Monocular Object and Plane SLAM in Structured Environments

Shichao Yang, Sebastian Scherer

***Abstract*—In this paper, we present a monocular Simultaneous Localization and Mapping (SLAM) algorithm using high-level object and plane landmarks. The built map is denser, more compact and semantic meaningful compared to feature point based SLAM. We first propose a high order graphical model to jointly infer the 3D object and layout planes from single images considering occlusions and semantic constraints. The extracted objects and planes are further optimized with camera poses in a unified SLAM framework. Objects and planes can provide more semantic constraints such as Manhattan plane and object supporting relationships compared to points. Experiments on various public and collected datasets including ICL NUIM and TUM Mono show that our algorithm can improve camera localization accuracy compared to state-of-the-art SLAM especially when there is no loop closure, and also generate dense maps robustly in many structured environments.**

***Index Terms*—SLAM, Semantic Scene Understanding, Object and Plane SLAM**

I. Introduction

SEMANTIC understanding and SLAM are two fundamental problems in computer vision and robotics. In recent years, there has been great progress in each field. For example, with the popularity of Convolutional Neural Network (CNN), the performance of object detection [1], semantic segmentation [2], and 3D understanding [3] has been improved greatly. In SLAM or Structure from Motion (SfM), approaches such as ORB SLAM [4] and DSO [5] are widely used in autonomous robots and Augmented Reality (AR) applications. However, the connections between visual understanding and SLAM are not well explored. Most existing SLAM methods represent the environments as sparse or semi-dense point cloud, which may not satisfy many applications. For example in autonomous driving, vehicles need to be detected in 3D space for safety and in AR applications, 3D objects and layout planes also need to be localized for virtual interactions.

There are typically two categories of approaches to combine visual understanding and SLAM. The decoupled approach first builds the SLAM point cloud then further labels [6] [7] or detects 3D objects [8] and planes [9], while the coupled approach jointly optimizes the camera pose with object and plane location. Most existing object SLAM [10] [11] requires prior object models to detect and model objects, which limits the application in general environments. Some prior works also utilize architectural planes for dense 3D reconstruction but mostly rely on RGBD [12] or LiDAR scanner [13].

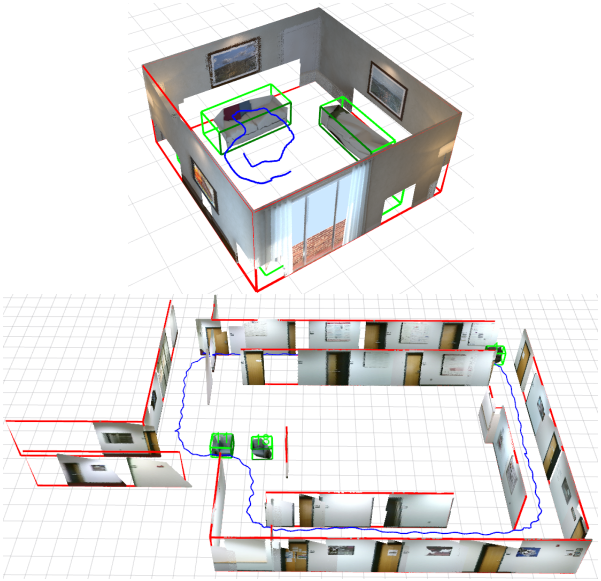


Fig. 1. Example result of dense SLAM map with points, objects (green box), planes (red rectangle) reconstructed using only a monocular camera. (top) ICL living room dataset. (bottom) Collected long corridor dataset.

In this work, we propose a monocular object and plane level SLAM, without prior object and room shape models. It is divided into two steps. The first step is single image 3D structure understanding. Layout plane and cuboid object proposals are generated and optimized based on geometric and semantic image features. The second step is multi-view SLAM optimization. Planes and objects are further optimized with camera poses and point features in a unified bundle adjustment (BA) framework. Objects and planes provide additional semantic and geometric constraints to improve camera pose estimation as well as the final consistent and dense 3D map. Accurate SLAM pose estimation on the other hand improves the single image 3D detection. In summary, our contributions are as follows:

- A high order graphical model with efficient inference for single image 3D structure understanding.
- The first monocular object and plane SLAM, and show improvements on both localization and mapping over state-of-the-art algorithms.

In the following, we first introduce the related work and single image 3D understanding in Sec III, then explain multi-view SLAM optimization in Sec IV, followed by experiments in Sec V.

Manuscript received February 24, 2019; revised May 6, 2019; accepted June 7, 2019. This paper was recommended for publication by Editor Cyrill Stachniss upon evaluation of the reviewers' comments. The work was supported by the Amazon Research Award #2D-01038138. (*Corresponding author: Shichao Yang*)

The authors are with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. Email of first author: {shichaoy@andrew.cmu.edu, 2013ysc@gmail.com}; Second author: basti@andrew.cmu.edu

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org. The enclosed video demonstrates SLAM experimental results.

Digital Object Identifier (DOI): see the top of this page.

II. Related Work

A. Single image understanding

The classic 3D object detection depends on hand-crafted features such as edge and texture [14]. CNNs are also used to directly predict object poses from images [15]. For layout detection, the popular room model based on vanishing point is proposed by Hedau *et al* [16]. There are also some CNN learning based approaches including [17] and RoomNet [3]. Most of them only apply to the restricted four-wall Manhattan room models and are not suitable for general indoor environments.

For the joint 3D understanding of object and planes, Most works mostly utilize RGBD camera and cannot run in real time [18]. More recent works directly predict the 3D position of objects and planes utilizing deep networks [19].

