2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Enabling Large-scale Heterogeneous Collaboration with Opportunistic Communications

Fernando Cladera, Zachary Ravichandran, Ian D. Miller, M. Ani Hsieh, C. J. Taylor, and Vijay Kumar

***Abstract*— Multi-robot collaboration in large-scale environments with limited-sized teams and without external infrastructure is challenging, since the software framework required to support complex tasks must be robust to unreliable and intermittent communication links. In this work, we present *MOCHA* (Multi-robot Opportunistic Communication for Heterogeneous Collaboration), a framework for resilient multi-robot collaboration that enables large-scale exploration in the absence of continuous communications. MOCHA is based on a gossip communication protocol that allows robots to interact opportunistically whenever communication links are available, propagating information on a peer-to-peer basis. We demonstrate the performance of MOCHA through real-world experiments with commercial-off-the-shelf (COTS) communication hardware. We further explore the system's scalability in simulation, evaluating the performance of our approach as the number of robots increases and communication ranges vary. Finally, we demonstrate how MOCHA can be tightly integrated with the planning stack of autonomous robots. We show a communication-aware planning algorithm for a high-altitude aerial robot executing a collaborative task while maximizing the amount of information shared with ground robots.**

The source code for MOCHA and the high-altitude UAV planning system is available open source[1].

I. Introduction

Air-ground collaboration has proven to be an effective approach to performing autonomous large-scale exploration of unknown environments [1], [2], [3]. Unmanned ground vehicles (UGVs) provide stable platforms for large sensors, and can safely approach and inspect objects of interest on the ground. However, the sensors' field of view and range can be limited by terrestrial obstacles like trees and the UGV's motion is constrained by traversability limitations and speed considerations.

Unmanned aerial vehicles (UAVs), by contrast, are not confined to two-dimensional motion, can avoid many obstacles by flying at higher altitudes, and can provide vantage points that cover more of the scene. However, UAVs expend significantly more energy on locomotion [4]. Therefore, heterogeneous robot teams can harness the advantages of both air and ground vehicles when exploring large-scale environments to outperform either modality alone.

All authors are with GRASP Laboratory, University of Pennsylvania. Corresponding author: fclad@seas.upenn.edu.

We gratefully acknowledge the support of ARL DCIST CRA W911NF-17-2-0181, NSF Grants CCR-2112665, ONR grant N00014-20-1-2822, ONR grant N00014-20-S-B001, NVIDIA, and C-BRIC, a Semiconductor Research Corporation Joint University Microelectronics Program program cosponsored by DARPA.

[1]http://github.com/KumarRobotics/MOCHA
http://github.com/KumarRobotics/air_router

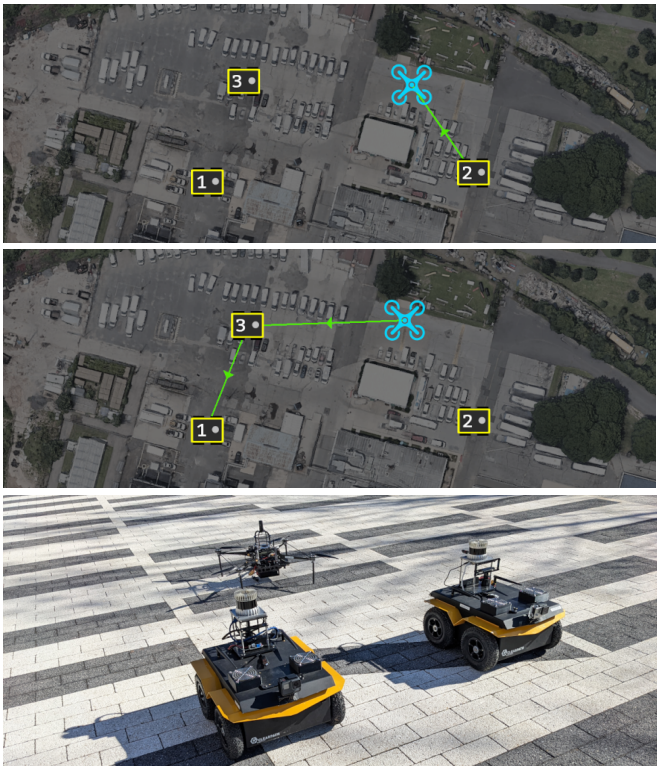

Fig. 1. Top: Visualization of the real-world experiments using MOCHA to enable communications between a UAV and three UGVs. The UAV first communicates with robot 2 and exchanges information. Middle: The UAV acts as a data mule transmitting robot 2's information to robot 3, which itself propagates the information to robot 1. Bottom: Autonomous robots during our heterogeneous exploration experiments: the Falcon 4 UAV and Clearpath Jackals.

