16-833 Project Final Report
Collaborative Heterogeneous-Sensor SLAM

Namya Bagree[1], Arjun Sengupta[1], Sourojit Saha[1], Hannah He[1] and Talha Faiz[1]

***Abstract*— Heterogeneous multi-robot SLAM tackles the problem of combining information from multiple sensors (of different types), from multiple robots, to build a map and localize the agents within it. Though several advancements have been made in multi-robot SLAM, as well as heterogeneous sensor SLAM (on a single robot), their combination is relatively unexplored and introduces new challenges. This project studies this problem, and explores a new method to combine maps from multiple robots, possibly with heterogeneous sensor data. The stack leverages a global alignment module and a map-merging module using the generalized-ICP algorithm, to combine pointclouds from different robots, and produce a common global map. We implement the system on real robots equipped with LiDAR sensors and monocular cameras. The system, when tested with LiDAR data produces promising results, and we explain our assumptions for it to work well with visual data.**

I. INTRODUCTION

The Robot Localization and Mapping problem on multiple robots (Collaborative-SLAM or C-SLAM) is a research area that has seen significant developments in recent times. C-SLAM aims to combine data collected by multiple robots, into globally consistent estimates of a common map and of each robot's state [6]. While this coordination between multiple robots enables more accurate SLAM, it is accompanied by several other challenges - most of which have ample descriptions and potential solutions in the literature. Current C-SLAM techniques primarily rely on combining maps of similar (homogeneous) sensor data. A problem not discussed too often is the presence of a framework to allow multiple robots with heterogeneous sensor capabilities to build and work on the same map. Little work has been done [1] - [5] on combining maps obtained from different types of sensors (heterogeneous data) in multi-robot SLAM. In this paper, we investigate the problem of C-SLAM with robots equipped with heterogeneous sensors.

We provide a detailed overview of the problem in the "problem definition" sub-section. We also highlight the potential applications of such a system - supporting the motivation for work in this area. In addition to an extensive literature review, we propose a system to tackle the problem of heterogeneous sensor C-SLAM, using map-alignment from point-cloud data. We also test the implementation of our system's important modules on actual robots, and report the results of the experiments conducted by us.

The rest of this paper is organized as follows: The "related work" section summarizes the prior-work in the area of multi-robot SLAM with heterogeneous sensors, in addition to a literature survey of heterogeneous sensor fusion, map-merging techniques, and C-SLAM in general. Apart from that, it also summarizes existing map-alignment techniques - since this is a vital part of the system we propose in a subsequent section. The "methodology" section summarizes our overall system, before elaborating upon the intricacies of each module. It also describes the baseline models that we have chosen to build our work upon, and the reasons for selecting these models. The "experimental set-up" section describes the hardware and software platforms used to conduct our experiments, along with the description of the robots' environment. Following this, the outcomes of our experiments are reported in the "Results" section. Lastly, we summarize our learnings in the "conclusion" and "future work" sections.

[1]All authors are students at Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, USA (nbagree, arjunsen, sourojis, hannahhe, mfaiz)@andrew.cmu.edu

A. Problem Description

Mapping using heterogeneous sensor data has several methods discussed in literature. However, sensor fusion for this purpose typically implies that different sensors are present on the same robot. Having different types of sensors on different robots introduces several additional challenges. Correlating heterogeneous sensor data from different robots becomes a huge challenge because of different odometries, with little knowledge of the relative transformation between the robots - for a general case. A formal definition of the problem is provided in [2], where it is defined as computing the joint posterior of relative transformations and merged-map. We summarize [2]'s formulation of the problem. They split the problem into two sub-problems: Map-matching, and map-merging. Mathematically, the map-fusion problem is written as:

$$p(M_r^{R_r}, T_{r,R_r} | m_r, m_{R_r}) = \underbrace{p(T_{r,R_r} | m_r, m_{R_r})}_{\text{map matching}} \cdot \underbrace{p(M_r^{R_r} | T_{r,R_r,m_r,m_{R_r}})}_{\text{map merging}}$$

Where $R = \{r_1, r_2..\}$ are the set of robots in the environment. $R_r = \{r_n, r_n \in R_r\}$ is defined as the neighbouring robots of robot r. m_r is the partial map generated by each by each individual robot r. Robot r receives the set of partial maps m_{R_r}, from nearby robots, where $m_{R_r} = \{m_{r_n}, r_n \in R_r\}$. $T_{r,R_r} = \{T_{r,r_n}, r_n \in R_r\}$ is the relative transformation between robot r and r_n. The global map is represented by $M_r^{R_r}$.

For our implementations we only consider two robots, and the corresponding two sensors being LiDAR and monocular

camera respectively. While one robot shall run a LiDAR-based SLAM algorithm, the other shall run a Visual-SLAM algorithm. The objective is to investigate the possibility of collaborative SLAM between these two robots.

