UNIVERSITY OF CALIFORNIA, MERCED

Towards Optimal 3D Reconstruction and Semantic Mapping

A dissertation submitted in partial satisfaction of the
requirements for the degree
Doctor of Philosophy

in

Electrical Engineering and Computer Science

by

Guoxiang Zhang

2021

The dissertation of Guoxiang Zhang is approved, and
it is acceptable in quality and form for publication on
microfilm and electronically:

(Professor Dong Li)

(Professor Mukesh Singhal)

(Professor YangQuan Chen, Chair)

University of California, Merced

2021

To my wife, son, and parents

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to my advisor Prof. YangQuan Chen. Without Prof. Chen's help and patient guidance, I may not survive difficulties in my Ph.D. study. With regard to research, Prof. Chen not only introduced me to an excellent research topic but also taught me how to do good research with critical thinking and writing, which now becomes an invaluable treasure for my future career.

I would like to thank Prof. Dong Li and Prof. Mukesh Singhal for taking their precious time to serve on my qualification and dissertation committee. Thanks also go to Prof. Holley Moyes for coordinating the SmartCaveDrone project and providing domain knowledge and comments. Thanks go to Dr. Bo Shang for the collaboration at the initial stage of the SmartCaveDrone project. Thanks go to Peng Wang for his collaboration in Chapter 7.

Thanks go to my lab colleagues Dr. Tiebiao Zhao, Sina Dehghan, Derek Hollenbeck, Haoyu Niu, Jairo Viola, Di An, Furkan Guc, Prof. Qi Yang, Prof. Jun Miao, Dr. Panpan Gu, Tian Fen, Lihong Guo, and many others. Thanks go to the undergraduate researchers I have mentored and worked with: Christopher Currier, Marion Cooke, Mighty Chen, Jason Eckenrode, Kevin Ong, Zach Yinger, Jonathan Rivas, Alex Ho, Jeffrey Ng, and Habibatu Mboka.

The work in this dissertation was supported by CITRIS Seed Grant entitled "Drones for Cave Archeology and 3D Mapping", UC Merced Electrical Engineering and Computer Science Graduate Program, and GPU donation from NVIDIA Corporation.

I am grateful for the support from my parents, Shuliang Zhang and Xiangju Du, and sister Yingying Zhang. Their unconditional support guarded me against potential hardship.

Last but not least, a special thank goes to my wife, Zhirui Nie, who plays a vital role in my completion of this dissertation. I am very grateful for her encouragement, her strong support, and all her love.

VITA

2013	B. E. in Electronic Science and Technology, Xidian University, China
2015	M. S. in Signal and Information Processing, Xidian University, China
2021	Ph. D. in Electrical Engineering and Computer Science, University of California, Merced

PUBLICATIONS

Guoxiang Zhang, YangQuan Chen and Holley Moyes, Smart 3D Processing of Unconstrained Caves Scans Using Small Unmanned Aerial Systems and RGB-D Cameras. *International Journal of Advanced Robotic Systems*, To appear.

Guoxiang Zhang and YangQuan Chen, Self-Optimizing Loop Sifting and Majorization for 3D Reconstruction, In *Proc. of the 2021 International Design Engineering Technical Conferences & Computers and Information in Engineering Conference. arXiv preprint arXiv:2104.10826*.

Guoxiang Zhang and YangQuan Chen, More Informed Random Sample Consensus, In *Proc. of the 2020 8th International Conference on Control, Mechatronics and Automation (ICCMA)*, Moscow, Russia, 2020, pp. 197-201. **Best Presentation Award**.
DOI: 10.1109/ICCMA51325.2020.9301545.

Guoxiang Zhang, Jose Alcala, Jeffrey Ng, Mighty Chen, Xiangyu Wu, Mark Mueller and YangQuan Chen, Embedding Consequence Awareness in Unmanned Aerial Systems with Generative Adversarial Networks, In *Proc. of the 2019 International Conference on Unmanned Aircraft Systems (ICUAS)*, Atlanta, GA, 2019, pp. 129-134. DOI: 10.1109/ICUAS.2019.8798141.

Guoxiang Zhang, YangQuan Chen and Holley Moyes, Optimal 3D Reconstruction of Caves Using Small Unmanned Aerial Systems and RGB-D Cameras, In *Proc. of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, Dallas, TX, 2018, pp. 410-415. DOI: 10.1109/ICUAS.2018.8453277.

Guoxiang Zhang, Bo Shang, YangQuan Chen and Holley Moyes, SmartCaveDrone: 3D cave mapping using UAVs as robotic co-archaeologists, In *Proc. of the 2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, Miami, FL, USA, 2017, pp. 1052-1057. DOI: 10.1109/ICUAS.2017.7991499.

Under review

Guoxiang Zhang and YangQuan Chen, A metric for evaluating 3D reconstruction and mapping performance with no ground truthing, *arXiv preprint arXiv:2101.10402*, submitted.

Guoxiang Zhang and YangQuan Chen, Context Aware Optimized RANSAC With Applications in Point Cloud Registration, submitted.

Guoxiang Zhang, Peng Wang, Ning Chen and YangQuan Chen, 3D Semantic Mapping: A Benchmark and Baseline method, submitted.

ABSTRACT OF THE DISSERTATION

Towards Optimal 3D Reconstruction and Semantic Mapping

by

Guoxiang Zhang
Doctor of Philosophy in Electrical Engineering and Computer Science

University of California Merced, 2021

Professor YangQuan Chen, Chair

3D reconstruction and semantic mapping are of great importance for many tasks and applications, such as consumer robots, augmented reality, digital heritage, and autonomous vehicles. Despite the drastic advancements in solving the 3D reconstruction problem, it is still challenging to reconstruct accurate 3D models and create semantic maps. Within this dissertation, contributions are made to take steps closer towards optimal 3D reconstruction and semantic mapping.

It is crucial to have an easy performance evaluation method for advancing 3D mapping systems. Thus, in Chapter 5, we propose dense map posterior (DMP) for 3D reconstruction and mapping performance evaluation that can work without any ground-truth data. With this metric, one can evaluate 3D mapping systems on any public or new data without worrying about the availability of ground-truth data captured by expensive and bulky equipment. We also show that the DMP can be used beyond the evaluation of final results. It can act as a supervisory figure-of-merit signal during 3D reconstruction processes.

To improve 3D reconstruction results, we propose a novel 3D reconstruction system that corrects surface loops with sparse feature-based bundle adjustment. In the system, fast 3D surface-based loop detection is done by a GPU-accelerated random sample consensus algorithm with optimized randomness supported by fractional calculus, which is in Chapter 3. Then, to solve a low-precision problem in surface loop detection, in Chapter 4, an online method for loop sifting is proposed for real-time feedback to

users. For the best 3D reconstruction performance, an offline method for loop sifting and majorization is proposed in Chapter 6. State-of-the-art performance is observed in experiments on public and our datasets.

In Chapter 7, an exploration of semantic mapping is carried out. A simple and effective real-time 3D semantic mapping method is proposed. In addition, a benchmark suite with a dataset derived from the KITTI dataset and three novel metrics are developed for semantic mapping evaluation.

Finally, conclusions and future works are presented in the last chapter.

Chapter 1

Introduction

3D reconstruction and semantic mapping are of great importance for many tasks and applications, such as consumer robots, augmented reality, and autonomous vehicles. For the 3D reconstruction problem, research has been done in the literature. Existing methods can be classified into two categories based on their computational speed: online 3D reconstruction method and offline 3D reconstruction method.

1.1 Online 3D reconstruction

Visual simultaneous localization and mapping (Visual SLAM or vSLAM) can produce 3D reconstruction results in a real-time manner. It has been studied actively by researchers from different fields, such as robotics, computer vision, and computer graphics [14]. They solve this problem with their emphases and preferences, which lead to diverse visual SLAM systems [80, 114, 108, 95]. Sparse feature-based SLAM systems [80, 107] are well developed because sparse features can be used to downsample data from sensor reading (*e.g.* images) to sparse data representation as image keypoints and feature vectors. This means less computation since data from different frames are matched solely based on feature vectors of their keypoints. Extended Kalman filter or particle filter-based filtering approaches [30, 79] can take keypoints as visual landmarks and solve vSLAM as a data filtering process. A drawback of this approach is that the filter cannot be re-optimized again based on all previous data. Then, maximum a posteriori (MAP) based approaches are used to optimize all observed camera data in a batch setting [80], which utilizes bundle adjustment (BA) from Structure from Motion [17, 29], to achieve a better accuracy [80]. In order to run BA for loop closure, the loops need to be detected. In the sparse image feature setting, Bag-of-words (BoW) based loop closure is widely used. But it gives a high portion of false loops, which may severely degrade the performance of a vSLAM system, so very strict loop filtering is often used [80], where many loops are rejected. This causes a big problem when there are many loopy motions in camera movement.

Another line of vSLAM research focuses on surface reconstruction. With the parallel processing power of GPU, Newcombee *et al.* proposed KinectFusion [83], which performs real-time dense 3D camera tracking and model fusion. It has a volumetric scene representation, which can be rendered to a depth map at a given camera pose. Tacking is done through a frame-to-model projective iterative closest point (ICP), which is parallelized on GPU for real-time performance. Finally, new camera data is fused into the volumetric model using a running average. KinectFusion can be considered of fusing very local loops together using the model it maintains as a proxy, but it does not close large loops. To close large loops, it is important to detect loops and find relative poses between loop areas. In BoW, image keypoints and features are used for both loop detection and relative pose generation, but in dense 3D systems, there is no such comparably reliable point cloud feature. Whelan *et al.* use BoW in a dense SLAM system called Kintinuous [113], which is an extended KinectFusion system. Later, to better solve the loop detection and optimization problem, ElasticFusion [114] was proposed to use ICP to find relative poses of potential loops, which are proposed by using two sources of information: spatial prior and appearance-based place recognition. After surface loop detected, in Kintinous, a pose graph of keyframes was utilized, while the authors mentioned that mesh deformation was required to get smooth 3D models, which indicates loop correction is not done optimally. In ElasticFusion, the pose graph is replaced by a deformation graph distributed inside the dense model. This deformation graph does not have a backing physical meaning, because most of the scenes scanned are not elastic. BundleFusion [29] used bundle adjustment to optimize loops, but they do not close surface loops. Instead, they close sparse feature loops and only use the dense surface for feature correspondence search and tracking.

One important work in RGB-D-based 3D reconstruction literature is KinectFusion [57], which first combines projective ICP and volumetric scene representation to build a real-time dense 3D reconstruction system on a GPU. It first reveals the potential of a real-time dense 3D reconstruction system. Then Kintinuous [113] extends it so that it works on a scale larger than a single room. Later, Whelan *et al.* propose ElasticFusion [114], which is based on deformation graphs and can jointly minimize geometric and photometric error. We discuss it in detail to get more insights into real-time 3D reconstruction systems.

In ElasticFusion, surfels [92] are used as a data representation of 3D models. Each surfel has seven attributes: position, normal, color, weight, radius, initialization timestamp, and last updated timestamp. With a radius property, a surfel can represent a locally flat surface around a given a position $\boldsymbol{p}$. After recovering the poses of a new RGB-D frame, the new data is incrementally fused into the 3D model while minimizing visible holes. It divides all the surfels inside the model into two parts: active area and inactive area. The active area is the area that was most recently observed within a period of time δt, and the inactive area is the rest part of the 3D model. They split the model into two parts for the purpose of closing loops. For every new frame, it will try to register incoming data to the active model using projective data association to projected RGB-D data from the currently camera pose estimate. Then it minimizes a combination of Euclidean distance error of corresponding points and photometric re-projection error to update the camera pose estimate.

It also has two levels of loop closure: local loop closure and global loop closure. Local loops aim to remove small mismatches between two model parts, while global loops realign surfels that belong to the same place after a big drift. Local loops are identified by attempting to register the observed part of the active model from the current estimated camera pose with the underlaid observed portion of the inactive model. This is done for each frame. If this registration is successful, a local loop is identified. Global loops are recognized using appearance-based place recognition based on the randomized fern encoding [45]. After a loop is identified, underlying deformation graphs will be optimized to get a deform 3D model to minimize error while keeping deformation as small as possible. Figure 1.1 is a result during the online processing of ElasticFusion.

ElasticFusion like systems can provide reasonably good results. We can observe that the processed results will fail in some situations, such as image blur during fast motion, especially fast rotation, and flat areas without color variations.

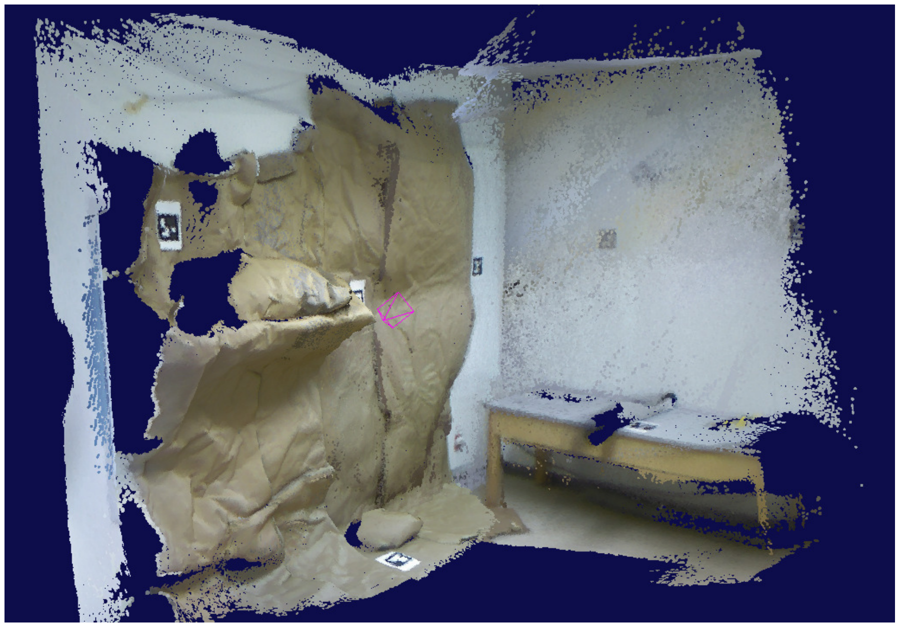


Figure 1.1: Result of online 3D reconstruction of a mock-up cave [126].

1.2 Offline 3D reconstruction

Other than different approaches to improve real-time performance, another branch of 3D reconstruction method is offline processing, which aims to achieve the best 3D reconstruction result by considering all the information jointly [119, 18, 128]. Structure from motion (SfM) [2, 24] and multi-view stereo (MVS) [101, 38, 39] have been actively explored, and they can be used to recover 3D models from sets of images. After the emerging of RGB-D cameras, Xiao *et al*. [119] run 3D SfM on both depth and RGB images. They extract and match image features across images and then get their 3D coordinates from depth images. After that, SfM is conducted using 3D coordinates of keypoints.

Choi *et al*. [18] produce good results among offline methods. They first use RGB-D visual odometry from Kintinous [113] to merge several frames into a scene fragment.

They do this under the assumption that RGB-D odometry is reliable in the short term, and it has two advantages. First, multiple sensor reading can average out high-frequency noise; Second, it can reduce the amount of computation by having fewer pieces of 3D models. Then it runs pairwise point cloud registration between all the pairs using a modified version of [100], which is the most time-consuming part of this method. This process will produce a relative pose between each pair. Among these poses, most are false positive ones, which means either that these pairs should not be register together or the relative poses are far from ground truth ones. The authors propose to use line processes to filter out these false positives by adding one weight term to each loop closure during least-squares optimization, which utilizes [1] as a back end. Finally, loop closures whose weights are smaller than a threshold are pruned. The remaining loop closures are used to construct the final model. This process turns out to be very effective in improving the precision of pairwise registration and quality of final 3D models. This method can improve the overall 3D reconstruction in terms of surface root mean square error (RMSE). We show one of the results of this method in Fig. 1.2.

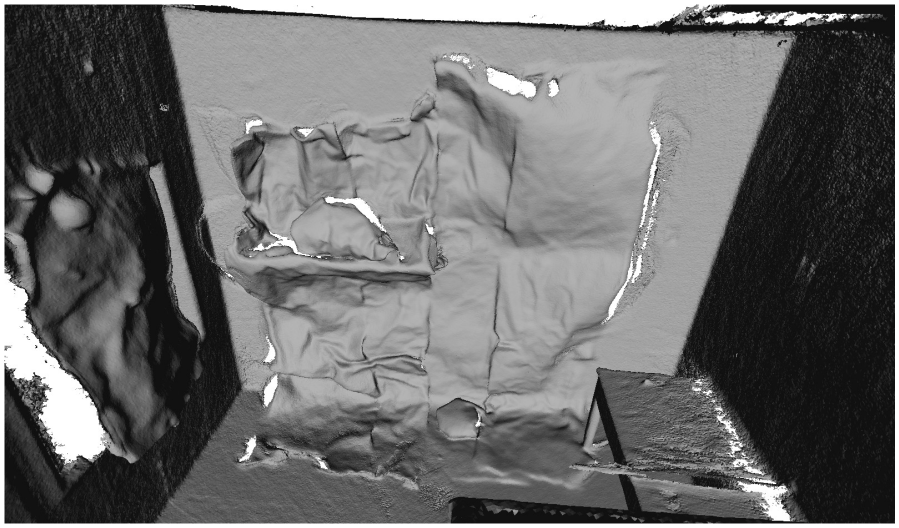


Figure 1.2: An offline refined 3D model of a mock-up cave [126].

1.3 Loop detection

1.3.1 Handcrafted feature-based approaches

Handcrafted features are first used for the loop closure detection problem. In 2007, Cummins and Newman [25] proposed a probabilistic framework for navigation using only appearance data, extended in [26, 27]. By learning a generative model of appearance, they can compute not only the similarity of two observations, but also the probability that they originate from the same location, and hence compute a probability density function (PDF) over observer location. In 2005, Wang *et al.* [111] present a coarse-to-fine global localization approach. Scale-invariant transformation feature descriptors as natural landmarks are indexed into a location vector space model (LVSM) and a location database. They are designed for two stages: coarse localization from the LVSM is fast, but not accurate enough, whereas localization from the location database using a voting algorithm is relatively slow, but more accurate. This system is tested in indoor and outdoor environments. In 2006, Ho and Newman [52] proposed an approach that relies instead upon matching distinctive 'signatures' of individual local scenes to prompt loop closure. In this work, they started using scale-invariant feature transform (SIFT) [75] image features and constructing visual vocabulary by clustering algorithms. They tested their algorithm on a dataset collected from an outdoor environment, which contains 155 images and laser scans.

In 2008, Angeli *et al.* propose an online method to run visual words-based loop detection within the framework of an online image retrieval task [3]. They made it possible to detect when an image comes from an already perceived scene using local shape and color information. Their approach extends the bag-of-words method used in image classification to incremental conditions and relies on Bayesian filtering to estimate loop-closure probability. Later, Angeli *et al.* [4] extend their work to be real-time capable. In 2011, Williams *et al.* [115] describe a relocalization module, in which relocalization is performed by first using a randomized lists classifier to establish landmark correspondences in the image and then random sample consensus (RANSAC) to determine the pose robustly from these correspondences.

Another widely used approach, DBOW [40], is proposed by Galvez-Lopez and Tardos in 2012, which use BoW for visual place recognition with FAST keypoint detector and BRIEF features. For the first time, they build a vocabulary tree that discretizes a binary descriptor space, and use the tree to speed up correspondences for geometrical verification. It shows competitive results in very different datasets. After [40], there are new attempts that try to make improvements on top of previous work, including attempts to make modifications on bag-of-words in [85, 43, 121, 3, 22], making improved detection by image sequences [7, 49, 53, 76, 6]. There are also works that try to make BoW run faster by working memory [67], visual memory using a Fuzzy ART [96], online binary feature [42], online vocabulary building [85], Gaussian Mixture Model with KD-tree [12], Sparsity-Cognizant with convex optimization [68] and online BoW with group similar images close in time [43].

There are also attempt to detect loops with more than a single image, including topological loop closure [34], image-to-image link recovery [54], hypothesis verification [60], object graph [89], and sequence-submap-based long-term [49, 48] loop detection. Some works try to improve loop detection precision with loop verification threshold learning in RANSAC based on geometric [69]. Others try to make improvements for special environments, such as dynamic environment [122], very large scale environment with 20 million key locations [116].

1.3.2 Learning-based approaches

After convolutional neural networks (CNN) and CNN features became the dominating method in vision-related tasks, research works try to build loop detection systems on top of CNN. The default way to try it is to use CNN features as the feature descriptor for similarity search. [55] use a pre-trained CNN model as a method of generating an image representation appropriate for visual loop closure detection. [117] utilizes PCANet features, while CNN features are processed before matching in [127]. An SVM is combined with CNN features in [118]. CNN features are weighted with hand-crafted features in [48]. Combining covisibility graph with CNN features is proposed in [16]. Using CNN feature with submaps [94] is also experimented.

There is another line of work trying to build special CNN architecture for loop detection. Siamese networks are adapted for similarity search in [74]. An end-to-end network is proposed in [74] to jointly optimize the two parts in a unified framework for further enhancing the interworking between these two parts. First, a two-branch siamese network is designed to learn respective features for each scene of an image pair. Then a hierarchical weighted distance (HWD) layer is proposed to fuse the multi-scale features of each convolutional module and calculate the distance between the image pair. Finally, by using the contrastive loss in the training process, the effective feature representation and similarity metric are learned simultaneously.

1.4 Challenges

Even though significant progress has been made to solve the 3D reconstruction problem, it is still challenging to reliably and efficiently reconstruct accurate 3D models and create semantic maps. First, sensors, such as RGB-D cameras, have a limited field of view and working range, which can bring in two problems: 1) more 3D model pieces need to be fused together; 2) each view only covers a small portion of the scene with limited information. These two problems make tracking prone to failure, especially when some areas of a scanned environment do not have enough shape and color variations. Second, environments usually contain sophisticated geometric structures and objects, and must be scanned from complex camera trajectories for better coverage, which means one place may be observed multiple times from different view angles. In theory, this should give more opportunities to minimize reconstruction error by detecting and optimizing loop closures. However, in practice, it often causes problems due to false loops added.

1.5 Dissertation contribution

The major contributions of this dissertation include, but are not limited to the following:

1. Developed a novel 3D reconstruction system that corrects surface loops with sparse feature-based bundle adjustment.

2. Proposed a fast 3D surface-based loop detection method, which is based on a new CUDA-accelerated point cloud registration algorithm.

3. Proposed a method that utilizes optimized randomness in random sample consensus (RANSAC) for point cloud registration.

4. Proposed a novel objective function for surface loop sifting with a sparse feature-based optimization graph. This graph is more robust to different scan patterns and can cope with tracking failure and recovery.

5. Proposed a metric, dense map posterior (DMP), for 3D reconstruction and mapping performance evaluation that can work without any costly ground-truth data.

6. Proposed an offline algorithm that can sift loop detections based on their impact on loop optimization results.

7. Proposed a simple and effective real-time 3D semantic mapping method. This method takes per-frame bounding box detections and sensor (camera) extrinsic transformation estimates as inputs and produces a set of static 3D bounding boxes in a world coordinate system as 3D semantic mapping results.

8. Derived a new semantic mapping benchmark dataset from the KITTI object tracking dataset. In the new benchmark, ground-truth semantic maps are constructed based on GPS-IMU data and labeled 3D bounding boxes of KITTI.

9. Proposed three novel semantic map-centered metrics for better evaluation of semantic mapping methods.

1.6 Dissertation organization

This dissertation is structured as follows. The research motivations and contributions are introduced in Chapter 1. In Chapter 2, we introduce preliminaries for 3D reconstruction.

In Chapter 3, a method that utilizes optimized randomness in RANSAC is proposed. The proposed method samples data with a Lévy distribution on ranked data. In the hypothesis sampling step of the method, data are ranked with a sorting metric we proposed, which sorts data based on the likelihood of a data point being from the inlier set. Then, hypotheses are sampled from the sorted data with Lévy distribution.

In Chapter 4, we propose a novel 3D reconstruction system for 3D reconstruction of caves. It corrects surface loops with sparse feature-based bundle adjustment. To build such a system, we propose a novel objective function for surface loop filtering with a sparse feature-based optimization graph. This graph is more robust to different scan patterns and can cope with tracking failure and recovery.

In Chapter 5, we propose a metric, dense map posterior (DMP), for 3D reconstruction and mapping performance evaluation that can work without any ground truth data. Instead, it calculates a relative dimensionless value, reflecting a map posterior probability, from dense point cloud observations.

In Chapter 6, an algorithm for offline sifting and majorization of loop detections is proposed and presented. With this algorithm, only correct and essential loops are fed into the following optimization steps. The proposed method highly couples with the dense map posterior (DMP) metric presented in Chapter 5 that can evaluate 3D reconstruction performance without ground truth measurement. Our proposed algorithm can compare the usefulness and effectiveness of different loops and ultimately sifts out false and unimportant loops.

In Chapter 7, a simple and effective real-time 3D semantic mapping method is proposed. In addition, a benchmark suite with a dataset derived from the KITTI dataset and three novel metrics are developed for semantic mapping evaluation.

Finally, Chapter 8 concludes the dissertation with discussions of future research directions.

1.7 Results reproducibility

Code for the methods presented in this dissertation will be published here `https://gzhang8.github.io/3DMapping/` so that readers can reproduce the results more easily.

Chapter 2

Preliminaries

2.1 Camera model

2.1.1 Pinhole camera model

A pinhole camera model is shown in Fig. 2.1. A camera could be approximated by a projective model, often called pinhole projection. The simplest representation of a camera is a light sensible surface (sensor): an image plane, a lens (projective projection) at a given position and orientation in space. The distance between the image plane and the principal point O is the focal length f.

Let $P = [X, Y, Z]^T$ be a point on a 3D object visible to the pinhole camera. P will be mapped or projected onto the image plane, resulting in point $P' = [X', Y']^T$, which follows

$$\frac{Z}{F} = \frac{X}{X'} = \frac{Y}{Y'}. \tag{2.1}$$

When considering that there is a translation and scale factor in order to convert to digital image space

$$\begin{cases} u = \alpha X' + c_x = f_x \frac{X}{Z} + c_x \\ v = \beta Y' + c_y = f_y \frac{Y}{Z} + c_y, \end{cases} \tag{2.2}$$

which can be written in matrix form

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{Z} \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \frac{1}{Z} \boldsymbol{K} P. \tag{2.3}$$

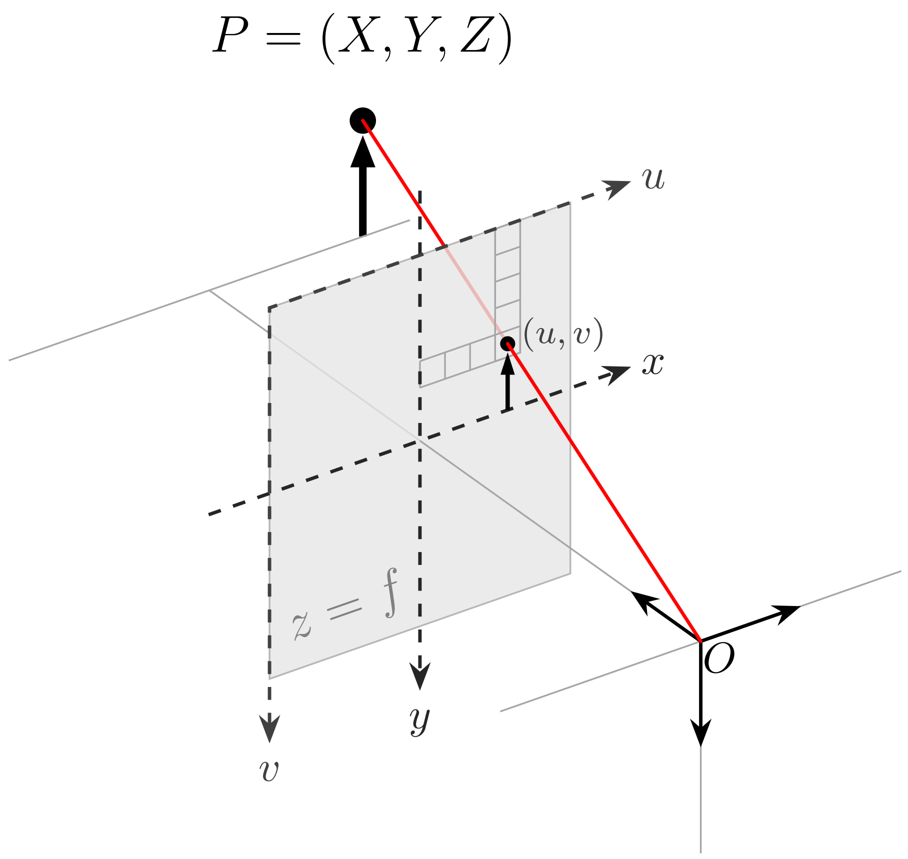


Figure 2.1: Pinhole camera model

$\boldsymbol{K}$ is the camera intrinsics, which can be obtained from a camera datasheet or calibration process. $\boldsymbol{p} = [u, v, 1]^T$ is a homogeneous coordinate for point $[u, v]^T$, such that any point $[x, y]^T$ becomes $[x, y, 1]^T$. Similarly, any point $[x, y, z]^T$ becomes $[x, y, z, 1]^T$. This augmented space is referred to as the homogeneous coordinate system. For simplicity, we define $\boldsymbol{u} = \pi(P; \boldsymbol{K})$ as the camera projection function.

A camera can also have a camera extrinsic matrix $\boldsymbol{T}$, which denotes the translation between the camera and the world coordinate system. Applying both, one can obtain

$$\boldsymbol{p} = \boldsymbol{K}\boldsymbol{T}P_w, \tag{2.4}$$

where P_w is a point in the world coordinate.

2.1.2 Stereo camera model

With a single pinhole camera, it is difficult to get the precise position of a point because depth z is unknown from only one image. One way to recover depth z is to use more than one camera to get observations of the same point. A stereo camera is one of these types of cameras. It has two pinhole cameras which usually share y and z axes. Considering that a point P in space is observed by both cameras with observations denoted as $\boldsymbol{p}_R$ and $\boldsymbol{p}_L$. Since the two cameras are placed only differently on the x-axis, $\boldsymbol{p}_R$ and $\boldsymbol{p}_L$ only have a difference on u readings in digital image space. We denote them as u_R and u_L separately. Then we get

$$\frac{z-f}{z} = \frac{b-u_L+u_R}{b}, \tag{2.5}$$

which leads to

$$z = \frac{fb}{d}, \tag{2.6}$$

where $d = u_L - u_R$ is called disparity. b is the baseline between the two cameras.

2.2 Scene representation

One key component to high-quality 3D reconstruction is the choice of underlying representation for fusing multiple sensor measurements. Approaches range from point-based representations [119], surfels [92, 51, 114], to volumetric approaches [57, 18, 128].

There are two types of point clouds: organized or unorganized. An organized point cloud is a point cloud with a structure in it, like a point cloud transferred from depth data. In an organized point, nearby points can be easily found by its structure, which enables projective ICP [99] for fast registration. One problem of point clouds is that when merging multiple point clouds, even points in the overlapping region will not merge with their neighbor points, so the data size of point-cloud models can be very large, which makes it unpractical to use. Also, it is difficult to visualize, since points do not have a size property.

Due to the aforementioned problems of point clouds, volumetric methods, which are based on implicit truncated signed distance functions (TSDF), have become widely used for high-quality 3D reconstruction [57, 18, 128]. These volumetric methods model continuous surfaces, regularize noise using running average, and efficiently perform incremental updates. However, one major drawback of the volumetric representation is that it is uniform for all places. Thus, it requires a huge amount of memory to be stored.

Another scene representation is surfels, where each surfel has the following attributes: a position $\boldsymbol{p} \in \mathbb{R}^3$, normal $\boldsymbol{n} \in \mathbb{R}^3$, color $\boldsymbol{c} \in \mathbb{N}^3$, weight $w \in \mathbb{R}$, radius $r \in \mathbb{R}$. The radius of each surfel is to represent the local surface area around a given point while minimizing visible holes. Surfels can be used to optimize color consistency [51] and model reflection and lighting conditions [114].

