Article

The International Journal of Robotics Research
2015, Vol. 34(1) 105–128
© The Author(s) 2014
Reprints and permissions:
sagepub.co.uk/journalsPermissions.nav
DOI: 10.1177/0278364914546173
ijr.sagepub.com
SAGE

Decentralized rigidity maintenance control with range measurements for multi-robot systems

Daniel Zelazo[1], Antonio Franchi[2], Heinrich H. Bülthoff[3,4] and Paolo Robuffo Giordano[5]

Abstract

This work proposes a fully decentralized strategy for maintaining the formation rigidity of a multi-robot system using only range measurements, while still allowing the graph topology to change freely over time. In this direction, a first contribution of this work is an extension of rigidity theory to weighted frameworks and the rigidity eigenvalue, which when positive ensures the infinitesimal rigidity of the framework. We then propose a distributed algorithm for estimating a common relative position reference frame amongst a team of robots with only range measurements in addition to one agent endowed with the capability of measuring the bearing to two other agents. This first estimation step is embedded into a subsequent distributed algorithm for estimating the rigidity eigenvalue associated with the weighted framework. The estimate of the rigidity eigenvalue is finally used to generate a local control action for each agent that both maintains the rigidity property and enforces additional constraints such as collision avoidance and sensing/communication range limits and occlusions. As an additional feature of our approach, the communication and sensing links among the robots are also left free to change over time while preserving rigidity of the whole framework. The proposed scheme is then experimentally validated with a robotic testbed consisting of six quadrotor unmanned aerial vehicles operating in a cluttered environment.

Keywords

Graph rigidity, decentralized control, multi-robot, distributed algorithms, distributed estimation

1. Introduction

The coordinated and decentralized control of multi-robot systems is an enabling technology for a variety of applications. Multi-robot systems benefit from an increased robustness against system failures due to their ability to adapt to dynamic and uncertain environments. There are also numerous economic benefits by considering the price of small and cost-effective autonomous systems as opposed to their more expensive monolithic counterparts. Currently, there is a great interest in implementing these systems from deep-space interferometry missions and distributed sensing and data collection, to civilian search and rescue operations, among others (Bristow et al., 2000; Akyildiz et al., 2002; Murray, 2006; Anderson et al., 2008a; Michael et al., 2009; Mesbahi and Egerstedt, 2010; Lindsey et al., 2011).

The challenges associated with the design and implementation of multi-agent systems range from hardware and software considerations to the development of a solid theoretical foundation for their operation. In particular, the *sensing* and *communication* capabilities of each agent will dictate the distributed protocols used to achieve team objectives. For example, if each agent in a multi-robot system is equipped with a GPS-like sensor, then tasks such as formation keeping or localization can be trivially accomplished by communication between robots of their state information in a common world-frame. However, in

[1]Faculty of Aerospace Engineering, Technion - Israel Institute of Technology, Haifa, Israel
[2]Centre National de la Recherche Scientifique (CNRS), Laboratoire d'Analyse et d'Architecture des Systèmes (LAAS), Toulouse, France
[3]Max Planck Institute for Biological Cybernetics, Tübingen, Germany
[4]Department of Brain and Cognitive Engineering, Korea University, Seoul, Korea
[5]CNRS at Irisa and Inria Rennes Bretagne Atlantique, Rennes, France

Corresponding author:
Paolo Robuffo Giordano, CNRS at Irisa and Inria Rennes Bretagne Atlantique, Campus de Beaulieu, 35042 Rennes Cedex, France.
Email: prg@irisa.fr

applications operating in harsher environments, i.e. indoors, underwater, or in deep space, GPS is not a viable sensing option (Scaramuzza et al., 2014). Indeed, in these situations, agents must rely on sensing without knowledge of a common inertial reference frame (Franchi et al., 2012a). In these scenarios, *relative sensing* can provide accurate measurements of, for example, range or bearing, but without any common reference frame.

A further challenge related to the sensing capabilities of multi-robot systems is the availability of these measurements. Sensing constraints such as line-of-sight requirements, range, and power limitations introduce an important system-level requirement, and also lead to an inherently time-varying description of the sensing network. Successful decentralized coordination protocols, therefore, must also be able to manage these constraints.

These issues lead to important architectural requirements for the sensing and communication topology in order to achieve the desired higher-level tasks (i.e. formation keeping or localization). The *connectivity* of the sensing and communication topology is one such property that has received considerable attention in the multi-robot communities (Ji and Egerstedt, 2007; Robuffo Giordano et al., 2011, 2013). However, connectivity alone is not sufficient to perform certain tasks when only relative sensing is used. For these systems, the concept of *rigidity* provides the correct framework for defining an appropriate sensing and communication topology architecture. Rigidity is a combinatorial theory for characterizing the “stiffness” or “flexibility” of structures formed by rigid bodies connected by flexible linkages or hinges.

The study of rigidity has a rich history with contributions from mathematics and engineering disciplines (Laman, 1970; Tay and Whiteley, 1985; Jacobs, 1997; Eren et al., 2004; Connelly and Whiteley, 2009; Krick et al., 2009; Shames et al., 2009). Recently, rigidity theory has taken an outstanding role in the motion control of mobile robots. The rigidity framework allows for applications, such as formation control, to employ control algorithms relying on only *relative distance measurements*, as opposed to *relative position measurements* from a global or relative internal frame (Olfati-Saber and Murray, 2002; Baillieul and McCoy, 2007; Smith et al., 2007; Anderson et al., 2008a,b; Krick et al., 2009). For example, Krick et al. (2009) showed that formation stabilization using only distance measurements can be achieved only if rigidity of the formation is maintained. Moreover, rigidity represents also a necessary condition for estimating relative positions using only relative distance measurements (Aspnes et al., 2006; Calafiore et al., 2010b).

In a broader context, rigidity turns out to be an important architectural property of many multi-agent systems when a common inertial reference frame is unavailable. Applications that rely on sensor fusion for localization, exploration, mapping and cooperative tracking of a target, can all benefit from notions in rigidity theory (Aspnes et al., 2006; Shames et al., 2009; Calafiore et al., 2010a; Wu et al., 2010; Williams et al., 2014). The concept of rigidity, therefore, provides the theoretical foundation for approaching *decentralized* solutions to the aforementioned problems using *distance measurement sensors*, and thus establishing an appropriate framework for relating system-level architectural requirements to the sensing and communication capabilities of the system.

1.1. Main contributions

In general, rigidity as a property of a given formation (i.e. of the robot spatial arrangement) has been studied from either a purely combinatorial perspective (Laman, 1970), or by providing an algebraic characterization via the state-dependent *rigidity matrix* (Tay and Whiteley, 1985). In our previous work (Zelazo et al., 2012), we introduced a related matrix termed the *symmetric rigidity matrix*. A main result of Zelazo et al. (2012) was to provide a necessary and sufficient condition for rigidity in the plane in terms of the positivity of a particular eigenvalue of the symmetric rigidity matrix; this eigenvalue we term the *rigidity eigenvalue*. This result is in the same spirit as the celebrated Fiedler eigenvalue[1] and its relation to the connectivity of a graph (Godsil and Royle, 2001). A first contribution of this work is the extension of the results on the rigidity eigenvalue provided by Zelazo et al. (2012) to three-dimensional frameworks, as well as the introduction of the concept of *weighted rigidity* and the corresponding *weighted rigidity matrix*. This notion allows for the concept of rigidity to include state-dependent weight functions on the edges of the graph, weights which can then be exploited to take into account inter-agent sensing and communication constraints and/or requirements.

A gradient-based *rigidity maintenance action* aimed at ‘maximizing’ the rigidity eigenvalue was also proposed by Zelazo et al. (2012). However, while this gradient control law was decentralized in structure, there was still a dependence on the availability of several global quantities, namely, of the robot relative positions in some *common reference frame*, of the value of the rigidity eigenvalue, and of the *rigidity eigenvector* associated with the rigidity eigenvalue. A main contribution of this work is then the development of the machinery needed to distributedly estimate all of these global quantities by resorting to only *relative distance measurements* among neighbors, so as to ultimately allow for a *fully distributed and range-based* implementation of the rigidity maintenance controller. To this end, we first show that if the formation is infinitesimally rigid, it is possible to *distributedly* estimate the relative positions of neighboring robots in a common reference frame from only range-based measurements. Our approach relies explicitly on the form of the symmetric rigidity matrix developed here, in contrast to other approaches focusing on distributed implementations of centralized estimation schemes, such as a Gauss–Newton approach used by Calafiore et al. (2010a). This first step is then instrumental for the subsequent development of the distributed estimation of the rigidity eigenvalue and eigenvector needed by the rigidity gradient controller. This is obtained by exploiting an appropriate modification of the *power iteration method* for eigenvalue estimation following from the works Yang et al. (2010) and Robuffo Giordano et al. (2011) for the distributed estimation of the connectivity eigenvalue of the graph Laplacian and now applied to rigidity. Finally, we show how to exploit the weights on the graph edges to embed constraints and requirements such as inter-robot and obstacle avoidance, limited communication and sensing ranges, and line-of-sight occlusions, into a unified gradient-based *rigidity maintenance control law*.

Our approach, therefore, can be considered as a contribution to the general problem of distributed strategies for maintaining certain architectural features of a multi-robot system (i.e. connectivity or rigidity) with minimal sensing requirements (only relative distance measurements). In addition, we also provide a thorough experimental validation of the entire framework by employing a group of six quadrotor unmanned aerial vehicles (UAVs) as robotic platforms to demonstrate the feasibility of our approach in real-world conditions.

The organization of this paper is as follows. Section 1.2 provides a brief overview of some notation and fundamental theoretical properties of graphs. In Section 2, the theory of rigidity is introduced, and our extension of the rigidity eigenvalue to three-dimensional weighted frameworks is given. We then proceed to present a general strategy for a distributed rigidity maintenance controller in Section 3. This section will provide details on certain operational constraints of the multi-robot team and how these constraints can be embedded in the control law. This section also highlights the need to develop distributed algorithms for estimating a common reference frame for the team, outlined in Section 4, and estimation of the rigidity eigenvalue and eigenvector, detailed in Section 5. The results of the previous sections are then summarized in Section 6 where the full distributed rigidity maintenance controller is given. The applicability of these results are then experimentally demonstrated on a robotic testbed consisting of six quadrotor UAVs operating in a obstacle populated environment. Details of the experimental setup and results are given in Section 7. Finally, some concluding remarks are offered in Section 8.

1.2. Preliminaries and notation

The notation employed is standard. Matrices are denoted by capital letters (e.g. A), and vectors by lowercase letters (e.g. x). The ij-th entry of a matrix A is denoted $[A]_{ij}$. The rank of a matrix A is denoted $\mathbf{rk}[A]$. Diagonal matrices will be written as $D = \mathbf{diag}\{d_1,\ldots,d_n\}$; this notation will also be employed for block-diagonal matrices. A matrix and/or a vector that consists of all zero entries will be denoted by $\mathbf{0}$; whereas, '0' will simply denote the scalar zero. Similarly, the vector $\mathbb{1}_n$ denotes the $n \times 1$ vector of all ones. The $n \times n$ identity matrix is denoted as I_n. The set of real numbers will be denoted as $\mathbb{R}$, and $\|\cdot\|$ denotes the standard Euclidean 2-norm for vectors. The Kronecker product of two matrices A and B is written as $A \otimes B$ (Horn and Johnson, 1991).

Graphs and the matrices associated with them will be widely used in this work (see, e.g. Godsil and Royle, 2001). An undirected (simple) weighted graph $\mathcal{G}$ is specified by a vertex set $\mathcal{V}$, an edge set $\mathcal{E}$ whose elements characterize the incidence relation between distinct pairs of $\mathcal{V}$, and diagonal $|\mathcal{E}| \times |\mathcal{E}|$ weight matrix W, with $[W]_{kk} \geq 0$ the weight on edge $e_k \in \mathcal{E}$. In this work we consider only finite graphs and denote the cardinality of the node and edge sets as $|\mathcal{V}| = n$ and $|\mathcal{E}| = m$. Two vertices i and j are called *adjacent* (or neighbors) when $\{i,j\} \in \mathcal{E}$. The *neighborhood* of the vertex i is the set $\mathcal{N}_i = \{j \in \mathcal{V} | \{i,j\} \in \mathcal{E}\}$. An *orientation* of an undirected graph $\mathcal{G}$ is the assignment of directions to its edges, i.e. an edge e_k is an ordered pair (i, j) such that i and j are, respectively, the initial and the terminal nodes of e_k.

The incidence matrix $E(\mathcal{G}) \in \mathbb{R}^{n \times m}$ is a $\{0, \pm 1\}$-matrix with rows and columns indexed by the vertices and edges of $\mathcal{G}$ such that $[E(\mathcal{G})]_{ik}$ has the value '$+1$' if node i is the initial node of edge e_k, '-1' if it is the terminal node, and '0' otherwise. The degree of vertex i, d_i, is the cardinality of the set of vertices adjacent to it. The degree matrix, $\Delta(\mathcal{G})$, and the adjacency matrix, $A(\mathcal{G})$, are defined in the usual way (Godsil and Royle, 2001). The (graph) Laplacian of $\mathcal{G}$, $L(\mathcal{G}) = E(\mathcal{G})E(\mathcal{G})^{\mathrm{T}} = \Delta(\mathcal{G}) - A(\mathcal{G})$, is a positive-semidefinite matrix. One of the most important results from algebraic graph theory in the context of collective motion control states that a graph is connected if and only if the second smallest eigenvalue of the Laplacian is positive (Godsil and Royle, 2001).

Table 1 provides a summary of the notations used throughout the document.

2. Rigidity and the rigidity eigenvalue

In this section we review the fundamental concepts of graph rigidity (Graver et al., 1993; Jackson, 2007). A contribution of this work is an extension of our previous results on the concepts of the *symmetric rigidity matrix* and *rigidity eigenvalue* for three-dimensional ambient spaces (Zelazo et al., 2012), and the notion of *weighted frameworks*.

2.1. Graph rigidity and the rigidity matrix

We consider graph rigidity from what is known as a *d-dimensional bar-and-joint framework*. A framework is the pair $(\mathcal{G}, p)$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a graph, and $p : \mathcal{V} \to \mathbb{R}^d$ maps each vertex to a point in $\mathbb{R}^d$. In this work we consider frameworks in a three-dimensional ambient space, i.e. $d = 3$. Therefore, for node $u \in \mathcal{V}$, $p(u) = [p_u^x \quad p_u^y \quad p_u^z]^{\mathrm{T}}$ is the position vector in $\mathbb{R}^3$ for the mapped node. We refer to the matrix $p(\mathcal{V}) = [p(v_1) \quad \cdots \quad p(v_n)]^{\mathrm{T}} \in \mathbb{R}^{n \times 3}$ as the *position matrix*. We now provide some basic definitions.

Definition 2.1. *Frameworks* $(\mathcal{G}, p_0)$ *and* $(\mathcal{G}, p_1)$ *are* equivalent *if* $\|p_0(u)-p_0(v)\| = \|p_1(u) - p_1(v)\|$ *for all* $\{u, v\} \in \mathcal{E}$, *and are* congruent *if* $\|p_0(u)-p_0(v)\| = \|p_1(u)-p_1(v)\|$ *for all* $\{u, v\} \in \mathcal{V}$.

Definition 2.2. *A framework* $(\mathcal{G}, p_0)$ *is* globally rigid *if every framework which is equivalent to* $(\mathcal{G}, p_0)$ *is congruent to* $(\mathcal{G}, p_0)$.

Definition 2.3. *A framework* $(\mathcal{G}, p_0)$ *is* rigid *if there exists an* $\epsilon > 0$ *such that every framework* $(\mathcal{G}, p_1)$ *which is equivalent to* $(\mathcal{G}, p_0)$ *and satisfies* $\|p_0(v) - p_1(v)\| < \epsilon$ *for all* $v \in \mathcal{V}$, *is congruent to* $(\mathcal{G}, p_0)$.

Table 1. Notation.

