MAM3SLAM: Towards underwater robust multi-agent visual SLAM

Juliette Drupt, Andrew Comport, Claire Dune, Vincent Hugel

MAM^{3}SLAM: Towards underwater-robust multi-agent visual SLAM

Juliette Drupt[a,*], Andrew I. Comport[b], Claire Dune[a] and Vincent Hugel[a]

[a]Université de Toulon, Avenue de l'Université, La Garde, 83130, France
[b]Université Côte d'Azur, CNRS-I3S, Euclide B, 2000 Rte des Lucioles, Sophia Antipolis, 06900, France

ARTICLE INFO

Keywords:
Remotely Operated Vehicles (ROVs)
Multi-robot systems
Visual SLAM

ABSTRACT

Some underwater applications involve deploying multiple underwater Remotely Operated Vehicles in a common area. Such applications require the localization of these vehicles, not only with respect to each other but also with respect to a previously unknown environment. To this end, this work presents MAM^{3}SLAM, a new fully centralized multi-agent and multi-map monocular Visual Simultaneous Localization And Mapping (VSLAM) framework. Multi-agent evaluation metrics are introduced to provide an extensive evaluation of MAM^{3}SLAM compared to the state-of-the-art multi-agent VSLAM on four two-agent scenarios: one standard airborne dataset and three new underwater datasets recorded in a pool and the sea. The results show that MAM^{3}SLAM is robust to underwater visual conditions and tracking failures, outperforms the other evaluated methods in estimating the individual and relative poses of the agents and in collaborative mapping accuracy. MAM3SLAM successfully estimates the individual and relative localization of the agents with an error lower than 5 cm on three out of the four test sequences, and is twice as accurate as competing multi-agent works in challenging visual conditions with frequent visual dropouts, poor textures, low framerate and fast motion. MAM^{3}SLAM's source code is made available, as are the underwater datasets.

1. Introduction

In underwater robotics, the most common missions involve covering large areas, for instance exploration and mapping, biological or archaeological sample collection or marine infrastructures inspection and maintenance. A now well-accepted strategy for speeding up large-area mapping is to use fleets of robots (Murphy et al., 2012). Multiple Remotely Operated Vehicles (ROVs) can also be deployed to manage the cable which provides real-time wired communication between a ROV and a teleoperation station (Laranjeira et al., 2020; Drupt et al., 2022). These ROVs are then chained along the cable to control its shape, prevent entanglement, and counteract the forces generated by the drag on the cable. The aforementioned applications need individual but also inter-robot localization within their environment to operate safely, hence the interest in multi-agent simultaneous localization and mapping (SLAM) algorithms. Conversely, multiple agents involve largest region coverage with multiple views, such that multi-agent SLAM can be expected to map a wider area in a restrained time, and with an improved robustness due to the multiplication of the viewing points. However, underwater multi-agent SLAM is generally addressed in the case of Autonomous Underwater Vehicles (AUVs) which have limited communication through the water medium (Song and Mohseni, 2014; Mangelson et al., 2018; Özkahraman and Ögren, 2022). The specificity of the present work is to consider ROVs instead of AUVs, allowing real time communication between the robots and a central surface server.

*Corresponding author
juliette.drupt@gmail.com (J. Drupt); andrew.Comport@cnrs.fr (A.I. Comport); claire.dune@univ-tln.fr (C. Dune); vincent.hugel@univ-tln.fr (V. Hugel)
ORCID(s): 0000-0003-3311-7742 (J. Drupt); 0000-0002-3959-3195 (A.I. Comport); 0000-0003-0221-5614 (C. Dune); 0000-0003-3675-4894 (V. Hugel)

Underwater SLAM usually relies on multi-sensor fusion, including visual, inertial, depth and acoustic measurements, but some works highlight the interest of visual SLAM for underwater applications (Joshi et al., 2019). Cameras have low cost, weight, and power consumption compared to acoustic technologies. In addition, they provide rich information about their environment, and monocular cameras are the main feedback sensors used for remotely operated structure inspection and scientific exploration. While visual SLAM (VSLAM) is widely investigated for airborne applications, underwater visual conditions are more challenging due to backscattering, selective color absorption, turbidity, and the effect of embedded light on the scene aspect in deep sea missions. However, recent works demonstrate that VSLAM algorithms based on ORB-SLAM (Mur-Artal and Tardós, 2017) can be robust to underwater conditions to some extent, even in the monocular case (Quattrini Li et al., 2017; Joshi et al., 2019; Hidalgo et al., 2018). In particular, ORB-SLAM Atlas (Elvira et al., 2019) is identified as a promising solution for underwater, monocular VSLAM (Drupt et al., 2023), which can cope with most tested scenarios.

Building on ORB-SLAM Atlas (Elvira et al., 2019), this paper presents a new multi-agent and multi-map monocular VSLAM framework, namely MAM^{3}SLAM, as a solution for the localization of multiple ROVs operating in a common area. In this multi-ROV scenario, allowing real-time communication with a powerful server without communication restrictions, the proposed approach is fully centralized, meaning that all the computations are performed on this central server. The contributions of the current work can be listed as follows:

- The introduction and release of MAM^{3}SLAM[1], which extends ORB-SLAM Atlas to a multi-agent

[1]https://github.com/LaboratoireCosmerTOULON/MAMMM-SLAM

case, where multiple agents can localize in the same maps and access and update their data collaboratively

- The collection and release of an underwater benchmark dataset for two-agent systems, composed of two pool sequences and one sea sequence[2]
- The proposal of several multi-agent SLAM evaluation metrics; an evaluation of MAM³SLAM on one standard airborne dataset and these three new underwater sequences[3] and an intensive comparison with state-of-the-art CCM-SLAM (Schmuck and Chli, 2019) and ORB-SLAMM (Daoud et al., 2018) works, resulting in the first comparative benchmark of these approaches in the literature, to the authors' knowledge.

While the introduction of MAM³SLAM is more of an engineering contribution to demonstrate the potential of ORB-SLAM Atlas-based multi-agent SLAM for underwater, visually challenging scenarios, the recording and release of the underwater datasets and the proposed benchmark are the main contributions of the current work.

Section 2 presents the related work, including VSLAM for underwater applications and the corresponding specificities, and the state-of-the-art in terms of multi-agent VSLAM. Section 3 describes the algorithm of MAM³SLAM. Underwater multi-agent datasets collection is detailed in Section 4 and multi-agent VSLAM evaluation methodology is given in Section 5. Evaluation results are presented and discussed in Section 6, before concluding in Section 7.

2. Related work

The first mention of visual-based SLAM for underwater applications can be found in (Eustice et al., 2005, 2006), where the wreck of the Titanic was mapped using a visual-based SLAM information filter, combined with navigation measurements involving a tilt sensor, a magnetometer, a Doppler Velocity Log (DVL), and pressure and altitude sensors. In line with this first work, many more recent underwater visual-based SLAM algorithms rely on additional navigation sensors (Zhang et al., 2022) and are, therefore, not purely visual works, such that they are finally out of the scope of the VSLAM definition. Although VSLAM is widely investigated for airborne applications, only a few works focus on underwater VSLAM, and even fewer ones investigate multi-agent VSLAM for underwater.

2.1. Challenges in underwater vision

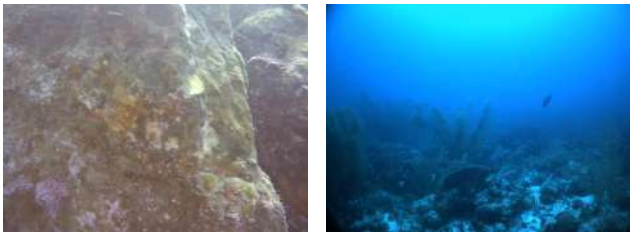
VSLAM algorithms usually assume that the scene is rigid, static and Lambertian. However, these assumptions are challenged by underwater visual conditions. Light propagation through the water is affected by back-scattering, selective color absorption, and turbidity (Akkaynak et al., 2017; Wang et al., 2019). This phenomenon can be compensated by estimated the water model from one view and the corresponding depth map (Akkaynak and Treibitz, 2019) or multiple views of the scene (Boittiaux et al., 2023b). Although very efficient, those techniques require a prior dense 3D mapping of the scene and are therefore not straightforward for the usual online monocular VSLAM algorithms. Attempts of image restoration steps in VSLAM works will be further discussed in Section 2.2.

The presence of embedded lights in deep sea missions also affects the aspect of the scene (Ferrera et al., 2019) and, conversely, scenes in shallow water can feature surface effects, known as flickering. As a result, the image of an underwater three-dimensional point depends on different factors including its position with respect to the camera, its depth with respect to the surface, but also the weather conditions. In extreme cases, a camera placed in open water or too far from the surrounding objects may not be able to observe the seabed or any other environment feature. This scenario can occur when a robot moves down the water column towards its working depth or if the camera is facing away from any landmark. Such a dropout example is given in Figures 1.

(a) Before dropout (b) During dropout

Figure 1: Visual dropout in the sea, off the coast of Saint-Raphael, France (Drupt et al., 2023)

In addition, while most airborne VSLAM works investigate indoor, industrial, or urban landscapes featuring few mobile objects and mostly artificial environments, underwater applications can feature a large variety of landscapes, including low-textured, natural environments and mobile elements including fish, seaweeds, and suspended particles, as illustrated in Figure 2.



(a) Suspended particles (b) Fishes and seaweeds (Joshi et al., 2019)

Figure 2: Examples of mobile objects in underwater, natural environments

[2] https://github.com/LaboratoireCosmerTOULON/2-agent-datasets
[3] https://youtu.be/tmDzvdISuMk

2.2. Monocular VSLAM for underwater

Coping with the aforementioned underwater-specific visual conditions is crucial when applying VSLAM techniques underwater. Some works investigate an image restoration step for each incoming frame (Salvi et al., 2008; Cho and Kim, 2018) to correct underwater visual distortion effects. However, this operation is always an approximation and can be computationally expensive, depending on the physical accuracy of the model used. As a result, an interest has been shown for off-the-shelf, underwater robust algorithms. Evaluations of state-of-the-art monocular VSLAM algorithms on underwater scenarios demonstrate that VSLAM works based on ORB-SLAM (Mur-Artal et al., 2015), and more specifically ORB-SLAM Atlas (Elvira et al., 2019; Campos et al., 2021) can be robust to underwater visual conditions (Joshi et al., 2019; Drupt et al., 2023).

ORB-SLAM (Mur-Artal et al., 2015; Mur-Artal and Tardós, 2017) is a monocular VSLAM algorithm based on ORB features (Rublee et al., 2011), which strongly impacted VSLAM by providing a publicly available algorithm able to perform tracking, local mapping in KeyFrame (KF) windows, a tracking failure recovery module based on relocalization and loop detection and closing, along with outstanding accuracy. Relocalization and loop detection are performed by a Bags-of-Words (BoW) place recognition module based on DBoW2 (Galvez-López and Tardos, 2012), where candidate matches are validated a second time according to consistency criteria over a window of connected KFs to reject bad loop closures. Even though it is quite robust to underwater visual conditions, ORB-SLAM is still reported to suffer from critical tracking failures caused by incorrect feature matching and triangulation or visual dropouts, from which the system fails to recover. Indeed, ORB-SLAM's tracking failure recovery consists of a relocalization attempt in the map, which becomes useless when the camera moves outside the already mapped area.

ORB-SLAM Atlas (Elvira et al., 2019) extends ORB-SLAM with a new, improved tracking failure recovery strategy, designed to handle relocalization failures. If lost, ORB-SLAM Atlas initially tries to relocalize in the KF database, similarly to ORB-SLAM, but initializes a new map if relocalization fails within a timed window. All KFs are stored in the same database, which is queried by DBoW2 at each new KF insertion with the same place recognition module as ORB-SLAM's loop closure. Two matched maps are fused into one through a global bundle adjustment. The system can thus reuse previous mapping data after a fusion between the current map and an old one. In addition, ORB-SLAM Atlas extends relocalization to all maps of the system. An implementation of ORB-SLAM Atlas is provided in the ORB-SLAM3 library (Campos et al., 2021), which unifies recent developments on ORB-SLAM. Underwater evaluations of this work demonstrate that the new, multi-map SLAM recovery strategy of ORB-SLAM Atlas makes it significantly more robust to tracking inconsistencies and visual dropouts than ORB-SLAM (Drupt et al., 2023), making it a promising candidate for underwater visual-based localization and mapping.

2.3. Multi-agent SLAM architectures

Multi-agent SLAM systems involve several robotic agents moving around in the same area and providing local information to the system. The multi-agent SLAM problem consists of localizing these multiple agents while building a consistent map of the scene using these multiple local observations inputs. On the one hand, multi-agent SLAM allows the localization of several agents with respect to each other and their environment, while on the other hand, multiple agents involve larger scene coverage and multiple viewing angles, which is expected to increase mapping accuracy by fusing observations with wide baselines (Cieslewski et al., 2018; Schmuck and Chli, 2019). In the current underwater application target, these properties can be expected to reduce tracking failures by relying on a more accurate map.

