2019 International Conference on Robotics and Automation (ICRA)
Palais des congres de Montreal, Montreal, Canada, May 20-24, 2019

A-SLAM: Human in-the-loop Augmented SLAM

Abbas Sidaoui, Mohammad Kassem Zein, Imad H. Elhajj, Daniel Asmar

***Abstract*— In this work, we are proposing an intuitive Augmented SLAM method (A-SLAM) that allows the user to interact, in real-time, with a robot running SLAM to correct for pose and map errors. We built an AR application that works on HoloLens and allows the operator to view the robot's map superposed on the physical environment and edit it. Through map editing, the operator can account for errors affecting real environment's representation by adding navigation-forbidden areas to the map in addition to the ability to correct errors affecting the localization. The proposed system allows the operator to edit the robot's pose (based on SLAM request) and can be extended to sending navigation goals to the robot, viewing the planned path to evaluate it before execution, and tele-operating the robot. The proposed solution could be applied on any 2D-based SLAM algorithm and can easily be extended to 3D SLAM techniques. We validated our system through experimentation on pose correction and map editing. Experiments demonstrated that through A-SLAM, SLAM run-time is cut to half, post-processing of maps is totally eliminated, and high quality occupancy grid maps could be achieved with minimal added computational and hardware costs.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) plays an important role in a number of indoor and outdoor applications. For instance, industrial robots rely heavily on SLAM to accomplish their tasks while navigating autonomously. Moreover, SLAM is used for tasks such as autonomous aerial surveillance [1-3], underwater reef monitoring [4,5] 3D reconstruction of underwater morphologies [6], and underground mine exploration [7,8]. SLAM can be implemented in 2D or 3D using sensors such as LIDARS or cameras. Although in this paper, we apply our solution using 2D SLAM, the proposed method is extendable to 3D.

The challenge is that SLAM estimates are affected by sensory errors, modeling errors, and map representation errors. While modeling errors are due to imperfections in the models used for the robot's dynamics and those used to model sensors, map representation errors are related directly to the environment being mapped. When talking about map representation errors, it is important to distinguish between two different types of errors: *those affecting the localization*, and *those affecting real environment's representation*. Noisy range measurements, changes in the environment being mapped, and uncertain pose estimates are possible causes of map errors that could affect localization. Fig. 1 shows an example of some conditions that lead to errors affecting the real environment representation. Objects of **Type 1** are

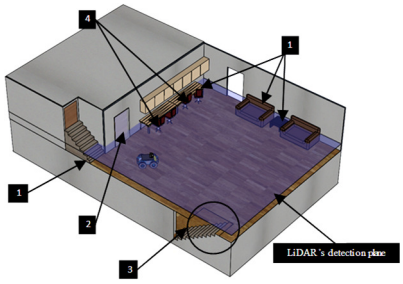
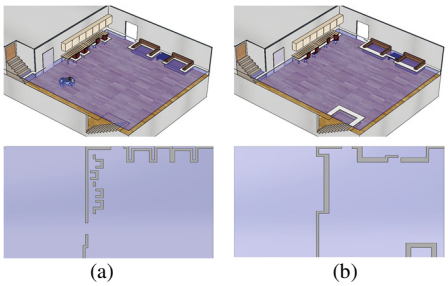


Figure 1. Conditions that affect the map representation

obstacles lower or higher than the LiDAR and thus can't be detected. **Type 2** objects are transparent obstacles, such as glass. **Type 3** objects represent unnavigable areas such as down stairs or through holes in the ground. Although not detecting these types of obstacles has little impact on the localization part of SLAM, the robot might consider such areas as acceptable for navigation, and the robot might plan a path through them. Finally, **Type 4** objects are obstacles that the user might want to include in the output map to steer the robot around them during path planning. Fig. 2 shows a comparison between the map obtained by a robot running SLAM in a sample environment (Fig. 2a) and a map that of the real environment, in which the four mentioned error types are taken into account (Fig. 2b). This sample scenario stresses on the importance of evaluating and assessing the quality of the robot's map, especially when the robot is operating in an environment that is shared with humans.

Traditionally, to ensure safe operations of a robot in critical environments, post-processing of the robot's map is applied to correct for errors [36]. Usually this procedure is time consuming and very challenging when the errors in the obtained map are numerous. In fact, it would be more efficient and more accurate if the operator could determine, in real-time, whether the robot's map and calculated pose match the actual location of both the robot and the obstacles, and could



(a) (b)

Figure 2. The Map obtained from traditional SLAM versus the map that reflects the real environment.

A. Sidaoui and I. H. Elhajj are with the Maroun Semaan Faculty of Engineering and Architecture, Electrical and Computer Engineering Department, American University of Beirut, 1107 2020, Riad El Solh, Beirut, Lebanon; email: ams108@mail.aub.edu ie05@aub.edu.lb. M. Kassem Zein and D. Asmar are with the Maroun Semaan Faculty of Engineering and Architecture, Mechanical Engineering Department, American University of Beirut, 1107 2020, Riad El Solh, Beirut, Lebanon; email: mhk50@mail.aub.edu da20@aub.edu.lb.

interact intuitively with these internal states in order to correct them.

