2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

COSMO-Bench: A Benchmark for Collaborative SLAM Optimization

Daniel McGann, Easton R. Potokar, and Michael Kaess

Name	ntu_r3_00	
Source	MCD	
Trials	D1, N4, N8	
Duration	10:01	
Length	7.07km	
LC	1593(3.8%)	
IRLC	Wifi	Pro-Radio
	403(3.2%)	820(2.4%)

Name	ntu_r3_01	
Source	MCD	
Trials	N4, N8, N13	
Duration	8:19	
Length	5.10km	
LC	1072(2.7%)	
IRLC	Wifi	Pro-Radio
	224(2.2%)	420(3.1%)

Name	ntu_r3_02	
Source	MCD	
Trials	D2, N4, N13	
Duration	5:47	
Length	3.33km	
LC	714(4.2%)	
IRLC	Wifi	Pro-Radio
	26(0.0%)	194(2.6%)

Name	ntu_r4_00	
Source	MCD	
Trials	D1, N4, N8, N13	
Duration	10:01	
Length	8.30km	
LC	1855(3.5%)	
IRLC	Wifi	Pro-Radio
	626(2.7%)	510(5.9%)

Name	ntu_r5_00	
Source	MCD	
Trials	D1, D2, N4, N8, N13	
Duration	11:25	
Length	8.94km	
LC	2019(3.6%)	
IRLC	Wifi	Pro-Radio
	868(2.9%)	586(3.1%)

Name	kth_r3_00	
Source	MCD	
Trials	D6, D9, D10	
Duration	14:58	
Length	3.35km	
LC	432(12.0%)	
IRLC	Wifi	Pro-Radio
	231(8.7%)	845(7.9%)

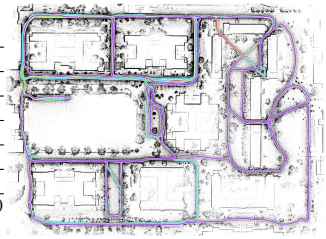
Name	kth_r3_01	
Source	MCD	
Trials	N1, N4, N5	
Duration	16:20	
Length	3.34km	
LC	402(11.4%)	
IRLC	Wifi	Pro-Radio
	284(12.3%)	650(7.7%)

Name	kth_r4_00	
Source	MCD	
Trials	D6, D10, N4, N5	
Duration	14:50	
Length	4.24km	
LC	491(13.0%)	
IRLC	Wifi	Pro-Radio
	700(10.1%)	928(8.3%)

Name	tuhh_r3_00	
Source	MCD	
Trials	D2, D3, D4	
Duration	14:18	
Length	2.15km	
LC	358(10.9%)	
IRLC	Wifi	Pro-Radio
	384(8.9%)	734(6.9%)

Name	tuhh_r3_01	
Source	MCD	
Trials	N7, N8, N9	
Duration	12:06	
Length	2.13km	
LC	356(13.5%)	
IRLC	Wifi	Pro-Radio
	280(9.6%)	719(6.4%)

Name	kittredge_loop	
Source	CU-Multi	
Trials	K1, K2, K3, K4	
Duration	42:36	
Length	7.38km	
LC	1064(19.5%)	
IRLC	Wifi	Pro-Radio
	2821(15.7%)	2382(19.4%)



Name	main_campus	
Source	CU-Multi	
Trials	M1, M2, M3, M4	
Duration	46:12	
Length	9.23km	
LC	1458(10.0%)	
IRLC	Wifi	Pro-Radio
	2020(2.8%)	3217(7.1%)

Fig. 1: The COSMO-Bench datasets. For each sequence, we plot the reference solution and enumerate metadata including – The sequence name, source data, component trials, total duration (MM:SS), total distance traveled, and the number (#) of measurements plus outlier rate (%) for both intra-robot (LC) and inter-robot (IRLC) loop-closures (#, %). For each sequence, we generate a dataset using both the Wi-Fi and Pro-Radio communication models for a total of 24 datasets. Component trial names are shortened for brevity – "D" for "Day" and "N" for "Night" for the MCD data and "K" for "Kittredge Loop" and "M" for "Main Campus" for the CU-Multi data. Note: Plots are not to scale.

***Abstract*— Recent years have seen a focus on research into distributed optimization algorithms for multi-robot Collaborative Simultaneous Localization and Mapping (C-SLAM). Research in this domain, however, is made difficult by a lack of standard benchmark datasets. Such datasets have been used to great effect in the field of single-robot SLAM, and researchers focused on multi-robot problems would benefit greatly from dedicated benchmark datasets. To address this gap, we design and release the *C*ollaborative *O*pen-Source *M*ulti-robot *O*ptimization *Bench*mark (COSMO-Bench) – a suite of 24 datasets derived from a baseline C-SLAM front-end and real-world LiDAR data. Data DOI: 10.1184/R1/29652158.**

This work was partially supported by NASA award 80NSSC24CA020 and the NSF Graduate Research Fellowship Program.
The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. {danmcgann, potokar, kaess}@cmu.edu

I. Introduction

Recent years have seen a resurgence in research into distributed optimization algorithms for Multi-Robot Collaborative Simultaneous Localization and Mapping (C-SLAM) [1]. These algorithms (often referred to as "back-ends") synthesize noisy and potentially incorrect measurements (produced by a "front-end") into an optimal state estimate [2]. However, a lack of standard benchmark datasets for distributed C-SLAM has hindered research into novel back-end methods.

