Multirobot rendezvous with visibility sensors in nonconvex environments

Anurag Ganguli, Jorge Cortés, and Francesco Bullo

***Abstract*—This paper presents a coordination algorithm for mobile autonomous robots. Relying upon distributed sensing, the robots achieve rendezvous, that is, they move to a common location. Each robot is a point mass moving in a simply connected, nonconvex, unknown environment according to an omnidirectional kinematic model. It is equipped with line-of-sight limited-range sensors, i.e., it can measure the relative position of any object (robots or environment boundary) if and only if the object is within a given distance and there are no obstacles in-between. The Perimeter Minimizing Algorithm is designed using the notions of robust visibility, connectivity-preserving constraint sets, and proximity graphs. The algorithm provably achieves rendezvous if the inter-agent sensing graph is connected at any time during the evolution of the group. Simulations illustrate the theoretical results and the performance of the proposed algorithm in asynchronous setups and with measurement errors, control errors and non-zero robot size. Simulations to illustrate the importance of visibility constraints and comparisons with the optimal centralized algorithm are also included.**

***Index Terms*—Multi-robot coordination, Cooperative control, Distributed algorithm, Visibility, Nonlinear systems and control**

I. Introduction

Multi-agent robotic systems have been receiving increasing attention in recent times due, in no small part, to the remarkable advances made in recent years in the development of small, agile, relatively inexpensive sensor nodes. Large number of such simple nodes offer a more economical, scalable, and robust solution than the use of fewer more expensive and sophisticated ones.

Inspired by the work in [1], we consider the multirobot rendezvous problem; in this basic coordination problem, we aim to design distributed control laws to steer all robots to a common location. The robots are assumed to move in a nonconvex environment, and have minimal sensing capabilities. Each robot is only equipped with an *omnidirectional limited-range visibility sensor*; the nomenclature is adopted from [2, Section 11.5]. Such a sensor is a device that determines within its line of sight and its sensing range the following quantities: (i) the relative position of other robots, and (ii) the relative position of the boundary of environment. Examples of some such visibility sensors can be found in [3], [4], [5]. The sensor data can be processed to obtain a geometric representation of the area visible from a robot, e.g., see [6]. The *robots do not have any global knowledge about the environment or the position of other robots, do not share a common reference frame, and do not communicate.* We also assume that the algorithm regulating the robots' motion is memoryless, i.e., we consider static feedback laws. Given the aforesaid model, the goal is to design a discrete-time algorithm which ensures that the robots converge to a common location within the environment. Please refer to Figure 1 for a graphical illustration of rendezvous in a nonconvex environment.

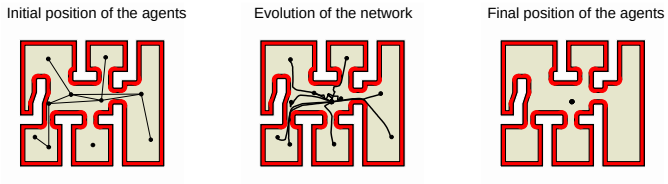

Fig. 1. Illustration of rendezvous for a group of robots distributed in a nonconvex environment shaped like a typical floor plan. The left-most figure shows the inter-robot sensing graph.

A sub-problem addressed in this paper is that of connectivity maintenance with visibility sensors; in other words, how should the robots move if they are to ensure that the graph generated by inter-robot visibility remains connected.

Our interest in the rendezvous problem is justified as follows. First, rendezvous is the most basic formation control problem and can be used as a building block for more sophisticated behaviors. A rendezvous algorithm can be used by the robots to agree on an origin of a global reference frame, or to come closer and form complex structures (modular robotics). Constraints on power or the geometry of the environment may sometimes require that the robots come closer to be able to communicate. Also, many data collection applications require deploying robots over a region of interest and subsequently retrieving them. A rendezvous algorithm can enable easy retrieval. The problem addressed in this paper is particularly relevant when the intended application is set in an urban or indoor environment.

The sensing and communication limitations are motivated in part by the problem of characterizing the minimal robot capabilities necessary to perform the required task and in part by the intended application setting. For example, in the urban canyon or in indoor environments, GPS signals are unreliable. Compass accuracies are affected by local magnetic anomalies. Communication might be expensive or unreliable. Also, a sensor-based approach avoids altogether the correspondence problem between what a robot senses and what it receives from communication with other robots.

Submitted as a Regular Paper on November 5, 2006. An early version of this work appeared in the IEEE Conference on Decision and Control, in Seville, Spain, 2005, with title "On rendezvous for visually-guided agents in a nonconvex polygon." This version revised on November 19, 2008.

A. Ganguli is with Center for Control, Dynamical Systems and Computation, University of California, Santa Barbara, CA 93106, anurag.ganguli@gmail.com. Corresponding author.

J. Cortés is with the Department of Mechanical and Aerospace Engineering, University of California, San Diego, CA 92093, USA, cortes@ucsd.edu

F. Bullo is with the Center for Control, Dynamical Systems and Computation, University of California, Santa Barbara, CA 93106, bullo@engineering.ucsb.edu

The connectivity maintenance sub-problem is another fundamental problem motivated by cooperative and distributed robotics applications; see for example the recent article in the IEEE Transactions on Robotics [7]. The novel connectivity maintenance algorithm in this paper can be used in various problems concerning nonconvex environments and visibility-based sensing beyond rendezvous, such as leader following and formation control.

The literature on multirobot systems is very extensive. Examples include the survey [8] and the special issue [9] of the IEEE Transaction on Robotics and Automation. Our multi-robot model is inspired by the literature on networks of mobile interacting robots: an early contribution is the model proposed in [10] consisting of a group of identical "distributed anonymous mobile robots" characterized as follows. Each robot completes repeatedly a cycle of operations: it senses the relative position of all other robots, elaborates this information, and moves. The robots share a common clock. A related model is presented in [11], where the robots evolve asynchronously, have limited visibility, and share a common reference frame. For these types of systems, the "multi-agent rendezvous" problem and the first "circumcenter algorithm" have been introduced in [1]. This algorithm has been extended to various asynchronous strategies in [12], [11], where rendezvous is referred to as the "gathering" problem. The circumcenter algorithm has been extended to arbitrary higher dimensions in [13], where its robustness properties are also characterized. Multirobot rendezvous with line-of-sight sensors is considered in [14], where solutions are proposed based on the exploration of the unknown environment and the selection of appropriate rendezvous points at pre-specified times. The problem of computing a rendezvous point in polyhedral surfaces made of triangular faces is considered in [15]. Formation control and rendezvous problems have been widely investigated with different assumptions on the inter-robot sensing. For example, a control law for groups with *time-dependent* sensing topology is proposed in [16]; this and similar works, however, depend upon a critical assumption of connectivity of the inter-agent sensing graph. This assumption is imposed without a sensing model. In this paper, we consider *position-dependent* graphs and, extending to visibility sensors a key idea in [1], we show how to constrain the robots' motion to maintain connectivity of the inter-robot sensing graph.

Our main contribution is a novel, provably correct algorithm that achieves rendezvous in a simply connected, nonconvex, unknown environment among robots with omnidirectional range-limited visibility sensors. Rendezvous is achieved among all robots if the inter-agent sensing graph is connected at any time during the group evolution. Another relevant contribution is the introduction of a novel set of notions and tools for connectivity maintenance in nonconvex environments; these notions are applicable to visibility-based multi-robot scenarios beyond the rendezvous problem. The technical approach proceeds as follows. In Section II, we review useful geometric notions, such as robust visibility [17] and proximity graphs [18], and introduce various novel visibility graphs. In Section III, we state the rendezvous and connectivity maintenance problems. To maintain connectivity during the system evolution, we design in Section IV novel constraint sets that (i) ensure that the visibility between two robots is preserved, and (ii) change continuously (in an appropriate technical sense) with the robots' positions. We define the novel locally-cliqueless visibility graph, which contains fewer edges than the visibility graph, and has the same connected components. This construction is useful in the connectivity maintenance problem because it imposes fewer constraints on the group evolution. In Section V, we provide a careful analysis of our solution to the rendezvous problem: the Perimeter Minimizing Algorithm. The main convergence result is proved via our recent version of the LaSalle Invariance Principle for set-valued maps [13]. As a novel Lyapunov function, we consider the perimeter of the relative convex hull of the robot positions. Finally, extensive simulations in Section VI validate our results and establish the convergence of our algorithm beyond the assumptions made in the theoretical analysis. The simulations show that our algorithm performance is still adequate assuming asynchronous agent operation, noise errors in sensing and control, and finite-size disk robots. Additional analysis results and all proofs are presented in the electronically available technical report [19].

II. Basic geometric notions

In this section we introduce some useful geometric notions. Let $\mathbb{Z}_{\geq 0}$, $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, and $\mathbb{R}_{>0}$ denote the sets of nonnegative integer, real, nonnegative real, and positive real numbers, respectively. For $p \in \mathbb{R}^2$ and $r \in \mathbb{R}_{>0}$, let $B(p,r)$ denote the *closed ball* centered at p of radius r. Given a bounded set $X \subset \mathbb{R}^2$, let $\mathrm{co}(X)$ denote the convex hull of X, and let $\mathrm{CC}(X)$ denote the *circumcenter* of X, i.e., the center of the smallest-radius circle enclosing X. For $p, q \in \mathbb{R}^2$, let $]p,q[= \{\lambda p + (1-\lambda)q \mid 0 < \lambda < 1\}$ and $[p,q] = \{\lambda p + (1-\lambda)q \mid 0 \leq \lambda \leq 1\}$ denote the *open* and *closed segment* with extreme points p and q, respectively. The *versor map* $\mathrm{vers} : \mathbb{R}^d \to \mathbb{R}^d$ is defined by $\mathrm{vers}(0) = 0$ and $\mathrm{vers}(p) = p/\|p\|$ for $p \neq 0$. Let $|X|$ denote the cardinality of a finite set X in $\mathbb{R}^2$. Given a compact set of points $X \subset \mathbb{R}^2$, and another point $p \in \mathbb{R}^2$, let $\mathrm{dist}(p, X)$ denote the minimum Euclidean distance of p to any point in the set X. The diameter $\mathrm{diam}(X)$ of a compact set X is the maximum distance between any two points in X.

