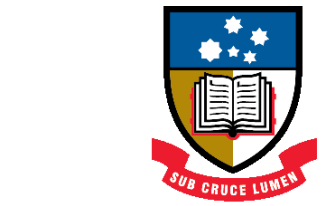
Real-Time Structure and Object Aware Semantic SLAM



THE UNIVERSITY of ADELAIDE

Mehdi Hosseinzadeh

School of Computer Science
The University of Adelaide

This dissertation is submitted for the degree of
Doctor of Philosophy

Supervisors:
Prof. Ian Reid
Prof. Anton van den Hengel

February 2019

To my family,
my beloved wife,
who made all of this possible,
for their endless encouragement and patience.

Declaration

I certify that this work contains no material which has been accepted for the award of any other degree or diploma in my name, in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made in the text. In addition, I certify that no part of this work will, in the future, be used in a submission in my name, for any other degree or diploma in any university or other tertiary institution without the prior approval of the University of Adelaide and where applicable, any partner institution responsible for the joint-award of this degree.

I acknowledge that copyright of published works contained within this thesis resides with the copyright holder(s) of those works.

I also give permission for the digital version of my thesis to be made available on the web, via the University's digital research repository, the Library Search and also through web search engines, unless permission has been granted by the University to restrict access for a period of time.

Mehdi Hosseinzadeh
February 2019

Acknowledgements

First and foremost, I would like to thank my principal supervisor, Prof. Ian Reid, for inspiring this work, for giving me invaluable guidance, and for his tremendous support during my PhD. Reaching this target would not be conceivable without his trust, encouragement, and insights. I have been extremely lucky to have him as my supervisor and mentor in this journey.

I would like to thank Dr. Yasir Latif for his generous and invaluable support and mentoring during this research. I appreciate his immense guidance towards this work. My sincere thanks also goes to Dr. Trung Pham, Dr. Alireza Khosravian, Dr. Hamid Rezatofighi for the great discussions, insights, and advice, Dr. Niko Sünderhauf for the useful remote meetings.

Many thanks also goes to all my former and present lab mates, for the fruitful discussions, helpful supports, and unforgettable memories. I especially thank my great friends Dr. Saroj Weerasekera, and Dr. Gabriel Maicas Suso. Also many thanks to Kejie Li, Ming Cai, Huangying Zhang.

I am grateful for the opportunity that was given to me to be a part of a great research team in the School of Computer Science at the University of Adelaide, the Australian Centre for Robotic Vision, the Australian Institute for Machine Learning (formerly Australian Centre for Visual Technologies).

Last but not least, I am immensely grateful to my parents and family for their support, my lovely wife, for her generous and timeless support, and patience during this journey.

Abstract

Simultaneous Localization And Mapping (SLAM) is one of the fundamental problems in mobile robotics and addresses the reconstruction of a previously unseen environment while simultaneously localising a mobile robot with respect to it. For visual-SLAM, the simplest representation of the map is a collection of 3D points that is sparse and efficient to compute and update, particularly for large-scale environments, however it lacks semantic information and is not useful for high-level tasks such as robotic grasping and manipulation. Although methods to compute denser representations have been proposed, these reconstructions remain equivalent to a collection of points and therefore carry no additional semantic information or relationship. Man-made environments contain many structures and objects that carry high-level semantics and can potentially act as landmarks of a SLAM map, while encapsulating semantic information as opposed to a set of points. For instance, planes are good representations for feature deprived regions, where they provide information complimentary to points and can also model dominant planar layouts of the environment with very few parameters. Furthermore, a generic representation for previously unseen objects can be used as a general landmark that carries semantics in the reconstructed map. Integrating visual semantic understanding and geometric reconstruction has been studied before, however due to various reasons, including high- level geometric entities in the SLAM framework has been restricted to a slow, offline structure-from-motion context, or high-level entities merely act as regulators for points in the map instead of independent landmarks. One of those critical reasons is the lack of proper mathematical representation for high-level landmarks and the other main reasons are the challenge of detection and tracking of these landmarks and formulating an observation model – a mapping between corresponding image observable quantities and estimated parameters of the representations.

In this work, we address these challenges to achieve an online real-time SLAM framework with scalable maps consisting of both sparse points and high-level structural and semantic landmarks such as planes and objects. We explicitly target real-time

performance and keep that as a beacon which influences critically the representation choice and all the modules of our SLAM system. In the context of factor graphs, we propose novel representations for structural entities as planes and general unseen and not-predefined objects as bounded dual quadrics that decompose to permit clean, fast and effective real-time implementation that is amenable to the nonlinear least-square formulation and respects the sparsity pattern of the SLAM problem. In this representation we are not concerned with high-fidelity reconstruction of individual objects, but rather to represent the general layout and orientation of objects in the environment. Also the minimal representations of planes is explored leading to a representation that can be constructed and updated online in a least-squares framework. Another challenge that we address in this work is to marry high-level landmark detections based on deep-learned frameworks, with geometric SLAM systems. Due to the recent success of CNN-based object detections and also depth and surface normal estimations from single image, it is feasible now to detect and estimate these semantic landmarks from single RGB images, therefore leading us seamlessly from RGB-D SLAM system to pure monocular SLAM thanks to the real-time predictions of the trained CNN and appropriate representations. Furthermore, to benefit from deep-learned priors, we incorporate high-fidelity single-image reconstructions and hallucinations of objects on top of the coarse quadrics to enrich the sparse map semantically, while constraining the shape of the coarse quadrics even more. Pertinent to our beacon, proposed landmark representations in the map also provide the potential for imposing additional constraints and priors that carry crucial semantic information about the scene, without incurring great extra computational cost. In this work, we have explored and proposed constraints such as priors on the extent and shape of the objects, point-plane regularizer, plane-plane (Manhattan assumption), and plane-object (supporting affordance) constraints.

We evaluate our proposed SLAM system extensively using different input sensor modalities from RGB-D to monocular in almost all publicly available benchmarks both indoors and outdoors to show its applicability as a general-purpose SLAM solution. The extensive experiments show the efficacy of our SLAM through different comparisons and ablation studies including high-level structures and objects with imposed constraints among them in various scenarios. In particular, the estimated camera trajectories have been improved significantly in varied sequences of visual SLAM datasets and also our own captured sequences with UR5 robotic arm equipped with a depth camera. In addition to more accurate camera trajectories, our system yields enriched sparse maps

with semantically meaningful planar structures and generic objects in the scene along with their mutual relationships.

Publications

This thesis contains the following works that have been published:

- **M. Hosseinzadeh**, Y. Latif, I. Reid, "Sparse Point-Plane SLAM", *ACRA 2017*.
- **M. Hosseinzadeh**, Y. Latif, T. Pham, N. Sünderhauf, I. Reid, "Structure Aware SLAM using Quadrics and Planes" (Spotlight Talk) *RSS 2018* – Learning and Inference in Robotics Workshop.
- **M. Hosseinzadeh**, Y. Latif, T. Pham, N. Sünderhauf, I. Reid, "Structure Aware SLAM using Quadrics and Planes", *ACCV 2018*.
- **M. Hosseinzadeh**, K. Li, Y. Latif, I. Reid, "Real-Time Monocular Object-Model Aware Sparse SLAM", *ICRA 2019*.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

In this chapter we motivate the research with a brief background and overview of our problem in this thesis. The summary of the key contributions and the outline of the thesis comes at the end.

1.1 Motivation

As the evolution of eye in species led to a visually guided intelligent behavior, the computer vision and robotics researchers aim to create the ability of visual understanding for robots and artificial agents in order to make their real-world interactions more intelligent. Creating robots that "see and understand" the world is not conceivable without two fundamental capabilities: understanding the scene *geometrically* and interpreting it *semantically*.

Geometric understanding or reconstruction of the scene is the ability to infer and retrieve the three-dimensional structure of the scene using only the perceived two-dimensional projections of the world on image frames, while localizing with respect to the modeled map of the scene. This geometric modeling is essential for any complex task that involves physical interactions with the real world, such as navigation and motion planning in different types of mobile robots from domestic vacuum cleaners to industrial robotic arms, autonomous cars, mining vehicles, or even modern applications such as Augmented Reality (AR). On the other hand, semantic scene understanding is the ability to infer any higher level semantics or relational information out of the 2D observations of the scene. Again all complicated visual robotic or virtual tasks require the level of semantic understanding close to human level in order to identify various objects in the scene and infer their relationships. For instance in the recent appealing

area of autonomous vehicles, the systems are expected to identify pedestrians or other cars in order to reason about their behavior and control the car accordingly to avoid any mishap.

Geometric reconstruction of the environment emerges in different problem setups due to various reasons, such as sensor modalities, online or offline streams of captured images, map representations, etc. It typically involves multi-views of the scene in order to compensate for the inherent ambiguity of the projected observations or problems such as occlusions. The main categories of the geometric reconstruction problem are Simultaneous Localization and Mapping (SLAM) and Structure from Motion (SfM) methods. In the latter methods, input captured images are processed in an offline manner, while visual SLAM methods operate in an online incremental fashion and refine the reconstructed map and camera trajectory gradually by observing more images of the scene. The modality of the input stream depends on the imaging equipment and varies from monocular and stereo to RGB-Depth sensors. In both of the mentioned categories, 3D reconstruction needs sufficient geometric and photometric correspondences in multiple views to retrieve the vanished information, while localizing the camera/robot pose in the reconstructed map. The density of these correspondences leads to different map representations from sparse to semi-dense and dense maps. Dense map reconstruction captures larger portions of the scene however it depends heavily on the quality of per-pixel matching and even with the recent success of deep learning methods, because of limited texture and poor photometric conditions or distortions in captured images, they are restrained to room-scale applications. Furthermore, these methods have high computational and memory cost that limits them to almost non real-time scenarios. On the contrary, sparse representations are computationally more tractable, more robust, and scalable to city-scale applications and widely used in autonomous robots and augmented reality applications. In the particular interest of this thesis we are interested in the sparse real-time SLAM methods.

Semantic scene understanding contains all the methods that extract high-level information present in the captured image of the scene. This high-level information can be any semantic cue regarding the identity of objects (for instance indoor furnitures, outdoor facilities, persons, etc), temporal action recognition, physical or causal relationships, or even understanding 3D structure of the scene from a single image. The wide-range of these applications can be categorized in three main streams: object detections (detecting objects in 2D or 3D with/without pose estimation), image classification, and semantic segmentation. All of these streams involve estimating or learning

a mapping from image features to some desired higher-level space, such as object semantics, object relations (such as spatial relationships encoded in scene graphs), etc. In traditional methods, features required for these applications are handcrafted in a task-specific manner, however recently deep neural networks and particularly Convolutional Neural Networks (CNNs) enable learning general features applicable for these visual tasks and surpass almost all traditional methods. In this work, we are interested in detecting objects in the scene and identifying their semantics, along with extracting the dominant planar structure of the scene and their relationships that is well-known as Manhattan world assumption for indoor environments. In addition to the semantics and planar structures, we are also keen to extract the high-fidelity reconstruction of objects from single images and use them as prior knowledge, analogous to humans that are able to hallucinate the occluded part of the objects seen only from a single view.

Combining scene understanding and geometric reconstruction (in our interest, SLAM) has widely been at the center of attention of both computer vision and robotics researchers, but has never been this close to realization, thanks to deep learning. The mixture of semantic and geometric understanding together in one framework has vital importance in any autonomous behavior of artificial agents. For instance, a driver-less car is required to know "what" is in front of the car and "where" is the location and pose of that to reason about further control actions. It is necessary for an intelligent humanoid robot to recognize the identity of objects and poses of them if it is expected to manipulate and deliver them to a human counterpart.

Two different approaches are conceivable for combining geometric reconstruction as visual SLAM and semantic understanding of the scene. The first approach, which is more traditional and we call it *semantic mapping*, utilizes SLAM as a backbone of the map building and localization and then performs semantic scene understanding as post-processing to reason about high-level content of the map. In other words, after establishing a 3D model of a scene and localizing with respect to that map with a SLAM pipeline, a *mapping* is inferred from the 3D map to a desired semantic space. An standard example of this approach is parsing the scene, detecting or segmenting 3D objects after reconstructing the map. The second approach, which we call *semantic SLAM*, attempts to jointly infer and optimize/refine geometric reconstructions and semantic awareness of the scene simultaneously. In the context of SLAM, this approach is translated to detecting high-level semantically meaningful landmarks in the scene, not only points, and represent them appropriately in the SLAM framework, while refining

them incrementally and when possible imposing or inferring high-level relationships among these landmarks. In this approach, both geometric reconstruction and scene understanding have mutual influence on each other simultaneously in a multitude of ways from data-association to tracking and bundle-adjustment (optimization of all landmarks and camera poses together after gradual inclusion of them in the map). One of the major impediments to the second approach is the ability to *detect* and *match* high-level entities and landmarks in the input image stream. Deep learning achievements in this area eliminates these obstacles in favor of the second approach and makes it more successful than the first one both in geometric reconstruction and semantic inference. For the scope of this thesis, we focus on the second approach to jointly reason about SLAM and semantically meaningful entities.

In this thesis, we propose novel representations for high-level structural and semantic entities to seamlessly integrate them into the sparse SLAM framework, in order to enrich the sparse point-based map with structural landmarks like planes and semantic landmarks like objects, while improving the accuracy of the SLAM in terms of the camera pose trajectory without incurring huge computational and memory cost, keeping the performance near real-time. We also propose new geometric and semantic constraints between the landmarks present in the scene, such as the Manhattan world assumption and supporting affordances, and refine the landmarks and camera poses in a unified bundle-adjustment framework. To move from RGB-D input modality to purely monocular SLAM system, we take advantage of the achievements of CNNs to detect, estimate, and track these landmarks in the monocular case. Hence our main idea is incorporating geometrically and semantically meaningful entities and constraints along with on demand high-fidelity reconstructions in different input modalities, from RGB-D to purely monocular, in a sparse graph-based SLAM system. The following section provides a brief background of our problem in the areas of sparse SLAM, landmark representations, and object-based SLAM. We also review the literature of these concepts in more detail in Chapter 2. At various points of the next section we also discuss our contributions and the way that they fit in the storyline of this thesis.

1.2 Background and Overview

Retrieving the lost 3D information of the scene from an array of numbers which a projective camera "sees" as a 2D image of the scene, is a daunting task for computer vision researchers who commonly deal with *inverse problems*, such as this task known

generally as *geometric reconstruction*. Understanding the geometry of the scene has a wide range of applications in robotics and computer vision, such as mobile robot motion planning and navigation, domestic or industrial robotic manipulation, autonomous vehicles, Augmented Reality (AR) applications. To reconstruct the 3D point corresponding to an observed 2D key-point on the image plane, multiple views of that point must be captured (over time by moving camera or simultaneously in a stereo or multi-camera rig) to enable *triangulation* for estimating the location of the point in 3D space. It is essential to match multiple observations of the same landmark in acquired images (known as the *data association* problem) before the process of geometric reconstruction.

In some applications such as localization, where the aim is to determine the camera's relative pose or camera's trajectory (known as Visual Odometry), maintaining and recovering the set of rigid landmarks in the map is unnecessary and relative pose of the subsequent camera frames can be estimated by marginalizing out the landmarks ([26], [15]). However in geometric reconstruction we are interested in estimating both the camera poses and landmarks in the map, classified in two problem setups: visual Structure from Motion (SfM) and Simultaneous Localization and Mapping (SLAM). The goal of SfM problem is to reconstruct the 3D map and camera poses from a set of images where there are no particular temporal or ordinal relations between the images. These methods are performed in an offline manner as a batch optimization problem to minimize the discrepancy between the estimation of landmarks and camera poses and their visual observations. In contrast, SLAM methods process images incrementally (there is an order in the input stream) in real-time that makes them suitable for robotic vision or real-time applications. In particular, SLAM is vital for autonomous mobile robots in order to acquire the model of the environment while localizing itself within the estimated map, by gradually observing the scene. Likewise any other estimation problem, SfM or SLAM problems presume some reasonable assumptions to restrict the space of feasible solutions. For instance in typical reconstruction problems, most of the 3D scene is considered rigid and brightness-invariant with respect to the perspective viewpoint. Depending on the extent of the problem, additional assumptions are also made such as a dominant structure in the scene or piecewise-planar assumptions mainly in indoor environments.

In this thesis, we are focused on the "visual SLAM" category of geometric reconstruction problem. SLAM is a well-studied problem in robotics and computer vision ([7],[1], [33]) that is traditionally formulated as a *maximum a posteriori* (MAP) problem

represented as a Bayes Net. In particular we define $\mathbf{T} = \{\mathbf{T}_i^w\}_{i=1}^N$ as the camera's trajectory over time, with $\mathbf{T}_i^w$ as a pose of the camera parametrized in the set of rigid Euclidean transformations $\mathbf{SE}(3)$, $\mathbf{L} = \{\mathbf{l}_j\}_{j=1}^M$ denotes the set of landmarks which are parametrized in their appropriate representation space, and $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$ maintains the set of observations of the detected landmarks, and $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^{N-1}$ is the set of odometry measurements between camera poses. Hence the solution of the SLAM problem is the optimal MAP estimate of the following

$$\mathbf{T}^*, \mathbf{L}^* = \arg\max_{\mathbf{T},\mathbf{L}} p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}, \mathbf{U}) \tag{1.1}$$

where $p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}, \mathbf{U})$ is the joint probability of all latent estimate variables given all the associated observations and measurements.

The modern formulation has been applied to SLAM problem recently ([14], [5]) after the introduction of *factor graphs* ([16]) as a modern probabilistic tool for factorization and inference over arbitrary distribution functions.

Factor graph $\mathcal{G}(\mathcal{V}, \mathcal{F}; \mathcal{E})$ is a bipartite graph that explicitly determines how a global function of multiple variables factorizes into a product of local functions ([16]). The set of vertices representing latent variables participating in the estimation problem is denoted by $\mathcal{V}$, $\mathcal{F}$ is the set of factors representing the prior knowledge regarding variable nodes or constraints between them, and $\mathcal{E}$ denotes the edge connections between variable and factor nodes. An example of a factor graph denoting a classic SLAM problem is depicted in Fig 1.1 where joint probability distribution of the MAP estimation problem is factorized as a product over observation factors.

Using the notion of factor graphs, the classical SLAM estimation problem shown in Equation 1.1 can be rewritten ([5]) without odometry measurements as follows:

$$\begin{aligned} \mathbf{T}^*, \mathbf{L}^* &= \arg\max_{\mathbf{T},\mathbf{L}} p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}) \\ &= \arg\min_{\mathbf{T},\mathbf{L}} \sum_{k=1}^{K} \| h_k(\mathbf{T}_{i_k}^w, \mathbf{l}_{j_k}) \ominus \mathbf{z}_k \|_{\mathbf{\Sigma}_k}^2 \end{aligned} \tag{1.2}$$

where h_k is the factor of observing landmark $\mathbf{l}_j$ from the camera pose of $\mathbf{T}_i^w$ as $\mathbf{z}_k$ (*sensor model*), and $\|\mathbf{x}\|_{\mathbf{\Sigma}}^2$ is the squared Mahalanobis norm of vector $\mathbf{x}$ with covariance matrix $\mathbf{\Sigma}$, and $\ominus$ is the desired difference operator defined in the measurement space.

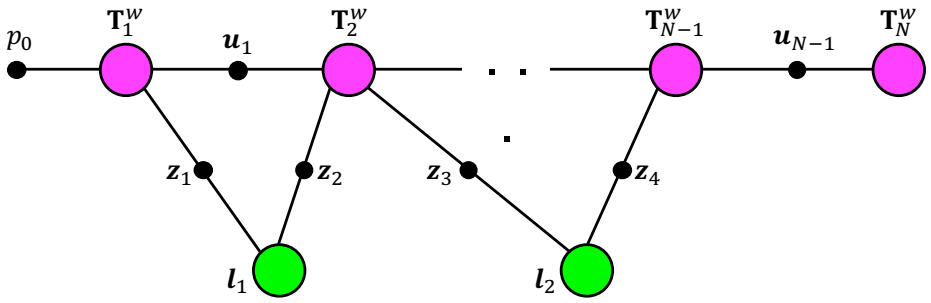


Fig. 1.1 A factor graph formulation of a typical visual-SLAM problem that contains Bundle Adjustment (BA). The camera poses are represented as $\mathbf{T}_i^w$ and landmarks are denoted as $\mathbf{l}_j$. The observations of the landmarks from different camera poses are represented as $\mathbf{z}_k$ and $\mathbf{u}_i$ is the odometry between two camera poses, if there is any. The prior knowledge about the initial pose of the camera is denoted as a prior factor p_0. The joint probability distribution of the corresponding MAP problem can be written as the product of the depicted factors.

Utilizing factor graphs provides an elegant probabilistic tool to intuitively describe and formalize different SLAM problems with novel landmarks and observations/constraints.

Modern SLAM systems consists of two components: "back-end" and "front-end" ([3]). So far we described the back-end component whose objective is optimizing the factor graph problem to yield the optimum estimate for variable nodes, consisting of camera poses and landmark states. This back-end optimization refines the initialize estimate of the camera poses and landmark states in two situations: **(1)** after including a new *key-frame* (a live frame which has less considerable overlap with preceding frames) to the factor graph by considering a "local neighborhood" of that key-frame in the graph, that we call *local bundle adjustment*, and **(2)** after detecting a *loop closure* by considering the whole graph, that we call *global bundle adjustment*.

The back-end optimization happens at an abstract level and typically is unaware of the sensor modalities and equipments of gathering input information for the system. Although the back-end is robust to false association to a certain degree, the task of data association for matching observations and landmarks, initializing the estimate of variables, and in general extracting the factor graph from the raw sensory information takes place in the front-end. Hence the front-end is sensor modality-specific and in general task-specific. For instance the front-end might use different interest point feature descriptors according to the application varying from SIFT([18]), ORB([30]), or even deep-leaned features.

The appropriate representation for camera pose and different kinds of landmarks has a profound effect on the performance of the SLAM problem and one of outcomes of this thesis is the emphasis on the fact that "representation matters" for the SLAM systems. While representation of the camera pose depends on the degrees of freedom (DoF) of motion of the camera (in 3D environment it is typically considered as rigid Euclidean transformations $\mathbf{SE}(3)$), many possible representation for the map have been introduced. The most widely used representation for the map is a set of sparse 3D points that permits SLAM systems, like ORB-SLAM ([22]), to benefit from the sparsity in the structure of the factor graph problem and efficiently solve the optimization problem with well established and effective sparse solvers ([4]). While sparse point-based SLAM systems are successful in city-scale environments, they are not useful for high-level robotic tasks such as grasping, motion and path planning, and virtual applications.

There has been efforts to densify the map by more complex representations such as considering significant gradient regions in the image for *semi-dense* mapping or considering all pixels for representing the scene by a set of surfaces for *dense* mapping. However these methods are still limited to room-scale or building-scale environments while very expensive in terms of computation and memory ([9, 8, 25, 27, 24]). Even when provided sufficient computational resources, these methods generate maps that remain equivalent to a cluster of points and carry no additional high-level semantic information.

In this thesis, as expressed in Section 1.1, we aim to develop a SLAM system that has the advantages of sparse SLAM systems in city-scale and near real-time performance while integrating high-level semantically meaningful landmarks to not only densify the map with these complex entities but also to create a framework for mutual interaction among geometric reconstruction and semantic scene understanding that eventually leads to more accurate camera trajectory and maps. Joint geometric and semantic understanding of the scene by encapsulating semantic information and relationships in structural primitives and objects, is the core of this thesis that yields a real-time SLAM system working in different (RGB-D and monocular) modalities. Our main tool for formulation of the back-end of these different SLAM systems is the mentioned factor graph mechanism. The main challenge in introducing these high-level landmarks is proposing novel mathematical representations that can seamlessly permit them to be integrated in the least-squares formulation of the sparse factor graph. To do so we propose novel representation for plane structural entities and objects to lie on

specific *manifolds* with proposed incremental update rule that let the back-end of the system be optimized in real-time (Chapters 3 and 4). As a consequence of incorporating these landmarks, new semantic and geometric relationships (such as Manhattan world assumption and supporting affordance) are realized as new observation or constraint factors, enriching the semantic understanding of the scene while boosting the accuracy of our SLAM system (Chapters 3, 4, 5). To extract the formulated factor graph from different sensor modalities in the front-end of the system, appropriate plane detections for different modalities are proposed (Chapters 3 and 5) and state-of-the-art deep-learned object detectors are incorporated along with the proposed plane and object matching schemes (Chapters 3 and 5). High-fidelity reconstructions of the objects are combined with coarse sparse representation of the map in Chapter 5 to enrich the real-time monocular SLAM even more and impose additional geometric/semantic constraints on the system.

Towards the aim of this thesis, in Chapter 3 we integrate planar structural primitives in the point-based SLAM systems. A map consisting of sparse points carries no structural or semantic information. Considering other geometric entities like edges or line segments has been studied ([28, 17]) as additional geometric constraints for points, however they carry insufficient semantics while they have complicated high-dimensional representation (6 DoF) and problematic matching/tracking algorithms. The most convenient structural entity to incorporate in our SLAM that is widely seen in man-made environments is planar surface that has 3 DoF. Planes as localization and mapping landmarks have been explored recently in ([32, 13, 39]), however these SLAM systems are only special-purpose non-real-time systems consisting of only plane landmarks. Detecting and including planes in addition to points, provides not only additional regularization (kind of *prior knowledge*) for points that lie on the planes, but also captures large feature-deprived portions of the scene with only few parameters which makes them "memory efficient" parametric landmarks. As representation plays a huge role in the quality of the SLAM solution, in Chapter 3 we propose an effective representation for planes which is amenable to factor graph formulation of sparse point-plane SLAM. We also extend the ability to encode semantic relationships among structural entities by considering Manhattan world assumption (the planar surfaces are mainly mutually parallel or perpendicular to each other) for indoor structured scenes. The appropriate constraint factors for these assumptions are proposed in Chapter 3. The memory efficient reconstructed map in our proposed sparse point-plane SLAM

system is illustrated for a sequence of RGB-D images in Fig 1.2 against the point-only semantic-deprived state-of-the-art ORB-SLAM2 ([21]) reconstruction.

The most vivid representative of a semantically meaningful entity that can be used as a landmark in indoor and outdoor scenes is an *object*. Towards an object-oriented SLAM system, previous efforts have been primarily special-purpose rather than a general-purpose large-scale system. Some works like SLAM++ ([31]) rely on predefined object models that needs to be recognized and matched precisely which is a fundamental limitation for extending to general unseen objects (with respect to the predefined set of objects). Other work ([2]) admits more general objects however in a very slow offline SfM context limited to room-scale. Also other works such as ([19], [36]) attempt to incorporate semantics in the mapping without informing the localization. So in this stream of works there is no joint geometric-semantic understanding, so we call them "semantic mapping" methods instead of "semantic-aware SLAM" systems.

To incorporate semantic-carrying objects in the SLAM framework, there are spectrum of representations from high-fidelity models (such as polygon mesh, TSDF, voxel grids, point clouds), which we term "non-parametric models", to coarse representations (such as cuboid, quadric, super-quadric), which we call "parametric models". Three main challenges for optimizing or refining non-parametric objects restrict them from wide application in general purpose real-time SLAM systems: **(1)** to refine the object model it is essential to represent different category of objects in a high-dimensional manifold that even with recent advances in CNN based object reconstructions, it is a tedious and daunting task to extract and explore this object *latent space*. Works such as [31] avoid this step by assuming predefined object models; **(2)** data-association and matching/tracking is extremely expensive for arbitrary high-dimensional object models, but essential for the success of general geometric reconstitutions; **(3)** maintaining and exploring non-parametric models are very expensive for memory and computational resources that yields mostly room-scale and slow SLAM systems ([20], [2]).

