Rigidity and Persistence of Directed Graphs

Julien M. Hendrickx, Brian D.O. Anderson and Vincent D. Blondel

***Abstract*— Motivated by [1], [2] and [3], we consider here formations of autonomous agents in a 2-dimensional space. Each agent tries to maintain its distances toward a pre-specified group of other agents constant, and the problem is to determine if one can guarantee that the structure of the formation will persist, i.e., if the distance between any pair of agents will remain constant. A natural way to represent such a formation is to use a directed graph. We provide a theoretical framework for this problem, and give a formal definition of persistent graphs (a graph is persistent if almost all corresponding agents formations persist). Note that although persistence is related to rigidity (concerning which much is known [4]), these are two distinct notions. We then derive various properties of persistent graphs, and give a combinatorial criterion to decide persistence. We also define the notion of minimal persistence (persistence with least number of edges), and eventually, we apply these notion to the interesting special case of cycle-free graphs.**

I. Introduction

From the recent increasing development of autonomous agents systems arise new questions in graph theory. Consider a formation of n agents able to move in a 2-dimensional space. To each agent, one assigns a set of distance constraints: Agent i has to maintain its distance d_{ij} from agent j. It is important to understand that this is a constraint for i but not for j, which will a priori not be required to do anything in order to maintain its distance from agent i constant. Moreover, as long as a particular agent satisfies all its distance constraints, no other hypothesis is made about its movement. Agent 4 in Figure 1(a) can thus move freely on a circle of radius d_{41} centered on agent 1. We are interested in knowing if one can guarantee that, provided that each agent is trying to satisfy all its distance constraints, the structure of the formation will be conserved. In other words, we want to know if the distance in either direction between any pair of agents (whether or not there is a distance constraint in either direction between the pair) will remain constant along any continuous move. As shown in Figure 1, this kind of system can be represented by a directed graph: To each agent corresponds a vertex, and there is a directed edge from i to j if i has a constraint on the distance it must maintain from j. Note that double edges are allowed and represent a situation where both i and j have to maintain the distance between them constant.

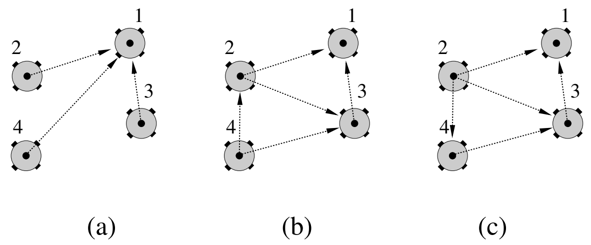

Fig. 1. Examples of autonomous agents systems; each arrow represents a distance constraint. In (a) for example, agents 2, 3 and 4 try to maintain their respective distances toward agent 1 constant. We will show that only (b) is persistent.

This issue is evidently related to the notion of rigidity of frameworks and graphs: A framework is represented by a graph $G = (V, E)$, where V is the set of vertices representing the articulations, and E is the set of undirected edges representing the beams or any other type of links. Suppose now that we assign arbitrary positions in[1] $\Re^2$ to all the vertices, and consider all the continuous moves such that the distance between the positions of any two vertices connected by an edge remains constant. The graph is called *rigid* if for almost all positions assignments, every such move preserves the distance between the positions of any pair of vertices, as shown in the examples in Figures 2(a) and 2(b). Note that Figures 1(a) and 2(c) present non rigid graphs. A graph is said to be *minimally rigid* if it is rigid and if there is no rigid graph having the same vertices but less edges, as shown in the example in Figure 2(b) (we will discuss this notion more extensively in Section IV). This class of graphs is interesting to study, not only because it provides an optimally efficient number of edges, but also because every rigid graph contains a minimally rigid graph.

However, the graphs that represent our autonomous agents-systems are directed. And although the definition of rigidity can be applied to directed graphs, it essentially remains an undirected notion and is thus inadequate to handle our problem. Consider indeed the system represented in Figure 1(c). Although the undirected graph is rigid, the

B. Anderson is with National ICT Australia , 216 Northbourne Ave, Canberra ACT 2601 Australia ; `brian.anderson@nicta.com.au`.
His work is supported by an Australian Research Council Discovery Projetc GranT and by National ICT Australia, which is funded by the Australian Government's Department of Communications, Information Technology and the Arts and the Australian Research Council through the Backing Australia's Ability Initiative.