Now, let us turn our attention to the environments we are interested in. Given any compact and connected subset Q of $\mathbb{R}^2$, let ∂Q denote its boundary. A point q of ∂Q is *strictly concave* if for all $\epsilon > 0$ there exists q_1 and q_2 in $B(q,\epsilon) \cap \partial Q$ such that the open interval $]q_1, q_2[$ is outside Q. A *strict concavity* of ∂Q is either an isolated strictly concave point or a connected set of strictly concave points. Accordingly, a strict concavity is either an isolated point (e.g., points r_1 and r_2 in Figure 2) or an arc (e.g., arc a_1 in Figure 2). Also, any strictly concave point belongs to exactly one strict concavity.

Definition II.1 (Allowable environment) *A set $Q \subset \mathbb{R}^2$ is* allowable *if*

(i) *Q is compact and simply connected;*
(ii) *∂Q is continuously differentiable except on a finite number of points; and*

(iii) *∂Q has a finite number of strict concavities.*

Recall that, roughly speaking, a set is simply connected if it is connected and it contains no holes. A particular case of the environment described above is a polygonal environment, the concavities being the reflex vertices[1] of the environment.

One can define the *internal tangent half-plane* $H_Q(v)$ at the following strictly concave points:

(i) At any point v where ∂Q is continuously differentiable, $H_Q(v)$ is the half-plane whose boundary is tangent to ∂Q at v and whose interior does not contain any points of the concavity; see point v' in Figure 2.

(ii) At any point v which is the end point of a strictly concave arc, $H_Q(v)$ is the half-plane whose boundary is tangent to the arc at v and whose interior does not contain any points of the concavity; see point v'' in Figure 2.

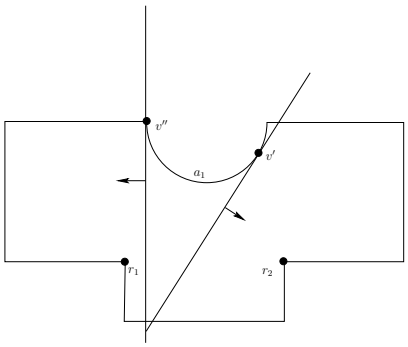

Fig. 2. An allowable environment Q: the closed arc a_1 and the isolated points r_1, r_2 are strict concavities. v' is a point on a_1 where the slope of ∂Q is defined. $H_Q(v')$ is the half-plane with the tangent to ∂Q at v' as the boundary and the interior in the direction of the arrow. v'' is an end point of arc a_1. $H_Q(v'')$ is the half-plane with the tangent to a_1 at v'' as the boundary and the interior in the direction of the arrow.

A point $q \in Q$ is *visible* from $p \in Q$ if $[p, q] \subset Q$. The *visibility set* $\mathcal{V}(p) \subset Q$ is the set of points in Q visible from p. This notion can be extended as follows (see [17]):

Definition II.2 (Robust visibility) *Take $\epsilon > 0$ and $Q \subset \mathbb{R}^2$.*

(i) *The point $q \in Q$ is* ϵ-robustly visible *from the point $p \in Q$ if $\cup_{q' \in [p,q]} B(q', \epsilon) \subset Q$.*

(ii) *The* ϵ-robust visibility set *$\mathcal{V}(p, \epsilon) \subset Q$ from $p \in Q$ is the set of points in Q that are ϵ-robustly visible from p.*

(iii) *The* ϵ-contraction *Q_ϵ of the set Q is the set $\{p \in Q \mid ||p - q|| \geq \epsilon$ for all $q \in \partial Q\}$.*

These notions are illustrated in Figure 3. We present the following properties without proof in the interest of brevity.

Lemma II.3 *Given an allowable environment Q and $\epsilon > 0$, the following statements hold:*

(i) *$p, q \in Q$ are ϵ-robustly visible if and only if $[p, q] \subset Q_\epsilon$;*

(ii) *if ϵ is sufficiently small, then Q_ϵ is allowable; and*

(iii) *all strict concavities of ∂Q_ϵ have non-zero length and are continuously differentiable.*

[1]A vertex of a polygon is reflex if its interior angle is strictly greater than π.

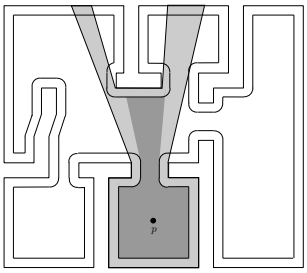

Fig. 3. Robust visibility notions. Q is the outer polygonal environment; the ϵ-contraction Q_ϵ is the region with the curved boundary and containing the point p; the visibility set $\mathcal{V}(p)$ is the region shaded in light gray; the ϵ-robust visibility set $\mathcal{V}(p, \epsilon)$ is the region shaded in darker gray. Note that the isolated concavities of Q give rise to strictly concave arcs in Q_ϵ.

Remarks II.4 (i) In light of Lemma II.3(ii), in what follows we assume that ϵ is small enough for Q_ϵ to be connected and therefore allowable.

(ii) Robust visibility is a useful concept in many practically meaningful ways. For example, according to this notion, points are visible only if they are at least at a distance ϵ from the boundary. This is useful when an object is arbitrarily close to the boundary and is indistinguishable from the boundary itself. Additionally, the parameter ϵ might be thought of as a measure of the physical size of the robot. Thus confining the robots to the ϵ-robust visibility set guarantees free movement of the robot in the environment. The notion of ϵ-contraction is related to the classical work on motion planning [20], [2]. □

We now define some graphs which are useful in describing the interactions between robots.

Definition II.5 (Proximity graphs) *A* proximity graph *is a graph whose nodes are a set of points $\mathcal{P} = \{p_1, \ldots, p_n\}$ and whose edges are a function of $\mathcal{P}$. Given $\mathcal{P} \subset Q$, $\epsilon > 0$ and $r > 0$, define:*

(i) *The* visibility graph *$\mathcal{G}_Q$ at $\mathcal{P}$ is the graph with node set $\mathcal{P}$ and with edges defined as follows: (p_i, p_j) is an edge if and only if $[p_i, p_j] \subset Q$.*

(ii) *The* ϵ-robust visibility graph *$\mathcal{G}_Q(\epsilon)$ is the visibility graph at $\mathcal{P}$ for Q_ϵ.*

(iii) *The* r-range ϵ-robust visibility graph *$\mathcal{G}_Q(\epsilon; r)$ at $\mathcal{P}$ is the graph with node set $\mathcal{P}$ and with edges defined as follows: (p_i, p_j) is an edge if and only if $[p_i, p_j] \subset Q_\epsilon$ and $\|p_i - p_j\| \leq r$.*

In other words, two points p, q are neighbors in the r-range visibility graph, for instance, if and only if they are mutually visible and separated by a distance less than or equal to r. Example graphs are shown in Figure 4. General properties of proximity graphs are defined in [18], [13].

We say that two proximity graphs $\mathcal{G}_1$ and $\mathcal{G}_2$ *have the same connected components* if, for all sets of points $\mathcal{P}$, the graphs $\mathcal{G}_1(\mathcal{P})$ and $\mathcal{G}_2(\mathcal{P})$ have the same number of connected components consisting of the same vertices. Given a set of points $\mathcal{P} = \{p_1, \ldots, p_n\}$ and a proximity graph $\mathcal{G}$, we let $\mathcal{N}_i(\mathcal{G})$ at $\mathcal{P}$ denote the *set of neighbors including itself* of p_i.

In other words, if $\{p_{i_1}, \ldots, p_{i_m}\}$ are the neighbors of p_i in $\mathcal{G}$ at $\mathcal{P}$, then $\mathcal{N}_i(\mathcal{G})$ at $\mathcal{P}$ is $\{p_{i_1}, \ldots, p_{i_m}\} \cup \{p_i\}$.

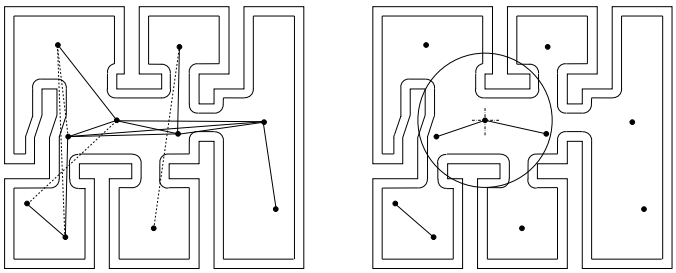

Fig. 4. The figure on the left shows the visibility graph (whose edges are the solid lines as well as the dashed lines) and the ϵ-robust visibility graph (whose edges are the solid lines alone) of a set of points in a nonconvex polygon. The figure on the right shows the r-range ϵ-robust visibility graph. The disk in the figure shows the sensing range for one of the robots.

The last key notion used in our technical approach is that of relative convex hull.

Definition II.6 (Relative convex hull) *Take an allowable environment Q.*

(i) $X \subseteq Q$ *is* relatively convex *if the shortest path inside Q connecting any two points of X is contained in X.*

(ii) *The* relative convex hull $\mathrm{rco}(X, Q)$ *of $X \subset Q$ is the smallest*[2] *relatively convex subset of Q that contains X.*

(iii) *If X is a finite set of points, then a* vertex *of* $\mathrm{rco}(X, Q)$ *is a point $p \in X$ with the property that* $\mathrm{rco}(X \setminus \{p\}, Q)$ *is a strict subset of* $\mathrm{rco}(X, Q)$*. The set of vertices of* $\mathrm{rco}(X, Q)$ *is denoted by* $\mathrm{Ve}(\mathrm{rco}(X, Q))$.