In this context of robot collaboration, a reliable communication system is critical for successful mission completion. Communication is necessary for coordinating goals, sharing maps, and informing operators about the progress of a mission [2], [5], [6]. Existing works focus on addressing the challenges of data fusion [7], development of robust consensus algorithms [8], [9], efficient task allocation [10], and probabilistic and graph-theoretic approaches for achieving collaborative behavior [11], [12]. However, most frameworks do not address the specific needs of multi-robot collaborative missions when guarantees on communication capability cannot be provided. Instead, they attempt to maintain connectivity [13] or establish links in a predictable fashion at desired rendezvous points [14], [15], [16]. Nevertheless, when team sizes are limited and the physical scale of the environment is large, it makes sense to allow vehicles to move out of each other's communication range in order to maximize the area explored by the team in a finite amount of time. As such, we

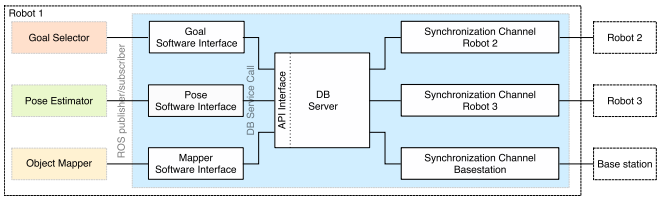

Fig. 2. Communication stack architecture from a single robot point of view. A shared configuration file defines the topics committed to the database and network topology, which generates the required software interfaces and synchronization channels.

would expect individual vehicles to be operating in isolation or forming small communication cliques for most of the task execution time in real-world settings.

In this work we present MOCHA, a communication and coordination framework for collaboration in large-scale environments with a limited number of robots. We assume that the individual communication range of each robot is a small fraction of the environment's total size. MOCHA adopts a gossip communication strategy, where two or more robots exchange information pairwise whenever they are in communication range. Gossip approaches have numerous benefits [17]: they converge to a global consistent state, are simple to implement and analyze, are robust to network disruptions, and are particularly well-suited for low-volume exchanges such as those in robotics applications with relatively small teams. MOCHA provides most of the advantages of consensus algorithms [18] for communication without the computational complexity. Moreover, as the communication between robots is negotiated, MOCHA incurs less channel contention than flooding protocols [19].

We provide a complete system for real-world robot team communication, including a usable software stack to enable further multi-robot research in the field using commercial-off-the-shelf (COTS) communication hardware. **To sum up, our contributions with MOCHA are**:

- An open-source collaboration framework for robots operating with intermittent communications, extensively tested through in multiple multi-robot deployments in real-world [2], [20] and simulations experiments. Existing and future coordination algorithms can be incorporated into our framework for large-scale field deployments of heterogeneous robot teams.
- An open-source planner for a high-altitude UAV that demonstrates the utility of using MOCHA to inform autonomy for actively optimizing communications.

II. Related Work

Communication in field robotics can be separated into two paradigms: 1) ensured network connectivity and 2) opportunistic communication.

A. Ensured Network Connectivity

Ensured network connectivity approaches rely on guaranteeing constant communication between all robots. These approaches require deploying enough agents to cover a particular area, some of them acting as relay nodes, and constructing a mesh network where packets can be routed between the nodes. Ad-hoc networks with the proper routing protocols [21] or COTS solutions, such as mobile ad-hoc mesh networks (MANETs), can generate appropriate mesh networks. For instance, in [14], robots can perform a mission while keeping connectivity through reactive controllers. Other works focus on deploying robots in specific configurations [13] to guarantee continuous communication [22].

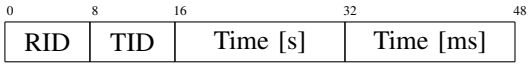

Fig. 3. MOCHA header bit field. RID stores the robot ID, whereas TID stores the topic ID. Our communication system can manage a maximum of 256 nodes, each with up to 256 topics.

Ensured network connectivity systems are advantageous when the nodes require real-time feedback from an operator to provide high-level commands. For instance, some teams at the DARPA SubT challenge deployed MANETs nodes, generated mesh networks to extend communications among all nodes, or used wired backbones to extend communication [5].

B. Opportunistic Communication

Opportunistic strategies do not aim to maintain connectivity, but rather take advantage of it when available. Opportunistic communication offers significant advantages over ensured network connectivity approaches: by relaxing the fully-connected constraint, the number of nodes in the network can be reduced, and mobile agents can act as *data mules* [23], propagating messages far beyond the communication range of the wireless transceivers.

Recently, DDS-based [24] approaches with ROS 2 [25] have been used for communication in multi-robot settings, such as CHORD [26]. However, ROS 2/DDS approaches based on the *reliable* QoS are only suitable for short intermittent disconnections, as networks become congested when a node reconnects after an extended period of time. The authors aimed to improve congestion management when nodes reconnect after a long period of inactivity in ACHORD [27]. They did this by modeling communication channels and using communications-aware coordination based on the size of message queues. Still, there is a limit to how long robots can explore since they need to get within communication range periodically to flush their queues.

