OCTREE-BASED APPROACH FOR REAL-TIME 3D INDOOR MAPPING USING RGB-D VIDEO DATA

J. Hou [1], M. Goebel [1], P. Hübner [1], D. Iwaszczuk [1]

[1] Technical University of Darmstadt, Dept. of Civil and Environmental Engineering Sciences,
Remote Sensing and Image Analysis, Darmstadt, Germany
(jiwei.hou, mona.goebel, patrick.huebner, dorota.iwaszczuk)@tu-darmstadt.de

KEY WORDS: 3D Indoor Mapping, Real-time, Visual SLAM, Octree, RGB-D Camera.

ABSTRACT:

3D indoor mapping is becoming increasingly critical for a variety of applications such as path planning and navigation for robots. In recent years, there is a growing interest in how low-cost sensors, such as monocular or depth cameras, can be used for 3D mapping. In our paper, we present an octree-based approach for real-time 3D indoor mapping using a handheld RGB depth camera. One benefit of the generated octree map is that it requires less storage and computational resources than point cloud models. Moreover, it explicitly represents free space and unmapped areas, which are essential for the robot's navigation tasks. In this work, on the basis of the ORB-SLAM3 system (Campos et al., 2021), we developed an octree mapping system, which directly calls the keyframes and estimated poses provided by ORB-SLAM3 algorithms. Furthermore, we used point cloud library (PCL) for the dense point cloud mapping and then OctoMap for the point cloud to octree map conversion. Finally, we implemented an efficient probabilistic 3D mapping in the robot operating system (ROS) environment. We used the TUM RGB-D dataset to evaluate the estimated trajectories of the camera. The evaluation shows an average translational RMSE of 5.9 cm on the TUM RGB-D dataset. Besides, we also compared the ground truth point clouds and our generated point clouds. The result shows the mean cloud-to-cloud distance in the corridor scene is about 6 cm. All the evaluation results show our proposed approach is a promising solution for advanced indoor voxel mapping and robotic navigation systems.

1. INTRODUCTION

With the increasing demand for robotics and autonomous systems, 3D indoor mapping is becoming more critical for various applications such as robot indoor navigation, building inspection, augmented reality (AR), and virtual reality (VR). Especially in complex and dynamic environments, to quickly generate and maintain accurate 3D maps with low-cost sensors. Due to the low cost and the intuitive approach to create a 3D map, many visual simultaneous localization and mapping (vSLAM) systems have been published in the past decades, such as LSD-SLAM (Engel et al., 2013), RGB-D SLAM (Endres et al., 2014) and ORB-SLAM3 (Campos et al., 2021).

This paper proposes an octree-based approach for real-time 3D indoor mapping using RGB-D video data. The proposed approach extends ORB-SLAM3 (Campos et al., 2021) algorithm with RGB-D video data captured by Intel RealSense D455 camera to generate an octree-based voxel map in real-time. It utilizes an octree structure to divide the indoor space into smaller regions, allowing for efficient processing of data and reduced computational cost.

The main contribution of this paper is that we extend the state-of-the-art ORB-SLAM3 (Campos et al., 2021) algorithm to octree-based voxel mapping. We utilize an octree structure for spatial partitioning of 3D point clouds, enabling a real-time 3D indoor voxel mapping in a robot operating system (ROS). Our extended system can smoothly run with an Intel RealSense D455 camera. We evaluate the extended system using TUM RGB-D benchmark dataset (Sturm et al., 2012) and a set of ground truth point clouds scanned by an industrial laser mobile mapping system.

The experimental results demonstrate that the proposed system is effective in generating the octree map and occupancy grid map required for robot navigation in real time. This work provides a promising solution for indoor environment mapping tasks that require real-time performance and lightweight 3D mapping, which could be further applied in robotic indoor autonomous navigation, AR and VR applications.

This paper consists of three main parts:

1. We extend the ORB-SLAM3 (Campos et al., 2021) algorithm for real-time 3D indoor mapping using an octree-based approach.

2. We apply the proposed system and evaluate its accuracy and efficiency in real indoor environments with varying levels of complexity and clutter.

3. We compare the performance of the proposed system with other advanced methods for 3D indoor mapping, in terms of accuracy and efficiency.

This paper is organized as follows. Section 2 gives a brief review of the main algorithms in the field of RGB-D based SLAM. Section 3 describes the proposed octree-based indoor mapping system and its three main components. The evaluation of the proposed approach using datasets with ground truth is presented in Section 4, along with a discussion of the obtained results. Section 5 makes a conclusion of this work and provides an outlook on future research directions.

2. RELATED WORK

In the following, we review the literature related to RGB-D based SLAM systems, and highlight the main milestones achieved in the field of RGB-D SLAM.

The RGB-D camera-based SLAM algorithm was first proposed by (Henry et al., 2010). This paper explores the potential applications of such cameras in the field of robotics, particularly for the purpose of constructing dense 3D maps of indoor environments. The proposed algorithm detects features using the scale invariant feature transform (SIFT) (Lowe, 2004) method and extracts descriptors from two adjacent RGB frames. Depth information is then added to generate 3D-3D feature point pairs information. The random sample consensus (RANSAC) (Fischler and Bolles, 1981) method is utilized to align the 3D-3D matched point pairs and to derive the corresponding transformation matrix. The motion transformation is subsequently optimized using the iterative closest point (ICP) method (Besl and McKay, 1992). To achieve global optimization of the 3D map, the tree-based network optimizer (TORO) algorithm (Grisetti et al., 2008) is employed at the back end. Finally, the view-based loop closure detection is added to the algorithm to obtain globally consistent 3D maps.