The relative convex hull of an example set of points and its vertices are shown in Figure 5.

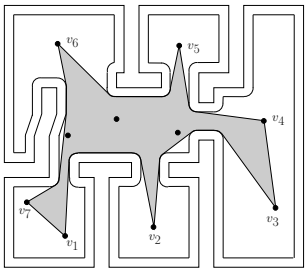

Fig. 5. Relative convex hull $\mathrm{rco}(X, Q_\epsilon)$ of a set of points X (solid disks) inside a the ϵ-contraction of an allowable set Q. The set of vertices $\mathrm{Ve}(\mathrm{rco}(X, Q_\epsilon))$ is the set $\{v_1, \ldots, v_7\}$.

The perimeter of the relative convex hull of a collection of points is defined next.

Definition II.7 (Perimeter) *For an allowable environment Q and a closed subset $X \subset Q$, the* $\mathrm{perimeter}(\mathrm{rco}(X, Q))$ *is the length of the shortest measurable closed curve inside Q enclosing X.*

The key property of Definition II.7 is that, if X is a finite set of points in Q, then the perimeter of $\mathrm{rco}(X, Q)$ depends continuously on the points in X.

[2]That is, $\mathrm{rco}(X, Q)$ is the intersection of all relatively convex subsets of Q that contain X.

III. Synchronous robots with visibility sensors and the rendezvous and connectivity maintenance problems

In this section we model a group of n robots with visibility sensors in a given allowable environment Q. We assume that ϵ is a known positive constant sufficiently small so that Q_ϵ is allowable. For $i \in \{1, \ldots, n\}$, we model the ith robot as a point $p_i \in Q$ and we refer to Section VI for an extension to a disk model. We make the following modeling assumptions:

Synchronized controlled motion model: Robot i moves at time $t \in \mathbb{Z}_{\geq 0}$ for a unit period of time, according to the discrete-time control system

$$p_i[t+1] = p_i[t] + u_i[t]. \tag{1}$$

We assume that there is a maximum step size $s_{\max} > 0$ for any robot, that is, $\|u_i\| \leq s_{\max}$. The robots are *synchronized* in the sense that the calculation of $u[t]$ in equation (1) takes place at the same times t for all robots.

Sensing model: The environment Q is unknown to the robots. Robot i senses (i) the presence and the position of any other robot that is visible and within distance r from p_i, and (ii) the subset of ∂Q that is visible and within distance $(r+\epsilon)$ from p_i. This in turn implies that the robot can sense the subset of ∂Q_ϵ that is visible and within distance r from p_i. It is convenient to define the *sensing region from position* p_i to be $\mathcal{S}(p_i) = \mathcal{V}(p_i, \epsilon) \cap B(p_i, r)$. The range r is the same for all robots.

Remark III.1 (No common reference frame) The model presented above assumes the ability of robots to sense absolute positions of other robots; this assumption is only made to keep the presentation as simple as possible. In this and subsequent remarks, we treat the more realistic setting in which the n robots have n distinct reference frames $\Sigma_1, \ldots, \Sigma_n$. We let Σ_0 denote a fixed reference frame. Notation-wise, a point q, a vector w, and a set of points S expressed with respect to frame Σ_i are denoted by q^i, w^i and S^i, respectively. For example, this means that Q^i is the environment Q as expressed in frame Σ_i. We assume that the origin of Σ_i is p_i and that the orientation of Σ_i with respect to Σ_0 is $R_i^0 \in \mathrm{SO}(2)$. Therefore, changes of reference frames are described by the equations: $q^0 = R_i^0 q^i + p_i^0$, $w^0 = R_i^0 w^i$, and $S^0 = R_i^0 S^i + p_i^0$. If we let $\mathcal{V}_{Q^j}(p_i^j, \epsilon)$ denote the visibility set expressed in Σ_j, for $j \in \{0, 1, \ldots, n\}$, then one can define

$$\mathcal{S}(p_i^j, Q^j) = \mathcal{V}_{Q^j}(p_i^j, \epsilon) \cap B(p_i^j, r),$$

and verify $\mathcal{S}(p_i^0, Q^0) = R_i^0 \mathcal{S}(p_i^i, Q^i) + p_i^0$. Note that $p_i^i = 0$.

Finally, we can describe our motion and sensing model under the no common reference frame assumption. Robot i moves according to

$$p_i^0[t+1] = p_i^0[t] + R_i^0[t] u_i[t], \tag{2}$$

and it senses the robot positions p_j^i and the subset of $(\partial Q)^i$ that are within the sensing region $\mathcal{S}(p_i^i, Q^i)$. □

We end this section by stating the two control design problems addressed in this paper.

Problem III.2 (Rendezvous) The *rendezvous problem* is to steer each agent to a common location inside the environment Q_ϵ. This objective is to be achieved (1) with the limited information flow described in the model above, and (2) under the reasonable assumption that the initial position of the robots $\mathcal{P}[0] = \{p_1[0], \ldots, p_n[0]\}$ gives rise to a connected robust visibility graph $\mathcal{G}_Q(\epsilon)$ at $\mathcal{P}[0]$. □

As one might imagine, the approach to solving the rendezvous problem involves two main ideas: first, the underlying proximity graph should not lose connectivity during the evolution of the group; second, while preserving the connectivity of the graph, the robots must move closer to each other. This discussion motivates a second complementary objective.

Problem III.3 (Connectivity maintenance) The *connectivity maintenance problem* is to design (state dependent) control constraint sets with the following property: if each agent's control takes values in the control constraint set, then the robots move in such a way that the number of connected components of $\mathcal{G}_Q(\epsilon)$ (evaluated at the robots' states) does not increase with time. □

IV. The connectivity maintenance problem

In this section, we maintain the connectivity of the group of robots with visibility sensors by designing control constraint sets that guarantee that every edge of $\mathcal{G}_Q(\epsilon; r)$ (i.e., every pair of mutually range-limited visible robots) is preserved. We have three objectives in doing so. First, the sets need to depend continuously on the position of the robots. Second, the sets need to be computed in a distributed way based only on the available sensory information. Third, the control constraint sets should be as "large" as possible so as to minimally constrain the motion of the robots. Our solution to this problem is a geometric strategy that allows us to compute appropriate constraint sets. We discuss it in detail in the next section.

A. Preserving mutual visibility: The Constraint Set Generator Algorithm

Consider a pair of robots in an environment Q that are ϵ-robustly visible to each other and separated by a distance not larger than r. To preserve this range-limited mutual visibility property, we restrict the motion of the robots to an appropriate subset of the environment. This idea is inspired by [1] and we begin by stating the result therein. Let the sensing region of robot i located at p_i be $\mathcal{S}(p_i) = B(p_i, r)$, for some $r > 0$. If at any time instant t, $\|p_i[t] - p_j[t]\| \leq r$, then to ensure that at the next time instant $t+1$, $\|p_i[t+1] - p_j[t+1]\| \leq r$, it suffices to impose the following constraints on the motion of robots i and j:

$$p_i[t+1],\ p_j[t+1] \in B\Big(\frac{p_i[t]+p_j[t]}{2}, \frac{r}{2}\Big),$$

or, equivalently,

$$u_i[t] \in B\Big(\frac{p_j[t]-p_i[t]}{2}, \frac{r}{2}\Big),\ u_j[t] \in B\Big(\frac{p_i[t]-p_j[t]}{2}, \frac{r}{2}\Big).$$

In summary, $B(\frac{p_j-p_i}{2}, \frac{r}{2})$ is the control constraint set for robot i. This constraint is illustrated in Figure 6 (left).

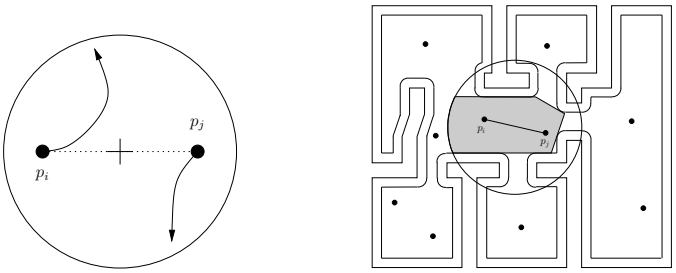

Fig. 6. In the figure on the left, starting from p_i and p_j, the robots are restricted to move inside the disk centered at $\frac{p_i+p_j}{2}$ with radius $\frac{r}{2}$. In the figure on the right, the robots are constrained to move inside the shaded region which is a convex subset of Q_ϵ intersected with the disk centered at $\frac{p_i+p_j}{2}$ with radius $\frac{r}{2}$.

Let us now consider the case when a robot i is located at p_i in a nonconvex environment Q with sensing region $\mathcal{S}(p_i) = \mathcal{V}(p_i, \epsilon) \cap B(p_i, r)$. If at any time instant t, we have that $\|p_i[t] - p_j[t]\| \leq r$ and $[p_i[t], p_j[t]] \in Q_\epsilon$, then to ensure that $\|p_i[t+1] - p_j[t+1]\| \leq r$ and $[p_i[t+1], p_j[t+1]] \in Q_\epsilon$, it suffices to require that:

$$p_i[t+1],\ p_j[t+1] \in \mathcal{C},$$

where $\mathcal{C}$ is any convex subset of $Q_\epsilon \cap B\big(\frac{p_i[t]+p_j[t]}{2}, \frac{r}{2}\big)$; see Figure 6 (right). Equivalently,

$$u_i[t] \in \mathcal{C} - p_i[t],\ u_j[t] \in \mathcal{C} - p_j[t],$$

where $\mathcal{C} - p_i[t]$ and $\mathcal{C} - p_i[t]$ are the sets $\{p - p_i[t] \mid p \in \mathcal{C}\}$ and $\{p - p_j[t] \mid p \in \mathcal{C}\}$, respectively. Note that both robots i and j must independently compute the same set $\mathcal{C}$. Given the positions p_i, p_j in an environment Q, Table I describes the Constraint Set Generator Algorithm, a geometric strategy for each robot to compute a constraint set $\mathcal{C} = \mathcal{C}_Q(p_i, p_j)$ that changes continuously with p_i and p_j. Figure 7 illustrates a step-by-step execution of the algorithm.

