Graph Rigidity and Distributed Formation Stabilization of Multi-Vehicle Systems

Reza Olfati-Saber Richard M. Murray

California Institute of Technology
Control and Dynamical Systems 107-81
Pasadena, CA 91125
{olfati,murray}@cds.caltech.edu

February 22, 2001

Abstract

In this paper, we provide a graph theoretical framework that allows us to formally define formations of multiple vehicles and the issues arising in graph realizations and unicity and their connections to stability of formations. The notion of graph rigidity is crucial in identifying the shape variables of a formation. This eventually leads to tools for formation stabilization, tacking, and formal representation of split, rejoin, and reconfiguration maneuvers for multi-vehicle formations. We introduce an algebra that consists of performing some basic operations on graphs which allow creation of larger rigid-by-construction graphs by combining smaller rigid subgraphs. This is particularly useful in performing and representing rejoin/split maneuvers of multiple formations in a distributed fashion.

1 Introduction

Coordinated control of multi-agent/multi-vehicle systems in a distributed fashion has attracted several researchers with rather backgrounds in control theory, computer science, biology, and physics. Multi-agent systems arise in broad areas including formation flight of unmanned aerial vehicles (UAVs), coordination of satellite clusters, automated highways, understanding the coordination and movement of flocks of birds or schools of fish [1], and molecular conformation problems [2].

The applications that are of primary interest in our work include performing maneuvers by UAVs which (possibly) require doing split/rejoin maneuvers in case a group of vehicles come across an obstacle, or changing the communication configuration of the network of vehicles due to the loss of line of sight and/or failure of a communication link (Fig. 1 (b)). In addition, we are interested in reconfiguration of the formation of a group of vehicles (Fig. 1 (a)) due to a change of the team-strategy in team-on-team competitive games like playing capture the flag using mobile robots known as the *RoboFlag* [3]. Competitive games of this sort can be also performed in the presence of obstacles to provide more realistic situations that might occur in uncertain environments and/or combat.

As a result, our theoretical objective is to provide analytical and computational tools for representation and manipulation of formations of multiple vehicles such as performing split, rejoin, and reconfiguration maneuvers in a distributed manner. As we realized from one of our earlier works [4], a notion from graph theory called *graph rigidity* turns out to be instrumental in both representation and distributed coordinated control of formations of multiple vehicles. Minimally rigid graphs (i.e. rigid graphs with n nodes and $2n-3$ directed edges, see section 3.3) are an important class of rigid graphs that their edges are closely related to

shape variables of formations of n vehicles. This in turn leads to automatic generation of potential functions from the interconnection graph of the group of vehicles that guarantee local *structural formation stabilization* [4].

One of the contributions of this paper is introducing new properties of minimally rigid graphs that allow composition of smaller rigid subgraphs that construct a larger rigid graph. This is used to represent and perform rejoin/split maneuvers for groups of vehicles. We introduce an *algebra over graphs* that allows performing some basic operations on graphs including rejoining two graphs, node augmentation to a graph, and attaching graphs via their edges. We presented a *hybrid system* framework for consecutive execution of a set of maneuvers for a group of agents/vehicles that allows high-level planning of operations in multi-vehicle systems (Fig. 2).

In the past, several methodologies have been exploited by different researchers in distributed control and coordination of multi-vehicle systems. In [5], graph theoretic tools were exploited to address navigational stability of formations of multiple vehicles with linear dynamics. The linearity of the dynamics of each vehicle plays an important role is applicability of the methods developed in [5]. *Distributed structural stabilization of formations* of multiple vehicles using bounded control inputs is addressed in [4]. This is done by construction of a structural potential function from a minimally rigid graph that has a unique global minimum (up to rotation, translation, and folding [4]). In [1], different types of potential functions are used. In the context of that work, a unique global minimum of the overall potential function is not desirable. Furthermore, according to [1] construction of a global potential function with a unique minimum requires adding several virtual vehicles. This complicates implementation of such a strategy in practice, since eventually the real vehicles and the virtual vehicles need to communicate and such a communication is costly in the presence of several extra virtual vehicles.

Though, the majority of the researchers have focused on formation stabilization problems, there have been almost no results available on performing rejoin, split, and reconfiguration maneuvers for groups of multiple vehicles. Formal representation of these maneuvers is one of the main contribution of this work. We also formally define what we mean by a formation in dimension $m \geq 2$ (dimension 1 is trivial).

The field of graph rigidity is very broad and several scientists and engineers from rather diverse backgrounds in mathematics, physics, chemistry, biology, computer science, and mechanical and civil engineering have been actively working on this subject over the past three decades (see [6] for a complete survey and the history of the subject that goes back to Euler in 1766 and Cauchy in 1813). Here, we are interested in *combinatorial rigidity* [7], [8] as supposed to *infinitesimal rigidity* [9], [10]. For a complete treatment of combinatorial rigidity we refer the reader to [6]. One of the first results in combinatorial characterization of rigidity of graphs in $\mathbb{R}^2$ was obtained by Laman in 1970 [7]. A *combinatorial characterization of graph rigidity for dimension 3* (and higher) has been an outstanding open problem for more than three decades. This necessitates the development of tools that allow generation of rigid-by-construction graphs in dimension 3. Here, we take a preliminary step towards this goal by developing techniques that allow creation of graphs that are rigid-by-construction and represent formations, and rejoin of multi-formations or split of a single formation to multi-formations.

A combinatorial characterization of rigidity in $\mathbb{R}^3$ is vital in problems involving conformation of molecules and solving relaxations of certain NP-hard combinatorial optimization problems [2]. Construction of molecules has broad applications in chemistry, physics, and biology.

Here is an outline of the paper: We begin with an introduction. Then, in section 2, the main motivation behind our work in the form of a hybrid system framework for consecutive operations on formations of multi-agent/multi-vehicle systems is presented. The split/rejoin maneuvers are (informally) described in section 2. The mathematical preliminaries required to discuss multi-agent formations, graph rigidity, and minimally rigid graphs are presented in section 3. Our main results on split/rejoin of groups of agents are presented in section 4. Finally, concluding remarks are made.

2 Motivation

In many applications, it is sometimes necessary to change the formation of a group of vehicles due to either a change in the coordinated task specification for the group, or the presence of uncertainty or adversarial vehicles. Reconfiguration of the formation is an example of switching from one desired formation to another. This is schematically shown in Fig. 1 (a). Another situation arises when temporarily due to the presence of an obstacle, some of the vehicles in a group cannot maintain their communication with other vehicles. Though, still all the vehicles might be interested to maintain their formation. In this situation, a communication reconfiguration is necessary among the vehicles. This is demonstrated in Fig. 1 (b). Later, we will see that both graphs in Fig. 1 (b) are rigid (see section 3.2).

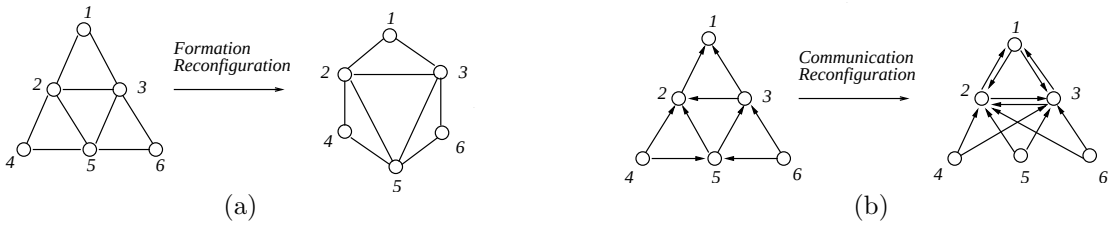

Figure 1: (a) Formation reconfiguration maneuver for a group of 6 vehicles moving in $\mathbb{R}^2$ and (b) communication reconfiguration for a group of 6 vehicles that leaves the formation unchanged.

In general, performing several types of operations on formations might be necessary. For example, performing split/rejoin maneuvers, shown in Fig. 3 (a) (see section 4 for a detailed discussion). A *Hybrid System* can be used to provide a high-level representation of a sequence of admissible operations on formations through performing certain maneuver as discussed earlier. In Fig. 2, a diagram of such a hybrid system is presented. Each discrete-state of this hybrid systems is a graph that represents the information flow (or communication configuration) among a group of vehicles. The continuous-time dynamics of the system in each discrete-state s_i is *induced* by the interconnections of the graph [4]. The hybrid system itself is a graph in which each edge of the graph is an admissible maneuver performed on a single (multiple) Formation(s).

