Effectiveness of Embedded Topology Controllers in a Multi-robot Network

Israel C. S. Rocha∗ Carlos H. C. Ribeiro **
Cinara Ghedini ***

∗ Computer Science Division, Aeronautics Institute of Technology, Sao Jose dos Campos, Brazil (e-mail: israel.c.rocha@gmail.com).
** Computer Science Division, Aeronautics Institute of Technology, Sao Jose dos Campos, Brazil (e-mail: carlos@ita.br)
*** Computer Science Division, Aeronautics Institute of Technology, Sao Jose dos Campos, Brazil (e-mail: cinarag@gmail.com)

Abstract: This paper presents analysis and results for several experiments performed to verify the effectiveness of topology control algorithms running on a multiagent network (e-puck robots), in a multi-objective realistic hardware-based scenario. The main goal of such a network can be, for instance, disaster response, wildfire and environment monitoring, security tasks and also exploration and mapping (Scherer and Rinner, 2020). In doing so, the agents should be controlled in such a way that they have to ensure connectivity maintenance and robustness to failures while improving the coverage area. These issues were addressed by Ghedini et al. (2016, 2017, 2018), who proposed topology controllers regarding all these aspects and tested them successfully in a simulated scenario. However, it lacked the validation on a hardware-based domain, accomplished by this work. By running several experiments from different initial topologies, it was possible to analyze and verify the effectiveness of the developed topology controllers proposed in Ghedini et al. (2016, 2017, 2018). The hardware-based experiments shown results compatible with the simulated ones.

Keywords: Networked robotic system modeling and control; Autonomous robotic systems; Guidance navigation and control.

1. INTRODUCTION

Consider the scenario depicted in Figure 1, which illustrates a viable application of a multi-robotic system with autonomous unmanned aerial vehicles (UAVs) communicating among each other and dynamically positioned by topology controllers to provide a sufficiently good network service to a group of uncontrolled on-the-ground mobile clients.

![Fig. 1. An example of a possible application scenario for topology control in a multi-agent system (Ghedini et al., 2016).](image)

An overall target for this team might be to improve the coverage area to provide service to more clients, but, if only this aspect is considered, the network could get disconnected by pushing the point-to-point links to the limits (nodes A and C on the left of Figure 2), or by having a highly covered area by a highly vulnerable network as illustrated in Figure 2 (right), in which, if node B is attacked or fails, the network gets fragmented. Here, we define a network as vulnerable if it is potentially able to fragment if some nodes fail. Similarly, a node is defined as vulnerable if the paths between it and other nodes are highly relying on a few connections (node B in our example).

![Fig. 2. Left: Example of a connected network. Node B is connected to nodes A and C. Nodes A and C are on the verge of a disconnection. Right: example of a vulnerable topological configuration.](image)

Since the agents must exchange information with their neighbors (Varadharajan et al., 2019), a disconnection...
compromises the effectiveness of what the team is supposed to provide (communication, monitoring and so forth). Furthermore, it is not desirable to have inner uncovered areas (namely, holes) in the overall coverage area (Figure 2, left), because the service provided by the team will be intermittent and fail when the ground group (in our example) makes even small displacements, since everyone inside the hole is out of range.

Thus, while the control regarding the coverage area acts to maximize the distance among agents while preventing holes from appearing, a connectivity control should act to prevent a “pushed to the limit” disconnection from happening and a robustness control should also act to avoid topological configurations vulnerable to failures of elements as shown in Figure 2 (right).

To approach all these issues, a combined strategy based on local information with regard to coverage area, connectivity and robustness has been addressed and extensively validated in all aspects through MATLAB™ simulations by Ghedini et al. (2016, 2017, 2018). Validation in a real hardware-based setup is addressed in this paper by considering a testbed (Figure 3) composed of a 1.8 m by 1.8 m squared wooden arena with real agents - model e-puck (GCTronic, 2018) - on it, over a handling computer and exchanging information through Bluetooth.

Fig. 3. Testbed for the experiments.

In addition, an external camera was necessary because the e-puck robotic agents have no positioning system (like GPS or GLONASS). Thus, computer vision had to be used to recognize each robot on the arena and to evaluate its Cartesian position and angle (posture). To do so, the camera should recognize each fiducial (Figure 4, left), which is an univocal image placed on the top of each robot that identifies it biunivocally.