TABLE I
Constraint Set Generator Algorithm

Goal:	Generate convex sets to act as constraints to preserve mutual visibility
Given:	$(p_i, p_j) \in Q_\epsilon^2$ such that $[p_i, p_j] \subseteq Q_\epsilon$ and $p_j \in B(p_i, r)$

Robot $i \in \{1, \ldots, n\}$ executes the following computations:

1: $\mathcal{C}_{\text{temp}} := \mathcal{V}(p_i, \epsilon) \cap B(\frac{p_i+p_j}{2}, \frac{r}{2})$
2: **while** $\partial\mathcal{C}_{\text{temp}}$ contains a concavity **do**
3: $\quad v :=$ a strictly concave point of $\partial\mathcal{C}_{\text{temp}}$ closest to the segment $[p_i, p_j]$
4: $\quad \mathcal{C}_{\text{temp}} := \mathcal{C}_{\text{temp}} \cap H_{Q_\epsilon}(v)$
5: **end while**
6: **return:** $\mathcal{C}_Q(p_i, p_j) := \mathcal{C}_{\text{temp}}$

Note that in step 3 of the algorithm, there can be many distinct points belonging to distinct concavities that satisfy the required property. In that case, v can be chosen to be *any* one of them. The following lemma justifies this observation.

Lemma IV.1 *Throughout the execution of the* Constraint Set Generator Algorithm *in Table I, let v_1, v_2 be two strictly concave points on $\partial\mathcal{C}_{\text{temp}}$ that are closest to $[p_i, p_j]$. Then $v_1 \in \mathcal{C}_{\text{temp}} \cap H_{Q_\epsilon}(v_2)$ and vice versa.*

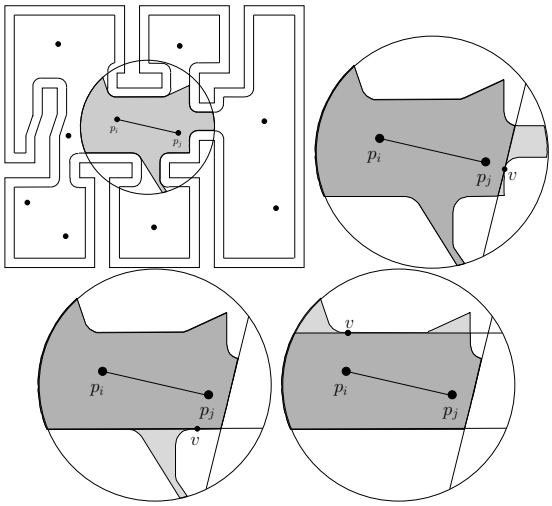

Fig. 7. From left to right and top to bottom, a sample incomplete run of the Constraint Set Generator Algorithm (cf. Table I). The top left figure shows $\mathcal{C}_{\text{temp}} := \mathcal{V}(p_i, \epsilon) \cap B(\frac{p_i+p_j}{2}, \frac{r}{2})$. In all the other figures, the lightly and darkly shaded regions together represent $\mathcal{C}_{\text{temp}}$. The darkly shaded region represents $\mathcal{C}_{\text{temp}} \cap H_Q(v)$, where v is as described in step 3. The final outcome of the algorithm, $\mathcal{C}_Q(p_i, p_j)$, is shown in Figure 6 (right).

Next, we characterize the main properties of the Constraint Set Generator Algorithm and the corresponding convex sets.

Proposition IV.2 (Properties of the Constraint Set Generator Algorithm) *Given an allowable environment Q with κ strict concavities, $\epsilon > 0$ and $(p_i, p_j) \in J = \{(p_i, p_j) \in Q_\epsilon^2 \mid [p_i, p_j] \in Q_\epsilon,\ \|p_i - p_j\| \leq r\}$, the following hold:*

(i) *the* Constraint Set Generator Algorithm *terminates in at most κ steps;*
(ii) *$\mathcal{C}_Q(p_i, p_j)$ is nonempty, compact, and convex;*
(iii) *$\mathcal{C}_Q(p_i, p_j) = \mathcal{C}_Q(p_j, p_i)$; and*
(iv) *the set-valued map $\mathcal{C}_Q$ is closed*[3] *at every point of J.*

Remark IV.3 (No common reference frame: continued) Consider a group of robots with visibility sensors and no common reference frame. With the notation and assumptions of Remark III.1, one can verify that the constraint sets transform under changes of coordinate frames according to

$$\mathcal{C}_{Q^0}(p_i^0, p_j^0) = R_i^0 \mathcal{C}_{Q^i}(p_i^i, p_j^i) + p_i^0. \tag{3}$$

We omit the proof in the interest of brevity. □

For each pair of mutually visible robots, the execution of the Constraint Set Generator Algorithm outputs a control constraint set such that, if the robots' motions are constrained to it, then the robots remain mutually visible. Clearly, given a connected graph at time t, if every robot remains connected with all its neighbors at time $t+1$ (i.e., each pair of mutually visible robots remain mutually visible), then the connectivity

[3]Let Ω map points in X to all possible subsets of Y. Then the set-valued map, Ω, is open at a point $x \in X$ if for any sequence $\{x_k\}$ in X, $x_k \to x$ and $y \in \Omega(x)$ implies the existence of a number m and a sequence $\{y_k\}$ in Y such that $y_k \in \Omega(x_k)$ for $k \geq m$ and $y_k \to y$. The map Ω is closed at a point $x \in X$ if for any sequence $\{x_k\}$ in X, $x_k \to x$, $y_k \to y$ and $y_k \in \Omega(x_k)$ imply that $y \in \Omega(x)$. Ω is continuous at any point $x \in X$ if it is both open and closed at x.

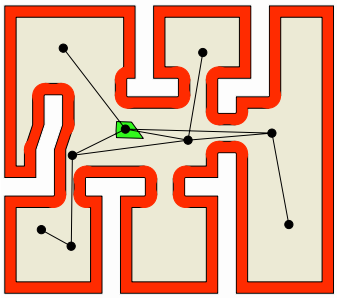

Fig. 8. The green convex set in the center represents $C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_Q(\epsilon; r)))$. The black disks represent the position of the robots. The straight line segments between pairs of robots represent edges of $\mathcal{G}_Q(\epsilon; r)$. Here, p_i is the black disk contained in the constraint set.

of the graph is preserved. This can be accomplished as follows. For robot i at $p_i \in Q_\epsilon$, define the control constraint set

$$C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_Q(\epsilon; r))) = \bigcap_{p_j \in \mathcal{N}_i(\mathcal{G}_Q(\epsilon; r))} \mathcal{C}_Q(p_i, p_j). \tag{4}$$

Now, if $u_i \in C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_Q(\epsilon; r))) - p_i$, for all $i \in \{1, \ldots, n\}$, then all neighboring relationships in $\mathcal{G}_Q(\epsilon; r)$ are preserved at the next time instant. Using inputs that satisfy these constraints, the number of edges in $\mathcal{G}_Q(\epsilon; r)$ is guaranteed to be nondecreasing.

B. The locally-cliqueless visibility graph

In this section, we propose the construction of constraint sets that are, in general, larger than $C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_Q(\epsilon; r)))$. To do this, we define the notion of *locally-cliqueless graph*. The locally-cliqueless graph of a proximity graph $\mathcal{G}$ is a subgraph of $\mathcal{G}$, and therefore has generally fewer edges, but it has the same number of connected components as $\mathcal{G}$. This property is fundamental because it directly leads to the design of less conservative constraint sets.

Before defining the locally-cliqueless graph, let us recall that (i) a *clique* of a graph is a complete subgraph of it, and (ii) a *maximal clique of an edge* is a clique of the graph that contains the edge and is not a strict subgraph of any other clique of the graph that also contains the edge. In the field of combinatorial optimization, it is well-known that finding the maximal clique of a graph is an NP complete problem. However, efficient polynomial time heuristics exist [21]. Additionally, we define a useful proximity graph. A *Euclidean Minimum Spanning Tree* $\mathcal{G}_{\text{EMST}}(\mathcal{G})$ at $\mathcal{P}$ of a proximity graph $\mathcal{G}$ is a minimum-length spanning tree of $\mathcal{G}(\mathcal{P})$, where edges of the form (p_i, p_j) have length $\|p_i - p_j\|$. If $\mathcal{G}(\mathcal{P})$ is not connected, then $\mathcal{G}_{\text{EMST}}(\mathcal{G})$ at $\mathcal{P}$ is the union of Euclidean Minimum Spanning Trees of its connected components.

Definition IV.4 (Locally-cliqueless graph of a proximity graph) *Given a point set $\mathcal{P}$ and an allowable environment Q, the* locally-cliqueless graph $\mathcal{G}_{\text{lc}}(\mathcal{G})$ *at $\mathcal{P}$ of a proximity graph $\mathcal{G}$ is the proximity graph with node set $\mathcal{P}$ and with edges at $\mathcal{P}$ defined as follows: (p_i, p_j) is an edge in $\mathcal{G}_{\text{lc}}(\mathcal{G})$ if and only if (p_i, p_j) is an edge of $\mathcal{G}(\mathcal{P})$ and (p_i, p_j) is an edge of $\mathcal{G}_{\text{EMST}}(\mathcal{G})$ at $\mathcal{P}'$ for any maximal clique $\mathcal{P}'$ of the edge (p_i, p_j) in $\mathcal{G}$.*

For simplicity, we will refer to the locally-cliqueless graph of the proximity graphs $\mathcal{G}_Q$, $\mathcal{G}_Q(\epsilon)$ or $\mathcal{G}_Q(\epsilon; r)$ as *locally-cliqueless visibility graphs*. Figure 9 shows an example of a locally-cliqueless visibility graph.