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	A graph defined by its vertex and edge sets
$\mathcal{N}_i(t)$	*Time-varying* neighborhood of node $v_i \in \mathcal{V}$
$p(i)$	Position vector in $\mathbb{R}^3$ of the mapped node $v_i \in \mathcal{V}$
p_i^s	$s \in \{x, y, z\}$ coordinate of position vector for node i
$p(\mathcal{V})$	Stacked position matrix of all nodes ($\mathbb{R}^{n \times 3}$)
$\xi(i)$	Velocity vector in $\mathbb{R}^3$ of the node $v_i \in \mathcal{V}$
$(\mathcal{G}, p, \mathcal{W})$	A weighted framework
$R(p, \mathcal{W})$	Rigidity matrix of a weighted framework
$\mathcal{R}$	Symmetric rigidity matrix of a weighted framework
λ_7, $\mathbf{v}_7$ $(\mathbf{v})$	Rigidity eigenvalue and eigenvector
ℓ_{ij}	Distance between nodes $v_i, v_j \in \mathcal{V}$, i.e., $\|p(v_i) - p(v_j)\|$
$\hat{\lambda}_7^i$	Agent i's estimate of the rigidity eigenvalue
$\hat{\mathbf{v}}_i^s$	s coordinate of the agent i estimation of the rigidity eigenvector
$\hat{p}_{i,c}$	Agent i estimate of relative position vector $p_i - p_c$
$\hat{p}$	Stacked vector of the relative position vector estimate $p_i - p_c$, $i = 1 \ldots n$
$\mathbf{avg}\ (x)$	The average of a vector $x \in \mathbb{R}^n$, $\mathbf{avg}(x) = \frac{1}{n}\sum_{i=1}^n x_i$
$\bar{\mathbf{v}}_i^x$	Agent i estimate of $\mathbf{avg}(\hat{\mathbf{v}}^x)$
$\bar{\mathbf{v}}_i^{2x}$	Agent i estimate of $\mathbf{avg}(\hat{\mathbf{v}}^x \circ \hat{\mathbf{v}}^x)$
z_i^{xy}	Agent i estimate of $\mathbf{avg}(\hat{p}^{y,c} \circ \hat{\mathbf{v}}^x - \hat{p}^{x,c} \circ \hat{\mathbf{v}}^y)$
z_i^{xz}	Agent i estimate of $\mathbf{avg}(\hat{p}^{z,c} \circ \hat{\mathbf{v}}^x - \hat{p}^{x,c} \circ \hat{\mathbf{v}}^z)$
z_i^{yz}	Agent i estimate of $\mathbf{avg}(\hat{p}^{y,c} \circ \hat{\mathbf{v}}^z - \hat{p}^{z,c} \circ \hat{\mathbf{v}}^y)$

Definition 2.4. *A* minimally rigid graph *is a rigid graph such that the removal of any edge results in a non-rigid graph.*

Figure 1 shows three frameworks illustrating the above definitions. The frameworks in Figure 1(a) are both minimally rigid and are equivalent to each other, but are not congruent, and therefore not globally rigid. By adding an additional edge, as in Figure 1(b) (the edge $\{v_4, v_5\}$), the framework becomes globally rigid. The key feature of global rigidity, therefore, is that the distances between *all* node pairs are maintained for different framework realizations, and not just those defined by the edge set.

By parameterizing the position map by a positive scalar representing time, we can also consider *trajectories* of a framework. That is, the position map now becomes $p : \mathcal{V} \times \mathbb{R} \to \mathbb{R}^3$ and is assumed to be continuously differentiable with respect to time. We then explicitly write $(\mathcal{G}, p, t)$ so as to represent a *time-varying framework*. In this direction, we can define a set of trajectories that are *edge-length preserving*, in the sense that for each time $t \geq t_0$, the framework $(\mathcal{G}, p, t)$ is equivalent to the framework $(\mathcal{G}, p, t_0)$. More formally, an edge-length preserving framework must satisfy the constraint

$$||p(v, t) - p(u, t)|| = ||p(v, t_0) - p(u, t_0)|| = \ell_{vu}, \quad \text{for all } t \geq t_0 \tag{1}$$

and for all $\{v, u\} \in \mathcal{E}$.

One can similarly assign velocity vectors $\xi(u, t) \in \mathbb{R}^3$ to each vertex $u \in \mathcal{V}$ for each point in the configuration space such that

$$(\xi(u, t) - \xi(v, t))^{\mathrm{T}}(p(u, t) - p(v, t)) = 0, \text{ for all } \{u, v\} \in \mathcal{E} \tag{2}$$

Note that this relation can be obtained by time-differentiation of the length constraint described in (1). These motions are referred to as *infinitesimal motions* of the mapped vertices $p(u, t)$, and one has

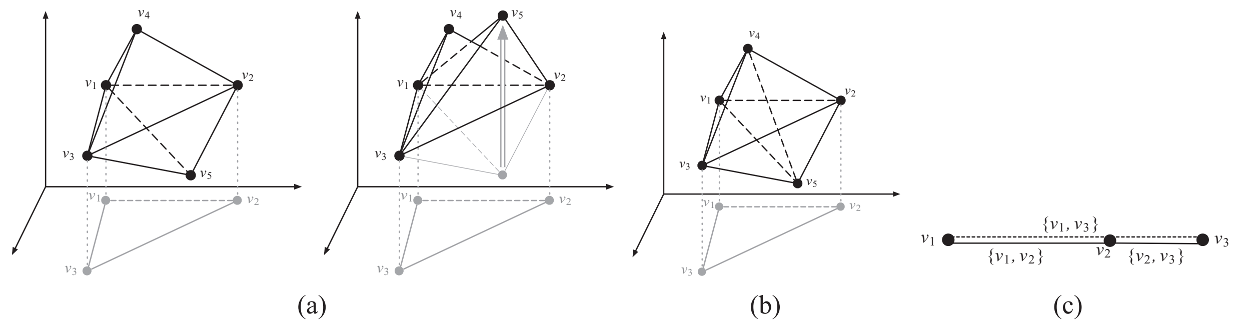

Fig. 1. Examples of rigid and infinitesimally rigid frameworks in $\mathbb{R}^3$. (a) Two equivalent minimally rigid frameworks in $\mathbb{R}^3$. The framework on the right-hand side is obtained by the reflection of the position of v_5 with respect to the plane characterized by the positions of v_1, v_2, and v_3 (as illustrated in gray). (b) An infinitesimally and globally rigid framework in $\mathbb{R}^3$. (c) A non-infinitesimally rigid framework (note that vertexes v_1 and v_3 are connected). Note that in (a) and (b) the 3D points associated with each vertex do not lie on the same plane, while in (c) the 3D points are aligned.

$$\dot{p}(u,t) = \xi(u,t) \tag{3}$$

For the remainder of this paper, we drop the explicit inclusion of time for frameworks and simply write $(\mathcal{G}, p)$ and $p(u)$ and $\xi(u)$ for the time-varying positions and velocities. The velocity vector $\xi(u)$ will be treated as the *agent velocity input* throughout the rest of the paper (see Section 3).

Infinitesimal motions of a framework can be used to define a stronger notion of rigidity.

Definition 2.5. *A framework is called* infinitesimally rigid *if every possible motion that satisfies (2) is trivial (i.e. consists of only global rotations and translations of the whole set of points in the framework).*

An example of an infinitesimally rigid graph in $\mathbb{R}^3$ is shown in Figure 1(b). Furthermore, note that infinitesimal rigidity implies rigidity, but the converse is not true (Tay and Whiteley, 1985), see Figure 1(c) for a rigid graph in $\mathbb{R}^3$ that is not infinitesimally rigid.

The infinitesimal motions in (2) define a system of m linear equations in the vector of unknown velocities $\xi = [\xi^T(v_1)\dots\xi^T(v_n)]^T \in \mathbb{R}^{3n}$. This system can be equivalently written as the linear matrix equation

$$R(p)\xi = \mathbf{0}$$

where $R(p) \in \mathbb{R}^{m\times 3n}$ is called *rigidity matrix* (Tay and Whiteley, 1985). Each row of $R(p)$ corresponds to an edge $e = \{u, v\}$ and the quantity $(p(u) - p(v))$ represents the non-zero coefficients for that row. For example, the row corresponding to edge e has the form

$$\begin{bmatrix} -\mathbf{0}- & \underbrace{(p(u)-p(v))^T}_{\text{vertex } u} & -\mathbf{0}- & \underbrace{(p(v)-p(u))^T}_{\text{vertex } v} & -\mathbf{0}- \end{bmatrix}$$

The definition of infinitesimal rigidity can then be restated in the following form:

Lemma 2.6 (Tay and Whiteley, 1985). *A framework $(\mathcal{G}, p)$ in $\mathbb{R}^3$ is infinitesimally rigid if and only if* $\mathbf{rk}[R(p)] = 3n - 6$.

Note that, as expected from Definition 2.5, the six-dimensional kernel of $R(p)$ for an infinitesimally rigid graph only allows for six independent *feasible* framework motions, that is, the above-mentioned collective roto-translations in $\mathbb{R}^3$ space. Note also that, despite its name, the rigidity matrix is actually characterizing *infinitesimal rigidity* rather than *rigidity* of a framework.

2.2. Rigidity of weighted frameworks

We now introduce an important generalization to the concept of rigidity and the rigidity matrix by introducing weights to the framework. Indeed, as discussed in the introduction, our aim is to propose a control law able to not only maintain *infinitesimal rigidity* of the formation as per Definition 2.5, but to also concurrently manage additional constraints typical of multi-robot applications such as collision avoidance and limited sensing and communication. This latter objective will be accomplished via the introduction of suitable state-dependent weights, thus requiring an extension of the traditional results on rigidity to a weighted case.

Definition 2.7. *A d-dimensional* weighted *framework is the triple $(\mathcal{G}, p, \mathcal{W})$, where $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a graph, $p : \mathcal{V} \to \mathbb{R}^d$ is a function mapping each vertex to a point in $\mathbb{R}^d$,* and *$\mathcal{W} : (\mathcal{G}, p) \to \mathbb{R}^m$ is a function of the framework that assigns a scalar value to each edge in the graph.*

Using this definition, we can also define the corresponding *weighted rigidity matrix*, $R(p, \mathcal{W})$, as

$$R(p, \mathcal{W}) = W(\mathcal{G}, p)R(p) \tag{4}$$

where $W(\mathcal{G}, p) \in \mathbb{R}^{m\times m}$ is a diagonal matrix containing the elements of the vector $\mathcal{W}(\mathcal{G}, p)$ on the diagonal. Often we will simply refer to the weight matrix $W(\mathcal{G}, p)$ as W when the underlying graph and map p is understood.

Remark 2.8. *Note that the rigidity matrix R(p) can also be considered as a weighted rigidity matrix with $W(\mathcal{G}, p) = I$. Another useful observation is that the unweighted framework $(\mathcal{G}, p)$ can also be cast as a weighted framework $(K_n, p, \mathcal{W})$, where K_n is the complete graph on n nodes and $[W(\mathcal{G}, p)]_{ii}$ is 1 whenever $e_i \in \mathcal{E}(K_n)$ is also an edge in $\mathcal{G}$, and 0 otherwise.*

Weighted rigidity can lead to a slightly different interpretation of infinitesimal rigidity, where the introduced weights might cause the rigidity matrix to lose rank. That is, an unweighted framework might be infinitesimally rigid, whereas a weighted version might not. This observation is trivially observed by considering a minimally infinitesimally rigid framework $(\mathcal{G}, p)$ and introducing a weight with a 0 entry on any edge. We formalize this with the following definitions.

Definition 2.9. *The* unweighted counterpart *of a weighted framework $(\mathcal{G}, p, \mathcal{W})$ is the framework $(\widehat{\mathcal{G}}, p)$ where the graph $\widehat{\mathcal{G}} = (\mathcal{V}, \widehat{\mathcal{E}})$ is such that $\widehat{\mathcal{E}} \subset \mathcal{E}$ and the edge $e_i \in \mathcal{E}$ is also an edge in $\widehat{\mathcal{G}}$ if and only if the corresponding weight is non-zero (i.e. $[W(\mathcal{G}, p)]_{ii} \neq 0$).*

Definition 2.10. *A weighted framework is called infinitesimally rigid if its unweighted counterpart is infinitesimally rigid.*

We now present a corollary to Lemma 2.6 for weighted frameworks.

Corollary 2.11. *A weighted framework $(\mathcal{G}, p, \mathcal{W})$ in $\mathbb{R}^3$ is infinitesimally rigid if and only if* $\mathbf{rk}[R(p, \mathcal{W})] = 3n - 6$.

Proof. The statement follows from the fact that $\mathbf{rk}[R(p, \mathcal{W})] = \mathbf{rk}[\hat{R}(p)]$, where $\hat{R}(p)$ is the rigidity matrix for the unweighted counterpart of $(\mathcal{G}, p, \mathcal{W})$.

2.3. The rigidity eigenvalue

In our previous work (Zelazo et al., 2012), we introduced an alternative representation of the rigidity matrix that transparently separates the underlying graph from the positions of each vertex. Here we recall the presentation and extend it to the case of three-dimensional frameworks.

Definition 2.12 (Zelazo et al. (2012)). *Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and its associated incidence matrix with arbitrary orientation $E(\mathcal{G})$. The* directed local graph *at node v_j is the sub-graph $\mathcal{G}_j = (\mathcal{V}, \mathcal{E}_j)$ induced by node v_j such that*

$$\mathcal{E}_j = \{(v_j, v_i) | e_k = \{v_i, v_j\} \in \mathcal{E}\}$$

The local incidence matrix *at node v_j is the matrix*

$$E_l(\mathcal{G}_j) = E(\mathcal{G})\mathbf{diag}\{s_1, \ldots, s_m\} \in \mathbb{R}^{n \times m}$$

where $s_k = 1$ if $e_k \in \mathcal{E}_j$ and $s_k = 0$ otherwise.

Note, therefore, that the local incidence matrix will contain columns of all zeros in correspondence to those edges not adjacent to v_j. This also implicitly assumes a predetermined labeling of the edges.

Proposition 2.13 (Zelazo et al., 2012). *Let $p(\mathcal{V}) \in \mathbb{R}^{n \times 3}$ be the position matrix for the framework $(\mathcal{G}, p)$. The rigidity matrix $R(p)$ can be defined as*

$$R(p) = \begin{bmatrix} E_l(\mathcal{G}_1)^{\mathrm{T}} & \cdots & E_l(\mathcal{G}_n)^{\mathrm{T}} \end{bmatrix} (I_n \otimes p(\mathcal{V})), \qquad (5)$$

where $E_l(\mathcal{G}_i)$ is the local incidence matrix for node v_i.

A more detailed discussion and examples of these definitions are provided in Appendix B.

Lemma 2.6 and Corollary 2.11 relate the property of infinitesimal rigidity for a given (weighted) framework to the rank of a corresponding matrix. A contribution of this work is the translation of the rank condition to that of a condition on the spectrum of a corresponding matrix that we term the *symmetric rigidity matrix*. For the remainder of this work, we will only consider weighted frameworks, since from the discussion in Remark 2.8, any framework can be considered as a weighted framework with appropriately defined weights.

The symmetric rigidity matrix for a weighted framework $(\mathcal{G}, p, \mathcal{W})$ is a symmetric and positive-semidefinite matrix defined as

$$\mathcal{R} : = R(p, \mathcal{W})^{\mathrm{T}} R(p, \mathcal{W}) \in \mathbb{R}^{3n \times 3n} \qquad (6)$$

An immediate consequence of the construction of the symmetric rigidity matrix is that $\mathbf{rk}[\mathcal{R}] = \mathbf{rk}[R(p, \mathcal{W})]$ (Horn and Johnson, 1985), leading to the following corollary.

Corollary 2.14. *A weighted framework $(\mathcal{G}, p, \mathcal{W})$ is infinitesimally rigid if and only if $\mathbf{rk}[\mathcal{R}] = 3n - 6$.*

The rank condition of Corollary 2.14 can be equivalently stated in terms of the eigenvalues of $\mathcal{R}$. Denoting the eigenvalues of $\mathcal{R}$ as $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_{3n}$, note that infinitesimal rigidity is equivalent to $\lambda_i = 0$ for $i = 1,\ldots, 6$ and $\lambda_7 > 0$. Consequently, we term λ_7 the *rigidity eigenvalue*. We will now show that, in fact, for any connected graph,[2] the first six eigenvalues are always 0.

The first result in this direction shows that the symmetric rigidity matrix is similar to a weighted Laplacian matrix.

Proposition 2.15. *The symmetric rigidity matrix is similar to the weighted Laplacian matrix via a permutation of the rows and columns as*

$$P\mathcal{R}P^{\mathrm{T}} = (I_3 \otimes E(\mathcal{G})W) Q(p(\mathcal{V})) \big(I_3 \otimes WE(\mathcal{G})^{\mathrm{T}}\big) \qquad (7)$$

with

$$Q(p(\mathcal{V})) = \begin{bmatrix} Q_x^2 & Q_x Q_y & Q_x Q_z \\ Q_y Q_x & Q_y^2 & Q_y Q_z \\ Q_z Q_x & Q_z Q_y & Q_z^2 \end{bmatrix} \in \mathbb{R}^{3m \times 3m} \qquad (8)$$

where Q_x, Q_y, and Q_z are $m \times m$ diagonal weighting matrices for each edge in $\mathcal{G}$ such that for the edge $e_k = (v_i, v_j)$,

$$[Q_s]_{kk} = (p_i^s - p_j^s), \quad s \in \{x, y, z\}$$

and p_i^x (p_i^y,p_i^z) represents the x coordinate (y coordinate, z coordinate) of the position of agent i.