Compared to existing solutions, MOCHA is completely opportunistic. It does not require periodic rendezvous, radio deployments, or continuous communications between robots. Our communication approach is robust and predictable and reduces network congestion by cleverly propagating only the latest information available from each node. Similar to [27], we provide metrics to inform higher-level planning decisions in the nodes. Our framework, however, does not impose restrictions on how long the nodes can explore without communication.

III. Methods

A. Distributed Communication System

The architecture of the communication stack for each robot in the team is shown in Fig. 2. The communication stack is composed of the following elements:

- **Database (DB) Server**: a key-value storage that records all the system's messages. Robots can store messages produced locally through software interfaces or get messages from other robots through a synchronization channel. The DB provides a simple API to store and retrieve messages. To reduce network traffic, all payloads are throttled and compressed with LZ4. It is worth noting that data from each robot is stored separately, allowing robots to store-and-forward other agents' data.
- **Software Interfaces**: These interface the rest of the robot middleware with MOCHA. For instance, when using ROS, the software interfaces subscribe to other ROS topics and perform the required DB API calls to insert the data into the DB Server. Conversely, when the DB server receives messages from another robot, the software interfaces publish the messages in the namespaced ROS topics.
- **Synchronization channels**: perform the peer-to-peer synchronization between two nodes. The synchronization channels are based on ZeroMQ [28], following the request/reply model. We have experimentally proven that this approach provides the highest reliability for intermittent communication. The synchronization process is described in detail in Sec. III-A.2.

1) Message Headers: MOCHA relies on a key-value storage model. All messages are stored in an in-memory database on each robot, where the key is the message header, and the value is the binary payload of the message. Headers are computed when a robot inserts a message into its own database.

A header is composed of the following elements, shown in Fig. 3:

- **RID** and **TID**: These fields identify the robot and topic ID of the message, respectively.
- **Time [s]** and **Time [ms]**: store the local robot time that produced the message.

MOCHA relies on a configuration file shared among all nodes that describes the different nodes in the network, the topology, and which topics are published by each robot. RID and TID are computed from the configuration file.

2) Synchronization: The synchronization process is shown in Fig. 4. For simplicity, we will consider the point of view of a *client* robot fetching information from a remote *server*. Any node in the system can perform the client or remote server role, and in general, perform both roles in different exchanges.

The client starts by requesting the lists of headers from the server, which is a concatenated stream of all its most recent headers. The header list size is $S_h = \sum_i r_i$, where r_i is the number of topics provided by robot i on the network. As the size of the header is only 6 bytes (Fig. 3), the exchange of all the headers is a fast operation that usually happens within a single ZeroMQ frame[2]. This is crucial to keep the amount of information exchanged bounded: a client i can request at most $S_h - r_i$ messages in the worst-case scenario.

[2]https://rfc.zeromq.org/spec/13/

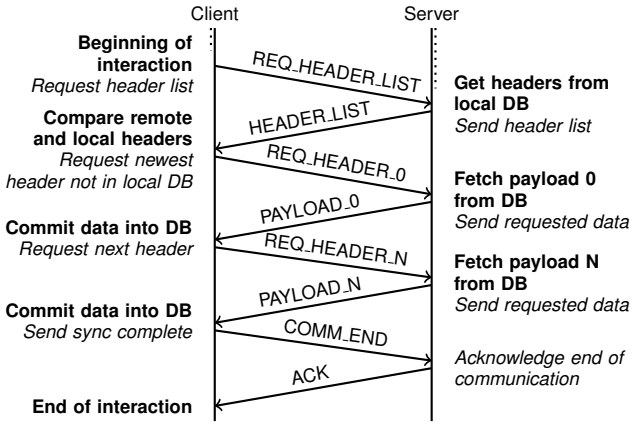

Fig. 4. Synchronization procedure for the distributed communication system. Any node in the system can perform the client or server role. Bold text represents the action of MOCHA in the robot, whereas the text in italics represents the message transmitted to the other peer. We consider the synchronization complete when the client reaches the **end of interaction** stage.

The client robot compares its local list of headers with the one from the remote server and decides which messages to request based on the timestamp field (i.e. requesting messages that are newer than the locally stored messages). Messages are requested in order based on their predefined priority. Payload transmissions are performed individually through synchronization channels. The client commits the message into its database once a transmission from the server is successful.

The client does not immediately discard old messages for a particular RID and TID to remove potential race or lock conditions during the communication exchange. This scenario occurs when three robots form a communication chain: $A \rightarrow B \rightarrow C$. In this case, C may request information from A through B, while B may be updating messages from A directly.

In case of disconnection, there are two possible scenarios:

- Short-term disconnection: ZeroMQ can recover from intermittent disconnection, and a message may continue transmission. A polled timeout of 5 seconds was set for this condition in our experiments.
- Long-term disconnection: The synchronization fails before reaching the end of interaction stage. The synchronization is interrupted and it will be performed again from the beginning if opportunity allows. In this case, the client would request a new list of headers. Only the messages that have not been synchronized previously will be requested.