B. Application Areas

Solving the problem of merging maps generated by two robots, wherein one uses LiDAR data, and the other uses another sensor (eg. visual) data - has tremendous potential. Such a system may have widespread use in search-and-rescue operations, and military applications. Specifically, a system of this type may useful in scenarios where:

1) Multiple robot-bases are readily available, but sophisticated sensors of a particular type are limited.
2) Sensors are damaged/ obstructed mid-operation and the robot/s can still execute the mission. The un-damaged sensors would yield heterogeneous map data, which shall be used to continue the C-SLAM process.
3) An agent can be localized within a previously generated map that was obtained by a different agent with a different type of on-board sensor.

II. RELATED WORK

Within SLAM - the subfields of Super Odometry, Intra-robot Loop Closures, and Pose Estimation have been active areas of research. In recent years, certain complete C-SLAM systems have been developed and deployed in realistic scenarios. For example, algorithms deployed in large-scale scenes during the DARPA Sub-T [23] [24] led to the developments of new C-SLAM systems, such as the robust lidar-based approach of [25]. For different modalities, [26] similarly proposes a vision-based centralized C-SLAM system incorporating inertial measurements, which has been tested with up to 12 robots in simulation. In another line of work, [27] presents a distributed and decentralized system robust to spurious measurements, along with online experiments on real robots, and a publicly available implementation. To compensate for system communication challenges, the authors of [28], put together a completed decentralized and distributed C-SLAM system including a novel robust distributed pose graph optimization back-end, and a front-end 10 producing globally consistent metric-semantic 3D mesh models of the explored environment. Those works are some of the best starting points for researchers and engineers looking to implement, improve and deploy practical C-SLAM systems in challenging conditions

Within the C-SLAM Front-End, the subfields of Direct and Indirect Inter Robot Loop Closures as well as Heterogeneous Sensing have been active areas of research. While Section I lists techniques such as LOAM, Super Odometry, and ORB to collect valid point cloud reconstructions, a problem not often discussed is the presence of a framework to allow multiple robots with heterogeneous sensor capabilities to build and work on the same map. [19] describes the use of the SegMap data structure for loop closure, determining map transforms, map matching, and map merging. A global target map is kept along with several partial maps from multiple robots and a GeometricConsistencyVerification test is performed on the feature vectors of each map (represented as segments). Thus, this centralized system efficiently updates and aligns partial maps for an accurate global map.

Within the C-SLAM Back-End, the subfields of Extended Kalman Filters, Particle Filters, Pose Graph Optimization, and Perceptual Aliasing Mitigation have been active areas of research. Heterogeneous sensing comes with the additional challenge of matching map data in different representation to perform relocalization and/or map fusion. There are also potentially unsolved problems with respect to various state of the art (SOTA) methods using different methods to solve the same problem.

1) [1] Describes a method to accomplish joint localization of multiple robots using distributed algorithms. Robots are equipped with various quality and types of sensors. The sensor data is fused locally and shared with a central server. The server then computes the optimal estimates of the poses of the robots.
2) [2] Proposes a probabilistic approach to integrate maps, which are independent of the type of sensor and the SLAM algorithm used. The problem was sub-divided into two parts, map matching and map merging. Relative transformation for map matching was obtained by applying Expectation Maximization to map matching. The maps were merged with a relative entropy filter to integrate the measurements of partial maps and decrease the uncertainty of the global map.
3) [5] Proposes a distributed algorithm amongst multiple robots to fuse any inter-robot measurement to reduce dead reckoning error. The problem is formulated as an optimization where the constraints are a manifold of lower dimension (Riemannian optimization). This performs better than the Extended Kalman Filter and the distributed pose graph method.

Within the System-Level challenges, the subfields of Map Representation and Communication Constraints have been active areas of research. While some early techniques simply share all the data from one robot to another, new sensors produce increasingly rich and dense data. The days of raw sensor data transmission are over and most current techniques in literature opt for some sort of compression or reduction. [3] surveys performances of several 3D keypoint detectors (Harris3D, ISS, KPQ, KPQ-SI, NARF), which are used to find keypoints matched between various sensor measurements and pointclouds for loop closure amongst multiple robots. If there are no computational restraints, KPQ-SI performs the best by finding the most keypoints and high repeatability. Otherwise, NARF allows for real-time performance and finds consistent, repeatable keypoints. For evaluating these techniques, papers like [4] sets upper bounds on the positioning accuracy of multiple robots (maximum expected localization uncertainty) in the context of cooperative localization. It is represented as a weighted connectivity graph representing the relative positions of the robot group.

III. METHODOLOGY

A. Overall Approach

Our system is broken-down into multiple submodules to tackle the problem of map fusion (described in the next section). Each robot will produce its own map through an existing SOTA SLAM algorithm - specific to the sensor that it is using. The robots shall send this data to a centralized fusion system in the form of point-clouds. After running a series of algorithms, the fusion system shall relay the fused map back to the robots, which would use the information for collaborative operations.

We make the following assumptions for our system to work well:

- The point-clouds produced by each of the individual robots should preferably be dense. The more dense the point clouds are, the better we expect our system to work.
- The region explored by one robot should partially overlap with the region explored by another robot, at least to a certain extent. Our system will fail if there is no overlap in explored regions among any robot

The following figure depicts the various modules of our system, for a specific case of two robots - one running visual SLAM, and the other using a LiDAR-based SLAM. Both robots produce point-clouds of the region that they are exploring (a considerable overlap is expected) and send it to the "global map alignment" module - implemented on a centralized base-station. This module estimates a relative transformation between the overlapping regions of the map, and sends this information to the "map-merging" module. The map-merging module also gets the original point-cloud information and fuses the two maps using the "generalized-ICP" algorithm. The fused maps are then sent from the base-station, back to the robots.

