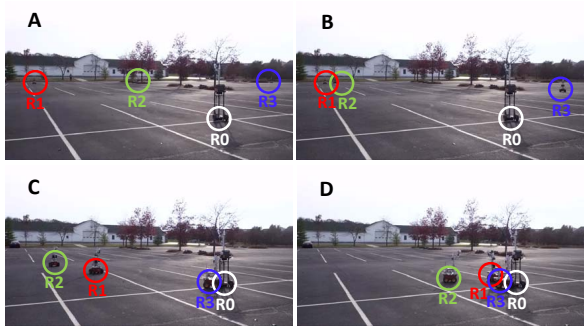
This is similar to Scenario 2, while there is a complete node fault with R_1 (both mobility and communication faults). We simulate the node fault by suddenly powering off R_1 while R_2 is moving toward R_1. Recall that at the beginning, R_3 moves toward R_0 and R_2 moves toward R_1, and R_1 waits until R_2 reaches it. Thus, when R_1 is switched off, R_2 is unable to communicate and track R_1 in its vicinity, resulting that the network hierarchy gets updated with R_1 removed as can be seen in Figure 5 (right). As soon as this update is made, R_2 will switch its parent to R_3, and R_3 will stop moving toward R_0, because it has to wait for its new child R_2. After R_2 reaches R_3, both move towards R_0 and complete the rendezvous task.

6. Results and discussion

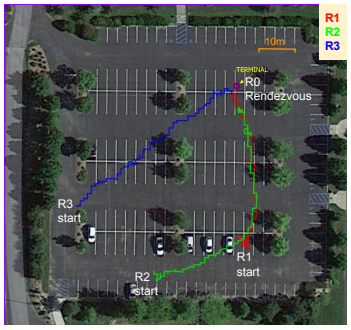
We performed two trials in each scenario, and have reported one trail in each case as they both were very similar in results. The paths taken by the robots (captured by a GPS sensor, because the robot odometer was inaccurate) overlaid on the satellite image are depicted in Figure 6 for all the scenarios. A temporal sequence of the experiments in all the scenarios is also shown in Figure 6. Video recordings of the experiments are available at `https://youtu.be/fJEsAPsx_Mw`. It can be seen that the resulting behavior in each scenario was as expected. Note that the robots took the least distant path possible while avoiding the trees and obstacles on their way. Several cars are seen on the satellite images; however, actually, they did not exist during the experiment. Also, note that the GPS reading has a mean accuracy of 2.5 m [49] typically, and hence the trajectory overlaps with positions of trees in some places, although the robots successfully avoided them.

6.1. Scenario 1

Figure 6(a) shows several images of the experiment in Scenario 1 as a sequence. It shows that R_3 moved toward R_0 and R_2 moved toward R_1 in the images A and B. The images B, C, and D show that after R_2 met R_1

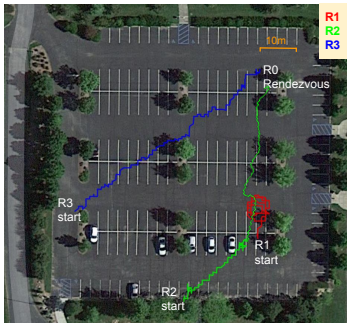


(a) Scenario 1 - temporal sequence

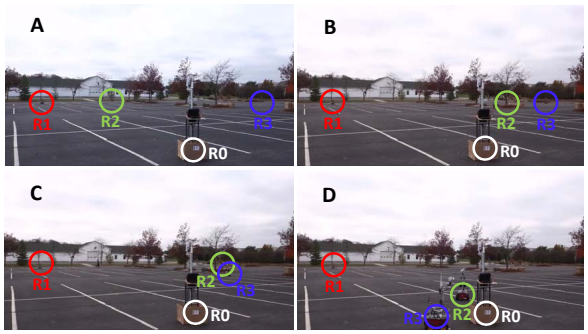


(b) Scenario 1 - trajectories

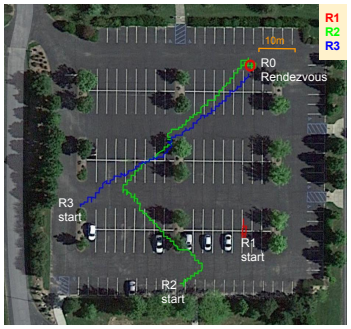
(c) Scenario 2 - temporal sequence



(d) Scenario 2 - trajectories



(e) Scenario 3 - temporal sequence



(f) Scenario 3 - trajectories

Figure 6: (a), (c), and (e) show a sequence of stills from experiments in Scenarios 1, 2, and 3. The temporal sequence is sorted alphabetically. (b), (d), and (e) are trajectories of the robots in Scenarios 1, 2, and 3 overlaid on the satellite image of the actual experimental site (a large and obstacle-filled parking lot). The trajectory of R_1 is depicted in red, the trajectory of R_2 is depicted in green, and the trajectory of R_3 is depicted in blue.