B. Object and Plane SLAM

For object and plane SLAM, the decoupled approach is to first build classic point SLAM then detect 3D objects and planes [8], but it may fail if the point cloud is sparse and not accurate. We here focus on the SLAM which explicitly uses objects and planes as landmarks. Semantic Structure from Motion [20] jointly optimizes various geometry components. Several object based SLAM [10] [11] are also proposed but all depend on the prior object models. The recent Quadric-SLAM [21] and CubeSLAM [22] propose two different object representations for monocular SLAM without prior models. Fusion++ [23] uses RGBD camera to build dense volumetric object models and SLAM optimization.

Concha [24] utilizes superpixel to provide local planar depth constraints in order to generate a dense map from sparse monocular SLAM. Lee [12] estimates the layout plane and point cloud iteratively to reduce mapping drift. Similarly, planes are shown to provide long-range SLAM constraints compared to points in indoor building environments [25] [26]. Recently, [27] proposes similar work to jointly optimize objects, planes, points with camera poses. The difference is that we use a monocular camera instead of RGBD camera and also have different object representations.

III. Single Image Understanding

We represent the environment as a set of cuboid objects and layout planes such as wall and floor. The goal is to simultaneously infer their 3D locations from a 2D image. We first generate a number of object and plane proposals (hypothesis), then select the best subset of them by Conditional Random Field (CRF) optimization, as shown in Fig. 2.

To represent the layout planes, CNNs can directly predict the 3D plane positions but may lose some details as the predicted layout may not exactly match the actual plane boundary. Therefore the large measurement uncertainty makes it unsuitable to be SLAM landmarks. Instead, we directly detect and select ground-wall line segments which are more reliable and reproducible.

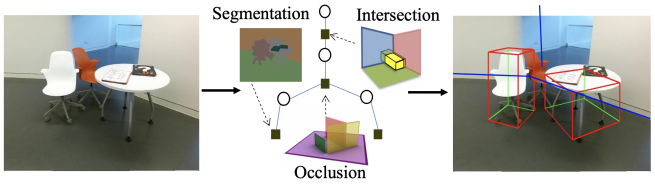


Fig. 2. Overview of single image 3D object and layout detection. We first generate many high-quality object and layout proposals then formulate a graphical model to select the optimal subset based on evidence of semantic segmentation, intersections, occlusions, and so on.

A. Proposal generation

1) Layout Plane Proposal: We first detect all image edges then select some of them close to the ground-wall semantic segmentation [2] boundary. For room environments, layout plane prediction score [17] is additionally used to filter out possible edges. If the edge lies partially inside object regions due to occlusions, we further extend it to intersect with other edges.

2) Object Cuboid Proposal: We follow CubeSLAM [22] to generate cuboid proposals based on 2D object detection and then score proposals based on image edge features. For each object instance, we select the best 15 cuboid proposals for latter CRF optimization. More cuboid proposals may improve the final performance but also increase computation a lot.

B. CRF Model definition

Given all the proposals, we want to select the best subset from them. We assign a binary variable $x_i \in \{0, 1\}$ for each plane and cuboid proposal, indicating whether it will be selected or not. Note that CRF only determines whether it appears or not and doesn't change the proposal's location. The labels are optimized to minimize the following energy function, which is also called potentials in CRF:

$$E(\mathbf{x}|\mathbf{I}) = \sum_i \psi^U(x_i) + \sum_{i<j} \psi^P(x_i, x_j) + \sum_{\mathbf{x_c} < \mathcal{C}} \psi^{Ho}(\mathbf{x_c}) \tag{1}$$

where $\mathbf{x}$ contains all the variables and I is the image. ψ^U and ψ^P are the unary and pairwise potential energy. $\psi^{Ho}(\mathbf{x_c})$ is the high order potential of clique $\mathbf{x_c}$. A clique is a set of variables x_i which have relationships with each other. These potentials are explained in the following:

1) Unary potential: The unary energy indicates the quality of the proposal. For each proposal, we assign negative unary energy to encourage it. Then in the optimization stage, due to the positive potential from pairwise or high order constraints, only part of them can be selected.

For the wall plane edges, the unary cost is determined based on the semantic segmentation. In more detail, from the segmentation, we can find the ground and wall boundary contour denoted as c. Then for each plane proposal edge x_i, we sample ten points on the edge and summarize their distance to the contour denoted as $D(x_i, c)$. To compare different edges effectively, we normalize the distance to $d(x_i, c) \in [0, 1]$. Then the plane edge unary is defined as:

$$\psi^U(x_i) = -w_i x_i \theta(x_i)\,(1 - d(x_i, c)) \tag{2}$$

where w_i is the weight for plane unary. $\theta(x_i)$ is the edge's field of view angle to the camera center. A larger angle usually indicates smaller edge detection error.

For objects, we use the normalized cuboid fitting error explained in [22]. If the cuboid's edges align better with the detected image edges and vanishing points, the unary will be smaller.

2) Pairwise Potential: There are different forms of pairwise relationship between objects and planes, for example the semantic co-occurrence [18]. Here we only utilize the geometric relationship to minimize the 3D occlusion and intersection.