Human-robot Interaction (HRI) combines human and robot perception in order to maximize the information provided to the human on one hand, and the feedback given to the robot on the other hand [10]. With the advancement in the field of Augmented Reality (AR), it is possible now to achieve more intuitive and user-friendly collaboration between humans and robots; thus increasing the efficacy and accuracy of robotic tasks.

The rest of the paper is as follows: Section II presents state-of-the-art augmentation techniques used in different SLAM algorithms and the advantages of Human-Robotic Collaborative systems. Section III discusses the OpenSLAM gmapping algorithm. Section IV gives an overview of our proposed system, while the proposed AR interface is presented in Section V. Section VI describes the implementation procedure. Finally, results of the conducted experiments are shown in section VII and we conclude our work in section VIII.

II. Background

In the literature, several methods were proposed to increase the accuracy of SLAM. Alsayed et al. [9] proposed two approaches based on Supervised Artificial Neural Networks to predict corrections to be applied on 2D based SLAM estimations. Their proposed methods require off-line training to predict the correction model. Hewitt and Marshall [10] presented an intensity-augmented SLAM approach where they used LiDAR and ToF sensor models in the sparse bundle adjustment (SBA) estimation problem. However, this method was evaluated in simulation and it takes into consideration Lambertian reflecting surfaces only. In monocular SLAM approaches, [11] introduced object detection to the system to fix scale ambiguity and drift problem using bundle adjustment (BA). BA was used in the work of [12-15] as the optimal solution for different SLAM problems. Although bundle adjustment outperforms filter based methods when it comes to accuracy [16], this accuracy comes at a high computational cost.

In the context of the localization problem in SLAM, a lot of methods were suggested to improve the robot pose estimation. In the work of [17], the absolute error between the true and predicted pose is bounded using overhead images from road networks, but, their technique needs special infrastructure and cannot be used in other sensing technologies such as 2D LiDAR. Recently, [18] presented improvement to the localization by bounding the robot pose to the geometry of a prior map. Despite that this system utilizes a map matching method that is easily integrated to a particle filter, a prior knowledge of the occupancy grid is needed.

Acquiring a collaborative system, where the efficiency and perception of human aids robots' tasks, has gained interest in the Human-Robot Interaction (HRI) research. For instance, Lin et. al [32] improved the productivity of a 6-DoF robot by combining the intellect of humans with the robot's flexible performance. In SLAM, [19-21] adopted the advantages of human robot collaboration to use it for map augmentation. They proved that the semantic information added by humans in the loop increased the total map accuracy. But, their methods required post processing and were not user friendly. However, in our previous work [36], we presented a human augmented mapping (HAM) interface. Our system proved superior over other methods by being real-time, user accessible, and more efficient in increasing map accuracy.

The fast emergence of Augmented Reality (AR) technology and its advancements are also promoting human-machine collaborative systems [22,23]. Clearly, this is significant through its ability for overlaying virtual graphics to the real world visualized by the user. For example, [24] proposed an interface which generates an augmented free-viewpoint images that increase the accuracy of robot teleoperation. As well, [25,26] investigated novel techniques that accomplish more intuitive robotics teleoperation using AR.

III. OpenSLAM Gmapping

OpenSLAM gmapping is a SLAM technique that applies Rao-Blackwellized particle filter (RBPF) to build grid maps from laser range measurements and odometry data [27]. RBPF was introduced first by Murphy et al. [28]. It consists of N particles in set R_t where each particle $R_t^{(i)} = (\mathfrak{s}_t^{(i)}, \eta_t^{(i)}, \mathrm{w}_t^{(i)})$ is presented by a proposed map $\eta_t^{(i)}$, pose $\mathfrak{s}_t^{(i)}$ inside this map, and an importance weight $\mathrm{w}_t^{(i)}$. OpenSLAM [27] can be summarized by five main steps: Pose Propagation, Scan Matching, Importance Weighting, Adaptive resampling, and Map Update.

Pose Propagation: at every iteration, odometry data u_{t-1} is used to propose the new poses from the previous pose $\mathfrak{s}_{t-1}$. Every particle $R_t^{(i)}$ is sampled from $R_{t-1}^{(i)}$ by giving it an initial pose $\mathfrak{s}'^{(i)}_t = \mathfrak{s}^{(i)}_{t-1} \oplus u_{t-1}$; $\oplus$ is the standard pose compound operator [29].

Scan Matching: for every new observation, z_t, scan-matching between z_t and the previous map η_{t-1} is applied and a score is given to this pose based on matching percentage. Each particle's pose is then adjusted within a predefined window till the maximum scan matching score is achieved for this single particle.