within the threshold distance ϵ, both R_2 and R_1 moved and arrived at R_0. Figure 6(b) shows the trajectory taken by the robots in this scenario, where you can also see all the three robots could successfully rendezvous at the designated point.

Figure 7(a) represents the changes in RSSI readings (filtered) measured by the rotational directional antenna of the robots (the best RSSI in a 180 degree scan). R_1 and R_3 have R_0 as its parent (AP), while R_2 has R_1 as its parent (AP). As shown in this figure, the RSSI at R_3 increases as it moves toward R_0. Similarly, the RSSI at R_2 increases as it moves toward R_1. However, after R_2 reaches R_1, R_2 maintains its parent as R_1 while R_1 moves to R_0. Thus, R_2 also follows R_1, while the RSSI at R_2 balances with the movement of R_1, but the RSSI at R_1 increases.

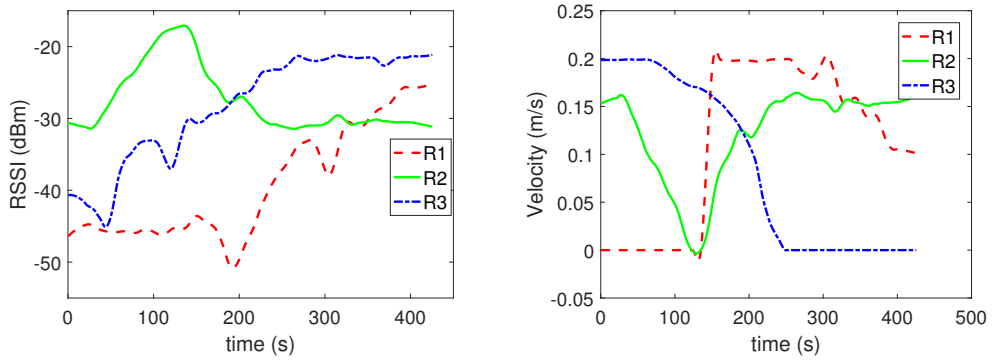
In Figure 7(b), the corresponding changes in robots' velocity (filtered) are presented. The robot's maximum velocity is set as 0.2 m/s as mentioned above. See around 150 seconds that shows the node R_1 waited until R_2 reached it, and then both moved toward R_0., whereas R_3 moved toward R_0 from the beginning and reached early as expected. Note the velocity was proportional to the relative distance between the parent and the child as described in Sec. 5.1. Therefore, when the child was farther from the parent initially, the velocity was higher, whereas it reduced the velocity when they were closer.

6.2. Scenario 2

Figure 6(d) shows the trajectory taken by the robots in Scenario 2. Figure 7(c) shows the velocity plots for this task. Note that R_1 stopped recording the velocity information after around 80 seconds due to technical reasons. As shown in the Figures 6(c) and 6(d), R_2 and R_3 were able to rendezvous at R_0 while R_1 exhibited chaotic movement and circles around its original position. Notably from Figure 7(c), one can see that the speed of R_2 was decreasing as it was approaching R_1 from $t = 0$ second to $t = 40$ second. However, as soon as the parent of R_2 node was switched from R_1 to R_0 after the detection of the failure of R_1, the velocity of R_2 was increasing. This result demonstrates that the proposed rendezvous control strategy can handle fault cases in nodes.

