Coxgraph: Multi-Robot Collaborative, Globally Consistent, Online Dense Reconstruction System

Xiangyu Liu, Weicai Ye, Chaoran Tian, Zhaopeng Cui, Hujun Bao and Guofeng Zhang*

***Abstract*— Real-time dense reconstruction has been extensively studied for its wide applications in computer vision and robotics, meanwhile much effort has been made for the multi-robot system which plays an irreplaceable role in complicated but time-critical scenarios, e.g., search and rescue tasks. In this paper, we propose an efficient system named *Coxgraph* for multi-robot collaborative dense reconstruction in real-time. In our system, each client performs volumetric mapping in a producer-consumer manner. To facilitate transmission, we propose a compact 3D representation which transforms the SDF submap to mesh packs. During the recovery of submaps from mesh packs, the system can perform loop closure outlier rejection based on geometry consistency, trajectory collision and fitness check. Then we develop a robust map fusion method through joint optimization of trajectories and submaps. Extensive experiments demonstrate that our system can produce a globally consistent dense map in real-time with less transmission load, which is available as open-source software [1].**

I. INTRODUCTION

Reconstructing dense volumetric scenes is an important task in the fields of computer vision and robotics, with many applications in factory automation, search & rescue, augmented reality [1], cultural heritage preservation [2], and city modelling. Although existing single robot reconstruction systems [3], [4] have shown their good performance on online localization and mapping, they are still difficult to be applied in large-scale scenarios such as city-level reconstruction and time-critical scanning. In view of this, we highlight that multi-robot collaborative dense reconstruction deserves more attention as they permit rapid exploration and higher redundancy than a single-robot system.

Recently, some multi-robot Simultaneous Localization and Mapping (SLAM) [5], [6] systems based on vision or LiDAR have emerged and shown state-of-the-art performance on inter-robot localization. However, most of them focus on the improvement of localization and optimization of camera poses, instead of dense mapping results. The major challenge in practice is to exchange dense mapping data with practical bandwidth usage. While most of existing multi-robot reconstruction methods try to perform dense reconstruction, they either assume high-quality network connection among clients and focus more on task distribution and collaborative path planning [7], or only optimize and transmit trajectories [5], [8], then adjust only local dense maps [9]. Note that none of them can achieve a globally consistent multi-robot dense reconstruction with limited transmission bandwidth.

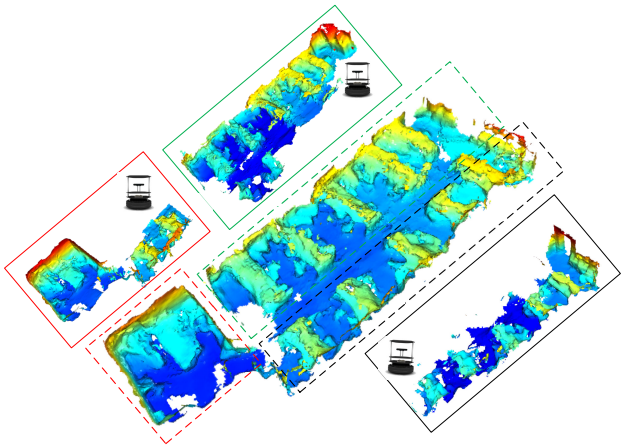


Fig. 1: A multi-robot reconstruction result of CVG lab. The center picture shows the global mesh generated online by server, and the surroundings show meshes from three clients corresponding to different regions.

To address the aforementioned problem, we propose an efficient system for multi-robot collaborative dense reconstruction with compact transmission data format and optimization on submaps. Our proposed compact representation can replace the heavy Signed Distance Function (SDF) map with comparable accuracy, which can be quickly transmitted to the server for online optimization on dense maps, achieving global consistency.

In our system, each client performs visual SLAM and SDF reconstruction in a producer-consumer manner and integrate SDF into submap at a fixed frequency. We then transform the submap to a compact representation, named as *mesh pack* in this work, which can fulfill low bandwidth communication. Then we propose a trajectory query-based method to convert the mesh pack back into submaps in the server. Experiments show that our compressed transmission method decreases the accuracy of original map negligibly. Based on the recovered submaps, the system can perform SDF-based loop closure outlier rejection and map fusion optimization.

The contributions of this paper are summarized as follows:

- We propose an efficient system named *Coxgraph* for centralized multi-robot collaborative dense reconstruction in real-time.
- We present a compact transmission representation which

* Corresponding Author: Guofeng Zhang

This work was partially supported by the National Key Research and Development Program of China under Grant 2020AAA0105900, Zhejiang Lab (2021PE0AC01) and NSF of China (No. 61822310). All authors are with State Key Lab of CAD&CG, Zhejiang University, China. Emails: mike.liu@ntu.edu.sg; {yeweicai, tiancr, zhpcui, baohujun, zhangguofeng}@zju.edu.cn

[1]https://github.com/zju3dv/coxgraph

enables transmitting local 3D submaps with minimal bandwidth requirement.
- Our system achieves global consistency across robots, by extending online map fusion optimization and loop closure correction methods.

