An Enhanced Visual SLAM Supported by the Integration of plane Features for the Indoor Environment

Bingxin Zi
School of Computing
Ulster University
Belfast, United Kingdom
zi-b@ulster.ac.uk

Haiying Wang
School of Computing
Ulster University
Belfast, United Kingdom
hy.wang@ulster.ac.uk

Jose Santos
School of Computing
Ulster University
Belfast, United Kingdom
ja.santos@ulster.ac.uk

Huiru Zheng
School of Computing
Ulster University
Belfast, United Kingdom
h.zheng@ulster.ac.uk

***Abstract*—This paper presents an enhanced indoor RGB-D simultaneously localisation and mapping (SLAM) system based on the integration of plane and point features. A new method was proposed to register each point feature to a corresponding plane feature and then modify its position accordingly. The plane features are parallelly extracted from depth data sources and used jointly to solve the camera pose with point features. Plane features are stored on the map as the same as point features. Both point and plane features are used for backend optimisation, where the weights associated with features can be dynamically updated. At the same time, the on-plane feature points are fixed during the optimisation. The proposed method has been tested with open-source benchmarks, including the scenarios with or without a structured environment. Experiment results demonstrated that the proposed algorithm performs better than other widely cited visual SLAM systems in some structured environments.**

Keywords—SLAM, Visual SLAM, Geometry Information, Plane Feature

I. INTRODUCTION

The concept of simultaneous localisation and mapping (SLAM) is to estimate the sensor's location and orientation (pose) and construct a map of the surrounding environment simultaneously. It has received massive attention during the last 20 years and has been applied to various applications, including autonomous robots, self-driving, augmented reality, and virtual reality[1]. The primary sensors that can implement visual SLAM include monocular cameras, stereo cameras and RGB-D cameras. In this type of SLAM, point features are commonly used to solve robot poses and construct a map that describes the environment. Thanks to the increasing computational power available over the last decades, many visual SLAM algorithms[2] have achieved real-time performance.

While in the field in the presence of structural information, there are other geometric features, like lines and planes, that can be used, besides point features, to improve the robustness and accuracy of visual SLAM systems[3][4][5][6]. One of the primary sensors used in a visual SLAM system is an RGB-D camera. Apart from RGB images, this type of camera can provide a corresponding depth map that reflects the structural geometry information of the environment. Because not every plane could be detected in the scene, Zhang, Wang, Qi, Liao and Wei[7] exploited the detected plane and generated supposed planes to add more constraints to the system. But they assumed that planes are either parallel or perpendicular to each other in the indoor scene. Ma and Cremers[3] introduced a direct SLAM that labels each pixel to a plane with probabilities. Their work required acceleration from GPU devices to achieve real-time performance. Besides, Zhang, Liao, Qi and Wang[8] developed a SLAM algorithm based on a stereo camera in which plane features were generated using the intersecting lines. Li et al. [9] used line features to support key point selection in a direct monocular SLAM system. A study conducted by Sun, Yuan, Zhang and Duan[4] utilised plane-edge fusion and probabilistic plane fitting. Li, Yunus, Brasch, Navab, and Tombari[5] decoupled the rotation estimation and translation estimation and used both plane features and line features. While various degrees of success has been achieved, the studies mentioned earlier are facing issues such as excessive filter states and over-parameterised features. Besides, for the aforementioned SLAM systems that have already used geometric features, they just parallelly add geometric features to the optimisation system without discussing the relationship between different types of features. This research aims to find an visual SLAM system that utilises plane features and explores the relationship between different kinds of features.

This paper proposed a novel approach to the integration of indoor plane features with the point-based RGB-D SLAM system. Point features are extracted from RGB images in each frame, while plane features are extracted from depth images. Each point feature is paired with a plane feature, and its location is adjusted by considering that the point locates on that plane. A cost function is then constructed following the constraints of both the plane and point features. The cost function is used for optimising the estimated poses from each frame and for optimising the joint point-plane bundle adjustment as well. During the optimisation, the point features that are registered with the plane will be set fixed. The plane features are stored on the map as land markers in the same way as point features. The main contributions of this paper are summarised as follows:

- Proposing a visual SLAM framework for the indoor structural environment, in which a new method to combine point and plane features has been proposed.
- Developing methods for pairing a point with a plane feature and adjusting the point's position by the plane feature.
- Based on the point-plane relationships, we proposed a novel approach enabling both point and plane features to participate in the optimisation process

A series of experiments have been carried out on the open-source benchmarks[10][11]. Results were evaluated in terms of the errors of camera trajectory, and they show that the proposed method could achieve better performance than state-of-the-art algorithms in an environment with structure features.

The remainder of this paper is organised as follows. Section II overviews the related works of visual SLAM systems that utilise structural information. Section III outlines the proposed methods used in this study, followed by the presentation of the experimental results in Section IV. The paper concludes with a discussion of the experiment results, current limitations, and future research.