In contrast, parametric models capture the desired and important specifications of objects, are memory and computation friendly, and permit the SLAM system to operate in real-rime and at larger scales. However prudent considerations are necessary for the parametric object representation to not only capture semantic label, rough 3D extent and shape, and 6D pose (location and orientation), but also to make them amenable to the least-squares framework of sparse SLAM systems that we are interested in. Different coarse object representations have been explored such as *cuboids* ([38]), bounded *quadrics* (quadratic surfaces, generalization of conic sections in plane) ([11]),

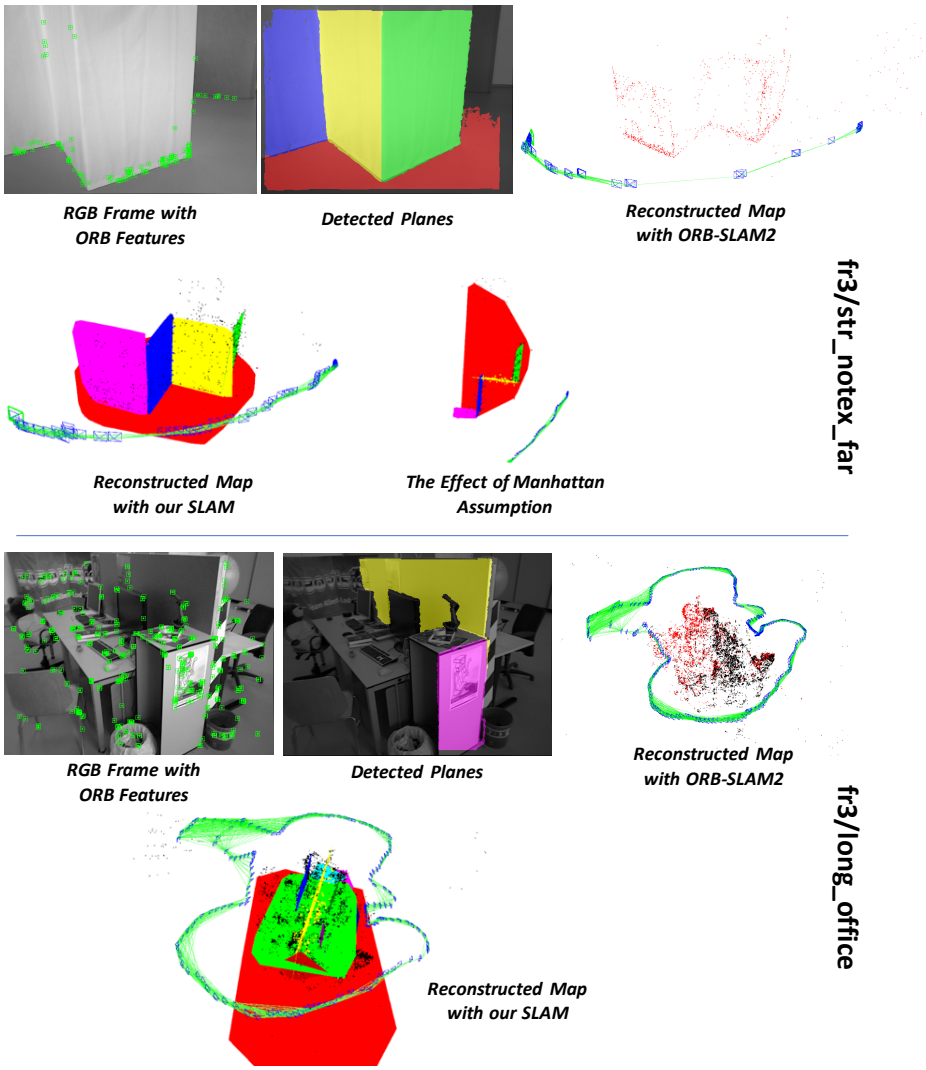


Fig. 1.2 **Sparse Point-Plane SLAM**. The reconstructed map and estimated camera trajectories by our proposed sparse SLAM system in Chapter 3 have been illustrated for two challenging sequences of TUM benchmark [34]. The proposed plane landmark representation along with the point-plane regularizer and Manhattan world assumption as new factors permit a real-time integration of planar structures in the point-based SLAM system in order to capture feature deprived regions such as the ones depicted for `fr3/str_notex_far`. The large loop closure present in `fr3/long_office` sequence boosts the localization and mapping accuracy even more for this sequence, due to the participation of all landmarks in the global bundle adjustment. Furthermore, our system enriches the reconstructed map with more semantically meaningful structural landmarks with negligible computational and memory overload, while enhancing the localization accuracy. For instance, with considerably less than 1 KB memory overload we can represent the reconstructed maps more accurately compared with the vanilla ORB-SLAM2 [21] which yields less interpretable sparse maps.

super-quadrics ([6]). A compromise between simplicity and mathematical convenience makes bounded quadrics (ellipsoids) the desired object landmark representation for our SLAM, since among the mentioned ones it is the only bounded representation whose perspective projection on the image plane is again a quadric but in a lower dimension (conic section). This simplicity is exploited and explored in Chapter 4. However using quadrics has some fundamental limitations such as initialization, matching scheme, etc, that restrains the previous methods ([29, 11, 35]) to SfM context or quadric-only map representations using ground-truth data associations in non-real-time live scenarios.

In Chapter 4, we address these problems by proposing novel representations and incremental updates for *dual quadrics* to enable the combining of the CNN-based object detectors with the real-time general purpose point-plane sparse SLAM system (Chapter 3) for the first time at this scale, to the best of authors' knowledge. In our proposed SLAM in Chapter 4, not only we do benefit from the semantic labels of object detections in data association, but we also use the geometric clues about the extent and pose of the observed objects as bounding boxes. This novel point-plane-object SLAM exploits the semantic and geometric information of CNNs to enrich the sparse point-only map efficiently with a set of covering planes and wide variety of coarse objects, by marrying CNNs and traditional graph-based SLAM approaches operating in near real-time.

In addition to the seamless integration of quadrics in our SLAM, we benefit from the coarse representation to associate the object detection with the landmarks and also initializing the quadric estimates effectively by the proposed decoupled dual representation (Chapter 4) to avoid ill-posed initialization problem that hinders previous SfM and SLAM methods ([29, 11, 35]). The other consequence of the proposed dual quadrics is the introduction of the supporting affordance in the sparse SLAM systems for the first time in this thesis between object landmarks and their supporting surfaces to encapsulate the supporting/tangency relationship which in addition to geometric meaning carries semantic information (Chapter 4).

The point-plane-object SLAM system (Chapter 4) while enriching the map memory-efficiently in real-time, also boosts the accuracy of camera trajectory for the experienced benchmarks. A sample of these results for RGB-D benchmarks is shown in Fig 1.3.

In Chapters 3 and 4, we demonstrate the efficacy of including visual semantic clues in geometric reconstruction (SLAM) and the mutual impact of them on enhancing the localization and mapping's accuracy while producing richer maps. Semantic clues consisting of planar structures and coarse quadric objects are extracted from raw input

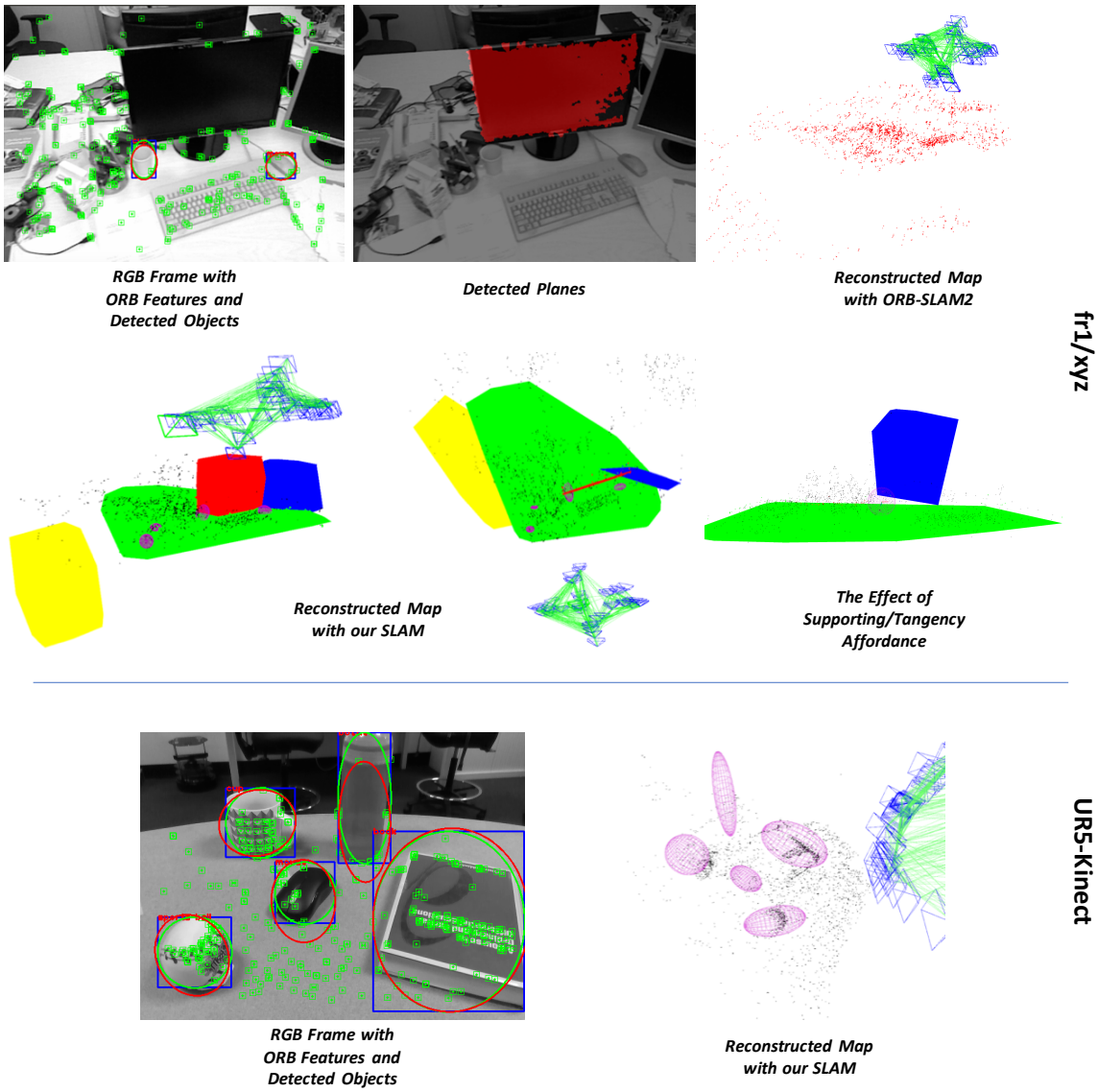


Fig. 1.3 **Structure Aware SLAM using Quadrics and Planes**. The estimated camera trajectory alonfg with the reconstructed map of our proposed structure and object aware SLAM system proposed in Chapter 4 are rendered for `fr1/xyz` sequence from publicly available TUM benchmark [34] and an RGB-D sequence captured with the UR5 robotic arm equipped with Kinect sensor in our Robotic Vision Lab. In addition to the advantages inherited from the sparse point-plane SLAM, in this system we introduce novel representations for coarse objects as dual quadrics, in order to involve high-level semantic landmarks directly in the bundle adjustment problem of our semantic SLAM. Besides boosting the camera trajectories by introducing these landmarks along with supporting affordance relationships, the reconstructed map is semantically rich as the product of the marriage between traditional geometric SLAM methods and deep-learned object detectors.

imagery using a depth-dependent plane detector and precomputed CNN-based object detections, respectively, in the front-end of our SLAM and are fed to the back-end via proposed manifold representations.

Towards the final goal of this thesis i.e., to encapsulate structural and semantic object entities to the real-time monocular sparse SLAM, we facilitate the integration of high-fidelity reconstructions on top of the coarse object landmarks and detect planar structures from RGB modality in Chapter 5. We propose a CNN-based monocular plane detector to segment planar regions and estimate their parameters from the redundancy existing in different output channels of a joint CNN for surface normals, depth and semantic segmentations. This detector is also compared in Chapter 5 against the state-of-the-art CNN-based plane detector that is dedicated to extract planes directly from raw images. To hallucinate the full 3D point-cloud object model from its single-view 2D detection, we train a Point-Set-Generation CNN ([10]) that leads to seamless integration of point-cloud model on top of the coarse dual quadric representation in the sparse map. In our SLAM, since we have both coarse and fine reconstructions, on demand, we use the coarse quadrics to introduce an elegant data-association scheme to track quadric objects based on robustly matched key-points (Chapter 5) that is a major bottleneck for object-oriented SfM or SLAM systems such as ([29, 11, 35]). This matching scheme performs effectively in cluttered scenes and overlapped detections.

In addition to the mentioned new components in the front-end of our monocular SLAM, we propose new observation factors in the back-end (Chapter 5) to overcome some limitations of factors introduced in Chapters 3 and 4, such as aligned observations or fixed global reference frames for planes and objects (refer to Chapter 5) for more details). In Chapter 5, we have established three links between traditional purely geometric SLAM methods and modern CNNs to enhance the performance of SLAM systems: detecting and initializing planar structures, semantic labels and rough extent and pose of objects, and also hallucinating object fine models. To complement the rough reconstruction of objects as quadric that comes from SLAM pipeline, with the fine model from CNN, we propose a novel prior factor on quadric (Chapter 5) induced from the point-cloud model to jointly refine the shape of the quadric further, feasible thanks to the decoupled quadric representation. The intuition of this prior factor comes from the idea of *intersection over union* loss.

Finally, we demonstrate in Chapter 5 (see Figure 1.4) that incorporating high-level landmarks and semantic information (planar structures, coarse and fine objects) jointly in the sparse SLAM system, yields an efficient real-time SLAM that enriches the sparse

map with semantically meaningful landmarks and relationships and affordances among them with negligible memory and computation overload, while boosting the accuracy of camera trajectory and implicitly the mapping.

At the end, note that unlike most of the SLAM systems that have been designed recently as special-purpose localization and mapping tools, our endeavor in this thesis is to present a *general-purpose* real-time monocular structure and object aware SLAM system. We exhaustively evaluate our system, unlike most of the other special-purpose ones [13, 32, 17, 28], on publicly available benchmarks, qualitatively and quantitatively through ablation studies by considering various landmarks and constrains in a multitude of scenarios. Our hope is that the way we have initiated in this thesis to contemplate about *Semantic SLAM* problem reaches to the introduction of more complicated semantic constraints or priors and even translation of more global relational *scene graphs* to the factor graphs of SLAM.

1.3 Contributions

The general contribution of this thesis is to propose a new semantic SLAM framework to directly incorporate high level semantically meaningful entities (structures and/or objects) as independent landmarks to the underlying graph-based bundle-adjustment of the sparse SLAM system. By proposing a SLAM framework, we mean not only proposing the new mathematical representations that are amenable for least-squares framework of graph-based SLAM and subsequently new observation factors and constraints, but also to benefit from the structures/objects in the front-end to propose new components for detecting and associating observations and hypothesizing the constraints in order to yield a more robust topology for the underlying factor graph.

For convenience, we briefly point out chapter-wise the key contributions. Note that all these contributions have been explained in further detail in Section 1.2 as they fit in the overall picture of this thesis.

- We propose a representation for introducing structural planar entities that is amenable to the factor graph formulation of the SLAM problem. By using this representation, we capture large portions of the scene, mainly feature-deprived regions, using a few parameters and introduce them as independent landmarks, instead of clusters of points, that participate directly in the bundle-adjustment of the point-based SLAM problem. Unlike works such as ([37]) that use planes

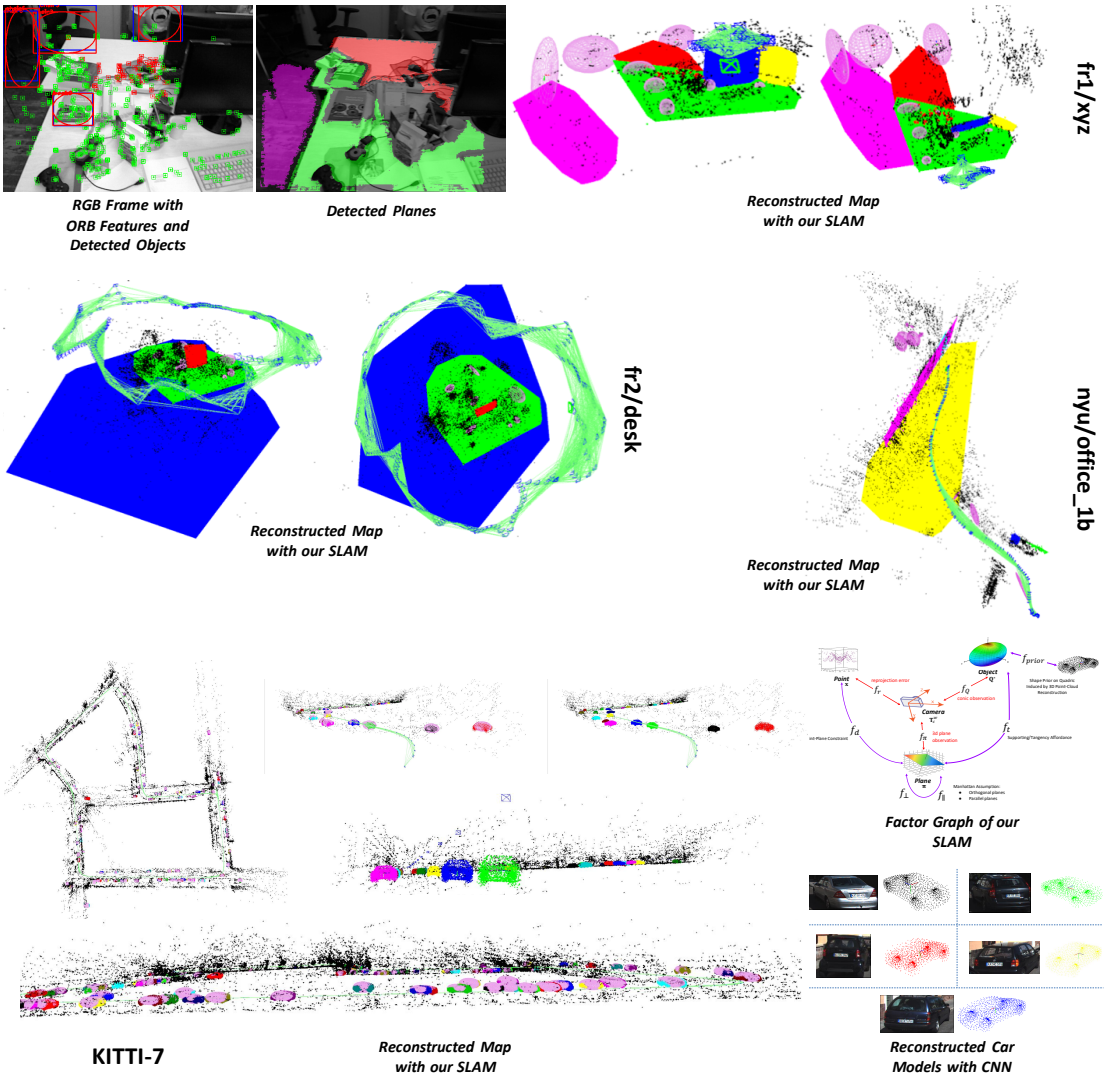


Fig. 1.4 **Real-Time Monocular Object-Model Aware Sparse SLAM**. We propose the structure and object model aware monocular SLAM system as the final aim of our thesis in Chapter 5. More accurate camera trajectories with semantically richer maps are the result of introducing heterogeneous high-level landmarks in our sparse SLAM, as demonstrated in the factor graph of our system. In addition to the novel proposed multi-edge factors for plane and object observations, new integrated CNN-based components in the front-end make the monocular plane detection and object point-cloud model hallucination possible in our SLAM. Wherever the fine reconstruction of the object is available, we impose a new shape prior factor which is induced from the point-cloud model of the object, on top of the coarse dual quadric in order to refine it even more. The resulted map with planar structures, coarse dual quadric objects, and high-fidelity car reconstructions are illustrated for some sample sequences from publicly available TUM [34], NYU-v2 [23], and KITTI [12].

for depth fusion of points, or special-purpose SLAM systems with plane-only landmarks ([13]), thanks to the factor graph compliant representation, we achieve a general-purpose sparse point-plane SLAM system that operates in real-time. (Chapter 3, Sections 3.3.3, 3.4.2)

- One of the prevalent approaches in combining deep-learned priors to traditional geometric SLAM systems is to introduce predicted depth or surface normals by CNNs to dense SLAM frameworks. However imposing these priors typically has local effect and enforcing more global priors particularly in sparse SLAM systems is a changeling task. By introducing the point-plane constraint in our SLAM we effectively apply additional global planar regularizer on top of the point landmarks in the map and refine the reconstructed map even more. (Chapter 3, Section 3.4.3)

- We are not content with only using point-plane constraints as prior or regularizer in our sparse SLAM and based on evidence we propose Manhattan world assumption to leverage the structure present in the man-made environments. We introduce Manhattan assumption in our sparse SLAM system as two new factors for perpendicularity and parallelism between planar structures of the scene. (Chapter 3, Section 3.4.4)

- The proposed factor graph of our system participating in the bundle-adjustment in back-end, is required to be extracted first from the raw imagery. We introduce simple yet effective front-end to detect and track planes in real-time in the RGB-D modality and hypothesize about constraints and priors, (Chapter 3, Section 3.5).

- In order to incorporate semantic objects directly in the incremental bundle-adjustment (BA) as independent landmarks while avoiding expensive high-dimensional latent spaces for objects, we propose a novel mathematically elegant dual quadric coarse representation for objects along with the proposed approximate incremental update for the BA problem. This representation permits the integration of dual quadrics in the general-purpose sparse point-plane SLAM system as coarse objects for the first time that in addition to providing elegant data association, elevates the accuracy of localization and mapping. (Chapter 4, Section 4.3.1)

- We propose dual quadric observation factor based on the projected dual conic section inside detected bounding box in order to benefit from the geometric clues

(about the rough pose and extent of the objects) provided by the deep-leaned 2D object detections in addition to their semantic label information that is useful for more robust data association in the front-end. This part is the first tie between traditional geometric approaches of SLAM and modern deep-learned CNNs in our SLAM system. (Chapter 4, Sections 4.4.1, 4.5.1)

- Integrating quadrics in the point-plane graph-based SLAM facilitates the introduction of *supporting affordances* - another prevailing structural information in man-made environments - to our real-time SLAM framework for the first time thanks to the proposed representation in dual space. The introduced constraint factor enforces the tangency, based on evidence, between dual quadrics and their supporting planar surfaces in the scene. (Chapter 4, Section 4.4.5).

- We propose the decoupled dual quadric initialization approach by exploiting robustly matched key-points in live frame followed by another intermediate observation factor optimization. This initialization method overcomes the ill-posed conditions that makes it a major bottleneck in previously quadric-based SfM or SLAM systems ([29, 11, 35]). (Chapter 4, Section 4.5.1).

- Towards one of aims of this thesis for presenting a purely monocular SLAM, we propose an RGB-only plane detection pipeline to replace the only depth-dependent component in the front-end. We exploit the redundancy that is present in three different output predictions (depth, surface normal, and semantic segmentation) of a real-time joint CNN in order to detect and segment planes and regress its parameters. This part is the second tie between traditional geometric approaches of SLAM and modern deep-learned CNNs in our SLAM system.(Chapter 5, Section 5.4).

- Multi-edge new observation factors are introduced for planes and quadric objects to introduce relative reference key-frame concept for the observed landmarks in order to increase the global consistency of the map encoded in the topology of the multi-edge factor graph participating in the local and global bundle-adjustment. (Chapter 5, Sections 5.4.3, 5.5.3).

- To overcome the limitations of the previously imposed observation factor for quadrics (Chapter 4) in enforcing axis-aligned observations, we propose a novel observation factor for objects inspired by the *intersection-over-union (IoU) loss*

and utilizing the confidence score of CNN-based object detections. (Chapter 5, Section 5.5.3)

- One of the advantages of using coarse representations for objects is the opportunity to introduce elegant data association methods that is exploited to propose a point-based object tracking scheme in our monocular SLAM. The proposed matching approach in the front-end performs effectively in multi-object overlapped cluttered indoor/outdoor scenes. (Chapter 5, Section 5.5.1)

- Eventually we register Point-Set based high-fidelity object reconstruction on top of its coarse quadric representation in our monocular sparse SLAM system to perform in near real-time while densifying the map for desired objects with their fine point-cloud reconstructions. We also propose a new unary *shape prior factor* for incorporating prior knowledge about the extent of the quadric to the framework. In this factor we impose the prior by the idea of 3D IoU loss induced from the reconstructed point-cloud model of objects. (Chapter 5, Section 5.5.2)

- We evaluate our SLAM system extensively on almost all the publicly available SLAM benchmarks, such as TUM([34]), NYU-v2([23]), and KITTI([12]), in both RGB-D and monocular input modalities. In addition to the qualitative results, we perform quantitative ablation studies by considering different landmarks and observations/constraints in a multitude of scenarios for comparing the camera trajectory errors against the baselines as an indicator for the accuracy of the localization and implicitly for the mapping. Furthermore, we explored the performance of our framework by considering different components in the front-end, for instance the effect of using various plane detection methods in the accuracy of the SLAM is examined. Unfortunately most of the proposed SLAM systems ([13, 32, 17, 28]) lack this kind of exhaustive evaluations and they are restricted to special-purpose scenarios and limited benchmarks. (Chapters 3, 4, 5; Sections 3.6, 4.6, 5.7)

1.4 Thesis Outline

An outline for the remainder of this thesis is as follows:

- **Chapter 2:** In this chapter we discuss the traditional geometric SLAM methods and the recent advances towards introducing semantics and objects in the SLAM

framework. We discuss various approaches towards integrating semantic objects and structures in the SLAM frameworks.

- **Chapter 3:** In this chapter we introduce our sparse point-plane SLAM system with new back-end and front-end components. In this system, in addition to integrating planes as independent landmarks, we also introduce constraints between them (plane-plane) and other heterogeneous landmarks in the scene (point-plane).

- **Chapter 4:** In this chapter we incorporate coarse representations for generic objects as dual quadrics to the previously introduced graph-based sparse point-plane SLAM in addition to imposing supporting affordance relationships. This chapter introduces the first ties between traditional purely geometric based SLAM approaches and modern deep-learned CNNs in our system.

- **Chapter 5:** In this chapter we introduce novel factor representations in the back-end and new CNN-based front-end components to represent our real-time monocular structure and object model aware sparse SLAM system.

- **Chapter 6:** In this chapter we provide a summary of the research in this thesis along with a discussion about potential future work directions.

Bibliography

[1] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[2] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2012.

[3] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32 (6):1309–1332, 2016.

[4] T. A. Davis et al. Suitesparse: a suite of sparse matrix software. *URL: http://www. suitesparse. com*, 2015.

[5] F. Dellaert, M. Kaess, et al. Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139, 2017.

[6] P. Drews, P. Núñez, R. Rocha, M. Campos, and J. Dias. Novelty detection and 3d shape retrieval using superquadrics and multi-scale sampling for autonomous mobile robots. In *2010 IEEE International Conference on Robotics and Automation*, pages 3635–3640. IEEE, 2010.

[7] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[8] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[9] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[10] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, volume 2, page 6, 2017.

[11] P. Gay, V. Bansal, C. Rubino, and A. D. Bue. Probabilistic structure from motion with objects (psfmo). In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3094–3103, Oct 2017. doi: 10.1109/ICCV.2017.334.

[12] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

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

[14] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. isam2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3281–3288. IEEE, 2011.

[15] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 29(8): 941–957, 2010.

[16] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

[17] T. Lemaire and S. Lacroix. Monocular-vision based slam using line segments. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2791–2796. IEEE, 2007.

[18] D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[19] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4628–4635. IEEE, 2017.

[20] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018.

[21] 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*, 33(5): 1255–1262, 2017.

[22] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[23] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[24] 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 *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[25] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[26] D. Nistér, O. Naroditsky, and J. Bergen. Visual odometry. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1, pages I–I. Ieee, 2004.

[27] V. A. Prisacariu, O. Kähler, S. Golodetz, M. Sapienza, T. Cavallari, P. H. Torr, and D. W. Murray. Infinitam v3: A framework for large-scale 3d reconstruction with loop closure. *arXiv preprint arXiv:1708.00783*, 2017.

[28] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer. Pl-slam: Real-time monocular visual slam with points and lines. In *Proc. International Conference on Robotics and Automation (ICRA), IEEE*, 2017.