II. Related Work

We briefly review the related methods of multi-robot SLAM and multi-robot reconstruction in this section.

A. Multi-Robot SLAM

With the development of SLAM and multi-robot systems, the research on multi-robot SLAM has attracted more attention. CoSLAM [10] presents a collaborative monocular SLAM system to improve robustness in dynamic environments which relies on computation on GPU. As a centralized collaborative SLAM system, CVI-SLAM presents a novel visual-inertial framework [11] for each agent outsourcing its computationally expensive tasks and sharing all information with a central server. CCM-SLAM [12] presents a centralized collaborative monocular visual SLAM system, running on clients with small processing units. Each client only performs feature tracking and the server collects experiences of all clients, detects loop closures and optimizes the global poses. In contrast, CORB-SLAM [6] uses extended ORB-SLAM2 as clients, and detects loop closure and optimizes in server. Since its clients runs complete visual odometry, it requires more computation capacity than the former ones. Bartolomei et al. propose VINS-Client-Server as a module of the multi-robot navigation architecture [13]. VINS-Client-Server extends VINS-Mono [14] as the client frontend, while its pose graph backend collects keyframe data and detects loop closure, similar to CVI-SLAM and CCM-SLAM. DOOR-SLAM [8] develops a fully distributed SLAM system using NetVLAD [15] to detect loop closure with robust outlier rejection methods. It also introduces a communication procedure to reduce communication load by exchanging global NetVLAD keyframe descriptors first and transmitting the complete keyframe data only if their descriptors match.

B. Multi-Robot Reconstruction

CoScan [7] performs collaborative scanning for dense 3D reconstruction of unknown indoor environments with the focus on task distribution and path planning. Kimera-Multi [9] extends Kimera [16] with similar communication and outlier rejection methods as DOOR-SLAM [8] to develop a system for distributed multi-robot metric-semantic SLAM. They optimize distributed pose graph and adjust local deformed mesh. Based on Voxblox [17], Voxblox-Multi-Agent in [13] collects voxel-filtered compressed point clouds from clients and fuses them based on the poses of anchor keyframes. Note that systems mentioned above basically optimize the map by firstly optimizing the trajectory and then adjusting dense maps accordingly. In this work, our proposed multi-robot dense reconstruction system *Coxgraph* can exchange dense map data with minimal communication load, and optimize jointly on trajectory and submap poses, to realise online collaborative volumetric mapping with global consistency across robots.

III. System Overview

The architecture of our system, i.e. *Coxgraph*, is depicted in Figure 2. *Coxgraph* aims at transmitting dense reconstruction data at a minimal requirement of network traffic, and optimizing dense maps globally to get optimized and globally consistent reconstruction results.

A. Client

Each client runs modules of visual-inertial odometry and dense SDF reconstruction in a client-server manner. When executing a reconstruction task, the visual SLAM module tracks the camera pose and transmits keyframe data to server. Meanwhile the reconstruction module integrates point clouds into TSDF (Truncated Signed Distance Function) voxels based on camera poses and transforms between sensors. After each certain period T_{submap}, the current TSDF map in clients is published to the server as a submap. The integration in the server then starts with an empty TSDF map. To reduce communication usage of transmission of TSDF submaps, the submaps are sent as *mesh packs*, and then recovered to SDF submaps in the server (refer to Section V).

B. Server

Loop Closure Detection See Section IV-A. In the server end, the loop closure detection module collects keyframe data from client visual odometry frontend, searches keyframe matches and computes transformations between keyframes from different clients. Then keyframe matches and transformations are forwarded to optimization module.

Client Handler Client handlers receive submap data from clients, recover mesh packs to TSDF submaps which are then converted to ESDF (Euclidean Signed Distance Function) for later optimization, and forward odometry constraints to the following pose graph optimization.

Optimization We develop a robust map fusion method through joint optimization of trajectories and the recovered submaps. The optimization module optimize poses of submaps based on three types of constraints: odometry, loop closure and registration constraints, see Section VI. After optimization, relative transformations of client maps can be determined and used for inter-robot localization. Finally, SDF submaps and meshes are combined and filtered based on submap poses to obtain global reconstruction results.