II. RELATED WORKS

There has been a growing interest in introducing planes as a kind of feature to visual SLAM in the indoor environment. For example, Gee, Chekhlov, Mayol-Cuevas, and Calway [12] discovered the state space for plane features in a filter-based SLAM system. The introduction of planes would reduce the number of points' states. Kim and Coltin[13] introduce the orthogonal plane into the linear Kalman Filter SLAM system as well. Servant, Marchand, Houlier and Marchal [14] applied planes in the Extended Kalman Filter (EKF) SLAM. But the filter-based SLAM system requires a large number of states, and it fails to handle the long-term drift errors well. Recently, a variant of the filter-based SLAM was proposed [15], which is based on data fusion of light detecting and ranging sensor (LiDAR), inertial measurement units (IMU), and Cameras. The plane landmarks are extracted from the LiDAR pipeline and tracked as a member of status. In addition to the large number of state spaces, the multiple sensors' calibration accuracy could also affect their work, as they concluded.

In the way of graph-based SLAM, [3][4][5][6] used plane features in their studies. Sun, Yuan, Zhang and Duan[4] introduced probability to avoid the noise influence in fitting a plane. Trevor, Rogers and Christensen[6] used both plane and line features. Li, Yunus, Brasch, Navab, and Tombari[5] decoupled translation and rotation in estimating the camera pose. Ma and Cremers [3] extracted planes from RGB-D images and then used the expectation-maximisation (EM) framework to decide if a pixel belongs to a detected plane. The EM frame provides a soft approach to associating a point with a plane compared with a hard label. To avoid the singularities of the plane representation, they forced the angle of the plane feature to fall into $(-\pi, \pi]$. In addition, the system would still suffer from the changing illumination problem as it is an extension of directly SLAM. Besides, their work needs work with a GPU support machine to accelerate probability computation. That means it would be hard to be implemented on the mobile platform. The SALM system introduced by Hsiao, Westman and Kaess[18] detected small planes from RGB-D data and merged them into larger planes. A modified point-to-plane Iterative closest point (ICP) algorithm was used to estimate frame pose by considering photometric, geometric, and plane matching errors. Due to the high computational cost, their algorithm also needs to run on a GPU to reach a near real-time performance. Furthermore, the communal problem of the above studies is they all used the over-parameterised Hessian form to represent the Plane feature. In the meantime, the [16] and [17] define a plane with plane azimuth θ, plane elevation ϕ and the distance d from the origin to the plane norm. But the azimuth and elevation way also need to be transformed to Hessian form during the optimisation procedure. In addition, the radian representation suffers from noise, which usually fluctuates largely and results in failed plane association [17]. In addition to the detected points and planes, the SALM system developed by Zhang, Wang, Qi, Liao and Wei[7] can generate some supposed planes by utilising the edge of the detected real planes. The contours of the real planes are used as the first criterion for associating the plane features, followed by associating planes with an angle threshold of 30 degrees. Their algorithm works under the assumption that planes are either perpendicular or parallel to each other in the scene. However, the planes do not necessarily support this assumption except the floor, ceiling, and walls. In the recent work by Zhang et al.[8], a visual SLAM system was introduced, which detects planes from stereo camera based on a novel plane detecting algorithm. It is capable of estimating planes based on the intersecting lines. However, as they concluded, the inaccurately estimated plane features still exist and remains a big challenge to their system.

To overcome the above-discussed challenges, including excessive states, plane feature over parameterisation, GPU requirements and other issues and efficiently utilise plane features and the point-plane relationships, we proposed a graph visual SLAM system that can detect plane features, parameters plane with three variables, register point feature to plane feature, modify point landmark's location and jointly

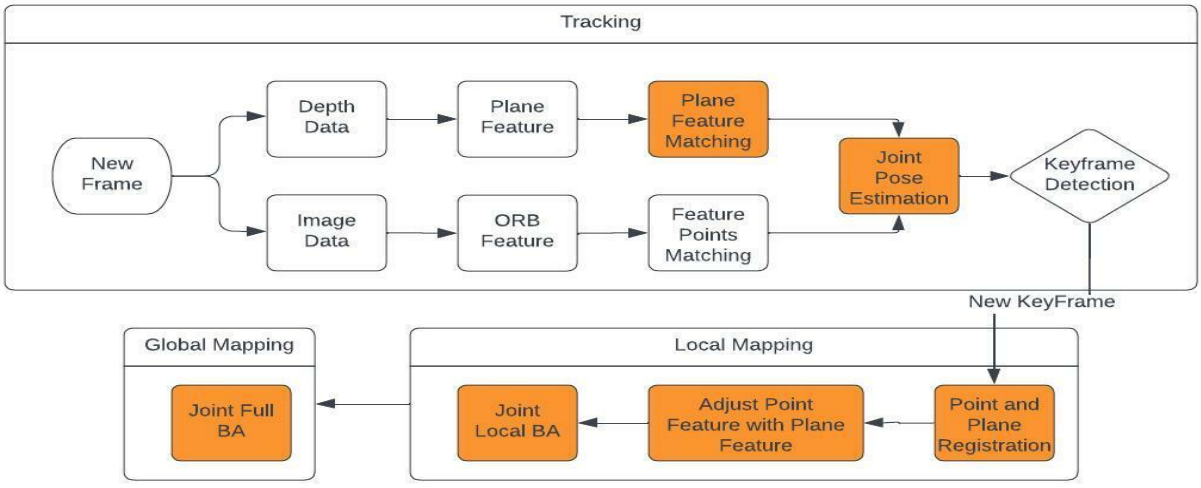


Fig. 1. The flow structure of the proposed visual SLAM system, which consists of a tracking module and two mapping modules.

optimise the camera pose without GPU and Manhattan hypothesis.

