HAC-SLAM: Human Assisted Collaborative 3D-SLAM Through Augmented Reality

Malak Sayour, Mohamad Karim Yassine, Nadim Dib, Imad H. Elhajj, Boulos Asmar, Elie Khoury, Daniel Asmar

***Abstract*— Simultaneous Localization and Mapping (SLAM) has emerged as a prime autonomous mobile agent localization algorithm. Despite the global research effort to improve SLAM, its mapping component remains limited and serves little more than to satisfy the coupled localization problem. We present a collaborative 3D SLAM approach leveraging the power of augmented reality (AR). The system introduces a trio of diverse agents, each with its unique capability to become an active member in the mapping process: mobile robots, human operators, and AR head-mounted display (AR-HMD). A 3D complementary mapping pipeline is developed to utilize the built-in SLAM capabilities of the AR-HMD as shareable data. Our system aligns and merges the AR-HMD and the robot's local map automatically, triggered by a human-dictated initial guess. The created merged map proves advantageous in scenarios where the robot is restricted from navigating in certain areas. To correct map imperfections resulting from problematic objects such as transparent or reflective surfaces, the fused map is overlayed onto the environment, and hand gestures are used to add or delete 3D map features in real-time. Our system is implemented in both a lab and a real industrial warehouse setup. The results show a significant improvement in the map quality and mapping duration.**

I. INTRODUCTION

Recent advancement in the robotics and automation field [1] [2] has created a pressing need for accurate agent localization and high-precision spatial information. While Simultaneous Localization And Mapping (SLAM) has emerged as a prime algorithm for agent localization, its mapping component remains limited, resulting in poor spatial representations.

The limited capabilities of SLAM in reconstruction are often caused by the sensor's inability to capture the navigated environment's geometrical information [3]. Sensor Noise, reflective surfaces, transparent objects, and other factors, contribute to incomplete scene representations and can lead to catastrophic failure in the agent's navigation [4]. In such scenarios, human intervention becomes imperative to rectify inaccuracies and ensure safe navigation. Manual map post-processing has been widely adopted to ensure correct spatial representations. However, such a process can prove to be an arduous and time-consuming endeavor, since completing a robot-generated map requires multiple site visits for missing measurements. In certain scenarios, post-processing could require manual measurements of areas where a robot is prohibited or incapable of navigating.

Collaborative SLAM has emerged as a technique to improve the mapping capabilities of traditional SLAM. Merging multiple agents' sub-maps [5], or fusing sensor data of multiple entities [6] all improve the generated map accuracy either through the redundancy of data or by capitalizing on the advantageous characteristics of one entity that complements those of the other. Moreover, the remarkable surge in the development and integration of AR has blurred the boundaries between the virtual and physical worlds [7]. These cutting-edge technologies have opened new possibilities for 3D map visualization. AR can offer a user-friendly 3D map visualization, which replaces two-dimensional screens and allows for a deeper understanding of the mapped environment [8].

We present a human-robot-AR collaborative SLAM, reshaping the way agents interact, while minimizing the mapping process. Through our innovative approach, we combine the unique properties of human operators, AR-HMD headsets, and mobile robots to generate a complete and accurate 3D representation of a navigable environment. Once the robot and AR-HMD maps become aligned and merged, the human operators are provided with real-time map visualization through the AR-HMD. The visualization enables the human operators to correct map imperfections, solve other agents' mapping conflicts, and most importantly overlay the generated map over the real world. The key contribution of our paper can be summarized as follows:

- Heterogeneous collaborative mapping: allowing the fusion of a mobile robot, AR-HMD 3D map, and human operator input.
- Agent localization in the global map: by utilizing intuitive interface tools, the human operator can localize other agents' submaps in the global map.
- AR Map Visualization: allowing the human operator to overlay the 3D map directly onto the real world.
- Error correction and map refinement: with direct access to the generated map, users can inspect and correct the map in real-time and in-situ.

The paper is structured as follows: Section II provides a concise overview of the relevant literature, Section III

M. Sayour, M. K. Yassine, 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: ms423@aub.edu.lb, mhy16@mail.aub.edu, ngd04@mail.aub.edu, ie05@aub.edu.lb, da20@aub.edu.lb. (M. Sayour and M. K. Yassine 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

introduces our proposed methodology, Section IV discusses the experiments and their results, and Section V serves as the paper's conclusion. The implemented code is made available on GitHub[1].

