Human-Robot Collaborative SLAM-XR

Karim Yassine, Malak Sayour, Adam Manasfi, Maya Hachach,
Nadim Dib, Imad H. Elhajj, Boulos Asmar, Elie Khoury, Daniel Asmar

***Abstract*—In this paper, we propose a collaborative centralized 3D mapping and localization framework that harnesses the capabilities of both SLAM (Simultaneous Localization And Mapping) and XR (eXtended Reality). On one hand, our framework allows for integrating local maps generated by a multitude of heterogeneous agents (*e.g.* robots) into a unified map. On the other hand, it allows human intervention at multiple levels: first, humans can inspect and intervene in the mapping process in situ to produce 3D maps, overlay virtual assets, and add annotations, all of which can contribute towards enhanced autonomy and navigation. Second, beyond the mapping aspect, a human can also intervene in the localization task of any collaborating robot by inspecting and correcting its generated paths, and, if necessary, enforcing a desired trajectory. Experiments inside two real settings demonstrated the superiority of the proposed system.**

I. Introduction

The ability of autonomous systems to navigate complex and dynamic environments hinges on their capacity for Simultaneous Localization and Mapping (SLAM). Initially conceived as a method for robotic navigation [1], SLAM has transcended its original confines and has become a useful tool in a myriad of applications [2], [3]. From self-driving cars [4] to digital twins [5], SLAM has proven its mettle in providing machines with the spatial awareness necessary for intelligent decision-making. However, despite these advancements, SLAM systems still struggle with uncertain environments, sensor limitations, and dynamic obstacles, leading to inaccuracies that can impact autonomous decision-making [6].

Similarly, the world of eXtended Reality (XR) has significantly evolved [7], transcending the realm of entertainment to revolutionize diverse sectors [8]. Industries ranging from healthcare [9] and education [10] to manufacturing and design [11] have embraced XR, leveraging their potential to enhance training, visualization, and collaboration [12], [13]. The ability of XR to bridge the gap between the physical and digital worlds has opened avenues for innovative applications that redefine traditional workflows.

Given the complementary strengths of SLAM and XR, their convergence presents a promising avenue for advancing autonomous navigation, interactive mapping, and human-in-the-loop control. By leveraging XR's visualization and interaction capabilities, humans can not only visualize SLAM-generated maps but also refine them—correcting errors, adding semantic information, and guiding robot navigation. This human-robot collaboration is particularly valuable in complex, cluttered, or ambiguous environments, where SLAM alone may fail to capture critical details.

M. Sayour, K. Yassine, A. Manasfi, M. Hachach, N. Dib, I. H. Elhajj, and D. Asmar are with the Vision and Robotics Lab, Maroun Semaan Faculty of Engineering and Architecture, American University of Beirut, 1107 2020, Riad El Solh, Beirut, Lebanon; email: mhy16@mail.aub.edu, ms423@aub.edu.lb, manasfiadam4@gmail.com, mayahashash00@gmail.com ,ngd04@mail.aub.edu, ie05@aub.edu.lb, da20@aub.edu.lb. (K. Yassine and M. Sayour are co-first authors.)

B. Asmar and E. Khoury are with IDEALworks, Riesstraße 22, 80992 München, Germany; email: Boulos.Asmar@idealworks.com, Elie.Khoury@idealworks.com

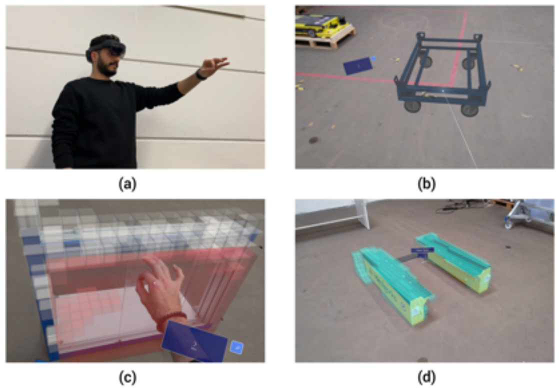


Fig. 1. Overview of the proposed system. (a) a human wearing a Microsoft HoloLens 2; (b) adding an asset into a digital twin; (c) human editing the produced map; (d) automatic labeling of assets.

In this paper, we introduce a modular framework designed to facilitate collaborative mapping and navigation by harnessing the synergies of XR and SLAM. Our proposed system focuses on critical aspects such as map quality enhancement, robot navigation and localization precision, semantic enrichment for enhanced autonomy, and its implications for multi-agent mapping scenarios as shown in Fig. 1. The aim is to create a dynamic digitized version of a navigated environment using the same agents responsible for their respective daily autonomous tasks, while at the same time empowering a human operator with the capability to intervene in the digitization process, and enabling immersive monitoring capabilities.

The proposed approach enables a collaborative environment between humans and robots via XR, allowing heterogeneous agents to participate in the mapping process reducing time and effort. Through this collaborative framework, SLAM inaccuracies arising from sensor noise or reflective and translucent surfaces can be mitigated. Errors in robot ego pose estimates can be examined and corrected. Finally, desired semantic data can be integrated for enhanced autonomous navigation and scene understanding. The key contributions of our paper

Manuscript 1273 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received February 28, 2025.

include:

- A modular multi-agent framework enabling collaborative heterogeneous mapping.
- Enhanced mapping accuracy by leveraging human-robot collaboration.
- Improved localization and navigation due to aided path planning on human-corrected maps, human-assisted localization, and the use of maps from multi-agents.
- Semantic enrichment by incorporating generated semantic information into the mapping process.
- Advanced visualization and editing capabilities, by allowing for predefined CAD insertion of irregular shapes.

