HAL
open science

A Packet Loss Tolerant Rendezvous Algorithm for Wireless Networked Robot Systems

Sabato Manfredi, Enrico Natalizio, Claudio Pascariello, Nicola Roberto Zema

To cite this version:

Sabato Manfredi, Enrico Natalizio, Claudio Pascariello, Nicola Roberto Zema. A Packet Loss Tolerant Rendezvous Algorithm for Wireless Networked Robot Systems. Asian Journal of Control, 2017, Advances in Control and Optimization Over Wireless Sensor and Actuator Networks, 19 (4), pp.1413-1423. 10.1002/asjc.1470 . hal-01672152

HAL Id: hal-01672152

https://hal.science/hal-01672152

Submitted on 14 Mar 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A PACKET LOSS TOLERANT RENDEZVOUS ALGORITHM FOR WIRELESS NETWORKED ROBOT SYSTEMS

Sabato Manfredi*, Enrico natalizio †, Claudio Pascariello*, Nicola Roberto Zema †
* Department of Electrical Engineering and Information Technology, University of Naples Federico II
smanfred@unina.it; cl.pascariello@studenti.unina.it
† Sorbonne universités, Université de technologie de Compiègne, CNRS, Heudiasyc UMR 7253, CS 60319
60203 Compiègne CEDEX, France. e-mail: {enrico.natalizio,nicola.zema}@hds.utc.fr

***Abstract*—This paper studies the coordination problem for a Wireless Networked Robot system (WNR). The objective is to drive the robots to keep a desired formation through local exchange of information. Nevertheless, packet losses may occur during communications among robots, thus preventing the system to reach its specific target. Specifically, the effects of an unreliable channel on the WNR performance are analyzed by considering the simulation of a *corrective consensus* algorithm into a network simulator. The use of a *packet-loss-tolerant* protocol is suggested to cope with heavy communication disruption. NS-3 (Network Simulator) simulation results validate the effectiveness of the proposed approach.**

I. Introduction

In the mid 2010, the development of compound systems made by networked vehicles, sensors and actuators has been led by tremendous technological improvements in robotics, computing and communications [1]. These improvements have enabled researchers and developers to envisage new robotic systems that interact cooperatively and wirelessly with other robots and even human beings. The namesake of this new technology is "Wireless Networked Robot systems" (WNR), as identified by the *IEEE technical committee on networked robots* [2]. According to the standard, this system is identified by two elements: (i) Autonomous Capabilities and (ii) Network-based Cooperation. The first element refers to the necessity, for a robotic node, to autonomously move and interact with the environment, while the second refers to its capability of communicating with its peers using radio technology. The growth of WNRs is supported by a wide base of industrial partners and there is a strong connection between robotics and sensor network enterprises. A *Robotic Network* is a subset of wireless sensor and actuator networks (WSANs) where the nodes have the added features to both autonomously interact with the physical world and adapt their own perception of the environment thanks to the informations exchanged. One of the most prominent research issue on this kind of system is how to introduce cooperation and coordination among multiple autonomous units in order to accomplish an arbitrary task. Solving this issue is a pivotal challenge that requires the use of combined approach from computer science, automatic control, telecommunication and artificial intelligence. Nevertheless, current approaches to cooperation among robots do not display resilience to communication disruption, commonplace in wireless networks. In this paper we introduce a novel cooperative *packet-loss-tolerant* algorithm, designed to accomplish a specific task even in presence of *link losses*. We tackle the issue from different points of view. From a second-order consensus algorithm applied to a *leader-follower* formation control, we (i) develop a broadcast-based network protocol able to obtain single-hop consensus in a wireless network; (ii) introduce a *corrective* phase both into the algorithm and into the protocol and (iii) we demonstrate the improvement over basic solution in an extremely disrupted wireless network environment. The paper is organized as follows: Section II presents a wide scope view of the related literature for network robotic and consensus control; Section III presents our wireless networked robot system approach. How we apply corrective consensus to the Wireless Networked Robot System is detailed in Section III-A. In Section IV is shown our design of a real protocol implementation of the corrective consensus and the results for a set of simulative experiments using it are shown in Section V. Section VI concludes the paper.

II. Related Works

Wireless networked robots extend the concept of multiple robot systems (MRS). The presence of networking provides mobile robots with the capability to work in a wide range of environments. In this section we briefly review the related literature. Seminal works are rooted on underwater [3] terrestrial [4] and urban [5] monitoring as well as home security [6] applications. Moreover, many studies on multiple unmanned aerial vehicles (UAV) coordination for terrain mapping, rescue and fire-fighting task have been carried out [7].

Recent research on formation control of WNRs has been developed to formulate decentralized rendezvous algorithm to guarantee robots to track the leader-robot position or velocity (i.e. rendezvous or consensus equilibrium). In this respect several approaches are detailed in [8], [9], [10], [11], [12], [13] and references therein. Although the effect of the communication network over the formation control performance (i.e. tracking of leader position) has been analyzed for centralised [14] and decentralised [15] architecture, the formulation of an effective algorithm to cope with network disruption of WNRs is still a challenging problem.