While this example depicts the "global-map alignment" and "map-merging" modules to be on a centralized base-station (as per our current implementation), we note that this can be extended to be implemented in a decentralized manner. The system may be part of the collaborative module on each robot itself. We elaborate on this in the "future-work" section.

B. Module-wise Description

1) Global Alignment Module: The goal of this module is to determine if two maps (possibly from multiple different sensors) align with each other, and to find the relative transformation between the two maps. To do this, we first convert the point-clouds obtained from the 2 robots, into 2D occupancy maps, in order to follow the method implemented in [19]. After obtaining the occupancy maps, the alignment process is followed - exactly as explained in [19]. Correspondences are obtained between the two maps using SIFT and SURF feature descriptors, and the similarity scores are calculated to infer matching points between the maps. These matches are then passed to the RANSAC module-which use the "multi-hypothesis RANSAC" algorithm as described in the cited paper. This is done to find the transform that would produce the most inliers between the two input maps. Features are extracted from the input and then sent to RANSAC to differentiate between inliers and outliers. Multiple possible transformations are maintained, where the probability of each possible transformation is represented as the sum of Gaussians. During RANSAC, the matching likelihood is optimized for the two correspondences iteratively until a set number of maximum iterations, before returning the transform associated with the largest weight SOG (Sum Of Gaussian). This entire process is done to obtain a transformation matrix between the maps - which is to be sent to the map-merging module.

2) Map Merging Module: We implement the generalized-ICP [18] algorithm for fusing the maps. It assumes an initial transformation between the overlapping regions of the provided point-clouds. The algorithm first computes correspondences between the two scans, and then computes a transformation which minimizes distance between corresponding points using Maximum Likelihood Estimation. It attaches a probabilistic model to the minimizations step for better estimation. These steps are performed iteratively until it converges to the desired transformation.

3) Iterative Merging: The map-merging step is performed iteratively when the two robots under consideration are estimated to be in similar positions and orientations. This iterative approach further refines the fused map. The estimation is done based on certain heuristics to determine the closeness between the robots. Specifically, our system checks whether the pose between the two robots, position as well as orientation, lie within a specific threshold. The time period between two merges is also maintained at a threshold to prevent excessive computation use. Further, the quality of each merge is measured using the mean squared distances between the two LiDAR scans, and if greater than a certain threshold, the parameters are reset and the merge transformation is shelved. Further smarter selection of LiDAR points based on the global map and location to get an accurate larger map is to be improved in a future setup. This would potentially help increase confidence in the iterative merges as time progresses and both robots explore larger regions of the environment.

4) Collaborative Module: This module is responsible for communication with the base-station and other peer robots. It sends the point-cloud information produced by the online SLAM system, to the base-station, and receives the fused map. It also sends its own odometry information to the other robots, and receives the odometry information from peer robots for collaborative operations. If our map-fusion is to be implemented in a decentralized manner the map-alignment and map-merging modules shall be implemented within the collaborative modules of each robot. The collaborative module can also be extended to relay other information- like robot health diagnostics, mission data etc. - depending on the specific task that the robot is working on.

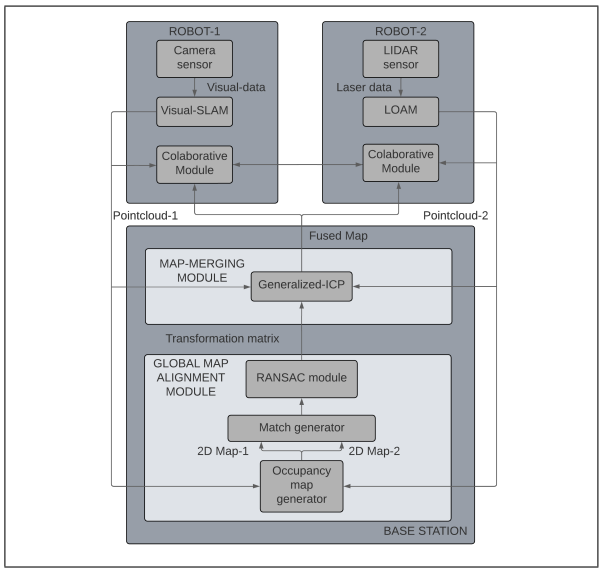

Fig. 1. Our framework

C. Baseline Models

1) LOAM: LOAM (LiDAR Odometry and Mapping in Real-time) [9] is among the best performing LiDAR odometry algorithms available. The LOAM ranks 3rd in the KITTI SLAM evaluation (best among the algorithms using only LiDAR). Hence, it was decided that we would begin with the open source version of LOAM, ALOAM(Advanced Implementation of LOAM) [10]. This algorithm is accomplished in two steps: (a) First, the high speed step running at 10Hz. This estimates the odometry and velocity of the LiDAR using low fidelity measurements. (b) Second, the low speed step runs at 1Hz. This step performs fine matching registration of point clouds and creates the map of the surrounding area.