The remainder of the paper is structured as follows: Section II provides a concise overview of the relevant literature, Section III introduces our proposed methodology, Section IV discusses the experiments and their results, and Section V serves as the paper's conclusion. The implemented Unity project and the developed ROS workspace are available on GitHub[1].

II. Related Work

Numerous efforts have been made to integrate XR and robotic systems for enhanced human robot interaction [14]–[17] with enhanced robot navigation being one of the most dominant task studied [17]. Kozlov *et al.* [18] suggested the utilization of Augmented Reality (AR) as a tool for debugging, assessing, and refining SLAM algorithms. They contended that AR offers the capability to visualize the internal program state of a robot and pertinent information, specifically within the realm of testing and debugging SLAM algorithms. Stedman *et al.* [19] presented a configurable immersive teleoperation framework based on the RTAB-Map SLAM library. VRTAB was proposed as a real-time immersive analytics methodology to enhance situational awareness and decision-making in teleoperation missions. In addition, Chacko *et al.* [20] proposed an AR-based spatial referencing system named AR-SRS to help non-expert users provide task-specific spatial information to mobile service robots in dynamic domestic environments. Their system allows users to create virtual AR objects with semantic labels, which are mapped to real-world coordinates for accurate task allocation. On the other hand, Mu *et al.* [21] proposed a VR based solution for in-oil corrosion inspection without opening storage tanks. Their solution allows for robot control and inspection monitoring in invisible environments. Similarly, Gu *et al.* [22] propose an approach to visualizing mobile robots through an AR headset when there is no line-of-sight visibility between the robot and the human. Their system utilizes 3D models of the robots and virtual arrows to indicate the robot's planned movements, along with a 2D virtual grid to represent the navigation plane. In addition, as an alternative to keyboards and joysticks, Gai *et al.* [23] proposed a VR-based telerobotic navigation method that synchronizes the movement of a remote robot with the real walk of an operator in a limited space. By integrating natural head and body interactions with real-time VR video feedback, their approach enhances the operator's sense of presence in remote environments, improving teleoperation efficiency.

While VR provides a safe and realistic environment for interaction, it can introduce perceptual biases affecting both humans and robots [24]. Grzeskowiak *et al.* [24]proposed a technical framework integrating motion tracking, VR devices, and robot sensor simulations for enhanced human-robot interactions in navigation tasks. The authors evaluated the impact and importance of VR-induced biases and concluded their work favoring VR in human-robot interaction.

In our previous work [25], we examined the impact of human-assisted SLAM utilizing AR in improving the quality of generated maps. The system proved efficient in increasing the quality of maps generated by one robot agent. Serving as a base for our proposed work, we propose a new architecture allowing for heterogeneous multi-agent collaboration with an enhanced focus on robot navigation and localization within the AR-assisted SLAM framework. The proposed approach serves as a multi-agent framework allowing for collaborative human-assisted mapping and navigation all while including semantic data integration for enhanced autonomy and scene understanding.

III. System Overview

We adopt a centralized approach, where all computations except for mapping are executed at the central unit (Fig. 2). The connectivity framework involves mapping agents linked to the central unit through the ROS2 DDS server, while XR devices establish connections via the ROS TCP endpoint package within Unity [26]. Heterogeneous agents can participate in the mapping process. Mobile robots equipped with 3D LiDARs or cameras generate 3D octomaps of the navigated environment using LiDAR-based and vision-based SLAM, respectively. The XR agent participates in multi-map alignment, map accuracy inspection, edits application, and labeling objects of interest. The central unit comprises four key nodes: the Communication node, the Merger node, the Semantic node, and the Navigation node.

A. System Architecture

1) Communication node: the communication node operates as a multi-threaded entity responsible for managing communication among various agents, functioning as a communication layer with multiple callbacks. The $mapcallback$ oversees the retrieval of sub-maps from all mapping agents, storing them in an octomap dictionary. Each mapping agent, identified by a unique ID, publishes its generated map to the $/octo-ID$ topic as an octomap ROS message. Additional callbacks, namely add, $delete$, and $label$, were designed to handle the acquisition of voxels added to the map, voxels removed from the map, and voxels assigned specific labels by the human agent, respectively. Beyond these callbacks, services are established and activated from the communication node based on human input. All services are part of a re-entrant callback group, facilitating multi-threaded execution. The key services are:

[1]https://github.com/AUBVRL/HAC-SLAM

Manuscript 1273 submitted to 2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Received February 28, 2025.