3) Additional Features: MOCHA requires no explicit clock synchronization between nodes, as the message for a particular RID may only be modified by the robot that produced it. Thus, all the header data processing on the client side compares only previous timestamps from the same robot.

The synchronization procedure can be triggered based on information from the physical layer (PHY), such as a predefined signal-to-noise ratio (SNR) or a received signal strength indicator (RSSI) threshold. Alternatively, a recurrent

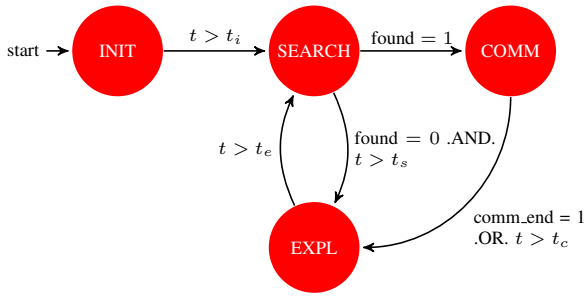

Fig. 5. State machine describing the communication-aware exploration mission for the UAV. Transitions between states occur when timers elapse (timeout) or when an event occurs, such as finding a ground robot or ending communications. In our experiments, we used $t_i = 3\,\text{min}$, $t_e = 2\,\text{min}$, $t_s = 45\,\text{s}$, and $t_c = 20\,\text{s}$.

synchronization timer can be set for every communication channel.

Finally, while our system does not explicitly support sending incremental updates of topics, the user may aggregate small messages into larger database updates which reduces network overhead substantially and allows the synchronization complexity to scale linearly with the number of topics rather than the number of messages. For example, if we desire to transmit the history of poses of a robot to a ground station, we could concatenate these poses into a single message and commit it into the database.

B. Network and Communication Metrics

MOCHA provides high-level information about the communication status between nodes. These include:

- Last synchronization time with a particular node.
- Link quality, measured as round-trip time (RTT) and network bandwidth. Metrics from the network physical layer can also be provided, such as SNR or RSSI.
- Status of the exchange: idle, comm start, comm end, timeout.

IV. Experiments

A. Communication-aware Exploration

We leveraged the metrics provided by MOCHA to inform the planning stack for a high-altitude UAV. This experiment builds on our previous work [2]: the UAV performs an exploration task in an unknown environment by following a set of predefined waypoints. While exploring the environment, the aerial robot generates a semantic map transmitted to multiple ground robots in the field. Ground robots use this map to localize themselves, find objects of interest (such as cars), and plan trajectories toward these objects of interest, which we refer to as goals.

We observed that when the UAV performs only exploration and passive communication, increasing the number of ground robots resulted in diminishing returns as maps are not transmitted back to ground robots regularly. The communication-aware exploration proposed in this section addresses the shortcomings of our previous work.

In our strategy, the UAV transitions between two main modes: exploration and communication. The state machine describing the algorithm is shown in Fig. 5. The UAV starts

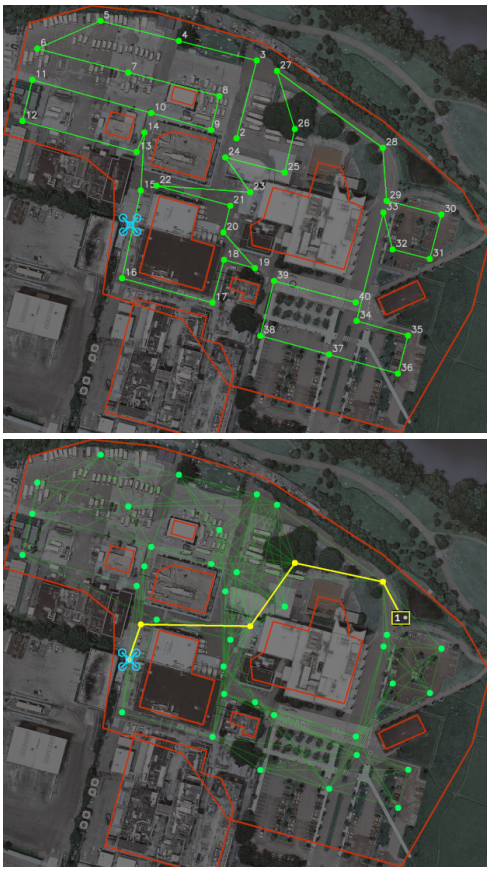

Fig. 6. Communication-aware planning. Top: A high-altitude UAV performs an exploration task on a set of predefined waypoints and transmits the aerial map to ground robots deployed in the field (INIT and EXPL states). Bottom: The UAV transitions to a search task (SEARCH state), searching for UGV 1 using predefined routes. Red lines represent the no-fly zones.

its mission with an initial exploration task (INIT), following a predefined set of waypoints. After a predetermined time, it transitions to a search task (SEARCH), where it seeks one of the ground robots to communicate in a round-robin fashion. This search is performed using the same waypoints as the exploration task, planning new routes between them, and avoiding no-fly zones. The last known position of the ground robot or the robot's current goal are used as heuristics for where to search. If the ground robot is found, the UAV acquires the current position of the ground robot and proceeds to transition to COMM, where it flies over the new robot position to maximize communication bandwidth until all communications are finished. Otherwise, the UAV transitions into a new exploration task (EXPL), where it continues to explore waypoints. If the exploration task finishes (all waypoints are reached), the UAV only performs search tasks.

