REGULAR PAPER

Performance Comparison of ROS2 Middlewares for Multi-robot Mesh Networks in Planetary Exploration

Loïck Pierre Chovet[1] · Gabriel Manuel Garcia[1] · Abhishek Bera[1] · Antoine Richard[1] · Kazuya Yoshida[2] · Miguel Angel Olivares-Mendez[1]

Abstract

Recent advancements in Multi-Robot Systems (MRS) and mesh network technologies pave the way for innovative approaches to explore extreme environments. The Artemis Accords, a series of international agreements, have further catalyzed this progress by fostering cooperation in space exploration, emphasizing the use of cutting-edge technologies. In parallel, companies across various sectors' widespread adoption of the Robot Operating System 2 (ROS 2) underscores its robustness and versatility. This paper evaluates the performances of available ROS 2 MiddleWare (RMW), such as FastRTPS, CycloneDDS and Zenoh, over a mesh network with a dynamic topology. The final choice of RMW is determined by the one that would most fit the scenario: an exploration of the extreme extra-terrestrial environment using a Multi-Robot Systems (MRS). The conducted study in a real environment highlights Zenoh as a potential solution for future applications, showing a reduced delay, reachability, data overhead and CPU usage while being competitive on the RAM usage over a dynamic mesh topology.

Keywords ROS 2 · RMW · Mesh Network · Decentralized · MRS

1 Introduction

Multi-Robot Systems (MRS) have become popular in recent years due to their potential for improving efficiency, flexibility, and robustness in various applications. The application of MRS is already extensively studied in many fields such as agriculture [1], search and rescue [2], hospital logistics [3] or environmental monitoring [4]. However, research in MRS still requires extensive work and normalization of the existing technologies with the goal of reaching interoperability.

The Space industry focuses on MRS, especially with the growth of In-Situ Resources Utilisation (ISRU), which promotes MRS as a promising tool in a growing market [5]. Future missions focus on exploring celestial bodies and require teams of robots to roam in extreme and unknown environments to provide a map and discover potential areas of interest. The work described in [6] distinguishes four primary architectures for MRS: the Centralized approach, the Hierarchical approach, the Decentralized approach, and the Hybrid approach. Decentralized and Hybrid heterogeneous MRS are especially relevant for this context because such applications require redundancy, scalability, and adaptability. However, those approaches face various challenges, such as more complex coordination due to their complexity. The work in [2] shows that a decentralized MRS offers way more scalability and adaptability than centralized MRS. By providing scalability and a better distribution of the resources, MRS offers a reliable approach for ISRU [7].

Loïck Pierre Chovet and Gabriel Manuel Garcia contributed equally to this work

✉ Loïck Pierre Chovet
  loick.chovet@uni.lu

  Gabriel Manuel Garcia
  gabriel.garcia@uni.lu

  Abhishek Bera
  abhishek.bera@uni.lu

  Antoine Richard
  antoine.richard@uni.lu

  Kazuya Yoshida
  yoshida.astro@tohoku.ac.jp

  Miguel Angel Olivares-Mendez
  miguel.olivaresmendez@uni.lu

1 SnT-SpaceR, University of Luxembourg, rue Richard Coudenhove-Kalergi, L-1359 Luxembourg, Luxembourg

2 Space Robotic Lab, Tohoku University, Aoba Ward, 980-8577 Sendai, Miyagi, Japan

Existing approaches even highlight the possibility of a *coopetitive* system, where robots from various stakeholders are on the same mesh network [8]. These robots have the opportunity to collaborate or cooperate autonomously. However, the space environment caters to additional technical issues and constraints. Since the planetary surface is less cluttered with obstacles and interference, the coverage of a robot is larger. However, this implies communication delay and lower data overhead. Connectivity maintenance is the most crucial aspect of a space system, losing the connection to a robot is not acceptable since it couldn't be recovered [9].

Many companies are proposing robotic platforms that can perform several tasks, but interoperability is a need. The open robotic foundation proposes this through Robot Operating System (ROS) 2, the newest version of ROS, focusing on aspects such as ROS and scalability [10]. ROS 2 is characterized by enhanced security features, real-time capabilities, and a flexible and dynamic architecture, allowing robots to interact in complex and dynamic environments. Most of the features are brought by the use of the Data Distribution Service (DDS) technology, acting as a ROS MiddleWare (RMW) between ROS 2 and the network layer.

DDS is a communication protocol often used in domains such as automotive, defence, finance, and simulation [11]. DDS plays the role of messenger that allows different parts of a system, such as robots in a MRS, to talk to each other effectively through mechanisms of subscription and publishment. DDS can handle large amounts of data, deal with complex communication patterns, and is designed to work well even in challenging environments. Therefore, DDS has become pivotal in MRS because it ensures that all the actors of a system are connected, updated, and working together seamlessly [12]. However, alternatives to DDS are emerging, such as Zenoh [13] and propose newer approaches with announced better performances [14].

Along with the Open Systems Interconnection (OSI) model, ROS 2 and the middleware rely on the network layer to work. If many network architectures exist, exploration requires some specificity, such as scalability and the ability to react to topology changes. In the traditional star topology of wireless networks, each device connects to a central node. The mesh networks allow each device, or "node", to communicate with all other participating nodes in the network either directly or via multi-hops. Therefore, it enhances the robustness of the network by offering multiple pathways for data transmission. Consequently, mesh networks are more reliable, as the failure of a single node rarely leads to a breakdown of the entire network. In addition, mesh networks are highly scalable, allowing the integration of more nodes without significant degradation in network performance [15]. This feature is critical in MRS, where the number of robots may vary depending on the task at hand. Despite these advantages, one of the biggest issues of the mesh networks lies in resource utilization. It requires more antennas to operate simultaneously compared to a centralized system where there would be one main antenna, leading to more power consumption. It also induces additional delay and potential consistency and synchronization issues. Middlewares such as DDS play a crucial role in MRS. To operate in extreme environments, these MRS need to be supported by meshed networks. Therefore, it is important to study and assess the impact of different middleware on such configurations. Hence, this study can guide the selection of the right RMW for a specific mission.

This paper investigates the impact of using mesh networks and ROS 2 alliance for MRS-driven missions in extreme environments. The applied research focuses on a data-sharing scenario in extreme environments, implying some specific constraints such as long-distance communication, potential obstacles and topology changes. It also poses issues with energy consumption and sensing due to the unexpected environments. The contributions of this work are as described below :

- We compare the different RMW technologies, presenting a Non-Line-of-Sight (NLOS) topology.
- We present guidelines to choose the most suited RMW depending the mission, for a MRS operating on a mesh network.

The paper is structured as follows. The Section 2 explores the state-of-the-art of MRS and Mesh Networks. The Section 3 highlights the needs of the state of the art. The Section 4 details the robot agnostic architecture proposed. The Section 5 presents the scenario where this architecture would be applied. The Section 6 explains the experimental setup used for the quantitative study. The Section 7 displays the results. Finally, Section 8 discusses the results and future works.

2 State of the Art

This section presents the details and state of the art of the various technologies involved in this paper.

2.1 Decentralized MRS and ROS 2

As explained in [6], we can distinguish four primary architectures of MRS :

- Centralized: This approach coordinates all the robots from one single point. The solution focuses on the most optimized approaches since all the data for decision-making are known in one place. It also allows the use of simpler robots designed for highly specific tasks.
- Hierarchical: In this approach, an overseer is always overseeing a small group of robots, where some of them can

also oversee a team. This allows the architecture to be strengthened against global failure.

- Decentralized: The decentralized approach allows each robot to make its own decisions independently. It is, by design, the strongest architecture but often leads to suboptimal results and a higher difficulty in sharing complex tasks.
- Hybrid: This approach combines the previous approaches with the goal of allying robustness with efficiency.

Most of the studies done on the decentralized approach could also relate to the hybrid. The recent improvements in service-oriented architectures (SOA) allow us to consider more economic interest in Sensing-as-a-Service and Robot-as-a-Service. SOA is a design paradigm that allows services to communicate over a network through a loosely coupled relationship to facilitate business processes and data exchange. This would allow any client to use the functionalities of any robots available [16]. For example, a research institute on earth could use the service of a robot already present on-site instead of creating its own mission, creating economic interest for the research institute and the robot owners [8]. This application facilitates special interest in extreme environments where missions are expensive and hardly reachable by humans, such as space exploration and ISRU. However, due to the cost of launching and designing similar missions, it is expected to have multiple robots from multiple owners on astral bodies. The robots from different owners still need a way to collaborate to propose an optimal SOA. However, the robots may remain from different companies and must keep their independence. This state has been studied since 1996 by [17] and is called coopetition. In the case of a Coopetiting system, the decentralized or hybrid architecture is the only architecture leaving enough freedom for each robot, or consortium to take its own decisions.

The authors in [18] extensively studied cooperative heterogeneous and decentralized robotics and presented most of the current applications. They highlighted the main concerns of decentralized MRS, which are :

- The Control architecture: How to ensure a global behaviour when robots only have local information. They classify three different controls: Reactive, Deliberative, and Hybrid.
- Consensus: Finding a consensus without a proper leader is still a research topic.
- Containment: The robot should be less interconnected to avoid a global failure.
- Formation: Robots must learn how to move in groups.
- Task Allocation: Most of the research focuses on this aspect.

- Intelligence: With the growing research in AI, the issue of how and if robots should share training data and learning is a hot topic.
- Optimisation: Due to the intrinsic suboptimal aspects of decentralized MRS, it remains one of the principal concerns.
- Communication: Two main approaches exist: intrinsic, where robots communicate through sensors, and explicit, where the communication is done on a communication channel. Most applications rely on the second, while the other one is destined for specific use cases; it suffers from many issues, such as limited distance or the obligation to add sensors.

The authors insisted on the need for more research in the field of decentralized MRS **communication**. The main challenges in communication are:

1. Efficiency
2. Environment tolerance
3. Consensus for interactions
4. Optimizing Speed and Energy

In the case of space exploration, environment tolerance and optimization of energy consumption are the biggest challenges of decentralized MRS, where the centralized counterparts currently offer better solutions and more developed approaches.

One of the first applications of a decentralized MRS was presented in [19] with ACTRESS. ACTRESS is based on *Universal Modular ACTOR formalism* by [20]. The system describes multiple autonomous components called robotor, which can communicate in peer to peer to any other robotor. This architecture has been designed to allow any autonomous architecture to join the system. This architecture already presents the interest of a decentralized MRS using a networking technology allowing peer-to-peer communication with its neighbours.

The author in [21] presented ALLIANCE, a software architecture focusing on fault tolerance, often referenced as a basis for more recent works. It focused on scenarios where a robot loses the capability to achieve its tasks, and the other robots recover the task.

The number of effective space robotic missions remains limited, but some scenarios have been proposed to study solutions to face the limitations posed by lunar exploration. The project LUNARES [22] involved a heterogeneous team of robots for autonomous data sampling and represents the interest of the scientific community for MRS in the context of space exploration. In the project ARCHES [23], researchers from the DLR demonstrated various technologies

and collaborative processes during a space analogue mission at Mount Etna. They performed sample collections through a heterogeneous multi-robot collaboration, demonstrating the efficiency of multi-robotic teams for space missions.

With the objective of motivating companies and laboratories to explore new solutions for space resource exploration, the European Space Agency (ESA) and the European Space Resources Innovation Centre (ESRIC) created the Space Resources Challenge [24]. Among the various implementations, REALMS2 [25] offered a decentralized approach to tackling space-related challenges such as communication delays.