One of our main goals in this paper is to provide a formal representation of each of these maneuvers and provide the required connections in the resulting graph(s) after performing an operation/maneuver. In section 4 an appropriate *algebra of graphs* is developed that allows to achieve our objective and facilitates performing and representing important maneuvers for formations of multiple agents.

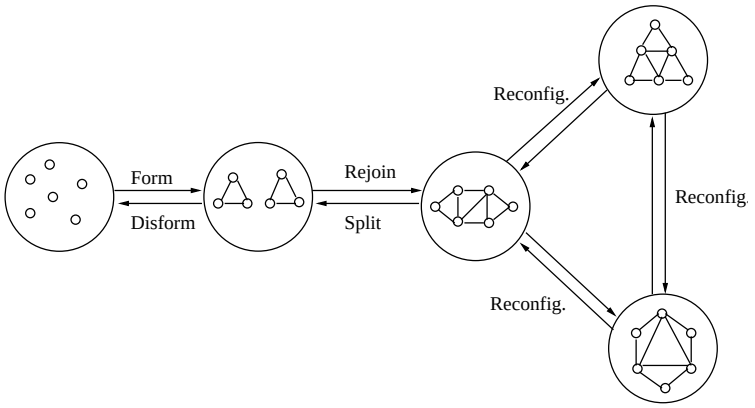

Figure 2: A Hybrid System representing the switching between multiple formations via performing a set of maneuvers.

Fig. 3 (a) demonstrates a split/rejoin maneuver for a group of vehicles that come across an obstacle. The vehicles need to *split* to avoid collision to the obstacle. After all the vehicles pass the obstacle, they need to *rejoin* and create a new desired formation. The desired path of the center of mass of the vehicles is shown in Fig. 3 (b). Here, we assume the desired attitude during the split/rejoin maneuver over the time interval $[0, T]$ is fixed.

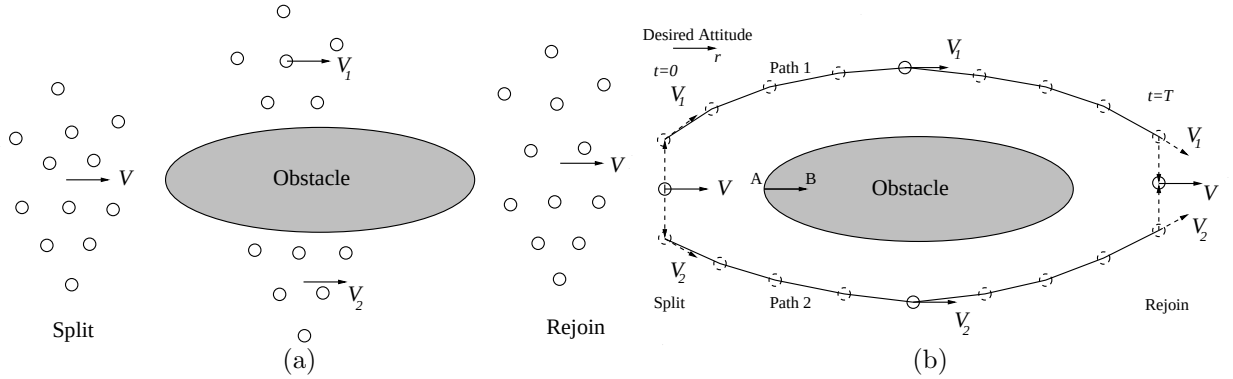

Figure 3: (a) Split/Rejoin maneuver for a group of 12 vehicles moving in $\mathbb{R}^2$ and (b) the pathes of centers of mass of two subgroups of vehicles in split/rejoin maneuver for a group of 12 vehicles moving in $\mathbb{R}^2$.

According to Fig. 3 (a), the vehicles divide into two subgroups of vehicles called *class 1* and *class 2*, respectively. Notice that the dynamics of the virtual navigation vehicle for each class is substantially different than the dynamics of an individual vehicle. To obtain the trajectory of the navigation vehicle associated with each class of vehicles, one can use either of the following three methods:

i) solving an optimization problem in the form of nonlinear trajectory generation for the dynamics of the virtual navigation vehicle [11].

ii) using the method of exact navigation by potential functions in the presence of the obstacles [12].

iii) solving a formation tracking problem for two vehicles that their center of mass has to stay on a straight line connecting the start and destination points of the maneuver, their attitude is fixed and equal to $\mathbf{r}$, and their distance d is greater than the length of the cross-section profile of the obstacle along $\mathbf{r}$.

Among all of the above, only method *iii*) would reduce the overall navigation problem for classes 1, 2 of the vehicles to a formation tracking problem for only two vehicles. This is a great reduction in computational complexity of path planning for all the vehicles. The main drawback of method *ii*) is designing the appropriate potential function that—roughly speaking—generates an acceptable path for each virtual navigation agent.

3 Preliminaries: Formations and Graph Rigidity

In this section, we provide the mathematical preliminaries that allow us to define formations of multiple agents/vehicles and its connection to graph rigidity.

3.1 Formations of Multi-Vehicles

In this section, we define a formation of n-agents, the position, and the attitude of a formation. Consider a group of n agents ($n \geq 2$) each with the following dynamics

$$\begin{cases} \dot{q}_i &= p_i \\ \dot{p}_i &= u_i \end{cases} \tag{1}$$

where $q_i, p_i, u_i \in \mathbb{R}^m$ for all $i \in \mathcal{I} = \{1, \ldots, n\}$. Therefore, each agent has a *linear dynamics*.

Remark 1. This assumption is made for the sake of presenting the main geometric and graph-theoretic ideas rather than getting involved in the technical details of dealing with nonlinear control of underactuated/nonholonomic mechanical systems. Control of formations of mobile robots that are underactuated or posses nonholonomic velocity constraints is the topic of a sequel of this paper.

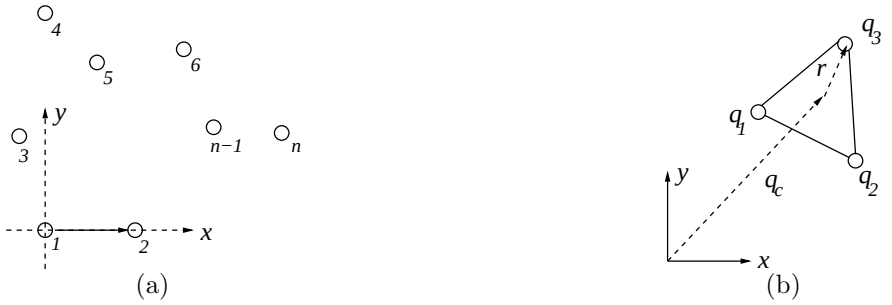

Figure 4: (a) A formation of $n \geq 2$ agents with a base $(1, 2)$ in $\mathbb{R}^2$, and (b) Position and attitude (q_c, r) of the formation of three vehicles.

We refer to a set of n points in $\mathbb{R}^m$ as an *n-grid*. The column vector $q = (q_1, \ldots, q_n)^c \in \mathbb{R}^{mn}$ is called the *configuration* of the n-grid. Identifying an agent $i \in \mathcal{I}$ by its position q_i, an agent can be viewed as a point in $\mathbb{R}^m$. Assume $\|q_2 - q_1\| > 0$ and connect the agents 1 and 2 by a directed partial-line e_{12} that is called the *base-edge* of the n-grid. An n-grid in which the distance between each two agents is greater than zero is called *collision-free*, i.e. $\|q_j - q_i\| > \gamma, \forall i, j \in \mathcal{I}, i \neq j$. Define $\gamma = \min_{1 \leq i < j \leq n} \|q_j - q_i\|$. We call γ the *safety margin* of the n-grid. Notice that in any collision-free n-grid, the safety margin is positive ($\gamma > 0$). In Fig. 4 (a), an n-grid of agents and its base-edge e_{12} is shown. For any n-grid in $\mathbb{R}^2$, a *body-axes* can be defined by taking e_{12} as the x-axis and $e_{12}^{\perp} = Te_{12}$ (read "orthogonal e_{12}") as the y-axis where T is a rotation by $\pi/2$ as the following

$$T = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Notice that for a vector $x = (x_1, x_2)^T$, $x^{\perp} = Tx = (-x_2, x_1)$. Let $\mathbf{n}(x) = x/\|x\|$ and for $q_i \neq q_j$ define

$$\mathbf{n}_{ij} = \mathbf{n}(q_j - q_i) = \frac{q_j - q_i}{\|q_j - q_i\|}$$

Then, $(\phi_1, \phi_2) = (\mathbf{n}_{12}, \mathbf{n}_{12}^{\perp})$ defines the bases of the *body-axes*. In general, in a collision-free n-grid, any two arbitrary agents can be used to define an orthonormal bases $(\mathbf{n}_{ij}, \mathbf{n}_{ij}^{\perp})$. For the special choice of e_{12} as the base-edge, the coordinates of points 1 and 2 in the body-axes are given by $(0, 0)^T$ and $(0, l)^T$ (with $l = \|q_2 - q_1\|$), respectively. l is a single degree of freedom (DOF) that determines the distance between agents $1, 2$. The position of the remaining $(n - 2)$ is each specified in the body-axes by their (x, y)-coordinates. Therefore, each remaining agent introduces 2 more degrees of freedom. Thus, the total number of degrees of freedom of an n-grid is $f = 1 + 2(n - 2) = 2n - 3$ given that $n \geq 2$. In the body-axes, an n-grid is uniquely specified by a $(2n - 3)$-dimensional vector

$$\varphi = (l, x_3, y_3, x_4, y_4, \ldots, x_n, y_n) \in Q := \mathbb{R}_{\geq 0} \times \mathbb{R}^{2n-4} \tag{2}$$