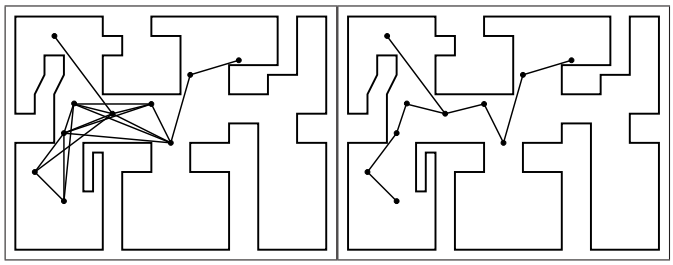

Fig. 9. Visibility graph (left) and locally-cliqueless visibility graph (right).

Theorem IV.5 (Properties of a locally-cliqueless graph of a proximity graph) *Let $\mathcal{G}$ be a proximity graph. Then, the following statements hold:*

(i) $\mathcal{G}_{\text{EMST}}(\mathcal{G}) \subseteq \mathcal{G}_{\text{lc}}(\mathcal{G}) \subseteq \mathcal{G}$*; and*

(ii) $\mathcal{G}_{\text{lc}}(\mathcal{G})$ *and $\mathcal{G}$ have the same connected components.*

In general, the inclusions in Theorem IV.5(i) are strict. Figure 10 shows an example with $\mathcal{G}_{\text{EMST}}(\mathcal{G}_Q) \subsetneq \mathcal{G}_{\text{lc}}(\mathcal{G}_Q) \subsetneq \mathcal{G}_Q$.

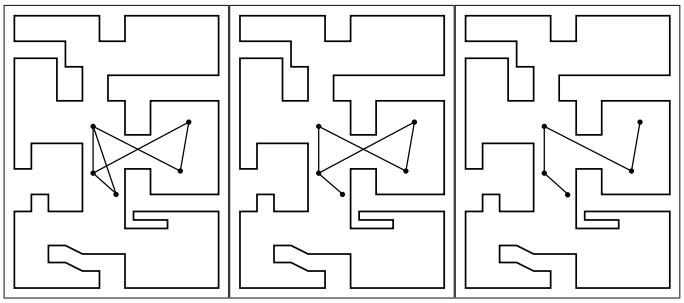

Fig. 10. From left to right, visibility graph, locally-cliqueless graph and Euclidean Minimum Spanning Tree of the visibility graph.

We will invoke Theorem IV.5 for $\mathcal{G}_Q$, $\mathcal{G}_Q(\epsilon)$ or $\mathcal{G}_Q(\epsilon; r)$ defined over allowable environments Q and Q_ϵ, $\epsilon > 0$.

We are now ready to define new constraint sets that are in general larger than the ones defined in (4). For simplicity, let $\mathcal{G} = \mathcal{G}_Q(\epsilon; r)$, and consider its locally-cliqueless graph $\mathcal{G}_{\text{lc}}(\mathcal{G})$. For robot $i \in \{1, \ldots, n\}$ located at p_i, define the constraint set

$$C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{lc}}(\mathcal{G}))) = \bigcap_{p_j \in \mathcal{N}_i(\mathcal{G}_{\text{lc}}(\mathcal{G}))} \mathcal{C}_Q(p_i, p_j). \quad (5)$$

Since $\mathcal{G}_{\text{lc}}(\mathcal{G})$ is a subgraph of $\mathcal{G}$ according to Theorem IV.5(i)), we have $\mathcal{N}_i(\mathcal{G}_{\text{lc}}(\mathcal{G})) \subseteq \mathcal{N}_i(\mathcal{G}) = \mathcal{N}_i(\mathcal{G}_Q(\epsilon; r))$, and therefore

$$C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_Q(\epsilon; r))) \subseteq C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{lc}}(\mathcal{G}))).$$

In general, since $\mathcal{G}_{\text{lc}}(\mathcal{G})$ is a strict subgraph of $\mathcal{G}$, the set $C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{lc}}(\mathcal{G})))$ is strictly larger that $C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_Q(\epsilon; r)))$. Note that, if $u_i \in C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{lc}}(\mathcal{G}))) - p_i$ for all $i \in \{1, \ldots, n\}$, then all neighboring relationships in the graph $\mathcal{G}_{\text{lc}}(\mathcal{G})$ are preserved at the next time instant. As a consequence, it follows from Theorem IV.5(ii) that the connected components of $\mathcal{G}_Q(\epsilon; r)$ are also preserved at the next time instant. Thus, we have found constraint sets (5) for the input that are larger than the constraint sets (4), and are yet sufficient to preserve the connectivity of the overall group.

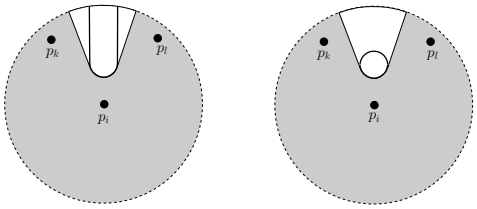

Fig. 11. The dashed circle is centered at p_i and is of radius r. The thick curves represent the boundary of Q_ϵ; the one on the left represents the outer boundary whereas the one on the right represents a hole in the environment.

Remark IV.6 (Distributed computation of locally-cliqueless visibility graphs) According to the model specified in Section III, each robot can detect all other robots in its sensing region $\mathcal{S}(p_i) = \mathcal{V}(p_i, \epsilon) \cap B(p_i, r)$, i.e., its neighbors in the graph $\mathcal{G}_Q(\epsilon; r)$. Given the construction of the constraint sets in this section, it is important to guarantee that the set of neighbors of robot i in the locally-cliqueless graph $\mathcal{G}_{\text{lc}}(\mathcal{G})$ can be computed locally by robot i. From the definition of the locally-cliqueless graph, this is indeed possible if a robot i can detect whether another robot j in its sensing region $\mathcal{S}(p_i)$ belongs to a clique of the graph $\mathcal{G}_Q(\epsilon; r)$. This is equivalent to being able to check if two robots $p_k, p_l \in \mathcal{S}(p_i)$ satisfy the condition that $p_k \in \mathcal{S}(p_l)$ and vice versa. Note that $p_k \in \mathcal{S}(p_l)$ is equivalent to $\|p_k - p_l\| \leq r$ and $[p_k, p_l] \subseteq Q_\epsilon$. Given that $p_k - p_l = (p_k - p_i) - (p_l - p_i)$, the vector $p_k - p_l$ (and hence $\|p_k - p_l\|$) can be computed based on local sensing alone. Now, checking if $[p_k, p_l] \subseteq Q_\epsilon$ is possible only if Q_ϵ does not contain any hole; see Figure 11. In such a case, it suffices to check if the entire line segment $[p_k, p_l]$ is visible from p_i or not.

Along similar lines, we can state that the locally-cliqueless visibility graph is computable under the "no common reference frame" model described in Remarks III.1 and IV.3. □

V. The rendezvous problem: Algorithm design and analysis results

In this section, we solve the rendezvous problem through a novel Perimeter Minimizing Algorithm. The algorithm is inspired by the one introduced in [1] but is unique in many different ways. The rendezvous algorithm uses different graphs to maintain connectivity and to move closer to other robots. Instead of moving towards the circumcenter of the neighboring robots, the robots move towards the center of a suitably defined motion constraint set.

We present the algorithm in Section V-A followed by its main convergence properties in Section V-B.

A. *The* Perimeter Minimizing Algorithm

We begin with an informal description of the Perimeter Minimizing Algorithm over graphs $\mathcal{G}_{\text{sens}}$ and $\mathcal{G}_{\text{constr}}$. The sensing graph $\mathcal{G}_{\text{sens}}$ is $\mathcal{G}_Q(\epsilon; r)$ while the constraint graph $\mathcal{G}_{\text{constr}}$ is either $\mathcal{G}_{\text{sens}}$ or $\mathcal{G}_{\text{lc}}(\mathcal{G}_{\text{sens}})$:

Every robot i performs the following tasks: (i) it acquires the positions of other robots that are its neighbors according to $\mathcal{G}_{\text{sens}}$; (ii) it computes a point that is "closer" to the robots it senses, and (iii) it moves toward this point while maintaining connectivity with its neighbors according to $\mathcal{G}_{\text{constr}}$.

The algorithm is formally described in Table II; Figure 1 in the Introduction illustrates an example execution.

TABLE II
Perimeter Minimizing Algorithm

Assumes: (i) Q is allowable
(ii) $\mathcal{G}_{\text{sens}}$ is $\mathcal{G}_Q(\epsilon; r)$; $\mathcal{G}_{\text{constr}}$ is either $\mathcal{G}_{\text{sens}}$ or $\mathcal{G}_{\text{lc}}(\mathcal{G}_{\text{sens}})$
(iii) $s_{\max} > 0$ is the maximal step size

Each robot $i \in \{1, \dots, n\}$ executes the following steps at each time instant:

1: acquire $\{p_{i_1}, \dots, p_{i_m}\}$:= positions of robots within p_i sensing region
2: compute $\mathcal{N}_i(\mathcal{G}_{\text{sens}})$ and $\mathcal{N}_i(\mathcal{G}_{\text{constr}})$
3: compute $X_i := C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{constr}})) \cap \text{rco}(\mathcal{N}_i(\mathcal{G}_{\text{sens}}), \mathcal{V}(p_i, \epsilon))$
4: compute $p_i^* := \text{CC}(X_i)$
5: **return:** $u_i := \min\left(s_{\max}, \|p_i^* - p_i\|\right) \text{vers}(p_i^* - p_i)$

Remarks V.1 (i) In the algorithm proposed in [1], robots move towards the circumcenter of their neighbors' position. In the Perimeter Minimizing Algorithm, robots move towards the circumcenter of their constraint set.

