

VIP-SLAM: An Efficient Tightly-Coupled RGB-D Visual Inertial Planar SLAM

Danpeng Chen^{1,2,3} Shuai Wang^{2,3} Weijian Xie^{1,2} Shangjin Zhai²,
Nan Wang^{2,3} Hujun Bao¹ Guofeng Zhang^{1*}

¹ State Key Lab of CAD&CG, Zhejiang University ²SenseTime Research ³Tetras.AI

Abstract—In this paper, we propose a tightly-coupled SLAM system fused with RGB, Depth, IMU and structured plane information. Traditional sparse points based SLAM systems always maintain a mass of map points to model the environment. Huge number of map points bring us a high computational complexity, making it difficult to be deployed on mobile devices. On the other hand, planes are common structures in man-made environment especially in indoor environments. We usually can use a small number of planes to represent a large scene. So the main purpose of this article is to decrease the high complexity of sparse points based SLAM. We build a lightweight back-end map which consists of a few planes and map points to achieve efficient bundle adjustment (BA) with an equal or better accuracy. We use homography constraints to eliminate the parameters of numerous plane points in the optimization and reduce the complexity of BA. We separate the parameters and measurements in homography and point-to-plane constraints and compress the measurements part to further effectively improve the speed of BA. We also integrate the plane information into the whole system to realize robust planar feature extraction, data association, and global consistent planar reconstruction. Finally, we perform an ablation study and compare our method with similar methods in simulation and real environment data. Our system achieves obvious advantages in accuracy and efficiency. Even if the plane parameters are involved in the optimization, we effectively simplify the back-end map by using planar structures. The global bundle adjustment is nearly 2 times faster than the sparse points based SLAM algorithm.

I. INTRODUCTION

6DoF motion estimation in unknown environments is a very critical and challenging technology for many applications, such as robot navigation, autonomous driving and AR/VR. To solve the indoor positioning problem, in recent years, many researchers have concentrated on using onboard sensors for real-time Simultaneous Localization and Mapping (SLAM).

At present, many visual SLAM [1], [2], [3], [4], [5], [6] can achieve high positioning accuracy. However, only relying on external scene information such as vision and geometry is easily affected by the external environment. To solve the problem of environment dependence, a lot of visual inertial odometry [7], [8], [9] and visual inertial SLAM [10], [11], [12] are proposed. They integrate IMU information that does not depend on the external environment into the system.

Although the existing visual inertial systems have achieved very high accuracy and robustness, there are still many problems in practical applications. The visual inertial odometry has cumulative drift. Visual inertial SLAM needs to optimize

many sparse map points, keyframe poses and IMU states, and these high-complexity problems are difficult to solve in real-time on mobile devices. There are many planar structures in the indoor environment. The plane constraints have two major advantages. 1) The plane usually is a larger structure feature, and these large features may effectively suppress the cumulative drift of SLAM; 2) Compared with feature points, plane can use fewer parameters to model a larger environment. Therefore, how to use these higher-level plane features to improve the localization accuracy and stability of the SLAM and reduce the system complexity are significant problems that needs to be solved.

So far, there are many literatures [13], [14], [15], [16], [17], [18], [19], [20], [21], [22] using line and planar features to improve the accuracy and robustness of SLAM in challenging environments. But most of them are just odometry or pure visual SLAM, and there is no literature to propose a complete RGB-D inertial plane tightly-coupled SLAM. There is also work [22] discussing the use of planes to reduce the dimension of the Hessian matrix and improve the solving speed of BA, but it is the only odometry, and re-parameterization does not reduce the number of constraints.

In summary, many previous works have shown that IMU helps to improve the robustness of the system, and structured plane helps to improve the accuracy and robustness of the system. Moreover, compared with line and point features, the plane can use fewer parameters to model the environment. Based on this, we make full use of the characteristics of multiple sensors to propose a highly robust and precise system, which integrates IMU, RGB, Depth, and Plane information. There are three contributions in this article:

- We are the first to propose a complete tightly-coupled multi-sensor fusion SLAM system to fuse RGB, Depth, IMU, and structured plane information. All the information are integrated into an unified non-linear optimization framework, which jointly optimizes the parameters of keyframe poses, IMU states, points, and planes.
- We introduce plane information to reduce the number of map points and speed up the optimization of BA. We use homography to remove the states of the point in the optimization and compress multiple constraints into one at the same time. These measures reduce the optimization time. The Fig 4(a) and 4(b) show the process.
- The plane information is integrated into the entire SLAM system to realize high-precision tracking. We use pure geometric single-frame point-to-plane constraints to improve the accuracy and stability of plane estimation in

*Corresponding author: Guofeng Zhang (zhangguofeng@zju.edu.cn)

This work was partially supported by the National Key Research and Development Program of China under Grant 2020AAA0105900. All the authors are also affiliated with ZJU-SenseTime Joint Lab of 3D Vision.

