A Novel Modular Map Construction Method for VR/MR Glasses

Siyan Ma, Yaoyu Lv, Hao Zhang, Lili Chen, Yuanxing Zhao, Peng Han, Juanjuan Shi, Haofei Guo

BOE Technology Group Co., LTD., Beijing, China

Abstract

A novel modular map construction method for VR/MR Glasses is proposed in this paper, which combines the advantages of the sparse map's low memory cost and the advantages of the dense map's good performance and user experience. By using this method, the VR/MR System can realize modular map construction of indoor scenes to meet users' needs of reality-virtual interaction, obstacle avoidance and navigation.

Author Keywords

VR/MR Glasses; Modular Map Construction; Semantic Segmentation; 3D Object Detection

1. Introduction

When wearing VR/MR Glasses, users' vision is blocked by the glasses and they cannot see the outside world. In order to ensure the safety of users and meet the needs of virtual-real interaction, the VR/MR System needs to have the ability of navigation and obstacle avoidance. And the foundation of this ability is localization and map construction. At present, SLAM (Simultaneous Localization and Mapping) algorithms based on color cameras and IMUs have excellent performance and achieve precise position, such as ORB-SLAM3, but there is still room for improvement in map construction. In terms of the density of the map's feature points, map construction algorithms can be divided into sparse map construction algorithms and dense map construction algorithms, each with its own advantages and disadvantages. Sparse map construction algorithms have lower computational and memory costs and high real-time performance, but too sparse feature points can not reproduce the real scenes in the virtual world, and can not meet the users' needs of interaction and obstacle avoidance; The point cloud data of dense maps are mostly in the form of 3D coordinate and color or the 3D Gaussian Distribution representation, the amount of map's data is proportional to the density of the scene (the number of points). On the VR/MR Glasses, it is difficult to store such a large amount of data locally.

This work presents a novel modular map construction method, which has the advantages of low memory cost and excellent experience effect at the same time. In other words, the method proposed meets all the needs of map construction in navigation, obstacle avoidance and virtual-real interaction scenarios simultaneously.

2. Methodology

Figure 1 and 2 show the VR/MR map construction scheme proposed in this paper. In terms of hardware, the VR/MR Glasses and the server built based on RTX3090 are included. The scheme consists of two stages: map construction stage and real-time rendering stage. During the map construction stage, the data of RGB and Depth cameras and the current pose results outputted by SLAM module are respectively collected and transmitted to the server through the WebRTC module. On the server, the current map is generated by the 3D object detection module and the map update module. During the real-time rendering stage, the rendering module renders images which will be passed to the screen from the bird's-eye view or user's perspective according to the user's current position.

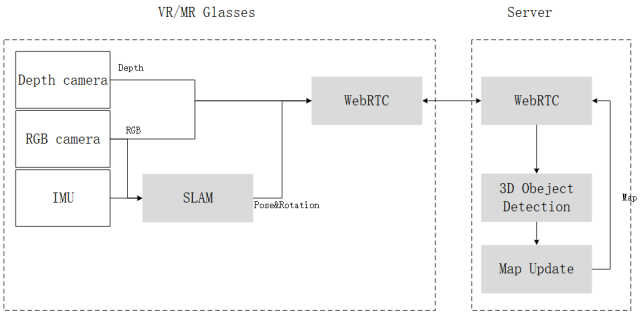

Figure 1. The map construction stage

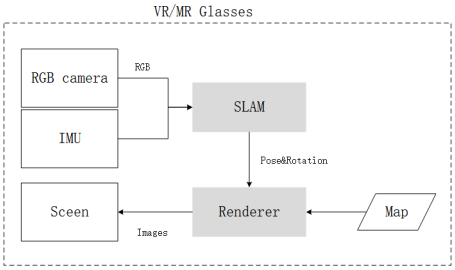

Figure 2. The real-time rendering stage

2.1 Map Construction Stage

This stage mainly includes two parts: 3D object detection, map update and storage.

2.1.1. 3D Object Detection

The 3D object detection algorithm proposed in this paper is implemented based on PAConv point cloud semantic segmentation algorithm and spatial geometry calculation.