In the single-robot SLAM domain, benchmark datasets (e.g. M3500 [3], Intel [4], MIT-Killian [4], CSAIL [4], Garage [5], etc.), which consist of geometric measurements output by a SLAM front-end, are widely used to develop and test novel algorithms [6–8]. Their use reduces the barrier of entry to try new ideas, provides a means by which researchers can reproduce results, enables relative performance comparisons across works, and broadly aids the community's mission to advance the state-of-the-art.

Researchers would benefit greatly from benchmark datasets for distributed C-SLAM back-ends. The need for such datasets is exemplified by recent work's reliance on single-robot benchmarks that have been partitioned to, unrealistically, simulate multi-robot scenarios [9–15].

In this work we develop the **C**ollaborative **O**pen-**S**ource **M**ulti-robot **O**ptimization **Bench**mark (COSMO-Bench)[1] – a suite of 24 benchmark datasets for distributed C-SLAM back-end evaluation (Fig. 1). We work to make COSMO-Bench as realistic as possible by deriving the datasets from real-world sensor data, a baseline LiDAR-based C-SLAM front-end, and a communication model developed from actual multi-robot communication data.

II. Benchmark Requirements

The most useful distributed C-SLAM benchmark datasets will be those that most accurately reflect the data that is observed by a multi-robot team in a real-world deployment. However, we additionally need benchmark datasets that permit accurate and insightful evaluation of C-SLAM back-end performance. In this section we outline the traits of a dataset that achieves these goals. These traits, in turn, act as the requirements for our proposed benchmark datasets.

Representative Measurements: The real world is a complex place for robots. To test the ability of distributed C-SLAM methods to operate in such conditions, we need the measurements in our datasets to reflect this complexity (e.g. measurements with non-parametric/non-stationary noise, measurements corrupted by outliers). Due to the intractability of simulating such conditions, the only way to generate such measurements will be to derive them using front-end algorithms parsing real-world data.

Loop-Closures: The key advantage that C-SLAM algorithms provide is to bound drift using loop-closures both within a robot's trajectory and between robots. We, therefore, will need to derive our benchmark datasets from data that contains plentiful loop-closure opportunities.

[1]Hosted in perpetuity at doi.org/10.1184/R1/29652158 and accessible with a user-friendly interface at cosmobench.com.

Long Traversals: Over short distances $O(100m)$, odometry algorithms can maintain state estimates that are sufficiently accurate for downstream tasks (e.g. planning) [16]. Therefore, for the effects of C-SLAM (i.e. inclusion of loop-closures) to actually be observable, and in turn for our benchmarks to be useful, we will need to derive our benchmark datasets from data of long traversals (i.e. $> 1km$).

Accurate Reference Solutions: With our datasets supporting observing the effects from different distributed C-SLAM back-ends, we will also need them to support metric computation to quantitatively evaluate these differences. To support this, we will need our benchmarks to include high-quality reference solutions (often called "groundtruth") as well as reference outlier measurement classifications.

Temporal Information: In addition to metric accuracy, a key performance measure for back-ends is their ability to compute solutions to problems in real-time. To support analysis of this performance, we will further require benchmark datasets that maintain accurate temporal information.

Convenient: For benchmark datasets to be useful, the community should want to use them. To facilitate this, the benchmark datasets should be convenient and easy to use so that researchers adopt them to test and evaluate their work.

It is only by meeting all of these requirements that our benchmark datasets will properly support researchers to test and evaluate novel C-SLAM back-end algorithms.

III. Related Work

Due to the lack of standard benchmark datasets, prior works have primarily taken three general approaches to evaluating the performance of proposed algorithms: 1) simulated multi-robot datasets [10–15, 17–21], 2) partitioned single-robot benchmark datasets [9–15], and 3) closed-source real-world data [10, 18]. While each can support the efficacy of an algorithm, even together they don't permit accurate evaluations and comparisons.

Simulated datasets are an excellent tool to evaluate C-SLAM back-end algorithms. They can be used to evaluate algorithms up to and beyond the bounds of current robotic hardware limitations (e.g. runtime, physical mobility, and team size). Simulated datasets, however, will never be capable of capturing the nuances of real-world data.

Partitioned single-robot benchmark datasets, when based on real-world data, bring increased realism. However, simply partitioning measurements from a single trial fails to accurately represent the topological structure, viewpoint variance, or measurement distribution that we expect from a multi-robot team operating in tandem and communicating to derive inter-robot measurements. Additionally, these benchmarks lack outlier measurements and, in their current formats (`g2o` [5] and `toro` [22]), lack temporal information.

A small number of works that develop distributed C-SLAM back-ends have evaluated on real-world data from multi-robot teams [10, 18]. Both test on closed-source multi-robot data. While this evaluation is great evidence for the performance of these methods, the closed-source data does not provide a means to reproduce results nor help current

researchers in the development of new algorithms.

Open-source multi-robot sensor data does exist and is being released with increased frequency [23–28]. This data, however, has not seen widespread adoption, as it requires researchers to implement their own distributed C-SLAM front-end, which can be a time-intensive and complex task. Additionally, if all researchers implement their own C-SLAM front-ends to utilize this data, then differences in these front-ends are likely to dominate performance differences and obfuscate back-end comparisons between works.

Though they have seen only limited use, there are two datasets that are close to achieving the requirements for our distributed C-SLAM benchmark datasets.