2.3 Camera tracking

In order to build a 3D model from a sequence of RGB-D frames, the first step is usually to build an initial camera trajectory estimate from camera tracking [119, 18]. We describe the basics of camera tracking in the following.

Sparse feature-based methods

Sparse feature-based camera tracking solves camera tracking by extracting image keypoints and then matching keypoints by their feature vectors, such as Oriented FAST and rotated BRIEF (ORB) [98] or SIFT [75] feature vectors. After feature points are matched, a perspective-n-point (PnP) problem [36] is solved to get the relative transformation from the previous frame to the current camera frame. The PnP problem can be solved by direct linear transform, efficient PnP solvers, or bundle adjustment solvers. The bundle adjustment formulation of the camera tracking problem is to minimize reprojection errors between frames, as in:

$$E(\boldsymbol{T}_{k,k-1}, \{P\}) = \sum_{i=1}^{n} ||\boldsymbol{u}_i - \pi(\boldsymbol{T}_{k,k-1} P_i; \boldsymbol{K})||^2 \tag{2.7}$$

where P_i is a point in the previously built sparse map, and $\boldsymbol{u}_i$ is the observation of point P_i. This objective function is a non-linear quadratic function, which can be minimized by the Levenberg-Marquardt algorithm.

Direct methods

Direct methods target to minimize a photometric error, in other words, the image intensity difference between the current color image frame I_k and the predicted color image from the frame I_{k-1}. The aim is to find the motion parameters $\boldsymbol{T}_{k,k-1}$ that minimize the error E as in:

$$E(\boldsymbol{T}_{k,k-1}) = \sum_{\boldsymbol{u} \in I_k} ||i_{k-1}(\boldsymbol{u}') - i_k(\pi(\boldsymbol{T}_{k,k-1}P; \boldsymbol{K}))||^2, \tag{2.8}$$

where $i_k(\boldsymbol{u})$ is a function that takes image intensity at a homogeneous point $\boldsymbol{u}$ from the image I_k.

In the cases that both RGB and depth are available, (2.7) and (2.8) can be combined to get better results.

2.4 Optimization

The bundle adjustment (BA) problem can be solved as a least-squares optimization of an error function that can be represented by a graph, as in Fig. 2.2. When more than one connection is added to the optimization of bundle adjustment, it can be solved by a pose graph. Each node of the graph represents a state variable to optimize. Each edge between two variables represents a pairwise observation of the two nodes it connects. For a pose graph, it tries to solve the following minimization problem:

$$\sum_{i,j \in \varepsilon} e_{ij}^T \Sigma_{ij}^{-1} e_{ij}, \tag{2.9}$$

where e_{ij} is an error term, and Σ_{ij} is the covariance matrix.

It can also be formed as a factor graph, where the objective is to maximize a posterior probability. A factor graph is a bipartite graph consisting of factors connected to variables. The variables represent the unknown random variables: camera pose $\boldsymbol{x}$ and landmarks l in the estimation problem, whereas the factors represent probabilistic information on those variables, derived from measurements z or prior knowledge. This can be written as

$$\underset{\boldsymbol{x}_0}{\operatorname{argmax}} \prod P(\boldsymbol{x}_k | \boldsymbol{x}_{k-1}, u_k) \prod P(\boldsymbol{z}_k | \boldsymbol{x}_i, l_i). \tag{2.10}$$

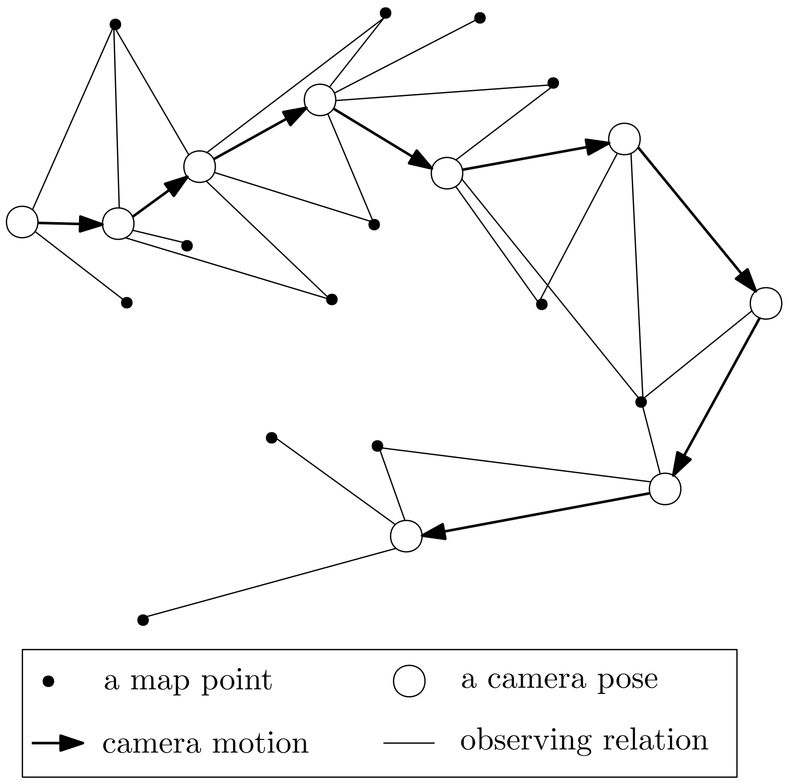


Figure 2.2: Bundle adjustment problem as a graph

2.5 Point cloud registration

Point cloud registration has been extensively explored [58, 120, 100, 93], and its definition is the following. Let two 3D point-sets $\boldsymbol{P} = \{\boldsymbol{p}_i\}, i = 1, ..., N$ and $\boldsymbol{Q} = \{\boldsymbol{q}_j\}, j = 1, ..., M$, where $\boldsymbol{p}_i, \boldsymbol{q}_j \in \mathbb{R}^3$ are point coordinates, be the *data* point-set and the *model* point-set respectively. The goal is to estimate a rigid motion with rotation $\boldsymbol{R} \in SO(3)$ and translation $\boldsymbol{t} \in \mathbb{R}^3$, which locates the optimal value of the following L$_2$-error E,

$$E(\boldsymbol{R}, \mathbf{t}) = \sum_{i=1}^{N} e_i(\boldsymbol{R}, \mathbf{t})^2 = \sum_{i=1}^{N} \|\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t} - \boldsymbol{q}_{j^*}\|^2, \tag{2.11}$$

where $e_i(\boldsymbol{R}, \mathbf{t})$ is the per-point residual error for $\boldsymbol{p}_i$.

A popular and widely used method is the so-called iterative closet point (ICP) [5] method, which uses alternate optimization. First, given $\boldsymbol{R}$ and $\mathbf{t}$, the point $\boldsymbol{q}_{j^*} \in \boldsymbol{Q}$ is denoted as the optimal correspondence of $\boldsymbol{p}_i$, which is the closest point to the transformed $\boldsymbol{p}_i$ in $\boldsymbol{Q}$, *i.e.*

$$j^* = \underset{j \in \{1,..,M\}}{\operatorname{argmin}} \|\boldsymbol{R}\boldsymbol{p}_i + \mathbf{t} - \boldsymbol{q}_j\|. \tag{2.12}$$

Note the notation used here: j^* varies as a function of $(\boldsymbol{R}, \boldsymbol{t})$ and also depends on $\boldsymbol{p}_i$. Second, equation (2.11) is minimized using correspondence found in previous steps. Equations (2.11) and (2.12) actually form a well-known *chicken-and-egg* problem: if the true correspondences are known *a prior*, the transformation can be optimally solved in closed-form; if the optimal transformation is given, correspondences can also be readily found. However, the joint problem cannot be trivially solved. Given an initial transformation $(\boldsymbol{R}, \boldsymbol{t})$, ICP iteratively solves the problem by alternating between estimating the transformation with (2.11), and finding closest-point matches with (2.12). Such an iterative scheme only guarantees convergence to a local minimum [11].

Since ICP can produce an accurate result when initialized near the optimal pose, but is unreliable without such initialization. Yang *et al.* [120] provide a globally optimal solution by using a branch-and-bound (BnB) scheme that searches the entire 3D motion space $SE(3)$, which is much slower than standard ICP. Zhou *et al.* [128] use Fast Point Feature Histogram feature descriptor to generate point correspondences and do not need to update correspondences during iterations in order to reduce computational complexity.

2.6 Chapter summary

In this chapter, we presented several key concepts in 3D reconstruction and semantic mapping. They are camera projection models, 3D scene representation, camera tracking, back-end least-squares optimization, and point cloud registration. Now we are ready to present our key innovations in 3D reconstruction and semantic mapping. We start with fractional order RANSAC for point cloud registration, a key technique used in our proposed system for surface loop detection.

Chapter 3

Fractional Order Random Sample Consensus

3.1 Introduction

Motivated by the fact that humans can notice mismatches in 3D models very easily by looking at the spatial displacement of surfaces, we propose to detect surface loops in addition to image loop detections. To formulate surface loop detection formally, we denote a surface fragment as a set of points with their normal. Then the surface fragment-based loop detection problem can be solved by point cloud registration methods. In a 3D reconstruction system, it is desired to get results quickly. In ElasticFusion [114], Whelan *et al*. resort to projective ICP, which can be performed very quickly on GPU. But, when surface mismatches are big, the initialization-dependent nature of ICP makes it difficult to converge to the right solution. So we turn to global point cloud registration in our framework, which does not depend on initial alignment at all. We find that a point cloud registration method [18] has desired performance except that it does not run fast enough for online SLAM methods. In order to solve this problem, improvements from two aspects are made to make it faster: 1) algorithm level with better randomness in random sample consensus (RANSAC), and 2) parallel implementation on GPU.

The RANSAC [36] is a robust model parameter estimation algorithm that can work with data which contain a large proportion of outliers. Due to its robustness to outliers, RANSAC is widely used in many fields including signal estimation [86], image-stitching [19], visual odometry [63], pattern detection [59] and point cloud registration [125, 124].

However, the RANSAC algorithm requires long computation time [86] because it needs to test a large number of hypotheses to find a good model. The number of hypotheses is set by a formula that depends on the number of inliers. But, in practice, the number of inliers is typically unknown. Thus RANSAC may be used with fewer iterations. Also, the probability of getting a correct model estimation dramatically decreases when the initial inlier ratio is low [71]. Moreover, it has been observed that a noise-contaminated outlier-free hypothesis may lead to a bad model estimate, which further requires more hypotheses to be tested [21].

In algorithm level, we propose a method that utilizes better randomness in RANSAC. The proposed method samples data with a Lévy distribution on ranked data. In the hypothesis sampling step of our method, data are ranked with a sorting metric we proposed, which sorts data based on the likelihood of a data point being from the inlier set. Then, hypotheses are sampled from the sorted data with Lévy distribution.

This Lévy distribution-based sampling strategy improves the probability of sampling a good hypothesis thus increase the chance of finding correct solutions. On the other hand, our method can converge to correct solutions with a similar probability with a smaller number of iterations. Our experiments on simulation and real-world data confirm the advantage of our method.

At the implementation level, we propose a new CUDA-accelerated acceleration of the improved RANSAC algorithm. We accelerated the most time-consuming parts using GPU programming with an efficient nearest neighbor search method. Traditionally, RANSAC is formulated as an iterative process with proved convergence [36]. But different iterations and different hypotheses can be considered to be totally independent of each other. This means different hypotheses can be mapped to different processing cores to be tested in parallel. Experiments show that it is fast enough for online use.

3.2 Related work

After RANSAC [36] proposed by Fischler and Bolles, research has been done to improve its performance. LO-RANSAC [21] proposes a local optimization step after the minimal sample model, which is helpful in correcting an incorrect assumption that a model computed from outlier-free samples is consistent with all inliers. In MLESAC [109], Torr and Zisserman proposed a new way of accessing model quality by choosing the solution that maximizes the likelihood rather than just the number of inliers. It is reported to be superior to the inlier counting of the plain RANSAC and less sensitive to threshold setting [8]. A differentiable RANSAC layer is introduced in [13] that can be used in neural networks in an end-to-end manner, which provides promising results. Latent RANSAC [65] presents an approach that can evaluate a hypothesis independent of input size. This method is based on the assumption that correct hypotheses are tightly clustered together in the latent parameter domain.

Among all the different improvements on RANSAC, a few of them are closely related to our method. PROSAC [20] orders the set of correspondences by a similarity function. Its samples are drawn from progressively larger sets of top-ranked correspondences. EVSAC [37] proposes a probabilistic parametric model that allows assigning a confidence value to each matching correspondence and thus accelerates iterations with hypothesis models. In NAPSAC [81] a new sampling strategy is proposed under the assumption that inliers tend to be closer to one another than outliers. GroupSAC [84] assumes that there exists some grouping between features in data. The grouping can come from prior information such as optical flow based clustering. To utilize this information, a binomial mixture model is introduced for sampling. Compared with these previous works, our method has a totally different sampling strategy with Lévy distribution.

3.3 The proposed method

The overall steps of our proposed fractional order RANSAC (FO-RANSAC) algorithm, as shown in Algorithm 3 and Fig. 3.1, has a similar structure to RANSAC but with additional ranking and nonuniform sampling steps. First, data association results $\mathcal{X}$ of input data are ranked with a similarity metric. Then hypothesis samples are drawn

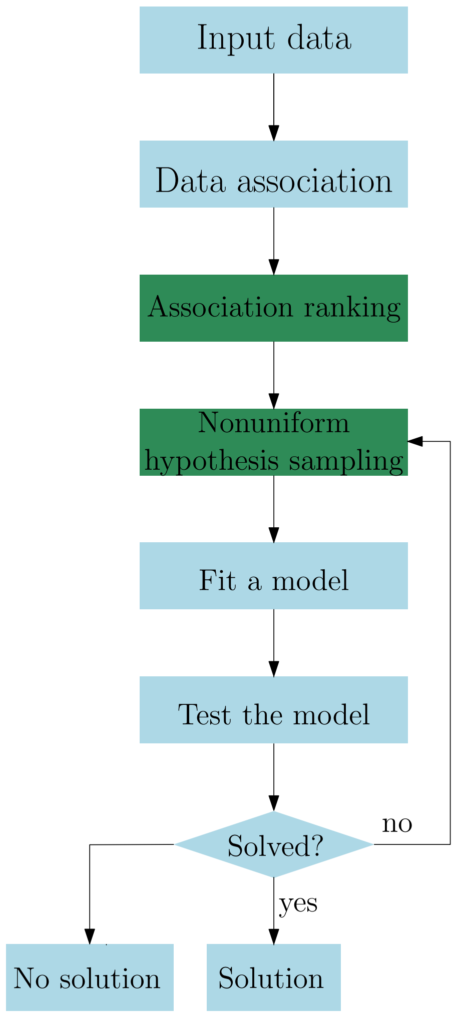


Figure 3.1: Diagram of the proposed fractional order RANSAC. The FO-RANSAC has a similar structure to RANSAC but with additional ranking and nonuniform sampling steps.

from ranked data with a nonuniform distribution rather than a uniform distribution as in the plain RANSAC. These hypotheses are tested to get a set of inlier points. Finally, the model which leads to the largest inlier set is selected as the result. For details of the ranking and sampling steps, we describe them in Secs. 3.3.1 and 3.3.2 respectively.

Algorithm 1: Fractional order RANSAC algorithm

```
   Input  : $\mathcal{X}, k_{max}, \tau$
   Output : $\theta^*, \mathcal{I}^*$
1  $k \leftarrow 0, \mathcal{I}^* \leftarrow \varnothing$
   // Data ranking
2  $\mathcal{X}^r \leftarrow \texttt{rank}(\mathcal{X}, by = \text{similarityMetric})$
3  while $k < k_{max}$ do
       // Hypothesis generation
4      Sample a subset of $m$ points with Lévy distribution from $\mathcal{X}^r$
5      Estimate model parameters $\theta_k$
       // Verification
6      $\mathcal{I}_k \leftarrow \texttt{findInliers}(\mathcal{X}, \theta_k, \tau)$
7      if $|\mathcal{I}_k| < |\mathcal{I}_{max}|$ then
8          $\theta^* \leftarrow \theta_k, \mathcal{I}^* \leftarrow \mathcal{I}_k$
9      end
10     $k \leftarrow k + 1$
11 end
```

3.3.1 Data association ranking

In many RANSAC use cases, input data are fed into a data association step, such as in image matching and point cloud registration. For these cases, it is beneficial to explore useful information for better convergence in an equal or fewer number of iterations. In our proposed method, we rank these correspondence pairs in descending order in terms of the likelihood to be an inlier pair. Ranking data helps to put good pairs of point matches to the front and the ones that are more likely to be the wrong ones to the end.

Different ranking methods may be chosen for different problems. In section 3.5, we provide a simple yet efficient ranking metric for point cloud registration problem.

3.3.2 Nonuniform sampling

Lévy distribution is a probability distribution that is both continuous, for non-negative random variables, and stable [46]. The Lévy distribution is a heavy-tailed distribution because its tail probabilities decay more slowly than those of any exponential distribution. This heavy-tailedness of the Lévy distribution has a direct connection to fractional calculus [88]. Thus, we call the proposed method fractional order RANSAC. The Lévy distribution follows the probability distribution function as in (3.1).

$$f(x;\mu,c)=\sqrt{\frac{c}{2\pi}}\ \frac{e^{-\frac{c}{2(x-\mu)}}}{(x-\mu)^{3/2}}, \tag{3.1}$$

where c is the scale parameter ($c>0$) and μ is the location parameter. It is meaningful when $x \geq \mu$.

We sample from Lévy distribution such that it can give more weight on top-ranked correspondences while not leave the less likely ones behind. It can also approximate uniform distribution so that it can degenerate to the plain RANSAC in the worst case.

A special step needs to be considered because hypothesis sampling is integer index sampling. These indices are within a range. To solve this, we sample from truncated Lévy distribution in range $[0, m]$. Then we scale the random numbers to $[0, n]$ and round them to the nearest integer. With the truncated Lévy distribution that follows a probability distribution function (PDF) as (3.1), we can get PDFs of different shape if they have different parameters. In Fig. 3.2, when we truncate and normalize the PDFs within $(0, \text{sample range})$ with different μ, we get different curves. When $\mu \to \infty$, the curve is flatter thus more similar to the PDF of a uniform distribution.

3.4 Simulation

In order to test the performance of different hypotheses sampling strategies, we create a simulation that is informative and easy to perform. In this simulation, a vector of Boolean is generated as input data. This vector simulates a set of feature correspondences

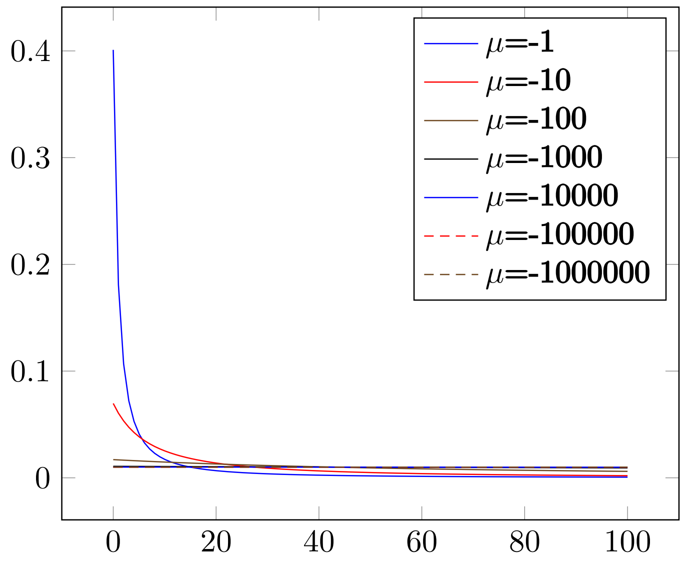


Figure 3.2: PDFs of Lévy distribution within $(0, 100)$ for $c = 1$ with different parameter μ values. By adjusting μ, we can get different variations of nonuniform distribution.

with `true` as an inlier pair and `false` as an outlier pair. This vector is generated in a way that the indices of `true` elements follow truncated Lévy distribution.

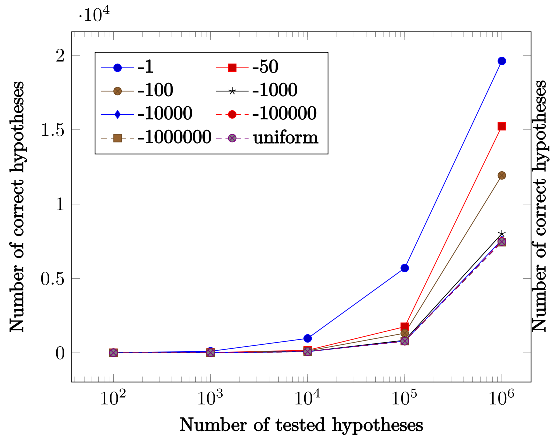
In our experiments, we create Boolean input vector data using Lévy distribution with $\mu = -10$ and $c = 1$. The Boolean vector will have `true` value on Lévy sampled index positions and `false` on all the other locations. We use 3000 as the length of data and 300 is the number of inliers.

We evaluate eight different hypothesis sampling strategies:

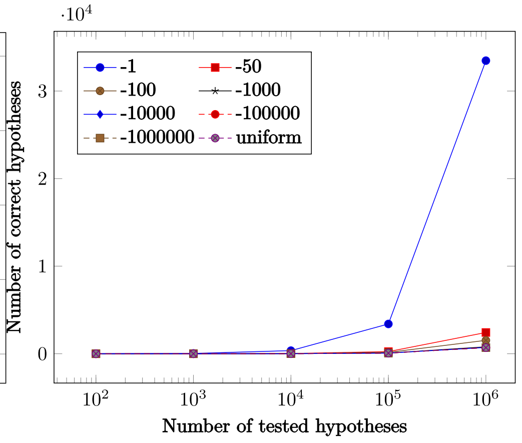
- Uniform distribution
- $\mu = \{-1, -50, -10^2, -10^3, -10^4, -10^5, -10^6\}$, $c = 1$ for Lévy distributions

We also change number of samples in a hypothesis from $\{2, 3, 4\}$ and number of hypotheses from $\{10^2, 10^3, 10^4, 10^5, 10^6\}$ to explore performance difference under different coverage settings.

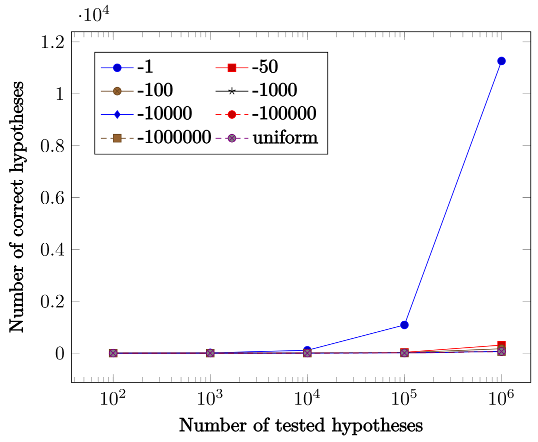
We run experiments with all the configurations specified above and plot results in Fig. 3.3. Under all the different sampling strategies, the Lévy distribution with $\mu = -1$ performs the best, much better than uniform sampling distribution. In Fig. 3.3 (a), when the number of hypotheses goes up, the results become less diverge. But when the hypothesis number is low, a more skewed distribution leads to a better result. More



(a) Sample size 2



(b) Sample size 3



(c) Sample size 4

Figure 3.3: Number of unique outlier-free hypotheses under difference experiment configurations

interestingly, In Fig. 3.3 (c), the performance is better with 10^5 hypotheses tested when $u = -1$ than the rest of u values even with an order of magnitude more hypotheses tested.

3.5 Point cloud registration

We further evaluate our method on the point cloud registration problem. We use the augmented ICL-NUIM point cloud registration dataset [18] to quantitatively analysis performance of our method. This dataset has four sets of point clouds and groundtruth rigid transformation between all possible point cloud pairs that overlap more than 30%. The evaluation metrics are precision and recall defined as in (3.2):

$$\begin{aligned} \text{precision} &= \frac{TP}{TP + FP} \\ \text{recall} &= \frac{TP}{TP + FN} \end{aligned} \tag{3.2}$$

where TP is true positive; TN is true negative; FP is false positive and FN is false negative. Both precision and recall are the higher the better.

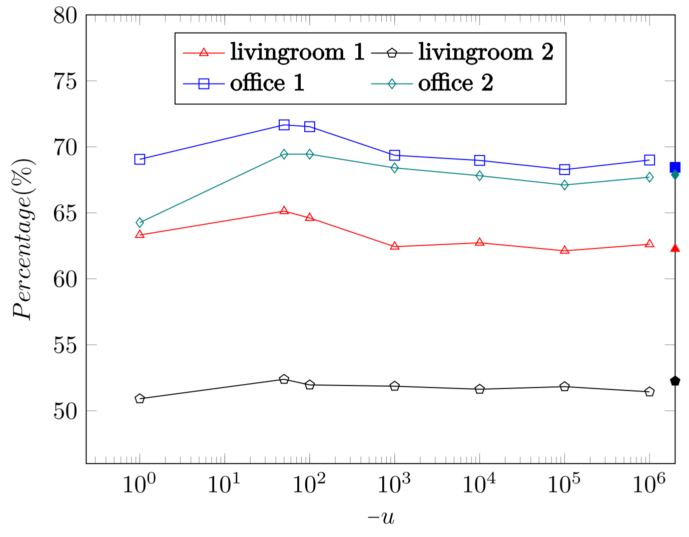
For data association ranking in point cloud registration, we use a simple yet efficient ranking metric. Given a pair of feature vectors (x_1, x_2) from two points, we have

$$r = \left|\left\{(x_1, x_2) : \|(x_1 - x_2)\|^2 < \tau_d^2\right\}\right|. \tag{3.3}$$

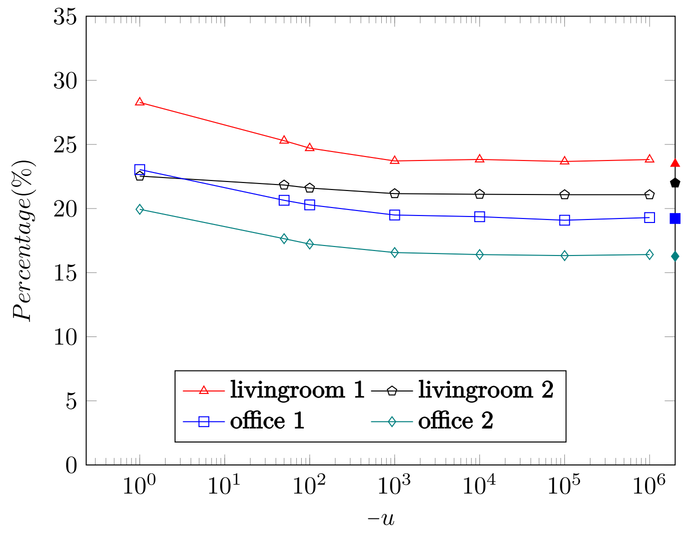
Equation (3.3) maps the feature vector pair to a metric space $r \in R$ by counting the number of data points that fall in the hypersphere spanned by radius τ_d such that $r > r'$ when $p(x_1, x_2) > p(x_1', x_2')$, where $p(x_1, x_2)$ denotes the probability of the point correspondence pair being a correct one. τ_d was set to an empirical value of 12 in all our experiments. One may tune this parameter with training data on a new dataset.

3.5.1 Experiments on the Aug-ICL-NUIM dataset

We compare the performance of our proposed method to the baseline RANSAC that samples from a uniform distribution. We run experiments on this dataset with different sampling distributions. We first use the same setting as in section 3.4: eight Lévy

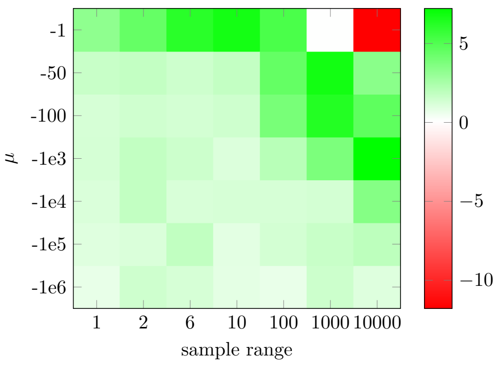


(a) Recall

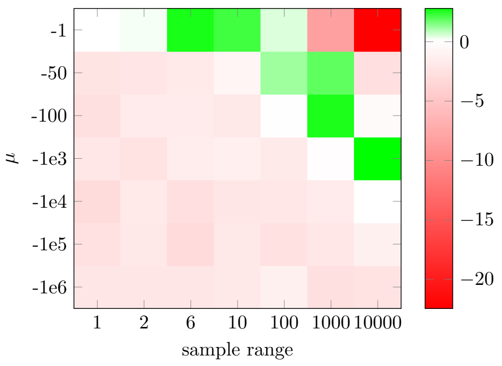


(b) Precision

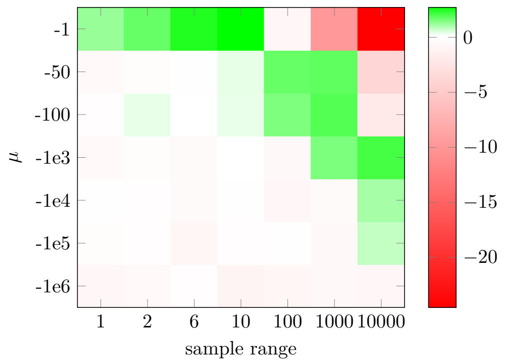
Figure 3.4: Recall and precision of RANSAC on Aug-ICL-NUIM data with our hand-crafted ranking method. All the performance results are from the average of 20 Monte Carlo runs. For each subfigure, the marks on the right most axis are the performance of the uniform distribution. One can see that there are a few μ values that lead to better recall results than the baseline. There is consistency in the performance along μ. The performance starts low when μ is very small. Then it goes up when μ is around -50. Then converge to the performance of uniform distribution. From this figure, we can see a clear performance benefit. More importantly, from the performance curves, we notice that the best value for μ is consistent across all the data sequences, which means that the method generalizes well on different data.



(a) 10,000 iterations



(b) 100,000 iterations



(c) 4,000,000 iterations

Figure 3.5: Average recall results of 10 runs on the Aug-ICL-NUIM data. The matrix plots are a visualization of recall performance of our nonuniform sampling strategy subtracted by uniform baseline recall. For each matrix cell, green means nonuniform sampling performing better and red means the uniform baseline method working better and white color means two sampling methods have a similar performance. Each subfigure is for a total iteration number configuration. From the three subfigures, one can see that the performance improvement of our method is greater when RANSAC runs a smaller number of iterations.

distributions and one uniform distribution. Experiments are repeated for 20 times to remove the influence of randomness on the final results. Results are shown in Fig. 3.4. Each curve is for a point cloud set.

In the results, we value more on a better recall because precision can be improved by post-processing, but recall cannot. We can see that there are a few μ values that lead to better recall results than the baseline. There is a consistency in the performance along μ. The performance starts low when μ is very small. Then it goes up when μ is around -50. Then converge to the performance of uniform distribution. From Fig. 3.4, we can see a clear performance benefit. More importantly, from the performance curves, we notice that the best value for μ is consistent across all the data sequences, which means that the method generalizes well on different data.

We further evaluate the effect of different scale parameter values for Lévy distribution by changing the sampling range. This sampling range was set to seven different values: $\{1, 2, 6, 10, 10^2, 10^3, 10^4\}$. Combined with the seven μ values from section 3.4, we get 50 configurations: 49 Lévy distribution configurations plus uniform sampling. After evaluation, the results, visualized as heatmaps in Fig. 3.5, show that there are several configurations that can lead to better recall performance. Another interesting observation is that these good configurations are continuous in an area. Thus applying a search algorithm becomes possible. Also, when comparing with just one parameter Lévy as in Fig. 3.4, the green area goes from top right to down left, which means it does not lose too much improvement if we only search on the location parameter μ.