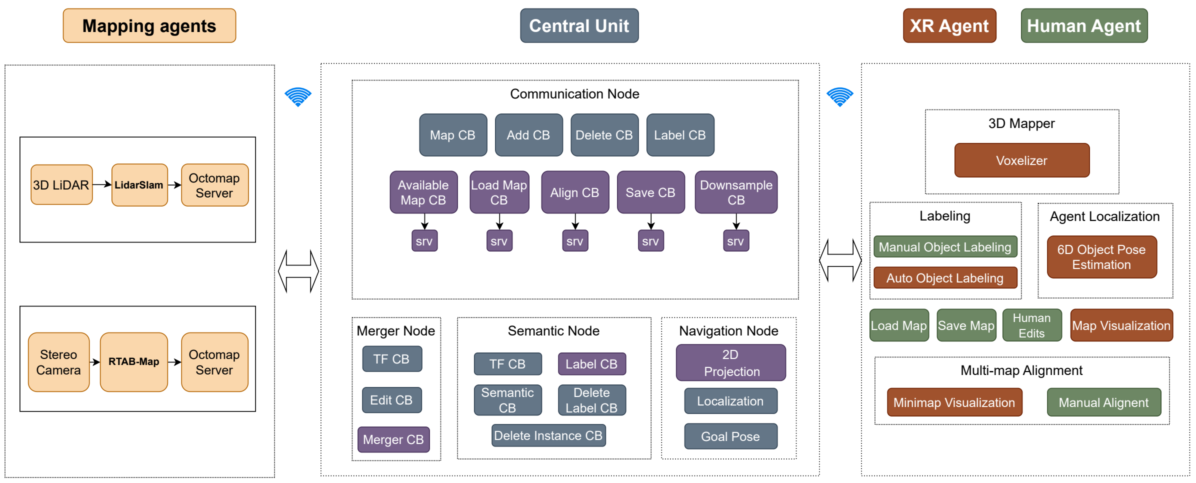


Fig. 2. System architecture illustrating the communication between heterogeneous agents and the central unit, including the modules for localization, semantic mapping, 3D reconstruction, and manual labeling, with the fusion of multi-agent data into a global map for visualization and navigation.

- The $downsampler$ service: designed to handle the XR headset's limited computational capabilities by downsampling sub-maps for more efficient communication and visualization.
- The $align$ service: developed based on the Iterative Closest Point (ICP) algorithm. The service receives the initial manual alignment of the sub-maps and refines it for accurate map fusion.
- The $saveMap$ service: developed for saving the fused map allowing map re-usage.
- The $loadMap$ service: the service loads user-selected offline map into the fusion process. Offline $.bt$ files are loaded and published into the $/octo - ID$ topic with an ID equal to the maximum number of agents. The loaded map is then treated as an extra agent.

2) Merger node: the merger node undertakes the task of integrating sub-maps from all agents into a cohesive and comprehensive map and is designed to handle n heterogeneous agents. This node operates with two primary callbacks on a single thread. The $Edit$ callback handles the incorporation of human edits received from the communication node into the merging process. The pivotal functionality of the merger node unfolds in the $Merge$ callback, a periodic callback triggered at a frequency set by the user. During each invocation, the $Merge$ callback iterates over all sub-maps and identifies those that have been aligned for further processing. Aligned maps, if updated, are then considered in the merging process. Each agent is assigned a pre-defined confidence level that influences its weight in contributing to the final merged map. A weighted average is subsequently employed to calculate the occupancy probability of a particular voxel as follows:

$$MT[n] = \frac{(WT[n] * MTO[n] + AW * SO[n])}{(WT[n] + AW)}, \quad (1)$$

$$WT[node] = WT[n] + AW * WT[n], \quad (2)$$

where MT stands for $mergedTree$ occupancy tree, WT for $weightTree$ storing the cumulative node weight, AW for $agentWeight$ specified in the central unit database, SO for the agent's sub-map occupancy value and n for the node in question. Voxels that have been modified by a human agent (either by adding or deleting voxels) are assigned specific and clearly defined log-odds values, ensuring their distinct identification from other voxels and preserving their integrity in subsequent fusion processes.

3) Semantic node: the semantic node is responsible for associating semantic data with the fused map for enhanced autonomy and scene understanding. It features three key callbacks; the $semantic$ callback is tasked with integrating semantic data into the merged map to generate a semantic point cloud. The resultant point cloud exhibits three custom fields, in addition to the regular xyz fields: a probability field indicating the confidence of occupancy, a label field denoting the class to which a point belongs, and an instance field used for tracking the available instances of a certain class. To enhance flexibility and manipulation, the $deleteLabel$ callback facilitates the deletion of all instances associated with a specific class. Simultaneously, the $deleteInstance$ callback is designed to remove a particular instance linked to a specific class from the fused map.

4) Navigation node: the navigation node plays a crucial role in preparing the necessary data for enabling robot navigation, featuring two key callbacks: the $occupancyGrid$ callback and the $localize$ callback. The $occupancyGrid$ callback generates a 2D occupancy grid derived from the 3D octomap, projecting obstacles within a specified height range onto the robot plane for navigation purposes. On the other hand, the $localize$ callback establishes the transformation between the map frame, upon which the merged map is based, and the $baselink$ frame of the robot. Following this calibration, the robot gains the capability to navigate the environment autonomously, utilizing the $nav2BringUp$ node provided by the navigation stack of ROS2.

B. XR Agent

In addition to the capabilities that the XR agent had in our previous work [25], the current system introduces the following features:

1) Voxel grid mapping: the XR agent utilizes the map generated by the device to create a voxelized version of the environment. The $Voxelizer$ node handles the voxelization and refinement of the map simultaneously. To attain the concurrency of addition and deletion, raycast techniques are replaced by box collider overlaps, which are instantiated in a grid-like manner in the scene with dimensions of 5 cm for each side. Each collider checks if it is overlapping another one; if the discovered collider is the spatial mesh generated by the XR agent, then the voxel's probability at the collider's position is increased by 5%. If the overlapped collider is that of a misplaced voxel (where it does not intersect with the spatial mesh), then that voxel's probability is decreased by 1%. The voxel's probability is assessed after each frame to either instantiate or destroy the voxel based on a preset threshold of 60%.