First is the University of Toronto Institute for Aerospace Studies Multi-Robot Cooperative Localization and Mapping benchmark [29]. While this benchmark can be an excellent resource for researchers, its artificial construction and controlled lab environment mean that it is not representative of the data we expect from deployed multi-robot systems.

Second is the Nebula multi-robot benchmark [30]. This benchmark consists of four datasets from multi-robot deployments in subterranean tunnel environments for the DARPA Sub-T Challenge. These datasets consist of 3D pose-graphs that were generated at operation time by a centralized collaborative SLAM system (LAMP 2.0 [30]). The Nebula datasets provide an excellent representation of real-world multi-robot data and include outlier measurements, temporal information, good intra/inter-robot loop-closures, and decent reference solutions from LiDAR scan matching to a survey-grade map. However, loop-closures are computed in a centralized manner, making them unrealistic to what we expect from distributed C-SLAM systems, and as it consists of only four datasets in subterranean environments, this benchmark is limited in its ability to test a back-end's generalization.

With only a single, limited benchmark that achieves the requirements outlined in Sec. II, we are motivated to develop additional benchmark datasets to support the community and facilitate future research.

IV. Methodology

To develop a suite of distributed C-SLAM benchmark datasets, we first identify open-source real-world data that supports our benchmark requirements (Sec. IV-A). We then define a baseline C-SLAM front-end used to derive the measurements that make up the benchmarks (Sec. IV-B) and a communication model required for deriving inter-robot measurements (Sec. IV-C). We finally derive the reference solution, noise models, and measurement classifications required to complete these benchmark datasets (Sec. IV-D).

A. Data Selection

To develop realistic benchmark datasets, we must begin with high-quality real-world data. We specifically focus on LiDAR data, as LiDAR-based SLAM systems are currently preferred in research and industry. Therefore, LiDAR-based benchmarks will be representative of real-world deployments for large sections of the robotics community.

The best real-world LiDAR data would be that gathered by actual multi-robot teams like the recent releases discussed above [23–27]. These datasets, however, all fail to support our requirements, as each fails to provide at least one of the following – an accurate reference solution, long trajectories, or opportunities for loop-closures.

Despite the lack of viable multi-robot data, there is another path to derive our benchmark datasets. We can use multiple independent trials of single-robot data gathered at different times in the same environment to simulate a multi-robot data sequence by temporally aligning the data. By using independent trials, we achieve realistic multi-robot viewpoint variance and topological structure. Additionally, by temporally synchronizing the data, we can simulate communication and, in turn, derive realistic inter-robot measurements.

This approach is made possible by our focus on LiDAR data, as it is invariant to visual differences, meaning that even data from different times will be consistent. While there may be physical differences in the environment when data is collected, these changes are not unrepresentative of dynamic objects moving during multi-robot operations.

Specifically, we synchronize multiple trials into a multi-robot sequence by selecting one to act as the anchor time t^*. For all other trials, we sample a relative start time $\Delta_i \sim \mathcal{N}(0s, 40s)$ and temporally transform its data to act as though the robot had started operation at t_i where:

$$t_i = t^* + \Delta_i \tag{1}$$

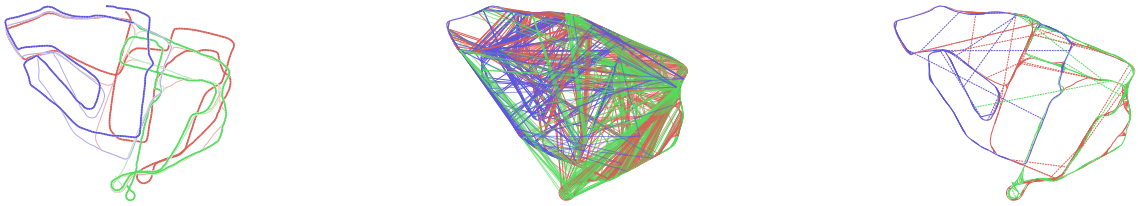
Remark 1 (Synchronization Side-Effects)**:** *While we synchronize single-robot trials to the same general operation time, we do not synchronize their measurements nor the period in which they are active. Each robot's sensors gather data at slightly different rates and offsets. Additionally, since each trial is a different length, not all robots may be active at once. We consider all robots to be "off" until the timestamp of their first measurement. Inversely, after a robot's final local measurement, we assume that the robot stays active but remains stationary such that it can continue to communicate and even derive new inter-robot measurements.*

For synchronization, however, we still require single-robot data that supports our benchmark requirements. For this, we selected the Multi-Campus Dataset (MCD) [31] and the CU-Multi Dataset [28]. Together the datasets consist of multiple trials collected by four different platforms in five unique university campus environments. The CU-Multi dataset was even designed intentionally for multi-robot applications, making an argument for temporal synchronization similar to ours. For additional details on these datasets, please see the original publications. To handle these datasets and ensure proper calibration, we utilize `evalio` [32].

B. C-SLAM Front-End

To generate benchmarks from synchronized data, we next implement a representative C-SLAM front-end algorithm consisting of an odometry algorithm, a keyframe selection method, a loop-closure detection module, and a loop-closure computation method. For each, we use existing methods representative of the state of the art.