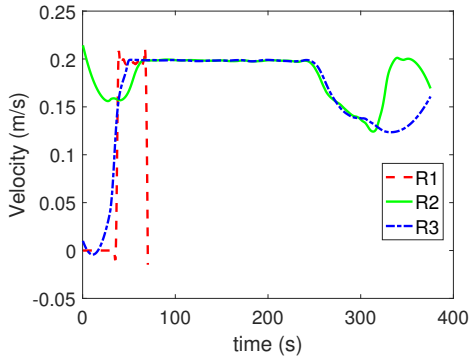
6.3. Scenario 3

Figure 6(e) shows a sequence of the experiment and Figure 6(f) shows robots' trajectories in Scenario 3 where the node R_1 was powered off at a

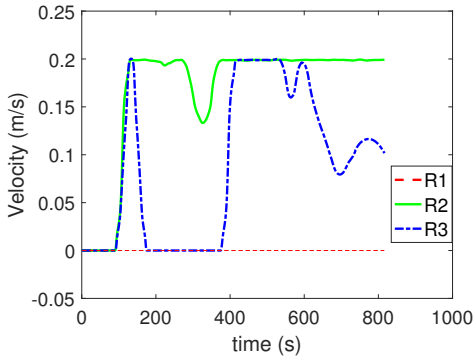


(a) RSSI Scenario 1

(b) Velocity Scenario 1



(c) Velocity Scenario 2



(d) Velocity Scenario 3

Figure 7: (a) shows the RSSI changes during rendezvous task in Scenario 1. (b)-(d) show the change in robot velocity during each of the scenarios.

random instant before R_2 reaches R_1. The robot trajectories were observed as expected in Sec. 5.4.3, which was also evident in Figure 7(d) showing the velocities in one of the trials. Observe that as soon as R_1 was out of the network hierarchy, node R_2 switched its parent to R_3 and moved toward R_3, and R_3 stopped moving until R_2 reached R_3. After that, both R_3 and R_2 moved to R_0.

The results of this Scenario 3 demonstrate the advantage of a dynamic hierarchical tree in handling the situation in which one of the nodes has failed including its ability to communicate.

6.4. Discussion

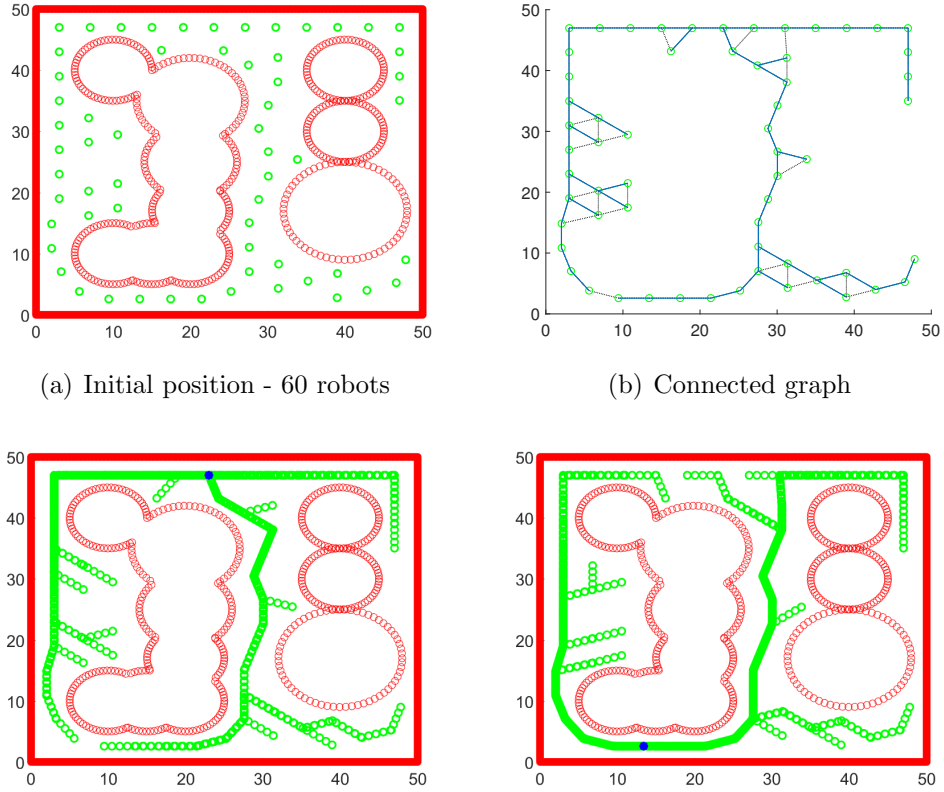
The field experiments have validated the following features of the proposed rendezvous control: boundedness, global connectivity, and broken nodes handling. Also, it shows that all robots exhibit strong capability in avoiding obstacles in the cluttered and dynamic environment. Nonetheless, an important feature to be demonstrated is the *scalability* of the control scheme as we are dealing with a multi-robot application. As we had only limited hardware resources for the field test, we conducted simulation experiments for Scenario 1 with the larger number of robot nodes, discussed in the next section. It is worth noting that robot coordinates are not needed in the implementation, which reflects the flexibility of the proposed rendezvous algorithm.