Two kinds of measurements can be involved in multi-agent SLAM systems. Intra-agent, or *individual* measurements only characterize the localization and local surroundings of the agent performing the measurement. Conversely, inter-agent, or *relative* measurement, characterizes the relative localization of several agents (Cieslewski et al., 2018). In addition, multi-agent SLAM approaches can be classified according to the distribution of their computations among the agents and an optional central server, as illustrated in Figure 3. In fully centralized approaches, all computations are done on a central server, which can be one of the agents or an external server. Other agents only perform measurements and send their data to the server. Optionally, the server can send back the estimated locations to the agents. Centralized approaches strongly rely on the server having sufficient computational resources and bandwidth. As a result, they are not scalable to many agents and are not robust to communication failures. Conversely, decentralized approaches distribute the computations among the agents (fully decentralized) or between the agents and a central server (partially decentralized). The main motivations of partially decentralized approaches are to reduce the server's computational payload and provide better robustness to communication failures since agents can keep estimating a localization based on their measurements if the server is unavailable (Schmuck and Chli, 2019). The scalability of such methods still depends on the server's computation resources and requires to exchange significant amounts of data. Fully decentralized algorithms do not involve any servers. Agents communicate their measurements and, optionally, a state prior to each other, and all computations are distributed among them. Such approaches are motivated to reduce the communication bandwidth and computational requirements for a more scalable multi-agent localization scheme. Decentralized works focus on reducing bandwidth requirements through communication sparsification, data marginalization, and inter-robot data exchanges condensation to improve scalability

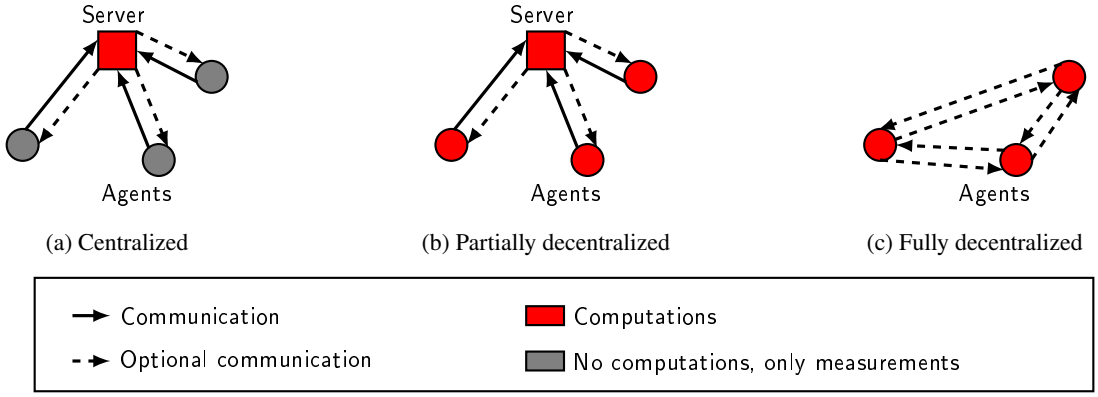


Figure 3: Multi-agent SLAM communication schemes

Table 1
Communication schemes and computations distribution in recent ORB-SLAM-based multi-agent works CORB-SLAM (Li et al., 2018) ,ORB-SLAMM (Daoud et al., 2018), CCM-SLAM (Schmuck and Chli, 2019) and SwarmMap (Cao et al., 2023) and in the proposed algorithm MAM³SLAM, where 'T' stands for 'tracking', 'LM' for 'local mapping', 'LC' for 'loop closing', 'MM' for 'map merging' and 'R' for 'recovery'.

	Communications	Category	Computations on agents	Computations on server
CORB-SLAM	Maps: agents → server	Partially decentralized	T, LM, LC	MM
ORB-SLAMM	Frames: agents → server	Centralized	∅	T, LM, LC, MM, R
CCM-SLAM	KFs: agents ↔ server	Partially decentralized	T, LM	LC, MM
SwarmMap	Map operations: agents → server	Partially decentralized	T, LM	LC, MM
MAM³SLAM	Frames: agents → server	Centralized	∅	T, LM, LC, MM, R

and robustness to communication restrictions (Luft et al., 2018; Dubois et al., 2019; Özkahraman and Ögren, 2022). However, fully decentralized algorithms are less accurate than those involving a central server since observations are never fused at a global scale.

2.4. Multi-agent Visual SLAM

Historically, multi-agent VSLAM has been investigated alongside single-agent VSLAM (Zou et al., 2019). Individual measurements are provided by embedded cameras and, in a few works, camera views from multiple neighbor agents are used to provide a relative measurement using multiple-view geometry (Zou and Tan, 2013; Cieslewski et al., 2018). Although some recent Bayesian works investigate fully decentralized approaches (Leonardos and Daniilidis, 2017; Cieslewski et al., 2018), most works, including the ones based on ORB-SLAM, are centralized or only partially decentralized (Li et al., 2018; Daoud et al., 2018; Schmuck and Chli, 2019).

With regards to underwater applications, some works investigate the problem of multi-agent VSLAM for AUVs (Mangelson et al., 2018; Özkahraman and Ögren, 2022), where the main challenge consists in managing information sharing between AUVs under the bandwidth limitations of underwater wireless communications in a distributed, sparsely communicating manner. However, by allowing real-time, large-bandwidth communication with a surface server, multi-ROV applications fit the standard communication configuration of airborne multi-agent VSLAM systems, where inter-agent data fusion can be performed on a central server in a centralized or partially decentralized scheme (Zou et al., 2019). In addition, having a physical connection between the ROVs and the server reduces the risk of data transmission failure between the agents and the server. According to this configuration and the observations reported in Section 2.2, we focus only on ORB-SLAM-based multi-agent works in the following.

ORB-SLAM being both efficient, highly accurate, and open source, it is used as a basis for multiple multi-agent works (Li et al., 2018; Daoud et al., 2018; Schmuck and Chli, 2019; Cao et al., 2023). Therefore, these works implement the main functionalities involved in ORB-SLAM, including (i) tracking, (ii) local mapping, (iii) loop closing, and (iv) an optional tracking failure recovery module. In addition, multiple agents involve a need to fuse local mapping information from each agent and, therefore, a map merging module. Communication schemes and computation distributions in these works are given in Table 1. CORB-SLAM (Li et al., 2018) introduces the first ORB-SLAM-based multi-agent architecture. Each agent runs ORB-SLAM individually and builds its own maps. These maps are sent to a central server, which tries to detect overlapping regions using a DBoW2 place recognition similar to ORB-SLAM's loop closure. Matched maps are then fused into a single common map. The global map is sent to the agents after each update. ORB-SLAMM (Daoud et al., 2018) is a fully centralized algorithm

designed for single or multiple agents. It implements a multi-map tracking failure recovery strategy that systematically creates of a new map in case of tracking failure, which is intended to be aligned with the previous maps by a place recognition module. Conversely, CCM-SLAM (Schmuck and Chli, 2019) implements a monocular, partially decentralized framework to apply to vehicles with limited onboard memory and computational resources under communication bandwidth constraints. Each agent conducts ORB-SLAM's tracking and local mapping and only maintains a window of local KFs. New, local map information is returned to the server, which stores map information and handles loop closing and map merging. A key contribution to that work is the ability of the agents to keep performing individual visual odometry if communication with the server is broken. A downside of this approach is that no SLAM failure handling strategy is implemented. Extending the scalability of such multi-agent approaches is also a challenge, which has been investigated recently in SwarmMap (Cao et al., 2023). This work presents a partially decentralized architecture similar to the one of CCM-SLAM but reduces the communication costs significantly by not sending the local maps to the server but rather the operations to be performed on the global map, given the local observations. These map update tasks are then handled by the server using a SLAM-specific task scheduling module. Their implementation is open source but not straightforward for existing systems because it relies on CUDA (NVIDIA et al., 2020).

2.5. Map merging in multi-map VSLAM

Environment mapping can be decomposed into several disconnected local maps in multi-map SLAM systems. Such systems include some multi-agent SLAM algorithms, like ORB-SLAMM (Daoud et al., 2018) and CCM-SLAM (Schmuck and Chli, 2019), but also SLAM algorithms with a multi-map tracking failure recovery strategy, like single-agent ORB-SLAMM (Daoud et al., 2018), ORB-SLAM Atlas (Elvira et al., 2019) or Dual-SLAM (Huang et al., 2020). In order to produce consistent environment mapping, multi-map SLAM systems implement map merging algorithms, which include region overlap detection between disconnected maps and map alignement.

These map merging algorithms can differ from a SLAM system to another. In Dual-SLAM (Huang et al., 2020), where new map creation is triggered by tracking failure, map merging is driven by spatial and temporal consistency using a backward SLAM to reach the previous map and connect it to the new one. Other works mainly rely on a loop-closing-like map merging, where inter-map place recognition is performed similarly to loop closure detection, and map and loop fusions are conducted similarly. This second category of map merging algorithms, implemented in all other aforementionned ORB-SLAM-based works, does not involve a spatial or temporal prior and is thus more suitable for fusing multiple maps created by multiple agents, where such a prior is usually unavailable.

In ORB-SLAMM (Daoud et al., 2018), a map merging thread continuously iterates place recognition among all KFs from all maps. This thread is unique in the system and considers all KFs independently from their agent creator. ORB-SLAMM's map overlap detection is similar to ORB-SLAM's loop closure by DBoW2 (Galvez-López and Tardos, 2012) for KF matching and deducing an initial-guess inter-map transformation from this match. However, unlike ORB-SLAM's loop closure, ORB-SLAMM does not implement inter-map place recognition consistency validation over a window of connected KFs, reducing place recognition computational cost and decreasing its robustness by making it quite permissive. In addition, ORB-SLAMM does not fuse inter-map points observations when computing map alignment, and merged maps remain distinct entities with separate KFs and BoW databases. Therefore, data from old maps cannot be reused in the localization process. It should also be noted that the computational cost of ORB-SLAMM's map merging thread is unnecessarily high because it continuously loops over all KFs from all maps to try to find a match instead of querying the KF database only at new KF insertion.

CORB-SLAM (Li et al., 2018), CCM-SLAM (Schmuck and Chli, 2019) and ORB-SLAM Atlas (Schmuck and Chli, 2019) implement an almost identical map-merging strategy, which is very similar to ORB-SLAM's loop closure, both with regards to map overlap detection, but also transformation estimation between the matched KFs, including several steps of validation checks and transformation refinement. In addition, merged maps become a single entity such that, in CCM-SLAM (Schmuck and Chli, 2019), multiple agents can localize in the same map simultaneously. Although SwarmMap relies on a quite different map information aggregation pipeline, its map-merging strategy is similar to these works.

2.6. Multi-agent VSLAM benchmarking

Multi-agent VSLAM benchmarking involves both datasets and evaluation metrics. Multi-agent VSLAM datasets can be either recorded using multiple devices, or simulated by simultaneously playing several video sequences recorded in the same environment. The first solution features real multi-agent cases, where agents may see each other. Such datasets can also integrate dynamic changes in the environment, which will be observed synchronously by the multiple agents. However, they represent a higher acquisition cost and are quite rare in the literature. While some works record their multi-agent datasets for evaluation (Schmuck and Chli, 2019), there is no such public dataset at the time of writing, to the best of the authors' knowledge. The second solution above is less realistic for a real multi-robot application, but is far easier to obtain. On the one hand, it is impossible for the simulated agents to observe each other, and possible dynamic changes in the environments will be observed out of sync among the simulated agents. On the other hand, their acquisition only involves deploying a single device or can be realized from existing sequences recorded in the same

static environment. In (Schmuck and Chli, 2019), several sequences of the EuRoC Machine Hall dataset (Burri et al., 2016) are used to simulate multiple agents, while other works rely on the KITTI odometry dataset (Geiger et al., 2012), including (Cieslewski et al., 2018; Daoud et al., 2018).

However, while airborne VSLAM evaluation can rely on standard, public datasets recorded in different environments, featuring several sequences in each of these environments and visual conditions with various trajectories of gradual difficulty (Geiger et al., 2012; Burri et al., 2016; Schubert et al., 2018), there is no equivalent yet in the underwater field, because of the important cost and resources required for acquiring such data. Although a few underwater visual datasets have been released these past few years (Quattrini Li et al., 2017; Joshi et al., 2019; Ferrera et al., 2019; Boittiaux et al., 2023a), none of them include overlapping sequences without significant visual conditions or environment change. As a result, existing public underwater datasets do not allow multi-agent VSLAM evaluation in underwater conditions.