V. Blondel and J. M. Hendrickx are with Department of Mathematical Engineering, Université catholique de Louvain, Avenue Georges Lemaitre 4, B-1348 Louvain-la-Neuve, Belgium; `blondel@inma.ucl.ac.be` `hendrickx@inma.ucl.ac.be`.
Their work is supported by the Belgian Programme on Interuniversity Attraction Poles initiated by the Belgian Federal Science Policy Office, and The Concerted Research Action (ARC) "Large Graphs and Networks" of the French Community of Belgium. The scientific responsibility rests with its authors. Julien Hendrickx holds a FNRS fellowship (Belgian Fund for Scientific Research).

[1]This could be done in any other space, but in the sequel we will always work in $\Re^2$.

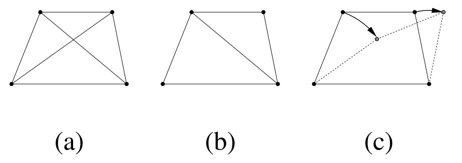
Fig. 2. (a) is rigid, (b) is minimally rigid, and (c) is not rigid.

structure of the formation may not be preserved: Agent 4 has an out-degree 1 and has thus only one distance constraint. If it moves on a circle of radius d_{43} centered on the position of agent 3, this constraint will remain satisfied. But, if agents 3 and 1 remain at the same position (and none of their constraints would force them to move) there will then (for almost all of the possible positions of agent 4) be no position for agent 2 where it could satisfy its three distance constraints, which implies that the structure of the formation is in some way ill-posed.

In the control literature, the characterization of such autonomous agents-systems has started to be attempted using the notion of *rigidity of a directed graph* [1], [2]: a directed graph is called rigid if the structure of the corresponding formation is conserved along any continuous move. Since this does not correspond to a simple transposition of the definition of rigidity for undirected graphs over to directed graphs, we prefer here to call this notion *persistence of a graph* in order to avoid confusion. Some results and conjectures concerning persistence are already used in the literature, especially for minimally persistent graphs (these can be defined similarly to minimally rigid graphs, see Section IV) or sufficient conditions for a graph to be (minimally) persistent [1], [2]. We propose here a formal definition of persistence that would provide a theoretical framework for this issue and allow us to prove these results. We will also derive some new properties of persistent graphs and give an operational criterion to determine if a graph is persistent.

The definition, which we provide in Section II, has the following intuitive meaning: a graph is persistent if, provided that all the agents are trying to satisfy their distance constraints, the global structure of the agent formation is preserved. We will see that *rigidity of the underlying undirected graph is a necessary but not sufficient condition.* This will lead us to the notion of constraint consistence of a graph, which is the additional condition for a rigid graph to be persistent. Intuitively, a graph is *constraint consistent* if every agent is able to satisfy its distance constraints provided that all the others are trying to do so. We will then show that a graph is persistent if and only if it is rigid and constraint consistent. So, in Figure 1, (a) is not rigid and (c) is not constraint consistent. The only persistent graph is thus (b). Note that all the proofs are omitted due to space limitations; a fuller version of the work is available in preprint form from the authors [5].

In Section III, we give some of the main properties of persistent graphs and the following criterion: A graph is persistent if and only if all those subgraphs are rigid which are obtained by removing outgoing edges from vertices with out-degree larger than 2 until all the vertices have an out-degree smaller than or equal to 2. We define then in Section IV *minimal persistence* analogously to minimal rigidity. We discuss some differences and similarities between the two notions, and give a characterization of minimally persistent graphs. Finally, we turn our attention to cycle-free graphs in Section V and give some more powerful results that exist in this special case, such as a polynomial time criterion to decide persistence.

II. Persistence of directed graphs

A *representation* of a graph $G = (V, E)$ is a function $p : V \to \Re^2$. We say that $p(i) \in \Re^2$ is the *position* of the vertex i, and define the distance between two representations p_1 and p_2 of the same graph by

$$d(p_1, p_2) = \max_{i \in V} ||p_1(i) - p_2(i)|| \,. \qquad (1)$$

A *distance set* d for G is a set of distances $d_{ij} > 0$, defined for all edges $(i, j) \in E$. A distance set is *realizable* if there exists a representation p of the graph for which $||p(i) - p(j)|| = d_{ij}$ for all $(i, j) \in E$. Such a representation is then called a *realization*. Note that any representation p of a graph is always a realization of the distance set defined by $d_{ij} = ||p(i) - p(j)|| \,, \forall ij \in E$.