7. Simulation Experiments

We conducted simulation experiments in MATLAB to evaluate the proposed rendezvous control method at a higher scale (regarding the number of robots). We simulated a 2D environment (plane of 50 m $\times$ 50 m) setup with pre-defined obstacles.In the MATLAB simulation, we introduced a constraint on the sensing capability; thus one robot can detect another robot only if a LOS path exists between them (i.e., obstacle-free movements). We assumed that both range and bearing sensors are available, and thus we were able to obtain the relative position of the neighboring robots. Each robot moves along the shortest path to the root, since the constructed tree is the shortest-path tree (compared to a shortest-hop tree in the field experiments).

In the simulations, the sampling interval of a robot's velocity controller (1) is set as 0.1 seconds. This implies that the velocity of a robot changes at a rate of 10 Hz. Note, we consider a fixed network graph and update the network tree at the same rate of the velocity controller. The maximum velocity of each robot is $S_m = 0.5$ m/s. We set the sensing range $R_m = 5$ m and distance threshold $\epsilon = 0.1$ m. Other settings are similar to the ones in the field experiments.

We conducted simulation experiments with 60 robots with two trials with different rendezvous point. Figure 8 shows the results of the simulations validating the scalability of the proposed method. Figures 8(a) and 8(b) show the nodes position and the resulting connected tree (both G_0 and T) respectively. The outcomes of the robots trajectories in each trial are shown in the Figures 8(c) and 8(d). Note that each robot moves along T. The only



(a) Initial position - 60 robots

(b) Connected graph

(c) Trajectories - Trial 1

(d) Trajectories - Trial 2

Figure 8: Simulation with 60 robots is shown. (a) Initial positions of all robots. Green circles are robot nodes and red curves are obstructions for communication/sensing. (b) Connected graph and network hierarchy. All edges are depicted as black dotted-line segments. All shortest-path connections are depicted as blue line segments. (c) and (d) Trajectories of all robots after rendezvous at the designated position (marked as blue asterisk) in different trials.

difference is the rendezvous point in both cases. The denser (darker green) the circles are in the plots, the higher the number of robots visited those positions. A time lapsed demonstration of the simulation results is shown in the video mentioned in Sec. 6.

8. Conclusion

We proposed rendezvous control laws for the coordinate-less multi-agent rendezvous problem based on network topology. Our framework included building and dynamically updating a hierarchical network tree. The main characteristics of the proposed control are as follows: boundedness, scalability, and global connectedness preservation. Our control laws can handle realistic application scenarios in cluttered environments and are easy to implement in practice. Additionally, the proposed method can handle minor fault cases such as mobility and communication faults without disrupting the whole rendezvous task.

We conducted extensive field experiments and simulations to validate the proposed control laws under practical application-oriented rendezvous scenarios. In field experiments, we used three mobile robots and demonstrated that the robots could rendezvous at the desired point in different scenarios even if there were mobility and communication faults in one node. In simulations, we further demonstrated the scalability of the proposed control scheme.

In future work, we will investigate the rendezvous problem using multiple squads with the primary goal of enabling both dynamic parent and child nodes (e.g., without the need to wait until all their children arrive) and moving rendezvous points, in addition to improving efficiency regarding energy and communication. Also, a possible extension is to fuse multiple sensor information for movement control. For instance, using vision integrated with wireless measurements and range information to track the trajectory of a dynamic parent robot with a goal of reducing the communication workload and early detection of failure scenarios.

Acknowledgement

This work was partially supported through the "NSF Center for Robots and Sensors for the Human Well-Being, under NSF Grant No. 1439717".

References

[1] J. Kim, S. Maxon, M. Egerstedt, F. Zhang, Intruder capturing on a topological map assisted by information networks, in: proc. of IEEE Conference on Decision and Control, Orlando, USA, pp. 6266 – 6271.

[2] J. Kim, S. Kim, Motion control of multiple autonomous ships to approach a target without being detected, International journal of advanced robotics systems 15(2) (2018).

[3] J. Kim, F. Zhang, M. Egerstedt, Simultaneous cooperative exploration and networking based on Voronoi diagrams, in: proc. of IFAC Workshop on Networked Robotics, Colorado, USA, pp. 1–6.