Regarding evaluation metrics, most multi-agent works use the Root-Mean-Square (RMS) Absolute Position Error (APE) to characterize the localization accuracy of each agent's trajectory or *individual* localization accuracy (Cieslewski et al., 2018; Daoud et al., 2018; Schmuck and Chli, 2019; Dubois et al., 2019). However, while relative localization is crucial when deploying multi-agent systems, a criterion based on relative localization is usually missing, notably in (Daoud et al., 2018; Schmuck and Chli, 2019). Mapping accuracy and collaboration of each agent to the map — denoted *collaborative mapping* — is also rarely considered and is not studied in (Daoud et al., 2018; Schmuck and Chli, 2019). Some collaborative mapping criteria can, however, be found in collaborative exploration strategy validation (Yu et al., 2022), such as ratio of areas explored by the agents to the entire explorable space, the average overlapping explored area over each pair of agents, or the duration required for having a given percentage of the scene mapped. Although these criteria are defined to evaluate active exploration efficiency of agents with adaptive trajectories, they may be used as a basis for collaborative mapping efficiency characterization in the VSLAM works under consideration.

Finally, it is worth noticing that, to date, there is no comparative reference for ORB-SLAM-based multi-agent work.

3. MAM³SLAM algorithm

MAM³SLAM builds on the ORB-SLAM Atlas (Elvira et al., 2019) implementation provided by ORB-SLAM3 (Campos et al., 2021). MAM³SLAM algorithm is represented in Figure 4.

3.1. Overall architecture

MAM³SLAM is a central multi-agent SLAM, meaning all computations are performed on a central server. Agents only send frames to the server. A tracking thread and a local mapping thread are run for each agent, on the central server. Loop closing and map merging tasks are conducted by one unique, common thread in the multi-agent system. A system of n agents then runs on $2n + 1$ threads, including n tracking threads (one per agent), n local mapping threads (one per agent), and 1 single, common loop closing and map merging thread. Communication schemes and computations distribution in MAM³SLAM are summarized in Table 1 for comparison to the state-of-the-art works. Similarly to ORB-SLAM Atlas, an additional global bundle adjustment thread is launched after a loop correction or map merging. The differences between ORB-SLAM Atlas and MAM³SLAM are the following:

- The main contribution lies in creating multi-agent VSLAM instances and making maps a shared resource between multiple agents, which was not implemented in ORB-SLAM Atlas. To this end, multiple agents can localize on the same map and access and update its data.
- Multi-threading support was implemented to allow concurrent map access and update, hence the need to protect the map from concurrent modification.
- A new KF insertion algorithm was implemented for agents locating on the same map. This involves protection from concurrent modifications.

3.2. Shared multi-map resources

Maps are initialized individually by the server for each agent but are shared among them using the multi-map representation proposed in ORB-SLAM Atlas and denoted as the *Atlas*. KFs from all maps are stored in a common database to which all agents contribute. When a new KF is inserted, a place recognition query is performed over the KF database. If a match is found, a loop closing or a map merging operation is performed, depending on whether the matched KF belongs to the same map or not. ORB-SLAM3 inspires this process since having multiple agent inputs does not modify the intrinsic behavior of the Atlas. Two matched maps $\mathcal{M}_i$ and $\mathcal{M}_j$ become a single map $\mathcal{M}_k$ that includes all KFs from the original $\mathcal{M}_i$ and $\mathcal{M}_j$, and merges their map point observations. All the data originally contained in $\mathcal{M}_i$ and $\mathcal{M}_j$ is thus made available for tracking and local mapping *via* $\mathcal{M}_k$. Making the Atlas a central resource for multiple agents allows fusing maps created by different agents, enabling any agent to reuse the map KFs and map points created by another agent in its tracking process. If lost, agents can also relocalize on any map, even if they never contributed to it. Furthermore, several agents can localize on the same map, enabling relative localization estimation and collaborative map incrementation.

A map is called *active* if at least one agent is currently localizing on it. Active maps are being updated by new KF

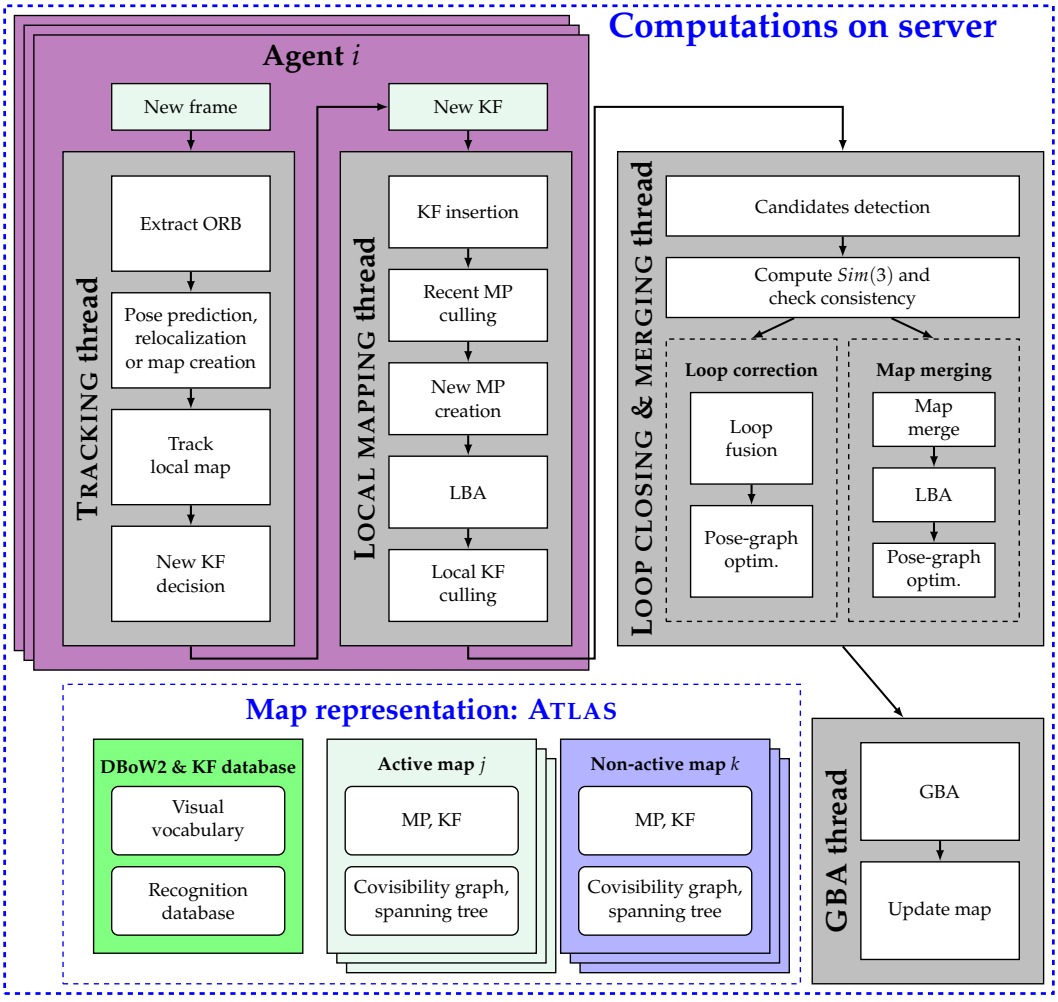


Figure 4: MAM³SLAM multi-map representation and workflow. LBA, GBA and MP stand, respectively, for 'Local Bundle Adjustment,' 'Global Bundle Adjustment,' and 'Map Points'.

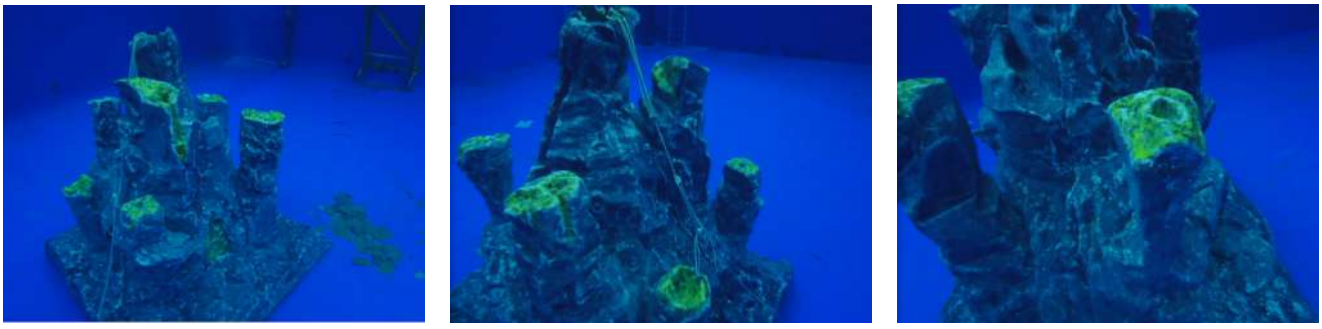
insertions and local mapping. Contrary to ORB-SLAM3, where map fusion can only happen between the currently active map and an old released map, MAM³SLAM allows the fusion of two active maps. This operation is, however, fully consistent with the ORB-SLAM3 merging algorithm and does not affect the SLAM. When several agents localize on the same map, the map is protected from concurrent updates from agents' local mapping threads by a mutex.

3.3. New KF insertion

In ORB-SLAM3, a new KF is inserted if (i) less than 90% of the map points of the current reference KF are visible in the current frame and if either (ii) more than 1 second has passed since the last insertion or (iii) the local mapping is idle. The local mapping thread periodically checks if some new KFs are to be inserted and processes them individually within a given period. However, in a multi-agent scenario, two agents on the same map may slow down their local mapping due to the need to wait for the mutex, thus delaying the insertion of new KFs. Therefore, in MAM³SLAM, KF insertion is modified so that the local mapping inserts all KFs into an insertion queue at each iteration. If many KFs are inserted, however, this may reduce the idle time of the local mapping, resulting in a decreasing number of new KF creation. As a result, in MAM³SLAM, the KF insertion criterion of ORB-SLAM3 is modified to force KF insertion if more than five consecutive frames satisfy criterion (i) but not (ii) or (iii).

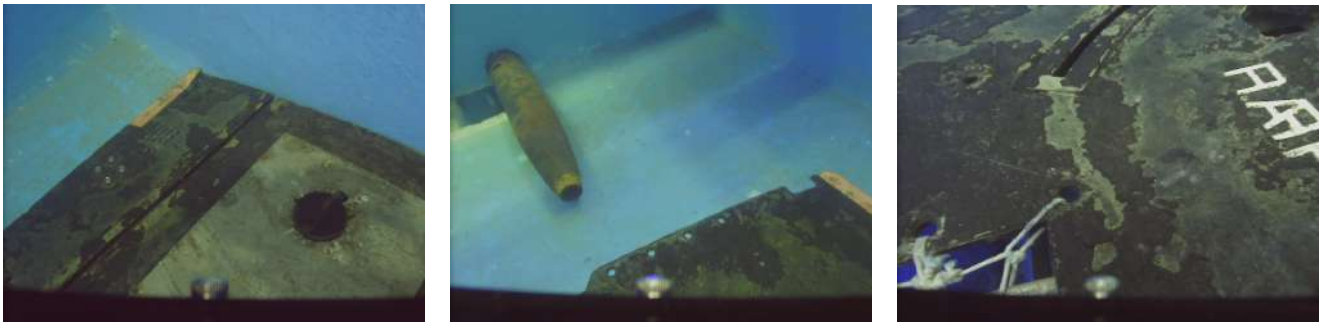
4. Underwater multi-agent datasets collection

As reported in Section 2.6, only a few public underwater VSLAM datasets are available, and unfortunately, none of them includes sequences recorded in the exact same environment or that may be divided into several disconnected sequences with spatial overlap. Evaluating the robustness of multi-agent works to underwater visual conditions and trajectory constraints was not straightforward. There was thus a need for creating underwater multi-agent VSLAM datasets. Because of the operational cost of deploying multiple underwater robots simultaneously, we privileged playing multiple agents from multiple videos acquired on the same site without major aspect changes in the environment. Three



(a) (b) (c)

Figure 5: Overview of the *Tank 1* dataset



(a) (b) (c)

Figure 6: Overview of the *Tank 2* dataset

two-agent datasets were collected this way, featuring different environments and challenges, denoted *Tank 1*, *Tank 2* and *Sea diving*, and illustrated in Figures 5, 6 and 7 respectively. The main characteristics of the three datasets are summarized in Table 2.

Tank 1 and *Tank 2* datasets are both recorded in a pool using the embedded camera of a BlueROV2. *Tank 1* is quite an easy sequence featuring slow motion around a highly textured artificial marine reef. The agents move around the reef with different radii and depths (Figure 9a). This dataset allows for evaluation of the SLAM with easy underwater visual conditions and validation of the proposed collaborative mapping approach.