Proof. The proof is by direct construction using Proposition 2.13 and (6). Consider the permutation matrix P as

$$P = \begin{bmatrix} I_n \otimes [1 \quad 0 \quad 0] \\ I_n \otimes [0 \quad 1 \quad 0] \\ I_n \otimes [0 \quad 0 \quad 1] \end{bmatrix} \qquad (9)$$

and let $\hat{E} = \begin{bmatrix} E_l(\mathcal{G}_1)^{\mathrm{T}} & \cdots & E_l(\mathcal{G}_n)^{\mathrm{T}} \end{bmatrix}$. It is straightforward to verify that

$$(I_n \otimes (p^x)^{\mathrm{T}}) \hat{E}^{\mathrm{T}} W = E(\mathcal{G}) W \underbrace{\begin{bmatrix} \ddots & & \\ & (p_i^x - p_j^x) & \\ & & \ddots \end{bmatrix}}_{\text{diagonal matrix of size } m \times m}$$

where p^x represents the first column of the position vector. The structure of the matrix in (7) then follows directly.[3]

The representation of the symmetric rigidity matrix as a weighted Laplacian allows for a more transparent understanding of certain eigenvalues related to this matrix. The next result shows that the first six eigenvalues of $\mathcal{R}$ must equal zero for any connected graph $\mathcal{G}$.

Theorem 2.16. *Assume that a weighted framework $(\mathcal{G}, p, \mathcal{W})$ has weights such that the weight matrix $W(\mathcal{G}, p)$ is invertible and the underlying graph $\mathcal{G}$ is connected. Then the symmetric rigidity matrix has at least six eigenvalues at the origin; that is, $\lambda_i = 0$ for $i \in \{1,\ldots, 6\}$. Furthermore, a possible set of linearly independent eigenvectors associated with each 0 eigenvalue is*

$$\left\{ P^{\mathrm{T}}\begin{bmatrix}\mathbb{1}_n\\ \mathbf{0}\\ \mathbf{0}\end{bmatrix}, P^{\mathrm{T}}\begin{bmatrix}\mathbf{0}\\ \mathbb{1}_n\\ \mathbf{0}\end{bmatrix}, P^{\mathrm{T}}\begin{bmatrix}\mathbf{0}\\ \mathbf{0}\\ \mathbb{1}_n\end{bmatrix}, \right.$$
$$\left. P^{\mathrm{T}}\begin{bmatrix}(p^y)\\ -(p^x)\\ \mathbf{0}\end{bmatrix}, P^{\mathrm{T}}\begin{bmatrix}(p^z)\\ \mathbf{0}\\ -(p^x)\end{bmatrix}, P^{\mathrm{T}}\begin{bmatrix}\mathbf{0}\\ (p^z)\\ -(p^y)\end{bmatrix}\right\},$$

where P is defined in (9).

Proof. Recall that for any connected graph, one has $E(\mathcal{G})^{\mathrm{T}}\mathbb{1}_n = 0$ (Godsil and Royle, 2001). Therefore, $P\mathcal{R}P^{\mathrm{T}}$ must have three eigenvalues at the origin, with eigenvectors $u_1 = \begin{bmatrix}\mathbb{1}_n^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$, $u_2 = \begin{bmatrix}\mathbf{0}^{\mathrm{T}} & \mathbb{1}_n^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$, and $u_3 = \begin{bmatrix}\mathbf{0}^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}} & \mathbb{1}_n^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$. We now demonstrate that the remaining three eigenvectors proposed in the theorem are indeed in the null-space of the symmetric rigidity matrix.

Let $u_4 = \begin{bmatrix}(p^y)^{\mathrm{T}} & -(p^x)^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$. Observe that $(I_3 \otimes WE(\mathcal{G})^{\mathrm{T}})u_4 = \begin{bmatrix}b_1^{\mathrm{T}} & b_2^{\mathrm{T}} & \mathbf{0}^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$ is such that b_1 is $\pm[W]_{kk}(p_i^y - p_j^y)$ only for edges $e_k = \{v_i, v_j\} \in \mathcal{E}$, and 0 otherwise. Similarly, b_2 is $\pm[W]_{kk}(p_j^x - p_i^x)$ only for edges $e_k = \{v_i, v_j\} \in \mathcal{E}$. The invertibility assumption of the weight matrix also guarantees that $[W]_{kk} \neq 0$. It can now be verified that from this construction one has

$$\begin{bmatrix} Q_x^2 & Q_xQ_y & Q_xQ_z\\ Q_yQ_x & Q_y^2 & Q_yQ_z\\ Q_zQ_x & Q_zQ_y & Q_z^2\end{bmatrix}(I_3 \otimes WE(\mathcal{G})^{\mathrm{T}})u_4 = 0.$$

The remaining two eigenvectors follow the same argument as above. It is also straightforward to verify that u_4, u_5, and u_6 are linearly independent of the first three eigenvectors.

Theorem 2.16 provides a precise characterization of the eigenvectors associated with the null-space of the symmetric rigidity matrix for an infinitesimally rigid framework.

Remark 2.17. *It is important to note that the chosen eigenvectors associated with the null-space of the symmetric rigidity matrix are expressed in terms of the* absolute positions *of the nodes in the framework. We note that these eigenvectors can also be expressed in terms of the* relative position *of each node to any arbitrary reference point* $p_c = [p_c^x \ \ p_c^y \ \ p_c^z]^{\mathrm{T}} \in \mathbb{R}^3$. *For example, vector* u_4 *could be replaced by*

$$u_4^{p_c} = P^{\mathrm{T}}\begin{bmatrix}p^y - p_c^y\mathbb{1}_n\\ p_c^x\mathbb{1}_n - p^x\\ \mathbf{0}\end{bmatrix}$$

that is a linear combination of the null-space eigenvectors u_1,u_2 *and* u_4. *The use of eigenvectors defined on relative positions, in fact, will be necessary for the implementation of a distributed estimator for the rigidity eigenvector and eigenvalue based on only* relative measurements *available from onboard sensing.*

Theorem 2.16 can be used to arrive at the main result relating infinitesimal rigidity to the rigidity eigenvalue.

Theorem 2.18. *A weighted framework* $(\mathcal{G}, p, \mathcal{W})$ *is infinitesimally rigid if and only if the rigidity eigenvalue is strictly positive, i.e.* $\lambda_7 > 0$.

Proof. The proof is a direct consequence of Corollary 2.14 and Theorem 2.16.

Another useful observation relates infinitesimal rigidity of a framework to connectedness of the underlying graph.

Corollary 2.19. *Rigidity of the weighted framework* $(\mathcal{G}, p, \mathcal{W})$ *implies connectedness of the graph* $\mathcal{G}$.

The connection between infinitesimal rigidity of a framework and the spectral properties of the symmetric rigidity matrix inherits many similarities between the well studied relationship between graph connectivity and the graph Laplacian matrix (Mesbahi and Egerstedt, 2010).

In the next section, we exploit this similarity and propose a *rigidity maintenance* control law that aims to ensure the rigidity eigenvalue is always positive. Such a control action will be shown to depend on the rigidity eigenvalue, on its eigenvector, and on relative positions among neighboring pairs expressed in a common frame. The issue of how every agent in the group can distributedly estimate these quantities will be addressed in Sections 4 and 5.

3. A decentralized control strategy for rigidity maintenance

The results of Section 2 highlight the role of the rigidity eigenvalue λ_7 as a measure of the "degree of infinitesimal rigidity" of a weighted framework $(\mathcal{G}, p, \mathcal{W})$. It provides a linear algebraic condition to test the infinitesimal rigidity of a framework and, especially in the case of weighted frameworks, provides a means of quantifying "how rigid" a weighted framework is. Moreover, the symmetric rigidity matrix was shown to have a structure reminiscent of a weighted graph Laplacian matrix, and thus can be considered as a naturally *distributed* operator.

The basic approach we consider for the maintenance of rigidity is to define a scalar potential function of the rigidity eigenvalue, $V_\lambda(\lambda_7) > 0$, with the properties of growing unbounded as $\lambda_7 \to \lambda_7^{\min} > 0$ and vanishing (with vanishing derivative) as $\lambda_7 \to \infty$ (see Figure 2 for one possible shape or V_λ with $\lambda_7^{\min} = 5$). Here, $\lambda_7^{\min}$ represents some predetermined minimum allowable value for the rigidity eigenvalue determined by the needs of the application. In addition to maintaining rigidity, the potential function should also capture additional constraints in the system, such as collision avoidance or formation maintenance. Each agent should then follow the anti-gradient of this potential function, that is

$$\xi(u) = \dot{p}_u(t) = -\frac{\partial V_\lambda}{\partial p_u(t)} = -\frac{\partial V_\lambda}{\partial \lambda_7}\frac{\partial \lambda_7}{\partial p_u(t)} \qquad (10)$$

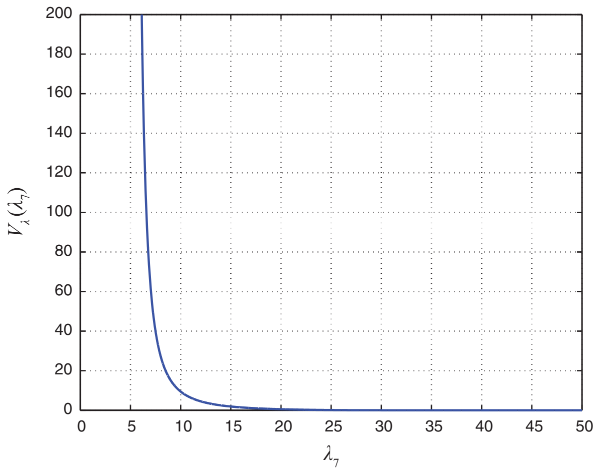

Fig. 2. A possible shape for the rigidity potential function $V_\lambda(\lambda_7)$ with $\lambda_7^{\min} = 5$.

where $\xi(u)$ is the velocity input of agent u, as defined in (3), and $p_u = [p_u^x \quad p_u^y \quad p_u^z]^T$ is the position vector of the u-th agent. This strategy will ensure that the formation maintains a "minimum" level of rigidity (i.e. $\lambda_7^{\min}$) at all times. Of course, this strategy is an inherently *centralized* one, as the computation of the rigidity eigenvalue and of its gradient require full knowledge of the symmetric rigidity matrix. Nevertheless, we will proceed with this strategy and demonstrate that it can be implemented in a fully decentralized manner.

In the following, we examine in more detail the structure of the control scheme (10). First, we show how the formalization of weighted frameworks allows to embed additional weights within the rigidity property that enforce explicit inter-agent sensing and communication constraints and group requirements such as collision avoidance and formation control. For instance, the weighting machinery will be exploited so as to induce the agents to keep a desired inter-agent distance ℓ_0 and to ensure a minimum safety distance $\ell_{\min}$ from neighboring agents and obstacles. With these constraints, the controller will simultaneously maintain a minimum level of rigidity while also respecting the additional inter-agent constraints. We then provide an explicit characterization of the gradient of the rigidity eigenvalue with respect to the agent positions, and highlight its distributed structure. Finally, we present the general control architecture for implementing (10) in a fully decentralized way.

3.1. Embedding constraints in a weighted framework

In real-world applications a team of mobile robots may not be able to maintain the same interaction graph throughout the duration of a mission because of various sensing and communication constraints preventing mutual information exchange and relative sensing. Furthermore, additional requirements such as collision avoidance with obstacles and among robots, as well as some degree of formation control, must be typically satisfied during the mission execution. Building on the design guidelines proposed in Robuffo Giordano et al. (2013) for dealing with *connectivity* maintenance, we briefly discuss here a possible design of weights $\mathcal{W}$ aimed at taking into account the above-mentioned sensing and communication constraints and group requirements within the rigidity maintenance action.

To this end, we start with the following definition of *neighboring agents*:

Definition 3.1. *Two agents u and v are considered* neighbors *if and only if* (*i*) *their relative distance* $\ell_{uv} = \|p(u) - p(v)\|$ *is smaller than* $D \in \mathbb{R}^+$ (the sensing range), (*ii*) *the distance* ℓ_{uvo} *between the segment joining u and v and the closest obstacle point o is larger than* $\ell_{\min}$ (*the minimum* line-of-sight visibility), *and (iii) neither u nor v are closer than* $\ell_{\min}$ *to any other agent or obstacle.*

Conditions (i) and (ii) are meant to take into account two typical sensing constraints in multi-robot applications: maximum communication and sensing ranges and line-of-sight occlusions. The purpose of condition (iii), which will be better detailed later on, is to force disconnection from the group if an agent is colliding with any other agent or obstacle in the environment. In the following we will denote with $\mathcal{S}_u$ the set of neighbors of agent u induced by Definition 3.1.

This neighboring definition can be conveniently taken into account by designing the inter-agent weights $\mathcal{W}_{uv}$ as state-dependent functions smoothly vanishing as any of the above constraints and requirements are not met by the pair (u, v) with the desired accuracy. Indeed, the use of state-dependent weights allows us to consider the ensemble of robots in the context of weighted frameworks, as introduced in Definition 2.7. In particular, we take the underlying graph to be the complete graph K_n and the map p corresponds to the physical position state of each agent in a common global frame. The weights are the maps $\mathcal{W}_{uv}$, and the weighted framework is the triple $(K_n, p, \mathcal{W})$ with, therefore, $\mathcal{N}_u = \{v \in \mathcal{V} | \mathcal{W}_{uv} \neq 0\}$.

Following what was proposed in Robuffo Giordano et al. (2013), and recalling that ℓ_{uvo} represents the distance between the segment joining agents u and v and the closest obstacle point o, we then take

$$\mathcal{W}_{uv} = \alpha_{uv}\beta_{uv}\gamma_{uv}^a\gamma_{uv}^b \tag{11}$$

with $\alpha_{uv} = \alpha_{uv}(\ell_{uk}|_{k\in\mathcal{S}_u}, \ell_{vk}|_{k\in\mathcal{S}_v})$, $\beta_{uv} = \beta_{uv}(\ell_{uv})$, $\gamma_{uv}^a = \gamma_{uv}^a(\ell_{uv})$, $\gamma_{uv}^b = \gamma_{uv}^b(\ell_{uvo})$ and such that:

- we have
  - $\lim_{\ell_{uk}\to\ell_{\min}} \alpha_{uv} = 0$, for all $k \in \mathcal{S}_u$,
  - $\lim_{\ell_{vk}\to\ell_{\min}} \alpha_{uv} = 0$, for all $k \in \mathcal{S}_v$, and
  - $\alpha_{uv} \equiv 0$ if $\ell_{uk} \leq \ell_{\min}$ or $\ell_{vk} \leq \ell_{\min}$, for any $k \in \mathcal{S}_u$, $k \in \mathcal{S}_v$;
- $\lim_{|\ell_{uv}-\ell_0|\to\infty} \beta_{uv} = 0$ with $\beta(\ell_{uv}) < \beta(\ell_0)$ for all $\ell_{uv} \neq \ell_0$;

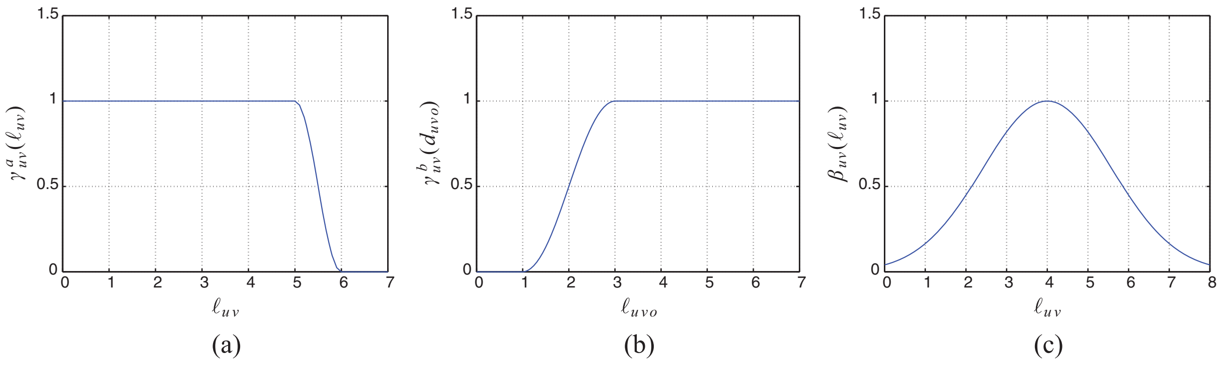

Fig. 3. The shape of $\gamma^a_{uv}(\ell_{uv})$ for $D = 6$ (a), $\gamma^b_{uv}(\ell_{uvo})$ for $\ell_{\min} = 1$ (b), and $\beta_{uv}(\ell_{uv})$ for $\ell_0 = 4$ (c).

- $\lim_{\ell_{uv}\to D} \gamma^a_{uv} = 0$ with $\gamma^a_{uv}\equiv 0$ for all $\ell_{uv} \geq D$;
- $\lim_{\ell_{uvo}\to \ell_{\min}} \gamma^b_{uv} = 0$ with $\gamma^b_{uv}\equiv 0$ for all $\ell_{uvo} \leq \ell_{\min}$.

As explained, $\ell_{\min}$ is a predetermined minimum safety distance for avoiding collisions and line-of-sight occlusions. Figures 3(a)–(c) show an illustrative shape of weights γ^a_{uv}, γ^b_{uv} and β_{uv}. The shape of the weights α_{uv} is conceptually equivalent to that of weights γ^b_{uv} in Figure 3(b).