III. METHODOLOGY

Our approach is a graph-based visual SLAM system that cooperates with an RGB-D camera to detect point and plane features. In each frame, small and isolated planes, which are usually seen at an edge, will be removed if they are well separated from large planes due to the influence of noise. At the same time, the adjacent planes are merged into a large one. Each point feature close to a plane is considered on that plane. In the map, the point's location is then modified accordingly. When associating two planes, the system considers both the plane's geometric location and the point-plane pair attributions. In the tracking part, along with point features, plane features are used to estimate and optimise the camera pose by constructing a point-plane joint least-square function. In addition, all the plane features are also applied to local and global backend optimisation. Our modified SLAM system has been developed and implemented based on the ORBSLAM2 framework [19].

A. System Overview

As a graph SLAM system, a tracking module is responsible for processing adjacent frames to calculate the current camera dynamics. All the features extracted from each keyframe and solved dynamics are stored in the mapping module, used to construct and maintain the feature map, optimise the camera poses and check for loop closure to reduce the long-term drift.

Fig.1 shows the structure of our system, which consists of a tracking module and two mapping modules. Since the proposed system has been implemented based on the framework provided by ORBSLAM2[19], the main modification modules are highlighted with the colour orange. The tracking module extracts point features from the 2D image data source, and plane features are derived from the depth data source. The extracted features are used for pairing with map landmarks. Both point and plane features are utilised to calculate the motion between two adjacent frames. While plane features are not involved in the keyframe detection [19]. The plane features are added to the map once a new keyframe is detected. After that, the point landmarks are registered to nearby plane landmarks, and their locations are modified by that plane. During the backend optimisation, both the point and plane features are used to construct the optimising function jointly. We introduced dynamic weight to adjust their contributions. The point features registered with the plane will be set fixed in the optimisation process. The joint optimisation will be explained in section E. The point and plane features are maintained within the entire system life cycle. Each module will be described in detail in the following sections.

B. Plane Feature Extraction and Representation

Parallel to extracting 2D point features from the RGB data source, all the 3D plane features are extracted from the depth data source. Each depth frame is segmented by the region growing algorithm from the PCL library [20]. Then the RANSAC algorithm [21] is applied to each segment to fit a plane. Small planes that are close to each other will be merged into large planes instead.

To avoid the singularity issue and over-parameterised issue discussed above [3][6][16][17], in this paper, a plane feature is represented following the closest point (CP) definition [22] that is a point that resides on the plane and is closest to the origin. The CP definition only takes three variables to describe the plane feature. Then the plane representation is defined as:

$$\Pi = nd \tag{1}$$

$$\begin{bmatrix} n^T \\ d \end{bmatrix} = \begin{bmatrix} \Pi^T / \|\Pi\| \\ \|\Pi\| \end{bmatrix}. \tag{2}$$

Where n is the plane norm vector and d is the distance scalar. Π represents the closest point which is the product of n and d, and $\|\Pi\|$ is the norm of Π.

The corresponding plane transform is performed using the equation (3):

$$\begin{bmatrix} n^L \\ d^L \end{bmatrix} = \begin{bmatrix} R_R^L & 0 \\ -p_L^{R\,T} & 1 \end{bmatrix} \begin{bmatrix} n^R \\ d^R \end{bmatrix}. \tag{3}$$

Where the superscript *L* indicates a local frame and *R* indicates the reference frame. The R_R^L stands for the rotation matrix rotates from the reference frame to the local frame, and P_L^R stands for the position of the location frame seen from the reference frame [22].

C. Point-Plane registration

In the common way of [3][6][16][17], the plane and line features have just been added to the system parallel to the point feature. Under the Manhattan assumption, plane features may introduce parallel or perpendicular constraints to the system. But in most cases, these geometric features serve the same function as point features. Usually, the point number is typically one to two orders of magnitude higher than the plane number, which would overwhelm the plane contributions during optimisation. To avoid this and explore more inner connections between the point and plane features, the proposed system prefers to connect the point feature with the plane feature.

Before creating a new keyframe and adding it to the map, each candidate point landmark in that new keyframe is paired with a plane landmark by calculating the Euclidean distance between them. Any candidate point landmark close to a plane landmark will be registered to that plane. Once a point-plane pair relationship is established, the candidate map point is replaced with the intersection point of the vector, which is point to the feature point from the origin, and the paired plane. In the pose optimisation, local bundle adjustment and global bundle adjustment procedures, the point-plane relationships indicate that the point should stay on that plane, so these points will be set fixed during the optimisation procedure.