Fig. 4. Example of a Square Fiducial Marker with c centimeters of edge (left). Two connected robots at a distance \( d \leq 2R \) and with overlapping in the coverage area (right).

By using this testbed (Figure 3), several experiments were performed on 12 e-pucks to verify the effectiveness of the topology controller algorithms. Each e-puck runs an embedded firmware to move, communicate through Bluetooth, and process topology control regarding connectivity maintenance. Due to hardware limitations, the controls regarding robustness and coverage area were executed on the handling computer and combined with the connectivity maintenance control in each e-puck.

The rest of this paper is organized as follows. The background on topology algorithms and related controls is presented in Section 2. Each initial topology, control gain and benchmark score involved in the experiments are shown in Section 3. The results are discussed in Section 4 and lastly, Section 5 ends with a conclusion.

2. PRELIMINARIES

2.1 Connectivity Maintenance

Consider \( n \) mobile robots \( v_1, v_2, \ldots, v_n \), all of them having a sensing range \( R \). Let \( p_i \in \mathbb{R}^2 \) be the bidimensional position vector of robot \( v_i \); referred to a predefined Cartesian plane, \( p = [p_1, p_2, \ldots, p_n]^T \) (\( p \in \mathbb{R}^{n \times 2} \)) be the position matrix and let \( \| \bullet \| \) be the norm of \( \bullet \), so \( v_i \) and \( v_j \) are connected if and only if \( d = \| p_i - p_j \| \leq 2R \) (Figure 4, right).

The algorithm regarding connectivity maintenance is based on Graph Theory (Bollobás, 1998; Godsil and Royle, 2001). It is known that the second smallest eigenvalue \( \lambda_2 \) of the Laplacian Matrix \( (L \in \mathbb{R}^{n \times n}) \) of an undirected graph \( G \) corresponds to its algebraic connectivity \( \lambda \). Let \( D \in \mathbb{R}^{n \times n} \) be the degree matrix of \( G \) and let \( A \in \mathbb{R}^{n \times n} \) be the adjacency matrix, so \( L \triangleq D - A \). For connectivity purposes, \( A \) is such that (Sabattini et al., 2013):

\[
a_{i,j} = \begin{cases} 
\exp \left( -\frac{\| p_i - p_j \|^2}{2\sigma^2} \right) & \text{if } v_i \text{ and } v_j \text{ are connected} \\
0 & \text{otherwise}
\end{cases}
\]

with \( \sigma^2 = -\frac{R^2}{2\ln\Delta} \); and \( \Delta \) is a small threshold.

The control strategy regarding connectivity maintenance considers the energy function \( E \) (Equation 2) and ensures that \( \lambda \) never falls below the desired algebraic connectivity lower-bound (\( \epsilon \)) (Sabattini et al., 2013).

\[
E(\lambda) = \begin{cases} 
\coth(\lambda - \epsilon) & \text{if } \lambda > \epsilon \\
0 & \text{otherwise}
\end{cases}
\]

The connectivity maintenance algorithm takes the position vector \( p \) of a graph of connected robots and evaluates, for each robot \( v_i \), the velocity vector \( p_i \) which will lead \( v_i \) to a new position without losing its connectivity. The corresponding control input is given by \( u_c^i \triangleq p_i \) and is derived from the gradient descent of the energy function:

---

1. Full HD model Logitech c920, hanging on a metallic support.
2. Self-localization and mapping were not addressed in this work, due to the limited sensing capabilities of the robots.
3. Developed by the authors using Microchip MPLAB X IDE version 4.15 and compiler XC16 version 1.33 which is compatible with dsPIC30F6014A devices, such as epucks.
4. The messages handlings through Bluetooth and computer vision was performed by a software developed (in Java 1.8.0 update 144) by the authors for this purpose and running on Intel i7 vPro 7th Generation with 8 GB and Ubuntu Linux with kernel version 4.15.0-30-generic. Theoretically, the testbed can accommodate 5 epucks per Bluetooth adapter.
5. The sensing range \( R \) is defined as half of the communication range.
\[
\begin{align*}
\dot{u}_i &= -\frac{\partial E(\lambda)}{\partial p_i} = -\frac{\partial E(\lambda)}{\partial \lambda} \frac{\partial \lambda}{\partial p_i} \\
&= -\partial \frac{\partial \cosh(\lambda - \epsilon)}{\partial \lambda} = -\frac{\partial}{\partial \lambda} \left[ \text{csch}^2(\lambda - \epsilon) \right] \\
&= -\frac{2}{\left( \text{csch}^2(\lambda - \epsilon) \right)^2} = \text{csch}^2(\lambda - \epsilon)
\end{align*}
\]  