We also test the influence of number of iterations for RANSAC in Fig. 3.5. The subfigures (a), (b), and (c) are from experiments with a different number of RANSAC iterations: 10^4, 10^5, and 4×10^6 respectively. From the three subfigures, one can see that the performance improvement of our method is greater when RANSAC runs a smaller number of iterations. This agrees with the results of the simulations. It means that the proposed method provides a way to lower the number of iterations with less performance penalty.

3.5.2 Experiments on the KITTI dataset

To evaluate the proposed method on real-world Lidar data in an autonomous driving scenario, we run experiments on the 3D visual odometry / SLAM dataset of the KITTI benchmarks [44] which is widely used for testing algorithms for autonomous vehicles. This 3D visual odometry / SLAM dataset consists of 22 stereo sequences, with a total length of 39.2 km. Each sequence contains images, LiDAR point clouds, and ground truth camera trajectories generated from IMU-RTK GPS data.

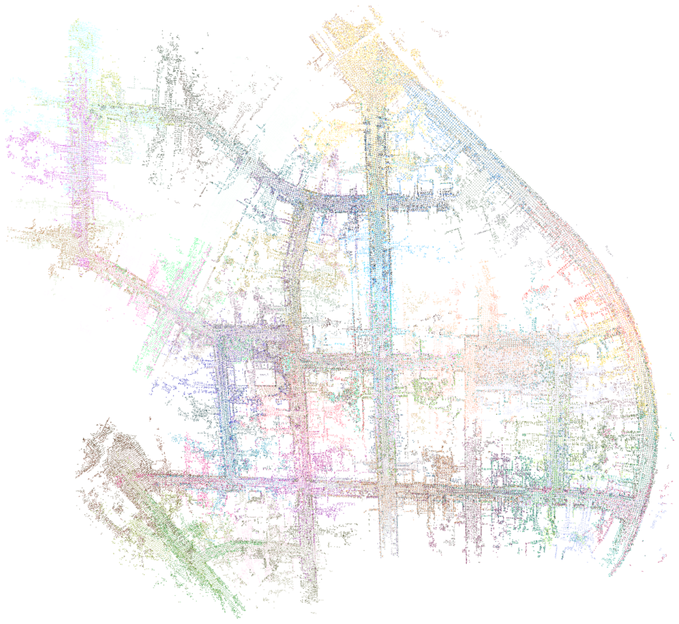
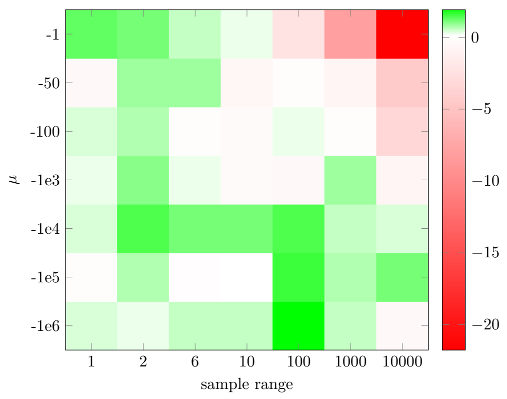
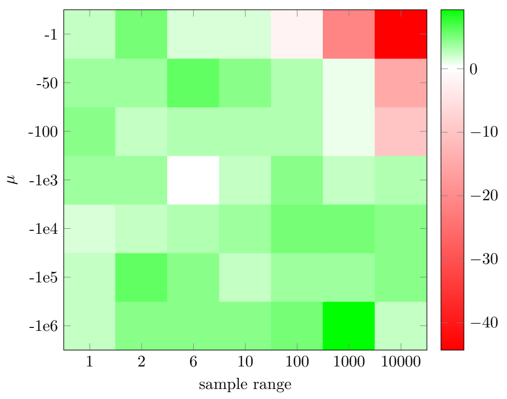


Figure 3.6: KITTI scene fragments visualization. We merge every $k = 50$ consecutive LiDAR frames into a scene fragment. Then these scene fragments are transformed to the world frame with the ground-truth trajectory. Each color represents a scene fragment.

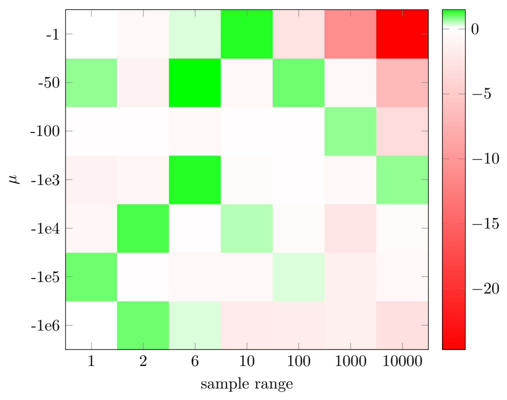
We further process this dataset to fit our point cloud registration evaluation purpose. Similar to how the Aug-ICL-NUIM dataset [18] is generated, we merge every $k = 50$ consecutive frames into a scene fragment, then a transformation is generated from the ground-truth camera trajectory. Registration ground truth data are generated when a pair of fragment shares more than 50% of matched points, which are defined as nearest-neighbor points within ϵ when they are transformed to the world frame. There are 11 sequences that have ground-truth trajectories publicly available. We use four sequences: `00`, `02`, `05`, `08` because other sequences do not have enough number of frames for



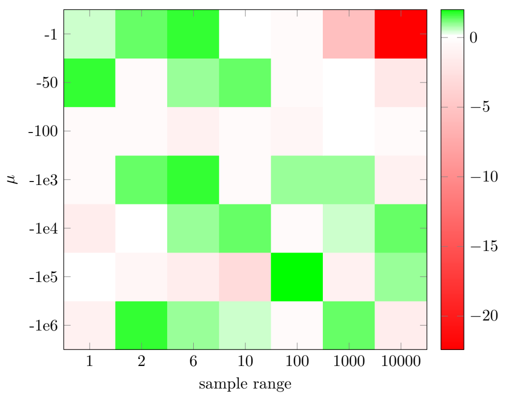
(a) Sequence 00



(b) Sequence 02



(c) Sequence 05



(d) Sequence 08

Figure 3.7: Average recall results of 10 runs on the selected KITTI sequences. The matrix plots are visualization of recall performance of our nonuniform sampling strategy subtracted by uniform baseline recall. For each matrix cell, green means nonuniform sampling performing better and red means the uniform baseline method working better and white color means two sampling methods have a similar performance. The results in these subfigures show that the proposed method performs better than the uniform baseline method in many cases for each sequence.

meaningful evaluation.

We run experiments on our processed data with the 50 configurations specified in section 3.5.1. The results in Fig. 3.7 shows that the proposed method performs better than the uniform baseline method in many cases for each sequence.

To test generalization ability of selected Lévy parameters, we run experiments with the best parameters from the Aug-ICL-NUIM dataset: $(\mu = -50, \text{sample range(sr)} = 100)$ and parameters for the best average result from Fig. 3.7: $(\mu = -10^6, \text{sample range} = 10^3)$, we get results in Table 3.1. We can see that even on a different dataset, the best parameter selected on the Aug-ICL-NUIM dataset can provide a performance improvement. This parameter generalization ability enables making parameter selection on a dataset with ground truth and then gets improvements on a new dataset with no ground truth information.

Table 3.1: Recall performance of selected Lévy parameters vs. uniform sampling.

Recall (%)	00	02	05	08	Average
Uniform	48.8	64.3	51.1	32.0	49.1
Ours $(\mu = -50, \text{sr} = 100)$	48.5	67.1	**51.9**	31.6	49.8
Ours $(\mu = -10^6, \text{sr} = 10^3)$	**49.3**	**73.6**	49.6	**33.2**	**51.4**

3.6 Fast GPU implementation

To make RANSAC run faster, we accelerated the most time consuming parts using GPU programming with an efficient nearest neighbor (NN) search method, and that normal checking is moved from pre-rejection part into hypotheses testing. Traditionally, RANSAC is formulated as an iterative process with proved convergence [36]. But different iterations and different hypotheses can be considered to be totally independent to each other. This means different hypotheses can be mapped to different processing cores to be tested in parallel.

As shown in Algorithm 2, point clouds are downsampled to the resolution of the typical precision of RGB-D sensors to reduce unnecessary computation. Fast Point Feature Histograms (FPFH) features are extracted for each point in $\boldsymbol{P}$ and $\boldsymbol{Q}$ for point correspondence pairs generation. To make nearest neighbor search more efficient, we

Algorithm 2: Global Registration on GPU

```
Input:  A pair of point cloud $\boldsymbol{P}$ and $\boldsymbol{Q}$
Output: $\boldsymbol{T_{QP}}$ if $\boldsymbol{P}$ and $\boldsymbol{Q}$ can be aligned together
1  Downsample $\boldsymbol{P}$ and $\boldsymbol{Q}$; Compute FPFH features $\boldsymbol{F}(\boldsymbol{Q})$ and $\boldsymbol{F}(\boldsymbol{Q})$;
2  $\boldsymbol{T}_{QP} \leftarrow \boldsymbol{I}$; $feature\_NN\_cache \leftarrow \emptyset$; Quadruple point set samples $S_P \leftarrow \emptyset$ ,
     $S_Q \leftarrow \emptyset$
   /* Parallel FPFH feature pre-matching                                   */
3  parallel for $i \leftarrow 1$ to $\boldsymbol{P}\_count$ do on GPU cores
4      $feature\_NN\_cache[i] \leftarrow$ NN of $\boldsymbol{F}(\boldsymbol{p}_i)$ in $\boldsymbol{F}(\boldsymbol{Q})$;
5  end
   /* Randomly sample hypotheses on GPU                                    */
6  parallel for $i \leftarrow 1$ to $\boldsymbol{P}\_count$ do on GPU cores
7      $S_P[i] \leftarrow$ randomly picked $(\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3)$ from $\boldsymbol{P}$ ;
8      forall $(\boldsymbol{p}_0, \boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3)$ do
9          $\boldsymbol{q}_j \leftarrow$ NN of $\boldsymbol{p}_j$ in $\boldsymbol{Q}$ using $feature\_NN\_cache[j]$
10     end
11 end
12 Stream compact non-rejected hypotheses;
   /* Consensus for the remaining quadruple pairs                          */
13 parallel for $i \leftarrow 1$ to $remain\_number$ do on GPU cores
14     Estimate transformation $\boldsymbol{T_{QP}}$ from $S_P[i]$ to $S_Q[i]$ ;
15     Find all inliers under $\boldsymbol{T_{QP}}$ hypothesis;
16     $test\_log \leftarrow$ inlier ratio, fitness score
17 end
   /* Get final result from test_log                                       */
18 Find the Hypothesis that has max inlier ratio and fitness score $>$ threshold in
     $test\_log$.
19 if Hypothesis found then
20     return its $\boldsymbol{T_{QP}}$;
21 end
```

pre-cached all the nearest neighbors of $\boldsymbol{P}$ in $\boldsymbol{Q}$ using FPFH feature distance. Then, for each hypothesis, 4 points are randomly sampled from $\boldsymbol{P}$, and their correspondences are found through the pre-cached nearest neighbors. After that, a pre-rejection step, which rejects hypotheses whose point pairs cannot make a similar polygon, is performed. τ is a similarity threshold and set to 0.9 in all our experiments. Then, hypotheses testing, the most time consuming step, tests both inlier ratio and fitness score on GPU. When implement it, we test each hypothesis on a thread block with efficient parallel reductions. For the NN search during hypotheses testing, we utilized a 3D grid to replace the k-d tree to fit special need of a GPU, since a GPU will slow down when different threads go to different code branches during k-d tree search. We propose to use a 3D grid for NN search, given that the point clouds to be searched only span in a limited area. This guarantees that we can use a grid with limited size for NN searching without jeopardizing searching accuracy. When a point is stored into the search grid, the i_x, i_y, i_z indices of its cell is calculated by (3.4).

$$\begin{aligned} i_x &= (x_p - x_c)/l, \\ i_y &= (y_p - y_c)/l, \\ i_z &= (z_p - z_c)/l, \end{aligned} \tag{3.4}$$

where the x_c, y_c, z_c are the coordinate of the center of target point cloud. l is the cell edge length of the NN 3D grid. It is subtracted so that the translation of point cloud does not affect searching. When a point wants to query its nearest neighbor, the searching is accomplished through table looking up, which has a time complexity of $\mathcal{O}(1)$, given cell edge length the same as point cloud downsample resolution. It is faster than the k-d tree which has a $\mathcal{O}(\log n)$ time complexity. We observed speedup by only replacing k-d tree with 3D search grid in CPU only code. More importantly, there is no branching during searching, so it fits much better on a GPU than the k-d tree.

3.6.1 Results

We run experiments to compare speed performance against the CPU baseline method implemented by CZK [18] on redwood pairwise registration evaluation dataset by Choi *et al*. We report results in Table 3.2. For both methods, we use the same hyper-parameter values as in published code of [18]: 0.05 as point cloud downsampling leaf size, 0.1 m as normal estimate radius, 0.25 m as FPFH feature estimate radius, 4,000,000 as hypotheses count and 0.075 m as maximum point correspondence distance. We use an Intel i7-6850K clocked at 3.6 GHz and an NVIDIA Titan X Pascal for our evaluation.

Our global point cloud registration can finish in around 20 milliseconds, which is around 366 times faster than CZK as in Table 3.2. With this speed, it can run at 50 Hz, which means we can process more loop candidates.

Table 3.2: Computational performance evaluation of different point cloud registration methods. Execution time in milliseconds per point cloud pair

	CZK on CPU	Ours
Living room 1	7606.20	**24.71**
Living room 2	7469.58	**18.19**
Office 1	7556.02	**21.63**
Office 2	7418.12	**17.45**
Average	7512.23	**20.50**

3.7 Chapter summary

In this chapter, we propose a RANSAC based algorithm that samples data with a Lévy distribution after data ranking. The proposed method is evaluated on both simulation and real-world public datasets. In experiments, our method shows better results compared with the uniform baseline method.

In the future, data-driven approaches for the ranking algorithm can be explored so that the ranking becomes more accurate, which enables more skewed nonuniform distributions.

We also propose a new CUDA-accelerated acceleration of the improved RANSAC algorithm. We accelerated the most time-consuming parts using GPU programming with an efficient nearest neighbor search method. Experiments show that it is fast enough for online use. The algorithm implementation can finish in around 20 milliseconds on a NVIDIA Titan X Pascal GPU.

Chapter 4

Online Sifting of Loop Detections for 3D Reconstruction of Caves

4.1 Introduction

It has long been recognized that cave sites often contain the best-preserved material in the archaeological record. Cave archaeology has developed its methodologies for mapping and recording sites, yet few sites are mapped to true 3D models because it is a slow and tedious process for archaeologists to record and book-keep caves. They need to incrementally set up baseline along the cave and then measure the distance from the baseline to cave walls or objects of interest and mark walls or objects in a 2D map by hand [126]. This slow process has a major negative impact on cultural relic preservation. Typically, archaeological teams will visit a site and begin to record it in one year, but when they come back to finish data collection it has been looted, artifacts stolen, architecture destroyed and the archaeological record disturbed. Therefore, archaeologists need a faster, more efficient method of surveying and recording the sites.

To accelerate cave mapping, a system that can automate the process needs to be developed. In this system, we first focus on building globally consistent and accurate 3D models using RGB-Depth (RGB-D) data recorded with unmanned aerial vehicles (UAVs).

Motivated by the fact that humans can notice mismatches in 3D models very easily

Figure 4.1: A photo shows that our team members are collecting RGB-D data in a cave at Las Cuevas, Belize.

by looking at the spatial displacement of surfaces, we propose to resolve mismatches directly by closing surface loops to get a consistent 3D model and a precise camera trajectory estimate in the vSLAM system. After surface loops detected, instead of optimizing surface directly to propagate correction introduced by surface loop, in this chapter, the surface loop correction is done through sparse feature bundle adjustment, so that all the past camera poses can be corrected based on their observations. By running extensive experiments on different datasets, we observed that combining sparse features with surface loop closure can produce better results. Not only 3D models get improved, but also camera trajectories estimate becomes more accurate. This is because our framework can detect loops in the dense surface domain and optimize loops in the sparse feature domain. Note that our framework can detect surface loops, yet other means of detecting loops can still be utilized.

In the following, we summarize the key contributions of our method:

1. We propose a novel 3D reconstruction system that corrects surface loops with sparse feature-based bundle adjustment. We demonstrate that this novel system

can give much-improved camera tracking and dense modeling results.

2. We propose a novel objective function for surface loop filtering with a sparse feature-based optimization graph. This graph is more robust to different scan patterns and can cope with tracking failure and recovery so that there is more flexibility for UAVs to fly and record data. In addition to the flexibility, experiments show that it performs better than state-of-art methods when only a limited number of loops are detected.

4.2 Overview of the proposed system

The method proposed aims at producing accurate 3D models by detecting and optimizing surface loops in sparse feature-based visual SLAM systems. By adding surface loop closure into a vSLAM system, we can get globally consistent and optimal 3D models. With our proposed algorithms, we build a novel system with five components: 1) tracking, 2) surface model fusion, 3) fast surface loop detection, 4) surface loop filtering, 5) loop optimization.

Tracking. We employ the tracking part from ORB-SLAM2 [80], which is a very well implemented sparse feature-based SLAM system. Inside this tracking module, the Oriented FAST and Rotated BRIEF (ORB) features are extracted for keypoint matching. Then frames are tracked against keyframes with motion estimate and then refined with a local sparse map. Keyframes are generated when tracking is weak, or the local bundle adjustment thread is free. Local BA is used to correct the re-projection error of feature correspondences among co-visible keyframes in a background thread. This tracking module provides camera poses for each frame and a co-visibility graph across keyframes.

Surface model fusion. We fuse surface models on a GPU using surfels as a map representation similar to [61, 114]. Each surfel has a position $\boldsymbol{p}$, normal vector $\boldsymbol{n}$, radius r, confidence c. We fuse keyframes within every k frames ($k = 50$ for all experiments) into a surface fragment. These surface fragments are generated for two reasons. One is to integrate out raw RGB-D data noise. Another one is to reduce the number of 3D pieces, so that loop detection computation is accelerated. After each scene fragment is generated, for the later optimization process, it is linked to the keyframe whose timestamp is closest to the first frame within the range.

Fast surface loop detection. The proposed surface loop detection is done by point cloud registration on surface fragments. To detect surface loops effectively, the co-visibility graph from tracking is utilized to pre-filter co-visible surface fragment pairs that are already connected. Thus a majority of unnecessary computation can be avoided. To detect surface loops efficiently, we propose using CUDA to accelerate point cloud registration. Details of this acceleration is described in Chapter 3.

Surface loop sifting. After loop candidates are detected, They need to go through a novel online loop sifting algorithm, so that false loops would not diverge the subsequent optimization process. This is in section 4.3.

Loop optimization. After surface loops are detected and verified, the loop pairs are used to connect pose graph vertices and also trigger more image loop detection, which again uses spatial prior and ORB feature matching. Then, the pose graph is optimized to give a coarse pose correction and then a full BA is performed in order to get optimally fine-tuned camera trajectory estimate. Details are in section 4.4.

4.3 Online surface loop sifting

The results out of the surface loop detection algorithm in Chapter 3 have a low precision problem. Choi *et al*. proposed to use a line process-based optimization to solve it [18]. But that method requires scanning to be tracked fully successfully from the beginning to the end, and each surface fragment cannot be empty. However, during a long scanning session, it is almost impossible to guarantee that RGB-D cameras always have surfaces observed within their effective range. A failure to maintain that will lead to an empty surface fragment. Thus the graph vertices are potentially divided into more than one sub-graph, which leads to erroneous optimization results. A similar problem will occur when tracking failure is present in the tracking part, which will break the optimization graph as well. To address this problem, we propose to minimize (4.1) instead, which has a supporting optimization graph that is always fully connected whenever sparse features are matched from either tracking or failure recovery.

$$\begin{aligned}\min_{\mathbb{X},\mathbb{T},\mathbb{S}} \quad & \sum_{c}\sum_{v\in V(c)} \|\tilde{\boldsymbol{x}}_v^c - \boldsymbol{K}\boldsymbol{T}_{cw}\boldsymbol{X}_v\|_\Sigma \\ & + \lambda\sum_{ij} f(\boldsymbol{T}_{iw}, \boldsymbol{T}_{jw}, s_{ij}|\mathcal{K}_{ij}) \\ \text{s.t.} \quad & 0 < s_{ij} < 1, \text{for all } i,j \text{ pairs,}\end{aligned} \tag{4.1}$$

where

$$\begin{aligned}& f(\boldsymbol{T}_{iw}, \boldsymbol{T}_{jw}, s_{ij}|\mathcal{K}_{ij}) \\ = & \sum_{(\boldsymbol{p}_i,\boldsymbol{p}_j)\in\mathcal{K}_{ij}} \left\|\Psi(s_{ij})\cdot(\boldsymbol{p}_i - \boldsymbol{T}_{iw}\boldsymbol{T}_{jw}^{-1}\boldsymbol{p}_j)\right\|^2 \\ & + ||\gamma_{ij} - s_{ij}||^2_{\boldsymbol{\Omega}_{ij}}.\end{aligned} \tag{4.2}$$

The objective function is minimized over a set of point cloud $\mathbb{X} = \{\boldsymbol{X}_v \in \mathbb{E}^3\}$ estimated from sparse feature key points, camera trajectory $\mathbb{T} = \{\boldsymbol{T}_{cw} \in SE(3)\}$, and a set of switch variable $\mathbb{S} = \{s_{ij} \in \mathbb{R}\}$. The first term in (4.1) builds a least squares optimization graph for bundle adjustment from sparse keypoint observations, where $\boldsymbol{X}_v$ is the 3D coordinate of a point visible in the c-th camera. $\tilde{\boldsymbol{x}}_v^c$ is the 2D pixel observation coordinate of the 3D point $\boldsymbol{X}_v$. $\boldsymbol{K}$ is the camera intrinsics matrix. Σ is the covariance matrix associated to the scale of the keypoint. The second term, weighted by a factor λ, is surface loop connections with switchable constraints [105] with s_{ij} as a switch variable for surface loop connection that connects keyframes i and j. Let P_i be the surface fragment referred by keyframe i, $\mathcal{K}_{ij}$ is the set of nearest neighbor correspondence pairs between $\boldsymbol{T}_{iw}^{-1}\boldsymbol{P}_i$ and $\boldsymbol{T}_{jw}^{-1}\boldsymbol{P}_j$ that are within distance ε = 0.05 m, which is typical noise level of RGB-D sensor.

In (4.2), $\Psi(s_{ij})$ is a switch function and we use a linear function $\Psi(s_{ij}) = s_{ij}$ as suggested in [105]. $\boldsymbol{\Omega}_{ij}$ is information of switchable prior constraints. It controls the influence of the loop candidate tested. Let $\boldsymbol{T}_{ij}$ be the transformation that align all the $\boldsymbol{p}_j$ to related $\boldsymbol{p}_i$ in $\mathcal{K}_{ij}$, *i.e.*

$$\boldsymbol{p}_i \approx \boldsymbol{T}_{ij}\boldsymbol{p}_j, \text{ for all } (\boldsymbol{p}_i, \boldsymbol{p}_j) \in \mathcal{K}_{ij}. \tag{4.3}$$

Then

$$\begin{aligned}&\sum_{(\boldsymbol{p}_i,\boldsymbol{p}_j)\in\mathcal{K}_{ij}} \left\|\Psi(s_{ij})\cdot(\boldsymbol{p}_i - \boldsymbol{T}_{iw}\boldsymbol{T}_{jw}^{-1}\boldsymbol{p}_j)\right\|^2 \\ \approx &\sum_{(\boldsymbol{p}_i,\boldsymbol{p}_j)\in\mathcal{K}_{ij}} \left\|\Psi(s_{ij})\cdot(\boldsymbol{p}_i - \boldsymbol{T}_{iw}\boldsymbol{T}_{jw}^{-1}\boldsymbol{T}_{ij}^{-1}\boldsymbol{p}_i)\right\|^2.\end{aligned} \tag{4.4}$$

To accelerate computation of (4.4), we follow an approximation proposed in [18]. The local parameterization of $\boldsymbol{T}_{iw}\boldsymbol{T}_{jw}^{-1}\boldsymbol{T}_{ij}^{-1}$ is represented with a 6D vector $\boldsymbol{\xi} = (\boldsymbol{\omega}, \boldsymbol{t})^T = (\alpha, \beta, \gamma, x, y, z)^T$, which consists of three rotational angles α, β, γ and three translation components x, y, z. Since its rotation is small under the assumption that the registration result is good and camera pose estimates are not far away from the correct solution, the approximations $\sin(\theta) \approx \theta$ and $\cos(\theta) \approx 1$ are utilized for all α, β, γ. Then

$$\begin{aligned}\boldsymbol{T}_{iw}\boldsymbol{T}_{jw}^{-1}\boldsymbol{T}_{ij}^{-1}\boldsymbol{p}_i &\approx \left[\begin{array}{ccc|c} 1 & -\gamma & \beta & x \\ \gamma & 1 & -\alpha & y \\ -\beta & \alpha & 1 & z \\ \hline 0 & 0 & 0 & 1 \end{array}\right]\boldsymbol{p}_i \\ &= \begin{bmatrix} 1 & -\gamma & \beta \\ \gamma & 1 & -\alpha \\ -\beta & \alpha & 1 \end{bmatrix}\boldsymbol{p}_i + \boldsymbol{t} \\ &= \left(\boldsymbol{I} + \begin{bmatrix} 0 & -\gamma & \beta \\ \gamma & 0 & -\alpha \\ -\beta & \alpha & 0 \end{bmatrix}\right)\boldsymbol{p}_i + \boldsymbol{t} \\ &= \boldsymbol{p}_i + \boldsymbol{\omega} \times \boldsymbol{p}_i + \boldsymbol{t} \\ &= \boldsymbol{p}_i - \boldsymbol{p}_i \times \boldsymbol{\omega} + \boldsymbol{t}.\end{aligned} \tag{4.5}$$

Plug (4.5) into (4.4):

$$\begin{aligned}&\sum_{(\boldsymbol{p}_i,\boldsymbol{p}_j)\in\mathcal{K}_{ij}} \left\|\Psi(s_{ij})\cdot(\boldsymbol{p}_i - \boldsymbol{T}_{iw}\boldsymbol{T}_{jw}^{-1}\boldsymbol{p}_j)\right\|^2 \\ \approx &\sum_{(\boldsymbol{p}_i,\boldsymbol{p}_j)\in\mathcal{K}_{ij}} \left\|\Psi(s_{ij})\cdot(-\boldsymbol{p}_i \times \boldsymbol{\omega} + \boldsymbol{t})\right\|^2 \\ = &\sum_{(\boldsymbol{p}_i,\boldsymbol{p}_j)\in\mathcal{K}_{ij}} \left\|\Psi(s_{ij})\cdot[-[\boldsymbol{p}_i]_\times|\boldsymbol{I}]\cdot\boldsymbol{\xi}\right\|^2 \\ = &\left\|\Psi(s_{ij})\cdot\boldsymbol{\xi}\right\|^2_{\Lambda_L},\end{aligned} \tag{4.6}$$

where

$$\Lambda_L^2 = \sum_{(\boldsymbol{p}_i, \boldsymbol{p}_j) \in \mathcal{K}_{ij}} [-[\boldsymbol{p}_i]_\times | \boldsymbol{I}]^T \cdot [-[\boldsymbol{p}_i]_\times | \boldsymbol{I}], \tag{4.7}$$

where,

$$[-[\boldsymbol{p}_i]_\times | \boldsymbol{I}] = \begin{bmatrix} 0 & z_{\boldsymbol{p}i} & -y_{\boldsymbol{p}i} & 1 & 0 & 0 \\ -z_{\boldsymbol{p}i} & 0 & x_{\boldsymbol{p}i} & 0 & 1 & 0 \\ y_{\boldsymbol{p}i} & -x_{\boldsymbol{p}i} & 0 & 0 & 0 & 1 \end{bmatrix}, \tag{4.8}$$

where $x_{\boldsymbol{p}i}, y_{\boldsymbol{p}i}, z_{\boldsymbol{p}i}$ are the three components of $\boldsymbol{p}_i$. Then our (4.2) becomes:

$$f(\boldsymbol{T}_{iw}, \boldsymbol{T}_{jw}, s_{ij} | \mathcal{K}_{ij}) = \|\Psi(s_{ij}) \cdot \boldsymbol{\xi}\|_{\Lambda_L}^2 + ||\gamma_{ij} - s_{ij}||_{\boldsymbol{\Omega}_{ij}}^2. \tag{4.9}$$

After optimizing (4.1), keyframes i and j should be connected as a loop if the optimized value of switch variable s_{ij} is greater than a threshold.

4.3.1 Surface loop sifting evaluation

Experiment design. To understand the performance of the proposed method and compare it with CZK, we run experiments on the Augmented ICL-NUIM dataset for loop filtering evaluation. In the experiments, RGB-D frames are first fused into scene fragments with implementation provided by CZK. Point cloud registration results from CZK are used as loop detections. Two sets of experiments are conducted: One takes all the successful point cloud registration results as loop detections, denoted as All pairs in Table 4.1. Another set of experiments only consider point cloud pairs that are not co-visible in ORB-SLAM2 tracking results as loop detections, which is denoted as Only non-co-visible pairs in Table 4.1. A pair of scene fragments is co-visible if there are any co-visible frames between two sets of frames contained in the two scene fragments. This only non-co-visible set of loops is more close to practical use cases because the co-visible pairs can be ignored for computational speed consideration and have been well connected in SLAM systems, already.

Observations. When only non-co-visible pairs are presented, the proposed method outperforms CZK in average precision, while recall is only 2% less. For `office 1` sequence, our method gives better results in both precision and recall, while CZK output only 62.5% precision. Such a low precision usually causes serious problems because

Table 4.1: Performance evaluation of different loop filtering methods reported in percentage

Recall (%) / Precision (%)	All pairs			Only non-co-visible pairs		
	Registration	CZK	Ours	Registration	CZK	Ours
Lvingroom 1	61.2 / 27.2	**57.6 / 95.1**	43.5 / 93.7	48.8 / 13.0	**48.8 / 92.2**	36.4 / 91.7
Livingroom 2	49.7 / 17.0	**49.7 / 97.4**	47.1 / 97.3	30.2 / 5.2	**30.2 / 94.1**	28.3 / 93.8
Office 1	64.4 / 19.2	**63.3** / 98.3	34.4 / **98.4**	27.0 / 2.3	**27.0** / 62.5	**27.0 / 100.0**
Office 2	61.5 / 14.9	**60.7 / 100.0**	58.5 / 96.3	29.3 / 2.9	22.0 / **100.0**	**29.3** / 92.3
Average	59.2 / 19.6	**57.8 / 97.7**	45.9 / 96.4	33.8 / 5.9	**32.0** / 87.2	30.3 / **94.5**

too many false loops are wrongly involved in the loop optimization step. For the other three sequences, our method gives competitive results. When all pairs are considered, CZK performs so well that our method is closely under it. The difference is mainly in recall, while precision difference is very small. Considering precision is more important than recall for loop optimization, we would like to note that this minor difference rarely impacts on the final loop optimization result.

In addition to the cases where our method performs better, note that our method is less strict to use in practical scenarios, especially on a fast-moving UAV platform. Because our method does not require maintaining RGB-D cameras facing surfaces all the time, which it is required by CZK. This requirement difference is inherently implied by underlying optimization pose-graphs.

4.4 Loop optimization

After a surface loop passing verification, map optimization is followed to reduce mismatches and errors. We employ pose-graph optimization and new data association and finally run a full bundle adjustment to get a maximum-a-posteriori (MAP)-based optimization to correct the camera trajectory estimate and thus improve the 3D model.