To overcome the slow processing speed of the SIFT method for feature extraction, an improved algorithm was proposed by (Henry et al., 2012). Instead of SIFT, the features from accelerated segment test (FAST) (Bay et al., 2006) method based on accelerated segments is used. Additionally, the sparse beam adjustment (SBA) method (Triggs et al., 1999), with better performance, is used for global optimization instead of the TORO algorithm (Grisetti et al., 2008). Furthermore, scene identification is used to improve the efficiency of loop closure detection.

A handheld RGB-D SLAM system was developed by (Engelhard et al., 2011), from the Department of Computer Science at the University of Freiburg , utilizing a Kinect camera as a sensor. The feature detection and extraction of feature descriptors from RGB images are conducted at the front end of the SLAM algorithm using the speeded up robust features (SURF) algorithm (Bay et al., 2008). Feature matching is carried out between adjacent RGB frames, and the 3D spatial coordinates of matched feature points are calculated using depth information from the depth image. For motion estimation and optimization, the RANSAC method is used to estimate the motion between two frames, and a modified ICP method (Segal et al., 2010) is used to optimize the motion transfer matrix of the camera. The pose graph solver (Grisetti et al., 2010) is used to globally optimize the motion transfer matrix, resulting in the optimal global pose. The output of the system is a globally consistent 3D model of the perceived surrounding, represented as a colored point cloud.

Later, (Endres et al., 2012) developed a new RGB-D SLAM system. In the new system, they proposed to use the open-source general graph optimization (g2o) library (Kümmerle et al., 2011) for global pose graph optimization to generate a global dense 3D model of the environment. Finally, the point cloud was converted into a voxel representation through OctoMap, an octree-based 3D mapping Framework. This resulted a volumetric 3D map of the environment, which can be utilized for robot localization, path planning, and navigation. In our research, we will compare with this RGB-D SLAM system in the context of trajectories estimation.

To evaluate RGB-D SLAM algorithms proposed by researchers from various institutions and universities around the world, researchers from the Technical University of Munich and the University of Freiburg produced a standard RGB-D SLAM dataset (Sturm et al., 2012). They used a motion capture system to record the accurate and time-synchronized poses data of the Microsoft Kinect depth camera. These data are used as ground truth and can be compared with the poses estimated by the RGB-D SLAM algorithms. The image sequences from the Microsoft Kinect depth camera contain both color and depth images at full sensor resolution (640 × 480) with a video frame rate (30 Hz). Finally, a total of 39 sequences were recorded across an office environment and an industrial hall to create the dataset. The dataset provides a diverse range of scenes and camera motions for analysis. The dataset has gained significant popularity and is commonly utilized for the evaluation of RGB-D SLAM systems. In our research, we also use this dataset for system performance evaluation.

KinectFusion (Newcombe et al., 2011) is the first SLAM algorithm based on the Microsoft Kinect depth camera, which is capable of accurate real-time mapping of complex and arbitrary indoor scenes under different lighting conditions. Especially, it allows real-time reconstruction of dense surfaces. KinectFusion system is made up of four components, including surface measurement, surface reconstruction update, surface prediction and sensor pose estimation. Surface measurement as a pre-processing stage to generate a dense vertex map and normal map pyramid. In their paper, KinectFusion relies on the Kinect camera to calculate the camera's poses and generate a 3D map of the environment, but the algorithm is not Kinect specific and can be applied to any RGB-D camera, including Intel RealSense depth camera.

While the original KinectFusion is restricted to small-scale scenes, there are meanwhile extensions for large-scale. (Niesner et al., 2013) contributed an online system for large and fine scale volumetric reconstruction, their system extends a simple spatial hashing technique (Teschner et al., 2003) that compresses space and allows real-time access and update of implicit surface data. Surface data is only densely stored in cells where measurements are observed. In addition, surface data can be efficiently flowed into and out of the hash table, enabling further scalability during sensor motion. However, their approach is designed to be efficient for parallel graphics processing unit (GPU) hardware.

ORB-SLAM (Mur-Artal et al., 2015) employs Oriented FAST and Rotated BRIEF (ORB) (Rublee et al., 2011) algorithm for feature detection and matching, sparse map creation, and loop closure detection. ORB-SLAM algorithm is lightweight and can therefore be run on a standard central processing unit (CPU) host. From a monocular SLAM system by incorporating support for stereo and RGB-D cameras (Mur-Artal and Tardos, 2017), ORB-based SLAM systems have been continuously updated and improved in the past few years, with the release of ORB-SLAM3 (Campos et al., 2021), ORB-SLAM3 has become one of the best performing feature-based SLAM systems that operates in real time, both indoors and outdoors. In our previous research (Hou et al., 2023), we have extended ORB-SLAM3 algorithm and implemented real-time tracking and 3D dense reconstruction using an Intel RealSense D455 camera. Based on our previous 3D dense mapping work, we further extend ORB-SLAM3 algorithm to an octree-based 3D mapping system in this paper, which can generate 3D voxel map of the environment in real time.