Firstly, the RGB and Depth data at the same time are fused to generate point cloud data, and the PAConv algorithm is used for semantic segmentation.

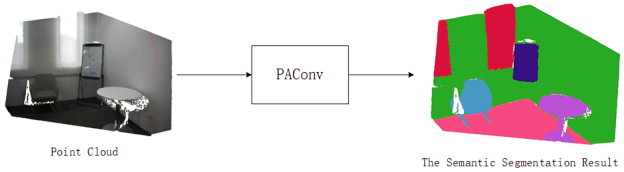

Figure 3. The semantic segmentation result

Secondly, the point cloud data of floor, walls and various indoor furniture are obtained from the semantic segmentation results.

Finally, aimed at different kinds of objects, we adopt different spatial computing methods to obtain their three-dimensional space representations.

Floor Detection: Floor detection is mainly realized based on RANSAC plane fitting algorithm, which obtains the largest plane's model from the point cloud of floor. The plane is

ISSN 0097-996X/25/5603-1559-$1.00 © 2025

represented by $Ax + By + Cz + D = 0$. And Establish the floor coordinate system with the normal vector $(A, B, C)^T$ as the positive direction of the Y-axis, the projection of the optical center of the camera on the floor as the origin of the coordinate system, the projection of the camera coordinate system's X-axis as the X-axis, and finally, the Z-axis is determined by the right hand rule.

Walls Detection: The walls detection is also realized by RANSAC plane fitting algorithm. Unlike the the point cloud of floor, the walls' point cloud may contain multiple sections of walls. The walls detection algorithm flow is shown in Figure 5. In each loop, we extract the point cloud of the largest plane from the whole point cloud of walls. In order to obtain the representations of the walls, we need to project the point cloud of the i^{th} wall onto the plane of the floor, which is fitted to the linear equation $a_i x + z = b_i$. The parameters can be calculated by the least square method. We can also get the start point and end point of the line segment, P_{i_start} and P_{i_end}. The plane model of the wall is determined by the normal vector of the floor's plane and any point on the line segment. We denote it as $A_i x + B_i y + C_i z + D_i = 0$ and remove the point cloud of the wall. This process is repeated until the number of points in the walls' point cloud is smaller than the threshold defined.

Other Objects Detection: For objects other than the floor and walls, multiple instances may be included in the point cloud of one kind in the same frame, such as the two chairs shown in Figure 6. For this case, the first thing we need to do is to separate the two point cloud data of the two chairs. In this paper, a novel clustering algorithm based on K-Means++ is presented. Take the point cloud of the two chairs in the image, the algorithm flow is as follows:

a. Denote the maximum radius of the point cloud of chair as r_{chair} and select one point $C_0 = (X_0, Y_0, Z_0)^T$ randomly as the center of the first chair, label it with 0.

b. Compute the distance d_i^j between every point $p_j = (x_j, y_j, z_j)^T$ who has no label and the every chair's center C_i, the minimum $d_{min} = min(d_i^j) = d_k^j$. If $d_{min} < r_{chair}$, label the point with i;

c. Compute the geometric center C_i of all points with the same label and the distance d_i^j between the point (denoted as p_j) labeled with i and C_i. If $d_i^j > r_{chair}$, remove the label of p_j;

d. Repeat step c until the distance the every center moves is smaller than $thresh_{distance}$. Count the number of points with the same label, if the number is smaller than $thresh_{number}$, remove the points' labels;

e. Count the number of points which has no label. If the number is smaller than $thresh_n$, the flow ends; Conversely, select one point which has no label randomly as the new center of $chair_{i+1}$;

f. Repeat steps b~f until the flow ends;