The *Tank 2* dataset features fast motion, including pure rotations, around submarine spare parts. It is recorded at a low frame rate (5 Hz), and the camera sometimes faces poorly textured areas. In addition, it simulates the visual dropouts that may occur in open water (see Section 2.1) by occasionally facing only the low-textured walls of the pool, as represented in Figure 8. This dataset is thus particularly difficult. Agents' trajectories are represented in Figure 9b. In addition, Agent 1's sequence is easier than Agent 0's, with more global scene views and less motion blur. This dataset aims to evaluate the robustness of the SLAM in particularly difficult visual conditions leading to repeated tracking failure and to show the interest of collaborative mapping in improving individual localization when one of the agents has difficult visual conditions.

Finally, the *Sea diving* dataset extends the evaluation to a real underwater field scenario. It is recorded by divers with a GoPro in the Mediterranean Sea, at shallow depth, in clear water. Suspended particles, fish, and a second diver appear in some frames. Different agent recordings are made, moving slowly around a rock with quite similar trajectories at slightly different altitudes (Figure 9c).

As shown in Table 2, the volumic coverage of the two tank datasets is limited to 70 and 48 m^3, respectively, due to pool size limitations and size of the submerged objects. However, the volumic coverage of the *Sea diving* dataset is significantly larger, reaching approximately 200 m^3, which is consistent with the size of a small wreck or, for instance, an underwater wind turbine. This spatial span also matches the order of magnitude of usual SLAM datasets recorded with aerial drones (Burri et al., 2016). Although the volumes and durations of our new datasets may be considered short with regard to the vastness of the oceans, we argue that the main objective of these new datasets is not to account for the long-term exploration of large fields but rather for the specificities of underwater visual conditions. Long-term maintenance and reuse of underwater maps is a significantly different problem than the one addressed here since the aspect of an underwater site can change significantly in a few

(a) (b) (c)

Figure 7: Overview of the *Sea diving* dataset

(a) Before dropout (b) During dropout

Figure 8: Simulation of a visual dropout in the *Tank 2* dataset (Agent 1)

hours, depending on the weather conditions at visiting times. We, therefore, believe that our new datasets are sufficient for extending multi-agent SLAM algorithm validation to the underwater field.

Generating ground truth is particularly challenging in underwater environments. Similarly to (Ferrera et al., 2019), a comparative baseline for SLAM evaluation is computed for all of our new underwater datasets using the Structure-from-Motion software Colmap (Schönberger and Frahm, 2016), which performs an offline reconstruction of the scene with exhaustive matching between images and outputs an accurate estimation of the camera trajectory. This output is not ground truth but is still a fair reference for online SLAM evaluation. The scaling factor is retrieved from the known dimensions of the submerged objects. The trajectories of the agents are represented in Figure 9 with respect to the surrounding objects.

5. Evaluation methodology

In order to evaluate MAM³SLAM and compare its performances with respect to state-of-the-art multi-agent and multi-map VSLAM works, a choice of datasets and evaluation criteria is necessary. Section 5.1 presents the competing multi-agent works evaluated, Section 5.2 indicates the evaluation datasets used and Section 5.3 introduces evaluation criteria.

5.1. Competing works

Following the state-of-the-art multi-agent VSLAM presented in Section 2, we select ORB-SLAMM (Daoud et al., 2018) and CCM-SLAM (Schmuck and Chli, 2019) for conducting a comparative evaluation with MAM³SLAM. While, to the best of our knowledge, ORB-SLAMM is the only state-of-the-art multi-agent VSLAM work with a multi-map recovery strategy, CCM-SLAM is the most complete state-of-the-art real-time multi-agent VSLAM framework at the time of writing. The main differences between these two works and MAM³SLAM are summarized in Table 3.

5.2. Datasets

In order to evaluate MAM³SLAM on a standard VSLAM benchmark, a first evaluation is conducted using the *Machine Hall (MH)* sequences from the *EuRoC* MAV Dataset (Burri et al., 2016). In a second part, the underwater evaluation is performed on the three new underwater datasets introduced in Section 4. The *EuRoC MH* sequences are captured in the same industrial environment with the same lighting conditions and are provided with a ground truth position from a Leica Total Station. Similarly to (Schmuck and Chli, 2019), sequences `MH_02` and `MH_03` were used to simulate two agents. The agents start from a close position, but do not overlap much after takeoff, each exploring a different part of the hall. An overview of this dataset is given in Figure 10, and a brief summary in Table 4 and the corresponding trajectories per agent in Figure 11.

5.3. Evaluation criteria

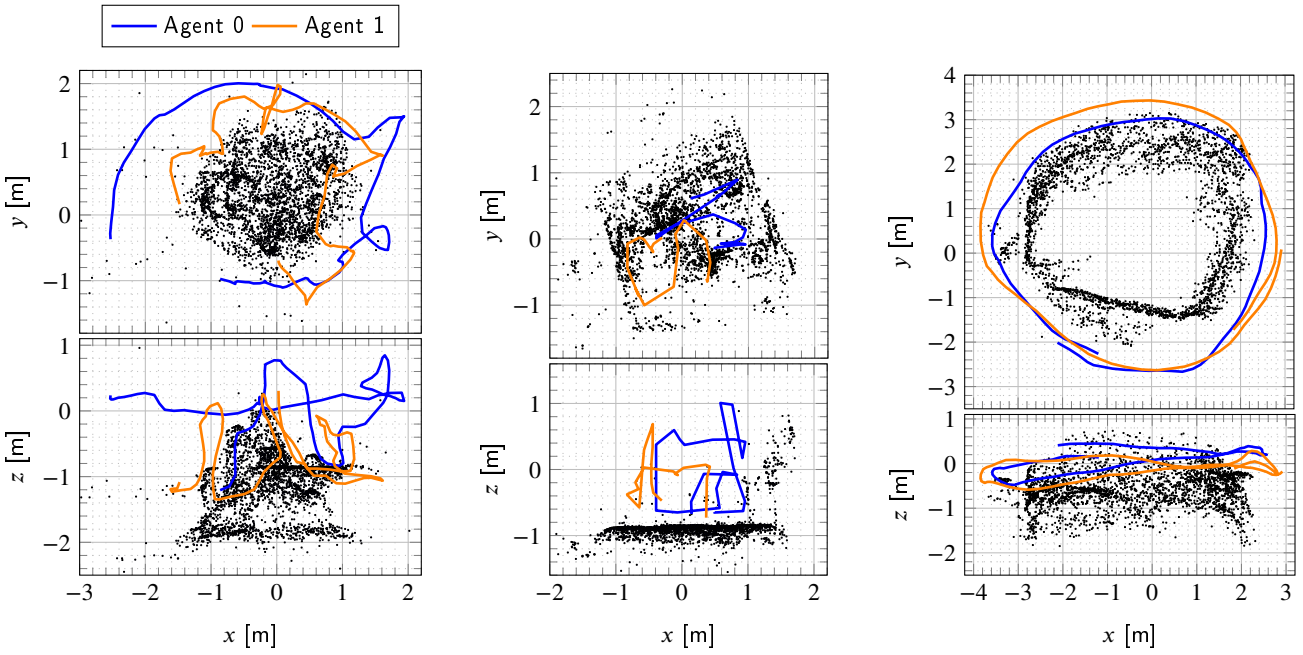
Two evaluations are conducted: localization and mapping performances and real-time performances. In related multi-agent VSLAM works (Daoud et al., 2018; Schmuck and Chli, 2019), localization and mapping performances are only evaluated by the RMS Absolute Position Error (APE) on each agent's trajectory. However, multi-agent SLAM also aims to estimate inter-agent relative poses and their collaborative environment mapping. This is why additional comparison metrics are introduced here.

Individual localization is evaluated by the RMS APE and the RMS Relative Position Error (RPE) between two consecutive frames and the percentage of localization failure with respect to the number of frames.

Relative localization is characterized by the percentage of the sequence for which the agents are localized on the same map, and by the RMS Absolute Relative Position Error

Table 2
Underwater multi-agent datasets description. All of them are recorded with an RGB camera and between 1 and 5 m depth, without embedded lights. Trajectory span is characterized by the volumetric and horizontal coverage of the sequence, abbreviated 'vol.' and 'horiz.,' respectively.

	Camera	Duration	Trajectory span	Description
Tank 1	480×640 pixels 10 Hz	100 s	vol.: 70 m^3 horiz.: 20 m^2	Textured fake reef in pool. Slow motion.
Tank 2	480×640 pixels 5 Hz	75 s	vol.: 48 m^3 horiz.: 16 m^2	Submarine spare parts, in pool. Some poorly textured areas (walls, floor). Fast motion and motion blur. Agent 1's sequence easier than Agent 0's.
Sea diving	380×640 pixels 8 Hz	100 s	vol.: 189 m^3 horiz.: 52 m^2	Around a rock, at sea, at shallow depth, in clear water. Presence of suspended particles, fishes and a fellow diver.



(a) *Tank 1* (b) *Tank 2* (c) *Sea diving*

Figure 9: Agent trajectories on sequences. The black points in Fig. 9a, 9b, and 9c are a sub-sample of the SfM reconstruction and give an idea of the position of the surrounding objects with respect to the agents' trajectories.

(ARPE) between the two agents. If the system fails to merge all maps into a single one, these metrics are given both as a statistic over all maps and for the map with the largest number of KFs, denoted as the *main map*.

Collaborative mapping is first evaluated by the number N_{maps} of unmerged maps at the end of the sequence. Additional metrics are computed on the main map. Agent contribution and map size are characterized by the number of KFs created by each agent. The map's spatial cover L_{map} is computed as the sum of the edges of the minimum spanning tree among the positions of the KF in the map, and evaluates the size of the mapped area. L_{map} is computed using the ground truth pose of the KF to avoid introducing a bias due to poor estimation of the KF pose. The RMS APE on the pose of the KFs of the main map is also computed.

Given an error metric e on a set of N poses $\mathbf{T}_j \in SE(3)$ defined on n disconnected maps denoted $\mathcal{M}_i, i \in \{0...n-1\}$, the global RMS associated to e is defined by:

$$e_{RMS} = \sqrt{\frac{1}{N}\sum_{i=1}^{n}\sum_{\mathbf{T}_j \in \mathcal{M}_i} e(\mathbf{T}_j)^2} \quad (1)$$

where $\mathbf{T}_j \in \mathcal{M}_i$ denotes that pose $\mathbf{T}_j$ is defined in map $\mathcal{M}_i$.

In order to compare the real time performances of the algorithms evaluated, the durations of the tracking (T), local mapping (LM) and place recognition (PR) operations are recorded. Place recognition includes inter and intra map

Table 3
Competing algorithms. Unless otherwise stated, tracking (T), local mapping (LM), loop closing (LC), place recognition (PR), and GBA operations are the same as the one implemented in ORB-SLAM. Map merging is abbreviated into MM.

	Agents	Server	Map merging	Recovery
MAM³SLAM	Send frames to the server	1 T thread per agent 1 LM thread per agent 1 LC & MM thread (1 LC GBA thread)	Similar to ORB-SLAM3's MM	Multi-map
ORB-SLAMM (Daoud et al., 2018)	Send frames to the server	1 T thread per agent 1 LM thread per agent 1 LC thread per agent (1 LC GBA thread per agent) 1 inter-map PR thread	Close to ORB-SLAM's LC but with less geometric consistency checks	Multi-map
CCM-SLAM (Schmuck and Chli, 2019)	1 T thread 1 LM thread + send new KF to the server	1 inter-map PR thread (1 MM GBA thread) + send local KF to the agents	Similar to ORB-SLAM's LC + GBA	∅

(a) (b) (c)

Figure 10: Overview of the *EuRoC MH* dataset

loop detection, loop fusion and map merging. Let t_{op}, $op \in \{T, LM, PR\}$ denote the total duration of the op operation summed for all iterations for all agents on a sequence. Let $N_i, i \in \mathbb{N}$ define the total number of frames of Agent i during the same sequence. Since the different datasets evaluated have different durations and framerates, the run time of each operation op is characterized by the quantity:

$$\overline{t_{op}} = \frac{t_{op}}{\sum_i N_i} \tag{2}$$

Because the tracking operation is triggered by a new incoming frame, $\overline{t_T}$ is also the average duration of a tracking operation.

6. Evaluation

A quantitative evaluation of MAM³SLAM is performed on two-agent scenarios on the four datasets described in Section 5.2, including one aerial dataset and three underwater datasets, by comparing MAM³SLAM to multi-agent ORB-SLAMM (Daoud et al., 2018) and to CCM-SLAM (Schmuck and Chli, 2019). These two competing works have been selected for the current evaluation because they are the most recent multi-agent ORB-SLAM-based work offering a fully functional open-source implementation and not requiring NVIDIA GPU acceleration (NVIDIA et al., 2020). In addition, ORB-SLAM3 (Campos et al., 2021) is used as a reference

Table 4
EuRoC MH dataset description. Trajectory span is characterized by the volumetric and horizontal coverage of the sequence, abbreviated 'vol.' and 'horiz.' respectively.