A realization p of a distance set d is *rigid* if there exists $\epsilon > 0$ such that for all realizations p' of d satisfying $d(p, p') < \epsilon$, there holds $||p'(i) - p'(j)|| = ||p(i) - p(j)||$ for all $i, j \in V$. (We say in this case that p and p' are *congruent*). A graph is said to be *rigid*[2] if almost all its realizations are rigid. Note that we could have said "almost all representations", since as explained above, a realization is a representation and a representation is always a realization of a certain distance set. Although this definition is given here for directed graphs, rigidity is essentially an undirected notion. It is indeed not affected by modification of the edges directions.

Our definition of rigidity is slightly different from those usually given in the literature, but we have the following equivalence:

Theorem 1: The following conditions are equivalent for a graph $G = (V, E)$.

- G is (generically) rigid.
- There exists a realization of a certain distance set d for which any continuous displacement of the positions (such that at all time the positions of the vertices remain a realization of d) is a rigid motion, i.e., is such that all these realizations are congruent to each other. (This

[2]For simplicity we use the term *rigidity* but this notion is known in the literature as *generic rigidity* [1], [4].

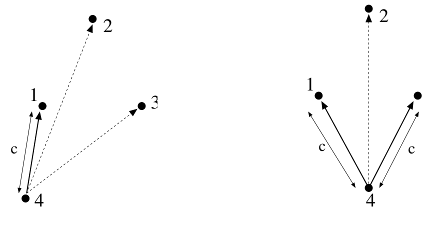

Fig. 3. Suppose that $d_{41} = d_{42} = d_{43} = c$. The position of 4 in (a) is not fitting because it only makes $(4, 1)$ active while there exists a position that would make both $(4, 1)$ and $(4, 3)$ active. On the other hand, its position in (b) is fitting because no point can be at the same time at a distance c from 1, 2 and 3.

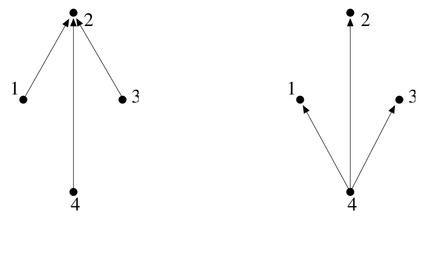

Fig. 4. The graph represented in (a) is constraint consistent. Each of 1, 3 and 4 can indeed always satisfy its unique distance constraint. On the other hand, the one represented in (b) is not constraint consistent because there always exists a configuration of positions of 1, 2 and 3 such that 4 is unable to satisfy its three distance constraints.

is equivalent to the usual definition of generic rigidity [8]).

- *Laman's criterion:* There is a subset $E' \subseteq E$ satisfying the following two conditions:
  (1) $|E'| = 2|V| - 3$.
  (2) For all non empty $E'' \subseteq E'$, the number $|V(E'')|$ of vertices that are end-vertices of the edges in E'' satisfies $|E''| \leq 2|V(E'')| - 3$.

As already mentioned, rigidity is an undirected notion, and is therefore insufficient to characterize persistence. The rigidity of a realization only means that if an external observer (or some physical properties) makes sure that the distance between the positions of any pair of vertices connected by an edge remains d_{ij}, then all the sufficiently close realizations of the same graph are congruent to each other. But, in our system of autonomous agents, there is no such external observer. Each agent is only aware of its own distance constraints, and can "move freely" as long as these particular constraints are satisfied. In order to have a more formal definition of persistence, we first need to characterize the fact that each agent is trying to keep the distances from its neighbors constant.

Let us thus fix a directed graph G, distances $d_{ij} > 0$ $\forall (i, j) \in E$, and a representation p. We say that the edge $(i, j) \in E$ is *active* if $||p(i) - p(j)|| = d_{ij}$. We also say that the position of the vertex $i \in V$ is *fitting* for d if it is not possible to increase the set of active edges leaving i by modifying the position of i while keeping the positions of the other vertices unchanged. This condition intuitively means that the agent i cannot satisfy additional distance constraints without breaking some that it already satisfies, as shown in the example in Figure 3. A representation of a graph is a *fitting* representation for a certain distance set d if all the vertices are at fitting positions for d. Note that any realization is a fitting representation for its distance set.