When a surface loop pair $\{i, j\}$ pass loop verification, we then try to find data association in the sparse feature domain. This is done by first retrieve all the co-visible keyframes for both keyframe i and j noted as $\mathcal{F}_i$ and $\mathcal{F}_j$. Then collect all the local sparse map points $\{X_{\boldsymbol{p}}\}_i$ and $\{X_{\boldsymbol{p}}\}_j$ that are observed by $\mathcal{F}_i$ and $\mathcal{F}_j$ respectively. Then data association between $\{X_{\boldsymbol{p}}\}_i$ and $\{X_{\boldsymbol{p}}\}_j$ is constructed by both distance in image feature space and Euclidean space. After that, we run RANSAC to filter out outliers in the matches and also improve the transformation $\boldsymbol{T}_{ij}$. If this process converges with enough inlier matches, we add a loop connection for keyframe i and j. And update keyframe connections by merging the observations of matched sparse map points. These merged map points will help improve during the final BA process. If it does not converge, we still add a loop connection with the $\boldsymbol{T}_{ij}$. We do this because in some cases, even though the areas related to keyframe i and j do not have enough sparse map points for a good matching result, a loop connection can lead to a better initial start point for bundle adjustment.

Finally, a full bundle adjustment is performed for all the keyframes and observed map points by optimizing:

$$\min \sum_{c} \sum_{v \in V(c)} \left\| \tilde{\boldsymbol{x}}_p^c - \boldsymbol{K}\boldsymbol{T}_{cw}\boldsymbol{X}_v \right\|_{\Sigma}. \tag{4.10}$$

4.5 Evaluation of the full 3D reconstruction system

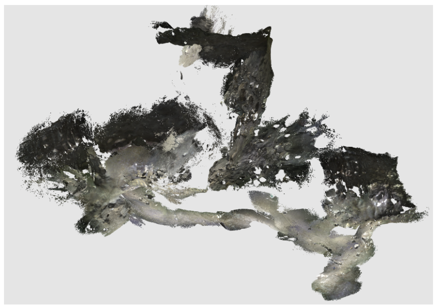
In addition to experiments in the fast surface loop detection section and surface loop filtering section, extensive experiments are performed to evaluate the proposed full 3D reconstruction system on multiple datasets: our Maya cave dataset, augmented ICL-NUIM [18], TUM RGB-D dataset [104], SUN3D dataset [119] and some other public data sequences. Comparisons are made with other online and offline methods. There are many SLAM algorithms and implementations. Here, we choose baseline methods in a way that they can best show the characteristics of our proposed system: We choose ORB-SLAM2 [80] since we use the tracking part of it and offline method CZK [18] because our loop filtering part is inspired by it. We denote the tracking part of ORB-SLAM2 as tracking and full ORB-SLAM2 as ORB-SLAM2 in all experiments. Results of baseline methods are from original papers or their authors when available.

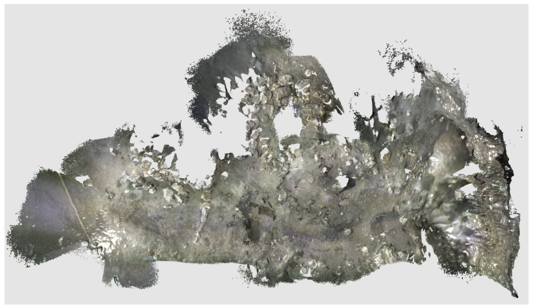
4.5.1 Maya cave dataset

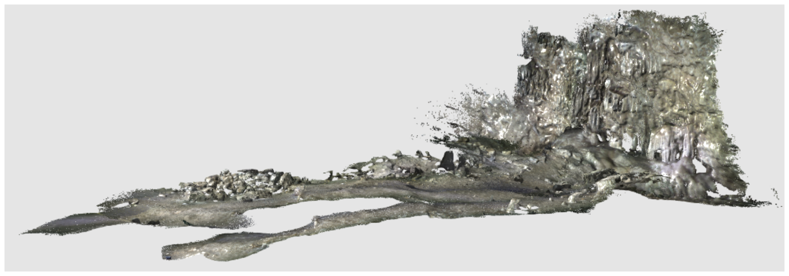
The Maya cave dataset is a dataset collected by a team of archaeologists using an RGB-D sensor Kinect V1 in Maya caves at Las Cuevas, Belize. LED lights are used to light up the environment as in Fig. 4.1. In this dataset, caves are scanned with a loopy motion for more loop optimization.

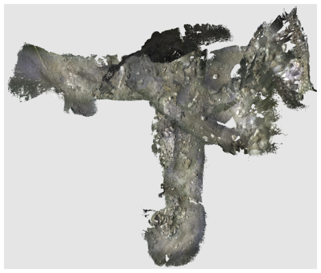
Experiment design and baseline methods. We evaluate modules of both tracking and loop closure from different approaches on the data we collected.

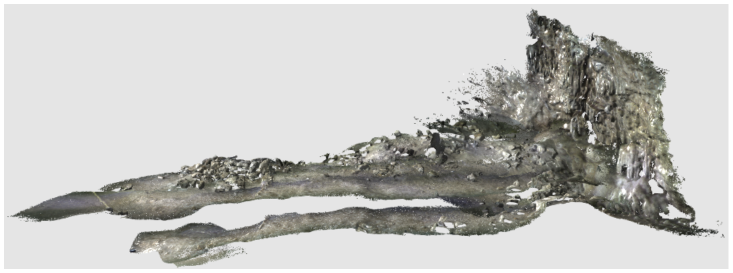
For RGB-D data, there are two major different camera tracking methods, which are sparse feature-based and dense frame-to-model approaches. We choose the ORB-SLAM2 as the implementation for the sparse feature tracking and the ElasticFusion for the dense frame-to-model implementation. In experiments for tracking, both implementations have their loop closure disabled so that the difference can reflect tracking


(a) ElasticFusion RGB-D tracking


(b) ElasticFusion Depth tracking


(c) ORB-SLAM2 tracking


(d) ElasticFusion full


(e) ORB-SLAM2 full

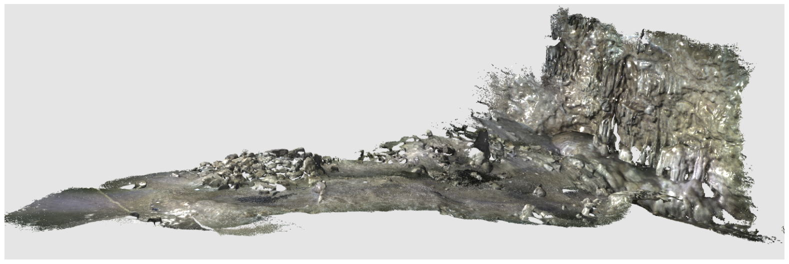
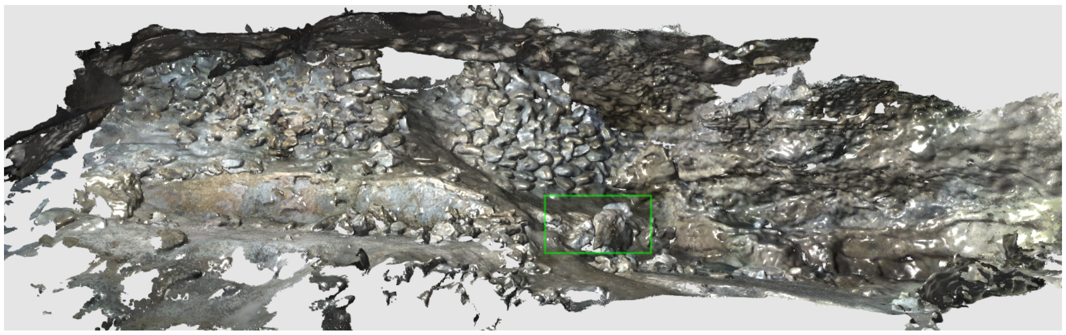
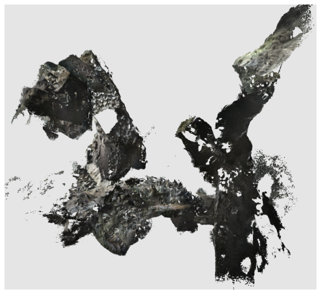

(f) Ours

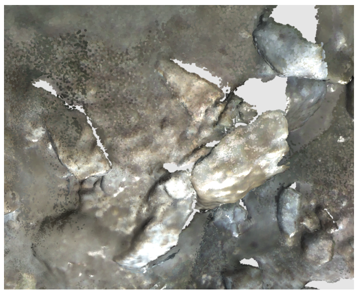
Figure 4.2: Results on the chamber-floor-walking sequence. RGB-D (a) and depth (b) tracking of ElasticFusion do not work reliably on cave data. Also, it is almost impossible for them to recover from the errors in the full system (d), because there are no strong data associations. The tracking part of ORB-SLAM2 (c) performs relativly well but the full system (e) failed to connect a few important loops so that the cave floor gets seperated into two layers. In (f), the proposed system produces the best result with no major mismatch.



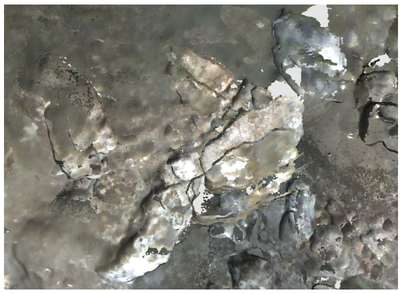
(a) Ours (full view)



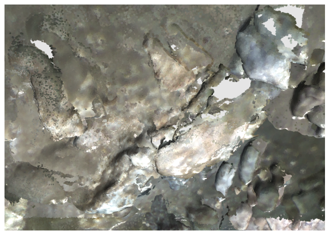
(b) ElasticFusion RGB-D tracking (full view)



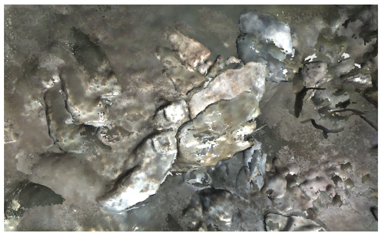
(c) ElasticFusion Depth tracking (zoom in)



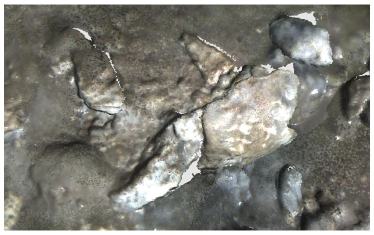
(d) ORB-SLAM2 tracking (zoom in)



(e) ElasticFusion full (zoom in)

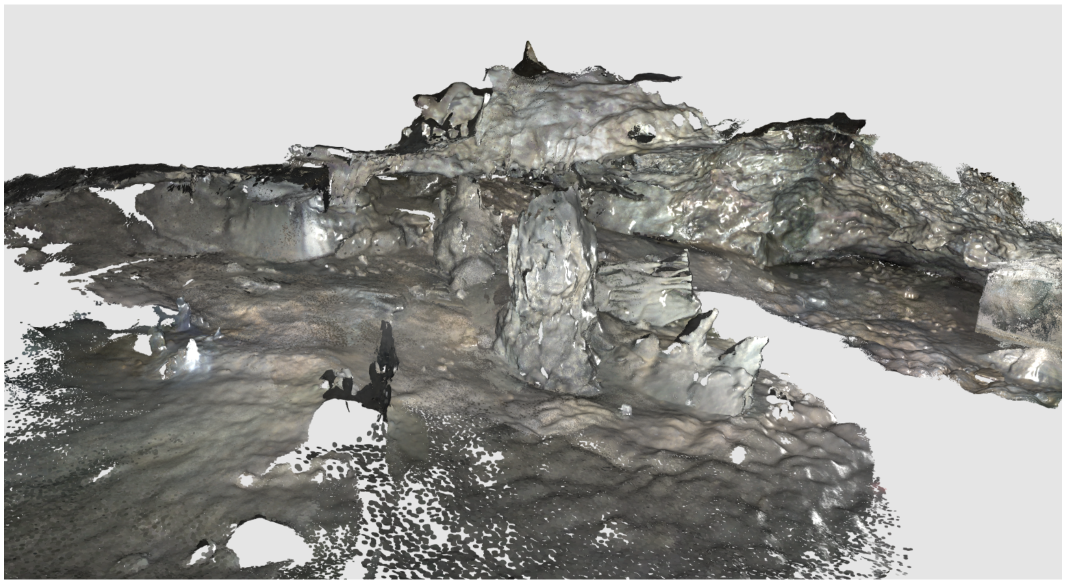


(f) ORB-SLAM2 full (zoom in)

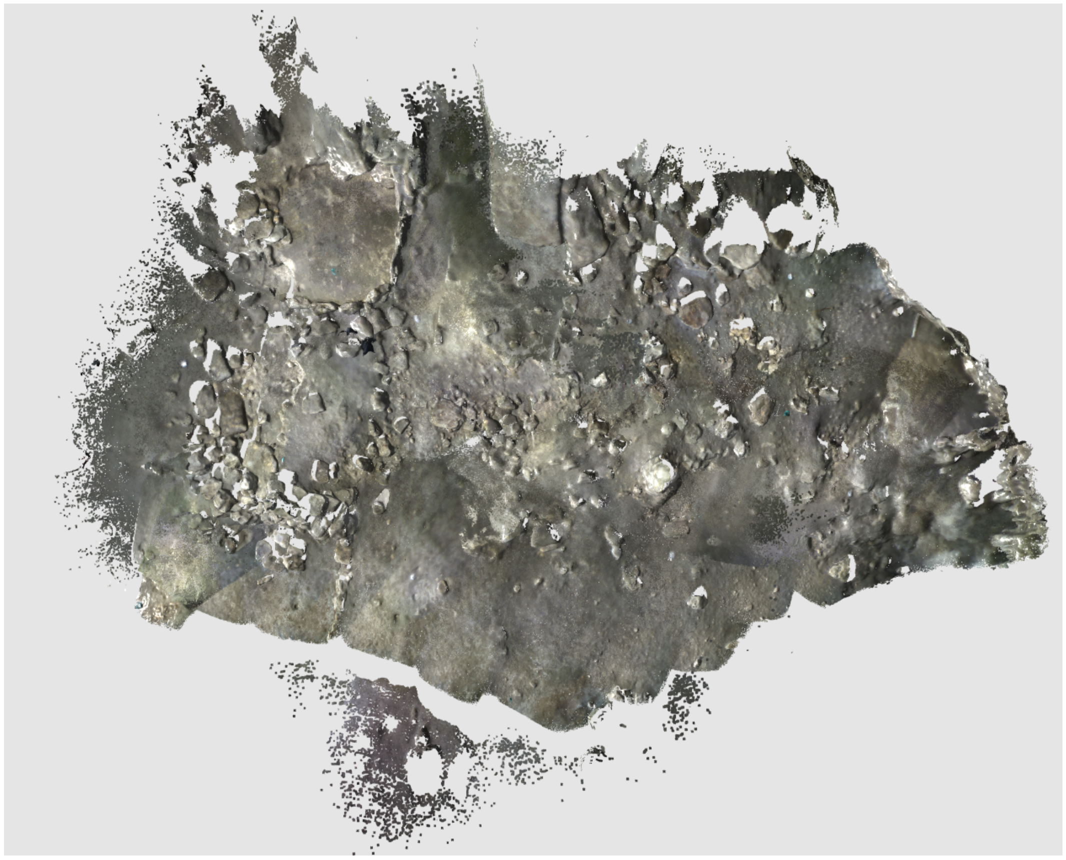


(g) Ours (zoom in)

Figure 4.3: Results on the chamber-entrance data sequence. A full view of the site is shown in (a). RGB-D tracking in ElasticFusion failed to provide meaningful results as in (b). For other methods, zoom in views of green region marked in (a) are compared. Visually, the proposed system (g) provides similarly good results to depth tracking (c), while full ElasticFusion (e), ORB-SLAM2 tracking (c) and full system (f) all have mismatches in their results.



(a) Our result on the chamber-alcove data sequence



(b) Our result on the chamber-cave-floor data sequence

Figure 4.4: Our results on other two data sequences. All methods can provide meanful results with minor differences.

performance. In tracking of ElasticFusion, there is one important parameter that controls the weight of RGB in tracking. We use the default 10 for RGB-D tracking, and a number greater than 100 that can disable RGB completely so that the tracking is totally based on the depth. For loop closure, there are three different types. ElasticFusion is using Ferns-ICP based approach. ORB-SLAM2 utilized BoW. Our point cloud registration-based surface loop detection approach is the third type compared.

Observations on tracking. From Figs. 4.2 and 4.3, we can see that RGB-D tracking of ElasticFusion does not work reliably on cave data, especially compared with the case when it uses depth only. We think this is due to the moving light source. RGB-D tracking calculates a transformation matrix partially by minimizing the intensity difference of two aligned images, which assumes the lighting condition of scenes is static. The ElasticFusion depth tracking working quite well in most cases excepts on the chamber-floor-walking data sequence. We can see that, in Fig. 4.2 (b), a half of the floor data get rotated around 90 degree clock-wise. It is almost impossible for a dense direct tracking to recover from the error, due to that there are no strong correspondences. The tracking of ORB-SLAM2 performed very well in all the data sequences, and it provides the possibility of globally optimize the map. The robustness of feature-based tracking implemented by ORB-SLAM2 is the reason that we use it as our tracking module.

Observations on loop detection and optimization. When we compare the performance of loop closure, our surface-focused method performs the best. It connects important loops in all the four sequences. The difficult data is the chamber-floor-walking one, shown in Fig. 4.2. Neither ElasticFusion nor ORB-SLAM2 tracks the camera trajectory correctly. Even after their loop closure, mismatches are still significant. Our method shows its robustness by reconstructing consistent 3D models on all data.

4.5.2 Augmented ICL-NUIM dataset

We use the augmented ICL-NUIM dataset [18] to quantitatively analyze the performance of our system. This dataset is a synthetic dataset with ground-truth surface models and camera trajectories. It has two indoor scenes: a living room and an office, and four RGB-D sequences, two sequences for each scene.

Experiment design. Evaluation metrics are camera trajectory translation RMSE

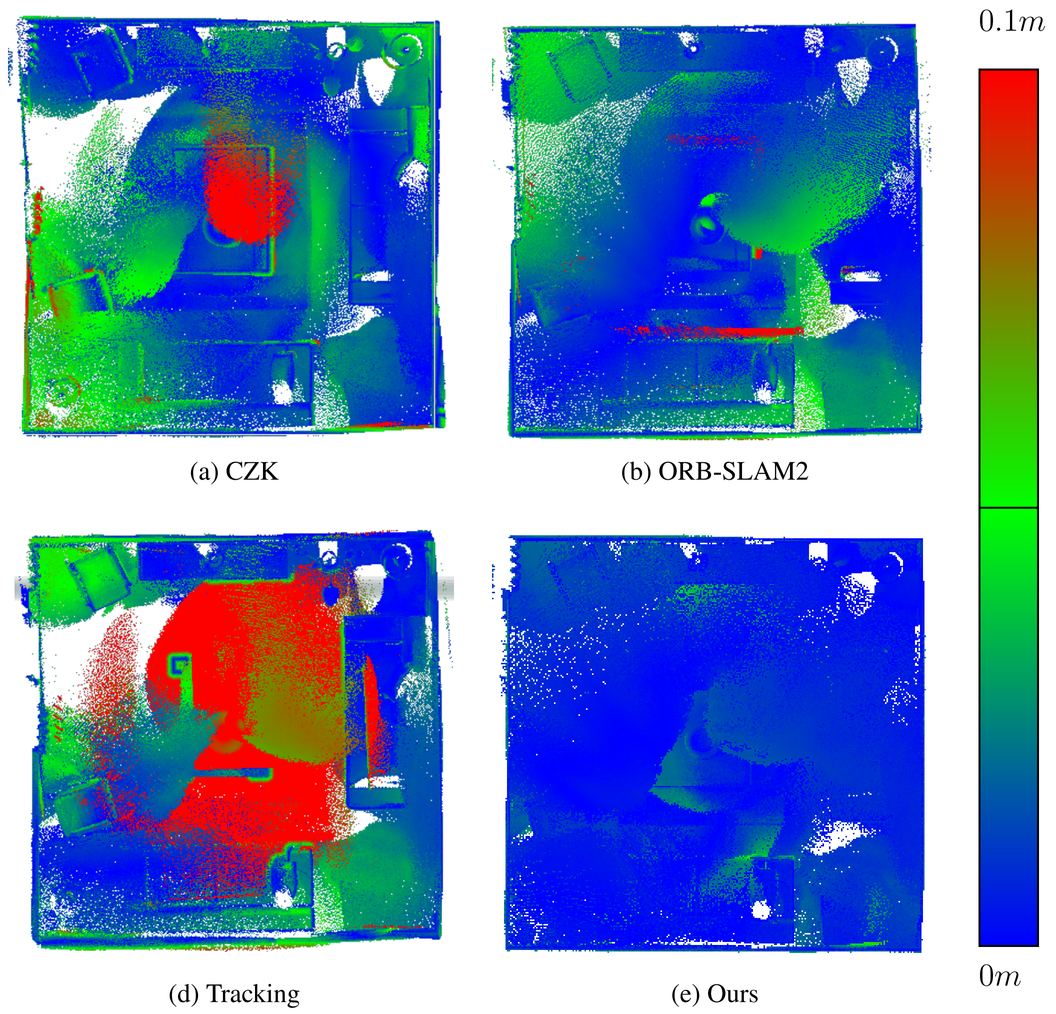


Figure 4.5: Distance error map of reconstructed models from different methods against ground-truth on Living room 1 data sequence. One can see our results have the lowest error across the whole model.

described by Handa *et al.* and the mean distance of the reconstructed surfaces to the ground-truth surfaces in the same way as Whelan *et al.* We report them separately in Table 4.2 and Table 4.3. Since different systems use different ways to fuse 3D models, for a fair comparison, we fuse 3D models using ElasticFusion using the same parameters with a camera trajectory estimate from each system. We use truncating depth distance of 4 meter and 10 as the surfel confidence threshold for fusion.

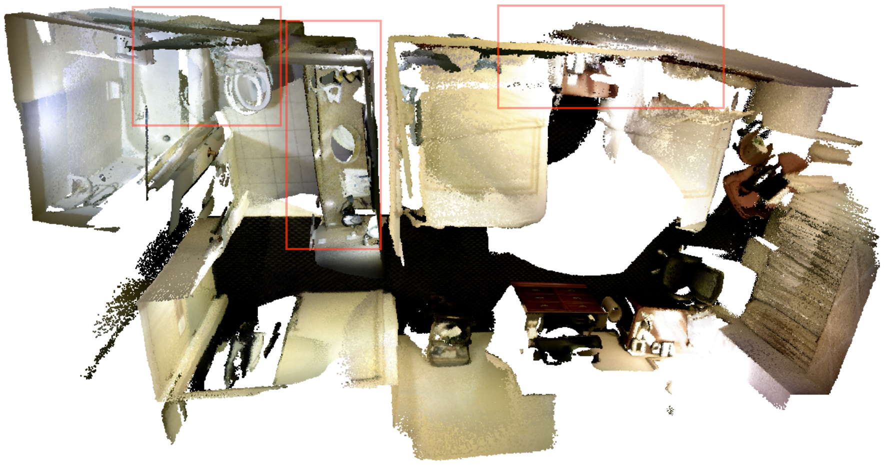
Table 4.2: Mean surface reconstruction error (in meters) on augmented ICL-NUIM sequences

	Livingroom 1	Livingroom 2	Office 1	Office 2	Average
CZK	0.033	0.028	0.019	0.022	0.026
Tracking	0.031	0.022	0.019	0.014	0.022
ORB-SLAM2	0.017	0.010	0.015	0.013	0.014
Ours	**0.007**	**0.007**	**0.013**	**0.010**	**0.009**

Table 4.3: RMSE (in meters) of estimated camera trajectories

	Livingroom 1	Livingroom 2	Office 1	Office 2	Average
CZK	0.10	0.13	0.06	0.07	0.09
Tracking	0.14	0.05	0.05	0.03	0.07
ORB-SLAM2	0.10	0.03	0.04	0.03	0.05
Ours	**0.03**	**0.02**	**0.03**	**0.02**	**0.03**

Observations. From Table 4.2 and Table 4.3, our system can give best results on all data sequence in terms of both trajectory and surface estimation accuracy. To give a more informative comparison, we report an error map of the reconstructed model in Fig. 4.5 on Livingroom 1 data sequence. We can see our results have the lowest error across the whole model. Our method performs better because more loops get detected for improving the final loop optimization results. In ORB-SLAM2, the loop detector has trouble detecting some important loops because its consistency check can hardly be satisfied due to very local view overlappings. On the other hand, our surface loop detector does not have this problem. Compared to CZK, the performance gain comes from the advantage of sparse-feature based bundle adjustment optimization, which can produce MAP results, thus more accurate camera trajectories and better reconstructed 3D models.


(a) Tracking only

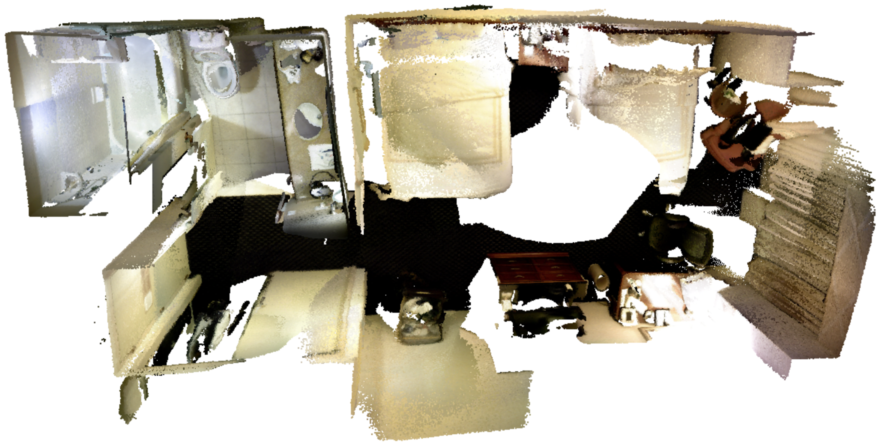
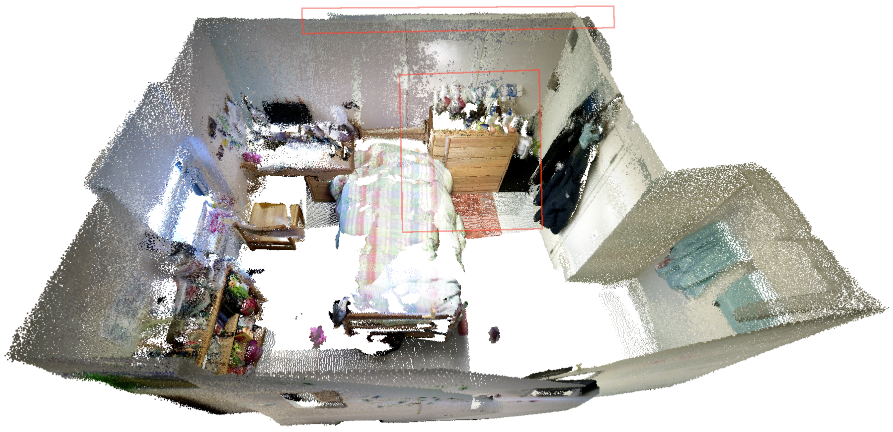
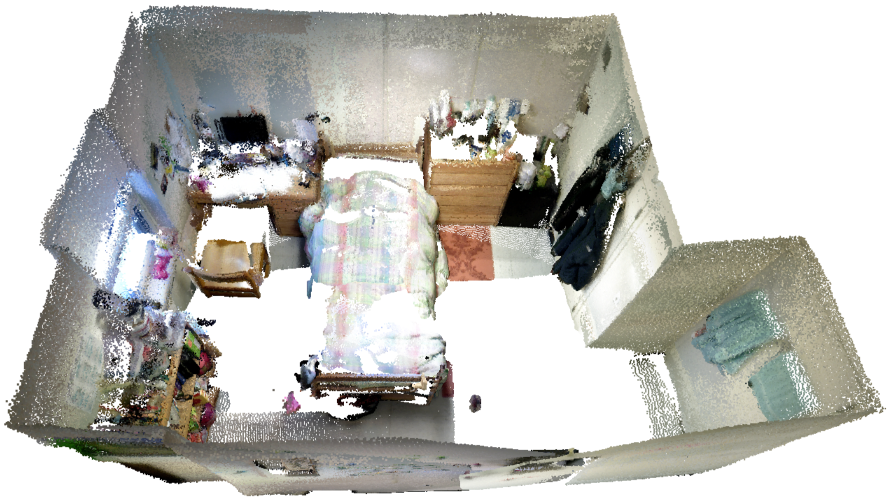

(b) With loop optimization

Figure 4.6: Results on sequence `maryland_hotel3` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.

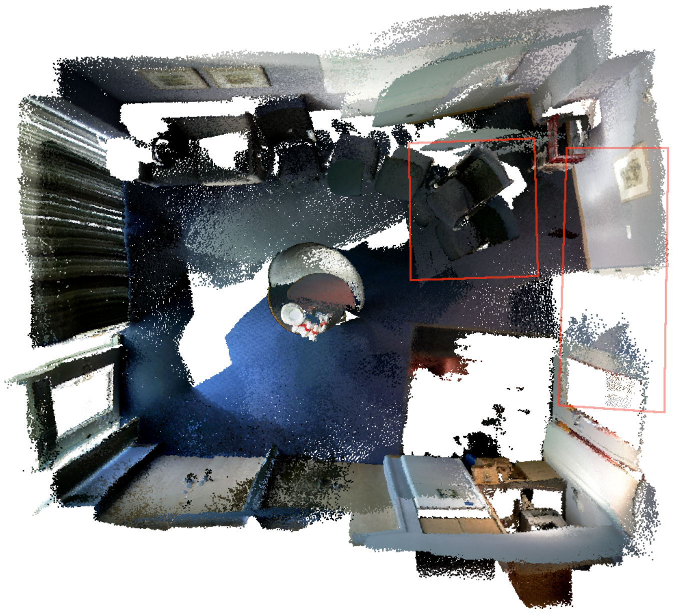


(a) Tracking only



(b) With loop optimization

Figure 4.7: Results on sequence `mit_dorm_next` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.