3. METHODOLOGY

As shown in Figure 1, the processing pipeline of our system consists of three main components: 1) ORB-SLAM3 based trajectory estimation for RGB-D cameras. 2) 3D dense mapping based on RGB-D data and poses of camera. 3) Octree-based indoor mapping with ROS in real-time. In the following sections, we will describe the three main parts of our processing pipeline in more detail.

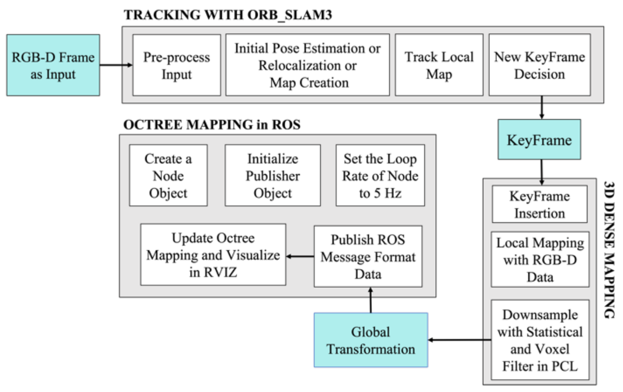
TRACKING WITH ORB_SLAM3
RGB-D Frame as Input
Pre-process Input
Initial Pose Estimation or Relocalization or Map Creation
Track Local Map
New KeyFrame Decision
KeyFrame
OCTREE MAPPING in ROS
Create a Node Object
Initialize Publisher Object
Set the Loop Rate of Node to 5 Hz
Update Octree Mapping and Visualize in RVIZ
Publish ROS Message Format Data
Global Transformation
3D DENSE MAPPING
KeyFrame Insertion
Local Mapping with RGB-D Data
Downsample with Statistical and Voxel Filter in PCL

Figure 1. Pipeline of our real-time 3D indoor octree mapping system, extended from ORB-SLAM3 (Campos et al., 2021).

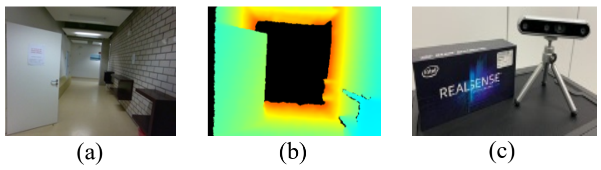

Figure 2. Example of an RGB-D image sequence from input data. (a) RGB image. (b) Depth image with pseudo color. captured by Intel RealSense Depth Camera D455 (c).

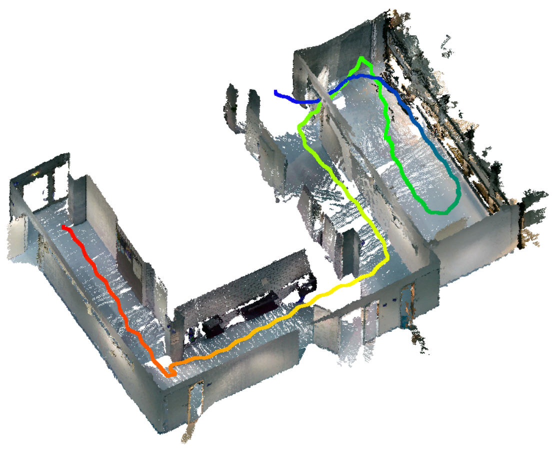

Figure 3. The estimated trajectory of Intel RealSense Depth Camera D455, it starts at blue and ends at red.

3.1 ORB-SLAM3 based trajectory estimation for RGB-D cameras

ORB-SLAM3 (Campos et al., 2021) is an accurate vSLAM algorithm that supports visual, visual–inertial, and multimap SLAM with monocular, stereo and RGB-D cameras, using pin-hole and fisheye lens models. One major feature of ORB-SLAM3 is visual-inertial SLAM, it relies entirely on maximum a posteriori (MAP) estimation, enabling robust real-time operation in a wide range of indoor and outdoor environments. In addition, ORB-SLAM3 is also characterised by its multiple map system, which provides excellent positioning accuracy, loop closure detection and relocalization capability.

In our research, for RGB-D camera tracking and pose estimation, we rely on ORB-SLAM3 as it offers a good trade-off between speed and accuracy. ORB-SLAM3 system includes four main components: 1) Feature extraction and pose estimation. 2) Local mapping and local bundle adjustment. 3) Loop closure detection and map merging, followed by full bundle adjustment. 4) Altas is used to generate a unique DBoW2 (Gálvez-López and Tardós, 2012) database of keyframes for relocalization, loop closing, and map merging. By performing these individual tasks in parallel threads, ORB-SLAM3 can achieve a globally consistent, long-term tracking capability while maintaining a lightweight profile that can be run on a standard CPU.

In our pipeline, we mainly use ORB-SLAM3 for estimating camera motion and processing keyframes derived from RGB-D videos. ORB-SLAM3 supports two different running modes, namely normal mode and ROS mode. The former involves using an RGB-D camera or RGB-D image dataset as input, while the latter involves using the ROS to process the input data obtained from an RGB-D camera. In this paper, we will conduct the octree-based mapping in ROS mode.