An observer based feedback controller using the linear matrix inequality (LMI) approach is proposed in [16]. The reason is to guarantee the robustness of the system by using a mean square value approach. The work in [17] studied a second order consensus problem for a set of mobile agents subject to noisy communication network check out, time delays and occasional packet losses modeled by a Bernoulli random process. They provided consensuability, i.e. consensus seeking ability, for such networked multi-agent systems. In [18] the averaging consensus problem with uniform link losses is considered. In this work, all communication links in the network may fail simultaneously. The authors provide a *memory consensus protocol*, so that an agent updates its own state according to the latest information stored in memory. Both works consider a simplistic link model where the link are modeled as bidirectional and instantaneous. The work in [19] proposed a nonlinear protocol for fixed-time consensus in networks with directed and intermittent communications. In our work we consider networks subject to more severe form of faults, as in presence of heavy disrupted environment. Moreover, we analyse and cope with the effect of packet losses on the decentralized control architecture of WNRs. In [20] a multi-hop dynamic consensus algorithm is proposed and a sufficient condition for stability is provided in order to have a faster convergence speed even in presence of heterogeneous time delays. Optimal parameters settings are also studied to reduce the packet collision phenomena. A Kalman filtering problem in a networked system subject to packet losses modeled by two state Markovian process, is studied in [21]. With these assumptions, sufficient conditions of the stable estimator are provided. In [22] the effect of packet losses on an industrial wireless sensor networked (IWSN) is investigated. New solution for network delay and packet loss problem are also investigated. A predictive compensator, a modified linear quadratic regulator and a combination of them, were proposed in order to reduce the effect of network disruption. In [23], simulation and experimental test of a single networked robot (Khepera mobile robot) have been performed to evaluate the effect of packet loss on quality of control (QoC) over network quality of service (QoS). They provided a critical limit for which QoC is acceptable. The impact of medium access protocol on average consensus problem over wireless networks for a group of quad-rotors is analyzed in [24]. This work provides a simulation environment that models different network communication layers. In this way, the impact of the CSMA/CA and TDMA can be evaluated. As a result, the averaging consensus presents a large convergence times using CSMA/CA protocols while TDMA is more suitable for real-time communications. In summary, the existing solutions deal with symmetric packet losses and balanced graph, that are far from realistic wireless networked robot system operative scenarios. However, a corrective algorithm for the first-order consensus problem in the case of Bernoulli packet-loss process with asymmetric link losses is introduced in [25].

Herein, we extend the first-order corrective consensus based algorithm [25] in order to deal with more realistic WNRs scenarios under different kinds of packet loss models. Specifically, we consider a leader-follower scenario where a team of robots (followers) has to track the leader velocity while maintaining a desired formation despite the presence of packet losses. We propose and apply for the first time a second-order corrective consensus-based model, thus extending the first order one [25] to the WNRs application scenario. Secondly, we formulate a broadcast-based network protocol in order to implement the corrective consensus over WNR system. Finally, the proposed algorithm and protocol are validated in a realistic simulative environment NS-3 in different scenarios.

III. Wireless Networked Robot Model

Let consider a WNR composed of $n+1$ robots with n *followers* (later labelled by subscript F) and one *leader* (later labelled by subscript L) as in Figure 1. A graph $G = (V, E)$ is used to denote the network topology among $n+1$ vehicles with $V = \{1, ..., n+1\}$ as the node set and $E \subset V \times V$ as the edge set.

$F = \{1, ..., n\}$ denotes the set of followers. An undirected edge $(j, i) \in E$ exists if vehicle i and vehicle j can access information from each other. It is also assumed that $(i, i) \notin E$. The adjacency matrix $A = [a_{ij}] \in \mathbb{R}^{(n+1)\times(n+1)}$ is defined as $a_{ij} = 1$ if $(j, i) \in E$ and $a_{ij} = 0$ otherwise. The neighbor set N_i of vehicle i is defined as $N_i = \{j : a_{ij} = 1\}$. The graph G is connected if there is a path between every pair of nodes. The graph G is balanced if the number of input links (i.e. in-degree) is equal to the number of output links (out-degree). This model assumes full-duplex, instantaneous and perfect communications. We consider a discrete-time second-order model [26] to describe the dynamic of leader/followers, namely:

$$p_i(t+1) = p_i(t) + T \cdot v_i(t) \tag{1}$$

$$v_i(t+1) = v_i(t) + T \cdot \begin{cases} u_i(t, p_i, v_i, p_j, v_j), & \begin{cases} i \in V \\ j \in V \end{cases} \\ f(t), & \text{if } i = n+1 \end{cases}$$

where $p_i(t)$, $v_i(t) \in \mathbb{R}^2$, $u_i(t, p_i, v_i, p_j, v_j) : [0, +\infty[\times\mathbb{R}^{2(n+1)} \to \mathbb{R}^2$ are, respectively, position, velocity and control input[1] associated with the $i-$th vehicle; $f(t) : [0, +\infty[\to \mathbb{R}^2$ is a signal describing the leader acceleration and T is the step-size.

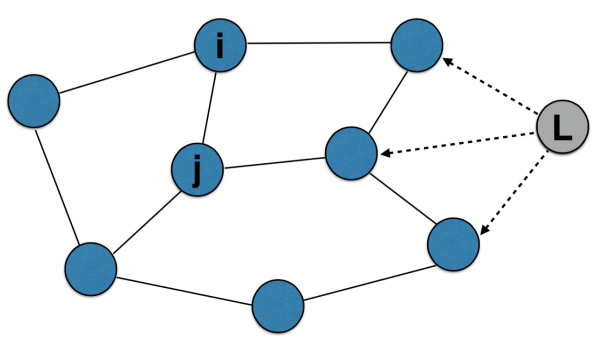

Fig. 1: A network of double integrator agents in which agent i receives the states p_j and v_j of its neighbour, agent j, if there is a link (i, j) connecting the two nodes; the gray circle is the leader.

The objective is to design $u_i(t)$ such that the followers maintain a desired formation and are able to follow the leader. Let $\overline{D}_{ij} : \mathbb{R}^{n+1} \to \mathbb{R}^{n+1}$ the desired euclidean distance between vehicle i and vehicle j, we assume the related matrix $\overline{D} = [\overline{D}_{ij}]$ being a skew-symmetric matrix with $\overline{D}_{ij} = -\overline{D}_{ij}$ for all i, j.