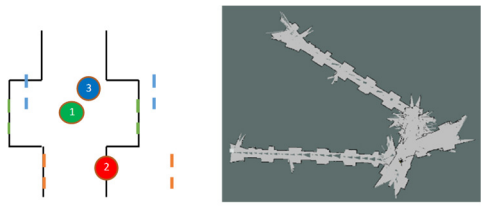
Importance Weighting: The observation model $p(z_t \,|\mathfrak{s}_t, \eta)$ uses observations z_t to calculate an importance weight $w_t^{(i)}$ for every particle according to the importance sampling principle shown in (1).

$$w_t^{(i)} = w_{t-1}^{(i)} \cdot p(\mathrm{z}_j |\, \mathfrak{s}_{t-1}^{(i)}, \eta_{t-1}) \quad (1)$$

$$\cong w_{t-1}^{(i)} \cdot \textstyle\sum_{j=1}^{K} p(z_t \mid \eta_{t-1}^{(i)}, \mathfrak{s}_j) \cdot p(\mathrm{z}_j |\, \mathfrak{s}_{t-1}^{(i)}, \eta_{t-1}) \quad (2)$$

Where K is the number of sampled points $\{x_j\}$ resulting from a Gaussian proposal for each particle around the pose obtained from scan matching.

Adaptive Resampling: This step is important for replacing particles of low importance weights by those of high weights.



(a) (b)

Figure 3. (a) Particle 1 is the correct pose. Although particle 3 is a better estimate of the pose, particle 2 will be given a higher weight. (b) A new map is formed as a result of odometry drift.

First, dispersion of the particles importance weights is measured through the effective sample size N_{eff}, see equation (3). In other words, N_{eff} could be used to assess how well the particles poses are close to the target posterior [27]. Then resampling is performed if N_{eff} drops below a pre-defined threshold (i.e. $\frac{N}{2}$).

$$N_{eff} = \frac{1}{\sum_{i=1}^{N}(\tilde{\omega}^{(i)})^2} \quad (3)$$

Where $\tilde{\omega}$ is the normalized importance weight of particle $R_t^{(i)}$.

Map update: Each map $\eta_t^{(i)}$ is updated based on the particle's new pose $z_t^{(i)}$ and observation z_t.

Although this method is known for its high accuracy, it is prone to failure in many scenarios. When navigating in areas with high symmetry levels such, scan matching might give high weights to incorrect pose estimates. This happens due to the fact that some obstacles look like those around the true pose, see Fig. 3a. Moreover, when scan matching score drops, odometry is used to estimate the new pose. In this case, if a slippage is introduced or odometry noise/drift is high, the algorithm will fail to find its way back to the true pose and will start overlaying a new map over the initial one. Here localization accuracy will be high at the local level, but wrong at the global level, see Fig. 3b.

IV. A-SLAM System Overview

In this section we will introduce our proposed A-SLAM system. Fig. 4 presents the flowchart of our methodology. Since blocks in black are adopted from OpenSLAM and already described in section III, we will only focus on discussing the Augmented Resampling and Augmented mapping blocks.

A. Augmented Resampling

This block presents our augmented pose correction method. It can be treated as adaptive human-in-the-loop resampling. The new augmented set of particles $A_t^{(i)} = (\acute{a}_t^{(i)}, \eta_t^{(i)}, w_t^{(i)})$ is sampled from $R_t^{(i)}$through the following steps:

1) Dispersion check: N_{eff} is calculated via equation (3). If $N_{eff} > \check{r}$ (where $\check{r}$ is a predefined resampling threshold) no augmented resampling will be performed, thus steps 2 to 5 are ignored.

2) Sampling from augmented pose: When $N_{eff} < \check{r}$ an augmented pose $\acute{a}_t^i$ is calculated for each particle:

$$\acute{a}_t^i = z_t^{(i)} \oplus d'^i_t \quad (4)$$

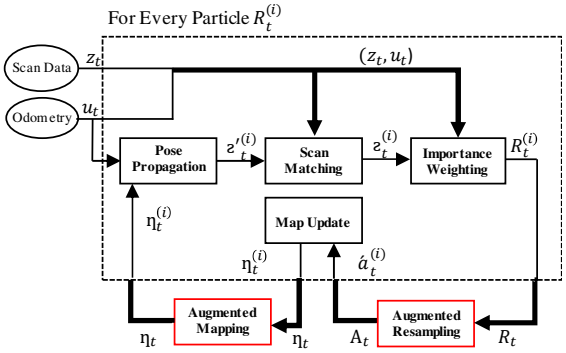


Figure 4. A-SLAM pipeline. Blocks in black are adopted from OpenSLAM and blocks in red represents our modifications.

Where d'^i_t is the pose difference vector between the desired human augmented pose and the pose of each particle. We then compute a Gaussian approximation $\acute{a}_t^i \sim \mathcal{N}(\mu, \Sigma)$ to propose the new particles distribution. This approximation is applied here since human correction itself would include errors. The mean μ of the distribution is equal to $\acute{a}_t^{(i)}$ and variance Σ is set to be (0.1m, 0.1m, 0.08rad). This value was obtained through experimentation.