To address the evolving needs of the robotic industry, the Open Robotics Foundation introduced ROS 2. Unlike its predecessor, ROS 2 is designed to provide reliable communication over distributed systems. This makes it particularly well-suited for MRS operating in harsh or remote conditions. ROS 2 uses a middleware-oriented network approach for high-performance inter-node communication, ensuring data integrity and timeliness crucial for coordinating complex robot tasks in unpredictable scenarios. ROS would require a centralized master to handle the system, making it hard to conceive a decentralized system without using workarounds. Thanks to the DDS, the default middleware for ROS 2, most ROS 2 nodes can communicate with any other node in the same network. DDS is a middleware protocol and API standard from the Object Management Group (OMG) for data-centric connectivity. Its core system is based on the Data-Centric-Publish-Subscribe approach and focuses on the real-time environment and scalable systems [26]. More details about the various middleware implementations of ROS2 can be found in Section 4.2.

Multiple works exist on MRS and ROS 2; however, the focus is mostly on the control architecture [27], leaving the concern of communication with a lack of research. Often, the communication is assumed to be peer-to-peer with no flaw, which is far from what is expected in the context of extreme environment exploration when it is highlighted that communication resource optimization is one of the biggest disadvantages of the decentralized system compared to the centralized approaches.

2.2 Mesh Networks

Mesh networks are quite common in large-scale infrastructures and are the default topology constructed by separate entities connecting to each other. In the scenario of different robots owned by companies that share interests but not property over the system, the default architecture constructed by connecting all those companies will result in a mesh network. According to [28], the resulting network topology is

the best trade-off between security, neutrality and the following points:

- Due to their decentralized nature, mesh networks are more **robust and reliable**. If one node fails or becomes disconnected, the network can dynamically reroute traffic through alternative paths, ensuring reliable connectivity.
- Mesh networks can easily **scale** up by adding additional nodes without causing significant disruptions or requiring extensive reconfiguration. Each new node increases the network's capacity and coverage area.
- Even more, with multiple paths available for data transmission, mesh networks provide built-in **redundancy**. This redundancy enhances fault tolerance and minimizes the impact of node or link failures, improving network performance and operational time.
- Mesh networks excel in providing wide and **extended coverage**, particularly in large or challenging environments where traditional network setups may encounter limitations. The ability of nodes to communicate directly with one another facilitates broader coverage without relying on a centralized infrastructure.
- Mesh networks can be easily deployed and reconfigured to suit various scenarios due to their inherent **flexibility and adaptability**. They are suitable for dynamic environments where network nodes may be mobile or where the network's structure needs to be adjusted to accommodate changing requirements.

MRS can operate in various and challenging environments that require higher flexibility than traditional-centred networks. The extended coverage and resilience of that network are crucial points that MRS greatly appreciates, as each robot is considered an antenna expanding the area used by the robots. Depending on the environment, rock formation can present obstacles to communication. However, with a fleet large enough, the complex topology adaptability of the mesh can tackle these challenges. If a path becomes too long for the packages to go through, the mesh protocol can find a new path going through other robots. Furthermore, this **flexibility and adaptability** of the network allows robots to be automatically reconnected to their pairs during a running experiment if the topology of the network changes.

As highlighted by the literature [28, 29], one of the biggest issues with mesh networks is scalability since studies lack information about their behaviour for too large networks.

In the following subsection, an introduction to the most commonly used mesh routing protocols will be achieved.

2.2.1 AODV

AODV stands for Ad-hoc On-demand Distance Vector, developed in 2003 as a low resource utilization routing

protocol. It was designed with low network device resource usage and with the mindset to maximize packet communication between different devices [30].

2.2.2 B.A.T.M.A.N Advanced

B.A.T.M.A.N Advanced stand for Better Approach To Mobile Ad-hoc Networking (Advanced) and is a low-level decentralized routing protocol operating on the second OSI layer. This routing protocol is a part of the Linux Kernel and only needs to be activated as a module to run on every device, even low-end ones [31].

Each Linux device, with the B.A.T.M.A.N Advanced module activated, will act as a node equipped with a switch over the network. This solution assumes that the devices used are running on an OpenWRT system, which provides a configurable and writable file system with package management.

B.A.T.M.A.N development was derived from the Optimized Link State Routing Protocol (OLSR), which is mostly like AODV proactive routing protocol [31]. The main difference between those two protocols is the solution used to pass information: Multipoint Relay (MP) in the case of OLSR and AODV and through available neighbours in the case of B.A.T.M.A.N.

2.2.3 HWMP and HWMP+

Hybrid Wireless Mesh Protocol (HWMP), as described in [32], is defined by the IEEE standard 802.11s as a standard meshing protocol, which also defines the default metric as the airtime link. On each link, the metric C_a described in [33] is computed as follow:

$$C_a = [O + \frac{B_t}{r}]\frac{1}{1 - PER} \tag{1}$$

With PER (Packet Error Rate):

$$PER = \frac{Frame\,Resent}{Frame\,Sent} \tag{2}$$

Where O represent the channel access overhead, B_t is a frame of standard size, and r is the transmission rate.

Later, HWMP+ was developed with the improvement of the PER computation in mind to get a more accurate estimation of the link quality and, as a result, allocate resources efficiently [33]. The authors of this solution [33] emphasized that HWMP+ is significantly more efficient than the previous solution, with an increase of 3%-6% additional receiving packets; hence, it generated 2.8% to 10.4% less routing overhead in general scenarios.

The author of [33] highlighted that most of the metrics used for the Mesh system are less adapted to variable wireless links that can be affected by noises, interferences and

even the limited energy of the nodes. The HWMP+ protocol proposed in this paper is designed with respect to a dynamic environment and real-time constraints.

Finally, Mesh Networks have already been experimented with in extreme environments. The winning team of the Darpa Subterranean challenge, Cerberus, used various robots dropping "breadcrumbs". The "breadcrumbs" are relay nodes, made of an antenna and a battery, dropped in strategic places to ensure connection, creating an extended Mesh Network [34].

2.3 Exploring the Performances of ROS 2

The authors of [35] studied the network properties of ROS 2 and its behaviour depending on various DDS implementations. They point out that using the DDS technology makes it suitable for real-time distributed embedded systems and scalable. However, it also comes with an overhead since all ROS 2 messages must be converted to the DDS format before. This overhead and the resulting latencies are the main metrics studied in this paper.

It is important to note that this paper was published in 2016 and that the experiment was realized on an alpha version of ROS 2. Also, it focused on three DDS: FastDDS, OpenSplice, and Connext, while the current implementation of ROS 2 presents FastDDS, Connext, Cyclone, and Gurum. And that Zenoh, a RMW different than DDS is foreseen to be implemented. These RMW's are detailed in Section 4.2.

During various experiments, the authors of [35] highlighted that ROS implied a loss of the initial message, whereas ROS 2 fixed this issue. They also presented different experiments to compare the three DDSs. Using a DDS in ROS implies delay, which grows exponentially as the message size increases. The experiments showed that FastDDS could not share a message bigger than 256 bytes. However, it is also shown later that FastDDS uses fewer threads to run a ROS 2 node, the same amount as ROS1.

They concluded that even if ROS 2 implies an overhead compared to ROS, it can be overcome using its QoS Policy and not by using a master-node policy. They also highlight that FastDDS is, at that time, the most suited DDS application for real-time distributed embedded systems in terms of threads and memory.

3 Research Gap

As highlighted in state of the art, MRS is a widely studied topic, and ROS 2 is proposing a strong solution for developing and studying strong control architectures for various situations. However, there is often a lack of communication architecture, and the existing works are based on a hypothetical perfect case. However, bigger and more complex MRS

architectures can lead to bottlenecks and unexpected situations on unadapted networks. More studies are needed on network architecture for robotic experiments and to propose some simple-to-implement architecture for any MRS application using ROS 2.

4 Architecture Proposed

4.1 Network Architecture

We consider a heterogeneous MRS deployed in an outdoor area to perform a set of tasks. A set $R = \{r_1, r_2, r_3, ..., r_n\}$ consists of n different robots connected through a mesh network M and all participating in it. For $i \in \{1, n\}$, a robot r_i is composed of $r_i = \{c_i, s_i, a_i\}$ with c_i a set of m embedded computers $c_i = \{c_{i_1}, ..., c_{i_m}\}$, a network switch s_i, and a mesh antenna a_i. The embedded computers represent each device connected to the mesh that is directly on a robot. The focus of this system is to propose high adaptability to all robotic platforms by connecting all the embedded computers to the switch, which is itself connected to the mesh router. We also consider a set of l operators and each operator is represented as $o_k = \{c_k, s_k, a_k\}$, where $k \in \{1, l\}$, c_k denotes associated computer, s_k refers the associated network switch, and a_k the respective antenna. In that case, the mesh network comprises n robots and one operator o_k is represented by set $M = \{a_1, a_2, ..., a_n, a_k\}$. Figure 1 presents the proposed architecture.

We consider that all robots have various capabilities, such as movement, data collection, and data treatment. They also have to face real-time mechanics.

As explained in the state of the art, HWMP+ is designed with large dynamic systems in mind. According to [33], HWMP+ has better metric computation, which results in less packet loss, reduced delay, and higher throughput compared to the standard HWMP. This is particularly noticeable in dynamic environments. Among the different mesh protocols, HWMP+ presents the fitting properties for the architecture proposed here, which will be the mesh network used.

4.2 ROS 2 Architecture

Every embedded computer c_{i_j} for $j \in \{1, m\}$ operates on a specific set of ROS 2 nodes that run the chosen RMW implementation. In its latest release, ROS 2 provides native support to two free DDS implementations, which are listed below:

- **FASTDDS**: Also called FASTRTPS, this DDS has been used with ROS 2 since its beginning, always proposed as the default version. Since the implementation described in [35], there have been significant changes in its implementation, such as using shared memory transport, which allows the transport of larger messages [36]. The authors announce a delay below $20\mu s$ for packets up to 15kB. According to their studies, in ideal conditions, they also present a throughput in the order of magnitude of $10^4 MB/s$.

- **Cyclone DDS**: Eclipse Cyclone DDS is designed as a free and open-source middleware. It is focused on high throughput and low delay, even if no official data are provided. The developers of FASTDDS have conducted a comparison with Cyclone, as cited in [37]. For intra-process communication, FastDDS presents significantly

Fig. 1 Mesh Network architecture
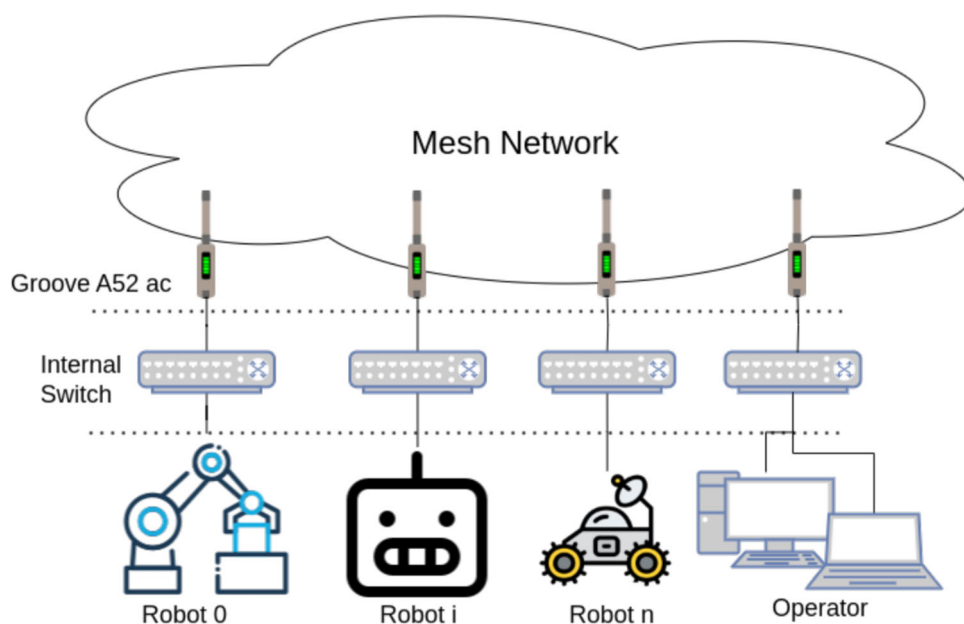
better delay results, especially for larger payloads. In the case of inter-process communication, FastDDS claims to have a delay of around 30% lower than Cyclone. They also highlight the stability of FastDDS, represented by the low delay increase as the payload gets larger. In terms of throughput, both DDSs present the same results.