II. RELATED WORK

Multiple techniques have been proposed to improve the mapping capabilities of SLAM [9], [10]. Since many uncertainties originate from reflective and transparent objects, Koch et al. [11] presented a mirror detector and reflection classifier using multi-echo laser scanners in order to classify transparent and specular reflective objects. Their results showed that although it is possible to discriminate materials by the behavior of their reflective characteristics, transparent object detection remains a challenging problem as their appearance in the sensor data strongly depends on the background. In addition, Zhu et al. [12] implemented a SLAM approach named transfusion that allows transparent object detection and reconstruction. Their approach includes Transparent objects Cut Iterative Closest Points (TC-ICP) followed by Transparent Objects Reconstruction (TO-Reconstruction). Their results showed improvement in transparent object detection and reconstruction but did not tackle reflective objects. Also, their approach required high computational complexity and a GPU-equipped processor.

Collaborative SLAM has been previously attempted to increase mapping accuracy; nevertheless, local map merging remains among the most demanding problems in multi-agent mapping. Wu et al. [13] presented a map merging method for collaborative LiDAR-SLAM, based on GPS measurements and improved Iterative Closest Point (ICP). The GPS measurements provided an initial guess for the ICP algorithm, where KD-tree and normal vectors were used to register overlapping regions. However, the proposed approach cannot be applied to indoor environments or any outdoor environments void of GPS. To solve the global localization problem in indoor areas, Tian et al. [14] proposed a distributed multi-robot visual SLAM based on loop closure. Their work included a two-stage method for outlier-robust distributed Pose Graph Optimization (PGO). The accuracy of the proposed solution however depends crucially on detecting sufficient loop closures. In addition, the proposed Visual SLAM solution only performs visual loop closure detection, which makes it sensitive to changes in the viewpoint.

Augmented reality head-mounted displays (AR-HMD), such as the HoloLens 2, have gained significant attention in recent years due to their potential to enhance various applications across different fields. One of the emerging areas of interest is their utilization in collaborative 3D SLAM systems. Hofe et al. [8] employed the HoloLens 2 to visualize the 3D map generated by the Boston Dynamics Spot robot and overlay it onto the user's environment. However, aligning the map with the real environment required manual map positioning from the human, which can be time-consuming and affected by human error. Al-Sabbag et al. [15] proposed a Human–Machine Collaborative Inspection (HMCI) system that offers a novel approach where an AR-HMD undergoes inspections for structural defects in an environment, and anchors the location of the defects in a robot's map. Although HMCI leverages the strengths of the AR-HMD to enable real-time visualization of structural defects, it does not allow the inspector to visualize the mapped environment. Therefore HMCI is not suitable for scenarios where geometric modifications are needed in the mapped environment. In our previous work [16], we presented a collaborative 2D SLAM system that unites a robot, a human operator, and an AR-HMD. The innovation allows real-time map editing through hand gestures, utilizing the AR-HMD's SLAM capabilities to enhance mapping accuracy. The maps generated by the agents are aligned using an AR marker mounted on the robot. The marker is detected by the AR-HMD, which triggers the map merging process. While the system did well in enabling human-driven editing and map merging in a 2D occupancy grid context, this research extends this advancement to a 3D voxel grid domain. This progression tackles the non-intuitive problem of voxelizing an environment, requiring efficient memory management, efficient communication methodologies, and streamlined rendering techniques that tap into richer spatial representation and mapping complexity.

III. SYSTEM OVERVIEW

The system architecture is presented in Fig. 1. Each agent generates its own 3D map of the navigated environment and sends it to the central unit. In addition to the generated maps, when needed, the human agent provides the central unit with map edits. At the central unit level, local maps are aligned and merged. The central unit is also responsible for applying human edits to the generated global map. At this point, the human can visualize the merged map overlayed onto the real environment and continue editing if necessary. The flowchart for the the proposed approach is presented in Fig. 2.