Object-object potential $\psi^P_{\mathcal{O}-\mathcal{O}}$ is defined as the 3D intersection of union between two cuboids as in Eq (3). In the equation, if two proposals both appear, namely $x_i, x_j = 1$, the potential becomes positive otherwise it is zero. Note that there is no pairwise potential between cuboid proposals belonging to the same object instance. Object-plane $\psi^P_{\mathcal{O}-\mathcal{L}}$ potential depends on the volume ratio of the object occluded by plane shown in Eq (4) and Fig. 3(a). Similarly, plane-plane $\psi^P_{\mathcal{L}-\mathcal{L}}$ is defined as the angle overlapping ratio between each other in Eq (5) in Fig. 3(b). Since large plane occlusion is strongly discouraged, an infinite penalty cost is assigned if their overlapping angle is greater than 5°.

$$\psi^P_{\mathcal{O}-\mathcal{O}}(x_i, x_j) = x_i x_j \frac{V(x_i) \bigcap V(x_j)}{V(x_i) \bigcup V(x_j)} \tag{3}$$

$$\psi^P_{\mathcal{O}-\mathcal{L}}(x_i, x_j) = x_i x_j \frac{V^{occ}(x_i)}{V(x_i)} \tag{4}$$

$$\psi^P_{\mathcal{L}-\mathcal{L}}(x_i, x_j) = x_i x_j \frac{A(x_i) \bigcap A(x_j)}{A(x_i) \bigcup A(x_j)} \tag{5}$$

Where $V(x)$ denotes the 3D object volume and $A(x)$ denotes plane angle range to camera center. $\bigcap$ represents intersection and $\bigcup$ is union.

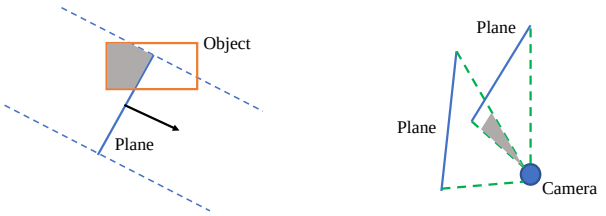


Fig. 3. Top view of potential definition. (a) Object-plane $\psi^P_{\mathcal{O}-\mathcal{L}}$ potential. The grey part of the object is occluded by the plane. The ratio of grey volume is defined as the potential. (b) Plane-plane $\psi^P_{\mathcal{L}-\mathcal{L}}$ potential. The two planes have angle overlapping and occlusion (grey area) between each other. The potential is defined as the ratio of overlapped angle by total angle.

3) High order potential: As explained in Section III-A, for each 2D object instance, many 3D cuboid proposals are generated from it but at most one of them can be selected. Thus these 3D proposals from one object form a clique $\mathbf{x_c}$ and the high order potential is defined as:

$$\psi^{Ho}(\mathbf{x_c}) = \begin{cases} 0 & \text{if } \sum_{x_i \in \mathbf{x_c}} x_i \leq 1 \\ \infty & \text{otherwise} \end{cases} \tag{6}$$

C. Efficient CRF inference

Efficient inference of high order discrete CRFs is still a challenging problem [28]. We observe that the high order term in Eq 6 is very sparse because at most one variable can be 1 in one clique $\mathbf{x_c}$. We therefore design efficient inference based on max-product loopy belief propagation [29]. After the iterative message passing, we select the state with minimum potential as the final result. The computationally expensive part is the message from clique c to variable node i:

$$m^t_{c\to i}(x_i) = \min_{\mathbf{x}_c^{-i}} \left(f_c(\mathbf{x}_c) + \sum_{j \in c \setminus \{i\}} m^{t-1}_{j\to c}(x_j) \right) \tag{7}$$

where $\mathbf{x}_c^{-i}$ denotes all the variables in clique c except variable i. t and $t-1$ represents different iteration steps. $m^{t-1}_{j\to c}(x_j)$ is the message from node to clique. For a clique with N binary nodes, there are totally 2^N clique states of $\mathbf{x}_c$. However there are only $N+1$ valid states in our problem $\{1,0,...0\},...\{0,0,...1\},\{0,0,...0\}$ denoted as $\{\mathbf{y}_1, \mathbf{y}_2, ...\mathbf{y}_{N+1}\}$. Therefore, we only need to check $N+1$ states and find the minimum in Eq 7. We can further observe that every adjacent state vector $\mathbf{y}_i$ only has two different variables, therefore $\sum_{j\in c\setminus\{i\}} m^{t-1}_{j\to c}(x_j)$ for each $\mathbf{y}_i$ can be computed iteratively. The average time complexity of computing $m^t_{c\to i}(x_i)$ is $O(1)$ instead of the naive $O(2^N)$. More details can be found at the appendix.

IV. SLAM Optimization

The selected object and plane proposals from single image detection are used as SLAM landmarks and optimized together with camera poses through multi-view BA. We also include points in SLAM because there are usually only a few objects and planes in the environments and they cannot fully constrain camera poses. In the following, we first formulate the optimization problem, then explain the parameterization and various measurement costs.

A. Bundle Adjustment Formulation

Consider a set of camera poses $C = \{c_i\}$, 3D objects $O = \{o_j\}$, planes $\Pi = \{\boldsymbol{\pi}_k\}$ and points $P = \{p_m\}$, bundle adjustment can be formulated as nonlinear least squares optimization problem:

$$C^*, O^*, \Pi^*, P^* = \arg \min_{\{C,O,\Pi,P\}} \sum_{i\in C, j\in O, k\in\Pi, m\in P} \mathbf{e}^{\mathbf{T}} \Sigma \mathbf{e} \tag{8}$$

where $\mathbf{e}$ is the measurement error between each other. Σ is covariance matrix of different error measurements. The optimization problem can be solved by Gauss-newton or Levenberg-Marquardt algorithm in many libraries such as g2o and iSAM.

B. Parameterization

For camera pose and point, we utilize the standard form $T_c \in SE(3)$ and $P \in \mathbb{R}^3$. The cuboid pose is similarly defined in [22] by 9 DoF parameters: $O = (T_o, D)$, where $T_o \in SE(3)$ is 3D object pose, and $D \in \mathbb{R}^3$ is dimensions.

We adopt the infinite plane representation [30] $\boldsymbol{\pi} = (\mathbf{n}^\top, d)^\top$ st. $\|\boldsymbol{\pi}\| = 1$. $\mathbf{n}$ is the plane normal and d is the plane distance to the world origin. In some environments, we use the Manhattan assumptions, namely the plane normal is fixed and parallel to one of the world frame axes, therefore only d is needed to represent it.

