Advanced Robotics

ISSN: (Print) (Online) Journal homepage: https://www.tandfonline.com/loi/tadr20

Structure SLAM with points, planes and objects

Benchun Zhou, Maximilian Gilles & Yongqi Meng

To cite this article: Benchun Zhou, Maximilian Gilles & Yongqi Meng (2022) Structure SLAM with points, planes and objects, Advanced Robotics, 36:20, 1060-1075, DOI: 10.1080/01691864.2022.2123253

To link to this article: https://doi.org/10.1080/01691864.2022.2123253

ADVANCED ROBOTICS
2022, VOL. 36, NO. 20, 1060–1075
https://doi.org/10.1080/01691864.2022.2123253

FULL PAPER

Structure SLAM with points, planes and objects

Benchun Zhou, Maximilian Gilles and Yongqi Meng

Institute for Material Handling and Logistics, Karlsruhe Institute of Technogy, Karlsruhe, Germany

ABSTRACT

Simultaneous localization and mapping (SLAM) is a fundamental problem for indoor mobile robots operating in unknown environments. While visual SLAM systems often use geometry features, the reconstructed maps lack semantic information. On the other hand, current object detection methods provide rich information about the scene from the image. In this paper, we present a structure SLAM system with feature points, geometry planes, and semantic objects. Unlike other systems modeling planes and objects as collections of points, we choose a parametric representation for these landmarks. For every single frame, we start by generating cuboid candidates of detected objects with varying dimensions and orientations, then use 2D-3D fitting constraints to calculate the cuboid's translation, and finally introduce 3D spatial and 2D image constraints to select the best cuboid candidate. For SLAM optimization, the detected planes and objects can provide geometry constraints to improve the localization result, and act as landmarks to reconstruct a semantic map. Experiments on the ICL NUIM RGB-D dataset show that the proposed point-plane-object SLAM system can slightly improve localization accuracy, and is able to build a semantic map of the environment.

ARTICLE HISTORY
Received 21 March 2022
Accepted 11 August 2022

KEYWORDS
3D object detection; 2D-3D fitting; point-plane-object SLAM

1. Introduction

Semantic understanding and simultaneous localization and mapping are two essential tasks in computer vision and robotics [1]. Traditional visual SLAM systems reconstruct the map as a collection of feature points, which is sparse and efficient to compute and update. However, this map does not contain object information, and is not convenient for high-level tasks such as robot manipulation. In man-made environments, many structures and objects carry semantics, and can potentially help reconstruct the map. For example, planes are good geometric elements to model the dominant structural layout of the environment [2]. Objects can also provide semantic information to help understand the surrounding [3]. Incorporating planes and objects will not only add landmarks to the map, but also provide additional constraints for SLAM optimization. Previous plane-based SLAM [4] and object-oriented SLAM [5] systems focused either on planes or objects, but hardly considered both. The reason is that these high-level landmarks have different representations, making it challenging to explore the spatial relationship between them. In this paper, we do not consider dense reconstruction with points or meshes, but rather represent planes and objects with parametric representations. Based on these representations, a point-plane-object SLAM system is proposed to jointly optimize the camera pose and reconstruct a semantic map.

We propose a point-plane-object SLAM system and address three main challenges: object detection from a single frame, multi-view SLAM optimization, and data association. Given a single frame, we firstly estimate the objects assuming 3D cuboid models. Then, the detected objects, together with planes, feature points and camera poses, are involved in a unified bundle adjustment framework for further optimization. In addition, to ensure a stable and robust system, the spatial relationship among points, planes and objects is exploited.

In summary, the main contributions of this work can be described as:

- A 3D object detection pipeline, where cuboid candidates are sampled from RGB images and scored by 3D and image constraints.
- A structure SLAM system, where feature points, structural planes, and semantic objects are optimized within a unified bundle adjustment framework.
- A data association method, which explores the spatial relationship among points, planes, and objects with mathematical representations.

CONTACT Benchun Zhou benchun.zhou@kit.edu

This article was originally published with errors, which have now been corrected in the online version. Please see Correction (http://dx.doi.org/10.1080/01691864.2022.2151269)

Supplemental data for this article can be accessed here. https://doi.org/10.1080/01691864.2022.2123253

- Experiments on living room and office datasets demonstrating the effectiveness of our point-plane-object SLAM system.

2. Related work

2.1. Plane estimation and point-plane SLAM

Given a single frame, there are many approaches to estimate planes. An efficient way is to convert the depth image together with the camera's intrinsic parameters into a point cloud and to segment it by applying plane estimation methods, such as RANSAC [6], region growing [7] or organized point cloud approaches [8]. With the development of deep neural networks, plane estimation from RGB images has also become possible: PlaneNet [9], PlaneRCNN [10], PlaneRecover [11] are some examples in this field.

Planes are important geometric features in a structural environment and can be represented in different ways. Point cloud representation is one common model, which defines a plane as a group of points [12]. However, since one plane may contain hundreds of points, optimization and data association operations have high computational costs regarding speed and memory. Kaess et al. [13] introduced a homogeneous representation of planes and refined the map with incremental solvers. Similar to quaternions, which are commonly used to represent rotations, a plane in homogeneous representation can be normalized to lie on the unit sphere and be updated in the tangent space using the exponential mapping.

When involved into a SLAM system, planes can act as landmarks to build a map, and provide geometry constraints to improve the camera localization. Hsiao et al. [14] modeled planes as point clouds and proposed a key-frame based planar SLAM method to reconstruct dense indoor environments. Zhang et al. [15] exploited the boundaries of planes and generated perpendicular planes, this method added more constraints between planes but required higher computational costs. Yang et al. [16] proposed pop-up planes and estimated them from wall-floor boundaries. These plane landmarks improved the robustness of the SLAM system, especially in low-texture scenes, but were limited to wall planes.

In this work, planes are represented as homogeneous vectors. Unlike other systems only considering structural planes, such as wall or ground, we also take object surface planes into account.

2.2. 3D object detection and object-oriented SLAM

3D object detection from a monocular image remains a challenging problem because only 2D information is available. Regarding different representation models, object detection methods can be divided into two categories: with or without object models. If prior knowledge is available in the form of 3D models [17], objects can be detected by matching key points to the models. However, the requirement for prior 3D object models limits the application area. Artificial object models are more flexible and compact. Rubino et al. [18] proposed to initialize quadrics from multi-view frames. 2D bounding boxes with corresponding fitted ellipses can be formed as a closed-form system to initialize a 3D quadric. Nicholson et al. [19] back-projected the edge of the 2D object bounding box in order to get 3D enveloping planes, thus determining an optimal dual quadric tangent to the planes. Both approaches [18,19] require more than 3 observations for object initialization. Cuboid, another geometry model, can be formulated from a single image. Yang et al. [20] sampled points from the 2D bounding box as cuboid corners, calculated the remaining corners using vanishing lines, and then scored them with image features. Their method assumes that the objects are laying on the ground and requires camera height to recover the objects' scale. Mousavian et al. [21] proposed to predict the object dimension and orientation with deep neural networks. However, for these data-driven methods to generalize well to new environments, a lot of training data is needed, limiting the application area.

The introduction of objects in a SLAM system has a two-fold benefit. One is that objects can serve as geometry landmarks to improve the localization accuracy. QuadricsSLAM [19] used dual quadrics as 3D landmark representations. Their experiments showed that quadric landmarks provide valuable information for correcting odometry errors. CubeSLAM [20] represented objects as cuboids and could demonstrate that object detection and SLAM benefit from each other in both indoor and outdoor scenarios. Another significant benefit of using objects in SLAM systems is scene understanding, where the estimated map is enriched with the objects as semantic elements. Sünderhauf et al. [22] proposed a semantic mapping system that contains object-level entities and mesh-based geometric representations. With a similar idea, Dengler et al. [23] extended a static system to be capable of online semantic mapping and object updating.

In our work, we focus on single frame object detection and object-oriented SLAM. In order to overcome the lack of generality deep learning based approaches are facing, we choose cuboids as representation and use a sample-score strategy to generate cuboid proposals. The detected objects, together with other landmarks, are involved in a unified bundle adjustment framework for localization and mapping.

2.3. Point-plane-object SLAM and data association

The point-plane-object SLAM problem can be defined as follows: given feature points P, geometric planes Y and semantic objects S measurements, estimate the sensor state trajectory X and the position T of objects in the environment [24]. Hosseinzadeh et al. [25] proposed a structure-aware SLAM system using quadrics and planes, where the quadrics and planes were estimated from RGBD frames, together with a supporting affordance relationship between them. Liao et al. [26] estimated planes and objects from the depth image, and exploited the spatial relationship between these landmarks. Yang et al. [27] proposed a monocular framework by combing points and object and plane landmarks through unified bundle adjustment optimization. Their work showed that semantic scene understanding and traditional SLAM optimization can benefit each other.

Data association among landmarks is another crucial problem in a SLAM system, especially when duplicated objects exist. Bowman et al. [24] formulated an optimization problem over sensor states and semantic landmark positions and decomposed them into two interconnected problems: an estimation of discrete data association and landmark class probabilities, and a continuous optimization over the metric states. The estimated landmarks and robot poses affected the association and class distributions, which in turn affected the robot-landmark pose optimization. Li et al. [28] proposed a novel object-level data association algorithm based on the bag of words algorithm, using both geometric and appearance information of the object. Wu et al. [29] proposed an ensemble data association strategy to integrate parametric and non-parametric statistic tests, which can effectively aggregate different measurements of the objects to improve association accuracy.

Our paper aims to build the point-plane-object SLAM system representing all landmarks in parametric form: feature points as pixels, planes as homogeneous vectors, and objects as cuboids. In addition, we also exploit the data association problem among different landmarks to provide more constraints for SLAM optimization.