1) Odometry: Odometry measurements, observed at the rate of LiDAR scans, are derived using LiDAR Odometry



(a) Odometry + Keyframes **(b)** Loop-Closure Detections **(c)** Loop-Closure Measurements

Fig. 2: Example results from each step in our distributed C-SLAM front-end on the `kth_r3_00` dataset. 2a depicts the odometry estimated by LOAM in solid colors, the reference trajectories in opaque colors, and the selected keyframes as dots on the odometry estimate, with each color representing one of the 3 robots. 2b depicts all potential loop-closures detected by ScanContext, with detections colored based on the robot that generated the detection. 2c depicts the final set of measurements computed by KISS-Matcher, with inliers drawn with solid lines and outliers drawn with dashed lines.

and Mapping (LOAM) – a standard baseline for LiDAR-only odometry [33]. We use an open-source implementation of LOAM provided through `evalio`.[2]

2) Keyframe Selection: From this high-rate odometry, we sub-sample keyframes – a standard practice in SLAM pipelines to reduce the problem size. We use a distance-based keyframe selection metric to sample keyframes when the odometry measurements indicate that the robot has traveled more than a threshold distance $d_{kf} = 2m$. Odometry between keyframes is computed directly by integrating the high-rate odometry measurements.

3) Intra-Robot Loop-Closure Detection: For each keyframe, we perform intra-robot loop-closure detection using ScanContext [34]. A ScanContext descriptor and RingKey are computed for the keyframe scan and compared against a database of descriptors from all prior keyframes using the "Two-Phase Search" method described in the original work. We use an open-source implementation of the ScanContext descriptor and database.[3]

4) Inter-Robot Loop-Closure Detection: To detect and compute inter-robot measurements, robots communicate raw LiDAR scans (for communication details, see Sec. IV-C). When a scan is received from a teammate, we again use ScanContext and perform the "Two-Phase Search" against the database of local keyframe descriptors. We then store the descriptor in a unique database for scans received from the teammate. When a local keyframe is selected, it is also compared against the databases for all teammates.

5) Loop-Closure Computation: For each detected loop-closure (both intra-robot and inter-robot), when detected, we attempt to compute a loop-closure measurement using KISS-Matcher [35]. We use the open-source implementation of KISS-Matcher provided by the original authors.[4] This algorithm does attempt to reject outlier measurements, but as expected, it does compute spurious measurements both due to poor detections as well as limitations in the LiDAR scan alignment, resulting in realistic outliers that we want for our benchmark datasets.

Remark 2 (Dewarping)**:** *To decouple the results of the odometry module and the loop-closure detection/measurement modules, we dewarp LiDAR scans using the reference solution before passing each scan to the detection/measurement modules.*

This front-end produces the measurements that form the basis of our benchmark datasets. An example of the results from this front-end can be seen in Fig. 2. For completeness, we also include all hyperparameters used for the front-end component algorithms alongside the online dataset release.

[2] https://github.com/contagon/evalio
[3] https://github.com/DanMcGann/scan_context
[4] https://github.com/MIT-SPARK/KISS-Matcher

C. Communication Model

To compute a realistic set of inter-robot loop-closure measurements using the pipeline above, we need to accurately model communication between robots. In this section, we derive a model from real-world data that can be used to simulate inter-robot communications for this task.

1) Communication Data Analysis: Open-source data on inter-robot communication during a multi-robot deployment is very limited. Recent data collected by Lajoie et al., however, provides us with an opportunity to analyze the characteristics of inter-robot communications [36].

The communication data was collected during a three-robot experiment in a planetary analog environment. As robots traversed, they communicated using an ad-hoc Wi-Fi network. The authors utilized network profiling tools to record the throughput and latency between all robots over periods of $1s$. In their analysis, Lajoie et al. note that the dominant indicator for network behavior is inter-robot distance, which we utilize below to summarize the data for each channel shown below in Fig. 3.

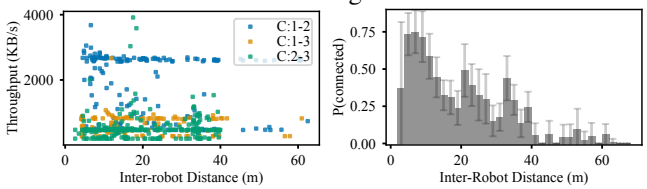


Fig. 3: Summary of Wi-Fi communication data recorded by Lajoie et al. [36], including per-channel throughput (left) and probability of inter-robot connectivity (right) against inter-robot distance. In the connectivity plots, we additionally plot 99% confidence intervals, though these are overly optimistic as the data is correlated.

From these figures, we can draw some conclusions about the communication network. Firstly, we can observe that bandwidth is uncorrelated with inter-robot distance; rather, when robots are able to connect, they appear to utilize all the available bandwidth. However, we can see that the bandwidth is not equally shared; we hypothesize that this is a side effect of running multiple instances of network profilers as each attempts to flood the network. When used for actual inter-robot communication, we predict that better network sharing is possible. Second, while bandwidth is not correlated with inter-robot distance, connectivity is correlated. Notably, connectivity between robots is probable at close range and

unlikely at larger inter-robot distances.

2) Generic Communication Model: Using the analysis of this data, we defined a generic communication model that can be used to simulate inter-robot communication.

We assume that all robots engage in a single peer-to-peer communication at a time (to minimize interference) on a shared channel (e.g. Wi-Fi, UWB, acoustic) that has a maximum bandwidth $\mathcal{B}$. Simultaneous communications will cause interference and require bandwidth sharing, which we assume to be allocated equally. However, this interference will be spatially localized due to limited communication ranges. Specifically for a communication between robots a and b, there will be interference if there exists an actively communicating robot within a distance d_{intf} meters of either a or b. We define the total number of communications occurring around a and b at timestep t as $\mathcal{I}_{ab}(t, \Theta^*)$, which uses the reference solution Θ^* to compute inter-robot distances. From this we can define the available bandwidth between robots a and b as $\mathcal{B}/\mathcal{I}_{ab}(t, \Theta^*)$.