In the case that feature points are located on edges and corners, a point belonging to the foreground plane may be incorrectly assigned to the background plane due to the interruption from sensor noise. To avoid this, the system prefers to get rid of the feature points with a high level of uncertainty. We selected the feature points from a smooth area. That is to say, the depth of a feature point should be close to its nearby pixels. So, the depths of each feature point and its surrounding patch are checked. Only the points with a tiny depth deviation of the surrounding patch will be accepted. In this system, the depth deviation threshold is set to 0.05 since we observed that the edge and corner point usually have a more than 0.1 depth deviation.

D. Plane feature association and management

When associating a local frame plane with an existing map plane landmark, the plane in the local frame must first be transformed to the map frame by Equations (1) − (3). Then the following Euclidean residual equation is applied to measure the CP distance between the transformed plane and the plane landmark.

$$error(\Pi)=\left\|\Pi^m-\widehat{\Pi}^m\right\|_2 \tag{4}$$

Where Π^m represents the plane landmark on the map, $\widehat{\Pi}^m$ is the transformed plane.

Besides the naïve plane Euclidean distance, point features are also involved in the plane association procedure. After the local plane detection, the map points will be transformed into the local frame. When the distance from a transformed point to a local plane is less than a certain threshold, this map point is registered with that local plane. Each point can only be registered to one single plane, and their point-plane pair relationships are recorded. When a close CP plane pair is firstly detected by Equation (4), then their point-plane pairs are compared. Ideally, two individual planes should not contain any common point features (except the points on plane intersection). This kind of measurement could help identify two individual but close planes. For example, a photo frame hung on a wall. This paper applies a small common point feature number threshold of 10 since we observed that most individual planes contains less than 10 common points.

When a new local plane did not associate with any map plane landmark under both the Euclidean distance criterion and common point-plane pair criterion, it should be considered a new plane feature to be added to the feature map.

E. Pose optimising and Weighted Point-Plane Bundle Adjustment

In the point-based graph SLAM, the pairs of 2D feature point observations and 3D map points are used to construct a reprojection error function and to estimate the robot motion by minimising the error function :

$$E_{pt}=\sum_{i=1}^{m}\left\|u_i-p\left(h(T,P_i)\right)\right\|_2. \tag{5}$$

Where u_i is the feature point's 2D observation, T is the camera pose, P_i is the paired 3D map point, $h()$ stands for the transform function and $p()$ represents the camera projection function.

With Equation (1) − (4), the plane error function and the joint pose optimisation function could be defined. The plane error function is defined as:

$$E_\pi=\sum_{i=1}^{m}\left\|\Pi_i^m-H(\Pi_i^l)\right\|_2. \tag{6}$$

Where Π_i^l denotes the plane feature detected in the local frame, Π_i^m is the matched plane landmark, and $H()$ represents the plane transform function (3).

The corresponding joint pose estimation function is defined as:

$$T_m^c=\underset{T_m^c}{argmin}\left(\sum w_p*Hu(E_P)+\sum w_\pi*Hu(E_\pi)\right). \tag{7}$$

Where $Hu()$ is the Huber robust kernel. w_p is the weight of point error and w_π is the weight of plane error.

The weight of the plane and point could be calculated as:

$$w_p=\frac{E_\pi}{E_P+E_\pi}, \tag{8}$$

$$w_\pi=\frac{E_p}{E_P+E_\pi}. \tag{9}$$

Where w_p and w_π denote the weights associated with the point feature's error E_p and plane feature's error E_π respectively.

Usually, both the point feature and the camera pose are adjustable in the optimisation procedures, but in this paper, since some point features are considered located on a plane, those points are set fixed in the optimisation process to matin the "point on the plane" constrains.

The optimisation process was implemented using the G2O library [23]. The optimisation was iterative, and the edges with a significant error were removed from the iterations.

TABLE I. THE TRAJECTORY EVALUATION RESULTS BETWEEN THE PROPOSED METHOD AND POINT-BASED SLAMS. THE BEST PERFORMANCE IS HIGHLIGHTED IN BOLD. THE MARK X INDICATES THE RESULT DOES NOT APPLY TO THEIR ORIGINAL PUBLICATION.

	Proposed Method	***ORB-SLAM2***[19]	***Elastic-SLAM***[24]
fr3/stf	**0.013**	0.020	**0.013**
fr3/stn	**0.011**	0.019	0.015
fr3/ntf	**0.047**	0.076	0.074
fr3/ntn	0.020	0.023	**0.016**
lr-kt0	0.032	0.026	**0.009**
lr-kt1	0.010	**0.008**	0.009
lr-kt2	**0.014**	0.023	**0.014**
lr-kt3	**0.008**	0.022	0.106
of-kt0	**0.034**	0.036	X
of-kt1	**0.039**	0.047	X
of-kt2	**0.035**	0.039	X
of-kt3	**0.021**	0.046	X

TABLE II. THE TRAJECTORY EVALUATION RESULTS BETWEEN THE PROPOSED METHOD AND SLAMS WITH GEOMETRY INFORMATION. THE BEST PERFORMANCE IS HIGHLIGHTED IN BOLD. THE MARK X INDICATES THE RESULT DOES NOT APPLY TO THEIR ORIGINAL PUBLICATION.

