Distributed formation control using robust asynchronous and broadcast-based optimization schemes

Vinicio Modolo [*] **Damiano Varagnolo** [**] **Ruggero Carli** [*]

[*] *University of Padova, Padova, Italy*
[**] *Luleå University of Technology, Luleå, Sweden*

Abstract: We consider the problem of letting a network of mobile agents distributedly track and maintain a formation while using communication schemes that are asynchronous, broadcasts based, and prone to packet losses.
To this purpose we revisit and modify an existing distributed optimization algorithm that corresponds to a distributed version of the Newton Raphson (NR) algorithm. The proposed scheme uses then robust asynchronous ratio consensus algorithms as building blocks, and employs opportune definitions for the local cost functions to achieve the desired coordination objective. In our algorithm, indeed, we code the position of the to-be-followed target as the minimum of a shared global cost, and capture the desired inter-robots behaviors through dedicated distances-based potential barriers.
We then check the effectiveness of the strategy using field tests, and verify that the scheme achieves the desired goal of introducing robustness to changes in the agents positions due to unexpected disturbances. More precisely, if an agent breaks the formation, then the update mechanism embedded in our scheme make that agent move back to a meaningful position as soon as some packets are successfully received by the misplaced agent.

Keywords: Newton Raphson Consensus, packet losses, target tracking

1. INTRODUCTION

In many fields and applications such as surveillance, distributed moving sensor networks, cooperative control of mobile vehicles, smart transportation, and multi-vehicle robotic, there is the need for letting a set of robots keep a stable formation while simultaneously tracking a specified trajectory in space. This situation arises specially when this group of robots needs to perform a task that requires the various agents to maintain specific relative positions with respect to each other, but also follow a target that is also moving in space (for example, a space debris that shall be surrounded by a formation of chasing satellites).

Among the problems spanned by the situations mentioned above, we focus on a framework where agents shall simultaneously maintain a formation that has been designed a priori and whose center moves in time. In other words, we consider thus the specific and simplified situation where the to-be-followed formation is known and given, and where agents can noisily measure how much the actual currently reached formation differs from the ideal one.

In this paper we focus on finding how to distributedly transform this locally and noisily measured "formation error" into a strategy suggesting agents how to cooperatively move so to reach the desired ideal formation. We thus focus not on how to compute formations, but rather on how to distributedly compute how to return to a certain desired and a-priori known formation using collaboration schemes that are completely distributed, robust to inter-robot communication losses, and using asynchronous broadcast communication schemes.

Literature review A number of different works have been focusing on multi-agent formation control using a variety of approaches such as output feedback Zhou et al. (2016), complex Laplacian Lin et al. (2014) Jagtap et al. (2015), model independent coordination Egerstedt and Hu (2001) and extremum seeking control Zhang and Ordòñez (2011).

For example, output feedback methods for formation control have been proposed by Zhou et al. in Zhou et al. (2016) to solve the formation control problem in multi-agent systems with directed communication topologies. Here the time-varying formation control problem is converted first to a consensus problem and then to a stabilization problem, the sufficient conditions guaranteeing stability being then achieved through opportune Lyapunov stability analyses. Notice that in this case the target tracking problem was not specifically addressed.

An other example of technique, based on complex Laplacian frameworks, was developed by Lin et al. in Lin et al. (2014). Here authors address the problem of designing a distributed and stable formation control strategy first determining theoretical results for formation shapes that are specified by inter-agent relative positions. The authors

[⋆] The research leading to these results has received funding from the Norrbottens Forksningsråd. Email-addresses: `vinicio.modolo@studenti.unipd.it | damiano.varagnolo@ltu.se | carlirug@dei.unipd.it`

show then that all the formations that are subject to only shape constraints are those that lie in the null space of a complex Laplacian satisfying certain rank condition and that a formation shape can be realized almost surely if and only if the graph modeling the inter-agent specification of the formation shape is 2-rooted, and then providing a distributed and linear control law based on knowing the shape target formation a priori. In the same branch of techniques, Jagtap et al. (2015) designed a multi-agent motion synchronization scheme integrating output regulation techniques with the aforementioned complex Laplacian and opportune consensus algorithms. Notice that also here the tracking issue was not specifically addressed.