3) Scan Matching: Like OpenSLAM, Vasco scan-matcher from CARMEN Toolkit [30] is implemented here to find the best pose that maximizes scan-matching score. For each particle, the algorithm searches for the best pose $\acute{a}_t^{*(i)}$ within a pre-defined window in map $\eta_{t-1}^{(i)}$ starting from the initial guess $\acute{a}_t^{(i)}$:

$$\acute{a}_t^{*(i)} = argmax\ p(\acute{a} \mid \eta_{t-1}^{(i)}, z_t, \acute{a}_t^{(i)}) \quad (5)$$

4) Importance Weighting: described in section III.

B. Augmented Mapping

In this work, we are considering Occupancy Grid Maps (OGM) only. This map representation technique was introduced first by Moravec and Elfes [31] and is known for its reduced complexity. In OGM, maps are formed out of cells with assigned occupancy probabilities where a probability between 0 and 0.5 means that the cell is free, probability between 0.5 and 1 means that it is occupied, and a probability of -1 means that this cell is still undiscovered. A detailed description of our methodology is presented in [36]. In summary, augmented mapping block receives the map of the best particle η_t^* as a 1D occupancy grid of size N:

$$\eta_t^* = \{n_1, n_2, n_3 \ldots . n_N\} \quad (6)$$

Every element n_i is assigned a value equal to the occupancy probability of the corresponding map's cell. Our proposed augmented mapping method allows the user to override the probability of any cell by toggling its status between free, occupied, and undiscovered.

V. AR User Interface

The developed AR interface allows the user to interact with SLAM intuitively -in real time- through Microsoft HoloLens (HL). In this work, we assume that the user has already calibrated the HL through the built-in inter-pupillary distance

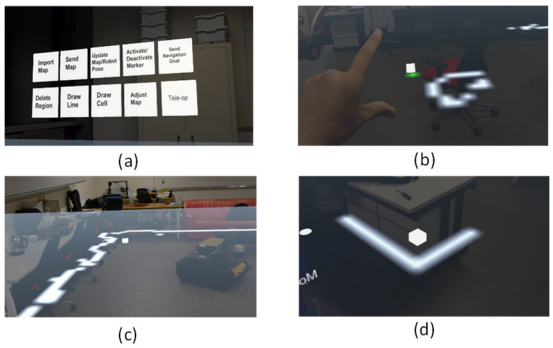


Figure. 5 A-SLAM user interface. (a) Presents the menu items, (b) shows the "tap" gesture,(c) and (d) show examples of map augmentation.

calibration routine. Fig. 5 shows real snaps of the user interface developed in Unity3D engine. It overlays a map on the LiDAR's plane of the robot and menu items are always rendered to the left of the user. In the augmented map, white cells are considered occupied and gray cells are considered free. Fig. 6 illustrates the interface alongside with the used frames and poses:

Map Frame M_F: Placed on the bottom left corner of the augmented map.

World Frame W_F: HL world frame, initialized and fixed by the HL when the application runs.

Map pose P_M: Pose of M_F in W_F, fixed at the initialization phase of the application.

Robot pose P_R: Pose of the LiDAR in P_M, acquired from the SLAM algorithm and updated in real-time.

HoloLens pose P_H: Pose of the HL inside the world frame, acquired through the HL built-in visual SLAM.

Gaze point pose P_G: the intersection between the viewing ray (gaze line) and the 3D augmented model. Gaze line vector is acquired from the HL itself in real-time.

Through our interface, the augmented map can be edited by adding or deleting occupied cells. Menu items manage the tasks inside the application and handle sending/receiving requests and messages to/from SLAM algorithm. Below are the buttons description and functionalities:

Import Map: sends a request to SLAM algorithm to get the latest map available. The map is received as a 1D occupancy grid array η_t^* of size N alongside with the width W, height H, and cell's resolution я of the 2D grid map. We are converting η_t^* to a 2D grid map and visualizing it as an image plane of $W \times H$ pixels, where each pixel represents a cell. The plane's dimension is $(W \times я)$ by $(H \times я)$.

Activate/Deactivate Marker: Activate and deactivate AR marker recognition and tracking. A predefined AR marker must be placed on the top of the LiDAR; it is used to initialize the map's pose inside the world frame. Vuforia SDK for Unity is used for AR marker recognition and tracking. When the Marker recognition is enabled, and the marker is in the view field of the HoloLens, the system waits for the most recent $\mathbf{P_R}$ then it calculates the map's position in the world frame. To account for localization drifting or lost tracking in HoloLens' visual SLAM, the user can re-initialize the map's pose at any time. Marker deactivation capability was added because we noticed instability and drifting in the HL visual odometry when keeping the marker tracking enabled.