such that, for \( \lambda > \epsilon \):

\[
-\frac{\partial E(\lambda)}{\partial \lambda} = -\frac{\partial \coth(\lambda - \epsilon)}{\partial \lambda} = \ldots
\]

may fragment the network (Ghedini et al., 2018). Therefore, the topology strategy should aim at increasing this value.

The Voronoi tessellation is an optimal distribution that presents minimal energy regarding the distance among the points. Consider the following energy function (Nair and Guruprasad, 2020; Breitenmoser et al., 2010):

\[
H_i(\rho) = \sum_{i=1}^{n} \int_{\zeta_i} \frac{1}{2} ||q - \rho_i||^2 \phi(q) dq
\]

To find its minimum, we need to equal its gradient to zero:

\[
\frac{\partial H_i(\rho_i)}{\partial p_i} = 0
\]

From Equation 10, we have:

\[
\frac{\partial H_i(\rho_i)}{\partial p_i} = \int_{\zeta_i} \frac{1}{2} \frac{\partial}{\partial p_i} (||q - \rho_i||^2) \phi(q) dq = \\
\int_{\zeta_i} -\phi(q) dq - \int_{\zeta_i} q \phi(q) dq = (12)
\]

\[
\rho_i = \int_{\zeta_i} \phi(q) dq - \int_{\zeta_i} q \phi(q) dq = (13)
\]

From Equations 11 and 13, we conclude that a local minimum-energy state is reached when each node is positioned in the centroid of its Voronoi cell, i.e., \( \rho_i = C_i \).

In order to get the input control regarding the coverage area, it is necessary to create a Voronoi diagram for the region \( \Omega \), to evaluate each centroid and then determine the associated input control:

\[
u_i^q \propto |C_i - \rho_i|
\]  

This means each agent will move from its current position towards its related centroid.

### 2.2 Coverage Area

Another topology aspect addressed in this paper is how to increase the coverage area, avoiding holes and redundantly (overlapped) monitored areas (Figure 4, right). We consider here the proposal from Ghedini et al. (2018) which combines Voronoi Tessellations with an algorithm to close the cell in the boundaries from local information only (Wang et al., 2008; He et al., 2013). To accomplish the control, first it is necessary to partition the space into regions, which is achievable by determining the Centroidal Voronoi Tessellation (Breitenmoser et al., 2010). Let \( \Omega \subset \mathbb{R}^{n \times 2} \) be a polygonal space with boundary \( \Gamma^b \) that corresponds to the operating area of the robotic network, with points \( \rho = [\rho_1, \rho_2, \ldots, \rho_n] \) in that space. Then the Voronoi cell \( \zeta_i \) related to \( \rho_i \) is given by:

\[
\zeta_i = \{ x \in \Omega \mid ||\rho_i x|| < ||\rho_j x||, \forall j \neq i \}
\]

Given a mass density function \( \phi: \Omega \rightarrow \mathbb{R}_+ \), the centroid of \( \zeta_i \), \( C_i \) is given by (Nair and Guruprasad, 2020):

\[
C_i = \frac{1}{M_i} \int_{\zeta_i} q \phi(q) dq
\]
Assume that it is possible to acquire information from the 1-hop and 2-hops neighbors of a given node \( v \). Then, let \( \Pi(v) \) be the set of nodes from which \( v \) can acquire information from shortest paths (\( d \)) with 1 or 2 hops, and let \( \Pi_2(v) \) the set of nodes from which \( v \) can acquire information from shortest paths (\( d \)) with exactly 2 hops (Ghedini et al., 2018). Thus

\[
\Pi(v) = \{ u, v \in V \mid d(u, v) \leq 2 \} \quad (16)
\]

and

\[
\Pi_2(v) = \{ u, v \in V \mid d(u, v) = 2 \} \quad (17)
\]