C. Measurements

Different measurement functions between the map components are proposed to formulate factor graph optimization. Camera-point observation model is the standard point reprojection error [4]. We here explain the new measurements in more detail.

1) Camera-plane: Different from RGBD based plane SLAM which can directly get plane measurement from point cloud plane fitting [27] [30], we need to back-project 2D plane edge l to the 3D space to get the measurement shown as the blue plane in Fig. 4(a), then compare it with the grey plane landmark plane $\boldsymbol{\pi}$ using log quaternion error:

$$e_{cp} = \| \log \left(\boldsymbol{\pi}_{obs}(l), T_c^T \boldsymbol{\pi} \right) \| \tag{9}$$

Note that in Eq 9, we transform the global plane landmark to camera frame by $T_c^T \boldsymbol{\pi}$ instead of comparing them in the world frame. This is because when camera moves far away from world origin, plane parameter d becomes very large compared to normal $\mathbf{n}$ and dominate the error.

For the back-projection process, suppose K is the camera intrinsic calibration and p is one of the endpoints of edge l, then its corresponding 3D point P is the intersection of back projected ray $K^{-1}p$ with the ground plane $(\mathbf{n}_g^\top, d_g)$ in camera frame:

$$P = \frac{-d_g}{\mathbf{n}_d^\top (K^{-1}p)} K^{-1} p \tag{10}$$

Similarly we can compute the other endpoint and get the 3D vertical wall plane $\boldsymbol{\pi}_{obs}$ passing through the two points. We can find that this process depends on the camera pose to the ground plane. Therefore, we need to update it each iteration during the optimization.

2) Camera-object: We follow the cuboid observation functions defined in the prior work [22]. The cuboid landmark's 3D corners are first projected onto the image plane then a 2D rectangle hull is found shown as the red rectangle in Fig. 4(b). Then it is compared with the blue actual detected 2D bounding box:

$$e_{2D} = \| \ [\mathbf{c}, \mathbf{s}] - [\mathbf{c_m}, \mathbf{s_m}] \ \|_2 \tag{11}$$

where $[\mathbf{c}, \mathbf{s}]$ is the center and dimension of the 2D box. This 2D measurement error has much less uncertainty compared to 3D cuboid error as explained in [22]. To make the optimization robust, we assign different weights to different objects based on their distance to camera and 2D semantic object detection confidence.

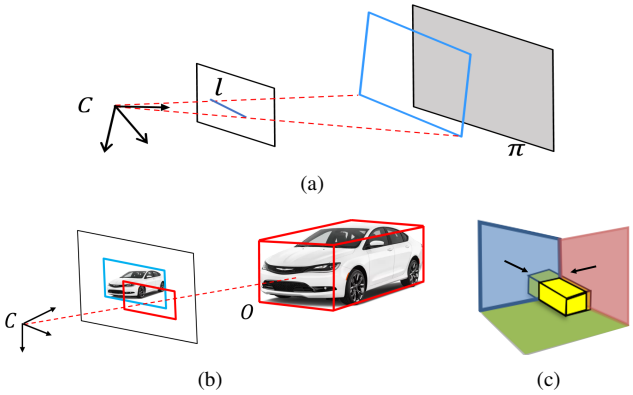


Fig. 4. SLAM observation functions. (a) Camera-plane observations. The detected ground edge is back-projected to 3D space to compare with landmark plane. (b) Camera-object observations. 3D cuboid landmark is projected onto images and compared with the detected 2D box. (c) Object-plane measurement error depends on the object volume occluded by planes.

3) Object-plane: There are different forms of object-plane constraints depending on the environment assumptions for example objects are supported by planes [27] or object orientation matches the nearby plane normal. We here propose a weaker but more general constraint that objects should not be occluded by nearby planes in the camera view shown in Fig. 4(c). The error is the sum of 3D corners' signed distance to plane:

$$e_{op} = \sum_{i=1:8} \max(0, -\boldsymbol{\pi} P_{oi}) \tag{12}$$

where P_{oi} is one of the eight cuboid corners. If the cuboid lies on the positive side of the plane meaning that there is no occlusion, e_{op} will be zero.

4) Point-plane: If a feature point belongs to a plane region, we also add a constraint of the point's 3D distance to plane. However, it is usually difficult to accurately determine if a point belongs to a plane from 2D image as layout planes are usually the background and points may belong to the foreground objects. To improve the robustness, we first select feature points in the 2D wall plane polygon then filter out points that are farther away from the 3D plane than a threshold. The point-plane error is defined as:

$$e_{pp} = \|\boldsymbol{\pi} P\|_2 \tag{13}$$

Note that to be robust to outliers, huber loss is applied to all above error functions.

D. Data association

Data association for different landmarks across multiple views is necessary to build a SLAM graph. For point association, we use the point feature matching in ORB SLAM [4]. Object association follows the work of CubeSLAM [22]. Basically, each object contains a set of feature points belonging to it, then we can find object matching which has the most number of shared map points in different views. This approach is easy to implement and can effectively handle occlusion, repetitive textures and dynamic movement.

For the plane association, we first check whether the plane normal difference is within 30° and distance to each other is smaller than $1m$. We then find the plane matching with the most shared feature points similar to object matching. In Sec IV-C4, we already determine which feature points belong to the specific plane.

V. Experiments

A. Implementation details

For object detection, we use similar settings as object SLAM in [22]. Yolo [1] detector is used for the 2D object detection. For plane detection, we first detect line segments using [31] and merge them to long edges. Segnet [2] is used for the 2D semantic segmentation. We then filter out lines whose length is shorter than 50 pixels and more than 50 pixels away from the wall-ground segmentation boundary. In video case, SLAM pose estimation is used in the single image 3D detections.