IV. Loop Closure Detection and SDF-based Outlier Rejection

A. Loop Closure Detection

Similar to other inter-robot localization methods [6], [5], [13], in this work, keyframes are firstly matched by querying the bag-of-words database, and a single database is shared among all clients to enable loop closure detection across clients. Then correspondence searching is performed on the best N candidates. Each match candidate is checked for its associate 3D landmarks, which are reprojected from the

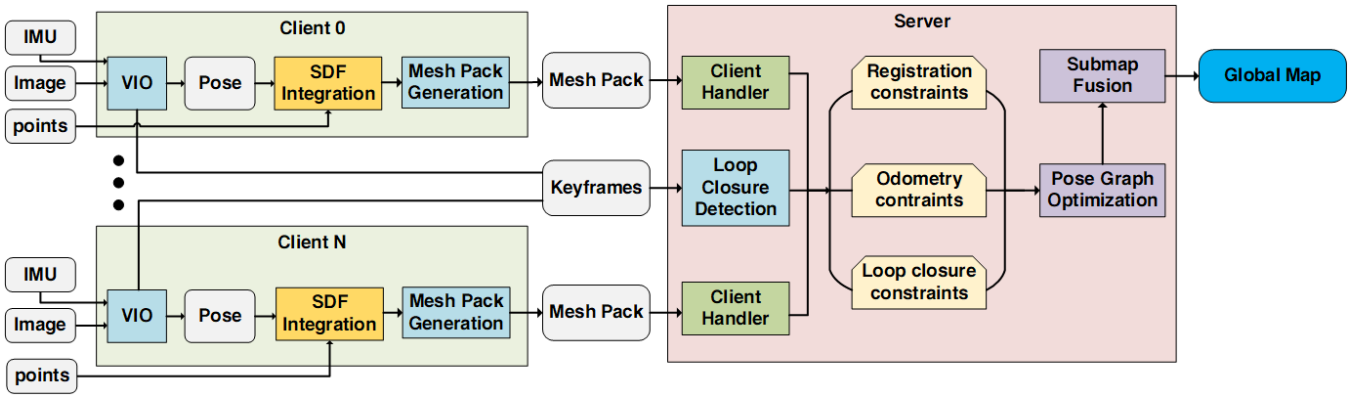


Fig. 2: Overview of Coxgraph system architecture. A multi-robot dense mapping system requires minimal network traffic and onboard computation on clients, and meanwhile maintaining global consistency intra- and cross- robots.

candidate frame to the current frame, and vice-versa. After 3D-2D RANSAC outlier rejection, valid keyframe matches and relative pose T_{ij} are published if sufficient inliers are found. Here we extend the client-server version of VINS-Mono in [13] to work as the VIO frontend and loop closure detector in our system.

B. SDF-based Outlier Rejection

Thanks to the SDF transmission method introduced in Section V, we are able to obtain clients' SDF maps in the server. Therefore for each computed loop closure transformation T_{ij} between submap S_i and S_j, we can take extra steps of loop closure outlier rejection based on dense maps, besides commonly used keyframe-based pairwise consistency checking steps as in [8].

1) Trajectory Collision Check: For each loop closure candidate pair, i.e., submaps S_i and S_j, we check the validity of received loop closure transformation by checking trajectory collision, given the fact that, transformed to the second submap S_j, all poses in the trajectory P_i of the first submap S_i should always lie in the free space of the second submap with a static environment assumed, i.e., the corresponding voxels in S_j haven't been created or have the default free space distance value.

Poses are considered valid if their transformed positions in S_j is not occupied, i.e., its SDF distance value is bigger than the default value for free space. This step of checking is bidirectional, i.e., the trajectory of S_j is also transformed to S_i for checking. If a minimal fraction of trajectory poses remain in the free space in the collision check, we consider the loop closure transformation T_{ij} passes this checking.

2) Fitness Check: For fitness checking, we take a similar method as SDF-based fitness evaluation in [18]. Isosurface points p^i_{iso} of S_i are firstly transformed to S_j. Given a correct relative transformation $T_{S_iS_j}$, transformed isosurface points should still lie in the isosurface of the other submap. If not, a fitness score can be determined by the fraction of p^i_{iso} to return a valid SDF value near to 0. Naturally, this step is also performed bidirectionally by transforming p^j_{iso} into S_i.

V. Submap Transmission

Millane et al. propose C-Blox in [19] that uses SDF-based submaps as the representation of map, and in Voxgraph [20], Reijgwart et al. optimize submap pose graph to reach global consistency in large-scale dense SDF mapping and high computation efficiency. In this work, we extend their work for multi-robot application, by introducing a method of submap transmission with minimal network usage, and meanwhile maintaining the advantage of global consistency.

As introduced in Voxgraph [20], there are two types of data needed to construct registration constraints in submap pose graph optimization: isosurface points and SDF maps containing distance and direction information.