[29] C. Rubino, M. Crocco, and A. D. Bue. 3d object localisation from multi-view image detections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP (99):1–1, 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2701373.

[30] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Computer Vision (ICCV), 2011 IEEE international conference on*, pages 2564–2571. IEEE, 2011.

[31] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: simultaneous localisation and mapping at the level of objects. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 1352–1359, 2013. doi: 10.1109/CVPR.2013.178. URL https://doi.org/10.1109/CVPR.2013.178.

[32] R. F. Salas-Moreno, B. Glocken, P. H. J. Kelly, and A. J. Davison. Dense planar slam. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 157–164, Sept 2014. doi: 10.1109/ISMAR.2014.6948422.

[33] C. Stachniss, J. J. Leonard, and S. Thrun. Simultaneous localization and mapping. In *Springer Handbook of Robotics*, pages 1153–1176. Springer, 2016.

[34] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[35] N. Sünderhauf and M. Milford. Dual quadrics from object detection boundingboxes as landmark representations in slam. *arXiv preprint arXiv:1708.00965*, 2017.

[36] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid. Meaningful maps with object-oriented semantic mapping. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 5079–5085. IEEE, 2017.

[37] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng. Point-plane slam for hand-held 3d sensors. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5182–5189. IEEE, 2013.

[38] S. Yang and S. Scherer. Cubeslam: Monocular 3d object detection and slam without prior models. *arXiv preprint arXiv:1806.00557*, 2018.

[39] S. Yang, Y. Song, M. Kaess, and S. Scherer. Pop-up slam: Semantic monocular plane slam for low-texture environments. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 1222–1229. IEEE, 2016.

Chapter 2

Literature Review

In this chapter after a brief review on different geometric reconstruction methods, in line with the main focus of this thesis, we discuss in more detail about the introduction of high-level entities, semantics, and objects in the SLAM framework. We present a new perspective towards categorization of these methods and clarify their distinction by using modern factor graphs as our language for explaining different geometric reconstruction methods. In conclusion, we justify our research in the context of the proposed perspective in this chapter towards integrating semantics and objects in SLAM.

2.1 Geometric Scene Reconstruction

Solving the problem of geometric 3D scene retrieval from a set of 2D projective observations, has continuously been in the center of attention of computer vision and robotics researchers. Similar to all *inverse* problems, this estimation problem is also tackled with considering certain assumptions, such as brightness-invariance with respect to the perspective viewpoint, rigidity or limited dynamics in the scene, etc.

Geometric reconstruction problem can be formulated in different settings depending on a multitude of elements, such as the modality of the input imagery, density of the map representation, and incremental or batch methods. In this section we discuss some of these different approaches towards scene reconstruction and focus mainly on the SLAM methods in the particular interest of this thesis.

2.1.1 Structure from Motion (SfM)

Structure from Motion (SfM) is a well-studied problem in computer vision comprises the 3D scene retrieval from a set of 2D projective images from multiple views, while estimating the pose of the camera captured those images. The key technique for tackling SfM problem is "adjusting" the "bundles" of light rays passing through the camera centers and feature points such that they converge on 3D points. The method is well-known as *Bundle Adjustment* (BA) [157]. In the BA methods, both 3D structure and camera parameters are optimized in a coupled way.

We can formulate the SfM as a *maximum a posteriori* (MAP) estimation problem represented as a Bayes net which estimates the trajectory of the camera poses $\mathbf{T} = \{\mathbf{T}_i^w\}_{i=1}^N$ and landmarks in the map $\mathbf{L} = \{\mathbf{l}_j\}_{j=1}^M$, simultaneously, given the noisy observations of those landmarks $\mathbf{Z} = \{\mathbf{z}_k\}_{k=1}^K$. The camera pose $\mathbf{T}_i^w$ can be parametrized as a Euclidean rigid transformation $\mathbf{SE}(3)$ and the parametrization of the landmark $\mathbf{l}_j$ depends on the representation of the map, for instance for 3D points it is a vector in $\mathbb{R}^3$. More precisely, the solution of the SfM problem is the optimal MAP estimate of the following

$$\mathbf{T}^*, \mathbf{L}^* = \arg\max_{\mathbf{T},\mathbf{L}} p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}) \tag{2.1}$$

where $p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z})$ is the joint probability of all latent estimate variables given all the associated observations. Finding the optimal solution of the mentioned SfM typically involves solving a batch non-linear least squares problem using gradient-based approaches such as Levenberg-Marquardt algorithm ([157]).

2.1.2 Simultaneous Localization and Mapping (SLAM)

The inherent batch optimization flavor in SfM approach towards geometric scene reconstruction, limits the applicability of this approach for real time and incremental applications, such mobile robot navigation and planning, autonomous vehicles, and augmented reality (AR) applications. Incremental and real-time methods have been designed over the past decades ([29, 4, 55, 23, 24]) to formulate the geometric reconstruction problem in an incremental fashion and solve it in real-time which makes it appealing for robotic and computer vision applications, giving rise to the new category of approaches known as *Simultaneous Localization and Mapping* (SLAM). Moving from expensive bulky laser range finders to ubiquitous, cost-effective, and power efficient

cameras as sensor equipments for observing the environment, make *visual* SLAM an attractive area for SLAM researchers.

Despite the enormous progress in the theory of visual SLAM ([29]), there are yet many application-specific issues that need to be addressed, such as robust landmark and feature representation, robust data association, scalability, dynamic environments, and computational effectiveness.

Traditional SLAM formulations known as filter based approaches consider the landmarks in the map and current camera pose as state variables by marginalizing out the previous camera poses, however in modern *key-frame* based approaches (after PTAM [72]) the whole camera poses (key-frames) trajectory along with the landmarks in the map are considered as state variables of the MAP estimation problem. Modern key-frame based approaches, in addition to resulting fewer interdependencies between camera poses and landmarks that are needed for BA, provide an elegant scheme to reduce computational complexity of the underlying BA, due to the inherent sparsity in the state covariance matrix of these methods ([138]). In key-frame based methods, the SLAM system consists of two parts: *front-end* and *back-end*. Feature detection, camera tracking with respect to key-frames, and map initialization ([105]) occur in the front-end, while back-end performs the bundle adjustment with the initialization of state variables that is fed from the front-end. For instance in a modern state-of-the-art SLAM system such as ORB-SLAM2 [99], BA takes place in a key-frame based fashion in an independent computing thread by the tracking and initialization provided by the front-end. Visual key-frame based SLAM frameworks has been adopted widely in recent modern SLAM implementations, such as [32, 72, 75, 100, 138, 139], which permits them to scale to long operating times, however they are limited by the amount of the partial coverage of a scene by the moving camera.

In spite of a typical common underlying bundle adjustment in the back-end of SLAM, the front-end of SLAM systems varies widely due to various reasons, such as differences in imaging equipment modalities such as monocular/stereo/multi-camera/depth-sensors setups, and tracking and initialization schemes. Furthermore, different map representations, from sparse to semi-dense and dense maps, also make an impact on the components in the front end of SLAM.

The simplest representation of the map is a set of 3D points corresponding to 2D image interest points [100, 99]. This sparse representation is computationally efficient for bundle adjustment in back-end, and allows the SLAM system to map city-scale environments successfully. However, sparse SLAM systems generate maps lacking

high-fidelity reconstructions that might be imperative for high level tasks such as robotic manipulation and grasping. A more computationally intensive but meaningful method is to consider all high gradient image pixels as candidates for 3D points in the map, that leads to *semi-dense* map representation [32]. These methods have been shown to operate successfully at a building-scale environment. A fully *dense* map represents the scene as a set of surfaces [116, 102], that supplies the most information required for high level tasks such as robotic grasping and manipulation, while being the most computationally expensive. Dense SLAM frameworks are still limited to room-scale environments. The map representations for semi-dense or dense methods, such as [33, 32, 103, 116, 102, 163, 178, 164, 12, 21], remain equivalent to a set of points without carrying any additional semantic information.

In our proposed SLAM systems in the next chapters, we adopt the key-frame based approach and integrate our proposed structural (plane) and semantic (object) entities in the bundle adjustment within the back-end, along with the appropriate components in the front-end for detecting and tracking landmarks and providing data-association. Our final objective is to propose a sparse pure monocular SLAM system that is aware of structural and semantic objects in the scene (Chapter 5).

2.1.2.1 SLAM as a Factor Graph

As discussed earlier, SLAM problem can be formulated as a Bayes net. After introduction of *factor graphs* by [81] as a powerful modern probabilistic tool for factorization and inference over probability distributions and functions, it has been applied widely to SLAM problems ([68, 25, 26]). In this section we introduce factor graph as an elegant tool to formalize our SLAM frameworks in Chapters 3, 4, 5 and also utilize it as a "language" in order to intuitively discuss about recent approaches towards introduction of semantics and objects in SLAM framework (Section 2.2) and how our proposed SLAM fits in among other modern SLAM systems.

A factor graph is a bipartite graph that indicates how a global multi-variable function factorizes into a product of local functions, possibly single or multi-variable. The first part of the nodes represents the set of latent variables which need to be estimated and, as context permits, it is called *variables*, and the second part of the nodes which represents the set of constraints and information between or on these variables is called *factors*. If we use factor graph to factorize a joint probability distribution over some random variables, intuitively we encode the inherent conditional independence of some local variables into the joint probability distribution.

The joint probability distribution of all the latent estimate variables of the SLAM problem, as mentioned in Equation 2.1, can be written as following, given all associated observations,

$$p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}, \mathbf{U}) \propto p(\mathbf{T}_0^w) \prod_{k=1}^{K} p(\mathbf{z}_k | \mathbf{T}_{i_k}^w, \mathbf{l}_{j_k}) \prod_{i=1}^{N} p(\mathbf{T}_i^w | \mathbf{T}_{i-1}^w, \mathbf{u}_{i-1}) \tag{2.2}$$

where $\mathbf{U} = \{\mathbf{u}_i\}_{i=1}^{N-1}$ denotes the possible odometry measurements $\mathbf{u}_i$, if there is any, between camera poses in the trajectory, $p(\mathbf{T}_0^w)$ is the prior knowledge on the initial pose of the camera, $p(\mathbf{z}_k | \mathbf{T}_{i_k}^w, \mathbf{l}_{j_k})$ is the effect of landmark observation $\mathbf{z}_k$ given the data-association (i_k, j_k), $p(\mathbf{T}_i^w | \mathbf{T}_{i-1}^w, \mathbf{u}_{i-1})$ is the state update given the motion model. By assuming zero-mean Gaussian noise for observations $\mathbf{Z}$ and odometry $\mathbf{U}$ with covariance $\mathbf{\Sigma}_k$ and $\mathbf{\Sigma}_o$ respectively, the above joint probability distribution can be written as following

$$\begin{aligned} &p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}, \mathbf{U}) \propto \\ &\underbrace{\prod_{k=1}^{K} \exp\Big(-\frac{1}{2} \Big\| h_k(\mathbf{T}_{i_k}^w, \mathbf{l}_{j_k}) \ominus \mathbf{z}_k \Big\|_{\mathbf{\Sigma}_k}^2 \Big)}_{\text{The effect of observations}} \underbrace{\prod_{i=1}^{N} \exp\Big(-\frac{1}{2} \Big\| f_o(\mathbf{T}_{i-1}^w, \mathbf{u}_{i-1}) \ominus \mathbf{T}_i^w \Big\|_{\mathbf{\Sigma}_o}^2 \Big)}_{\text{The effect of odometry}} \end{aligned} \tag{2.3}$$

where h_k is sensor model and f_o is motion model, $\|\mathbf{x}\|_{\mathbf{\Sigma}}^2$ is the squared Mahalanobis norm with the corresponding covariance, $\ominus$ denotes the difference between vectors in the vector space tangent to the corresponding manifold, observation and camera pose spaces, respectively. By factorizing the joint probability distribution, the optimal MAP estimate can be determined by solving the following equivalent least-squares form,

$$\begin{aligned} \mathbf{T}^*, \mathbf{L}^* &= \arg\max_{\mathbf{T},\mathbf{L}} \; p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}, \mathbf{U}) \\ &= \arg\min_{\mathbf{T},\mathbf{L}} \; -\log p(\mathbf{T}, \mathbf{L} \mid \mathbf{Z}, \mathbf{U}) \\ &= \arg\min_{\mathbf{T},\mathbf{L}} \; \underbrace{\sum_{k=1}^{K} \Big\| h_k(\mathbf{T}_{i_k}^w, \mathbf{l}_{j_k}) \ominus \mathbf{z}_k \Big\|_{\mathbf{\Sigma}_k}^2}_{\text{Observation Factors}} + \underbrace{\sum_{i=1}^{N} \Big\| f_o(\mathbf{T}_{i-1}^w, \mathbf{u}_{i-1}) \ominus \mathbf{T}_i^w \Big\|_{\mathbf{\Sigma}_o}^2}_{\text{Odometry Factors}} \end{aligned} \tag{2.4}$$

that can be interpreted graphically as a factor graph, depicted in Fig 1.1, with associated observation and odometry factors connecting variable nodes of the graph.

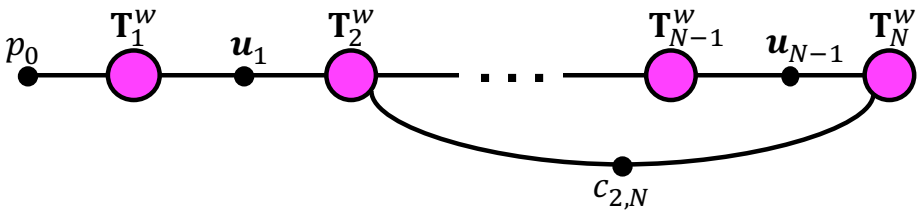


Fig. 2.1 Pose-Graph SLAM as a factor graph. A typical factor-graph formulation of Pose SLAM, where the odometry factors $\mathbf{u}_i$ constrain the relative camera poses with potential loop-closure factors denoted as c_{i_1,i_2} . Variable nodes are camera poses depicted in purple circles, while factor nodes are filled-in black nodes. Unary factor p_0 encodes the prior knowledge about the initial pose of the camera.

Pose SLAM In some tasks, such as localization, it is not required to maintain the set of landmarks and their associated observations into the future. The formulation of these tasks leads to Pose SLAM [76, 77, 94] which marginalizes out all the landmarks in the map by registering two sets of landmarks and identifying the relative pose constraint for the camera with the uncertainty that is propagated to the relative pose estimated. The factor graph of a typical Pose SLAM problem is illustrated in Fig 2.1 in which the subsequent camera nodes are connected with odometry factors that constrain the relative poses of the camera. If the system identifies a previously visited scene (*loop-closure* detection), the loop-closure factors are imposed as relative pose constraints by registering the previously visited landmark observations with the current view of them.

Bundle Adjustment (BA) In classical bundle adjustment [157, 56] the variable nodes of the factor graph can be considered as camera poses and 3D landmark points, and the objective is to minimize the re-projection error factors. BA applications can make use of other sensory information such as wheel odometry in mobile robots, or inertial measurement unit (IMU) to further improve the accuracy of camera trajectory and mapping. The factor graph of a typical bundle adjustment problem interpreted as factor graph is depicted in Fig 2.2 with potential loop-closure factors. As proposed in this thesis, the landmark representations and factors can be extended to some high-level entities and more complicated constraints and factors.

As mentioned earlier about key-frame based approaches, in the context of factor graphs, the back-end of the SLAM system optimizes this factor graph which is exploited, built, and initialized from the raw imagery in the front-end. By adding a new

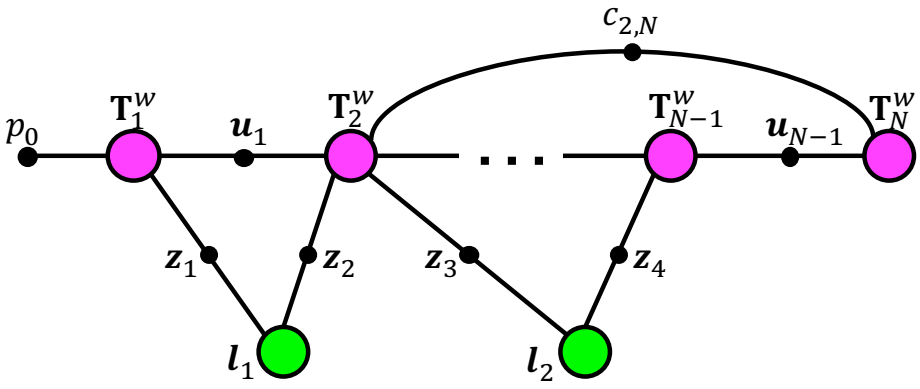


Fig. 2.2 Visual-SLAM Bundle Adjustment (BA) contained in a factor graph. A typical factor graph formulation of bundle adjustment, where the potential odometry factors $\mathbf{u}_i$ constrain the relative camera poses with potential loop-closure factors denoted as c_{i_1,i_2} , landmarks are represented as $\mathbf{l}_j$. The observation of various landmarks from different camera poses are indicated as $\mathbf{z}_k$ factors. Variable nodes are camera poses depicted in purple circles, and landmarks in green circles, while factor nodes are filled-in black nodes. Unary factor p_0 encodes the prior knowledge about the initial pose of the camera.

key-frame, the topology of the graph is updated and bundle-adjustment is performed on the graph. In typical visual SLAM frameworks, there are no odometry measurements and the only factors exist in the graph are observation factors and probably potential constraints between landmarks in the map. However, note that the variable nodes or observation factors in the graph carry no semantic information about the scene or about the configuration of landmarks in the map.

2.2 Semantics and Objects in SLAM

There has been many efforts in the last few decades to design and implement a SLAM system that fulfills the following four criteria: accurate, fast, robust, and cost-effective [29]. However these methods generate pure geometric maps which carry no additional semantic information that is required for high-level tasks, such as robotic planning, navigation, or AR applications [33, 32, 103, 116, 102].

Recently, in particular after the success of Convolutional Neural Networks (CNNs) in different visual recognition tasks such as object detection, and semantic segmentation, there has been an increasing interest in combining both geometric reconstruction and scene understanding in various ways in a single framework, for numerous autonomous

and virtual tasks. The aim of this combination is to create maps that include "meanings", both to human operators and moving robots or artificial agents. The *meanings* can contain semantic or detection information or even spatial or temporal relationships and affordances[1] between objects. Maps with additional semantic information facilitate the communication between humans and robots in order to reason about goals [153].

There are a multitude of approaches in the literature that have been applied to the problem of incorporating semantics and objects into the geometric reconstruction framework. We have proposed a new perspective towards the categorization of these methods in this chapter and a unified framework to explain them in the context of factor graphs. We classify these approaches based on the amount of influence of this incorporation on *localization* in next sections. We introduce the main research works in each category in the last recent years and also explain how our work in this thesis fits in this classification.

2.2.1 Semantic Mapping

The definition of *Semantic Mapping* that we propose in this chapter is as follows: "incorporating semantics in the mapping process without informing localization". In other words, semantic mapping does not involve with the localization process and the only concern is to assign high-level semantic information to the existing geometric landmarks in the map, unlike our proposed SLAM systems in this thesis which are informed by the semantics and objects during the localization process, not only mapping. In the context of factor graphs, semantic mapping approaches make an attempt to estimate or learn a mapping from geometric landmarks to some high-level semantic space, as illustrated in Fig 2.3. For instance the 3D points as landmarks in the map are semantically segmented and assigned some labels, such as assigning label s_1 (e.g. *chair*) to landmark $\mathbf{l}_1$ in Fig 2.3.

The estimation of this mapping from the space of landmarks to the space of semantics or high-level information, is determined by two different approaches in the recent works which we call them *offline*, and *online/incremental*. We explain some recent works that utilize these methods in the following sections.

[1] In particular interest of this thesis, *affordance* means the quality or property of an object that defines its possible uses, or actions that can afford.

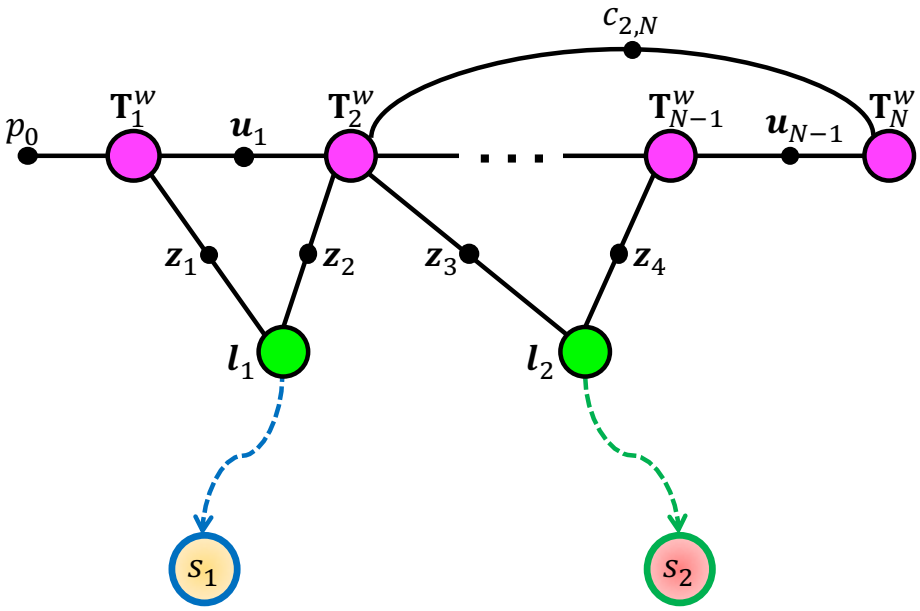


Fig. 2.3 A factor graph interpretation of a typical *Semantic Mapping* approach. The aim in this approach is predicting or learning a mapping from landmark representation space to some high-level semantic space *without* informing the localization, either after completing the whole map reconstruction (*offline method*) or incrementally during the map generation (*online/incremental method*).

2.2.1.1 Offline Approaches

In offline semantic mapping approaches, map reconstruction is followed by some 3D semantic segmentation method. In other terms, first the map of the environment is built by a geometric reconstruction system, such as SfM or SLAM framework, then this map is fed to another pipeline for semantic segmentation.

Unlike methods such as [34, 132, 133, 88, 135] which focus on single-frame 2D segmentations, offline semantic mapping approaches aim to yield semantically annotated 3D maps. In [69], the 3D map is reconstructed first of all by KinectFusion algorithm [102] from aligned RGB-D images, then the 3D scene is labeled in a structured learning framework by using Decision Tree Field [107] and Regression Tree Field [65], which learn the unary and pair-wise terms of a Conditional Random Field (CRF) from training data. Other work [3], builds the 3D point-cloud map by KinectFusion [102] and RGBDSLAM [31], then the point-cloud is over-segmented and labeled by using structured Support Vector Machines (SVM) [66].

Similar to [3] and [69], [112] generates the indoor scene, then segments the 3D scene by relaxing the labeling problem to a regression, and modeling higher-order potentials,

not only pair-wise CRFs, in order to describe complex relations and context based on robust associative P^n Potts models [73, 74], and regression forests [65] encoding Gaussian densities. In city-scale environments, [96] reconstructs a semi-dense 3D city model by SfM algorithms [165, 42, 64] from a street-scene, then classifies each point in the point-cloud into one semantic label using a Random Forest classifier trained on 3D features.

In addition to point-cloud model, other methods use different 3D representation for the scene, such as [160] which reconstructs a triangulated meshed representation of the scene by fusing multiple depth information of RGB-D stream by the algorithm presented in [20]. Then segments the scene by defining a CRF over this mesh, however besides 3D consistency features, it exploits appearance properties from 2D images. This methods generates globally consistent 3D maps, however the inference is performed on the whole map instead of incrementally fusing the predictions online. [78] also segments the completed 3D modeled indoor scene into vertices of a graphical model and infers the final semantic labels by using edge potentials based on handcrafted features.

In previous mentioned methods, the classification of the reconstructed 3D model is based on trained models on 2D or 3D semantic segmentation data. However there are some methods that avoid large ground-truth datasets and perform unsupervised 3D segmentation on the reconstructed scene [130, 108, 115, 122]. For instance, [113] instead of using supervised approaches, presents a geometric based method for the segmentation of the reconstructed 3D point cloud – using InfiniTAM [116] – into planes and meaningful objects by clustering an adjacency graph over surface patches. An organized point cloud segmentation approach is presented in [155] which segments planes then performs Euclidean clustering for small object segmentation. Their approach is based on connected component labeling similar to the two-pass algorithm, also known as Hoshen–Kopelman algorithm [61]. Without relying on plane fitting, [136] presents an efficient algorithm for object segmentation. Every adjacent pairs of local patches are classified into convexity or concavity using estimated surface normals from depth information, with the hope of indicating different objects based on concavity relations.

2.2.1.2 Online Approaches

Unlike offline approaches, in some recent works such as [83, 97, 172], map reconstruction and semantic segmentation are performed incrementally as the camera moves in the environment as we call them online or incremental approaches. Note that similar to

offline methods, there is no involvement of semantics or objects in the localization process in these works, unlike our SLAM system in this thesis.

As a straightforward solution [131] back-projects 2D image labels to 3D without further 3D optimization. Aiming towards a dense 3D semantic map, [140, 142] and [59] both predict per-pixel labels using Random Decision Forests. [142] uses a Multi-Resolution Surfel Map-based system, however does not keep a global semantic map during the mapping operation. [59] registers the predictions in the reference frame using only camera tracking phase and regularizes its predictions using fully-connected CRF inference method presented in [79]. [161] uses a Random Forest with a CRF for semantic labeling to produce an incremental map reconstruction from outdoor stereo pairs in KITTI benchmark [45], without having a full SLAM system to perform loop closure to yield a globally consistent map structure.

After the recent success of CNNs in classification and semantic segmentation applications [80, 92, 106] on various benchmarks [143, 101, 45, 123] and different modalities, in particular RGB [106, 92], depth [17, 53], and surface normals [30, 50, 51], recent works [97, 172, 151] use CNNs to predict semantic labels then combining them with the mapping process.

One of the recent successful works in combining CNNs for building indoor semantic maps incrementally is SemanticFusion [97] which utilizes ElasticFusion [164] dense SLAM system to represent the map by collections of surfels and provide correspondences between RGB-D frames. Then after reconstructing each frame, the trained CNN segments the 2D image semantically and in the last step the correspondences allow the Bayesian update of the semantic predictions from multiple views to be fused into the map, and this process continues incrementally.

Another work that combines 2D semantic segmentations using CNN and CRF optimization with geometric mapping is [172]. In this work the 3D geometric map is reconstructed by ORB-SLAM [100] from input stereo image pairs of KITTI [45] and densified by occupancy mapping [36, 129]. Then CNN segmentation [173] is utilized as prior unary potentials for a CRF model in addition to super-pixels [1] to enforce label consistency within a region. Eventually the final semantic mapping is refined incrementally by optimizing the hierarchical CRF in 3D grid space, unlike [83] which labels voxel's occupancy in one unified CRF. In addition to incremental semantic mapping, [150] explores further simultaneous object recognition and pose estimation on 3D data. The 3D reconstruction is based on KinectFusion approach [102], followed by an incremental merging of 3D segments based on [149]. Finally, for 3D object

recognition [150] computes the 3D descriptor [2] directly on each segment acquired from the incremental segmentation phase and match it with the descriptor calculated on the fully 3D object model.

Most of the above mentioned semantic mapping systems use stereo or RGB-D as input, however CNN-SLAM [151] fuses CNN-predicted dense depth maps – based on the ResNet architecture [57, 84] – with depth measurement acquired from direct monocular LSD-SLAM [32] and incrementally fuses semantic labels predicted from a single 2D image frame with dense map based on the approach in [149].

In an attempt to build an object-oriented map with both semantic and geometric entities, [145] performs camera localization and sparse mapping on every RGB-D frame using ORB-SLAM [100], followed by per RGB frame object detection by SSD [91]. Then this work incrementally assigns a 3D point cloud unsupervised segment [113, 38] to every detection, and associates object segments and detections based on a threshold on the "distance" of the point clouds.