2) Automatic labeler: in an industrial setting, most of the recurring objects in the environment are known objects that possess a 3D CAD model such as robots, charging stations, pallets, etc. These objects can serve as important landmarks for better scene understanding as well as for enhanced navigation. Therefore, a 6D object-pose estimation model is employed to automatically detect and track objects of interest, in addition to injecting them into the fused map as known assets for semantic enrichment. For that purpose, a single CAD of the object of interest is utilized for training the neural network (NN) model. Once the training is complete, the model is outputted as a deployable Unity package to utilize onboard the XR device. Inference requires the device's RGB camera to be enabled for detection. Once the object of interest is in the frame of the camera, the NN infers its position and orientation with respect to the XR device's origin. To confirm an accurate alignment, the XR device renders the CAD of the detected object on top of its physical twin using the acquired transformation. The user assesses the alignment to confirm a correct transformation. Upon confirmation, the CAD is transformed into labeled voxels and sent as a point cloud message to the $SemanticNode$ to incorporate them into the global map.

3) Agent localization: to enhance the localization of autonomous agents, the XR agent can intervene in correcting the robot localization using the 6D object pose estimation model. The transformation of the physical agent is obtained and used to localize the agent in the XR's local map. Knowing the transformation between the XR's local map and the global map from the alignment step, the agent's location is transformed from the local map frame to the global map frame. The resulting transformation is published to the navigation node to update the global map with the agent's location, making the agent virtually visible on the grid for other agents and itself.

4) Path visualization and editing: autonomous navigation in dynamic environments can result in catastrophic collisions due to the lack of special infrastructure. Immersive visualization of the planned path can be crucial in providing efficient navigation supervision, especially during early deployment phases. Our framework allows the XR agent to examine the generated path by overlaying it onto the real environment. The user can then assess if the agent has successfully planned a safe path around the obstacles in the environment. Knowing that some obstacles might be problematic for the autonomous robots to map, human intervention alleviates these limitations by manually editing any part of the map that is not accurate. These edits will be reflected in the autonomous agent's planned routes since the agent will now accommodate the previously unseen obstacles. This also facilitates robots and humans working side by side by enhancing safety.

C. Human-in-the-loop

The human agent's involvement is cardinal as it ensures the refinement and accuracy of the global map. The key influences on the system are the following:

1) Multi-map alignment: the human agent possesses the flexibility to align any agent's sub-map using the XR's user-friendly interface. Initially, the user aligns the XR agent's sub-map with any available agent's sub-map, ensuring that the two sub-maps share overlapping features from the environment for successful alignment, resulting in the creation of a global map accessible to all agents on-site. Subsequently, the user can align any agent's sub-map into the global map using the same approach. Each successful alignment integrates the sub-map into the global map, showcasing the collaborative mapping capabilities inherent in our system.

2) Voxel editing and asset injection: the XR agent offers the user the choice of editing the map in real time. The edits are of three types:

Voxel addition: when an agent fails to map the environment correctly (due to problematic objects or unreachable areas), the user can correct the global map by adding voxels to the environment. The added voxels are published to the semantic node to update the global map.

Voxel deletion: the process follows a similar methodology to that mentioned above. The deleting feature is crucial for scenarios where an agent maps an undesired area or obstacle.

Asset injection: for easier robot deployment and enhanced navigation, assets such as charging stations, pallet dockers, etc. can be injected virtually into the fused map even before the physical twin is deployed. The 3D CADs of the real assets are pre-loaded into the XR agent beforehand. The human agent injects these assets into the environment, which automatically incorporates the global map with the changes. The added assets are also labeled with the CADs' names and published to the $SemanticNode$.

3) Manual labeling: manual labeling is applied to recurring objects in the environment that do not have a pre-loaded CAD

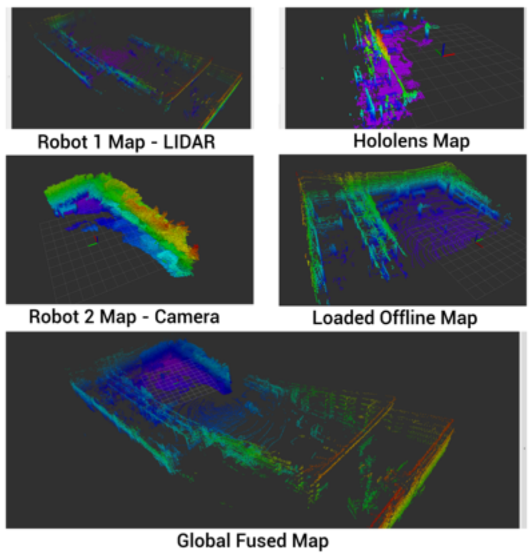


Fig. 3. Sub-maps generated by heterogeneous agents: top-left map corresponds to a LiDAR generated map, the top-right map represents the map outputted by the developed HoloLens application, the middle-left represents the SLAM product of a stereo camera, and the middle-right map depicts a loaded offline map. The larger map below represents the fused global map combining data from the four sub-maps.

file, yet still play a valuable role in improving autonomy. This feature utilizes the same selection technique applied earlier in Section III-C2. Following the confirmation of the selected voxels, the user is prompted to either input a new label name or select a previously added label. The labeled voxels are then published to the semantic node. The labeling feature empowers the human agent to delete a specific instance of an object by merely selecting it on-site. Additionally, the user can opt to delete all instances of a particular object from the environment.

IV. EXPERIMENTS AND RESULTS

The system was tested in two different settings: one in a real industrial environment at the IDEALworks GmbH research and development center in Munich, Germany, and the other on the fourth floor of the Ray R. Irani Oxy Building at the American University of Beirut. In the first setting, the experiment focused on qualitative data, assessing the clarity of the XR interface, and visually evaluating the effectiveness of human interventions in the mapping process. In contrast, the second setting was dedicated to quantitative analysis, providing objective measures of the improvements achieved through the proposed system.