- Each agent produces a 3D map as it navigates the environment.
- Once significant overlap between the maps is achieved, the align functionality is triggered by the human agent.
- The human agent receives a down-sampled version of the robot map and visualizes it on the AR-HMD as a mini-map.
- The human roughly aligns the mini-maps and publishes the transformation to the central unit.
- The central unit uses the provided transformation as an initial guess to the ICP algorithm.
- Once aligned, the agents' maps are merged.
- Beyond this step, every map update from any agent triggers the map merger.
- Once merged, the human agent can visualize the generated map overlayed onto the real environment in real-time
- If adjustments to the map are needed, the human can manually add or delete voxels using hand gestures.

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

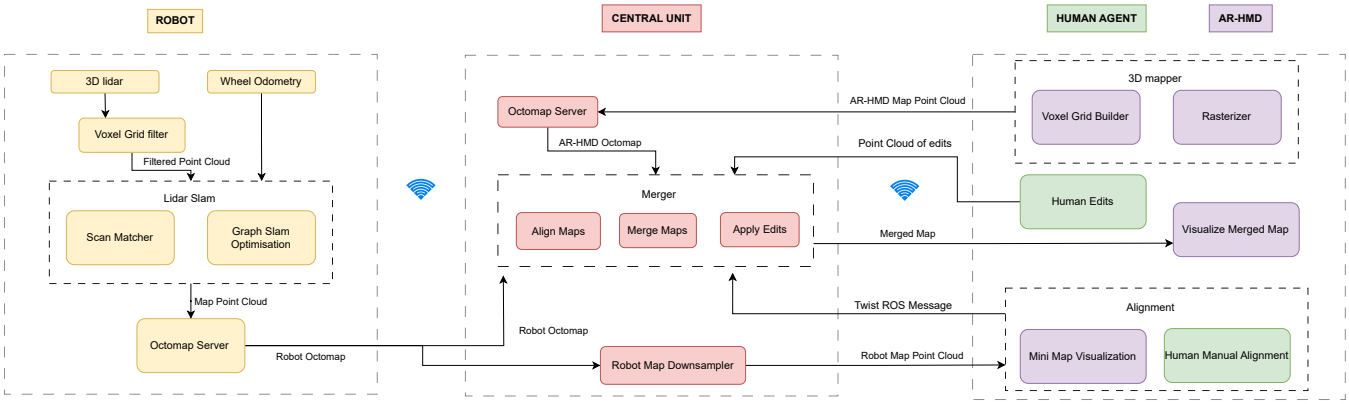


Fig. 1. System architecture

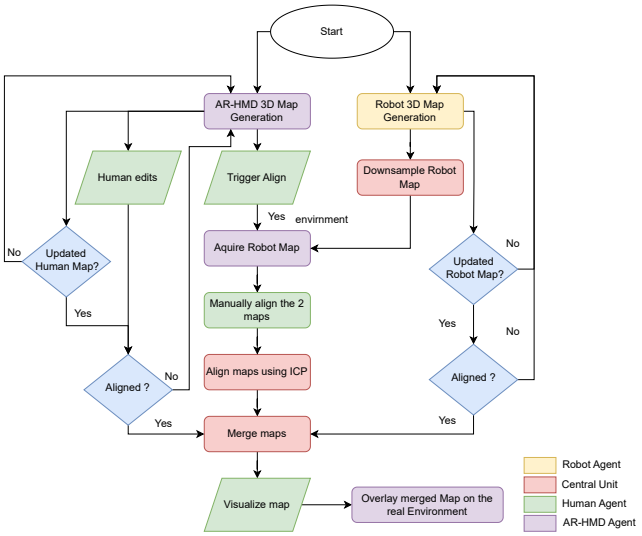


Fig. 2. Operational Flowchart

A. Robot Agent

The robot is equipped with a 3D LiDAR and is controlled through Robot Operating System (ROS 2). In order to generate the 3D map, the input point cloud is first downsampled by the $filterNode$ using the point cloud library (PCL) voxel grid filter at a resolution of 0.05 meters. The filtered point cloud is then used to generate the 3D map using LiDAR-SLAM [17]. LiDAR-SLAM consists of 2 nodes, the $scanMatching$ front-end node, and the $graphBasedSlam$ back-end node. The $scanMatching$ node estimates the sensor pose and builds the environment map by iteratively applying scan matching between consecutive frames using OpenMP-boosted normal distributions transform (NDT) scan matching. The $graphBasedSlam$ node compensates for the resultant accumulated error by performing loop detection and optimizes the generated pose graph. For reliable communication, the generated point cloud is converted into an octomap structure [18] and published to the ROS 2 data distribution service (DDS) server.