3. Methods

3.1. Single frame object detection

Objects are high-level entities of the environment that carry semantic and geometric information. In this paper, a novel method is proposed to estimate them from a single frame. We start by sampling object dimension and orientation to generate cuboid candidates (see Subsubsection 3.1.1), and then score them with 3D spatial constraints and 2D image constraints to get the best object proposal (see Subsubsection 3.1.2). The whole process is shown in Figure 1.

3.1.1. Object representation and candidates generation

Objects are represented as cuboids with 9 DoF parameters: 3 DoF position $\mathbf{T} = (t_x, t_y, t_z)^\top$, 3 DoF orientation $\mathbf{R} = (\theta_x, \theta_y, \theta_z)^\top$, and 3 DoF dimension $\mathbf{D} = (d_x, d_y, d_z)^\top$. The cuboid coordinate frame is located at the cuboid center, aligned with the main axes.

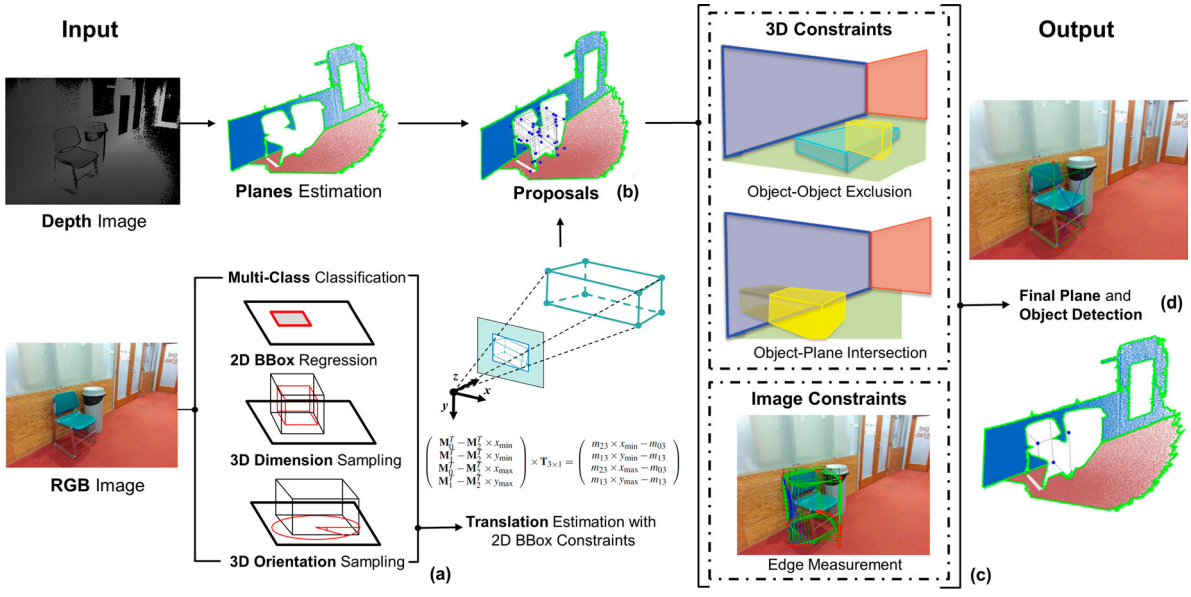

Figure 1. Overview of our proposed 3D object detection method. We take a single frame as input, sample object dimension and orientation to generate cuboid candidates (*a*), and score these candidates (*b*) with spatial and image constraints (*c*). The best candidate is selected to represent the object (*d*).

Using a 2D object detector to detect object bounding box and class category from a single RGB image, we generate cuboid candidates by sampling dimension and orientation. The object dimension is sampled based on average value from measurement, while the object orientation is sampled related to the supporting plane. Since most objects are supported by ground, desk, or wall, it is sufficient to sample one angle θ relative to the supporting plane normal.

Once we know the 2D bounding box, dimension and orientation, the translation of each candidate can be calculated by the 2D-3D fitting constraints [21]: the projected vertexes of a 3D cuboid should fit tightly into each side of its 2D detection box. That means, four out of eight vertexes of the 3D bounding box should be projected right on the four sides of the 2D bounding box. To do that, we formulate the perspective camera projection function as:

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{P_{roj}} \begin{pmatrix} \mathbf{X_{3D}} \\ 1 \end{pmatrix}, \tag{1}$$

where (u, v) is the pixel position in image, $\mathbf{P_{roj}}$ is camera projection matrix, and $\mathbf{X_{3D}}$ is 3D cuboid vertexes in camera coordinates. Given the cuboid translation $\mathbf{T}$, dimension $\mathbf{D}$ and orientation $\mathbf{R}$, (1) can be rewritten as:

$$\begin{aligned} \lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} &= \mathbf{P}_{\text{roj}} \begin{pmatrix} \mathbf{I}_{3\times3} & \mathbf{R}_{3\times3}(\theta) \cdot \mathbf{D}_{3\times1} \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{T}_{3\times1} \\ 1 \end{pmatrix} \\ &= \mathbf{M}_{3\times4} \begin{pmatrix} \mathbf{T}_{3\times1} \\ 1 \end{pmatrix}, \end{aligned} \tag{2}$$

where $\mathbf{M} = \mathbf{P_{roj}} \times (\mathbf{I}|\mathbf{R} \times \mathbf{D})$. If we assume $\mathbf{M}_i^T = [m_{i0}, m_{i1}, m_{i2}]$, and m_{ij} is the (i, j) elements of the matrix $\mathbf{M}$, the equation can be rewritten as:

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{M}_0^T \times \mathbf{T}_{3\times1} + m_{03} \\ \mathbf{M}_1^T \times \mathbf{T}_{3\times1} + m_{13} \\ \mathbf{M}_2^T \times \mathbf{T}_{3\times1} + m_{23} \end{pmatrix}. \tag{3}$$

Introducing the 2D bounding box as constraint, for example the right border of the bounding box $x_{\max}$, we get the following equation:

$$x_{\max} = \frac{\mathbf{M}_0^T \times \mathbf{T}_{3\times1} + m_{03}}{\mathbf{M}_2^T \times \mathbf{T}_{3\times1} + m_{23}}, \tag{4}$$

$$\left(\mathbf{M}_0^T - \mathbf{M}_2^T \times x_{\max}\right) \times \mathbf{T}_{3\times1} = m_{23} \times x_{\max} - m_{03}. \tag{5}$$

As we can see, one border of the bounding box can formulate one equation to solve for $\mathbf{T}$. Taking four borders, left $x_{\min}$, right $x_{\max}$, top $y_{\min}$ and bottom $y_{\max}$ into account, we obtain:

$$\begin{aligned} &\begin{pmatrix} \mathbf{M}_0^T - \mathbf{M}_2^T \times x_{\min} \\ \mathbf{M}_1^T - \mathbf{M}_2^T \times y_{\min} \\ \mathbf{M}_0^T - \mathbf{M}_2^T \times x_{\max} \\ \mathbf{M}_1^T - \mathbf{M}_2^T \times y_{\max} \end{pmatrix} \times \mathbf{T}_{3\times1} \\ &= \begin{pmatrix} m_{23} \times x_{\min} - m_{03} \\ m_{13} \times y_{\min} - m_{13} \\ m_{23} \times x_{\max} - m_{03} \\ m_{13} \times y_{\max} - m_{13} \end{pmatrix}, \end{aligned} \tag{6}$$

$$\mathbf{A} \times \mathbf{T}_{3\times1} = \mathbf{b}(\mathbf{b} \neq \mathbf{0}), \tag{7}$$

$$\mathbf{T}_{3\times1} = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\mathbf{b}, \tag{8}$$

It is an over-constrained system, and the translation $\mathbf{T}$ can be solved by the least-squares method. Each constraint of the 2D bounding box can correspond to any of the 8 corners of the 3D box, which results in $8^4 = 4096$ configurations. As discussed in [21], when the object is assumed to be parallel to the ground plane, one can narrow the configuration to 64, and choose the minimal error of the least square equation.

Until now, based on a single RGB image, we generate many cuboid candidates with class name, 2D bounding box, dimension, orientation, and translation. Then, to select the best cuboid proposal, we utilize the 3D spatial and 2D image features constraints.

3.1.2. Candidate selection with constraints

Given a set of cuboid candidates, our goal is to select the best proposal that not only satisfies the spatial constraints of the physical world, but also matches the local surface geometry estimated from image cues.