(a) Tracking only

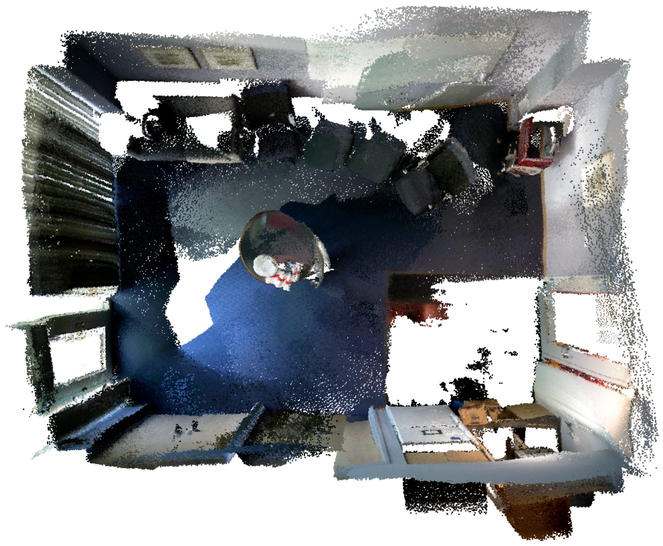
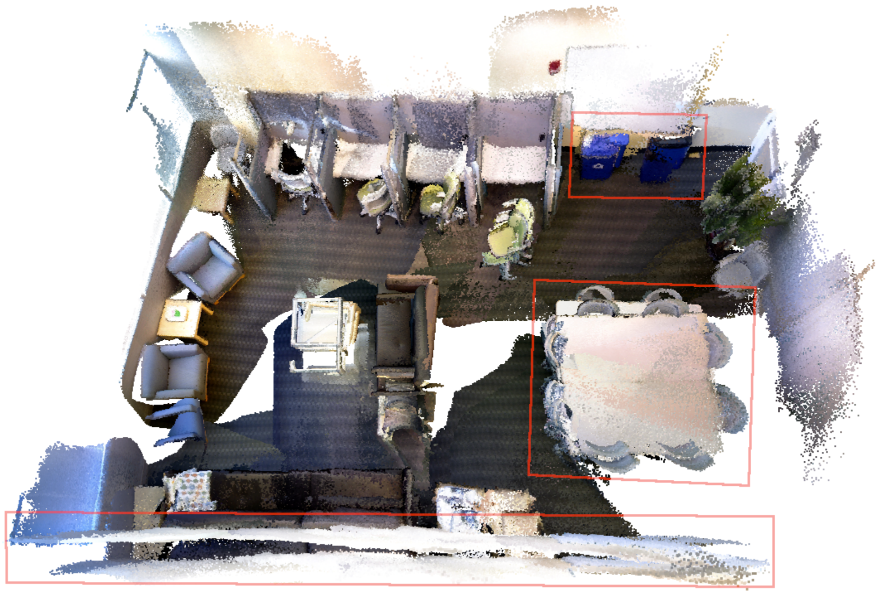
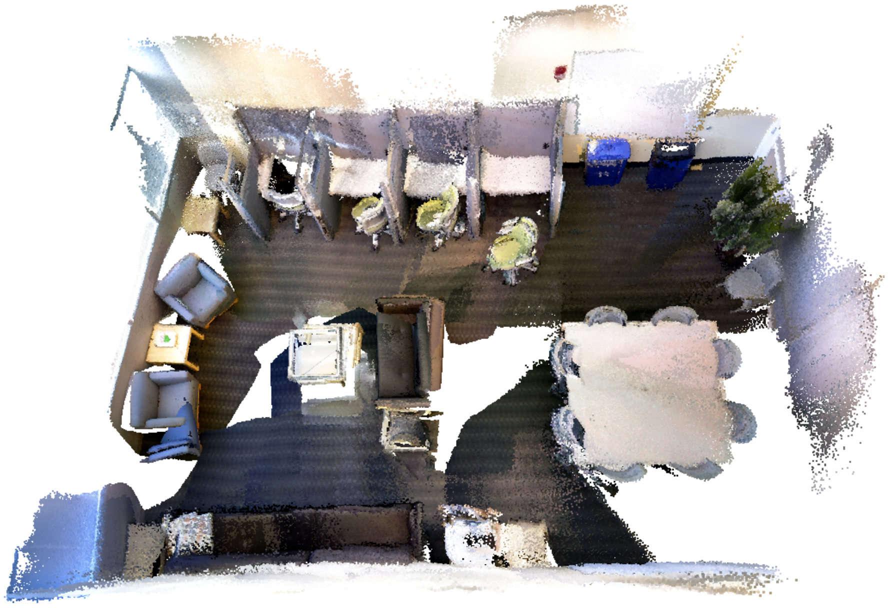

(b) With loop optimization

Figure 4.8: Results on sequence `mit_lab_hj` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.

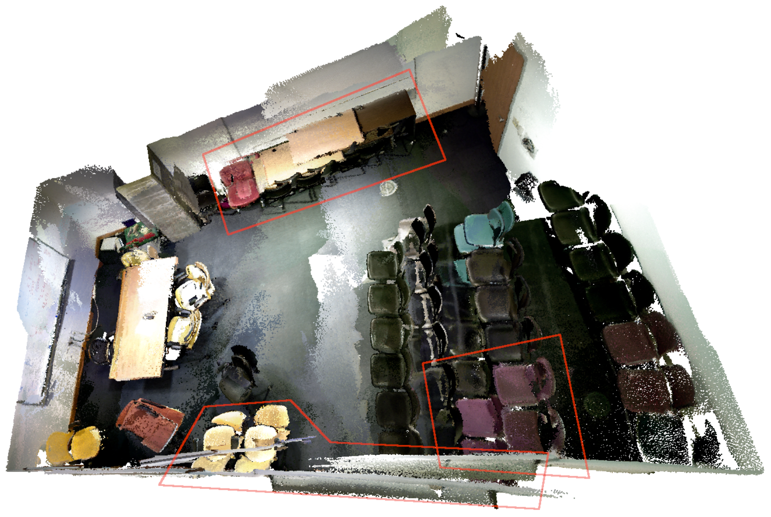


(a) Tracking only

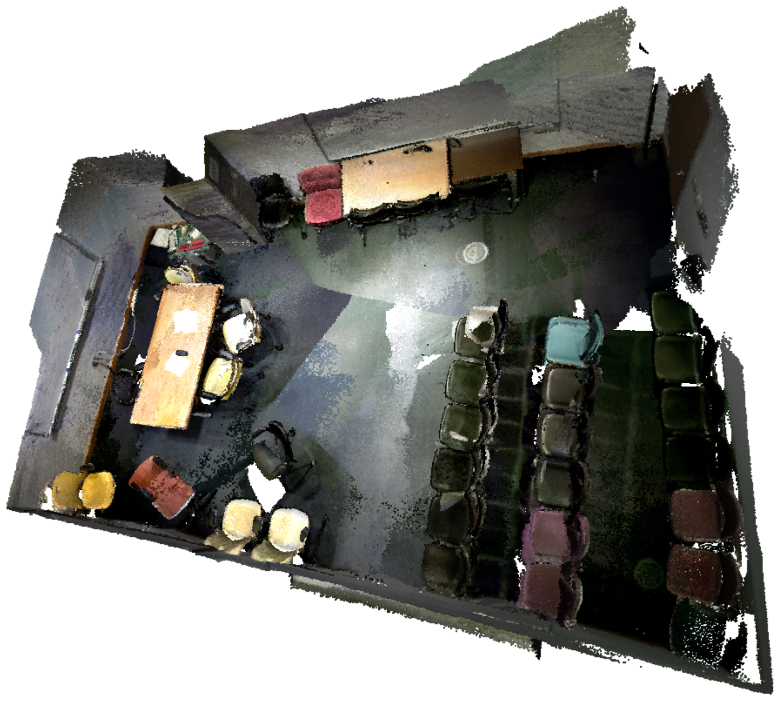


(b) With loop optimization

Figure 4.9: Results on sequence `76_studyroom` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.



(a) Tracking only



(b) With loop optimization

Figure 4.10: Results on sequence `mit_32_d507` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.

4.5.3 SUN3D dataset

The SUN3D dataset [119] is a large-scale RGB-D database that captures many places. It contains many data sequences. There are eight sequences `http://sun3d.cs.princeton.edu/listNow.html` that are labeled with object annotations and widely used for evaluating SLAM and 3D reconstruction systems. Since there is no ground truth available, we follow this practice and run experiments on these sequences. We show qualitative results in the form of screenshots of reconstructed 3D models in Figs. 4.6, 4.7, 4.8, 4.9 and 4.10.

For the sequences in Figs. 4.6, 4.7, 4.8, 4.9 and 4.10, we highlight the mismatches in tracking results so that readers can better compare them with our results. For sequences `harvard_c5`, `harvard_c6` and `harvard_c8`, there is no loops detected on top of tracking. So we do not include screenshots for them. **Observations:** SUN3D data sequences are scanned with very loopy motion in some area but only once for some other scene parts, thus is considered more difficult. Even though our method does not produce perfect reconstructed 3D models, it dramatically removed some significant mismatches. Also, our results are on par or better than other methods. Interested readers may compare with the results of CZK on this webpage `http://redwood-data.org/indoor/models.html`.

4.5.4 TUM RGB-D dataset

The TUM dataset [104] is an RGB-D dataset that is commonly used to evaluate SLAM systems. This dataset has 39 RGB-D sequences recorded in office and industrial environments with a large variety of camera motions and scenes. Along with RGB-D sequences, ground truth camera trajectories that are recorded with a motion capture system are also available. Following common practice, we run experiments only on sequences that are commonly used for SLAM evaluation [80].

Observations. From the results listed in Table 4.4, it shows that for data sequence `fr1/desk`, `fr1/room`, `fr2/desk`, and `fr3/office` our method makes improvement on tracking and achieves competitive results comparing full ORB-SLAM2. Among these sequences, `fr1/room` has high ATE error in tracking, and our surface loop significantly

Table 4.4: TUM RGB-D dataset comparison absolute trajectory error (ATE) (m).

Sequence name	ORB-SLAM2	Tracking	Ours
fr1/desk	0.016	0.021	0.019
fr1/desk2	0.022	0.028	0.028
fr1/room	0.047	0.295	0.068
fr2/desk	0.009	0.016	0.015
fr2/xyz	0.004	0.011	0.011
fr3/office	0.010	0.025	0.013
fr3/nst	0.019	0.034	0.034

reduced the error. For the small performance difference, we think that it is due to the implementation difference of the tracking part and that the difference only has only marginal effects on reconstructed 3D models when ATE error is less than 0.02 as shown in Table 4.5 and Fig. 4.3. For sequences `fr1/desk2` and `fr2/xyz`, the tracking has provided good enough results that most of the frames are connected in the co-visibility graph such that there are no loops to be detected. The sequence `fr3/nst` is a scan of a flat wall with rich texture but no geometry changes, so our method cannot detect surface loops in it; thus, no improvement is made. The TUM dataset indicates that, even though our method is designed for larger-scale environments with rich geometry changes, it can produce competitive results for some small-scale environments.

4.6 Chapter summary

This chapter presents a novel 3D reconstruction system that maps both large archaeological caves and general indoor environments with RGB-D cameras. The proposed system produces accurate 3D models by detecting and optimizing surface loops in sparse feature-based visual SLAM systems. By adding surface loop closure into a vSLAM system, globally consistent and optimal 3D models are generated accurately. The proposed system consists of five components: 1) sparse feature-based camera tracking from ORB-SLAM2, 2) surface model fusion powered by Surfels, 3) a novel fast surface loop detection algorithm, 4) a novel surface loop filtering method, and 5) loop optimization based on sparse feature-based bundle adjustment.

In the surface loop filtering component part, a novel objective function is proposed to remove false-positive loops from entering the loop optimization step. The proposed objective function formulates a least-squares pose-graph with a bundle adjustment term as the supporting backbone graph and robust least squares terms with switchable constraints for surface loop detections. Due to the inherent difference in underlying optimization pose-graphs, compared with its closely related work CZK, our method is less strict to use in practical scenarios, especially on a fast-moving UAV platform. Because our method does not require maintaining RGB-D cameras facing surfaces all the time, but CZK requires it. In addition to the flexibility, the proposed method is benchmarked against CZK on the Augmented ICL-NUIM dataset in terms of filtered loop detection precision and recall. Experiments show that the proposed method performs better (with $+37.5\%$ precision and equally better recall) than CZK when only a limited number of loops are detected, and provides competitive performance on all other scenarios.

Moreover, experiments are conducted to evaluate the performance of the full novel system on multiple datasets, including our Maya cave dataset, augmented ICL-NUIM, TUM RGB-D dataset, SUN3D dataset, and some other public data sequences. The results are evaluated with: absolute trajectory error or trajectory RMSE when ground-truth camera trajectories are available, surface reconstruction error when ground-truth 3D models are accessible, and visual comparisons when no ground-truth is available. The results show that sparse feature-based camera tracking performs the best in cave environments. The results also show that the proposed system produces the most reliable and accurate 3D reconstruction performance when surface loops are detected, filtered, and optimized on a sparse feature-based objective function. Other than in cave environments, experiments on other datasets show that the proposed system produces results on-par or better than baseline methods.

Chapter 5

Dense Map Posterior: A Novel Quality Metric for 3D Reconstruction

5.1 Introduction

Background and motivation. Simultaneous localization and mapping (SLAM) [14, 80] and 3D reconstruction [124, 18, 41] methods have achieved dramatic advancements recently due to the increasing need for autonomous vehicles and consumer robots. But, it is still not an easy task to collect data for performance evaluation. Existing metrics require additional ground truth data, such as ground truth camera trajectories or ground truth 3D models, that need to be collected by special and expensive instruments, such as motion capture systems, RTK-GPS, and LiDAR. This hardware requirement makes it impossible to add ground truth to existing data. As a result, there are only a limited number of datasets with ground truth information available [35].

To address this problem, some researchers turn to generate synthetic data from graphical rendering on virtual scenes [78, 103]. This makes accessing ground truth data easy, but it introduces new challenges to create realistic virtual environments and camera trajectories. Some datasets [10, 28, 56] choose to use pseudo-ground truth data which are generated by estimation algorithms, while some others resort to external markers [62, 97].

Key contributions. In this chapter, we propose a metric, dense map posterior (DMP), for 3D reconstruction and mapping performance evaluation that can work without any ground truth data. Instead, it calculates a comparable value, reflecting a map posterior probability, from dense point cloud observations. In our experiments, the proposed metric is benchmarked against ground truth-based metrics. Results show that the proposed DMP can provide a similar evaluation capability.

Significance. The proposed metric makes a broad impact to SLAM and 3D mapping. We list a few major ones here:

1. It makes 3D mapping evaluation more flexible and easily accessible.
2. It provides a supervisory figure-of-merit signal for robust loop closure optimization.
3. It helps build large real-world SLAM datasets with minimal effort.
4. It makes it possible to introduce self-supervised machine learning algorithms to 3D reconstruction methods.

5.2 Related work

When evaluating 3D mapping results, the common error metrics are the absolute trajectory error (ATE) [80, 104, 114], the relative position error (RPE) [104], and surface distance error [124, 18, 50, 114]. The ATE represents the difference between the ground truth and an estimated trajectory in a common world frame. The RPE calculates the relative relation differences between estimated and ground truth trajectories. The surface distance error specifies the mean and median of distances between a reconstructed surface and a ground-truth surface. All these metrics require ground truth data that is not trivial to collect.

The work closely related to ours is [90]. Olson and Kaess discussed using sparse map posteriors for map evaluation, but they concluded that the posterior subjects to over-fitting during the optimization process. Thus it is not reliable. In their formulation, the sparse map posterior is calculated on a map of sparse landmarks. We agree that, with sparse data only, it is not as reliable as ground truth-based metrics. However, in our method, we consider the posterior of a dense 3D map given geometry-related observations. The over-fitting is no longer a problem.

5.3 Dense map posterior (DMP)

3D reconstruction and mapping methods usually take a set of sensor reading and output both an estimated 3D model (*e.g.* mesh or surfels, *etc.*) $\mathcal{M}$ and an estimated camera or LiDAR trajectory $\mathbb{T}$. For simplicity, we denote $M = \{\mathcal{M}, \mathbb{T}\}$ to include both outputs. For the input sensor readings, we denote geometry related readings as $\mathbb{Z}$, where $\mathbb{Z}$ can be a set of depth images from RGB-D cameras or a set of point clouds from LiDARs.

The idea behinds the proposed metric, dense map posterior (DMP), is to calculate a value reflecting a posterior probability $p(M|\mathbb{Z})$. For this calculation, DMP takes both M and $\mathbb{Z}$ as evaluation input. Then, performance can be evaluated by comparing metric values of different 3D reconstruction estimates $\{M_i\}$.

Because there is no direct way to calculate $p(M|\mathbb{Z})$, derivations are made to get a computable form. First, the Bayes' rule is applied to get

$$p(M|\mathbb{Z}) = \frac{p(\mathbb{Z}|M)p(M)}{p(\mathbb{Z})}. \tag{5.1}$$

Then, for the same set of observations $\mathbb{Z}$, if there are two different 3D reconstruction estimates M_1 and M_2, we can get a ratio

$$\frac{p(M_1|\mathbb{Z})}{p(M_2|\mathbb{Z})} = \frac{\frac{p(\mathbb{Z}|M_1)p(M_1)}{p(\mathbb{Z})}}{\frac{p(\mathbb{Z}|M_2)p(M_2)}{p(\mathbb{Z})}}. \tag{5.2}$$

If we assume $p(M_1) = p(M_2)$, we can simplify (5.2) to be

$$\frac{p(M_1|\mathbb{Z})}{p(M_2|\mathbb{Z})} = \frac{p(\mathbb{Z}|M_1)}{p(\mathbb{Z}|M_2)}, \tag{5.3}$$

where

$$p(\mathbb{Z}|M) = \prod_{Z \in \mathbb{Z}} p(Z|M), \tag{5.4}$$

where Z is a depth image or a single point cloud scan of LiDAR. Note that Z consists of many independent observations (*i.e.*, depth image pixels or 3D points). We denote each of them as $z \in Z$, then

$$p(Z|M) = \prod_{z \in Z} p(z|M). \tag{5.5}$$

Assume that z follows a Gaussian distribution:

$$p(z|M) \sim \mathcal{N}(z', \sigma^2) \tag{5.6}$$

where z' and σ are the mean and covariance of the Gaussian distribution. z' is produced by applying sensor model $\mathcal{S}(\cdot)$ on M with the same observing configuration as z:

$$z' = \mathcal{S}_z(M) = \mathcal{S}_z(\mathcal{M}, T_z), \tag{5.7}$$

where $\mathcal{S}(\cdot)$ is a camera projection model for depth sensors and a range-bearing model for LiDARs. T_z is the estimated sensor observing perspective for z. One can easily get T_z from the estimated trajectory $\mathbb{T}$. For numerical stability, we take log on both sides of (5.3). We get a log-likelihood ratio (LLR)

$$LLR = \log p(\mathbb{Z}|M_1) - \log p(\mathbb{Z}|M_2). \tag{5.8}$$

We define

$$r(M, \mathbb{Z}) = -\log p(\mathbb{Z}|M). \tag{5.9}$$

When $LLR > 0$, which equivalents to $r(M_1, \mathbb{Z}) < r(M_2, \mathbb{Z})$, M_1 is more likely to be better than M_2 in terms of dense map posterior probability. This means that we can compare different $r(M_i, \mathbb{Z})$ directly. The lower, the better. To further simplify (5.9), one can plug (5.4), (5.5), and (5.6) into (5.9) to get

$$\begin{aligned} -\log p(\mathbb{Z}|M) &= -\log \left[\prod_{Z \in \mathbb{Z}} \prod_{z \in Z} \frac{1}{\sigma\sqrt{2\pi}} e^{-(z-z')^2/2\sigma^2} \right] \\ &= \sum_{Z \in \mathbb{Z}} \sum_{z \in Z} \frac{(z - z')^2}{2\sigma^2} + L_0, \end{aligned} \tag{5.10}$$

where L_0 is a constant. Since L_0 is the same for the same set of $\mathbb{Z}$, then (5.9) can be simplified to be

$$r(M, \mathbb{Z}) \sim \overline{r}(M, \mathbb{Z}) = \sum_{Z \in \mathbb{Z}} \sum_{z \in Z} (z - z')^2, \tag{5.11}$$

where $\overline{r}(M, \mathbb{Z})$ is the proposed metric DMP, the lower, the better. To calculate its value, one needs to provide a 3D reconstruction estimate M, sensor readings $\mathbb{Z}$, and a sensor model $\mathcal{S}(\cdot)$.

5.4 Evaluation of DMP – the proposed metric

To evaluate the proposed metric DMP, we conduct experiments to compare the results of different metrics on multiple datasets: TUM RGB-D dataset [104] and Augmented ICL-NUIM [18]. In all of our experiments, 3D models are fused using Surfels implemented by ElasticFusion [114].

5.4.1 TUM RGB-D dataset

The TUM RGB-D dataset [104] is widely used for evaluating SLAM systems. The dataset has RGB-D sequences with ground truth camera trajectories available. We run experiments on a subset of sequences that are commonly used for SLAM evaluation [80]. Following [104], we adapt the absolute trajectory error (ATE) as the baseline metric.

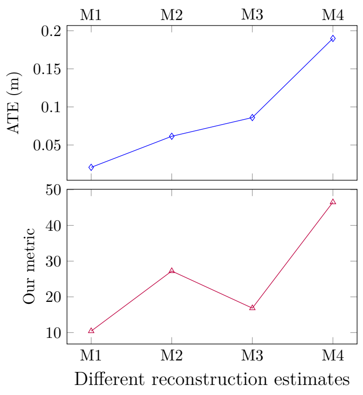
For each data sequence, four 3D reconstruction estimates, M_1, M_2, M_3, and M_4 are used as benchmarking data. These estimates are from SLAM systems with different characteristics. The results are reported in Fig. 5.1. For better visualization, M_1 to M_4 are sorted with the ATE metric.

Key observation. In the results, the proposed metric DMP can report the same ascending order as the ATE metric in most cases. This means our metric can provide similar evaluation results as the ATE.

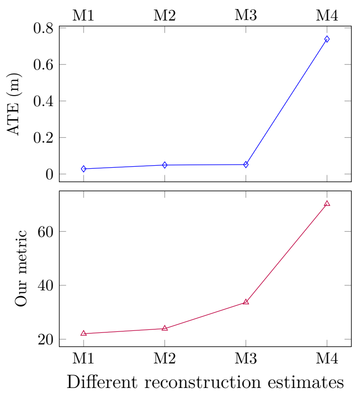
5.4.2 Augmented ICL-NUIM dataset

The augmented ICL-NUIM (AIN) dataset [18] is a synthetic dataset with ground-truth surface models and camera trajectories. Experiments are performed to quantitatively compare performance of the proposed metric DMP with ground truth-based metrics: trajectory RMSE and surface mean distance (SMD).

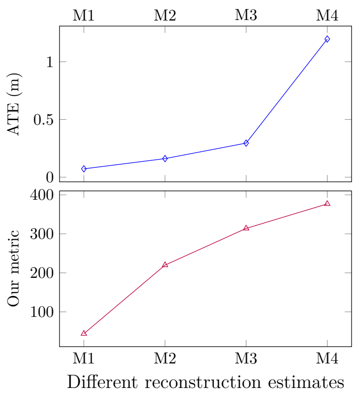
Results are reported in Fig. 5.2. Reconstruction estimates data $M_1 - M_5$ for evaluation are from SLAM methods and ranked with the SMD metric. **Key observations:** Our metric DMP positively correlates to SMD and Traj. RMSE, which means our metric can provide almost the same ranking results as ground truth-based metrics. To further visualize our metric, we include screen-shots of different 3D models and corresponding DMP values in Fig. 5.3.



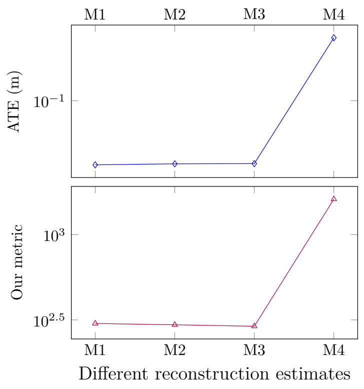
(a) fr1/desk



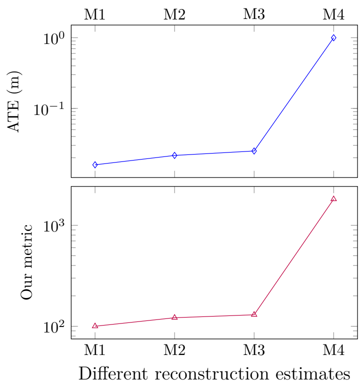
(b) fr1/desk2



(c) fr1/room

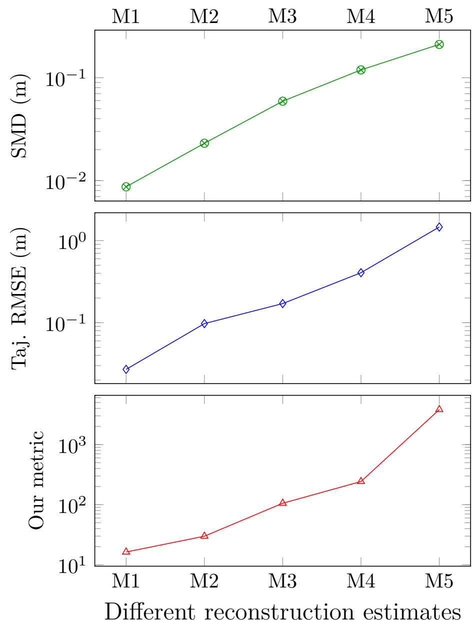


(d) fr2/desk

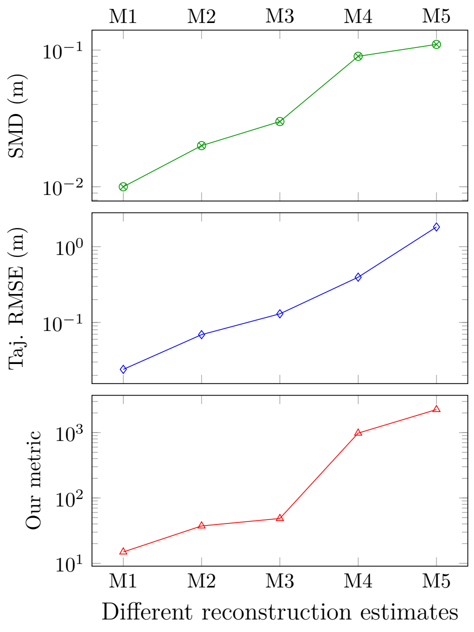


(e) fr3/office

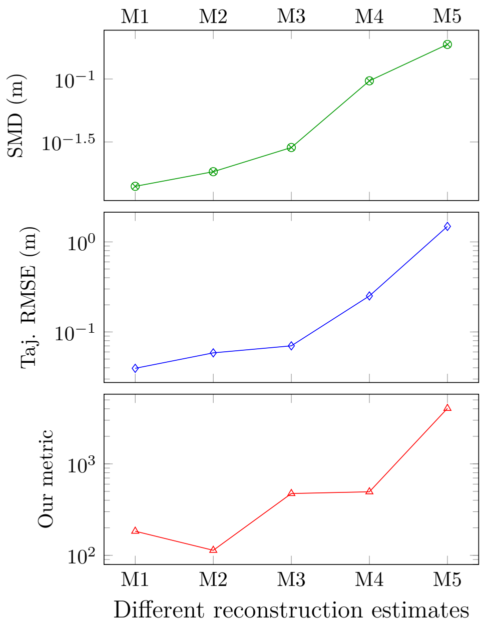
Figure 5.1: Comparison of ATE and our metric DMP. For each sequence, curves of two metrics follow almost the same ascending order, which means metrics have very similar comparison results. The outlier case in (a) is discussed in section 5.4.3.



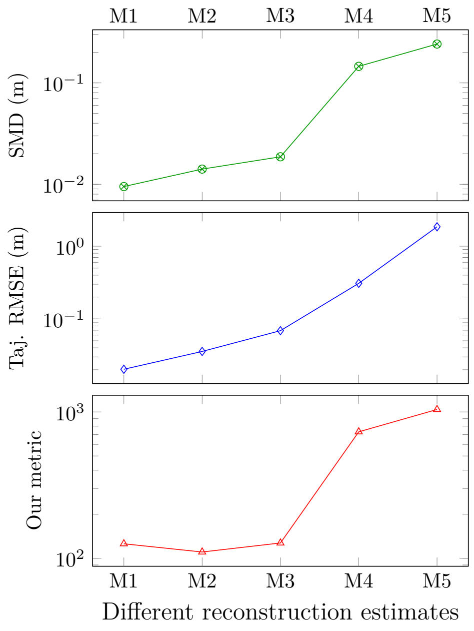
(a) livingroom 1



(b) livingroom 2

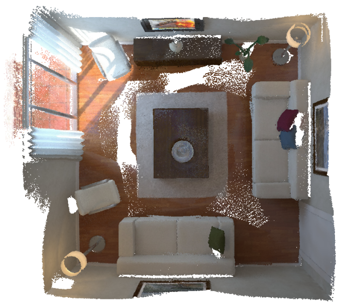


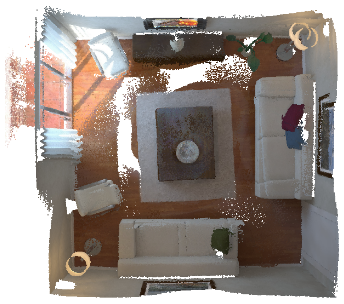
(c) office 1

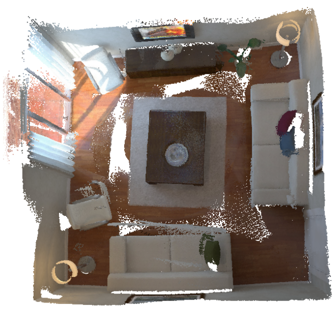


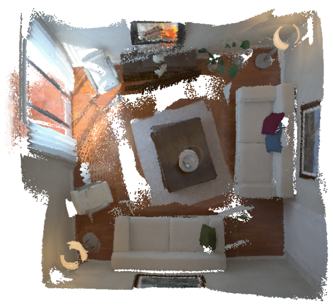
(d) office 2

Figure 5.2: Comparison of surface mean distance (SMD), trajectory RMSE (Traj. RMSE), and the proposed metric DMP. Our metric DMP positively correlates to SMD and Traj. RMSE, which means our metric can provide almost the same ranking results as ground truth-based metrics. The two outliers can be avoided by overlapping scans.


(a) M1, DMP: 14.9


(b) M2, DMP: 37.3


(c) M3, DMP: 48.5


(d) M4, DMP: 982.4

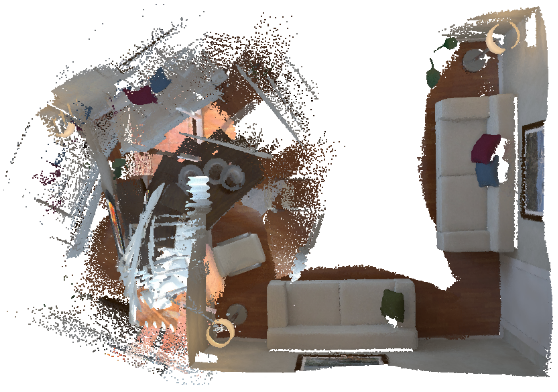

(e) M5, DMP: 2254.0

Figure 5.3: Visualization of estimated 3D models of the livingroom 2 sequence and their corresponding DMP values. In (a), M_1 leads to the best 3D mapping quality among the five estimates. There are not any noticeable mismatches. In (b), there are mismatches at the upper right-hand corner. The lamp and sofa are not well reconstructed. In (c), the sofa at the buttom of the image is not perfectly matched between frames. In (d), the coffee table at the center is off. (e) shows a collapsed 3D model optimized with a false positive loop.

5.4.3 Ineffective cases

Among all the evaluations on the two datasets, there are rare cases where the proposed metric DMP gives different results from ground truth-based metrics: M_3 in `TUM/fr1/desk`, M_2 in `AIN/office 1`, and M_2 in `AIN/office 2`. We believe that it is because these data sequences have most of the space scanned only once without loopy coverage. That is the case where our metric cannot handle perfectly. However, it is simple to avoid this problem by adding overlapping scans.

5.5 Computational performance

Our metric can be efficiently parallelized using OpenGL and CUDA. In our implementation, on an NVIDIA Titan X Pascal, the average evaluation time is 2.7 s for a TUM model and 4.2 s for an Augmented ICL-NUIM model. The speed can be further improved if only a sampled subset of data frames are used for evaluation.

5.6 Chapter summary

In this chapter, we propose a metric, which can work without any ground truth data, for evaluating 3D reconstruction and mapping performance. In our experiments, the proposed metric DMP is benchmarked against ground truth based metrics. Results show that DMP can provide a similar evaluation capability. The proposed metric not only makes 3D mapping evaluation simpler, but also opens many new opportunities. We envision that more can be done with this metric, such as self-supervised methods and more available datasets.

Chapter 6

Offline Sifting and Majorization of Loop Detections

6.1 Introduction

Due to rapid development in autonomous vehicles and consumer robots, there is an increasing need for precise 3D maps for route and action planning and navigation. Among 3D mapping methods, visual simultaneous localization and mapping (vSLAM) and 3D reconstruction methods are very promising because they can map large-scale environments such as cities and indoor environments without the need for much human effort involved.

vSLAM and 3D reconstruction methods have gone through impressive progress. In camera tracking, there are different methods, such as sparse keypoint point-based methods [80, 32, 64, 29, 72], direct methods [131, 33], and dense surface-based methods [83, 114]. Additionally, IMU are added to methods [23, 110, 87, 130, 31, 73, 23] to make tracking more accurate. Even though camera tracking algorithms have good performance and low drift, the build-up error can still not be ignored [14]. To solve this problem, loop closure detection [40, 26] and optimization [66] are often leveraged to counter the problem, and it has provided plenty of improvements. However, the problem is not fully solved yet. Intuitively, the more loops in data, the more information to recover more precise camera trajectories and 3D models. But, in practice, when running existing vSLAM systems on datasets with loopy motions, mismatches can always be found in the final results. This means that loops are not successfully detected and utilized.

vSLAM systems tend to add the loops very conservatively to reduce the severe influence of the false loops [14]. These conservative checks are the result of the non-perfect precision performance of loop detection methods. There are high chances that detected loops are incorrect ones.