Another possibility, commonly used by MRS, is to rely on another middleware than DDS. Eclipse Zenoh is very promising in this way, being close to natively implemented in the latest version of ROS 2. It offers the first easy-to-use RMW not relying on DDS. This technology is designed to streamline the discovery process and improve efficiency. During the discovery phase, robots attempt to identify the topics of other robots, which can be a crucial network operation. Fortunately, Zenoh has been developed to significantly reduce the overhead required for this process, cutting it down from 97% to 99.9%. This technology delivers results similar to those of DDS while minimizing the required resources. [38] Discovery overhead is frequently cited as a major problem with DDS technology in wireless networks, as it can cause network downtime.

In the paper [39], the authors explained that Zenoh is less likely to cause network failure compared to typical ROS 2 DDS. It is specifically created to operate smoothly even in unstable network conditions. One other advantage of Zenoh is the existence of pico Zenoh, a version compatible with microcontrollers. The only DDS implementation offering microcontroller-compatible features is FastDDS; however, various robotic applications use microcontrollers for motion. In the case of the usage of Zenoh in the architecture, all the Robot Operating System (ROS) 2 nodes within the robot r_i should share the same domain id D_i, but the domain id of each robot r_i should be different, $D_i \neq D_j$, $\forall \{i, j\} \in \{1, n\}(i \neq j)$. We note that all the available studies and comparisons [36, 37] use Gigabit Ethernet connections and no quantitative data are given for Wi-Fi, mesh networking or any other wireless technology such as 5g/6g.

5 Problem Scenario

In our experiment, we only evaluate the performance of RMW. However, depending on the scenario, the ideal RMW might differ. This is why we created a fictional yet realistic scenario that will allow us to select the ideal RMW. For this fictional scenario, we consider a group of robots exploring a harsh environment, specifically a lunar environment. They extend the group capabilities by working together in a decentralised manner while also maintaining their independence. Each robot continuously generates and transmits sets of data called 'messages'. As an example, these messages could represent small chunks of a point cloud acquired by various sensors. Multiple robots are expected to share their data (i.e. point clouds) for merging or optimisation operations [40, 41]. In order to map even more efficiently, a robot can obtain the point cloud data of another robot to prevent the need to map an area already visited, thus conserving energy. The state-of-the-art provides many mapping, task allocation, and control solutions [42]. However, most applications are based on simulations or highly controlled networks, and the networking aspects are often considered out of the scope. Especially given the recent developments with ROS 2. The state-of-the-art needs more propositions of MRS-proof network architecture. Currently, many works and projects rely on access points and utilize the default settings of Robot Operating System (ROS) 2. As stated in [43], evolving network topologies remain an open question. As a result, there is a need for a study about the network aspects of MRS relying on a decentralized architecture. In this context, we use Robot Operating System (ROS) 2 to implement a decentralized architecture. The main concerns of this problem to be surveyed are:

- Data throughput
- Stability of a robot on the network
- Power consumption

6 Experiments

6.1 Experimental Setup

6.1.1 Hardware

Most of the embedded computers come with an internal antenna. The straightforward approach would be to use the embedded antenna and run the routing protocol on the embedded computer. However, this solution faces multiple issues.

- Limitations: Internal antennas offer less data throughput and a limited range due to power and space limitations.
- Resource usage: Embedded computers are optimized for specific purposes and are usually used for computationally intensive tasks such as image processing. Introducing a network protocol layer would increase the amount of computation required and make the system more susceptible to internal crashes.
- Bottlenecks: Robotic systems frequently incorporate multiple computers within the platform, all connected via Ethernet. However, relying on one computer to manage all networking tasks could result in significant data throughput limitations and reduce efficiency.

On the other hand, using an external router presents multiple advantages. The first one lies in using a dedicated internal board designed for network computation, allowing a better process of the packages and optimized routing. An external power source can provide a more significant data throughput and broader coverage. One of its key features is the ability to quickly implement any networking protocol, which is crucial when experimenting. Notably, using an external antenna can increase power consumption, which should be taken into account as a potential concern.

We use the *Groove A52 ac* router from the brand *Mikrotik* [44] as the mesh node for every agent of the MRS. This router offers more than 200 meters of coverage and is compatible with the norms 802.11ac (Wi-Fi 5). The router's design also meets the need for outdoor robotics, with a weatherproof design that easily fits all current robots. Finally, it can be powered over ethernet, providing a more straightforward wiring and power distribution approach.

The architecture proposed in Fig. 1 is highly scalable and adaptable to any robot. Since many robots rely on multiple embedded computers, they all need to be connected to a Switch connected to the Groove A52 AC. The router used in our setup is an *EdgeRouter X* from the brand *Ubiquity* set up as a switch [45]. However, any switch providing power over ethernet is suitable.

The robots used for the experimentations are upgraded LeoRovers [46] as in Fig. 2, equipped with an RGBD *intel realsense_d455* camera [47], a *RPlidar* [48], an embedded computer used for the computational heavy tasks *Nvidia Jetson Xavier* [49] and the setup previously mentioned.

The overall system is designated as Resilient Exploration And Lunar Mapping System 2 (REALMS2) and has been successfully tested in a real environment during the ESA-ESRIC Space resource challenge as detailed in [25]. This experiment benefits from this architecture however it would be possible to repeat the result only with a *Raspberry Pi* and a *Groove A52 ac* router.

6.1.2 Software

As explained in [33], IEEE 802.11s defines HWMP as the default mesh protocol. The authors of [33] propose HWMP+ as a version more adapted to dynamic topology and optimising the shared channel resources. Most of the existing work was conducted in simulation and now requires further examination in mobile robotics.

To establish the mesh network between the robots, Mikrotik provides instructions [50], creating a 2.4GHz Wi-Fi mesh network relying on the HWMP+ protocol. The **2.4GHz** band is chosen for its higher resistance to interferences and longer range. The authors of [33] propose HWMP+ as a version more adapted to dynamic topology and optimising the shared channel resources. Most of the existing work was conducted in simulation and now requires further investigation.

All the network communications of ROS 2 are handled by a RMW, a middleware providing a publish-subscribe model for sending and receiving data, events, and commands among the nodes. The RMW solves many issues when it comes to distributing real-time data, sending and receiving data over a network, and providing consistency in the data model. It is acting on layers 5 and 4 of the OSI network model. By definition, the mesh protocol assures the data link, so layer 2 of the OSI model. This means that there is no difference in ROS 2 between a mesh network and a regular network.

All the embedded computers are running on Linux Ubuntu 22.04 and use ROS 2 Iron distribution.

6.2 Metrics

We consider multiple metrics to compare and validate the more appropriate system. The first set of metrics allows us to quantify the network during the two crucial steps of the scenario, which are as follows:

- **Data throughput**: Assessing the data throughput provided by the mesh network is crucial to fully understand how different middlewares work together.
- **Jitter**: The delay measurements vary along a measurement. This variation is called the jitter and indicates the stability of the network. In a system with low jitter, messages consistently take about the same time to travel from sender to receiver.

The second set of metrics is measured during the whole experiment, measuring the performances of each RMW implementation :

- **Reachability**: Since the experiment consists of sending data from a robot to a lander, this metric evaluates when

Fig. 2 Experimental robotic platform
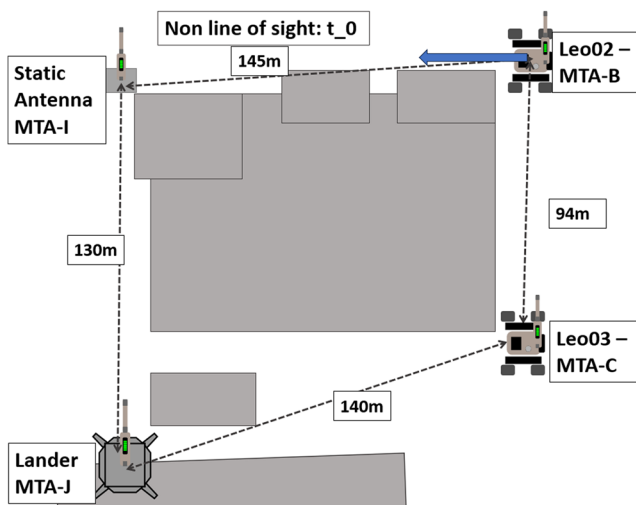
(a) Initial configuration of the experiment.
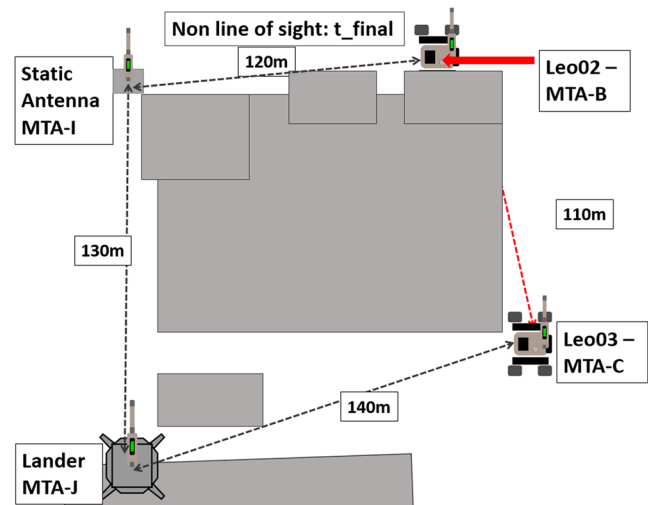
(b) Final configuration of the experiment. No connection between Leo02 and Leo03.

Fig. 3 Experimental scenarios

the robot is reachable from the lander. A device is considered reachable from another device if a connection can be established. We define the average reachability $R(t)$ for an RMW as follows: With $t \in [0, 120]$, $\tau_i(t) \in \{0, 1\}$:

$$R(t) = \frac{1}{N} \sum_{i=1}^{N} \tau_i(t) \tag{3}$$

with $\tau_i(t)$ the reachability for a given run i, N the total number of run for the RMW. The average reachability

Fig. 4 Urban terrain used for the first experimental scenario, the width of the visible environment is 175m

$R(t)$ is a floating-point value between 0 and 1. Higher values indicate a more stable connection.

- **Data overhead**: During the experiment, the robots will share messages of a fixed size over the network. However, each technology adds its own headers and additional content to the message. This metric studies the size of this overhead depending on the size of the message to quantify the network impact of each middleware.
- **Delay**: The delay is a crucial metric for real-time systems, representing the time it takes for a message to travel between a publisher and a subscriber.
- **CPU Usage**: Most embedded computers have limited resources, so monitoring the usage of the resources by the various middleware implementations is helpful.
- **RAM Usage**: Along with the CPU Usage, the RAM usage remains crucial since the memory is shared between many of the running processes and should remain available.

6.3 Experimental Scenario

As stated previously, ROS 2 handles communication through different middlewares. Since all the existing studies present results only in ideal conditions, we consider an approach that pushes the different middleware to their limits. As detailed in Fig. 3, the experiment involves two robots (Leo02 and Leo03), a lander, and a static antenna. The chosen environment is signal-wise noisy and filled with obstacles in the urban environment (Fig. 4), pushing the network to its limits.

During the experiment, all used devices on the mesh network use IRON ROS 2 distribution. Furthermore, Leo02 is

Table 1 Leo Rover composition

	Internal	External
CPU	Raspberry Pi 4B	Raspberry Pi 4B
Memory	4GB	4GB
OS	Ubuntu 22.04	Ubuntu 22.04
ROS Distro	Humble	Iron

composed of two embedded devices, one internal and one external, as described in Table 1. The internal onboard computer is used for low-level control (rover teleoperation) and is accessed out of the mesh network whereas the external computer is in charge of the communication of the data, and this is where the RMW changes take place. However, Leo03 is only used as a relay point.