Every cuboid candidates in 3D space should have a non-zero finite volume. We formulate the spatial constraints [30] as object-object exclusion and object-plane constraints, implying that the volumes occupied by different objects are mutually exclusive, and every object should be close to its associated planes without intersection. When projecting the 3D cuboid to the image, the feature constraints are defined as the alignment between the projected cuboid edges and detected lines in the image. In summary, the overall cost can be formulated as:

$$E(O, \Pi, I) = k_1 \times \Psi_{O-O} + k_2 \times \Psi_{O-\Pi} + k_3 \times \Psi_{O-I}, \tag{9}$$

where O, Π, and I are defined as objects, planes and image respectively. k_1, k_2 and k_3 are coefficients to balance the 3D and 2D constraints. Ψ_{O-O} represents object-object constraints, and we use 3D intersection over union (3D IoU) between two cuboids $x_i, x_j (i \neq j)$ to measure it, as

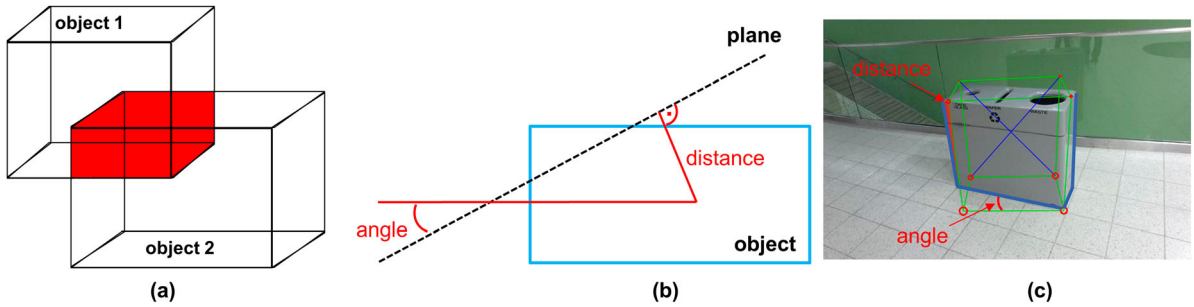

Figure 2. Proposed constraints for object selection. (*a*) Object-object constraints are calculated by object 3D IoU. (*b*) Object-plane constraints are calculated with angle and distance error. (*c*) Object-image constraints are calculated by angle and distance error of the image lines and projected cuboid edges.

shown in (10) and Figure 2(a). $\Psi_{O-\Pi}$ represents object-plane constraints. If a plane π_j is associated to an object x_i, the angle and distance cost are computed as in (11) and Figure 2(b). For object-image constraints Ψ_{O-I}, we measure the angle and distance error between the projected cuboid's edges l_i and the detected image lines l_j, as shown in (12) and Figure 2(c). To calculate the distance cost $dist(l_i, l_j)$, we sample 10 points from image lines and calculate their orthogonal distance to the projected edges. Based on (9), the cuboid candidate with the smallest cost $E(O, \Pi, I)$ is selected as the object proposal.

$$\Psi_{O-O}(x_i, x_j) = x_i x_j \frac{V(x_i) \cap V(x_j)}{V(x_i) \cup V(x_j)}, \tag{10}$$

$$\Psi_{O-\Pi}(x_i, \pi_j) = x_i \pi_j [(\theta_{x_i} - \theta_{\pi_j}) + dist(x_i, \pi_j)], \tag{11}$$

$$\Psi_{O-I}(x_i, I) = \sum_{l_i \in x_i, l_j \in I} l_i l_j [(\theta_{l_i} - \theta_{l_j}) + dist(l_i, l_j)], \tag{12}$$

3.2. Point-plane-object SLAM

We treat detected objects as semantic landmarks to improve the localization and mapping result. A point-plane-object SLAM system (see Figure 3) is proposed based on monocular ORB-SLAM2 [31]. Plane estimation and object detection are only implemented at selected key-frames (key-frame based SLAM). We adopt the for indoor environment suitable Manhattan world assumption [32] and assume that planes are parallel or perpendicular to each other. We filter planes with regard to three main directions, associate them with cuboids to score the candidates, and add selected cuboid proposals to the local bundle adjustment framework. The whole system is initialized by ORB feature points, and loop closure is implemented using the bag of words algorithm [31]. In the following, we first formulate the optimization problem, then explain various measurement costs.

3.2.1. Bundle adjustment formulation

Bundle adjustment is an optimization process to jointly optimize camera poses $C = \{c_i\}$, cuboids $O = \{o_j\}$, planes $\Pi = \{\pi_k\}$ and feature points $P = \{p_m\}$. It can be formulated as a nonlinear least squares problem:

$$C^*, O^*, \Pi^*, P^* = \underset{\{C,O,\Pi,P\}}{\arg\min} \sum_{i \in C, j \in O, k \in \Pi, m \in P} \mathbf{e}^T \mathbf{\Sigma} \mathbf{e}, \tag{13}$$

where $\mathbf{\Sigma}$ is a covariance matrix of different error measurements. The measurement error between different landmarks is represented by $\mathbf{e}$ and will be explained in Subsubsection 3.2.2. The problem can be solved by Gauss–Newton or Levenberg–Marquardt algorithm available in many libraries, such as g2o [33]. Huber robust cost function [31] is applied to all measurement errors to improve the robustness.

3.2.2. Measurements

In our system, the landmarks are camera poses $\mathbf{T}_c \in SE(3)$, feature points $\mathbf{P} = (x, y, z, 1)^\top \in \mathbb{R}^3$, planes $\boldsymbol{\pi} = (n_x, n_y, n_z, d)^\top \in \mathbb{R}^3$, and objects $\mathbf{O} = (\mathbf{T}_o, \mathbf{D})$, where $\mathbf{T}_o \in SE(3)$ and $\mathbf{D} \in \mathbb{R}^3$ are object pose and dimension. As shown in Figure 4, we integrate all the landmarks into a unified bundle adjust framework, explore the measurements between these landmarks, and jointly optimize them in a key-frame based SLAM system. Here we present a brief overview of these measurements.

(1) *Camera-Point Measurement:* We follow ORB-SLAM2 [31] to minimize the geometric re-projection error as:

$$e(\mathbf{P}, \mathbf{T}_c) = \mathbf{u}_c - \rho(\mathbf{T}_{cw}^{-1}, \mathbf{P}_w), \tag{14}$$

where $\mathbf{u}_c$ is the image pixel location of the 3D point P_w in the global world frame, $\rho(.)$ is a function that projects a world 3D point into the image plane.

(2) *Camera-Plane Measurement:* Since the homogeneous representation of the plane $\boldsymbol{\pi} = (n_x, n_y, n_z, d)^\top$ will cause an over-parameterization in optimization problem [15], we use minimal representation of planes

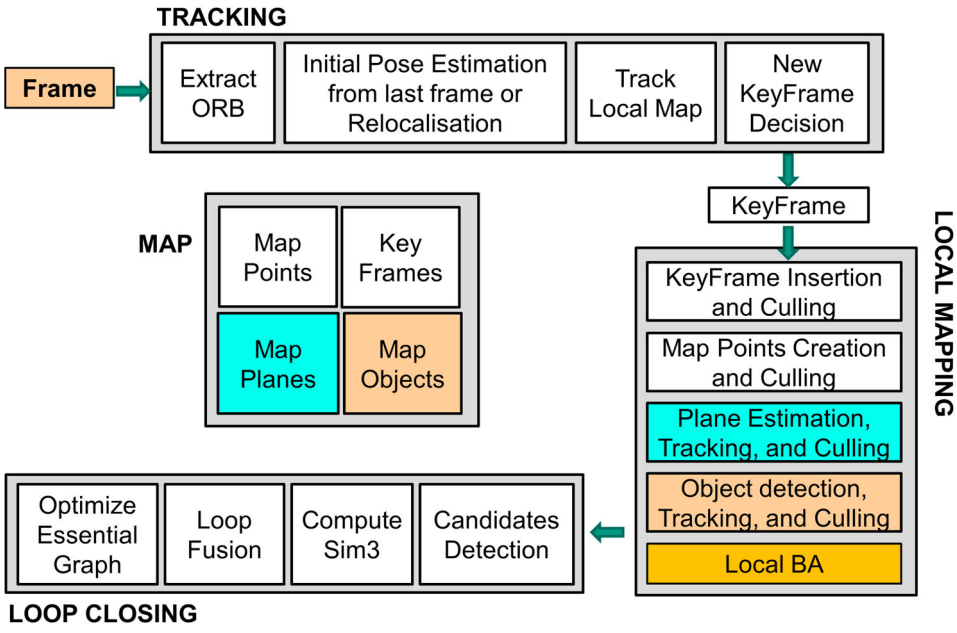

Figure 3. Our point-plane-object SLAM framework based on monocular ORB-SLAM2. We detect planes and objects in every key-frame, track them and optimize them with a local bundle adjustment framework. As a result, the reconstructed map also contains planes and objects.

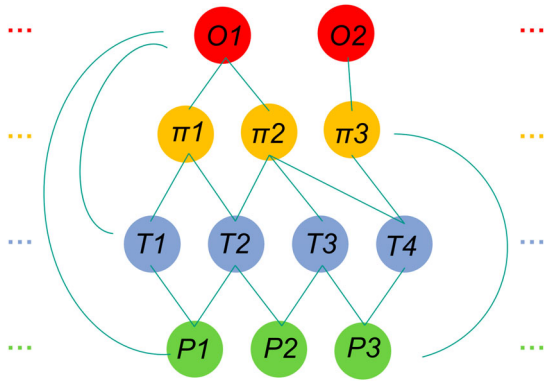

Figure 4. SLAM optimization graph of our system, where the blue nodes with *T* denote the camera key-frames, while the red nodes with *O*, yellow nodes with Π and green nodes with *P* denote the objects, planes, feature points landmarks respectively. The edges represent the measurements between them.

$q(\boldsymbol{\pi}) = (\phi, \psi, d)^\top$ to formulate the camera-plane measurement as:

$$e(\boldsymbol{\pi}, \mathbf{T}_c) = q(\boldsymbol{\pi}_c) - q\left(\mathbf{T}_{cw}^{-\top} \boldsymbol{\pi}_w\right), \tag{15}$$

where $\boldsymbol{\pi}_c$ is the observed plane in the current frame, $\mathbf{T}_{cw}^{-\top} \boldsymbol{\pi}_w$ means to transform 3D planes $\boldsymbol{\pi}_w$ from world coordinates to current frame, and $q(\boldsymbol{\pi})$ is defined as

$$q(\boldsymbol{\pi}) = \left(\phi = \arctan \frac{n_y}{n_x}, \psi = \arcsin n_z, d\right)^\top, \tag{16}$$

where ϕ and ψ are the azimuth and elevation angle of the plane normal respectively and restricted to $(-\pi, \pi]$ to avoid singularities.