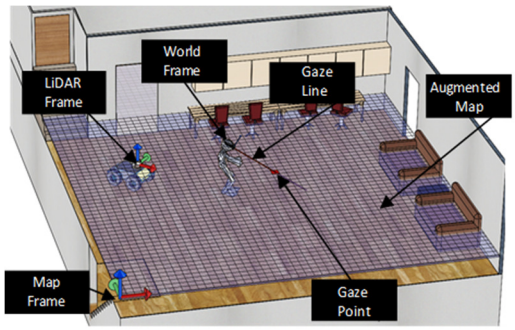


Figure. 6 AR interface components and reference frames

Adjust Map: allows for manual adjustment of the map's pose by applying translations (orientation adjustment is done through the AR Marker). The user has to click on a reference cell in the map, then click on the true position of that cell (considering the physical world).

Draw Line: enables a set of cells that form a single straight line to be set as occupied or free. The user has only to click on the starting and end points of the line.

Delete region: A rectangular region could be selected by clicking on two cells that form the diagonal of the desired rectangle. All cells within this area would be set to free.

Send Map: sends the human augmented map to the SLAM algorithm. This is done through converting the 2D pixels array of the image plane back to 1D occupancy array η_t^*.

Interaction with augmented objects is done through the HL-supported "tap" gesture. When looking on the augmented menu items, a round cursor is rendered on the intersection point $\mathbf{P_G}$. However, to facilitate the map editing procedure, if the gaze line happens to intersect with the map plane, $\mathbf{P_G}$ is rounded to the nearest cell's (pixel) center and a cube is rendered inside that cell of interest.

VI. Implementation

The proposed framework was implemented using Robotics Operating System (ROS) Indigo distro and Unity3D 2017.3.1f1. A detailed flowchart of the software architecture is presented in Fig. 7. Our implementation consists of 3 main nodes:

A. *SLAM Node:*

This node runs on a Lenovo M93P 10A8-A0PW00 mini workstation with Ubuntu 14.04 operating system. The proposed augmented resampling and augmented mapping methods are integrated in Gmapping, a ROS package that implements OpenSLAM's Gmapping [33]. SLAM node sends/receives the best particle's map and pose to the Supervisor node through RosBridge_suite [34] ROS package. Pose correction is handled through Rviz in this node too.

B. *Supervisor Node:*

This node handles the communication between ROS and our HoloLens' AR application. Map and pose messages are received from SLAM node through RosbridgeLib that uses RosBridge Protocol [36] to communicate with ROS JSON

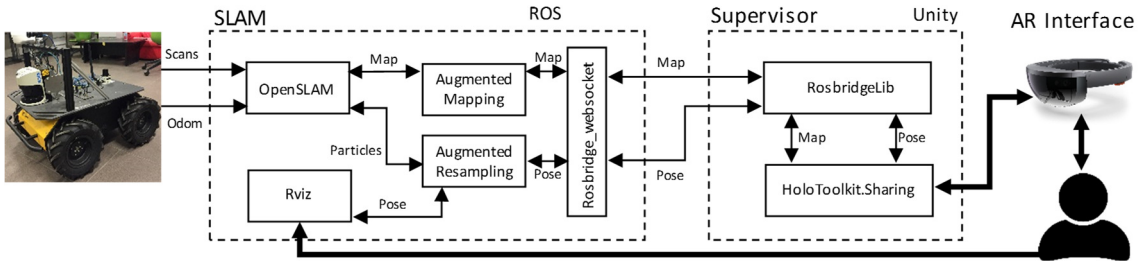


Figure 7. A-SLAM System pipeline

messages. The Supervisor node sends and receives messages from HL through HoloToolkit.Sharing library [35].

C. AR User Interface

The user interface was developed in Unity3D software and deployed on Microsoft HL headset.

VII. Experimentation and Results

Testing was done in an indoor-open environment on the sixth floor of Ray R. Irani Oxy Building at the American University of Beirut. Fig. 8 shows the blueprint with images reflecting the real area. The testing environment was chosen so that it lacks semantic features and glass walls cover a major part of the structure. This introduces challenges to traditional 2D grid-based SLAM techniques i.e. OpenSLAM. The Clearpath Husky A200 Explorer robot, with a 2D LiDAR mounted on it and wheel encoders installed, was utilized during the whole experimentation process. In the first set of experiments, the robot was teleoperated using a joystick. The operator had to cover a distance of about 170 meters while traversing the same trajectory through all tests. The path that should be traversed by the robot is shown in Fig. 8, shaded areas are not to be mapped. The second set of experiments was conducted to evaluate the AR user interface and its features. In all experiments, the rate of map update was set to 0.5 second and map resolution was 0.1m.

Pose Correction ExperimentsThis set of experiments was performed to assess correcting the robot's pose on the overall accuracy of the resulting SLAM maps. Several tests were done while varying lighting condition, number of particles in the RBPF, and bound of laser ranges. Table I documents the different sets of experiments with their corresponding parameters. Each set of tests was run three times resulting in a total of 3×2×3×3=54 maps. Obtained maps were analyzed by overlaying the ground truth map on the resulting map and visually evaluating the geometric matching. Through the procedure, we chose best and worst maps from each set for further discussion. Furthermore, we conducted three experiments with our pose correction method. These experiments were run using parameters that resulted in worst map representation for each set.