A. IDEALworks setup:

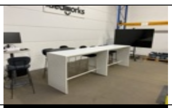
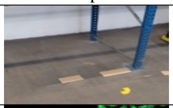
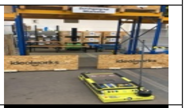
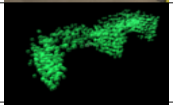
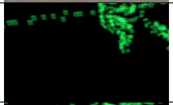
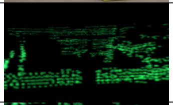
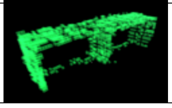
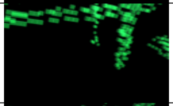
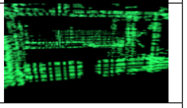
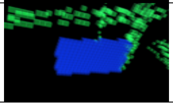
This section details the qualitative evaluation of our system at IDEALworks. Three heterogeneous agents collaborated in the mapping process:

- The iw.hub Version 4 robot, equipped with an LS CH32R 3D LiDAR and operating on Ubuntu 20 with ROS2 Foxy. Being a LiDAR-equipped agent, the robot runs the $lidarSlam$ package onboard, explained in [25].
- The iw.hub Version 4 robot, equipped with an Intel D435 camera and operating on Ubuntu 20 with ROS2 Foxy. This agent maps the navigated environment using the RTAB-Map package [27].
- The Microsoft HoloLens 2 running the developed application detailed in section III-B.

Each mapping agent navigates a different part of the warehouse and builds its local sub-map. The generated octomaps are subsequently published to the DDS server, each under a specific $/octoID$ topic for inclusion in the fusion process. After covering a significant portion of the environment, the human agent aligns the XR's generated map with the robot's sub-map as detailed in [25]. The alignment generates a merged global map that the user uses to align the other sub-maps to it. All aligned sub-maps are then merged, generating a fused global map. The result is a complete representation of the environment that is achieved in a remarkably fast time. Instead of waiting for each agent to map the entire environment, partial maps are accurately integrated along with previously saved maps as shown in Fig. 3.

To assess the resultant map, the user toggles the $ViewMergedMap$ button, allowing for the overlay of the generated map onto the real environment. Table I shows how various environmental features achieved accurate representation only upon the fusion of the agent's sub-maps. For areas that are inaccessible by the robot, the human agent maps the area in question and fuses the new region with the global map, making this region virtually accessible to all other agents on site. After examining the 1:1 generated map, the user identifies mapping errors in the environment that are not accurately captured by any of the agents and proceeds to edit the generated map by adding the missing obstacles.

TABLE I
MAPPING RESULTS. GROUND TRUTH (ROW 1); ROBOT 3D MAP (ROW 2); MERGED MAP (ROW 3); HUMAN EDITED MAP (ROW 4). THE GROUND PLANE AND WALLS ARE CLIPPED FOR VISUALIZATION PURPOSES.

	IDEALworks GmbH Research and Development center		
	Table	Glass panel	Inaccessible area
Ground Truth			
Robot Map			
Merged Map			
Edited Map	No Edits Needed		No Edits Needed

To enrich the map with semantic data, the XR agent navigates the environment and automatically detects pre-trained objects using a 6D pose estimation model provided by Vuforia. The developed algorithm assigns on-the-go labels to mobile robots, charging stations, dollys, and docking stations. Fig. 4 shows the automatic labeling of the iw.hub robot. In instances where the human operator desires to label a custom object of interest not covered by the pre-trained models, manual labeling is accomplished by pinching and selecting the volume of interest, as depicted in Fig. 5. This modular and flexible approach for semantic map creation allows for a focused representation of only the pertinent objects, preventing unnecessary data overload on the server.

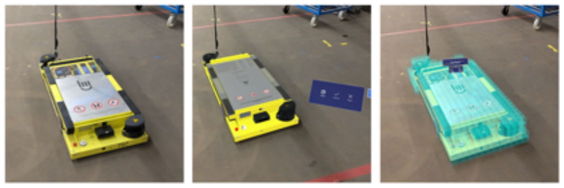


Fig. 4. Automatic object pose estimation and labeling. From left to right, the human agent positions the iw.hub robot in the field of view. The NN model detects the robot and overlays it's corresponding CAD model on top of its real counterpart. The CAD model is then voxelized and labeled iw.hub after the user confirmed the alignment.

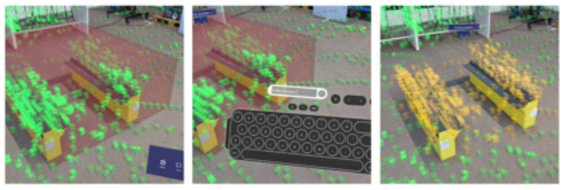


Fig. 5. Manual labeling. From left to right, the human agent first selects the voxels of interest using hand gestures. The label name is then inputted using a virtual keyboard (middle), and the voxels of interest are successfully assigned the inputted semantic value (shown in yellow).

At this stage, a complete and correct representation of the surrounding environment with semantic-enriched data is achieved. For the facilitation of modular robot deployment and accurate environment representation, the user is capable of adding obstacles and assets to the generated map, even if the physical version is absent from the scene. Fig. 6 demonstrates the process of adding a charging station in its anticipated future location facilitating robot deployment.