(3) *Camera-Object Measurement:* Similar to camera-point measurement, we project the cuboid landmark $\mathbf{O}_w$ onto the image plane to get 8 vertexes and associate the projected cuboid vertexes to the detected object vertexes $\mathbf{z}_m$, measurement is the sum of geometry distance between these vertexes:

$$e(\mathbf{O}, \mathbf{T}_c) = \sum_{m=1}^{8} \mathbf{z}_m - \rho(\mathbf{T}_{cw}^{-1}, \mathbf{O}_w). \tag{17}$$

(4) *Point-Plane Measurement:* If a feature point $\boldsymbol{P}$ lies on a specific plane $\boldsymbol{\pi}$, the point-plane measurement is defined as the orthogonal distance from that point $\boldsymbol{P}$ to its associated plane $\boldsymbol{\pi}$:

$$e(\mathbf{P}, \boldsymbol{\Pi}) = \boldsymbol{\pi} \mathbf{P}. \tag{18}$$

(5) *Point-Object Measurement:* If a point $\boldsymbol{P}$ belongs to an object, it should lie inside the 3D bounding box. We transform the point into the associated cuboid frame and compare its location with the cuboid's dimensions, defined in [20]:

$$e(\mathbf{P}, \mathbf{O}) = \max\left(\left|\mathbf{T}_o^{-1} \mathbf{P}\right| - \mathbf{D}, \mathbf{0}\right), \tag{19}$$

where max operator is used because we only encourage points to lie inside the cuboid instead of the surface.

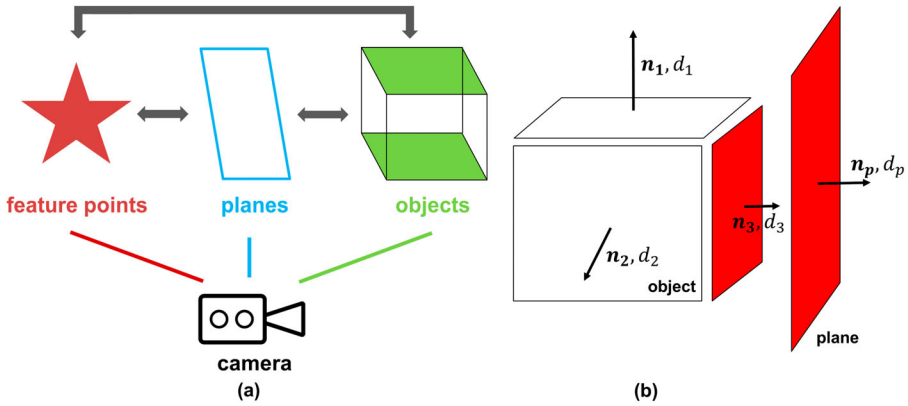

Figure 5. Landmarks overview and object-plane measurements. (*a*) In every key-frame, the camera can observe feature points, planes and objects landmarks, and explore the relationship among them. (*b*) A demonstration of object-plane measurements, where the associated planes should have a similar angle and minimum distance to one of the cuboid surface planes.

(6) *Plane-Object Measurement:* Two different types of planes are associated with objects. One is the structural plane, such as the ground or the wall. They are parallel to the world structure and provide supporting constraints for objects. The other plane type is the object surface plane, which belongs to the object and should be close to the cuboid proposal. These different planes can be assigned with individual weights. For simplification, we treat them in the same way.

Since an object is represented by a cuboid, the six surface planes $\boldsymbol{\pi_{oi}}, i \in \{1, 6\}$ can be defined by the center position, orientation and dimension of the object. If a plane $\boldsymbol{\pi}$ is associated with the object, as shown in Figure 5(b), it should have a similar angle and minimum distance to one of the cuboid surface planes. Then, the plane-object measurement can be replaced by plane-plane measurement and calculated as:

$$e(\mathbf{\Pi}, \mathbf{O}) = \min(q(\boldsymbol{\pi}) - q(\boldsymbol{\pi}_{oi})). \tag{20}$$

3.2.3. Data association

Data association among different landmarks is an important part of the SLAM system, as it happens in single frame detection, across frame tracking, and frame-map matching. For every key-frame, feature points, planes and objects are detected and associated. The point-plane association is computed by the orthogonal distance between them. Points are selected as in-plane points P_{plane} when the distance satisfies a threshold $d_{\pi p}$ and they are observed in at least 3 key-frames. For point-object association, we check if the pixel position of points lies inside the object's 2D bounding box. If the feature points are inside and observed in more than 3 key-frames, they will become in-object points P_{object}. Finally, we associate plane and object by checking if their orientation and orthogonal distance meet the angle threshold $\theta_{\pi o}$ and distance threshold $d_{\pi o}$.

We also consider landmark tracking across frames. The feature points are matched by bag of words algorithm [31]. For plane-plane association, three conditions are required: the orientation threshold $\theta_{\pi\pi}$, the orthogonal distance threshold $d_{\pi\pi}$, and the minimum number of in-plane points $N_{\pi\pi}$. For object-object association, two objects that satisfy a 2D IoU threshold IoU_{2D} and share a minimum number of in-object points N_{oo} are regarded as associated.

To reduce the visual odometry drift, we initialize and maintain the point map from ORB-SLAM2 and enrich it with plane and object landmarks. For every key-frame, we transfer map landmarks to the current frame and match them to the currently detected landmarks. The thresholds are the same as across frame tracking. If the detected landmarks do not satisfy these conditions, they will be initialized as new landmarks and added to the map.

4. Experiments

4.1. Experiments–single frame object detection

4.1.1. Implementation details

We presented 3D object detection experiments on SUN RGB-D dataset [34], it assumes that all objects are aligned with the gravity direction, making it suitable to evaluate our proposed method. There are more than 1000 different classes in the dataset, we select 7 common objects for further evaluation and compute their average dimension in Table 1. To focus on 3D object detection, we adopt the 2D bounding box and class name from the ground truth. To generate 3D cuboid candidates, we sample object dimension around the average value between

Table 1. Object average dimension in SUN RGB-D dataset.

Object	Number	Length (m)	Width (m)	Height (m)
bed	1286	0.7723	1.012	0.5151
night stand	548	0.3166	0.2541	0.3503
lamp	880	0.2015	0.1894	0.3959
sofa	1333	0.9186	0.4973	0.4218
sofa chair	1770	0.4198	0.4174	0.4167
chair	19294	0.277	0.2941	0.4202
garbage bin	1016	0.2025	0.1852	0.2824

Notes: We collect the data from ground truth and sample around the average value to generate cuboid candidates.

80% and 120% with a sample step of 20% to avoid invalid parameters. For orientation sampling, it is sufficient to only sample yaw angle from 0° to 180° with a sample step of 5°. So, for every object, we generate 972 cuboid candidates for scoring. The plane parameters are estimated from the depth image with the fast plane segmentation algorithm in near real-time [8]. In our experiment, the coefficients k_1, k_2, k_3 are set to 1.0, 0.5 and 0.8.

The official SUN RGB-D dataset [34] provides two metrics for 3D object detection: 3D intersection over union (IoU) and the orientation difference. We compare our method with other single frame 3D object detection methods: SUN Primitive [35] and CubeSLAM [20].

4.1.2. Quantitative results and analysis

As described in Figure 1 and Section 3.1.2, spatial and image feature constraints are introduced for object selection. To compare different constraints, we take *garbage bin* as an example and visualize the detection result in Figure 6. The first column shows the estimated planes and detected 2D bounding boxes, acting as the input of our method. In the second column, we only sample object orientations and adopt the average object dimension from Table 1. In this case, the image feature constraint Ψ_{O-I} is the main factor in object selection. It is possible to select the cuboid proposal, but the results are not satisfying. In the third column, we sample both dimension and orientation, but only use image feature constraints for selection. We could observe that the results are not sufficient, because cuboid candidates with different dimensions can have similar projections when projected into image plane. This scale problem always exists in monocular 3D object detection. It is not sufficient to rely solely on image features for selecting cuboid proposals. Therefore, in our work, we introduce plane-object constraints $\Psi_{O-\Pi}$ to score the cuboid candidates. The plane-object constraint is calculated by the angle difference and center distance, which encourages the cuboid proposal to align with planes with a suitable dimension. The results are shown in the fourth and fifth column of Figure 6. Experiments on other object types can be found in Table 2. It can be observed, that for most objects, the introduction of plane-object constraints improves the 3D object IoU performance.

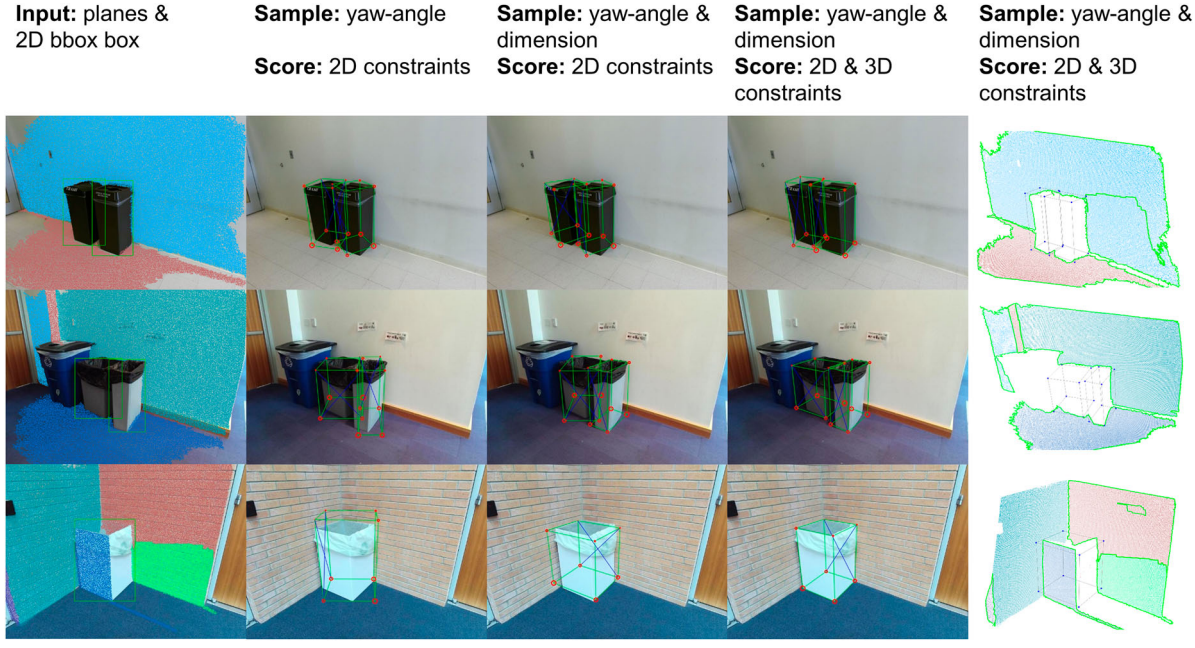

Figure 6. Single frame object detection result on SUN RGB-D dataset. The first column shows the 2D bounding box and estimated planes acting as input. In the second column, we only sample yaw and use image feature constraints for selection. In the third column, we sample yaw and dimension, but only use image feature constraints for selection. In the fourth and fifth column, we sample yaw and dimension and use both image feature and plane constraints for selection. The results are shown in image and 3D space.