	Camera	Duration	Trajectory span	Description
EuRoC MH (Burri et al., 2016)	Grayscale 480×752 pixels 20 Hz	135 s	vol.: 288 m^3 horiz.: 144 m^2	Flying drone in an industrial hall.

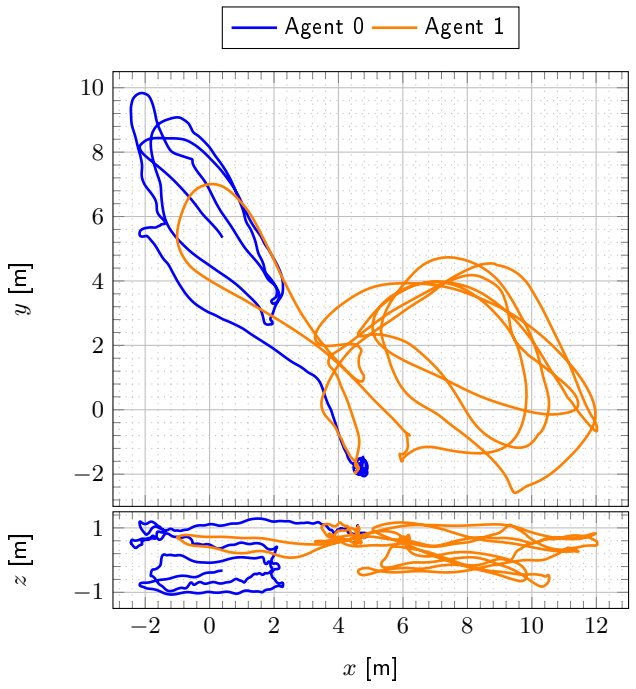


Figure 11: Agent trajectories on the *EuRoC MH* dataset

for individual localization performance, as it represents the current state-of-the-art for monocular single agent VSLAM. Evaluations are carried out in real-time on a computer with an Intel i7-10610U CPU @ 1.80GHz × 8, 16 GB RAM, running Ubuntu 18.04 and ROS Melodic.

6.1. Localization and mapping evaluation

A localization and mapping performance evaluation is performed according to the metrics described in Section 5.3. Given the non-deterministic behavior of multithreaded applications, all localization and mapping evaluation metrics are given as their median values among five runs. The results are reported in Tables 5, 6, 7 and 8. If there are more than one map in the system, the RMS APE and RPE on the main map are indicated in parentheses in the table. Since ORB-SLAMM does not completely fuse the maps in the system but only computes an alignment, the total number of maps in the system is shown in parentheses in the table. Estimated and reference trajectories are aligned using the $Sim(3)$ Umeyama alignment of the EVO library (Grupp, 2017) to compute the error metrics. In Tables 5, 6, 7 and 8, the best result for each evaluation criterion among the multi-agent SLAM works is shown in bold. If ORB-SLAM3 (Campos et al., 2021) gives a better result than the multi-agent approaches, ORB-SLAM3's result is also indicated in bold. A graphical, synthetic comparison of the performances of the three multi-agent SLAM works is given in Figure 12. More statistics on the position errors for the multi-agent SLAM algorithms are provided in Figure 16, in the form of box-and-whisker diagrams.

Figure 13 shows the trajectories estimated by MAM³SLAM on one of the five runs, for the four datasets. Estimated and reference trajectories overlap particularly well for the *EuRoC MH*, *Tank 1* and *Sea diving* datasets.

CCM-SLAM fails to initialize on the *Tank 1* and *Tank 2* datasets, denoting a lack of robustness. We assume that their initialization solver may not have time to converge due to the number of features, their layout, or their characteristics.

MAM³SLAM demonstrates the best individual localization performances among the multi-agent approaches evaluated, with the lowest RMS APE and RPE in all sequences, reaching centimetric accuracy on the *EuRoC MH* and *Tank 1* datasets. CCM-SLAM's performances are close to MAM³SLAMs but still with a lower accuracy, showing up to ten times higher RMS APE and RPE on *EuRoC MH*'s Agent 1 and *Sea diving*'s both agents. ORB-SLAMM shows particularly poor individual localization performances, with RMS APE and RPE exceeding one meter in *EuRoC MH* and RMS APE 50 times higher than all other approaches on *Sea diving*'s Agent 1. This is mainly due to bad inter-map transformation estimation caused by the lack of robustness of their inter-map place recognition and map merging algorithm compared to the other approaches. An example of ORB-SLAMM map misalignment is provided in Figure 14, where trajectories estimated by ORB-SLAMM are aligned with respect to the reference trajectories using Umeyama alignment. In Figure 14a, trajectories estimated on different maps are aligned together using the inter-map transformation estimated by ORB-SLAMM, and the resulting trajectory is aligned with respect to the reference ones. In Figure 14b, trajectories estimated on different maps are aligned individually with respect to the reference, without using the inter-map transformation estimate from ORB-SLAMM. It can be observed that ORB-SLAMM produces inconsistent inter-map alignments, which deteriorate significantly the accuracy of the global trajectory. The same phenomenon happens on all datasets.

One can notice that even if MAM³SLAM and CCM-SLAM show similar performances, ORB-SLAM3 produces the most accurate individual localization on the *EuRoC MH*, *Tank 1*, and *Sea diving* datasets. This can be explained by the computing resources available since more parallel computations are required by multi-agent approaches. This can also explain the non-zero failure rates observed for multi-agent approaches on these three datasets. However, MAM³SLAM outperforms ORB-SLAM3 in individual localization on *Tank 2*, highlighting the interest in collaborative map construction and map sharing for particularly difficult sequences. In this dataset, Agent 1's sequence is easier than Agent 0's. As a result, ORB-SLAM3 outputs two times higher RMS APE and RPE on Agent 0 than on Agent 1. However, in MAM³SLAM, Agent 0's localization relies on more complete and reliable mapping data through the collaborative scene mapping. This results in agent performance smoothing, reaching a similar accuracy of about 10 cm for both agents.

Table 5
Results on the *EuRoC MH* dataset

		MAM³SLAM	ORB-SLAMM	CCM-SLAM	*ORB-SLAM3*
#0	RMSAPE (m)	**0.021**	1.237	0.055	***0.016***
	RMSRPE (m)	0.015	0.147	**0.008**	***0.005***
	failure rate (%)	0.72	**0.00**	**0.00**	***0.00***
#1	RMSAPE (m)	**0.034 (0.033)**	2.624	0.152	***0.027***
	RMSRPE (m)	**0.011 (0.011)**	1.436	0.025	***0.010***
	failure rate (%)	1.55	1.48	**0.00**	***0.00***
% same map		99.38	**100.00**	**100.00**	Ø
RMSARPE (m)		**0.025**	2.162	0.135	Ø
N_{maps}		2	**1 (4)**	**1**	Ø
KF per agent (#0,#1)		243, 249	306, 479	281, 360	Ø
L_{map} (m)		135.292	**150.224**	145.347	Ø
KF RMSAPE (m)		**0.031**	2.500	0.159	Ø

Table 6
Results on the *Tank 1* dataset

		MAM³SLAM	ORB-SLAMM	CCM-SLAM	*ORB-SLAM3*
#0	RMSAPE (m)	**0.026**	0.489	Ø	***0.012***
	RMSRPE (m)	**0.026**	0.461	Ø	*0.038*
	failure rate (%)	**0.00**	**0.00**	100.00	***0.00***
#1	RMSAPE (m)	**0.037**	0.208	Ø	***0.011***
	RMSRPE (m)	**0.038**	0.213	Ø	***0.030***
	failure rate (%)	**0.00**	3.16	100.00	***0.00***
% same map		**100.00**	**100.00**	Ø	Ø
RMSARPE (m)		**0.023**	0.389	Ø	Ø
N_{maps}		**1**	**1 (5)**	Ø	Ø
KF per agent (#0,#1)		101, 128	89, 125	Ø	Ø
L_{map} (m)		**32.086**	29.366	Ø	Ø
KF RMSAPE (m)		**0.012**	0.509	Ø	Ø

MAM³SLAM also outperforms the other evaluated methods on inter-agent relative localization. All methods localize the two agents on the same map in almost 100% of the *EuRoC MH* and *Tank 1* datasets, but MAM³SLAM outperforms the other approaches on relative pose estimation with centimetric precision. It is also the only SLAM approach capable of localizing the two agents on the same map for some periods of time on the *Tank 2* dataset, with a fair 20 cm accuracy. Finally, on the *Sea diving* dataset, MAM³SLAM and CCM-SLAM localize the agents in the same map 100 % of the sequence and with a similar relative localization accuracy, whereas ORB-SLAMM fails to fuse all maps of the system, resulting in a very small sequence percentage for which a relative localization can be computed.

Considering collaborative mapping, ORB-SLAMM produces inaccurate map alignments and sometimes fails to detect map overlap, as with the *Tank 2* and the *Sea diving* datasets. Map merging is far more robust in MAM³SLAM and CCM-SLAM, which output a common, single map in most sequences. It is, however, worth noticing that CCM-SLAM's agents cannot initialize a new map, limiting the maximum number of maps in the system to two agents. One can see that MAM³SLAM produces two maps on the *EuRoC MH* dataset but localizes the agents on the same map during almost all the sequence. This is explained by the failure of one of the agents to track the scene soon after initialization and its relocalization in the map initialized by the other agent, which is incremented collaboratively along the sequence. The second map is thus very small, with a median size of 3 KF. Finally, in the general case, MAM³SLAM produces a more accurate main map than the other approaches for an equivalent number of KF per agent and map size L_{map}.

6.2. Discussion on ORB-SLAM3 localization accuracy

As mentioned in Section 6.1, ORB-SLAM3 shows slightly better individual localization accuracy on the *EuRoC MH*, *Tank 1*, and *Sea diving* datasets. The purpose of this section is to discuss these results in more details.

Table 7
Results on the *Tank 2* dataset

		MAM³SLAM	ORB-SLAMM	CCM-SLAM	*ORB-SLAM3*
#0	RMSAPE (m)	**0.125 (0.142)**	0.217 (0.327)	ø	*0.204 (0.241)*
	RMSRPE (m)	0.139 **(0.152)**	**0.127** (0.158)	ø	*0.143 (0.159)*
	failure rate (%)	**27.78**	33.33	100.00	***24.18***
#1	RMSAPE (m)	**0.115 (0.026)**	0.286 (0.174)	ø	***0.074 (0.022)***
	RMSRPE (m)	**0.043 (0.021)**	0.099 (0.066)	ø	*0.050 (0.025)*
	failure rate (%)	23.08	**18.85**	100.00	*26.28*
% same map		**13.92**	0.00	ø	ø
RMSARPE (m)		**0.204**	ø	ø	ø
N_{maps}		**4**	6 (6)	ø	ø
KF per agent (#0,#1)		74, 30	41, 0	ø	ø
L_{map} (m)		**8.412**	2.855	ø	ø
KF RMSAPE (m)		**0.107**	0.327	ø	ø