(ii) One can prove that the set X_i is convex; see [19]. Therefore, $\text{CC}(X_i) \in X_i$, and hence $p_i^* \in X_i$. Also, $p_i \in X_i$. Therefore, $u_i \in X_i - p_i \subseteq C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{constr}})) - p_i$ and, in turn, p_i at the next time instant belongs to $C_{p_i,Q}(\mathcal{N}_i(\mathcal{G}_{\text{constr}}))$. From Section IV, this implies that the graph $\mathcal{G}_{\text{constr}}$ remains connected (or, more generally, that the number of connected components of $\mathcal{G}_{\text{constr}}$ does not decrease). Therefore, by Theorem IV.5, the number of connected components of $\mathcal{G}_{\text{sens}}$ also does not decrease.

(iii) If the initial positions of the robots are in Q_ϵ, then the robots will remain forever in Q_ϵ because $p_i^* \in X_i \subseteq Q_\epsilon$.

(iv) All information required to execute the steps in the algorithm is available to a robot through the sensing model described in Section III. The constraint on the input size, $\|u_i\| \leq s_{\max}$, is enforced in step 5. □

Finally, we conclude this section by completing our treatment of robots without a common reference frame.

Remark V.2 (No common reference frame: continued) Consider a group of robots with visibility sensors and no common reference frame as discussed in Remarks III.1 and IV.3. Because the relative convex hull and the circumcenter of a set transform under changes of coordinate frames in the same way as the constraint set does in equation (3), one can verify that

$$u_i(p_1^0, \dots, p_n^0) = R_i^0 u_i(p_1^i, \dots, p_n^i),$$

where $u_i(p_1^0, \dots, p_n^0)$ is computed with environment Q^0 and $u_i(p_1^i, \dots, p_n^i)$ is computed with environment Q^i. This equality implies that the robot motion with control $u_i(p_1^0, \dots, p_n^0)$ in equation (1) is identical to the robot motion with control $u_i(p_1^i, \dots, p_n^i)$ in equation (2). □

B. Convergence properties

To state the main results on the correctness of the Perimeter Minimizing Algorithm, we require some preliminary notation. First, note that given the positions of the robots $\{p_1, \dots, p_n\}$ at time instant t, the algorithm computes the positions at time instant $t+1$. The Perimeter Minimizing Algorithm, therefore, is a map, say $T_{\mathcal{G}_{\text{sens}},\mathcal{G}_{\text{constr}}} : Q_\epsilon^n \to Q_\epsilon^n$. Second, we work with tuple of points $P = (p_1, \dots, p_n) \in Q^n$ for convenience. We let $\mathcal{G}(P)$ denote the proximity graph $\mathcal{G}(\mathcal{P})$ and $\text{rco}(P, Q)$ denote the relative convex hull of the set $\mathcal{P}$ inside Q, where $\mathcal{P}$ is the point set given by $\{p_i \mid i \in \{1, \dots, n\}\}$. Third, we introduce a Lyapunov function that encodes the rendezvous objective. Given an allowable environment Q, we recall the notions of relative convex hull and of perimeter from Section II, and define $V_{\text{perim},Q} : Q^n \to \mathbb{R}_{\geq 0}$ by

$$V_{\text{perim},Q}(P) = \text{perimeter}(\text{rco}(P, Q)).$$

Lemma V.3 (Properties of Lyapunov function) *The function $V_{\text{perim},Q}$ has the following properties:*

(i) *$V_{\text{perim},Q}$ is continuous and invariant under permutations of its arguments; and*

(ii) *$V_{\text{perim},Q}(P) = 0$ for $P = (p_1, \dots, p_n)$ if and only if $p_i = p_j$ for all $i, j \in \{1, \dots, n\}$.*

This result implies that achieving the rendezvous objective is equivalent to making $V_{\text{perim},Q_\epsilon}$ equal to zero. The proof strategy is based on establishing the monotonic decreasing evolution of this function along the executions of Perimeter Minimizing Algorithm. We now state the main result of the paper.

Theorem V.4 (Rendezvous is achieved via the Perimeter Minimizing Algorithm) *Let Q and Q_ϵ be allowable environments. Let $p_1, \dots, p_n$ be a group of robots with visibility sensors in Q_ϵ. Any trajectory $\{P[t]\}_{t \in \mathbb{Z}_{\geq 0}} \subset Q_\epsilon$ generated by $P[t+1] = T_{\mathcal{G}_{\text{sens}},\mathcal{G}_{\text{constr}}}(P[t])$ has the following properties:*

(i) *if the locations of two robots belong to the same connected component of $\mathcal{G}_{\text{sens}}$ at $P[t_0]$ for some t_0, then they remain in the same connected component of $\mathcal{G}_{\text{sens}}$ at $P[t]$ for all $t \geq t_0$;*

(ii) *$V_{\text{perim},Q_\epsilon}(P[t+1]) \leq V_{\text{perim},Q_\epsilon}(P[t])$; and*

(iii) *the trajectory $\{P[t]\}_{t \in \mathbb{Z}_{\geq 0}}$ converges to a point $P^* \in Q_\epsilon$ such that either $p_i^* = p_j^*$ or $p_i^* \notin \mathcal{S}(p_j^*)$ for all $i, j \in \{1, \dots, n\}$.*

As a direct consequence of the theorem, note that if the graph $\mathcal{G}_{\text{sens}}$ is connected at any time during the evolution of the system, then all the robots converge to the same location in Q_ϵ.

VI. Practical implementation issues

The analysis of the Perimeter Minimizing Algorithm presented in Section V is valid for an ideal model of *point* robots, operating *synchronously*, with *perfect sensing and actuation capabilities*. However, such an ideal model is not realistic in practical situations. In this section, we investigate, via extensive computer simulations, the effects of deviations from this ideal scenario.

A. Nominal experimental set-up

The computer simulation was written in C++ using the Computational Geometry Algorithmic Library [22]. However, it was found that Boolean operations on polygons using the utilities present in CGAL were not sufficiently fast for the purpose of running extensive simulations. Hence, Boolean operations on polygons were performed using the General Polygon Clipping Library [23].

For the purpose of simulations, the environment considered is a typical floor plan; see Figure 1. The environment size is roughly 80×70, the step size of a robot is $s_{\max} = 0.5$ and the sensing radius $r = 30$. For simplicity, $\mathcal{G}_{\text{constr}}$ is the same as $\mathcal{G}_{\text{sens}}$. To use the ϵ-robust visibility notion in providing robustness to asynchronism and sensing and control errors, at each time instant, ϵ is set to be 0.97 times the value of ϵ at the previous time instant. Initially, ϵ is equal to 3. In case a robot approaches a reflex vertex of the environment closely, it reduces its speed. This is done to reduce the risk of collision due to errors in sensing the exact location of the reflex vertex.

The algorithm performance is then evaluated using the following measures: (i) the average number of steps taken by the robots to achieve the rendezvous objective; and (ii) the number of connected components of $\mathcal{G}_{\text{sens}}$ at the end of the simulation compared with the number of connected components at the start.

B. Robustness against asynchronism, sensing and control noise, and finite-size disk robot models

In this section, we begin by describing the various non-ideal conditions introduced into the experiments. We then describe the performance of the Perimeter Minimizing Algorithm under these non-ideal conditions.

1) Asynchronism: The robots operate asynchronously, i.e., do not share a common processor clock. All the robots start operating at the same time. Each robot's clock speed is a random number uniformly distributed on the interval $[0.9, 1]$. At integral multiples of its clock speed, a robot wakes up, senses the positions of other robots within its sensing region, and takes a step according to the Perimeter Minimizing Algorithm.

2) Distance error in sensing and control: The visibility sensors measure the relative distance of another object according to the following multiplicative noise model. If d_{act} is the actual distance to the object, then the measured distance is given by $(1 + e_{1,\text{dist}} + e_{2,\text{dist}} d_{\text{act}}) d_{\text{act}}$, where $e_{1,\text{dist}}$ and $e_{2,\text{dist}}$ are random variables uniformly distributed in the intervals $[-0.1, 0.1]$ and $[-0.003, 0.003]$, respectively. Therefore, the measured distance is correct up to an error of maximum magnitude equal to 20% of the actual distance. The objects to which distances are measured are other robots in the sensing range and the boundary of the environment. For simplicity, instead of measuring a sequence of points along the boundary, as a real range sensor does, we assume that only the vertices of the environment are measured and the sensed region is reconstructed from that information. The sensor error, therefore, occurs in the measurement of other robots and environment vertices. The actuators moving the robots are also subject to a multiplicative noise distance model with the same error parameter.

3) Direction error in sensing and control: The visibility sensors measure the relative angular location of another object according to the following additive noise model. If θ_{act} is the actual angular location of any object in the local reference frame of a robot, the measured angular location is given by $\theta_{\text{act}} + e_\theta$, where e_θ is a random variable uniformly distributed in the interval $[-5, 5]$. As before, the actuators moving the robots are also subject to an additive noise directional model with the same error parameter.

4) Disk robot model (with asynchronism, distance error, and direction error): The robots are assumed to be disks of radius 0.5. This disk model also implies that robots may occlude the view of other robots. We assume that two robots can detect each other only if the line segment joining their centers does not intersect with any other robot disk. Also, any strictly concave point of the boundary is visible from a robot only if the line segment joining the center of the robot to that point does not intersect with another robot disk. During any step, a robot moves a distance of at most $s_{\max} = 0.5$. The next position of the center of the robot, therefore, lies in a *motion disk* of radius 1.0 centered at the center of the robot. A *colliding neighbor* of a robot i is any neighbor j according to $\mathcal{G}_{\text{sens}}$ such that the motion disks of i and j intersect and that the motion disk of j intersects the physical disk of i on the path between i and its next point. If a robot has no colliding neighbors, then its motion is executed according to Perimeter Minimizing Algorithm. On the other hand, if a robot has exactly one colliding neighbor, then it tries to swerve around it while reducing its speed. Finally, if the robot has more than two colliding neighbors then it stays at the current location. To ensure free movement of the robots inside the environment, ϵ is not allowed to fall below 0.5, which is the radius of a robot disk.