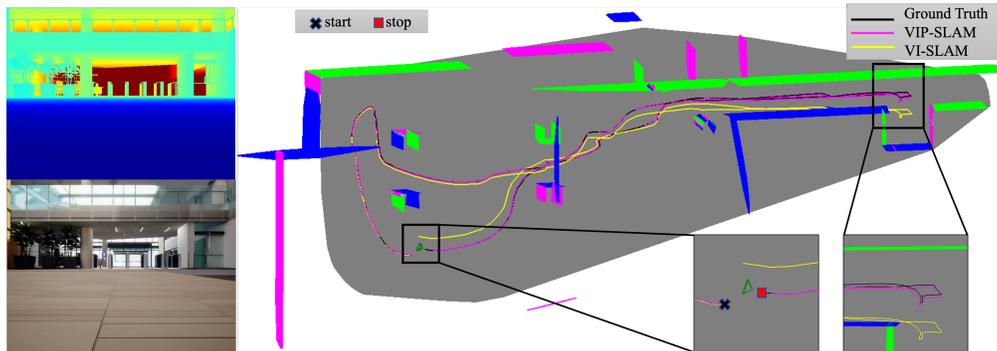


Fig. 1: Trajectories and reconstructed planes from sequence 02. Top left is depth image, bottom left is RGB image. Right image: The gray area is horizontal plane. Pink, green and blue areas are different vertical planes. Black, pink and yellow trajectories are Ground Truth, our VIP-SLAM and normal VI-SLAM.

textureless scenes. Moreover, we convert the reprojection of the plane point to homography constraints to establish the relationship between multiple frames and planes to further correct the drift. Fig. 1 shows comparison of trajectories and reconstructed planes.

We evaluate our system both on simulation and real data. The data includes slow and fast movements, low texture, small indoor rooms and large scenes. Experimental results show that our system not only achieves higher accuracy and robustness but also has a significant improvement in the efficiency of solving BA.

II. RELATED WORK

RGB-D Inertial SLAM: In the past few years, RGB-D sensors have become more and more popular. Many methods [4], [6], [5], [3], [2] are proposed to use RGB-D cameras for real-time 3D dense reconstruction or SLAM. However, traditional RGB-D SLAM easily suffer from the robustness problem in special scenarios, such as fast motion, weak textures, dynamic environments, flat white walls, etc. To reduce the dependence on vision and geometry, researchers proposed a few RGB-D inertial SLAM systems. [23] proposes to integrate IMU and RGB-D information to achieve the first tightly-coupled RGB-D inertial SLAM system. Their experiments showed the advantages of their proposed system in fast motion, weak texture, and weak geometric scenes. VINS-RGBD [10] integrates depth information to the monocular visual inertial system VINS-Mono [9]. They use depth information to speed up system initialization, stabilize system scale and improve system accuracy and robustness. DPI-SLAM [13] proposes the first plane inertial SLAM system which tightly couples IMU, visual odometry (VO), and plane information. However, the loose coupling of IMU and VO may encounter the same robustness issues as RGB-D SLAM. In ORB-SLAM3 [11], inertial constraints and the methods of multi-maps are fused into ORB-SLAM2 [2]. However, ORB-SLAM3 only supports monocular/stereo visual inertial SLAM or RGB-D SLAM.

Plane Based SLAM: Some researchers have explored ways to improve the accuracy of SLAM systems using planar structures. [14] proposes a RANSAC-based registration method for localization based on points and planes. [15] uses the global plane model to reduce the RGB-D SLAM drift. Kimera [17] proposes to use 2D image Delaunay triangulation and corresponding sparse point cloud to reconstruct 3D

Mesh. SP-SLAM [16] adds features to ORB-SLAM2, where parallel and perpendicular plane constraints also are included. [18] forces the points on the plane to fall exactly on the plane, and add new feature points and plane constraints based on this assumption. KDP-SLAM [24] develops a fast dense planar SLAM system, which optimizes the keyframe poses and planes in a global factor graph. But, the loose coupling of planes and VO may lead to a decrease in the accuracy of motion estimation in scenes with one or two planes. Some works model the environment as the Manhattan or Atlanta world and use this assumption to reduce the cumulative localization error in rotation. [19] decouples rotation and translation estimation based on Manhattan assumption, and combines point, line, and surface features to improve the accuracy of translation. [20]’s system is also based on the Manhattan world hypothesis, but it can support seamless switching to the non-Manhattan world. These world hypothesis algorithm can work well under special scenes. However, it does not work well under complex environments. There are also some works focused on improving the speed of plane bundle adjustment (PBA). [25], [26], [27] propose to merge multiple measurements related to the same state into one constraint, which significantly improve the efficiency of PBA. Inspired by these works, we also merge point-to-plane constraints and extend them to homography observations. [22] forces the points associated with the plane to fall on the plane and uses the plane and the corresponding anchor pose to represent the point. This form of representation eliminates the point state and reduces the dimensionality of the Hessian matrix, which will increase BA solved speed.

III. OVERVIEW

A. System Overview

The overview of the proposed system is shown in Fig. 2. Our system takes RGB, Depth, and IMU as input and has three main components including front end, plane module, and back end.

The front-end module is a VIO system based on a sliding window and estimates 6-DoF poses in real-time. Our VIO is similar to [7], except that we do not consider the SLAM features and add additional depth measurements.

The plane module receives the front-end and back-end data as input. The high-frequency front-end information is only used to expand the plane. The low-frequency but

high-precision back-end information is accepted for new plane detection, plane expansion, point-to-plane association, and plane-to-plane merging. Since most planes of indoor environment are horizontal or vertical, we only consider the detection of horizontal and vertical planes. However, the fusion optimization about planes is suitable for general planes.

The back-end module uses local bundle adjustment (LBA) or global bundle adjustment (GBA) to jointly optimize planes, points, IMU states and keyframe poses and corrects the drift of the front-end pose.