[1]in the following we use $u_i(t)$ as the acceleration, for the sake of simplicity of notation

At each follower is applied the following Standard Consensus control law [27]:

$$u_i(t) = u_{iF}(t) + u_{iL}(t) = \sum_{j=1}^{n} a_{ij}(t)\big[\big(p_j(t) - p_i(t)\big) + \overline{D}_{ij} + \gamma\big(v_j(t) - v_i(t)\big)\big] + a_{iL}(t)\big[\big(p_L(t) - p_i(t)\big) + \overline{D}_{iL} + \gamma\big(v_L(t) - v_i(t)\big)\big] \tag{2}$$

where $\gamma > 0$ is the coupling strength.

Definition 1:

The desired formation is asymptotically reached if $\| p_j(t) - p_i(t) \| \to \overline{D}_{ij}$ and $\| v_j(t) - v_i(t) \| \to v_L(t)$ $\forall i, j$.

The main aim of the algorithm is to achieve the desired formation, as in Definition 1, under the assumption of balanced graph. Unfortunately this assumption is violated in practice due to packet-losses on real systems, causing the robots not approach the desired formation [28], [14]. Therefore, in the next section we will formulate the definition of a *packet-loss-tolerant* algorithm: SOC^2 (Second Order Corrective Consensus algorithm), capable to perform the desired behavior for a network of mobile robots through local exchange of information.

A. A Packet loss tolerant Rendezvous Algorithm

Starting from the already defined distributed rendezvous algorithm of equation (2), we define a new set of variables $\phi_{ij}(t)$ with $(i, j) \in V^2$.

$$\phi_{ij}(t+1) = \phi_{ij}(t) + a_{ij}(t)[(p_j(t) - p_i(t)) + \overline{D}_{ij} + \gamma(v_j(t) - v_i(t))] \tag{3}$$

$$\phi_{ij}(0) = 0. \tag{4}$$

For each node i, the auxiliary variables $\phi_{ij}(t)$ represent the amount of change that node i has made to its state variables $p_i(t)$ and $v_i(t)$ due to neighbor j. Also, note that keeping $\phi_{ij}(t)$ and updating them according to (3) and (4) do not need any additional message exchange because the robot already execute (2) at every iteration. If robot i and j always take the same action, then the changes they make should be symmetric, i.e., $\phi_{ij}(t)$ = -$\phi_{ji}(t)$. Therefore, it is natural to define a new set of variables:

$$\Delta_{ij}(t) = \phi_{ij}(t) + \phi_{ji}(t) \tag{5}$$

that represent the amount of *bias* (as the sum of the states) that a robot has accumulated on both directions of the (i, j) *link*. In this way robots should reduce the error in a distributed and iterative manner: each robot i corrects its own state value.

Specifically, node i collects $\phi_{ij}(t)$ $(j \in N_i)$ from its neighbors to calculate the $\Delta_{ij}(t)$. Then node i

adjusts its control input u_i using the $\Delta_{ij}(t)$'s. After this correction, robots resume the standard consensus shown in (2) while periodically performing the corrective step described above. In summary, there are two types of iterations in corrective consensus: *Standard* and *Corrective* iterations.

1) During a *Standard iteration*, robots exchange state values and update the values in accordance with (2). In addition, each node also updates the ϕ_{ij}'s as defined in (3) and (4).
2) During a *Corrective iteration*, robots exchange ϕ_{ij} to calculate the Δ_{ij}'s and use it to adjust their control input, u_i, and auxiliary variables ϕ_{ij}.

Corrective consensus starts with the standard iterations and after every k consecutive standard iterations, one corrective iteration takes place using the following control input:

$$u_i(k+1) = -\frac{1}{2}\sum_{j=1}^{n+1}\Delta_{ij}(k), \quad i = 1,\dots,n \tag{6}$$

The auxiliary variables, instead, after k consecutive standard iterations update their value according to:

$$\phi_{ij}(k+1) = \phi_{ij}(k) - \frac{1}{2}\Delta_{ij}(k), \quad i,j = 1,\dots,n+1 \tag{7}$$

Overall, SOC^2 iterates as follows:

$$u_i(t) = \begin{cases} -\frac{1}{2}\sum_{j=1}^{n+1}\Delta_{ij}(t) \quad if \quad t = l(k+1)-1 \\ \text{otherwise} \\ \sum_{j=1}^{n} a_{ij}(t)\big[\big(p_j(t)-p_i(t)\big) + \overline{D}_{ij} + \gamma\big(v_j(t)-v_i(t)\big)\big] \\ + a_{iL}(t)\big[\big(p_L(t)-p_i(t)\big) + \overline{D}_{iL} + \gamma\big(v_L(t)-v_i(t)\big)\big] \end{cases}$$

where $i = 1,\dots,n$ and $\Delta_{ij}(t)$ is given by (5) with

$$\phi_{ij}(t+1) = \begin{cases} \phi_{ij}(t) - \Delta_{ij}(t)/2 & if \quad t = l(k+1)-1 \\ \phi_{ij}(t) + a_{ij}(t)[(p_j(t)-p_i(t)) & otherwise \\ + \overline{D}_{ij} + \gamma(v_j(t)-v_i(t))] & \end{cases} \tag{8}$$

where $(i,j) \in V^2$ and $l \in \mathbb{N}$ is the l-th corrective iteration.

Remark 1: Notice that the WNR closed loop system is composed of the model (1), the control law (2) and the corrective consensus (6) and its stability analysis can be carried out by leveraging on the existing results and approaches presented in the literature. Specifically let an infinite sequence of contiguous time intervals $T_k = [t_k, t_{k+1})$, $k = 0,1,...$ with $t_0 = 0$ and $t_{k+1} - t_k \leq T$, for some uniform $T > 0$, such that the topology is fixed on each interval but may change over different time intervals; the network topology is said "jointly connected" if the union of graphs over the infinite sequence of time intervals T_k is connected [29]. Under the following connectivity assumption:

Assumption 1: The WNR graph G is jointly connected.

it is possible to prove convergence and stability of the second order closed loop WNR system composed of (1) and (2) (see i.e. [26] and references therein). The stability of the overall WNR closed loop system, also including the corrective consensus algorithm (6), may be derived by using the analytic approach formulated in [25]. As herein we are aimed to validate the performance of the second order corrective consensus algorithm in a realistic scenario by a well known simulator also giving a realistic protocol to implement the proposed algorithm and for space constraints, we give indications about the main results and assumption to guarantee stability of the closed loop system. For the same motivation we notice that herein we assumed that the variables ϕ_{ij}'s are always delivered, but this assumption may be dropped still guaranteeing network stability and algorithm convergence (see the approach in [25]).