3.2 3D dense mapping based on RGB-D data and poses of camera

In the second stage of our pipeline, we perform a dense point cloud mapping based on RGB-D data and estimated camera poses. Here, we mainly use the powerful cross-platform open source point cloud library (PCL) proposed by (Rusu and Cousins, 2011) to generate global point clouds. PCL implements a large number of common algorithms related to point clouds, such as acquisition, filtering, segmentation and surface reconstruction.

In the first stage of the point cloud generation, we create a single thread to extend the ORB-SLAM algorithm, and then use a multi-step process involving various algorithms. First, we invoke keyframes from ORB-SLAM3 and RGB-D information to generate local point clouds for each session. Finally, we transform these point clouds into a global map based on the estimated camera poses in keyframes.

To improve the quality and accuracy of the point cloud mapping, we use a double-layer filtering method. In the first layer, we perform statistical filtering in the local map to remove outlier points. This filtering method identifies points that are too far from the mean of the local patches and removes them. This helps to improve the accuracy of the reconstructed 3D model by eliminating points that are unlikely to be part of the actual scene. In the second layer of filtering, we use voxel filtering in the point clouds to downsample the number of points in the point cloud. This filtering method reduces memory usage without causing significant distortion of the shape features of the 3D model. The voxel filtering algorithm divides the 3D space into voxels, keeping only one point per voxel. This reduces the number of points while retaining the general shape of the object. In this task, we set the voxel resolution to 0.01 m for downsampling.

By using this multi-step approach with a double-layer filtering method, we have successfully carried out real-time 3D dense point cloud mapping with the Intel RealSense D455 camera.

3.3 Octree-based indoor mapping with robot operating system (ROS) in real-time

For the octree-based indoor mapping, we adopt the work of (Hornung et al., 2013) also known as OctoMap, which is an octree-based framework for efficient probabilistic 3D mapping. In addition, the real-time display of the octomap is achieved by means of the ROS visualization (RVIZ). ROS is an open source software framework widely used in robotics research and

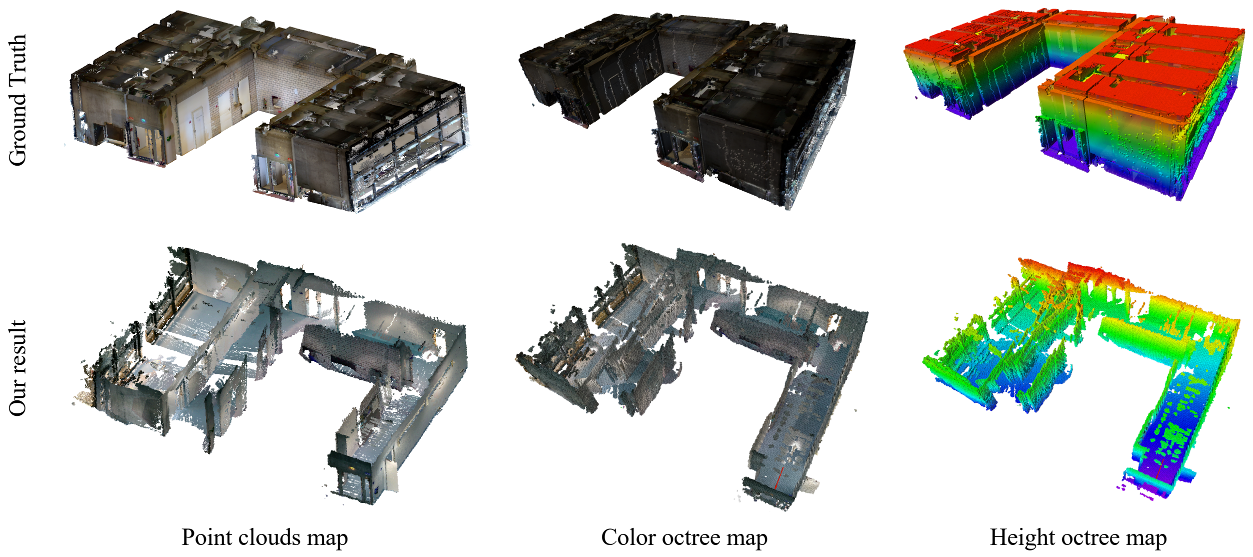

Figure 4. Result of real-time 3D indoor octree mapping using RGB-D video. The ground truth data is scanned by an industrial NavVis VLX laser mobile mapping system.

development. It provides a set of tools and libraries for building robot applications, such as communication mechanisms, and algorithms for perception, planning, and control.

The input data for this algorithm is the voxel filtered point clouds, generated during the 3D dense point clouds mapping process. The objective is to generate a 3D model of the indoor environment in real-time using an octree data structure. The algorithm can be split into the following steps:

1) Create a node object to communicate with other nodes in the ROS environment. This object allows the current node to subscribe to topics, publish messages, and interact with the ROS parameter server.

2) After creating the node object, a publisher object is then created to advertise a ROS point cloud message “sensor_msgs/PointCloud2” to topic “cloud_in” as data input for octree mapping. The topic “cloud_in” should be the same as the topic in the octree server launch file. In addition, we empirically set the message queue size to 100,000, which determines how many messages can be cached if the subscriber cannot keep up with the publishing speed.