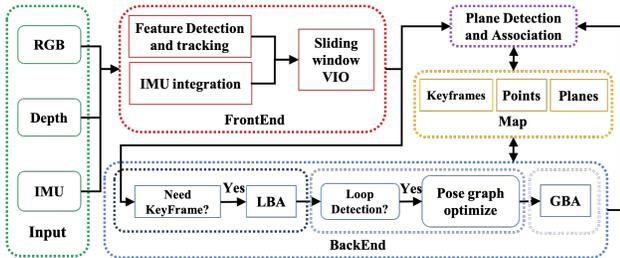


Fig. 2: The pipeline of the proposed system.

B. Notation

We first define notations that used throughout the paper. We consider $(\cdot)^W$ as the world frame. $(\cdot)^C$ is the camera frame and $(\cdot)^I$ is the IMU frame. We represent a pose by $T \in SE(3)$ which consists of rotation $R \in SO(3)$ and translation $t \in R^3$. We use $\pi = [n; d]^T$ to represent a plane, where n is the plane normal, and d is the distance between the origin and plane. We adopt CP [28] vector to parameterize plane π , i.e., $\eta = nd$. X are the states need to be estimated, including poses, velocities, IMU bias, points and plane landmarks.

IV. FRONT END

Feature Detection and Matching When a new image is received, we detect ORB feature [1] points and compute corresponding descriptors. First, we use KLT to track them from the last image to the current image. Then, we project features that have 3D information onto the current image and use the Hamming distance to find the closest ORB feature point. The remaining features are matched by using the result of KLT tracking as the initial value to find the optimal observation. Finally, we use the RANSAC-based fundamental matrix to remove outliers.

Motion Estimation Motion estimation is a sliding window based VIO, which tightly integrates RGB, depth, and IMU. Our VIO is similar to [7], which uses a square root inverse filter to fuse all measurements. The major difference is that we do not consider the SLAM features and add depth information to visual measurements. We will describe visual measurements with depth in detail in Section VI-A.1.

V. PLANE DETECTION AND ASSOCIATION

A. Plane Detection and Merging

Plane Detection The plane module only detects planes with back-end data. Once a plane is detected, the plane module expands the plane with front-end and back-end data and associates the plane with map landmarks. Like [17], we use

Delaunay triangulation to create 3D mesh and histogram to detect planes. We only detect vertical and horizontal planes, by checking out whether the mesh normal is vertical or parallel to the gravity. Our plane module employs some additional methods to improve the plane accuracy. When a plane is detected from histogram, we will refine its parameters with the data and 3D plane points in the histogram, instead of using the scale value of the histogram directly. For horizontal plane, we just set $n = [0, 0, 1]^T$ and the plane distance is the average of the z axis of plane points. For vertical plane, we set $n = [n_x, n_y, 0]^T$, and the vertical plane parameters can be refined with the following Equation:

$$[\bar{P}_{f_1}^W, \bar{P}_{f_2}^W, \dots, \bar{P}_{f_n}^W]^T \cdot \begin{bmatrix} n_x/d \\ n_y/d \end{bmatrix} = [-1, -1, \dots, -1]^T \quad (1)$$

where n is the number of plane points, $P_{f_k}^W$ is the position of k^{th} landmark under the world frame, and $\bar{P}_{f_k}^W = [P_{f_k}^W(0), P_{f_k}^W(1)]^T$. We use QR decomposition to solve the Equation (1). When plane parameters are detected, the plane can associate 3D mesh by angle and distance. Our angular and distance thresholds are about 10 degrees and 5 centimeters between the plane and 3D mesh. Fig. 3 shows this process.

Plane Merging The plane merging has 2 strategies. First, we check whether a plane satisfies a certain angle and distance threshold with others. Once we find satisfied planes, secondly, we will check the boundary points of satisfied planes whether contain each other. Our angular and distance thresholds are 10 degrees and 10 centimeters. Plane merging occurs when a new plane is detected or old planes are adjusted.

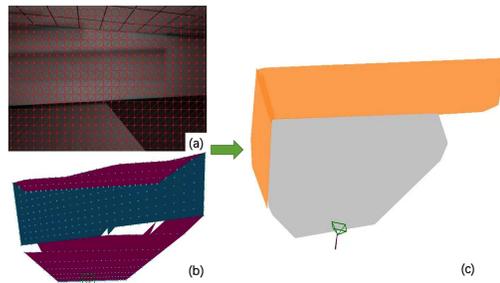


Fig. 3: Plane detection: (a) 2D mesh. (b) 3D mesh, where red and blue areas indicate horizontal and vertical meshes respectively. (c) Planes where vertical planes are yellow, and horizontal plane is grey.

B. Point and Plane Association

We use 3D meshes to associate more map points with planes. Once a 3D mesh from the depth map has been associated with a plane, we will find its 2D mesh. If both the 2D coordinate of a map point is in the 2D mesh and the distance from the map point to the plane is less than 10 centimeters, the map point will be added into the candidate set associated with the plane. If a point on the candidate set is observed in more than 3 keyframes, we will check its geometric consistency. We calculate the reprojection error of the point, force the point to be associated with the plane, and then calculate another reprojection error. If the two reprojection errors are similar, and the maximum reprojection error is lower than a certain threshold, we consider the point to

be a plane point. If a point fails to pass geometric consistency many times, it will be removed from the candidate point set.