An other distributed formation control strategy, called model independent coordination, was developed by Egerstedt and Hu in Egerstedt and Hu (2001); here authors combined a desired reference path for a virtual leader with the formation control problem described above. The tracking problem was then specifically resolved for a group of nonholonomic robots – even if no navigation strategy was proposed therein. In this work, moreover, the relative distance and orientation of the agents was exponentially stabilized by feedback controllers working in a leader-follower setup and using local information from local sensors.

Lastly, we mention that extremum seeking techniques have also been used for the formation control problem, in particular in combination with approaches based on potential functions Olfati-Saber and Murray (2002). This approach includes creating potential functions that embed information about the scalar signal to be tracked (i.e., the moving target) and information about the interconnectivity among agents. Formation control and target tracking are then realized by minimizing this potential function; extremum seeking control strategies can then be used to autonomously find how the closed-loop system for maintaining the formation shall operate, while simultaneously maintaining stability and boundedness of the actuation signals Zhang and Ordòñez (2011).

Statement of contributions All the methods above require the agents to share information through opportune wireless communications. Often, to ensure the stability of the formation and tracking control algorithms, scholars require agents to implement synchronized communication schemes. Relaxing the need for synchronization, though, would simplify the implementability of these schemes in practical applications.

In this paper we analyse how the formation and tracking control problems can be solved in contexts where agents shall implement broadcast and lossy asynchronous communication schemes, and in which each agent has a limited amount of information about the status of the other nodes in the network.

More specifically, we cast the formation and tracking control problem so that it is amenable to be distributedly solvable using an opportunely modified version of the Newton Raphson Consensus (NRC) algorithm, a primal-based distributed numerical optimization scheme specifically designed for multi-agents convex optimization in situations where it is not possible to guarantee neither synchronicity nor losslessness in the information exchange schemes. The strategy will thus work by minimizing the cost above cooperatively and iteratively. By finding the minimum of this given function, the agents will indeed find how to move in space and converge towards the desired formation.

The paper briefly describes: *i)* how to design functions leading to particular formations, *ii)* how to implement robust asynchronous NRC algorithms, and *iii)* how to guarantee achieving formation control under the assumption of noise in the measurements of the relative distances among the nodes.

Structure of the manuscript Section 2 introduces the notation used in our distributed optimization scheme. Section 3 describes how to cast the formation and tracking problem using opportunely designed cost functions. Section 4 describes the NRC scheme and defines how to modify this algorithm so that it can serve as a formation and tracking control scheme. Section 5 shows results collected from field trials involving robots equipped with bluetooth communication modules. Section 6 concludes the manuscript with the learned lessons and some potential future research directions.

2. NOTATION

The algorithm works by letting each agent maintain in its memory an *estimate* of the positions of all the various other agents in the network. Considering thus agent i, we collect the set of estimates of i be the column vector $\widehat{x}^i$, in the sense that the estimate that node i has of the position of node j is the j-th component of $\widehat{x}^i$, indicated with $\widehat{x}^i_j$. Note that each component $\widehat{x}^i_j$ is actually a column vector that may have dimension two or three (two for terrestrial robots, three for aerial ones. Hereafter we will consider for simplicity a dimension two, generalizations being immediate). Stacking all the local state vectors $\widehat{x}^1, \dots, \widehat{x}^N$ we then obtain the vector of all the various estimated positions within the network, that we indicate with $\widehat{x}$. This vector will later on be extensively used within the NRC-based formation control scheme.

The previous set of states $\widehat{x}$ indicates *estimated* positions; in contrast to these quantities there is then the set of *actual* positions of the nodes within the used reference system. These states (either two or three dimensional, again depending on the application) will be indicated with the column vector x, whose components are the column vectors $x^1, \dots, x^N$.