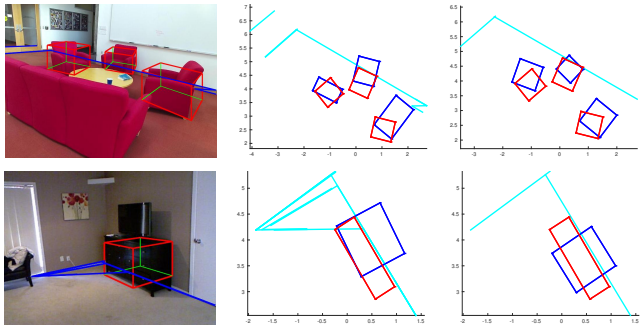
For the SLAM part, our system is built on the feature point based ORB SLAM, augmented with our objects and planes. We compute jacobians of new observation functions then perform BA using g2o library. Since the outlier associations and measurements of objects and planes have more severe effects on the optimization compared to outlier points, strict outlier rejections have to be utilized. In our system, the object and plane landmark will be deleted if it has not been observed by 3 frames in recent 15 frames after creation or if there are less than 10 stable feature points associated with it. In most of the room environments, we use the Manhattan plane representation with a fixed surface normal as mentioned in Section IV-B to provide more constraints on the planes and improve the overall performance. If the initially generated wall surface normal difference with Manhattan direction exceeds 30 degrees, it will also be treated as an outlier.

In addition to being used as SLAM landmarks, objects and planes also provide depth initialization for those feature points which are difficult to triangulate due to small motion parallax. When less than 30% of the feature points are matched to map points, we create some new map points directly using depth from objects and planes. This can improve monocular SLAM performance in low texture environments and large rotation scenarios. Compared to the prior work of monocular plane SLAM [26], ground plane is not used in this work because there is no actual edge measurement corresponding to the ground plane.

For the final dense map generation, we back-project pixels in the plane regions onto the optimized plane landmarks. For feature points belonging to objects, we create triangular meshes in 3D space to get a dense 3D model. Note that in the SLAM optimization, planes are represented as infinite planes, but for visualization purposes, we also keep track of the plane boundary polygon.

B. Single Image Result

We first show the single image object and plane result. Some examples of proposal generation and CRF optimization are shown in Fig. 5. The middle and right columns show the top view of object proposals before and after CRF optimization.



(a) (b) (c)

Fig. 5. Single image raw proposal generation and CRF optimization illustrations. (a) Raw plane and object proposals. (only one cuboid proposal is drawn for brevity) (b) Top view of raw proposals. The red rectangle is ground truth objects and blue are the estimated. Cyan lines are wall plane proposals. (c) Top view of CRF selected proposals. Object poses are more accurate after optimization. Plane and object intersection and occlusion is also reduced.

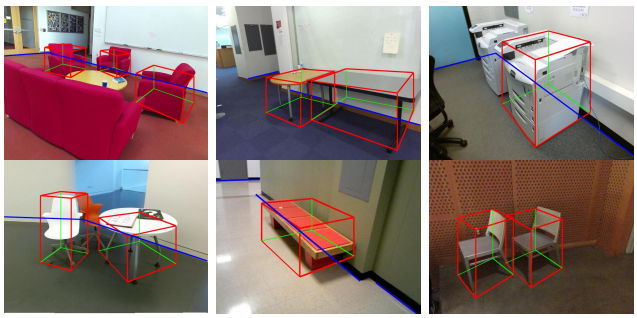


Fig. 6. More single image 3D detection results, where blue line represents the wall plane edge and the box represents the object.

We can see that CRF can select non-overlapped wall edges and better cuboid proposals to reduce occlusion and intersection. Since CRF only selects the proposals without changing their actual locations, there might still be occlusion after optimization.

More results of CRF selected object and plane proposals are shown in Fig. 6. The algorithm is able to work in different environments including rooms and corridors but it may still miss some planes and objects when there is severe object occlusion and unclear edges for example in the right column of Fig. 6.

We also evaluate quantitatively on the SUN RGBD dataset using the 3D object intersection over union (IoU) as the metric. We select 1670 images with visible ground planes and ground objects fully in the field of view, then compare with the prior work [22], the latest deep network based scene understanding [19] and two other model-based algorithms: SUN primitive [32] and 3D Geometric Phrases (3dgp) [33]. Note that ground truth camera pose is used in our method as well as [32] [33]. Huang *et al.* [19] predict camera pose and layouts jointly so it is difficult to modify their algorithm to use the provided camera pose. From Table I, our prior work [22] performs similar to other work. [19] performs worse because it utilizes the predicted camera pose therefore it is not quite comparable. Compared to [32] [33], our method detects much more objects. The proposed CRF joint optimization improves

TABLE I
3D OBJECT IOU ON SUN RGBD SUBSET DATA

Method	Huang [19]	Xiao [32]	3dgp [33]	Ours [22]	Our CRF
3D IoU	0.27	0.30	0.35	0.38	0.43

(a) (b) (c)

Fig. 7. (a) Layout prediction score map [17] (b) Semantic segmentation by [2] (c) Our single image object and plane detections. It cannot detect the occluded wall surface while multi-view SLAM can build a complete map in Fig. 1, demonstrating the advantage of our multi-view object SLAM.

the IoU by 5% compared to [22]. Note that to emphasize the optimization effect, we only evaluate on images where CRF generates different results compared to the single image detection. This is because in most images, there are no wall visible planes or planes are far from objects, therefore planes have no constraints on object positions.