Table 2. Object 3D IoU with different constraints.

Object	Number	Sample yaw with 2D constraints	Sample yaw and dimension with 2D constraints	Sample yaw and dimension with 2D and 3D constraints
bed	149	0.4467	0.4352	**0.4979**
night stand	101	0.2317	0.2301	**0.3559**
lamp	75	**0.2843**	0.2575	0.2578
sofa	40	0.2820	0.3032	**0.3189**
sofa chair	77	0.2964	0.3586	**0.3740**
chair	163	0.3387	0.3688	**0.3988**
garbage bin	77	0.3580	0.2693	**0.3786**

Table 3. Object detection results for our proposed method on SUN RGB-D dataset.

Object	Number	Position error (m)	Yaw error (rad)	3D IoU
bed	149	0.3971	0.1602	**0.4979**
night stand	101	0.2420	0.1634	0.3559
lamp	149	0.2376	0.1371	0.2578
sofa	40	0.4352	0.1569	0.3189
sofa chair	77	0.3244	0.1601	0.3740
chair	163	0.2280	0.1637	0.3988
garbage bin	77	**0.1633**	**0.1090**	0.3786

A subset of the object detection results in image and 3D space is shown in Figure 7. A quantitative analysis of our proposed method regarding the position, orientation and 3D IoU for different object classes can be found in Table 3. The object category *bed* performed best with the highest 3D IoU. We explain this by its large 2D bounding box and that surface planes can be easily detected, potentially helping to select the cuboid proposal. The object category *garbage bin* has the smallest yaw error, because they always face the wall plane and have clear edges in this dataset.

To compare our proposed detection method with SUN Primitive [35] and CubeSLAM [20], we select 400 images with visible ground planes and fully visible ground objects. SUN Primitive [35] calculates object orientation from vanishing lines, randomly samples object dimension, and uses the image lines to select the best cuboid proposal. CubeSLAM [20] samples cuboid corners on the 2D bounding box, recovers object dimension with camera height, and scores objects with image line. The average results on 3D object IoU together with the reported results from [20,35] are shown in Table 4. From the Table, we can see that our method achieves a higher 3D IoU, showing that our proposed two stage strategy of sampling and scoring is an effective method for considered the use-case.

Table 4. Object 3D IoU of different methods on SUN RGB-D dataset.

	SUN primitive [35]	CubeSLAM [20]	Ours
3D IoU	0.30	0.38	**0.4134**

4.2. Experiments–point-plane-object SLAM

4.2.1. Implementation details

The SLAM part of our system is built on top of monocular ORB-SLAM2 [31] and enriched with object and plane information. Considering the modality of the input, we estimate reliable 3D planes from depth images. However, this is not a fundamental bottleneck, because we only use the depth information at key-frames to estimate planes in order to score object cuboid proposals and add landmarks for semantic mapping. There is no additional fusion of depth information over frames happening in the front-end. It can be easily replaced by other RGB-based plane detection methods, such as PlaneNet [9].

In our system, the measurement and association of planes and objects are important for optimization. Therefore, we add a strict outlier rejection to data association, where the distance threshold is defined as $d = d_{\pi p} = d_{\pi o} = d_{\pi\pi} < 0.1m$, angle threshold $\theta = \theta_{\pi o} = \theta_{\pi\pi} < 8^{\circ}$, IoU threshold $IoU_{2D} > 0.5$, minimum number of in-plane points and in-object points $N = N_{\pi\pi} = N_{oo} > 15$.

The performance of point-plane-object SLAM is evaluated on ICL-NUIM dataset [36] because of its rich plane and object information. We adopt the root mean squared error (RMSE) of absolute pose error (APE) [37] as metric to evaluate the SLAM performance. We perform baseline experiments against ORB-SLAM2 [31] and benchmark our proposed point-plane-object SLAM system against state-of-the-art ORB-SLAM3 [38].

4.2.2. SLAM result and analysis

The ICL-NUIM virtual dataset [36] provides a home and an office scene with eight different sequences. The home scene includes sofas, chairs, vases, while the office scene includes monitors, tables, cabinets, etc.

We firstly show an example of our point-plane-object SLAM in living room sequences kt-2 in Figure 8. On the left side, the first two columns show the single frame object detection result on RGB images, as well as in 3D space, the camera trajectory and the reconstructed map are shown on the right. Taking points, planes, and objects into a unified bundle adjustment optimization framework, our system can achieve a good camera localization and build a sparse semantic map. In the reconstructed point-plane-object map, 5 objects and 9 planes are shown, the associated points, planes, and objects share the same color, while other unassociated planes such as ceilings are not displayed. Note that in the SLAM optimization, planes are represented as infinite planes, but for visualization purposes, we keep the point cloud model as the plane representation.

We also conduct experiments on other living room sequences. The mapping results with landmark information,

Figure 7. Single frame object detection results on SUN RGB-D dataset based on 2D and 3D constraints.

Table 5. Landmark information of our point-plane-object map on ICL NUIM dataset.

Sequence	Frame number	Key-frame number	Object number	Plane number
livingroom kt-0	1508	43	1	10
livingroom kt-2	880	57	6	16
livingroom kt-3	1240	120	0	14
office kt-0	1508	57	6	7
office kt-2	880	58	7	10
office kt-3	1240	62	1	8

including the key-frame number, object number, and plane number, are reported in Table 5. The localization performances with RMSE-APE are shown in Figure 9. In most sequences, our SLAM system can achieve a good localization and mapping result.

For comparison, we run the ORB-SLAM3 [38] in these sequences. To evaluate the benefit of introducing planes and objects, we also study two variation of our system, namely point-plane SLAM and point-object SLAM system. For camera trajectory comparison, we plot the results in Figure 10. Since the ORB feature based initialization [31] relies on RANSAC, we run each system 5 times in each sequence, and the mean value of RMSE-APE is reported in Table 6. We can observe that the added object and plane landmark constraints in our proposed method improve the camera pose estimation. We believe the improvement comes from two aspects: (1) the plane detection with dominant direction contributes to better pose estimation, because there are significant improvements between point SLAM and point-plane SLAM. (2) the strict outlier rejection method ensures a robust data association, and the constraints between different landmarks encourage a reliable and precise result. However, in sequence kt-1, both ORB-SLAM3 and our system do not work and are therefore not shown in the table. In sequence kt-3, when no objects are detected, our point-plane-object SLAM is reduced to point-plane SLAM, the results are different because of RANSAC initialization. For semantic mapping results, we show the point map, point-object map, point-plane map, and point-plane-object map in Figure 11.

For office sequences, we do similar experiments as for the living room sequences. Figure 12 shows the detection and mapping result on sequence kt-0 as an example, where 6 objects and 5 associated planes are visualized. The RMSE-APE between estimated camera poses and ground truth are compared in Figure 13, while the landmark information of the reconstructed map can be found in Table 5. We also compare our point-plane-object SLAM system with other SLAM systems, the camera trajectories are plotted in Figure 14, the localization errors are reported in Table 6, and the mapping results are shown in Figure 15.