	Proposed Method	**PlanarSLAM[5]**	**CPA-SLAM[3]**	**Plane-edge-SLAM[4]**	**L-SLAM[13]**	**LPVO[26]**	**PL-SLAM[25]**
fr3/stf	0.013	X	X	0.014	0.212	0.17	**0.009**
fr3/stn	**0.011**	X	X	X	0.156	0.11	0.013
fr3/ntf	**0.047**	X	X	X	X	X	fail
fr3/ntn	0.020	X	0.016	**0.015**	X	X	0.021
lr-kt0	0.032	**0.006**	X	X	0.012	0.015	0.008
lr-kt1	**0.010**	0.015	X	X	0.027	0.039	**0.010**
lr-kt2	**0.014**	0.020	X	X	0.053	0.034	0.019
lr-kt3	**0.008**	0.012	0.028	X	0.143	0.102	0.012
of-kt0	0.034	0.041	X	X	**0.020**	0.061	**0.020**
of-kt1	0.039	0.020	X	X	**0.015**	0.052	0.022
of-kt2	0.035	**0.011**	X	X	0.026	0.039	0.022
of-kt3	0.021	0.014	X	X	**0.011**	0.030	0.018

IV. EXPERIMENTS

The plane feature is common within manufactured environments, including roads, tunnels, walls, ceilings, and floors. Our research scenario is focused on the indoor environment, so the most common plane feature would be floor, wall and ceilings. The experiments were carried out on the open-source TUM RGB-D benchmark[10] and ICL-NUIM benchmark[11]. The TUM benchmark provides multiple real-world scenarios and the corresponding ground truth obtained by an external motion capture system. In order to comprehensively evaluate the system, both structured scenarios (fr3/structure_texture_far known as fr3/stf and fr3/structure_texture_near known as fr3/stn) and no structured scenarios (fr3/nostructure_texture_far known as fr3/ntf and fr3/nostructure_text_near_withloop known as fr3/ntn) were selected. Since the proposed algorithm was developed based on the ORBSLAM2 frame, which could fail to track in the low texture area, we did not pick the non-textured scenarios. The texture structure scenario has a zig-zag wooden panel structure located in the centre of the environment. The sensor was moved by half a meter height around that structure. The wooden panel was wrapped in a colourful plastic foil to provide a strong texture for point features. While in the no structure scenario, the camera was facing the floor for the all-time. The ICL-NUIM dataset is a synthetic dataset that provides eight tracks of a living room scene and an office room scene, containing several planes that could be utilised. Every track (known as lr-kt0 to lr-kt3, of-kt0 to of-kt3) is tested.

The proposed method was compared with other state-of-the-art SLAM algorithms including ORBSLAM2 [19], Elastic-SLAM[24], CPA-SLAM [3], Plane-edge-SLAM [4], Planar-SLAM[5], PL-SLAM [25], L-SLAM [13] and LPVO[26]. The ORBSLAM2[19] is one of the most popular point-based visual graph SLAM. The Elastic-SLAM [24] is a frame-to-model tracking SLAM without any backend graph optimisation. The CPA-SLAM [3], Plane-edge-SLAM [4], and L-SLAM [13]are among the most cited plane related visual SLAM in recent years. On the other hand, the PL-SLAM [25] uses point and line features in their system. And LPVO [26] explores both plane features and line features. Planar SLAM[5] is a Planar SLAM that utilises point, line and plane features, decouples the translation and rotation estimation and applies Manhattan constraints if applicable.

The trajectory evaluation tool EVO [27] calculated the absolute pose error (APE). To alleviate the effect of randomness, each trajectory was tested ten times, and the mean values were provided. All experiments were carried out with an Intel Core i9 CPU(2.90GHz) without GPU acceleration.

V. RESULTS AND DISCUSSION

We first compared the performance of our approach with SLAM systems [19][24], which do not consider geometry information, followed by comparisons with state-of-the-art Visual SLAMs that utilise geometry information like planar features [3][4][5][13] and line features [25][26]. The ORBSLAM2 system was implemented along with the proposed method, while the results of other SLAM systems were taken from their original publications, respectively.

A. Comparison with Point-based SLAM

Table I. Shows the evaluation results of the proposed method and other algorithms without using geometry information, i.e., ORB-SLAM2[19], Elastic-SLAM[24]. The mark X indicates that the result does not apply to their original publication. As illustrated in Table I., our proposed method outperforms other algorithms in most scenarios. It demonstrates the performance gain by introducing geometry information and adding more constraints to the pose estimation procedure.

However, the Elastic SLAM performs better for the fr3/ntn track. In this scenario, the camera was facing the floor the whole time. So there was not too much geometry information that could be detected, except for the only floor plane. The Elastic-SLAM is a dense model-frame SLAM system which frequently checks whether there is a global or local closing loop in the current frame and uses pixel intensity information to optimise the pose results. It may partially explain why it could do better when the camera is close to the floor and the details of each frame are clearer. However, when the camera is away from the floor and the detail information becomes coarse, our method can outperform Elastic SLAM, as demonstrated in the fr3/ntf sequence.