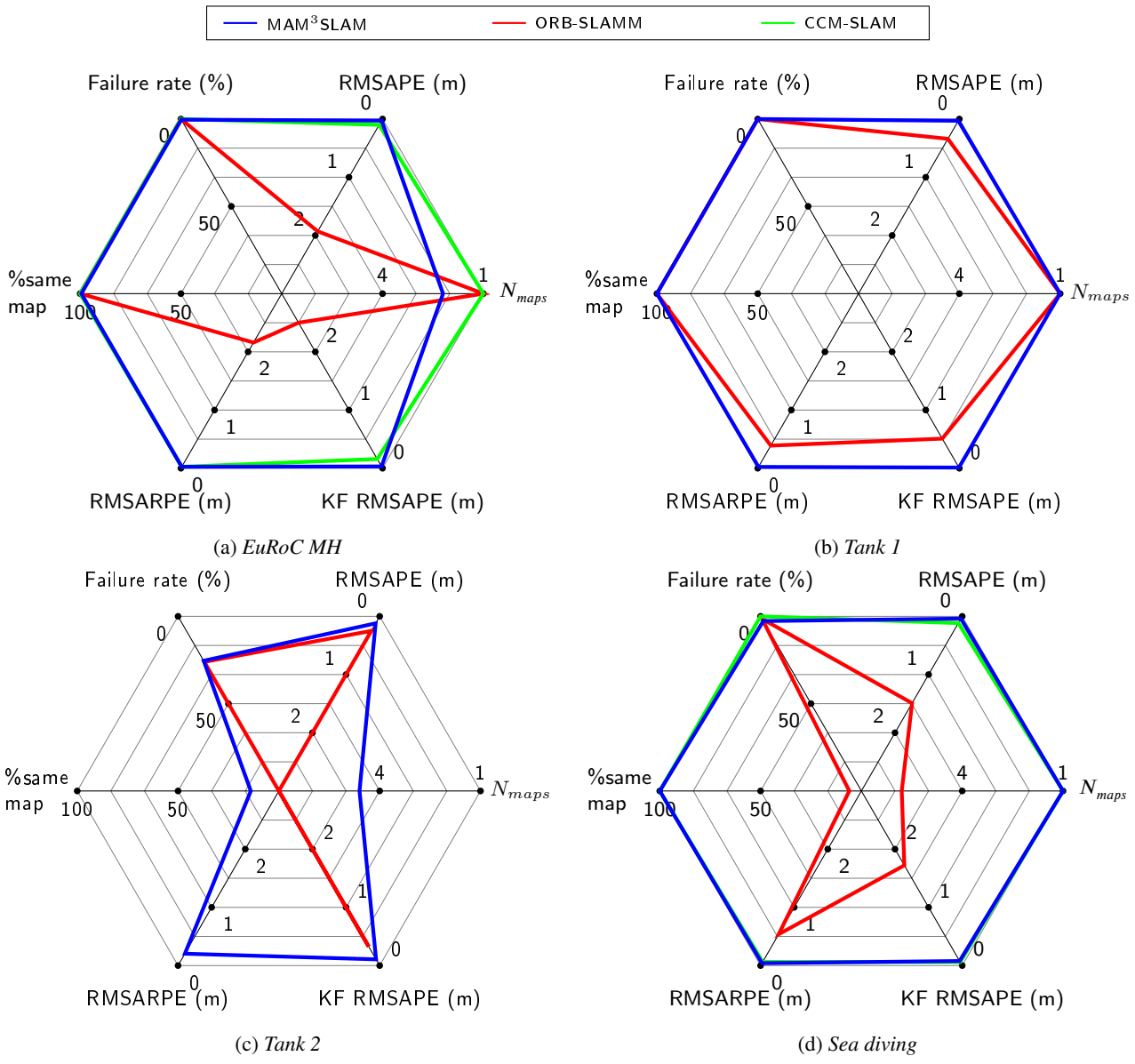
Table 8
Results on the *Sea diving* dataset

		MAM³SLAM	ORB-SLAMM	CCM-SLAM	*ORB-SLAM3*
#0	RMSAPE (m)	**0.040**	0.722 (0.601)	0.046	***0.032***
	RMSRPE (m)	**0.042**	0.325 (0.168)	0.186	***0.041***
	failure rate (%)	5.41	0.75	**0.00**	***0.00***
#1	RMSAPE (m)	0.039	1.492 (1.770)	**0.034**	***0.026***
	RMSRPE (m)	**0.051**	0.139 (0.155)	0.179	*0.049*
	failure rate (%)	**0.00**	1.88	**0.00**	***0.00***
% same map		**100.00**	6.00	**100.00**	ø
RMSARPE (m)		**0.041**	0.538	0.062	ø
N_{maps}		**1**	5 (6)	**1**	ø
KF per agent (#0,#1)		269, 285	21, 187	199, 202	ø
L_{map} (m)		**49.051**	20.132	44.706	ø
KF RMSAPE (m)		0.078	1.722	**0.061**	ø

On the one hand, one can expect that using multiple maps from different agents will result in better accuracy since it will bring additional mapping information and a wider variety of viewing poses. On the other hand, running one instance of single-agent ORB-SLAM3 requires less computational resources than running any of the compared multi-agent SLAM works on a 2-agent sequence. As a result, the computational resources available for agent-level operations have decreased. Because the system is bound to real-time by the image input, the SLAM threads are intended to make more computations in the same limited time. Although the difference in processing time available for agent-level operations is minor, it is still sufficient to have the multi-agent SLAM outputs slightly less optimized than the ones of a single-agent ORB-SLAM3.

The two phenomena depicted have opposite effects on the result's accuracy: the first tends to increase it, and the second tends to decrease it. In the EuRoC MH, *Tank 1*, and *Sea diving* datasets, the second phenomenon is predominant for different reasons. In EuRoC MH, the agents split into two different areas of the hall after a few seconds. Their respective mapping information, therefore, extends the map but does not provide much observational redundancy that would improve map accuracy. On the other hand, *Tank 1* and *Sea diving* feature quite slow motion and mostly easy visual conditions, allowing an already good reconstruction without fusing observations from the two agents, and combining agent information does not improve the map sufficiently to impact trajectory estimation accuracy. Even though localization and mapping accuracy are not improved in these datasets by fusing views from multiple agents, one can still notice that CCM-SLAM and MAM³SLAM accuracy is still very close to that of ORB-SLAM3 in these datasets.

Conversely, the second depicted phenomenon becomes predominant in the *Tank 2* dataset. Because Agent 1's motion is very fast, with a low frame rate, motion blur, and low textures, a single-agent SLAM using only Agent 1's data will have limited accuracy. This can be observed by the result given by ORB-SLAM3 for this agent in this sequence. However, because Agent 0 has simpler motion, its camera input is better suited for computing an accurate map and trajectory. When running a multi-agent SLAM, map data



(a) *EuRoC MH* (b) *Tank 1*

(c) *Tank 2* (d) *Sea diving*

Figure 12: Graphical overview of the performances of the three multi-agent methods compared, on the four test datasets, according to the quantities reported in Tables 5, 6, 7, and 8. The RMSAPE and failure rate reported in the charts correspond to the mean value among the two agents. Note that because CCM-SLAM fails to initialize on the *Tank 1* and *Tank 2*, no results are indicated in Figures 12b and 12c, accounting for a zero performamce. Note also that the results of MAM³SLAM and CCM-SLAM are overlapping in Figure 12d

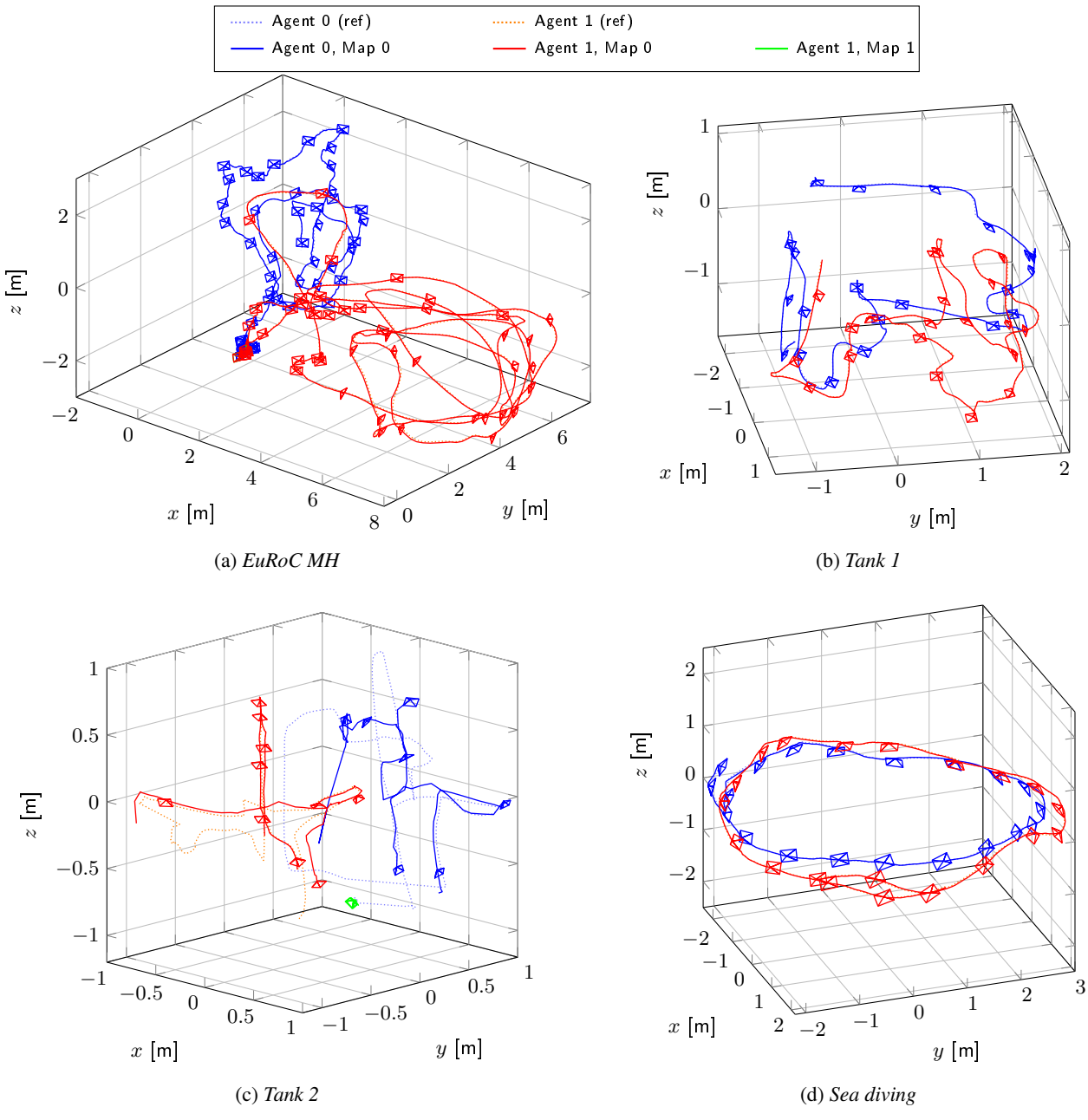
created by Agent 0 can be used in Agent 1's localization, significantly improving its positioning accuracy.

Lastly, but importantly, ORB-SLAM3 does not allow locating multiple agents in the same referential frame at the same time, contrary to the multi-agent works studied. However, getting this relative localization information is our main motivation for focusing on multi-agent SLAM.

6.3. Computing time evaluation

In order to compare the real time performances of the algorithms evaluated, the durations of the tracking (T), local mapping (LM) and place recognition (PR) operations are recorded. In the current evaluation, we group under the name "Place recognition" inter- and intra-map loop detection *via* the database queries followed by consistency and geometrical checks, loop fusion and map merging operations when they occur, including pose graph optimization. The total duration of each of these operations is summed for all iterations for all agents. Figure 15 compares the total computational time dedicated to T, LM, and PR operations in processing each one of the test datasets. Since these datasets have different framerates and duration, the durations indicated in the figure correspond to the sum of all iteration durations divided by the number of frames in the sequence.

One can see that tracking durations are quite similar



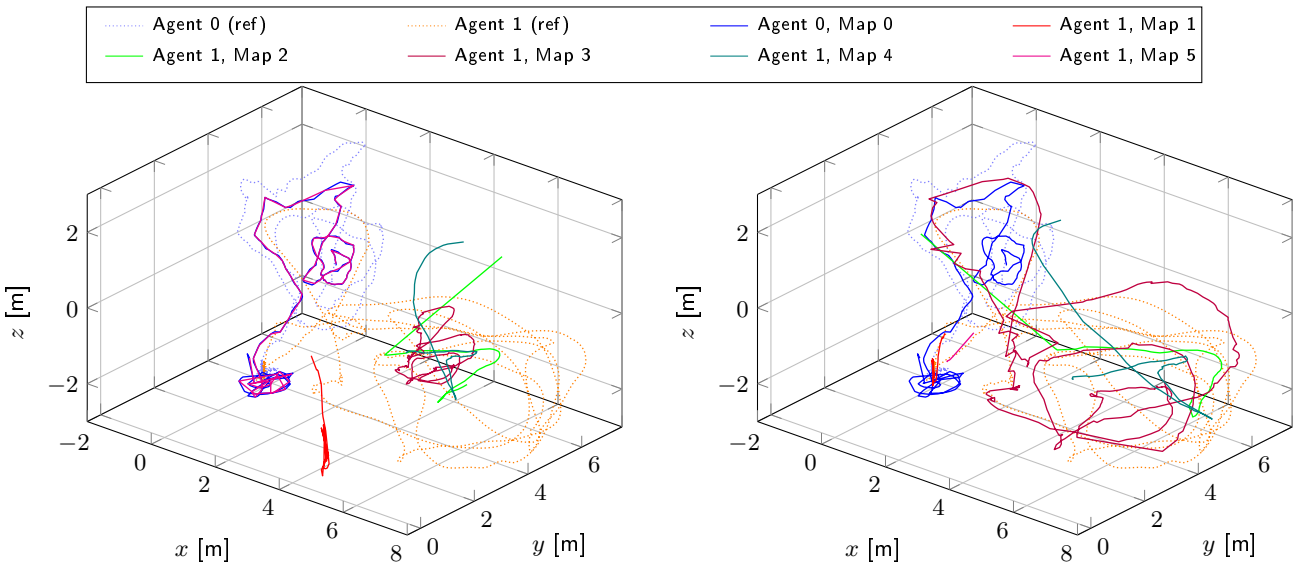
(a) *EuRoC MH* (b) *Tank 1*

(c) *Tank 2* (d) *Sea diving*

Figure 13: Trajectories estimated by MAM³SLAM (solid) and reference trajectories (dashed) on the test datasets, after Umeyama alignment, showing estimated camera poses every 5s.

from an algorithm to another. For the *EuRoC MH*, *Tank 2*, and *Sea diving* datasets, tracking duration represents about 10 ms per frame per agent. The tracking duration per frame is much higher for the *Tank 1* dataset, which can be explained by the high number of ORB features tracked (5000 ORB features per frame are extracted, *vs* 1000 for *EuRoC MH*, and *Sea diving*). 5000 ORB features per frame are extracted for the *Tank 2* dataset too, but most of them are not matched correctly and thus unused for pose computation, hence having a lower computational duration.