3) After the definition of a publisher object, a rate object with a frequency of 5 Hz is set, which is the loop rate of the node that was created in Step 1, this step ensures that the node runs at a consistent frequency, regardless of how quickly or slowly the loop computation completes. It is important for real-time applications and for maintaining efficient use of system resources.

4) Using a 3D affine transformation matrix provided by the Eigen library to transform the inputted point clouds. This transformation is useful for aligning point clouds or correcting orientation differences. In this paper, it is used to adjust the pose of the currently acquired global point cloud. Eigen is a C++ template library for linear algebra. It is esigned to be fast, efficient and easy to use for numerical computation, particularly linear algebraic operations.

5) After the affine transformation, we then use the method “toROSMsg” provided by PCL to convert the transformed global point clouds into a ROS point cloud message format. followed by setting the ROS frame ID. Using the publisher object created in step 2 to publish the ROS point cloud message format data to the topic “cloud_in”.

6) Launch the octree server and start the octree mapping. The topic "cloud_in" will be subscribed by RVIZ so that the generated octree map can be visualised in real time.

The above is the procedure of real-time octree mapping with the tool of OctoMap. In our research, the resolution of octree map is set as 0.05 m, the result is shown in Figure 4.

4. EVALUATION

To evaluate our system performance quantitatively. On the one hand, we use the TUM RGB-D benchmark dataset (Sturm et al., 2012) to evaluate the translational root mean square error (Transl. RSME) in the estimated camera trajectory by comparing it with the ground-truth, as shown in Table 1. On the other hand, we apply our octree mapping system with the self-collected data scanned by a handheld Intel RealSense D455 camera. Figure 2 shows our Intel RealSense D455 camera and example of an RGB-D image sequence from input data. As the octree map is built from point clouds, the quality of the generated 3D point clouds directly affects the quality of the octree mapping, we also use ground truth 3D point cloud data to evaluate the generated 3D point clouds. The ground truth 3D point cloud data is scanned by an industrial NavVis VLX laser mobile mapping system. We evaluate the results with the software CloudCompare. All experiments are conducted on a computer with Ubuntu18.04 operating system, AMD Epyc 7402p, 24-core processor, 256G RAM and NVIDIA RTX A4000 GPU.

In this section, we first present the sequences we selected from the benchmark dataset for the evaluation of the camera trajectory, as shown in Table 1. Then, we detail the evaluation process and results, followed by a discussion of the results of our system for octree-based 3D indoor mapping.

Sequence Name	Duration	Length	Avg. Transl Velocity	Avg. Angular Velocity	Transl. RMSE*	Transl. RMSE
fr1 xyz	30.09 s	7.11 m	0.24 m/s	8.92 deg/s	0.021 m	**0.010 m**
fr1 rpy	27.67 s	1.66 m	0.06 m/s	50.15 deg/s	0.042 m	**0.022 m**
fr1 360	28.69 s	5.82 m	0.21 m/s	41.60 deg/s	**0.103 m**	0.227 m
fr1 floor	49.87 s	12.57 m	0.26 m/s	15.07 deg/s	0.055 m	**0.043 m**
fr1 desk	23.40 s	9.26 m	0.41 m/s	23.33 deg/s	0.049 m	**0.017 m**
fr1 desk2	24.86 s	10.16 m	0.43 m/s	29.31 deg/s	0.102 m	**0.027 m**
fr1 room	48.90 s	15.99 m	0.33 m/s	29.88 deg/s	0.219 m	**0.072 m**
fr1 plant	41.53 s	14.80 m	0.37 m/s	27.89 deg/s	0.142 m	**0.019 m**
fr1 teddy	50.82 s	15.71m	0.32 m/s	21.32 deg/s	0.138 m	**0.096 m**

Table 1. The statistical information on fr1 sequences of TUM RGB-D dataset. Duration, Length, Avg. translational velocity, Avg. angular velocity from TUM RGB-D dataset official website. Transl. RMSE* is the evaluation of RGB-D SLAM from (Endres et al., 2012). Transl. RMSE is the evaluation of our octree-based 3D indoor mapping system.

4.1 Dataset used for evaluation

TUM RGB-D dataset: The TUM RGB-D dataset is a large-scale dataset primarily for texturing office (named as "fr1") and industrial hall (named as "fr2") scenes. The dataset includes both color and depth images taken by the Microsoft Kinect camera, as well as the ground truth trajectory of the camera. The data was recorded at full frame rate (30Hz) and sensor resolution (640x480). The ground truth trajectories were obtained from a high precision motion capture system using eight high speed tracking cameras (100 Hz). In addition, the dataset also provides an automated evaluation script for comparing the estimated trajectory with the ground truth trajectory.

In our research, we selected nine sequences of fr1 for evaluation. Among these selected sequences, as shown in Table 1, the sequences fr1/xyz and fr1/rpy are very simple and the result normally represents the best case. The rest of the sequences are more challenging as they cover a larger office space and more unstable camera movements.