B. AR-HMD Agent

1) Voxel Grid Builder: the AR-HMD can generate a spatial mesh of the dynamic environment; raycasting techniques are leveraged to construct a discrete, cube-like representation of the environment called voxels. Furthermore, a novel algorithm is designed to harness the raycasting lines derived from the AR-HMD to instantiate voxels [16]. Initially a set of lines with varying incident angles are cast from the AR-HMD. If a ray returns a hit point, it is used as a voxel instantiation position. The generated position is then rounded with respect to the discretisation value, keeping the instantiated voxels from overlapping or being randomly spaced.

2) Raycast Rasterizer: in addition to populating the voxel grid, the raycasted lines are used to destroy any misplaced voxels in the environment originating from the dynamic spatial mesh of the AR-HMD. Whenever the generated spatial mesh is updated, the voxels occupying the previous mesh stay in place. The raycasted lines propagate in the environment until they hit a collider enabled GameObject, which in our case is set to the spatial mesh. Thus, the penetrated parts of the environment are defined as free. In order to acquire the position of these penetrated voxels, the line is rasterized in 3D. The algorithm for rasterizing a line is listed in Algorithm 1. The functions representing the line between the AR-HMD position and the hit point in the yx, zx, and yz planes are respectively denoted as $f(x)$, $g(x)$, and $h(z)$. The coordinates of the hit point are compared with those of the AR-HMD position and the results are saved as either 1 or -1 in (**DirectionVector**):

$$v = \left[\frac{\text{end.x} - \text{start.x}}{|\text{end.x} - \text{start.x}|} \quad \frac{\text{end.y} - \text{start.y}}{|\text{end.y} - \text{start.y}|} \quad \frac{\text{end.z} - \text{start.z}}{|\text{end.z} - \text{start.z}|} \right]^T,$$

where v is the vector that indicates the direction of the rasterizer ray in the 3D environment.

C. Human Agent

1) Map Alignment: the alignment phase requires the AR-HMD to first have a substantial portion of the environment mapped. Afterward, the human agent triggers an interactive step on the AR-HMD that allows for the alignment of

Algorithm 1 Rasterizer

Require: start, end
Ensure:
$f(x) \leftarrow$ Equation in $yx - plane$
$g(x) \leftarrow$ Equation in $zx - plane$
$h(z) \leftarrow$ Equation in $yz - plane$
$v \leftarrow$ DirectionVector$(start, end)$
$temp = start$
while $temp \neq end$ **do**
  if $temp.y = f(temp.x + v.x)$ **then**
    if $temp.z = g(temp.x + v.x)$ **then**
      $temp.x = temp.x + v.x$
    else
      $temp.z = temp.z + v.z$
    end if
  else
    if $temp.y = h(temp.z + v.z)$ **then**
      $temp.z = temp.z + v.z$
    else
      $temp.y = temp.y + v.y$
    end if
  end if
  Destroy voxel at position $temp$
end while

the local frame with the global frame. The agent views a downsampled version of the robot's 3D map, along with a miniaturized representation of the area explored earlier by the AR-HMD. The agent roughly localizes the AR-HMD map into the robot's map using intuitive hand gestures. Upon reaching satisfactory results, the agent triggers the AR-HMD to publish the transformation between the two maps as a ROS twist message to be used by the central unit's ICP as an initial guess. Upon reaching a precise alignment by the central unit's ICP, the final transformation between the two maps is sent back to the AR-HMD to be applied on the merged map generated by the central unit for accurate overlay over the operator's environment.

2) Map Editing: the system leverages the ability of the human agent to manipulate a map. The AR-HMD prompts the option to either delete or add voxels in the visualized environment through a user-friendly interface. Using hand gestures, the operator can select the lines, areas, and volumes of interest in 3D. After satisfactory selection, the operator then confirms the edits, triggering the AR-HMD to publish the newly added or deleted voxels over a designated ROS topic.