IV. Protocol Implementation

As the algorithms of Section III rely only the exchange of small messages, we have developed a network protocol based on broadcast messages that is able to translate the methods and assumptions of corrective consensus algorithm into a practical implementation. We called it *BESSEL*, as for distriButed corrEctive conSenSus fanEt protocoL. For each robotic node j, every T_{tx} seconds, the broadcasting of the following information is triggered: robot's unique *id*, *position*, *velocity vector* and a time-stamp. The source of this information is outside the scope of this paper, but different approaches for inertial navigation and localization for wireless nodes are already widespread in literature. Whenever node i receives the packet, it stores the information and calculates the $\phi_{ij}(t)$ according to Equation 7. After having collected a set of $\phi_{ij}(t)$ coming from other nodes, each t_ϕ seconds, every node j broadcasts a packet containing a time-stamp and the list of all the $\phi_{ji}(t)$s it has knowledge of in the time-frame T_u. Whenever another node i receives the list of $\phi_{ij}(t)$s from another node j, it extracts his own $\phi_{ji}(t)$. In this way the nodes exchange the information necessary to run the standard consensus algorithm by calculating and applying a new control input each T_c ms. Each k iterations of the standard consensus, the nodes apply instead the corrective one using the

TABLE I: 802.11g Simulation Parameters.

PHY Parameter	Value		MAC Parameter	Value
Frequency	2.4 GHz		SlotTime	9 μs
Receive Sensitivity	-86 dBm		SIFS	16 μs
Transmission Power	18 dBm		Preamble Length	96 bit

information they have previously exchanged. As the simulated nodes' memory is finite, all the information older than the current time minus T_u is deleted from a node. As *BESSEL* is based only on broadcast messages, it can be run on any kind of networking solution that supports packet radio, thus it can be implemented even at level 2 of the ISO/OSI stack. Furthermore, it can be implemented as an application and be run in parallel with any other kind of network services and applications. By these choices, it is possible to imagine that *BESSEL* can be implemented directly on real drones equipped with commercial, off-the shelf, network devices and our simulative study is bent to adhere to this scenario.

V. Simulative Validation

In order to evaluate the performance of SOC^2 and *BESSEL* we tested them in the NS-3 simulator [30]. To assess the capability of the corrective consensus to keep the formation even in disruptive environments we implemented *BESSEL* as an application built on top of UDP/IP that, in turn, is built on top of a IEEE 802.11 wireless network in Ad-Hoc mode without any routing. We used the stock NS-3 simulated devices with the radio parameters taken from the market [31] and summarized in Table I. In this way the information that has to be exchanged between nodes is directly serialized and sent as packets whose single transmission and reception is handled by the simulator's loss models.

A. Simulation Scenario

We consider all nodes equipped with a single wireless interface and we set the simulator to employ a two-folded loss model. As the line-of-sight signal and the reflection of this last one by the ground are much stronger than any other possible path, in this flying vehicles environment, the simulator is set to employ both a Two-Ray-Ground [32] and ITU-R 1411 [33] propagation models. The first one will take into account the prominence of the two aforementioned paths and the second one will take into account the losses due to the employment in urban or rural environments. It is important to remark that using this set up, the very concept of stable topology falls because each single transmission is capable to address a complete different set of nodes from the previous or the next one. From another point of view this approach is capable to render more closely what happens in a real scenario and thus is better suited to test network protocols implementations. The simulator takes into account also some networking issues: we assume no fragmentation of packets and we simulate a IEEE 802.11n [34] protocol stack, the current consumer standard, so that packet collisions are treated accordingly. Other protocols have been discarded as unsuited for the specific tasks as they will have issues of insufficient range (IEEE 802.15 family), power and space constraints (IEEE 802.16 family) or necessity of external infrastructures (TDMA-based cellular network family). In the simulated scenario, 30 nodes will start from random positions inside a $15m \times 15m$ squared area. They have to reach consensus on a set of predetermined inter-distances, $\overline{D}_{ij}$, in order to keep a circular formation. The center of the circular formation is an additional *leader* node, which constantly moves in a fixed direction. The inter-distances $\overline{D}_{ij}$ are precomputed for each node and used as the only input at the beginning of the experiment. The circular formation radius, indicated with δ, is 200 meters. The nodes flight height is set to 50 meters and kept constant during the simulations. For all our results, we sample the Maximum Error, $E_{MAX}(t)$, among all the nodes, according to Definition 1. It represents the maximum error between nodes formation at time t and the desired one, defined by $\overline{D}$. All the output values are averaged over 20 runs by using a 90% confidence interval. The values of t_u, T_c and T_{tx} are respectively fixed to 4 s, 10 ms and 100 ms. To highlight the capability of SOC^2 and *BESSEL* to work in heavily disruptive environments, i.e. in the presence of heavy jamming, we set the simulator to periodically lower the transmission range of all the nodes to few meters. In detail, we set different possibilities for the channel availability: 40, 60 and 70 as the percentage of time the network is unavailable over a period of 5 s, and we refer to it as *ND* (i.e. Network Disruption). As *BESSEL* is time-driven, these values also correspond to the gross data volume exchanged over the total sent, saved for losses due to standard network dynamics. Our results are three-folded. Initially, we demonstrate the capability of SOC^2 and *BESSEL* to achieve better results in terms of positioning error in comparison to the standard consensus. Then, we tune the parameters of our schemes to ameliorate their performance by introducing the concept of *Corrective Impulse* and, at the end, we propose some network design perspectives for *BESSEL*. For the first set of results, we varied the value of k, the number of standard iterations before a corrective one is executed, among 50, 100 and 200, i.e. a corrective iteration is executed every $\rho = k \times T_c$, that is 500, 1000 and 2000 ms. The values of t_ϕ are set

TABLE II: Values of t_ϕ.