Recently, [111] fuses a truncated signed distance function (TSDF) representation of the whole scene with CNN-based semantic segmentation, followed by a progressive CRF. For instance segmentation, it clusters semantically labeled 3D voxels. As a recent work towards dense reconstruction and tracking of moving instances, [125] uses ElasticFusion [164] surfel model for objects and background static map, and Mask R-CNN [58] predictions for object instance detections.

2.2.2 Semantic SLAM

The definition of *Semantic SLAM* that we propose in this chapter in contrast to semantic mapping approaches, is as follows: " integrating semantics and object-level entities into the mapping of the scene, while informing the localization process". This incorporation of high-level information can indirectly influence the localization and camera tracking by improving the data-association and changing the topology of the factor graph representing the SLAM problem, or by actively involving in "object representation space" optimization within the bundle adjustment.

The current advances in machine learning and in particular deep CNNs [80, 92, 106, 57], facilitates exploiting rich priors from data with different modalities, from proposal based object detections [48, 47, 121, 91, 120] to dense pixel-wise semantic segmentations [92, 6, 176, 166], and instance-level semantic segmentations [174, 22, 175, 159]. These deep priors can enrich the geometric reconstructions semantically while improving the accuracy of mapping and localization simultaneously. Our proposed SLAM system in

this thesis lies in this category since by the proposed structural entities (planes) and semantic coarse object landmarks, it improves the accuracy of the estimated camera trajectory and mapping, in contrast to semantic mapping approaches which use SLAM as a backbone and perform mapping processes in parallel.

In the context of factor graph, semantic SLAM approach can be defined as any method which involves introducing semantics/objects into the factor graph representation of the SLAM problem. With our proposed perspective towards semantic SLAM in this chapter, we classify the related approaches in two different categories which both have been studied in the recent literature. This classification is based on the way in which semantics or objects participate in the factor graph, explained in the following sections along with some of the major works in each category. Finally, we justify our work in the proposed categorization of the related semantic SLAM works.

2.2.2.1 First Approach

In the first category of approaches, semantics and objects can affect the tracking of the camera indirectly by changing the topology of the graph with data-association, or analogous to Pose SLAM methods [76, 77, 94] described earlier, by marginalizing the shape of the object/semantic landmarks and optimizing the pose graph within the back-end. In the works related to this approach [71, 127, 13, 98], typically the representation of the object in the pose graph is the 6D object pose or 3D location of the centroid of object. In other words, the shape of the objects do not involve directly in the pose graph optimization and only pose or location of the objects is refined in the back-end. The relative pose constraint factor between camera key-frames is generally based on iterative closest point (ICP) [9] error metric resulting from registering object reconstructions from different viewpoints. In the front-end of these SLAM systems, the object models are retrieved from a predefined database [127] or refined gradually by observing more segments of them [98]. Furthermore, they make use of ICP-like algorithms for live camera frame tracking and localization. A typical factor graph of semantic SLAM systems that apply the first introduced approach is illustrated in Fig 2.4 and in the following we describe some of the related works in more detail. Note that our work is different from these methods in a sense that we actively engage the shape and pose of our proposed object landmarks during the bundle adjustment not only pose graph optimization.

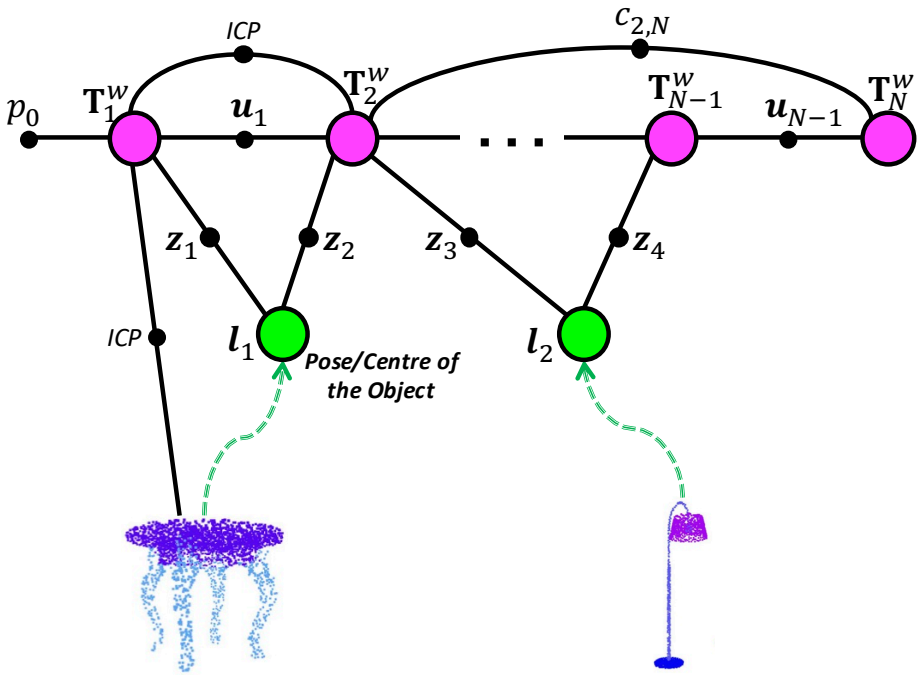


Fig. 2.4 A factor graph interpretation of a typical *first approach* for *Semantic SLAM* problem. The aim of this approach is incorporating high-level semantics and objects in the SLAM framework while informing the localization process, however analogous to Pose SLAM methods, by marginalizing the shape of the object/semantic landmarks and optimizing typically the 6 DoF pose or 3D location of the object in the pose graph. The generic factors used in these approaches are ICP-based errors for registration of 3D object models.

One early work to combine object recognition and SLAM is [119] which uses a 2D laser/camera system to recognize objects in order to create discrete entities in the map (tree trunks) instead of using raw measurements.

SLAM++ [127] is an early RGB-D object-oriented SLAM thats uses pair features for object detection [28, 5] and a pose graph for global optimization [82, 137]. The live camera tracking is estimated using a fast dense ICP algorithm [126] against a complete multi-object model prediction. One drawback of [127] is the requirement of that the full collection of object instances along with their very elaborate geometric shape models, have to be known beforehand and preprocessed in an offline phase. They create an object database by using KinectFusion [102] and marching cubes algorithm [93]. In another similar work, [71] uses a depth camera to scan the scene and all data is then fused into a single map representation. Then offline learned object models

are matched against the map and optimization is performed with both similarity and configuration of objects.

In [13], the camera localization is done online [154, 156] and objects are discovered using the unsupervised segmentation approach in [155, 60, 38] and matched by a graph-based strategy [16]. The discovered objects are added as landmarks into the pose graph, however they only optimize the object's point-cloud centroid in the graph.

Unlike approaches such as [15, 141, 40, 95, 37, 114, 27] which mainly use SLAM to assist object discovery, recently published Fusion++ [98] uses discovered objects as landmarks in the pose graph formulation. Analogous to [177], [98] aims to optimize the quality of object reconstructions by marginalizing them in the pose graph. Objects are detected using Mask-RCNN [58] and represented by truncated signed distance functions (TSDFs) [20] instead of point clouds. The 6 DoF $\mathbf{SE}(3)$ pose of the object are refined in the pose graph optimization within the back-end of the system.

Finally, the same idea about incorporating object recognition and scene reconstruction has also been used in offline structure from motion mode such as [7], which represents a map as a collection of points, objects, and regions in two-view SfM, solving very slowly and jointly in a graph optimization fashion.

2.2.2.2 Second Approach

In contrast to the fist approach towards incorporating semantics and objects in SLAM, in the second approach both shape and pose of the object are engaged during the bundle adjustment. In the context of factor graph, this engagement necessitates an independent representation for objects or semantically meaningful structures as variable nodes in the factor graph, in order to directly participate in the optimization within the back-end, as shown in Fig 2.5. Both shape and pose of the object participate in the bundle adjustment, unlike the first approach in which only pose or location of the 3D object involves in the pose graph optimization. In the works using this approach, live camera frame tracking also benefits from the factor graph optimization with the presence of objects as independent landmarks.

Appropriate representations are critical for the related works in this category and in general two types of representation are conceivable in the literature: **1)** high-fidelity or fine, and **2)** coarse representations.

Recently with the availability of large CAD model repositories [11], there has been many attentions to high-fidelity 3D object model reconstruction or hallucination from single or multiple views using data hungry deep CNNs [14, 46, 148, 54, 35, 87, 147],

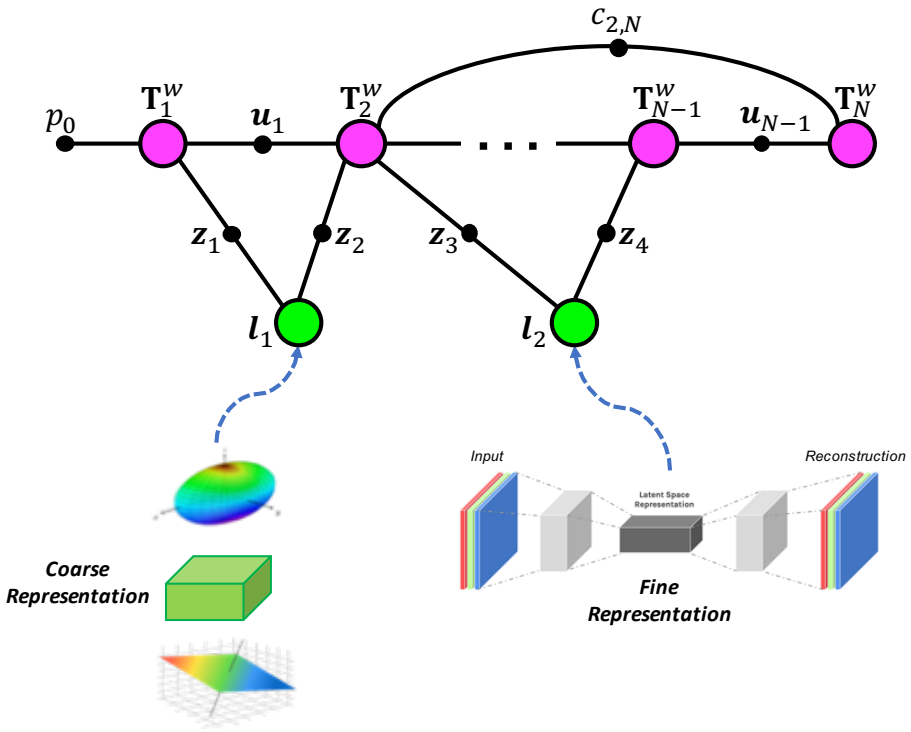


Fig. 2.5 A factor graph interpretation of a typical *second approach* for *Semantic SLAM* problem. The aim of this approach is incorporating high-level semantics and objects in the SLAM framework while informing the localization process, however in contrast to the first approaches, both shape and pose of the objects are engaged during the bundle adjustment. In this approach objects or structural entities are represented – either coarse or fine – as independent variable nodes participating in the BA optimization.

that led to the latent space representation for 3D objects. In these methods, various kinds of representations have been used for modeling 3D objects, such as point-clouds, signed distance function (SDF) or truncated SDF (TSDF), voxels. Most of the object hallucination or reconstruction methods only make use of single-view of the object [147, 35, 86], however there has been efforts to refine the shape of the object from multiple views by optimizing in the latent space [14, 87, 85] but these limited approaches are expensive in terms of time and computation and not robust enough against occlusion, hence not applicable yet for real-time SLAM systems.

The more popular representation for objects in this category of semantic SLAM approach is the coarse representation for the shape and pose of the objects. Different coarse representations has been proposed for the pose (location and orientation) and extent or rough shape of 3D objects, such as cuboid [168], quadrics [144], super quadrics [8]. Data-association is more elegant with this coarse representations compared to fine

object models. For structural entities some works integrate line segments or edges into the SLAM framework [117, 49], however despite the more complex mathematical representation for line segments and non-robust data-association for them, they carry very limited high-level or semantic information. The most popular structural entity for SLAM framework which also carries semantic information is plane. There are recent works that make use of planes in SLAM framework [146, 67, 171, 62], however unlike our proposed SLAM, they use planes for fusing depth information of points and 3D data registration [118, 162, 110, 109] not as independent landmarks, or only using planes in the map with no further landmark [67] that restrain their generalization to broader indoor or outdoor environments.

Our proposed SLAM systems in this thesis lie in this category of approaches towards semantic SLAM. We propose a novel integration of quadrics as coarse representation for objects that permits real-time performance, and also incorporate planes along with other landmarks in the map such as points that makes our system applicable to more generic scenes. In addition to high-level constraints between landmarks in the map, we also integrate fine 3D reconstruction of objects on top of the coarse quadrics in order to enrich the map and refine the quadric shapes even more, without engaging fine 3D point-clouds directly in the BA. In the following we explain some of the related works in more detail.

Built on top of point-based ORB-SLAM [100], [117] amends point and line correspondences to the framework, which improves the initialization of the monocular pipeline. [49] also considers a loop closure detection with the presence of points and lines in the stereo modality.

Unlike some recent works that predict planar structures from single images particularly with deep CNNs [167, 52, 90, 89, 158], the early work [10] identifies known planar regions in the scene and incorporates their geometry into the extended Kalman Filter formulation of [23]. [154] uses a mixture of large field-of-view (FOV) 2D laser scanner and a small FOV 3D sensor to use both plane-to-plane and line-to-plane correspondences in 3D data registration within a SLAM system. By using RGB-D camera, [146] registers 3D data in two different frames using various combinations of point and plane primitives using a RANSAC framework [41]. In RGB-D modality, [128] maps the scene by a collection of surfels based on [70] and labels each surfels in 3D map with either one of a number of discrete planar regions or no label. Then data-associated planes are unified and merged in one reference world frame. In fact these planar regions help point depth fusion to occur more accurately. [67] includes infinite planes in the least-squares

formulation for mapping with depth cameras, using a homogeneous representation of planes and refines the map consisting of only planes with incremental solvers such as iSAM [68]. For low-texture scenes, [171] trains a CNN-based monocular 3D plane model based on [170] applied to each single image, then includes plane detections in a least-squares framework of [67] with further loop-closure constraints. [62] A key-frame based approach proposed in [62] which estimates odometry, fuses depth measurements, and detects planes from the fused depth map, and refines the key-frame poses and only-plane landmarks with an incremental factor graph solver (iSAM) [68]. [63] extends the previous work to also include inertial measurements by IMUs.

For coarse representation of objects in 3D map, cuboids have been recently used as a generalization of 2D bounding boxes. Closely related to our work in this thesis, [168, 169] propose single image 3D cuboid object detection and multi-view SLAM with object and plane landmarks. In this thesis objects are represented differently by bounded dual quadrics [19, 56] which are mathematically more convenient in projective setup and more amenable for real-time performance. They infer 3D objects and layout planes from a single image using a CNN and refine them in a unified SLAM framework [82]. Superquadrics (a general family of geometric shapes unified by formulas that resemble those of ellipsoids and cuboids, except that the squaring operations are replaced by arbitrary powers [8]) has been used to model 3D deformable objects [8, 152, 134, 39], however they are not suitable and popular for SLAM frameworks, due to the computational complexity inherent in their representation particularly their projective geometry. On the other hand, quadrics (generalization of conic sections in 3D space) based representation for geometric reconstruction was first proposed in [19] and later has been used in a structure from motion setup [18, 43, 124, 44]. [144] reconstructs quadrics based on detected 2D bounding boxes in frames, however the objects are not explicitly modeled to remain bounded during graph optimization. Addressing previous drawback, [104] still relies on ground-truth data-association in a non-real-time quadric-only framework.

2.3 Conclusion

In this chapter we described the problem of geometric reconstruction with its main two categories: SfM and SLAM. In the category of SLAM, factor graph is introduced as the modern probabilistic tool for formulating the SLAM problem. To integrate visual scene understanding and geometric reconstruction, there has been works to incorporate

semantics and objects into the SLAM framework. We proposed definitions to distinguish two major streams of works in this regard: **1)** *Semantic Mapping* which incorporates semantics in the mapping without informing localization, and **2)** *Semantic SLAM* which integrates objects and semantically meaningful entities in the mapping with influencing the localization process. Our work in this thesis falls in the semantic SLAM class. However, in this category we create two subclasses based on the current literature that in the first one, only pose or location of the objects participates in the factor graph optimization, analogous to Pose SLAM methods, and in the second subclass the whole shape and pose of the object involve directly in the bundle adjustment.

Our work lies in the second subclass of semantic SLAM approaches, since we introduce novel coarse representation for objects and also planar structures in order to integrate them in the bundle adjustment. Despite recent latent space representations for fine reconstruction of 3D objects, they are computationally expensive for engaging directly in the BA and not suitable for our real-time aim. We introduce representations for bounded quadrics (ellipsoids) and planes that are amenable for factor graph framework and include them in the point-based sparse SLAM. We also benefit from the mixture of landmarks in data-association and initialization of the vertices of the graph, which eventually allows real-time and more accurate performance compared with the related works that are non-real-time and use quadrics in the SfM fashion [18, 43, 124, 44] or with ground-truth data-association frameworks [144, 104]. The coarse quadric object landmark, also permits elegant integration of fine point-cloud reconstruction of objects to be included on top of the quadric in the map and refine the coarse representation even more, besides further enrichment of the map. Note that our proposed SLAM, to the best of our knowledge, is the first sparse SLAM that includes all points, planes, and coarse objects in single real-time monocular framework.

Bibliography

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE transactions on pattern analysis and machine intelligence*, 34(11):2274–2282, 2012.

[2] A. Aldoma, F. Tombari, J. Prankl, A. Richtsfeld, L. Di Stefano, and M. Vincze. Multimodal cue integration through hypotheses verification for rgb-d object recognition and 6dof pose estimation. In *2013 IEEE International Conference on Robotics and Automation*, pages 2104–2111. IEEE, 2013.

[3] A. Anand, H. S. Koppula, T. Joachims, and A. Saxena. Contextually guided semantic labeling and search for three-dimensional point clouds. *The International Journal of Robotics Research*, 32(1):19–34, 2013.

[4] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): Part ii. *IEEE Robotics & Automation Magazine*, 13(3):108–117, 2006.

[5] D. H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[6] A. Bansal, X. Chen, B. Russell, A. Gupta, and D. Ramanan. Pixelnet: Representation of the pixels, by the pixels, and for the pixels. *arXiv preprint arXiv:1702.06506*, 2017.

[7] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2703–2710. IEEE, 2012.

[8] A. H. Barr. Superquadrics and angle-preserving transformations. *IEEE Computer graphics and Applications*, 1(1):11–23, 1981.

[9] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*, volume 1611, pages 586–607. International Society for Optics and Photonics, 1992.

[10] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray. Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4102–4107. IEEE, 2007.

[11] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[12] S. Choi, Q.-Y. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5556–5565, 2015.

[13] S. Choudhary, A. J. Trevor, H. I. Christensen, and F. Dellaert. Slam with object discovery, modeling and mapping. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1018–1025. IEEE, 2014.

[14] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[15] A. Collet, B. Xiong, C. Gurau, M. Hebert, and S. S. Srinivasa. Exploiting domain knowledge for object discovery. In *2013 IEEE International Conference on Robotics and Automation*, pages 2118–2125. IEEE, 2013.

[16] C. Couprie, C. Farabet, Y. LeCun, and L. Najman. Causal graph-based video segmentation. In *2013 IEEE International Conference on Image Processing*, pages 4249–4253. IEEE, 2013.

[17] C. Couprie, C. Farabet, L. Najman, and Y. LeCun. Indoor semantic segmentation using depth information. *arXiv preprint arXiv:1301.3572*, 2013.

[18] M. Crocco, C. Rubino, and A. Del Bue. Structure from motion with objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4141–4149, 2016.

[19] G. Cross and A. Zisserman. Quadric reconstruction from dual-space geometry. In *Computer Vision, 1998. Sixth International Conference on*, pages 25–31. IEEE, 1998.

[20] B. Curless and M. Levoy. A volumetric method for building complex models from range images. 1996.

[21] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4):76a, 2017.

[22] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158, 2016.

[23] A. J. Davison. Real-time simultaneous localisation and mapping with a single camera. In *null*, page 1403. IEEE, 2003.

[24] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007.

[25] F. Dellaert. Factor graphs and gtsam: A hands-on introduction. Technical report, Georgia Institute of Technology, 2012.

[26] F. Dellaert, M. Kaess, et al. Factor graphs for robot perception. *Foundations and Trends® in Robotics*, 6(1-2):1–139, 2017.

[27] T. Dharmasiri, V. Lui, and T. Drummond. Mo-slam: Multi object slam with run-time object discovery through duplicates. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1214–1221. IEEE, 2016.

[28] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3d object recognition. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 998–1005. Ieee, 2010.

[29] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE robotics & automation magazine*, 13(2):99–110, 2006.

[30] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015.

[31] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the rgb-d slam system. In *Icra*, pages 1691–1696, 2012.

[32] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[33] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, March 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2658577.

[34] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[35] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 605–613, 2017.

[36] Z. Fang, S. Yang, S. Jain, G. Dubey, S. Roth, S. Maeta, S. Nuske, Y. Zhang, and S. Scherer. Robust autonomous flight in constrained and visually degraded shipboard environments. *Journal of Field Robotics*, 34(1):25–52, 2017.

[37] M. Fehr, F. Furrer, I. Dryanovski, J. Sturm, I. Gilitschenski, R. Siegwart, and C. Cadena. Tsdf-based change detection for consistent long-term dense reconstruction and dynamic object discovery. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 5237–5244. IEEE, 2017.

[38] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *International journal of computer vision*, 59(2):167–181, 2004.

[39] F. P. Ferrie, J. Lagarde, and P. Whaite. Darboux frames, snakes, and superquadrics: Geometry from the bottom up. *IEEE transactions on pattern analysis and machine intelligence*, 15(8):771–784, 1993.

[40] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard. Toward lifelong object segmentation from change detection in dense rgb-d maps. In *2013 European Conference on Mobile Robots*, pages 178–185. IEEE, 2013.

[41] 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*, 24(6):381–395, 1981.

[42] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *IEEE transactions on pattern analysis and machine intelligence*, 32(8):1362–1376, 2010.

[43] P. Gay, C. Rubino, V. Bansal, and A. Del Bue. Probabilistic structure from motion with objects (psfmo). In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3075–3084, 2017.

[44] P. Gay, S. James, and A. Del Bue. Visual graphs from motion (vgfm): Scene understanding with object geometry reasoning. *arXiv preprint arXiv:1807.05933*, 2018.

[45] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3354–3361. IEEE, 2012.

[46] R. Girdhar, D. F. Fouhey, M. Rodriguez, and A. Gupta. Learning a predictable and generative vector representation for objects. In *European Conference on Computer Vision*, pages 484–499. Springer, 2016.

[47] R. Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[48] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[49] R. Gomez-Ojeda, D. Zuñiga-Noël, F.-A. Moreno, D. Scaramuzza, and J. Gonzalez-Jimenez. Pl-slam: a stereo slam system through the combination of points and line segments. *arXiv preprint arXiv:1705.09479*, 2017.

[50] S. Gupta, R. Girshick, P. Arbeláez, and J. Malik. Learning rich features from rgb-d images for object detection and segmentation. In *European Conference on Computer Vision*, pages 345–360. Springer, 2014.

[51] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Aligning 3d models to rgb-d images of cluttered scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4731–4740, 2015.

[52] O. Haines and A. Calway. Recognising planes in a single image. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1849–1861, 2015.

[53] A. Handa, V. Patraucean, V. Badrinarayanan, S. Stent, and R. Cipolla. Understanding real world indoor scenes with synthetic data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4077–4085, 2016.

[54] C. Häne, S. Tulsiani, and J. Malik. Hierarchical surface prediction for 3d object reconstruction. In *2017 International Conference on 3D Vision (3DV)*, pages 412–420. IEEE, 2017.

[55] C. G. Harris and J. Pike. 3d positional integration from image sequences. *Image and Vision Computing*, 6(2):87–90, 1988.

[56] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[57] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[58] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[59] A. Hermans, G. Floros, and B. Leibe. Dense 3d semantic mapping of indoor scenes from rgb-d images. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2631–2638. IEEE, 2014.

[60] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab. Adaptive neighborhood selection for real-time surface normal estimation from organized point

cloud data using integral images. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2684–2689. IEEE, 2012.

[61] J. Hoshen and R. Kopelman. Percolation and cluster distribution. i. cluster multiple labeling technique and critical concentration algorithm. *Physical Review B*, 14(8):3438, 1976.

[62] M. Hsiao, E. Westman, G. Zhang, and M. Kaess. Keyframe-based dense planar slam. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5110–5117. IEEE, 2017.

[63] M. Hsiao, E. Westman, and M. Kaess. Dense planar-inertial slam with structural constraints. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6521–6528. IEEE, 2018.

[64] M. Jancosek and T. Pajdla. Multi-view reconstruction preserving weakly-supported surfaces. In *CVPR 2011*, pages 3121–3128. IEEE, 2011.

[65] J. Jancsary, S. Nowozin, T. Sharp, and C. Rother. Regression tree fields—an efficient, non-parametric approach to image labeling problems. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2376–2383. IEEE, 2012.

[66] T. Joachims, T. Finley, and C.-N. J. Yu. Cutting-plane training of structural svms. *Machine Learning*, 77(1):27–59, 2009.

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

[68] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. isam2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3281–3288. IEEE, 2011.

[69] O. Kahler and I. Reid. Efficient 3d scene labeling using fields of trees. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3064–3071, 2013.

[70] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. Real-time 3d reconstruction in dynamic scenes using point-based fusion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 1–8. IEEE, 2013.

[71] Y. M. Kim, N. J. Mitra, D.-M. Yan, and L. Guibas. Acquiring 3d indoor environments with variability and repetition. *ACM Transactions on Graphics (TOG)*, 31(6):138, 2012.

[72] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc. Sixth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR'07)*, Nara, Japan, November 2007.

[73] P. Kohli, M. P. Kumar, and P. H. Torr. P3 & beyond: Solving energies with higher order cliques. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[74] P. Kohli, P. H. Torr, et al. Robust higher order potentials for enforcing label consistency. *International Journal of Computer Vision*, 82(3):302–324, 2009.

[75] K. Konolige and M. Agrawal. Frameslam: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008.

[76] K. Konolige, J. Bowman, J. Chen, P. Mihelich, M. Calonder, V. Lepetit, and P. Fua. View-based maps. *The International Journal of Robotics Research*, 29 (8):941–957, 2010.

[77] K. Konolige, G. Grisetti, R. Kümmerle, W. Burgard, B. Limketkai, and R. Vincent. Efficient sparse pose adjustment for 2d mapping. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 22–29. IEEE, 2010.

[78] H. S. Koppula, A. Anand, T. Joachims, and A. Saxena. Semantic labeling of 3d point clouds for indoor scenes. In *Advances in neural information processing systems*, pages 244–252, 2011.

[79] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In *Advances in neural information processing systems*, pages 109–117, 2011.

[80] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[81] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

[82] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE, 2011.

[83] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *European Conference on Computer Vision*, pages 703–718. Springer, 2014.

[84] I. Laina, C. Rupprecht, V. Belagiannis, F. Tombari, and N. Navab. Deeper depth prediction with fully convolutional residual networks. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 239–248. IEEE, 2016.

[85] K. Li, R. Garg, M. Cai, and I. Reid. Optimizable object reconstruction from a single view. *arXiv preprint arXiv:1811.11921*, 2018.

[86] K. Li, T. Pham, H. Zhan, and I. Reid. Efficient dense point cloud object reconstruction using deformation vector fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 497–513, 2018.

[87] C.-H. Lin, C. Kong, and S. Lucey. Learning efficient point cloud generation for dense 3d object reconstruction. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[88] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[89] C. Liu, K. Kim, J. Gu, Y. Furukawa, and J. Kautz. Planercnn: 3d plane detection and reconstruction from a single image. *arXiv preprint arXiv:1812.04072*, 2018.

[90] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa. Planenet: Piece-wise planar reconstruction from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2588, 2018.

[91] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[92] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[93] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. In *ACM siggraph computer graphics*, volume 21, pages 163–169. ACM, 1987.

[94] F. Lu and E. Milios. Globally consistent range scan alignment for environment mapping. *Autonomous robots*, 4(4):333–349, 1997.