According to the authors in [35], quality-of-service (QoS) is one of the new technologies added by ROS 2 and significantly impacts network performance, especially with large data messages such as point clouds. Any software using ROS 2 uses a QoS profile, and choosing the appropriate one can affect the networks. For this work, we consider the default QoS profile as only the RMW is evaluated and not the different available QoS.

The experiment is divided into the three evaluated RMWs that are each subdivided into seven fixed-sized messages (from a kilobyte to 64 kilobytes). This upper limit of 64 kilobytes has been found by preliminary tests, larger message sizes break the mesh network. To improve the quality of the result and get a proper sample, five successful runs are completed per fixed-size messages leading to 3x7x5=105

runs. A run has a duration of 120 seconds and consists of Leo02 moving toward the static antenna while sending fixed-size messages to the Lander. This trajectory, Fig. 3b, has the purpose of changing the network topology (because of the surrounding buildings) from a square shape (Fig. 7a) to a line shape (Fig. 7b) and investigate its impacts over each RMW. The messages sent on the network allow us to monitor the Key performance indicator (KPI)s seen in Section 6.2.

7 Results and Discussion

As explained in Section 6.3, each RMW has been studied with seven message types over five distinct runs. Figure 5 demonstrates an overview of the experiment, including the robot's trajectory given each RMW. Figure 6 presents the network overview of each of these singular runs. Both figures emphasise the connection lost between 60 and 90 seconds, allowing the network to change its topology.

The Fig. 5 has a zoom on the trajectories of each RMW. The missing values in the trajectories represent the loss of connection of the rover. During those 'blackout' moments, the network is either saturated or in reconfiguration. A closer look at the trajectories allows us to identify three different phases. During the first phase (I), the rover is connected to the mesh network in a full square topology (see Fig. 7a) and therefore enhances good communication of the packages over the mesh allowing multiple routes. The second phase (II) is the transition phase where the mesh changes its topology due to the moving antenna on Leo02. During this phase, loss of connection, delayed messages and an unstable
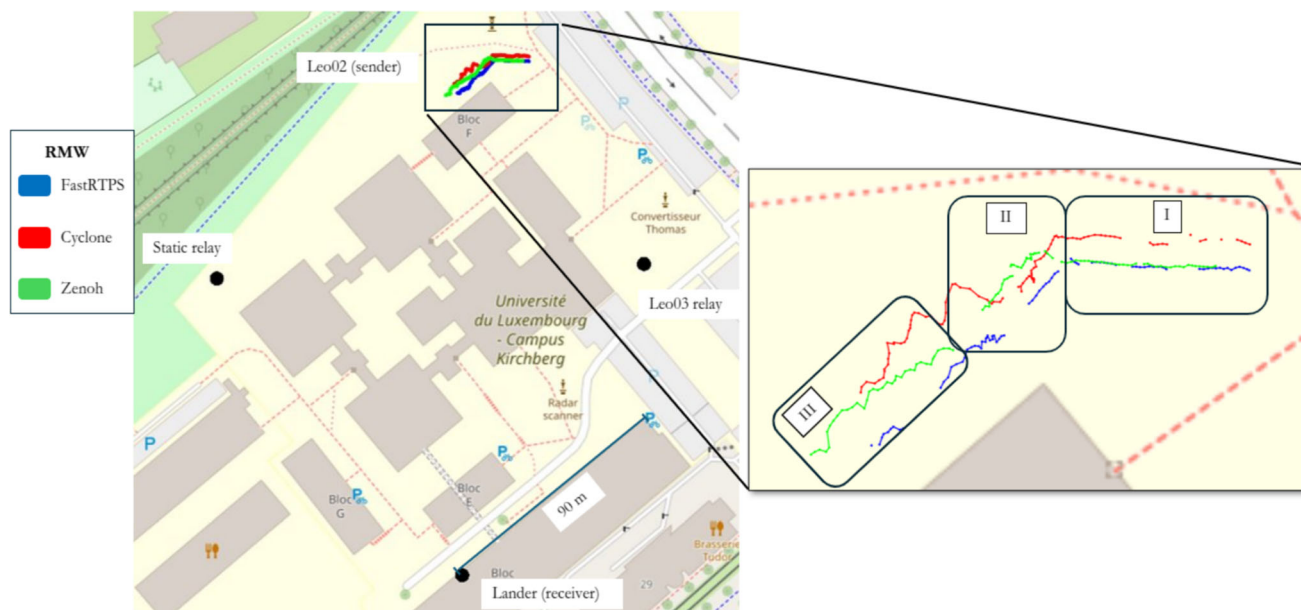
Fig. 5 Global overview of the experiment
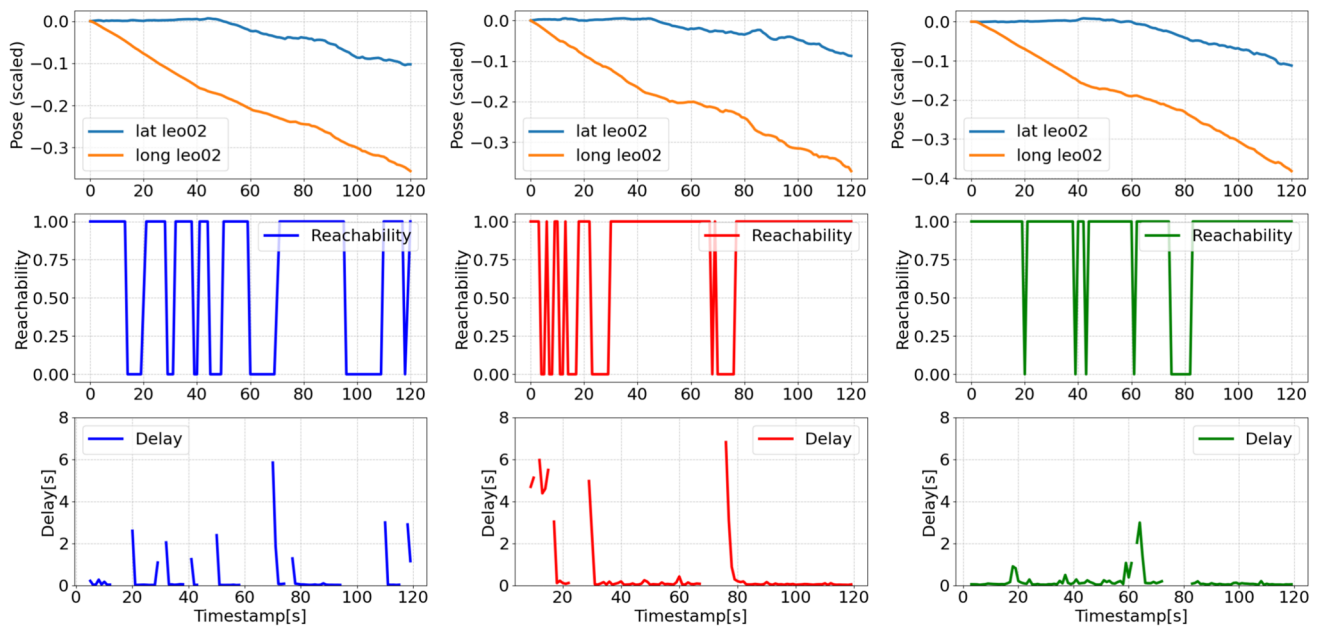
(a) Fifth run of Fast DDS   (b) First run of Cyclone DDS (c) third run of Zenoh RMW

Fig. 6 Overviews of a singular run for each RMW for a message size of 16 KiloBytes. Trajectory, reachability and delay are shown

network is expected. Finally, the third phase (III) shows a reconfiguration of the network in 'line' (see Fig. 7b).

In addition to the map overview, the Fig. 6 shows how the network behaves during the experiment. The top graph represents the position of the rover related to its starting point, the middle graph is a metric called 'reachability' (if the rover is connected to the network or not), and lastly, the delay. Missing data in the delay graph are due to connection loss between the rover and the mesh network.

A GUI tool helps us to keep an overview of the network topology during the experiment. Figure 7a displays the topology at the beginning of the experiment (t=0), while Fig. 7b displays the topology at the end of the experiment (t=120s). This 120-second duration was decided after measuring the time for the network to change its topology while driving at a speed of 0.5m/s. After measurement, 75 seconds was enough

to reorganize the network. However, all the runs have a duration of 120 seconds to fully understand the behaviour of the network before and after the change in topology. Table 2 presents the average capabilities of the mesh network at the beginning of the scenario (t=0) and the end of the scenario (t=120).

Since the **packet loss** and **delay** of the ping command do not vary in any scenario, they are not included in the table. It is noticeable in Table 2 that the distance between the robot and the relay impacts the data throughput and that relays should be used carefully. This behaviour can be explained by the number of packet losses: the further, the antennas are spread, the higher the chances of packet loss. Therefore reduced usable bytes are received per second.

In the scenario where node A and node B want to communicate, we learn that if node C wants to offer relaying services,

Fig. 7 Result of the network topology analysis during the experiment done with a GUI. A circle represents a node of the mesh network (antenna) connected to its direct neighbour
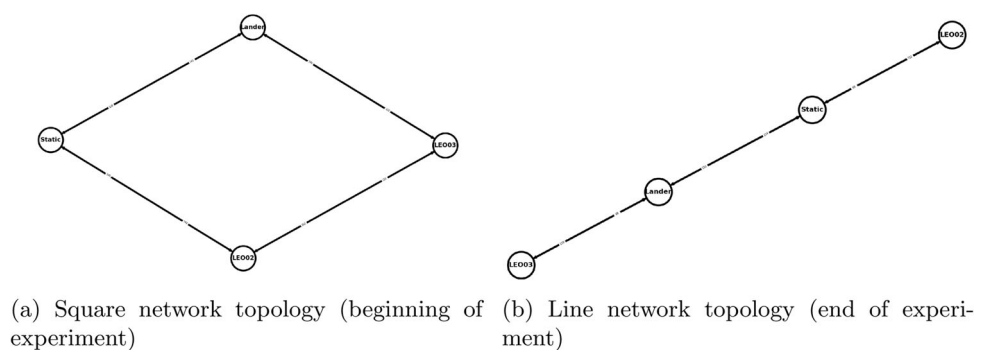
(a) Square network topology (beginning of experiment)

(b) Line network topology (end of experiment)

Table 2 Network Metrics over the mission without ROS 2

Scenario	t=0		t=120	
	lander to leo02	leo02 to lander	lander to leo02	leo02 to lander
Data throughput (TCP)	270 Kbits/s	164 kbits/s	1.31 Mbits/s	1.15 Mbits/s
Jitter (UDP)	0.000ms	245.385ms	0.000 ms	7.850ms

it should only get concerned when A and B are on the edge of the direct communication span. It is also noticeable in Table 2 that the jitter is highly impacted by the distance between each node. Also, using the lander as an emitter provides better results than the robot. Those results can be explained by the lander's antenna being attached to the first floor of a building and therefore offering better coverage.

One hundred five runs were initially expected; however, we encountered some failures during the experiment, leading us to perform 136 runs instead, giving a success rate of 77%. It is noticed that this success rate varies depending on the RMW, 73% for fast, 77% for cyclone and 83% for Zenoh.

The following subsections describe the evaluations of the studied KPIs: reachability, delay, data overhead, CPU usage, and RAM usage over time.

7.1 Reachability

In our scenario, the reachability of Leo02 from the lander is evaluated. Based on the introduced formula for the reachability, Eq. 3, N is the total amount of run per RMW (i.e 35), $\tau_i(t) \in \{0, 1\}$ represents the reachability of Leo02 at time t and $R(t)$ is the average reachability of a RMW at a time t. Figure 8 displays the average reachability $R(t)$ with its variance for each RMW given time.