4.2 Evaluation of the estimated trajectory

First, we evaluated our system on all fr1 sequences of TUM RGB-D dataset, see Table 1. However, because our octree mapping system is based on ORB-SLAM3 algorithm, it provides higher accuracy camera poses estimation, which will guarantee our system to achieve greater accuracy in the mapping process. For all sequences except the "fr1_360" sequence, the pose estimation results outperform RGB-D SLAM (Endres et al., 2012). On the simple "fr1_xyz" sequence, we obtained the best value of 1.0 cm Transl. RSME error. Even in the challenging "fr1_room" sequence, it was still possible to obtain a 7.2 cm Transl. RSME error. We achieved the worst 22.7 cm Transl. RSME in "fr1_360" sequence, because this sequence contains a fast 360 degree turn, and it has a relatively high average velocity in terms of both translational and angular, this sequence is more challenging for the extended ORB-SLAM3 system.

Overall, this evaluation shows that the extended ORB-SLAM3 system is reliable, which can provide relatively high pose estimation for our real-time octree-based mapping.

4.3 Evaluation of the mapping accuracy

To verify the accuracy of the 3D dense point clouds mapping results, we use the point cloud data scanned by a high precision NavVis VLX mobile mapping system as ground truth to conduct a comparison experiment. We evaluate the mapping accuracy using tools on CloudCompare. First, we align point clouds through point pair picking, and then make a fine registration with the iterative closest point (ICP) algorithm. Finally, we compare the cloud-to-cloud (C2C) distances between the ground truth point clouds and our generated point clouds. All parameter settings are selected as default parameters on CloudCompare. We use 0.5 m and 0.2 m C2C absolute distance filters to downweight the influence of outlier points caused by noise, respectively.

The evaluation result is shown in Figure 5 and Table 2, and the input RGB-D video stream details are shown in the Table 3. For the entire point cloud generated, the mean distance of computed distances is 7.4 cm, and the standard deviation of computed distances is 8.5 cm. We can see that in the left room scene, the distance error is large near the window without curtains. But inside the corridor, the accuracy of the obtained point cloud is higher. Then we filter out outliers with distances greater than 0.2 m and recalculate the cloud-to-cloud distances, the obtained mean distance of computed distances is 5.8 cm, and the standard deviation of computed distances is 4.2 cm. In general, this evaluation shows the accuracy of our indoor dense point clouds mapping is good, although the 3D indoor scenes are not as complete due to the camera's perspective. However, the mapping accuracy is also affected by the way of the camera movement and the light situation of the environment, these are classical problems for visual SLAM. Here, we only focus on the mapping with an ideal camera moving state, the direction of D455 camera movement is shown in Figure 3.

Filter by distance	Mean Distance	Standard Deviation
No filter	0.074 m	0.085 m
0.5 m	0.070 m	0.066 m
0.2 m	0.058 m	0.042 m

Table 2. The statistics of computed distances.