D. Central Unit

The central unit is responsible for acquiring and fusing maps received from different agents (Fig 1). To acquire the map generated by the AR headset, the ROS TCP endpoint package from Unity technologies was used [19]. The published map is received as a point cloud and transformed using the octomap server to an octomap structure. Four primary processes are running to ensure a smooth map fusion: the robot's map down sampler for smooth communication, the map aligner, the map merger, and the map editor.

1) Robot Map Downsampling: to optimize the manual alignment process, the $downSampler$ node filters the robot-acquired map using the PCL voxel grid filter with a resolution of 0.5 meters and publishes the down-sampled map to the DDS server for it to be used by the AR-HMD agent's manual alignment component. The set resolution ensures the persistence of the main map features while respecting the AR-HMD rendering capabilities and network bandwidth.

2) Map Alignment: the $ARFrame$ and $robotFrame$ represent the AR-HMD local map frame and the robot local map frame, respectively. In order to merge the maps generated by each agent, a transformation between $T^{global}_{ARFrame}$ and $T^{global}_{robotFrame}$ is needed to link each agent's local map to the global map. As we only have 2 agents, the robot's local map frame is considered the global frame. Therefore $T^{robotFrame}_{ARFrame}$ needs to be calculated before any merging process can be triggered. For this purpose, ICP is used; however, in scenarios where no overlap is detected between 2 scenes, the ICP algorithm fails to find the appropriate transformation matrix. This problem is often solved by using GPS data as an initial guess [13]. In an indoor environment, GPS cannot be used, and instead we rely on the human to initiate ICP with a rough manual alignment. The human can also reset the map in case of catastrophic SLAM failure. The merger acquires the transformation as a ROS 2 twist message and convertes into the initial preliminary transformation matrix $T^{robotFrame}_{ARFrame}$. Applying the transformation to the AR-HMD map, the central unit can detect and filter the map overlay. Subsequently, the filtered common area and the provided matrix are fed into the ICP algorithm, which refines the provided transformation between the 2 frames.

3) Map Merger: once the corresponding transformation matrix is generated, the map merging begins fusing the robot map with the acquired AR-HMD map to produce a merged map M. The robot R and AR-HMD's H acquired octomaps are both stored in an octree structure based on the type of octomap message received. Our system supports both full and binary resolution octomaps. Each octree node stores an occupancy probability in the form of a log-odds value. The first step is to generate the aligned octree A by multiplying H with the transformation matrix generated. Then, for each Node r in R, the merger finds the corresponding node a in A and sets the log-odd value of the corresponding node m in the output merged tree M to

$$\max(log - odd_a, log - odd_r), \tag{1}$$

giving more weight to the occupied cells. In order to keep track of the conflicts between the two agents maps, we define a conflict octree denoted by C. C is appended whenever a node is defined occupied by the robot yet free by the AR-HMD. At Last, M and C are published to the DDS server for it to be accessible by the robot and the AR-HMD. The complete merge logic is presented in table I. Following the first merging process, the merger keeps on checking for updated maps at every time step. Whenever an update is

TABLE I

MERGER DECISION TABLE, FREE (F), OCCUPIED (O), AND UNKNOWN (U).

M_{t-1}	F									O									U								
R_t	F			O			U			F			O			U			F			O			U		
A_t	F	O	U	F	O	U	F	O	U	F	O	U	F	O	U	F	O	U	F	O	U	F	O	U	F	O	U
M_t	F	O	F	O	O	O	F	O	F	F	O	F	O	O	O	F	O	O	F	O	F	O	O	O	F	O	U