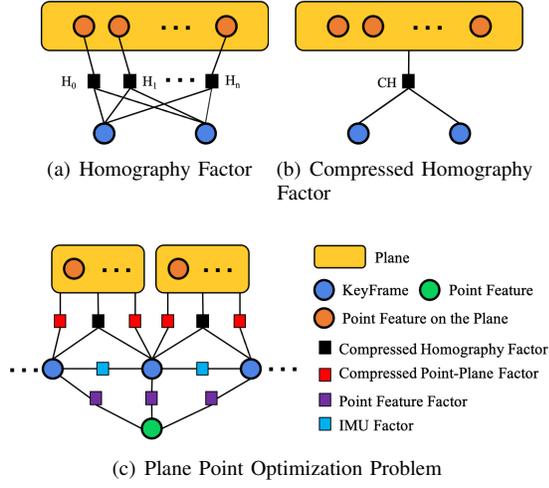


Fig. 4: Our optimization problem. (a) and (b) are the factor graphs before and after homography constraints are compressed. (c) is our global bundle adjustment problem.

VI. BACK END

A. Measurement

1) *IMU and Point Feature Measurement*: The IMU data between two continuous keyframes are processed by the preintegration method [8], [9]. We define the cost term of preintegration based IMU data, which is the same as [9].

Since depth image is valid, we integrate depth information into visual point feature measurement. The projection and depth residuals for the l^{th} feature observed in the i^{th} keyframe is defined as:

$$\begin{aligned} \mathbf{r}^C(\mathbf{X}) &= \text{Proj}(\mathbf{P}_i^{C_i}) - \mathbf{z}_i \\ \mathbf{r}^{C_\lambda}(\mathbf{X}) &= (\mathbf{P}_i^{C_i}) \cdot z() - \lambda \end{aligned} \quad (2)$$

where $\mathbf{P}_i^{C_i} = \mathbf{R}_C^l T (\mathbf{R}_W^l (\mathbf{P}_i^W - \mathbf{t}_i^W) - \mathbf{t}_C^l)$ is the 3D position of l^{th} feature in the i^{th} keyframe. \mathbf{R}_C^l , \mathbf{t}_C^l are rotation and translation from the IMU frame to the Camera frame. Proj is the project function that project the point from camera coordinate to image coordinate. \mathbf{z}_i is the l^{th} feature observation in the i^{th} image. $z()$ is the third component of the vector. λ is the corresponding depth obtained from the depth image.

2) *Compressed Homography Measurement*: When the 3D map point is associated with the plane, we enforce the point must fall on the plane. So instead of using the common point-to-plane distance constraint, we use the homography matrix to constrain two keyframes and a plane. If the point on the plane $\boldsymbol{\pi}^W$ is observed by frames i^{th} and j^{th} , we can write the following Equations of point-to-plane and reprojection:

$$\begin{aligned} \mathbf{n}_\pi^W T (\mathbf{R}_{C_i}^W \mathbf{p}_i \lambda + \mathbf{t}_{C_i}^W) + d_\pi^W &= 0 \\ \mathbf{R}_{C_j}^W T (\mathbf{R}_{C_i}^W \mathbf{p}_i \lambda + \mathbf{t}_{C_i}^W - \mathbf{t}_{C_j}^W) &= \mathbf{p}_j s \end{aligned} \quad (3)$$

We define $\mathbf{p}_i = (x_i, y_i, 1)^T$ and $\mathbf{p}_j = (x_j, y_j, 1)^T$, where $(x_i, y_i, 1)^T = \mathbf{K}^{-1}(u_i, v_i, 1)^T$ is the point feature observation on the normal plane of the frame i^{th} . \mathbf{K} is the intrinsic matrix,

and (u_i, v_i) is a 2D image feature. λ is the corresponding depth. s is an unknown scale. Combining the above Equations (3), we can get:

$$\mathbf{p}_j s = \mathbf{R}_{C_j}^W T \left[\mathbf{I} - \frac{(\mathbf{t}_{C_i}^W - \mathbf{t}_{C_j}^W) \mathbf{n}_\pi^W T}{d_\pi^W + \mathbf{n}_\pi^W T \mathbf{t}_{C_i}^W} \right] \mathbf{R}_{C_i}^W \mathbf{p}_i = \mathbf{H} \mathbf{p}_i \quad (4)$$

where \mathbf{I} is an identity matrix, and \mathbf{H} is a homography matrix. So the homography constraint and the reprojection constraint are equal when the point is on the plane. **It is worth noting that homography does not require the 3D positions of point features.** In the BA problem, we convert the reprojection constraint to homography constraint, which is equivalent to remove many state variables of points on the plane. Finally, the efficiency of bundle adjustment will be greatly improved through the smaller and sparser Hessian matrix. There is also a similar work [22] to remove the states of the plane points in optimization, but they use the reprojection representation method, it is difficult to compress multi-observation constraints, which limits the further improvement of the optimization speed. Homography associates the states of two keyframes and a plane. These three states may have many common observations. We merge these observations into one observation to further improve the optimization speed. Assume there are N point features on the plane $\boldsymbol{\pi}$ observed by i^{th} and j^{th} keyframes. According to Equation (4), the homography constraint Equation of l^{th} point feature is:

$$s \begin{bmatrix} x_j \\ y_j \\ 1 \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} & H_{13} \\ H_{21} & H_{22} & H_{23} \\ H_{31} & H_{32} & H_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} \quad (5)$$