As for the communication graph, we assume that the nodes communicate via a directed communication graph represented as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with nodes $\mathcal{V} = \{1, \dots, N\}$ and edges $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ so that $(i, j) \in \mathcal{E}$ represents the fact that node j can directly receive information from node i. We let moreover $\mathcal{N}^{\text{out}}_i$ be the set of *out-neighbors* of node i, i.e., let $\mathcal{N}^{\text{out}}_i = \{j \in \mathcal{V} \mid (i, j) \in \mathcal{E}, i \neq j\}$ be the set of nodes receiving messages from i. Similarly, we let $\mathcal{N}^{\text{in}}_i$ be the set of *in-neighbors* of i, i.e., $\mathcal{N}^{\text{in}}_i = \{j \in \mathcal{V} \mid (j, i) \in \mathcal{E}\ i \neq j\}$. Finally, we also let their cardinality be indicated by $|\mathcal{N}^{\text{out}}_i|$ and $|\mathcal{N}^{\text{in}}_i|$ respectively. As a standing assumption, the communication graphs considered in this manuscript are always strongly connected.

3. DEFINING FORMATIONS THROUGH COST FUNCTIONS

We now cast the problem of designing a formation for a set of agents as a problem of designing an opportune cost function. We consider then the following two requirements:

(1) were them let alone, all the various agents would individually converge towards a point that represents the target that these agents should track. Importantly, the position of the target can potentially be time-varying;
(2) when in the presence of other agents, the various nodes shall maintain a certain sparsity pattern (or formation) around the target point defined above.

Designing particular formations with complex function design can then be performed as we exemplify hereafter: assume for instance that the network is composed by six agents, and that we may want them to follow an hexagonal formation where the target position is in the centre of the hexagon, as in Figure 1 below.

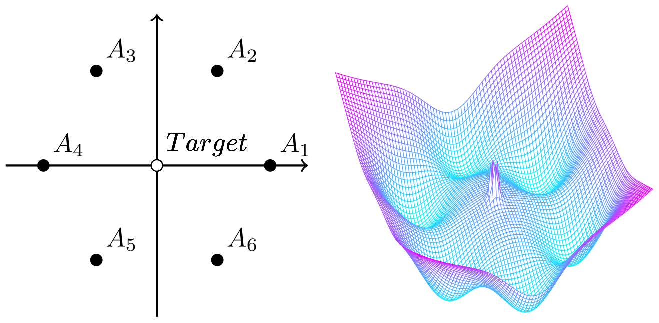


Fig. 1. An example of a plausible hexagonal formation for six agents, for which the target position is in the centre of the hexagon (on the left) and of the corresponding cost Φ as defined in (2).

The intuition is that the desired formation structure can then be seen as a cost function that has several local minima around the target and placed where we would like the agents to be (so that the number of local minima corresponds to the number of agents in the formation). Assume then to be able to phrase this "centralized" cost into a sum of opportune local costs. Intuitively speaking, this enables (as we will see in the forthcoming sections) casting the problem of maintaining a formation and tracking a target point into the problem of implementing an opportune distributed numerical minimization procedure.

Instrumental for the consequent derivations, we complete the notation listed in Section 2 by introducing:

- the quantity x_{form}, a column vector collecting the global coordinates of the N minima defining the formation. As an example, the vector x_{form} for the case of Figure 1 is a column vector stacking the coordinates of the various points $A_1, \ldots, A_6$ in the global reference system;
- the quantity x_{tgt}, indicating the position of the target defined above in the reference system used by the agents (and thus again an either two-dimensional or three-dimensional vector). In practice it practically always holds that $x_{\text{tgt}} = \frac{1}{N}\sum_{j=1}^{N} x_{\text{form}}^{j}$ where x_{form}^{j} refers to that part of x_{form} that correspond to the j-th minimum in the formation, so formally introducing x_{tgt} would not be needed. However, this notation is handy for our future derivations.

To express the centralized cost as a sum of local costs we then propose to consider the generic functional structure

$$f_i = \Phi\left(\widehat{x}_i^i, x_{\text{tgt}}, x_{\text{form}}\right) + \gamma \sum_{i \neq j} \Psi\left(\widehat{x}_i^i, \widehat{x}_j^i\right) \qquad (1)$$

with $i = 1, \ldots, N$, and where:

- Φ will serve the purpose of inducing agent i to move towards a location that is compatible with the position of the target x_{tgt} and the positions of the various local minima x_{form};
- Ψ will serve the purpose of capturing desired inter-robot behaviors, in the sense of inducing agents to respect certain distances by acting as a potential barrier;
- γ will serve as a regularization parameter trading off the importance of the "global" behavior induced by Φ against the "local" behavior induced by Ψ.