[95] L. Ma and G. Sibley. Unsupervised dense object discovery, detection, tracking and reconstruction. In *European Conference on Computer Vision*, pages 80–95. Springer, 2014.

[96] A. Martinovic, J. Knopp, H. Riemenschneider, and L. Van Gool. 3d all the way: Semantic segmentation of urban scenes from start to end in 3d. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4456–4465, 2015.

[97] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *2017 IEEE International Conference on Robotics and automation (ICRA)*, pages 4628–4635. IEEE, 2017.

[98] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018.

[99] R. Mur-Artal and J. D. Tardós. ORB-SLAM2: an open-source SLAM system for monocular, stereo and RGB-D cameras. *CoRR*, abs/1610.06475, 2016. URL http://arxiv.org/abs/1610.06475.

[100] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardós. Orb-slam: A versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5): 1147–1163, Oct 2015.

[101] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[102] 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 *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[103] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[104] L. Nicholson, M. Milford, and N. Sünderhauf. Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam. *IEEE Robotics and Automation Letters*, 4(1):1–8, 2019.

[105] D. Nister. An efficient solution to the five-point relative pose problem. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–195. IEEE, 2003.

[106] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[107] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli. Decision tree fields. In *2011 International Conference on Computer Vision*, pages 1668–1675. IEEE, 2011.

[108] A. Nüchter and J. Hertzberg. Towards semantic maps for mobile robots. *Robotics and Autonomous Systems*, 56(11):915–926, 2008.

[109] C. Olsson, F. Kahl, and M. Oskarsson. The registration problem revisited: Optimal solutions from points, lines and planes. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 1206–1213. IEEE, 2006.

[110] K. Pathak, A. Birk, N. Vaskevicius, and J. Poppinga. Fast registration based on noisy planes with unknown correspondences for 3-d mapping. *IEEE Transactions on Robotics*, 26(3):424–441, 2010.

[111] Q.-H. Pham, B.-S. Hua, D. T. Nguyen, and S.-K. Yeung. Real-time progressive 3d semantic segmentation for indoor scene. *arXiv preprint arXiv:1804.00257*, 2018.

[112] T. T. Pham, I. Reid, Y. Latif, and S. Gould. Hierarchical higher-order regression forest fields: An application to 3d indoor scene labelling. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2246–2254, 2015.

[113] T. T. Pham, M. Eich, I. Reid, and G. Wyeth. Geometrically consistent plane extraction for dense indoor 3d maps segmentation. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4199–4204. IEEE, 2016.

[114] S. Pillai and J. Leonard. Monocular slam supported object recognition. *arXiv preprint arXiv:1506.01732*, 2015.

[115] J. Poppinga, N. Vaskevicius, A. Birk, and K. Pathak. Fast plane detection and polygonalization in noisy 3d range images. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3378–3383. IEEE, 2008.

[116] V. Prisacariu, O. Kahler, M. Cheng, C. Ren, J. Valentin, P. Torr, I. Reid, and D. Murray. A Framework for the Volumetric Integration of Depth Images. *ArXiv e-prints*, 2014.

[117] 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)*, pages 4503–4508. IEEE, 2017.

[118] S. Ramalingam and Y. Taguchi. A theory of minimal 3d point to 3d plane registration and its generalization. *International journal of computer vision*, 102 (1-3):73–90, 2013.

[119] F. Ramos, J. Nieto, and H. Durrant-Whyte. Combining object recognition and slam for extended map representations. In *Experimental Robotics*, pages 55–64. Springer, 2008.

[120] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[121] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[122] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich, and M. Vincze. Segmentation of unknown objects in indoor environments. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4791–4796. IEEE, 2012.

[123] G. Ros, L. Sellart, J. Materzynska, D. Vázquez, and A. M. López. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016.

[124] C. Rubino, M. Crocco, and A. Del Bue. 3d object localisation from multi-view image detections. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1281–1294, 2018.

[125] M. Runz, M. Buffier, and L. Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. In *2018 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 10–20. IEEE, 2018.

[126] S. Rusinkiewicz and M. Levoy. Efficient variants of the icp algorithm. In *3dim*, page 145. IEEE, 2001.

[127] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013.

[128] R. F. Salas-Moreno, B. Glocken, P. H. Kelly, and A. J. Davison. Dense planar slam. In *2014 IEEE international symposium on mixed and augmented reality (ISMAR)*, pages 157–164. IEEE, 2014.

[129] S. Scherer, J. Rehder, S. Achar, H. Cover, A. Chambers, S. Nuske, and S. Singh. River mapping from a flying robot: state estimation, river detection, and obstacle mapping. *Autonomous Robots*, 33(1-2):189–214, 2012.

[130] R. Schnabel, R. Wahl, and R. Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library, 2007.

[131] S. Sengupta, E. Greveson, A. Shahrokni, and P. H. Torr. Urban 3d semantic modelling using stereo vision. In *2013 IEEE International Conference on robotics and Automation*, pages 580–585. IEEE, 2013.

[132] N. Silberman and R. Fergus. Indoor scene segmentation using a structured light sensor. In *2011 IEEE international conference on computer vision workshops (ICCV workshops)*, pages 601–608. IEEE, 2011.

[133] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer, 2012.

[134] F. Solina and R. Bajcsy. Recovery of parametric models from range images: The case for superquadrics with global deformations. *IEEE transactions on pattern analysis and machine intelligence*, 12(2):131–147, 1990.

[135] S. Song, S. P. Lichtenberg, and J. Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 567–576, 2015.

[136] S. C. Stein, F. Wörgötter, M. Schoeler, J. Papon, and T. Kulvicius. Convexity based object partitioning for robot applications. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3213–3220. IEEE, 2014.

[137] H. Strasdat. *Local accuracy and global consistency for efficient visual SLAM*. PhD thesis, Department of Computing, Imperial College London, 2012.

[138] H. Strasdat, J. Montiel, and A. J. Davison. Real-time monocular slam: Why filter? In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 2657–2664. IEEE, 2010.

[139] H. Strasdat, A. J. Davison, J. M. Montiel, and K. Konolige. Double window optimisation for constant time visual slam. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2352–2359. IEEE, 2011.

[140] J. Stückler and S. Behnke. Model learning and real-time tracking using multi-resolution surfel maps. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*, 2012.

[141] J. Stückler and S. Behnke. Hierarchical object discovery and dense modelling from motion cues in rgb-d video. In *Twenty-Third International Joint Conference on Artificial Intelligence*, 2013.

[142] J. Stückler and S. Behnke. Multi-resolution surfel maps for efficient dense 3d modeling and tracking. *Journal of Visual Communication and Image Representation*, 25(1):137–147, 2014.

[143] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[144] N. Sünderhauf and M. Milford. Dual quadrics from object detection bounding-boxes as landmark representations in slam. *arXiv preprint arXiv:1708.00965*, 2017.

[145] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid. Meaningful maps with object-oriented semantic mapping. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5079–5085. IEEE, 2017.

[146] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng. Point-plane slam for hand-held 3d sensors. In *2013 IEEE International Conference on Robotics and Automation*, pages 5182–5189. IEEE, 2013.

[147] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Multi-view 3d models from single images with a convolutional network. In *European Conference on Computer Vision*, pages 322–337. Springer, 2016.

[148] M. Tatarchenko, A. Dosovitskiy, and T. Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3d outputs. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2088–2096, 2017.

[149] K. Tateno, F. Tombari, and N. Navab. Real-time and scalable incremental segmentation on dense slam. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4465–4472. IEEE, 2015.

[150] K. Tateno, F. Tombari, and N. Navab. When 2.5 d is not enough: Simultaneous reconstruction, segmentation and recognition on dense slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2295–2302. IEEE, 2016.

[151] K. Tateno, F. Tombari, I. Laina, and N. Navab. Cnn-slam: Real-time dense monocular slam with learned depth prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6243–6252, 2017.

[152] D. Terzopoulos and D. Metaxas. Dynamic 3d models with local and global deformations: Deformable superquadrics. In *[1990] Proceedings Third International Conference on Computer Vision*, pages 606–615. IEEE, 1990.

[153] A. J. Trevor, J. G. Rogers, C. Nieto-Granda, and H. I. Christensen. Tables, counters, and shelves: Semantic mapping of surfaces in 3d. Georgia Institute of Technology, 2010.

[154] 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*, pages 3041–3048. IEEE, 2012.

[155] A. J. Trevor, S. Gedikli, R. B. Rusu, and H. I. Christensen. Efficient organized point cloud segmentation with connected components. *Semantic Perception Mapping and Exploration (SPME)*, 2013.

[156] A. J. Trevor, J. G. Rogers, and H. I. Christensen. Omnimapper: A modular multimodal mapping framework. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1983–1990. IEEE, 2014.

[157] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle Adjustment?A Modern Synthesis. In *International Workshop on Vision Algorithms*, pages 298–372. Springer, 1999.

[158] S. Tulsiani, R. Tucker, and N. Snavely. Layer-structured 3d scene inference via view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 302–317, 2018.

[159] J. Uhrig, M. Cordts, U. Franke, and T. Brox. Pixel-level encoding and depth layering for instance-level semantic labeling. In *German Conference on Pattern Recognition*, pages 14–25. Springer, 2016.

[160] J. P. Valentin, S. Sengupta, J. Warrell, A. Shahrokni, and P. H. Torr. Mesh based semantic modelling for indoor and outdoor scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2067–2074, 2013.

[161] V. Vineet, O. Miksik, M. Lidegaard, M. Nießner, S. Golodetz, V. A. Prisacariu, O. Kähler, D. W. Murray, S. Izadi, P. Pérez, et al. Incremental dense semantic stereo fusion for large-scale semantic scene reconstruction. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 75–82. IEEE, 2015.

[162] J. Weingarten and R. Siegwart. 3d slam using planar segments. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3062–3067. IEEE, 2006.

[163] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald. Kintinuous: Spatially extended kinectfusion. 2012.

[164] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison. Elasticfusion: Dense slam without a pose graph. Robotics: Science and Systems, 2015.

[165] C. Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE, 2013.

[166] Z. Wu, C. Shen, and A. Van Den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 90:119–133, 2019.

[167] F. Yang and Z. Zhou. Recovering 3d planes from a single image via convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018.

[168] S. Yang and S. Scherer. Cubeslam: Monocular 3d object detection and slam without prior models. *arXiv preprint arXiv:1806.00557*, 2018.

[169] S. Yang and S. Scherer. Monocular object and plane slam in structured environments. *arXiv preprint arXiv:1809.03415*, 2018.

[170] S. Yang, D. Maturana, and S. Scherer. Real-time 3d scene layout from a single image using convolutional neural networks. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2183–2189. IEEE, 2016.

[171] S. Yang, Y. Song, M. Kaess, and S. Scherer. Pop-up slam: Semantic monocular plane slam for low-texture environments. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1222–1229. IEEE, 2016.

[172] S. Yang, Y. Huang, and S. Scherer. Semantic 3d occupancy mapping through efficient high order crfs. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 590–597. IEEE, 2017.

[173] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. 2016.

[174] S. Zagoruyko, A. Lerer, T.-Y. Lin, P. O. Pinheiro, S. Gross, S. Chintala, and P. Dollár. A multipath network for object detection. *arXiv preprint arXiv:1604.02135*, 2016.

[175] Z. Zhang, S. Fidler, and R. Urtasun. Instance-level segmentation for autonomous driving with deep densely connected mrfs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 669–677, 2016.

[176] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[177] Q.-Y. Zhou and V. Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (ToG)*, 32(4):112, 2013.

[178] Q.-Y. Zhou, S. Miller, and V. Koltun. Elastic fragments for dense scene reconstruction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 473–480, 2013.

Chapter 3

Sparse Point-Plane SLAM

In this chapter we propose a representation for planes to introduce them as higher level sparse geometric entities that capture the dominant structure of the scene and incorporate them into the sparse point-based SLAM framework. We first provide a brief introduction of the relevant problem by reviewing the related literature, and then explain our SLAM system in three sections: first in 3.3 we discuss our proposed mathematical representations for these new entities as variable nodes of the factor graph (state variables) and then we propose the observation/constraint factors between landmarks in 3.4, such as point-plane constraints and plane-plane (Manhattan assumption). Finally the front-end of the SLAM system that extracts and builds the sparse factor graph of our problem from sensory information (RGB-D input) is elaborated in 3.5. At the end we evaluate our system extensively against the state-of-the-art point-based SLAM systems as an ablation study and comparative study with publicly available benchmarks, qualitatively and quantitatively, unlike most of the visual SLAM systems that lack an exhaustive evaluation against the state-of-the-art benchmarks. We have shown a significant boost in the accuracy of the SLAM system while keeping the real-time performance with a negligible additional memory load.

3.1 Introduction

Simultaneous Localization And Mapping (SLAM) is one of the fundamental problems in mobile robotics [2] and addresses the reconstruction of a previously unseen environment while simultaneously localizing a mobile robot with respect to it. While the representation of the robot pose depends on the degrees of freedom of motion, the representation of the map depends on a multitude of factors including the available

sensors, computational resources, intended high level task, and required precision. Many possible representations have been proposed.

For visual-SLAM, the simplest representation of the map is a collection of 3D points which correspond to salient image feature points. This representation is sparse and efficient to compute and update. Point based methods have been successfully used to map city-scale environments. However, this sparsity comes at a price: points-based maps lack semantic information and are not useful for high level task such as grasping and manipulation.

A more computationally demanding but meaningful approach is to consider all pixels in the image that have significant gradient and to use them as candidates for points in the map. This leads to so called "semi-dense" mapping, where the structure of the environment is more interpretable and can be used for some high level tasks. Semi-dense system have been shown to work at a building-scale environment successfully. At the extreme end of the density scale lies a fully dense map representation where the environment is represented as a collection of surfaces. This contains the most information for tasks such as manipulation, while being the most expensive to compute. Dense mapping is still limited to room-scale environments. The map representations for these semi-dense or dense methods, such as [4, 3, 20, 21, 19], remain equivalent to a set of points without carrying any additional semantic information.

A map based on sparse points carries no semantic information. Geometric entities such as lines and planes, on the other hand, carry semantic information such as edges or surfaces present in the scene. These primitives are useful for map representation as they provide informative constraints on the scene geometry as well as the robot's pose, while still being sparse.

Man-made environments contain many planar surfaces and it is easy to infer relationships between them using a Manhattan world assumption: the planes are either parallel or perpendicular to each other. Detecting these planes and using them in addition to a point-based representation provides additional constraints for the points that lie on one of the planes and in-turn inform the camera pose. These planar constraints are also useful in areas where features can not be found because of uniform texture. The main challenge of considering additional geometric entities in the SLAM problem is choosing an effective mathematical representation for state variables and observation/constraints between them that is suitable to the SLAM formulation and also efficient in terms of memory and time that is required for solving the consequent optimization problem. Including planes in a SLAM map has been explored before

[24, 14], however unlike those previous works which only use planes and ignore points, we target to exploit both, by incorporating planes along with sparse points, more precisely, by proposing mathematical representations of landmarks and constraints that are amenable for graph-based formulation of the real-time sparse point-plane SLAM.

In this chapter, we explore the effect of introducing planes as independent landmarks, with our proposed representation, into the map and the consequent geometric constraints that these landmarks impose between the sparse points and their underlying planes, and also the geometric constraints between planes that are imposed to realize Manhattan world assumption in 3D map reconstruction. Planes are extracted from input imagery[1] and integrated into a state-of-the-art sparse mapping system to further constrain a sparse environment representation and enrich the map with more semantically meaningful landmarks. The extracted planes are used for both localization and mapping, in other words, the newly introduced planes play a role both in tracking of the camera while the map representation is considered fixed, and also in the Bundle Adjustment (BA) problem to refine the estimation of the state variables of the local map. Unlike most of the visual SLAM systems [14, 24, 16, 9], we exhaustively evaluate our proposed system with the additional landmarks and observation/constraint factors, as an ablation study, and comparative study against state-of-the-art systems, in publicly available datasets such as RGB-D TUM [25] and NYU-v2 [18] benchmarks and show that this greatly reduces the error in the estimated camera trajectory without incurring great computational cost either in time and memory. A point and plane based representation is richer and more informative than a sparse point representation and can be used for reasoning about semantics and higher level information encoded in the scene such as object affordances and supporting surfaces.

The rest of the chapter is organized as follows. Section 3.2 presents a short overview of methods that have already been developed to address the SLAM problem based on a combination of points and other geometric primitives. Then we present our SLAM system in *three* separate sections: first we elaborate the mathematical representations that we utilize to estimate the state variables (variable nodes of our factor graph) in Section 3.3 and we propose our representation for plane landmarks in this section. Then mathematical representations for our proposed observation/constraint factors are explained in Section 3.4. Sections 3.3 and 3.4 are comprising the back-end of our system. The last section that explains the front-end of our SLAM is Section 3.5. In

[1] In this chapter we utilize the RGB-D input modality, however towards a pure monocular SLAM, we propose a monocular plane detection pipeline in Chapter 5.

3.5 we build our factor graph from the sensory information (RGB-D input stream) by detecting, initializing, and tracking the landmarks and their observations from our sensor models and then we hypothesis regarding the constraint factors between different landmarks in the map. The evaluations and experiments of the SLAM system are presented in Section 3.6 and finally the conclusion is presented in Section 3.7.

3.2 Related Work

SLAM is a well studied problem in mobile robotics and many different solutions have been proposed for solving it. The most recent of these is the graph-based approach that formulates SLAM as a non-linear least squares problem [11]. SLAM with cameras has also seen advancement in theory and good implementations that have lead to many real-time systems from sparse ([17],[4]) to semi-dense ([3], [5]) to fully dense ([20], [19], [21]).

We are interested in sparse map representations incorporate geometric primitives such as lines and/or planes, without focusing on the modality of the sensor used (monocular vs. RGB-D). Recently, there has been a lot of interest in extending the capability of a point-based representation by either applying the same techniques to other geometric primitives or fusing points with lines or planes to get better accuracy. Several methods have explored replacing points with lines [16], [7]. However, lines present especial difficulty because of the lack of a good mathematical representation that is amenable to the least-squares framework. Some works have explored the possibility of using lines and points in the same framework [22], [9] and have been more successful.

Recently, [14] proposed a representation for infinite planes that is amenable for use in a least-squares framework. Using this representation, they presented a method that works using only information of planes visible in the environment. Similarly, [28] use a monocular input to generate plane hypotheses using a Convolutional Neural Network (CNN) which is then refined over time using both the planes as well as points in the images. However unlike our proposed SLAM in this chapter, these methods only consider planes as independent landmarks and neglect points in the map. A method is proposed in [26] that fuses points and planes using an RGB-D sensor. In the latter works, they try to fuse the information of planar entities to increase the accuracy of depth inference for point landmarks and ignore planes as independent landmarks in the bundle adjustment of the SLAM. The evaluation of the previously proposed

plane-based SLAM systems has been limited to special-purpose scenarios and scenes, however we evaluate our framework exhaustively in generic public SLAM benchmarks.

3.3 Landmark Representations

As introduced in Chapter 2, we formulate our sparse SLAM problem as a factor graph optimization problem and use the benefits of the least squares estimation problems.

In this chapter, our problem is point-plane localization and mapping problem, hence we want to estimate the N camera poses with respect to the world reference frame, $\mathbf{T}_1^w, \mathbf{T}_2^w, ..., \mathbf{T}_N^w$, n 3D points in the map $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ and L infinite planes in the map $\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, ..., \boldsymbol{\pi}_L$ using the point and plane measurements in the camera frames. We represent the mentioned estimation problem as a factor graph shown in Fig. 3.1. In this bipartite graph, variable nodes of the graph, or *variables* if context permits, are corresponding to 3D points $\mathbf{x}_i$, 3D planes $\boldsymbol{\pi}_i$, and camera coordinates frame $\mathbf{T}_c^w$, all represented with respect to the world global frame. Factor nodes, referred as *factors*, of the graph consist of measurements of the 3D points and planes in the camera frame and also the constraints between 3D planes and their corresponding inlier points. The objective function of our optimization problem consists of the multiplication of these factors of the joint probability distribution over variables given observations.

In the rest of this section we explain the appropriate mathematical representation for each of these variable nodes that is amenable to the least squares optimization problem and are used in our system. In the next section 3.4 we describe the above mentioned observations/constraints factors in detail. In section 3.5 we explain how to build this factor graph and initialize the variable nodes of the graph from input sensory information.

3.3.1 Point Landmarks

Due to the projective nature of the camera observations, the most common representation of 3D point variable $\mathbf{x}$ of the factor graph is by its homogeneous coordinates $\mathbf{x} = (x, y, z, w)^\top$ as a point in the 3D projective space $\mathbb{P}^3$. However without loss of generality, by normalizing the last coordinate we assume $\mathbf{x} = (x, y, z, 1)^\top$ as the representation for points in 3D space. By using this representation, point variables are updated in the graph during optimization process based on the gradient of the loss functions (associated connected factors) in the underlying $\mathbb{R}^3$ space. Note that in this

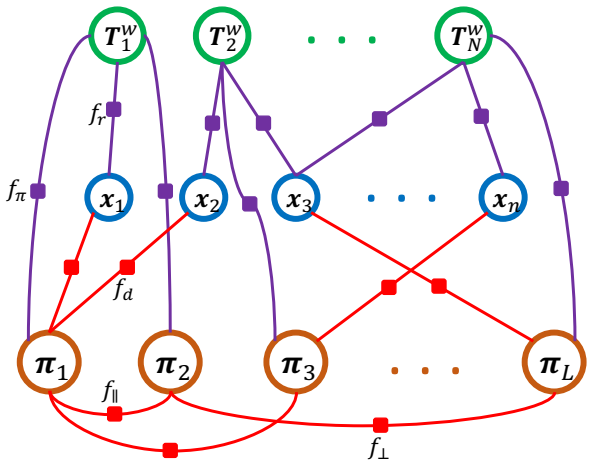


Fig. 3.1 Our sparse point-plane SLAM factor graph. The variable nodes include camera poses $\mathbf{T}_c^w$, points $\mathbf{x}_i$, and plane landmarks $\boldsymbol{\pi}_i$. The factor nodes are point and plane observations (observation factor nodes) and also the constraint factors between points-planes and planes-planes (Manhattan assumption).

simple case, $\mathbb{R}^3$ is a Lie group with corresponding Lie algebra of $\mathbb{R}^3$ itself. The 3D point in the world and its projective observation is illustrated in Fig. 3.2.

3.3.2 Camera Poses

The camera pose in the factor graph is represented as an isometric 3D Euclidean transformation, $\mathbf{T}_c^w$, between a global coordinates frame and local camera coordinates frame. This *proper* isometric Euclidean transformation, $\mathbf{T}_c^w$, is a bijective mapping from $\mathbb{R}^3$ to itself that preserves distances between any pairs of points (or equivalently *inner product* of vectors in the *vector space*) and also preserves the orientation of the vector space.

The set of proper Euclidean isometrics (rigid motions), denoted as $\mathbf{SE}(3)$, form a group along with the group operation of composition of mappings, that we denote them as *poses*. There are different equivalent ways for parameterizing 6D poses, such as:

- 3D translation and 3D rotation defined by Euler angles
- 3D translation and 4D rotation defined by unit quaternions
- 4×4 transformation matrices

Since we model 3D space as 3D projective space $\mathbb{P}^3$, the latter parametrization of this group is more convenient for applying projective operations. Hence we parametrize

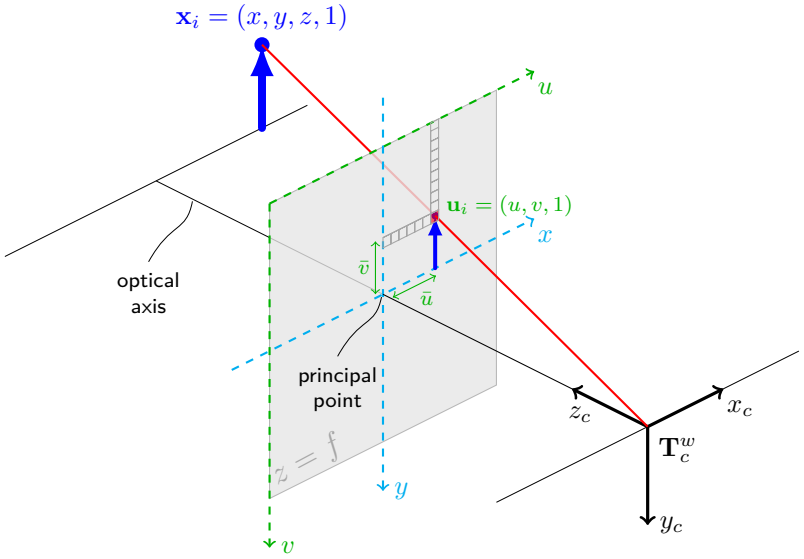


Fig. 3.2 The representation of a 3D point $\mathbf{x}_i$ in the camera coordinates frame $\mathbf{T}_c^w$ and its projective measurement on the image plane $\mathbf{u}_i$, both are represented in 3D and 2D projective spaces ($\mathbb{P}^2$ and $\mathbb{P}^3$), respectively.

this group as a subgroup of projective transformations of $\mathbb{P}^3$ (the set of all invertible 4×4 matrices $\mathbf{GL}(4, \mathbb{R})$) that satisfies the following structure:

$$\mathbf{T}_c^w = \left[\begin{array}{c|c} \mathbf{R}_{3\times3} & \mathbf{t}_{3\times1} \\ \hline \mathbf{0}_{1\times3}^\top & 1 \end{array}\right] \tag{3.1}$$

where $\mathbf{R} \in \mathbf{SO}(3)$ is a proper rotation matrix (*orthogonal matrix* with determinant $+1$), $\mathbf{t} = (t_x, t_y, t_z)^\top \in \mathbb{R}^3$ is a translation vector and group operation is the standard matrix product.

In our notation, $\mathbf{T}_c^w$ is the rigid transformation between global *world* coordinates frame (w) and local *camera* coordinates frame (c) that has the following effect on the coordinates of the points:

$$\begin{gathered} \mathbf{x}_c = \mathbf{T}_c^w \mathbf{x}_w \\ \begin{pmatrix} x_c \\ y_c \\ z_c \\ 1 \end{pmatrix} = \left[\begin{array}{ccc|c} & \mathbf{R} & & \begin{matrix} t_x \\ t_y \\ t_z \end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \begin{pmatrix} x_w \\ y_w \\ z_w \\ 1 \end{pmatrix} \end{gathered} \tag{3.2}$$

where $\mathbf{x}_c$ and $\mathbf{x}_w$ are the homogeneous coordinates of the point in local camera frame and global world frame, respectively.

Note that $\mathbf{SE}(3)$ group is a 6D smooth manifold (*Lie Group*) embedded in the 16 dimensional manifold $\mathbf{GL}(4,\mathbb{R})$ and has the structure of semi-direct product of groups $\mathbf{SO}(3)$ and $\mathbb{R}^3$ ($\mathbf{SO}(3) \ltimes \mathbb{R}^3$).

Camera pose variables are updated in the factor graph during optimization process based on the gradient of the loss functions (associated connected factors) in the underlying associated 6D Lie algebra $\mathfrak{se}(3)$, whose bases are corresponding to either infinitesimal rotations or infinitesimal translations along each axis, found by "linearization" of the manifold $\mathbf{SE}(3)$.

3.3.3 Plane Landmarks

We represent *planes* as 2 dimensional linear subspaces of 3D projective space $\mathbb{P}^3$. In other words, a homogeneous vector $\boldsymbol{\pi} = (a, b, c, d)^\top \in \mathbb{P}^3$ represents a plane in projective space and a homogeneous point $\mathbf{x} = (x, y, z, 1)^\top \in \mathbb{P}^3$ lies on that plane if and only if

$$\boldsymbol{\pi}^\top \mathbf{x} = 0 \tag{3.3}$$

or the more familiar plane equation of

$$ax + by + cz + d = 0 \tag{3.4}$$

where we represent the normal vector of the plane by $\mathbf{n} = (a, b, c)^\top$ and find its orthogonal distance from the origin by

$$\frac{|d|}{\|\mathbf{n}\|_2} \tag{3.5}$$

where $\|.\|_2$ is the Euclidean norm defined in the corresponding vector space.