Running two algorithms parallely allows the SLAM problem to be solved real time. For feature point extraction, only reliable points should be selected and this has been performed as follows: (a) Only those points were selected for which none of its surrounding points have been selected. (b) Points on a plane that are roughly on a plane parallel to LiDAR beams have been discarded. (c) The number of selected points within a sub region were not exceeded more than a threshold. The reported error is about 0.88% (0.55% on the KITTI website) of the distance traveled. This makes ALOAM a reliable and accurate SOTA LiDAR odometry algorithm.

2) ORB: We selected ORB-SLAM-3 our V-SLAM baseline because it is the SOTA method for V-SLAM, reporting best performance in accuracy and robustness. It outperforms other SOTA methods like SLAMM [11], VINS-MONO [12] and CCM-SLAM [13] according to the original paper [8]. Therefore, we believe that it will be widely used, and evaluating changes to such an algorithm could benefit the research community. This also uses the 'ORB' feature extractor, as does ORB-SLAM [14], ORB SLAM-2 [15] and ORB-SLAM Visual Inertial [16]. Our previously proposed hypothesis of the 'ORB' feature extractor being a potential bottleneck (given the emergence of newer deep-learning-based feature descriptors which are potentially faster) can potentially be tested on this algorithm. ORB-SLAM-3 in fact proposes 'Atlas' , a system to seamlessly fuse multiple maps. When a robot gets lost, it starts building a new map - which will be merged with the previous map while revisiting areas. This map-merging capability directly relates to our larger aim of heterogeneous map fusion. Though atlas is far from solving this problem, the very fact that it provides map-merging capabilities may serve as a starting point for our goal.

3) Super Odometry: Super Odometry is an IMU-centric SLAM system combining the advantages of tightly-coupled methods with loosely coupled methods. It is shown in [22] the improvements such a SLAM method can provide over LOAM based on accuracy, resilience, computational efficiency and extensibility. We make use of Super Odometry in our LiDAR framework due to the above mentioned advantages.

The code for our system formulation is made available on GitHub.[1]

IV. EXPERIMENTAL SETUP

A. Hardware

For this project, the robots used were "Traxxas remote control (RC) trucks". We use two of these robots, containing specially mounted payloads, comprising of Velodyne 16 LiDAR sensors, monocular cameras and an Epson g365 IMU. These robots are depicted in Figure 2. The LiDAR was mounted on top to get a higher perspective of the environment. For the purpose of collecting data, we controlled these robots remotely- one using an x-box controller, and the other through a laptop, which served as a local base-station.

Fig. 2. Hardware setup: robots used

B. Software

We use Robot Operating System (ROS) to program our system. It has most open source implementations. Most of our software stack was programmed in C++, and built on ROS. We use the C++ pointcloud library (PCL) to implement the generalized-ICP algorithm. Python was used to script modifying some of our rostopics for calibration and conversion of 3D scans to 2D for alignment testing. We also use the open source repositories of ALOAM [10] and the repository linked in paper [19] An attempt was made to

[1] https://github.com/sourojit-saha/Slam-Project

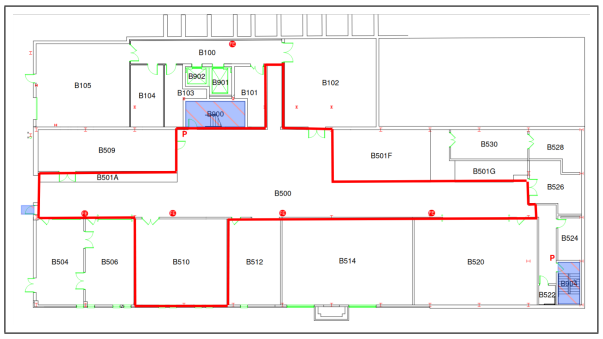

Fig. 3. Evironment floor plan

Fig. 4. Evironment view

make use of the official ORB-SLAM-3 repository, but this implementation was abandoned due to open issue with the codebase as mentioned in [17].

C. Environment Description

The robots were tested in Carnegie Mellon University's Newell Simon Hall (NSH), specifically - in the building's B-level. They were initially placed within a room, separated from one another by roughly a couple meters and 10-20 degrees angle orientation. This initialization would ensure a significant overlap between the initial maps generated produced by the two robots, in consonance with the assumption for our system to work well. The NSH B-level is a long corridor environment, extended into a parking lot. There are several obstacles in the scene, specifically boxes, chairs, tables and parked cars. These act as good feature points for both visual and LiDAR SLAM. We aim to test our system in that extend a long distance without losing features as it satisfies the way we want to constraint our environment. For this reason, this environment was selected for testing.

D. Experiment

Two RC trucks were used to collect the data. The data was collected in a tunnel environment. The RC trucks started from a similar location, separated roughly by a couple of meters from each other. Two operators controlled the RC trucks independently to move around and explore the environment.