Lastly, let \( L(u, v) \) be the number of paths between nodes \( v \) and \( u \); and let \( \text{Path}_k(v) \subseteq \Pi_2(v) \) be the set of \( v \) with 2-hop neighbors reachable in at most \( k \) paths. Then

\[
\text{Path}_k(v) = \{ u \in \Pi_2(v) \mid L(u, v) \leq k \} \quad (18)
\]

The threshold \( \beta \) defines “the maximal number of paths between a node \( v \) and each of its \( u \) neighbors that are necessary to include \( u \) in \( \text{Path}_k(v) \).” apud Ghedini et al. (2018).

It is possible to identify the most weakly connected 2-hop neighbors by setting \( \beta = 2 \), which leads us to define a vulnerability level \( (P_2) \) as follows (Ghedini et al., 2018):

\[
P_2(v) = \frac{|\text{Path}_2(v)|}{|\Pi_2(v)|} \quad (19)
\]

where \(| \cdot |\) means the cardinality of \( \cdot \). If a node \( v_i \) identifies itself as vulnerable, then it needs a control law to increase the number of connections towards its current 2-hop neighbors \( \in \text{Path}_2(v_i) \) (Ghedini et al., 2018, 2016). This is achieved by moving each vulnerable node \( v \) towards the barycenter \( \mu(v_i) \) of the positions \( p \in \mathbb{R}^{|\text{Path}_2(v_i)| \times 2} \) of robots in \( \text{Path}_2(v_i) \).

Let \( p_i \in \mathbb{R}^2 \) be the position of node \( v_i \), so its barycenter \( \mu \in \mathbb{R}^2 \) is given by

\[
\mu(v_i) = \frac{1}{|\text{Path}_2(v_i)|} \sum_{j \in \text{Path}_2(v_i)} p_j \quad (20)
\]

and the topology control is given by:

\[
u_i' \propto \frac{\mu(v_i) - p_i}{\|\mu(v_i) - p_i\|} \quad (21)
\]

2.4 The Overall Combined Topology Control

Each one of the topology controls (Equations 7, 14 and 21) can be normalized and linearly combined into a controller by adjusting control gains \( k^c, k^r, k^a \in \mathbb{R}_+ \) such that the overall control action is

\[
u_i = k^c \frac{u_i^c}{\|u_i^c\|} + k^r \frac{u_i^r}{\|u_i^r\|} + k^a \frac{u_i^a}{\|u_i^a\|} \quad (22)
\]

These gains can be empirically adjusted, whether to suppress or boost some topological control aspect. What was exhaustively tested by Ghedini et al. (2017). In this work, several experiments were performed for different sets of gains to observe topology aspects of connectivity, robustness and coverage area.

3. EXPERIMENTS

Three topologies (\( T_1 \), \( T_2 \) and \( T_3 \)), each one with 12 e-pucks, were tested applying different control gains \( (K_i) \), where \( K_i = [k_i^c, k_i^r, k_i^a] \) is the \( i \)-th set of applied gains and \( k_i^c, k_i^r, k_i^a \in [0, 1] \) are the gains for connectivity, robustness and coverage area, respectively.

**Topology** \( T_1 \) was a square configuration with e-pucks equally distant from its two neighbors and with a hole inside the boundary (Figure 5, left). This topology was chosen for having a huge hole inside it that was minimized by the algorithms controls, as shown in bit.ly/topctr01.

**Topology** \( T_2 \) was a topology consisting of a highly connected network with high overlapping in the coverage area (Figure 5, center), thus a dense topology with high connectivity and robustness, but with a low coverage area. This topology was used to check the behavior of the algorithms expected to minimize the overlaps, as shown in sample video available at bit.ly/topctr02.

Finally, the third topology (\( T_3 \)) had several vulnerable nodes (Figure 5, right). This topology was chosen to visually show the algorithms acting to minimize these vulnerabilities, as shown in sample video bit.ly/topctr03.

In the first round of experiments, we applied the gain\(^6\) \( K_1 = [1, 1, 1] \) over topologies \( T_1, T_2 \) and \( T_3 \). In the second one, the gain \( K_2 = [1, 1, 0.5] \) over the three topologies. And finally, the gain \( K_3 = [0, 1, 0.5] \) over topologies \( T_1 \) and \( T_3 \).