This weight design results in the following properties: for a given pair of agents (u, v), the weight $\mathcal{W}_{uv}$ will vanish (because of the term $\gamma^a_{uv}\gamma^b_{uv}$) whenever the sensing and communication constraints of Definition 3.1 are violated (maximum range, obstacle occlusion), thus resulting in a decreased degree of connectivity of the graph $\mathcal{G}$ (edge $\{u, v\}$ is lost). The same will happen as the inter-distance ℓ_{uv} deviates too much from the desired ℓ_0 because of the term β_{uv}. Finally, the term α_{uv} will force *complete disconnection* of vertexes u and v from the other vertexes and therefore a complete loss of connectivity for the graph $\mathcal{G}$ whenever a collision with another agent is approached.[4]

We now recall from Corollary 2.19 that infinitesimal rigidity implies graph connectivity. Therefore, any decrease in the degree of graph connectivity due to the weights $\mathcal{W}_{uv}$ vanishing will also result in a decrease of rigidity of the weighted framework $(K_n, p, \mathcal{W})$ (in particular, rigidity is obviously lost for a disconnected graph). By maintaining $\lambda_7 > 0$ (in the context of weighted frameworks) over time, it is then possible to preserve formation rigidity while, at the same time, explicitly considering and managing the above-mentioned sensing and communication constraints and requirements.

Remark 3.2. *We note that the purpose of the weight β_{uv} in (11) is to embed a basic level of formation control into the rigidity maintenance action: indeed, every neighboring pair will try to keep the desired distance ℓ_0 thanks to the shape of the weights β_{uv}. More complex formation control behaviors could be obtained by different choices of functions β_{uv} (e.g. for maintaining given relative positions). Furthermore, formation shapes can be uniquely specified owing to the infinitesimal rigidity property of the configuration.*

Remark 3.3. *We further highlight the following properties whose explicit proof can be found in Robuffo Giordano et al. (2013): the chosen weights $\mathcal{W}_{uv}$ are functions of only relative distances to other agents and obstacles, while their gradients with respect to the agent position p_u (respectively p_v) are functions of* relative positions *expressed in a common reference frame. Furthermore, $\mathcal{W}_{uv} = \mathcal{W}_{vu}$ and $\dfrac{\partial \mathcal{W}_{uv}}{\partial p_u} = 0$, for all $v \notin \mathcal{N}_u$. Finally, the evaluation of weights $\mathcal{W}_{uv}$ and of their gradients can be performed in a decentralized way by agent u (respectively v) by only resorting to local information and* 1 *-hop communication.*

As shown in the next developments, these properties will be instrumental for expressing the gradient of the rigidity eigenvalue as a function of purely relative quantities with respect to only 1-hop neighbors.

3.2. The gradient of the rigidity eigenvalue

We now present an explicit characterization of the gradient of the rigidity eigenvalue with respect to the agent positions, as used in the control (10). We first recall that the rigidity eigenvalue can be expressed as

$$\lambda_7 = \mathbf{v}_7^{\mathrm{T}} \mathcal{R} \mathbf{v}_7$$

where $\mathbf{v}_7$ is the *normalized* rigidity eigenvector associated with λ_7. For notational convenience, we consider the permuted rigidity eigenvector $P\mathbf{v}_7 = \begin{bmatrix} (\mathbf{v}^x)^{\mathrm{T}} & (\mathbf{v}^y)^{\mathrm{T}} & (\mathbf{v}^z)^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, where P is defined in Theorem 2.16. For the remainder of the work, we drop the subscript and reserve the bold font $\mathbf{v}$ for the rigidity eigenvector. Note that in fact, the rigidity eigenvalue and eigenvector are state-dependent, and therefore also time-varying when the formation is induced by the spatial orientation of a mobile team of robots, or due to the action of state-dependent weights on the sensing and communication links.

We can now exploit the structure of the symmetric rigidity matrix for weighted frameworks. Using the form of the symmetric rigidity matrix given in (7), we define $\tilde{Q}(p(\mathcal{V})) = (I_3 \otimes W)Q(p(\mathcal{V}))(I_3 \otimes W)$ as a generalized weight matrix, and observe that

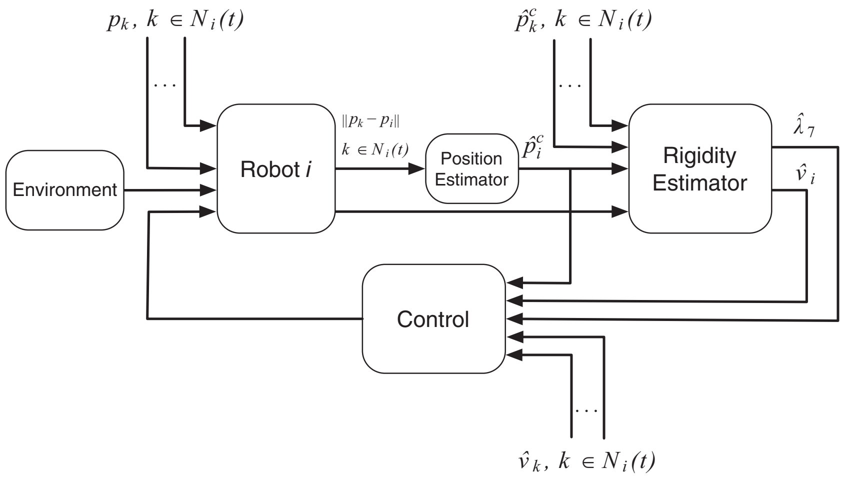

Fig. 4. Control architecture for distributed rigidity maintenance.

$$\mathcal{PRP}^{\mathrm{T}} = (I_3 \otimes E(\mathcal{G}))\tilde{Q}(p(\mathcal{V}))\left(I_3 \otimes E(\mathcal{G})^{\mathrm{T}}\right).$$

The elements of $\tilde{Q}(p(\mathcal{V}))$ are entirely in terms of the relative positions of each agent and the weighting functions defined on the edges as in (11).

The rigidity eigenvalue can now be expressed explicitly as

$$\lambda_7 = \sum_{(i,j)\in\mathcal{E}} \mathcal{W}_{ij}\Big((p_i^x - p_j^x)^2(\mathbf{v}_i^x - \mathbf{v}_j^x)^2 + (p_i^y - p_j^y)^2(\mathbf{v}_i^y - \mathbf{v}_j^y)^2 + (p_i^z - p_j^z)^2(\mathbf{v}_i^z - \mathbf{v}_j^z)^2 + 2(p_i^x - p_j^x)(p_i^y - p_j^y)(\mathbf{v}_i^x - \mathbf{v}_j^x)(\mathbf{v}_i^y - \mathbf{v}_j^y) + 2(p_i^x - p_j^x)(p_i^z - p_j^z)(\mathbf{v}_i^x - \mathbf{v}_j^x)(\mathbf{v}_i^z - \mathbf{v}_j^z) + 2(p_i^y - p_j^y)(p_i^z - p_j^z)(\mathbf{v}_i^y - \mathbf{v}_j^y)(\mathbf{v}_i^z - \mathbf{v}_j^z)\Big) = \sum_{(i,j)\in\mathcal{E}} \mathcal{W}_{ij}S_{ij}. \tag{12}$$

From (12), one can then derive a closed-form expression for $\frac{\partial\lambda_7}{\partial p_i^s}$, $s\in\{x, y, z\}$, i.e. the gradient of λ_7 with respect to each agent's position. In particular, by exploiting the structure of the terms S_{ij} and the properties of the employed weights $\mathcal{W}_{ij}$ (see, in particular, the previous Remark 3.3), it is possible to reduce $\frac{\partial\lambda_7}{\partial p_i^x}$ to the following *sum over the neighbors*,

$$\frac{\partial\lambda_7}{\partial p_i^x} = \sum_{j\in\mathcal{N}_i} \mathcal{W}_{ij}\Big(2(p_i^y - p_j^y)(\mathbf{v}_i^x - \mathbf{v}_j^x)(\mathbf{v}_i^y - \mathbf{v}_j^y) + 2(p_i^x - p_j^x)(\mathbf{v}_i^x - \mathbf{v}_j^x)^2 + 2(p_i^z - p_j^z)(\mathbf{v}_i^x - \mathbf{v}_j^x)(\mathbf{v}_i^z - \mathbf{v}_j^z)\Big) + \frac{\partial\mathcal{W}_{ij}}{\partial p_i^x}S_{ij} \tag{13}$$

and similarly for the y and z components.

The gradient (13) possesses the following key feature: it is a function of relative quantities, in particular of (i) relative components of the eigenvector $\mathbf{v}$, (ii) relative distances, and (iii) relative positions with respect to *neighboring agents* (see, again, Remark 3.3 for what concerns weights $\mathcal{W}_{ij}$), thus allowing for a distributed computation of its value once these quantities are locally available. Sections 4 and 5 will detail two estimation schemes able to recover all of these relative quantities by resorting to only *measured distances* with respect to 1-hop neighbors owing to the infinitesimal rigidity of the group formation.

3.3. The control architecture

The explicit description of the gradient of the rigidity eigenvalue in (13) motivates the general control architecture for the implementation of the rigidity maintenance action in (10). We observe that each agent requires knowledge of the rigidity eigenvalue, appropriate components of the rigidity eigenvector, and relative positions with respect to neighboring agents in a common reference frame. As already mentioned, all of these quantities are inherently global quantities, and thus a fully distributed implementation of (10) must include appropriate estimators for recovering these parameters in a distributed manner.

As a preview of the next sections in this work, Figure 4 depicts the general architecture needed by each agent to implement the rigidity maintenance control action (10).

1. Exploiting measured distances with respect to its 1-hop neighbors, and owing to the formation rigidity, each agent distributedly estimates relative positions in a common reference frame, labeled as the *position estimator* in the figure. This block is fully explained in Section 4.
2. The output of the position estimator is then used by each agent to perform a distributed estimation of the rigidity eigenvalue ($\hat{\lambda}_7$) and of the relative components of the eigenvector ($\hat{\mathbf{v}}$), labeled as the *rigidity estimator* in the figure. This procedure is explained in Section 5.
3. Thanks to these estimated quantities (relative positions, $\hat{\lambda}_7$ and $\hat{\mathbf{v}}$), each agent can finally implement the control action (10) in a distributed way for maintaining infinitesimal rigidity of the formation during the group motion

(while also coping with the various constraints and requirements embedded into weights $\mathcal{W}$). Maintaining infinitesimal rigidity guarantees in turn convergence of the position estimator from measured distances of step 1), and thus closes the 'estimation-control loop.'

We finally note that the proposed control architecture also implicitly assumes the *initial* spatial configuration of the agents (i.e. their positions $p(\mathcal{V})$ at time 0) to be infinitesimally rigid (with, in particular, a $\lambda_7 > \lambda_7^{\min}$). This assumption on the group initial condition is formally stated below.

Assumption 3.4. *The initial spatial configuration of the agents*, $p(\mathcal{V})$ *at time* $t = 0$, *is infinitesimally rigid with* $\lambda_7 > \lambda_7^{\min}$.

The purpose of requiring a minimum level of rigidity ($\lambda_7^{\min}$) is discussed in greater detail in Section 7.

4. Decentralized estimation of positions in a common frame

As explained, evaluation of the gradient control (13) requires that each agent has access to the *relative positions* of its neighboring agents. A main focus of this work, however, is to achieve rigidity maintenance using only *relative distance* measurements. In this section, we leverage the infinitesimal rigidity of the formation to estimate the relative position with respect to a *common reference point*, p_c, shared by all agents. In particular, each agent i, with $i = 1\ldots n$, will be able to compute an estimate $\hat{p}_{i,c}$ of its relative position $p_{i,c} = p_i - p_c$ to this common point. By exchanging their estimates over 1-hop communication channels, two neighboring agents i and j can then build an estimate $\hat{p}_{j,c} - \hat{p}_{i,c}$ of their actual relative position $p_j - p_j$ in a common reference frame. Note that both the graph (i.e. neighbor sets, edges, etc.) and the robot positions are time-varying quantities. However, in this section we omit dependency on time for the sake of conciseness.

We also note that this common reference point does not need to be stationary, i.e. it can move over time. In the following, we choose the point p_c to be attached to a *special agent* in the group, determined *a priori*. This agent will be denoted with the index i_c and, in the remainder of this section, we set $p_c = p_{i_c}$. We now proceed to describe a distributed scheme able to recover an estimation of the relative position $p_{i,c} = p_i - p_{i_c}$ for any agent in the group by exploiting the measured relative distances and the rigidity property of the formation.

To achieve this estimation, we first introduce additional assumptions on the capabilities of the special agent i_c. While all agents other than i_c are able to measure only the *relative distance* to their neighbors, the special agent i_c is required to be endowed with an additional sensor able to also measure, at any time t, the *relative position* (i.e. distance *and* bearing angles) of at least two non-collinear neighbors;[5] these two sensed neighbors will be denoted with the indexes $(\iota(t), \kappa(t)) \in \mathcal{N}_{i_c}(t)$.

Remark 4.1. *We stress that the agent indexes* $\iota(t)$ *and* $\kappa(t)$ *are* time-varying; *indeed, contrarily to the special agent* i_c, $\iota(t)$ *and* $\kappa(t)$ *are not preassigned to any particular agent in the multi-robot team. Therefore the special agent* i_c *only needs to measure its relative positions* $p_{\iota(t)} - p_{i_c}$ *and* $p_{\kappa(t)} - p_{i_c}$ *with respect to* any *two agents within its neighborhood (*ι *and* κ *are effectively arbitrary), with the points* p_{i_c}, $p_{\iota(t)}$ *and* $p_{\kappa(t)}$ *being non-collinear for all* $t \geq t_0$. *We believe this assumption is not too restrictive in practice, as it only require the presence of at least one robot equipped with a range plus bearing sensor while all the remaining ones can be equipped with simple range-only sensors.*

In the following we omit for brevity the dependency upon the time t of the quantities ι and κ.

In order to perform the distributed estimation of $p_{i,c} = p_i - p_c$, for all $i \in \{1,\ldots,n\}$, we follow the approach presented in Calafiore et al. (2010b), with some slight modifications dictated by the nature of our problem. Consistently with our notation, we define $\hat{p} = \left[\hat{p}_{1,c}^{\mathrm{T}} \;\ldots\; \hat{p}_{n,c}^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbb{R}^{3n}$. For compactness, we also denote by ℓ_{ij} the measured distance $\|p_j - p_i\|$, as introduced in Definition 3.1. We then consider the following least-squares estimation error:

$$e(\hat{p}) = \frac{1}{4}\sum_{\{i,j\}\in\mathcal{E}} \left(\|\hat{p}_{j,c} - \hat{p}_{i,c}\|^2 - \ell_{ij}^2\right)^2 + \frac{1}{2}\|\hat{p}_{i_c,c}\|^2 + \frac{1}{2}\|\hat{p}_{\iota,c} - (p_\iota - p_{i_c})\|^2 + \frac{1}{2}\|\hat{p}_{\kappa,c} - (p_\kappa - p_{i_c})\|^2 \tag{14}$$

Note that the quantities $\ell_{ij}, p_\iota - p_{i_c}$, and $p_\kappa - p_{i_c}$ are measured while all of the other quantities represent local estimates of the robots.

The non-negative error function $e(\hat{p})$ is zero if and only if:

- $\|\hat{p}_{j,c} - \hat{p}_{i,c}\|$ is equal to the measured distance ℓ_{ij} for all the pairs $\{i,j\} \in \mathcal{E}$;
- $\|\hat{p}_{i_c,c}\| = 0$;
- $\hat{p}_{\iota,c}$ and $\hat{p}_{\kappa,c}$ are equal to the measured relative positions $p_\iota - p_{i_c}$ and $p_\kappa - p_{i_c}$, respectively.

Note that the estimates $\hat{p}_{i_c,c}$, $\hat{p}_{\iota,c}$ and $\hat{p}_{\kappa,c}$ could be directly set to 0, $(p_\iota - p_{i_c})$, and $(p_\iota - p_{i_c})$, respectively, since the first quantity is known and the last two are measured. Nevertheless, we prefer to let the estimator obtaining these values via a 'filtering action' for the following reasons: first, the estimator provides a relatively simple way to filter out noise that might affect the relative position measurements; second, implementation of the rigidity maintenance controller only requires that $(\hat{p}_{j,c} - \hat{p}_{i,c}) \to (p_j - p_i)$, which is achieved if $\hat{p}_{j,c} \to p_j - \hat{p}_{i_c,c}$ and $\hat{p}_{i,c} \to p_i - \hat{p}_{i_c,c}$ for *any* common value of $\hat{p}_{i_c,c}$. Therefore, any additional hard constraint on $\hat{p}_{i_c,c}$ (e.g. $\hat{p}_{i_c,c} \equiv 0$) might unnecessarily over-constrain the estimator.