Unlike bandwidth, according to the analysis of Fig. 3, connectivity between agents is correlated with the inter-robot distance. We assume that this correlation is modeled by a function ϕ that, given an inter-robot distance $d_{ab}(t, \Theta^*)$ computed from the reference solution Θ^*, outputs the probability that the robots a and b can successfully communicate at this timestep t. This allows us to sample the connectivity $\mathcal{C}(t)$ between two robots as $\mathcal{C}_{ab}(t, \Theta^*) \sim Bern(\phi(d_{ab}(t, \Theta^*)))$.

Together these models for bandwidth and connectivity allow us to simulate the throughput $\mathcal{T}$ between robots a and b at a time t over a small timestep δ (during which we assume that all network traits are constant) as:

$$\mathcal{T}_{ab}(t) = \underbrace{(\mathcal{B}/\mathcal{I}_{ab}(t, \Theta^*))}_{\text{Bandwidth}} \cdot \underbrace{\mathcal{C}_{ab}(t, \Theta^*)}_{\text{Connectivity}} \cdot \delta \tag{2}$$

Putting all of this together, we can define a simple process to simulate inter-robot communications as outlined in Alg. 1.

Remark 3 (Communication Initialization)**:** *To utilize this model, we need a method to initiate communication between robots. We expect that transmissions will require multiple timesteps δ of connectivity. To help ensure this, we define an initialization threshold d_{init} based on ϕ such that robots initialize communications only when they are close enough to have relatively reliable throughput to complete the communication. An attempt to initialize a new communication is successful if the two robots connectivity $\mathcal{C}_{ab}$ is successfully sampled according to the model above.*

Remark 4 (Communication Timeouts)**:** *We also define a communication timeout $T = 2s$ to terminate stale connections. This timeout is both representative of typical network protocols and also prevents our simulation from becoming deadlocked if agents travel too far apart while attempting to communicate.*

Algorithm 1 Simulate Communication

In: start/end time t_s/t_e, step δ, robots R, ref. solution Θ^*
$C \leftarrow \{\}$ ▷ Active Communications
$A \leftarrow R$ ▷ Available Robots
for t from t_s to t_e by δ **do**
  $C, A \leftarrow$ InitializeNewComms(t, A, Θ^*) ▷ Remark 3
  for $c \in C$ **do**
    $c \leftarrow$ UpdateActiveComm(c, t, Θ^*) ▷ Sample+Integrate Eq. 2
    $A \leftarrow$ CheckTermination(c, t) ▷ Complete/Fail/Maintain

3) Concrete Models: From this general model structure, we define two concrete instances, defining the parameters of the model based on expectations for common hardware. First is a Wi-Fi-based model derived from the communication data by Lajoie et al. [36]. The closed-form equation for $\phi(d) = \min\left(P_{max}, \alpha\left(\beta^{d/r_{max}} - \beta\right)\right)$ was derived from manually fitting a curve to the connectivity data in Fig. 3.

Based on the Wi-Fi model, we also define a model based on "Professional" radio equipment (Pro-Radio). Such hardware is expected to have increased range but lower bandwidth due to the increased power requirements. The remaining hyperparameters are scaled from those of the Wi-Fi model to maintain a similar structure. Both the Wi-Fi and Pro-Radio models are summarized in Fig. 4.

Model Hyperparameters		
Parameter	Wi-Fi	Pro-Radio
$\mathcal{B}$ (KB/s)	2000	1000
d_{init} (m)	30	150
d_{intf} (m)	40	150
P_{max}	0.7	0.8
α	1.1	1.8
β	0.1	0.3
r_{max} (m)	70	200

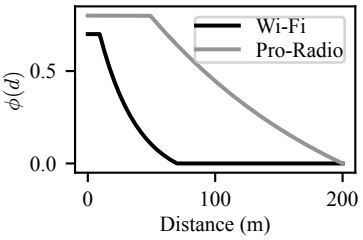


Fig. 4: Model hyperparameters (left) and connectivity functions $\phi(d)$ (right) for the Wi-Fi and Pro-Radio communication models.

4) Interactions with Front-End: To utilize these concrete models to simulate inter-robot communication for the purpose of supporting the front-end described above, there are two additional details to specify – the selection of LiDAR scans to send when a communication is initialized and the amount of data required to transmit that scan.

LiDAR scan selection is performed naively. When a communication is initialized, the transmitting robot randomly selects (uniformly) a keyframe scan that has not yet been successfully sent to the receiving robot.

We compute the data transmission size of a LiDAR scan assuming that agents only need to send $[x, y, z]$ data represented by 32-bit floating-point numbers, that the packet header sizes are negligible relative to the size of the scan, and that the data is first compressed using `xz`.[5] We assume that the compression factor provided by `xz` is modeled by a normal distribution such that the compressed size $\mathfrak{s}$ can be computed from the original size S as $\mathfrak{s} = f \cdot S$ and the compression factor f is sampled from $\mathcal{N}(\mu_{xz} = 0.653, \sigma_{xz} = 0.04)$. Where we empirically computed this distribution using a random sample of 20 LiDAR scans from the KTH Day 6 trial of the Multi-Campus Dataset [31].