The data reveals a similar behaviour for Fast and Cyclone, all relying on the DDS technology; however, Zenoh is showing significant improvements in terms of network stability. The Figs. 9, 10, 11 shows the average reachability for three different message groups. Figure 9 corresponds to the smallest messages: one and two kilobytes. Figure 10 highlights the medium messages (4, 8, and 16Kb), while Fig. 11 shows the bigger messages (32 and 64Kb). Both small and medium

messages as shown in Figs. 9 and 10 emphasize Zenoh that outperforms the DDS (Fast and Cyclone) in terms of reachability. However, it is interesting to say that the results for bigger messages shown in Fig. 11 mitigate the performance of Zenoh with comparable results with Cyclone. On the other hand, Fast is underperforming compared to the other RMW.

7.2 Delay

In this section, the delay between each message received is studied. For reference, a message is sent by Leo02 every 0.5 seconds on the network but might be lost or corrupted during the transfer. This is why the delay in receiving the messages is studied; a higher delay means more time to update a potential point cloud or outdated information about the respective rover's residual energy.

Figure 12, shows the average delay in seconds for various message sizes transmitted using the different middleware implementations. Compared to reachability, the variance is not displayed to enhance readability (extremums lead to the flattening of the graph). However, it is notable that FastDDS presents way more variance than Zenoh or Cyclone, reaching values up to a fifty-second delay. Figure 12c shows that Zenoh outperforms the DDSs by an order of scale. It is also interesting to notice that FastDDS performs better than Cyclone for very small messages, which follows FastDDS's claims. On the other hand, Zenoh has comparable results to Cyclone for large messages following the reachability tendency. It is also important to notice that Cyclone and Fast seem to be impacted by a less stable network while Zenoh remains stable all along. Around the end of the experiment, when the data throughput exceeds 1Mb/s (Table 2), both DDSs seem more stable.
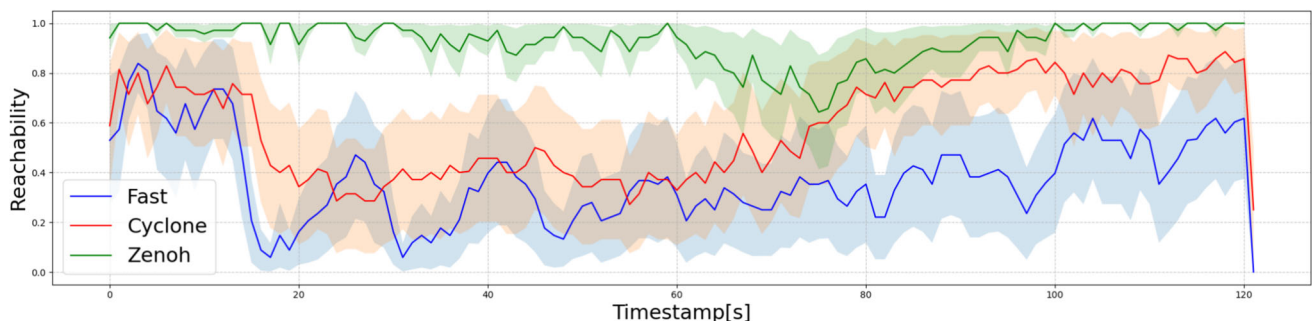
Fig. 8 Average reachability over all the runs for each RMW with the variance
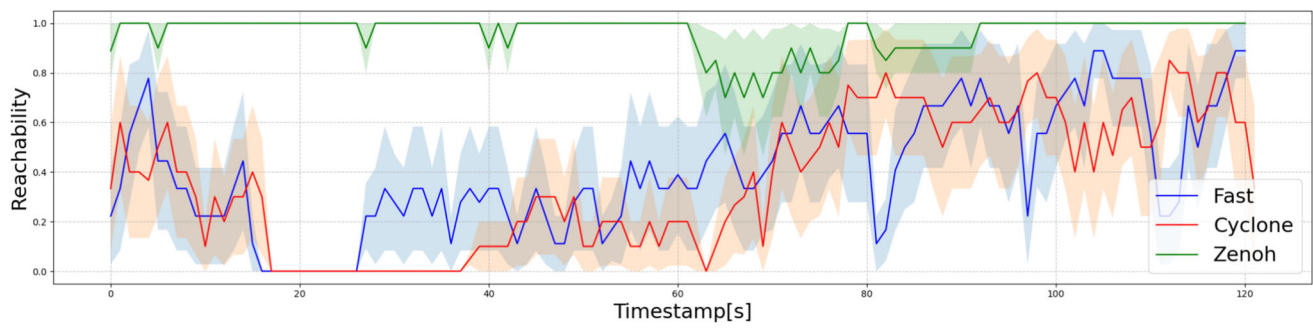
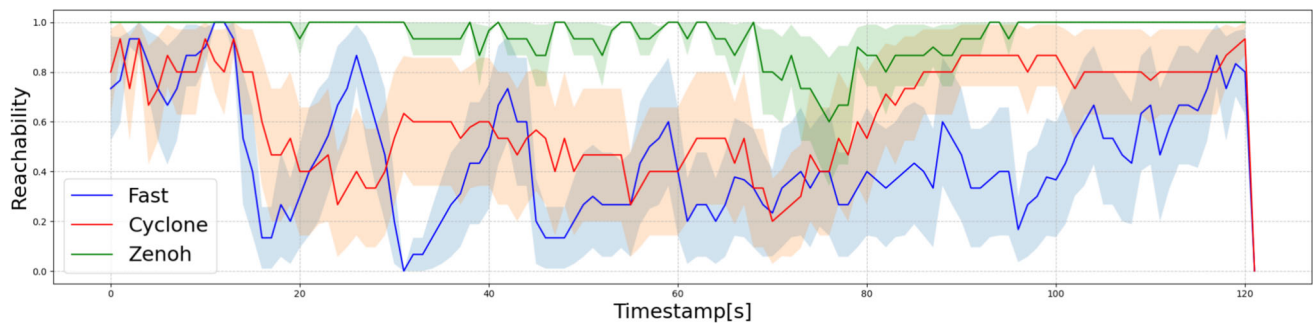**Fig. 9** Average reachability for messages of sizes one and two kilobytes

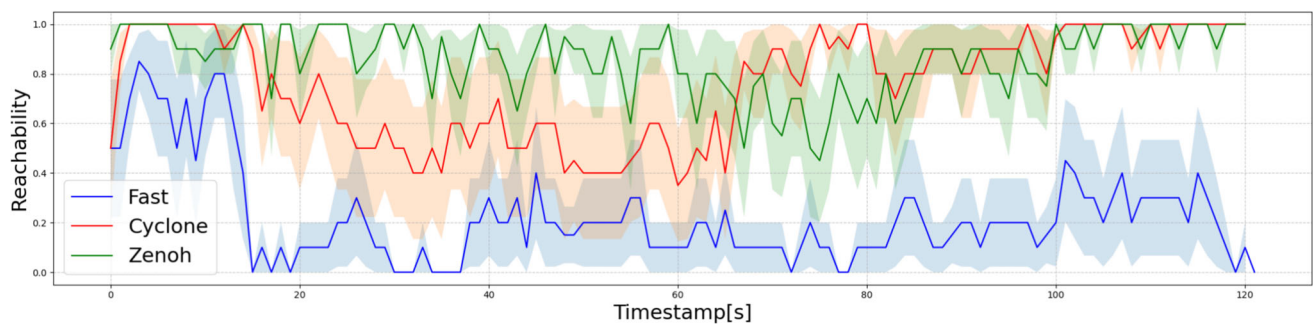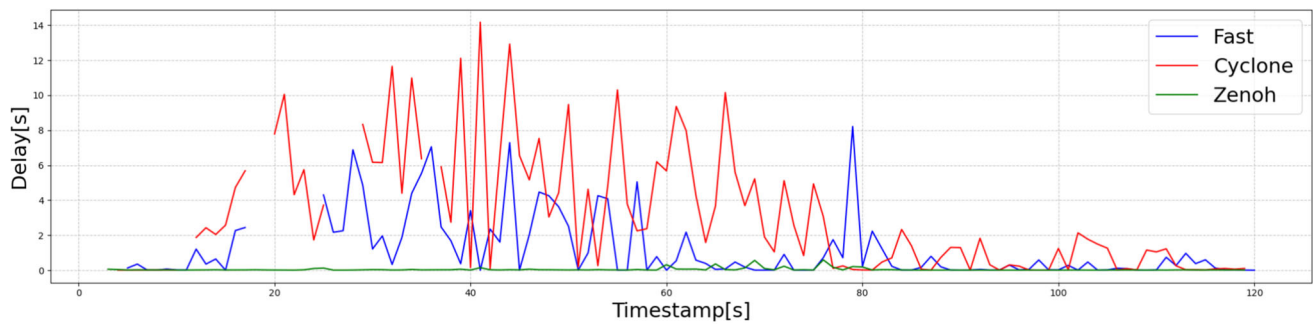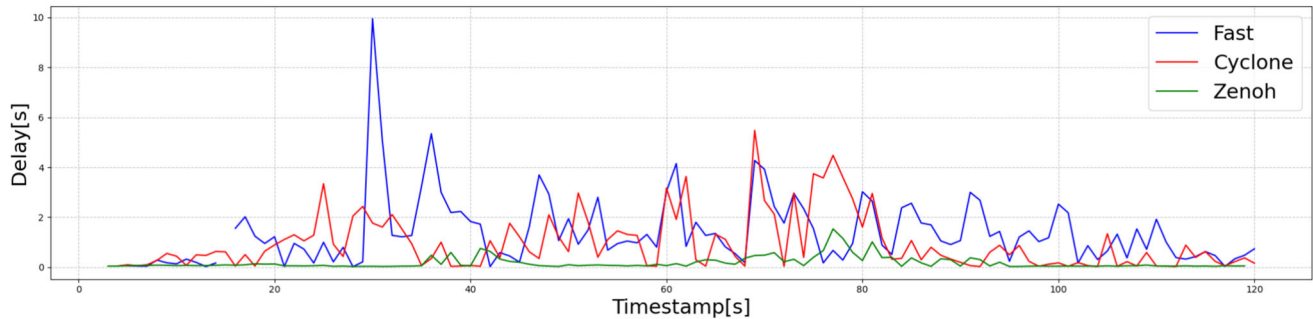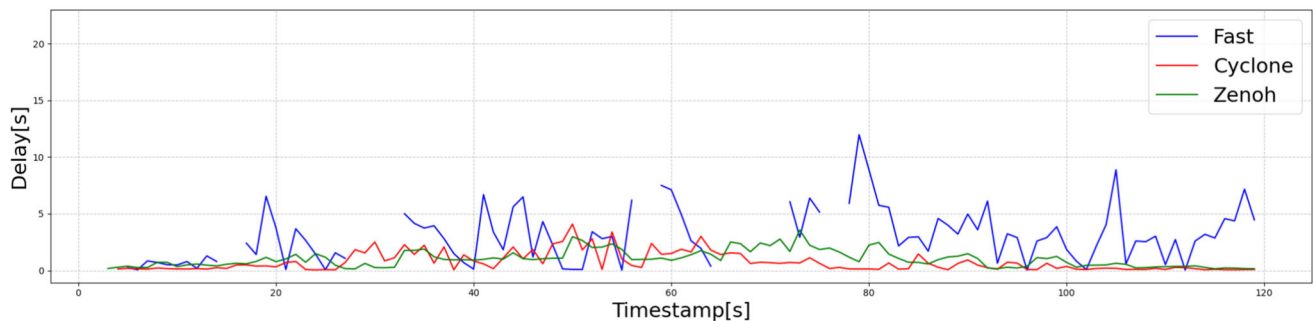**Fig. 10** Average reachability for messages of sizes four, eight, and sixteen kilobytes

Fig. 11 Average reachability for messages of size 32 and 64 kilobytes for each RMW with the variance

(a) Average delay for messages of sizes one and two kilobytes

(b) Average delay for messages of sizes four, eight, sixteen kilobytes

(c) Average delay for messages of sizes thirty-two and sixty-four kilobytes

Fig. 12 Average delay in second over three packs of message size for each RMW without the variance

In the context of extra-terrestrial extreme environments, the delay of communication is a less crucial parameter since any delay induced by the network would be negligible compared to the delay between the operator and the robot. For example, in the case of lunar exploration, we can expect a delay between the Earth and the moon of 2 seconds. Such delay already removes the robot's real-time teleoperation capabilities.