1) Both vehicles begin exploring the region to their left.
2) Vehicle B decides to take an earlier turn and explore the regions on the opposite end of the environment.
3) Vehicle A moves to the other region after exploring its side of the map until it meets vehicle B.
4) Both vehicles turn and return to the starting position.

The data was collected by the RC trucks and transferred to a local machine for processing. The data was then used for simulating the RC trucks as simultaneously moving and Calibration algorithm was applied to create an aligned map.

V. RESULTS

The demo for our experiment can be seen in the video.[2]

A. Performance of Baseline Models

1) A-LOAM and Super Odometry on NSH Data: Based on the collected LiDAR data, two base SLAM implementations were tested to create the environment map. The maps created by A-LOAM and Super Odometry can be seen in Figures 5 and 6 respectively. The results given by Super Odometry were superior to the A-LOAM implementation. This can be attributed to two aspects, Super Odometry makes initial predictions at each step based on inertial data and Super Odometry is an IMU-centric SLAM built on the LOAM framework to improve it as shown in [22].

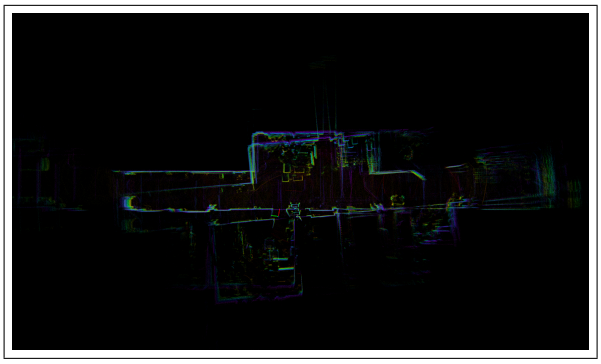

Fig. 5. Map created using LiDAR data through ALOAM

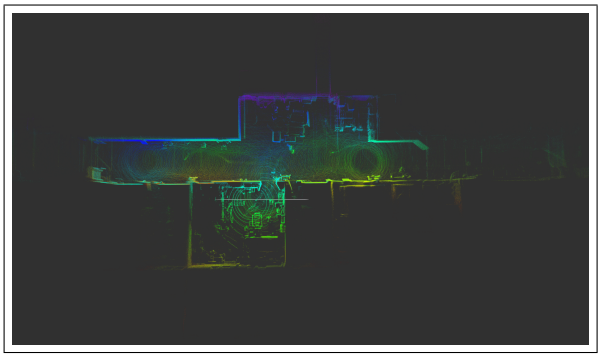

Fig. 6. Map created using LiDAR and IMU data through Super Odometry

2) ORB-SLAM on datasets: Based on the sensors available on the physical robots (i.e. monocular) visual-data was first collected from the KITTI, EuRoC, and NUance datasets. Then ORB-SLAM was implemented to create the environment map and can be seen in the figure below.

ORB-SLAM running on KITTI sequences 00 and 07 simultaneously. It took 562.2 seconds to get the final map

[2]https://www.youtube.com/watch?v=nB7C6FMvg0I

which contains 1824 keyframes, with translation error of 1% of trajectory's dimensions.

ORB-SLAM on the EuRoC MAV dataset took 374 seconds to get the final map consisting of 1200 features. The depth threshold was 3.852, Scale factor = 1.2, and the RGB color order was selected (even though the input was in grayscale).

B. Combining Point Clouds from Two LiDAR Sources

For the following sections, we made use of LiDAR maps created through Super Odometry to minimize drifts due to individual vehicle SLAM. As explained in Section III-B.2 (Map Merging Module), the maps were created through the two robot setup described in Section IV (Experimental Setup). Both vehicles start at a similar location in the environment. The timing at which each robot begins its operation doesn't affect our algorithm. The second robot is given a transformation (rotation and translation) based on the initial LiDAR points from the first robot. The Generalized-ICP algorithm has a few tunable parameters that include number of LiDAR scans, maximum iterations and max correspondence distance. A decent estimate from the start position or global alignment module is assumed, so the maximum iterations doesn't create a significant variation on the results. Max correspondence distance sets the maximum distance threshold between two corresponding points in source and target scans. This creates certain variations in map merging and the quality of the merge is measured with the fitness score (Mean squared error) as shown in Table I. The optimal maximum distance threshold is chosen through this analysis (5.0). Increasing the number of LiDAR scans increases computation time but also improves the point matching. However, if the robots are moving while the LiDAR scans are selected, outlying LiDAR points might get included that affect our map merging as shown in Figures 7 - 10. Here, comparisons between 1, 2, 5 and 10 frame overlaps are seen. The variation in quality between different numbers of frames while robots might not be stationary displays the importance of the iterative map merging to get a better understanding of the environment without spending as much compute on an alignment. The mean square error between LiDAR points is measured along with visual feature differentiations to compare the quality of the merged maps.