C. SLAM Result

We then evaluate the SLAM tracking and mapping performance on both public datasets ICL-NUIM [34], TAMU Indoor [35], TUM mono [36], and our collected datasets by KinectV2 sensor.

1) Qualitative Results: A sample frame of ICL sequence is shown in Fig. 7. The left and middle images show the raw image overlayed by layout prediction and the semantic segmentation. Both of them have noise and our CRF optimization in Fig. 7(c) shows a roughly correct 3D model but it cannot fully detect the occluded wall segments. After the multi-view SLAM optimization, the algorithm is able to build a more consistent and complete map shown in Fig. 1.

More examples of 3D mapping and camera pose estimation in different datasets and environment configurations are shown in Fig. 8. The green box is the object location and red rectangle is the plane boundary. After BA, objects and planes' locations are more accurate compared to the single view detection and most objects lie inside the room. Note that not all objects are mapped because the 2D object detector might miss some and SLAM might also classify some of them as outliers due to inconsistent observations. In some scenarios such as the top left of Fig. 8, our algorithm cannot detect the complete wall plane due to severe object occlusions. To improve the visualization robustness, if there are not enough map points observed in some region of a plane polygon, the pixels won't be back-projected to generate dense maps, shown as the missing segments on the wall surface.

2) Quantitative Results: We then show the quantitative camera pose comparison with ORB SLAM and DSO. For datasets in Table II, the initial maps of both ORB SLAM and ours are scaled by the ground truth initial camera height. Then we can directly evaluate the absolute translation error without aligning the pose in scale, to show that object and planes can improve the pose estimation and reduce monocular drift. Each algorithm runs 5 times in each sequence and the mean error is reported here. From the table, we can see that in most of the scenarios, the added object and plane landmark constraints improve the camera pose estimation. There are two main reasons for this. One is that even though there is no explicit loop closure, due to object and plane's long-range visibility properties, the algorithm may still associate with the old plane landmark to reduce the final drift. The second reason is that more feature points' depth can be initialized by object and planes especially when there is large camera rotations. Due to the strict outlier rejection and robust BA optimization, even if objects and planes don't improve the results, they won't seriously damage the system.

TABLE II
ABSOLUTE CAMERA TRANSLATION ERROR ON VARIOUS DATASETS (cm FOR ICL, m FOR OTHERS)

Method	ORB [4]	ORB-No LC [4]	Ours
ICL living 0	2.35	3.08	**0.8**
ICL living 2	3.54	3.25	**2.06**
ICL living 3	**4.68**	5.36	5.38
ICL office 0	**5.67**	6.23	5.93
ICL office 2	3.82	5.00	**2.63**
Tamu corridor	1.74	3.87	**0.97**
Our room 1	0.14	0.15	**0.05**
Our corridor 1	1.49	2.25	**0.30**
Our corridor 2	1.05	2.93	**0.24**
Our corridor 3	0.87	1.84	**0.49**

From the table, we also find that loop closure in ORB SLAM has some benefits in small office environments, but in large corridors with loops at the sequence end, it doesn't perform well compared to ours, because SLAM already has large scale drift before the loop closure, the final global BA cannot fully recover the drift.

For TUM mono data in Table III, there is no ground truth camera height available thus we evaluate the monocular scale alignment error proposed in [5]. Results of DSO and ORB are taken from the supplementary material of DSO. Our semantic SLAM can work robustly in these challenging datasets even though there is large camera rotation and sometimes the camera may be upside down. In the cluttered dataset such as Room 37, there are only a few planes in a few observed frames thus our algorithm almost reduces to point SLAM and achieves. In Corridor 38, our algorithm and ORB SLAM are much worse compared to DSO because there are many areas of only one white wall with few feature points which are difficult for feature and plane based SLAM.

3) Time Analysis: We also provide the run-time analysis on Intel i7-4790 CPU at 4.0 GHz and Nvidia 980 Ti GPU. GPU is used for 2D object detector and semantic segmentation. All SLAM parts are implemented in C++ on CPU. As shown in Table IV, there are several single image pre-processing steps. The CNN algorithms we used cannot run in real time but they actually depend on the model complexity which can be

TABLE III
POSE ALIGNMENT ERROR ON TUM-MONO DATASET

Method	ORB-No LC [4]	DSO [5]	Ours
Corridor 36	1.81	4.01	**0.94**
Room 37	0.60	0.55	**0.35**
Corridor 38	23.9	**0.55**	7.65

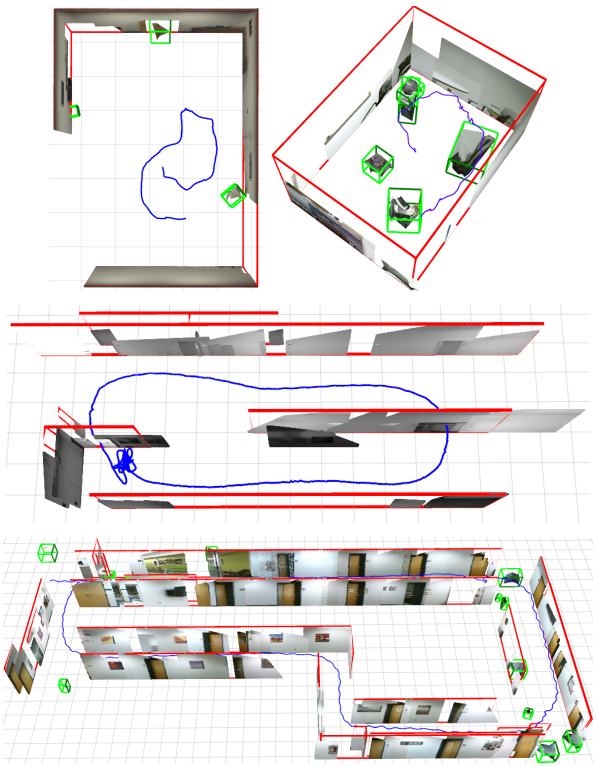


Fig. 8. More dense mapping results with objects and planes. (top) ICL-NUIM office 2 and collected room data. (middle) TUM-mono 36. (bottom) Our collected long corridor. The red rectangle is the plane boundary and green cuboid is the object. The blue curve is the estimated camera trajectory.

replaced by recent lightweight CNNs.