Aiming to compare the outcomes of these experiments, let us define Coverage Area (\( A \)) as a function of time and also relative and normalized scores \( S_C \) for connectivity, \( S_R \) for robustness and \( S_A \) for coverage area, each of them as a function of the experiment runtime. Therefore, \( \forall i = 1, 2, ..., n \), we define

\[
S_C(t) = \frac{\text{mean}(\mu(t))}{\max(\text{mean}(\mu(t)))} \\
S_R(t) = \frac{\text{mean}(\Theta(t))}{\max(\text{mean}(\Theta(t)))} \\
S_A(t) = \frac{\text{Area}(t)}{\max(\text{Area}(t))} \quad (23)
\]

4. RESULTS

When \( K_1 \) and \( K_2 \) were applied over \( T_1 \), the topology was meant to keep the connectivity above the lower bound 0.02 and to diminish the hole in the coverage area, while improving robustness. This expected behavior was observed in the experiments, as shown in Figures 6 and 7. Figure 9 (left and center) presents the topology by the end of the experiment.

By reducing the coverage area gain by 50\%, more holes were expected, which is perceived when comparing the outcomes. Additionally, \( K_2 \) over \( T_1 \) converged to a dense topology, similarly to \( T_2 \) at the beginning.

When \( K_3 \) was applied over \( T_1 \), connectivity should not even be upheld, since the associated gain is zero. But in the experiments we verified that robustness gain maintained

\(^6\) Ghedini et al. (2017) presents further discussion about the gain parameter.
connectivity and also diminished the hole in the coverage area (Figure 8). Figure 9 (right) presents the topology by the end of the experiment.

When we applied $K_1$ and $K_2$ over $T_2$, the initially very dense topology was meant to keep the connectivity above the lower bound 0.02, while improving coverage area and robustness. This expected behavior was observed in the experiments, as shown in Figures 10 and 11. Figure 12 (left and right) presents the topology by the end of the experiment.

When we applied $K_1$ and $K_2$ over $T_3$, the topology was meant to keep the connectivity above the lower bound 0.02 and to decrease the vulnerable nodes by improving robustness while maximizing the coverage area. This behavior was observed in the experiments, as shown in Figures 13 and 14. Figure 16 (left and center) presents the topology by the end of the experiment.

When we applied $K_1$ and $K_2$ over $T_3$, connectivity was not meant to be upheld, but improvements on coverage area and robustness were expected. All this was observed in the experiments, as shown in Figure 15. Figure 16 (right) presents the topology by the end of the experiment.

In all experiments, except when happened issues related to a dead battery or light disturbances impairing tracking by...
the camera, the topology controls were able to keep connectivity above the predefined lower bound, while improving robustness and coverage area. The results corroborate the simulations performed by Ghedini et al. (2016, 2017, 2018), proving the effectiveness of the algorithms running in a realistic hardware-based setup. During the experiments, we also experienced robots slipping on the arena and going to an unexpected position. We faced Bluetooth bottlenecks causing small delays in message exchanges, but none of this troubled the results.

5. CONCLUSION

This paper presented an analysis and results for experiments performed to verify the effectiveness of topology control algorithms on a multiagent robotic network in a multi-objective realistic hardware-based scenario. The robotic agents are meant to be controlled in such a way that they have to ensure connectivity maintenance and robustness to failures, while improving the coverage area.

Ghedini et al. (2016, 2017, 2018) proposed an approach to ensure all these aspects and proved it to work in simulated scenarios. The experiments performed in this work were run on a team of e-puck robots and corroborated the simulations, proving the effectiveness of these topology controls with regard to connectivity maintenance, while improving coverage area and robustness. In future work, this validation could be performed in a robotic team with Wi-Fi and more powerful on memory and processing, in such a way that all the procedures — including localization — would be performed locally. We also intend to evaluate the proposed scheme in UAV or satellite teams, considering alternative domains and application scenarios.

ACKNOWLEDGEMENTS

The authors thank FAPESP - Sao Paulo State Research Foundation (proc. no. 2017/02055-8) for the support to carry out this research.

REFERENCES