TABLE I

AVERAGE FITNESS VALUE VARIATION WITH MAXIMUM DISTANCE THRESHOLD OVER MULTIPLE FRAME VALUES

Max. Dist. Threshold (m)	0.1	1.0	5.0	8.0
Average Fitness Score	0.46	0.42	0.27	0.29

C. Results With and Without Loop Merging

The importance of loop merging was explained in Section III-B.3 (Iterative Merging) and that can be validated through our results. As expected, the initial estimate of the robot locations through the starting position or global alignment process may or may not be very accurate. This directly affects the quality of the Map Merging Module that is performing a non-linear optimization over a significant number of LiDAR points. To get rid of such uncertainties and increase robustness to our system performance, the comparison results of a complete run of the two robots is seen in Figures 11 and 12. It can clearly be seen that features such as walls, doors, corners and corridors, especially in regions further from the initial position don't align as well without the Iterative Merging Module.

Fig. 7. Map created with 1 LiDAR frame matching

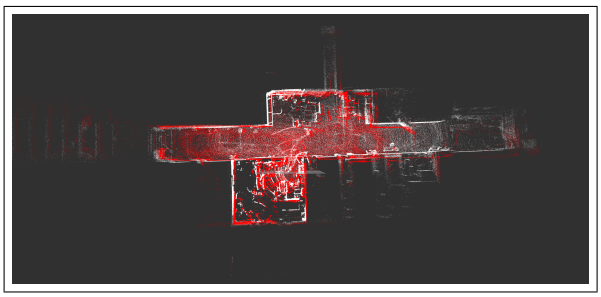

Fig. 8. Map created with 2 LiDAR frames matching

D. Global Feature Matching

As mentioned in Section III-B.1, a Global Alignment Module is required to handle cases when the robots start operations in highly varied locations that the other modules wouldn't be able to handle on their own. Feature matching among overlapping regions of the maps was used to help solve this. This was accomplished by dividing the map in small patches and finding similarity among the patches. The patches with the highest similarity value was used to calculate the transformation. Due to presence of noise and sparsity of the map used, false matches were detected and

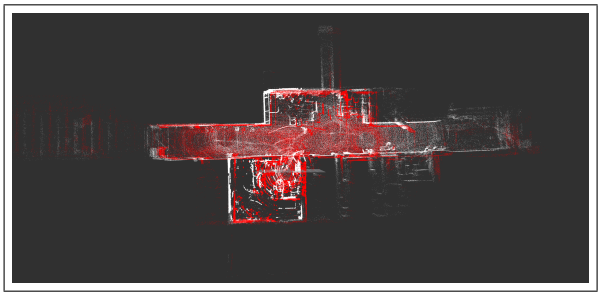

Fig. 9. Map created with 5 LiDAR frames matching

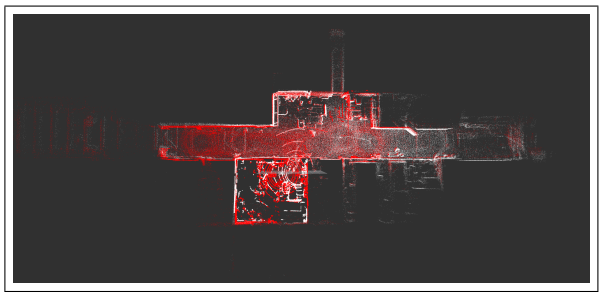

Fig. 10. Map created with 10 LiDAR frames matching

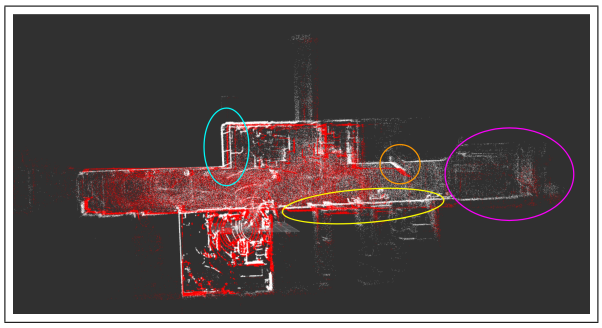

Fig. 11. Map created after one initial merge

affected the accuracy of our method. This could potentially be improved in the future with the help of a probability occupancy map. Nevertheless, on allowing the robots to move around the area and explore a significant region, we were able to get accurate global matching, that helped display the working of our pipeline. It can be seen in Figure 13 that partially overlapped maps are able to match features. Figure 14 displays the final overlapped maps, following the alignment steps.

E. Challenges

While the A-LOAM and Super Odometry using LiDAR data was easier to implement and could generate consistent point clouds. Implementing ORB-SLAM was challenging due to errors in compiling and motion blur leading to mixed pixels in the point clouds. ORB-SLAM remains the SOTA algorithm when working with visual data, and the discrepancies in the output are not indicative of the model's performance, but instead of broken code dependencies in publicly available repos which can be fixed with time.