7.3 Data Overhead

The data overhead is characterised by two metrics: the bytes sent by Leo02 and the bytes received by the lander. Since for all the RMWs, the messages are of the same size, measuring the total bytes sent gives an overview of the message overhead of each RMW. Figure 13 shows the average bytes sent by leo02 over the different RMW. As shown in the graph, this value does not fluctuate a lot except for Fast during a run. These peaks may be one of the reasons for the largest network instability led by Fast. It seems that Zenoh has an overall greater data overhead, this represents an issue in the data measuring. The function used to measure the data overhead took into account not only the traffic on the Wifi interface but also the Zenoh router, multiplying by two the real overhead. In that case, it appears that Zenoh shows a way smaller data overhead than the two other RMW.
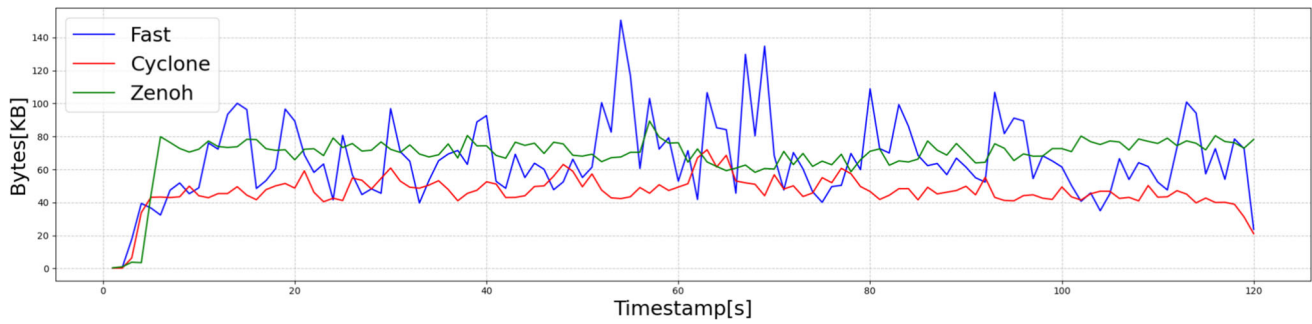
Fig. 13 Average bytes sent by leo02 over all the runs for each RMW without the variance

Since the average bytes sent by each RMW do not fluctuate much over time, Fig. 14 offers a Box Plot graph featuring the bytes sent by Leo02 for each message size and RMW Fig. 14a along with a Box Plot featuring the bytes received by the lander for each message size and RMW Fig. 14b. These graphs also take into account the latest remarks, dividing the bytes sent and received for Zenoh.

Figure 14 provides a better understanding of the impact of each RMW over different message sizes. The results of the lander and leo02 are comparable as expected, except for a slight increase of bytes on the lander side (scale differs on both graphs; Fig. 14b has a larger scale).

Based on those results, Zenoh appears to add less overhead but also tends to be more predictable for all message sizes, whereas Fast and Cyclone tend to lead to more inconsistency and, therefore, more network instabilities.

7.4 CPU and RAM usage

In this section, we discuss the CPU and RAM usage of ROS 2 processes. It is important to emphasize the running processes on the rover. We only consider the packages designed for the experiment to be running (sender for leo02 and receiver for lander), excluding even the ROS daemon. As an embedded system has a limited amount of resources, this metric allows us to quantify the impact of each RMW on the system. An RMW taking more resources leaves less room for other algorithms that might be running on the system, such as SLAM, navigation, or exploration. Furthermore, taking more resources has a direct impact on the power consumption of the embedded device and, therefore, the battery life of the robot.

Figure 15a illustrates the CPU usage on the Raspberry of leo02 when sending messages of fixed sizes for both
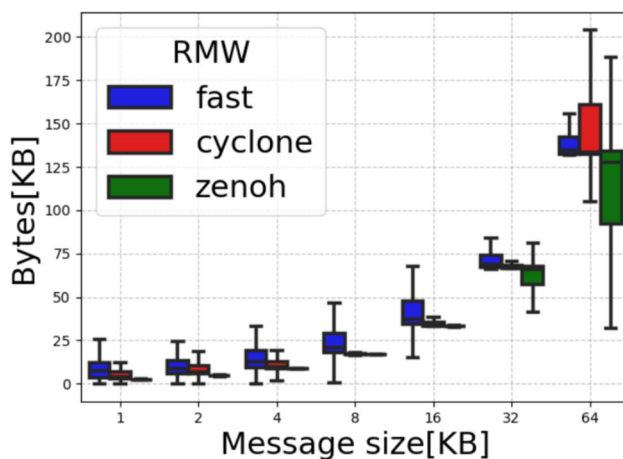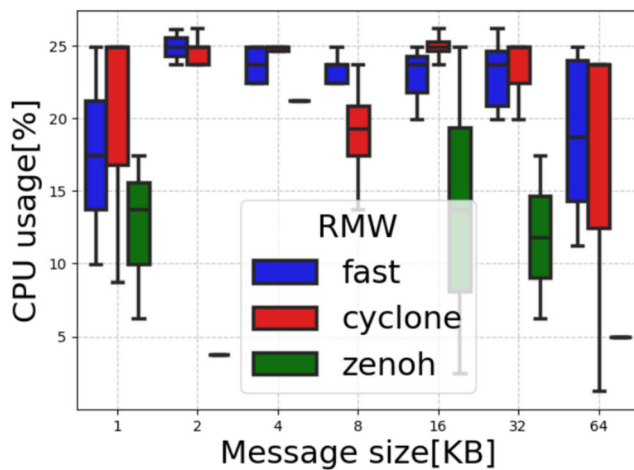

(a) Data overhead of leo02's ROS 2 related processes for every fixed size message
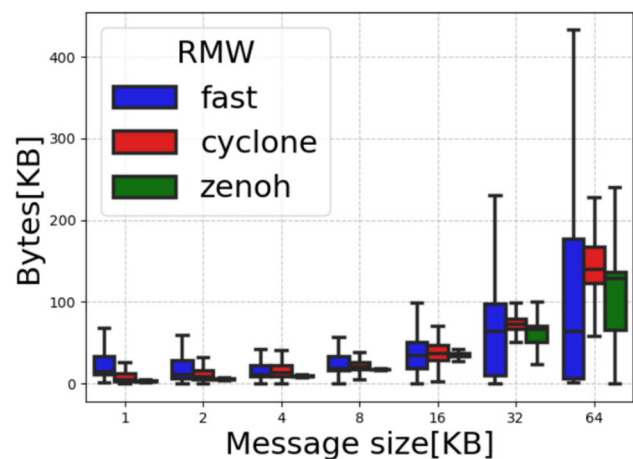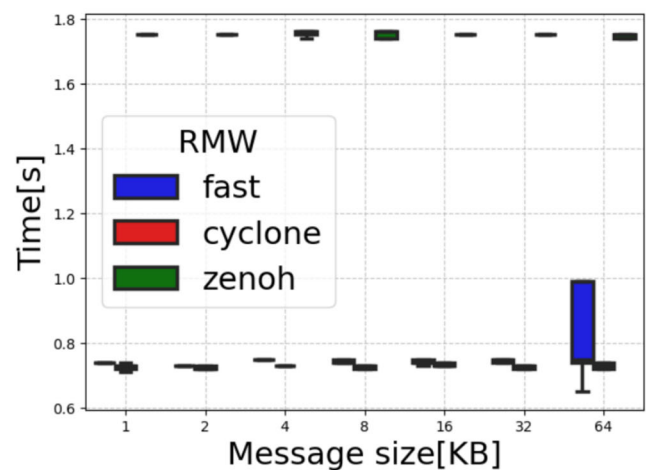
(b) Data overhead of the lander ROS 2 related processes for every fixed size message

Fig. 14 Box Plot representing the data overhead, for each RMW at every fixed size

(a) CPU usage in percentage on Leo02


(b) CPU time in second on Leo02

Fig. 15 Box Plot representing the CPU usage in percentage

scenarios, while Fig. 15b displays the CPU time. Comparing both metrics shows interesting results, while Zenoh uses in median of around two times less the CPU, it seems to use two times longer. This shows a behaviour that tends to be more lightweight for the embedded computers while being slightly slower to compute

Notably, Zenoh uses a router that adds overhead to the overall resource consumption. However, even with this overhead, Zenoh depicts very good performances compared to the other RMW. While CPU consumption has a direct impact on the battery life and other algorithms' performance, RAM (Random access memory) is required to save data while computing. To this extent, Fig. 16a displays the RAM percentage

usage of ROS2 processes (i.e. RMW consumption) while Fig. 16b the bytes consumption of such process.

Both diagrams show the same data but use different representations (one as a percentage, the other one in bytes) to get the RAM consumption's magnitude fully. Those diagram have very narrow box plots but depict a constant behaviour for each RMW. Starting from the left: Fast, Cyclone and Zenoh show the same amount of used memory no matter the size of the transmitted data which is really surprising for us. Cyclone and Fast use the least amount of memory with respectively 29.3MB and 34MB representing less than 0.45% of the 4GB of RAM of the Raspberry. On Zenoh's side, this number grows up to 54.6MB representing 0.66%. Zenoh takes more
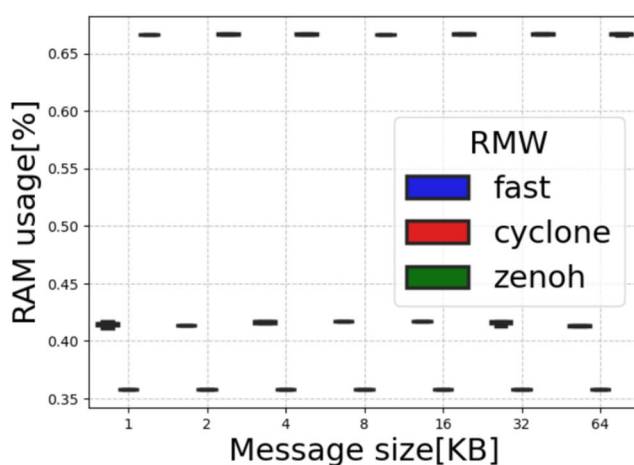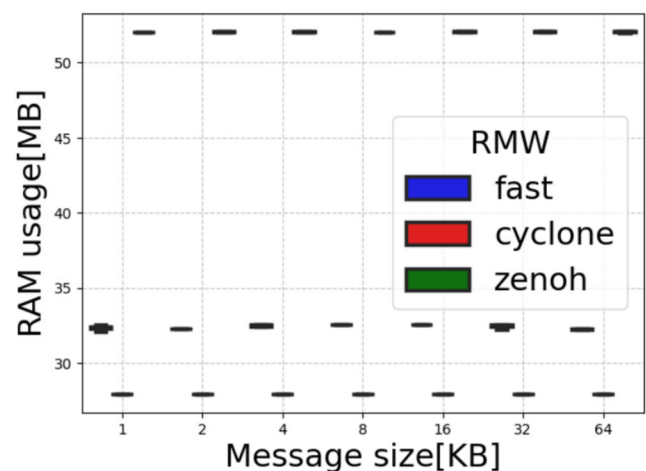

(a) RAM usage in percentage on Leo02

(b) RAM usage in 10^7 bytes on Leo02

Fig. 16 Box Plot representing the RAM usage in percentage during the scenario

Table 3 Summary table of RMW performances compared to each message size

		Fast	Cyclone	Zenoh
Small Messages	Delay	−	−−	++
	Reachability	−	−	++
	Data overhead	−	+/−	+
	CPU Usage	+/−	+/−	++
	RAM Usage	+/−	+	−
Medium Messages	Delay	−−	−	++
	Reachability	−	+/−	++
	Data overhead	−	+/−	+
	CPU Usage	+/−	+/−	++
	RAM Usage	+/−	+	−
Larger Messages	Delay	−−	+	+
	Reachability	−	+	+
	Data overhead	+	+/−	+
	CPU Usage	+/−	+/−	++
	RAM Usage	+/−	+	−

Table 4 Comparison table between Zenoh and the DDSs on global performances

	Zenoh to Fast	Zenoh to Cyclone
Delay (reduced)	76%	69.86%
Reachability (increased)	146.93%	58.17%
Data overhead per message	−47.82%	−25.93%
CPU Usage (reduced)	41.27%	39.76%
RAM Usage (increased)	60.50%	86.03%

RAM than the other RMW, however it relies on a router to be functional. This router might add overhead to the initial 'cost' of sending a message on the network and therefore increase RAM consumption as it might have increased CPU consumption.