ρ	t_ϕ
500ms	400ms
1000ms	900ms
2000ms	1900ms

TABLE III: Scenario Parameters.

Parameter	Values
Network Disruption *ND*	40%, 60%,70%
Formation Radius δ	200m
$\rho = k \times T_c$	500, 1000, 2000 [ms]
T_{corr}	1,3,5,10
T_c	10 [ms]
T_{tx}	100 [ms]

according to SOC^2 and summarized in Table II. These values have been chosen to provide the nodes that are executing a corrective consensus iteration with the most updated set of $\phi_{ij}(t)$ possible. Along with SOC^2, we also evaluate the behavior of the standard action in the same environment. The value *S.C.*, i.e. *Standard Consensus*, in the graphs represents the case where SOC^2 is not applied. For the second set of results we introduce the concept of *Corrective Period* (T_{corr}) as the number of consecutive *corrective* consensus actions in a row, called for every T_{tx}. Finally, for our third set of results we introduce the concept of *Consensus Packet Units* (CPU). A CPU is composed by two three-dimensional vectors representing position and velocity. In this way, in *BESSEL* a node i sends a *standard consensus* packet made by one CPU and a *corrective consensus* packet made by $\frac{N_{nodes} \times CPU}{2}$ (the list of $\phi_{ij}(t)$). As the broadcasts of *BESSEL* are all time-driven, after a time t and if all nodes can communicate with any other, the total number of sent packets by the entire network is given by:

$$P_{sent} = t \times CPU \times \left[\frac{1}{T_{tx}} + \frac{N_{nodes}}{2\rho} \right] \times N_{nodes}$$

given that the broadcast events are independent and N_{nodes} is the number of nodes in the network. The total number of packet sent is used to estimate the network resources needed to reach consensus in the various scenarios. All the parameters are summarized in Table III.

B. Results

We have discovered that high levels of ND produces interesting behavioral changes. For our first set of results, we focus our analysis on this case. In Figures 2, 3 and 4 it is shown the evolution of E_{MAX} when the *ND* and the ρ vary. Specifically, the two horizontal lines correspond to the thresholds where the distance difference is below 5% and 2% of the precomputed distances $\overline{D}_{ij}$. The various curves represent the different ρ and the different figures represent the different *ND*. Higher variations of E_{MAX} can be seen for the highest levels of disruption. Specifically, it can be observed that, when the jamming is below 50%, the performances of the system with SOC^2 are similar to the case when *Standard Consensus* is applied. This can be noticed in Figure 2, that points out the strength of SOC^2 in presence of "good" network conditions, since it introduces negligible delays. Furthermore, with a *ND* of 60% and 70%, it gives multiple advantages over the standard action. First and foremost, the *Standard Consensus* is never able to minimize the error below the threshold of 2% while SOC^2 and *BESSEL* are capable to reach the 2% threshold even with 70% of packet losses due to jamming (Figure 4). Looking at the results with different ρ, it can be seen that in Figure 3 there is a direct relationship between a higher convergence speed and lower ρ, as show in [25], in fact a more frequent corrective action, i.e. $\rho = 500$ms guarantees a faster convergence time, instead with $\rho = 2000$ms the convergence speed is slower. Instead, in Figure 4, E_{MAX} is lower with a ρ of 1000ms. This is due to the fact that with a higher *ND*, a faster exchange of data could lead to congestion and packet losses, thus leaving nodes with outdated information for a certain time. Employing instead a lower exchange ratio, the losses due to collisions are minimized and data is exchanged more efficiently. In a second set of results we analyze the impact of a corrective action application multiple times in a row. Figures 5 and 6 represent the evolution of E_{MAX} when a T_{corr} is applied in the following way: in Figure 5, a corrective impulse of 1, 3 and 5 iterations in a row is applied to a ρ of 1000ms; in Figure 6, a corrective impulse of 1, 5 and 10 iterations in a row is applied to a ρ of 2000ms, considering that in this last case the time window in which new iterations can be executed is larger. Synchronization issues are predominant again. In fact, in both figures, a higher T_{corr} always leads to a smaller E_{MAX}, but it is always the middle value for the same parameter that leads to the best result. As discussed before, the higher level of disruption of the analyzed scenario means that the information available to SOC^2 is often outdated due to the loss of most recent packets. In this case applying more times in a row the same control input leads to longer delays in the minimization of E_{MAX}. More in detail, it can be seen in for Figure 5 that the accuracy increase is minimal in respect to Figure 6. In the latter, the value of ρ is already optimal and thus, with a T_{corr} of 3, its effects are increased. Applying instead a higher T_{corr}, the nodes overshoot their optimal positions and E_{MAX} is increased.