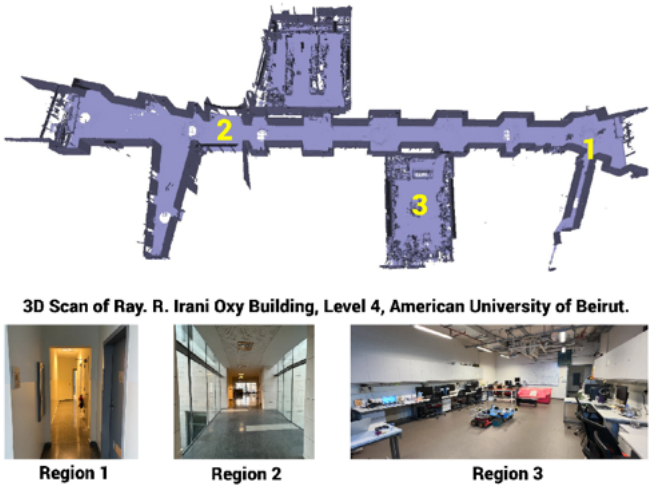


Fig. 3. Ray Irani Oxy Building level 4 3D representation.

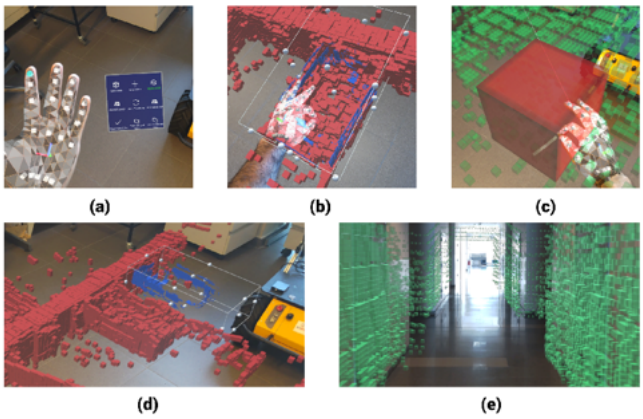


Fig. 4. Captures from the AR-HMD. Figure (a) features the interactive AR menu. Figure (b) and (d) show the process of aligning the minimaps. Figure (c) displays the 3D selector for deleting and adding voxels. Figure (e) shows a one-to-one representation of the generated map overlayed on the real.

received from one of the agents, the merging process is repeated.

4) Map editing: In addition to aligning and merging maps, the merger node runs a parallel process monitoring the map edits published by the human agent. Obstacles to be added are published on the $humanEditsAdd$ topic as a point cloud. Whenever data is published on this topic, the edit callback is triggered. First, the acquired data is transformed to match the global map frame; second, the function iterates over the transformed points and generates the octree key associated with the point 3D coordinates. Following this step, the log-odd value corresponding to the generated key is set to 100. Since log-odd values never reach the value of 100 in octrees, this value will help the merger identify the human-edited voxels and make them non-modifiable by the merging thread. Similar logic applies to deleting obstacles, where deleted points are sent on the $humanEditsDelete$ topic, and the log-odd values are set to -100.

IV. EXPERIMENTS AND RESULTS

We test our system in two experimental settings. The first experiment is conducted on the fourth floor of Ray. R. Irani Oxy Building at the American University of Beirut. The second experiment is conducted in a real industrial setup at the idealworks warehouse at the BMW plant in Munich, Germany.

A. Experiment 1

In this experiment, we validate our results on the Clearpath Husky A200 four-wheel robot equipped with a Mini-ITX computer and a Velodyne VLP-16 3D LiDAR. The onboard system is running Ubuntu 20 operating system and ROS 2 Foxy. The robot is tele-operated using a Sony Playstation 4 joystick. For the AR-HMD, We use the Microsoft HoloLens 2 (HL2).

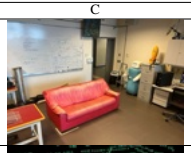
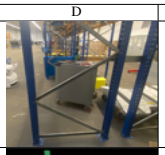
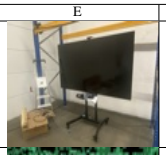
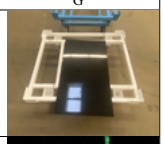
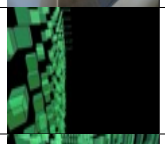
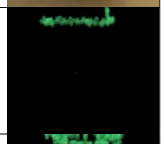
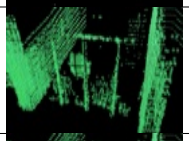
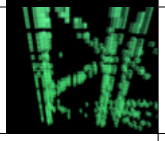
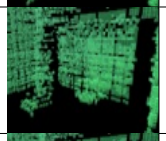
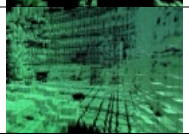
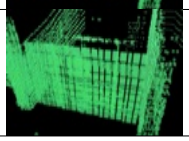
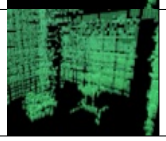
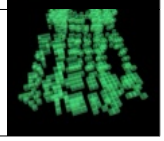
The 3D representation of the mapped floor shown in Fig.3 was generated using the Leica BLK 360 laser scanner. Region 1, features a narrow hallway in which the robot cannot navigate, and region 2 references a featureless hallway with 2 glass walls facing each other. Regions 3 and 4 feature the Vision and Robotics Lab (VRL) and Mobile and Distributed Computing Lab (MDCL) respectively, in which problematic objects such as reflective objects and transparent obstacles were placed.