Compared to the living room dataset, office sequences include objects that are not located on the ground. For example, the monitors are on the top of the desk. Since we sample object dimensions and calculate translation in local coordinates, we can detect objects even if they are not on the ground. We notice that in the third row of Figure 12, there are sometimes detection errors from the single image due to the clustered background or occlusion. These imperfect object measurements are optimized in the SLAM framework by sacrificing

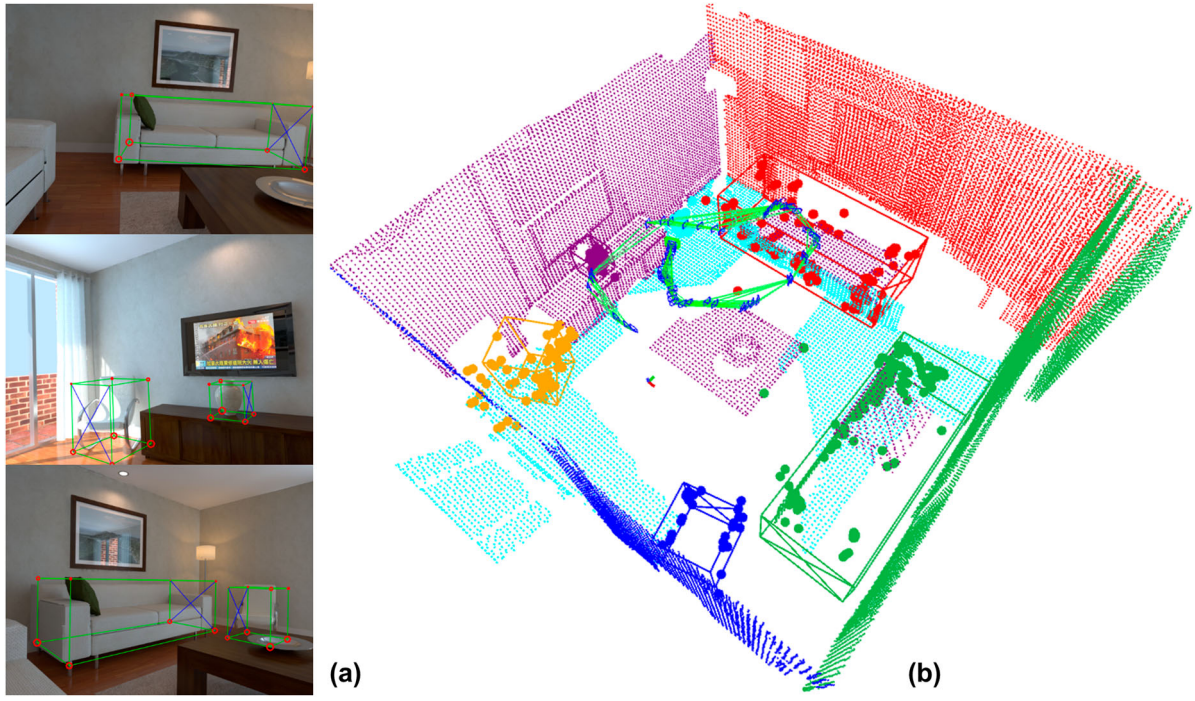

Figure 8. SLAM result on living room kt-2 sequence of ICL NUIM dataset, where the left part shows the object detection results in image and 3D space, while the right part shows the camera trajectory (green line) and the reconstructed point-plane-object map. The associated points, planes, and objects share the same color, and other unassociated planes such as ceilings are not displayed. we utilize the point cloud model as the plane representation only for visualization.

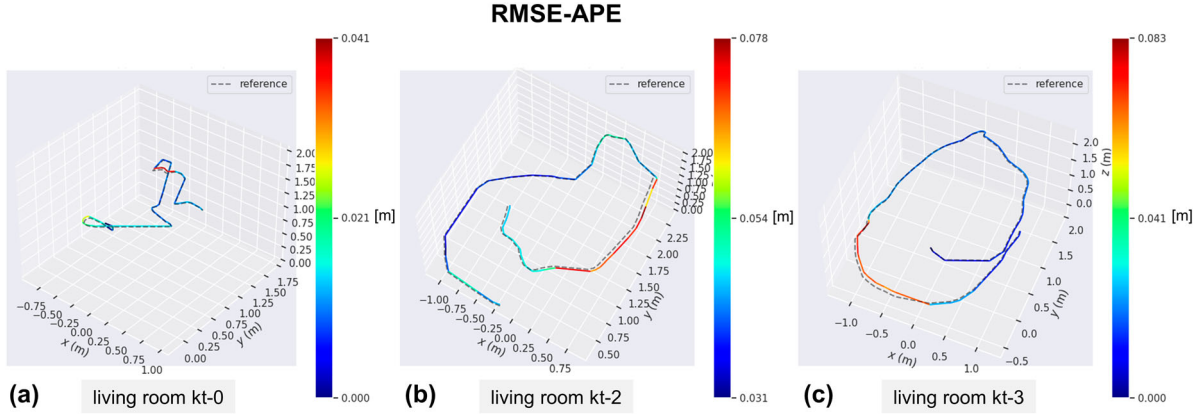

Figure 9. Evaluation results of various living room sequences of ICL NUIM dataset, these figures show the comparison of the estimated trajectories and corresponding ground truth.

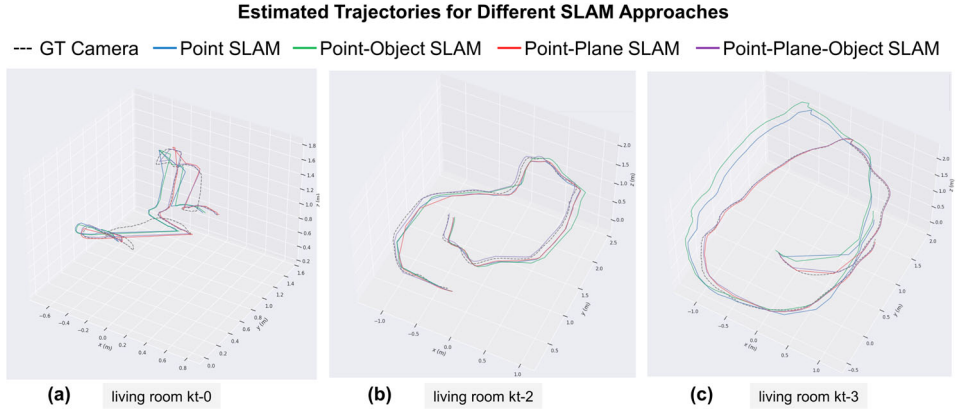

Figure 10. Comparison of the camera trajectories using different SLAM system in various living room sequences of ICL NUIM dataset.

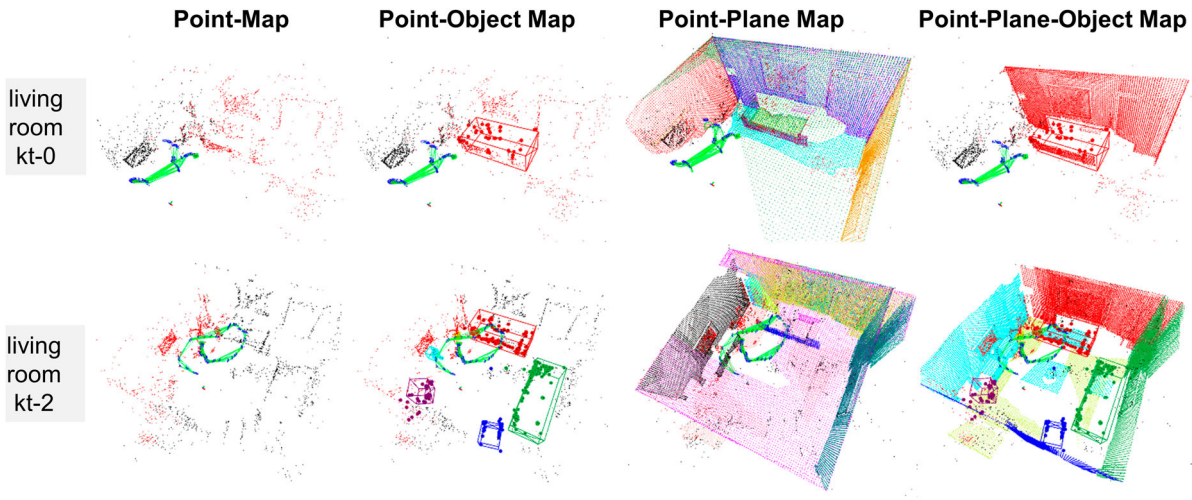

Figure 11. Comparison of the mapping results using different SLAM system in various living room sequences on ICL NUIM dataset. Note that in point-plane-object map, only associated planes are visualized.

Table 6. Evaluation results of root mean squared error of absolute camera pose error (RMSE-APE) on ICL NUIM dataset (cm).

Sequence	Point SLAM (ORB-SLAM2) [31]	Point SLAM (ORB-SLAM3) [38]	Point-plane SLAM	Point-object SLAM	Point-plane-object SLAM
livingroom kt-0	**0.39712**	0.75052	0.49278	0.40216	0.99162
livingroom kt-2	2.57732	2.17204	2.3053	2.0543	**1.77044**
livingroom kt-3	2.64344	3.16458	2.0557	2.55202	**1.75516**
office kt-0	12.83628	6.31114	**5.85264**	6.32358	6.83028
office kt-2	2.61264	1.86694	**1.42606**	5.3889	1.7747
office kt-3	3.92022	4.18088	**2.90242**	3.11014	3.66788