We defined the UAV waypoints manually for safety reasons, but alternatively, these can be generated automatically with a coverage algorithm. The UAV performs passive communication during EXPL and INIT states too, but the quality of the channel deteriorates rapidly as the UAV flies away from the ground robots. The UAV mission ends when the battery level falls under a pre-specified threshold, and the safety pilot manually lands it.

Comm Radius [m]	Configuration		Performance		Message wait	
	UAVs	UGVs	Goals [%]	Mean time [min]	Mean [s]	SD [s]
5	1	3	36.70%	23	0.17	0.87
5	1	5	70.00%	16.4	0.18	1.29
5	1	10	**76.70%**	15.8	0.53	5.13
5	2	3	40.00%	19.7	0.23	1.44
5	2	5	73.30%	18.6	0.24	2.42
5	2	10	70.00%	**15.2**	0.63	6.90
10	1	3	46.70%	16.5	0.32	1.80
10	1	5	55.60%	14.9	0.40	2.85
10	1	10	**76.70%**	14	1.33	13.89
10	2	3	53.30%	14.1	0.61	3.27
10	2	5	70.00%	13.8	0.62	4.93
10	2	10	73.30%	**11.7**	1.11	13.95
20	1	3	50.00%	14.6	0.38	2.66
20	1	5	55.60%	13.7	0.49	3.76
20	1	10	**86.70%**	14	1.47	25.73
20	2	3	43.30%	16	1.05	6.02
20	2	5	66.70%	**12.1**	0.95	8.43
20	2	10	75.60%	13.8	3.91	55.39
30	1	3	40.00%	16.1	0.36	2.62
30	1	5	66.70%	14.9	0.95	6.85
30	1	10	**93.30%**	13.9	3.21	35.45
30	2	3	60.00%	18.3	1.20	6.60
30	2	5	62.20%	12	1.50	12.08
30	2	10	73.30%	**10.5**	6.80	80.92

TABLE I

SIMULATION RESULTS FOR MULTI-ROBOT EXPERIMENTS.

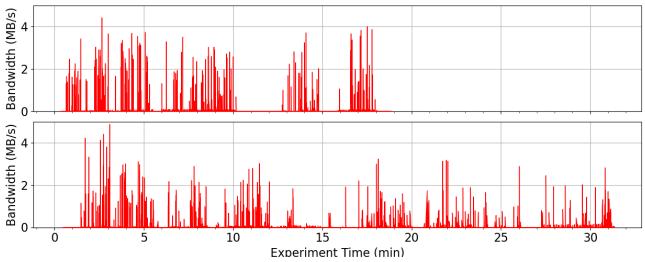

Fig. 7. Total bandwidth evolution for two real-world experiments with one UAVs and three UGVs. We can observe that communications are sporadic, with extended silent periods when robots are not within communication range.

B. Real-world deployments

We deployed the system described in Sec. III on our autonomous platforms, shown in Fig. 1:

- Ground robots: Clearpath Jackals, fitted with AMD Ryzen 3600 CPUs and an NVIDIA GTX 1650 GPU. These robots perform navigation and obstacle avoidance using an Ouster OS1-64 LiDAR.
- Aerial robots: our Falcon 4 UAV [29] was used for high altitude operations. This robot is equipped with a U-blox F9P GPS, a downward-facing OVC camera [30], and a PX4-based flight controller. This system executed the communication-aware exploration algorithm described in Sec. IV-A.

We tested both Rajant MANETs as well as ad-hoc WiFi with Babel [21] for the PHY, both working at 2.4 GHz. Experimentally, we observed that the Rajant radios perform better than WiFi+Babel, particularly since routing tables were created faster and the association time was lower. To decrease network congestion, we did not use the mesh capabilities of the Rajant nodes and only performed one-hop communication. We triggered the synchronization procedure when the RSSI threshold was greater than 20 dBm.

We performed tests in three locations, shown in Fig. 8. These locations showcase distinct communication conditions, from open spaces to dense urban environments. Experiments ran for 116 accumulated minutes, and the ground robots traveled an accumulated distance of 17.8 km. The real-time bandwidth evolution during two experiments in these locations is shown in Fig. 7. We observe that the bandwidth spikes sporadically only when robots are within communication range. Compared to the literature [27], MOCHA achieves similar bandwidth rates with far sparser communication, as the system can support extended periods of isolation between nodes. 99.5% of the 1720 interactions in these experiments finished successfully from header exchange to transmission end, and only seven interactions were interrupted. This can be explained by the high RSSI chosen to trigger communications. More importantly, we do not observe any clogging of the network when robots return to communication range.