Before proposing an example of how f_i may look like in practice, we note that the total cost in (1) depends only on the estimated states $\widehat{x}_i^i$ and $\widehat{x}_j^i$ (i.e., on where i believes to be and on where i believes j to be), and the (assumed to be known) positions x_{tgt} and x_{form} (i.e., the positions of the target and of the minima of the formation). What the optimization algorithm will then do later when we will introduce it is to find for each node i the "best" geographical position as the argument minimizing f_i.

The following is then an example of potential f_i that we used in our field experiments. Note that, for obvious reasons, we do not claim that this specific cost is optimal to respect to any norm – we however experienced that, from experimental perspectives, it gave us good results in terms of making agents reach the desired formation while keeping safe distances among each other. As for Φ, our choice is thus

$$\Phi = \left\|x_{\text{tgt}} - \widehat{x}_i^i\right\|^2 + \left\|x_{\text{tgt}} - \widehat{x}_i^i\right\|^2 - \sum_{j=1}^{N} \exp\left(-\left\|x_{\text{form}}^j - \widehat{x}_i^i\right\|^2\right) \qquad (2)$$

where x_{form}^j refers to that part of x_{form} that correspond to the j-th minimum in the formation (cf. Figure 1). Graphically speaking, defining Φ through (2) while considering the planar hexagonal formation in Figure 1 leads to a cost that is qualitatively as in the right panel of Figure 1.

Inspecting Figure 1 we notice that the centre of the formation (i.e., the target position x_{tgt}) corresponds to a peak induced by the term $\left\|x_{\text{tgt}} - x_i^i\right\|^{-2}$. Even if it would seem, from purely mathematical perspectives, that this term is actually useless, we empirically found that its presence helps significantly the agents reaching the formation in a faster and smoother way.

The term $\left\|x_{\text{tgt}} - x_i^i\right\|^2$, instead, pushes agents to get closer to the center of the formation, and plays a relevant role specially when some agent somehow "falls behind". The last term corresponds to a sum of functions that have the role of creating a series of local minima where appropriate.

Importantly, the cost Φ does not contain elements that prevent different agents to come "too close", i.e., it does not contain mechanisms avoiding the possibility that more than one agent tries to occupy one specific minimum. To this aim it helps introducing the function Ψ, that adds opportune potential barriers so that no more than one node will be orbiting in the neighborhood of one minimum. In our field experiments we found that the inverse of the Euclidean distance between the (estimated) positions of the nodes, i.e.,

$$\Psi = \left\| \widehat{x}^i_j - \widehat{x}^i_i \right\|^{-2}$$

leads to desirable practical behaviors.

As a concluding comment about the here introduced costs, we note that in this way one designs only the shape of the formation, and do not assign to each agents a particular position within the formation. This implies that the position that an agent will occupy in the formation will be determined by its initial position with respect to the other agents. If one would instead prefer to assign to each agent a specified position then it is immediate to extend the framework by adding costs that reflect this choice by opportunely modifying Φ. Since this modification would not add too much to the messages that we want to convey, but at the same time it would add complexity that would obfuscate our discussions, we will simply ignore it.

4. DISTRIBUTED FORMATION CONTROL THROUGH THE NEWTON RAPHSON CONSENSUS

In this section we introduce the building block of our formation control algorithm, namely the NRC algorithm introduced in Carli et al. (2015); Bof et al. (view). The algorithm works by considering that a network of N collaborative agents aim at solving the separable optimization problem

$$x^* = \arg\min_{x \in \mathbb{R}^n} \sum_{i=1}^{N} f^i(x) \tag{3}$$

with the costs $f^i : \mathbb{R}^n \mapsto \mathbb{R}$ known only locally by agent i and strongly convex. The NRC algorithm is then based on the observation that the standard Newton-Raphson update in the standard centralized scenario with a single agent can be written as