Following map generation, the user proceeds to save the fused map. The human agent triggers the save map button using the XR menu and inputs the desired file name for the current global map to be saved. The map is saved as a *.bt* file for loading it into the system, and a *.pcd* file for a point cloud representation of the environment.

At the deployment phase, the user loads the generated map and aligns it with the XR agent's current map. The navigation node generates the 2D occupancy grid by projecting a section of the 3D octomap onto it. Next, the XR agent automatically detects the iw.hub robot using the same 6D object pose estimation model used for automatic labeling. The obtained transformation of the robot is published to the navigation node. Following this step, the robot is then ready to

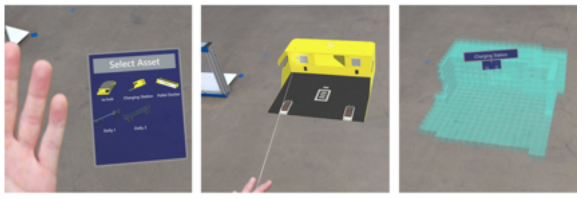


Fig. 6. Adding assets. From left to right, the human first chooses the asset to add and places it in the desired location using hand gestures. After confirmation, the asset is voxelized and added to the merged map along with its label.

navigate the environment autonomously. Tests are conducted to evaluate the impact of edits on the robot's navigation. Commands are issued to the iw.hub, allowing the user to visualize and inspect the planned path. As depicted in Fig. 7, the correction of the map's inaccuracies and the addition of unmapped obstacles plays a crucial role in securing a safe path. The path visualization feature allows the user to inspect the planned path which can be helpful for Day zero robot deployment. The result is a complete framework allowing for faster and more accurate mapping, flexible semantics, and modular robot deployment.

B. American University of Beirut

This section presents the quantitative evaluation conducted at the Ray R. Irani Oxy Building at the American University of Beirut. The leveraged devices that collaborated in the following section are:

- The Clearpath Husky A200 four-wheel robot equipped with a Mini-ITX computer and a Velodyne VLP-16 3D LiDAR. The system is operating on Ubuntu 20 with ROS2 Foxy. The robot is tele-operated using a Logitech F710 wireless game pad.
- The Microsoft HoloLens 2 running the developed application detailed in section III-B.
- Leica BLK360 G1 3D LiDAR scanner for providing ground-truth, dense point clouds of the environment.

Two distinct environments were selected to evaluate the accuracy of our system. The first setting, a glass corridor, presents a challenging scenario for robotic mapping due to the transparent surfaces found on each side of the corridor. In contrast, the second setting, the Vision and Robotics Lab

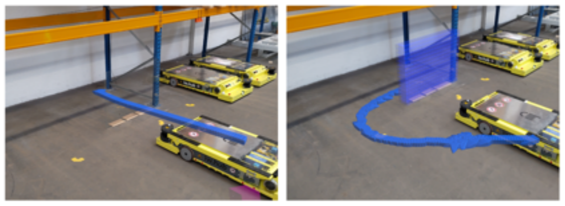


Fig. 7. Effect of human edits on robot navigation. Path before edits would have caused a robot collision with the glass pane that was not automatically detected.

(VRL), simulates a restricted area where the robot's maneuverability is limited. The Leica BLK360 scanner was employed for both settings to capture the ground truth data, outputting files in the $.e57$ format. However, as a laser-based device, the Leica did not capture the glass structures in either area. To address this, CloudCompare [28] was utilized to manually incorporate the glass windows based on measurements taken from the actual installations, which required approximately 40 minutes of manual work to achieve.

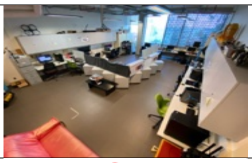
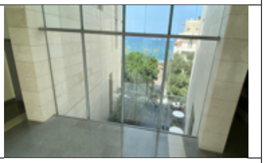
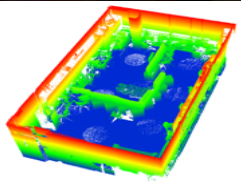
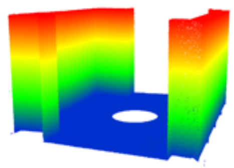
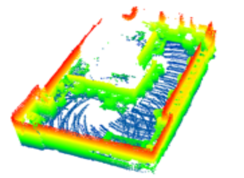
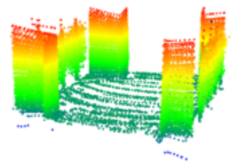
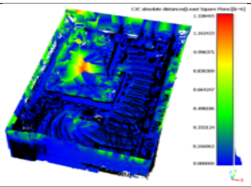
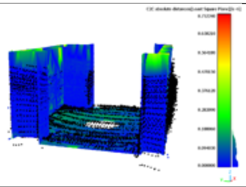
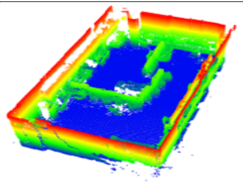
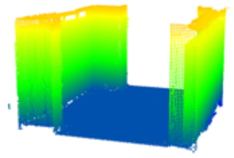
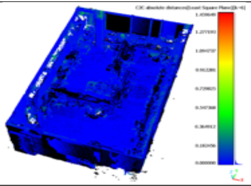
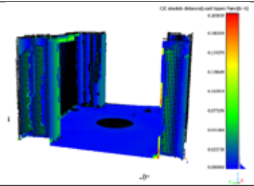
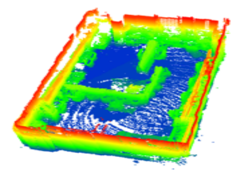
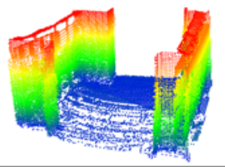
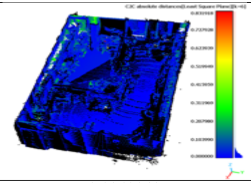
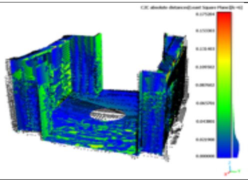
The robot was then teleoperated in both settings via a joystick while running the $LiDARSLAM$ package onboard. The resulting point clouds also failed to capture the glass windows and did not record any features from the restricted area of the lab. In parallel, the HoloLens scanned the same environments and generated a point cloud that similarly omitted the glass windows; this was rectified in situ using the system's editing feature, which injected the missing window data into the map. The corrections took approximately 3 minutes to complete. Finally, the corrected HoloLens point cloud and the robot-generated point cloud were merged using our proposed system. All the maps were transmitted as a $pointcloud2$ ROS message to the central unit. The messages were then converted to $.PLY$ format to facilitate the comparison with the ground truth maps recorded by the Leica.