Applying a first-order gradient descent method to $e(\hat{p})$, we finally obtain the following *decentralized* update rule for the *i*-th agent ($i \neq i_c$):

$$\dot{\hat{p}}_{i,c} = -\frac{\partial e}{\partial \hat{p}_{i,c}} = \sum_{j \in \mathcal{N}_i} (\|\hat{p}_{j,c} - \hat{p}_{i,c}\|^2 - \ell_{ij}^2)(\hat{p}_{j,c} - \hat{p}_{i,c})$$
$$-\delta_{ii_c}\hat{p}_{i,c} - \delta_{i\iota}(\hat{p}_{\iota,c} - (p_\iota - p_{i_c})) - \delta_{i\kappa}(\hat{p}_{\kappa,c} - (p_\kappa - p_{i_c})), \quad (15)$$

where δ_{ij} is the well-known Kronecker's delta.[6] The estimator (15) is clearly decentralized since:

- ℓ_{ij} is locally measured by agent *i*;
- $\hat{p}_{i,c}$ is locally available to agent *i*;
- $\hat{p}_{j,c}$ can be transmitted using one-hop communication from agent *j* to agent *i*, for every $j \in \mathcal{N}_i$;
- $(p_\iota - p_{i_c})$ and $(p_\kappa - p_{i_c})$ are measured by agent i_c and can be transmitted using one-hop communication to agents ι and κ, respectively.

In order to show the relation between the proposed decentralized position estimator scheme and the infinitesimal rigidity property, one can restate (15) in matrix form as

$$\dot{\hat{p}} = -\mathcal{R}(\hat{p})\hat{p} + R(\hat{p})\ell + \Delta^c \quad (16)$$

where $\mathcal{R}(\hat{p})$ and $R(\hat{p})$ are the symmetric rigidity matrix and the rigidity matrix computed with the estimated positions, $\ell \in \mathbb{R}^{|\mathcal{E}|}$ is a vector whose entries are ℓ_{ij}^2, for all $\{i,j\} \in \mathcal{E}$, and $\Delta^c \in \mathbb{R}^{|\mathcal{E}|}$ contains the remaining terms of the right-hand side of (15).

Proposition 4.2. *If the framework is (infinitesimally) rigid, then the vector of true values* $p - (\mathbb{1}_n \otimes p_c) = \left[(p_1 - p_c)^{\mathrm{T}} \ \cdots \ (p_n - p_c)^{\mathrm{T}}\right]^{\mathrm{T}}$ *is an isolated local minimizer of* $e(\hat{p})$*. Therefore, there exists an* $\epsilon > 0$ *such that, for all initial conditions satisfying* $\|\hat{p}(0) - p - (\mathbb{1}_n \otimes p_c)\| < \epsilon$*, the estimation* $\hat{p}$ *converges to* $p - (\mathbb{1}_n \otimes p_c)$.

We point out that the estimator in the form (16) is identical to the formation controller proposed in Krick et al. (2009). Consequently, we refer the reader to this work for a discussion on the stability and convergence properties of this model. A similar estimation scheme is also proposed in Calafiore et al. (2010b). We briefly emphasize that the property of having the true value of relative positions $p - (\mathbb{1}_n \otimes p_c)$ as an isolated local minimizer of (14) is a consequence of the definition of infinitesimal rigidity and of the non-collinearity assumption of the agents i_c, ι, and κ.

We finally note that, in general, the rate of convergence of a gradient descent method is known to be slower than other estimation methods. However, we opted for this method since is its directly amenable to a distributed implementation and requires only first-order derivative information.

5. Distributed estimation of the rigidity eigenvalue and eigenvector

As seen in Section 4, when the multi-robot team possesses the infinitesimal rigidity property, it is possible to distributedly estimate the relative positions in a common reference frame for each agent. However, the proposed distributed rigidity maintenance control action (10) requires knowledge of some additional global quantities that are explicitly expressed in the expressions (13) and (10). In particular, each agent must know also the current value of the rigidity eigenvalue and certain components of the rigidity eigenvector. In this section we propose a distributed estimation scheme inspired by the distributed connectivity maintenance solution proposed in Yang et al. (2010) for obtaining the rigidity eigenvalue and eigenvector.

For the reader's convenience, we first provide a brief summary of the *power iteration method* for estimating the eigenvalues and eigenvectors of a matrix. We then proceed to show how this estimation process can be distributed by employing *PI consensus filters* and by suitably exploiting the structure of the symmetric rigidity matrix.

5.1. Power iteration method

The power iteration method is one of a suite of iterative algorithms for estimating the dominant eigenvalue and eigenvector of a matrix. Following the same procedure as in Yang et al. (2010), we employ a continuous-time variation of the algorithm that will compute the smallest non-zero eigenvalue and eigenvector of the symmetric rigidity matrix.

The discrete-time power iteration algorithm is based on the following iteration,

$$x^{(k+1)} = \frac{Ax^{(k)}}{\|Ax^{(k)}\|} = \frac{A^k x^{(0)}}{\|A^k x^{(0)}\|}$$

Under certain assumptions for the matrix A (i.e. no repeated eigenvalues), the iteration converges to the eigenvector associated to the largest eigenvalue of the matrix.

To adapt the power iteration to compute the rigidity eigenvector and eigenvalue, we leverage the results of Theorem 2.16 and consider the iteration on a *deflated* version of the symmetric rigidity matrix, i.e. $\widetilde{\mathcal{R}} = I - TT^{\mathrm{T}} - \alpha\mathcal{R}$ for some small enough $\alpha > 0$. The power iteration method estimates the largest eigenvalue of a matrix. As all of the eigenvalues of the symmetric rigidity matrix are non-negative, the largest eigenvalue of the deflated version $\widetilde{\mathcal{R}}$ will correspond to $1 - \alpha\lambda_7$, and thus can be used to estimate λ_7. The constant α ensures the matrix $\widetilde{\mathcal{R}}$ is positive semi-definite. The columns of the matrix $T \in \mathbb{R}^{3n \times 6}$ contain the eigenvectors corresponding to the zero eigenvalues of $\mathcal{R}$, for example, as characterized in Theorem 2.16. Note that the power iteration applied to the matrix $\widetilde{\mathcal{R}}$ will compute the eigenvector associated with the rigidity eigenvalue.[7]

The continuous-time counterpart of the power iteration algorithm now takes the form (Yang et al., 2010)

$$\dot{\widehat{\mathbf{v}}}(t) = -\left(k_1 TT^{\mathrm{T}} + k_2\mathcal{R} + k_3\left(\frac{\widehat{\mathbf{v}}(t)^{\mathrm{T}}\widehat{\mathbf{v}}(t)}{3n} - 1\right)I\right)\widehat{\mathbf{v}}(t) \quad (17)$$

where $\widehat{\mathbf{v}}$ is the *estimate* of the rigidity eigenvector, and the constants $k_1, k_2, k_3 > 0$ are chosen to ensure the trajectories converge to the rigidity eigenvector.[8] We present here the main result and refer the reader to Yang et al. (2010) for details of the proof, noting that the proof methodologies are the same for the system (17) as that proposed in Yang et al. (2010).

Theorem 5.1. *Assume that the weighted framework $(\mathcal{G}, p, \mathcal{W})$ with symmetric rigidity matrix $\mathcal{R}$ is infinitesimally rigid and has distinct non-zero eigenvalues, and let* $\mathbf{v}$ *denote the rigidity eigenvector. Then for any initial condition $\widehat{\mathbf{v}}(t_0) \in \mathbb{R}^{3n}$ such that $\mathbf{v}^{\mathrm{T}}\widehat{\mathbf{v}}(t_0) \neq 0$, the trajectories of (17) converge to the subspace spanned by the rigidity eigenvector, i.e.* $\lim_{t\to\infty}\widehat{\mathbf{v}}(t) = \gamma\mathbf{v}$ *for $\gamma \in \mathbb{R}$, if and only if the gains k_1,k_2 and k_3 satisfy the following conditions:*

1. $k_1, k_2, k_3 > 0$;
2. $k_1 > k_2\lambda_7$;
3. $k_3 > k_2\lambda_7$.

Furthermore, for any choice of constants $k_1, k_2, k_3 > 0$, the trajectories of (17) remain bounded and satisfy

$$||\widehat{\mathbf{v}}(t)|| \leq \max\left\{||\widehat{\mathbf{v}}(t_0)||, \sqrt{3n}\right\}, \quad \textit{for all } t \geq t_0$$

In particular, the trajectory converges to the rigidity eigenvector with

$$\lim_{t\to\infty} ||\widehat{\mathbf{v}}(t)|| = \sqrt{3n\left(1 - \frac{k_2}{k_3}\right)\lambda_7}$$

Remark 5.2. *The power iteration proposed in (17) assumes that the symmetric rigidity matrix is static. However, in a dynamic setting the parameters of the rigidity matrix are a function of the state of the robots in a multi-robot system, and both the symmetric rigidity matrix and the expression of its null space are inherently* time-varying. *While the proof provided in Yang et al. (2010) does not explicitly address the time-varying case, our experience suggests that the dynamics of (17) is able to track even a time-varying rigidity eigenvector, so long as the dynamics of the robots are slower than the estimator. The speed of convergence of (17), of course, is also tunable by the constants k_i.*

Remark 5.3. *Another important subtlety of the dynamics (17) is the requirement that the rigidity eigenvalue is unique. When the rigidity eigenvalue is not unique, the associated eigenvector can belong to (at least) a two-dimensional subspace $\mathcal{L}$, so that (17) cannot be expected to converge to a unique eigenvector but rather to an equilibrium point in $\mathcal{L}$ (see, e.g., Yang et al., 2010). This can pose difficulties in real-world conditions since non-idealities such as noise in measuring the agent states (used in evaluating the symmetric rigidity matrix $\mathcal{R}$), and discretization when numerically integrating (17), can make the equilibrium point for (17) in $\mathcal{L}$ to abruptly vary over time, thus preventing a successful convergence of the estimation of* $\mathbf{v}$.

5.2. A distributed implementation

The results of Section 5.1 provide a continuous-time estimator for estimating the rigidity eigenvalue and eigenvector of the symmetric rigidity matrix. The estimator given in (17), however, is a *centralized* implementation. Moreover, certain parameters used in (17) are expressed using a common reference frame (i.e. the quantity TT^{T}, see Theorem 2.16 and Remark 2.17) or require each robot to know the entire estimator state (i.e. the quantity $\widehat{\mathbf{v}}(t)^{\mathrm{T}}\widehat{\mathbf{v}}(t)$ in (17)). We propose in this section a distributed implementation for the rigidity estimator that overcomes these difficulties, in particular by leveraging the results of Section 4. In the same spirit as the solution proposed in Yang et al. (2010), we make use of the *PI average consensus filter* (Freeman et al., 2006) to distributedly compute the necessary quantities of interest, and strongly exploit the particular structure of the symmetric rigidity matrix.

Our approach to the distribution of (17) is to exploit both the built-in distributed structure (i.e. the symmetric rigidity matrix $\mathcal{R}$) and the reduction of the other parameters to values that all agents can obtain via a distributed algorithm. In this direction, we now proceed to analyze each term in (17) and discuss the appropriate strategies for implementing the estimator in a distributed fashion.

Concerning the first term $TT^{\mathrm{T}}\widehat{\mathbf{v}}$, Theorem 2.16 provides an analytic characterization of the eigenvectors associated with the zero eigenvalues of the symmetric rigidity matrix (assuming the graph is infinitesimally rigid). To begin the analysis, we explicitly write out the matrix T and examine the elements of the matrix TT^{T}. Following the comments of Remark 2.17, we express the null-space vectors in terms of *relative positions* to an arbitrary point $p_c = [p_c^x \;\; p_c^y \;\; p_c^z] \in \mathbb{R}^3$; in particular, the point p_c will be the special agent i_c described in Section 4:

$$T = \begin{bmatrix} \mathbb{1}_n & \mathbf{0} & \mathbf{0} & p^y - p_c^y\mathbb{1}_n & p^z - p_c^z\mathbb{1}_n & \mathbf{0} \\ \mathbf{0} & \mathbb{1}_n & \mathbf{0} & p_c^x\mathbb{1}_n - p^x & \mathbf{0} & p^z - p_c^z\mathbb{1}_n \\ \mathbf{0} & \mathbf{0} & \mathbb{1}_n & \mathbf{0} & p_c^x\mathbb{1}_n - p^x & p_c^y\mathbb{1}_n - p^y \end{bmatrix}$$

For the remainder of this discussion, we assume that all agents have access to their state in an estimated coordinate frame relative to the point p_{i_c}, the details of which were described in Section 4:

$$TT^{\mathrm{T}} = \begin{bmatrix} \mathbb{1}_n\mathbb{1}_n^{\mathrm{T}} + p^{y,c}(p^{y,c})^{\mathrm{T}} + p^{z,c}(p^{z,c})^{\mathrm{T}} & -p^{y,c}(p^{x,c})^{\mathrm{T}} & -p^{z,c}(p^{x,c})^{\mathrm{T}} \\ -p^{x,c}(p^{y,c})^{\mathrm{T}} & \mathbb{1}_n\mathbb{1}_n^{\mathrm{T}} + p^{x,c}(p^{x,c})^{\mathrm{T}} + p^{z,c}(p^{z,c})^{\mathrm{T}} & -p^{z,c}(p^{y,c})^{\mathrm{T}} \\ -p^{x,c}(p^{z,c})^{\mathrm{T}} & -p^{y,c}(p^{z,c})^{\mathrm{T}} & \mathbb{1}_n\mathbb{1}_n^{\mathrm{T}} + p^{x,c}(p^{x,c})^{\mathrm{T}} + p^{y,c}(p^{y,c})^{\mathrm{T}} \end{bmatrix} \quad (18)$$

To simplify notation, we write as in Section 4, for example, $p^{y,c} = p^y - p_c^y\mathbb{1}_n$ and $p_{i,c} = p_i - p_c$. Following our earlier notation, we also partition the vector $\widehat{\mathbf{v}}$ into each coordinate, $\widehat{\mathbf{v}}^x$, $\widehat{\mathbf{v}}^y$, and $\widehat{\mathbf{v}}^z$. Let $\mathbf{avg}(r)$ denote the average value of the elements in the vector $r \in \mathbb{R}^n$, i.e. $\mathbf{avg}(r) = \frac{1}{n}\mathbb{1}_n^{\mathrm{T}}r$. Then it is straightforward to verify that

$$\mathbb{1}_n\mathbb{1}_n^{\mathrm{T}}\widehat{\mathbf{v}}^k(t) = n\,\mathbf{avg}(\widehat{\mathbf{v}}^k(t))\mathbb{1}_n, \quad k \in \{x,y,z\} \quad (19)$$

$$p_{i,c}(p_{j,c})^{\mathrm{T}}\widehat{\mathbf{v}}^k(t) = n\,\mathbf{avg}(p_{j,c}\circ\widehat{\mathbf{v}}^k)p_{i,c}, \quad i,j,k \in \{x,y,z\} \quad (20)$$

where '$\circ$' denotes the element-wise multiplication of two vectors.

This characterization highlights that, in order to evaluate the term $TT^{\mathrm{T}}\widehat{\mathbf{v}}$, each agent must compute the average amongst all agents of a certain value that is a function of the current state of the estimator and of the positions in some common reference frame whose origin is the point p_c. It is well known that the *consensus protocol* can be used to distributedly compute the average of a set of numbers (Mesbahi and Egerstedt, 2010). The speed at which the consensus protocol can compute this value is a function of the connectivity of the underlying graph and the weights used in the protocol. In this framework, however, a direct application of the consensus protocol will not be sufficient. Indeed, it is expected that each agent will be physically moving, leading to a time-varying description of the matrix TT^{T} (see Remark 5.2). In addition, the underlying network is also dynamic as sensing links between agents are inherently state dependent.

The use of a *dynamic* consensus protocol introduces additional tuning parameters that can be used to ensure that the distributed average calculation converges faster than the underlying dynamics of each agent in the system, as well as the ability to track the average of a time-varying signal. We employ the following *PI average consensus filter* proposed in Freeman et al. (2006),

$$\begin{bmatrix} \dot{z}(t) \\ \dot{w}(t) \end{bmatrix} = \begin{bmatrix} -\gamma I_n - K_P L(\mathcal{G}(t)) & K_I L(\mathcal{G}(t)) \\ -K_I L(\mathcal{G}(t)) & 0 \end{bmatrix}\begin{bmatrix} z(t) \\ w(t) \end{bmatrix} + \begin{bmatrix} \gamma I_n \\ 0 \end{bmatrix} u(t) \quad (21)$$

$$y(t) = \begin{bmatrix} I_n & 0 \end{bmatrix}\begin{bmatrix} z(t) \\ w(t) \end{bmatrix} \quad (22)$$

The parameters $K_P, K_I \in \mathbb{R}$ and $\gamma \in \mathbb{R}$ are used to ensure stability and tune the speed of the filter. An analysis of the stability and performance of this scheme with time-varying graphs is given in Freeman et al. (2006). Figure 5 provides a block diagram representation of how the PI consensus filters are embedded into the calculation of $TT^{\mathrm{T}}\widehat{\mathbf{v}}(t)$ (in only the x coordinate).

As for the second term in (17), as shown in Section 2.3 the symmetric rigidity matrix is by construction a distributed operator. The term $\mathcal{R}\widehat{\mathbf{v}}(t)$ can be computed using only information exchanged between neighboring agents, as determined by the sensing graph.