1) Design Guidelines: Up to now we have shown the behavior of SOC^2 and *BESSEL* from a control system standpoint but important insights on the design of practical systems implementing our proposals can be devised. In fact, in Figures 7 and 8, a pictorial description of the necessary data to exchange in order to achieve a desired E_{MAX} is given. The plots represent the evolution of E_{MAX} with a *ND* of 60% and 70%. Superimposed with the results for ρ of 500, 1000 and 2000 ms, the straight lines represent the total number of packets sent by all the nodes for the same scenario, expressed in CPU. Even if, for the considered number of nodes, the system does not incur in packet fragmentation, the resulting packet size and broadcasting interval for a ρ of 500ms can lead to great differences in the total data exchanged. When comparison are made with the case where SOC^2 is employed, it is necessary to distinguish between different levels of *ND*. For lower levels of disruption, as in Figure 7 a simple trade-off rule can be easily extracted. If energy is not an issue, selecting a ρ of 500ms is indeed the best option, at the expenses of exchanged data and thus congestion issues. For achieving the same level of E_{MAX}, with a ρ of 1000ms, *BESSEL* needs the same amount of data but twice the time. For higher levels of *ND*, as in Figure 8, the performance of SOC^2 and *BESSEL* with various ρ are flattened by synchronization issues and therefore, an optimal parameters combination is difficult to identify. Under certain circumstances, using a greater T_{corr} can be extremely beneficial for *BESSEL*. In Figure 9 we compare the evolution of E_{MAX} for two cases: (i) $\rho = 500ms$ and $T_{corr} = 1$ and (ii) $\rho = 2000ms$ and $T_{corr} = 5$, superimposed with the data exchange prediction for the same settings. For case (ii), we can see that E_{MAX} falls below the 2% threshold even faster than case (i). If we consider that the network load is independent of T_{corr}, we can see that using the right T_{corr}, SOC^2 and *BESSEL* can achieve the same performances associated with a higher network load using a fraction of the packets. Besides, these results underline that a synchronization system for the exchange of the $\phi_{ij}(t)$s is one of the most important issues to be addressed.

VI. Conclusions and Future Works

Distributed rendezvous algorithm provide an elegant way to keep cooperating robots in a desired formation or have a specific behavior That solution is however impractical when applied to wireless networks that naturally exhibit asymmetric packet losses. In this paper has been proposed a corrective consensus algorithm (e.g. *packet-loss-tolerant*) that enables the practical use of consensus in real-life mobile robot networks. Through the addition of new variables at each node and new corrective iterations, the proposed method let a wireless robotic network to converge to a desired formation despite heavy link losses. Furthermore, the performance of corrective and standard consensus algorithms in terms of convergence speed are compared. Additional work is prospected to devise a synchronization system for the timely exchange of updated information and to better tune the parameters of the algorithm and the related network protocol.

Acknowledgment

This work has been carried out in the framework of the DIVINA Challenge Team, which is funded under the Labex MS2T program. Labex MS2T is supported by the French Government, through the program Investments for the future managed by the National Agency for Research (Reference: ANR-11-IDEX-0004-02).

References

[1] B. Kehoe, S. Patil, P. Abbeel, and K. Goldberg, "A survey of research on cloud robotics and automation," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 398–409, 2015.

[2] "Ieee technical committee on networked robots." http://www-users.cs.umn.edu/~isler/tc/ (2016). Available at.

[3] H. Singh, J. Catipovic, R. Eastwood, L. Freitag, H. Henriksen, F. Hover, D. Yoerger, J. G. Bellingham, and B. A. Moran, "An integrated approach to multiple auv communications, navigation and docking," in *OCEANS'96. MTS/IEEE. Conf. Proc.*, vol. 1, pp. 59–64, IEEE, 1996.

[4] W. Kaiser, G. Pottie, M. Srivastava, G. Sukhatme, J. Villasenor, and D. Estrin, "Networked infomechanical systems (nims) for ambient intelligence," *Ambient Intelligence*, pp. 83–113, 2004.

[5] L. Chaimowicz, A. Cowley, D. Gomez-Ibanez, B. Grocholsky, M. Hsieh, H. Hsu, J. Keller, V. Kumar, R. Swaminathan, and C. Taylor, "Deploying air-ground multi-robot teams in urban environments," in *Multi-Robot Systems. From Swarms to Intelligent Automata Volume III*, pp. 223–234, Springer, 2005.

[6] D. Fox, J. Ko, K. Konolige, B. Limketkai, D. Schulz, and B. Stewart, "Distributed multirobot exploration and mapping," *Proceedings of the IEEE*, vol. 94, no. 7, pp. 1325–1339, 2006.

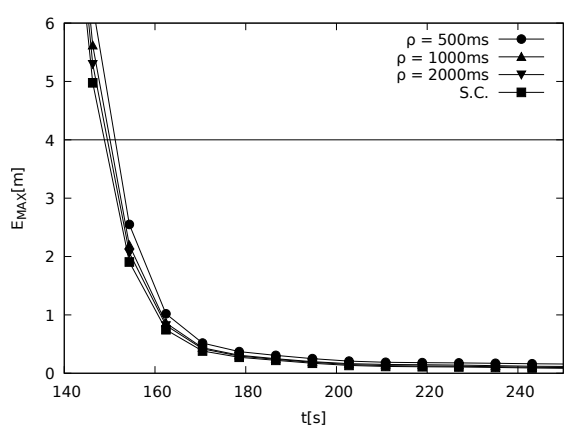

Fig. 2: Evolution of E_{MAX} with *ND* = 40%.

[7] A. Ollero, S. Lacroix, L. Merino, J. Gancet, J. Wiklund, V. Remuss, I. Perez, L. Gutierrez, D. Viegas, M. Benitez, *et al.*, "Architecture and perception issues in the comets multi-uav project. multiple eyes in the skies," *IEEE Rob. Autom Mag.*, vol. 12, pp. 46–57, 2005.

[8] Z. Hu, C. Ma, L. Zhang, A. Halme, T. Hayat, and B. Ahmad, "Formation control of impulsive networked autonomous underwater vehicles under fixed and switching topologies," *Neurocomputing*, vol. 147, pp. 291–298, 2015.

[9] D. Luo, W. Xu, S. Wu, and Y. Ma, "Uav formation flight control and formation switch strategy," in *8th International Conference on Computer Science & Education (ICCSE) Conf. Proc.*, pp. 264–269, IEEE, 2013.