All acquired point clouds were imported into CloudCompare, aligned with the Leica reference scan and analyzed to compute the mean error and standard deviation between the datasets. The error calculation in CloudCompare utilizes the least squares plane local model to compute the distance between two point clouds. The model first identifies, for each point in the compared cloud, the nearest point in the reference cloud. A local neighborhood is then selected around this nearest point based on a fixed number of neighbors, which are used to compute the least-square best-fitting plane. The distance from the point in the compared cloud to this fitted plane is calculated and used as the cloud-to-cloud distance measure. The results are tabulated in Table II

Quantitative analysis revealed that the mean error of the merged map relative to the ground truth was intermediate between the superior performance of the HoloLens alone and the higher error observed in the robot's output. This indicates that while the integration of outlier data from the robot slightly degrades the high-resolution, dense point cloud produced by the HoloLens, it simultaneously enhances the overall mapping quality compared to the robot's performance. Moreover, the standard deviation of the merged map compared to the ground truth was lower than that of each agent separately, since the fusion process effectively averages out random noise and reduces extreme discrepancies and variability.

Although the HoloLens independently yields more accurate results in the demonstrated, small-scale setting, its limited processing power restricts scalability in larger industrial environments. In such cases, the robot's mapping capabilities are advantageous for covering extensive areas, where the HoloLens intervenes in targeted regions to correct specific inaccuracies, such as missing glass surfaces and unpopulated

TABLE II
MAPPING RESULTS. REAL SETUPS (ROW 1); GROUND TRUTH MAP (ROW 2): ROBOT 3D MAP (ROW 3); ROBOT OVERLAPPED WITH GROUND TRUTH (ROW 4); HOLOLENS EDITED MAP (ROW 7); HOLOLENS OVERLAPPED WITH GROUND TRUTH (ROW 8); MERGED MAP (ROW 11): MERGED WITH GROUND TRUTH (ROW 12)

	American University of Beirut	
	Restricted Area	Glass Corridor
Ground Truth		
Ground Truth Point Cloud		
Robot Map		
Robot Map Overlap Error		
Mean Error	0.100994	0.0816562
Std. Dev	0.154399	0.0683867
HoloLens Edited Map		
HoloLens Overlap Error		
Mean Error	0.021105	0.017918
Std. Dev	0.030039	0.0249778
Merged Map		
Merged Map Overlap Error		
Mean Error	0.0240942	0.0247489
Std. Dev	0.0254207	0.0225431

restricted zones. These findings underscore the complementary strengths of both systems and confirm that our approach successfully balances detail and scalability to deliver enhanced mapping performance in diverse operational contexts.

V. Conclusion

In this paper, we presented a collaborative human-assisted SLAM approach based on XR. By fusing XR and SLAM, our system proved efficient in fusing maps from any number of heterogeneous agents, while increasing the generated map's quality. The proposed approach allows for enhanced focus on robot navigation and localization by creating semantically rich maps and allowing for supervised robot localization and navigation. The proposed system was successfully tested in a real experimental setting. Future work can explore the effect of the human agent on the efficiency of the proposed system as well as a measure of the effort needed compared to other frameworks. In addition, the collaborative nature of our human-robot framework signifies a step towards cognitive mapping, harmonizing human reasoning with the mapping capabilities of robots. This collaborative synergy lays the groundwork for advanced applications in mapping, semantic understanding, and autonomous navigation. Finally, scalability is an important factor to be studied in future work given the hardware limitation of AR headsets available in the market.

Acknowledgment

This work was supported by the DIDYMOS-XR Horizon Europe project (grant number 101092875–DIDYMOS-XR, www.didymos-xr.eu), and the Research Board at the American University of Beirut.

References

[1] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: part i," *IEEE robotics & automation magazine*, vol. 13, no. 2, pp. 99–110, 2006.

[2] L. Jinyu, Y. Bangbang, C. Danpeng, W. Nan, Z. Guofeng, and B. Hujun, "Survey and evaluation of monocular visual-inertial slam algorithms for augmented reality," *Virtual Reality & Intelligent Hardware*, vol. 1, no. 4, pp. 386–410, 2019.

[3] T. Lai, "A review on visual-slam: Advancements from geometric modelling to learning-based semantic scene understanding using multi-modal sensor fusion," *Sensors*, vol. 22, no. 19, p. 7265, 2022.