We refer to μ as the vector of *internal degrees of freedom* of an n-grid. Apparently, μ remains invariant under rotation and translation of all the points in an n-grid. Let $b \in \mathbb{R}^2$ denote the coordinates of agent 1 in the reference-frame and R be the rotation matrix by θ such that $\mathbf{n}_{12} = Re_1$ where $e_1 = (1, 0)^T$ denotes the 1st base of the reference-frame. Then $(b, R) \in SE(2)$ (which is a 3-dimensional manifold) represents

the 3 *external degrees of freedom* of an n-grid. We call (b, θ) the *navigational variables* of the n-grid. In general, any point $q_c = \sum_{i=1}^{n} \lambda_i q_i$ with a fixed set of λ_i's satisfying $\sum_{i=1}^{n} \lambda_i = 1$ and an arbitrary unit vector $\mathbf{r}$ satisfying

$$< \mathbf{r}, \mathbf{n}_{12} >= c_0 = \text{const.}$$

can be chosen to represent the *position and attitude* of the formation of an n-grid as $(q_c, \mathbf{r})$. The $(q_c, \mathbf{r})$ can be interpreted as the position and attitude of a *virtual agent* called *the navigation (virtual) agent* associated with the formation of n agents.

Remark 2. A special choice of λ_i's and c_0 is $\lambda_i = 1/n$ that gives the position of the center of mass of all agents and $c_0 = 0$ corresponding to $\mathbf{r} = \mathbf{n}_{12}^{\perp}$. Another interesting choice of the attitude $\mathbf{r}$ is $\mathbf{r} = (q_j - q_c)/\|q_j - q_c\|$ where q_c is the center of mass and agent j is an *attitude leader*.

Note. For the special case where all agents in an n-grid coincide, define $\mathbf{n}_{12} = e_1$. This case is excluded throughout the paper, unless otherwise is stated.

Definition 1. (formation) A *formation* φ of n-agents is a point on the manifold Q (defined in (2)) associated with the set of n-grids in $\mathbb{R}^2$. The *position and attitude* of a formation is defined as $\psi = (q_c, \mathbf{r})$.

Example 1. In Fig. 4 (b), an equilateral formation of three vehicles (i.e. a 3-grid) with its associated position and attitude $(q_c, \mathbf{r})$ is shown. Here, vehicle 3 is the attitude leader.

The method that is introduced here for representation of formations of planar n-grids can be directly generalized to any other dimension $m \geq 3$.

Question 1. *Is it possible to specify $f(n) = 2n - 3$ algebraic constraints in the form of the distance between the points in an n-grid in $\mathbb{R}^2$ that uniquely determines the formation associated with the n-grid?*

The correct answer depends on what we mean by uniqueness. Later on, we rephrase and answer Question 1 in the context of graph theory (section 3.2).

3.2 Graph Rigidity

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ be a weighted graph with the set of vertices $\mathcal{V} = \{v_1, \ldots, v_n\}$ (i.e. $|\mathcal{V}| = n$), the set of edges $\mathcal{E}$, and the set of weights $\mathcal{W}$. In addition, define $\mathcal{I} = \{1, 2, \ldots, n\}$ as the set of indices of the element of $\mathcal{V}$. Each agent in a multi-agent system can be viewed as a node of the graph $\mathcal{G}$ which represents the overall system.

Remark 3. Throughout this paper, we assume that controller of the multi-agent system is *distributed*. This means that each agent performs *sensing and communication* with all of its *neighbors* $J_i := \{j \in \mathcal{I} : e_{ij} \in \mathcal{E}\}$ in a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. As a special case, this definition of a neighbor includes the case of *spatial neighbors* of an agent that are located within a distance $d > 0$ of each agent (see [1]).

Let $q_i \in \mathbb{R}^m$ denote the coordinates vector assigned to node v_i of the graph. Then $q = (q_1, \cdots, q_n)^c \in \mathbb{R}^{mn}$ is called a *realization* of $\mathcal{G}$ iff

$$\|q_j - q_i\| = w_{ij}, \ \forall e_{ij} \in \mathcal{E}, q_i, q_j \in \mathbb{R}^m$$

where $\mathcal{W} = \{w_{ij}\}, \mathcal{E} = \{e_{ij}\}$. The pair $(\mathcal{G}, q)$ is called a *framework*. An *infinitesimal motion* is an assignment of a velocity vector p_i to the vertex v_i of the graph $\mathcal{G}$ such that

$$\langle p_j - p_i, q_j - q_i \rangle = 0, \quad \forall e_{ij} \in \mathcal{E} \tag{3}$$

where $\langle \cdot, \cdot \rangle$ denotes the inner product. A *flexing* of a framework $(\mathcal{G}, q)$ is a family of realizations of $\mathcal{G}$ parameterized by t, $q(t) : [0, 1] \to \mathbb{R}^{mn}$ such that $q(0) = q$ and $q(t)$ is a differentiable function of t. Let $p(t) = \dot{q}$ be the vector of velocities assigned to each node. Then, a flexing $q(t)$ satisfies the following relation

$$\langle p_j(t) - p_i(t), q_j(t) - q_i(t) \rangle = 0, \quad \forall e_{ij} \in \mathcal{E}, \forall t \in [0, 1] \tag{4}$$

Clearly, the rigid motions of $\mathbb{R}^m$ are length preserving and $p_i = p_i(t)$ defines an infinitesimal motion of the family of graphs $\mathcal{G}(t)$ with realization $q(t)$ where $t \in [0, 1]$. These rigid motions are called *trivial* flexings of a framework. A framework $(\mathcal{G}, q)$ is called *infinitesimally rigid*, iff the only infinitesimal motions of the framework are trivial motions. In other words, rigidity of a graph $\mathcal{G}$ concerns answering to the following question:

Question 2. *Consider a weighted graph $\mathcal{G}$ that can be embedded in $\mathbb{R}^m$. Is the realization of $\mathcal{G}$ unique up to a congruence (i.e. rotation and translation)?*

Apparently, Question 1 is a special case of question Question 2 in which the graph has $|\mathcal{E}| = f(m, n)$ edges. It turns out that rigidity of graphs is a generic property in the sense that almost all realizations of a particular graph are either infinitesimally rigid, or flexible [9]. Thus, rigidity is a *generic* property of graphs. This eliminates the necessity to check the rigidity property for all possible realizations of a graph.

Note. In the context of *rigidity* of graphs, it is assumed that *all the edges are directed* and the graph has neither undirected edges, nor edges from one node to itself.

Throughout this paper, we assume each agent performs *sensing and communication* with all of its neighbors that are defined as the following:

Definition 2. (neighbors) The indices of the *neighbors* of the node v_i in the graph $\mathcal{G} = (V, \mathcal{E})$ is denoted by J_i and defined as $J_i := \{j \in \mathcal{I} : e_{ij} \in \mathcal{E}\}$.

A combinatorial characterization of rigidity of graphs in $\mathbb{R}^2$ was first obtained by Laman [7]. First, we need to define the Laman subgraph of a graph.