Remark 5 (Communication Model Limitations)**:** *While designed for accuracy, our inter-robot communication model has non-trivial limitations. The one-active-communication assumption, equal-bandwidth-sharing assumption, and disregard of packet latency may not be perfectly reflective of actual operations. However, more impactfully, we know that our distance-based connectivity model will not reflect reality as it ignores physical interference (e.g. buildings, vegetation, walls, etc.). While our connectivity models ϕ are based on data collected in the presence of physical interference, lack of explicit modeling limits our model's accuracy.*

D. Additional Details

For completeness, in this section we provide a few final details on the dataset derivation process for COSMO-Bench.

[5] https://github.com/tukaani-project/xz

1) Prior: For all datasets, we assume that all robots have an accurate initial estimate in the global reference frame, which is taken from the reference solution. Since this estimate is quite accurate, we manually define a noise model with a rotational standard deviation of $\sigma_r = 1e^{-4}\ rad$ and a translational standard deviation of $\sigma_t = 1e^{-3}\ m$.

2) Reference Solution: Deriving a reference solution for our benchmarks is simple due to our data selection. Both the MCD and CU-Multi datasets have accurate reference solutions provided by alignment to a survey-grade map and a LiDAR-Inertial + RTK-GPS + Digital Elevation Model SLAM algorithm, respectively [28, 31]. To derive a reference solution for our benchmarks, we simply extract the poses from the provided references for each keyframe.

3) Measurement Noise Models: This reference solution also permits us to derive measurement noise models. For each data collection platform, we select a single trial and run it through the front-end above without any robot team. Using the computed measurements $m \in SE(3)$ and their respective "true" measurements m^* (derived from the reference solution), we compute a sample covariance Q for both the odometry measurements and loop-closure measurements. Since both are relative pose measurements, covariance is computed in the tangent space $\mathfrak{se}(3)$ according to:

$$Q = \sum_{m \in M} r_m r_m^\top \tag{3}$$

where $r_m = \text{Log}\left(m^{-1} \circ m^*\right)$, which is defined using standard Lie Group notation [37]. One confounding factor for computing this covariance is that it assumes none of the measurements are outliers. For loop-closure measurements that contain outliers, we compute the sample covariance on only "good" measurements, defined as having a translational and rotational error (as compared to the reference solution) less than a user-supplied threshold (e.g. $0.5\ m$ and $0.05\ rad$).

Remark 6 (Analytical Noise Models)**:** *We would prefer to use analytic noise models for each measurement over empirical models. However, the component algorithms we use in our front-end do not provide estimates for measurement noise, as doing so is extremely difficult due to iterative re-association within the algorithms [38].*

4) Measurement Classification: Finally, we make use of these noise models to define the final aspect of our dataset – reference outlier classifications. We assume that all measurements are affected by Gaussian noise. This means that the residuals for the measurements will be χ^2 distributed. We can use this fact along with the empirically computed noise models to classify our measurements. Specifically, we define a loop-closure measurement as an outlier if its measurement residual (weighted by the noise model above) exceeds the 95% χ^2 critical value.

V. COSMO-Bench

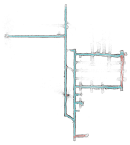
Using the process described in Sec. IV, we generate a total of 24 multi-robot benchmark datasets that make up COSMO-Bench. More specifically, we define 10 sequences using trials from the Multi-Campus data[6] and 2 from the two locations in the CU-Multi data. Due to the fact that the communication model will affect the data passed between teammates and, in turn, which inter-robot measurements are computed, we generate one dataset for each communication model for all sequences, resulting in a total of 24 datasets. For both models, we use $\delta = 0.1s$ for communication simulation. For a summary of the COSMO-Bench datasets, see Fig. 1.

Remark 7 (Noise Models)**:** *For all sequences in a common environment, we use a constant empirical noise model derived from a single trial. The trials used for noise models are as follows – NTU=Night 4, KTH=Day 6, TUHH=Day 2, and CU Boulder Campus=Kittredge Loop Robot 3.*

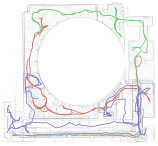
We make all benchmark datasets publicly available in JSON Robot Log (JRL) format.[7] Each dataset contains ordered prior/odometry/loop-closure measurements like familiar formats (e.g. `g2o`, `toro`). However, JRL unambiguously separates data per robot, includes temporal information for runtime evaluation, provides reference solutions for algorithm evaluation, and is built on JSON for easy parsing.

A. Nebula Multi-Robot Datasets

For the convenience of the community, we additionally provide the Nebula Multi-Robot Datasets in the same JRL format as our benchmarks (Fig. 5).



Name	tunnel
Robots	d, c
Dur.	71:06
Length	2.59km
LC	3130(7.0%)
IRLC	2426(8.7%)



Name	urban
Robots	a, d, e
Dur.	44:33
Length	1.55km
LC	846(21.4%)
IRLC	606(68.5%)

Name	ku
Robots	a, c, d, b
Dur.	57:54
Length	6.01km
LC	653(5.4%)
IRLC	232(37.1%)

Name	finals
Robots	g, h, e, c
Dur.	41:12
Length	1.21km
LC	662(11.9%)
IRLC	801(29.3%)

Fig. 5: The Nebula datasets. For each dataset, we plot the reference solution and enumerate metadata including – The dataset name, involved robots, total duration (MM:SS), total distance traveled, and the number (#) of measurements plus outlier rate (%) for both intra-robot (LC) and inter-robot (IRLC) loop-closures (#, %).