We can now give a formal definition of persistence: A realization p of a distance set d is *persistent* if there exists $\epsilon > 0$ such that every representation p' fitting for d and satisfying $d(p, p') < \epsilon$ is congruent to p. A graph is then *persistent* if almost all its realizations are persistent.

This definition is similar to the one of rigidity, and it is thus natural to ask if there is a relation between the two notions. Actually, a persistent graph is always rigid, and we will now define constraint consistence, which is a necessary and sufficient condition for a rigid graph to be persistent. A realization p of a distance set d is *constraint consistent* if there exists $\epsilon > 0$ such that any representation p' fitting for d and satisfying $d(p, p') < \epsilon$ is a realization of d. Intuitively, the constraint consistence of a realization means that if each agent tries to satisfy its distance constraints (i.e., is at a fitting position), then all the distance constraints will be satisfied, or equivalently, no agent will be in a situation where it cannot satisfy some constraint, as shown in the example in Figure 4. Again, we say that a graph is *constraint consistent* if almost all its realizations are constraint consistent. We have then the following useful equivalence:

Theorem 2: A graph is *persistent* if and only if it is *rigid* and *constraint consistent*.

III. Characterization of persistent graphs

In this section, we discuss properties of persistent graphs and give a combinatorial criterion to decide persistence. We begin by giving a lower bound on the number of active edges, and a first sufficient condition for a graph to be constraint consistent. In the sequel, $d^-(i)$ and $d^+(i)$ designate respectively the in- and out-degree of the vertex i.

Lemma 1: Let i be a vertex of a graph $G = (V, E)$. For almost all realizations p, there exists $\epsilon > 0$ such that in every representation p' fitting for the distance set corresponding to p and satisfying $d(p, p') < \epsilon$, the number of active edges leaving i is at least $\min(2, d^+(i))$. Consequently, a graph in which all the vertices have an out-degree smaller than or equal to 2 is always constraint consistent.

The next proposition allows us to delete edges in a persistence graph and maintain persistence.

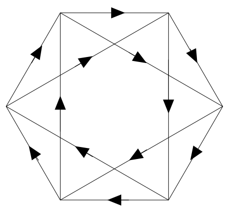

Fig. 5. All the vertices of this rigid graph have an out-degree 2. By Lemma 1 it is thus constraint consistent and therefore persistent, but the number of degrees of freedom of each vertex is 0.

Proposition 1: A persistent (resp. constraint consistent) graph remains persistent (resp. constraint consistent) after deletion of any edge (i, j) for which $d^+(i) \geq 3$.

An interesting corollary of Proposition 1 concerns the total number of degrees of freedom in the graph. The *number of degrees of freedom* of a vertex is the maximal dimension, over all representations of the graph, of the set of possible fitting positions for this vertex. In a 2-dimensional space, the vertices with zero out-degrees have 2 degrees of freedom, the vertices with out-degrees 1 have one degree of freedom, and the other have none. Note that a vertex with no degree of freedom can have more than one possible fitting position. Observe indeed that there is in almost all situations two possible fitting positions for a vertex with out-degree 2. However, since this set contains a finite number of points, its dimension is still 0. The following result provides a natural bound on their sum on a persistent graph.

Corollary 1: The sum of the degrees of freedom on all the vertices of a persistent graph cannot exceed 3.

Note that the total of three degrees of freedom is an upper bound. There are persistent graphs whose vertices do not have any degree of freedom, as shown in Figure 5.

Proposition 1 guarantees that a persistent graph remains persistent after deletion of any edge (i, j) for which $d^+(i) \geq 3$. After successive deletions, we can thus reach in this way a persistent graph whose vertices all have an outgoing degree that is smaller than or equal to 2. The next theorem states that a graph is persistent if and only if all the graphs obtained in this way are rigid.

Theorem 3: A graph is persistent if and only if all those subgraphs are rigid which are obtained by removing outgoing edges from vertices with out-degree larger than 2 until all the vertices have an out-degree smaller than or equal to 2.