Experiments were performed under the aforesaid non-ideal conditions with 20 robots starting from 100 randomly generated initial conditions from a uniform distribution. To account for the non-zero robot size for the purpose of simulations, the rendezvous objective is considered to be achieved if the robots belonging to each connected component of $\mathcal{G}_{\text{sens}}$ form a "cohesive" group[4]. We also place an upper bound on the total time for which any given experiment can run. The algorithm performance in each experiment is evaluated using the following two measures: (i) the ratio of the number of cohesive groups at the end of the experiment to the number of connected components in the sensing graph $\mathcal{G}_{\text{sens}}$ at the beginning of the experiment; and (ii) the number of steps on average taken by a robot until the completion of the experiment. For the sake of convenience, we shall refer to the above measures simply as Measure 1 and Measure 2 in the rest of the paper.

Experiments are conducted to investigate if the Perimeter Minimizing Algorithm under various non-ideal conditions is still able to achieve the primary rendezvous objective - preserve the connected components of the initial sensing graph and bring all the robots within each connected component into

[4]For each connected component of $\mathcal{G}_{\text{sens}}$, the graph having nodes as the robot locations and with an edge between two nodes whenever the corresponding motion disks of the robots intersect is connected.

a cohesive group. We compare three variants of the Perimeter Minimizing Algorithm: (a) the synchronous implementation assuming no errors and a point robot model, (b) the asynchronous implementation with sensing and control errors but with a point robot model, and (c) the asynchronous implementation with sensing and control errors with a disc robot model. The associated mean values and the 95 percent intervals [24] of the averages of the two performance measures described earlier are shown in Figure 12. The standard deviation for Measure 1 for cases (a), (b), and (c) above are 0.25, 0.34 and 0.29 respectively. The standard deviation for Measure 2 for cases (a), (b), and (c) above are 17.64, 12.76 and 80.94 respectively. Our observations are as follows.

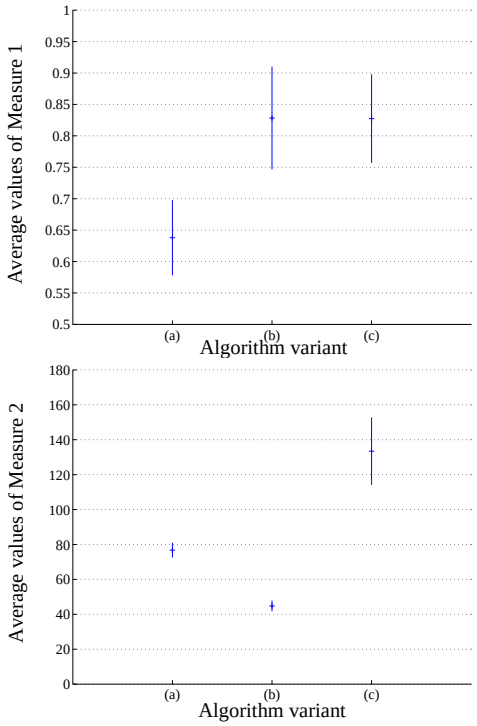

Fig. 12. Figure showing the average values of Measure 1 (top) and Measure 2 (bottom) for three different variants of the Perimeter Minimizing Algorithm. Also shown are the associated 95 percent confidence intervals of the averages.

(i) The Perimeter Minimizing Algorithm is robust in the presence of errors, asynchronism and finite robot size, i.e., in all the cases (a), (b), and (c), the primary rendezvous objective is achieved. We test this hypothesis, for example for case (c), as follows. We perform the Student's $t-$test [24] with the null hypothesis being that the average of the Measure 1 values is one against the alternate hypothesis that the average is less than one at the 95 percent confidence level. The $t-$test results are shown in Table III. The field *df* refers to the degrees of freedom. We assume that the Measure 1 value across different experiments is an independent and identically distributed random variable. Because of the large number of data points, we can assume that the average Measure 1 value is normally distributed according to the Central Limit Theorem; thus the $t-$test is applicable in this case [24]. The test results imply that we can reject the null hypothesis in favor of the alternate one. This is also evidenced by the fact that all the values within the 95 percent confidence interval for the average of Measure 1 are strictly less than one. This implies that the number of cohesive groups at the end of the simulation is less than the number of connected components in the sensing graph $\mathcal{G}_{\text{sens}}$ at the start of the experiment. In other words, all robots initially connected in $\mathcal{G}_{\text{sens}}$ do converge to form a cohesive group.

(ii) The performance of the Perimeter Minimizing Algorithm worsens with the introduction of non-ideal behavior as can be seen by the larger number of connected components (Measure 1) in cases (b) and (c) as compared to case (a) in Figure 12(top).

TABLE III
STUDENT'S T-TEST RESULTS FOR DEMONSTRATING THE ROBUSTNESS OF THE PERIMETER MINIMIZING ALGORITHM

Mean	Standard deviation	df	$t-$value	$p-$value
0.83	0.29	99	-5.85	3.19×10^{-8}

(iii) The disc robot model performance in the presence of errors (case (c)) is similar to the point robot model performance in the presence of errors (case (b)) in terms of Measure 1. We test this hypothesis as follows. We record the difference between the Measure 1 values of cases (b) and (c) for each of the 100 experiments. We perform the Student's $t-$test with the null hypothesis being that the average of the aforesaid differences is zero against the alternate hypothesis that the difference is not equal to zero at the 95 percent confidence level. The results, summarized in Table IV, show that the null hypothesis cannot be rejected. A comparison of the average Measure 2 values between case (b) and case (c) in Figure 12(bottom) shows that the average number of steps taken by the robots to converge is much greater in the case of the disc robot model.

(iv) The cohesive groups at the end of the experiments are sometimes arranged in chain-like formations as seen in Figure 13(bottom).

TABLE IV
STUDENT'S T-TEST RESULTS FOR COMPARISON OF PERFORMANCE MEASURE 1 OF THE DISC ROBOT MODEL WITH THE POINT ROBOT MODEL

Mean	Standard deviation	df	$t-$value	$p-$value
-0.02	0.82	99	-0.1968	0.84

The above discussion shows that the Perimeter Minimizing Algorithm is robust to various deviations from the ideal scenario in terms of preserving the connected components of the sensing graph and bringing the robots together.

C. Importance of the visibility constraints

Apart from being robust to asynchronism and errors in sensing and control, it is also important to understand how much impact do the visibility constraints have on the performance of the algorithm. By visibility constraints, we refer to the constraints placed on the motion of a robot to ensure that any other robot within its line-of-sight at a certain time instant remains within line-of-sight at the next time instant.

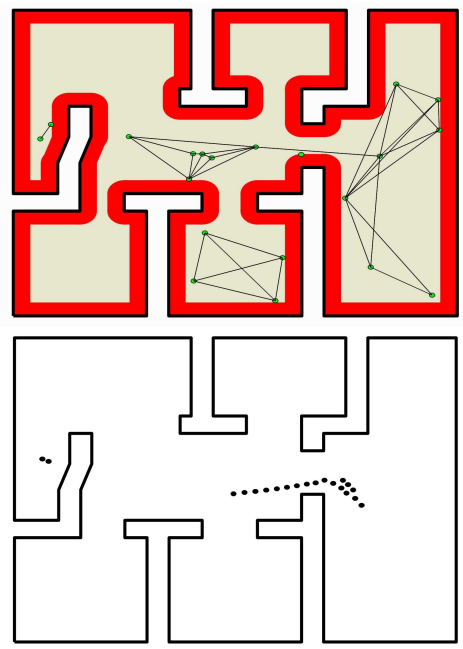

Fig. 13. Computer simulation results taking into account asynchronism, sensing and control errors, and non-zero robot size. The top plot represents the initial condition: green disks represent the robot positions, the graph depicts the sensing graph, $\mathcal{G}_{\text{sens}}$, the portion of the environment interior to the red boundary is Q_ϵ, where $\epsilon = 3$. Initially, $\mathcal{G}_{\text{sens}}$ has four connected components. One of the components is a single robot isolated from others due to the fact that its distance from the boundary is less than ϵ, and hence it cannot be sensed by any other robot. The bottom plot represents the final configuration: black disks represent the final positions of the robots. Note that the robots converge in two groups. The group on the left was disconnected from the other robots in the initial configuration and remained disconnected throughout the evolution.

We performed several executions of the Perimeter Minimizing Algorithm with and without enforcing the visibility constraints for connectivity maintenance. Since it is the visibility constraint that we are interested in, we use a sensing radius of 600, which is much greater than the diameter of the environment. Experiments are performed for a disk robot model with asynchronism and sensing and actuation errors. For each initial condition, we record the ratio of the number of cohesive groups obtained when enforcing the visibility constraints to the number of cohesive groups obtained without enforcing the visibility constraints. We then test the null hypothesis that the mean of the ratios is one against the alternative hypothesis that the mean of the ratio is less than one. As before, we use the Student's $t-$test to test the above null hypothesis at the 95 percent confidence level. The results, reported in Table V, lead us to reject the null hypothesis in favor of the alternate hypothesis. This also follows from the 95 percent confidence interval of the mean ratio which is given by $[0.20, 0.64]$. The 95 percent confidence intervals show that, on average, we can expect the Perimeter Minimizing Algorithm with visibility constraints to result in significantly fewer number of cohesive groups. This illustrates the importance of enforcing the visibility constraints.