We can eliminate unknown s from the above Equation (5) and get the residual function as:

$$\begin{aligned} \mathbf{r}_l(\mathbf{X}) &= \begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i x_j & -y_i x_j & -x_j \\ 0 & 0 & 0 & x_j & y_j & 1 & -x_i y_j & -y_i y_j & -y_j \\ [H_{11} & H_{12} & H_{13} & H_{21} & H_{22} & H_{23} & H_{31} & H_{32} & H_{33}]^T \end{bmatrix} \\ &= \mathbf{C}_l \bar{\mathbf{H}} \end{aligned} \quad (6)$$

The homography cost function of N point features on the plane $\boldsymbol{\pi}$ is:

$$\begin{aligned} \mathbf{C}^{ch}(\mathbf{X}) &= \frac{1}{2} \sum_{l=1}^N \mathbf{r}_l(\mathbf{X})^T \mathbf{r}_l(\mathbf{X}) = \frac{1}{2} \bar{\mathbf{H}}^T \left(\sum_{l=1}^N \mathbf{C}_l^T \mathbf{C}_l \right) \bar{\mathbf{H}} \\ &= \frac{1}{2} \bar{\mathbf{H}}^T \mathbf{G}_h \bar{\mathbf{H}} = \frac{1}{2} \bar{\mathbf{H}}^T \mathbf{L}_h \mathbf{L}_h^T \bar{\mathbf{H}} \end{aligned} \quad (7)$$

where $\mathbf{G}_h = \mathbf{L}_h \mathbf{L}_h^T$ is the matrix decomposition. To ensure the stability of the solution, we use eigenvalue decomposition. \mathbf{G}_h is a constant 9×9 matrix during the optimization and only depends on visual observations. We can calculate the matrix in advance. Here, we have merged observations of N point features on the plane $\boldsymbol{\pi}$ into one observation matrix \mathbf{G}_h . Fig. 4(a) and 4(b) show this process. Combining many cost functions into one cost function helps to improve the efficiency of bundle adjustment. The jacobian of compressed homography cost function can be define as $\mathbf{J}^{ch} = \mathbf{L}_h^T \frac{\partial \bar{\mathbf{H}}}{\partial \mathbf{X}}$, and residue is $\mathbf{r}^{ch} = \mathbf{L}_h^T \bar{\mathbf{H}}$

3) *Compressed Point-to-Plane Measurement*: The homography measurement relies on feature matching of map point, which is easily affected by the lighting and texture of environments. Therefore, we add geometric constraint of single frame point cloud and plane association to enhance the accuracy of plane estimation and the stability of motion estimation under textureless scenes. Similar to the work related to the Lidar plane SLAM [27], [26], [25], we use the compressed point-to-plane cost function. \mathbb{P}_{l_i} is the set of N points on the l^{th} plane $\boldsymbol{\pi}_l^W$ observed in the i^{th} keyframe. \mathbf{p}_{ilk} is the k^{th} 3D point of \mathbb{P}_{l_i} in world. The point-to-plane residue of k^{th} plane point can be defined as:

$$\mathbf{r}_{ilk}(\mathbf{X}) = \boldsymbol{\pi}_l^{W^T} \mathbf{T}_i^W \begin{bmatrix} \mathbf{p}_{ilk} \\ 1 \end{bmatrix} \quad (8)$$

where \mathbf{T}_i^W is the i^{th} keyframe pose. $\mathbf{r}_{ilk}(\mathbf{X})$ is only one dimension, so the point-to-plane cost function of N points on the l^{th} plane has the following form:

$$\mathbf{C}^{cpp}(X) = \frac{1}{2} \sum_{l=1}^N \mathbf{r}_{ilk}(\mathbf{X}) \mathbf{r}_{ilk}(\mathbf{X})^T = \frac{1}{2} \boldsymbol{\pi}_l^{W^T} \mathbf{T}_i^W \mathbf{G}_p \mathbf{T}_i^W \boldsymbol{\pi}_l^W \quad (9)$$

Similar to VI-A.2, \mathbf{G}_p is a constant 4×4 matrix, which only depends on observations, not states. Through the matrix decomposition of $\mathbf{G}_p = \mathbf{L}_p \mathbf{L}_p^T$, the jacobian of the new cost function can be written as $\mathbf{J}^{cpp} = \mathbf{L}_p^T \frac{\partial(\mathbf{T}_i^{W^T} \boldsymbol{\pi}_l^W)}{\partial \mathbf{X}}$, and residue is $\mathbf{r}^{cpp} = \mathbf{L}_p^T \mathbf{T}_i^W \boldsymbol{\pi}_l^W$.

B. Local Plane and Point Bundle Adjustment

When a new keyframe is inserted into the map, we perform LBA optimization. LBA optimizes the newest K keyframe poses, IMU states, as well as the points and planes observed by these keyframes. Other keyframes where observe these points and planes contribute to the cost function but remain fixed in LBA. Here, K is set to 20. We adopt the LM algorithm of Ceres¹ to solve this minimization problem. We set the maximum number of iterations to 10 and maximum solver time is 0.2s.