The above result provides a non-polynomial time algorithm to check the persistence of a graph: it suffices to check the rigidity of all subgraphs obtained by deleting edges leaving vertices with out-degree larger or equal to 3 until all vertices have an out-degree smaller to or equal to 2. An algorithm with a smaller complexity would be useful in the case of large graphs, especially if there is a high number of vertices with a high out-degree, but no such algorithm has been found yet and at the time of writing it is still unclear if the problem of determining if a directed graph is persistent can be solved in a polynomial time. Notice that such an algorithm is known to determine if a graph is rigid [9]. However, better results exist in some particular cases. In Section V, we will see that for cycle-free graphs, persistence can be checked in polynomial time, and in the next section we introduce the related notion of minimal persistence and give a decision criterion that can also be checked in a polynomial time.

IV. Minimal persistence

In this section we define the notion of minimal persistence, analogously to minimal rigidity. We then discuss the main properties of these minimally persistent graphs, and show some similarities and difference between minimal persistence and minimal rigidity.

But first, we recall a few facts about minimal rigidity. One way to define it is to say that a graph is *minimally rigid* if it is rigid and if there exists no rigid graph with the same number of vertices and a smaller number of edges. Another way is to say that a graph is *minimally rigid* if it is rigid and if no single edge can be removed without losing rigidity. These two definitions are provably equivalent and lead to the following criterion: A graph $G = (V, E)$ is minimally rigid if it is rigid and if $|E| = 2|V| - 3$ (with an exception if $|V| = 1$). Moreover, a necessary and sufficient condition for a graph to be rigid is the presence of a minimally rigid (edge) subgraph. This can be seen using for example the Laman's criterion (Theorem 1). A *Henneberg sequence* is a sequence of graphs $G_2, G_3, \ldots, G_{|V|}$ such that G_2 is the complete (undirected) graph with two vertices, and G_{i+1} ($i \geq 2$) can be obtain from G_i by performing either a vertex addition or an edge splitting (see [1], [7]). These operations are defined in Figure 6, and one can show that they preserve minimal rigidity. Moreover, every minimally rigid graph can be obtained as the result of a Henneberg sequence [4].

We now define minimal persistence as follows: A persistent graph is called *minimally persistent* if no edge can be removed without losing persistence. Note that a first important and surprising difference with the concept of minimal rigidity is that a graph having a minimally persistent (edge) subgraph is not necessarily persistent, as shown in the example in Figure 7. More generally, unlike the case of rigidity, it is possible to obtain a non-persistent graph by adding edges to a persistent graph. A necessary condition for a persistent graph to be minimally persistent is immediate from Proposition 1: the absence of a vertex with an out-degree exceeding 2. On the other hand, a sufficient condition is minimal rigidity. Suppose indeed that one removes an edge to a persistent minimally rigid graph. The obtained graph would by definition not be rigid and

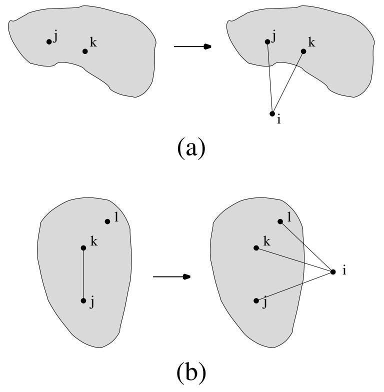

Fig. 6. (a) *vertex addition*: One adds a vertex and two incident edges. (b) *edge splitting*: One replaces an edge (j,k) by a vertex i and three edges (i,j), (i,k) and (i,l) where l is another vertex of the original graph. Both operations preserve (minimal) rigidity [6], [4].

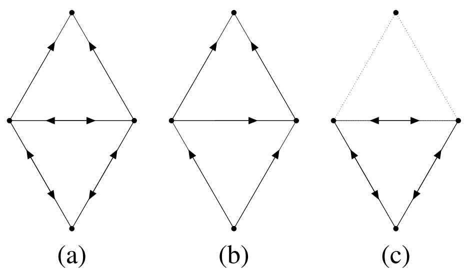

Fig. 7. The graph represented in (a) has a minimally persistent subgraph (b). However, by Theorem 3, it is not persistent because the subgraph represented in (c) is not rigid. In the corresponding multi-agent system, this could be seen as arising from a combination of unfortunate selections among the various possible information architectures available to the three agents of the cycle.

therefore not persistent.