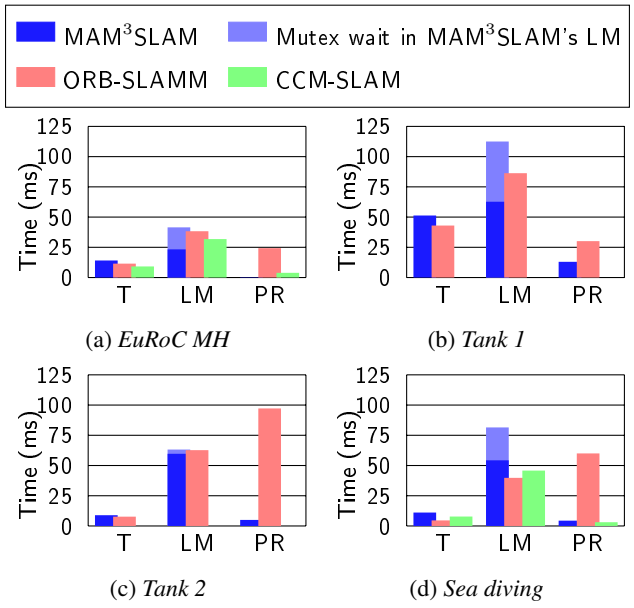
LM total computational duration is higher for MAM³SLAM. One can see that a significant proportion of this duration is spent waiting for mutex unlock. In the evaluated configurations, the maximum duration of local mapping operations divided by the number of incoming frames is about 110 ms. This indicates that, in this case, the system can create a new KF every 9 s, which is still a satisfying ratio. However, this maximum KF insertion rate will decrease as the number of agents increases since more agents will increase the total mutex wait duration. Our system is thus limited to a small number of agents. One can notice that the total duration of mutex wait in local mapping is very low for the *Tank 2* dataset. This is due to the agents being located on the same map simultaneously more rarely and the low framerate reducing the probability



(a) Using ORB-SLAMM's map alignment

(b) Without using ORB-SLAMM's map alignment

Figure 14: Trajectories estimated by ORB-SLAMM on the *EuRoC MH* dataset (plain) and ground truth trajectories (dashed). Trajectories are estimated over 5 maps, which are aligned by the system. Parts of the trajectory from different maps are represented in a different color.



(a) *EuRoC MH*

(b) *Tank 1*

(c) *Tank 2*

(d) *Sea diving*

Figure 15: Sum of the durations of tracking, local mapping and place recognition (loop closure and map overlap detection and fusion) iterations divided by the total number of frames, for each test dataset.

of a simultaneous KF creation attempt by the two agents.

Lastly, two main observations can be formulated regarding PR total computational duration $\overline{t_{PR}}$. First, one can observe that this quantity is significantly higher for ORB-SLAMM than for CCM-SLAM and MAM³SLAM. Indeed, ORB-SLAMM's inter-map place recognition consists of a loop over all KF from all maps, which is repeated continuously, while CCM-SLAM and MAM³SLAM only query the KF database at new KF creation, which prevents unnecessary queries if the maps and keyframes are idle. Second, $\overline{t_{PR}}$ depends on the sequence for each SLAM algorithm. First, Let us consider the cases of MAM3SLAM and CCM-SLAM, where place recognition is triggered by inserting a new KF by any of the involved agents. If a map merging or loop closing candidate is confirmed, the map will be updated and optimized. The duration of the initial database query is negligible compared to that of this correction step. The value of $\overline{t_{PR}}$ thus reflects the ratio of the number of map merging or loop closing operations compared to the number of frames, which depends, notably, on camera frame rate and trajectory. In the case of ORB-SLAMM, place recognition is not triggered by KF insertion but is an infinite loop over all Kfs from all maps. Thus, the complexity of a single iteration of this infinite loop is in $o({n_{KF}}^N)$ where n_{KF} accounts for the number of KF in a map, and N is the number of maps. $\overline{t_{PR}}$ is thus significantly higher in *Tank 2* and *Sea diving*, where there are five to six maps in the system. One can also notice that ORB-SLAMM's $\overline{t_{PR}}$ is higher in *Tank 2* than in *Sea diving*. We believe this is caused by ORB-SLAMM computing one map alignment in *Tank 2* versus 0 in *Sea diving*.

7. Conclusion and future work

This work focused on the problem of visual-based simultaneous localization and mapping for underwater multi-agent scenarios, which require the localization of the involved vehicles with respect to each other and a previously unknown environment. Following previous studies of visual

SLAM underwater capabilities, the current work focuses on ORB-SLAM-based works. MAM^{3}SLAM, a new fully centralized multi-agent and multi-map monocular VSLAM framework based on ORB-SLAM Atlas (Elvira et al., 2019) and ORB-SLAM3 (Campos et al., 2021), was introduced as a solution for underwater multi-ROV localization. In addition, the two most recent state-of-the-art multi-agent ORB-SLAM-based-works released, namely CCM-SLAM (Schmuck and Chli, 2019) and ORB-SLAMM (Daoud et al., 2018), were selected to perform a comparative benchmark and study the potential of ORB-SLAM-based multi-agent works for underwater. In the absence of previous underwater datasets for multi-agent visual SLAM benchmarking, a key contribution of the current work was to record and release three underwater two-agent datasets, including two in a pool and one in the sea. MAM^{3}SLAM, CCM-SLAM, and ORB-SLAMM were then evaluated on four two-agent scenarios, including a standard aerial dataset and our three new underwater datasets.

A run-time analysis demonstrated that MAM^{3}SLAM's place recognition computation time is significantly lower than competing works, and its tracking computational time is of the same order of magnitude, up to 10 times lower than CCM-SLAM's on the *EuRoC MH* dataset and about 100 times lower than ORB-SLAMM's in the general case. This analysis also shows that MAM^{3}SLAM's local mapping takes significantly more time than that of competing approaches — up to 1.4 more time in the worst case — because of the mutex wait imposed to prevent concurrent map updates from different agents when they localize with respect to the same map, and some low-level software engineering would be required to extend MAM^{3}SLAM to more than two agents in future works. This extension would be a prerequisite for using MAM^{3}SLAM in a real-life underwater robot chain configuration.

Even though the current implementation did not focus on handling this scalability problem and left it for future works, our two-agent MAM^{3}SLAM proved to outperform the two other evaluated approaches in terms of individual localization, relative localization, and mapping accuracy and demonstrates good robustness to poor visual conditions. Although showing a similar accuracy to CCM-SLAM, up to 2 cm on some of the datasets, MAM^{3}SLAM proves to be significantly more robust by successfully processing all the test sequences, while CCM-SLAM failed to initialize on two out of the four sequences. In addition, MAM^{3}SLAM produced significantly more accurate results than ORB-SLAMM on all sequences, where MAM^{3}SLAM localization errors are up to 80 times lower than ORB-SLAMM's. Finally, MAM^{3}SLAM proves to be a reliable multi-agent VSLAM approach for multi-ROV localization, reaching centimetric accuracy on two out of three underwater evaluation datasets.

In light of these results, MAM^{3}SLAM seems like a promising solution for multiple ROV localization, for instance, in underwater robot chains. Nonetheless, some low-level software developments are required to extend this approach to a larger number of agents while keeping the overall high-level principle. In addition, future works may focus on improving the robustness of the system by fusing multiple inputs to increase its robustness to underwater visual conditions and estimate the scale of the environment. In particular, in the specific context of a robot chain, inter-agent pose constraints might be used at different levels in the SLAM pipeline, including scale estimation, inter-map place recognition, and pose estimation optimization. The system may also be extended to include stereo camera input, fuse inertial, and, possibly, pressure measurements.

8. Acknowledgment

This work is funded by the French Research Ministry and the CARTT of the IUT of Toulon. We would like to thank the CEPHISMER of the French Navy and the IFREMER for their logistical support.

References

Akkaynak, D., Treibitz, T., 2019. Sea-thru: A method for removing water from underwater images, in: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1682–1691. doi:10.1109/CVPR.2019.00178.

Akkaynak, D., Treibitz, T., Shlesinger, T., Loya, Y., Tamir, R., Iluz, D., 2017. What is the space of attenuation coefficients in underwater computer vision?, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 568–577. doi:10.1109/CVPR.2017.68.

Boittiaux, C., Dune, C., Ferrera, M., Arnaubec, A., Marxer, R., Matabos, M., Audenhaege, L.V., Hugel, V., 2023a. Eiffel tower: A deep-sea underwater dataset for long-term visual localization. The International Journal of Robotics Research 42, 689–699. URL: https://doi.org/10.1177/02783649231177322, doi:10.1177/02783649231177322, arXiv:https://doi.org/10.1177/02783649231177322.

Boittiaux, C., Marxer, R., Dune, C., Arnaubec, A., Ferrera, M., Hugel, V., 2023b. SUCRe: Leveraging scene structure for underwater color restoration. arXiv:2212.09129.

Burri, M., Nikolic, J., Gohl, P., Schneider, T., Rehder, J., Omari, S., Achtelik, M.W., Siegwart, R., 2016. The EuRoC micro aerial vehicle datasets. The International Journal of Robotics Research 35, 1157–1163. URL: https://doi.org/10.1177/0278364915620033, doi:10.1177/0278364915620033, arXiv:https://doi.org/10.1177/0278364915620033.

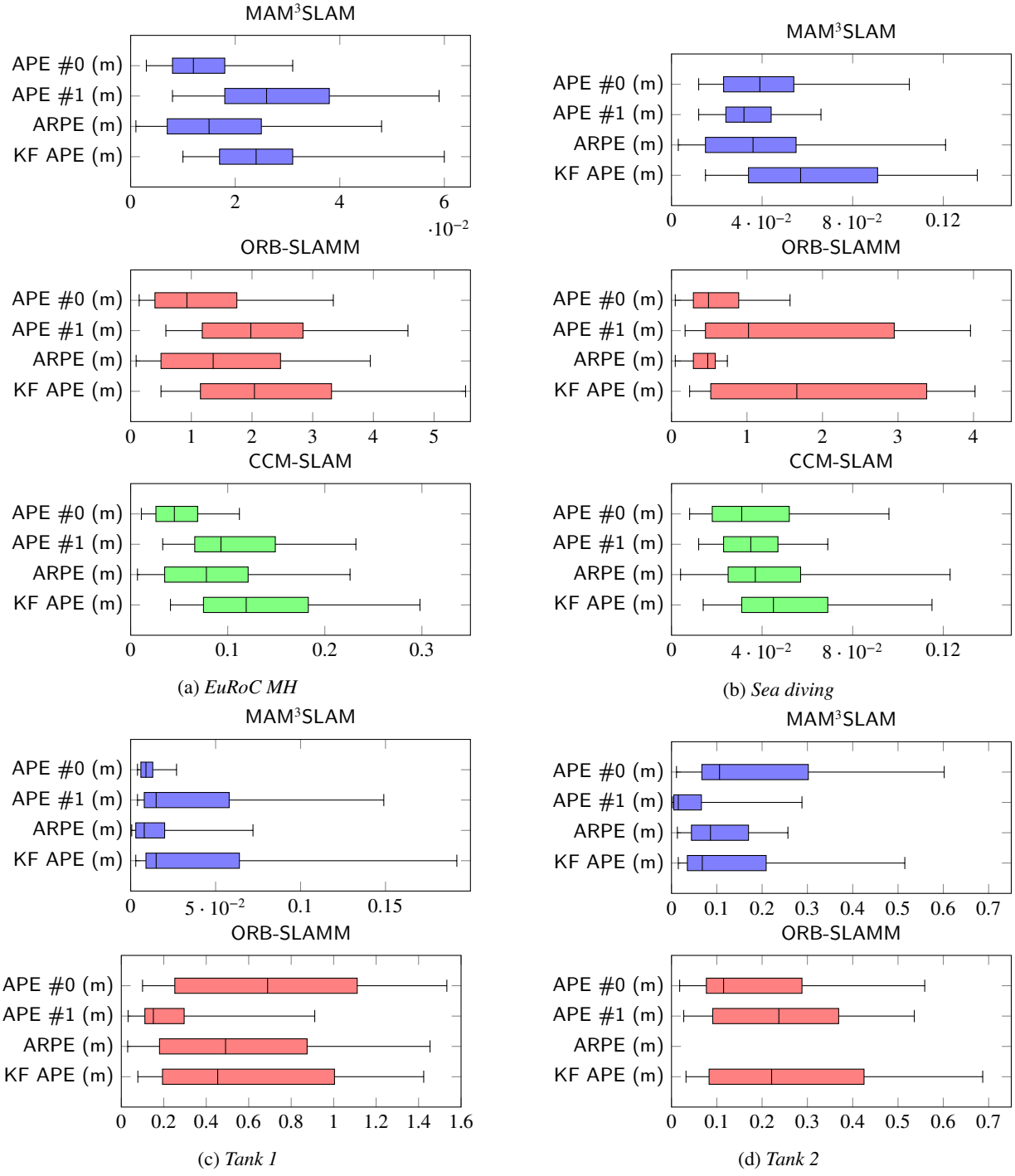
Campos, C., Elvira, R., Rodríguez, J.J.G., M. Montiel, J.M., D. Tardós, J., 2021. ORB-SLAM3: An accurate open-source library for visual, visual–inertial, and multimap SLAM. IEEE Transactions on Robotics 37, 1874–1890. doi:10.1109/TRO.2021.3075644.