C. Global Plane and Point Bundle Adjustment

Loop Detection and Pose Plane Graph Optimization Similar to many visual SLAM [2], [11], we use DBoW2 [29] to detect loop closure and use geometry information to verify their reliability. If the loop is accepted, graph optimization is firstly used to correct large drift. Our optimization problem is to minimize the following energy function:

$$\mathbf{E}_{PPG} = \sum_{i=1}^{N-1} \left\| (\mathbf{T}_{C_i}^{W^T} \mathbf{T}_{C_{i+1}}^W) \mathbf{T}_{C_i}^{C_{i+1}} \right\|_{\boldsymbol{\Sigma}_{i(i+1)}}^2 + \left\| (\mathbf{T}_{C_m}^{W^T} \mathbf{T}_{C_n}^W) \mathbf{T}_{C_m}^{C_n} \right\|_{\boldsymbol{\Sigma}_{mn}}^2 + \sum_{i=1}^N \sum_{l \in \mathbb{M}_i} \left\| \boldsymbol{\pi}_l^W - \mathbf{T}_{C_i}^W \boldsymbol{\pi}_l^{C_i} \right\|_{\boldsymbol{\Sigma}_i}^2 \quad (10)$$

Here, N is the number of all keyframes. m^{th} and n^{th} is a pair of loop keyframes. $\boldsymbol{\Sigma}$ is all the corresponding covariance matrix. \mathbb{M}_i is the set of planes observed by the i^{th} keyframe.

Optimization After pose plane graph optimization, we need to update all states including keyframe poses, IMU states, points, and planes. For the GBA problem, we fuse all measurements including preintegration, reprojection, compressed homography, compressed point-to-plane and prior plane in

a tightly-coupled form. Fig. 4 shows this problem. The optimization is to minimize the following energy function:

$$\mathbf{E}_{GBA} = \sum_i \left\| \mathbf{r}_{i(i+1)}^{IMU} \right\|_{\boldsymbol{\Sigma}_{IMU}}^2 + \sum_{ik} \left\| \mathbf{r}_{ik}^C \right\|_{\boldsymbol{\Sigma}_C}^2 + \sum_{ik} \left\| \mathbf{r}_{ik}^{C_\lambda} \right\|_{\boldsymbol{\Sigma}_{C_\lambda}}^2 + \sum_{ijs} \left\| \mathbf{r}_{ijs}^{ch} \right\|_{\boldsymbol{\Sigma}_{ch}}^2 + \sum_{is} \left\| \mathbf{r}_{is}^{cpp} \right\|_{\boldsymbol{\Sigma}_{cpp}}^2 \quad (11)$$

The first two items of Equation (11) are similar to most VI-SLAM systems [12], [11]. $\{\mathbf{r}_{i(i+1)}^{IMU}, \mathbf{r}_{ik}^C, \mathbf{r}_{ik}^{C_\lambda}\}, \{\mathbf{r}_{ijs}^{ch}\}, \{\mathbf{r}_{is}^{cpp}\}$, are respectively described in the Subsection VI-A.1, VI-A.2 and VI-A.3. We use LM algorithm to solve this minimization problem. We set the maximum number of iterations to 100 and maximum solver time is 2s.

VII. EXPERIMENT

We conduct many experiments to verify the accuracy and efficiency of our system. We perform an ablation study to test the effects of different modules and compare them with some similar algorithms. We ran our experiments on a computer with an i7-6700 CPU and 16G RAM.

A. Dataset

Our data comprises simulation data and real environment data. The simulation data is collected by AirSim [30]. AirSim is a simulator for drones, cars and more. We use a drone to collect data in an indoor scene of a large virtual office building. The scene is quite challenging, which is large and empty, with many textureless areas. To verify the effect in the actual scene, we use a mobile phone on the market with ToF sensor to collect RGB, Depth, and IMU data. We collected the data in a small VICON² room. The data in the VICON room has a high-precision trajectory provided by VICON. Sequences 01 to 05 are collected by AirSim, and 06 to 10 are mobile phone's data. The camera moves slowly on sequences 06 to 08, but rapidly on sequences 09 and 10.

We also test with EuRoC MAV visual inertial datasets [31]. Our algorithm fuses with RGB-D and IMU sensor, however, EuRoC datasets have no depth images. So we generate depth images through the 3D module from [32]. Considering the noise of the actual depth image, we generate a Gaussian random noise $n \sim N(0, 0.0017^2)$. Assuming original depth is d , we change it to $d' = d + d \cdot n$.

B. Ablation Study and Baseline

To verify the influence of different modules of our algorithm, we consider three variants of our algorithm. **VI-SLAM** is our basic VI-SLAM fused with RGB-D and IMU sensor, without plane-related constraints. **VI-SLAM-P** adds planar constraints without compression on the basis of VI-SLAM. **VI-SLAM-CP** compresses plane-related constraints on the basis of VI-SLAM-P. **VIP-SLAM** is our complete algorithm. Different from VI-SLAM-CP, VIP-SLAM removes the states of the plane points in the optimization. We also compare our algorithm with ORB-SLAM2 (RGB-D) [2], ManhattanSLAM [20] and Kimera [17]. To better demonstrate the influence of the plane on the cumulative error, we also test the version without loop closure.

¹<http://ceres-solver.org>

²<https://www.vicon.com/>

TABLE I: ATE RMSE(cm) of methods without/with loop closure.