C. Simulation Experiments

We performed simulation experiments with the twofold goal of 1) analyzing the behavior of MOCHA over varying team configurations (i.e., number of UAVs and UGVs) and communication ranges and 2) stressing the capabilities of MOCHA over congested communication regimes.

We used real-world data from our experiments to design a transmission model between robots which we implement in a photorealistic, physics-based simulator in Unity with ROS interfaces (Fig. 8). Furthermore, our simulated robots used the same perception and planning stack as our physical robots, which emulated realistic latencies and noise in our system. We limited the total experiment time to 30 minutes. Robots performed the communication-aware exploration mission described in Section IV-A. For configurations with multiple UAVs, the second UAV only performed a SEARCH task.

1) Transmission Model: We implemented a transmission model parameterized by exponential distributions describing latency per byte transmitted

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases} \quad (1)$$

where λ is related to the rate of the channel. We fitted three different distributions for the different stages of the communication: One for the header request (λ_h), one for each piece of data transmitted (λ_t), and one for the end of communication message (λ_e). These distributions aim to model the communication exchange latency as a Poisson point process. Experimentally, we obtained: $\lambda_h = 2.67$, $\lambda_e = 0.51$, and $\lambda_t = 7.63$.

We imposed the constraint that only a single robot can broadcast on the wireless channel at a time to simulate network contention. This is a worst-case scenario, as real-world experiments exhibit spatial multiplexing when node cliques are well separated.

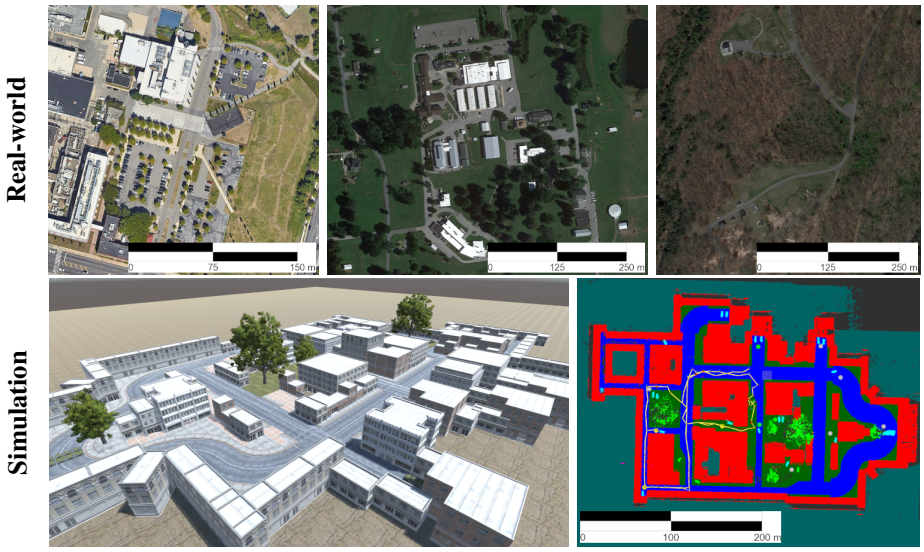

Fig. 8. **Top**: Real-world environments used for experiments. Left: urban environment showcasing tall buildings and cars. Center: semi-urban environment with open spaces and grass fields. Right: rural environment with dense forest. The areas for these environments are $62500\,\mathrm{m}^2$, $105000\,\mathrm{m}^2$ and $157000\,\mathrm{m}^2$ respectively. **Bottom**: Environment for simulation experiments. Left: Unity-based simulation overview, with a total area of $62570\,\mathrm{m}^2$. Right: Bird's-eye view of the semantic map captured by the UAV in the simulator.

2) Experimental Results: Tab. I summarizes our simulation configurations and results. We limited the communication radius to mimic different RSSI thresholds that trigger communication. We evaluated two performance metrics: the number of goals found, and the mean time to goal. We also evaluated the average message wait time and standard deviation, as a measure of channel contention.

The results indicate that task performance is primarily driven by the number of UGVs and channel contention. As expected, the number of found goals increases with the number of UGVs. Across fixed communication radii (or for fixed transmission power for the radios), the 1 UAV / 10 UGVs teams consistently outperform other configurations in terms of the number of found goals. Moreover, for a single UAV increasing the radius of the communication also has a positive impact. Teams with 3 and 5 UGVs generally benefit from an additional UAV that acts as a data mule. However, when operating with 10 UGVs or with a 30 m communication radius, the benefit of additional connectivity is offset by increased contention, which leads to decreased task performance.