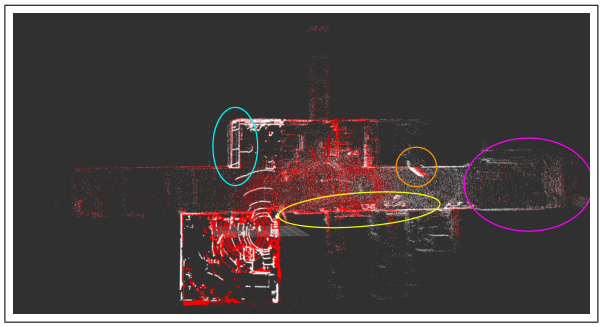

Fig. 12. Map created with iterative merging

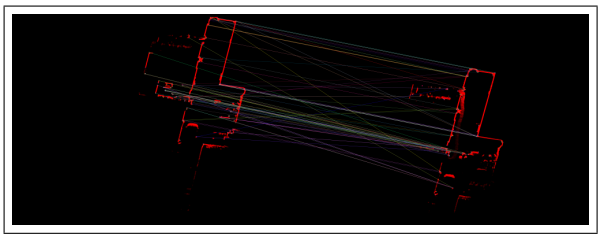

Fig. 13. Feature matching

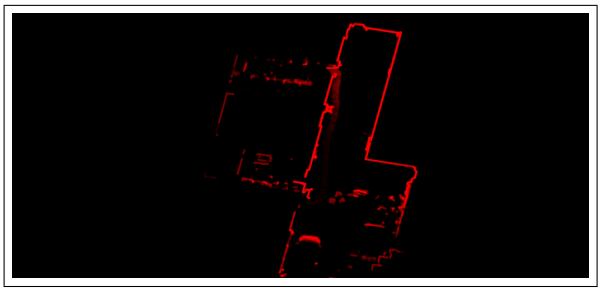

Fig. 14. Globally merged maps

During testing over datasets, some loops failed in the monocular camera case because the depth reconstruction is done using two consecutive frames. This is not very reliable. In order to get good depth reconstruction it is necessary that there should be some side shift between two frames. When one object can be observed in different locations in different pictures, depth calculation is more consistent. Because the robot is moving in a straight path, there is not enough side shift between consecutive images to recover good quality depth. This can be solved if stereo cameras were used instead because we get two images at every instant, and thus side shift information is available at every instance. To test this, the ORB-SLAM implementation was also tested over EuRoC's dataset. Because this dataset has both visual-inertial and stereo datasets for similar paths, the results were easy to compare.

VI. CONCLUSIONS

The goal of this research work was to enable the fusion of point clouds from two or more sensors, even if the sensors are fundamentally different, on different robots, and have a time-delay. Additionally, the baseline of this algorithm was set in place, it was important to test it on physical robots to test real-time performance and improve edge cases.

These objectives were achieved using two modalities – LiDAR data and Visual data. The C-SLAM algorithm was based on three baselines – ALOAM, Super Odometry and ORB-SLAM. An improved global alignment module and map merging module was implemented in the proposed framework. Experiments were done on physical robots and good performance benchmarks were observed. This direction of work allows for:

1) Modularity when multiple robot bases can be used while minimizing the need for sophisticated sensors (which are limited and sometimes expensive).

2) Sensors which get damaged/obstructed mid-operation to still execute the mission.
3) Localizing an agent within a previously generated map obtained by a different agent with a different type of on-board sensor.
4) Faster and more robust mapping and localization.

Some challenges were faced as discussed in section V-E, and based on our observations, this work can be extended in a few directions as elucidated in Section VII.

VII. FUTURE WORK

There are several other, more nuanced methods for map matching, merging, and loop closure of heterogeneous sensors on multiple robots that would be more efficient and effective than our current implementation. The expectation maximization approach [20] computes the joint posterior of relative transformations and merges the partial maps frame by frame, allowing for generalizability. Another method is the SegMap data structure [21] where feature vectors from each local partial map (from each robot) are compared with a global target map in the form of segments. Then, segment matching is performed using k-nearest neighbors and geometric consistency verification to verify the validity of the correspondence between two maps.

As for our current iterative merging module, we make use of LiDAR scans at a time when the robots satisfy certain conditions. This can be modified to only include a certain density of pointclouds at a location and to search over a larger region based on what has been explored previously by both robots.

ACKNOWLEDGMENT

We would like to thank Dr. Michael Kaess and the Teaching Assistants of the course 16-833 (Robot Localization and Mapping) at Carnegie Mellon University, for their constant guidance and support throughout the duration of this project.

REFERENCES

[1] T. Bailey, M. Bryson, H. Mu, J. Vial, L. McCalman and H. Durrant-Whyte, "Decentralised cooperative localisation for heterogeneous teams of mobile robots," 2011 IEEE International Conference on Robotics and Automation, 2011, pp. 2859-2865, doi: 10.1109/ICRA.2011.5979850.

[2] Y. Yue et al., "Multi-Robot Map Fusion Framework using Heterogeneous Sensors," 2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), 2019, pp. 536-541, doi: 10.1109/CIS-RAM47153.2019.9095798.

[3] E. R. Boroson and N. Ayanian, "3D Keypoint Repeatability for Heterogeneous Multi-Robot SLAM," 2019 International Conference on Robotics and Automation (ICRA), 2019, pp. 6337-6343, doi: 10.1109/ICRA.2019.8793609.

[4] A. I. Mourikis and S. I. Roumeliotis, "Analysis of positioning uncertainty in reconfigurable networks of heterogeneous mobile robots," IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004, 2004, pp. 572-579 Vol.1, doi: 10.1109/ROBOT.2004.1307210.