Definition 3. (Laman subgraph) A *Laman subgraph* of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a graph $\mathcal{H} = (\mathcal{V}_\mathcal{H}, \mathcal{E}_\mathcal{H})$ such that $\mathcal{V}_\mathcal{H} \subset \mathcal{V}$, $|\mathcal{V}_\mathcal{H}| \geq 2$, and $\mathcal{E}_\mathcal{H} = \mathcal{E}|_{\mathcal{V}_\mathcal{H}} := \{e_{ij} \in \mathcal{E} : v_i, v_j \in \mathcal{V}_\mathcal{H}\}$ (We read $\mathcal{E}_\mathcal{H}$ is the restriction of $\mathcal{E}$ to $\mathcal{V}_\mathcal{H}$)

Theorem 1. *(Laman, 1970 [7]) A planar graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $n \geq 2$ nodes is rigid iff there exists a subset $\mathcal{E}_\mathcal{H} \subset \mathcal{E}$ of $2n - 3$ edges of $\mathcal{G}$ such that for the graph $\mathcal{H} = (\mathcal{V}, \mathcal{E}_\mathcal{H})$ with n nodes, each Laman subgraph $Y = (\mathcal{V}_Y, \mathcal{E}_Y)$ of $\mathcal{H}$ satisfies the property $|\mathcal{E}_Y| \leq 2|\mathcal{V}_Y| - 3$.*

Definition 4. (essential subgraph) We refer to $\mathcal{H}$ in Theorem 1 as the *essential subgraph* of the rigid graph $\mathcal{G}$.

Definition 5. (independent/redundant edges) An edge e of a rigid graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is called *redundant*, iff after removing e from $\mathcal{E}$, the graph remains rigid. Otherwise, e is called an *independent* edge.

According to Laman's Theorem, *any planar rigid graph $\mathcal{G}$ ($n \geq 2$) has at least $2n - 3$ edges.* Laman's Theorem was later generalized by Lovász and Yemini in [8]. Based on their work, in [13] an $O(n^2)$ algorithm is give that determines whether a graph is rigid or not. Another ($O(n^{1.15})$ algorithm (worst case $O(n^2)$) is presented in [14] as a test for graph rigidity.

3.3 Minimally Rigid Graphs

Any rigid graph $\mathcal{G}$ with $n \geq 2$ nodes and $2n - 3$ edges is called a *minimally rigid graph* (MRG). Apparently, any MRG is *the essential subgraph of itself.* In addition, *every edge of an MRG is independent.*

Due to computational and communications costs in a network of n-vehicles, we are interested in the least possible number of edges between the agents that creates a rigid graph and thus a locally stabilizing distributed control law for each vehicle [4]. This makes minimally rigid graphs the ideal choice for us. Moreover, it will become clear later that MRGs benefit from nice analytic properties that allow one to construct bigger graphs through connecting minimally rigid subgraphs. This is explained in complete details in section 4.

The edges of a minimally rigid graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ define the following set of *shape variables* for the graph:

$$\eta_{ij} := \|q_j - q_i\| - w_{ij}, \quad \forall e_{ij} \in \mathcal{E} \tag{5}$$

We call the column vector η and manifold $Q(\mathcal{G})$ defined by

$$\eta = \{\eta_{ij}\} \in Q(\mathcal{G}) := \Pi_{e_{ij}\in\mathcal{E}}[-w_{ij}, \infty) \subset \mathbb{R}^{2n-3} \tag{6}$$

as the *shape configuration* and *shape manifold* of $\mathcal{G}$. *Any point at the boundary of $Q(G)$ corresponds to a collision between two agents.* The *structural potential function* of the graph $\mathcal{G}$ is defined as a smooth, proper, and positive definite function $V(\eta)$ that satisfies $V(0) = 0$. Two examples of $V(\eta)$ (or $V(q)$) are given in [4] as the following:

$$\begin{array}{rcl} V_1(\eta) & = & \sum_{e_{ij}\in\mathcal{E}} \eta_{ij}^2 \\ V_2(\eta) & = & \sum_{e_{ij}\in\mathcal{E}} [(1+\eta_{ij}^2)^{\frac{1}{2}} - 1] \end{array} \tag{7}$$

Clearly, $V_2(q) := V_2(\eta)$ has a bounded gradient w.r.t. q and this is the key in designing a bounded control input for structural formation stabilization [4].

4 Main Results: Rejoin/Split of Formations of Multi-Agents

To study and manipulate rejoin and split of formations of multiple agents, we need to formally define some basic operations on graphs that preserve the rigidity properties of the obtained graphs associated with rejoin/split maneuvers in multi-agent formations. For doing so, we discuss the problem of joining two rigid graphs to construct a new composite rigid graph. This operation is called *rejoining graphs*. Later, we discuss how a rigid graph can be decomposes to two sets of disjoint graphs that are both rigid. The second operation is called *splitting a graph*. These basic operations (together with some other operations) constitute a novel *graph algebra*. This graph algebra greatly facilitates representation of the rejoin operation of graph that is more complicated than the split operation.

4.1 Node Augmentation to Graphs

A key ingredient in formal definition of rejoin/split of rigid graphs is the *node augmentation* operation. Before, we describe this operation, we need to define proper Laman subgraphs and present an axiom.

Definition 6. (proper Laman subgraphs) A Laman subgraph $Y = (\mathcal{V}_Y, \mathcal{E}_Y)$ of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ satisfying the property $|\mathcal{E}_Y| \leq 2|\mathcal{V}_Y| - 3$ is called a *proper Laman subgraph*.

Axiom 1. *(single node rigidity) Any graph with a single node is rigid in $\mathbb{R}^m$ for dimension $m \geq 1$ is minimally rigid.*

Definition 7. (node augmentation for single node graphs) Consider two single node graphs $\mathcal{G}_1 = (\{v_1\}, \emptyset, \emptyset)$ $\mathcal{G}_2 = (\{v_2\}, \emptyset, \emptyset)$, the *node augmentation* is an operation that creates a graph $\mathcal{G} = (\{v_1, v_2\}, \{e_{21}\}, \{d_{21}\})$ and is denoted by

$$\mathcal{G} := \mathcal{G}_1 \oplus \mathcal{G}_2 \tag{8}$$

where $d_{21} = \|e_{21}\|$ and $\|e\|$ denotes the length of an edge e. By a slight abuse of notation, we write $\mathcal{G}_1 = \{v_1\}, \mathcal{G}_2 = \{v_2\}$ and then $\mathcal{G}$ can be rewritten as

$$\mathcal{G} := \{v_1\} \oplus \{v_2\}$$

Notice that a graph with two nodes and one edge is minimally rigid (due to Laman's theorem). Before we define the node augmentation for graphs with multiple nodes, we need to define the union of two graphs.

Definition 8. (union) Let $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1, \mathcal{W}_1)$ and $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2, \mathcal{W}_2)$ be two weighted graphs. The *union* of $\mathcal{G}_1$ and $\mathcal{G}_2$ is a graph

$$\mathcal{G} = \mathcal{G}_1 \cup \mathcal{G}_2 := (\mathcal{V}_1 \cup \mathcal{V}_2, \mathcal{E}_1 \cup \mathcal{E}_2, \mathcal{W}_1 \cup \mathcal{W}_2)$$

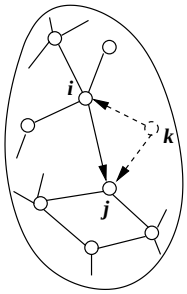

Figure 5: Augmentation of the node k at the edge (i, j) of a multi-node graph.

Definition 9. (node augmentation for multi-node graphs) Consider a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with $n \geq 2$ nodes and let $v_k \notin \mathcal{V}$ be the vertex of a single node graph $\mathcal{G}_0 = (\{v_k\}, \emptyset, \emptyset)$ with no edges. By a slight abuse of notation we denote $\mathcal{G}_0 = \{v_k\}$. Let $q_1, \ldots, q_n$ be a realization of $\mathcal{G}$ and assume q_k does not coincide with any vertices of $\mathcal{G}$, i.e. $\|q_k - q_i\| > 0$, for $i = 1, \ldots, n$. The *node augmentation* at (v_i, v_j) (i.e. an unordered pair of distinct vertices) is an operation on two graphs with n nodes and one node that creates a new graph $\mathcal{G}_a = (\mathcal{V}_a, \mathcal{E}_a, \mathcal{W}_a)$ with $n + 1$ nodes and $|\mathcal{E}_a| = |\mathcal{E}| + 2$ edges defined as the following

$$\mathcal{V}_a = \mathcal{V} \cup \{v_k\},\ \mathcal{E}_a = \mathcal{E} \cup \{e_{ki}, e_{kj}\},\ \mathcal{W}_a = \mathcal{W} \cup \{\|e_{ki}\|, \|e_{kj}\|\}, \tag{9}$$

where $\|e\|$ denotes the length of the edge e. The node augmentation operation is denoted by

$$\mathcal{G}_a := \mathcal{G}|_{(v_i, v_j)} \oplus \{v_k\} \tag{10}$$

whenever an edge $e_{ij} \in \mathcal{E}$, this operation is equivalently expressed as

$$\mathcal{G}_a := \mathcal{G}|_{e_{ij}} \oplus \{v_k\} \tag{11}$$

The operation of node-augmentation is shown in Fig. 5. In this figure, the augmented node and edges are all drawn by dashed lines.