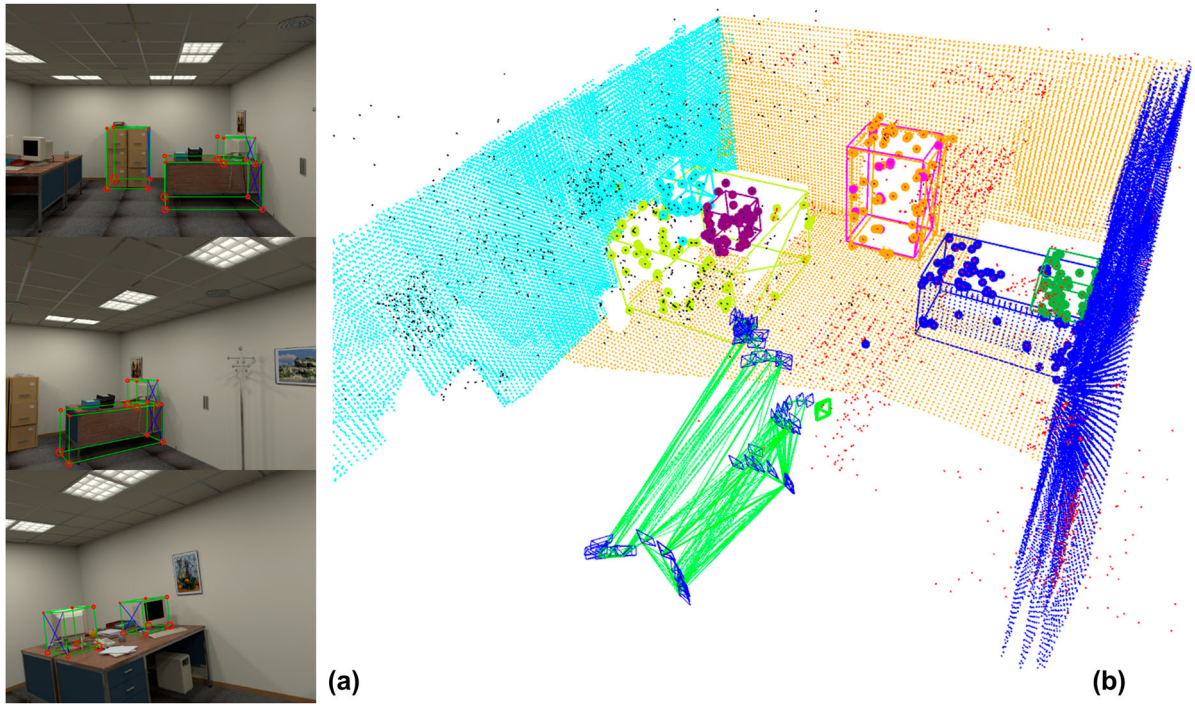

Figure 12. SLAM results on office kt-0 sequence of ICL NUIM dataset, where the left part shows the object detection results on image and 3D space, while the right part shows the camera trajectory (green line) and the reconstructed point-plane-object map.

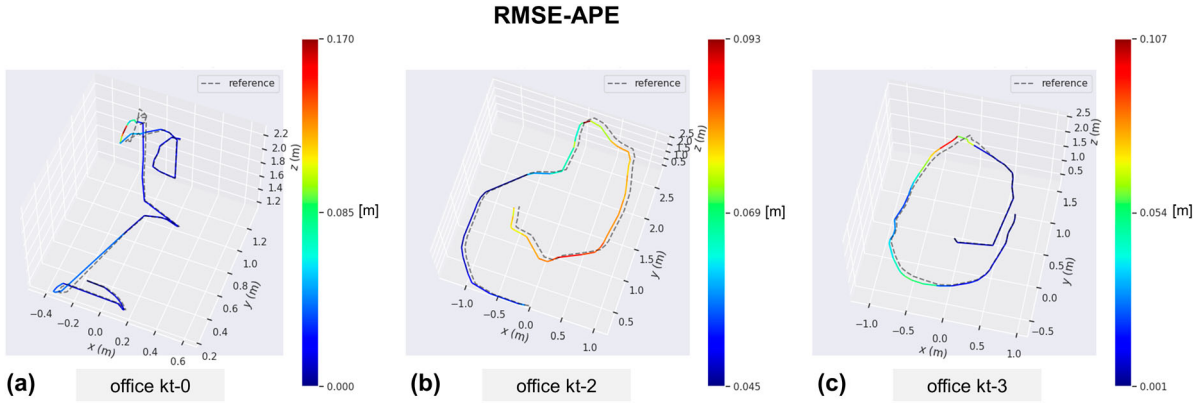

Figure 13. Evaluation results of various office sequences of ICL NUIM dataset, these figures show the comparison of the estimated trajectories and corresponding ground truth.

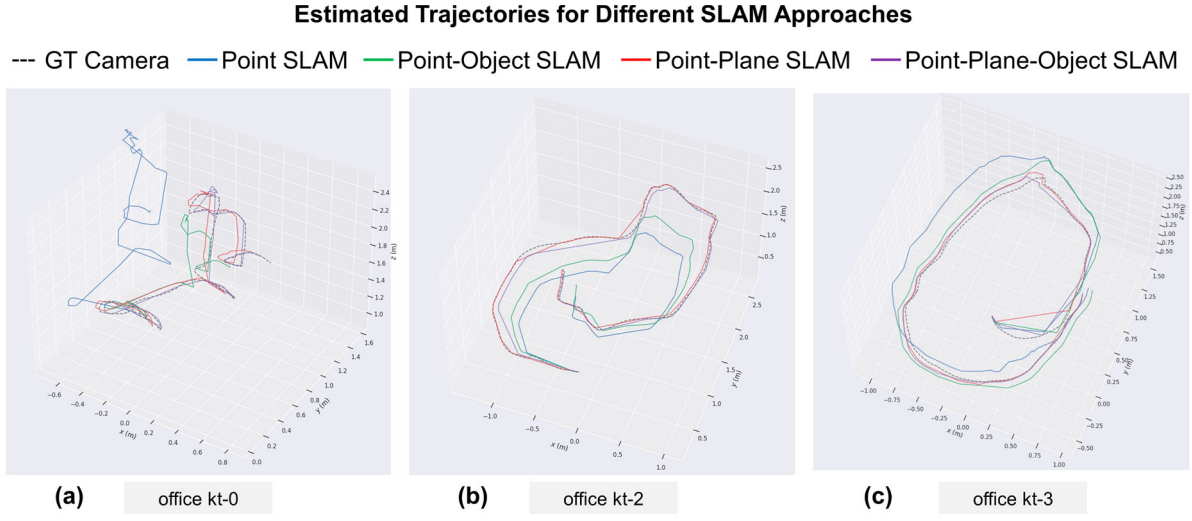

Figure 14. Comparison of the camera trajectories using different SLAM system in various office sequences of ICL NUIM dataset.

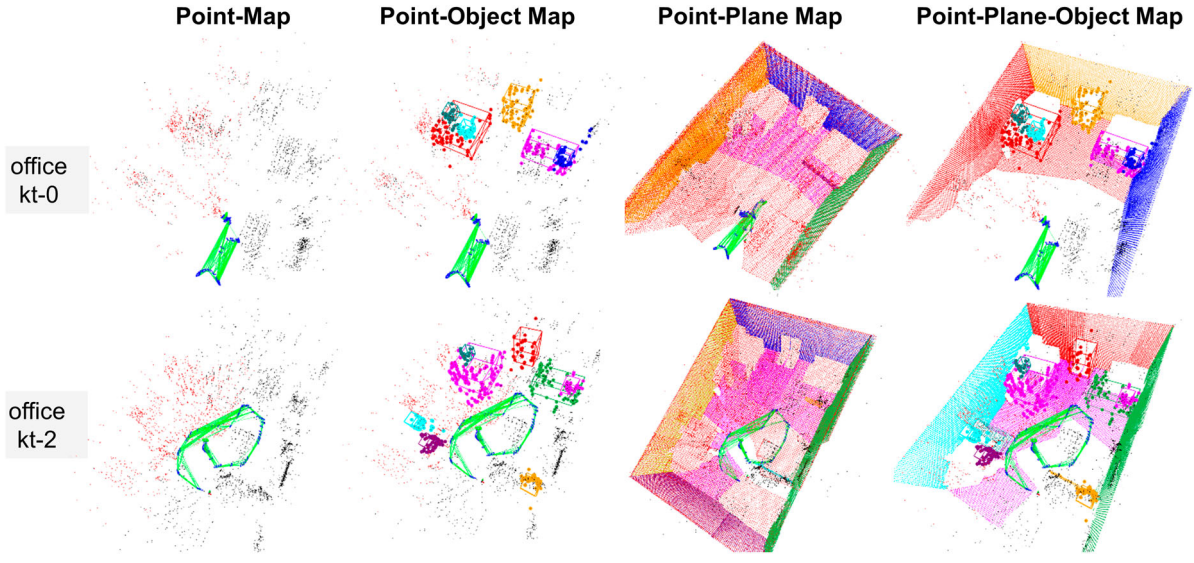

Figure 15. Comparison of the mapping results using different SLAM system in various office sequences on ICL NUIM dataset. Note that in point-plane-object map, only associated planes are visualized.

Table 7. Average runtime of different SLAM components.

Dataset	Tasks	Runtime (mSec)
Simple Image Preprocess	Plane Estimation	109.99
	Object Detection	97.386
	Edge Detection	18.831
Indoor ICL Room Dataset	Tracking Thread	17.886
	Point Only BA	63.240
	Point Plane Object BA	197.48

localization accuracies. Since there are only less than 10 objects but hundreds of feature points, even if objects do not improve the results, they won't seriously damage the system.

From the results, we can draw a similar conclusion as for the living room sequences: Our point-plane-object SLAM system can slightly improve the camera localization accuracy, and is able to reconstruct a semantic map with points, planes, and objects. If there are only few objects, our system will be reduced to point-plane SLAM.

4.2.3. Runtime analysis

Mapping and localization is usually conducted locally on a mobile robot, therefore runtime is a critical factor for real world systems with limited computing resources. All methods are implemented in C++ and evaluated on a laptop computer (i7-8565U 1.80 GHz CPU, 16 GB RAM, no GPU, Ubuntu 18.04). As shown in Table 7, there are several single image pre-processing steps. The plane estimation requires 109 ms, while the object detection needs 97 ms to select the best cuboid from 972 candidates. we run the 2D object detector offline and its inference time is not considered.

The SLAM run-time experiments are conducted on the ICL NUIM living room kt-2 sequence. The tracking thread, including feature detection, data associations, and camera pose tracking for each frame, can be executed in real-time (see Table 7). When a new key-frame is created, on average, 5 objects and 10 planes are involved in local bundle adjustment optimization. Compared to point only bundle adjustment, our system has higher optimizing costs because point-plane and point-object association have to be applied to many points, increasing the optimization time.