[10] S. Manfredi and D. Angeli, "Frozen state conditions for exponential consensus of time-varying cooperative nonlinear networks," *Automatica*, vol. 64, pp. 182–189, 2016.

[11] S. Manfredi and D. Angeli, "A criterion for exponential consensus of time-varying non-monotone nonlinear networks," *IEEE Trans. Autom. Control*, no. 99, pp. 1–1, 2016.

[12] Y. Zheping, L. Yibo, Z. Jiajia, and Z. Gengshi, "Moving target following control of multi-auvs formation based on rigid virtual leader-follower under ocean current," in *34th Chinese Control Conference (CCC), 2015.*, pp. 5901–5906, IEEE, 2015.

[13] S. Manfredi, "A consensus based rate control scheme for atm networks," *Int. J. Control Autom. Syst.*, vol. 10, no. 4, pp. 817–823, 2012.

[14] L. R. Buonocore, V. Lippiello, S. Manfredi, F. Ruggiero, and B. Siciliano, "Effects of packet losses on formation control of unmanned aerial vehicles," *IFAC Proc.*, vol. 47, no. 3, pp. 1234 – 1240, 2014.

[15] S. Manfredi, "An algorithm for fast rendezvous seeking of wireless networked robotic systems," *Ad Hoc Networks*, vol. 11, no. 7, pp. 1942–1950, 2013.

[16] X. Li and S. Sun, "Robust h control for networked systems with random packet dropouts and time delays," *Procedia Eng.*, vol. 29, pp. 4192–4197, 2012.

[17] Y. Zhang and Y.-P. Tian, "Consensus of data-sampled multi-agent systems with random communication delay and packet loss," *IEEE Trans. Autom. Control*, vol. 55, no. 4, pp. 939–943, 2010.

[18] S. Wang, C. An, X.-X. Sun, and X. Du, "Average consensus over communication channels with uniform packet losses," in *Chinese Control and Decision Conference. Conf. Proc.*, pp. 114–119, IEEE, 2010.

[19] Q. Wang, Y. Wang, and C. Sun, "Fixed-time consensus of multi-agent systems with directed and intermittent communications," *Asian J. Control*, 2016. asjc.1329.

[20] S. Manfredi, "Design of a multi-hop dynamic consensus algorithm over wireless sensor networks," *Control Eng. Pract.*, vol. 21, no. 4, pp. 381–394, 2013.

[21] B.-F. Wang and G. Guo, "Kalman filtering with partial markovian packet losses," *Int. J. Autom. Comput.*, vol. 6, no. 4, pp. 395–400, 2009.

[22] V.-A.-Q. Nguyen and M. Yoo, "Packet loss compensation for control systems over industrial wireless sensor networks," *Int. J. Distrib. Sens. Netw.*, p. 5, 2015.

[23] A. Mechraoui, Z. H. Khan, and J.-M. Thiriet, "Effect of packet loss on the quality of control of a networked mobile robot," in *30th IFAC Workshop on Real-Time Programming and 4th Int. Workshop on Real-Time Software*, pp. 97–101, 2009.

[24] J. A. Guerrero, Y. Challal, and P. Castillo, "Impact of wireless medium access protocol on the quadrotor formation control," *Flight Formation Control*, pp. 257–276, 2012.

[25] Y. Chen, R. Tron, A. Terzis, and R. Vidal, "Corrective consensus: Converging to the exact average," in *49th IEEE Conference on Decision and Control (CDC). Conf, Proc.*, pp. 1221–1228, IEEE, 2010.

[26] A. Eichler and H. Werner, "Closed-form solution for optimal convergence speed of multi-agent systems with discrete-time double-integrator dynamics for fixed weight ratios," *Systems & Control Letters*, vol. 71, pp. 7–13, 2014.

[27] W. Ren, "Consensus strategies for cooperative control of vehicle formations," *Control Theory & Application*, vol. 1, no. 2, pp. 505–512, 2007.

[28] W. Jiang and H. Schulzrinne, "Modeling of packet loss and delay and their effect on real-time multimedia service quality," in *NOSSDAV Conf. Proc.*, Citeseer, 2000.

[29] W. Ni and D. Cheng, "Leader-following consensus of multi-agent systems under fixed and switching topologies," *Systems & Control Letters*, vol. 59, no. 3, pp. 209–217, 2010.

[30] P. Fuxjaeger and S. Ruehrup, "Validation of the ns-3 interference model for ieee802.11 networks," in *2015 8th IFIP Wireless and Mobile Networking Conference (WMNC) Conf. Proc.*, pp. 216–222, Oct 2015.

[31] "Cisco Aironet 802.11a/b/g Wireless CardBus Adapter." http://www.cisco.com/c/en/us/products/collateral/wireless/aironet-1130-ag-series/product_data_sheet0900aecd801b9058.html (2016). Available at.

[32] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*, vol. 2. Prentice Hall PTR, New Jersey, 1996.

[33] R. Katuiski and A. Kiedrowski, "Calculation of the propagation loss in urban radio-access systems," *IEEE Antennas. Propag.*, vol. 50, no. 6, pp. 65–70, 2008.

[34] Y. Xiao, "Ieee 802.11 n: enhancements for higher throughput in wireless lans," *IEEE Wireless Commun.*, vol. 12, no. 6, pp. 82–91, 2005.

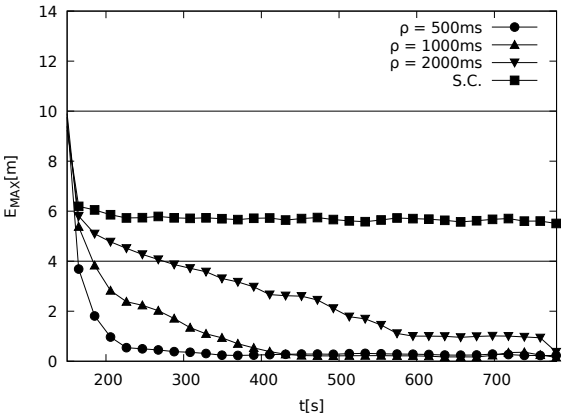

Fig. 3: Evolution of E_{MAX} with *ND* = 60%.