While in the synthetic dataset of ICL-NUIM[11], the proposed algorithm achieved the best results on more than half-tracks. ORBSLAM2 and Elastic-SLAM achieved the best results on one track, respectively. The Elastic-SLAM

TABLE III. THE POINT ERROR AND PLANE ERROR IN POSE OPTIMIZATION AND BUNDLE ADJUSTMENT

	Pose Optimization				**Bundle Adjustment**			
	Avg Point Error in pixel	***Avg Point Number***	***Avg Plane Error in pixel***	***Avg Plane Number***	***Avg Point Error in pixel***	***Avg Point Number***	***Avg Plane Error in pixel***	***Avg Plane Number***
lr/kt0	26.7	351.3	1.8	2.7	1.8	3495.5	2.8	4154.7
lr/kt1	18.0	272.1	9.5	3.1	1.2	3026.7	1.8	1701.2
lr/kt2	26.2	377.8	5.0	3.4	1.6	6834.9	3.5	2079.0
lr/kt3	29.6	303.2	6.9	3.0	1.8	1452.2	4.0	1500.0
of/kt0	35.6	369.6	3.0	3.8	1.1	10425.5	3.0	8882.9
of/kt1	26.1	348.4	2.8	2.7	1.6	496.7	3.4	1708.8
of/kt2	30.4	387.7	4.8	3.9	1.9	3591.6	3.7	2256.9
of/kt3	23.0	333.9	3.4	2.7	2.0	1553.8	3.6	1738.8

performed better in the lr-kt0 track. It could be because the camera was facing the white wall at some frame in those two scenes. It is similar to the situation discussed above. While in the other office scenes of of-kt1, the blocks on the ceiling and floor provide enough point features, which give enough constraints for ORBSLAM2. But the proposed method achieved a close result as well.

B. Comparison with SLAM using geometry features

Table II. shows the evaluation results of the proposed methods and other state-of-the-art SLAM algorithms that utilise plane features and line features in terms of APE. The mark X indicates that the result is not applicable from their original publication, and 'fail' indicates the SLAM could not complete the tracking and mapping job. The fr3/stf sequence results show that PL-SLAM performs best while our methods reached 2nd place. However, in the sequence of fr3/stn, the proposed method performed better than PL-SLAM and became the best. In the sequence of fr3/ntf, only the proposed method's results and PL-SLAM's results are applicable. PL-SLAM, however, fails to complete the sequence because ambiguity was detected[25]. In the last TUM sequence of fr3/ntn, CPA-SLAM and Plane-edge-SLAM achieve better performance. A close look at the sequence suggests that there were not too many plane features but only a floor plane. The proposed method could not find extra constraints, while the Plane-edge-SLAM could utilise edge information. CPA-SLAM is a dense direct visual SLAM algorithm that labels each RGB-D point with a probability of belonging to a plane. As a result, more constraints were added to the CPA-SLAM algorithm. It entails a high computational cost which requires GPU acceleration.

While in the synthetic dataset of ICL-NUIM[11], the proposed algorithm achieved the best result in three sequences. The Planar-SLAM achieved an outstanding result for the first living room scene (lr-kt0). As discussed above, the camera was facing a white wall in some frames, so there were not too many plane features that could be detected. But Planar-SLAM also utilised line features, bringing extra constraints to the system. The L-SLAM achieved the best result in three office scenes of of-kt0, of-kt1 and of-kt3. It could be because, in those scenes, the camera captured five planes (front, left and right walls, ceiling and floor) in the first frame. A solid MW constraint was established initially, which brings extra constraints.

C. Plane Feature Error Analysis

Table III. demonstrates the point and plane feature's error analysis during pose optimisation and bundle adjustments. In this paper, the plane is represented by CP points, and the errors of CP points are converted to pixel errors for the convenience of comparison. As shown in column 1 and column 3, it is obvious that the plane features' error is much less than the point feature's error. While at the same time, the number of planes is much less than the number of points. It led the contribution of the plane easily overwhelmed by the points. As discussed above, the proposed method's points on a plane were set fixed during the optimisation procedures. It brings the consequence that those fixed points could have a larger error than the normal points. But, If a feature point is very close to a plane. It is more reasonable that that point locates on that plane rather than floating in the air. Fixing those points on a plane could help avoid falling to the local minimum in the optimisation procedure. For those reasons, the higher error of fixed points is accepted in the proposed method.

D. The point on the plane constrains

Fig. 2 provides examples of the mapping difference between the proposed method and the ORBSLAM2 in ICL-NUIM living room sequences. The maps show that in our proposed method, the feature points have usually been constrained on a plane if it is close to that plane, as the red box indicates. While in the ORBSLAM2, the point based system ignores the point-plane relationships. As the red box indicates, some feature points are isolated and far away from the planes. It is often unreasonable for the planes such as exterior walls, ceilings and floors since feature points should not appear outside the room.