To solve this challenging problem, we propose an algorithm that can sift and majorize loop detections so that only correct and essential loops are fed into the following optimization steps. The proposed method highly couples with the dense map posterior (DMP) metric [123] that can evaluate 3D reconstruction performance without ground truth measurement. Our proposed algorithm can compare the usefulness and effectiveness of different loops and ultimately sifts out false and unimportant loops. To the best of our knowledge, the contributions of the proposed algorithm are:

1. The proposed algorithm can sift loop detections based on their impact on loop optimization results.

2. It is the first algorithm that can marjorize loop detection only to keep the important ones while ignoring the less relevant ones.

3. Experiments on public datasets show it outperforms state-of-the-art methods.

6.2 Related work

To avoid the severe consequence of optimizing with false loops, vSLAM and 3D mapping systems tend to add the loops very conservatively. ORB-SLAM2 [80] requires the presence of several consistent loops in consecutive keyframes to accept them, where at least one keyframe must be shared in order to be classified as consistent. With this consistency check, ORB-SLAM2 merely takes false loops into optimization but at the price that plenty of correct loops are rejected. ElasticFusion [114] evaluates several characteristics before taking a loop detection into optimization pipelines, including deformation cost and final state of the Gauss-Newton system. Even after all the evaluations, a good loop is often rejected, and not rare to see that an incorrect loop is accepted. BundleFusion [29] filters loop correspondences with cascade checks including local geometric and photometric consistency checks and check on correspondence residual after optimization. The local depth discrepancy check shares a small similarity with our work. However, the check is limited to a very local region with a downsampled depth resolution together with a user-specified threshold. Thus it is not informed about the effect of loop data correspondences impacting a full 3D model. A requirement on a user parameter also makes it ineffective and less adaptive. [32] do this by pruning edges after optimization based on the discrepancy between the individual transformation estimates before and after optimization. We share the idea of observing optimization consequences brought in with a loop, but their impact is measured on a sparse graph while ours is observed on a full 3D dense model.

Another approach to solving the problem is to treat false loops as outlier data and decrease their impact on the optimization [70, 106, 1, 105]. They work well in some cases, but the dependence on initial conditions and the ratio of outliers makes them prune to failures. Choi *et al*. further develop this idea into an algorithm that is highly coupled with the dense 3D reconstruction problem by specifying both pose graph construction and least squares information calculation [18]. This method is very effective when a desired camera scan pattern is followed but it requires keeping surface within camera range all the time thus limiting its flexibility. It also suffers dependence on initial condition and outlier ratio.

Due to the difficulty of balancing precision and recall of loop detections, SUN3D [119] turns to a human-in-the-loop approach by labeling objects in scenes and connect the same objects across frames. This method performs very well in terms of loop precision and recall, but it requires too much effort in labeling; thus is not practical to process data on a large scale.

6.3 The proposed method

To solve the loop sifting problem, we propose an algorithm specified in Algorithm 3. In the algorithm, a given set of loop detections is denoted as $\mathbb{O}$ among which each individual one is denoted as O. The supporting optimization pose graph is denoted as $\mathcal{G}$. The sensor (*e.g.* camera and LiDAR) data are denoted as $\mathbb{Z}$.

There are two parts in the algorithm. In the first part, all the loops are tested and evaluated individually on the given initial pose graph. This step first runs optimization with a single loop and then fuses a model with the optimized results $\mathbb{T}^i$. Then a DMP value r is evaluated for the fused 3D model M. This means that it tests each loop and sees how much improvement it provides by itself. Finally, all these loops are ranked by the calculated DMP value r in ascending order (more effective $\rightarrow$ less effective $\rightarrow$ negative impacts).

In the second part, all the loops are tested and evaluated one more time, but in a way that is different from the first time. In this part, the loops are tested in sorted order: the ones that provide more improvements are tested first. When a loop can provide performance improvement on the previous result, it will be added to an accepted set, thus will also impact consequent loop tests. In this way, loops are accepted when they can provide performance improvement on the current status. The first accepted one should make an improvement to the original results from tracking.

6.4 Implementation

The proposed method is general and agnostic to loop detection and optimization methods. Neither does it require a specific type of vSLAM system. For our experiments, we choose several well know implementations.

6.4.1 Tracking and optimization pose graph

The proposed method requires an optimization pose graph as the input data. The only requirement of the pose graph optimization is that it can handle loop closure optimization. In our implementation, we use sparse image feature-based tracking and mapping method implemented by ORB-SLAM2 [80] with loop detection disabled. The pose graph from ORB-SLAM2 is utilized as the optimization graph for the proposed method. For the purpose of loop sifting and majorization, we find pose graph optimization is good enough; thus, the more time-consuming full bundle adjustment is not included.

Algorithm 3: Loop sifting and majorization algorithm

```
Input  : $\mathbb{O}, \mathcal{G}, \mathbb{Z}$
Output : filtered loops $\mathbb{O}^*$
1  $\mathbb{O}^* \leftarrow \varnothing, r^* \leftarrow \texttt{r}(\mathcal{G}, \mathbb{Z}), \boldsymbol{r} \leftarrow \varnothing$
2  for $i \leftarrow 0$ to $\texttt{len}(\mathbb{O})$ do
3      $\mathbb{T}^i \leftarrow \texttt{optimize}(\mathcal{G}, \mathbb{O}[i])$
4      $M^i \leftarrow \texttt{fuseModel}(\mathbb{T}^i, \mathbb{Z})$
5      $\boldsymbol{r}[i] \leftarrow \texttt{r}(M^i, \mathbb{Z})$
6  end
   // Ranking $\mathbb{O}$ by our metric
7  $\mathbb{O}^r \leftarrow \texttt{rank}(\mathbb{O},\ by = \boldsymbol{r},\ order = \text{descending})$
   // Try $\mathbb{O}^r$ one by one and add the ones making improvements
8  for $i \leftarrow 0$ to $\texttt{len}(\mathbb{O}^r)$ do
9      $\mathbb{O}^*_{tmp} \leftarrow \texttt{union}(\mathbb{O}^*, \mathbb{O}[i])$
10     $\mathbb{T}' \leftarrow \texttt{optimize}(\mathcal{G}, \mathbb{O}^*_{tmp})$
11     $M' \leftarrow \texttt{fuseModel}(\mathbb{T}', \mathbb{Z})$
12     $r' \leftarrow \texttt{r}(M', \mathbb{Z})$
13     if $r' > r^*$ then
14         $\mathbb{O}^* \leftarrow \mathbb{O}^*_{tmp}$
15         $r^* \leftarrow r'$
16     end
17 end
```

The ORB-SLAM2 is a very well implemented sparse feature-based SLAM system. Inside this tracking module, the Oriented FAST and Rotated BRIEF (ORB) features are extracted for keypoint matching. Then frames are tracked against keyframes with motion estimate and then refined with a local sparse map. Keyframes are generated when tracking is weak, or the local bundle adjustment thread is free. Local BA is used to correct the re-projection error of feature correspondences among co-visible keyframes in a background thread. This tracking module provides camera poses for each frame and a co-visibility graph across keyframes.

6.4.2 Model fusion

It is a important step to fuse camera reading data into dense 3D models . For this step, surfels [92] are used as a data representation of 3D model. Each surfel has seven attributes: a position $\boldsymbol{p} \in \mathbb{R}^3$, normal $\boldsymbol{n} \in \mathbb{R}^3$, color $\boldsymbol{c} \in \mathbb{N}^3$, weight $w \in \mathbb{R}$, radius $r \in \mathbb{R}$, initialization timestamp t_0 and last updated timestamp t. With a radius property, a surfel can represent a local flat surface around a given a position $\boldsymbol{p}$.

Even though surfel fusion is fast with efficient implementation running on GPU, it takes a considerable amount of time. To speed up the efficiency, we leverage the advantage that surfels can easily be moved rigidly in space. We fuse k consecutive frames scene fragments as basic blocks and transform them based on optimized camera trajectories. In this way, the fusion of updated camera pose estimates is approximated with transforming scene fragments to updated location. Thus final results are calculated more efficiently.

6.4.3 Fragment loop to frame loop conversion

Since there are fewer scene fragments than frames, there is a need to convert scene fragment matches to camera frame loops. We do this by connecting a reference frame in one scene fragment and connecting it to all the frames of the other scene fragments and repeat for the other direction.

6.5 Experiments

Extensive experiments are performed to evaluate our proposed method on two datasets: augmented ICL-NUIM [18] and SUN3D dataset [119]. SMD stands for surface mean distance.

Table 6.1: Performance difference with only key loops vs. all correct loops agreed by ground truth. SMD is short for surface mean distance.

	Traj. RMSE		SMD		DMP	
	key loops	all loops	key loops	all loops	key loops	all loops
livingroom 1	**0.082**	0.175	**0.027**	0.059	**36.9**	122.8
livingroom 2	**0.037**	0.203	**0.012**	0.080	**19.5**	85.2
office 1	**0.051**	0.096	**0.020**	0.046	**143.5**	200.9
office 2	**0.036**	0.085	**0.014**	0.024	**110.5**	373.2
Average	**0.052**	0.140	**0.018**	0.052	**77.6**	195.5

6.5.1 Augmented ICL-NUIM dataset

We run experiments on the Augmented ICL-NUIM dataset [18]. This dataset is a synthetic dataset with ground-truth surface models and camera trajectories. The dataset has four data sequences of RGB-D data. For each sequence, there are merged scene fragments available with ground truth registration results. For this dataset, our baseline method is CZK [18] which is published in the same work as the Augmented ICL-NUIM dataset.

Table 6.2: Recall and precision performance on loops before and after loop filtering or sifting.

Recall/Precision (%)	Registration	CZK	Ours
livingroom 1	**61.2** / 27.2	57.6 / 95.1	5.5 / **100**
livingroom 2	**49.7** / 17.0	**49.7** / 97.4	3.9 / **100**
office 1	**64.4** / 19.2	63.3 / 98.3	2.8 / **100**
office 2	**61.5** / 14.9	60.7 / 100	0.7 / **100**

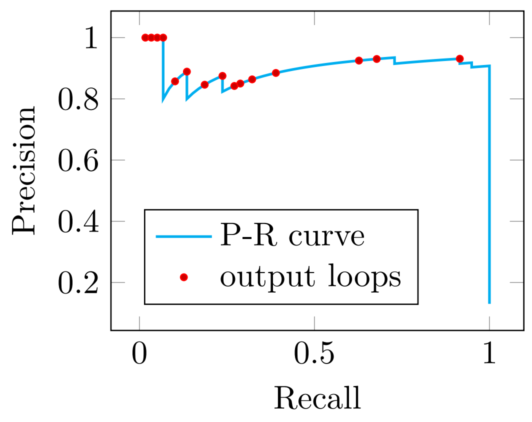
Experiments are conducted to evaluate the loop sifting and majorization performance of the proposed method. Performance is evaluated based on precision and recall of loops detected and remaining. Results are reported in Table 6.2. We can see that our method gets 100% percent precision, which is desired. You may notice that the recall reduced dramatically after sifting. The decrease is not because of the strict requirement but because many loops are not very useful. We note that the remaining loops are the core ones that matter most for a better reconstruction quality, which we call it loop majorization.

Many of the original loops are close to each other and connected accurately by ORB-SLAM tacking already. We prove this in another experiment that evaluates the trajectories and reconstructed 3D models of optimization with the key loops identified by our method and all the loops that agree with ground truth. Results are shown in Table 6.1. We can see that more loops do not improve performance instead decrease the performance. This is because some of the loops are not very precise. It will decrease accuracy if two loop regions are well connected originally.

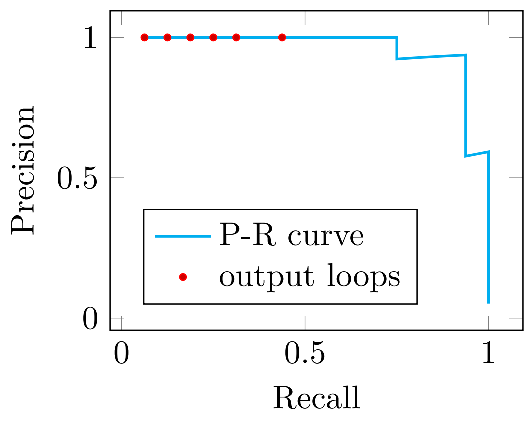
To further understand the proposed method, we draw precision-recall curves of the loop ranking results in the proposed algorithm in Fig. 6.1. In the results, we can see the curves all starts from 100% precision. Then the curves keep on high precision values when recall increases. There are a few drop points, which means false positive loops. The majority of the false-positive loops are at the end of the list reflected by the sharp drops when recall reaches 100%. These mean that the ranking has good performance with exceptions. These false-positive loops that remained in the ranking are well handled by the last part, which tests and decides acceptance of each ranked loop. It shows one more strength of our method: it decides the acceptance of correct loops without a single user parameter, even when the ratio of true/false positive loops are drastically different.

6.5.2 SUN3D dataset

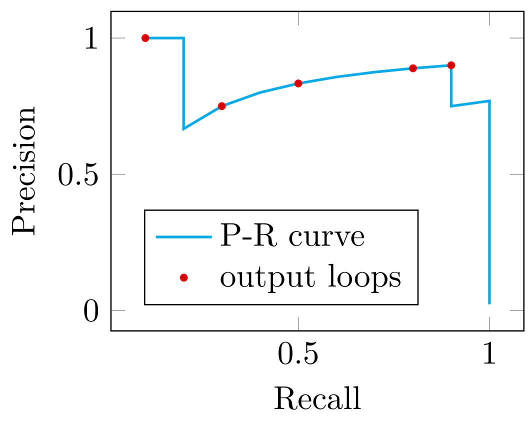
The SUN3D dataset [119] is a large-scale RGB-D database. It contains many data sequences captured at many places. Among them, there are eight sequences (listed here: `http://sun3d.cs.princeton.edu/listNow.html`) that are labeled with object annotations and widely used for evaluating SLAM and 3D reconstruction systems. We follow this common practice and run experiments on these sequences. For sequences `harvard_c5`, `harvard_c6` and `harvard_c8`, there is no loops detected on top of tracking. So we do not include results for them.



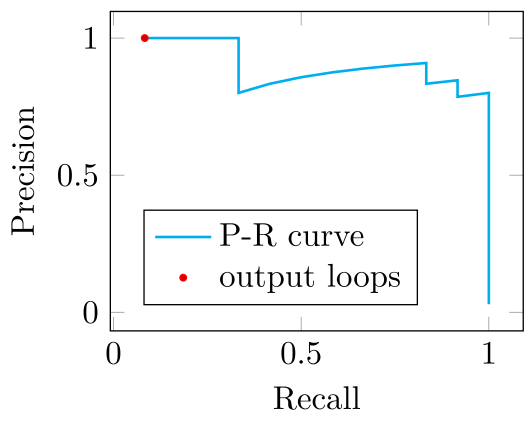
(a) livingroom 1



(b) livingroom 2



(c) office 1



(d) office 2

Figure 6.1: Precision and recall curves for ranking results of the proposed method. The curves show a good ranking performance with a few exceptions. These exceptions justifies the second pass on all the loops, so that they get removed properly. The locations of the accepted loops are marked with red dots on the plots. None of the false positive loops get passed. The system also do smart selection instead of choosing the top ranking loop detections.

Table 6.3: DMP performance difference of different methods on different sequences

	SUN3D	ORB SLAM2	CZK	BundleFusion	Tracking	Ours
mit_32_d507	573.95	750.60	334.17	441.15	904.25	**296.59**
maryland_hotel3	145.85	107.50	108.91	128.86	111.83	**96.56**
76_studyroom	448.84	1191.40	282.22	256.07	358.93	**193.94**
mit_dorm_next	46.38	51.88	734.50	944.81	87.04	**30.16**
mit_lab_hj	180.49	162.98	244.86	207.94	**155.86**	199.57

Table 6.4: Number of loops before and after loop sifting and majorization

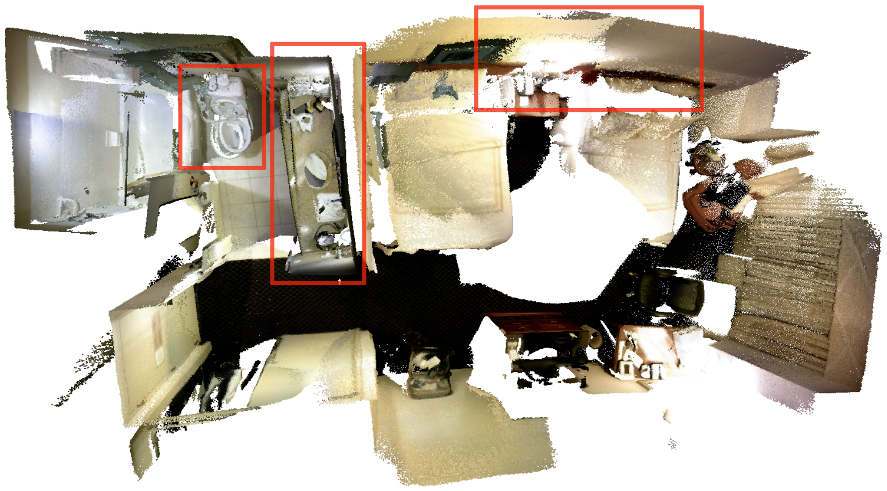
Number of loops	before	after
mit_32_d507	2135	34
maryland_hotel3	224	6
76_studyroom	442	7
mit_dorm_next	621	6
mit_lab_hj	219	7

Quantitatively, we evaluate the DMP metric of different methods and report them in Table 6.3. We compared with four different methods: 1) CZK, which is an offline method that targets the best surface reconstruction quality; 2) SUN3D, which is an offline method that adds manual object labeling as a source of loop closure; 3) ORB-SLAM2, which is a well known good SLAM that also has a tracking part in our system; 4) BundleFusion which is a well-engineered real-time dense SLAM system. The DMP metric evaluates that the proposed method makes reliable improvements on its initial start point: tracking result. Most importantly, it outperforms most methods. To understand the proposed method, we also include the number of loop detections and the number of loops that pass our algorithm, shown in Table 6.4.

Qualitative, we show results in the form of screenshots of reconstructed 3D models in Figs. 6.2, 6.3 and 6.4. In these figures, we highlight the mismatches in tracking results so that reader can better compare them with our results. SUN3D data sequences are scanned with very loopy motion in some areas but only once for some other scene parts, thus is considered difficult to process. Readers may refer to `http://redwood-data.org/indoor/models.html` for results of the CZK method and `https://graphics.stanford.edu/projects/bundlefusion/recons.html` for BundleFusion results for visual comparison.

6.6 Chapter summary

In this chapter, an algorithm that can sift and majorize loop detections is proposed. The algorithm tests and decides the acceptance of each loop without a single user-defined threshold. Experiments are conducted on public datasets, including the Augmented ICL-NUIM dataset and the SUN3D dataset. Results show that the proposed method outperforms the state-of-the-art methods. It can find key loops with 100% precision and eliminate significant mismatches when processing SUN3D sequences.


(a) Tracking only

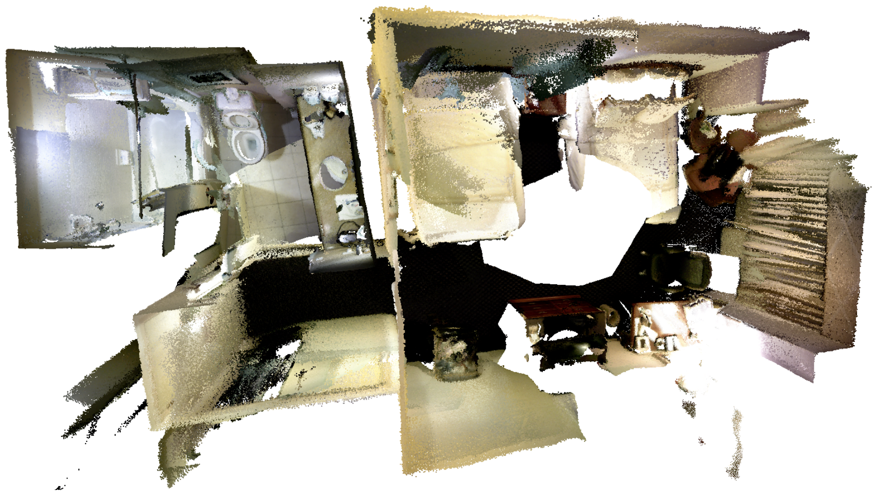
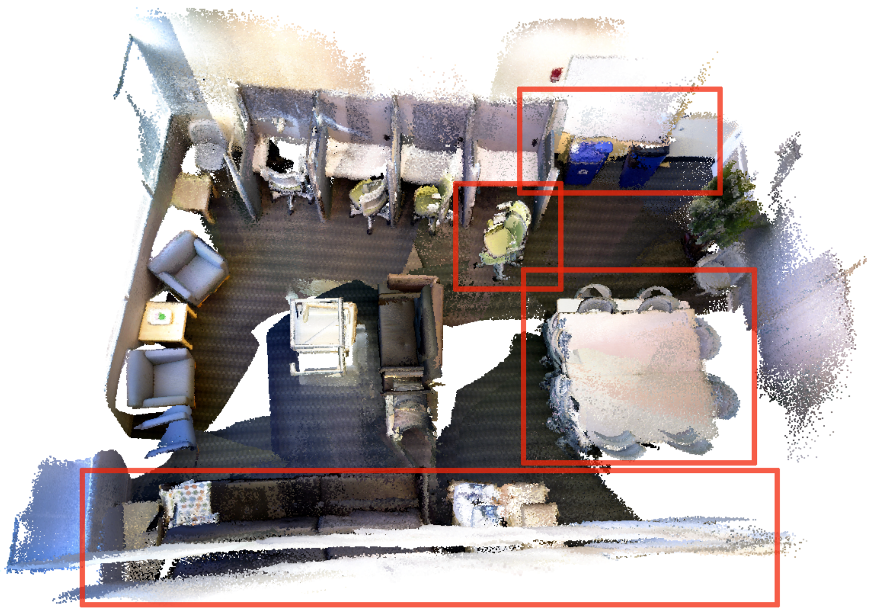
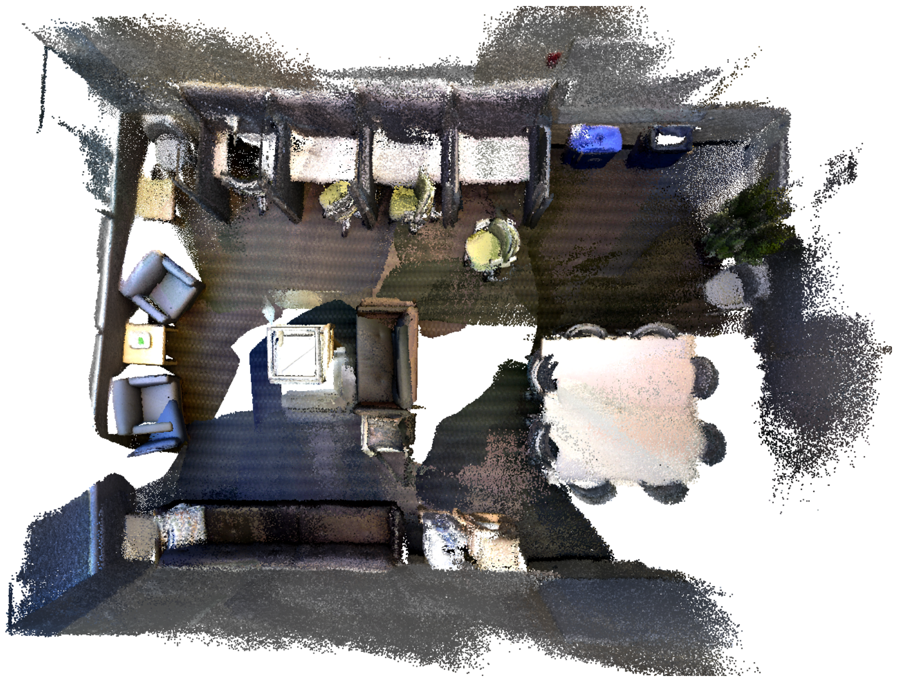

(b) With loop optimization

Figure 6.2: Results on sequence `maryland_hotel3` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.

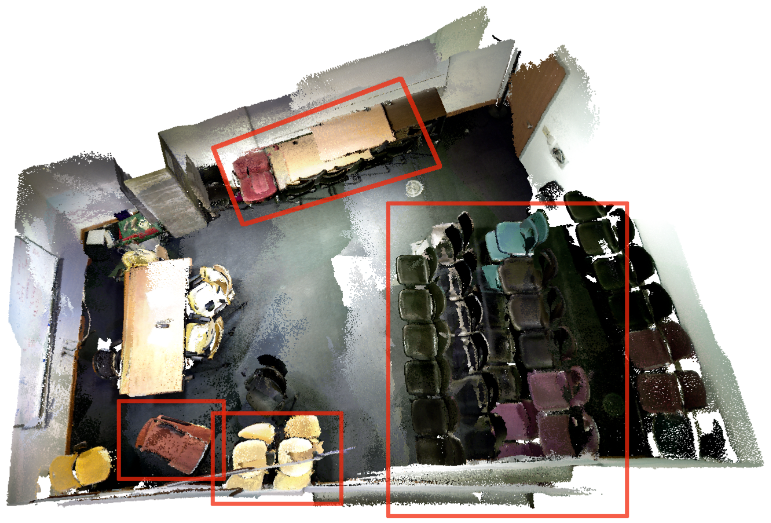


(a) Tracking only

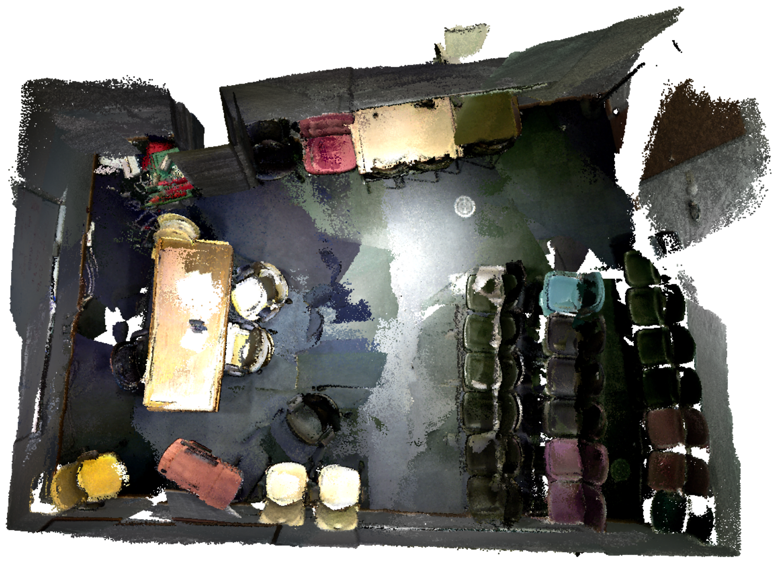


(b) With loop optimization

Figure 6.3: Results on sequence `76_studyroom` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.



(a) Tracking only



(b) With loop optimization

Figure 6.4: Results on sequence `mit_32_d507` of the SUN3D dataset. We highlight the mismatches in tracking results so that differences are easier to compare.

Chapter 7

3D Semantic Mapping of Cities for Autonomous Driving

7.1 Introduction

In previous chapters, we focus on the geometry aspect of 3D reconstruction and mapping. In this chapter, explorations are made on how to create semantic maps and benchmark semantic mapping methods. Semantic maps are important data abstraction from 3D mapping results. With higher-level semantic information, robots and autonomous vehicles can get better environmental awareness and do longer-term path and motion planning. It may also make the storing, transmitting, and retrieving of 3D mapping results more feasible, because semantic information usually has better long-term stability with smaller data footage.

At present, the concept of semantic mapping is not crystal clear yet [9] because it is still under active exploration. Some research works define semantic mapping as per 3D point/surfel/voxel labeling on dense mapping results [77, 82, 47], which, we believe, extends the idea of image segmentation to 3D space. Some others choose to build 3D point cloud maps with only traffic lines, and signs remained [95]. In this chapter, we focus on mapping 3D objects in the form of 3D bounding boxes for 3D semantic maps that are easy to store and transmit. Another motivation is that 3D object detection has shown promising performance improvements recently. Thus 3D bounding box input is increasingly accessible.

We list contributions of this chapter:

1. A simple and effective real-time 3D semantic mapping method is proposed. The proposed method takes per-frame bounding box detections and sensor (camera) extrinsic transformation estimates as inputs and produces a set of static 3D bounding boxes in a world coordinate system as 3D semantic mapping results.

2. A new benchmark is derived from the KITTI object tracking evaluation, since KITTI has no official ground truth semantic maps. In the new benchmark, ground-truth semantic maps are constructed based on GPS-IMU data and labeled 3D bounding boxes of KITTI.

3. Three novel semantic map-centered metrics are proposed for better evaluation of semantic mapping methods.

7.2 Related work

3D Object Detection. The research of 2D object detection has been very mature. Representative works can be divided into two main categories: Region Proposal Network (RPN) approaches, such as Faster R-CNN [91] and Mask R-CNN [129]. Single Shot MultiBox Detector (SSD) approaches like YOLO v1-YOLO v5. Compared with 2D object detection, there is a new requirement for 3D object detection. The 3D bounding box has three more angles: pitch, yaw, and roll, in addition to the position and size. In [102], PointRCNN for 3D object detection from raw point clouds is proposed. The whole framework is composed of two stages: stage-1 for the bottom-up 3D proposal generation and stage-2 for refining proposals in the canonical coordinates to obtain the final detection results. Instead of generating proposals from RGB images or projecting point clouds to bird's views, the stage-1 sub-network directly generates a small number of high-quality 3D proposals from point clouds in a bottom-up manner via segmenting the point cloud of the whole scene into foreground points and background. The stage-2 sub-network transforms the pooled points of each proposal to canonical coordinates to learn better local spatial features, which is combined with global semantic features of each point learned in stage-1 for accurate box refinement and confidence prediction.

3D Multi-Object Tracking. In [112], a 3D MOT system is proposed. An off-the-shelf 3D object detector is used to obtain oriented 3D bounding boxes from the LiDAR point cloud. Then, a combination of the 3D Kalman filter and Hungarian algorithm is used for state estimation and data association. Although it is a straightforward combination of standard methods, good results are observed on the KITTI dataset.

7.3 Semantic mapping benchmark dataset

It requires a large number of human efforts to annotate ground-truth labels for creating ground-truth semantic maps. So, instead of labeling new datasets, we turn to existing datasets and explore ways to convert feasible ones for semantic mapping evaluation.

Although there are several datasets with 3D bounding box annotations, as shown in Table 7.1, the purpose of these annotations is to provide ground truth for 3D object detection and 3D object tracking algorithms in autonomous driving scenes. For example, although nuScenes [15] dataset has the most labels, most of them are dynamic vehicles in highway scenes. For semantic mapping purposes, a large number of static objects are required. After comparison, we found that the multi-object tracking benchmark in the KITTI dataset meets the requirements and can be transformed for semantic mapping evaluation. This benchmark in KITTI has multiple data sequences with color stereo camera images, gray stereo camera images, LiDAR point clouds, GPS-IMU data together with ground-truth 3D bounding boxes for objects, and ground-truth data association among these objects.

Table 7.1: Datasets with ground-truth 3D bounding boxes available.