TABLE V
STUDENT'S T-TEST RESULTS FOR COMPARISON OF THE PERIMETER MINIMIZING ALGORITHM WITH AND WITHOUT ENFORCING THE VISIBILITY CONSTRAINTS FOR CONNECTIVITY MAINTENANCE.

Mean	Standard deviation	df	$t-$value	$p-$value
0.42	0.94	99	-6.1878	6.92×10^{-9}

D. Near-optimality with respect to distance traveled by robots

We include a comparison of the Perimeter Minimizing Algorithm with the optimal algorithm in terms of the maximum distance by any robot to rendezvous. Given a group of point robots in a nonconvex environment, the optimal rendezvous location is given by $\operatorname{argmin}_{q \in Q} \max_{i \in \{1,\dots,n\}} \mathrm{d}_{\text{geodesic}}(p_i, q)$, where $\mathrm{d}_{\text{geodesic}}(p, q)$ denotes the length of the shortest curve contained in Q that connects p and q. Of course, this location can only be computed centrally with full information about the environment and the robots' location.

Experiments were run for the one hundred randomly generated initial conditions described in Section VI-B for the point robot model with asynchronism and errors in control and sensing. In each case, for every group of robots that converges to a single location, we recorded the optimal rendezvous point, and compute the ratio of the maximum distance that any robot covers until rendezvous under the Perimeter Minimizing Algorithm and the optimal distance to rendezvous. The mean and the standard deviation of the ratio is given by 1.19 and 0.23, respectively. The 95 percent confidence interval of the mean ratio is given by $[1.13, 1.24]$. This shows that, at least for the point robot model, the performance of the (distributed) Perimeter Minimizing Algorithm with asynchronism and sensor and actuator noise is reasonably close to that of the optimal (centralized) algorithm.

E. Computational complexity with finite resolution sensing

An important consideration in the practical implementation of the Perimeter Minimizing Algorithm is the time taken for a robot to complete each step of the algorithm. This is dependent on the computational complexity, that we characterize next.

A visibility sensor, e.g., a range scanner, will sense the position of other robots and the boundary of the environment with some finite resolution; in particular, the boundary of the sensing region will be described by a set of points. It is reasonable to assume that the cardinality of this set of points is bounded, say by M, for all robots, irrespective of the shape of the environment and the location of the robot in it. For example, if a laser range sensor is used to measure the distance to the boundary and a measurement is taken at intervals of one degree, then M is equal to 360.

Proposition VI.1 (Computational complexity) *Let Q be any allowable environment. Let M be the resolution of the visibility sensor located at any robot in Q. Then the following statements are true:*

(i) *the computational complexity of the* Constraint Set Generator Algorithm *is $O(\kappa M)$;*
(ii) *the computational complexity of the* Perimeter Minimizing Algorithm *is $\tau(M) + O(M^3 \log M)$; and*

(iii) *if* $\mathcal{G}_{\text{constr}} = \mathcal{G}_{\text{sens}}$, *then the computational complexity of the* Perimeter Minimizing Algorithm *is* $O(M^2 \log M)$,

where $\tau(M)$ *is time taken for the computation of* $\mathcal{N}_i(\mathcal{G}_{\text{constr}})$ *given the set* $\mathcal{N}_i(\mathcal{G}_{\text{sens}})$ *assuming* $|\mathcal{N}_i(\mathcal{G}_{\text{sens}})| \leq M$, *and* κ *is the number of strict concavities of* Q.

As discussed in Section IV-B, if $\mathcal{G}_{\text{constr}} = \mathcal{G}_{\text{lc}}(\mathcal{G}_{\text{sens}})$, then the computation of $\mathcal{G}_{\text{constr}}$ from $\mathcal{G}_{\text{sens}}$ can be performed using efficient polynomial time heuristics. The time $\tau(M)$ above depends on the specific heuristic used. Thus, the running time of each step of the Perimeter Minimizing Algorithm is polynomial in the number of data points obtained by the visibility sensor. Based on the above analysis, we conclude the algorithm can be implemented on actual robots without demanding an unreasonable computational power.

VII. Conclusions

In this paper, we have presented a provably correct discrete-time synchronous Perimeter Minimizing Algorithm that achieves rendezvous of robots equipped with visibility sensors in a nonconvex environment. The proposed algorithm builds on a novel solution to the connectivity maintenance problem also developed in the paper. Extensive simulations have shown that the performance of the algorithm is satisfactory under asynchronous robot operation, noise in sensing and control, and nontrivial robot dimension. We have also compared the performance of Perimeter Minimizing Algorithm against a similar geometric centering strategy that does not incorporate any connectivity constraint and against the optimal (centralized) strategy in terms of distance traveled by the robots. These comparisons have shown the near-optimality of our (distributed) algorithm in terms of distance traveled, and the superior performance in terms of achievement of the rendezvous objective. Finally, we have characterized the computational complexity of the algorithm under the assumption of finite sensing resolution and found it to be feasible for implementation in real robotic systems.

VIII. Acknowledgments

This material is based upon work supported in part by AFOSR MURI Award F49620-02-1-0325, NSF Award CMS-0626457, NSF Award IIS-0525543, and NSF CAREER Award ECS-0546871. The authors thank Prof. Jana Košecká for an early inspiring discussion and Prof. Seth Hutchinson for his kind support.

References

[1] H. Ando, Y. Oasa, I. Suzuki, and M. Yamashita, "Distributed memoryless point convergence algorithm for mobile robots with limited visibility," *IEEE Transactions on Robotics and Automation*, vol. 15, no. 5, pp. 818–828, 1999.

[2] S. M. LaValle, *Planning Algorithms*. Cambridge University Press, 2006.

[3] "Hokuyo urg-04lx," http://www.hokuyo-aut.jp.

[4] "Swissranger sr-3000," http://www.swissranger.ch.

[5] R. Orghidan, J. Salvi, and E. Mouaddib, "Modelling and accuracy estimation of a new omnidirectional depth computation sensor," *Pattern Recognition Letters*, vol. 27, no. 7, pp. 843–853, 2006.

[6] D. Sack and W. Burgard, "A comparison of methods for line extraction from range data," in *IFAC Symposium on Intelligent Autonomous Vehicles (IAV)*, Lisbon, Portugal, July 2004.

[7] M. Ji and M. Egerstedt, "Distributed control of multiagent systems while preserving connectedness," *IEEE Transactions on Robotics*, vol. 23, no. 4, pp. 693–703, 2007.

[8] Y. U. Cao, A. S. Fukunaga, and A. Kahng, "Cooperative mobile robotics: Antecedents and directions," *Autonomous Robots*, vol. 4, no. 1, pp. 7–27, 1997.

[9] T. Arai, E. Pagello, and L. E. Parker, "Guest editorial: Advances in multirobot systems," *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 655–661, 2002.

[10] I. Suzuki and M. Yamashita, "Distributed anonymous mobile robots: Formation of geometric patterns," *SIAM Journal on Computing*, vol. 28, no. 4, pp. 1347–1363, 1999.

[11] P. Flocchini, G. Prencipe, N. Santoro, and P. Widmayer, "Gathering of asynchronous oblivious robots with limited visibility," *Theoretical Computer Science*, vol. 337, no. 1-3, pp. 147–168, 2005.

[12] J. Lin, A. S. Morse, and B. D. O. Anderson, "The multi-agent rendezvous problem: An extended summary," in *Proceedings of the 2003 Block Island Workshop on Cooperative Control*, ser. Lecture Notes in Control and Information Sciences, V. Kumar, N. E. Leonard, and A. S. Morse, Eds. Springer, 2004, vol. 309, pp. 257–282.

[13] J. Cortés, S. Martínez, and F. Bullo, "Robust rendezvous for mobile autonomous agents via proximity graphs in arbitrary dimensions," *IEEE Transactions on Automatic Control*, vol. 51, no. 8, pp. 1289–1298, 2006.

[14] N. Roy and G. Dudek, "Collaborative exploration and rendezvous: Algorithms, performance bounds, and observations," *Autonomous Robots*, vol. 11, no. 2, pp. 117–136, 2001.

[15] M. Lanthier, D. Nussbaum, and T.-J. Wang, "Calculating the meeting point of scattered robots on weighted terrain surfaces," in *Computing: The Australasian Theory Symposium (CATS)*, vol. 27, Newcastle, Australia, 2005, pp. 107–118.

[16] Z. Lin, M. Broucke, and B. Francis, "Local control strategies for groups of mobile autonomous agents," *IEEE Transactions on Automatic Control*, vol. 49, no. 4, pp. 622–629, 2004.

[17] F. Duguet and G. Drettakis, "Robust epsilon visibility," in *ACM Annual Conference on Computer graphics and Interactive Techniques (SIGGRAPH '02)*, San Antonio, Texas, July 2002, pp. 567–575.

[18] J. W. Jaromczyk and G. T. Toussaint, "Relative neighborhood graphs and their relatives," *Proceedings of the IEEE*, vol. 80, no. 9, pp. 1502–1517, 1992.

[19] A. Ganguli, J. Cortés, and F. Bullo, "Multirobot rendezvous with visibility sensors in nonconvex environments," Nov. 2006, Available electronically at `http://arxiv.org/abs/cs/0611022`.

[20] T. Lozano-Perez, "Spatial planning: A configuration space approach," *IEEE Transactions on Computers*, vol. 32, no. 2, pp. 108–120, 1983.

[21] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo, "The maximum clique problem," in *Handbook of Combinatorial Optimization - Supplement Volume A*, D.-Z. Du and P. M. Pardalos, Eds. Kluwer Academic Publishers, 1999, pp. 1–74.

[22] "Computational geometry algorithmic library," http://www.cgal.org.

[23] "General polygon clipping library," http://www.cs.man.ac.uk/~toby/alan/software/gpc.html.

[24] E. Kreyszig, *Advanced Engineering Mathematics*, 9th ed. John Wiley, 2005.