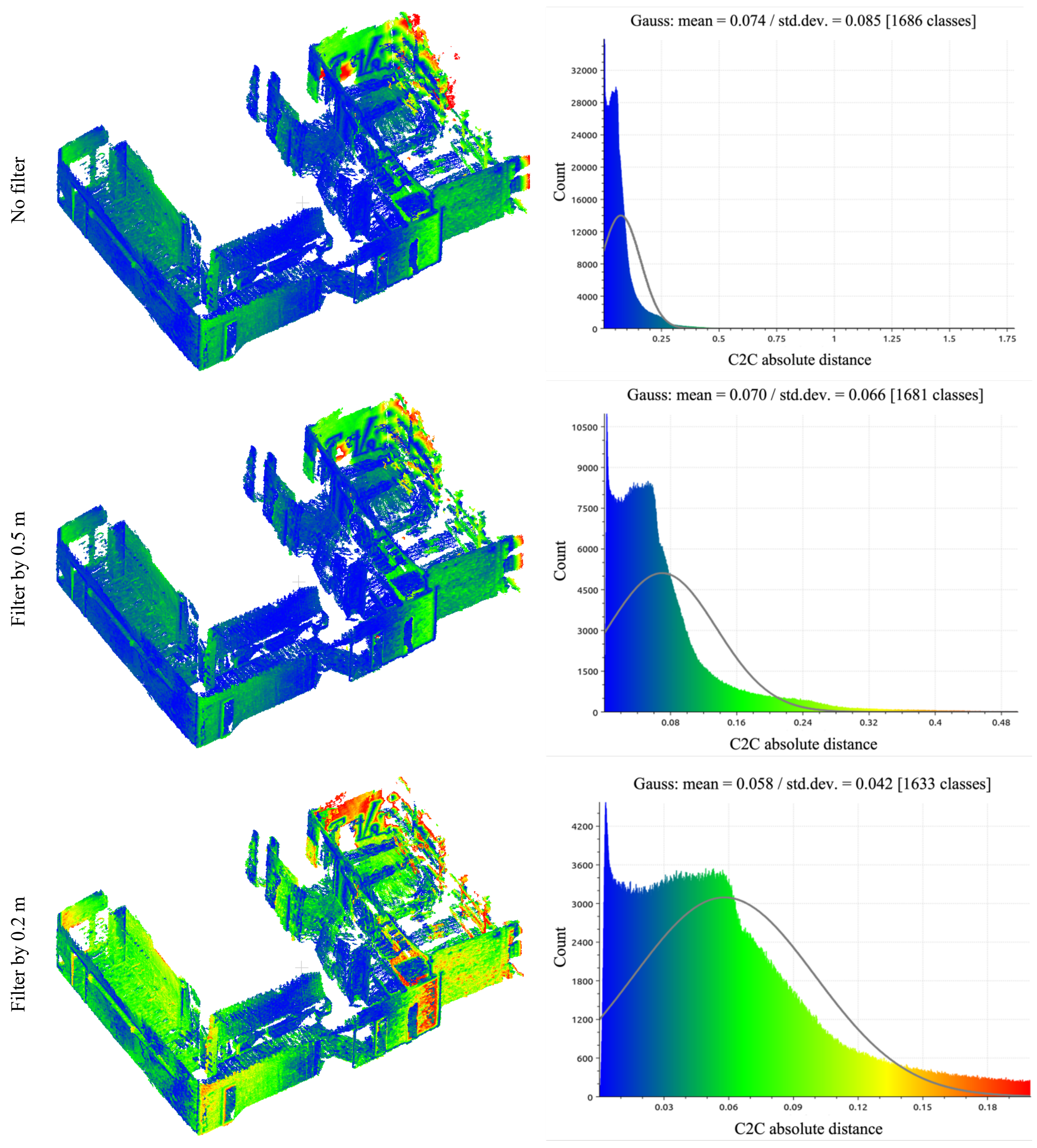

Figure 5. The comparison results of cloud-to-cloud distances between the ground truth point clouds and our generated point clouds. We use 0.5 m and 0.2 m C2C absolute distance filters to downweight the influence of outlier points caused by noise, respectively.

Data	Color point clouds	Color octree	Octree
Memory	46.6 M	6.7 M	288.6 kB
Compression rate	—	85.6%	99.4%

Table 4. The memory representation of color point clouds map in pcd format, color octree map in ot format and octree map in bt format. M represents megabyte and kB represents kilobytes, 1M = 1024 kB.

Sensor resolution	Frame rate (FPS)	Duration	Length
640 x 480	30Hz	153.8 s	35.28 m

Table 3. The details about the input RGB-D video. FPS: Frames per second.

4.4 The results of our octree-based 3D indoor mapping

Eventually, we succeeded in mapping an indoor octree model in real time with a handheld D455 camera, the result of our octree-based 3D indoor mapping is shown in Figure 4. As described in the methodology section, we set the default resolution of the octree map to 0.05 m. As can be seen in Table 4, the octree structure provides a compact memory representation for efficient storage of maps. The memory of the color point cloud data is reduced from 46.6M to 6.7M and 288.6kB, with compression rates of 85.6% and 99.4% respectively.

5. CONCLUSION & FUTURE WORK

We present an octree-based approach for real-time 3D indoor mapping using a handheld Intel RealSense D455 camera. In this work, we extend the ORB-SLAM3 (Campos et al., 2021) algorithm to build large-scale octree voxel maps on the fly. For our indoor mapping system, first, we invoke the keyframes and estimated poses from ORB-SLAM3. Then we apply point cloud library (PCL) to generate 3D dense point clouds. Finally, we use OctoMap, an octree-based mapping framework, to construct and update the octree map based on the published point clouds topic in the robot operating system (ROS) environment. The real-time octree mapping of indoor environments can be further used for robot path planning and navigation.

We evaluate our system on the TUM RGB-D dataset. The average translational RMSE on TUM RGB-D dataset is 5.9 cm. In addition, we use ground truth 3D point clouds to evaluate the accuracy of the dense point clouds generated by our system. However, the good accuracy of point clouds also ensures the high quality of the 3D octree mapping.

During the evaluation of our system, we have also discovered some aspects that we need to further improve and verify. For instance, we just utilized the rough camera trajectory from ORB-SLAM3 algorithm, without adding loop closure detection for our octree-based system. Moreover, the mapping results of 3D indoor scenes are not complete, and the point cloud processing still needs to be further improved. Therefore, in the future, we will mainly optimize our octree-based mapping system from the above aspects.

ACKNOWLEDGEMENT

This research is supported by China Scholarship Council (CSC), Grant/Award Number: 202108130064. We also thank Vivien Volland and Jakob Schmidt of Department of Geodetic Measuring Systems and Sensors at the Technical University of Darmstadt for providing their dataset (TU Darmstadt L501 room 44).

REFERENCES

Bay, H., Ess, A., Tuytelaars, T., Van Gool, L., 2008. Speeded-Up Robust Features (SURF). Comput. Vis. Image Underst. 110, 346–359. https://doi.org/10.1016/j.cviu.2007.09.014

Bay, H., Tuytelaars, T., Van Gool, L., 2006. Machine Learning for High-Speed Corner Detection 404–417.

Besl, P.J., McKay, N.D., 1992. A method for registration of 3-D shapes. EEE Trans. Pattern Anal. Mach. Intell. 14, 239–256.

Campos, C., Elvira, R., Rodriguez, J.J.G., Montiel, J.M.M., Tardos, J.D., 2021. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial, and Multimap SLAM. IEEE Trans. Robot. 37, 1874–1890. https://doi.org/10.1109/TRO.2021.3075644

Endres, F., Hess, J., Engelhard, N., Sturm, J., Cremers, D., Burgard, W., 2012. An evaluation of the RGB-D SLAM system. Proc. - IEEE Int. Conf. Robot. Autom. 3, 1691–1696. https://doi.org/10.1109/ICRA.2012.6225199

Endres, F., Hess, J., Sturm, J., Cremers, D., Burgard, W., 2014. 3-D Mapping with an RGB-D camera. IEEE Trans. Robot. 30, 177–187. https://doi.org/10.1109/TRO.2013.2279412

Engel, J., Sturm, J., Cremers, D., 2013. LSD-SLAM: Large-Scale Direct Monocular SLAM. Proc. IEEE Int. Conf. Comput. Vis. 1449–1456.