In order to publish necessary data for optimization with minimal data size, we leverage the ability of meshes to represent SDF maps. From a SDF map, meshes are extracted by connecting zero-level voxels using the marching cube method [21]. In this way, we can represent isosurface points using mesh vertices. Furthermore, to reconstruct SDF maps in server end, besides the isosurface points, we also need to know the observation ray of these points to re-integrate them into SDF maps. So we record and publish visibility information of each mesh triangle, i.e., in which frames it has been observed, and the poses of all frames in each submap. Therefore, having isosurface points and associated camera poses, we can re-integrate points to get a recovered SDF map with negligible onboard computation cost and minimal bandwidth usage.

In conclusion, three types of data are recorded during the compression of each submap S_i, named as *mesh pack* for the remainder of this work, including:

1) Mesh data of submaps, containing triangles.
2) Frame indices i_{obs} of each mesh triangles when the corresponding voxels are observed, noted as observation history vectors hv.
3) Camera poses in each submap.

Noticeably, each mesh triangle only contains three vertices and colors, but every triangle can be observed in many frames during the submap period T_{submap}, i.e., the duration how long a submap lasts and a new submap should be

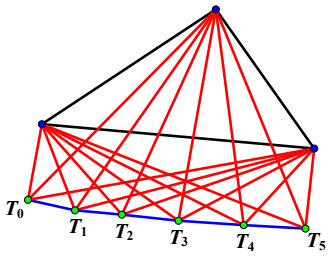


Fig. 3: Mesh pack generation schematic. Besides vertices of each triangle, the observation frame indices $i_{obs} \in (0,1,2,3,4,5)$, and pose histories $T_i, i \in (0,1,2,...,N)$ are also included in mesh packs.

created, which means a triangle may contain an observation history vector hv requiring larger communication amount than the triangle itself. However, intuitively the observation histories of triangle have the property of continuity, i.e., if a vertex V is observed at frame t, it is most likely V is also observed in adjacent frames around t, meanwhile discontinuous observation has higher possibility to be caused by noisy sensor data. Therefore, from the perspective of continuity of observation, history vector can be trimmed before transmission. In this step, history vectors are transformed into starting and terminating indices of subsequences, i.e., $hv = \{i_s, i_t\}_k,\ k \in subsequences$. Valid subsequences are extracted if a subsequence has the continuity where the difference between adjacent indices does not exceed K_{diff} and it contains at least K_n indices. In the experiments, we take $K_{diff} = 3$ and $K_n = 4$.

This concise method can reduce the data size of mesh packs effectively, and as a result, mesh packs require network traffic only 10% of original TSDF submaps, and about 30% of mesh packs without history vector trimming, as shown in Figure 8a and 8b. We evaluate the proposed recovery method in Section VII-A, and the quality of the recovered SDF is illustrated in Figure 5. We then optimize submap poses globally, and generate global maps as in Section VI.

VI. Map Fusion Optimization

After valid loop closure messages are received, different from other multi-robot localization systems which perform global optimization on loop closure constraints directly on trajectories to correct drifts, we propose a robust map fusion method through joint optimization of trajectories and SDF submaps with the awareness of dense mapping consistency, by including registration constraints between overlapping submaps as firstly proposed in [20].

A. Constraints

Three types of constraints are considered:

1) Loop Closure: The poses of submaps related to the loop closure message are firstly adjusted by transformation $T_{S_iS_j}$ as deducted as following, in which $T_{S_it_a}$ stands for the camera pose at t_a in the submap frame of S_i, also the case to $T_{S_jt_b}$, and $T_{t_at_b}$ is the received loop closure transformation between camera pose at t_a and t_b:

$$T_{S_iS_j} = T_{S_it_a} T_{t_at_b} T^{-1}_{S_jt_b} \tag{1}$$

Therefore the loop closure constraint is given as:

$$e^{i,j}_{loop}(T_{WS_i}, T_{WS_j}) = \log\left(T_{WS_i} T_{S_iS_j} T^{-1}_{WS_j}\right) \tag{2}$$

In the case of multi client map fusion, T_{WS_i}, T_{WS_j} are the poses of submap S_i and S_j in the world frame, and by default, the world frame is set identical to the odometry frame of client C_0, i.e. $T_{WC_0} = I$.