Definition 10. (a triangular graph) A *triangular graph* $\mathcal{T}$ is a graph with three nodes, i. e. $\mathcal{V} = \{v_1, v_2, v_3\}$, and a set of three directed edges $\mathcal{E}_\mathcal{T} \subset \mathcal{E} = \{e_{12}, e_{21}, e_{13}, e_{31}, e_{23}, e_{32}\}$.

A (special) triangular graph can be created using two consecutive node-augmentations as the following:

$$\mathcal{T} = (\{v_1\} \oplus \{v_2\}) \oplus \{v_3\}$$

Since the graph $\{v_1\} \oplus \{v_2\}$ only has a single edge, the edge location in the last equation is dropped.

Remark 4. The operation of augmentation of a node v_k at the pair (v_i, v_j) can be equivalently represented by union of two graphs as follows. Let

$$\mathcal{T} = \mathcal{T}(v_k, v_i, v_j) := (\{v_k, v_i, v_j\}, \{e_{ki}, e_{kj}\}, \{\|e_{ki}\|, \|e_{kj}\|\} \tag{12}$$

denote a partial triangular graph with two edges and three nodes. Then $\mathcal{G}_a$, defined in (10), can be expressed as $\mathcal{G}_a = \mathcal{G} \cup \mathcal{T}$.

Here is our first result on construction of rigid graphs via node augmentation.

Theorem 2. *Any rigid graph $\mathcal{G}$ in $\mathbb{R}^2$ remains rigid after node augmentation, i.e. node augmentation preserves rigidity in $\mathbb{R}^2$.*

Proof. See section A.1 in the Appendix. □

Corollary 1. *Any minimally rigid graph $\mathcal{G}$ in $\mathbb{R}^2$ remains minimally rigid after node augmentation.*

Proof. According to Theorem 2, the augmented graph $\mathcal{G}_a$ is rigid and has $|\mathcal{E}_{G_a}| = (2n-3)+2 = 2(n+1)-3$ edges. Thus, $\mathcal{G}_a$ satisfies the relation $|\mathcal{E}_{G_a}| = 2|\mathcal{V}_{G_a}| - 3$. The fact that G_a is an essential subgraph of itself follows from the proof of Theorem 2 and that establishes every Laman subgraph Y_a of $\mathcal{G}_a$ is proper. □

Corollary 2. *The augmented edges in node augmentation of a minimally rigid graph are independent edges of the obtained graph.*

Proof. This follows from the fact that the obtained graph after node augmentation is minimally rigid and all of its edges are independent.

The following corollary provides a simple proof of rigidity for minimally rigid graphs.

Corollary 3. *Any graph that can be constructed by an ordered sequence of node augmentations starting from a single node is minimally rigid.*

Example 2. Here is a sequence of nodes and edges that create a (minimally) rigid graph shown in Fig. 6 (a), (b).

$$\begin{aligned}
\mathcal{G}_a: &\quad 1; 2,(2,1); 3,(3,1),(3,2); 4,(4,2),(4,3); 5,(5,2),(5,4);\\
&\quad 6,(6,4),(6,3).\\
\mathcal{G}_b: &\quad 1; 2,(2,1); 3,(3,1),(3,2); 4,(4,2),(4,3); 5,(5,3),(5,4);\\
&\quad 6,(6,4),(6,5); 7,(7,5),(7,6).\\
\mathcal{G}_c: &\quad 1; 2,(2,1); 3,(3,1),(3,2); 4,(4,2),(4,3).\\
\mathcal{G}_d: &\quad 1; 2,(2,1); 3,(3,1),(3,2); 4,(4,2),(4,3); 5,(5,3),(5,4).
\end{aligned} \tag{13}$$

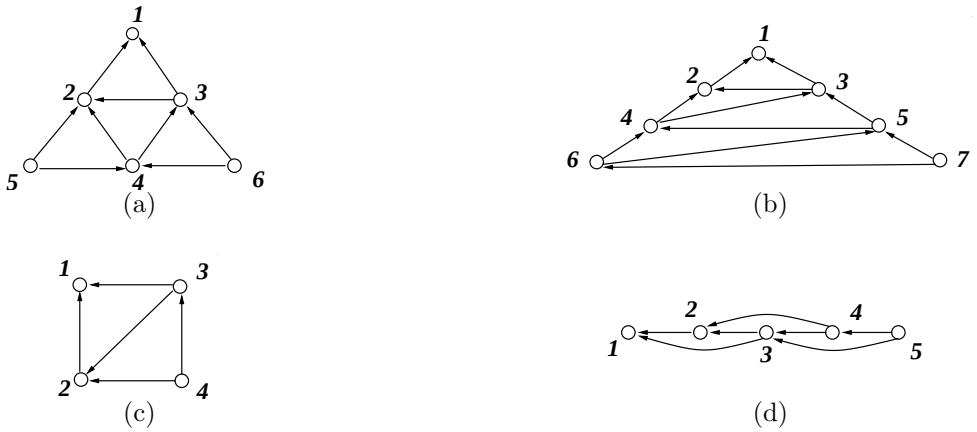

Figure 6: (a) A rigid graph $\mathcal{G}_a$ with $n = 6$ nodes and $n_e = 9$ edges, (b) A rigid graph $\mathcal{G}_b$ representing a V-formation of $n = 7$ vehicles with $n_e = 11$ edges, (c) a square graph $\mathcal{G}_c$ with a diagonal edge representing a diamond formation of four vehicles, and (d) A rigid graph $\mathcal{G}_d$ with $n = 5$ nodes on a line representing a platoon of five vehicles.

4.2 Rejoining Two Graphs

As described earlier in section 2, the *rejoin operation* of two subgroups of vehicles is very important after they pass an obstacle (see Figures 3 (a), (b)). In this section, we present the graph theoretical tools/operations that create a rigid composite graph by combining the graphs associated with each subgroup of agents/vehicles.

Let $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1, \mathcal{W}_1)$ and $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2, \mathcal{W}_2)$ be two graphs with disjoint set of vertices $\mathcal{V}_1 \cap \mathcal{V}_2 = \emptyset$ and $n_1 \geq 2$ and $n_2 \geq 2$ nodes, respectively. We say $\mathcal{G}_1$ and $\mathcal{G}_2$ are *edge-attachable* at $e_{ij} = (v_i, v_j) \in \mathcal{E}_1$ and $e_{kl} = (v_k, v_l) \in \mathcal{E}_2$ iff e_{ij} and e_{kl} have the equal lengths, i.e. $\|e_{ij}\| = \|e_{kl}\|$. Before describing the operation of edge-attachment of two graphs, we need to introduce three basic operations on graphs. Namely, the operations of renaming of vertices, edge-addition, and edge-subtraction in a graph.

Definition 11. (single-node renaming) Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and $v_k \notin \mathcal{V}$ be an extra node. The operation of *renaming* $v_i \in \mathcal{V}$ *in* $\mathcal{G}$ *by* v_k means creating a new graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ such that $\mathcal{V}' = (\mathcal{V} \setminus \{v_i\}) \cup \{v_k\}$ and $\mathcal{E}' = (\mathcal{E} \setminus \mathcal{E}_i) \cup \mathcal{E}_k$ where

$$\begin{array}{rcl} \mathcal{E}_i & = & \{e \in \mathcal{E} : e = (v_i, v_j) \vee e = (v_j, v_i)\}, \\ \mathcal{E}_k & = & \{(v_i, v_k) : (v_i, v_j) \in \mathcal{E}\} \cup \{(v_k, v_i) : (v_j, v_i) \in \mathcal{E}\} \end{array} \tag{14}$$

Definition 12. (multi-node renaming) Similarly, more than one node in a graph can be renamed. The operation of renaming an ordered list of nodes $v_1, v_2, \ldots, v_m$ in $\mathcal{G}$ by $v'_1, v'_2, \ldots, v'_m$ is called *multi-node renaming* and is denoted by

$$\mathcal{G}' = \mathrm{Ren}(\mathcal{G}; v_1, v_2, \ldots, v_m | v'_1, v'_2, \ldots, v'_m)$$

This operation means creating a new graph $\tilde{\mathcal{G}}$ iteratively as the following

$$\begin{array}{rcl} \mathcal{G}^0 & := & \mathcal{G} \\ \mathcal{G}^k & := & \mathrm{Ren}(\mathcal{G}^{k-1}; v_k | v'_k), k = 1, \ldots, m \\ \mathcal{G}' & := & \mathcal{G}^m \end{array} \tag{15}$$

With a minor abuse of notation, the renaming operation can be applied to the set of edges of a graph instead of the whole graph.