The Nebula datasets are provided as global factor graphs containing measurements generated by LAMP 2.0. To convert to JRL, we simply sequenced each robot's odometry measurements and distributed the loop-closure measurements. We additionally derive a new noise model for the datasets. Comparing the measurements in these datasets against the reference solution suggests that the recorded noise models are overly optimistic. We derive a new noise model for both odometry and loop-closure measurements empirically (Sec. IV-D.3) using the `tunnel` dataset. We additionally use this model along with the reference solution to classify inlier/outlier measurements (Sec. IV-D.4).

Remark 8 (The `urban` Reference Solution)**:** *WARNING: Careful inspection of the map produced by the reference solution of the `urban` datasets shows misaligned LiDAR scans, indicating that the reference solution provided for this dataset is somewhat inaccurate.*

VI. Conclusion

We hope that COSMO-Bench serves as a useful tool for the community as we collectively work to develop and evaluate the next generation of C-SLAM back-end algorithms.

[6]Note: The trial NTU Day 10 from MCD is omitted from any sequences due to issues with its reference solution.

[7]https://github.com/DanMcGann/jrl

REFERENCES

[1] P. Lajoie, B. Ramtoula, F. Wu, and G. Beltrame, "Towards collaborative simultaneous localization and mapping: a survey of the current research landscape," *Journal of Field Robotics*, vol. 2, no. 1, pp. 971–1000, 2022.

[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. on Robotics (TRO)*, vol. 32, no. 6, pp. 1309–1332, 2016.

[3] E. Olson, J. Leonard, and S. Teller, "Fast iterative alignment of pose graphs with poor initial estimates," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Orlando, USA), pp. 2262–2269, 2006.

[4] W. Burgard, C. Stachniss, G. Grisetti, B. Steder, R. Kümmerle, C. Dornhege, M. Ruhnke, A. Kleiner, and J. Tardös, "A comparison of SLAM algorithms based on a graph of relations," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 2089–2095, 2009.

[5] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "G^2o: A general framework for graph optimization," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Shanghai, CN), pp. 3607–3613, May 2011.

[6] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard, "SE-Sync: A certifiably correct algorithm for synchronization over the special euclidean group," *Intl. J. of Robotics Research (IJRR)*, vol. 38, no. 2-3, pp. 95–125, 2019.

[7] K. Chen, F. Bai, S. Huang, and Y. Sun, "iMCB-PGO: Incremental minimum cycle basis construction and application to online pose graph optimization," *IEEE Robotics and Automation Letters (RA-L)*, vol. 9, no. 11, pp. 10185–10192, 2024.

[8] S. Kim, R. Hsiao, B. Nikolic, J. Demmel, and Y. S. Shao, "Super-NoVA: Algorithm-hardware co-design for resource-aware slam," in *Proc. ACM Intl. Conf. on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, pp. 1035–1051, 2025.

[9] S. Choudhary, L. Carlone, H. I. Christensen, and F. Dellaert, "Exactly sparse memory efficient SLAM using the multi-block alternating direction method of multipliers," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, (Hamburg, DE), pp. 1349–1356, Oct. 2015.

[10] E. Cristofalo, E. Montijano, and M. Schwager, "GeoD: Consensus-based geodesic distributed pose graph optimization," arXiv:2010.00156 [cs.RO], arXiv preprint, 2020.

[11] Y. Tian, K. Khosoussi, D. M. Rosen, and J. P. How, "Distributed certifiably correct pose-graph optimization," *IEEE Trans. on Robotics (TRO)*, vol. 37, no. 6, pp. 2137–2156, 2021.

[12] Y. Tian, A. Koppel, A. S. Bedi, and J. P. How, "Asynchronous and parallel distributed pose graph optimization," *IEEE Robotics and Automation Letters (RA-L)*, vol. 5, no. 4, pp. 5819–5826, 2020.

[13] K. Matsuka and S. J. Chung, "Localized and incremental probabilistic inference for large-scale networked dynamical systems," *IEEE Trans. on Robotics (TRO)*, vol. 39, no. 5, pp. 3516–3535, 2023.

[14] T. Fan and T. D. Murphey, "Majorization minimization methods for distributed pose graph optimization," *IEEE Trans. on Robotics (TRO)*, vol. 40, pp. 22–42, 2024.

[15] D. McGann, K. Lassak, and M. Kaess, "Asynchronous distributed smoothing and mapping via on-manifold consensus ADMM," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Yokohama, JP), pp. 4577–4583, 2024.

[16] D. Lee, M. Jung, W. Yang, and A. Kim, "Lidar odometry survey: recent advancements and remaining challenges," *Intelligent Service Robotics*, vol. 17, no. 2, pp. 95–118, 2024.

[17] A. Cunningham, V. Indelman, and F. Dellaert, "DDF-SAM 2.0: Consistent distributed smoothing and mapping," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Karlsruhe, DE), pp. 5220–5227, May 2013.

[18] S. Choudhary, L. Carlone, C. Nieto, J. Rogers, H. Christensen, and F. Dellaert, "Distributed mapping with privacy and communication constraints: Lightweight algorithms and object-based models," *Intl. J. of Robotics Research (IJRR)*, vol. 36, no. 12, pp. 1286–1311, 2017.