[4] T. T. O. Takleh, N. A. Bakar, S. A. Rahman, R. Hamzah, and Z. Aziz, "A brief survey on slam methods in autonomous vehicle," *International Journal of Engineering & Technology*, vol. 7, no. 4, pp. 38–43, 2018.

[5] M. Sommer and K. Seiffert, "Scan methods and tools for reconstruction of built environments as basis for digital twins," *DigiTwin: An approach for production process optimization in a built environment*, pp. 51–77, 2022.

[6] A. Sidaoui, M. K. Zein, I. H. Elhajj, and D. Asmar, "A-slam: Human in-the-loop augmented slam," in *2019 International conference on robotics and automation (ICRA)*. IEEE, 2019, pp. 5245–5251.

[7] J. Singh, G. Singh, R. Verma, and C. Prabha, "Exploring the evolving landscape of extended reality (xr) technology," in *2023 3rd International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON)*. IEEE, 2023, pp. 1–6.

[8] S. H.-W. Chuah, "Why and who will adopt extended reality technology? literature review, synthesis, and future research agenda," *Literature Review, Synthesis, and Future Research Agenda (December 13, 2018)*, 2018.

[9] N. V. Valen, "The potential and challenges of extended reality (xr) technology in healthcare," Master's thesis, NTNU, 2023.

[10] A. Logeswaran, C. Munsch, Y. J. Chong, N. Ralph, and J. McCrossnan, "The role of extended reality technology in healthcare education: Towards a learner-centred approach," *Future healthcare journal*, vol. 8, no. 1, pp. e79–e84, 2021.

[11] Å. Fast-Berglund, L. Gong, and D. Li, "Testing and validating extended reality (xr) technologies in manufacturing," *Procedia Manufacturing*, vol. 25, pp. 31–38, 2018.

[12] T. Morimoto, T. Kobayashi, H. Hirata, K. Otani, M. Sugimoto, M. Tsukamoto, T. Yoshihara, M. Ueno, and M. Mawatari, "Xr (extended reality: virtual reality, augmented reality, mixed reality) technology in spine medicine: status quo and quo vadis," *Journal of Clinical Medicine*, vol. 11, no. 2, p. 470, 2022.

[13] J. M. M. Ferreira and Z. I. Qureshi, "Use of xr technologies to bridge the gap between higher education and continuing education," in *2020 IEEE Global Engineering Education Conference (EDUCON)*. IEEE, 2020, pp. 913–918.

[14] M. Sakashita, H. Kim, B. Woodard, R. Zhang, and F. Guimbretière, "Vroxy: Wide-area collaboration from an office using a vr-driven robotic proxy," in *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023, pp. 1–13.

[15] F. Sanfilippo, J. Smith, S. Bertrand, and T. H. S. Svendsen, "Mixed reality (mr) enabled proprio and teleoperation of a humanoid robot for paraplegic patients," in *2022 5th International Conference on Information and Computer Technologies (ICICT)*. IEEE, 2022, pp. 153–158.

[16] J. Steinke, J. Rischke, P. Sossalla, J. Hofer, C. L. Vielhaus, N. Vom Hofe, and H. F. Fitzek, "The future of dog walking–four-legged robots and augmented reality," in *2023 IEEE 24th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2023, pp. 352–354.

[17] X. Wang, L. Shen, and L.-H. Lee, "A systematic review of xr-based remote human-robot interaction systems," *arXiv e-prints*, pp. arXiv–2403, 2024.

[18] A. Kozlov, B. MacDonald, and B. Wünsche, "Towards improving slam algorithm development using augmented reality," in *Proceedings of Australasian Conference on Robotics and Automation*. Citeseer, 2007.

[19] H. Stedman, B. B. Kocer, M. Kovac, and V. M. Pawar, "Vrtabmap: A configurable immersive teleoperation framework with online 3d reconstruction," in *2022 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 2022, pp. 104–110.

[20] S. M. Chacko, A. Granado, A. RajKumar, and V. Kapila, "An augmented reality spatial referencing system for mobile robots," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4446–4452.

[21] H. Mu, Y. Li, D. Chen, J. Li, and M. Wang, "Design of tank inspection robot navigation system based on virtual reality," in *2021 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2021, pp. 1773–1778.

[22] M. Gu, A. Cosgun, W. P. Chan, T. Drummond, and E. Croft, "Seeing thru walls: Visualizing mobile robots in augmented reality," in *2021 30th IEEE International Conference on Robot Human Interactive Communication (RO-MAN)*, 2021, pp. 406–411.

[23] W. Gai, H. Li, Y. Zhao, M. Shi, W. Wang, Y. Sun, and C. Yang, "A new navigation method for vr-based telerobotic system via supervisor's real walking in a limited physical space," in *2020 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2020, pp. 494–498.

[24] F. Grzeskowiak, M. Babel, J. Bruneau, and J. Pettré, "Toward virtual reality-based evaluation of robot navigation among people," in *2020 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. IEEE, 2020, pp. 766–774.

[25] M. Sayour and M. Yassine, "Hac-slam: Human assisted collaborative 3d-slam through augmented reality," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024.

[26] Unity-Technologies, "Ros-tcp-endpoint," 2021. [Online]. Available: https://github.com/Unity-Technologies/ROS-TCP-Endpoint.git

[27] IntRoLab, "rtabmap_ros." [Online]. Available: https://github.com/introlab/rtabmap_ros/tree/ros2

[28] CloudCompare Team, "Cloudcompare," http://www.cloudcompare.org, 2020, open source 3D point cloud processing software.