$$\begin{aligned} x^+ &= x - \varepsilon \left(\nabla^2 f(x)\right)^{-1} \nabla f(x) \\ &= (1-\varepsilon)\, x + \varepsilon \left(\nabla^2 f(x)\right)^{-1} \left(\nabla^2 f(x) x - \nabla f(x)\right) \end{aligned} \tag{4}$$

with the superscript + indicating an updated value. Letting

$$h^i(x) := \nabla^2 f^i(x) \qquad g^i(x) := \nabla^2 f^i(x) x - \nabla f^i(x) \tag{5}$$

so that

$$\nabla^2 f(x) = \sum_{i=1}^{N} \nabla^2 f^i(x) = \sum_{i=1}^{N} h^i(x) \tag{6}$$

$$\nabla^2 f(x) x - \nabla f(x) = \sum_{i=1}^{N} \left(\nabla^2 f^i(x) x - \nabla f^i(x)\right) = \sum_{i=1}^{N} g^i(x) \tag{7}$$

we get the intuitive strategy of distributing (4) by letting the agents have different values of x^i and run in parallel the N local updates

$$x^{i+} = (1-\varepsilon)\, x^i + \varepsilon \left(\sum_{i=1}^{N} h^i(x^i)\right)^{-1} \left(\sum_{i=1}^{N} g^i(x^i)\right). \tag{8}$$

Two intuitions are then crucial to understand how (8) can work as a distributed version of (4):

(1) assume to be able to compute, through opportunely fast average consensus protocols, the quantities

$$z = \frac{1}{N} \sum_{i=1}^{N} h^i(x^i) \qquad y = \frac{1}{N} \sum_{i=1}^{N} g^i(x^i) \tag{9}$$

every time an x^i is updated. Then we could run (8) as $x^{i+} = (1-\varepsilon)\, x^i + \varepsilon\, (z)^{-1} (y)$ and expect that, since the dynamics of the N local systems would be identical and driven by the same forcing term, $x^i - x_j$ would tend to zero as soon as the dynamics are stable;

(2) once $x^i - x_j \approx 0$ for each i and j then the dynamics of each local system are practically equivalent to a standard centralized Newton-Raphson algorithm.

The previous intuition requires computing the two average consensus (9) and the local dynamics to be asymptotically stable. As shown in Varagnolo et al. (2016), local stability for the overall algorithm can be guaranteed assuming strong but bounded convexity of the local functions f^i, in the sense that it is guaranteed that there exists $\varepsilon^\star > 0$ and a neighborhood Ω of the global optimum so that for $\varepsilon < \varepsilon^*$ and initial conditions in Ω the algorithm is guaranteed to exponentially converge to the optimum. Importantly, the original NRC optimization algorithm has moreover been extended in Bof et al. (view) so to cope with communications that are broadcast, asynchronous, and prone to packet losses. The strategy is based on substituting standard average consensus algorithms with the *robust asynchronous ratio consensus algorithms* introduced in Bof et al. (2017).

Here we start from the algorithm proposed in Bof et al. (view), and extend it so that it can serve our purpose distributed formation control need. More precisely, we adapt it so that it can incorporate online information on the position of the nodes and account for situations where for some reasons a robot is not in the position where it is estimated to be (e.g., situations where agents are exposed to environmental forces that affect their position as wind or sea currents).

We thus address this by introducing this mechanism: assume that, after moving, the generic node i can measure its position in the global reference system, i.e., measure x^i . This newly obtained information can then be used to overwrite that element in the local set of estimates $\widehat{x}^i$ that corresponds to the own estimated position, i.e., i can perform the operation $\widehat{x}^i_i \leftarrow x^i$.

Summarizing, our distributed formation control algorithm works by updating the following variables:

- y^i and z^i, that are used to continuously keep track of the averages $\frac{1}{N}\sum_{i=1}^{N} h^i(x^i)$ and $\frac{1}{N}\sum_{i=1}^{N} g^i(x^i)$;

- σ_y^i, σ_z^i, $\rho_y^{i\leftarrow j}$ and $\rho_z^{i\leftarrow j}$, that help coping with broadcast asynchronous and packet-loss prone communications.