The final term in (17) is a normalization used to drive the eigenvector estimate to the surface of a sphere of radius $\sqrt{3n}$. Using the same analysis as above, it can be verified that

$$\left(\frac{\widehat{\mathbf{v}}(t)^{\mathrm{T}}\widehat{\mathbf{v}}(t)}{3n} - 1\right)\widehat{\mathbf{v}}(t) = (\mathbf{avg}(\widehat{\mathbf{v}}(t)\circ\widehat{\mathbf{v}}(t)) - 1)\widehat{\mathbf{v}}(t) \quad (23)$$

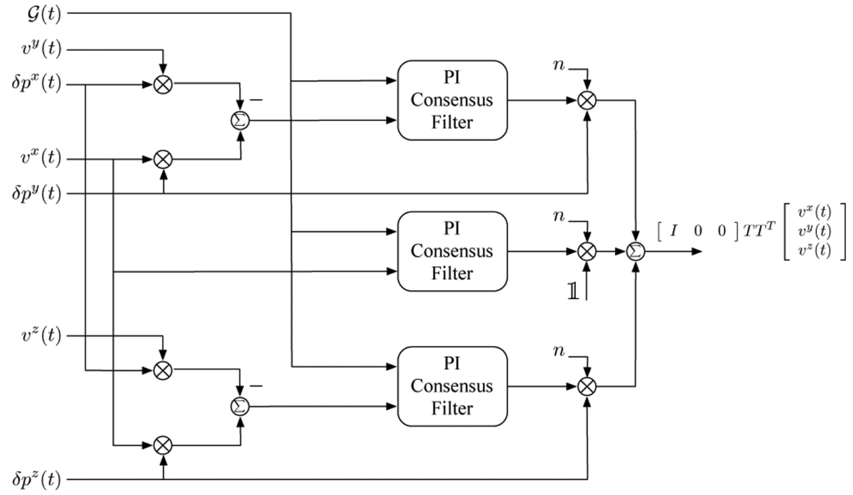

Fig. 5. Block diagram showing PI consensus filters in calculation of $TT^{\mathrm{T}}\widehat{\mathbf{v}}(t)$.

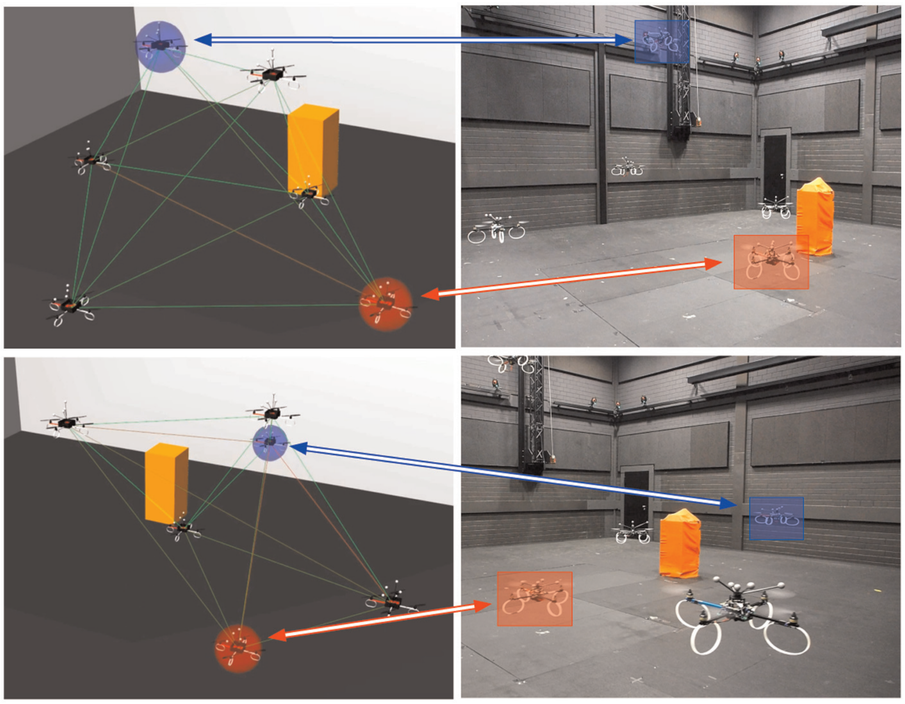

Fig. 6. Two snapshots of the reported experiment. Left: Simulated 3D views showing, in particular, the inter-agent links (red, almost disconnected link; green, optimally connected link). Right: Corresponding pictures of the experimental setup. The two highlighted quadrotor UAVs are partially controlled by two human operators.

This quantity can therefore be distributedly computed using an additional PI consensus filter.

Using the result of Theorem 5.1 and the PI consensus filters, each agent is also able to estimate the rigidity eigenvalue.

Corollary 5.4. *Let* $\overline{\mathbf{v}}_i^2(t)$ *denote the output of the PI consensus filter for estimating the quantity* $\mathbf{avg}(\widehat{\mathbf{v}}(t) \circ \widehat{\mathbf{v}}(t))$ *for agent i. Then agent i's estimate of the rigidity eigenvalue,* $\hat{\lambda}_7^i$*, can be obtained as*

$$\hat{\lambda}_7^i = \frac{k_3}{k_2}\left(1 - \overline{\mathbf{v}}_i^2(t)\right).$$

In summary, each agent implements the following filters:

- estimation of a common reference frame using (15);
- estimation of the rigidity eigenvector using (17);
- a PI-consensus filter for tracking the average of the estimate of the rigidity eigenvector (19);
- a PI-consensus filter for tracking the quantity described in (20);
- a PI-consensus filter for tracking the average of the square of the rigidity eigenvector estimate (23).

For completeness, we now present the full set of filters that each robot executes:

$$\begin{aligned}\dot{\hat{\mathbf{v}}}_i^x = & -k_1 n\left(\overline{\mathbf{v}}_i^x + z_i^{xy}(t)\hat{p}_{i,c}^y + z_i^{xz}\hat{p}_{i,c}^z(t)\right) \\ & - k_2 \sum_{j \in \mathcal{N}_i(t)} W_{ij}\left(\widehat{\mathbf{v}}_i^x(t) - \widehat{\mathbf{v}}_j^x\right) - k_3\left(\overline{\mathbf{v}}_i^x - 1\right)\widehat{\mathbf{v}}_i^x\end{aligned} \tag{24}$$

$$\begin{aligned}\dot{\hat{p}}_{i,c} = & \sum_{j \in \mathcal{N}_i(t)} (\|\hat{p}_{j,c} - \hat{p}_{i,c}\|^2 - \ell_{ij}^2)(\hat{p}_{j,c} - \hat{p}_{i,c}) \\ & - \delta_{ii_c}\hat{p}_{i,c} - \delta_{i\iota}(\hat{p}_{\iota,c} - (p_\iota - p_{i_c})) \\ & - \delta_{i\kappa}(\hat{p}_{\kappa,c} - (p_\kappa - p_{i_c}))\end{aligned} \tag{25}$$

$$\dot{\hat{\mathbf{v}}}_i^x = \gamma\left(\widehat{\mathbf{v}}_i^x - \overline{\mathbf{v}}_i^x\right) - K_P \sum_{j \in \mathcal{N}_i}\left(\overline{\mathbf{v}}_i^x - \overline{\mathbf{v}}_j^x(t)\right) + K_I \sum_{j \in \mathcal{N}_i(t)}\left(\bar{w}_i^x - \bar{w}_j^x\right) \tag{26}$$

$$\dot{\bar{w}}_i^x = -K_I \sum_{j \in \mathcal{N}_i(t)}\left(\overline{\mathbf{v}}_i^x - \overline{\mathbf{v}}_j^x\right) \tag{27}$$

$$\begin{aligned}\dot{\hat{\mathbf{v}}}_i^{2x} = & \gamma\left((\widehat{\mathbf{v}}_i^x)^2 - \overline{\mathbf{v}}_i^{2x}\right) - K_P \sum_{j \in \mathcal{N}_i(t)}\left(\overline{\mathbf{v}}_i^{2x} - \overline{\mathbf{v}}_j^{2x}\right) \\ & + K_I \sum_{j \in \mathcal{N}_i(t)}\left(\bar{w}_i^{2x} - \bar{w}_j^{2x}\right)\end{aligned} \tag{28}$$

$$\dot{\bar{w}}_i^{2x} = -K_I \sum_{j \in \mathcal{N}_i(t)}\left(\overline{\mathbf{v}}_i^{2x} - \overline{\mathbf{v}}_j^{2x}\right) \tag{29}$$

$$\begin{aligned}\dot{z}_i^{xy} = & \gamma((\hat{p}^y \circ \widehat{\mathbf{v}}^x - \hat{p}^x \circ \widehat{\mathbf{v}}^y) - z_i^{xy}) - K_P \sum_{j \in \mathcal{N}_i(t)}\left(z_i^{xy} - z_j^{xy}\right) \\ & + K_I \sum_{j \in \mathcal{N}_i(t)}\left(w_i^{xy}(t) - w_j^{xy}\right)\end{aligned} \tag{30}$$

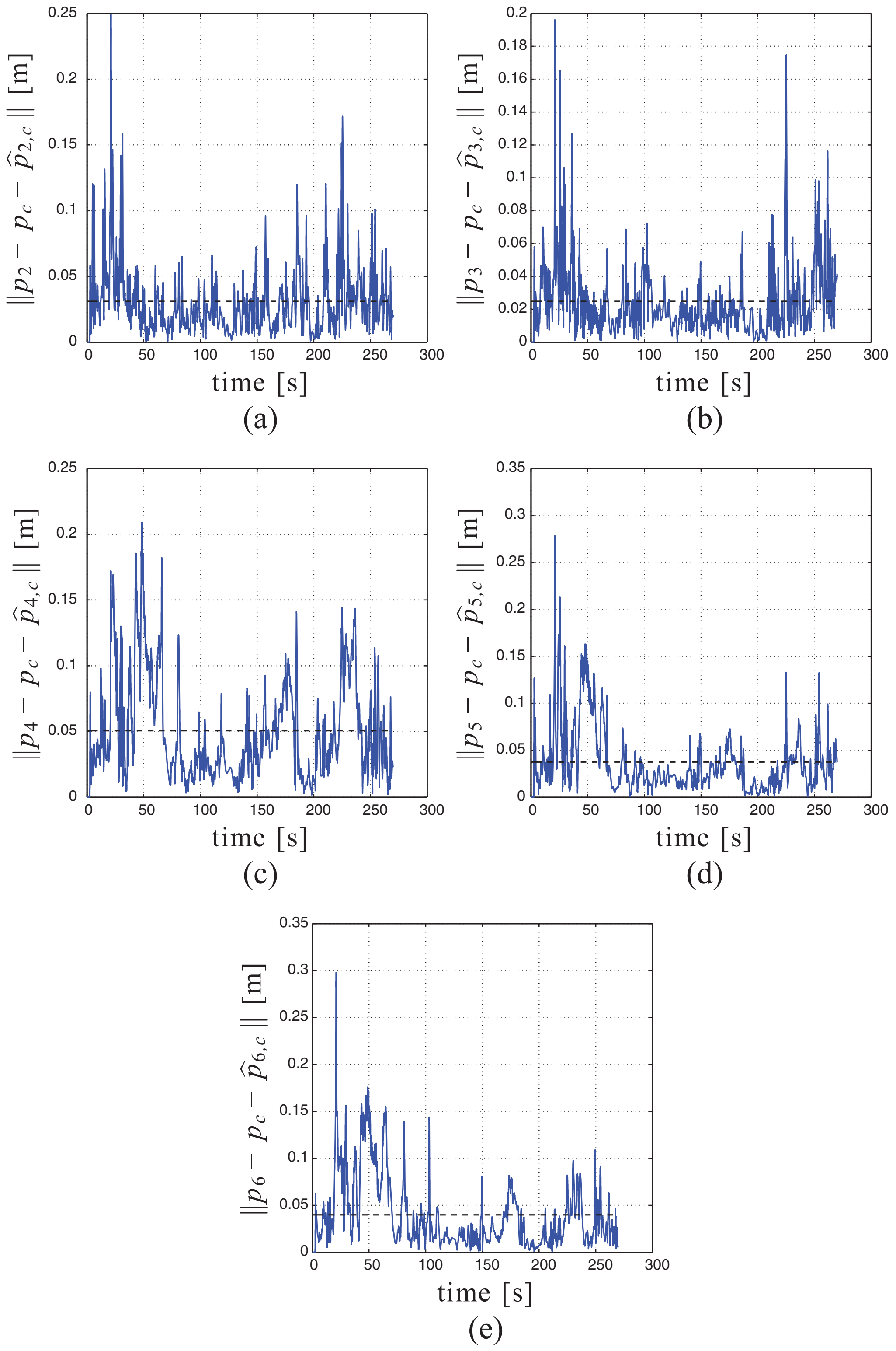

Fig. 7. Behavior of $||p_i - p_c - \hat{p}_{i,c}||$, $i = 2 \ldots 6$, the norm of the estimation error for the relative positions of agents 2…6 with respect to agent $i_c = 1$. The horizontal dashed black line represents the mean value of each error norm over time. Note how the estimation errors keep a low value during the group motion and thus indicate the ability of each robot to recover its relative position with respect to the robot $i_c = 1$ by only exploiting measured distances with respect to its neighbors and the infinitesimal rigidity of the formation.

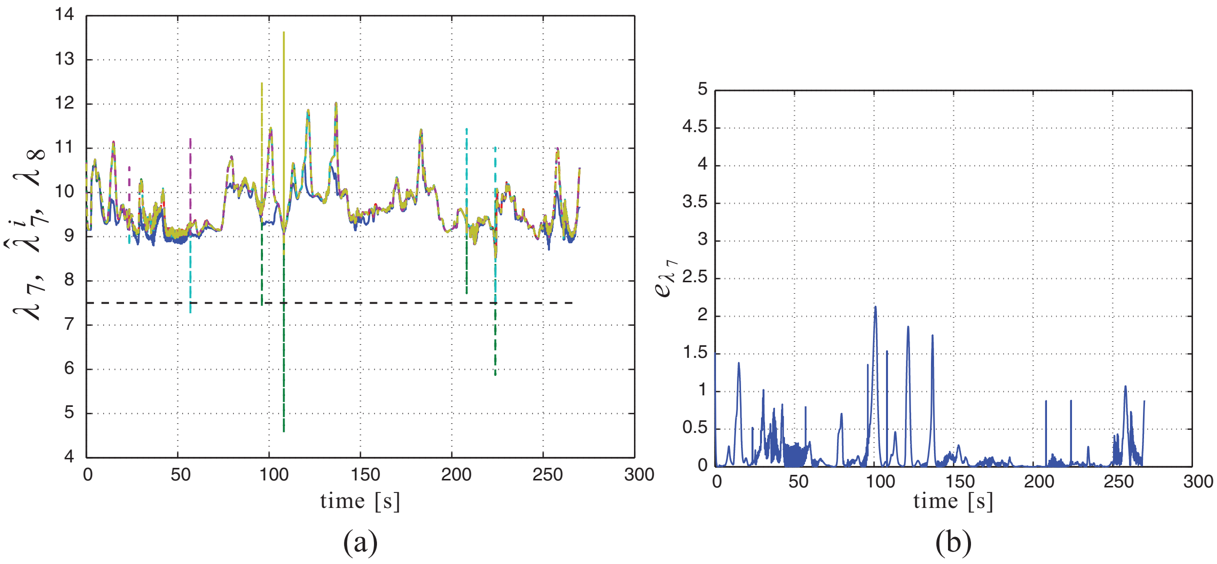

Fig. 8. (a) Behavior of $\lambda_7(t)$ (blue line) and the six estimations $\hat{\lambda}_7^i(t)$ (dashed colored lines) which result almost coincident. (b) Behavior of the overall rigidity eigenvalue estimation error $e_\lambda(t)$ as defined in (35).

$$\dot{w}_i^{xy} = -K_I \sum_{j\in\mathcal{N}_i(t)} \left(z_i^{xy} - z_j^{xy}\right) \quad (31)$$

$$\dot{z}_i^{xz} = \gamma\left((\hat{p}^z \circ \widehat{\mathbf{v}}^x - \hat{p}^x \circ \widehat{\mathbf{v}}^z) - z_i^{xz}\right) - K_P \sum_{j\in\mathcal{N}_i(t)} \left(z_i^{xy} - z_j^{xy}\right) + K_I \sum_{j\in\mathcal{N}_i(t)} \left(w_i^{xy} - w_j^{xy}\right) \quad (32)$$

$$\dot{w}_i^{xz} = -K_I \sum_{j\in\mathcal{N}_i(t)} \left(z_i^{xz} - z_j^{xz}\right). \quad (33)$$

These equations are written only for the x coordinate associated with all of the quantities. Observe, however, that the filters needed for the y and z coordinates do not require additional integrators, as similar filters can be vectorized (for example, the PI filters can be combined as in (21)). For the readers convenience, a summary of the notation and variable definitions used in (24)–(33) is provided in Table 1.