The average time to goal, on the other hand, improves with larger teams and larger communication radii. Target disambiguation is an important factor for time to goal, and disambiguation messages are less affected by contention as they have a high priority and a small size. Moreover, larger communication radii and an additional UAV helped the ground robots acquire maps and make initial plans more quickly, further improving time to goal.

V. Conclusion

In this work, we proposed MOCHA: a resilient, gossip-based opportunistic communication and coordination framework for multi-robot field deployments. MOCHA enables large-scale collaboration between teams of heterogeneous robots, without imposing limitations on disconnection time, and provides network and communications metrics that can be exploited to inform higher-level autonomy behaviors. We also presented an example of communication-aware exploration by an aerial vehicle.

We performed extensive testing of MOCHA in simulation and real-world experiments. We explored the limitations of our method as the number of agents increased, and we showed that network contention does not affect performance significantly in our 1 UAV / 10 UGVs simulation. Thanks to spatial multiplexing, where multiple robots can transmit simultaneously if they are separated enough, our system scales well when both the size of the environment and the number of robots grow. A higher number of nodes may reduce network performance due to contention, particularly if the communication radius is comparable to the size of the environment. In these situations ensured network communication schemes may be preferable.

Incorporating network models from the environment based on the semantic map obtained from the UAV is an appealing path for improvement. For instance, ground robots could plan routes towards dynamic rendezvous points, maximizing communication probability with other nodes in the system.

Finally, we should note that our system does not require any time synchronization between nodes. If strict temporal ordering of the updates is require, we can introduce a Lamport clock [31] as described in [32].

Acknowledgments

The authors would like to acknowledge Alex Zhou for helping assemble and service our UAVs. We would like to thank Alice Kate Li and Priyanka Shah, for their help during field experiments. Finally, we would like to thank Dr. Ethan Stump from ARL and Dr. Barbara Dallap Schaer from Penn Vet for their collaboration with test locations.

REFERENCES

[1] C. Shen, Y. Zhang, Z. Li, F. Gao, and S. Shen, "Collaborative Air-Ground Target Searching in Complex Environments," in *2017 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*, 2017, pp. 230–237.

[2] I. D. Miller, F. Cladera, T. Smith, C. J. Taylor, and V. Kumar, "Stronger Together: Air-Ground Robotic Collaboration Using Semantics," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 9643–9650, 2022.

[3] C. Liu, J. Zhao, and N. Sun, "A Review of Collaborative Air-Ground Robots Research," *Journal of Intelligent & Robotic Systems*, vol. 106, no. 3, p. 60, 2022.

[4] Y. Mulgaonkar, "Small, Safe Quadrotors for Autonomous Flight," Ph.D. dissertation, University of Pennsylvania, 2019.

[5] M. Tranzatto, T. Miki, M. Dharmadhikari, L. Bernreiter, M. Kulkarni, F. Mascarich, O. Andersson, S. Khattak, M. Hutter, R. Siegwart, and K. Alexis, "CERBERUS in the DARPA Subterranean Challenge," *Science Robotics*, vol. 7, no. 66, 2022.

[6] K. Otsu, S. Tepsuporn, R. Thakker, T. S. Vaquero, J. A. Edlund, W. Walsh, G. Miles, T. Heywood, M. T. Wolf, and A.-A. Agha-Mohammadi, "Supervised Autonomy for Communication-Degraded Subterranean Exploration by a Robot Team," in *2020 IEEE Aerospace Conference*, 2020, pp. 1–9.

[7] H. Durrant-Whyte and T. C. Henderson, "Multisensor Data Fusion," *Springer handbook of robotics*, pp. 867–896, 2016.

[8] A. Tahbaz-Salehi and A. Jadbabaie, "A Necessary and Sufficient Condition for Consensus Over Random Networks," *IEEE Transactions on Automatic Control*, vol. 53, no. 3, pp. 791–795, 2008.

[9] K. Saulnier, D. Saldaña, A. Prorok, G. J. Pappas, and V. Kumar, "Resilient Flocking for Mobile Robot Teams," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1039–1046, 2017.

[10] M. Malencia, V. Kumar, G. Pappas, and A. Prorok, "Fair Robust Assignment Using Redundancy," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4217–4224, 2021.

[11] V. Edwards, T. C. Silva, and M. A. Hsieh, "Stochastic Nonlinear Ensemble Modeling and Control for Robot Team Environmental Monitoring," *International Symposium on Distributed Autonomous Robotic Systems (DARS)*, 2022.

[12] D. Saldaña, A. Prorok, S. Sundaram, M. F. M. Campos, and V. Kumar, "Resilient Consensus for Time-Varying Networks of Dynamic Agents," in *2017 American Control Conference (ACC)*, 2017, pp. 252–258.

[13] D. Mox, V. Kumar, and A. Ribeiro, "Learning Connectivity-Maximizing Network Configurations," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5552–5559, 2022.