Algorithm 1 ra-NRC with feedback

```
    Initialization step (for every node)
    $x^i \leftarrow$ estimated initial position
    $y^i \leftarrow 0, \quad g^i \leftarrow 0, \quad g^{i,\text{old}} \leftarrow 0$
    $z^i \leftarrow I, \quad h^i \leftarrow I, \quad h^{i,\text{old}} \leftarrow I$
    $\sigma_y^i \leftarrow 0, \quad \sigma_z^i \leftarrow 0$
    $\rho_y^{i\leftarrow j} \leftarrow 0, \quad \rho_z^{i\leftarrow j} \leftarrow 0 \qquad \forall$ in-neighbor $j$
    When node $i$ broadcasts a packet
    a) prepare the data
1:  $y^i \leftarrow \frac{1}{|\mathcal{N}_i^{\text{out}}|+1} y^i$
2:  $z^i \leftarrow \frac{1}{|\mathcal{N}_i^{\text{out}}|+1} z^i$
3:  $\sigma_y^i \leftarrow \sigma_y^i + y^i$
4:  $\sigma_z^i \leftarrow \sigma_z^i + z^i$
    b) transmit the data
5:  broadcast $i, \sigma_y^i, \sigma_z^i$
    c) update the local estimates
6:  $\widehat{x}^i \leftarrow (1-\varepsilon)\widehat{x}^i + \varepsilon \left(z^i\right)^{-1} y^i$
7:  $g^{i,\text{old}} \leftarrow g^i$
8:  $h^{i,\text{old}} \leftarrow h^i$
9:  $h^i \leftarrow \nabla^2 f^i(\widehat{x}^i)$
10: $g^i \leftarrow h^i \widehat{x}^i - \nabla f^i(\widehat{x}^i)$
11: $y^i \leftarrow y^i + g^i - g^{i,\text{old}}$
12: $z^i \leftarrow z^i + h^i - h^{i,\text{old}}$
    When node $j$ receives a packet from node $i$
    a) receive the data
13: $y^j \leftarrow y^j + \sigma_y^i - \rho_y^{j\leftarrow i}$
14: $z^j \leftarrow z^j + \sigma_z^i - \rho_z^{j\leftarrow i}$
15: $\rho_y^{j\leftarrow i} \leftarrow \sigma_y^i$
16: $\rho_z^{j\leftarrow i} \leftarrow \sigma_z^i$
    b) update the local estimates
17: $\widehat{x}^j \leftarrow (1-\varepsilon)\widehat{x}^j + \varepsilon \left(z^j\right)^{-1} y^j$
18: $g^{j,\text{old}} \leftarrow g^j$
19: $h^{j,\text{old}} \leftarrow h^j$
20: $h^j \leftarrow \nabla^2 f^j(\widehat{x}^j)$
21: $g^j \leftarrow h^j \widehat{x}^j - \nabla f^j(\widehat{x}^j)$
22: $y^j \leftarrow y^j + g^j - g^{j,\text{old}}$
23: $z^j \leftarrow z^j + h^j - h^{j,\text{old}}$
    c) move, if the total number of received packets is a multiple of $K$
24: move towards the just computed $\widehat{x}_j^j$
25: measure the novel position $x^{j,\text{new}}$
26: $\widehat{x}_j^j \leftarrow x^{j,\text{new}}$
27: re-update the local estimates as in b) above
```

In our algorithm, σ_y^i (respectively σ_z^i) plays the role of a mass counter keeping track of the total information transmitted by node i about the variable y^i (z^i). Instead $\rho_y^{i\leftarrow j}$ ($\rho_z^{i\leftarrow j}$) keeps track of the total information received by node i from node j concerning the variable y^j (z^j). These mass counters have been introduced first in Vaidya et al. (2011) to robustify the standard average ratio consensus algorithm so to cope with the presence of packet losses. Intuitively, indeed, if a packet related to y^j is lost by node i, its value can be retrieved by computing the difference $\sigma_{j,y} - \rho_y^{i\leftarrow j}$ once the next packet $\sigma_{j,y}$ is received by node i.