The SLAM experiment runs on ICL-NUIM living room dataset. On average, there are 5 object landmarks in each local BA optimization. The tracking thread includes feature detection, associations, and camera pose tracking for each frame which can run in real time from the table. The BA map optimization occurs when a new keyframe is created, therefore it does not need to run in real-time. Compared to point only BA, adding objects into the system only increases the optimization time by 7%. Plane landmarks further double the optimization time because the point-plane constraints are applied to many points, bringing in more measurement costs to optimize. Another reason relates to the implementation of g2o. Since there are different types of edges with different dimensions such as camera-point, point-plane, we cannot pre-allocate the solver matrix dimensions.

VI. CONCLUSION

In this work, we propose the first monocular SLAM and dense mapping algorithm combining points with high-level object and plane landmarks through unified BA optimization. We show that semantic scene understanding and traditional SLAM optimization can improve each other.

TABLE IV
AVERAGE RUNTIME OF DIFFERENT SLAM COMPONENTS

Dataset	Tasks	Runtime (mSec)
Single image Preprocessing	Yolo 2D object detection	17.5
	SegNet semantic segmentation	71.5
	Edge detection	12.1
Indoor ICL room	Tracking thread	15.0
	Point only BA	49.5
	Point + object BA	55.3
	Point + object + plane BA	105.6

For the single image, we propose a fast 3D object and layout joint understanding for general indoor environments. Cuboid and plane proposals are generated from 2D object and edge detection. Then an efficient sparse high order CRF inference is proposed to select the best proposals. In the SLAM part, several new measurement functions are designed for planes and objects. Compared to points, objects and planes can provide long-range geometric and semantic constraints such as intersection and supporting relationships, to improve the pose estimation. Strict outlier rejection, robust data association and optimization are proposed to improve the robustness.

We evaluate the SLAM algorithm in various public indoor datasets including rooms and corridors. Our approach can improve the camera pose estimation and dense mapping in most environments compared to the state-of-the-art.

In the future, more general planes in addition to wall planes need to be considered to produce a denser and more complete map. Dynamic objects and object surface mapping can also be addressed to improve the robustness and mapping quality.

APPENDIX

We here explain the CRF inference of Section III-C in more detail shown in Algorithm 1. As mentioned before, there are $N+1$ special state for a clique with size N. For each state $\mathbf{y}_k$, we define $\mathbf{s}_k = \sum_{j \in \mathbf{y}_k} m_{j \to c}^{t-1}(y_k^j)$ as the sum of messages to the clique. The key observation is that $\mathbf{s}_k$ can be computed iteratively and efficiently.

REFERENCES

[1] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. *Computer Vision and Pattern Recognition (CVPR)*, pages 7263–7271, 2017.

[2] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. SegNet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[3] Chen-Yu Lee, Vijay Badrinarayanan, Tomasz Malisiewicz, and Andrew Rabinovich. Roomnet: End-to-end room layout estimation. In *IEEE International Conference on Computer Vision*, 2017.

[4] Raul Mur-Artal, JMM Montiel, and Juan D Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[5] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, 2017.

[6] Abhijit Kundu, Yin Li, Frank Dellaert, Fuxin Li, and James M Rehg. Joint semantic segmentation and 3D reconstruction from monocular video. In *European Conference on Computer Vision (ECCV)*, pages 703–718. Springer, 2014.

[7] Shichao Yang, Yulan Huang, and Sebastian Scherer. Semantic 3D occupancy mapping through efficient high order CRFs. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 590–597, 2017.

[8] Sudeep Pillai and John Leonard. Monocular SLAM supported object recognition. *Robotics: Science and systems*, pages 34–42, 2015.

[9] Sid Yingze Bao, Axel Furlan, Li Fei-Fei, and Silvio Savarese. Understanding the 3D layout of a cluttered room from multiple images. In *IEEE Winter Conference on Applications of Computer Vision*, pages 690–697. IEEE, 2014.

[10] Renato F Salas-Moreno, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly, and Andrew J Davison. SLAM+: Simultaneous localisation and mapping at the level of objects. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, 2013.

[11] Dorian Gálvez-López, Marta Salas, Juan D Tardós, and JMM Montiel. Real-time monocular object SLAM. *Robotics and Autonomous Systems*, 75:435–449, 2016.

[12] Jeong-Kyun Lee, Jaewon Yea, Min-Gyu Park, and Kuk-Jin Yoon. Joint layout estimation and global multi-view registration for indoor reconstruction. In *IEEE International Conference on Computer Vision (ICCV)*, pages 162–171, 2017.

[13] Jianxiong Xiao and Yasutaka Furukawa. Reconstructing the worlds museums. *International journal of computer vision*, 110(3):243–258, 2014.

[14] Joseph J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing IKEA objects: Fine pose estimation. In *IEEE International Conference on Computer Vision*, pages 2992–2999, 2013.

[15] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 292–301, 2018.

[16] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the spatial layout of cluttered rooms. In *International Conference on Computer Vision*, pages 1849–1856. IEEE, 2009.

[17] Yuzhuo Ren, Shangwen Li, Chen Chen, and C-C Jay Kuo. A coarse-to-fine indoor layout estimation (CFILE) method. In *Asian Conference on Computer Vision*, pages 36–51. Springer, 2016.