Definition 13. (edge-addition/subtraction) Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph and let $\mathcal{G}^+ := (\mathcal{V}, \mathcal{E} \cup \{e\})$. Then the operation $[\cdot]_e^+$ on a graph defined as $\mathcal{G}^+ = [\mathcal{G}]_e^+$ is called *edge-addition*. Similarly, let $\mathcal{G}^- := (\mathcal{V}, \mathcal{E} \setminus \{e\})$. Then, the operation $[\cdot]_e^-$ on a graph defined by $\mathcal{G}^- = [\mathcal{G}]_e^-$ is referred to as *edge-subtraction*. Notice that for weighted graphs, edge addition (or subtraction) operation adds (or subtracts) the element $w = \|e\|$ to (from) the set of weights $\mathcal{W}$.

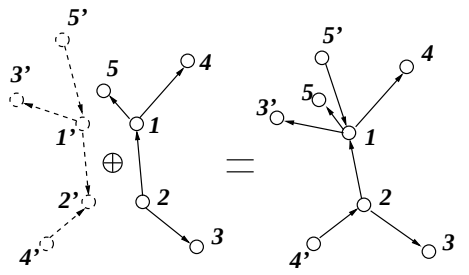

Figure 7: Edge-attachment between two graphs at the edges $(1, 2)$ and $(1,', 2')$ with equal length.

Definition 14. (edge-attachment) Let $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1, \mathcal{W}_1)$ and $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2, \mathcal{W}_2)$ be two graphs that are edge-attachable at $e_{ij} \in \mathcal{E}_1$ and $e_{kl} \in \mathcal{E}_2$. Define

$$\mathcal{G}_2^- = [\mathcal{G}_2]_{e_{kl}}^- := (\mathcal{V}_2, \mathcal{E}_2 \setminus \{e_{kl}\}, \mathcal{W}_2 \setminus \{w_{kl}\}),$$

and let $\mathcal{G}'_2 = \mathrm{Ren}(\mathcal{G}_2^-; v_k, v_l | v_i, v_j)$. Then, the *edge-attachment operation* of $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2)$ to $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$ at $e_{ij} \in \mathcal{E}_1$ is defined as

$$\mathcal{G} := \mathcal{G}_1 \cup \mathcal{G}'_2 \tag{16}$$

and denoted by

$$\mathcal{G} = \mathcal{G}_1|_{e_{ij}} \oplus \mathcal{G}_2|_{e_{kl}} \tag{17}$$

We call $\mathcal{G}$ the *attachment* (graph) of $\mathcal{G}_1$ and $\mathcal{G}_2$. The overall edge-attachment operation can be equivalently expressed as follows

$$\mathcal{G}_1|_{e_{ij}} \oplus \mathcal{G}_2|_{e_{kl}} := \mathcal{G}_1 \cup \mathrm{Ren}([\mathcal{G}_2]^-_{e_{kl}}; v_k, v_l | v_i, v_j) \tag{18}$$

Notice that the edge-attachment operation that is shown schematically in Fig. 7 creates a new graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathcal{W})$ with $\mathcal{V} = \mathcal{V}_1 \cup (\mathcal{V}_2 \setminus \{v_k, v_l\})$, $\mathcal{E} = \mathcal{E}_1 \cup \mathrm{Ren}(\mathcal{E}_2 \setminus \{e_{kl}\}; v_k, v_l | v_i, v_j)$, and $\mathcal{W} = \|\mathcal{E}\|$. Furthermore, the obtained graph $\mathcal{G}$ has $n = (n_1 + n_2) - 2$ nodes ($n \geq 2$) and $|\mathcal{E}| = |\mathcal{E}_1| + |\mathcal{E}_2| - 1$ edges.

Example 3. Consider two triangular graphs $\mathcal{T}_1 = (\{v_1, v_2, v_3\}, \{e_{12}, e_{23}, e_{31}\})$ and $\mathcal{T}_2 = (\{v_4, v_5, v_6\}, \{e_{45}, e_{56}, e_{64}\})$ and let $\|e_{23}\| = \|e_{56}\|$ then the graph $\mathcal{G} = \mathcal{T}_1|_{e_{23}} \oplus \mathcal{T}_2|_{e_{56}}$ is a graph with the set of vertices $\mathcal{V} = \{v_1, v_2, v_3, v_4\}$ and the set of edges $\mathcal{E} = \{e_{12}, e_{23}, e_{31}, e_{42}, e_{34}\}$.

Our main motivation to define the operation of edge-attachment of two weighted graphs is the following result.

Theorem 3. *(edge-attachment) Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be two rigid graphs with disjoint set of vertices. Assume that the corresponding essential subgraphs $\mathcal{H}_1$ and $\mathcal{H}_2$ of these graphs possess attachable edges $e_{ij} \in \mathcal{E}_{\mathcal{H}_1}$ and $e_{kl} \in \mathcal{E}_{\mathcal{H}_2}$. Then the attachment $\mathcal{G} = \mathcal{G}_1|_{e_{ij}} \oplus \mathcal{G}_2|_{e_{kl}}$ is a rigid graph.*

Proof. See section A.2 in the Appendix. □

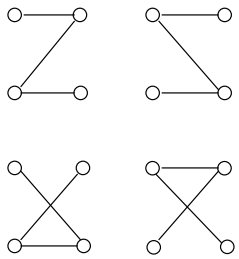

Figure 8: A Z-link in four possible configuration of edges.

Definition 15. (Z-link) We refer to a *bipartite graph* $\mathcal{K}_{2,2}$ with three edges, shown in Fig. 8, as a *Z-link*.

Remark 5. The name "Z-link" comes from the shape of the graph in the upper left corner of Fig. 8.

Definition 16. (rejoining two graphs) The operation of *rejoining two graphs* $\mathcal{G}_1 = (\mathcal{V}_1, \mathcal{E}_1)$ and $\mathcal{G}_2 = (\mathcal{V}_2, \mathcal{E}_2)$ via a *Z-link*, $Z = (\mathcal{V}_Z, \mathcal{E}_Z)$, means creating a new graph

$$\mathcal{G} = \mathcal{G}_1|_{e_1} \oplus_Z \mathcal{G}_2|_{e_2} \tag{19}$$

where $\mathcal{G} = \mathcal{G}_1 \cup Z \cup \mathcal{G}_2$ and $\mathcal{V}_Z = \mathcal{V}(e_1) \cup \mathcal{V}(e_2)$, i.e. the vertices of the Z-link are the end-points of two edges e_1 and e_2. The joining operation is schematically shown in Fig. 9. The edges that are removed after edge-attachment are shown with dashed lines.

Here is an important result that states how two graphs need to be connected such that the obtained graph after connection is rigid. This is particularly useful in rejoin maneuvers of two formations of multiple vehicles.

Theorem 4. *Let $\mathcal{G}_1, \mathcal{G}_2$ be two rigid graph each with more than two nodes. Assume e_1, e_2 are edges of essential subgraphs of $\mathcal{G}_1, \mathcal{G}_2$, respectively. Then, the connection of $\mathcal{G}_1$ and $\mathcal{G}_2$ via a Z-link given by $\mathcal{G} = \mathcal{G}_1|_{e_1} \oplus_Z \mathcal{G}_2|_{e_2}$ is a rigid graph.*

Proof. See section A.3 in the Appendix. □

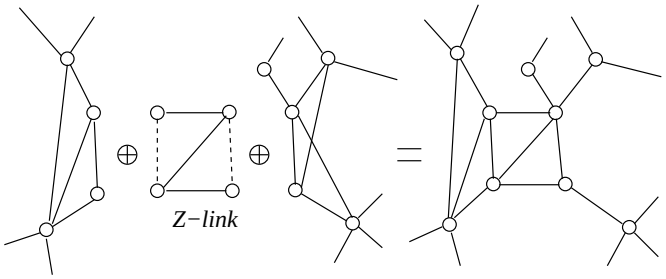

Figure 9: Joining two graphs using a Z-link (here the direction of the edges can be chosen arbitrarily).