7.5 Summary

Table 3 displays an overview of the performances of each middleware over different performance metrics. Symbols display how efficient each RMW is on a specific point of interest compared to the others. A "++" indicates that the system outperforms the others in the scenario, while a "- -" indicates major performance issues compared to the other approaches. The intermediary symbols show: "+" good results, "+/-" acceptable results and "-" less good results than the others. Finally, the "?" indicates missing values. As depicted earlier, Zenoh shows outperforming results compared to the other RMW.

For small messages, Zenoh outmatches the other RMW with smaller delays, better reachability, not too high data overhead per message, and reduced CPU usage while using slightly more RAM. However, the RAM usage is still within the acceptable margin. Medium messages still depict a good picture in favour of Zenoh with mitigated RAM usage. Large messages, also confirm the interest in using Zenoh. We recommend selecting the RMW based on the conducted experiment: if battery management and the reachability of the network are the key parameters, Zenoh might be more suited. On the other hand, if the data throughput is more important, Cyclone might be more interesting.

Table 4 emphasizes the difference between Zenoh and the other RMW. Out of five metrics, Zenoh surpasses the other RMW in terms of delay, reachability, data overhead and CPU usage. On the other hand, RAM usage has increased, but the order of magnitude is derisory compared to embedded computer resources nowadays. On average, Fast offers comparable results, whereas Cyclone adds around 50% less data overhead to transmit a ROS 2 message over the network.

A fixed communication delay cannot be avoided in the context of extra-terrestrial extreme environment exploration, forbidding any real-time operations and control. Any network-induced delay is negligible and less interesting for this scenario. However, data overhead, reachability, and CPU usage are of high interest. Since the mesh network can offer limited performance (depending on the used protocol, hardware, and environment), it is designed to kick out any node that uses too much data throughput, so the selected solution should focus on optimizing the data throughput usage and, thereby, the data overhead. Exploration rovers must execute various resource-expensive mapping and navigation software. Saving computational power should also be a concern in space-oriented scenarios. Also, computational power directly translates into electrical power, which is a critical resource in space. On the other hand, all of those metrics are pointless if the nodes in the network are unreachable. This is why we choose reachability as our key parameter regarding which RMW to choose.

In [14], the authors present similar results, where Zenoh stands out in the DDS implementation. They also highlight Message Queuing Telemetry Transport (MQTT) as the best solution, even if there is no known implementation compatible with ROS2. It could be worth investigating an implementation with ROS. Yet, MQTT is designed for IOT and handling small messages and would probably have good performance in specific scenarios.

8 Conclusion and Future Work

This study evaluates the performances of available ROS 2 middleware (FastRTPS, Cyclone DDS and Zenoh) over a mesh network given a dynamic topology. The final choice of

ROS 2 middleware will be given by the closest middleware able to fulfil the scenario: an exploration of the extreme extra-terrestrial environment using a Multi-Robot Systems (MRS).

The results emphasize Zenoh as a promising solution for using ROS 2 on a mesh network in an extra-terrestrial extreme environment exploration scenario for MRS. In this scenario, where robots have limited computing capabilities and energy while needing to maintain consistent connectivity, our focus is on power consumption and reachability, which are two key strengths of Zenoh.

However, Zenoh might not be suited for scenarios that require very low RAM usage. Furthermore, Zenoh's very good results tend to be more mitigated with larger messages. Further studies on larger messages (up to one Megabyte or even larger) need to be conducted to fully picture Zenoh's performance on very large messages.

In space missions where a robot needs to reach a given position, we need to ensure the connection is maintained. Future work will focus on developing collaboration mechanisms for Connectivity maintenance in mesh networks. In order to achieve optimal networking capabilities for space multi-robot systems, future work will also focus on optimizing network resources.

Author Contributions Loïck Pierre Chovet: Conceptualization, Methodology, Software, Data Analysis, Writing. Gabriel Manuel Garcia: Methodology, Software, Data Acquisition and formatting, Writing. Abhishek Bera: Methodology, Review and Editing. Antoine Richard: Methodology. Kazuya Yoshida: Review and Editing. Miguel Angel Olivares-Mendez: Review and Editing. The authors Loïck Chovet and Gabriel Garcia contributed equally to this work.

Data Availability The data are available on https://github.com/Gabryss/RMW-Mesh-Experiments, under the folder data_analysis/data

Declarations

Ethics Approval The authors declare that this study does not pertain to a subject that requires approval by an Ethics committee.

Consent to Participate The authors declare that this is not applicable to this work.

Consent for Publication The authors declare that this is not applicable to this work.

Conflicts of Interest The authors declare no conflicts of interest.

Competing interests The authors declare no competing interests.

References

1. Pretto, A., Aravecchia, S., Burgard, W., Chebrolu, N., Dornhege, C., Falck, T., Fleckenstein, F., Fontenla, A., Imperoli, M., Khanna, R., Liebisch, F., Lottes, P., Milioto, A., Nardi, D., Nardi, S., Pfeifer, J., Popović, M., Potena, C., Pradalier, C., Rothacker-Feder, E., Sa, I., Schaefer, A., Siegwart, R., Stachniss, C., Walter, A., Winterhalter, W., Wu, X., Nieto, J.: Building an Aerial-Ground Robotics System for Precision Farming: An Adaptable Solution. IEEE Robotics & Automation Magazine **28**(3), 29–49 (2021)

2. Yan, Z., Jouandeau, N., Cherif, A.A.: A Survey and Analysis of Multi-Robot Coordination. International Journal of Advanced Robotic Systems **10**(12), 399 (2013). https://doi.org/10.5772/57313. Publisher: SAGE Publications. Accessed 2023-03-15

3. Roldán Gómez, J., Peña-Tapia, E., Garzón Ramos, D., De León Rivas, J., Garzon, M., Cerro, J., Barrientos, A.: Multi-robot Systems, Virtual Reality and ROS: Developing a New Generation of Operator Interfaces. In: Studies in Computational Intelligence, pp. 29–64 (2019). https://doi.org/10.1007/978-3-319-91590-6_2 . Journal Abbreviation: Studies in Computational Intelligence

4. Espina, M.V., Grech, R., De Jager, D., Remagnino, P., Iocchi, L., Marchetti, L., Nardi, D., Monekosso, D., Nicolescu, M., King, C.: Multi-robot Teams for Environmental Monitoring. In: Remagnino, P., Monekosso, D.N., Jain, L.C. (eds.) Innovations in Defence Support Systems – 3: Intelligent Paradigms In Security. Studies in Computational Intelligence, pp. 183–209. Springer, Berlin, Heidelberg (2011). https://doi.org/10.1007/978-3-642-18278-5_8 Accessed 2023-11-10

5. Baima, R.L., Chovet, L., Sedlmeir, J., Fridgen, G., Olivares-Mendez, M.A.: Designing Trustful Cooperation Ecosystems is Key to the New Space Exploration E.https://doi.org/10.1145/3639476.3639760, arXiv:2402.06036 Accessed 2024-03-12

6. Parker, L.E.: Multiple Mobile Robot Systems. In: Siciliano, B., Khatib, O. (eds.) Springer Handbook of Robotics, pp. 921–941. Springer, Berlin, Heidelberg (2008). https://doi.org/10.1007/978-3-540-30301-5_41 Accessed 2023-03-15

7. Overview: In-Situ Resource Utilization - NASA (2024). https://www.nasa.gov/overview-in-situ-resource-utilization/ Accessed 2023-11-10

8. Chovet, L., Lima Baima, R., Hartwich, E., Bera, A., Sedlmeir, J., Fridgen, G., Olivares Mendez, M.A.: Trustful coopetitive infrastructures for the new space exploration era. In: PROCEEDINGS

OF THE 2023 ACM SYMPOSIUM ON APPLIED COMPUTING. ACM (2024). https://doi.org/10.1145/3605098.3635887 , https://orbilu.uni.lu/handle/10993/60481 Accessed 2024-03-20

9. Staudinger, E., Giubilato, R., Schuster, M.J., Pöhlmann, R., Zhang, S., Dömel, A., Wedler, A., Dammann, A.: Terrain-aware communication coverage prediction for cooperative networked robots in unstructured environments **202**, 799–8 (2024). https://doi.org/10.1016/j.actaastro.2022.10.050 . Accessed 2023-11-27

10. Macenski, S., Foote, T., Gerkey, B., Lalancette, C., Woodall, W.: Robot Operating System 2: Design, architecture, and uses in the wild. Science Robotics **7**(66), 6074 (2022). https://doi.org/10.1126/scirobotics.abm6074. Publisher: American Association for the Advancement of Science. Accessed 2023-05-25

11. Köksal, Tekinerdogan, B.: Obstacles in data distribution service middleware: A systematic review. Future Generation Computer Systems 68, 191–210 (2017 https://doi.org/10.1016/j.future.2016.09.020

12. DiLuoffo, V., Michalson, W.R., Sunar, B.: Robot Operating System 2: The need for a holistic security approach to robotic architectures. International Journal of Advanced Robotic Systems **15**(3), 1729881418770011 (2018). https://doi.org/10.1177/1729881418770011. Publisher: SAGE Publications. Accessed 2023-07-12

13. Corsaro, A., Cominardi, L., Hecart, O., Baldoni, G., Avital, J.E.P., Loudet, J., Guimares, C., Ilyin, M., Bannov, D.: Zenoh: Unifying communication, storage and computation from the cloud to the microcontroller. In: 2023 26th Euromicro Conference on Digital System Design (DSD), pp. 422–42 (2024). https://doi.org/10.1109/DSD60849.2023.00065 . ISSN: 2771-2508. https://ieeexplore.ieee.org/abstract/document/10456820 Accessed 2024-04-12

14. Liang, W.-Y., Yuan, Y., Lin, H.-J.: A Performance Study on the Throughput and Latency of Zenoh, MQTT, Kafka, and DDS (2023). arXiv. arXiv:2303.09419 , https://doi.org/10.48550/arXiv.2303.09419 . http://arxiv.org/abs/2303.09419 Accessed 2023-07-12

15. Sichitiu, M.: Wireless mesh networks: opportunities and challenges (2023)

16. Chen, Y., Du, Z., García-Acosta, M.: Robot as a Service in Cloud Computing. In: 2010 Fifth IEEE International Symposium on Service Oriented System Engineering, pp. 151–158 (2010). https://doi.org/10.1109/SOSE.2010.44

17. Brandenburger, A.M., Nalebuff, B.J.: Co-Opetition. Crown (2011)

18. Rizk, Y., Awad, M., Tunstel, E.W.: Cooperative Heterogeneous Multi-Robot Systems: A Survey. ACM Computing Surveys **52**(2), 1–31 (2020). https://doi.org/10.1145/3303848. Accessed 2023-06-14