Combining Equations 3.2 and 3.3 yields that the homogeneous plane representation in the global world coordinates frame, $\boldsymbol{\pi}_w$ is transformed to the local camera frame, $\boldsymbol{\pi}_c$, by the *inverse transpose* of the camera pose transformation, i.e.

$$\boldsymbol{\pi}_c = (\mathbf{T}_c^w)^{-\top} \boldsymbol{\pi}_w. \tag{3.6}$$

If we consider the homogeneous representation for plane landmarks in the factor graph optimization, since this representation is over-parametrized (there are only three degrees of freedom in a plane: one for orthogonal distance of plane from origin and two for its surface normal orientation), the Hessian (or *information matrix*) of the

problem will be rank-deficient. In incremental solving of non-linear least squares problem, in each step after "linearizing" the non-linear objective function a linear system should be solved by inverting the Hessian matrix. Rank-deficiency of Hessian matrix cause problems during optimization. The immediate solution to this rank-deficiency problem is using regularization with the cost of slow convergence that is not suitable for real-time applications.

Uncritical minimal (three degrees of freedom) parametrization of planes causes singularities and instabilities for numerical optimization, such as the parametrization of using two parameters for orientation of the surface normal and one parameter for the distance of the plane from origin: it is a minimal representation, however there are singularities similar to Gimbal lock problem of Euler angles and also the difference in the scale of the third parameter (distance) and the first two parameters (angles), due to their different nature, makes the numerical optimization instable. Hence we are looking for more *homogeneous* parametrization.

The immediate intuitive minimal representation driven from homogeneous vectors is normalizing them and representing planes by normalized homogeneous vectors with unit norm. The set of all unit homogeneous vectors defines a 3-sphere $\mathcal{S}^3$ in $\mathbb{R}^4$

$$\mathcal{S}^3 = \{(a, b, c, d) \in \mathbb{R}^4 \mid a^2 + b^2 + c^2 + d^2 = 1\}. \tag{3.7}$$

Note that $\mathcal{S}^3$ is a double cover for the set of all 3D planes, since antipodal points $\boldsymbol{\pi} \in \mathcal{S}^3$ and $-\boldsymbol{\pi} \in \mathcal{S}^3$ both are representing the same plane.

To solve the optimization problem by this minimal representation for planes, we should restrict the plane vector to remain on the $\mathcal{S}^3$ during the optimization steps. One method to solve this constrained optimization problem is adding Lagrange multipliers to enforce the constraint. The drawbacks of this method are: increasing the size of the optimization space and computational cost while complicating the objective function. As shown in [14], the better solution is using minimal representation to update plane landmarks during optimization on the manifold of planes. Hence we represent planes by finding a proper mapping from normalized homogeneous vectors to a smooth manifold on which we perform the optimization updates.

As $\mathcal{S}^3$ is a double cover for our plane representation space and also for *rotations group* in 3D, $\mathbf{SO}(3)$, we propose to identify our plane representation with a *rotation matrix* by using a two-to-one and onto homomorphism from $\mathcal{S}^3$ onto $\mathbf{SO}(3)$. In contrast to [14] that uses quaternions to represent planes, by our representation we

directly benefit from the straight-forward Lie algebra of $\mathfrak{so}(3)$ to update planes during optimization.

If we represent our plane landmark as $\boldsymbol{\pi} = (a, b, c, d)^\top \in \mathcal{S}^3$ then there is a 2:1 homomorphism, [8], from $\mathcal{S}^3$ to $\mathbf{SO}(3)$ that maps $\boldsymbol{\pi}$ to the following rotation matrix

$$\mathbf{R}_{\boldsymbol{\pi}} = \begin{bmatrix} 1-2b^2-2c^2 & 2ab-2cd & 2ac+2bd \\ 2ab+2cd & 1-2a^2-2c^2 & 2bc-2ad \\ 2ac-2bd & 2bc+2ad & 1-2a^2-2b^2 \end{bmatrix} \in \mathbf{SO}(3) \tag{3.8}$$

which represents a rotation ($\mathbf{R}_{\boldsymbol{\pi}}\mathbf{R}_{\boldsymbol{\pi}}^\top = \mathbf{R}_{\boldsymbol{\pi}}^\top\mathbf{R}_{\boldsymbol{\pi}} = \mathbf{I}$) around the vector $(a, b, c)^\top$ by an angle φ, where $\cos\frac{\varphi}{2} = d$.

Hence by the mentioned mapping instead of solving a constrained optimization problem on $\mathcal{S}^3$, we can update planes during optimization on $\mathfrak{so}(3)$ by using the *exponential* and its inverse, *logarithm*, maps of the $\mathbf{SO}(3)$ manifold. More precisely if we denote the increment $\mathbf{v} = (v_1, v_2, v_3)^\top$ in the tangent space of $\mathbf{SO}(3)$ at the identity or equivalently its corresponding skew-symmetric matrix in its Lie algebra by

$$[\mathbf{v}]_\times = \begin{bmatrix} 0 & -v_3 & v_2 \\ v_3 & 0 & -v_1 \\ -v_2 & v_1 & 0 \end{bmatrix} \in \mathfrak{so}(3) \tag{3.9}$$

then we update the current estimation of the plane node $\mathbf{R}_{\boldsymbol{\pi}_t}$ (or equivalently its rotation matrix) such as following

$$\mathbf{R}_{\boldsymbol{\pi}_{t+1}} = \exp\left([\mathbf{v}]_\times\right)\mathbf{R}_{\boldsymbol{\pi}_t} \tag{3.10}$$

where $\exp([\mathbf{v}]_\times)$ is the exponential map that maps one element of the Lie algebra on the manifold, such as following, also known as Rodrigues' formula [1],

$$\exp([\mathbf{v}]_\times) = \mathbf{I} + (\frac{\sin\varphi}{\varphi})[\mathbf{v}]_\times + (\frac{1-\cos\varphi}{\varphi^2})[\mathbf{v}]_\times^2 \tag{3.11}$$

where $\varphi = \|\mathbf{v}\|_2$. Another interpretation of tangent vectors $\mathbf{v}$ (projections of rotation matrices $\mathbf{R}_{\boldsymbol{\pi}}$ to the tangent space at the identity $\mathbf{I}$) is that they are local *axis-angle* representations for rotation matrices (in our case 3D plane landmarks).

Eventually, for going backward to find the new update of 3D homogeneous plane $\boldsymbol{\pi}_{t+1}$ from its updated rotation matrix representation $\mathbf{R}_{\boldsymbol{\pi}_{t+1}}$, we need the logarithm map of $\mathbf{SO}(3)$ that can be calculated by first determining $\cos(\|\mathbf{v}\|_2) = \frac{1}{2}(tr(\mathbf{R}_{\boldsymbol{\pi}_{t+1}}) - 1)$

and then computing $\mathbf{v}$ from symmetric differences by using 3.11. Then updated homogeneous plane representation is determined as the following [10],

$$\boldsymbol{\pi}_{t+1} = (\sin(\frac{\|\mathbf{v}\|_2}{2})\hat{\mathbf{v}}^\top,\ \cos(\frac{\|\mathbf{v}\|_2}{2}))^\top \tag{3.12}$$

where $\hat{\mathbf{v}} = \mathbf{v}/\|\mathbf{v}\|_2$ is the unit norm vector.

3.4 Observations and Constraints Factors

After introducing the representation of different variable nodes in the factor graph of our SLAM system in Section 3.3, in this section we want to describe in detail the proposed factor nodes of our sparse SLAM system. The factor nodes are representatives of different *observations* (of landmarks from the camera pose) and also *constraints* (between landmarks).

3.4.1 3D Point Observation Factors

In a traditional point-based SLAM system, observation factors exist between points and the camera that seek to minimize the geometric re-projection error. We also adopt these observation factors and introduce them in our system as

$$f_r(\mathbf{x}_w, \mathbf{T}_c^w) = \parallel \mathbf{u}_c - \Pi(\mathbf{x}_w, \mathbf{T}_c^w) \parallel^2_{\boldsymbol{\Sigma}_r} \tag{3.13}$$

where $\mathbf{u}_c$ is the observed pixel location (in the image plane) of the 3D point $\mathbf{x}_w$ represented in the global world frame, as illustrated in Fig. 3.2, $\|\mathbf{u}\|_{\boldsymbol{\Sigma}} = (\mathbf{u}^\top \boldsymbol{\Sigma}^{-1}\mathbf{u})^{1/2}$ notation is the Mahalanobis norm of vector $\mathbf{u}$ with covariance matrix $\boldsymbol{\Sigma}$ (in this factor node a 2×2 matrix), and $\Pi(.)$ is a function that projects a world 3D point into the image plane such as following

$$\Pi(\mathbf{x}_w, \mathbf{T}_c^w) = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{T}_c^w \mathbf{x}_w \tag{3.14}$$

where $\mathbf{K}$ is a 3×3 *camera calibration matrix* and $\mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{T}_c^w$ is called *camera projection matrix*.

Note that in this factor and all the other factors introduced in this thesis, to mitigate the influence of outlier data, we use robust error functions in the factors, such

as Huber loss [13] which is defined such as following

$$\rho_\delta(r) = \begin{cases} \frac{1}{2}r^2 & \text{for } |r| \leq \delta \\ \delta(|r| - \frac{1}{2}\delta) & \text{otherwise.} \end{cases} \tag{3.15}$$

where r is the residual and δ is the parameter of the robust kernel. For the sake of brevity, as context permits, we avoid to repeat the Huber robust kernel in every factor definition.

3.4.2 3D Plane Observation Factor

Due to the proposed representation of 3D plane landmarks in 3.3.3, the problem of introducing an observation factor for plane landmarks is reduced to a problem of choosing an appropriate *metric* or *distance* for the space of rotation matrices.

There are different distance measures that we can introduce between two rotation transformations, however we want to exploit the geometrical characteristics of the space of rotation transformations, so we choose the *geodesic* distance or the shortest distance between two points along the manifold, which captures the concept of the *closeness* between two elements of the manifold more precisely when compared to the Euclidean metric.

Therefore, the plane observation factor between a camera node variable $\mathbf{T}_c^w$ and a 3D plane landmark $\boldsymbol{\pi}$, that is observed as $\boldsymbol{\pi}_{obs}$ in the local camera coordinates frame, is imposed as following

$$f_\pi(\boldsymbol{\pi}, \mathbf{T}_c^w) = \| \, d(\mathbf{T}_c^{w-\top}\boldsymbol{\pi}, \boldsymbol{\pi}_{obs}) \, \|^2_{\boldsymbol{\Sigma}_\pi} \tag{3.16}$$

where $\mathbf{T}_c^{w-\top}\boldsymbol{\pi}$ is the transformed plane to the camera coordinates frame, refer to 3.6, $\boldsymbol{\Sigma}_\pi$ is the covariance matrix of the Mahalanobis norm, and d is the difference of the corresponding rotation matrices of the planes in the tangent space at the identity of $\mathbf{SO}(3)$, explained in 3.3.3, such as following

$$d(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \text{logm}(\mathbf{R}_{\boldsymbol{\pi}_2}^\top \mathbf{R}_{\boldsymbol{\pi}_1}) \tag{3.17}$$

where $\mathbf{R}_{\boldsymbol{\pi}}$ is the associated rotation matrix of the corresponding plane, and *logm* is the logarithm map of the rotation matrices explained in 3.3.3.

Note that f_π is formulating the closeness of the plane landmarks based on the geodesic distance rather than Euclidean distance that is also a good distance only locally (near identity) whereas the geodesic measure is valid globally.

3.4.3 Point-Plane Constraint Factor

If we believe that a point actually lies on a specific plane, it makes sense to impose a constraint between the point and the relevant plane landmark. To do so we introduce a factor node between point $\mathbf{x}$ and plane $\boldsymbol{\pi}$ variable nodes that models an orthogonal Euclidean distance of the point and plane such as following

$$f_d(\mathbf{x}, \boldsymbol{\pi}) = \| \mathbf{n}^\top (\mathbf{x} - \mathbf{x}_o) \|^2_{\sigma_d} \tag{3.18}$$

where $\mathbf{n}$ is the unit normal vector of the plane $\boldsymbol{\pi}$, σ_d is the variance of the Mahalanobis norm, and $\mathbf{x}_o$ is an arbitrary point on the plane that only needs to satisfy plane Equation 3.3 and in our system is considered as the closest point of the plane to the origin.

Note that the association between inlier points and planes have been established in the front-end during plane detection and segmentation that is explained in 3.5.2.

3.4.4 Plane-Plane Constraint Factor (Manhattan World Assumption)

In almost all of the man-made environments, particularly indoors, there are dominant underlying structures that are present in the scene and influence strongly all of the spatial and semantic relationships among geometric entities and semantic objects. These structures can be introduced as prior knowledge to the 3D modeling process of the scene.

One of the most useful and common underlying structures (especially indoors) are those associated with a Manhattan world in which planes are mostly mutually orthogonal or parallel This assumption acts as a prior knowledge about the scene based on the evidence.

We introduce this geometric assumption to our SLAM system and demonstrate the efficacy of such an assumption in 3.6 for improving the estimation of camera trajectory and implicitly accuracy of 3D mapping of the scene, particularly in textureless environments.

Imposing the mentioned constraint on relative plane orientation is only a matter of introducing a constraint factor on the plane surface normals. The constraint factor for parallel planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ is considered as following

$$f_{\parallel}(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \parallel |\mathbf{n}_1^\top \mathbf{n}_2| - 1 \parallel^2_{\sigma_{par}} \tag{3.19}$$

and the constraint factor for perpendicular planes is implemented as

$$f_{\perp}(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \parallel \mathbf{n}_1^\top \mathbf{n}_2 \parallel^2_{\sigma_{per}} \tag{3.20}$$

where $\mathbf{n}_1$ and $\mathbf{n}_2$ are the unit normal vectors of the planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ respectively and σ_{par} and σ_{per} are the variances of the Mahalanobis norm of parallel and perpendicular constraints, respectively.

Note that we actually apply this assumption by enforcing a constraint for the inner product of the unit normal vectors of the planes participating in the factor.

This Manhattan assumption is implemented as a *soft* constraint in our system, 3.5.2, means that we impose the constraint with high uncertainty without enforcing the planes to be exactly perpendicular or parallel and planes could end up with situations close to perpendicularity or parallelism according with the evidence.

3.5 Front-end of our Point-Plane SLAM

As discussed in Chapter 2 modern SLAM algorithms can be divided into two parts: the front-end, that deals with sensory information to convert them into constraints, and a back-end that can optimize over these constraints in a least-squares framework. In other words, so far we described in sections 3.3 and 3.4 the appropriate representations of landmarks consisting the map and the observations and constraints that form our SLAM problem as a factor graph optimization problem. Now in this section we explain in detail the front-end of our SLAM system to extract and build the factor graph from sensory input.

The key issues addressed in this section are how to detect 3D landmarks from input and how to track and match observations to landmarks (*data association* problem) and initialize the new ones when it is necessary and add them to the factor graph. Also we introduce the mechanisms that we use to detect and impose the soft constraints between existing landmarks in the map.

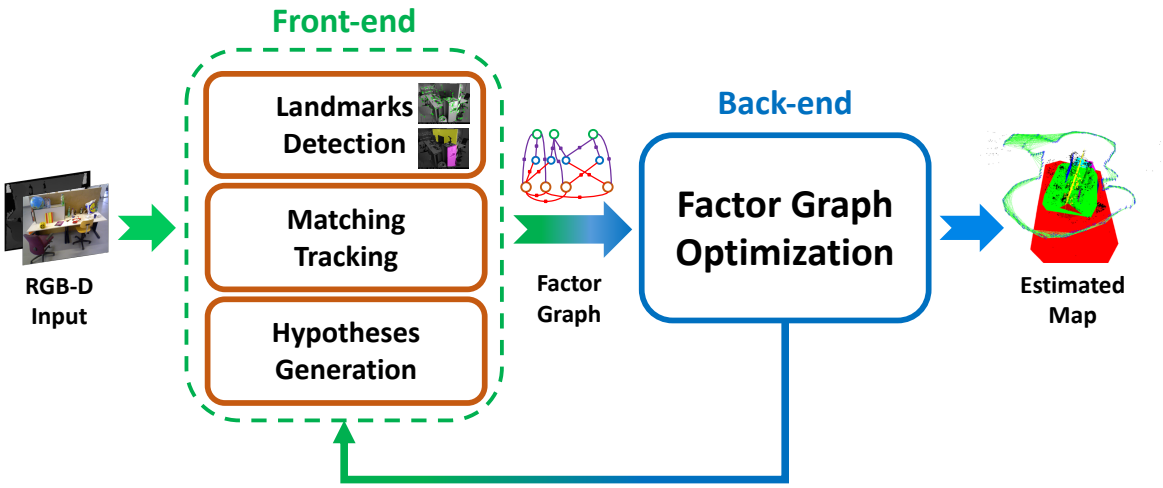


Fig. 3.3 The back-end and front-end of our sparse point-plane SLAM system.

In order to take advantage of the recent success of graph-based approaches to SLAM, we build our system on top of the state-of-the-art feature-based sparse SLAM framework, ORB-SLAM2 [17]. This allows us to benefit from the sparsity of the map and at the same time incorporate more semantically meaningful geometric primitives such as planes in the map. Therefore we are loyal to our initial goals to build an almost real-time, scalable, spares SLAM consisting of higher level entities.

In terms of the modality of the input, for the scope of this chapter, we use RGB-D input to detect and extract more reliable 3D planes from the RGB and depth maps in order to focus our study on the effect of introducing planes and corresponding constraints to a sparse SLAM system. However this is not a fundamental bottleneck, since as we see in this section we only use the depth information to initialize the position of 3D point features and also detect and initialize 3D planes in the map. There is no additional fusion of depth information over frames happening in the front-end of our SLAM and further refinement of the landmarks takes place within the bundle adjustment in the back-end.

In the rest of this section, first we describe our point feature extraction and tracking and then our plane detection and segmentation. Then we discuss in detail the data association problem related to plane landmarks. The back-end and front-end of our sparse point-plane SLAM is demonstrated in Fig 3.3.

3.5.1 Point Feature Extraction and Tracking

For point feature selection and matching we rely on the front-end of the state-of-the-art ORB-SLAM2 [17] to extract and match ORB features in a coarse-to-fine pyramid. Using the front-end of the ORB-SLAM2 for point features, makes our comparison in the ablation study in 3.6 fair as we incrementally add more landmarks and consider higher level constraints.

Points are tracked after extracting and matching ORB descriptors in each frame, which is much faster to compute than SIFT and SURF features. For initializing the 3D position of the detected feature point, the depth of the key-point is only used from the first frame which observes the key-point and further improvements over the estimate of the 3D point occur in the bundle adjustment.

3.5.2 3D Plane Detection and Tracking

Most plane fitting models for RGB-D data use RANSAC which is extremely slow for the purpose of building a near real-time online SLAM framework consisting of points and planes in the map. Keeping near real-time performance in mind, we are interested in detecting infinite planes and estimating its parameters from RGB-D data along with the finite boundary segmentation of the planes in each frame to use this information for associating point landmarks to the detected planes and even match planes across frames. Here we describe our approach which overcomes these limitations.

3.5.2.1 Plane Segmentation and Parameter Estimation

Inspired by [27], we efficiently cluster and segment multiple planes from point-clouds generated from RGB-D data.

After constructing the point-cloud from a pair of RGB image and depth map, first we calculate surface normals for valid depth measurements by smoothing in its neighborhood based on the proposed method of [12]. We smoothen the depth data before calculating the surface normals in order to yield a less noisy normals. The method in [12] uses *integral images* to perform border and depth dependent smoothing, since they can be calculated efficiently by accessing only four data elements of the point-cloud corresponding to the four corners of the rectangular smoothing region. Then we estimate the surface normal unit vector $\mathbf{n}$ at image location (u, v) by computing the cross product of the 3D vectors between its neighbors and then normalizing it such as following

$$\mathbf{n} = \frac{\mathbf{v}_{ud} \times \mathbf{v}_{lr}}{\| \mathbf{v}_{ud} \times \mathbf{v}_{lr} \|_2} \tag{3.21}$$

where $\mathbf{v}_{ud}$ is the 3D vector between the upper and lower neighbors and $\mathbf{v}_{lr}$ is the vector between the left and right neighbors of $\mathbf{u}$ in the point-cloud, as shown in Fig 3.4.

After estimating the unit surface normals, to segment multiple planes from the reconstructed point-cloud out of the input RGB-D stream, we cluster the cloud in two steps, as illustrated in Fig 3.4.

First we cluster the unit surface normals in the space of unit normal vectors to find the clusters corresponding to dominant planar orientations of the scene. In this step we also merge the clusters that satisfy the following criteria

$$\mathbf{n}_1^\top \mathbf{n}_2 < \theta_{thresh} \tag{3.22}$$

where $\mathbf{n}_i$ is the unit surface normal corresponding to the center of cluster and θ_{thresh} is the threshold on the angle between those two surface normals.

Then for each resulted cluster in unit normal space, we conduct another clustering in the orientation of that surface normal vector (unit surface normal associated with the center of the cluster) by computing a one dimensional voxel grid and merging the neighboring voxels. In this step we also merge the cluster bins that satisfy the following criteria for depth space

$$|d_1 - d_2| < d_{thresh} \tag{3.23}$$

where d_i is the perpendicular distance of the center of the cluster bin to the origin and d_{thresh} is the threshold on this distance for clusters. In other words, this threshold determines the resolution of the clustering of the planes and the measure on the closeness of the segmented planes.

The above mentioned two steps yield multi-plane segmentation of the reconstructed point-cloud per RGB-D input. By projecting the plane segmentations to the RGB frame, we find the finite boundary of the planes in the image. To estimate the plane parameters, we consider the unit surface normal $\mathbf{n}$ and perpendicular depth d of the center of the cluster bins to initialize the estimate the plane parameters $\boldsymbol{\pi} = (\mathbf{n}^\top, d)^\top$.

Sample results of the plane segmentation is shown in Fig 3.5 for some real and synthetic RGB-D inputs.

The plane segmentation and parameter initialization uses RGB-D data, and is the only part of our system (other than ORB feature depth initialization) which relies

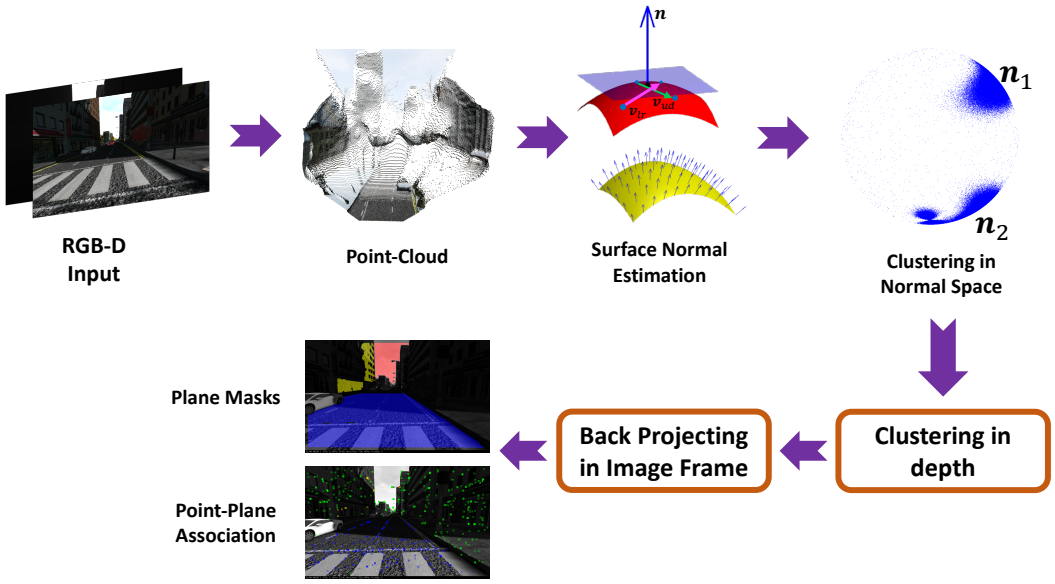


Fig. 3.4 To segment and estimate 3D plane parameters from RGB-D input images, first we build the corresponding point-cloud and then we estimate the surface normals as explained in 3.5.2.1. Then we cluster the normals in the normal space to find the dominant orientations in the scene and after merging the normal clusters, we perform the clustering in depth in the orientation of the resulted normals to complete the plane parameters and by back-projecting to image frame we find the segmentation of planes. Note that we determine the point-plane associations in this step.

on depth information. We overcome this depth-dependence in Chapter 5 towards proposing the monocular SLAM.

3.5.2.2 Data Association and Plane Matching

Planes detected in one frame need to be tracked into the subsequent frames and matched with planes detected in the new frames. This matching is an example of data association problem that enables the reconstruction of the factor graph. In other words, after detecting and initializing plane landmarks as variable nodes of the graph, we need to connect these nodes with the other nodes of the graph, such as 3D points, other planes in the map, and camera nodes (key-frames) via factor nodes. In this part, we determine these connecting edges of the factor graph for connecting planes to camera key-frames (by tracking planes in input RGB-D frames), connecting planes to the other planes (by creating hypotheses for Manhattan assumption based on the current estimate of the geometry of the planes), and at the end connecting planes to points (by assessing the inlier condition for planes).

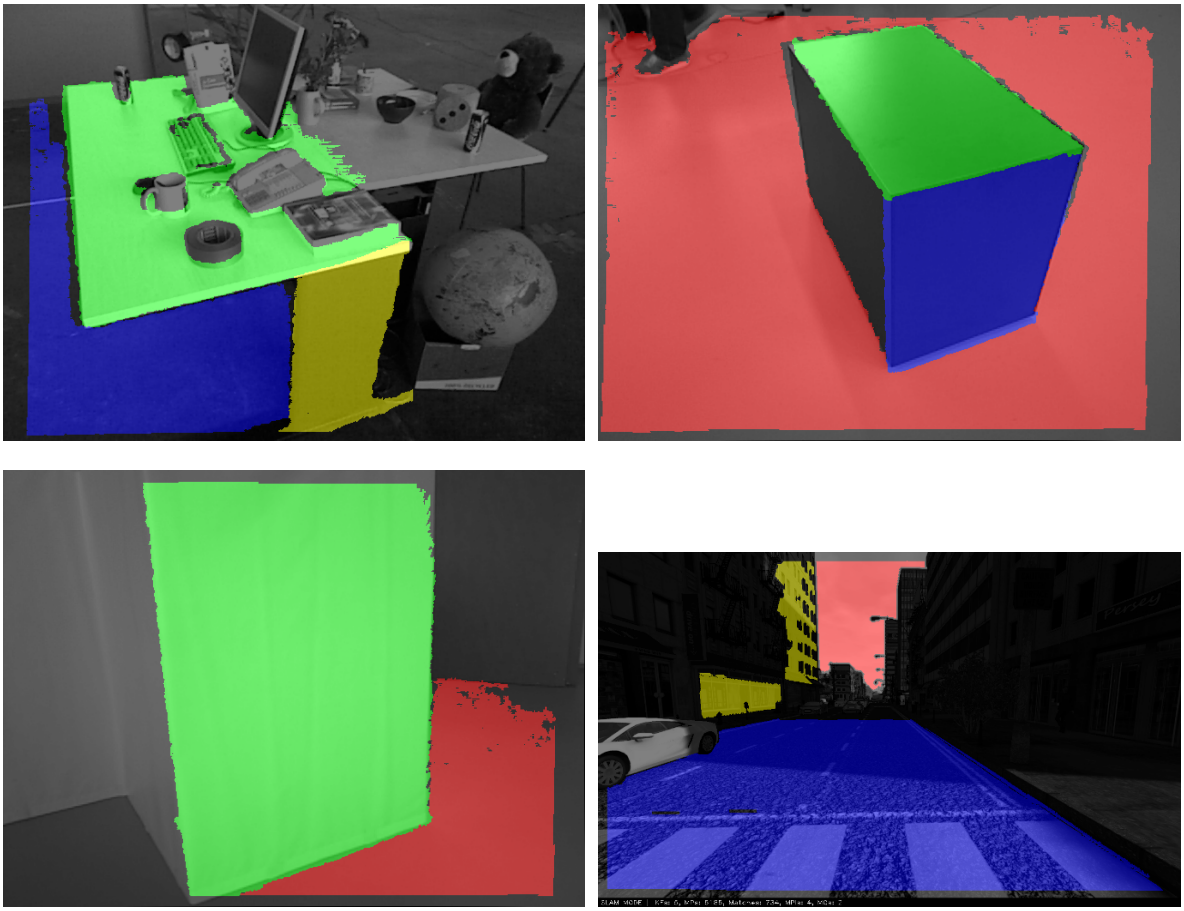


Fig. 3.5 Sample results of our plane segmentation for some sequences from RGB-D TUM benchmark [25] and synthetic SYNTHIA dataset [23].