Dataset	Seq	ORB-SLAM2[2]	Manhattan-SLAM[20]	VI-SLAM	VI-SLAM-P	VIP-SLAM
AirSim	01	16.8/9.1	12.6/X	5.6/2.9	2.2/1.7	1.2/1.4
	02	112.1/59.7	112.7/X	36.7/2.2	4.0/2.0	3.6/1.8
	03	15.6/9.1	6.5/X	9.3/2.7	2.1/3.3	2.1/1.5
	04	76.6/40.1	420.3/X	156.2/16.6	21.1/ 8.0	7.8/9.2
	05	63.9/62.5	113.4/X	8.4/5.1	1.8/1.9	2.0/1.1
Mobile Phone	06	5.9/1.7	14.8/X	9.3/2.3	9.6/2.7	7.9/1.8
	07	14.3/3.0	8.3/X	10.0/ 1.6	3.4/2.7	3.3/2.4
	08	10.4/3.6	6.2/X	6.5/ 2.3	3.6/2.7	3.0/2.3
	09	X/X	X/X	15.1/2.0	5.5/2.5	5.4/1.6
	10	X/X	X/X	34.4/ 2.4	14.0/2.8	24.5/4.0
Avg	X/X	X/X	29.2/4.0	6.7/3.0	6.1/2.7	

TABLE II: ATE RMSE(cm) of methods without/with loop closure.

Dataset	Seq	ORB-SLAM2[2]	Manhattan-SLAM[20]	Kimera[17]	VI-SLAM	VIP-SLAM
EuRoC	V101	2.2/2.2	6.4/X	5.0/5.0	2.9/2.9	2.2/2.0
	V102	21.0/ 2.1	32.9/X	8.0/11.0	11.7/5.4	2.2/2.4
	V103	24.0/6.2	7.2/X	7.0/12.0	5.2/3.1	2.1/2.2
	V201	5.1/6.0	7.3/X	8.0/7.0	5.4/ 3.1	3.9/3.2
	V202	5.3/4.0	9.3/X	10.0/10.0	5.5/4.4	2.9/3.1
	V203	X/X	X/X	21.0/19.0	20.5/8.1	7.8/7.5
	Avg	X/X	X/X	9.8/10.7	8.5/4.5	3.5/3.4

C. Accuracy

We run each sequence three times and show the average accuracy results in Tables I and II. The results of Kimera came from [17]. Under the cases of weak texture and fast motion, the VI-SLAM algorithm is more accurate and robust than ORB-SLAM2 and Manhattan-SLAM, which is mainly due to the integration of IMU does not rely on external information. Both VIP-SLAM and VI-SLAM-P achieve the best accuracy, which demonstrates that compressing multiple observations and replacing the reprojection constraints of the point on the plane with homography constraints are effective. However, because of the loop closure effect, the cumulative error of SLAM is effectively eliminated, resulting in the accuracy difference between VIP-SLAM and VI-SLAM is minor. With the loop closure module disabled, VIP-SLAM and VI-SLAM-P have achieved obvious advantages on all datasets, which effectively proves that plane constraints are very helpful to ease the accumulated errors of SLAM.

In all sequences, the performances of Manhattan-SLAM and Kimera are poor. For Manhattan-SLAM, we guess there are two reasons. 1) Our data is more challenging, with weak textures, large indoor areas, fast movements, etc. 2) Manhattan-SLAM does not make full use of plane information. Besides they only use observations from a single frame to a plane. We use the point-to-plane constraints of direct observation to achieve a more accurate estimation of single-frame observations. Moreover, we convert the reprojection of the plane point to homography constraints to establish the relationship between multiple frames and planes. As for Kimera, the poor accuracy of the plane and local plane information cause poor performance. They only consider the local planes within the small window.

TABLE III: Time-consuming and corresponding accuracy of GBA for different variant algorithms

	VI-SLAM	VI-SLAM-P	VI-SLAM-CP	VIP-SLAM
Iter.	10	10	10	10
Redisual (ms)	65	181	117	22
Jacobians (ms)	145	777	213	86
Linear (ms)	425	593	462	113
Pre-processor (ms)	24	84	27	3
Post-processor (ms)	3	7	3	1
Sum (ms)	722	1845	912	254
RMSE (m)	0.038	0.032	0.027	0.032

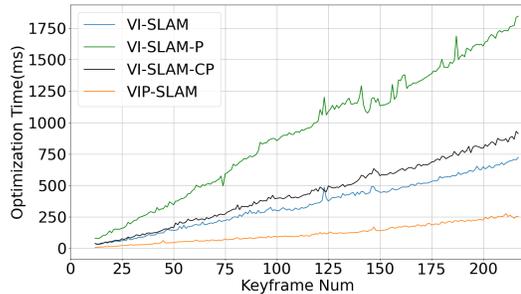


Fig. 5: The GBA time for variants of our algorithm.

D. Runtime

To verify VIP-SLAM cost time of optimization, we do an experiment of GBA with sequence 09. All algorithms have a maximum number of 10 iterations and do not limit the optimization time. Fig. 5 shows a comparison of the GBA time for variants of our algorithm. The horizontal axis is the numbers of keyframes, and the vertical axis is the optimization time of GBA. We find that the optimization time of VI-SLAM-P is much higher than VI-SLAM due to too many additional point-to-plane and homography measurements. After using the compressed measurements as VI-SLAM-CP, the optimization time is close to VI-SLAM but still higher. Finally, removing the state of plane points, optimization time is nearly 2 times faster than VI-SLAM. We also find the optimization time consumption will be reduced obviously as the number of keyframe increases. Table III shows the time-consuming and accuracy of GBA for different variant algorithms. The states of GBA include 215 keyframes, 2,236 planar points, 2,032 non-planar points, 10 vertical planes, and 6 horizontal planes. We list several important sections of optimization time-consuming, as we can see, time-consuming is reduced for all important sections of our VIP-SLAM with close accuracy.