Figure 4. Point cloud data of floor, walls and other objects

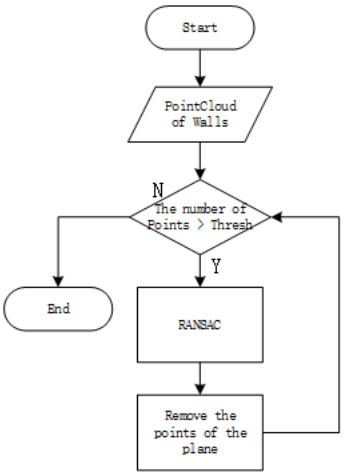

Figure 5. The walls detection algorithm flow

With the above steps, we have successfully get two separated sets of points from the image shown in Figure 6 and we denote as $cloud_{chair1}$ and $cloud_{chair2}$. Project the $cloud_{chairi}$ to the floor through $Cloud_{chairi}' = {}_{floor}T_c \cdot cloud_{chairi}$ and get the minimum 2D bounding box of them. And combine with the maximum value of points along the Y-axis denoted as h, we get the 3D bounding box of the $chair_i$. In this paper, we use the $\{p_j, j = 0,1,\cdots,7\}$ as the representation of the 3D detection result of $chair_i$.

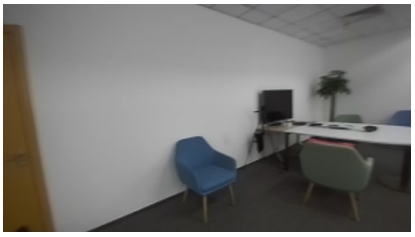

Figure 6. The image contains two chairs

2.1.2. Map Update and Storage

Map Update: This paper uses the transformation matrix from the current coordinate system to the world coordinate system or the reference coordinate system ${}_wT_c$ to convert the detection results of different frames to the same world coordinate, and make them fused based the NMS algorithm.

The location (X, Y, Z) and pose (x, y, z, w) of camera from SLAM module are transformed into matrix representation as ${}_wT_{ci}$;

The 3D object detection results are transformed to the world coordinates. For the planes, we use the formula $\vec{V}' = R_i \cdot \vec{V}$ and $(X_0', Y_0', Z_0')^T = R_i \cdot (X_0, Y_0, Z_0)^T + t_i$ to transform the normal vector and a random point of the plane into the world coordinate system. For the 3D bounding box, the 8 points $\{p_j, j = 0,1,\cdots,7\}$

are transformed to $\{p_j', j = 0,1,\cdots,7\}$;

Assume that at time t, we have the $map = \{Floor, Wall_0, Wall_1, Chair_0, Desk_0\}$, and the 3D object detection results of time $t+1$ is $\{Floor_i, Wall_{i0}, Chair_{i0}, Sofa_0\}$, the map update process is as follows:

- The fusion of floors: If $Floor$ and $Floor_i$ are parallel, the fusion result is $Ax + By + Cz + \frac{D_i+D}{2} = 0$; if $Floor$ and $Floor_i$ are not parallel, Cosine theorem is used to calculate the angle between the two normal vectors of the floors, and the angle's bisector is taken as the new normal vector. Then the representation of the new floor of time $t+1$ is generated by the the new normal vector and a random point on the intersection line of the two old floor plane;
- The fusion of walls: If there are no walls in the map, then brings $Wall_{i0}$ into the map; Conversely, if the map contains one or more walls, denoted as $Wall_j, j = 0,1,\cdots N$, we need to calculate the angle between the normal vector of $Wall_{i0}$ and each one wall's normal vector in map, the minimum of angles is denoted as $theta_{min} = min(theta_{i0}^0, theta_{i0}^1) = theta_{i0}^k$; If $theta_{min} < thresh_{angle}$, we think of $Wall_{i0}$ and $Wall_k$ as different parts of the same wall, and then merge the two planes of walls as described in a. The new start point and end point are the two farthest apart points of the two start points and two end points. If $theta_{min} \geq thresh_{angle}$, the $Wall_{i0}$ is a new wall, bring it to the map;
- The fusion of other 3D object detection results: For object that do not exist in the map, such as $Sofa_0$, just bring it to the map; For the $Chair_{i0}$, the map contains one or more chairs, like $Chair_0$, we should calculate the IOU of the 3D bounding box of $Chair_{i0}$ and each one chairs' 3D bounding boxes in map. In this paper, Monte Carlo method is used to obtain the IOU. We denote the maximum of IOU as $IOU_{max} = max(IOU_{i0}^m) = IOU_{i0}^k$. If $IOU_{max} > thresh_{IOU}$, we think the $Chair_{i0}$ and the $Chair_k$ are the same chair and the fusion result of the two chairs is the one with the bigger 3D bounding box based on the NMS Algorithm. If $IOU_{max} \leq thresh_{IOU}$, bring the $Chair_{i0}$ to the map;

With the above steps, we have successfully updated the map_t to map_{t+1}.