Cao, H., Xu, J., Yang, Z., Shangguan, L., Zhang, J., He, X., Liu, Y., 2023. Scaling up edge-assisted real-time collaborative visual slam applications. IEEE/ACM Transactions on Networking , 1–16URL: http://dx.doi.org/10.1109/TNET.2023.3330763, doi:10.1109/tnet.2023.3330763.

Cho, Y., Kim, A., 2018. Channel invariant online visibility enhancement for visual SLAM in a turbid environment. Journal of Field Robotics 35, 1080–1100. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21796, doi:https://doi.org/10.1002/rob.21796.

Cieslewski, T., Choudhary, S., Scaramuzza, D., 2018. Data-efficient decentralized visual slam, in: 2018 IEEE International Conference on Robotics and Automation (ICRA), pp. 2466–2473. doi:10.1109/ICRA.2018.8461155.

Daoud, H.A., Sabri, A.Q.M., Loo, C.K., Mansoor, A.M., 2018. SLAMM: Visual monocular SLAM with continuous mapping using multiple maps.



(a) *EuRoC MH*

(b) *Sea diving*

(c) *Tank 1*

(d) *Tank 2*

Figure 16: Position error box-and-whisker diagrams for the three compared multi-agent VSLAM algorithms on the four datasets.The rectangular box extends from the lower quartile to the upper quartile of the distribution. The median value is indicated by a plain line inside the box. The whiskers indicate the 5 percentile and 95 percentile. The errors analyzed are the Absolute Position Error of Agent 0 (APE #0), the Absolute Position Error of Agent 1 (APE #1), the Absolute Relative Position Error (ARPE) between the two agents and the KF Absolute Position Error (KF APE). These statistics include all runs of each algorithm. Note that the scale of the horizontal axis can change significantly from a diagram to another.

PLOS ONE 13, e0195878. URL: https://doi.org/10.1371/journal.pone.0195878, doi:10.1371/journal.pone.0195878.

Drupt, J., Dune, C., Comport, A.I., Hugel, V., 2023. Qualitative evaluation of state-of-the-art DSO and ORB-SLAM-based monocular visual SLAM algorithms for underwater applications, in: OCEANS 2023 - Limerick, pp. 1–7. doi:10.1109/OCEANSLimerick52467.2023.10244636.

Drupt, J., Dune, C., Comport, A.I., Seillier, S., Hugel, V., 2022. Inertial-measurement-based catenary shape estimation of underwater cables for tethered robots, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6867–6872. doi:10.1109/IROS47612.2022.9981980.

Dubois, R., Eudes, A., Frémont, V., 2019. On data sharing strategy for

decentralized collaborative visual-inertial simultaneous localization and mapping, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 2123–2130. doi:10.1109/IROS40897.2019.8967617.

Elvira, R., Tardós, J.D., Montiel, J., 2019. ORBSLAM-Atlas: a robust and accurate multi-map system, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 6253–6259. doi:10.1109/IROS40897.2019.8967572.

Eustice, R., Singh, H., Leonard, J., Walter, M., Ballard, R., 2005. Visually navigating the RMS Titanic with SLAM information filters, in: Proceedings of Robotics: Science and Systems, Cambridge, USA. doi:10.15607/RSS.2005.I.008.

Eustice, R.M., Singh, H., Leonard, J.J., Walter, M.R., 2006. Visually mapping the rms titanic: Conservative covariance estimates for SLAM information filters. The International Journal of Robotics Research 25, 1223–1242. URL: https://doi.org/10.1177/0278364906072512, doi:10.1177/0278364906072512, arXiv:https://doi.org/10.1177/0278364906072512.

Ferrera, M., Creuze, V., Moras, J., Trouvé-Peloux, P., 2019. Aqualoc: An underwater dataset for visual–inertial–pressure localization. The International Journal of Robotics Research 38, 1549–1559. URL: https://doi.org/10.1177/0278364919883346, doi:10.1177/0278364919883346, arXiv:https://doi.org/10.1177/0278364919883346.

Galvez-López, D., Tardos, J.D., 2012. Bags of binary words for fast place recognition in image sequences. IEEE Transactions on Robotics 28, 1188–1197. doi:10.1109/TRO.2012.2197158.

Geiger, A., Lenz, P., Urtasun, R., 2012. Are we ready for autonomous driving? the KITTI vision benchmark suite, in: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3354–3361. doi:10.1109/CVPR.2012.6248074.

Grupp, M., 2017. evo: Python package for the evaluation of odometry and SLAM. https://github.com/MichaelGrupp/evo.

Hidalgo, F., Kahlefendt, C., Bräunl, T., 2018. Monocular ORB-SLAM application in underwater scenarios, in: 2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO), pp. 1–4. doi:10.1109/OCEANSKOBE.2018.8559435.

Huang, H., Lin, W.Y., Liu, S., Zhang, D., Yeung, S.K., 2020. Dual-SLAM: A framework for robust single camera navigation, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 4942–4949. doi:10.1109/IROS45743.2020.9341513.

Joshi, B., Rahman, S., Kalaitzakis, M., Cain, B., Johnson, J., Xanthidis, M., Karapetyan, N., Hernandez, A., Li, A.Q., Vitzilaios, N., Rekleitis, I., 2019. Experimental comparison of open source visual-inertial-based state estimation algorithms in the underwater domain, in: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7227–7233. doi:10.1109/IROS40897.2019.8968049.

Laranjeira, M., Dune, C., Hugel, V., 2020. Catenary-based visual servoing for tether shape control between underwater vehicles. Ocean Engineering 200, 107018. URL: https://www.sciencedirect.com/science/article/pii/S0029801820300949, doi:https://doi.org/10.1016/j.oceaneng.2020.107018.

Leonardos, S., Daniilidis, K., 2017. A game-theoretic approach to robust fusion and kalman filtering under unknown correlations, in: 2017 American Control Conference (ACC), pp. 2568–2573. doi:10.23919/ACC.2017.7963339.

Li, F., Yang, S., Yi, X., Yang, X., 2018. CORB-SLAM: A collaborative visual SLAM system for multiple robots, in: Romdhani, I., Shu, L., Takahiro, H., Zhou, Z., Gordon, T., Zeng, D. (Eds.), Collaborative Computing: Networking, Applications and Worksharing, Springer International Publishing, Cham. pp. 480–490.

Luft, L., Schubert, T., Roumeliotis, S.I., Burgard, W., 2018. Recursive decentralized localization for multi-robot systems with asynchronous pairwise communication. The International Journal of Robotics Research 37, 1152–1167. URL: https://doi.org/10.1177/0278364918760698, doi:10.1177/0278364918760698, arXiv:https://doi.org/10.1177/0278364918760698.

Mangelson, J.G., Vasudevan, R., Eustice, R.M., 2018. Communication constrained trajectory alignment for multi-agent inspection via linear programming, in: OCEANS, MTS/IEEE, Charleston, SC, USA. doi:10.1109/oceans.2018.8604775.

Mur-Artal, R., Montiel, J.M.M., Tardós, J.D., 2015. ORB-SLAM: A versatile and accurate monocular SLAM system. IEEE Transactions on Robotics 31, 1147–1163. doi:10.1109/TRO.2015.2463671.

Mur-Artal, R., Tardós, J.D., 2017. ORB-SLAM2: An open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Transactions on Robotics 33, 1255–1262. doi:10.1109/TRO.2017.2705103.

Murphy, R.R., Dreger, K.L., Newsome, S., Rodocker, J., Slaughter, B., Smith, R., Steimle, E., Kimura, T., Makabe, K., Kon, K., Mizumoto, H., Hatayama, M., Matsuno, F., Tadokoro, S., Kawase, O., 2012. Marine heterogeneous multirobot systems at the great eastern japan tsunami recovery. Journal of Field Robotics 29, 819–831. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21435, doi:https://doi.org/10.1002/rob.21435.

NVIDIA, Vingelmann, P., Fitzek, F.H., 2020. Cuda, release: 10.2.89. URL: https://developer.nvidia.com/cuda-toolkit.

Özkahraman, O., Ögren, P., 2022. Collaborative navigation-aware coverage in feature-poor environments, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 10066–10073. doi:10.1109/IROS47612.2022.9981547.

Quattrini Li, A., Coskun, A., Doherty, S.M., Ghasemlou, S., Jagtap, A.S., Modasshir, M., Rahman, S., Singh, A., Xanthidis, M., O'Kane, J.M., Rekleitis, I., 2017. Experimental comparison of open source vision-based state estimation algorithms, in: Kulić, D., Nakamura, Y., Khatib, O., Venture, G. (Eds.), 2016 International Symposium on Experimental Robotics, Springer International Publishing, Cham. pp. 775–786.

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An efficient alternative to sift or surf, in: 2011 International Conference on Computer Vision, pp. 2564–2571. doi:10.1109/ICCV.2011.6126544.

Salvi, J., Petillot, Y., Batlle, E., 2008. Visual SLAM for 3D large-scale seabed acquisition employing underwater vehicles, in: 2008 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1011–1016. doi:10.1109/IROS.2008.4650627.

Schmuck, P., Chli, M., 2019. CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. Journal of Field Robotics 36, 763–781. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21854, doi:https://doi.org/10.1002/rob.21854.

Schönberger, J.L., Frahm, J.M., 2016. Structure-from-motion revisited, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4104–4113. doi:10.1109/CVPR.2016.445.

Schubert, D., Goll, T., Demmel, N., Usenko, V., Stückler, J., Cremers, D., 2018. The TUM VI benchmark for evaluating visual-inertial odometry, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 1680–1687. doi:10.1109/IROS.2018.8593419.

Song, Z., Mohseni, K., 2014. A distributed localization hierarchy for an AUV swarm, in: 2014 American Control Conference, pp. 4721–4726. doi:10.1109/ACC.2014.6859344.

Wang, Y., Song, W., Fortino, G., Qi, L.Z., Zhang, W., Liotta, A., 2019. An experimental-based review of image enhancement and image restoration methods for underwater imaging. IEEE Access 7, 140233–140251. doi:10.1109/ACCESS.2019.2932130.

Yu, C., Yang, X., Gao, J., Yang, H., Wang, Y., Wu, Y., 2022. Learning efficient multi-agent cooperative visual exploration, in: Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX, Springer-Verlag, Berlin, Heidelberg. p. 497–515. URL: https://doi.org/10.1007/978-3-031-19842-7_29, doi:10.1007/978-3-031-19842-7_29.

Zhang, S., Zhao, S., An, D., Liu, J., Wang, H., Feng, Y., Li, D., Zhao, R., 2022. Visual SLAM for underwater vehicles: A survey. Computer Science Review 46, 100510. URL: https://www.sciencedirect.com/science/article/pii/S1574013722000442, doi:https://doi.org/10.1016/j.cosrev.2022.100510.

Zou, D., Tan, P., 2013. CoSLAM: Collaborative visual SLAM in dynamic environments. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 354–366. doi:10.1109/TPAMI.2012.104.

Zou, D., Tan, P., Yu, W., 2019. Collaborative visual SLAM for multiple agents:a brief survey. Virtual Reality and Intelligent Hardware 1,

461–482. URL: https://www.sciencedirect.com/science/article/pii/S2096579619300634, doi:https://doi.org/10.1016/j.vrih.2019.09.002.
3D Vision.