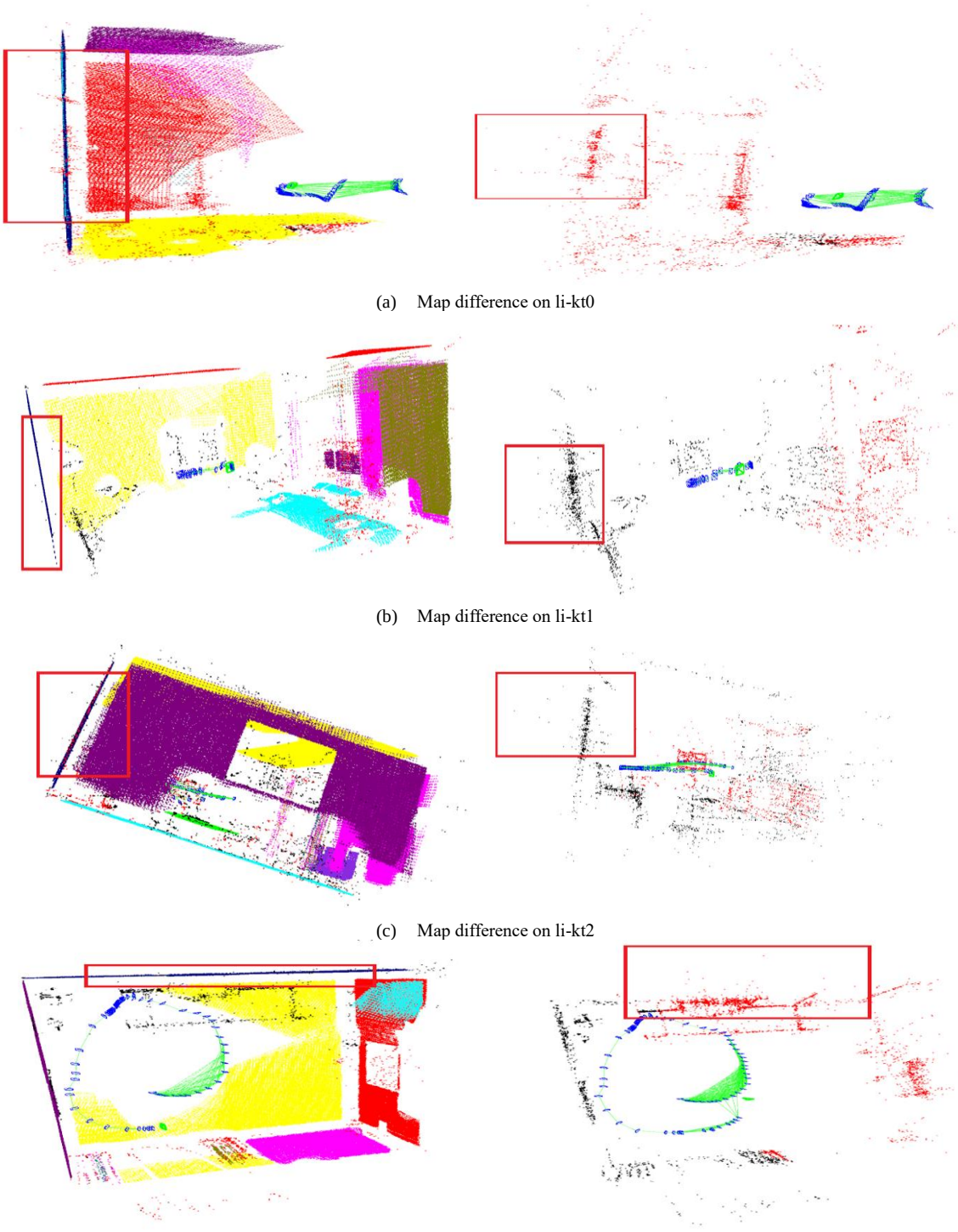
VI. CONCLUSION

This paper proposes an RGB-D visual SLAM system that utilises the point-on-plane relationship. The point-plane constraints were exploited to adjust the valid point features' locations. Along with point features, plane features were involved in solving the camera pose for each frame. A joint point-plane cost function wSas applied to optimise the pose estimation locally and globally. During the optimisation procedure, the position was marked fixed for the point features registered with a plane. The proposed method was tested in two open-source RGB-D SLAM datasets, and it achieved better performance than the widely cited state-of-the-art open-source SLAM point only algorithms in most scenarios. Compared with other recent SLAM algorithms that utilised geometry features, the proposed algorithm showed competitive results in some scenarios with rich plane features. The mapping results demonstrated that the proposed method would generate a more reasonable map that constraints some points on a plane. The proposed work can provide a novel

view for integrating the point and plane features. Since the proposed system has been implemented based on the ORBSLAM2 system, the proposed approach still faces some limitations. For example, the system needs to detect enough point features to initial the map construction. In the future, we would like to enable the system to work along with plane features and line features and extend the work to get involved with other sensors like LiDAR and explore the fusion of both LiDAR and camera.

REFERENCES

[1] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localisation and mapping: Toward the robust-perception age," *IEEE Transactions on robotics*, vol. 32, no. 6, pp.