The experiment initiates from the VRL by tele-operating the Husky to navigate the entire floor. Simultaneously, the human agent wearing the HL2 maps the areas of interest. After mapping a substantial portion of the environment, the human agent triggers the alignment process using the $View\ Minimaps$ button found on the hand menu shown in Fig. 4 (a). The button renders minimized versions of the local maps for the human agent to roughly align as displayed in Fig. 4 (b). The $Send\ Twist$ button sends the transformation that results from the rough alignment to the merger in the central unit. After the central unit generates the merged map, the HL2 utilizes it for a 1:1 overlay visualization over the real environment using the $View\ Merged\ Map$ button. Overlaying the map allows the human agent to easily check and correct for any inaccuracies in the mapped environment in situ as shown in Fig. 4 (c).

The results are tabulated in table II. Fusing the unedited

TABLE II

MAPPING RESULTS. ROW ONE SHOWS THE GROUND TRUTH OF THE MAPPED AREA OF INTEREST. ROW TWO SHOWS THE RESULTANT 3D MAP FROM THE ROBOT AGENT. ROW THREE SHOWS THE MERGED MAP RESULTING FROM THE ROBOT AND THE AR-HMD MAPS. ROW FOUR SHOWS THE EDITED MAP IN APPLICABLE SCENARIOS. FOR VISUALIZATION PURPOSES, THE GROUND PLANE WAS CLIPPED FROM ALL THE MAPS WHILE SOME OF THE WALLS WERE CLIPPED FROM IDEALWORKS MAPS.

	AUB			idealWorks			
	A	B	C	D	E	F	G
Ground Truth							
Robot Map							
Merged Map							
Edited Map	No Edits Needed		No Edits Needed	No Edits Needed		No Edits Needed	

local maps shows the ability to map the robot's unreachable area shown in Table II-A. Also, the robot map alone does not perfectly capture the environment features presented in Table II-C, unless merged with the HL2 map. The glass wall portrayed in Table II-B, was not captured by either the robot or the HL2. Therefore the human agent manually adds voxels for the glass wall using hand gestures. The edits appeared in place as shown in the edited map.

B. Experiment 2

In this experiment, we validate our results in Idealworks warehouse using the iw.hub Version 4 robot equipped with a LS CH32R 3D LiDAR running on Ubuntu 20 operating system and ROS 2 Foxy. The robot is tele-operated using a Logitech controller joystick. The HL2 is also used as the AR-HMD.

A similar testing procedure to that in Experiment 1 is applied. However, unique challenges are encountered in the industrial setting: Shelve supports are too narrow for the robot to map, which is easily repairable by the HL2 mapping, as depicted in Table II-D.

The TV stand is below the robot's LiDAR plane, and the HL2 cannot accurately map it due to its textureless black surface. Therefore, the human agent is required to manually edit the map for the stand to be captured as shown in Table II-E. Despite being inaccessible to the robot, the narrow area presented in Table II-F is completely mapped using the AR-HMD. Finally, the reflective plate in Table II-G is represented by neither the HL2 nor the robot map for being reflective and under the LiDAR's plane, which proved the necessity of manual human edits.

The experiments confirm the effectiveness of our proposed solution in generating accurate maps of the environment. Our future work will involve making the editing process more intuitive, combining the generated map with semantic data, and allowing the user to add, delete, and move objects. The user could then add points of interest such as charging stations or assign control points for improved navigation. The future steps will also concentrate on increasing scalability, allowing multiple agents to communicate together over large areas.