[4] J. Kim, Cooperative exploration and protection of a workspace assisted by information networks, Annals of Mathematics and Artificial Intelligence 70 (2014) 203–220.

[5] N. Ahmed, J. Cortes, S. Martinez, Distributed control and estimation of robotic vehicle networks: Overview of the special issue, IEEE Control Systems 36 (2016) 36–40.

[6] J. Kim, Capturing intruders based on voronoi diagrams assisted by information networks, International Journal of Advanced Robotic Systems Article ID 962523 (2017).

[7] J. Kim, Cooperative exploration and networking while preserving collision avoidance, IEEE Transactions on Cybernetics PP (2016) 1 – 11.

[8] M. Ji, A. Muhammad, M. Egerstedt, Leader-based multi-agent coordination: Controllability and optimal control, in: American Control Conference, 2006, IEEE, pp. 1358 – 1363.

[9] J. Kim, F. Zhang, M. Egerstedt, Battery level estimation of mobile agents under communication constraints, in: Sensor Networks, Ubiquitous, and Trustworthy Computing (SUTC), 2010 IEEE International Conference on, IEEE, pp. 291 – 295.

[10] J. Yu, S. M. LaValle, D. Liberzon, Rendezvous without coordinates, in: Decision and Control, 2008. CDC 2008. 47th IEEE Conference on, IEEE, pp. 1803–1808.

[11] R. Zheng, D. Sun, Multirobot rendezvous with bearing-only or range-only measurements, Robotics and Biomimetics 1 (2014) 4.

[12] T. Setter, M. Egerstedt, Energy-constrained coordination of multi-robot teams, IEEE Transactions on Control Systems Technology (2016).

[13] L. Sabattini, C. Secchi, N. Chopra, A. Gasparri, Distributed control of multirobot systems with global connectivity maintenance, IEEE Transactions on Robotics 29 (2013) 1326 – 1332.

[14] A. Ajorlou, A. Momeni, A. G. Aghdam, Connectivity preservation in nonholonomic multi-agent systems: A bounded distributed control strategy, IEEE Transactions on Automatic Control 58 (2013) 2366 – 2371.

[15] L. E. Parker, B. Kannan, X. Fu, Y. Tang, Heterogeneous mobile sensor net deployment using robot herding and line-of-sight formations, in: Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, volume 3, IEEE, pp. 2488–2493.

[16] N. Mathew, S. L. Smith, S. L. Waslander, Multirobot rendezvous planning for recharging in persistent tasks, IEEE Transaction on Robotics 31 (2015) 128 – 142.

[17] M. Gupta, J. Das, M. A. M. Vieira, H. Heidarsson, H. Vathsangam, G. S. Sukhatme, Collective transport of robots: Coherent, minimalist multi-robot leader-following, in: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5834–5840.

[18] B.-C. Min, E. T. Matson, Robotic Follower System Using Bearing-Only Tracking with Directional Antennas, Springer International Publishing, Cham, pp. 37–58.

[19] Alonso-Mora, E. M. Javier, M. Schwager, D. Rus, Distributed multi-robot formation control among obstacles: A geometric and optimization approach with consensus, in: Robotics and Automation (ICRA), 2016 IEEE International Conference on, IEEE, pp. 1–8.

[20] N. Roy, G. Dudek, Collaborative robot exploration and rendezvous: Algorithms, performance bounds and observations, Autonomous Robots 11 (2001) 117–136.

[21] W. Ren, H. Chao, W. Bourgeous, N. Sorensen, Y. Chen, Experimental validation of consensus algorithms for multivehicle cooperative control, IEEE Transactions on Control Systems Technology 16 (2008) 745–752.

[22] B. Li, B. Moridian, N. Mahmoudian, Underwater multi-robot persistent area coverage mission planning, in: OCEANS 2016 MTS/IEEE Monterey, pp. 1–6.

[23] M. Mesbahi, M. Egerstedt, Graph theoretic methods in multiagent networks, Princeton University Press, 2010.

[24] C. Gong, S. Tully, G. Kantor, H. Choset, Multi-agent deterministic graph mapping via robot rendezvous, in: Robotics and Automation (ICRA), 2012 IEEE International Conference on, IEEE, pp. 1278–1283.

[25] M. Ji, M. Egerstedt, Distributed coordination control of multi-agent systems while preserving connectedness, IEEE Transactions on Robotics 23 (2007) 693–703.