Corollary 4. *Two minimally rigid graphs that are connected using a Z-link construct a minimally rigid graph.*

Example 4. In Fig. 2, two triangular formations of vehicles are rejoined using a Z-link to create a formation of six vehicles with a minimally rigid interconnection graph.

4.3 Splitting of A Graph

In section 2, the *split operation* of a group of vehicles into two subgroups is discussed. This situation occurs when a group of vehicles come across an obstacle (see Figures 3 (a), (b)). In the splitting maneuver, we first classify the vehicles into two subgroups. Then, create a minimally rigid graph for n_1-grid and n_2-grid ($n_1 + n_2 = n$) associated with each subgroup (or class). The latter procedure is performed using successive node-augmentations or rejoining two subgraphs using Z-links.

The classification of the vehicles into two subgroups can be performed as follows. Let point A be the tip of the obstacle, meaning that if we move a line orthogonal to the attitude $\mathbf{r}$ along the direction of $\mathbf{r}$, it would first intersect the obstacle at point A. Let $A = (x_1, x_2)^T$. Define the location of the point $B = (y_1, y_2)^T$ as $B = A + \mathbf{r}$. Then the directed partial-line AB is parallel to the desired attitude $\mathbf{r}$ (see Fig. 3 (b)). Define the following *triangular orientation function*

$$h(x, y, z) := \det \begin{bmatrix} x_1 & x_2 & 1 \\ y_1 & y_2 & 1 \\ z_1 & z_2 & 1 \end{bmatrix} \tag{20}$$

that is widely used is computer vision and computational geometry. For all $i \in \{1, \ldots, n\}$, set

$$\mathrm{Ori}(q_i) := \begin{cases} \text{“right”} & \text{if } h(x, y, q_i) \leq 0 \\ \text{“left”} & \text{if } h(x, y, q_i) > 0 \end{cases} \tag{21}$$

If $\mathrm{Ori}(q_i) =$ “left”, then vehicle i is a member of class 1 and takes the left path around the obstacle, otherwise it is a member of class 2 and takes the right path around the obstacle. This is a *distributed algorithm* for splitting a group of vehicles into left and right subgroups. In the sense that all the agents have to come to a *consensus* [15, Chap. 12] regarding the values of $\mathbf{r}, (x_1, x_2)^T$. Further study of the consequences of the uncertainties and delays created by solving such a consensus problem in an asynchronous fashion [15, Chap. 14] on the separation of the two subgroups of the vehicles is the topic of future research. We have presented several examples. All the aforementioned methods are implemented and simulation results for split, rejoin, and reconfiguration of formations are available upon request (even in the form of movies) and will be included in the final presentation of the paper.

5 Conclusion

In this paper, we provide a unified graph-theoretical framework that allows us to formally define formations of multiple vehicles and their stabilization issues. We clarified the important role of graph rigidity and minimally rigid graphs in construction of structural potential functions and manipulation of multiple formations. This includes formal representation of split, rejoin, and reconfiguration maneuvers for formations of multi-vehicle systems. We presented a hybrid system framework for consecutive execution of a set of maneuvers for a group of agents/vehicles that allows high-level planning of operations in multi-vehicle systems. We introduce an algebra that formalizes performing some basic operations on graphs and allows creation of larger rigid-by-construction graphs by combining smaller rigid subgraphs (the size is measured in terms of the number of nodes in the graph). This is particularly useful in performing and representing rejoin/split maneuvers of multiple formations in a distributed fashion.

A Appendix

A.1 Proof of Theorem 2

The result is obvious if $\mathcal{G}$ is a single node graph. Therefore, assume $\mathcal{G}$ is a multi-node rigid graph. Based on Laman's Theorem, there exists an essential subgraph $\mathcal{H}$ of $\mathcal{G}$ with $n \geq 2$ nodes and $|\mathcal{E}_\mathcal{H}| = 2n-3$ edges such that each Laman subgraph Y of $\mathcal{H}$ satisfies $|\mathcal{E}_Y| \leq 2|\mathcal{V}_Y| - 3$. Define an augmented graph

$$\mathcal{H}_a := \mathcal{H}|_{(v_i,v_j)} \oplus \{v_k\}$$

with $n+1$ nodes and $|\mathcal{E}_{H_a}| = (2n-3)+2 = 2(n+1)-3$ edges, i.e. the relation $|\mathcal{E}_{\mathcal{H}_a}| = 2|\mathcal{V}_{\mathcal{H}_a}|-3$ is satisfied for $\mathcal{H}_a$). We prove that $\mathcal{H}_a$ is an essential subgraph of $\mathcal{G}_a$. Every Laman subgraph Y_a of $\mathcal{H}_a$ either contains v_k or not. In the latter case, $Y_a = Y$ where Y is a proper Laman subgraph of $\mathcal{H}$ and thus the relation $|\mathcal{E}_{Y_a}| \leq 2|\mathcal{V}_{Y_a}|-3$ holds. In the former case where v_k is a node of Y_a, the subgraph Y_a can be decomposed as

$$Y_a = Y|_{(v_i,v_j)} \oplus \{v_k\}$$

which implies $|\mathcal{E}_{Y_a}| = |\mathcal{E}_Y| + 2$ and $|\mathcal{V}_{Y_a}| = |\mathcal{V}_Y| + 1$, thus

$$\begin{array}{rcl} |\mathcal{E}_{Y_a}| & = & |\mathcal{E}_Y| + 2 \\ & \leq & (2|\mathcal{V}_Y| - 3) + 2 \\ & = & 2|\mathcal{V}_Y| - 1 \\ & = & 2|\mathcal{V}_{Y_a}| - 3 \end{array} \tag{22}$$

and this proves Y_a is a proper subgraph of $\mathcal{H}_a$. Therefore, $\mathcal{H}_a$ is an essential subgraph of $\mathcal{G}_a$ and $\mathcal{G}_a$ is rigid. □

A.2 Proof of Theorem 3

The set of edges $\mathcal{E}$ of $\mathcal{G}$ is the union of two disjoint set of edges $\mathcal{E}_1$ and $\mathcal{E}_2'$ where $\mathcal{E}_2'$ is the set of edges of $\mathcal{G}_2' = \text{Ren}([\mathcal{G}_2]^-_{e_2}; v_k, v_l | v_i, v_j)$. By definition of an essential subgraph, it follows that $\mathcal{H}_1, \mathcal{H}_2$ are graphs with n_1, n_2 nodes and $m_1 = 2n_1 - 3, m_2 = 2n_2 - 3$ edges, respectively. Define $\mathcal{H} = \mathcal{H}_1|_{e_1} \oplus \mathcal{H}_2|_{e_2}$. We prove that $\mathcal{H}$, the attachment of essential subgraphs of $\mathcal{G}_1, \mathcal{G}_2$, is a valid essential subgraph of the attachment $\mathcal{G}$. First, observe that

$$m := |\mathcal{E}_\mathcal{H}| = |\mathcal{E}_{\mathcal{H}_1}| + |\mathcal{E}_{\mathcal{H}_2}| - 1 = 2(n_1 + n_2 - 2) - 3 = 2n - 3$$

where $n = |\mathcal{V}_G|$ is the number of the nodes in the attachment $\mathcal{G}$. This means $\mathcal{H}$ has the correct number of edges. Let Y be a Laman subgraph of $\mathcal{H}$, then $\mathcal{E}_Y$ can be decomposed into two disjoint set of edges $\mathcal{E}_{Y_1} \subset \mathcal{E}_1$

and $\mathcal{E}_{Y_2'} \subset \mathcal{E}_2'$ such that $\mathcal{E}_Y = \mathcal{E}_{Y_1} \cup \mathcal{E}_{Y_2'}$. Define $Y_1 = (\mathcal{V}(\mathcal{E}_{Y_1}), \mathcal{E}_{Y_1})$ and $Y_2' = (\mathcal{V}(\mathcal{E}_{Y_2'}), \mathcal{E}_{Y_2'})$. Three cases arise: i) $v_i, v_j \in \mathcal{V}_{Y_2'}$, ii) either $v_i \in \mathcal{V}_{Y_2'}$, or $v_j \in \mathcal{V}_{Y_2'}$, and iii) $v_i, v_j \notin \mathcal{V}_{Y_2'}$. In case i) set

$$Y_2 = [\mathrm{Ren}(Y_2'; v_i, v_j | v_k, v_l)]_{e_{kl}}^+$$

in cases ii), without loss of generality assuming that $v_i \in \mathcal{V}_{Y_2'}$, set