V. CONCLUSION

In this paper, we presented a collaborative 3D SLAM approach benefiting from the unique characteristics of 3 heterogeneous agents. The mobile robot and AR-HMD's local 3D maps were fused together by the central unit using ICP with a human-defined initial guess. The system offers the possibility to visualize the one-to-one scale merged map overlay-ed over the real environment, allowing the human agent to correct map inaccuracies in real-time. The proposed approach proved to increase the generated map's quality by correctly representing problematic and incomplete objects. The designed system also proved its efficiency in creating complete maps in scenarios where the robot is not allowed (or not able) to navigate certain areas. The proposed approach can be implemented with any mapping agent producing 3D point clouds.

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] 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.

[2] Y. Jiang, S. Yin, K. Li, H. Luo, and O. Kaynak, "Industrial applications of digital twins," *Philosophical Transactions of the Royal Society A*, vol. 379, no. 2207, p. 20200360, 2021.

[3] 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.

[4] R. Koch, S. May, P. Koch, M. Kühn, and A. Nüchter, "Detection of specular reflections in range measurements for faultless robotic slam," in *Robot 2015: Second Iberian Robotics Conference: Advances in Robotics, Volume 1*. Springer, 2016, pp. 133–145.

[5] P. Schmuck and M. Chli, "Multi-uav collaborative monocular slam," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 3863–3870.

[6] R. Dubé, A. Gawel, H. Sommer, J. Nieto, R. Siegwart, and C. Cadena, "An online multi-robot slam system for 3d lidars," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 1004–1011.

[7] J. Carmigniani and B. Furht, "Augmented reality: an overview," *Handbook of augmented reality*, pp. 3–46, 2011.

[8] N. V. Hofe, P. Sossalla, J. Hofer, C. L. Vielhaus, J. Rischke, J. Steinke, and F. H. P. Fitzek, "Demo: Robotics meets augmented reality: Real-time mapping with boston dynamics spot and microsoft hololens 2," in *2023 IEEE 24th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2023, pp. 358–360.

[9] Y. Sun, M. Liu, and M. Q.-H. Meng, "Improving rgb-d slam in dynamic environments: A motion removal approach," *Robotics and Autonomous Systems*, vol. 89, pp. 110–122, 2017.

[10] J. Cheng, Y. Sun, and M. Q.-H. Meng, "Improving monocular visual slam in dynamic environments: An optical-flow-based approach," *Advanced Robotics*, vol. 33, no. 12, pp. 576–589, 2019.

[11] R. Koch, S. May, P. Murmann, and A. Nüchter, "Identification of transparent and specular reflective material in laser scans to discriminate affected measurements for faultless robotic slam," *Robotics and Autonomous Systems*, vol. 87, pp. 296–312, 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0921889015302736

[12] Y. Zhu, J. Qiu, and B. Ren, "Transfusion: A novel slam method focused on transparent objects," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 6019–6028.

[13] J. Wu, J. Wang, and C. Wei, "Robust map merging method for collaborative lidar-based slam using gps sensor," *Journal of Physics: Conference Series*, vol. 2402, no. 1, p. 012004, dec 2022. [Online]. Available: https://dx.doi.org/10.1088/1742-6596/2402/1/012004

[14] Y. Tian, Y. Chang, L. Quang, A. Schang, C. Nieto-Granda, J. P. How, and L. Carlone, "Resilient and distributed multi-robot visual slam: Datasets, experiments, and lessons learned," *arXiv preprint arXiv:2304.04362*, 2023.

[15] Z. A. Al-Sabbag, C. M. Yeum, and S. Narasimhan, "Interactive defect quantification through extended reality," *Advanced Engineering Informatics*, vol. 51, p. 101473, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1474034621002238

[16] A. Sidaoui, I. H. Elhajj, and D. Asmar, "Collaborative human augmented slam," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 2131–2138.

[17] K. Koide, J. Miura, and E. Menegatti, "A portable three-dimensional lidar-based system for long-term and wide-area people behavior measurement," *International Journal of Advanced Robotic Systems*, vol. 16, 02 2019.

[18] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

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