Remark 5.5. *Equations (24)–(33) show that each agent requires a 10th-order dynamic estimator for estimating the rigidity eigenvector and eigenvalue. This filter is composed of three PI-consensus filters, a relative position estimation filter, and the power iteration filter. An important point to emphasize is the order of the overall filter is independent of the number of agents in the ensemble, and thus is a scalable solution.*

6. The rigidity maintenance controller

The primary focus of this work until now was a detailed description of how the rigidity of a multi-robot formation can be maintained in a distributed fashion. The basic idea was to follow the gradient of an appropriately defined potential function of the rigidity eigenvalue; this control strategy was presented in (13). The fundamental challenge for the implementation of this control strategy was twofold: on the one hand, rigidity of a formation is an inherently *global* property of the network, and on the other hand, the control law depended on relative position measurements in a *common* reference fame.

A truly distributed solution based on this control strategy requires each agent to estimate a common inertial reference frame and also estimate the rigidity eigenvalue and eigenvector of the formation. The solution to these estimation problems was presented in Sections 4 and 5, with the complete set of filter equations summarized in (24)–(33). Note that both estimation strategies implicitly require that the underlying formation is infinitesimally rigid (see also Assumption 3.4). The final step for implementation of the rigidity maintenance controller is then to replace all of the state variables given in (13) with the appropriate estimated states computed by the relative position estimators and rigidity eigenvalue estimators. The local controller for each agent is thus given as,[9]

$$\xi_i^x = -\frac{\partial V(\hat{\lambda}_7^i)}{\partial \lambda_7} \sum_{j\in\mathcal{N}_i} W_{ij}\Big(2(\hat{p}_{i,c}^x - \hat{p}_{j,c}^x)(\widehat{\mathbf{v}}_i^x - \widehat{\mathbf{v}}_j^x)^2 + 2(\hat{p}_{i,c}^y - \hat{p}_{j,c}^y)(\widehat{\mathbf{v}}_i^x - \widehat{\mathbf{v}}_j^x)(\widehat{\mathbf{v}}_i^y - \widehat{\mathbf{v}}_j^y) + 2(\hat{p}_{i,c}^z - \hat{p}_{j,c}^z)(\widehat{\mathbf{v}}_i^x - \widehat{\mathbf{v}}_j^x)(\widehat{\mathbf{v}}_i^z - \widehat{\mathbf{v}}_j^z)\Big) + \frac{\partial W_{ij}}{\partial p_i^x}\hat{S}_{ij} \quad (34)$$

in conjunction with all of the estimation filters of (24)–(33).

Remark 6.1. *The interconnection of the relative position estimator, rigidity eigenvalue estimator, and gradient controller leads to a highly non-linear dynamics for which a*

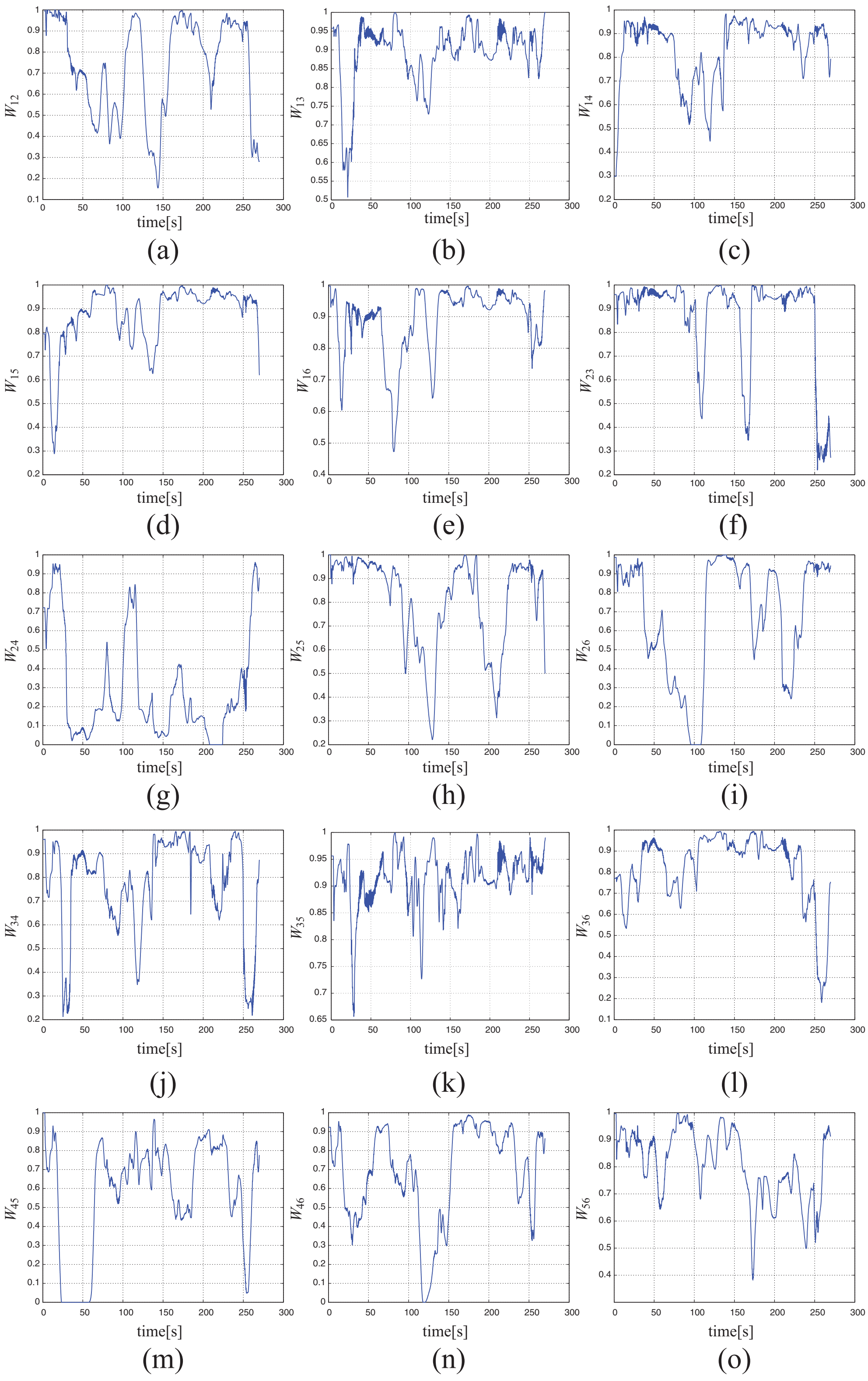

Fig. 9. Behavior of the 15 weights $W_{uv}(t)$ for all the possible edges of graph $\mathcal{G}$. Note how the values of weights $W_{uv}(t)$ vary over time because of the sensing/communication constraints and requirements embedded within their definition (see Section 3.1). Some weights (e.g. W_{24} and W_{45}) also temporarily vanish indicating loss of the corresponding edge (and, thus, the time-varying nature of graph $\mathcal{G}$).

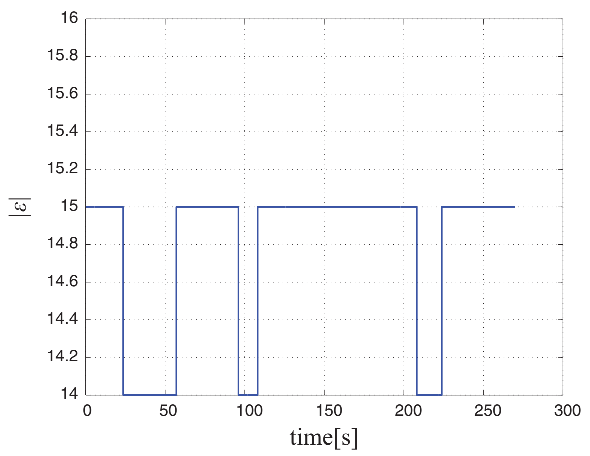

Fig. 10. Total number of edges in the graph $\mathcal{G}$ during the group motion.

formal proof analysis is not straightforward. While we are currently working towards a deeper analysis in this sense, the approach taken in this work is to exploit the typical (although informal) time-scale separation argument commonly found in many robotics applications relying on feedback control from an estimated state (as, e.g., when using an extended Kalman filter). Basically, the estimator dynamics is assumed "fast enough" such that its transient behavior can be considered as a second-order perturbation with respect to the robot motion (see also Yang et al., 2010) for an equivalent assumption in the context of decentralized connectivity maintenance control.

7. Experimental results

In this section we report some experimental results aimed at illustrating the machinery proposed so far for distributed rigidity maintenance. The experiments involved a total of $N = 6$ quadrotor UAVs (five real and one simulated) flying the environment shown in Figure 6. A video illustrating the various phases of the experiment (Extension 1) is attached to the paper.

All of the quadrotor UAVs were implementing the rigidity maintenance action (34) in addition to the estimation filters presented in (24)–(33). In addition, for two of the quadrotor UAVs (namely, quadrotors 1 and 2) an exogenous bounded velocity term $\xi_i^* \in \mathbb{R}^3$ was also added to (34); this allows for two human operators to independently control the motion of quadrotors 1 and 2 during the experiment, so as to steer the whole formation and trigger the various behaviors embedded in the weights $\mathcal{W}_{uv}$ (formation control, obstacle avoidance, sensing limitations).[10]

Our experimental quadrotor platform is a customized version of the MK-Quadro (see http://www.mikrokopter.de) implementing the TeleKyb ROS framework (see http://www.ros.org/wiki/telekyb) for flight control, experimental workflow management and human inputting. Attitude is stabilized with a fast inner loop that takes advantage of high-rate/onboard accelerometer and gyroscope measurements while the velocity stabilization is achieved by a slower control loop that measures the current velocity thanks to an external motion capture system. The motion capture system is also used to obtain relative distance measurements among the robots and the two bearing measurements needed by the special robot i_c. The reader is referred to Franchi et al. (2012b) for a detailed description of the quadrotor-based experimental setup.

We start illustrating the behavior of the relative position estimator described in Section 4 and upon which all of the subsequent steps are based (estimation of λ_7 and $\mathbf{v}$ and evaluation of the control action (10)). As explained in Section 4, owing to the formation infinitesimal rigidity, the scheme (15) allows each agent i to build an estimation $\hat{p}_{i,c}$ of its relative position $p_i - p_c$ with respect to the agent i_c, with $i_c = 1$ in this experiment. Figures 7(a)–(e) report the behavior of the norm of the estimation errors $||p_i - p_c - \hat{p}_{i,c}||$ for $i = 2\ldots6$ together with their mean values (dashed horizontal black line). It is then possible to verify how the relative position estimation errors keep low values over time, thus effectively allowing every agent to recover its correct relative position with respect to p_c from the measured relative distances.

As for the rigidity eigenvalue estimation of Section 5, Figure 8(a) reports the behavior of $\lambda_7(t)$ (solid blue line), of the six estimations $\hat{\lambda}_7^i(t)$ (solid colored lines almost superimposed to $\lambda_7(t)$), and of the minimum threshold $\lambda_7^{\min} = 7.5$ (horizontal dashed line). From the plot one can verify: (i) the accuracy in recovering the value of $\lambda_7(t)$ (note how the six estimations are almost superimposed on the real value) and (ii) that $\lambda_7(t) > \lambda_7^{\min}$ at all times apart from few isolated spikes, implying that *formation rigidity* was maintained during the task execution. As an additional indication of the eigenvalue estimation performance, Figure 8(b) shows the total estimation error for the rigidity eigenvalue

$$e_\lambda(t) = \frac{\sum_{i=1}^{N} \lambda_7(t) - \hat{\lambda}_7^i(t)|}{N} \tag{35}$$

which again confirms the accuracy of the estimation strategy.

Figures 9(a–o) report the behavior of the 15 weights W_{uv} defined in (11) and associated with all of the possible edges of graph $\mathcal{G}$ in order to show their *time-varying* nature because of the constraints and requirements listed in Section 3.1. Note how the value of some weight drops to zero over time (e.g. $W_{45}(t)$ at about $t = 25$ s or $W_{24}(t)$ at about $t = 210$ s), thus indicating loss of the corresponding edge. In the same spirit, Figure 10 shows the total number of edges $|\widehat{\mathcal{E}}|$ of the unweighted graph $\widehat{\mathcal{G}}$ (i.e. of non-zero weights W_{uv}, see Definition 2.9) during the group motion. These results highlight the time-varying nature of graph $\mathcal{G}$ which, as explained in the previous sections, is not constrained to keep a given fixed topology but is free to lose

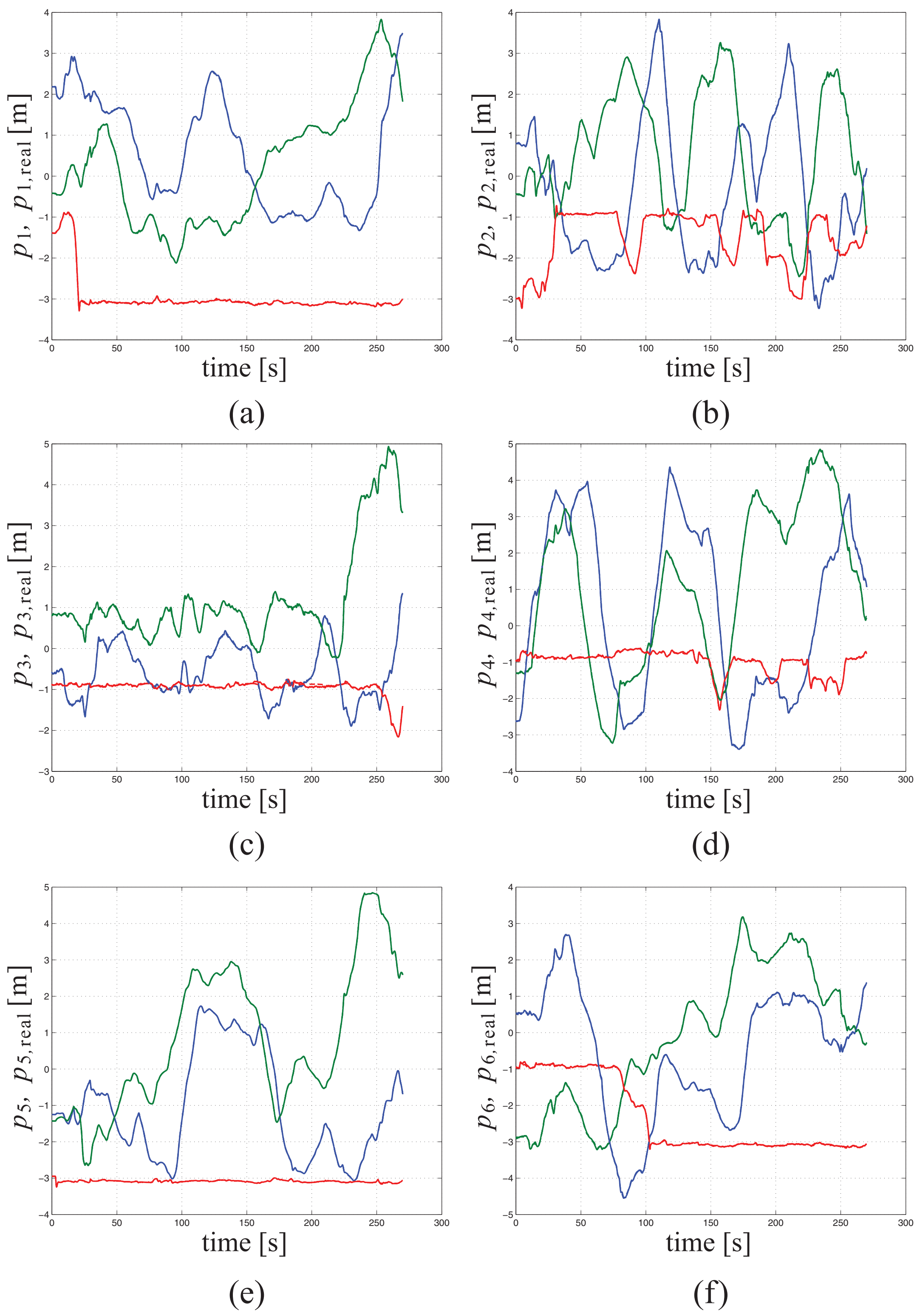

Fig. 11. Behavior of $p_i(t)$ (solid) and $p_{i,real}$ (dashed): these are basically superimposed, showing the accuracy of the quadrotors in tracking the reference trajectory $p_i(t)$. In the plots the following color code is used: blue/red/green solid/dashed lines correspond to the x / y / z components of $p_i(t)$ and $p_{i,real}$.

or gain edges as long as infinitesimal rigidity of the formation is preserved.