Note then that, even if the vector $\widehat{x}^i$ contains an estimate of the positions of all the various nodes, to decide where to move node i is interested only into its i-th component of $\widehat{x}^i$, i.e, $\widehat{x}_i^i$. However, if communications among agents happen at high frequency, we noticed that it is convenient, from a practical point of view, that i moves to the new desired position $\widehat{x}_i^i$ not every time this quantity is updated, but only after K updates have been performed, where K is a positive integer that is likely dependent on the application. This part of the algorithm accounts also, as one can see from the pseudo-code, for potential external disturbances. In this case, indeed, the algorithm is so that when the node moves in space then its own estimate $\widehat{x}_i^i$ is re-initialized to the measured position.

5. FIELD EXPERIMENTS

We now illustrate the effectiveness of our scheme via a field experiments. More precisely, to verify the practical usefulness of our algorithm we performed field experiments using three arduino-based wheeled robots from `minseg.com` that shall follow an equilateral triangle formation. using broadcast bluetooth communications, The actual positions of the robots were measured using a Vicon motion-capture system, and the measured positions were individually communicated to the various robots using a dedicated bluetooth communication module. Agents were communicating among themselves also using broadcast bluetooth communications. Also in this case the to-be-followed trajectory was composed by two parts: a first one where the target remains fixed, and a second one where the target follows a sinusoidal movement.

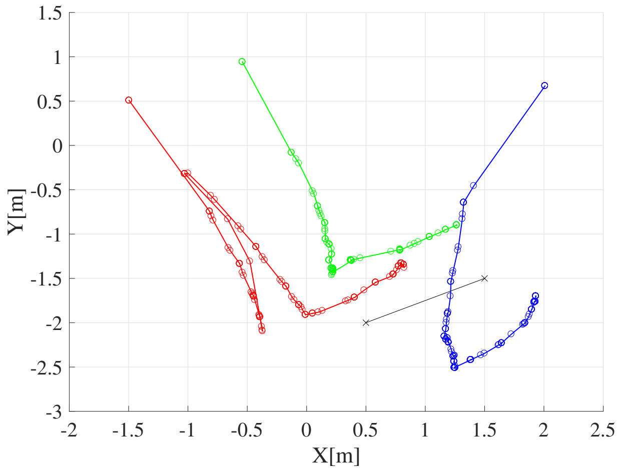
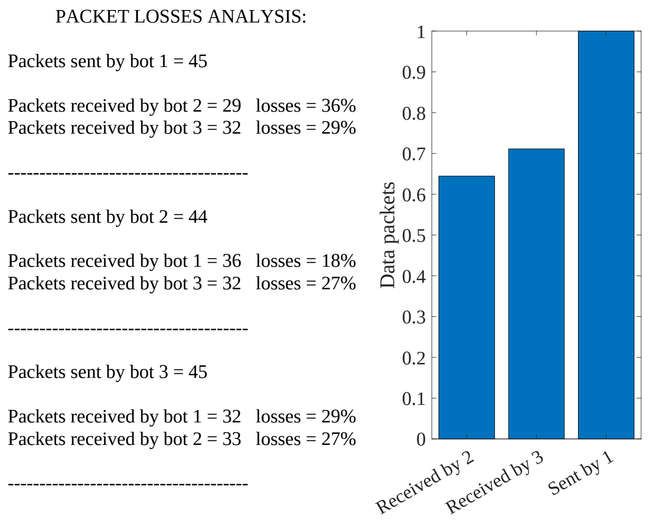


Fig. 2. Trajectories of the robots during our field experiment.

The results of the test are reported in Figure 2. In the first phase the target does not move, so that robots move only to reach the formation. In the second phase, instead, the target starts to move and the agents move to follow it while maintaining the formation. Note then that in this case there exists an external disturbance (more precisely, us) that was been applied to the red agent: indeed that robot was manually moved far away from its current position just after the target started to move. As the figure suggests, the agent quickly came back to the desired position in the indicated formation.

To complement the previous results we report in Figure 3 an analysis that quantifies the number of packets that were lost during the experiment. Despite experiencing more packet losses than we expected, the algorithm was able to cope with the task that it was assigned to it. As a concluding note, all the code used for our experiments is available on `github.com/damianovar/formation-control-via-NRC`.