[18] Dahua Lin, Sanja Fidler, and Raquel Urtasun. Holistic scene understanding for 3D object detection with RGBD cameras. In *IEEE International Conference on Computer Vision*, pages 1417–1424, 2013.

[19] Siyuan Huang, Siyuan Qi, Yinxue Xiao, Yixin Zhu, Ying Nian Wu, and Song-Chun Zhu. Cooperative holistic scene understanding: Unifying 3d object, layout, and camera pose estimation. In *Advances in Neural Information Processing Systems*, pages 207–218, 2018.

[20] Sid Yingze Bao, Mohit Bagra, Yu-Wei Chao, and Silvio Savarese. Semantic structure from motion with points, regions, and objects. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2703–2710. IEEE, 2012.

[21] Lachlan James Nicholson, Michael J Milford, and Niko Sunderhauf. QuadricSLAM: Dual quadrics from object detections as landmarks in object-oriented SLAM. *IEEE Robotics and Automation Letters*, 4(1):1–8, 2019.

[22] Shichao Yang and Sebastian Scherer. CubeSLAM: Monocular 3D object SLAM. *IEEE Transactions on Robotics*, 2019.

[23] John McCormac, Ronald Clark, Michael Bloesch, Andrew Davison, and Stefan Leutenegger. Fusion++: Volumetric object-level SLAM. In *International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018.

[24] Alejo Concha and Javier Civera. DPPTAM: Dense piecewise planar tracking and mapping from a monocular sequence. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 5686–5693. IEEE, 2015.

[25] Ming Hsiao, Eric Westman, Guofeng Zhang, and Michael Kaess. Keyframe-based dense planar SLAM. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5110–5117. IEEE, 2017.

[26] Shichao Yang, Yu Song, Michael Kaess, and Sebastian Scherer. Pop-up SLAM: a semantic monocular plane SLAM for low-texture environments. In *International conference on Intelligent Robots and Systems (IROS)*, pages 1222–1229. IEEE, 2016.

[27] Mehdi Hosseinzadeh, Yasir Latif, Trung Pham, Niko Suenderhauf, and Ian Reid. Structure aware SLAM using quadrics and planes. *Asian Conference on Computer Vision*, pages 410–426, 2018.

[28] Chaohui Wang, Nikos Komodakis, and Nikos Paragios. Markov random field modeling, inference & learning in computer vision & image understanding: A survey. *Computer Vision and Image Understanding*, 117(11):1610–1627, 2013.

[29] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[30] Michael Kaess. Simultaneous localization and mapping with infinite planes. In *International Conference on Robotics and Automation (ICRA)*, pages 4605–4611. IEEE, 2015.

[31] Rafael Grompone von Gioi, Jeremie Jakubowicz, Jean-Michel Morel, and Gregory Randall. LSD: A fast line segment detector with a false detection control. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (4):722–732, 2008.

[32] Jianxiong Xiao, Bryan Russell, and Antonio Torralba. Localizing 3D cuboids in single-view images. In *Advances in neural information processing systems (NIPS)*, pages 746–754, 2012.

[33] Wongun Choi, Yu-Wei Chao, Caroline Pantofaru, and Silvio Savarese. Understanding indoor scenes using 3D geometric phrases. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 33–40, 2013.

[34] A. Handa, T. Whelan, J.B. McDonald, and A.J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *IEEE Intl. Conf. on Robotics and Automation, ICRA*, pages 1524–1531, 2014.

[35] Yan Lu and Dezhen Song. Robustness to lighting variations: An RGB-D indoor visual odometry using line segments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 688–694, 2015.

[36] J. Engel, V. Usenko, and D. Cremers. A photometrically calibrated benchmark for monocular visual odometry. In *arXiv:1607.02555*, July 2016.

Algorithm 1: CRF Factor-to-Variable Message Passing

Input: Variable-to-factor message $m_{i\to c}^{t-1}(\cdot), i = 1, ..., N$

Output: All Factor-to-variable message:
$m_{c\to i}^{t}(x_i), i = 1, ..., N$

// compute all $\mathbf{s}_k$ recursively and record min

$\mathbf{s}_1 = \sum_{j=1,2,3...N} m_{j\to c}^{t-1}(\mathbf{y}_1^j)$

minS $= \mathbf{s}_1$; sndminS $= \infty$

for $k \leftarrow 2$ **to** $N+1$ **do**

  if $k \leq N$ **then**

    $\mathbf{s}_k = \mathbf{s}_{k-1} - m_{k-1\to c}^{t-1}(1) + m_{k-1\to c}^{t-1}(0) - m_{k\to c}^{t-1}(0) + m_{k\to c}^{t-1}(1)$

  else

    $\mathbf{s}_k = \mathbf{s}_{k-1} - m_{k-1\to c}^{t-1}(1) + m_{k-1\to c}^{t-1}(0)$

  end

  if $\mathbf{s}_k \leq$ *minS* **then**

    minS $= \mathbf{s}_k$

  else

    if $\mathbf{s}_k \leq$ *sndminS* **then**

      sndminS $= \mathbf{s}_k$

    end

  end

end

// compute final message

for $k \leftarrow 1$ **to** N **do**

  $m_{c\to k}^{t}(x_k = 1) = \mathbf{s}_k - m_{k\to c}^{t-1}(x_k = 1)$

  if $\mathbf{s}_k ==$ *minS$_{val}$* **then**

    $m_{c\to k}^{t}(x_k = 0) =$ sndminS $- m_{k\to c}^{t-1}(x_k = 0)$

  else

    $m_{c\to k}^{t}(x_k = 0) =$ minS $- m_{k\to c}^{t-1}(x_k = 0)$

  end

end