Dataset	Year	Sequence count	RGB frame #	LiDAR frame #	3D bounding box #
KITTI	2012	22	15K	15K	200K
AS lidar	2018	-	0	20K	475K
ApolloScape	2018	-	144K	0	70K
H3D	2019	160	83K	27K	1.1M
nuScenes	2019	1K	1.4M	400K	1.4M

Ground-truth camera trajectory. With the available GPS-IMU data, ground truth trajectory T_{wi} in the GPS-IMU coordinate frame can be derived. In the KITTI dataset, extrinsic parameters between sensors are also given. There are two transformation matrices that are useful for transforming GPS-IMU trajectory to camera trajectory. One is from the GPS-IMU coordinate to velodney coordinate T_{vi}. The another is from velodney coordinate to camera coordinate T_{cv}. Thus, the transformation from the GPU-IMU coordinate to the camera 0 frame is $T_{ci} = T_{P0} \cdot T_{cv} \cdot T_{vi}$. With this transformation, we can get the ground-truth trajectory of camera 0 with:

$$T_{wc} = T_{ci} \cdot T_{i0w} \cdot T_{wi} \cdot T_{ic}, \tag{7.1}$$

where T_{wc} is the transformation matrix from the camera coordinate to the world coordinate system.

Ground-truth semantic map. Considering that an object has several different 3D bounding box labels across multiple frames, and these boxes will not overlap perfectly due to a number of error factors. To address this problem, we fuse these boxes of the same object in the world coordinate system. First, the center and size of a 3D bounding box is computed by

$$C_w = \frac{\sum_{i=1}^{N_b} C_{wi}}{N_b}, \tag{7.2}$$

where C_w is x or y or z, which represents the center of a object 3D bounding box. N_b is the number of these boxes of the i-th object belongs to $(0, N_b)$.

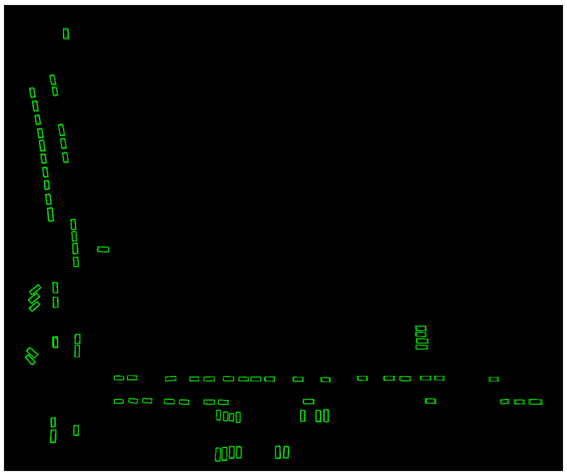
For the orientation of these bounding boxes, obviously, we can not use the same method. In the KITTI dataset, the θ is used to represent the orientation of the object. It can be converted to a rotation matrix

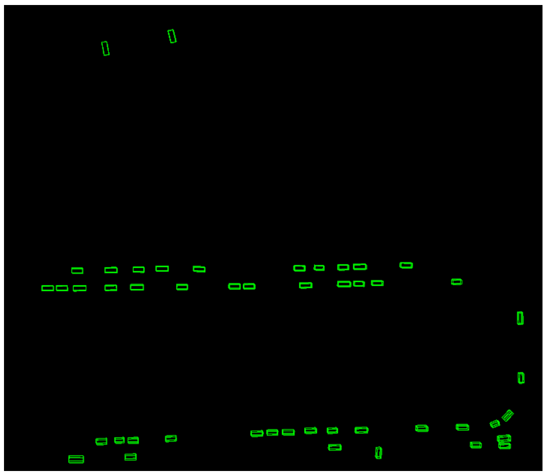
$$R_{cb}(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}. \tag{7.3}$$

Then, orientation R_{wb} of the 3D bounding box in the world coordinate system can be computed by

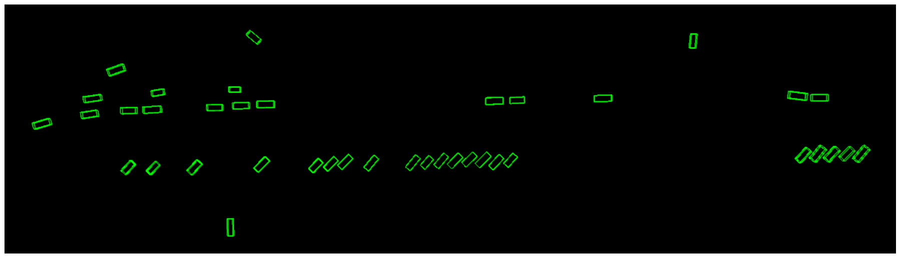
$$R_{wb} = R_{wc} R_{cb}, \tag{7.4}$$

By converting the rotation matrix R_{wb} to the Euler angle representation, we have three Euler angles. Then, we can use the same method as in equation (7.2).


(a) sequence 01


(b) sequence 07

(c) sequence 09


(d) sequence 11

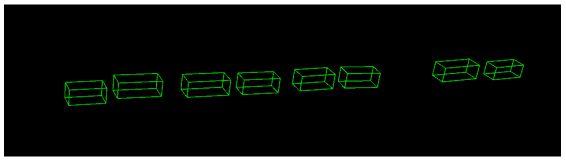

(e) sequence 14

Figure 7.1: Visualization of our generated ground-truth semantic maps. Each green box is for a parked car.

Table 7.2: Information of the generated ground-truth semantic maps. Storage is caculated when stored in 64 bit floating point numbers.

Sequence	Static object number	Odometry (m)	Storage (KB)
01	81	361	5.8
07	48	352	3.5
09	70	614	5.0
11	40	224	2.9
14	12	61	0.9

Finally, we get the 3D bounding boxes ground truth of each object in the world coordinate system. These ground-truth semantic maps are visualized in Fig. 7.1. For semantic mapping evaluation, we find five suitable sequences: 01, 07, 09, 11, 14 from the KITTI multi-object tracking benchmark. The information of these maps is shown in Table 7.2.

7.4 Semantic mapping method

The proposed semantic mapping method takes camera tracking and 3D object detection as input to process then generate 3D bounding boxes in world frames. This process will then fuse observation of objects across different frames into a map of objects in the form of 3D bounding boxes. The core of the mapping method is a Kalman filter for model update and the Hungarian method as data association.

We define states of a Kalman filter for object fusion as $x, y, z, \varphi, \theta, \psi, l, w, h, v_x, v_y, v_z$, which consists of bounding box location center x, y, z, the three angle that represents the direction of a bounding box, length l, width w, height h, and speed of the movement of the bounding box v_x, v_y, v_z. For the defined state, the state transition function is

$$\begin{cases} x^{k+1} = x^k + v_x^k + \sigma_x, \\ y^{k+1} = y^k + v_y^k + \sigma_y, \\ z^{k+1} = z^k + v_z^k + \sigma_z, \\ \varphi^{k+1} = \varphi^k + \sigma_\varphi, \\ \theta^{k+1} = \theta^k + \sigma_\theta, \\ \psi^{k+1} = \psi^k + \sigma_\psi, \\ l^{k+1} = l^k + \sigma_l, \\ w^{k+1} = w^k + \sigma_w, \\ h^{k+1} = h^k + \sigma_h, \\ v_x^{k+1} = v_x^k + \sigma_{vx} \\ v_y^{k+1} = v_y^k + \sigma_{vy} \\ v_z^{k+1} = v_z^k + \sigma_{vz} \end{cases} \tag{7.5}$$

where each σ represents state transition noise for a state dimension. The observation functions are simpler since all the states except the velocities are directly observable.

$$\begin{bmatrix} x_o^k \\ y_o^k \\ z_o^k \\ \varphi_o^k \\ \theta_o^k \\ \psi_o^k \\ l_o^k \\ w_o^k \\ h_o^k \end{bmatrix} = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0 \end{bmatrix} \begin{bmatrix} x^k \\ y^k \\ z^k \\ \varphi^k \\ \theta^k \\ \psi^k \\ l^k \\ w^k \\ h^k \\ v_x^k \\ v_y^k \\ v_z^k \end{bmatrix} + \boldsymbol{\sigma}_{obs} \tag{7.6}$$

where underscript o denotes observation of the data dimension. Note that the abstracted observation is in the world frame. Thus, the real observations need to be first transformed to the world frame. Given the detection bounding boxes set $\mathbb{B}_c$, where each bound box is denoted as B_c and camera pose Twc_i. $\mathbb{B}_c$ are transformed to world frame, denoted as

$$\mathbb{B}_w = Twc_i(\mathbb{B}_c), \tag{7.7}$$

where $Twc_i(\cdot)$ is a function that transforms bounding boxes with the transformation Twc_i. Given the abstracted observation $\mathbb{B}_w$ and bounding box predictions $\overline{\mathbb{B}_w}$, intersection over union (IoU) of two set of bounding boxes are calculated and fed into the Hungarian algorithm to find data association between them. If a $B_w \in \mathbb{B}_w$ is matched with predicted bounding boxes, the observation will be used to update the corresponding Kalman filter. Otherwise, it will be used to initialize a new object with a new Kalman filter on the semantic map.

7.4.1 Object state transition

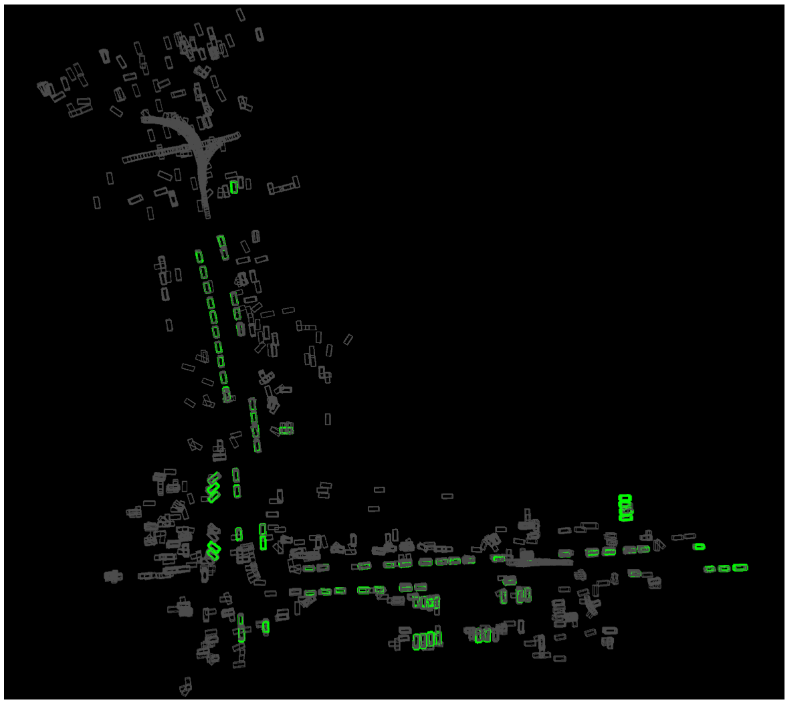


Figure 7.2: Visualization of noisy object detection input (in gray color) vs. the ground-truth objects (in green) for sequence 01. The noisy object detection results are transformed to the world frame with the ground-truth camera trajectory for a fair visual comparision with true objects in the ground-truth map.

There is noise in object detection results, as shown in Fig. 7.2, that are fed into the pipeline. The false positive object detections would cause false objects in mapping results if not handled. To deal with the issue, we propose adding object states to an object in a map. The states are 1) unstable object, which is the entry state when a new object is created for the map. An object will stay in the unstable state if the object is observed less than h times. 2) dynamic object, which is an object that is moving. Any object that ever moved with a velocity greater than v_t will have this state. 3) static activate, which is an object with a velocity smaller than v_t all the time and observed at least in k frames. 4) static inactivate, which is the destination of static activate object when not observed once in the last k frames. 5) inactivate state, which will take unstable and dynamic objects that are not observed once in the last k frames. The transition between the states is shown in figure 7.3. With these states, we take all the static objects as the mapping results.

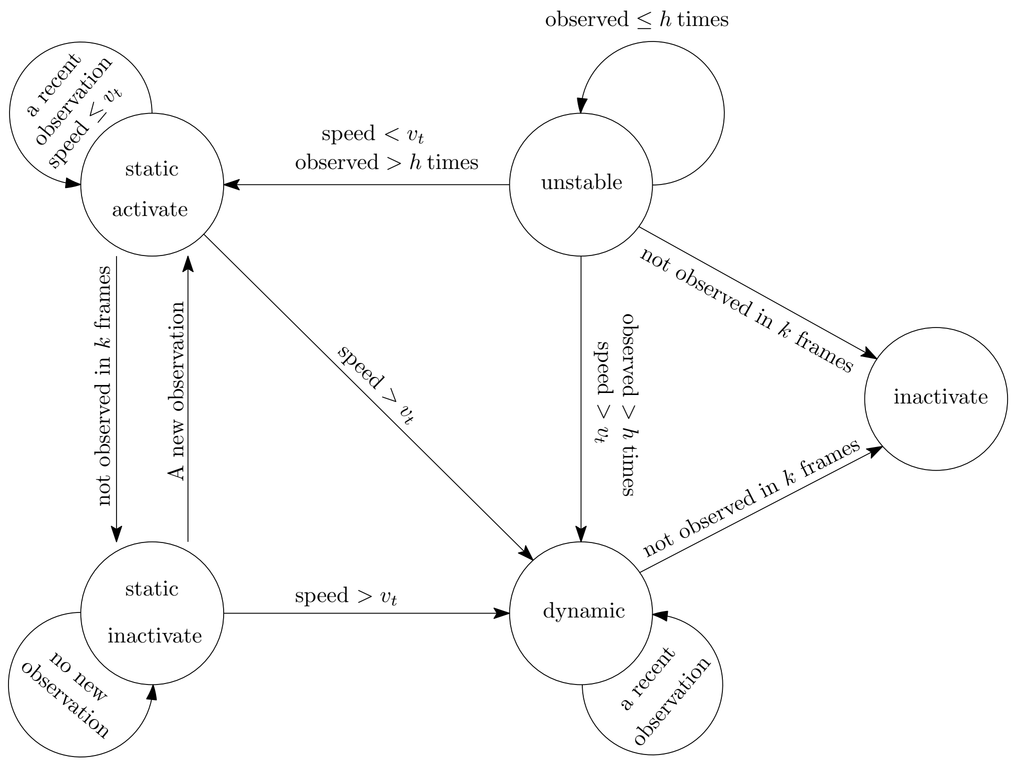

observed $\leq h$ times
a recent observation speed $\leq v_t$
static activate
speed $< v_t$ observed $> h$ times
unstable
not observed in k frames
A new observation
speed $> v_t$
observed $> h$ times speed $> v_t$
not observed in k frames
inactivate
not observed in k frames
static inactivate
speed $> v_t$
dynamic
no new observation
a recent observation

Figure 7.3: State transition finite state machine

7.5 Evaluation of semantic mapping

To evaluate semantic mapping methods, existing metrics: intersection over union and precision-recall are adopted. In addition to that, three additional map-focused metrics are proposed to better evaluate the performance of semantic mapping methods. Two of them, as in section 7.5.3, focus on full-map absolute distances between matched objects, while the other one, as in section 7.5.5, focuses on frame-based relative error in virtual observations.

We run simple baseline experiments to evaluate the performance of the proposed method on the aforementioned KITTI sequences with the metrics.

7.5.1 Visual evaluation

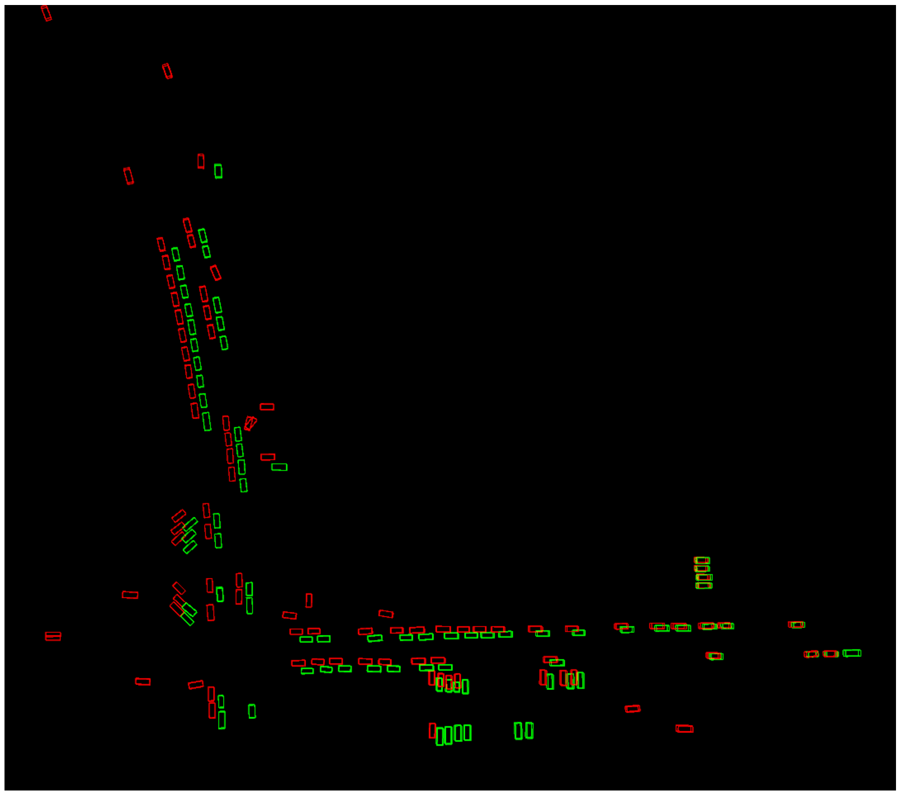
Semantic mapping results of the proposed baseline method are visualized and reported in Figs. 7.4, 7.5, 7.6, 7.7, and 7.8. To better understand mapping performance, the input camera trajectories are also visualized comparing ground-truth camera trajectories. In the figures, we can see that the estimated 3D bounding boxes in the estimated maps are accurate relative to estimated camera trajectories. Most of the objects presented in the ground-truth maps are detected in the estimated maps.

7.5.2 Precision and recall

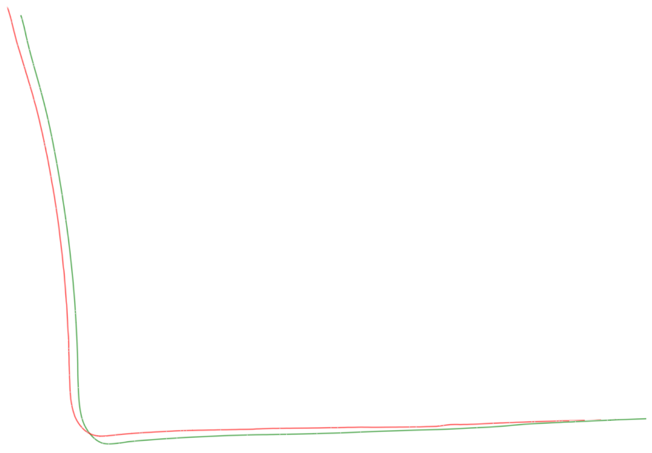
For object detection, precision and recall are usually used to analyze the effectiveness of the detection methods quantitatively. The true positive (TP) detections are the detections that agree with ground-truth labels with higher than threshold IoU values. False positive (FP) represents the bounding boxes that are detected by an algorithm but do not appear in the ground truth. False negative (FN) represents the number of ground-truth bounding boxes that are missing from detection results. Precision and recall then can be calculated as in (7.8) and (7.9), respectively.

$$\text{Precision} = \frac{TP}{TP + FP}, \tag{7.8}$$

$$\text{Recall} = \frac{TP}{TP + FN}. \tag{7.9}$$

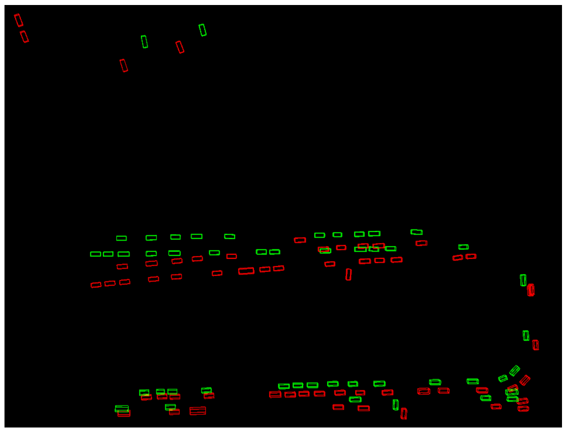


(a) Red: estimated map. Green: ground-truth map



(b) Red: estimated trajectory. Green: ground-truth trajectory

Figure 7.4: Visualization of estimated ground-truth semantic map and input camera trajectory for sequence 01. We can see that the estimated 3D bounding boxes in the estimated map are accurate relative to estimated camera trajectory. Most of objects presented in the ground-truth map are detected in the estimated map.


(a) Red: estimated map. Green: ground-truth map

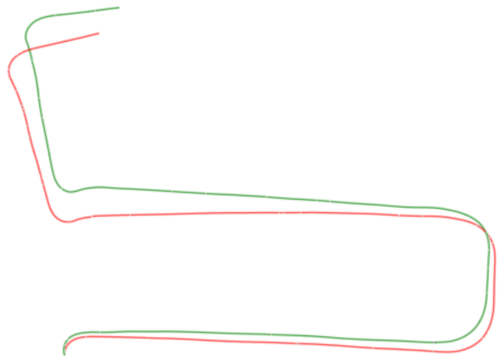
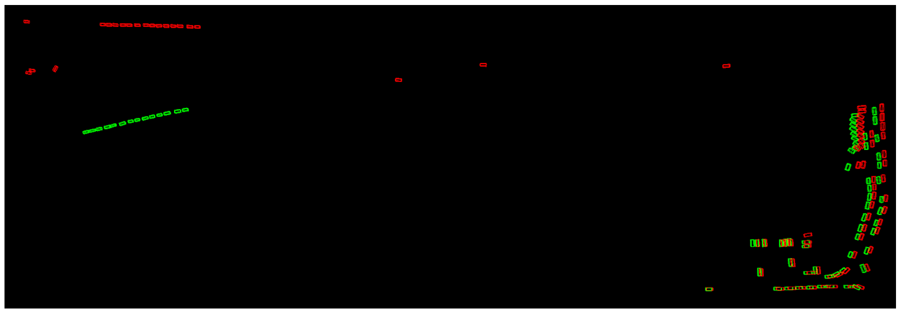

(b) Red: estimated trajectory. Green: ground-truth trajectory

Figure 7.5: Visualization of estimated ground-truth semantic map and input camera trajectory for sequence 07.


(a) Red: estimated map. Green: ground-truth map

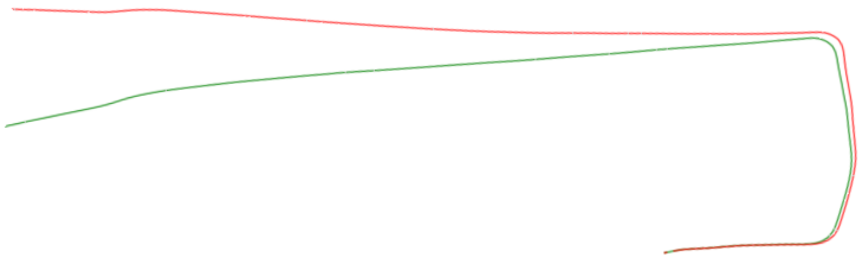
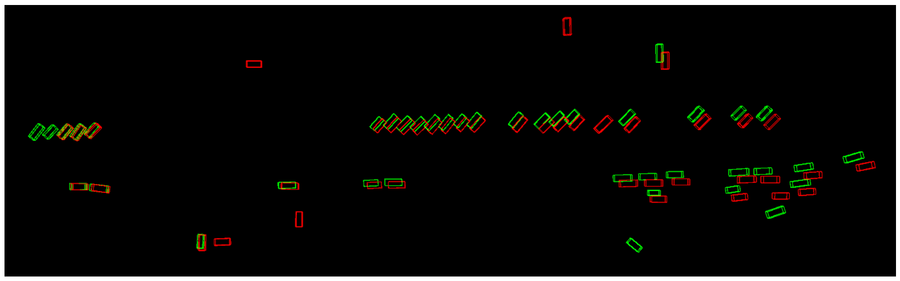

(b) Red: estimated trajectory. Green: ground-truth trajectory

Figure 7.6: Visualization of estimated ground-truth semantic map and input camera trajectory for sequence 09. We can see that the estimated 3D bounding boxes in the estimated map are accurate relative to estimated camera trajectory. Most of objects presented in the ground-truth map are detected in the estimated map.


(a) Red: estimated map. Green: ground-truth map

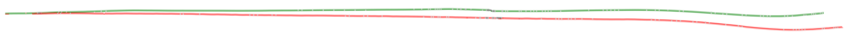
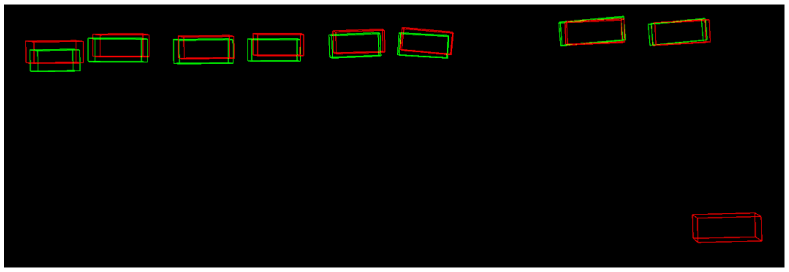

(b) Red: estimated trajectory. Green: ground-truth trajectory

Figure 7.7: Visualization of estimated ground-truth semantic map and input camera trajectory for sequence 11.


(a) Red: estimated map. Green: ground-truth map

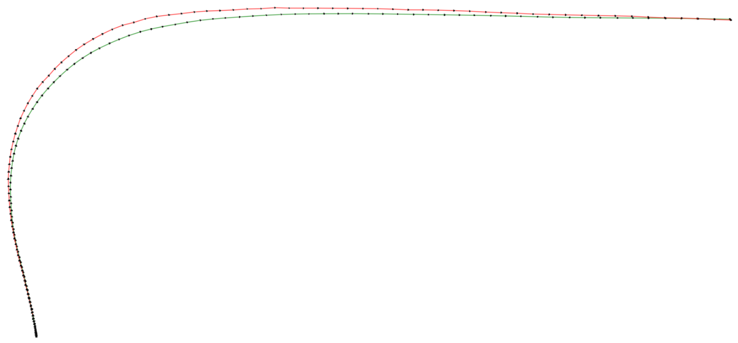

(b) Red: estimated trajectory. Green: ground-truth trajectory

Figure 7.8: Visualization of estimated ground-truth semantic map and input camera trajectory for sequence 14. We can see that the estimated 3D bounding boxes in the estimated map are accurate relative to estimated camera trajectory. Most of objects presented in the ground-truth map are detected in the estimated map.

Table 7.3: The precision-recall results of proposed algorithm in terms of object detected in maps.

Sequence	01	07	09	11	14
TP	71	47	68	35	8
FP	20	12	12	7	1
FN	10	1	2	5	4
Precision	78.0	79.7	85.0	83.3	88.9
Recall	87.7	97.9	97.1	87.5	66.7

Similarly, one can calculate the value of precision and recall for the proposed semantic map algorithm, except that data associations are determined by IoU in virtual observation space instead of in world coordinates. The results are shown in Table 7.3 ,which is based on IOU_{3D} not less than 35%. As we can see, the value of precision is no less than 78%, and the recall is larger than 66%, which shows the good performance of our algorithms.

7.5.3 Absolute object distance

One key evaluation aspect of 3D mapping methods is to know the positional error of 3D environments. For 3D object-centered semantic mapping methods, we propose to evaluate absolute object distance (AOD) between. The AOD metric calculates as follows. Given an estimated semantic map M^e, a ground-truth semantic map M^{GT}, and data association A that indicates the matched objects between the two maps, the metric calculates the average distance between two sets of matches objects. There are two variants of the metric. One calculates the distance directly, denoted as direct AOD (DAOD), which is useful for cases when absolute position error is important. Another one, denoted as aligned AOD (AAOD), first transfer estimated map M^e to be best aligned with the ground-truth map then calculate distance between them. The AAOD is more useful for cases where positioning between mapped objects is more important than the absolute positioning of a single object in a map.

$$\text{DAOD}(M^e, M^{GT}, A) = \frac{\sum_{a \in A} ||M^e_a - M^{GT}_a||_2}{|A|}, \tag{7.10}$$

Table 7.4: DAOD and AAOD of the proposed method

Sequence	DAOD (m)	AAOD (m)
01	3.55	1.18
07	6.37	2.42
09	17.06	7.59
11	1.71	0.56
14	0.40	0.12

where $||\cdot||_2$ is the L^2 norm of a vector and M_a means taking center of an object from M with index in data association a.

$$\text{AAOD}(M^e, M^{GT}, A) = \frac{\sum_{a\in A} ||T^A \cdot M_a^e - M_a^{GT}||_2}{|A|}, \tag{7.11}$$

where $T^A \in SE(3)$ is a 3D transformation that aligns M^e with M^{GT} by

$$T^A = \underset{T^A}{\operatorname{argmin}} \sum_{a\in A} ||T^A \cdot M_a^e - M_a^{GT}||_2. \tag{7.12}$$

For the data association A, it can be calculated by matching IoUs between virtual observations on both maps when both the estimated camera trajectory and the ground-truth trajectory are available. In the worst case, it can be generated with human labeling.

7.5.4 Experiments

When evaluating absolute object distance metrics, data association between ground-truth maps and estimated maps is done through IoU score in virtual observations as stated in PRVO metric with IoU threshold of 60%. Results for the two AOD metrics are in Table 7.4.

7.5.5 Precision and recall of virtual observations

To evaluate the performance of object detection fusion performance while isolating the influence of camera trajectory estimation error from SLAM methods, we propose another metric called precision and recall of virtual observations (PRVO). Inputs for the PRVO metric are: an estimated semantic map M^e with corresponding estimated camera trajectory $\{T\}^e$ and ground-truth semantic map M^{GT} with ground-truth camera trajectory $\{T\}^{GT}$.

Table 7.5: PRVO of the proposed method

Sequence	PRVO	
	Precision (%)	Recall (%)
01	50.0	59.9
07	36.6	47.4
09	44.1	48.7
11	58.5	62.6
14	61.1	74.0

For each data frame, virtual observations are generated for both the estimated map and the ground-truth map by applying sensor model $s(\cdot)$

$$\begin{aligned} \{B\}_v^e &= s(M^e, T^e) \text{ for } T^e \in \{T\}^e \\ \{B\}_v^{GT} &= s(M^{GT}, T^{GT}) \text{ for } T^{GT} \in \{T\}^{GT}. \end{aligned} \tag{7.13}$$

An IoU threshold is used to determine the number of false positive (fp), false negative (fn), and true positive (tn) between $\{B\}_v^e$ and $\{B\}_v^{GT}$ for a frame. For all the frames, aggregated total numbers are calculated as $TN = \sum tn$, $FN = \sum fn$, and $TP = \sum tp$. Finally, a pair of precision and recall values are computed from TN, FN, and TP.

Experiments

When evaluating PRVO metric, the IoU threshold is set to 50%. Results are reported in Table 7.5.

7.6 Chapter summary

In this chapter, a simple and effective real-time 3D semantic mapping method is proposed. The proposed method takes per-frame bounding box detections and sensor (camera) extrinsic transformation estimates as inputs and produces a set of static 3D bounding boxes in a world coordinate system as 3D semantic mapping results.

To evaluate the proposed method, a new benchmark is derived from KITTI object tracking evaluation. Ground-truth semantic maps are constructed based on GPS/IMU data of KITTI and the labeled 3D bounding box. By fusing multiple annotation bounding boxes of the same object from frames, we get one single 3D bounding box for each object in the world frame as ground-truth semantic maps. Three novel semantic map-centered metrics: DAOD, AAOD, and PRVO are also proposed. Experiments are conducted to evaluate the proposed method. The set of proposed method, metric and benchmarking dataset will serve as a new benchmark platform for easier comparison of new methods.