Due to the inherent robust structure of plane landmarks, we use a relatively straightforward approach for data association related to them by considering the geometric information of the detected planes.

In our system, due to the small number of planes (compared with 3D points), detected planes are matched with the existing planes in the map based on the difference between their normals and the Euclidean perpendicular distance between them. Our first geometric criterion is the angle between normal vectors of the planes and after filtering out the candidate matches (from the existing planes in the map), the second measure is the Euclidean perpendicular distance between them. We choose the best match based on the second criteria after removing the outlier candidates by thresholding with the first measure.

For identifying the inlier points of the planes and associating them via the point-plane constraint factor, described in 3.4.3, we use the plane segmentation mask produced from the plane detection process and to ensure fewer outliers we check another geometric measure before imposing this constraint factor. We postulate the point-plane inlier condition if the 3D key-point lies in the segmented mask and its orthogonal distance from the infinite plane in the map is less than a threshold th_{PP}, which we set as a function of the distance of the points from the camera, because further points have greater uncertainty.

Since the number of planes detected by our system is sufficiently small, we consider all possible pairs in the map and introduce plane-plane constraint factors (Manhattan world assumption) explained in 3.4.4, with very little impact on overall computational cost of our system. At present we adopt the expedient of imposing a parallel constraint factor $f_{\parallel}$ if the smallest angle between the pair of planes (current estimates in the map) is less than a threshold th_M^{par}. We introduce a perpendicular constraint factor $f_{\perp}$ if the smallest angle between the pair of the current estimated planes is greater than a threshold th_M^{per} near to 90 deg. For our experiments we have used $th_M^{par} = 15\,\text{deg}$ and $th_M^{\perp} = 75\,\text{deg}$ empirically in our system.

Note that the mentioned thresholds, similar to other hard thresholds introduced in this thesis, are determined empirically by considering the overall performance of the system, in this case the measure of camera trajectory accuracy. However, this expedient can be replaced by training deep neural networks in future works in order to learn about the introduced constraints and conditions that need to be satisfied before imposing the constraint factors.

Manhattan constraints are imposed in a conservative manner as a *soft* constraint with a large uncertainty which is set empirically three times the measurement noise, and act as a prior on the relative orientation of the planes. Based on evidence gathered over image frames, they might end up being perpendicular or parallel but are not forced to be in that configuration if the data strongly favors an alternative interpretation.

Due to the relatively reliable depth information and accurate plane segmentation and parameter estimation, this simple proposed data association scheme is efficacious as demonstrated in experiments section 3.6.

3.5.2.3 Plane Landmarks in the Back-End Optimization

Since the number of infinite planes observable in each frame is much less than the number of interest points (key-points), integrating planes in the back-end optimization

of our system has negligible additional computational cost on top of optimization for point landmarks. As shown in section 3.6, introducing planes has significant effect on *a) camera localization with respect to a key-frame* and *b) local bundle adjustment over the landmarks in the map including points and planes after adding a new key-frame.*

In the camera localization stage, the observed landmarks are assumed to be fixed and the pose of the camera is optimized according to the factor graph Fig 3.1. Using planes in this stage with its subsequent additional factor nodes improves the accuracy of the camera tracking significantly which is shown in the section 3.6. When current frame has common field-of-view with previous frames less than a specific threshold we add a new *key-frame* to our system and this key-frame is considered as one camera variable node in the factor graph of our SLAM system. The idea of using key-frames instead of considering all the streaming input frames is now common in visual SLAM systems, since it creates smaller number of variable and factor nodes in the graph and improves the optimization time and real-time performance of visual SLAM systems. After adding new key-frame local bundle adjustment is carried out by optimizing a local factor graph consisting of the new key-frame and adjacent key-frames which have observed the common points and planes landmarks. This stage also has a significant part in global consistency of the reconstructed map and its effect can be seen in section 3.6.

Note that the global bundle adjustment, when a loop closure is detected, is done in our point-plane SLAM using points and planes as landmarks. We detect loops by the front-end of ORB-SLAM2 which uses bag of words [6] based on ORB features.

3.6 Experiments

To evaluate the performance of the proposed method, we use the publicly available and widely-used TUM RGB-D dataset [25]. We show both qualitative and quantitative results for the proposed point-plane SLAM by comparing it with the state-of-the-art sparse mapping system, ORB-SLAM2 [17] and also report the computational cost and runtime analysis in 3.6.3. Results are also reported for comparison against the recent plane-based Pop-up SLAM [28] in low-texture environments.

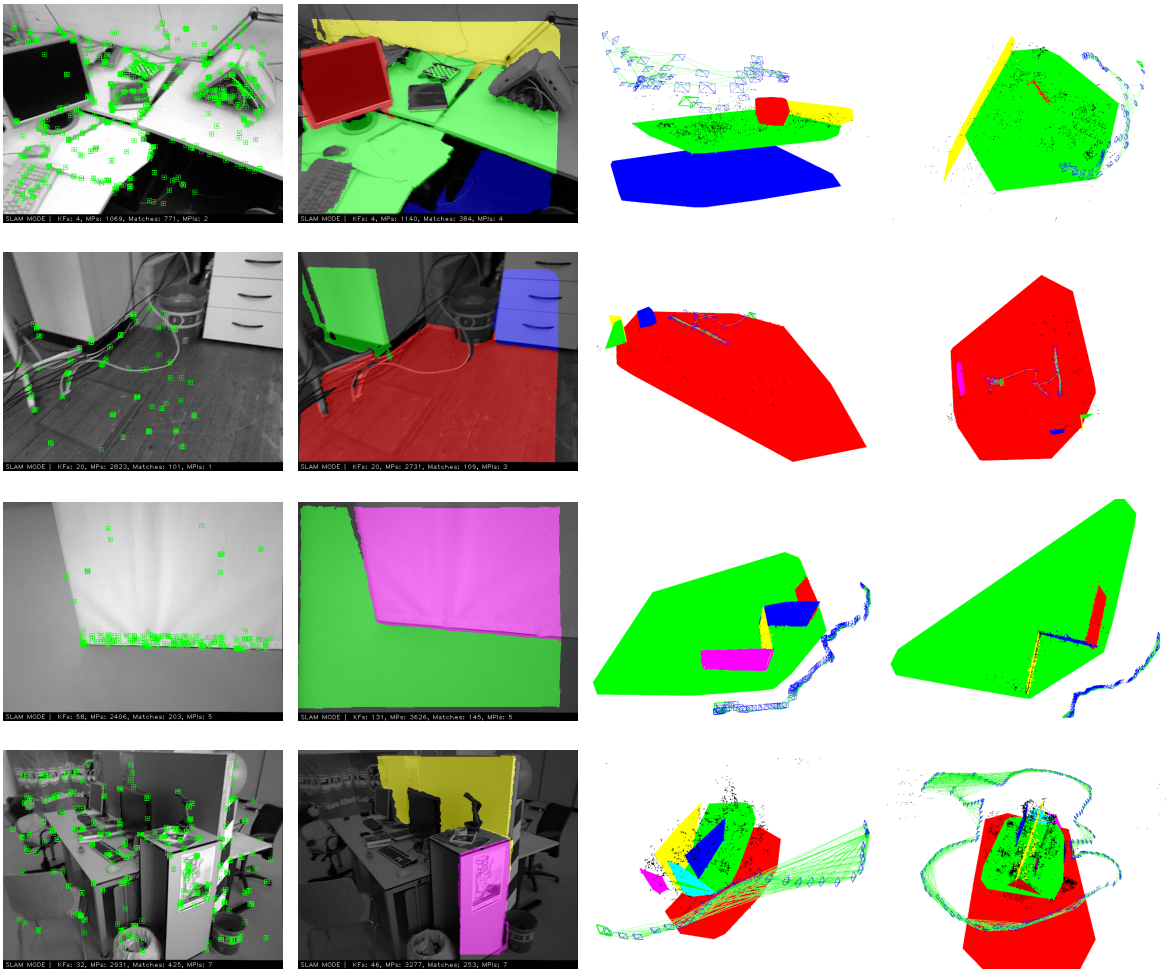
Sequences from the TUM dataset used in our experiments are introduced in Table 3.1 along with a short description of the type of scene observed in each sequence. These sequences have a wide range of conditions, from plane-rich scenes to scenes with little or no texture.

Table 3.1 RGB-D sequences from TUM benchmark [25] used in our experiments with their specific characteristics.

Dataset	Description
`fr1/xyz`	desk in an office environment
`fr1/desk`	sweeps over four desks in an office
`fr1/floor`	sweeps over wooden floor, planar regions
`fr2/xyz`	slow translational motion along the principal axes
`fr2/rpy`	slow rotational motion along the principal axes
`fr2/desk`	closing loop around the two desks in a typical office scene
`fr3/cabinet`	little texture, planar surfaces
`fr3/str_notex_near`	low-texture close, planar surfaces
`fr3/str_notex_far`	low-texture far, planar surfaces
`fr3/long_office`	large loop around office desks

3.6.1 Qualitative Results

Qualitative results for five different sequences have been shown in Fig 3.8 and Fig 3.6. The first column shows the input RGB frame along with the tracked ORB features. The detected planes along with their unique segmentation masks are shown in the second column and finally the last two columns depict the generated map consisting of points, planes, and key-frames from side and top view-points, respectively. In the first row of Fig 3.6, we can see that the generated map is consistent with the ground truth scene since we can see the desk as a unique green plane representing the four same-height office desk in the sequence. The same scenario happens for the second sequence in the second row of Fig 3.6 where we have the red plane reconstructed from the floor which is observable in all of the frames of this dataset. The mapping and tracking results for two challenging low-texture datasets have been illustrated in Fig 3.6 third row and Fig 3.8 for `fr3/str_notex_near` and `fr3/str_notex_far`, respectively. For these sequences, ORB-SLAM2 was unable to detect features in the environment with the default setting. The feature detection threshold had to be lowered to get some point features which also resulted in more outliers and inaccurate trajectories. However, our point-plane SLAM system improves the trajectory estimates significantly and yields richer maps for these sequences, particularly with Manhattan world assumption, with a negligible amount of additional memory load. For instance, the reconstructed map of our system for `fr3/str_notex_far` sequence needs only around 20 more parameters ($\sim$160 Bytes) to store additional landmarks in the memory and $\sim$500 more Bytes if storing the pair-wise constraints is desirable, compared to the sparse ORB-SLAM2.



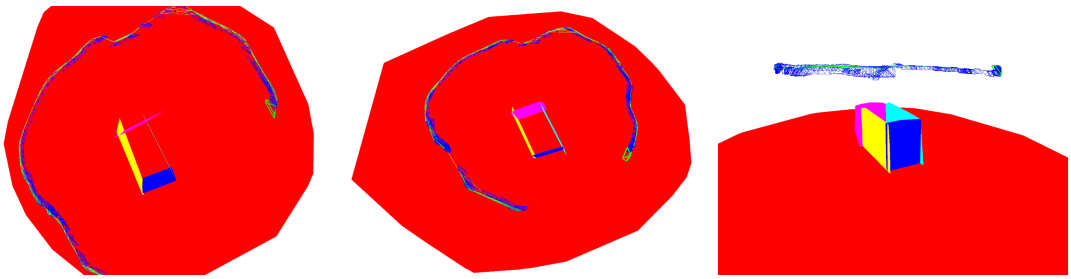
(a) RGB Frame with ORB Features (b) Detected Planes (c) Generated Map (Side) (d) Generated Map (Top)

Fig. 3.6 Qualitative Results for 4 different TUM RGB-D datasets.

However the map of our system is much richer and semantically more meaningful for further robotic tasks, such as planning or navigation, as shown in Fig 3.8.

Note that in all of the demonstrated reconstructed maps, all of the observation and constraint factors depicted in Fig 3.1 are effective in the back-end optimization of the system, consisting of camera-point, camera-plane, point-plane, and plane-plane (Manhattan assumption) factors.

To study the importance of introducing Manhattan assumption (plane-plane factor nodes) in the performance of our SLAM system, we design an experiment to reconstruct the cabinet in `fr3/cabinet` sequence with and without this assumption to assess the accuracy of the mapping of our system in reconstructing this cabinet. The sequence



(a) Generated Map (Before– Top) (b) Generated Map (After– Top) (c) Generated Map (After– Side)

Fig. 3.7 Qualitative comparison of the reconstructed planes representing *cabinet* before and after imposing Manhattan assumption between the planes in the RGB-D TUM `fr3/cabinet` sequence. Points and top-side plane of the cabinet have not been rendered for clarifying the difference in the map

contains a loop around a cabinet on the floor. All the faces of this cabinet are parallel or perpendicular to each other and the floor. Fig 3.7 demonstrates the difference in the quality of the reconstruction of the cabinet's sides without and with Manhattan assumption in columns (a) and (b), respectively.

Note that 3D point landmarks and the top-side of the cabinet have not been rendered to facilitate the comparison of the maps. As seen in Fig 3.7(a) that demonstrates the reconstruction of the cabinet without imposing the plane-plane constraint factors, sides of the cabinet are not congruent with the ground truth scene which all of the sides of the cabinet are parallel or perpendicular, for instance the yellow and light-blue sides are not parallel with each other. After imposing Manhattan assumption by considering the corresponding factors of the back-end factor graph, as seen in Fig 3.7 columns (b) and (c), the reconstructed sides of the cabinet are more congruent with the ground truth cuboid of the cabinet in terms of parallelism and perpendicularity. For example look at the difference in the configuration of the yellow and light-blue sides before and after adding the hypotheses.

In addition to the tracking accuracy in the frame-to-frame scenario, incorporating planes greatly impacts the trajectory estimate when there is a large loop closure: the system performs a global bundle adjustment with the points as well as plane landmarks. The produced map is more consistent as can be seen in the last row of Fig 3.6, which is from the `fr3/long_office` sequence. The green desk and red floor along with the monitors in this map are consistent with the ground truth scene.

3.6.2 Quantitative Comparison

To evaluate the accuracy of the state estimation (camera trajectory) and implicitly the consistency of map reconstruction, we compare our proposed system with the state-of-the-art point-based ORB-SLAM2. As it is common in evaluating visual SLAM systems, comparisons are reported in terms of the following metrics:

- Absolute Trajectory Error (ATE)
- Relative Translational Error (RTE)
- Relative Rotational Error (RRE)

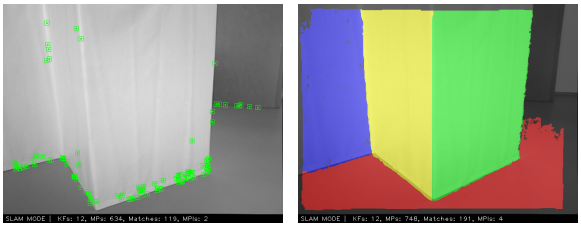
Results are given in Table 3.2 for the RGB-D sequences specified in Table 3.1 that compare ORB-SLAM2 and our proposed point-plane SLAM in two different scenarios as an ablation study.

In the first scenario we only consider point and plane variable nodes along with the observation and constraint factors between points and planes (without plane-plane constraint or Manhattan assumption). We denote this scenario by `PP` in Table 3.2. In the second scenario we also consider Manhattan assumption in our system and denote this by `PP+M` in Table 3.2.

It can be seen from the Table 3.2 that our point-plane SLAM improves the trajectory estimate in all of the sequences and up to 30% in some of them (the bold numbers) in both of the `PP` and `PP+M` scenarios. As expected, considering the plane-plane constraint factors significantly reduces the trajectory error when dominant planar structure is present in the scene. For instance, in the `fr1/desk` dataset, the Root-Mean-Squared-Error (RMSE) ATE has decreased significantly by 34.57%. The RTE and RRE has also decreased significantly which shows more accurate tracking and mapping. The reason for this improvement in this particular sequence is the presence of four desks at the same height (all belonging to the same infinite plane), as shown in Fig 3.6 first row. These planar surfaces impose a strong structural constraint on where points can lie in the scene and also enforces the reconstructed tables to be reconstructed parallel to the floor as evidence suggests. Our proposed method exploits these planar structures to improve the tracking and mapping of the scene. Another sequence in which we have a strong planar structure is `fr3/long_office`, where tables are connected to each other with monitors on top, as shown in Fig 3.6 last row. The presence of these planes particularly in the large loop closure present in this sequence improves the accuracy of tracking and mapping significantly, with a 34.55% reduction in ATE, 19.00% reduction in RTE, and 20.37% reduction in RRE.

Table 3.2 Comparison of RMSE for ATE, RTE, and RRE against RGB-D ORB-SLAM2 for 7 different TUM RGB-D datasets. PP and PP+M means two different scenarios that are considered in our experiments with points-planes only and points-planes+Manhattan constraint factors, respectively. Numbers in bold represent over 30% improvement over ORB-SLAM2.

Dataset	ATE (cm)			RTE (cm)			RRE (deg)		
	ORB-SLAM2	PP	PP+M	ORB-SLAM2	PP	PP+M	ORB-SLAM2	PP	PP+M
fr1/xyz	1.0457	0.9647	0.9231	1.5693	1.4675	1.2876	0.9871	0.9534	0.9037
fr1/desk	2.2668	1.5267	**1.4831**	4.0835	3.2994	3.1174	1.8547	1.7817	1.6932
fr1/floor	1.4399	1.3798	1.3246	4.2161	3.8789	3.6381	3.3229	2.8856	2.7839
fr3/cabinet	7.9602	7.3724	**2.1675**	15.1002	14.4081	**5.1328**	6.8639	6.5623	**2.9125**
fr3/str_notex_near	1.6882	1.0883	**1.0648**	4.0383	2.4540	**2.3533**	1.8476	1.1541	**1.1125**
fr3/str_notex_far	2.0007	1.9092	**1.3722**	3.4869	3.3523	3.0834	0.8479	0.7679	0.6695
fr3/long_office	1.5129	1.0601	**0.9902**	3.0555	2.6223	**2.4750**	0.8927	0.8062	**0.7109**



(a) RGB Frame with ORB Features (b) Detected Planes (c) Generated Map (Side) (d) Generated Map (Top)

Fig. 3.8 Our sparse point-plane SLAM results on the challenging low-texture `fr3/str_notex_far` sequence. Considering planes in spare mapping leads to more meaningful map representations.

Some of the most challenging scenarios for point-based ORB-SLAM2 are low-texture environments such as `fr3/str_notex_near` and `fr3/str_notex_far`. Our proposed point-plane SLAM shows great improvement of 36.93% and 31.41% respectively in the RMSE ATE, which implies the improvement in the reconstruction of walls and floor present in these sequences. The detected ORB features, in these sequences are on the intersections of planes (walls and floor). In these challenging low-texture environments, integrating planes in tracking and mapping shows its effectiveness demonstrably and yields richer and more semantically meaningful maps (Fig 3.6 third row and Fig 3.8) rather that just sparse points belonging to the intersection of walls and floor.

We further compare our point-plane SLAM with points-planes and Manhattan assumption against the recent plane-based SLAM work, Pop-up SLAM [28], which uses the point-based semi-dense LSD SLAM [3] to provide photometric odometry constraint along a free unconstrained direction. The comparison is given in Table 3.3, the number for Pop-up SLAM are taken from [28] for the `fr3/str_notex_far`. Even though Pop-up SLAM uses monocular input, the ground truth height of the camera is provided for initialization of their system and the reported results are with absolute scale and hence comparable to our work that uses RGB-D input only for *initializing* 3D points and planes in the back-end.

As Table 3.3 suggests, our SLAM system outperforms Pop-up SLAM on this challenging sequence. Pop-up SLAM loses the track of small movements of the camera due to the lack of enough plane landmarks (under-constrained) in this environment and lack of frame-to-frame odometry [28]. The generated map using our proposed system for this sequence can be seen in Fig 3.8.

Table 3.3 Comparison of Mean and Standard-Deviation (Std) for ATE in the challenging low-texture `fr3/str_notex_far` sequence for Pop-up SLAM and our point-plane SLAM.

Dataset	Pop-up SLAM		Ours	
	Mean	Std	Mean	Std
`fr3/str_notex_far`	18 cm	7 cm	1.75 cm	0.76 cm

3.6.3 Analysis of the Runtime

All the experiments have been performed on a commodity machine with an Intel Core i7-4790 CPU at 3.6 GHz and all the source code is implemented in C++. The back-end optimizations have been implemented also in C++ with the general framework for graph optimization, g2o [15], using Levenberg-Marquardt algorithm. The most time consuming part of the system is plane segmentation which takes on average 70 msec per frame, which is reduced by running in a multi-threaded pipeline to around 38 msec per frame. With the back-end optimization, our system runs near 19 Hz if we segment planes and track them in each frame. However we note that normally a plane in the scene can be easily tracked into the neighboring frames so it is enough to detect and segment planes in each key-frame instead of every frame and this allows our system to run in near real-time even above 19 Hz.

In Table 3.4 the detailed runtime analysis of our SLAM system and average statistics of the evaluated RGB-D datasets have been provided. All of theses timings have been measured when our system segment and match planes in each frame and the local bundle adjustment optimization is done with the presence of point and plane landmarks after adding each key-frame.

Table 3.4 Our sparse point-plane SLAM runtime average statistics on the evaluated RGB-D TUM sequences shown in Table 3.1 with plane segmentation and tracking in each frame in the loop.

Main Components	Runtime
`Plane segmentation`	38.1 msec
`Data association`	2.6 msec
`Local bundle adjustment optimization`	12.4 msec
`Total frame time`	53.1 msec

3.7 Conclusions

In this chapter we proposed a representation for plane landmarks as higher level structural entities amenable to graph-based SLAM systems. Also we imposed consequent observation/constraint factors pertinent to 3D planes, such as point-plane and plane-plane (Manhattan world assumption). In addition to introducing the mathematical representation for proposed landmarks and factors, we also elaborated the front-end of our system which builds and extracts the factor graph of SLAM problem from the input RGB-D sensory information.

We evaluated our system exhaustively against the state-of-the-art SLAM systems with publicly available benchmarks, qualitatively and quantitatively. The significant improved performance due to using points and planes along with additional constraints has been clearly shown by the experiments. Using planes is more effective in an environment where a dominant planar Manhattan structure is present. In cases where enough planes are not present, the point-based SLAM can still function as usual.

So far in this chapter, our SLAM system only uses the structural entities present in the scene and ignores higher level semantically meaningful landmarks, i.e. general objects. Therefore in the next chapter we explore methods to integrate generic objects (pose and extent) as independent landmarks into the graph-based SLAM problem, in addition to the sparse points and structural primitives such as planes.

Bibliography

[1] C. Brian. Hall. lie groups, lie algebras, and representations. an elementary introduction, volume 222 of. *Graduate Texts in Mathematics*, 2015.

[2] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32 (6):1309–1332, 2016.

[3] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[4] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[5] C. Forster, M. Pizzoli, and D. Scaramuzza. Svo: Fast semi-direct monocular visual odometry. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 15–22. IEEE, 2014.

[6] D. Gálvez-López and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[7] A. P. Gee and W. Mayol-Cuevas. Real-time model-based slam using line segments. In *International Symposium on Visual Computing*, pages 354–363. Springer, 2006.

[8] I. M. Gelfand, R. A. Minlos, and Z. Y. Shapiro. *Representations of the rotation and Lorentz groups and their applications*. Pergamon, London, 1963. URL http://cds.cern.ch/record/107042. Trans. from the Russian, Moscow, 1958.

[9] R. Gomez-Ojeda, F.-A. Moreno, D. Scaramuzza, and J. Gonzalez-Jimenez. Pl-slam: a stereo slam system through the combination of points and line segments. *arXiv preprint arXiv:1705.09479*, 2017.

[10] F. S. Grassia. Practical parameterization of rotations using the exponential map. *J. Graph. Tools*, 3(3):29–48, Mar. 1998. ISSN 1086-7651. doi: 10.1080/10867651.1998.10487493. URL http://dx.doi.org/10.1080/10867651.1998.10487493.

[11] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[12] S. Holzer, R. B. Rusu, M. Dixon, S. Gedikli, and N. Navab. Adaptive neighborhood selection for real-time surface normal estimation from organized point cloud data using integral images. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 2684–2689. IEEE, 2012.

[13] P. J. Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992.

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

[15] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011.

[16] T. Lemaire and S. Lacroix. Monocular-vision based slam using line segments. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2791–2796. IEEE, 2007.

[17] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[18] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[19] 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 *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[20] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[21] V. A. Prisacariu, O. Kähler, S. Golodetz, M. Sapienza, T. Cavallari, P. H. Torr, and D. W. Murray. Infinitam v3: A framework for large-scale 3d reconstruction with loop closure. *arXiv preprint arXiv:1708.00783*, 2017.

[22] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer. Pl-slam: Real-time monocular visual slam with points and lines. In *Proc. International Conference on Robotics and Automation (ICRA), IEEE*, 2017.

[23] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016.

[24] R. F. Salas-Moreno, B. Glocken, P. H. J. Kelly, and A. J. Davison. Dense planar slam. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 157–164, Sept 2014. doi: 10.1109/ISMAR.2014.6948422.

[25] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[26] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng. Point-plane slam for hand-held 3d sensors. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5182–5189. IEEE, 2013.

[27] A. Trevor, S. Gedikli, R. Rusu, and H. Christensen. Efficient organized point cloud segmentation with connected components. In *3rd Workshop on Semantic Perception Mapping and Exploration (SPME), Karlsruhe, Germany*, 2013.

[28] S. Yang, Y. Song, M. Kaess, and S. Scherer. Pop-up slam: Semantic monocular plane slam for low-texture environments. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 1222–1229. IEEE, 2016.

Chapter 4

Structure Aware SLAM using Quadrics and Planes

After incorporating 3D planes as independent landmarks in a factor graph of a sparse point-based SLAM system in Chapter 3, in this chapter towards a real-time semantic object-aware SLAM system, we consider further higher level entities such as objects for landmarks in the map that carry semantic information in addition to the geometric information. We propose a novel representation for generic objects as quadrics which allows object detections to be seamlessly integrated in a SLAM framework in section 4.3. Then we elaborate the consequent new observation/constraint factors between landmarks such as supporting/tangency affordances in section 4.4. The front-end of our proposed system, described in section 4.5, marries the classic geometric-based landmark detections and matching with the state-of-the-art deep-learned object detection methods which provide rich information about entities present in the scene from a single image, to extract and build the factor graph that is presented in preceding sections from the input RGB-D images. Our extensive evaluations and experiments show that the proposed points-planes-quadrics representation can easily incorporate Manhattan and object affordance constraints, greatly improving camera localization and is a step towards to semantically meaningful maps.

4.1 Introduction

Towards achieving a real-time semantic SLAM system that provides accurate geometric and semantic representation of the scene, we incorporated our first higher level entity, 3D plane, that also carries semantic information, to the sparse point-based SLAM

system in Chapter 3. Much of the large-scale structure of a general scene (especially indoors) comprises dominant planar surfaces. Planes are also a good representation for feature deprived regions, where they provide information complimentary to points and can represent significant portions of the environment with very few parameters. We explored our proposed representation for plane landmarks and the consequent point-plane and plane-plane (Manhattan assumption) constraint factors in Chapter 3. As shown in 3.6 we achieved a near real-time online and memory efficient SLAM system that represents a map with sparse points and 3D planes without incurring huge additional computation and memory.