$$Y_2 = \mathrm{Ren}(Y_2'; v_i | v_k)$$

and in case iii) set $Y_2 = Y_2'$. By definition, Y_1, Y_2 are Laman subgraphs of $\mathcal{H}_1, \mathcal{H}_2$, respectively. Since $\mathcal{G}_1, \mathcal{G}_2$ are rigid both Y_1, Y_2 are proper subgraphs, therefore $|\mathcal{E}_{Y_k}| \leq 2|\mathcal{V}_{Y_k}| - 3$ for $k = 1, 2$. In addition, in case i), Y_2' has one edge less than Y_2 which means $|\mathcal{E}_{Y_2'}| \leq 2|\mathcal{V}_{Y_2'}| - 4$. In cases ii) and iii), Y_2' has the same number of nodes and edges as Y_2 and thus $|\mathcal{E}_{Y_2'}| \leq 2|\mathcal{V}_{Y_2'}| - 3$. In all three cases, $|\mathcal{E}_Y| = |\mathcal{E}_{Y_1}| + |\mathcal{E}_{Y_2}|$. We obtain that in case i), the graph has $|\mathcal{V}_Y| = |\mathcal{V}_{Y_1}| + |\mathcal{V}_{Y_2'}| - 2$ nodes and the number of edges of Y satisfies the following upper bound

$$|\mathcal{E}_Y| \leq (2|\mathcal{V}_{Y_1}| - 3) + (2|\mathcal{V}_{Y_2'}| - 4) = 2|\mathcal{V}_Y| - 3 \qquad \text{(case i)}$$

In case ii), the graph Y has $|\mathcal{V}_Y| = |\mathcal{V}_{Y_1}| + |\mathcal{V}_{Y_2'}| - 1$ nodes and

$$|\mathcal{E}_Y| =\leq (2|\mathcal{V}_{Y_1}| - 3) + (2|\mathcal{V}_{Y_2'}| - 3) < 2(|\mathcal{V}_{Y_1}| + |\mathcal{V}_{Y_2'}| - 1) - 3 = 2|\mathcal{V}_Y| - 3 \qquad \text{(case ii)}$$

Finally, in case iii), the graph has $|\mathcal{V}_Y| = |\mathcal{V}_{Y_1}| + |\mathcal{V}_{Y_2}|$ nodes and

$$|\mathcal{E}_Y| \leq (2|\mathcal{V}_{Y_1}| - 3) + (2|\mathcal{V}_{Y_2'}| - 3) = 2(|\mathcal{V}_{Y_1}| + |\mathcal{V}_{Y_2'}|) - 6 < 2|\mathcal{V}_Y| - 3 \qquad \text{(case iii)}$$

As a result, in all three cases, Y is a proper Laman subgraph of $\mathcal{H}$, and therefore $\mathcal{G}$ is rigid. □

A.3 Proof of Theorem 4

Without loss of generality, assume $e_1 = (v_1, v_2)$ and $e_2 = (v_3, v_4)$. Define the following graph

$$Z' = (\{v_1', v_2', v_3', v_4'\}, \{z_1', z_2', z_3', e_1', e_2'\})$$

where Z' is obtained from the Z-link $Z = (\{v_1, v_2, v_3, v_4\}, \{z_1, z_2, z_3\})$ with

$$z_1 = (v_1, v_3),\ z_2 = (v_3, v_2),\ z_3 = (v_2, v_4)$$

using successive basic operations of edge-addition (twice) and renaming (once) as seen in the following

$$Z' = \mathrm{Ren}([[Z]_{e_1}^+]_{e_2}^+; v_1, v_2, v_3, v_4 | v_1', v_2', v_3', v_4')$$

In addition, $\|e_i'\| = \|e_i\|$ for $i = 1, 2$. Thus Z' is attachable to both $\mathcal{G}_1$ and $\mathcal{G}_2$ via e_1' and e_2', respectively. Observe that the graph Z' with $n(Z) = 4$ nodes and $n_e(Z) = 2n(Z) - 3 = 5$ edges is a minimally rigid graph. As a result, e_1', e_2' are edges of the essential subgraph of Z'. By direct calculations, $\mathcal{G} = \mathcal{G}_1|_{e_1} \oplus_Z \mathcal{G}_2|_{e_2}$ can be expressed as the successive attachment of $Z', \mathcal{G}_2$ and then a consecutive attachment to $\mathcal{G}_1$, i.e.

$$\mathcal{G} = \mathcal{G}_1|_{e_1} \oplus (\mathcal{G}_2|_{e_2} \oplus Z'|_{e_2'})|_{e_1'} \tag{23}$$

since attachment preserves rigidity, the obtained graph $\mathcal{G}$ is rigid. The only remaining technical point is to show that after attachment of Z' and $\mathcal{G}_2$ that creates $\mathcal{G}_3 = \mathcal{G}_2|_{e_2} \oplus Z'|_{e_2'})$. There exists an essential subgraph of $\mathcal{G}_3$ that contains the edge e_1'. Let $\mathcal{H}_2$ be an essential subgraph of $\mathcal{G}_2$ that contains e_2 and recall that Z' is a minimally rigid graph. Thus $\mathcal{H}_3 = \mathcal{H}_2|_{e_2} \oplus Z'|_{e_2'}$ is a minimally rigid graph that contains the edge e_1'. It is straightforward to show that $\mathcal{H}_3$ is the essential subgraph of $\mathcal{G}_3$. Therefore, we showed that by construction, there exists an essential subgraph of $\mathcal{G}_3 = (\mathcal{G}_2|_{e_2} \oplus Z'|_{e_2'})$ contains the edge e_1' and according to Theorem 3, the obtained graph $\mathcal{G} = \mathcal{G}_1|_{e_1} \oplus \mathcal{G}_3|_{e_1'}$ is rigid. □

References

[1] N. E. Leonard and E. Fiorelli, "Virtual leaders, artificial potentials, and coordinated control of groups," *Proc. of the 40th IEEE Conference on Decision and Control*, Orlando, FL, Dec. 2001.

[2] B. Hendrickson, "The molecule problem: exploring structure in global optimization," *SIAM J. Optimization*, vol. 5, no. 4, pp. 835–857, November 1995.

[3] R. D'Andrea, "Personal communication," Cornell University, Silby School of Mechnical and Aerospace Engineering, `http://www.mae.cornell.edu/raff/`, February 2002.

[4] R. Olfati-Saber and R. M. Murray, "Distibuted cooperative control of multiple vehicle formations using structural potential functions," *The 15th IFAC World Congress*, June 2002, `http://www.cds.caltech.edu/~murray/papers/2001l_om02-ifac.html`.

[5] A. Fax and R. M. Murray, "Information Flow and Cooperative Control of Vehicle Formations," *The 15th IFAC World Congress*, June 2002.

[6] J. Graver and H. Servatius, B. Servatius, *Combinatorial Rigidity*, vol. 2 of *Graduate Studies in Mathematics*, American Mathematical Society, 1993.

[7] G. Laman, "On graphs and rigidity of plane skeletal structures," *Journal of Engineering Mathematics*, vol. 4, no. 4, pp. 331–340, October 1970.

[8] L. Lovász and Y. Yemini, "On generic rigidity in the plane," *SIAM J. Alg. Disc. Meth.*, vol. 3, no. 1, pp. 91–98, March 1982.

[9] H. Gluck, "Almost all simply connected closed surfaces are rigid," in *Geometric Topology*, number 438 in Lecture Notes In Mathematics, pp. 225–239. Springer-Verlag, Berlin, 1975.

[10] B. Roth, "Rigidity and flexible frameworks," *The American Mathematical Monthly*, vol. 88, pp. 6–21, January 1982.

[11] M. Milam, K. Mushambi, and Murray R. M., "A new computational approach to real-time trajectory generation for constrained mechanical systems," *Proc. of the 39th IEEE Conf. on Decision and Control*, vol. 1, pp. 845–551, 2000.

[12] E. Rimon and D. E. Koditschek, "Exact robot navigation using artificial potential functions," *IEEE Trans. on Robotics and Automation*, pp. 501–518, 1992.

[13] H. N. Gabow and H. H. Westermann, "Forests, frames, and games: algorithms for matroid sums and applications," *Proc. 20th Ann. ACM Symp. on the Theory of Computing*, pp. 407–421, 1988.

[14] D. J. Jacobs and B. Hendrickson, "An algorithm for two-dimensional rigidity percolation: the pebble game," *Journal of Computational Physics*, vol. 137, pp. 346–365, 1997.

[15] N. A. Lynch, *Distributed Algorithms*, Morgan Kaufmann Publishers, Inc., 1997.