19. Asama, H., Matsumoto, A., Ishida, Y.: Design Of An Autonomous And Distributed Robot System: Actress. In: Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems '. (IROS '89) 'The Autonomous Mobile Robots and Its Applications, pp. 283–290 (1989). https://doi.org/10.1109/IROS.1989.637920

20. Hewitt, C., Bishop, P., Steiger, R.: A universal modular ACTOR formalism for artificial intelligence. In: Proceedings of the 3rd International Joint Conference on Artificial Intelligence. IJCAI'73, pp. 235–245. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1973)

21. Parker, L.E.: ALLIANCE: an architecture for fault tolerant multi-robot cooperation. IEEE Transactions on Robotics and Automation 14(2), 220–240 (1998). https://doi.org/10.1109/70.681242 . Conference Name: IEEE Transactions on Robotics and Automation

22. Cordes, F., Ahrns, I., Bartsch, S., Birnschein, T., Dettmann, A., Estable, S., Haase, S., Hilljegerdes, J., Koebel, D., Planthaber, S., Roehr, T.M., Scheper, M., Kirchner, F.: LUNARES: lunar crater exploration with heterogeneous multi robot systems. Intelli-

23. Schuster, M.J., Müller, M.G., Brunner, S.G., Lehner, H., Lehner, P., Sakagami, R., Dömel, A., Meyer, L., Vodermayer, B., Giubilato, R., Vayugundla, M., Reill, J., Steidle, F., Bargen, I., Bussmann, K., Belder, R., Lutz, P., Stürzl, W., Smíšek, M., Maier, M., Stoneman, S., Prince, A.F., Rebele, B., Durner, M., Staudinger, E., Zhang, S., Pöhlmann, R., Bischoff, E., Braun, C., Schröder, S., Dietz, E., Frohmann, S., Börner, A., Hübers, H.-W., Foing, B., Triebel, R., Albu-Schäffer, A.O., Wedler, A.: The ARCHES space-analogue demonstration mission: Towards heterogeneous teams of autonomous robots for collaborative scientific sampling in planetary exploration **5**(4), 5315–53 (2024). https://doi.org/10.1109/LRA.2020.3007468 . Conference Name: IEEE Robotics and Automation Letters. Accessed 2024-03-21

24. ESA - European Space Agency: 1ST FIELD TEST - ESA-ESRIC Challenge. https://www.spaceresourceschallenge.esa.int/general-3. Accessed on 2023-06-28 (2023)

25. Van Der Meer, D., Chovet, L., Garcia, G., Bera, A., Olivares Mendez, M.A.: REALMS 2 -RESILIENT EXPLORATION AND LUNAR MAPPING SYSTEM 2 (2024). https://orbilu.uni.lu/handle/10993/57411 Accessed 2023-11-17

26. About the Data Distribution Service Specification Version 1.4 (2024). https://www.omg.org/spec/DDS/1.4/About-DDS/ Accessed 2024-03-29

27. Rabbah, M.A., Rabbah, N., Belhadaoui, H., Rifi, M.: Real Time Distributed and Decentralized Peer-to-Peer Protocol for Swarm Robots. International Journal of Advanced Computer Science and Applications (IJACSA) **12**(11) (2021). https://doi.org/10.14569/IJACSA.2021.0121131 . Number: 11 Publisher: The Science and Information (SAI) Organization Limited. Accessed 2023-06-26

28. K.C, K.: Wireless mesh network: A survey. In: 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), pp. 1966–1970 (2016). https://doi.org/10.1109/WiSPNET.2016.7566486 . https://ieeexplore.ieee.org/abstract/document/7566486 Accessed 2023-10-25

29. Benyamina, D., Hafid, A., Gendreau, M.: Wireless Mesh Networks Design — A Survey. IEEE Communications Surveys & Tutorials **14**(2), 299–310 (2012). https://doi.org/10.1109/SURV.2011.042711.00007. Conference Name: IEEE Communications Surveys & Tutorials

30. Chakeres, I.D., Belding-Royer, E.M.: AODV routing protocol implementation design. In: 24th International Conference on Distributed Computing Systems Workshops, 2004. Proceedings., pp. 698–703 (2004). https://doi.org/10.1109/ICDCSW.2004.1284108 , https://ieeexplore.ieee.org/abstract/document/1284108 Accessed 2023-11-10

31. Jafri, S.T.A., Ahmed, I., Ali, S., Yahaya, J., Qamar, F., Abdullah, Z.H.: Split Hop Penalty for Transmission Quality Metrics in a Better Approach to Mobile Ad Hoc Networking (BATMAN) for IoT-Based MANET. Symmetry **15**(5), 969 (2023). https://doi.org/10.3390/sym15050969. Accessed 2023-07-06

32. IEEE 802.11s Mesh Networking, D1.06. Jul. 2007. (2007)

33. Yang, L., Chung, S.-H.: HWMP+: An Improved Traffic Load Sheme for Wireless Mesh Networks. In: 2012 IEEE 14th International Conference on High Performance Computing and Communication & 2012 IEEE 9th International Conference on Embedded Software and Systems, pp. 722–727 (2012). https://doi.org/10.1109/HPCC.2012.102

34. Tranzatto, M., Miki, T., Dharmadhikari, M., Bernreiter, L., Kulkarni, M., Mascarich, F., Andersson, O., Khattak, S., Hutter, M., Siegwart, R., Alexis, K.: CERBERUS in the DARPA subterranean challenge **7**(66), 97 (2024). https://doi.org/10.1126/scirobotics.abp9742 . Publisher: American Association for the Advancement of Science. Accessed 2023-11-15

35. Maruyama, Y., Kato, S., Azumi, T.: Exploring the performance of ROS2. In: Proceedings of the 13th International Conference on Embedded Software. EMSOFT '16, pp. 1–10. Association for Computing Machinery, New York, NY, USA (2016). https://doi.org/10.1145/2968478.2968502 Accessed 2023-05-25

36. eProsima Fast DDS Performance (2024). https://www.eprosima.com/index.php/resources-all/performance/eprosima-fast-dds-performance Accessed 2023-07-03

37. Fast DDS vs Cyclone DDS Performance (2024). https://www.eprosima.com/index.php/resources-all/performance/fast-dds-vs-cyclone-dds-performance Accessed 2023-07-03

38. Minimizing Discovery Overhead in ROS2 (2024). https://zenoh.io/blog/2021-03-23-discovery/ Accessed 2023-07-04

39. ZettaScale Technology: Improving the communications layer of robot applications with ROS 2 and Zenoh (2022). https://www.youtube.com/watch?v=1NE8cU72frk Accessed 2023-07-04

40. Chen, Y., Luo, Y., Ma, J., Qi, A., Huang, R., De Paulis, F., Qi, Y.: Non-Contact In-Vehicle Occupant Monitoring System Based on Point Clouds from FMCW Radar. Technologies **11**(2), 39 (2023). https://doi.org/10.3390/technologies11020039. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute. Accessed 2023-07-03

41. Fotheringham, M., Paudyal, D.R.: Combining terrestrial scanned datasets with uav point clouds for mining operations. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences V-4-2021, 129–138 (2021). https://doi.org/10.5194/isprs-annals-V-4-2021-129-2021

42. Multi-Robot Systems: Challenges, Trends and Application (2024). https://doi.org/10.3390/books978-3-0365-2847-2 . https://www.mdpi.com/books/reprint/5252-multi-robot-systems-challenges-trends-and-applications Accessed 2024-03-29

43. Distributed Robotic Systems in the Edge-Cloud Continuum with ROS 2: a Review on Novel Architectures and Technology Readiness (2024). https://ar5iv.labs.arxiv.org/html/2211.00985 Accessed 2024-03-29

44. MikroTik (2024). https://mikrotik.com/ Accessed 2023-11-10

45. ER-X-SFP Quick Start Guide (2024). https://dl.ubnt.com/qsg/ER-X-SFP/ER-X-SFP_EN.html Accessed 2023-11-10

46. Leo Rover | Robot Developer Kit | Open-source :ROS and for outdoor use (2024). https://www.leorover.tech/ Accessed 2023-07-04

47. Introducing the Intel® RealSense™ Depth Camera D455 (2024). https://www.intelrealsense.com/depth-camera-d455/ Accessed 2023-11-10

48. Peng, M.: RPlidar A2 - SLAMTEC Global Network Lidar Sensor RPlidar A2 cheap lidar scanner (2024). https://www.slamtec.ai/home/rplidar_a2/ Accessed 2023-11-10

49. NVIDIA Jetson Xavier Series (2024). https://www.nvidia.com/en-us/autonomous-machines/embedded-systems/jetson-xavier-series/ Accessed 2023-11-10

50. HWMPplus (Mesh) - RouterOS - MikroTik Documentation (2024). https://help.mikrotik.com/docs/pages/viewpage.action?pageId=8978441 Accessed 2024-04-12

Loïck Pierre Chovet received the Engineering degree in computer science and the master in robotics from the universityof Clermont-Ferrand in 2021 (a.k.a INP-ISIMA). He is currently pursuing a Ph.D. thesis at the Interdisciplinary Centerfor Security, Reliability and Trust (SnT), University of Luxembourg, in the Space Robotics Research Group (SpaceR),where he works on multi-robot collaboration, coopeitive scenarios and connectivity maintenance in mesh networks.

Gabriel Manuel Garcia received the Engineering degree from the University of Technology of Belfort-Montbéliard (UTBM).He is currently pursuing a Ph.D. thesis at the Interdisciplinary Center for Security, Reliability and Trust (SnT),University of Luxembourg, in the Space Robotics Research Group (SpaceR), where he works on underground-traversability analysis and exploration.

Abhishek Bera received the BTech degree from the West Bengal University of Technology, in 2009, the ME degreefrom Bengal Engineering and Science University Shibpur, in 2011, and the PhD degree from the AdvancedTechnology Development Centre, indian Institute of Technology Kharagpur, India. He is currently a postdoctoralresearcher with the University of Luxembourg. His current research interests include multi-UAV networks, Internet ofThings, aerial robotics, 5G, and mobile edge computing. His research works are published in different reputedjournals and many reputed conferences. He is a reviewer for several international journals and conferences.

Antoine Richard received the Ph.D. degree from Georgia Institute of Technology (a.k.a. Georgia Tech), in 2022.During his thesis, he studied the modeling and control of dynamic systems through the use of neural networks andreinforcement learning. He is currently a Research Associate with the Space Robotics Research Group (SpaceR),University of Luxembourg, where he works on environment generation and reinforcement learning for space robotics.

Kazuya Yoshida received B.E., M.S., and Dr.Eng, degrees in Mechanical Engineering Science from Tokyo Instituteof Technology, Japan, in 1984, 1986, and 1990, respectively. Since 2003, he has been Full Professor in Department ofAerospace Engineering at Tohoku University, Japan, leading the Space Robotics Laboratory. He has also taught atInternational Space University. His research interests include free-flying space robots and lunar/planetary explorationrobots.

Miguel Angel Olivares-Mendez received the Engineering degree in computer science from the University of Malaga,in 2006, and the M.Sc. and Ph.D. degrees in robotics and automation from the Technical University of Madrid, in 2009and 2013, respectively. During his Ph.D., he was a Visiting Researcher with EPFL, Switzerland, and ARCAA-QUT,Australia. He is currently a tenured Assistant Professor of space robotics and a Senior Research Scientist with SnTand the University of Luxembourg. He leads the Space Robotics Research Group (SpaceR), where he is the MainSupervisor for 11 Ph.D. students and seven postdoctoral researchers. He has published more than 120 peer-reviewedpublications. His research interests include aerial, planetary, and orbital robotics for autonomous navigation, situational awareness, perception, machine learning, multi-robot interaction in autonomous exploration, inspection,and operations.