VIII. CONCLUSIONS

Our goal is to make full use of multiple sensors' information to achieve a high-precision, lightweight and robust SLAM system. The experiments demonstrates the effectiveness of the proposed system. Using plane information not only helps to reduce the cumulative error of the system but also accelerates the GBA. Our points and planes GBA is nearly 2 times faster than the sparse points based SLAM algorithm. So far, the back-end map of our system still includes a lot of points. In the future, we will explore whether a more lightweight pure plane-based visual SLAM back end is feasible.

REFERENCES

- [1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "ORB-SLAM: a versatile and accurate monocular SLAM system," *IEEE transactions on robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.
- [2] 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.
- [3] H. Liu, C. Li, G. Chen, G. Zhang, M. Kaess, and H. Bao, "Robust keyframe-based dense SLAM with an RGB-D camera," *arXiv preprint arXiv:1711.05166*, 2017.
- [4] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon, "KinectFusion: Real-time dense surface mapping and tracking," in *Proceedings of 10th IEEE international symposium on mixed and augmented reality*. IEEE, 2011, pp. 127–136.
- [5] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger, "ElasticFusion: Real-time dense SLAM and light source estimation," *The International Journal of Robotics Research*, vol. 35, no. 14, pp. 1697–1716, 2016.
- [6] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald, "Kintinuous: Spatially extended kinectfusion," 2012.
- [7] K. Wu, A. Ahmed, G. A. Georgiou, and S. I. Roumeliotis, "A square root inverse filter for efficient vision-aided inertial navigation on mobile devices," in *Robotics: Science and Systems*, vol. 2, 2015, p. 2.
- [8] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "IMU preintegration on manifold for efficient visual-inertial maximum-a-posteriori estimation." Georgia Institute of Technology, 2015.
- [9] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.
- [10] Z. Shan, R. Li, and S. Schwertfeger, "RGBD-inertial trajectory estimation and mapping for ground robots," *Sensors*, vol. 19, no. 10, p. 2251, 2019.
- [11] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "ORB-SLAM3: An accurate open-source library for visual, visual-inertial, and multimap SLAM," *IEEE Transactions on Robotics*, 2021.
- [12] R. Mur-Artal and J. D. Tardós, "Visual-inertial monocular SLAM with map reuse," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 796–803, 2017.
- [13] M. Hsiao, E. Westman, and M. Kaess, "Dense planar-inertial SLAM with structural constraints," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 6521–6528.
- [14] L. Ma, C. Kerl, J. Stückler, and D. Cremers, "CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 1285–1291.
- [15] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, "Point-plane SLAM for hand-held 3D sensors," in *Proceedings of IEEE international conference on robotics and automation*. IEEE, 2013, pp. 5182–5189.
- [16] 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, p. 3795, 2019.
- [17] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," *arXiv preprint arXiv:1910.02490*, 2019.
- [18] C. Arndt, R. Sabzevari, and J. Civera, "From points to planes-adding planar constraints to monocular SLAM factor graphs," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 4917–4922.
- [19] Y. Li, N. Brasch, Y. Wang, N. Navab, and F. Tombari, "Structure-SLAM: Low-drift monocular SLAM in indoor environments," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6583–6590, 2020.
- [20] R. Yunus, Y. Li, and F. Tombari, "ManhattanSLAM: Robust planar tracking and mapping leveraging mixture of manhattan frames," *arXiv preprint arXiv:2103.15068*, 2021.
- [21] K. Joo, T.-H. Oh, F. Rameau, J.-C. Bazin, and I. S. Kweon, "Linear RGB-D SLAM for atlanta world," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 1077–1083.
- [22] X. Li, Y. Li, E. P. Örnek, J. Lin, and F. Tombari, "Co-planar parametrization for stereo-SLAM and visual-inertial odometry," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6972–6979, 2020.
- [23] T. Laidlow, M. Bloesch, W. Li, and S. Leutenegger, "Dense RGB-D-inertial SLAM with map deformations," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 6741–6748.
- [24] M. Hsiao, E. Westman, G. Zhang, and M. Kaess, "Keyframe-based dense planar SLAM," in *Proceedings of IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 5110–5117.
- [25] L. Zhou, D. Koppel, H. Ju, F. Steinbruecker, and M. Kaess, "An efficient planar bundle adjustment algorithm," in *Proceedings of IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2020, pp. 136–145.
- [26] L. Zhou, D. Koppel, and M. Kaess, "Lidar SLAM with plane adjustment for indoor environment," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7073–7080, 2021.
- [27] L. Zhou, S. Wang, and M. Kaess, " π -LSAM: LiDAR smoothing and mapping with planes."
- [28] P. Geneva, K. Eickenhoff, Y. Yang, and G. Huang, "Lips: Lidar-inertial 3d plane SLAM," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 123–130.
- [29] D. Gálvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.
- [30] S. Shah, D. Dey, C. Lovett, and A. Kapoor, "Airsim: High-fidelity visual and physical simulation for autonomous vehicles," in *Field and service robotics*. Springer, 2018, pp. 621–635.
- [31] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The euroc micro aerial vehicle datasets," *The International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.
- [32] H. Bao, W. Xie, Q. Qian, D. Chen, S. Zhai, N. Wang, and G. Zhang, "Robust tightly-coupled visual-inertial odometry with pre-built maps in high latency situations," *IEEE Transactions on Visualization & Computer Graphics*, no. 01, pp. 1–1, 2022.