[14] M. A. Hsieh, A. Cowley, V. Kumar, and C. J. Taylor, "Maintaining Network Connectivity and Performance in Robot Teams," *Journal of field robotics*, vol. 25, no. 1-2, pp. 111–131, 2008.

[15] Y. Kantaros and M. M. Zavlanos, "Distributed Intermittent Communication Control of Mobile Robot Networks Under Time-Critical Dynamic Tasks," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 5028–5033.

[16] X. Yu and M. A. Hsieh, "Synthesis of a Time-Varying Communication Network by Robot Teams With Information Propagation Guarantees," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1413–1420, 2020.

[17] K. Birman, "The Promise, and Limitations, of Gossip Protocols," *SIGOPS Oper. Syst. Rev.*, vol. 41, no. 5, p. 8–13, oct 2007.

[18] U. S. Aditya, R. Singh, P. K. Singh, and A. Kalla, "A Survey on Blockchain in Robotics: Issues, Opportunities, Challenges and Future Directions," *Journal of Network and Computer Applications*, vol. 196, p. 103245, 2021.

[19] I. Mezei, V. Malbasa, and I. Stojmenovic, "Robot to Robot," *IEEE Robotics & Automation Magazine*, vol. 17, no. 4, pp. 63–69, 2010.

[20] I. D. Miller, F. Cladera, A. Cowley, S. S. Shivakumar, E. S. Lee, L. Jarin-Lipschitz, A. Bhat, N. Rodrigues, A. Zhou, A. Cohen, A. Kulkarni, J. Laney, C. J. Taylor, and V. Kumar, "Mine Tunnel Exploration Using Multiple Quadrupedal Robots," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2840–2847, 2020.

[21] J. Chroboczek, "The Babel Routing Protocol," RFC, Tech. Rep., 2011.

[22] E. Stump, N. Michael, V. Kumar, and V. Isler, "Visibility-based Deployment of Robot Formations for Communication Maintenance," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 4498–4505.

[23] N. Cheng, W. Xu, W. Shi, Y. Zhou, N. Lu, H. Zhou, and X. Shen, "Air-Ground Integrated Mobile Edge Networks: Architecture, Challenges, and Opportunities," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 26–32, 2018.

[24] O. M. Group, "DDS Specification." [Online]. Available: https://www.omg.org/spec/DDS/1.4/PDF

[25] S. Macenski, T. Foote, B. Gerkey, C. Lalancette, and W. Woodall, "Robot Operating System 2: Design, Architecture, and Uses in the Wild," *Science Robotics*, vol. 7, no. 66, p. eabm6074, 2022.

[26] M. F. Ginting, K. Otsu, J. A. Edlund, J. L. Gao, and A. akbar Agha-mohammadi, "CHORD: Distributed Data-Sharing via Hybrid ROS 1 and 2 for Multi-Robot Exploration of Large-Scale Complex Environments," *IEEE Robotics and Automation Letters*, vol. 6, pp. 5064–5071, 2021.

[27] M. Saboia, L. Clark, V. Thangavelu, J. A. Edlund, K. Otsu, G. J. Correa, V. S. Varadharajan, A. Santamaria-Navarro, T. Touma, A. Bouman, H. Melikyan, T. Pailevanian, S.-K. Kim, A. Archanian, T. S. Vaquero, G. Beltrame, N. Napp, G. Pessin, and A.-a. Agha-mohammadi, "ACHORD: Communication-Aware Multi-Robot Coordination With Intermittent Connectivity," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 184–10 191, 2022.

[28] P. Hintjens, *ZeroMQ: messaging for many applications*. O'Reilly Media, Inc., 2013.

[29] X. Liu, G. V. Nardari, F. C. Ojeda, Y. Tao, A. Zhou, T. Donnelly, C. Qu, S. W. Chen, R. A. F. Romero, C. J. Taylor, and V. Kumar, "Large-Scale Autonomous Flight With Real-Time Semantic SLAM Under Dense Forest Canopy," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5512–5519, 2022.

[30] M. Quigley, K. Mohta, S. S. Shivakumar, M. Watterson, Y. Mulgaonkar, M. Arguedas, K. Sun, S. Liu, B. Pfrommer, V. Kumar, and C. J. Taylor, "The Open Vision Computer: An Integrated Sensing and Compute System for Mobile Robots," in *2019 International Conference on Robotics and Automation (ICRA)*, 2019, pp. 1834–1840.

[31] L. Lamport, "Time, Clocks, and the Ordering of Events in a Distributed System," in *Concurrency: the Works of Leslie Lamport*, 2019, pp. 179–196.

[32] C. Pinciroli, A. Lee-Brown, and G. Beltrame, "A Tuple Space for Data Sharing in Robot Swarms," in *9th EAI International Conference on Bio-inspired Information and Communications Technologies (BICT 2015)*. European Union Digital Library, May 2016.