Finally, Figures 11(a)–(f) report the behavior over time of $p_i(t)$ (the ith agent position, solid lines) and of $p_{i,\text{real}}$ (t) (the i th quadrotor position, dashed lines) while tracking the motion of $p_i(t)$. The two position vectors result almost perfectly coincident, thus indicating a successful tracking performance of the quadrotors (and the soundness of our

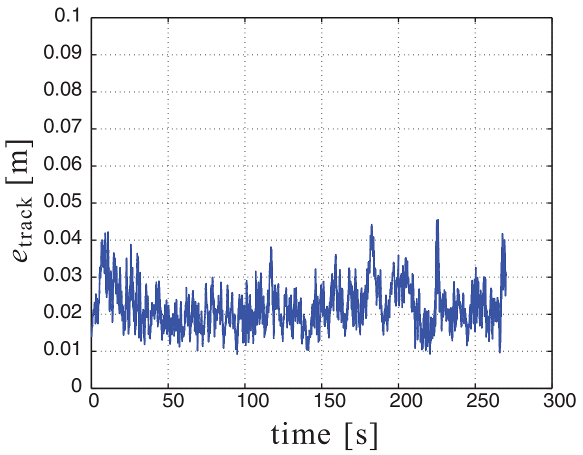

Fig. 12. Behavior of the tracking error e_{track} (t) defined in (36) showing again the good tracking performance of the six quadrotors.

modeling assumptions). As a further confirmation of this fact, the norm of the overall tracking error defined as

$$e_{\text{track}}(t) = \frac{\sum_{i=1}^{N} ||p_i(t) - p_{i,real}(t)||}{N} \tag{36}$$

is also reported in Figure 12.

8. Concluding remarks

This work presented a fully distributed solution for the rigidity maintenance control of a multi-robot system. As discussed in the introduction, rigidity is an important architectural feature for multi-robot systems that enables, for example, formation keeping and localization using only range-based measurements. The main theme of this work, therefore, was the distributed implementation of a number of algorithms for estimation and control in a multi-robot system related to rigidity maintenance. In particular, we demonstrated how the *rigidity eigenvalue* and eigenvector, used to decide whether a formation is infinitesimally rigid, can be distributedly estimated using a suite of estimators based on dynamic consensus filters and the power iteration method for eigenvalue estimation. The rigidity property also allowed for estimation of a common inertial reference frame using only *range-based measurements*, along with one single endowed agent that is able to sense both range and bearing. The estimation of these quantities were then embedded in a gradient-based distributed control action ensuring each agent moves in a way that guarantees rigidity of the formation is maintained. This control scheme also explicitly handles a variety of practical multi-robot constraints, including sensing and communication ranges, collision and obstacle avoidance, and line-of-sight requirements. The validity of the proposed algorithms was demonstrated by a team of six quadrotor UAVs flying in a cluttered environment.

This work also highlighted a number of directions for future research. In particular, the estimation of the rigidity eigenvalue assumed that there is a separation between the rigidity eigenvalue and the next largest eigenvalue, i.e. $|\lambda_7 - \lambda_8| > 0$. While the reported experimental results showed a large degree of robustness with respect to this effect, there remain both theoretical and practical questions related to this problem. For instance, it would be interesting to complement the rigidity maintenance controller with an additional term meant to maintain a minimum separation among λ_8 and λ_7. Another extension is to relax the requirement for having a special agent endowed with additional sensing capabilities (i.e. range and bearing). This would lead to a distributed solution involving only range measurements for all robots in the ensemble.

Despite these remaining challenges, this work has successfully demonstrated the power of distributed strategies for multi-robot systems. Indeed, it is remarkable to observe the behavior of the multi-robot team running many distributed filters to achieve a common global objective. The refinement of these strategies will no doubt become an important requirement as autonomous multi-robot systems are integrated more into a variety of application domains.

Funding

Part of Heinrich Bülthoff's research was supported by the Brain Korea 21 PLUS Program through the National Research Foundation of Korea funded by the Ministry of Education.

Notes

1. The second smallest eigenvalue of the graph Laplacian matrix.
2. If the graph is not connected, there will be additional eigenvalues at the origin corresponding to the number of connected components of the graph, see Godsil and Royle (2001).
3. A more detailed proof for the two-dimensional case is provided in Zelazo et al. (2012).
4. As for collision with obstacles, an equivalent behavior is automatically obtained from weights γ_{uv}^{b}, see again Robuffo Giordano et al. (2013) for a full explanation. Also note that, because of the definition of weights W_{uv}, one has $\mathcal{N}_u \subseteq \mathcal{S}_u$ but $\mathcal{S}_u \not\subset \mathcal{N}_u$.
5. Formation rigidity implies presence of at least two non-collinear neighbors for each agent (Laman, 1970).
6. $\delta_{ij} = 0$ if $I \neq j$ and $\delta_{ij} = 1$ otherwise.
7. Assuming the rigidity eigenvalue is unique and the framework is infinitesimally rigid (i.e. the rigidity eigenvalue is positive). We will discuss the implications of this assumption later.
8. Note that the constant α used to describe the deflated symmetric rigidity matrix is effectively replaced by k_2 in this formulation.
9. The control is shown in the x coordinate; a similar expression can be obtained for the y and z coordinates.
10. We note that, being ξ^*_i bounded, its effect does not threaten rigidity maintenance since the control action ξ_i in (10) always results dominant as $V_\lambda(\lambda_7) \to \infty$ if $\lambda_7(t) \to \lambda_7^{\min}$

11. Here, we assume that the directed edges (v_i,v_j) and (v_j,v_i) are equivalent to the undirected edge $\{v_i,v_j\}$.
12. This representation also assumes that all of the edges have been assigned a label, and this labeling is maintained even for the local graphs (local graphs do not relabel their edges; for example if edge 2 is not in local graph $\mathcal{G}_j$, then the second column of $E(\mathcal{G}_j)$ will be zero).

References

Akyildiz IF, Sankarasubramaniam Y and Cayirci E (2002) A survey on sensor networks. *IEEE Communications Magazine* 40(8): 102–114.

Anderson BDO, Fidan B, Yu C and van der Walle D (2008a) UAV formation control: Theory and application. In: Blondel VD, Boyd SP and Kimura H (eds.) *Recent Advances in Learning and Control* (Lecture Notes in Control and Information Sciences, vol. 371). New York: Springer, pp. 15–34.

Anderson BDO, Yu C, Fidan B and Hendrickx JM (2008b) Rigid graph control architectures for autonomous formations. *IEEE Control Systems Magazine* 28(6): 48–63.

Aspnes J, Eren T, Goldenberg DK, et al. (2006) A theory of network localization. *IEEE Transactions on Mobile Computing* 5(12): 1663–1678.

Baillieul J and McCoy L (2007) The combinatorial graph theory of structured formations. In: *2007 46th IEEE conference on decision and control*, pp. 3609–3615.

Bristow J, Folta D and Hartman K (2000) A formation flying technology vision. In: *AIAA space 2000 conference and exposition,* vol. 21, Long Beach, CA.

Calafiore GC, Carlone L and Wei M (2010a) A distributed Gauss–Newton approach for range-based localization of multi agent formations. In: *2010 IEEE international symposium on computer-aided control system design (CACSD)*, pp. 1152–1157.

Calafiore GC, Carlone L and Wei M (2010b) A distributed gradient method for localization of formations using relative range measurements. In: *2010 IEEE international symposium on computer-aided control system design*, Yokohama, Japan, pp. 1146–1151.

Connelly R and Whiteley WJ (2009) Global rigidity: the effect of coning. *Discrete Computational Geometry* 43(4): 717–735.

Eren T, Goldenberg OK, Whiteley W, et al. (2004) Rigidity, computation, and randomization in network localization. In: *IEEE INFOCOM* 2004, vol. 4. Piscataway, NJ: IEEE, pp. 2673–2684.

Franchi A, Masone C, Grabe V, Ryll M, Bülthoff HH and Robuffo Giordano P (2012a) Modeling and control of UAV bearing-formations with bilateral high-level steering. *The International Journal of Robotics Research* 31(12): 1504–1525.

Franchi A, Secchi C, Ryll M, Bülthoff HH and Robuffo Giordano P (2012b) Shared control: Balancing autonomy and human assistance with a group of quadrotor UAVs. *IEEE Robotics and Automation Magazine* 19(3): 57–68.

Freeman RA, Yang P and Lynch KM (2006) Stability and convergence properties of dynamic average consensus estimators. In: *45th IEEE conference on decision and control*, San Diego, CA, pp. 338–343.

Godsil CD and Royle G (2001) *Algebraic Graph Theory.* New York: Springer.

Graver J, Servatius B and Servatius H (1993) *Combinatorial Rigidity* (Graduate Studies in Mathematics). Providence, RI: American Mathematical Society.

Horn R and Johnson C (1985) *Matrix Analysis*. New York: Cambridge University Press.

Horn RA and Johnson CR (1991) *Topics in Matrix Analysis*. New York: Cambridge University Press.

Jackson B (2007) Notes on the rigidity of graphs. In: *Levico Conference Notes*.

Jacobs D (1997) An algorithm for two-dimensional rigidity percolation: the pebble game. *Journal of Computational Physics* 137(2): 346–365.

Ji M and Egerstedt M (2007) Distributed coordination control of multiagent systems while preserving connectedness. *IEEE Transactions on Robotics* 23(4): 693–703.

Krick L, Broucke ME and Francis BA (2009) Stabilisation of infinitesimally rigid formations of multi-robot networks. *International Journal of Control* 82(3): 423–439.

Laman G (1970) On graphs and rigidity of plane skeletal structures. *Journal of Engineering Mathematics* 4(4): 331–340.

Lindsey Q, Mellinger D and Kumar V (2011) Construction of cubic structures with quadrotor teams. In: *2011 robotics: science and systems*, Los Angeles, CA.

Mesbahi M and Egerstedt M (2010) *Graph Theoretic Methods in Multiagent Networks*. (Princeton Series in Applied Mathematics), 1st edn. Princeton, NJ: Princeton University Press.

Michael N, Fink J and Kumar V (2009) Cooperative manipulation and transportation with aerial robots. In: *2009 robotics: science and systems*, Seattle, WA.

Murray RM (2006) Recent research in cooperative control of multi-vehicle systems. *ASME Journal on Dynamic Systems, Measurement, and Control* 129(5): 571–583.

Olfati-Saber R and Murray RM (2002) The combinatorial graph theory of structured formations. In: *41th IEEE conference on decision and control*, Las Vegas, NV, pp. 3609–3615.

Robuffo Giordano P, Franchi A, Secchi C and Bülthoff HH (2011) Bilateral teleoperation of groups of UAVs with decentralized connectivity maintenance. In: *2011 Robotics: Science and Systems*, Los Angeles, CA.

Robuffo Giordano P, Franchi A, Secchi C and Bülthoff HH (2013) A passivity-based decentralized strategy for generalized connectivity maintenance. *The International Journal of Robotics Research* 32(3): 299–323.

Scaramuzza D, Achtelik MC, Doitsidis L, et al. (2014) Vision-controlled micro flying robots: from system design to autonomous navigation and mapping in GPS-denied environments. *IEEE Robotics and Automation Magazine* (in press).

Shames I, Fidan B and Anderson BDO (2009) Minimization of the effect of noisy measurements on localization of multi-agent autonomous formations. *Automatica* 45(4): 1058–1065.

Smith B, Egerstedt M and Howard A (2007) Automatic generation of persistent formations for multi-agent networks under range constraints. In: *1st international conference on robot communication and coordination*, pp. 1–8.

Tay T and Whiteley W (1985) Generating isostatic frameworks. *Structural Topology* 11(1): 21–69.

Williams RK, Gasparri A, Priolo A and Sukhatme GS (2014) Evaluating network rigidity in realistic systems: decentralization, asynchronicity, and parallelization. *IEEE Transactions on Robotics* (in press).

Wu C, Zhang Y, Sheng W and Kanchi S (2010) Rigidity guided localisation for mobile robotic sensor networks. *International Journal of Ad Hoc and Ubiquitous Computing* 6(2): 114.

Yang P, Freeman RA, Gordon GJ, Lynch KM, Srinivasa SS and Sukthankar R (2010) Decentralized estimation and control of graph connectivity for mobile sensor networks. *Automatica* 46(2): 390–396.

Zelazo D, Franchi A, Allgöwer F, Bülthoff HH and Robuffo Giordano P (2012) Rigidity maintenance control for multi-robot systems. In: *2012 Robotics: Science and Systems*, Sydney, Australia.

Appendix A: Index to Multimedia Extension

Archives of IJRR multimedia extensions published prior to 2014 can be found at http://www.ijrr.org, after 2014 all videos are available on the IJRR YouTube channel at http://www.youtube.com/user/ijrrmultimedia

Table of Multimedia Extension

Extension	Media type	Description
1	Video	Experiments of rigidity maintenance with a group of UAVs

Appendix B: Rigidity matrix example

The development of the alternative representation of the rigidity matrix given in Proposition 2.13 of the document is aided by a simple example. To begin, we make some qualitative observations of the rigidity matrix. For this example we consider a framework in $\mathbb{R}^2$ with the complete graph on three nodes (denoted K_3). The rigidity matrix can be written by inspection as

$$R(p) = \begin{bmatrix} p_1^x - p_2^x & p_1^y - p_2^y & p_2^x - p_1^x & p_2^y - p_1^y & 0 & 0 \\ p_1^x - p_3^x & p_1^y - p_3^y & 0 & 0 & p_3^x - p_1^x & p_3^y - p_1^y \\ 0 & 0 & p_2^x - p_3^x & p_2^y - p_3^y & p_3^x - p_2^x & p_3^y - p_2^y \end{bmatrix}$$

For the complete graph and an arbitrary orientation assigned to each edge, the incidence matrix $E(\mathcal{G})$ can be written as

$$E(\mathcal{G}) = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 0 & 1 \\ 0 & -1 & -1 \end{bmatrix}$$

The transpose of the incidence matrix functions as a "difference" operator. If the position of each agent is formed into a vector, we have

$$E(\mathcal{G})^{\mathrm{T}} \begin{bmatrix} p_1^x & p_1^y \\ p_2^x & p_2^y \\ p_3^x & p_3^y \end{bmatrix} = \begin{bmatrix} p_1^x - p_2^x & p_1^y - p_2^y \\ p_1^x - p_3^x & p_1^y - p_3^y \\ p_2^x - p_3^x & p_2^y - p_3^y \end{bmatrix}$$

The point to illustrate here is that this difference operation between positions is *redundantly embedded inside the rigidity matrix*. This fact can be made more precise by defining a *directed local graph at node v_i* from the graph $\mathcal{G}$ as in Definition 2.12 in the main text. Intuitively, the idea is that each node only has some local information about the connectivity of the entire graph; indeed, it only knows of the existence of other nodes that it can sense. In this way, we can define a sub-graph induced by each node in the graph as follows.

Let $\mathcal{G}_j = (\mathcal{V}, \mathcal{E}_j)$ be a sub-graph induced by node v_j such that

$$\mathcal{E}_j = \{(v_j, v_i) | \{v_i, v_j\} \in \mathcal{E}\}$$

Here we emphasize that the original graph $\mathcal{G}$ is undirected, while in the new induced graph $\mathcal{G}_i$ we assign a direction to the edge such that node v_j is always the tail. Furthermore, observe that $\cup_j \mathcal{G}_j = \mathcal{G}$.[11] This is illustrated in Figure 13.

To continue with the K_3 example, we can write the local incidence matrix for node v_1 as

$$E_l(\mathcal{G}_1) = \begin{bmatrix} 1 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & -1 & 0 \end{bmatrix}$$

Note that this matrix is not truly an incidence matrix for the graph $\mathcal{G}_1$; "placeholders" for the other edges in the graph $\mathcal{G}$

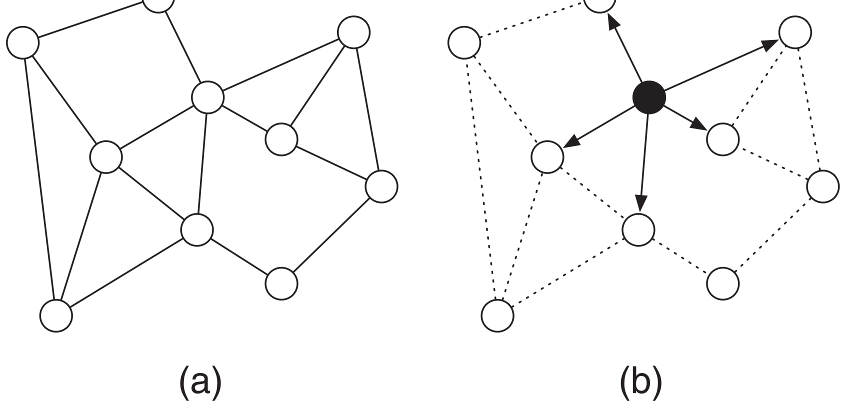

Fig. 13. Example of a directed local graph: (a) a graph; (b) local directed graph at a node.

are kept. As a result, the local incidence matrix is defined as $E_l(\mathcal{G}_j) \in \mathbb{R}^{|\mathcal{V}| \times |\mathcal{E}|}$ to have zero columns corresponding to the edges not in $\mathcal{E}_j$.[12]

Now, consider the local incidence matrix as the difference operator,

$$E_l(\mathcal{G}_1)^{\mathrm{T}} \begin{bmatrix} p_1^x & p_1^y \\ p_2^x & p_2^y \\ p_3^x & p_3^y \end{bmatrix} = \begin{bmatrix} p_1^x - p_2^x & p_1^y - p_2^y \\ p_1^x - p_3^x & p_1^y - p_3^y \\ 0 & 0 \end{bmatrix}$$

Note that this is identical to the first two columns of the rigidity matrix $R(p)$. In fact, this shows that the rigidity matrix can be written entirely in terms of local incidence matrices, as formally stated in Proposition 2.13.