PACKET LOSSES ANALYSIS:

Packets sent by bot 1 = 45

Packets received by bot 2 = 29   losses = 36%
Packets received by bot 3 = 32   losses = 29%

Packets sent by bot 2 = 44

Packets received by bot 1 = 36   losses = 18%
Packets received by bot 3 = 32   losses = 27%

Packets sent by bot 3 = 45

Packets received by bot 1 = 32   losses = 29%
Packets received by bot 2 = 33   losses = 27%

Fig. 3. Analysis of the frequencies of packet losses events during our field experiment.

6. CONCLUSIONS AND FUTURE WORKS

We proposed a new approach to distributedly solve the problem of tracking and maintaining a formation in an asynchronous and lossy communication scenario. To this purpose we employed a robustified Newton Raphson Consensus (NRC) algorithm, and checked using both numerical and tests that the scheme achieves the desired goals. More precisely, the algorithm introduces robustness to changes in the agents positions due to unexpected disturbances, so that if an agent breaks the formation, then the position estimates update mechanism embedded in our scheme makes that agent move back to a meaningful position.

This implies that the network of agents collectively responds to target motions by behaving in a coordinated fashion to the point that the target following motion qualitatively appears as a rigid translation of the whole flock (with a degree of rigidness that inversely depends – even if we did not show it in our experimental results section – on the probability that the packets sent by the various agents are getting lost).

Despite working from practical perspectives, the algorithm requires some future analysis and development efforts. For example, the stability of the original NRC (that has already been proved to hold under mild conditions) should now be revisited for this new practical situation in light of the additional mechanisms that we incorporated in our novel scheme.

An other interesting problem with both practical and theoretical sides spans from the possibilities of performing an on-line tuning of the stepsize ε, since this would likely improve the convergence speed of the scheme. However, letting stepsizes change and being able to prove convergence properties seems to be a notoriously non-trivial problem in the distributed optimization literature.

REFERENCES

Bof, N., Carli, R., Notarstefano, G., Schenato, L., and Varagnolo, D. (2018 (under review)). Newton-raphson consensus under asynchronous and lossy communications for peer-to-peer networks. *IEEE Transactions on Automatic Control.*

Bof, N., Carli, R., and Schenato, L. (2017). Average consensus with asynchronous updates and unreliable communication. *IFAC-PapersOnLine*, 50(1):601–606.

Carli, R., Notarstefano, G., Schenato, L., and Varagnolo, D. (2015). Analysis of newton-raphson consensus for multi-agent convex optimization under asynchronous and lossy communications. In *IEEE Conference on Decision and Control, 2015*, pages 418–424. IEEE.

Egerstedt, M. and Hu, X. (2001). Formation constrained multi-agent control. *IEEE International Conference on Robotics and Automation.*

Jagtap, P., Deshpande, A., Singh, N., and Kazi, F. (2015). Complex laplacian based algorithm for output synchronization of multi-agent systems using internal model principle. *IEEE Conference on Control Applications (CCA).*

Lin, Z., Wang, L., Han, Z., and Fu, M. (2014). Distributed formation control of multi-agent systems using complex laplacian. *IEEE Transactions on Automatic Control.*

Olfati-Saber, R. and Murray, R. (2002). Distributed cooperative control of multiple vehicle formations using structural potential functions. *IFAC Proceedings Volumes.*

Vaidya, N., Hadjicostis, C., and Dominguez-Garcia, A. (2011). Distributed Algorithms for Consensus and Coordination in the Presence of Packet-Dropping Communication Links Part II: Coefficients of Ergodicity Analysis Approach. In *arXiv.*

Varagnolo, D., Zanella, F., Cenedese, A., Pillonetto, G., and Schenato, L. (2016). Newton-raphson consensus for distributed convex optimization. *IEEE Transactions on Automatic Control*, 61(4):994–1009.

Zhang, C. and Ordòñez, R. (2011). Extremum-seeking control and applications: a numerical optimization-based approach. *IEEE International Conference on Robotics and Automation.*

Zhou, S., Qi, Y., Kang, Y., and Yan, S. (27-28 Aug. 2016.). Output feedback based formation control of multi-agent systems. *International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC).*