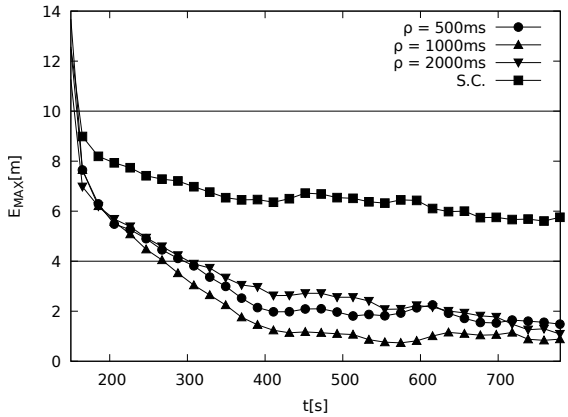

Fig. 4: Evolution of E_{MAX} with *ND* = 70%.

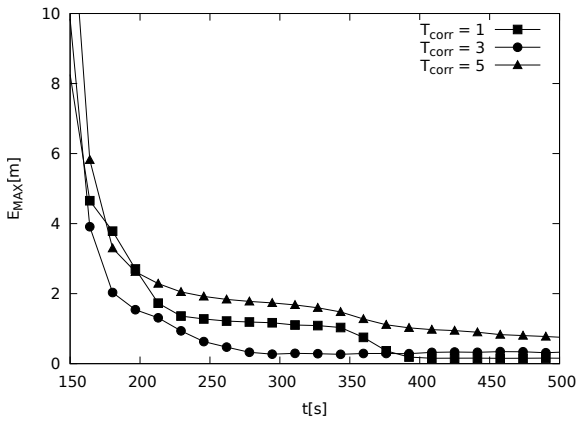

Fig. 5: Evolution of E_{MAX} varying T_{corr}. *ND* = 70%, $\rho = 1000ms$.

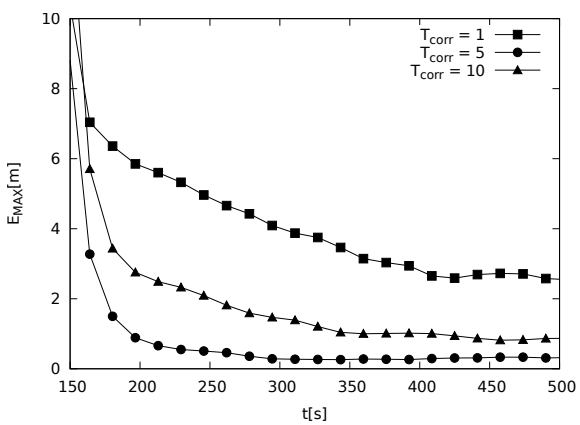

Fig. 6: Evolution of E_{MAX} varying T_{corr}. *ND* = 70%, $\rho = 2000ms$.

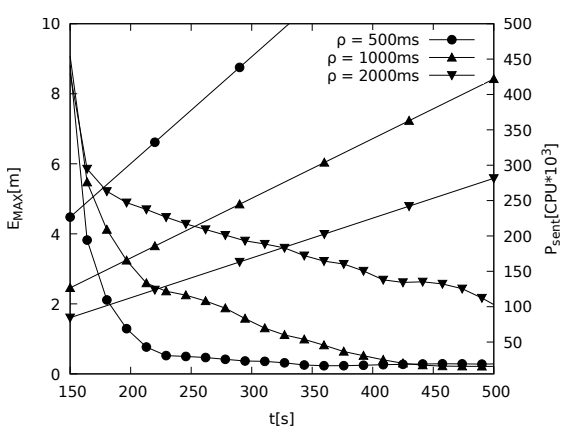

Fig. 7: Network design guidelines with *ND* = 60%.

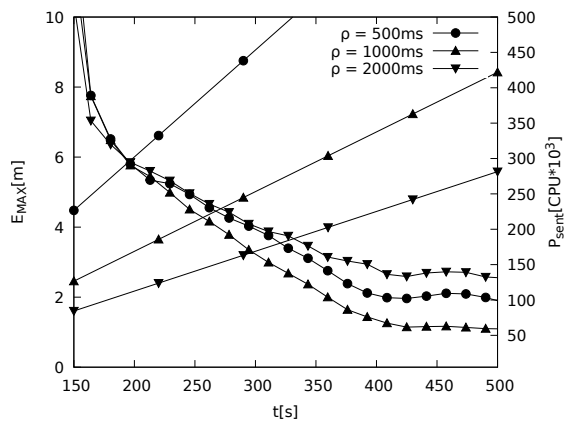

Fig. 8: Network design guidelines with *ND* = 70%.

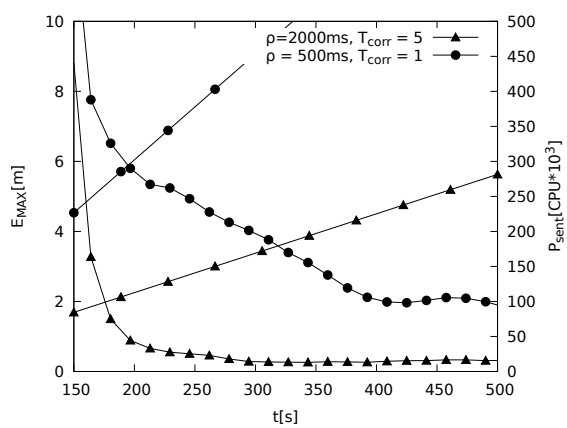

Fig. 9: Evolution of E_{MAX} with different combinations of T_{corr} and ρ. *ND* = 70%.