As explained above, every minimally rigid graph can be obtained from an initial seed of two vertices and one edge by a sequence of vertex additions and edge splittings. We define here the directed version of these operations as in [1] by giving a direction to the added arrows in a way such that the out-degrees of the already existing vertices are not affected, as represented in Figure 8. To perform a *(directed) vertex addition* on a graph $G=(V,E)$, one adds a vertex and two edges from this vertex to different vertices of V. The *(directed) edge splitting* consists in removing one edge $(j,k)\in E$ and adding a vertex i and three edges (j,i), (i,k) and (i,l) for some $l\in V, l\neq j,k$. In the sequel, these operations will always be considered with the directed meaning. A *Henneberg sequence (directed case)* is then a sequence of graphs $G_2,G_3,...,G_{|V|}$, such that each graph G_{i+1} $(i\geq 2)$ can be obtained by implementing a vertex addition or an edge splitting on G_i, and G_2 is a graph of two vertices connected by one directed edge. As in the undirected case, all the graphs of such a sequence are minimally rigid. Moreover, since the out-degree of each of their vertices is always smaller or equal to two, Lemma 1 guarantees that they are also constraint consistent and thus minimally persistent. This implies that one can

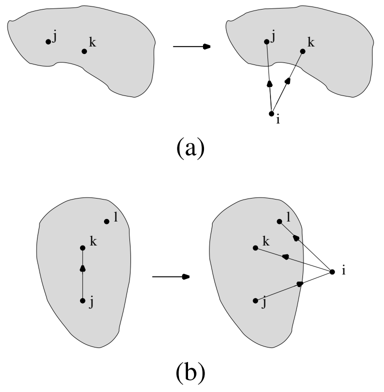

Fig. 8. Directed version of the *vertex addition* (a) and the *edge splitting* (b).

always assign a direction to all the edges of a minimally rigid undirected graph such that the resulting graph is minimally persistent. It is indeed possible to obtain every minimally rigid undirected graph by performing a sequence of (undirected) vertex additions and edge splitting on an initial seed of two vertices and one edge. [In order to obtain a minimally persistent graph, one can then simply perform the same sequence of the directed version of these operations]. However, it is still an open question as to whether, given an undirected rigid graph, there exists an assignment of directions to the edges such that the resulting directed graph is persistent.

Since every undirected minimally rigid graph can be obtained as the result of a Henneberg sequence, and since there always exists a minimally persistent graph resulting from the same sequence, it is natural to ask if every minimally persistent graph can be obtained in that way. Unfortunately, the existence of counterexamples force us to answer negatively to this question. Consider indeed the cycle of length 3 or any minimally persistent graph for which all the vertices have a positive out-degree: Both vertex addition and edge splitting conserve the out-degree of all the already existing vertices, and the first graph (G_2) of a Henneberg sequence contains a vertex with a zero out-degree. A graph having no vertex with a zero out-degree can thus never result from a Henneberg sequence. Actually, there also exist some graphs having a vertex with a zero out-degree and that still cannot be built using a Henneberg sequence.

As we already explained, minimal rigidity is a sufficient condition for a persistent graph to be minimally persistent. The next proposition states that it is also a necessary condition.

Proposition 2: A graph $G=(V,E)$ is minimally persistent if and only if it is persistent and satisfies $|E|=2|V|-3$.

Using the criterion provided by Proposition 2 and

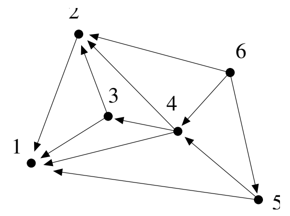

Fig. 9. Example of cycle-free persistent graph. The numbers correspond to an order in which the vertices can be added.

Corollary 1 about the number of degrees of freedom, it is possible to give a more specific characterization of minimal persistence that relies on the vertex out-degrees.

Theorem 4: A rigid graph with more than one vertex is minimally persistent if and only if one of the following two conditions is satisfied:

- Three vertices have an out-degree 1 and all the others have an out-degree 2.
- One vertex has an out-degree 0, one vertex has an out-degree 1, and all the others have an out-degree 2.