5. Conclusions

In this paper, we propose a structural SLAM framework with feature points, geometric planes, and semantic objects. For the single frame object detection part, objects are represented as cuboids and detected by a sample-and-score method. Firstly, object dimension and orientation are sampled to generate cuboid candidates. Secondly, 2D-3D fitting constraints are introduced to calculate the translations of these candidates. Then, the plane and feature constraints are adopted to score them in 3D and 2D space and the cuboid candidate with minimum error is selected as the final object proposal. For the SLAM part, the system is built on monocular ORB-SLAM2. In every key-frame, we detect planes and objects, track them, convert them into landmarks and integrate them into a unified bundle adjustment framework.

We evaluate our system on public datasets. The single frame object detection experiments are conducted on the SUN RGB-D dataset. The SLAM system is evaluated in living room and office scenes of ICL-NUIM dataset. Results show that our point-plane-object SLAM system improves camera pose estimation and is able to build a sparse semantic map with different landmarks, including feature points, planes and objects.

Since ORB feature based initialization may fail in some cases, future work should focus on a better initialization system with planes or other landmarks. Besides, our current method requires objects to be fully visible and ignores objects that are only partly observed in the image. However, considering these objects could provide additional information for the SLAM system.

Acknowledgments

The authors would like to thank Maximilian Ries for his kind support reviewing and discussing the work.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

This work was supported by Chinese Scholarship Council (CSC) Foundation [grant number 201906020168].

Notes on contributors

Benchun Zhou received his B.S. degree from School of Automation, Chongqing University, China in 2016 and his M.E. degree from the School of Automation Science and Electrical Engineering, Beihang University, China in 2019. He is currently a Ph.D. student at the Institute for Material Handling and Logistics, Karlsruhe Institute of Technology, Germany. His research interests include visual SLAM, computer vision, and field robotics.

Maximilian Gilles is working as research assistant at Institute for Material Handling and Logistics at Karlsruhe Institute of Technology (KIT). His research focuses on computer vision for stationary and mobile robots and its application in intralogistics.

Yongqi Meng received his B.S. degree from Beijing Institute of Technology, China, 2019, and M.S. degree from Karlsruhe

Institute of Technology, Germany, 2022. His research interests include computer vision, visual SLAM and SFM.

ORCID

Benchun Zhou http://orcid.org/0000-0002-8496-2674
Maximilian Gilles http://orcid.org/0000-0002-0528-5709

References

[1] Cadena C, Carlone L, Carrillo H, et al. Past, present, and future of simultaneous localization and mapping: toward the robust-perception age. IEEE Trans Robot. 2016;32(6):1309–1332.
[2] Li Y, Raza Y, Nikolas B, et al. RGB-D SLAM with structural regularities. Proceedings of IEEE International Conference on Robotics and Automation; 2021 May; Xi'an, China; p. 11581–11587.
[3] Xia L, Jiashuo C, Ran S, et al. A survey of image semantics-based visual simultaneous localization and mapping: application-oriented solutions to autonomous navigation of mobile robots. Int J Adv Robot Sys. 2020 May;17(3):1–17.
[4] Hosseinzadeh M, Latif Y, Reid I. Sparse point-plane SLAM. Proceedings of Australasian Conference on Robotics and Automation; 2017 Dec; Sydney, Australia; p. 148–155.
[5] Salas-Moreno RF, Newcombe RA, Strasdat H, et al. SLAM++: simultaneous localisation and mapping at the level of objects. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition; 2013 Jun; Portland, USA; p. 1352–1359.
[6] Holz D, Holzer S, Rusu RB, et al. Real-time plane segmentation using RGB-D cameras. RoboCup 2011: Robot Soccer World Cup XV; Berlin, Heidelberg: Springer; 2012; p. 306–317.
[7] Holz D, Behnke S. Fast range image segmentation and smoothing using approximate surface reconstruction and region growing. Proceedings of International Conference on Intelligent Autonomous Systems; 2012 Jun; Jeju Island, Korea; p. 61–73.
[8] Feng C, Taguchi Y, Kamat VR. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. Proceedings of IEEE International Conference on Robotics and Automation; 2014 May; Hong Kong, China; p. 6218–6225.
[9] Liu C, Yang J, Ceylan D, et al. PlaneNet: piece-wise planar reconstruction from a single RGB image. Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018 Jun; Salt Lake City, USA; p. 2579–2588.
[10] Liu C, Kim K, Gu J, et al. PlaneRCNN: 3D plane detection and reconstruction from a single image. Proceedings of IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2019 Jun; Long Beach, USA; p. 4450–4459.
[11] Yang F, Zhou Z. Recovering 3D planes from a single image via convolutional neural networks. Proceedings of European Conference on Computer Vision; 2018 Sep; Munich, Germany; p. 85–100.
[12] Le P, Košecka J. Dense piecewise planar RGB-D SLAM for indoor environments. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems; 2017 Sept; Canada: Vancouver; p. 4944–4949.
[13] Kaess M. Simultaneous localization and mapping with infinite planes. Proceedings of IEEE International Conference on Robotics and Automation; 2015 May; Seattle, USA; p. 4605–4611.
[14] Hsiao M, Westman E, Zhang G, et al. Keyframe-based dense planar SLAM. Proceedings of IEEE International Conference on Robotics and Automation; 2017 May; Singapore, Singapore; p. 5110–5117.
[15] Zhang X, Wang W, Qi X, et al. Point-plane SLAM using supposed planes for indoor environments. Sensors. 2019;19(17):3795.
[16] Yang S, Song Y, Kaess M, et al. Pop-up SLAM: semantic monocular plane SLAM for low-texture environments. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems; 2016 Oct; Daejeon, Korea; p. 1222–1229.
[17] Parkhiya P, Khawad R, Murthy JK, et al. Constructing category-specific models for monocular object-slam. Proceedings of IEEE International Conference on Robotics and Automation; 2018 May; Brisbane, Australia; p. 4517–4524.
[18] Rubino C, Crocco M, Del Bue A. 3D object localisation from multi-view image detections. IEEE Trans Pattern Anal Mach Intell. 2017;40(6):1281–1294.
[19] Nicholson L, Milford M, Sünderhauf N. QuadricSLAM: dual quadrics from object detections as landmarks in object-oriented SLAM. IEEE Robot Autom Lett. 2018;4(1):1–8.
[20] Yang S, Scherer S. CubeSLAM: monocular 3-d object SLAM. IEEE Trans Robot. 2019;35(4):925–938.
[21] Mousavian A, Anguelov D, Flynn J, et al. 3D bounding box estimation using deep learning and geometry. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition; 2017 Jul; Honolulu, USA; p. 7074–7082.
[22] Sünderhauf N, Pham TT, Latif Y, et al. Meaningful maps with object-oriented semantic mapping. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems; 2017 Sept; Canada: Vancouver; p. 5079–5085.
[23] Dengler N, Zaenker T, Verdoja F, et al. Online object-oriented semantic mapping and map updating. Proceedings of European Conference on Mobile Robots; 2021 Aug; Virtual; p. 1–7.
[24] Bowman SL, Atanasov N, Daniilidis K, et al. Probabilistic data association for semantic SLAM. Proceedings of IEEE International Conference on Robotics and Automation; 2017 May; Singapore, Singapore; p. 1722–1729.
[25] Hosseinzadeh M, Latif Y, Pham T, et al. Structure aware SLAM using quadrics and planes. Proceedings of Asian Conference on Computer Vision; 2018 Dec; Perth, Australia; p. 410–426.
[26] Liao Z, Wang W, Qi X, et al. RGB-D object SLAM using quadrics for indoor environments. Sensors. 2020;20(18):5150.
[27] Yang S, Scherer S. Monocular object and plane SLAM in structured environments. IEEE Robot Automat Lett. 2019;4(4):3145–3152.
[28] Li J, Koreitem K, Meger D, et al. View-invariant loop closure with oriented semantic landmarks. Proceedings of IEEE International Conference on Robotics and Automation; 2020 May; Virtual; p. 7943–7949.

[29] Wu Y, Zhang Y, Zhu D, et al. EAO-SLAM: monocular semi-dense object SLAM based on ensemble data association. Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems; 2020 Oct; Virtual; p. 4966–4973.
[30] Gupta A, Hebert M, Kanade T, et al. Estimating spatial layout of rooms using volumetric reasoning about objects and surfaces. Proceedings of Advances in Neural Information Processing Systems 23; 2010 Dec; Canada: Vancouver.
[31] Mur-Artal R, Tardós JD. ORB-SLAM2: an open-source SLAM system for monocular, stereo, and RGB-D cameras. IEEE Trans Robot. 2017;33(5):1255–1262.
[32] Coughlan JM, Yuille AL. Manhattan world: compass direction from a single image by bayesian inference. Proceedings of IEEE International Conference on Computer Vision; 1999 Sept; Kerkyra, Greece; p. 941–947.
[33] Kümmerle R, Grisetti G, Strasdat H, et al. G2o: a general framework for graph optimization. Proceedings of IEEE International Conference on Robotics and Automation; 2011 May; Shanghai, China; p. 3607–3613.
[34] Song S, Lichtenberg SP, Xiao J. SUN RGB-D: a RGB-D scene understanding benchmark suite. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition; 2015 Jun; Boston, USA; p. 567–576.
[35] Xiao J, Russell B, Torralba A. Localizing 3D cuboids in single-view images. Proceedings of Advances in Neural Information Processing Systems 25; 2012 Dec; Lake Tahoe, USA.
[36] Handa A, Whelan T, McDonald J, et al. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. Proceedings of IEEE International Conference on Robotics and Automation; 2014 May; Hong Kong, China; p. 1524–1531.
[37] Grupp M. EVO: python package for the evaluation of odometry and SLAM; 2017. Available from: https://github.com/MichaelGrupp/evo.
[38] Campos C, Elvira R, Rodríguez JJG, et al. ORB-SLAM3: an accurate open-source library for visual, visual–inertial, and multimap SLAM. IEEE Trans Robot. 2021;37(6):1874–1890.