Chapter 8

Conclusion and Future Opportunities

8.1 Concluding remarks

In this dissertation, various key contributions have been made and tested to make it closer for a system that can optimally reconstruct and map 3D environments.

In Chapter 3, a RANSAC-based algorithm that samples data with a Lévy distribution after data ranking is proposed. The proposed method is evaluated on both simulation and real-world public datasets. In experiments, our method shows better results compared with the uniform baseline method.

In Chapter 4, a novel 3D reconstruction system is presented. The system maps both large archaeological caves and general indoor environments with RGB-D cameras. The proposed system produces accurate 3D models by detecting and optimizing surface loops in sparse feature-based visual SLAM systems. By adding surface loop closure into a vSLAM system, globally consistent and optimal 3D models are generated accurately. The proposed system consists of five components: 1) sparse feature-based camera tracking from ORB-SLAM2, 2) surface model fusion powered by Surfels, 3) a novel fast surface loop detection algorithm, 4) a novel surface loop filtering method, and 5) loop optimization based on sparse feature-based bundle adjustment. Experiments on datasets show that the proposed system produces results on-par or better than baseline methods.

In Chapter 5, we propose a metric, which can work without any ground truth data, for evaluating 3D reconstruction and mapping performance. In our experiments, the proposed metric DMP is benchmarked against ground truth-based metrics. Results show that DMP can provide a similar evaluation capability. The proposed metric not only makes 3D mapping evaluation simpler, but also opens many new opportunities. Our experiments show that it can evaluate loop detection results and lead to good precision-recall performance. We envision that more can be done with this metric, such as self-supervised methods and more available datasets.

In Chapter 6, an algorithm that can sift and majorize loop detections is proposed. The algorithm tests and decides the acceptance of each loop without a single user-defined threshold. Experiments are conducted on public datasets, including the Augmented ICL-NUIM dataset and the SUN3D dataset. Results show that the proposed method outperforms the state-of-the-art method. It can find key loops with 100% precision and eliminate significant mismatches when processing SUN3D sequences.

In Chapter 7, a simple and effective real-time 3D semantic mapping method is proposed. The proposed method takes per-frame bounding box detections and sensor (camera) extrinsic transformation estimates as inputs and produces a set of static 3D bounding boxes in a world coordinate system as 3D semantic mapping results. To evaluate the proposed method, a new benchmark is derived from KITTI object tracking evaluation. Three novel semantic map-centered metrics: DAOD, AAOD, and PRVO are also proposed. Experiments are conducted to evaluate the proposed method. The set of proposed method, metric, and benchmarking dataset will serve as a new benchmark platform for easier comparison of new methods.

8.2 Future challenges and opportunities

3D reconstruction and semantic mapping are still far from being matured. Based on the current achievements, we present three directions for future investigation.

8.2.1 Machine learning methods with the DMP metric

We envision that the DMP metric, presented in Chapter 5, will open new opportunities for 3D machine learning. Especially, it enables self-supervised machine learning methods in 3D reconstruction and mapping research because the DMP metric can evaluate 3D reconstruction performance without costly ground-truth data. For example, one may use the DMP metric as the reward function to build reinforcement learning-based methods. Or, the DMP metric may be used as a self-supervising loss to train machine learning models. These new opportunities have the potential to create a brand new branch for 3D machine learning methods.

8.2.2 More advanced semantic mapping

There are many possible improvements on the semantic mapping method presented. A major one is to include object appearance into the data association step so that it will be more robust for cluttered environments. There are other potential improvements, such as reducing the influence of false-positive detections out of object detection methods, and adding more information, object property, etc., to map objects.

In addition to the improvements on the proposed method in Chapter 7, there are interesting works that can be done to feed information back to supporting object detections and camera trajectory estimation methods so that they can perform better with additional information made available.

8.2.3 Navigation and localization on saved semantic maps

One of the major purposes of 3D mapping is for navigation and localization. However, rare usage has been noticed in practical systems. One reason is that the memory footage of a traditional map is too big for practical storage and transmission. One advantage of 3D semantic mapping is that semantic maps are highly abstracted with only important information. This can greatly reduce the memory pressure imposed on practical systems. This lower memory requirement is great. However, this type of map is so abstracted that much geometry information is missing. So new explorations have to be done for reliable localization on such maps. For example, what information about objects need to be stored in a semantic map. Also, it will be beneficial to figure out how to track camera motion precisely over semantic maps for better localization accuracy.

8.2.4 Multi-session map fusion and map update over time

There are many use cases that require the ability to fuse multiple maps or update maps. For example, When mapping a large environment, such as large caves or city scale environments, it is almost impossible to capture enough data in one session. A reliable method for the multi-session map fusion method is highly desired. In addition to simple map fusion for coverage, another important one is to update maps with new data. Examples are: update a mapped city environment to reflect constructions of new buildings and seasonal changes in environment appearances. It is crucial to be able to efficiently identify and update changes in maps while preserving unchanged portions.

Bibliography

[1] Pratik Agarwal, Gian Diego Tipaldi, Luciano Spinello, Cyrill Stachniss, and Wolfram Burgard. Robust map optimization using dynamic covariance scaling. In *Proc. of the 2013 IEEE International Conference on Robotics and Automation*. IEEE, May 2013. 5, 72

[2] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M. Seitz, and Richard Szeliski. Building Rome in a day. In *Proc. of the 2009 IEEE 12th International Conference on Computer Vision*. IEEE, September 2009. 4

[3] A. Angeli, D. Filliat, S. Doncieux, and J. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics*, 24(5):1027–1037, October 2008. 6, 7

[4] Adrien Angeli, Stephane Doncieux, Jean-Arcady Meyer, and David Filliat. Real-time visual loop-closure detection. In *Proc. of the 2008 IEEE International Conference on Robotics and Automation*, pages 1842–1847. IEEE, May 2008. 6

[5] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, September 1987. 17

[6] L. Bampis, A. Amanatiadis, and A. Gasteratos. Encoding the description of image sequences: A two-layered pipeline for loop closure detection. In *Proc. of the IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 4530–4536, October 2016. 7

[7] Loukas Bampis, Angelos Amanatiadis, and Antonios Gasteratos. Fast loop-closure detection using visual-word-vectors from image sequences. *International Journal of Robotics Research*, 37(1):62–82, 2018. 7

[8] Daniel Barath and Jiří Matas. Graph-Cut RANSAC. In *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 21

[9] H. Bavle, P. De La Puente, J. P. How, and P. Campoy. VPS-SLAM: Visual Planar Semantic SLAM for Aerial Robotic Systems. *IEEE Access*, 8:60704–60718, 2020. 83

[10] Jan Bayer, Petr Čížek, and Jan Faigl. On construction of a reliable ground truth for evaluation of visual SLAM algorithms. In *Proc. of the 2016 Conference on Planning in Artificial Intelligence and Robotics*, volume 6, pages 1–5, November 2016. 61

[11] P. J. Besl and Neil D. McKay. A method for registration of 3-D shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, February 1992. 17

[12] M. Boulekchour and N. Aouf. Efficient real-time loop closure detection using GMM and tree structure. In *Proc. of the IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 4944–4949, September 2014. 7

[13] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - Differentiable RANSAC for Camera Localization. In *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 21

[14] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, Jose Neira, Ian Reid, and John J. Leonard. Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Transactions on Robotics*, 32(6):1309–1332, December 2016. 1, 61, 70, 71

[15] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom. nuScenes: A Multimodal Dataset for Autonomous Driving. In *Proc. of the 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11618–11628, 2020. 85

[16] Silvia Cascianelli, Gabriele Costante, Enrico Bellocchio, Paolo Valigi, Mario Luca Fravolini, and Thomas A. Ciarfuglia. Robust visual semi-semantic loop closure detection by a covisibility graph and CNN features. *Robotics and Autonomous Systems*, 92:53–65, 2017. 7

[17] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray. Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras. In *Proc. of 2007 IEEE International Conference on Robotics and Automation*, pages 4102–4107, April 2007. 2

[18] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *Proc. of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, June 2015. 4, 13, 14, 19, 27, 31, 35, 36, 40, 42, 46, 50, 61, 62, 65, 72, 76

[19] Sunglok Choi, Taemin Kim, and Wonpil Yu. Performance Evaluation of RANSAC Family. In *Proc. of the British Machine Vision Conference 2009*. British Machine Vision Association, 2009. 20

[20] O. Chum and J. Matas. Matching with PROSAC — Progressive Sample Consensus. In *Proc. of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. IEEE, 2005. 21

[21] Ondřej Chum, Jiří Matas, and Josef Kittler. Locally Optimized RANSAC. In *Lecture Notes in Computer Science*, pages 236–243. Springer Berlin Heidelberg, 2003. 20, 21

[22] T. A. Ciarfuglia, G. Costante, P. Valigi, and E. Ricci. A discriminative approach for appearance based loop closing. In *Proc. of the IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, pages 3837–3843, October 2012. 7

[23] Alejo Concha, Giuseppe Loianno, Vijay Kumar, and Javier Civera. Visual-inertial direct SLAM. In *Proc. of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2016. 70

[24] David Crandall, Andrew Owens, Noah Snavely, and Dan Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *Proc. of the CVPR 2011*. IEEE, June 2011. 4

[25] M. Cummins and P. Newman. Probabilistic appearance based navigation and loop closing. In *Proc. of the 2007 IEEE Int. Conf. Robotics and Automation*, pages 2042–2048, April 2007. 6

[26] Mark Cummins and Paul Newman. FAB-MAP: Probabilistic Localization and Mapping in the Space of Appearance. *International Journal of Robotics Research*, 27(6):647–665, 2008. 6, 70

[27] Mark Cummins and Paul Newman. Appearance-only SLAM at large scale with FAB-MAP 2.0. *International Journal of Robotics Research*, 30:1100–1123, 2011. 6

[28] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias NieBner. ScanNet: Richly-Annotated 3D Reconstructions of Indoor Scenes. In *Proc. of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017. 61

[29] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. BundleFusion: Real-Time Globally Consistent 3D Reconstruction Using On-the-Fly Surface Reintegration. *ACM Transactions on Graphics*, 36(3):1–18, May 2017. 2, 70, 72

[30] A. J. Davison and D. W. Murray. Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):865–880, July 2002. 1

[31] Jingming Dong, Xiaohan Fei, and Stefano Soatto. Visual-inertial-semantic scene representation for 3D object detection. In *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 70

[32] Felix Endres, Jurgen Hess, Jurgen Sturm, Daniel Cremers, and Wolfram Burgard. 3-D Mapping With an RGB-D Camera. *IEEE Transactions on Robotics*, 30(1):177–187, February 2014. 70, 72

[33] Jakob Engel, Thomas Schöps, and Daniel Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *Proc. of the Computer Vision – ECCV 2014*, pages 834–849. Springer International Publishing, 2014. 70

[34] Lorenzo Fernández, Luis Payá, Óscar Reinoso, Arturo Gil, and David Valiente. Visual hybrid SLAM: an appearance-based approach to loop closure. In Manuel A. Armada, Alberto Sanfeliu, and Manuel Ferre, editors, *Proc. of the ROBOT 2013: First Iberian Robotics Conference - Advances in Robotics, Vol. 1, Madrid, Spain, 28-29 November 2013*, volume 252 of *Advances in Intelligent Systems and Computing*, pages 693–701. Springer, 2013. 7

[35] Michael Firman. RGBD datasets: Past, present and future. In *Proc. of the 2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. IEEE, June 2016. 61

[36] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, June 1981. 14, 19, 20, 21, 33

[37] Victor Fragoso, Pradeep Sen, Sergio Rodriguez, and Matthew Turk. EVSAC: Accelerating Hypotheses Generation by Modeling Matching Scores with Extreme Value Theory. In *Proc. of the 2013 IEEE International Conference on Computer Vision*. IEEE, December 2013. 21

[38] Yasutaka Furukawa, Brian Curless, Steven M. Seitz, and Richard Szeliski. Towards Internet-scale multi-view stereo. In *Proc. of the 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, June 2010. 4

[39] Yasutaka Furukawa and Jean Ponce. Accurate, Dense, and Robust Multiview Stereopsis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1362–1376, August 2010. 4

[40] D. Galvez-López and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012. 6, 7, 70

[41] Adrian Garcea, Jiazhen Zhu, Dominik Van Opdenbosch, and Eckehard Steinbach. Robust map alignment for cooperative visual SLAM. In *Proc. of the 2018 25th IEEE International Conference on Image Processing (ICIP)*. IEEE, October 2018. 61

[42] E. Garcia-Fidalgo and A. Ortiz. On the use of binary feature descriptors for loop closure detection. In *Proc. of the IEEE Emerging Technology and Factory Automation (ETFA)*, pages 1–8, September 2014. 7

[43] E. Garcia-Fidalgo and A. Ortiz. ibow-LCD: An appearance-based loop-closure detection approach using incremental bags of binary words. *IEEE Robotics and Automation Letters*, 3(4):3051–3057, October 2018. 7

[44] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proc. of the 2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, June 2012. 31

[45] Ben Glocker, Jamie Shotton, Antonio Criminisi, and Shahram Izadi. Real-time RGB-D camera relocalization via randomized ferns for keyframe encoding. *IEEE Transactions on Visualization and Computer Graphics*, 21(5):571–583, May 2015. 3

[46] R. Gorenflo and F. Mainardi. Fractional calculus and stable probability distributions. *Archives of Mechanics*, 50:377–388, 1998. 24

[47] Margarita Grinvald, Fadri Furrer, Tonci Novkovic, Jen Jen Chung, Cesar Cadena, Roland Siegwart, and Juan Nieto. Volumetric instance-aware semantic mapping and 3D object discovery. *IEEE Robotics and Automation Letters*, 4(3):3037–3044, July 2019. 83

[48] F. Han, X. Yang, Y. Deng, M. Rentschler, D. Yang, and H. Zhang. SRAL: Shared Representative Appearance Learning for Long-Term Visual Place Recognition. *IEEE Robotics and Automation Letters*, 2(2):1172–1179, April 2017. 7

[49] Fei Han, Hua Wang, Guoquan Huang, and Hao Zhang. Sequence-based sparse optimization methods for long-term loop closure detection in visual SLAM. *Autonomous Robots*, 42(7):1323–1335, April 2018. 7

[50] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *Proc. of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014. 62

[51] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor

environments. *International Journal of Robotics Research*, 31(5):647–663, February 2012. 13, 14

[52] Kin Leong Ho and Paul Newman. Loop closure detection in SLAM by combining visual and spatial appearance. *Robotics and Autonomous Systems*, 54(9):740–749, 2006. 6

[53] Kin Leong Ho and Paul Newman. Detecting loop closure with scene sequences. *International Journal of Computer Vision*, 74(3):261–286, 2007. 7

[54] S. Hong, T. Kim, and J. Kim. Underwater visual SLAM with loop-closure using image-to-image link recovery. In *Proc. of the OCEANS 2015 - Genova*, pages 1–6, May 2015. 7

[55] Y. Hou, H. Zhang, and S. Zhou. Convolutional neural network-based image representation for visual loop closure detection. In *Proc. of the IEEE Int. Conf. Information and Automation*, pages 2238–2245, August 2015. 7

[56] Binh-Son Hua, Quang-Hieu Pham, Duc Thanh Nguyen, Minh-Khoi Tran, Lap-Fai Yu, and Sai-Kit Yeung. SceneNN: A scene meshes dataset with aNNotations. In *Proc. of the 2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, October 2016. 61

[57] Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew Davison, and Andrew Fitzgibbon. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera. In *Proc. of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM. 3, 13

[58] Bing Jian and B. C. Vemuri. Robust Point Set Registration Using Gaussian Mixture Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1633–1645, August 2011. 17

[59] Y. Kanazawa and H. Kawakami. Detection of Planar Regions with Uncalibrated Stereo using Distributions of Feature Points. In *Procedings of the British Machine Vision Conference 2004*. British Machine Vision Association, 2004. 20

[60] T. Kanji. Multi-model hypothesize-and-verify approach for incremental loop closure verification. In *Proc. of the IEEE/SICE Int. Symp. System Integration (SII)*, pages 762–767, December 2016. 7

[61] Maik Keller, Damien Lefloch, Martin Lambers, Shahram Izadi, Tim Weyrich, and Andreas Kolb. Real-Time 3D Reconstruction in Dynamic Scenes Using Point-Based Fusion. In *Proc. of the 2013 International Conference on 3D Vision*. IEEE, June 2013. 39

[62] Harsha Kikkeri, Gershon Parent, Mihai Jalobeanu, and Stan Birchfield. An inexpensive method for evaluating the localization performance of a mobile robot navigation system. In *Proc. of the 2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2014. 61

[63] Bernd Kitt, Andreas Geiger, and Henning Lategahn. Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. In *Proc. of the 2010 IEEE Intelligent Vehicles Symposium*. IEEE, June 2010. 20

[64] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *Proc. of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*. IEEE, November 2007. 70

[65] Simon Korman and Roee Litman. Latent RANSAC. In *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 21

[66] Rainer Kummerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. G2o: A general framework for graph optimization. In *Proc. of the 2011 IEEE International Conference on Robotics and Automation*. IEEE, May 2011. 70

[67] Mathieu Labbe and Francois Michaud. Appearance-based loop closure detection for online large-scale and long-term operation. *IEEE Transactions on Robotics*, 29(3):734–745, 2013. 7

[68] Yasir Latif, Guoquan Huang, John J. Leonard, and José Neira. An online sparsity-cognizant loop-closure algorithm for visual navigation. In Dieter Fox, Lydia E. Kavraki, and Hanna Kurniawati, editors, *Proc. of the Robotics: Science and Systems X, University of California, Berkeley, USA, July 12-16, 2014*, 2014. 7

[69] G. H. Lee and M. Pollefeys. Unsupervised learning of threshold for geometric verification in visual-based loop-closure. In *Proc. of the IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 1510–1516, May 2014. 7

[70] Gim Hee Lee, Friedrich Fraundorfer, and Marc Pollefeys. Robust pose-graph loop-closures with expectation-maximization. In *Proc. of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, November 2013. 72

[71] Xiangru Li and Zhanyi Hu. Rejecting Mismatches by Correspondence Function. *International Journal of Computer Vision*, 89(1):1–17, January 2010. 20

[72] Huai-Jen Liang, Nitin J. Sanket, Cornelia Fermuller, and Yiannis Aloimonos. SalientDSO: Bringing Attention to Direct Sparse Odometry. *IEEE Transactions on Automation Science and Engineering*, 16(4):1619–1626, October 2019. 70

[73] Haomin Liu, Mingyu Chen, Guofeng Zhang, Hujun Bao, and Yingze Bao. ICE-BA: Incremental, consistent and efficient bundle adjustment for visual-inertial SLAM. In *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 70

[74] Hong Liu, Chenyang Zhao, Weipeng Huang, and Wei Shi. An End-to-End Siamese Convolutional Neural Network for Loop Closure Detection in Visual SLAM System. In *Proc. of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018. 7, 8

[75] David G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. *International Journal of Computer Vision*, 60(2):91–110, November 2004. 6, 14

[76] A. Maldonado-Ramírez and L. A. Torres-Méndez. A discrete Bayes filter for visual loop-closing in image sequences of coral reef explorations taken by humans and AUVs. In *Proc. of the OCEANS 2017 - Anchorage*, pages 1–5, September 2017. 7

[77] John McCormac, Ankur Handa, Andrew Davison, and Stefan Leutenegger. SemanticFusion: Dense 3D semantic mapping with convolutional neural networks. In *Proc. of the 2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2017. 83

[78] John McCormac, Ankur Handa, Stefan Leutenegger, and Andrew J. Davison. SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-training on Indoor Segmentation? In *Proc. of the 2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE, October 2017. 61

[79] M. Montemerlo and S. Thrun. Simultaneous localization and mapping with unknown data association using FastSLAM. In *Proc. of the 2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 2, pages 1985–1991, September 2003. 1

[80] Raul Mur-Artal and Juan D. Tardos. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Transactions on Robotics*, 33(5):1255–1262, October 2017. 1, 2, 39, 46, 58, 61, 62, 65, 70, 71, 73

[81] D. R. Myatt, P. H. S. Torr, S. J. Nasuto, J. M. Bishop, and R. Craddock. NAPSAC: High Noise, High Dimensional Robust Estimation - it's in the Bag. In *Procedings of the British Machine Vision Conference 2002*. British Machine Vision Association, 2002. 21

[82] Yoshikatsu Nakajima, Keisuke Tateno, Federico Tombari, and Hideo Saito. Fast and Accurate Semantic Mapping through Geometric-based Incremental Segmentation. In *Proc. of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, October 2018. 83

[83] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *Proc. of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, October 2011. 2, 70

[84] Kai Ni, Hailin Jin, and Frank Dellaert. GroupSAC: Efficient consensus in the presence of groupings. In *Proc. of the 2009 IEEE 12th International Conference on Computer Vision*. IEEE, September 2009. 21

[85] T. Nicosevici and R. Garcia. Automatic visual bag-of-words for online robot navigation and mapping. *IEEE Transactions on Robotics*, 28(4):886–898, August 2012. 7

[86] Peter C. Niedfeldt and Randal W. Beard. Recursive RANSAC: Multiple Signal Estimation with Outliers. *IFAC Proceedings Volumes*, 46(23):430–435, 2013. 20

[87] Matthias Nießner, Angela Dai, and Matthew Fisher. Combining Inertial Navigation and ICP for Real-time 3D Surface Reconstruction. In *Proc. of the Eurographics (Short Papers)*, pages 13–16. Citeseer, 2014. 70

[88] Haoyu Niu, Jiamin Wei, and YangQuan Chen. Optimal Randomness for Stochastic Configuration Network (SCN) with Heavy-Tailed Distributions. *Entropy*, 23(1):56, December 2020. 24

[89] J. H. Oh, J. D. Jeon, and B. H. Lee. Place recognition for visual loop-closures using similarities of object graphs. *Electronics Letters*, 51(1):44–46, 2015. 7

[90] Edwin Olson and Michael Kaess. Evaluating the performance of map optimization algorithms. In *Proc. of the RSS Workshop on Good Experimental Methodology in Robotics*, volume 15, 2009. 62

[91] M. K. Pargi, B. Setiawan, and Y. Kazama. Classification of different vehicles in traffic using RGB and Depth images: A Fast RCNN Approach. In *Proc. of the 2019 IEEE International Conference on Imaging Systems and Techniques (IST)*, pages 1–6, 2019. 84

[92] Hanspeter Pfister, Matthias Zwicker, Jeroen van Baar, and Markus Gross. Surfels: Surface Elements as Rendering Primitives. In *Proc. of the 27th annual conference on Computer graphics and interactive techniques - SIGGRAPH '00*. ACM Press, 2000. 3, 13, 75

[93] Sai Manoj Prakhya, Liu Bingbing, Yan Rui, and Weisi Lin. A closed-form estimate of 3D ICP covariance. In *Proc. of the 2015 14th IAPR International Conference on Machine Vision Applications (MVA)*. IEEE, May 2015. 17

[94] H. Qin, M. Huang, J. Cao, and X. Zhang. Loop closure detection in SLAM by combining visual CNN features and submaps. In *Proc. of the Automation and Robotics (ICCAR) 2018 4th Int. Conf. Control*, pages 426–430, April 2018. 7

[95] Tong Qin, Tongqing Chen, Yilun Chen, and Qing Su. AVP-SLAM: Semantic visual mapping and localization for autonomous vehicles in the parking lot. In *Proc. of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, October 2020. 1, 83

[96] Karima Rebai, Ouahiba Azouaoui, and Nouara Achour. Fuzzy ART-based place recognition for visual loop closure detection. *Biological Cybernetics*, 107(2):247–259, 2013. 7

[97] John G. Rogers, Jason M. Gregory, Jonathan Fink, and Ethan Stump. Test your SLAM! the SubT-tunnel dataset and metric for mapping. In *Proc. of the 2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2020. 61

[98] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *Proc. of the 2011 International Conference on Computer Vision*, pages 2564–2571. IEEE, November 2011. 14

[99] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pages 145–152, 2001. 13

[100] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. Fast Point Feature Histograms (FPFH) for 3D registration. In *Proc. of the 2009 IEEE International Conference on Robotics and Automation*. IEEE, May 2009. 5, 17

[101] S. M. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms. In *Proc. of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 1 (CVPR'06)*. IEEE, 2006. 4

[102] S. Shi, X. Wang, and H. Li. PointRCNN: 3D Object Proposal Generation and Detection From Point Cloud. In *Proc. of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–779, 2019. 84

[103] Shuran Song, Fisher Yu, Andy Zeng, Angel X. Chang, Manolis Savva, and Thomas Funkhouser. Semantic scene completion from a single depth image. In *Proc. of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017. 61

[104] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *Proc. of*

the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. IEEE, October 2012. 46, 58, 62, 65

[105] Niko Sunderhauf and Peter Protzel. Switchable constraints for robust pose graph SLAM. In *Proc. of 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, October 2012. 41, 72

[106] Niko Sunderhauf and Peter Protzel. Switchable constraints vs. max-mixture models vs. RRR - A comparison of three approaches to robust pose graph SLAM. In *Proc. of the 2013 IEEE International Conference on Robotics and Automation*. IEEE, May 2013. 72

[107] Takafumi Taketomi, Hideaki Uchiyama, and Sei Ikeda. Visual SLAM algorithms: a survey from 2010 to 2016. *IPSJ Transactions on Computer Vision and Applications*, 9(1), June 2017. 1

[108] Keisuke Tateno, Federico Tombari, Iro Laina, and Nassir Navab. CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction. In *Proc. of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, July 2017. 1

[109] P. H. S. Torr and A. Zisserman. MLESAC: A New Robust Estimator with Application to Estimating Image Geometry. *Computer Vision and Image Understanding*, 78(1):138–156, April 2000. 21

[110] Vladyslav Usenko, Jakob Engel, Jorg Stuckler, and Daniel Cremers. Direct visual-inertial odometry with stereo cameras. In *Proc. of the 2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, May 2016. 70

[111] Junqiu Wang, R. Cipolla, and Hongbin Zha. Vision-based global localization using a visual vocabulary. In *Proc. of the 2005 IEEE Int. Conf. Robotics and Automation*, pages 4230–4235, April 2005. 6

[112] Xinshuo Weng, Jianren Wang, David Held, and Kris Kitani. 3D Multi-Object Tracking: A Baseline and New Evaluation Metrics. In *Proc. of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10359–10366, October 2020. 85

[113] T. Whelan, M. Kaess, M. F. Fallon, H. Johannsson, J. J. Leonard, and J. B. McDonald. Kintinuous: Spatially Extended KinectFusion. In *Proc. of the RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, Sydney, Australia, July 2012. 2, 3, 4

[114] Thomas Whelan, Stefan Leutenegger, Renato Salas Moreno, Ben Glocker, and Andrew Davison. ElasticFusion: Dense SLAM Without A Pose Graph. In *Proc. of the Robotics: Science and Systems XI*. Robotics: Science and Systems Foundation, July 2015. 1, 2, 3, 13, 14, 19, 39, 62, 65, 70, 71

[115] B. Williams, G. Klein, and I. Reid. Automatic relocalization and loop closing for real-time monocular SLAM. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1699–1712, September 2011. 6

[116] J. Wu, H. Zhang, and Y. Guan. An efficient visual loop closure detection method in a map of 20 million key locations. In *Proc. of the 2014 IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 861–866, May 2014. 7

[117] Y. Xia, J. Li, L. Qi, and H. Fan. Loop closure detection for visual SLAM using pcanet features. In *Proc. of the Int. Joint Conf. Neural Networks (IJCNN)*, pages 2274–2281, July 2016. 7

[118] Y. Xia, J. Li, L. Qi, H. Yu, and J. Dong. An evaluation of deep learning in loop closure detection for visual SLAM. In *Proc. of the Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData) 2017 IEEE Int. Conf. Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber*, pages 85–91, June 2017. 7

[119] Jianxiong Xiao, Andrew Owens, and Antonio Torralba. SUN3D: A Database of Big Spaces Reconstructed Using SfM and Object Labels. In *Proc. of the 2013 IEEE International Conference on Computer Vision*. IEEE, December 2013. 4, 13, 14, 46, 58, 72, 76, 77

[120] Jiaolong Yang, Hongdong Li, and Yunde Jia. Go-ICP: Solving 3D Registration Efficiently and Globally Optimally. In *Proc. of the 2013 IEEE International Conference on Computer Vision*. IEEE, December 2013. 17

[121] H. Yue and W. Chen. Comments on automatic visual bag-of-words for online robot navigation and mapping. *IEEE Transactions on Robotics*, 31(1):223–224, February 2015. 7

[122] G. Zhang, M. J. Lilly, and P. A. Vela. Learning binary features online from motion dynamics for incremental loop-closure detection and place recognition. In *Proc. of the 2016 IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 765–772, May 2016. 7

[123] Guoxiang Zhang and YangQuan Chen. A metric for evaluating 3D reconstruction and mapping performance with no ground truthing. arXiv:2101.10402. 71

[124] Guoxiang Zhang and YangQuan Chen. LoopSmart: Smart Visual SLAM Through Surface Loop Closure, January 2018. arXiv:1801.01572. 20, 61, 62

[125] Guoxiang Zhang, YangQuan Chen, and Holley Moyes. Optimal 3D Reconstruction of Caves Using Small Unmanned Aerial Systems and RGB-D Cameras. In *Proc. of the 2018 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 410–415. IEEE, June 2018. 20

[126] Guoxiang Zhang, Bo Shang, YangQuan Chen, and Holley Moyes. SmartCave-Drone: 3D cave mapping using UAVs as robotic co-archaeologists. In *Proc. of 2017 International Conference on Unmanned Aircraft Systems (ICUAS)*. IEEE, June 2017. viii, 4, 5, 37

[127] X. Zhang, Y. Su, and X. Zhu. Loop closure detection for visual SLAM systems using convolutional neural network. In *Proc. of the 2017 23rd International Conference on Automation and Computing (ICAC)*, pages 1–6. IEEE, September 2017. 7

[128] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Fast Global Registration. In *Proc. of the Computer Vision – ECCV 2016*, pages 766–782. Springer International Publishing, 2016. 4, 13, 17

[129] Z. Zhou, M. Wang, X. Chen, W. Liang, and J. Zhang. Box Detection and Positioning based on Mask R-CNN for Container Unloading. In *Proc. of the 2019 IEEE 4th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, volume 1, pages 171–174, 2019. 84

[130] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-Based Visual Inertial Odometry. In *Proc. of The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 70

[131] Jon Zubizarreta, Iker Aguinaga, and Jose Maria Martinez Montiel. Direct Sparse Mapping. *IEEE Transactions on Robotics*, 36(4):1363–1370, August 2020. 70

Appendix A

List of Abbreviations

AAOD aligned absolute object distance

AOD absolute object distance

ATE absolute trajectory error

BA bundle adjustment

BnB branch-and-bound

BoW bag-of-words

BRIEF binary robust independent elementary features

CNN convolutional neural networks

CZK Choi, Zhou, and Koltun

DAOD direct absolute object distance

DMP dense map posterior

FAST features from accelerated segment test

FO-RANSAC fractional order RANSAC

FPFH fast point feature histograms

GPS global positioning system

GPU graphics processing unit

ICP iterative closest point

IMU inertial measurement unit

IoU intersection over union

LiDAR light detection and ranging

MAP maximum a posteriori

MOT multi-object tracking

MVS multi-view stereo

NN nearest neighbor

ORB oriented FAST and rotated BRIEF

PDF probability density function

PnP perspective-n-point

PRVO precision and recall of virtual observations

RANSAC random sample consensus

RGB-D RGB-Depth

RMSE root mean square error

RPE relative position error

RPN region proposal network

RTK-GPS real-time kinematic GPS

SfM structure from motion

SIFT scale-invariant feature transform

SLAM simultaneous localization and mapping

SMD surface mean distance

TSDF truncated signed distance function

UAV unmanned aerial vehicle

vSLAM visual simultaneous localization and mapping

Appendix B

Code and Data

Code and data will be made available here `https://gzhang8.github.io/3DMapping/`.