2) Odometry: Odometry-estimated relative poses $T_{S_kS_l}$ of submaps from the same client, between submap S_k and S_l, are added as constraints as following,

$$e^{k,l}_{pose}(T_{WS_k}, T_{WS_l}) = \log\left(T_{WS_k} T_{S_kS_l} T^{-1}_{WS_l}\right) \tag{3}$$

3) Registration: The correspondence-free registration constraint proposed in [20] is included in optimization. Firstly, overlapping submaps S_m, S_n are detected by comparing poses and bounding boxes. Then points p_{S_m} on iso-surfaces are extracted from Euclidean Signed Distance Function (ESDF) of submap S_m and projected to the frame of submap S_n. The distance from projected point $p^i_{S_m}$ to iso-surface of submap S_n can be determined by reading the ESDF value at the point. Registration constraint is then formed expressing all squared distances from points $p^i_{S_m}$ to iso-surfaces of S_n:

$$e^{m,n}_{reg}(T_{WS_m}, T_{WS_n}) = \sum_{i=0}^{N_{S_m}} r_{S_mS_n}(p^i_{S_m}, T_{S_mS_n})^2, \tag{4}$$

where N_{S_m} are the number of iso-surface points of submap S_m. Then the residual $r_{S_mS_n}$ is given by:

$$\begin{aligned} r_{S_mS_n}(p^i_{S_m}, T_{S_nS_m}) &= \Phi_{S_m}(p^i_{S_m}) - \Phi_{S_n}(T_{S_nS_m} p^j_{S_m}) \\ &= -\Phi_{S_n}(T_{S_nS_m} p^j_{S_m}), \end{aligned} \tag{5}$$

where $\Phi_{S_m}(p^i_{S_m}) = 0$, for all $p^i_{S_m}$ lie on iso-surface, and $T_{S_nS_m}$ is a function of optimization variables in χ as:

$$T_{S_nS_m} = T^{-1}_{WS_n} T_{WS_m} \tag{6}$$

B. Optimization

In the stage of optimization, we firstly roughly align submap poses based on odometry and loop closure transformations, and then add registration constraints between all overlapping submap pairs. The pose graph of submap poses is illustrated in Figure 4. From global optimization, we generate two outputs for inter-robot localization and global mapping:

Client Frame Determination For inter-robot localization, after we optimize constraints on poses of each submap from clients, transformations between client odometry frames $T^{i,j}_{C_\alpha C_\beta}$ can be determined from pairs of submap poses from different clients.

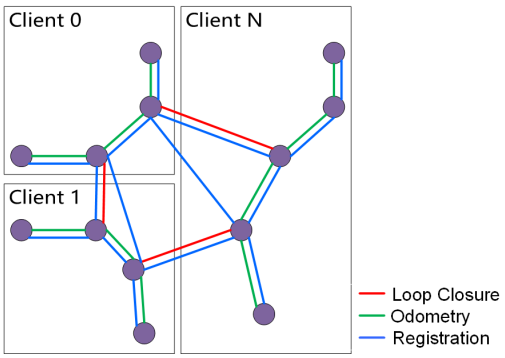


Fig. 4: Schematics depicting the pose graph in map fusion optimization. There are three client 0, 1 and N in this graph and contain client submaps indicated as purple circles. Red, greed and blue lines relatively stands for loop closure, odometry and registration constraints.

To obtain an optimized $T^{opt}_{C_\alpha C_\beta}$ that minimizes the total error over all $T^{i,j}_{C_\alpha C_\beta}$, we perform another lightweight optimization on client frames.

This second optimization takes transformation T_{WC_γ} from the world frame to the client odometry frames as nodes, aiming to minimize the transformation error on $T^{i,j}_{C_\alpha C_\beta}$, the constraint is formed as:

$$e^{\alpha,\beta}_{tf}(T_{WC_\alpha}, T_{WC_\beta}) = \log\left(T_{WC_\alpha} T^{i,j}_{C_\alpha C_\beta} T^{-1}_{WC_\beta}\right), \tag{7}$$

where, the world frame is initialized as $T_{WC_0} = I$, if not specified externally.

Therefore, from the result of client frame optimization, we can determine transformations from the world frame to the client frames.

Global Map Generation To generate global volumetric map, we combine all SDF submaps as [19] and [20]. In addition to the global SDF map, we also combine and filter the received meshes which are the same ones used for submap recovery to generate global map to compensate the completeness loss caused by SDF integration in complicate scene.

Figure 9 shows the comparison of meshes generated in different stages and methods. Figure 9b is generated by combining meshes, and 9d is the mesh generated from recovered SDF. The combined mesh can effectively compensate reconstruction completeness loss caused by submap merging.

VII. Experiments

We validate and evaluate the proposed submap transmission method and reconstruction performance of our multi-robot system with extensive experiments in this section.

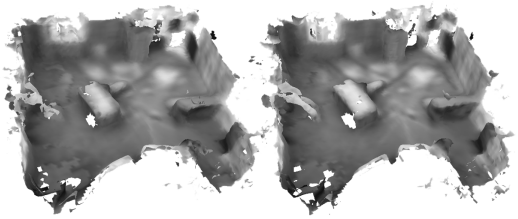
A. SDF Recovery

We firstly evaluate the performance of the proposed mesh-to-SDF recovery method, on both Machine Hall and Vicon Room 1 scene of EuRoC Dataset. The SDF map is reconstructed using voxblox and the camera poses provided by VINS-Mono, converted to mesh pack, and then recovered to SDF. For better comparison, the recovered SDF is converted to mesh again, and compared with the original mesh. The reconstruction error between the origin and the recovered mesh is shown in Table I. Based on the surface reconstruction error, the recovery method only cause slight error compared with the origin SDF map. Also the transformation process only costs negligible computation, since the recording of all required information is already finished while integrating point clouds into SDF.