Laser range was the lowest in Set A. This increased the effect of odometry drift especially in **region 2** where detection of indoor obstacles was unattainable and thus, scan matching failed rapidly. Best map obtained in this set is shown in Fig. 9 (a) and the worst is shown in Fig. 9b. Fig. 9c shows the result of the map using our proposed framework. Overlaying the ground truth map on our resulted map Fig. 9d shows the effectiveness of our approach. In Sets B and C, the laser range was increased to mid-range and high-range respectively. The higher range was chosen to make sure that the robot detects the inner walls while navigating in **region 2**. Fig. 10a and Fig. 11a both show an improvement over the best map that could be achieved through OpenSLAM. However, this improvement comes at a higher computational cost. The worst maps obtained using both ranges are also documented, see Fig. 10b and Fig. 11b. Maps built using our system are displayed in Fig. 10c and Fig. 11c and the results of our system is presented in Fig. 10d and Fig. 11d. Through these experiments we demonstrated that including a human in the loop and correcting the robot pose in real-time would result in higher quality maps even in challenging areas. However, even with pose correction, post-processing of maps would still be required if no map correction is done simultaneously. Below are some quantitative results:

1) Run time: Tests with pose correction took **35%-50% less time** than those performed with OpenSLAM. This is due to the fact that when applying pose correction throughout the run, there is no need to revisit areas and check for loop closures.

2) Computational Costs: Since no computational complexity is added by our proposal, no significant added computational costs were recorded.

3) Success rate: we define success rate as the ability to get a high quality map with no areas overlaying on each other. When using OpenSLAM gmapping, 31 out of 54 runs produced useful maps- that definitely need post processing- resulting in **57.4%** success rate. Through our framework, we were able to produce better quality maps in **100%** of the tests.

4) Human workload: Throughout tests with pose correction, the operator needed to change the robots pose **9.5%** of the times where N_{eff} dropped. **91.5%** of the time the user had only to confirm the robot's pose estimation. Indeed, this requires much less workload if compared with the large effort needed in post processing on OpenSLAM results.

For testing the AR system, a map of the Vision and Robotics Lab (VRL) at AUB was built using OpenSLAM Gmapping. The time needed for correction using HL was approximately 6 minutes, whereas it took 30 minutes to post process the map using GIMP software, and it was less accurate. Furthermore, we demonstrated sending navigation goals to the robot with and without adding a forbidden area, see Fig. 12 (a) and Fig. 12 (b). Fig. 12 (c) presents the corrected map in real-time.

TABLE I. EXPERIMENTATION SETS AND THEIR PARAMETERS

Experiment	A			B			C		
Laser Range (m)	5.5 - 6.1			7.5 - 8.1			9.5 - 10.1		
Lighting	D		N	D		N	D		N
Number of particles	5	50	100	5	50	100	5	50	100

D: Tests performed in day light, N: Test performed at night

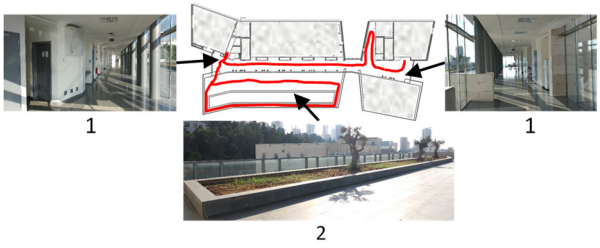


Figure 8. Testing Area at the American University of Beirut

VIII. CONCLUSION

In this paper we presented A-SLAM, a human-in-the-loop augmented SLAM framework. The proposed system allows real-time pose and map correction, and can be easily integrated in any 2D grid based SLAM technique. Pose correction is triggered by the SLAM algorithm and corrections are acquired through RViz. We also developed an intuitive AR interface for Microsoft HoloLens. Our interface overlays the SLAM's map on the real physical world so that the user can intuitively compare position of obstacles presented with the obstacles in the physical environment. The user can interact with this superposed map by adding or deleting obstacles, and adding navigation-forbidden areas. Through pose correction experimentation, we demonstrated that our system is able to produce more accurate maps in less number of trials and less run-time for each trial. A map obtained from one run has higher quality than the best map obtained from 54 runs using the traditional method. This was achieved with no additional hardware or significant computational cost. Moreover, the introduced AR interface increased the accuracy and efficacy of map editing and eliminated the post-processing task.

IX. ACKNOWLEDGMENT

This research was funded and supported by the AUB University Research Board.