[26] H. Park, S. Hutchinson, A distributed optimal strategy for rendezvous of multi-robots with random node failures, in: Proc. of IEEE International Conference on Intelligent Robots and Systems, USA, pp. 1155 – 1160.

[27] M. Franceschelli, M. Egerstedt, A. Giua, C. Mahulea, Constrained invariant motions for networked multi-agent systems, in: 2009 American Control Conference, IEEE, pp. 5749–5754.

[28] A. Jadbabaie, J. Lin, A. S. Morse, Coordination of groups of mobile autonomous agents using nearest neighbor rules, IEEE Transactions on automatic control 48 (2003) 988–1001.

[29] F. Xiao, L. Wang, T. Chen, Connectivity preservation for multi-agent rendezvous with link failure, Automatica 48 (2012) 25 – 35.

[30] X. Li, D. Sun, J. Yang, Preserving multirobot connectivity in rendezvous tasks in the presence of obstacles with bounded control input, IEEE Transactions on Control Systems Technology 21 (2013) 2306–2314.

[31] H. Ando, Y. Oasa, I. Suzuki, M. Yamashita, Distributed memoryless point convergence algorithm for mobile robots with limited visibility, IEEE Transactions on Robotics and Automation 15 (1999) 818–828.

[32] S. Martinez, Practical multiagent rendezvous through modified circumcenter algorithms, Automatica 45 (2009) 2010–2017.

[33] Z. Feng, C. Sun, G. Hu, Robust connectivity preserving rendezvous of multi-robot systems under unknown dynamics and disturbances, IEEE Transactions on Control of Network Systems PP (2016) 1–1.

[34] L. Sabattini, C. Secchi, C. Fantuzzi, Arbitrarily shaped formations of mobile robots: artificial potential fields and coordinate transformation, Autonomous Robots 30 (2011) 385 – 397.

[35] H. Park, S. Hutchinson, An efficient algorithm for fault-tolerant rendezvous of multi-robot systems with controllable sensing range, in: Robotics and Automation (ICRA), 2016 IEEE International Conference on, IEEE, pp. 358–365.

[36] J. G. Manathara, D. Ghose, Rendezvous of multiple uavs with collision avoidance using consensus, Journal of Aerospace Engineering 25 (2011) 480–489.

[37] M. Kriegleder, S. T. Digumarti, R. Oung, R. D'Andrea, Rendezvous with bearing-only information and limited sensing range, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 5941–5947.

[38] S. Zhao, R. Zheng, Flexible bearing-only rendezvous control of mobile robots, in: 2017 36th Chinese Control Conference (CCC), pp. 8051–8056.

[39] P. R. Giordano, A. Franchi, C. Seccos, H. H. Bulthoff, A passivity-based decentralized strategy for generalized connectivity maintenance, International Journal of Robotics Research 32 (2013) 299 – 323.

[40] M. Lindhe, P. Ogren, K. H. Johansson, Flocking with obstacle avoidance: A new distributed coordination algorithm based on voronoi partitions, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 1785–1790.

[41] B.-C. Min, R. Parasuraman, S. Lee, J.-W. Jung, E. T. Matson, A directional antenna based leader–follower relay system for end-to-end robot communications, Robotics and Autonomous Systems 101 (2018) 57–73.

[42] J. de Hoog, S. Cameron, A. Visser, Dynamic team hierarchies in communication-limited multi-robot exploration, in: 2010 IEEE Safety Security and Rescue Robotics, pp. 1–7.

[43] D. Mukherjee, D. Ghose, Generalized hierarchical cyclic pursuit, Automatica 71 (2016) 318–323.

[44] B. W. Douglas, Introduction to Graph Theory, Prentice Hall, Illinois, USA, 2 edition, 2001.

[45] S. M. Lavalle, Planning Algorithms, Cambridge University Press, 2006.

[46] Graphs of maximum diameter, Discrete Mathematics 102 (1992) 121 – 141.

[47] J. P. La Salle, The stability of dynamical systems, SIAM, 1976.

[48] B.-C. Min, E. T. Matson, J.-W. Jung, Active antenna tracking system with directional antennas for enhancing wireless communication capabilities of a networked robotic system, Journal of Field Robotics 33 (2016) 391–406.

[49] GLOBALSAT GPS Module: Hardware Data Sheet, Globalsat Technology Corporation, 2013. Version 1.4.