Engelhard, N., Endres, F., Hess, J., Sturm, J., Burgard, W., 2011. Real-time 3D visual SLAM with a hand-held RGB-D camera. Proc. RGB-D Work. 3D Percept. Robot. Eur. Robot. Forum, 2011.

Fischler, M.A., Bolles, R.C., 1981. Random sample consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Commun. ACM 24, 381–395. https://doi.org/10.1145/358669.358692

Gálvez-López, D., Tardós, J.D., 2012. Bags of binary words for fast place recognition in image sequences. IEEE Trans. Robot. 28, 1188–1197. https://doi.org/10.1109/TRO.2012.2197158

Grisetti, G., Kümmerle, R., Stachniss, C., Frese, U., Hertzberg, C., 2010. Hierarchical optimization on manifolds for online 2D and 3D mapping. Proc. - IEEE Int. Conf. Robot. Autom. 273–278. https://doi.org/10.1109/ROBOT.2010.5509407

Grisetti, G., Stachniss, C., Grzonka, S., Burgard, W., 2008. A tree parameterization for efficiently computing maximum likelihood maps using gradient descent. Robot. Sci. Syst. 3, 65–72. https://doi.org/10.15607/rss.2007.iii.009

Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D., 2012. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. Int. J. Rob. Res. 31, 647–663. https://doi.org/10.1177/0278364911434148

Henry, P., Krainin, M., Herbst, E., Ren, X., Fox, D., 2010. RGB-D Mapping: Using Depth Cameras for 3D Modeling of Indoor Enviroments. Proc. Int. Symp. Exp. Robot. 1–7.

Hornung, A., Wurm, K.M., Bennewitz, M., Stachniss, C., Burgard, W., 2013. OctoMap: An efficient probabilistic 3D mapping framework based on octrees. Auton. Robots 34, 189–206. https://doi.org/10.1007/s10514-012-9321-0

Hou, J., Goebel, M., Iwaszczuk, D., 2023. Real-Time Tracking and 3D Dense Reconstruction Based on ORB-SLAM3 Extensions Using a Depth Camera, in: Publications of the German Society for Photogrammetry, Remote Sensing and Geoinformation. pp. 87–97.

Kümmerle, R., Grisetti, G., Strasdat, H., Konolige, K., Burgard, W., 2011. G2o: A general framework for graph optimization. Proc. - IEEE Int. Conf. Robot. Autom. 3607–3613. https://doi.org/10.1109/ICRA.2011.5979949

Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 60, 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94

Mur-Artal, R., Montiel, J.M.M., Tardos, J.D., 2015. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. IEEE Trans. Robot. 31, 1147–1163. https://doi.org/10.1109/TRO.2015.2463671

Mur-Artal, R., Tardos, J.D., 2017. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. IEEE Trans. Robot. 33, 1255–1262. https://doi.org/10.1109/TRO.2017.2705103

Newcombe, R.A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A.J., Kohli, P., Shotton, J., Hodges, S., Fitzgibbon, A., 2011. KinectFusion: Real-time dense surface mapping and tracking. 2011 10th IEEE Int. Symp. Mix. Augment. Reality, ISMAR 2011 127–136. https://doi.org/10.1109/ISMAR.2011.6092378

Niesner, M., Zollhöfer, M., Izadi, S., Stamminger, M., 2013. Real-time 3D reconstruction at scale using voxel hashing. ACM Trans. Graph. 32. https://doi.org/10.1145/2508363.2508374

Rublee, E., Rabaud, V., Konolige, K., Bradski, G., 2011. ORB: An efficient alternative to SIFT or SURF. Proc. IEEE Int. Conf. Comput. Vis. 2564–2571. https://doi.org/10.1109/ICCV.2011.6126544

Rusu, R.B., Cousins, S., 2011. 3D is here: Point Cloud Library (PCL). Proc. - IEEE Int. Conf. Robot. Autom. 1–4. https://doi.org/10.1109/ICRA.2011.5980567

Segal, A. V., Haehnel, D., Thrun, S., 2010. Generalized-ICP. Robot. Sci. Syst. 5, 161–168. https://doi.org/10.15607/rss.2009.v.021

Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D., 2012. A benchmark for the evaluation of RGB-D SLAM systems. IEEE Int. Conf. Intell. Robot. Syst. 573–580. https://doi.org/10.1109/IROS.2012.6385773

Teschner, M., Hiedelberger, B., Müller, M., Pomeranets, D., Gross, M., 2003. Optimized Spatial Hashing for Collision Detection of Deformable Objects. Vmv 2003 8. https://doi.org/10.1.1.4.5881

Triggs, B., McLauchlan, P., Hartley, R., Fitzgibbon, A., 1999. Bundle Adjustment — A Modern Synthesis. Conf. Rec. IEEE Photovolt. Spec. Conf. 34099, 298–372.