[19] R. Murai, J. Ortiz, S. Saeedi, P. H. J. Kelly, and A. J. Davison, "A robot web for distributed many-device localization," *IEEE Trans. on Robotics (TRO)*, vol. 40, pp. 121–138, 2024.

[20] D. McGann and M. Kaess, "iMESA: Incremental distributed optimization for collaborative simultaneous localization and mapping," in *Proc. Robotics: Science and Systems (RSS)*, (Delft, NL), 2024.

[21] D. Hug, I. Alzugaray, and M. Chli, "Hyperion–a fast, versatile symbolic gaussian belief propagation framework for continuous-time SLAM," in *Proc. Eur. Conf. on Computer Vision (ECCV)*, pp. 215–231, Springer, 2024.

[22] G. Grisetti, C. Stachniss, and W. Burgard, "Nonlinear constraint network optimization for efficient map learning," *IEEE Tran. on Intelligent Transportation Systems*, vol. 10, no. 3, pp. 428–439, 2009.

[23] Y. Zhu, Y. Kong, Y. Jie, S. Xu, and H. Cheng, "GRACO: A multimodal dataset for ground and aerial cooperative localization and mapping," *IEEE Robotics and Automation Letters (RA-L)*, vol. 8, no. 2, pp. 966–973, 2023.

[24] Y. Tian, Y. Chang, L. Quang, A. Schang, C. Nieto-Granda, J. P. How, and L. Carlone, "Resilient and distributed multi-robot visual SLAM: Datasets, experiments, and lessons learned," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pp. 11027–11034, 2023.

[25] D. Feng, Y. Qi, S. Zhong, Z. Chen, Q. Chen, H. Chen, J. Wu, and J. Ma, "S3E: A multi-robot multimodal dataset for collaborative SLAM," *IEEE Robotics and Automation Letters (RA-L)*, vol. 9, no. 12, pp. 11401–11408, 2024.

[26] J. Kim, H. Kim, S. Jeong, Y. Shin, and Y. Cho, "DiTer++: Diverse terrain and multi-modal dataset for multi-robot SLAM in multi-session environments," arXiv:2412.05839 [cs.RO], arXiv preprint, 2024.

[27] Y. Zhou, L. Quang, C. Nieto-Granda, and G. Loianno, "CoPeD – advancing multi-robot collaborative perception: A comprehensive dataset in real-world environments," *IEEE Robotics and Automation Letters (RA-L)*, vol. 9, no. 7, pp. 6416–6423, 2024.

[28] D. Albin, D. McGann, M. Mena, A. Thomas, H. Biggie, X. Sun, S. McGuire, J. How, and C. Heckman, "CU-Multi: A dataset for multi-robot collaborative perception," arXiv:2509.19463 [cs.RO], arXiv preprint, 2025.

[29] K. Y. Leung, Y. Halpern, T. D. Barfoot, and H. H. Liu, "The UTIAS multi-robot cooperative localization and mapping dataset," *Intl. J. of Robotics Research (IJRR)*, vol. 30, no. 8, pp. 969–974, 2011.

[30] Y. Chang, K. Ebadi, C. E. Denniston, M. F. Ginting, A. Rosinol, A. Reinke, M. Palieri, J. Shi, A. Chatterjee, B. Morrell, A. Aghamohammadi, and L. Carlone, "LAMP 2.0: A robust multi-robot SLAM system for operation in challenging large-scale underground environments," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 4, pp. 9175–9182, 2022.

[31] T. M. Nguyen, S. Yuan, T. H. Nguyen, P. Yin, H. Cao, L. Xie, M. Wozniak, P. Jensfelt, M. Thiel, J. Ziegenbein, and N. Blunder, "MCD: Diverse large-scale multi-campus dataset for robot perception," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, (Seattle, US), pp. 22304–22313, 2024.

[32] E. R. Potokar and M. Kaess, "A comprehensive evaluation of LiDAR odometry techniques," arXiv:2507.16000 [cs.RO], arXiv preprint, 2025.

[33] J. Zhang and S. Singh, "LOAM: Lidar odometry and mapping in real-time.," in *Proc. Robotics: Science and Systems (RSS)*, Berkeley, CA, 2014.

[34] G. Kim and A. Kim, "Scan context: Egocentric spatial descriptor for place recognition within 3d point cloud map," in *Proc. IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, (Brisbane, AU), pp. 4802–4809, Oct. 2018.

[35] H. Lim, D. Kim, G. Shin, J. Shi, I. Vizzo, H. Myung, J. Park, and L. Carlone, "KISS-Matcher: Fast and robust point cloud registration revisited," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Atlanta, US), May 2025.

[36] P. Y. Lajoie, K. Soma, H. M. Bong, A. Lemieux-Bourque, R. Zhang, V. S. Varadharajan, and G. Beltrame, "Multi-robot decentralized collaborative SLAM in planetary analogue environments: Dataset, challenges, and lessons learned," *IEEE Trans. on Field Robotics (TFR)*, pp. 1–1, 2025.

[37] J. Solà, J. Deray, and D. Atchuthan, "A micro lie theory for state estimation in robotics," arXiv:1812.01537 [cs.RO], arXiv preprint, 2021.

[38] A. Censi, "An accurate closed-form estimate of ICP's covariance," in *Proc. IEEE Intl. Conf. on Robotics and Automation (ICRA)*, (Roma, IT), pp. 3167–3172, Apr. 2007.