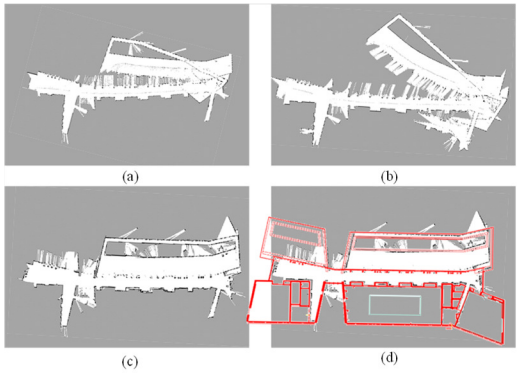


Figure 9. Maps obtained using set A

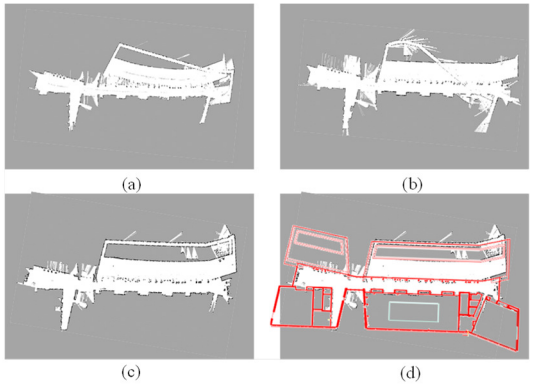


Figure 10. Maps obtained using set B

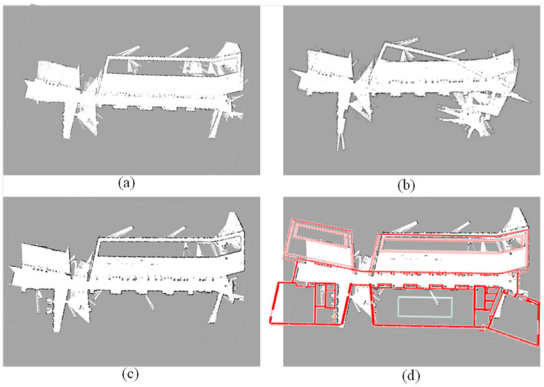


Figure 11. Maps obtained using set C

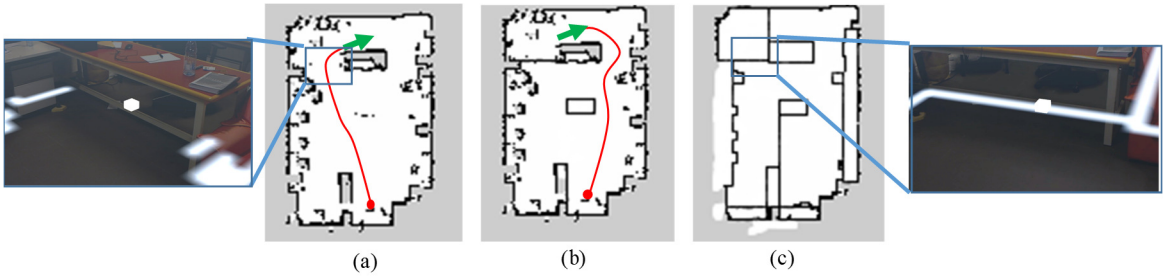


Figure 12. AR Interface test. (a) is the map obtained from OpenSLAM, the green arrow is the sent goal, while red line is the path planned by move_base ROS package. (b) result of map correction in real-time, after adding the obstacle, the path planner changed its navigation plan. (c) shows the result of correcting the map and adding forbidden areas in real time through HL.

REFERENCES

[1] J. Engel, J. Sturm, and D. Cremers, "Accurate figure flying with a quadrocopter using onboard visual and inertial sensing," *Imu,* vol. 320, p. 240, 2012.

[2] Z. Li, Y. Liu, R. Walker, R. Hayward, and J. Zhang, "Towards automatic power line detection for a UAV surveillance system using pulse coupled neural filter and an improved Hough transform," *Machine Vision and Applications,* vol. 21, no. 5, pp. 677-686, 2010.

[3] E. Semsch, M. Jakob, D. Pavlicek, and M. Pechoucek, "Autonomous UAV surveillance in complex urban environments," in *Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology-Volume 02*, 2009, pp. 82-85: IEEE Computer Society.

[4] N. Fairfield, G. Kantor, and D. Wettergreen, "Real-time SLAM with octree evidence grids for exploration in underwater tunnels," *Journal of Field Robotics,* vol. 24, no. 1-2, pp. 03-21, 2007.

[5] S. B. Williams *et al.*, "Monitoring of benthic reference sites: using an autonomous underwater vehicle," *IEEE Robotics & Automation Magazine,* vol. 19, no. 1, pp. 73-84, 2012.

[6] C. Beall, B. J. Lawrence, V. Ila, and F. Dellaert, "3D reconstruction of underwater structures," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, 2010, pp. 4418-4423: IEEE.

[7] A. Nuchter, H. Surmann, K. Lingemann, J. Hertzberg, and S. Thrun, "6D SLAM with an application in autonomous mine mapping," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, 2004, vol. 2, pp. 1998-2003: IEEE.

[8] S. Thrun *et al.*, "Autonomous exploration and mapping of abandoned mines," *IEEE Robotics & Automation Magazine,* vol. 11, no. 4, pp. 79-91, 2004.

[9] Z. Alsayed, G. Bresson, A. Verroust-Blondet, and F. Nashashibi, "2D SLAM correction prediction in large scale urban environments," in *ICRA 2018-International Conference on Robotics and Automation 2018*, 2018.