TABLE I: Reconstruction error of the proposed SDF recovery method compared to original SDF map.

Scene	V1_01	MH_01	MH_02	MH_03
RMSE(m)	0.059	0.065	0.069	0.029

Fig. 5: Comparison between meshes generated from original (left) and recovered (right) SDF maps of Vicon Room 1.

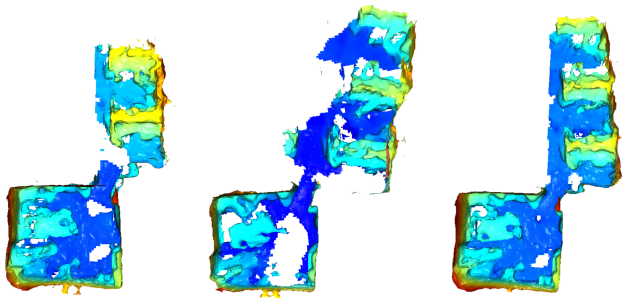


B. Multi-Robot Reconstruction

We further test the multi-robot reconstruction performance of *Coxgraph*. We reconstruct Machine Hall with flights MH_01, MH_02 and MH_03 of the EuRoC Dataset using depth maps generated from stereo matching, with the voxel size set to 5cm. The merged trajectories and meshes are demonstrated in Figure 7. Using Machine Hall flights, we evaluate our system with the metrics of Absolute Trajectory Error (ATE), reconstruction error and average data size transmitted for submaps. The SDF map generated offline by Maplab [22] is used as the ground truth. The trajectory error is demonstrated in Table II, and the surface reconstruction error is shown in Table III. Our system can effectively correct trajectory drift, as seen in the experimental results. Working in the direct mode, i.e., transmitting SDF directly, our system reaches a surface reconstruction RMSE of 0.116 m with bandwidth usage around 1 MB/s. Compared to the direct mode, the recovery mode provide dense reconstruction result with 0.129 m reconstruction RMSE as demonstrated by Figure 9d, and network traffic only about 100 KB/s. Also the mesh combination step of recovery mode can generate meshes with higher completeness as shown in Figure 9b with a reconstruction RMSE of only 0.111 m. Data sizes and bandwidth usage in Machine Hall experiment are illustrated in Figure 8.

C. Platform Experiment

In order to attest the performance and practicality of Coxgraph in a real-world scenario, we reconstruct the CVG Lab in Zhejiang University. Three clients are tested in the



(a) (b) (c)

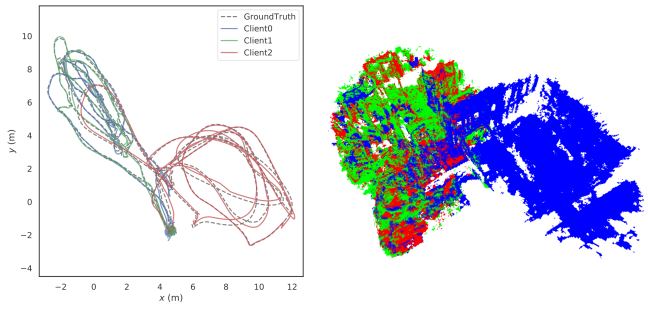
Fig. 6: Global consistency maintaining ability of our system. (a) is the mapping result of client 0 with accurate odometry and mapping result. In (b), client 1 is obviously affected by drifted odometry. The server still successfully corrects the odometry drift in client 1, and generates an accurate dense map (c) thanks to a correct loop closure and the optimization process.

TABLE II: Trajectory ATE comparison for EuRoC experiments. Our submap-based optimization system is compared against VINS-Mono realtime output running on clients, *direct* means optimization on raw SDF maps directly instead of *recovered* ones. RMSE (m) values are reported.

System	MH_01	MH_02	MH_03	merged
VINS-Mono(realtime)	0.24	0.22	0.21	-
Ours(direct)	0.22	0.10	0.10	0.17
Ours(recovered)	0.23	0.11	0.19	0.21

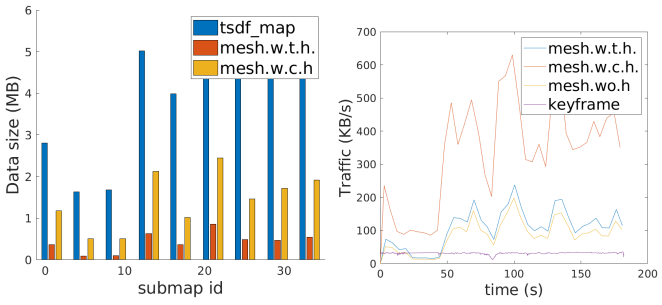
TABLE III: Surface reconstruction comparison between meshes generated from SDF transferred directly, transferred as mesh packs, and by combining mesh according to optimized submap poses. RMSE (m) values are reported.