[5] J. Knuth and P. Barooah, "Collaborative localization with heterogeneous inter-robot measurements by Riemannian optimization," 2013 IEEE International Conference on Robotics and Automation, 2013, pp. 1534-1539, doi: 10.1109/ICRA.2013.6630774.

[6] Lajoie, Pierre-Yves, et al. "Towards Collaborative Simultaneous Localization and Mapping: a Survey of the Current Research Landscape." arXiv preprint arXiv:2108.08325 (2021).

[7] D. Villagrá Guilarte, 'Collaborative Localization and Mapping with Heterogeneous Depth Sensors', Dissertation, 2020.

[8] Campos, Carlos, et al. "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam." IEEE Transactions on Robotics 37.6 (2021): 1874-1890.

[9] Zhang, Ji and Sanjiv Singh. "LOAM: LiDAR Odometry and Mapping in Real-time." Robotics: Science and Systems (2014).

[10] ALOAM - an open source implementation of LOAM - https://github.com/HKUST-Aerial-Robotics/A-LOAM

[11] Daoud HA, Md. Sabri AQ, Loo CK, Mansoor AM (2018) SLAMM: Visual monocular SLAM with continuous mapping using multiple maps. PLoS ONE 13(4): e0195878. https://doi.org/10.1371/journal.pone.0195878

[12] Qin, Tong, Peiliang Li, and Shaojie Shen. "Vins-mono: A robust and versatile monocular visual-inertial state estimator." IEEE Transactions on Robotics 34.4 (2018): 1004-1020.

[13] Schmuck, Patrik, and Margarita Chli. "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams." Journal of Field Robotics 36.4 (2019): 763-781.

[14] Mur-Artal, Raul, Jose Maria Martinez Montiel, and Juan D. Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system." IEEE transactions on robotics 31.5 (2015): 1147-1163.

[15] Mur-Artal, Raul, and Juan D. Tardós. "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras." IEEE transactions on robotics 33.5 (2017): 1255-1262.

[16] Mur-Artal, Raúl, and Juan D. Tardós. "Visual-inertial monocular SLAM with map reuse." IEEE Robotics and Automation Letters 2.2 (2017): 796-803.

[17] Open Issue with ORB-SLAM-3 GitHub repository: https://github.com/UZ-SLAMLab/ORB_SLAM3/issues/403

[18] Segal, Aleksandr, Dirk Haehnel, and Sebastian Thrun. "Generalized-icp." Robotics: science and systems. Vol. 2. No. 4. 2009.

[19] Blanco, Jose-Luis, Javier González-Jiménez, and Juan-Antonio Fernández-Madrigal. "A robust, multi-hypothesis approach to matching occupancy grid maps." Robotica 31.5 (2013): 687-701.

[20] Yue, Yufeng, et al. "Multi-Robot Map Fusion Framework using Heterogeneous Sensors." 2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM). IEEE, 2019.

[21] Barbas Laina, Sebastián. "Heterogeneous collaborative SLAM: localization between camera and depth sensors." (2021).

[22] Zhao, Shibo, et al. "Super odometry: IMU-centric LiDAR-visual-inertial estimator for challenging environments." 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). IEEE, 2021.

[23] Hudson, N., Talbot, F., Cox, M., Williams, J., Hines, T., Pitt, A., Wood, B., Frousheger et. al. (2021). Heterogeneous Ground and Air Platforms, Homogeneous Sensing: Team CSIRO Data61's Approach to the DARPA Subterranean Challenge. Submitted to the DRC Finals Special Issue of the Journal of Field Robotics.

[24] Agha, A., Otsu, K., Morrell, B., Fan, D. D., Thakker, R., Santamaria-Navarro, A., et al. (2021). NeBula: Quest for Robotic Autonomy in Challenging Environments; TEAM CoSTAR at the DARPA Subterranean Challenge. Accepted for publication in the Journal of Field Robotics, 2021.

[25] Ebadi, K., Chang, Y., Palieri, M., Stephens, A., Hatteland, A., Heiden, E., Thakur, A., Funabiki, N., Morrell, B., Wood, S., Carlone, L., and Agha-mohammadi, A.-a. (2020). LAMP: Large-Scale Autonomous Mapping and Positioning for Exploration of Perceptually-Degraded Subterranean Environments. arXiv:2003.01744.

[26] Schmuck, P., Ziegler, T., Karrer, M., Perraudin, J., and Chli, M. (2021). COVINS: Visual Inertial SLAM for Centralized Collaboration. In 2021 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct), pages 171–176.

[27] Lajoie, P.-Y., Ramtoula, B., Chang, Y., Carlone, L., and Beltrame, G. (2020). DOOR-SLAM: Distributed, Online, and Outlier Resilient SLAM for Robotic Teams. IEEE Robotics and Automation Letters, 5(2):1656–1663.

[28] Tian, Y., Chang, Y., Arias, F. H., Nieto-Granda, C., How, J. P., and Carlone, L. (2021). Kimera-Multi: Robust, Distributed, Dense Metric-Semantic SLAM for Multi-Robot Systems. arXiv:2106.14386.