[10] R. A. Hewitt and J. A. Marshall, "Towards intensity-augmented SLAM with LiDAR and ToF sensors," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015, pp. 1956-1961: IEEE.

[11] D. P. Frost, O. Kähler, and D. W. Murray, "Object-aware bundle adjustment for correcting monocular scale drift," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, 2016, pp. 4770-4776: IEEE.

[12] L. Zhao, S. Huang, L. Yan, J. J. Wang, G. Hu, and G. Dissanayake, "Large-scale monocular SLAM by local bundle adjustment and map joining," in *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*, 2010, pp. 431-436: IEEE.

[13] A. Salehi, V. Gay-bellile, S. Bourgeois, N. Allezard, and F. Chausse, "Large-scale, drift-free SLAM using highly robustified building model constraints," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, 2017, pp. 1586-1593: IEEE.

[14] X. Zuo, X. Xie, Y. Liu, and G. Huang, "Robust visual SLAM with point and line features," *arXiv preprint arXiv:1711.08654,* 2017.

[15] L. Zhao, S. Huang, and G. Dissanayake, "Linear MonoSLAM: A linear approach to large-scale monocular SLAM problems," in *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, 2014, pp. 1517-1523: IEEE.

[16] H. Strasdat, J. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 2657-2664: IEEE.

[17] A. Napier, G. Sibley, and P. Newman, "Real-time bounded-error pose estimation for road vehicles using vision," in *Intelligent Transportation Systems (ITSC), 2010 13th International IEEE Conference on*, 2010, pp. 1141-1146: Citeseer.

[18] A. R. Romero, P. V. Borges, A. Pfrunder, and A. Elfes, "Map-Aware Particle Filter for Localization."

[19] E. A. Topp and H. I. Christensen, "Tracking for following and passing persons," in *IROS*, 2005, pp. 2321-2327.

[20] S. Kim, H. Cheong, J.-H. Park, and S.-K. Park, "Human augmented mapping for indoor environments using a stereo camera," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, 2009, pp. 5609-5614: IEEE.

[21] A. Diosi, G. Taylor, and L. Kleeman, "Interactive SLAM using laser and advanced sonar," in *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, 2005, pp. 1103-1108: IEEE.

[22] T. Dias, P. Miraldo, N. Gonçalves, and P. U. Lima, "Augmented reality on robot navigation using non-central catadioptric cameras," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015, pp. 4999-5004: IEEE.

[23] W. Hoenig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian, "Mixed reality for robotics," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, 2015, pp. 5382-5387: IEEE.

[24] F. Okura, Y. Ueda, T. Sato, and N. Yokoya, "Teleoperation of mobile robots by generating augmented free-viewpoint images," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, 2013, pp. 665-671: IEEE.

[25] K. Krückel, F. Nolden, A. Ferrein, and I. Scholl, "Intuitive visual teleoperation for UGVs using free-look augmented reality displays," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, 2015, pp. 4412-4417: IEEE.

[26] B. Vagvolgyi, W. Niu, Z. Chen, P. Wilkening, and P. Kazanzides, "Augmented virtuality for model-based teleoperation," in *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, 2017, pp. 3826-3833: IEEE.

[27] G. Grisetti, C. Stachniss, and W. Burgard, "Improved techniques for grid mapping with rao-blackwellized particle filters," *IEEE transactions on Robotics,* vol. 23, no. 1, pp. 34-46, 2007.

[28] K. P. Murphy, "Bayesian map learning in dynamic environments," in *Advances in Neural Information Processing Systems*, 2000, pp. 1015-1021.

[29] F. Lu and E. Milios, "Globally consistent range scan alignment for environment mapping," *Autonomous robots,* vol. 4, no. 4, pp. 333-349, 1997.

[30] M. Montemerlo, N. Roy, and S. Thrun, "Perspectives on standardization in mobile robot programming: The Carnegie Mellon navigation (CARMEN) toolkit," in Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on, 2003, vol. 3, pp. 2436-2441: IEEE.

[31] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, 1985, vol. 2, pp. 116-121: IEEE. [32] H.-

[32] C. Lin, Y. Fan, T. Tang, and M. Tomizuka, "Human guidance programming on a 6-DoF robot with collision avoidance," in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, 2016, pp. 2676-2681: IEEE.

[33] Gmapping, ROS package. Accessed August, 2018. [online] http://wiki.ros.org/gmapping

[34] Rosbridge_suite ROS package. Accessed July, 2018. [online] http://wiki.ros.org/rosbridge_suite

[35] HoloToolKit Sharing Library. Accessed May, 2018. [online] https://github.com/Microsoft/MixedRealityToolkit-Unity/tree/master/Assets/HoloToolkit/Sharing

[36] A. Sidaoui, I. H. Elhajj, and D. Asmar, "Human-in-the-loop Augmented Mapping," in *Intelligent Robots and Systems (IROS), 2018 IEEE/RSJ International Conference on*, 2018.