Method	MH_01	MH_02	MH_03	merged	merged mesh
direct	0.238	0.267	0.264	0.116	-
recover	0.247	0.269	0.262	0.129	0.111



(a) (b)

Fig. 7: (a) Top view of trajectories in server map. (b) Result mesh colored by client id. *Red*: client 0 (MH_01). *Green*: client 1 (MH_02). *Blue*: client 2 (MH_03).



(a) (b)

Fig. 8: (a) Data size for each submap during reconstruction of Machine Hall. Blue bars are sizes of TSDF submaps, red bars are sizes of mesh packs with trimmed history vectors and yellow bars are sizes of mesh packs with complete history vectors. (b) Network traffic of transferring data of different types. The orange line is mesh pack with complete history vectors, the blue line is mesh pack with trimmed vectors, the yellow line is mesh without history data, and the purple line is keyframe data used for inter-robot localization.

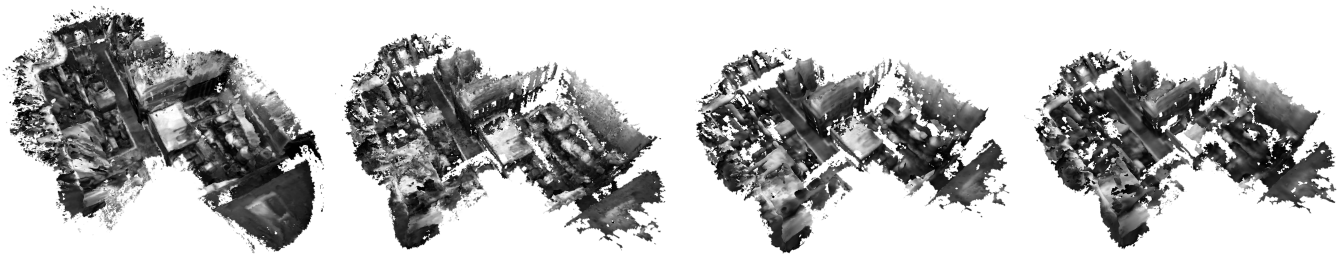
TABLE IV: Network traffic load of messages from clients to server during reconstruction of CVG lab.

Message Type	Mean Bandwidth	Std Deviation
Keyframes	20.25 KB/s	6.58 KB/s
Mesh Packs	25.24 KB/s	11.70 KB/s

experiment. Each client has a RealSense D435i depth camera for data collection, and a NUC10i7FNH onboard computer for processing. The server runs on a LENOVO LEGION laptop, of which the WiFi module is used as the router. The reconstruction result in Figure 1 is generated online with a voxel size of 0.1m and the submap interval of 5 seconds. The center of Figure 1 is the global mesh generated online by the server and colored by height, while the surroundings are the mapping results of three clients.

We also record CPU usage of client modules shown in Figure 10. Because the mesh observation history can be simultaneously recorded when updating SDF voxels, the mesh pack generation costs negligible CPU load. And since VINS-Mono frontend does not need to optimize the pose graph, it requires less computation. The percentage corresponds to how much of a single CPU thread is utilized. Our main client nodes TSDF integrator and VINS-Mono odometry frontend only consume approximately 110% and 180% of CPU load. Because only a local map of a time window is maintained by clients, our system unleashes the clients from the considerable memory usage of SDF reconstruction.

Sequences are also picked to show the ability of our proposed method to maintain inter-robot global consistency even with odometry drift in clients. As seen in Figure 6, the mapping result of the second client (Figure 6b) is severely affected by odometry drift when driving out the meeting room. However, in the global map generated by server, the error is effectively suppressed during the optimization thanks to a correct loop closure detected and the optimization



(a) (b) (c) (d)

Fig. 9: Comparison of meshes. (a) ground truth mesh generated offline by Maplab. (b) from global mesh generation step in our method. (c) generated from origin SDF submaps. (d) generated from recovered SDF submaps.

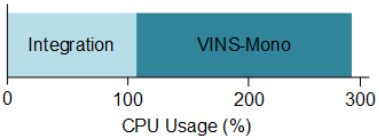


Fig. 10: CPU usage break down of *Coxgraph* client nodes, our client nodes cost only around 290% of CPU utilization.

method.