V. Cycle-free graphs

In this section, we provide a simple criterion to decide the persistence of cycle-free graphs and an explicit way to build all the persistent cycle-free graphs. Using the properties of the undirected vertex additions (see [6]) and Theorem 3, one can prove the following proposition.

Proposition 3: A graph obtained by adding one vertex to a graph $G = (V, E)$ and at least two edges leaving this vertex is persistent if and only if G is persistent.

We thus know that a cycle-free graph obtained by successively adding vertices all with out-degree larger than or equal to 2 to an initial seed of one directed edge connecting two vertices is persistent. The next theorem states that every persistent cycle-free graph can be obtained in such a way as shown in Figure 9. It also gives a simple criterion to decide persistence in the particular case of cycle-free graphs.

Theorem 5: A cycle-free graph having more than one vertex is persistent if and only if

- One vertex (called the leader) has an out-degree 0;
- One vertex (called the first follower) has an out-degree 1 and the corresponding edge is incident to the leader;
- Every other vertex has an out-degree larger or equal to 2.

Moreover, every such graph can be obtained from an original seed composed by the leader and first follower by adding vertices one by one in the way described in Proposition 3.

This result provides an algorithm with a low complexity to decide the persistence of a cycle-free graph. Moreover, if we apply it to a minimally persistent graph, we get the following corollary.

Corollary 2: A minimally persistent cycle-free graph with more than one vertex always has a *leader-follower* structure (see Theorem 5) and is always the result of Henneberg sequence containing only vertex additions.

VI. Conclusions and further works

As mentioned in the previous sections, several questions remain open; namely, the existence of a polynomial time criterion to decide if a graph is persistent, and an algorithm to assign directions to the edges of rigid graph in order to obtain a persistent graph. Among the possible extensions of this work, one can remark that we always assumed that the graph representations lie in $\Re^2$. From a practical point of view, it would be desirable to extend the results to $\Re^3$. However, this could give rise to new difficulties. For undirected graphs, there is no known equivalent of Laman's theorem in three dimensions, and not all minimally rigid graphs can be obtained by Henneberg sequences. Since we showed in Figure 7 that one cannot generally add edges indefinitely to a persistent graph without losing persistence, we could also define and characterize *maximally persistent* graphs. Finally, another issue would be to consider the robustness of a persistent graph. One could assign to each edge a probability of breakdown and to each unconnected pair of vertices a probability of parasite edge appearance. There might be in this case a maximally robust persistent graph, i.e., a graph for which the probability of losing persistence is minimal. It is evident that if there is a finite probability of losing an edge, it would be desirable to have persistence both with and without it. This observation emphasizes the need to understand better the circumstances under which edges can be added to a persistent graph without losing the persistence property.

References

[1] T. Eren, B.D.O. Anderson, A.S. Morse, and P.N. Belhumeur. Information structures to secure control of rigid formations with leader-follower structure. In *Proc. of the American Control Conference*, pages 2966-2971, Portland, Oregon, June 2005

[2] J. Baillieul and A. Suri. Information patterns and hedging brockett's theorem in controlling vehicle formations. In *Proc. of the 42nd IEEE Conf. on Decision and Control*, volume 1, pages 556–563, Hawaii, december 2003.

[3] A. Das, J. Spletzer, V. Kumar, and C. Taylor. Ad hoc networks for localization and control. In *Proc. of the 41st IEEE Conf. on Decision and Control*, Las Vegas, NV, 2002.

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

[5] J.M. Hendrickx, B.D.O. Anderson, V.D. Blondel and J.-C. Delvenne Rigidity and Persistence of directed graphs *Preprint*, 2005.

[6] G. Laman. On graphs and rigidity of plane skeletal structures. *J. Engrg. Math.*, 4:331–340, 1970.

[7] L. Henneberg. Die graphische Statik der starren Systeme. Leipzig, 1911.

[8] W. Whiteley. Some matroids from discrete applied geometry. In *Matroid theory (Seattle, WA, 1995)*, volume 197 of *Contemp. Math.*, pages 171–311. Amer. Math. Soc., Providence, RI, 1996.

[9] D.J. Jacobs and B. Hendrickson. An algorithm for two-dimensional rigidity percolation: the pebble game. *J. Comput. Phys.*, 137(2):346–365, 1997.