(a) Map difference on li-kt0

(b) Map difference on li-kt1

(c) Map difference on li-kt2

(d) Map difference on li-kt3

Fig. 2. The constructed maps of the proposed method and ORBSLAM2 in ICL-NUIM living room sequences. For each row, the image at the left represents the mapping result of the proposed method. The image at the right reprsents the mapping result of ORBSLAM2. The map is composed of point features and plane features.

1309–1332, 2016.

[2] G. Klein and D. Murray, “Parallel tracking and mapping for small AR workspaces,” in *Proceedings of the 2007 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007, pp. 1–10.

[3] C. K. J. S. Lingni Ma and D. Cremers, “CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 2016, pp. 1285–1291.

[4] Q. Sun, J. Yuan, X. Zhang, and F. Duan, “Plane-Edge-SLAM: Seamless fusion of planes and edges for SLAM in indoor environments,” *IEEE Transactions on Automation Science and Engineering*, 2020.

[5] Y. Li, R. Yunus, N. Brasch, N. Navab, and F. Tombari, “RGB-D SLAM with Structural Regularities,” in *2021 IEEE international conference on Robotics and automation (ICRA)*, 2021.

[6] A. J. Trevor, J. G. Rogers, and H. I. Christensen, “Planar surface SLAM with 3D and 2D sensors,” in *2012 IEEE International Conference on Robotics and Automation*, 2012, pp. 3041–3048.

[7] X. Zhang, W. Wang, X. Qi, Z. Liao, and R. Wei, “Point-Plane SLAM Using Supposed Planes for Indoor Environments,” *Sensors*, vol. 19, no. 17, 2019 [Online]. Available: https://www.mdpi.com/1424-8220/19/17/3795

[8] X. Zhang, Z. Liao, X. Qi, and W. Wang, “Stereo Plane SLAM Based on Intersecting Lines,” in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[9] S.-J. Li, B. Ren, Y. Liu, M.-M. Cheng, D. Frost, and V. A. Prisacariu, “Direct line guidance odometry,” in *2018 IEEE international conference on Robotics and automation (ICRA)*, 2018, pp. 5137–5143.

[10] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, “A benchmark for the evaluation of RGB-D SLAM systems,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 573–580.

[11] A. Handa, T. Whelan, J. McDonald, and A. J. Davison, “A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM,” in *2014 IEEE international conference on Robotics and automation (ICRA)*, 2014, pp. 1524–1531.

[12] A. P. Gee, D. Chekhlov, W. W. Mayol-Cuevas, and A. Calway, “Discovering Planes and Collapsing the State Space in Visual SLAM.,” in *BMVC*, 2007, pp. 1–10.

[13] P. Kim, B. Coltin, and H. J. Kim, “Linear RGB-D SLAM for planar environments,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 333–348.

[14] F. Servant, E. Marchand, P. Houlier, and I. Marchal, “Visual planes-based simultaneous localisation and model refinement for augmented reality,” in *2008 19th International Conference on Pattern Recognition*, 2008, pp. 1–4.

[15] X. Zuo, Y. Yang, P. Geneva, J. Lv, Y. Liu, G. Huang, and M. Pollefeys, “Lic-fusion 2.0: Lidar-inertial-camera odometry with sliding-window plane-feature tracking,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020, pp. 5112–5119.

[16] Q. Sun, J. Yuan, X. Zhang, and F. Sun, “RGB-D SLAM in Indoor Environments With STING-Based Plane Feature Extraction,” *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 3, pp. 1071–1082, 2018.

[17] R. Guo, K. Peng, W. Fan, Y. Zhai, and Y. Liu, “Rgb-d slam using point-plane constraints for indoor environments,” *Sensors*, vol. 19, no. 12, p. 2721, 2019.

[18] M. Hsiao, E. Westman, and M. Kaess, “Dense planar-inertial SLAM with structural constraints,” in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2018, pp. 6521–6528.

[19] R. Mur-Artal and J. D. Tardós, “Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras,” *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[20] R. B. Rusu and S. Cousins, “3D is here: Point Cloud Library (PCL),” in *IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[21] M. A. Fischler and R. C. Bolles, “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography,” *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[22] P. Geneva, K. Eckenhoff, Y. Yang, and G. Huang, “LIPS: LiDAR-Inertial 3D Plane SLAM,” in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 123–130.

[23] R. Kummerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, “G2O: A general framework for graph optimisation,” in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, 2011, pp. 3607–3613.

[24] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison, “ElasticFusion: Dense SLAM without a pose graph,” 2015.

[25] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, “PL-SLAM: Real-time monocular visual SLAM with points and lines,” in *2017 IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 4503–4508.

[26] P. Kim, B. Coltin, and H. J. Kim, “Low-drift visual odometry in structured environments by decoupling rotational and translational motion,” in *2018 IEEE international conference on Robotics and automation (ICRA)*, 2018, pp. 7247–7253.

[27] M. Grupp, “evo: Python package for the evaluation of odometry and SLAM.,” 2017.