VIII. Discussion and Conclusions

In this paper, we propose an efficient system named *Coxgraph* for multi-robot collaborative dense reconstruction in real-time. To facilitate transmission, we propose a compact representation which transforms the SDF map to mesh packs that can be recovered to SDF map, and the submaps of clients are optimized and merged to reach a globally consistent dense map. Our proposed system can be easily extended with a path planning module in the future, and also have the potential to be modified to a distributed system.

References

[1] O. Wasenmüller, M. Meyer, and D. Stricker, "Augmented reality 3D discrepancy check in industrial applications," *2016 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 125–134, 2016.

[2] M. Zollhöfer, C. Siegl, M. Vetter, B. Dreyer, M. Stamminger, S. Aybek, and F. Bauer, "Low-cost real-time 3D reconstruction of large-scale excavation sites," *J. Comput. Cult. Herit.*, vol. 9, no. 1, Nov. 2015. [Online]. Available: https://doi.org/10.1145/2770877

[3] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon, "KinectFusion: Real-time 3D reconstruction and interaction using a moving depth camera," in *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '11. New York, NY, USA: Association for Computing Machinery, 2011, p. 559–568. [Online]. Available: https://doi.org/10.1145/2047196.2047270

[4] A. Dai, M. Nießner, M. Zollöfer, S. Izadi, and C. Theobalt, "BundleFusion: Real-time globally consistent 3D reconstruction using on-the-fly surface re-integration," *ACM Transactions on Graphics 2017 (TOG)*, 2017.

[5] P. Schmuck and M. Chli, "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," *Journal of Field Robotics*, vol. 36, no. 4, pp. 763–781, 2019.

[6] F. Li, S. Yang, X. Yi, and X. Yang, "CORB-SLAM: a collaborative visual slam system for multiple robots," in *International Conference on Collaborative Computing: Networking, Applications and Worksharing*. Springer, 2017, pp. 480–490.

[7] S. Dong, K. Xu, Q. Zhou, A. Tagliasacchi, S. Xin, M. Nießner, and B. Chen, "Multi-robot collaborative dense scene reconstruction," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 4, pp. 1–16, 2019.

[8] P.-Y. Lajoie, B. Ramtoula, Y. Chang, L. Carlone, and G. Beltrame, "DOOR-SLAM: Distributed, online, and outlier resilient slam for robotic teams," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1656–1663, 2020.

[9] Y. Chang, Y. Tian, J. P. How, and L. Carlone, "Kimera-Multi: a system for distributed multi-robot metric-semantic simultaneous localization and mapping," *arXiv preprint arXiv:2011.04087*, 2020.

[10] D. Zou and P. Tan, "CoSLAM: Collaborative visual slam in dynamic environments," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2013.

[11] M. Karrer, P. Schmuck, and M. Chli, "CVI-SLAM—collaborative visual-inertial slam," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2762–2769, 2018.

[12] P. Schmuck and M. Chli, "CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams," in *Journal of Field Robotics (JFR)*, 2018.

[13] L. Bartolomei, M. Karrer, and M. Chli, "Multi-robot coordination with agent-server architecture for autonomous navigation in partially unknown environments," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[14] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[15] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "NetVLAD: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5297–5307.

[16] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020. [Online]. Available: https://github.com/MIT-SPARK/Kimera

[17] H. Oleynikova, Z. Taylor, M. Fehr, R. Siegwart, and J. Nieto, "Voxblox: Incremental 3d euclidean signed distance fields for on-board mav planning," in *2017 Ieee/rsj International Conference on Intelligent Robots and Systems (iros)*. IEEE, 2017, pp. 1366–1373.

[18] A. J. Millane, H. Oleynikova, C. Lanegger, J. Delmerico, J. Nieto, R. Siegwart, M. Pollefeys, and C. C. Lerma, "Freetures: Localization in signed distance function maps," *IEEE Robotics and Automation Letters*, 2021.

[19] A. Millane, Z. Taylor, H. Oleynikova, J. Nieto, R. Siegwart, and C. Cadena, "C-blox: A scalable and consistent TSDF-based dense mapping approach," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 995–1002.

[20] V. Reijgwart, A. Millane, H. Oleynikova, R. Siegwart, C. Cadena, and J. Nieto, "Voxgraph: Globally consistent, volumetric mapping using signed distance function submaps," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 227–234, 2019.

[21] W. E. Lorensen and H. E. Cline, "Marching cubes: A high resolution 3d surface construction algorithm," *ACM siggraph computer graphics*, vol. 21, no. 4, pp. 163–169, 1987.

[22] T. Schneider, M. Dymczyk, M. Fehr, K. Egger, S. Lynen, I. Gilitschenski, and R. Siegwart, "Maplab: An open framework for research in visual-inertial mapping and localization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1418–1425, 2018.