Map Storage: According to the dense map construction algorithms, the position and color information of all points like (X, Y, Z, R, G, B) need to be stored, the total amount of the information is $6n$. Storing in the form of a 3D Gaussian Distribution requires additional parameters of opacity o and radius r, the total amount of information is increased to $8n$. For an image with the resolution of 720p, the memory cost is 21MB and 28MB respectively. For our modular map, the memory cost is reduced to $24N$, N is the number of instances in the image. Therefore, the local memory cost of our method is significantly reduced.

2.2. Real-time Rendering Stage

In virtual-real interaction, navigation and obstacle avoidance scenarios of VR/MR System, it is necessary to show the rendered images of map from the bird's-eye view or the user's perspective. The following two different rendering methods can be selected according to user's needs or the glasses' computational power.

2.2.1. Simple Mode

In the scenarios, such as obstacle avoidance and navigation, users do not care about the details of map construction, we only need to use cubes of different colors and sizes to represent different objects in the scene (such as tables, chairs, etc.) on the designated position of the map and project them to the designated view.

2.2.2. Complex Mode

In the scenarios of virtual-real interaction and entertainment, users hope that the display of map construction will be more realistic and immersive under the premise of high real-time performance. This paper presents a rendering method based on the preset models. For common indoor objects, such as table, chair, sofa and other objects, the 3D models are collected in advance as shown in Figure 7, and rendered according to the target size, pose and other information, and then superimposed on the same image.

Figure 7. Common indoor objects' models

3. Results

In this paper, a ZED2i camera is used to simulate the RGB and Depth cameras integrated on VR/MR Glasses, and the pose outputs of ZED2i are used as SLAM results. The modular map construction result and the rendered images from the bird's-eye view of the real-world data are shown in Figure 8 and Figure 9.

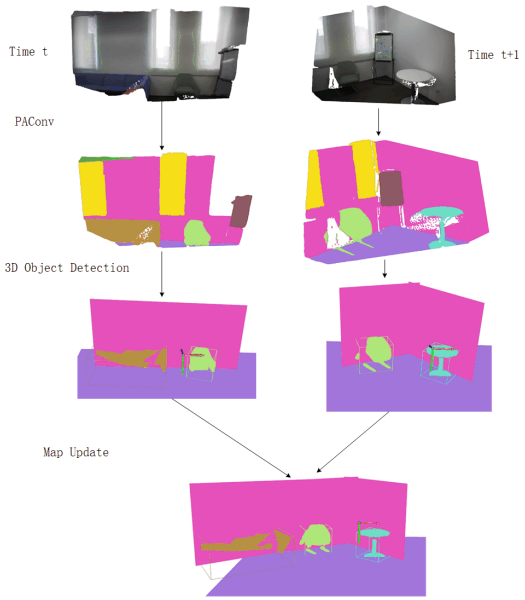

Figure 8. The map construction result of real data

4. Conclusions

This paper presents a semi-dense modular map construction method for VR/MR Glasses based on RGBD cameras and SLAM module, which requires less local memory and keeps the user experience. Based on the performance of the real-world data, our method is proved to be effective.

Figure 9. Images rendered from bird's-eye view in simple mode and complex mode

References

1. Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan and Jonathon Luiten. SplaTAM: Splat, Track & Map 3D Gaussians for Dense RGB-D SLAM. CVPR(2024).
2. Mutian Xu, Runyu Ding, Hengshuang Zhao, and Xiaojuan Qi. PAConv: Position Adaptive Convolution with Dynamic Kernel Assembling on Point Clouds. CVPR(2021).
3. Arthur David, S. Vassilvitskii. K-Means++: The Advantages of Careful Seeding. ACM (2007).
4. Campos Carlos, et al. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. (2020)