Man-made environments contain many objects that could potentially be used as landmarks in a SLAM map, encapsulating a higher level of information than a set of points. Previous object-based SLAM efforts have mostly relied on a database of predefined objects – which must be recognized and a precise 3D model fit to match the observation in the image to establish correspondence [24]. Other work [1] has admitted more general objects (and constraints) but only in a slow, offline structure-from-motion context. In contrast, we are concerned with live (real-time) SLAM, but we seek to represent a wide variety of objects. Like [1] we are not so concerned with high-fidelity reconstruction of individual objects, but rather to represent the location, orientation and rough shape and extent of objects. A suitable representation is therefore potentially a quadric [6, 27], which allows a compact representation of rough extent and also is mathematically more convenient than other 3D pose and extent representations such as cuboid.

Modern SLAM is usually formulated as an unconstrained sparse nonlinear least-square problem [9]. The sparsity structure of the problem greatly effects the computation time of the systems. If planes and quadrics are to be introduced in a SLAM system, they should be represented in a way which is amenable to the non-linear least squares formulation and respects the sparsity pattern of the SLAM problem.

Pertinent to our purpose, such representations also provide the potential for additional constraints for landmarks, such as for the points that lie on one of the planes or introduction of useful affordance constraints between objects and their supporting planes, as we explain later in the chapter. All these constraints lead to better estimates of the camera pose.

In this chapter, we propose a map representation that, in addition to points and planes, consists of higher level geometric entities for objects as landmarks. Unlike previous work such as [1] we explicitly target real-time performance, and integrate

within an online SLAM framework. Such performance would be impossible with uncritical choices of representation and to that end we propose a novel representation of objects based on quadrics that decomposes in the dual space to permit clean, fast and effective real-time implementation. We show that this representation, along with point-plane, plane-plane (Manhattan), and plane-object (supporting) constraints, greatly reduces the error in the estimated camera trajectory without incurring great extra computational cost. Because of the higher-level primitives in the map, the representation remains compact, but carries crucial semantic information about the scene.

The idea of semantic SLAM is to include semantically meaningful entities in the SLAM framework to improve both localization and mapping. Recall that we distinguish this from semantic mapping (Chapter 2), in which SLAM is used as a tool to reconstruct a semantic map, but this semantic map does not inform localization. In contrast, the SLAM system proposed in this chapter uses the semantic representation in both mapping and localization and falls under our definition of *Semantic SLAM*. To the best of our knowledge, ours is the first *real-time* semantic SLAM system proposed in literature that incorporates both higher level primitives of planes and previously unseen objects as landmarks. The main contributions of this chapter are articulated as follows: (1) proposing a novel representation and decomposition of dual quadrics, and its related factors for integrating objects in the SLAM factor graph that is amenable to the non-linear least-squares framework and allows CNN-based object detections to be seamlessly integrated in a SLAM framework, (2) introducing a supporting affordance relationship between quadric objects and planes in a SLAM factor graph thanks to the proposed representation, and (3) integrating all of the higher level primitives: planes and quadrics, along with points, and geometric relationships among them, Manhattan assumptions and supporting/tangency constraints, in a complete online key-frame based SLAM system that performs near real-time.

The remainder of the chapter is organized as follows. Section 4.2 presents a short overview of methods that have already been developed to address the SLAM problem based on a combination of points, planes, and higher level entities such as objects. Then we represent our SLAM system in *three* separate sections: first we give detailed descriptions of the mathematical representations of each landmark (variable nodes) in Section 4.3 from an object-oriented perspective. Then our proposed observation/constraint factor nodes are explained in Section 4.4. Section 4.5 presents the front-end of our SLAM and explains how the preceding sections is integrated into

the SLAM system by detecting and matching the landmarks and also hypothesizing about the constraint factors based on the input RGB-D images. Experiments showing the efficacy and comparative performance of our system are presented in Section 4.6. We conclude the chapter in Section 4.7.

4.2 Related Work

There has been a lot of attention recently in incorporating additional higher level landmarks to point-based SLAM systems by either considering them as independent landmarks in the map and applying the classic geometric techniques to them or as prior information to fuse points with those primitives such as lines or planes to increase the accuracy of the SLAM trajectory estimation and mapping. For example [15, 7] replace points with line segments, however due to the lack of a good mathematical representation for integrating lines to the least-squares framework, they present a especial difficulty. In [21, 8] they explore the possibility of integrating lines and points in the same SLAM framework and have been more successful.

For incorporating planes in SLAM framework, [11] proposes a representation for infinite planes and build its SLAM system containing only planes, based on the visible infinite planes present in the scene by using the RGB-Depth information. By using a monocular input and training a Convolutional Neural Network (CNN), [31] generates plane hypotheses and refines them over time. To fuse point and planes using RGB-D images, [30] proposes a method to increase the accuracy of depth estimation based on the information from detected 3D planes.

Quadrics-based representation was first proposed in [2] and later used in a structure from motion setup [6, 23]. [27] reconstructs quadrics based on detected 2D bounding boxes in frames, however the objects are not explicitly modeled to remain bounded during graph optimization. Addressing previous drawback, [20] still relies on ground-truth data-association in a non-real-time quadric-only framework. [28] presented a semantic mapping system that uses object detection coupled with RGB-D SLAM to reconstruct precise object models in the environment, however object models do not inform localization. [24] presented an object based SLAM system that uses pre-scanned object models as landmarks for SLAM but can not be generalized to unseen objects. [17] presented a system that fused multiple semantic predictions with a dense map reconstruction. SLAM is used as the backbone to establish multiple view

correspondences for fusion of semantic labels but the semantic labels do not inform localization.

4.3 Object-Oriented Representations

For object-oriented SLAM the map comprises not only points but higher-level entities representing landmarks which aim to be more semantically meaningful than sparse points. However to maintain real-time operation, there is a trade-off between complexity of the landmark representation and the computational cost of tracking and mapping. Keeping sparse points as a backbone of tracking, due to their more reliable matching nature, is beneficial for estimating camera pose but not enough to represent a rich semantic map.

In this chapter we consider two kinds of landmarks, which admit efficient implementation but can broadly capture the overall structure of many scenes, especially those captured indoors: **a)** plane landmarks, as been introduced and explored in Chapter 3, whose role is to encapsulate high-level structure of regions; and **b)** quadrics (more specifically ellipsoids) that serve as a general representation of objects in scene, capturing not detailed shape, but key properties such as size, extent, position and orientation. We introduced proper representation of planes for factor graph optimization problem in Chapter 3, and now in this section we introduce representations of quadrics (representatives for general objects) that allow for efficient implementation in a SLAM framework. Since we want to add object landmarks to the SLAM framework that contains sparse points and planes landmarks (refer to Chapter 3 for more detail), we propose a representation for objects that in addition to computational efficiency, admits clean and effective constraints between other primitives and landmarks of the map, such as supporting constraint factors between objects and planes.

4.3.1 Quadric Representation

As noted above, we represent general objects in a scene using an ellipsoid that is an example of non-singular quadric surfaces. Generally speaking, a *quadric surface* in 3D space, or *quadric* as context permits, can be represented by a homogeneous quadratic form defined on the 3D projective space $\mathbb{P}^3$ which satisfies the following equation

$$\mathbf{x}^\top \mathbf{Q} \mathbf{x} = 0 \tag{4.1}$$

where $\mathbf{x} \in \mathbb{P}^3$ is the homogeneous 3D point (denoted by 4 parameters in $\mathbb{R}^4$) and $\mathbf{Q} \in \mathbb{R}^{4\times4}$ is a homogeneous 4×4 symmetric matrix representing the quadric surface, also called a *point quadric* as Equation 4.1 defines a constraint on points in the space. Note that quadric $\mathbf{Q}$ has 9 degrees of freedom, corresponding to independent entries of a 4×4 symmetric matrix less one for scale.

The action of a projective camera on a quadric is that it converts a 3D quadric surface to a *conic* section on image plane. However the relationship between a point quadric and its projection into an image plane (a conic) is not straightforward and can be tricky [10]. A widely accepted alternative is to make use of the *dual* projective space [2, 6, 27] in which a *dual quadric* $\mathbf{Q}^*$ is represented as the envelope of a set of tangent planes, viz:

$$\boldsymbol{\pi}^\top \mathbf{Q}^* \boldsymbol{\pi} = 0 \tag{4.2}$$

where $\boldsymbol{\pi}$ is a plane tangent to the point quadric $\mathbf{Q}$. The corresponding point quadric and the dual quadric are related to each other by $\mathbf{Q}^* = \text{adjoint}(\mathbf{Q})$ such as following

$$\mathbf{Q}^*\mathbf{Q} = \mathbf{Q}\mathbf{Q}^* = \det(\mathbf{Q})\mathbf{I} \tag{4.3}$$

and if $\mathbf{Q}$ is invertible, then we have the following relation

$$\mathbf{Q}^* = \det(\mathbf{Q})\mathbf{Q}^{-1} \tag{4.4}$$

Before diving into our proposed representation and the approximation that we use for incremental optimization of dual quadrics in the factor graph framework, we derive the way that a dual quadric $\mathbf{Q}^*$ transforms under a Euclidean transformation like $\mathbf{T}_c^w$ (camera pose) which is a point transformation from global to local coordinates system as expressed in Equation 3.2. By substituting Equation 3.6 into Equation 4.2, we find the transformation of dual quadrics such as following

$$\begin{aligned} \boldsymbol{\pi}_c^\top \mathbf{Q}_c^* \boldsymbol{\pi}_c = 0 \xrightarrow{3.6} \boldsymbol{\pi}_w^\top \underbrace{(\mathbf{T}_c^w)^{-1}\mathbf{Q}_c^*(\mathbf{T}_c^w)^{-\top}}_{\mathbf{Q}_w^*} \boldsymbol{\pi}_w = 0 \\ \implies \mathbf{Q}_w^* = (\mathbf{T}_c^w)^{-1}\mathbf{Q}_c^*(\mathbf{T}_c^w)^{-\top} = \mathbf{T}_w^c \ \mathbf{Q}_c^* \ {\mathbf{T}_w^c}^\top \end{aligned} \tag{4.5}$$

where subscripts c and w indicate the representation of landmarks in local camera and global world coordinates frame respectively.

Using dual quadrics greatly simplifies the relationship between the quadric and its projection to a conic, as will be discussed in more detail in Section 4.4.1, however a

further problem remains in the context of optimization in a graph-SLAM framework. The issue is that an incremental update of $\mathbf{Q}^*$, given a 9-dim error vector $\mathbf{e}$ in the tangent space of $\mathbf{Q}^*$, should be constrained to lie along a geodesic of the manifold. But finding these geodesics and updating with respect to them is computationally expensive, making a "straightforward" quadric representation intractable for incremental optimization.

We seek to address both of these issues. For our object representation, we would like to restrict landmarks to belong to the set of bounded quadrics, namely *ellipsoids*. To do so requires imposing the constraint that $\mathbf{Q}^*$ must have 3 positive and 1 negative eigenvalues, i.e. the signature of eigenvalues of $\mathbf{Q}^*$ has to be of the form $(+,+,+,-)$. Due to the Sylvester's law of inertia [14] this signature does not change under a homography of 3D projective space. Therefore based on this restriction and the fact that $\mathbf{Q}^*$ is a real symmetric matrix and by considering Euclidean transformations the representation of dual ellipsoids $\mathbf{Q}^*$ can be decomposed as:

$$\mathbf{Q}^* = \mathbf{T}_Q \mathbf{Q}_c^* \mathbf{T}_Q^\top = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ \hline \mathbf{0}^\top & 1 \end{array}\right] \left[\begin{array}{ccc|c} a^2 & 0 & 0 & 0 \\ 0 & b^2 & 0 & 0 \\ 0 & 0 & c^2 & 0 \\ \hline 0 & 0 & 0 & -1 \end{array}\right] \left[\begin{array}{c|c} \mathbf{R}^\top & \mathbf{0} \\ \hline \mathbf{t}^\top & 1 \end{array}\right] \tag{4.6}$$

where $\mathbf{T}_Q \in \mathbf{SE}(3)$ transforms an axis-aligned (canonical) quadric at the origin, $\mathbf{Q}_c^*$, to a desired $\mathbf{SE}(\mathbf{3})$ pose, and a, b, c denote the scale of the canonical ellipsoid $\mathbf{Q}_c^*$ along its principal axes. Note that the Equation 4.6 is compatible with the fact in Equation 4.5, however it acts in reverse and transforms a dual quadric to another pose but in the same coordinates system. In 4.6 a canonical dual ellipsoid $\mathbf{Q}_c^*$ is transformed to a desired pose $\mathbf{T}_Q$ to represent a general dual ellipsoid in the current coordinates system. Setting the homogeneous normalization factor to -1 in Equation 4.6 ensures the single negative eigenvalue for $\mathbf{Q}^*$ while the squared diagonal elements give the other three positive eigenvalues. Note that the signature $(+,+,+,-)$ of the eigenvalues of $\mathbf{Q}_c^*$, guaranteeing that quadric $\mathbf{Q}^*$ remains an ellipsoid after $\mathbf{SE}(3)$ transformation.

Optimizing on the space of quadrics must impose constraints on the eigenvalues of $\mathbf{Q}^*$ to force the solution to be an ellipsoid. Recently [23] and [6] have parameterized ellipsoids to overcome this problem. They optimize on the space of ellipsoids, denoted by $\mathbf{E}$, to localize the quadric in 3D map by their respective conic observations. However their representation requires solving a constrained least squares problem, without exploiting the underlying structure of $\mathbf{E}$. While their parametrization is useful for utilizing in observation of quadrics on camera image plane as conics (as an observation

factor node in factor graph), it can not be used in generic constraint factor nodes in the graph SLAM problem due to its constrained nature. The authors in [6] decompose the translation part of the representation, mainly for numerical stability in the optimization because of the different scales of translation and the other parts of the representation, and impose some prior knowledge on the shape of the ellipsoids.

For a more efficient representation of dual ellipsoids in graph-based SLAM, we propose to exploit the underlying manifold structure of $\mathbb{E}$ to represent the dual quadric. Since the upper-left block of the representing matrix of $\mathbf{Q}_c^*$ in 4.6 is a symmetric positive definite matrix (in here it is also diagonal), based on Cholesky factorization [14], it can generally be factorized to lower triangular matrices (or in this special case to diagonal "square roots") and based on this we propose to represent the $\mathbf{Q}^*$ such as following

$$\mathbf{Q}^* = \mathbf{T}_Q \, \mathbf{Q}_c^* \, \mathbf{T}_Q^\top = \left[\begin{array}{c|c} \mathbf{R} & \mathbf{t} \\ \hline \mathbf{0}^\top & 1 \end{array}\right] \left[\begin{array}{c|c} \mathbf{L}\mathbf{L}^\top & \mathbf{0} \\ \hline \mathbf{0}^\top & -1 \end{array}\right] \left[\begin{array}{c|c} \mathbf{R}^\top & \mathbf{0} \\ \hline \mathbf{t}^\top & 1 \end{array}\right] \tag{4.7}$$

where

$$\mathbf{L} = \begin{bmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{bmatrix} \tag{4.8}$$

with real numbers a, b and c, and so $\mathbf{L}\mathbf{L}^\top$ guarantees the required positive eigenvalues.

Inspired by group theoretic notions we thus represent a dual ellipsoid $\mathbf{Q}^*$ using a tuple $(\mathbf{T}, \mathbf{L})$ where $\mathbf{T} \in \mathbf{SE}(3)$ and $\mathbf{L}$ lives in $\mathbf{D}(3)$ the space of real diagonal 3×3 matrices, i.e. an axis-aligned ellipsoid (canonical) accompanied by a rigid transformation. By using the proposed decomposition, we propose to approximate the underlying geometry of the space of dual ellipsoids $\mathbb{E}$ by the induced geometry from the algebraic structure of $\mathbf{SE}(3) \times \mathbb{R}^3$. Note that here we used $\mathbb{R}^3$ instead of $\mathbf{D}(3)$ because of isomorphism between those two by considering matrix addition as group operation for $\mathbf{D}(3)$ and real number addition as group operation for $\mathbb{R}^3$.

Due to the proposed approximation, we update the $\mathbf{Q}^* = (\mathbf{T}, \mathbf{L})$ during the incremental optimization of factor graph separately in the underlying 6D space of $\mathbf{SE}(3)$ and 3D space of $\mathbf{D}(3)$, where both of them are Lie groups and can be updated efficiently by their respective Lie algebra. Thus the proposed update rule is:

$$\mathbf{Q}^* \oplus \Delta\mathbf{Q}^* = (\mathbf{T}, \mathbf{L}) \oplus (\Delta\mathbf{T}, \Delta\mathbf{L}) = (\mathbf{T} \cdot \Delta\mathbf{T}, \mathbf{L} + \Delta\mathbf{L}) \tag{4.9}$$

where $\oplus : \mathbb{E} \times \ \mathbb{E} \longmapsto \mathbb{E}$ is the mapping for updating ellipsoids, $\Delta\mathbf{L}$ is the update for $\mathbf{L}$ which comes from the first 3 elements of error vector $\mathbf{e}$ (calculated from the associated observation or constraint factor nodes) and applies in the space of $\mathbf{D}(3)$ (or isomorphically in Euclidean space of $\mathbb{R}^3$) and $\Delta\mathbf{T}$ is the update for $\mathbf{T}$ which comes from the last 6 elements of error vector $\mathbf{e}$ and applies in 6D space of $\mathbf{SE}(3)$ by using its Lie algebra $\mathfrak{se}(3)$ and associated exponential and logarithm maps. This decoupled update is a good approximation given the incremental nature of evidence and its efficacy is shown in Section 4.6.

An analogy to the notion of our proposed approximate update is the optimization problem on the space of $\mathbf{SE}(3)$ when considering the structure of $\mathbf{SE}(3)$ as a direct product of its underlying groups $\mathbf{SO}(3) \times \mathbb{R}^3$ instead of semi-direct product of them $\mathbf{SO}(3) \ltimes \mathbb{R}^3$. Note that as mentioned earlier in Chapter 3, $\mathbf{SE}(3)$ group is a 6D smooth manifold (Lie Group) embedded in the 16 dimensional manifold $\mathbf{GL}(4, \mathbb{R})$ and has the structure of semi-direct product of underlying groups $\mathbf{SO}(3) \ltimes \mathbb{R}^3$. More precisely, the difference between direct and semi-direct product of underlying groups of $\mathbf{SE}(3)$ is clarified in the following

$$\begin{aligned} \text{Direct product:} \quad & (\mathbf{R}, \mathbf{t}) \times (\Delta\mathbf{R}, \Delta\mathbf{t}) = (\mathbf{R}\Delta\mathbf{R}, \mathbf{t} + \Delta\mathbf{t}) \\ \text{Semi-direct product:} \quad & (\mathbf{R}, \mathbf{t}) \ltimes (\Delta\mathbf{R}, \Delta\mathbf{t}) = (\mathbf{R}\Delta\mathbf{R}, \mathbf{t} + \mathbf{R}\Delta\mathbf{t}) \end{aligned} \tag{4.10}$$

where $\mathbf{R}$ and $\Delta\mathbf{R}$ are rotation matrices in $\mathbf{SO}(3)$ and $\mathbf{t}$ and $\Delta\mathbf{t}$ are 3D translation vectors in $\mathbb{R}^3$. As seen in above Equations 4.10 in semi-direct structure the update in translation is applied after some twist by the rotation matrix. The direct approximation can be used instead of semi-direct rule when we are interested in approximating the optimal solution in the problem that have incremental nature.

The above proposed representation of ellipsoids and approximate update rule is beneficial particularly when we want to impose constraints on different parts of this representation. For instance, this representation for $\mathbf{Q}^*$ makes it possible to apply prior knowledge for shapes and sizes of objects, using the $\mathbf{L}$ component, prior information about location and orientation of the objects using the $\mathbf{T}$ component, and adjacency/supporting constraints (see Section 4.4).

4.3.2 Point and Plane Representation

We use the trivial notion of 3D vectors to represent 3D sparse points and to represent planes as structural entities in the map, we represent a plane (see Chapter 3) $\boldsymbol{\pi}$ in

3D projective space by its normalized homogeneous coordinates $\boldsymbol{\pi} = (a, b, c, d)^\top \in \mathbb{P}^3$ where $\mathbf{n} = (a, b, c)^\top$ is the normal vector and its orthogonal distance from the origin is $|d|/\|\mathbf{n}\|_2$ where $\|.\|_2$ is the Euclidean norm defined in the corresponding vector space.

By identifying the normalized homogeneous planes (northern hemisphere of $\mathcal{S}^3$) and $\mathbf{SO}(3)$ Lie group through a mapping described in Section 3.3.3, we avoid rank-deficient information matrices in factor graph optimization and perform the optimization using three elements that locally represent an element of the $\mathbf{SO}(3)$ manifold (a vector in the tangent space) "close enough" to a member of the manifold. For more precise details and explanations refer to Section 3.3.3.

4.4 The Factor Graph of Structure Aware SLAM

As discussed earlier in chapters 2 and 3, we can formulate our structure aware sparse SLAM problem as a factor graph optimization problem. In a traditional point-based SLAM system, observation factors exist between 3D points in the map and the camera key-frames that seek to minimize the following re-projection error:

$$f_r(\mathbf{x}_w, \mathbf{T}_c^w) = \| \mathbf{u}_c - \Pi(\mathbf{x}_w, \mathbf{T}_c^w) \|^2_{\boldsymbol{\Sigma}_r} \tag{4.11}$$

where $\mathbf{x}_w$ represents a point in the world coordinates system, $\mathbf{T}_c^w$ is the pose of the camera (an $\mathbf{SE}(3)$ transformation) which takes a point in the world coordinates frame $\mathbf{x}_w$ to a point in the current camera frame $\mathbf{x}_c = \mathbf{T}_c^w \mathbf{x}_w$ that is observed at the pixel location $\mathbf{u}_c$, and Π is a function that projects a world point into the camera image plane. $\|\mathbf{u}\|_{\boldsymbol{\Sigma}}$ is the Mahalanobis norm of vector $\mathbf{u}$ and equals to $(\mathbf{u}^\top \boldsymbol{\Sigma}^{-1} \mathbf{u})^{1/2}$ where $\boldsymbol{\Sigma}$ is the covariance matrix associated with the factor node.

Likewise if odometry is known between two camera key-frames (or robot positions), a factor involving camera poses can be formulated as:

$$f_o(\mathbf{T}_c^w, \mathbf{T}_k^w) = \| \widetilde{\mathbf{T}_c^k} \ominus \mathbf{T}_c^k \|^2_{\boldsymbol{\Sigma}_o} \tag{4.12}$$

where $\mathbf{T}_c^w$ and $\mathbf{T}_k^w$ are two different camera poses related with the odometry factor node, $\mathbf{T}_c^k$ and $\widetilde{\mathbf{T}_c^k}$ are, respectively, the difference between those two poses and the measurement of that. The difference between a pose variable and its measurement is calculated with $\ominus$ operation in the space of camera poses ($\mathbf{SE}(3)$). The solution to the SLAM problem is a configuration of the variable nodes that minimizes the error over all the involved factors.

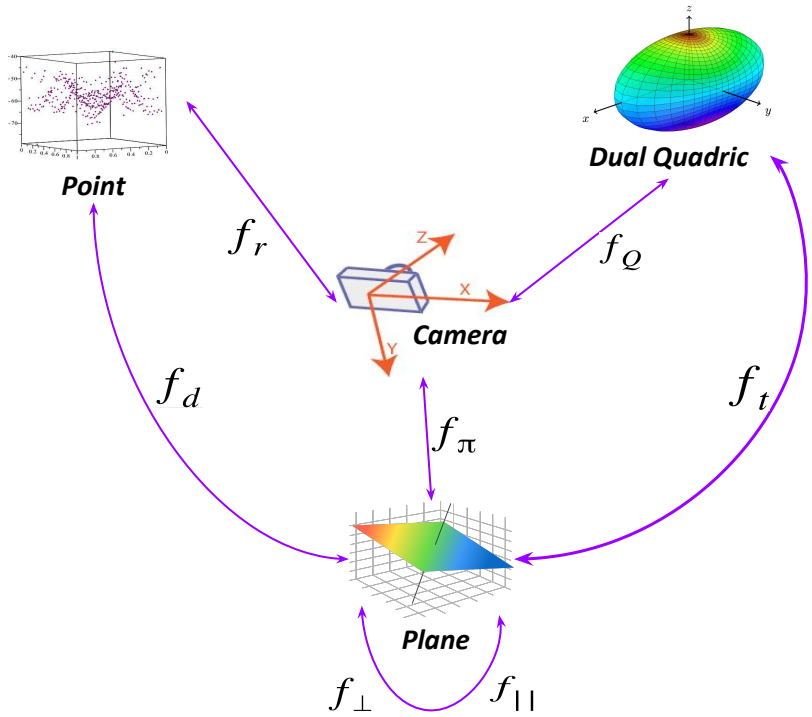


Fig. 4.1 The factor graph of our object-oriented structure aware SLAM system demonstrating all types of our landmark representations as variable nodes and observations/constraints as factors. For further details regarding these factors refer to section 4.4.

In our proposed object-oriented SLAM problem, the variable nodes in the SLAM factor graph consist not only of sparse points but potentially planes and/or general objects (represented by quadrics). For more details about sparse points and planes refer to Chapter 3. The various factors involving cameras, points, planes, and quadric objects in our system are illustrated in Fig. 4.1. Below we describe in more detail how the new components of our SLAM system are introduced as additional factor nodes in the graph.

4.4.1 Observation Factor for Object (Dual Ellipsoid)

Since we consider a 3D bounded dual quadric (ellipsoid) as a general representation of an object in the scene, the object will be observed from our SLAM system perspective as a 2D *conic section*, or *conic* as context permits, on the image plane, as illustrated in Fig 4.2, due to the geometric principles of projective camera [10]. As discussed in more details in Section 4.5.1, we use 2D bounding box detections from a deep-learned object detector to extract the conic observation out of that detected 2D bounding box.

As mentioned in Section 4.3.1, the relationship between a dual quadric and its perspective projection to a dual conic is more "straightforward" than point quadrics. The 2D dual conic $\mathbf{C}^*$ that is also represented by a homogeneous 3×3 symmetric matrix (so it has 5 degrees of freedom) is an envelope of tangent lines in the 2D image plane that satisfy the following

$$\mathbf{l}^\top \mathbf{C}^* \mathbf{l} = 0 \tag{4.13}$$

where $\mathbf{l}$ is a homogeneous 3D vector representing a line tangent to the conic section. Note that the relationship between point conics and dual conics is similar to that of quadrics and is calculated similarly by *adjoint* operator defined in Equations 4.3 and 4.4.

If the projection matrix of the perspective camera is $\mathbf{P}$, then we can back-project one tangent line $\mathbf{l}$ of the conic's envelope to find corresponding tangent plane $\boldsymbol{\pi}$ of the dual quadric's envelope such as following

$$\boldsymbol{\pi} = \mathbf{P}^\top \mathbf{l} \tag{4.14}$$

where $\mathbf{P}$ is the same projection matrix which projects a 3D point $\mathbf{x}$ into the pixel location $\mathbf{u}$ on the image plane by $\mathbf{u} = \mathbf{P}\mathbf{x}$. Now by substituting Equation 4.14 in Equation 4.2 we find the relationship between dual quadric $\mathbf{Q}^*$ and its projected dual conic $\mathbf{C}^*$ as following

$$\begin{aligned} \boldsymbol{\pi}^\top \mathbf{Q}^* \boldsymbol{\pi} = 0 \xrightarrow{4.14} \mathbf{l}^\top \underbrace{\mathbf{P}\mathbf{Q}^*\mathbf{P}^\top}_{\mathbf{C}^*} \mathbf{l} = 0 \\ \implies \mathbf{C}^* \simeq \mathbf{P}\mathbf{Q}^*\mathbf{P}^\top \end{aligned} \tag{4.15}$$

Note that $\mathbf{Q}^*$ and $\mathbf{C}^*$ both are homogeneous matrices and both sides of the Equation 4.15 are equal up to scale, hence $\simeq$ means an *equivalence relation* for equality of matrices up to scale. Recall that the camera projection matrix $\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{T}_c^w$ where $\mathbf{K}$ is the intrinsic camera calibration matrix and $\mathbf{T}_c^w$ is the extrinsic camera pose.

After calculating the projected dual conic from dual quadric and observing the dual conic from detected 2D bounding box (more details in Section 4.5.1) we propose the following observation factor node connecting dual ellipsoid landmark and camera key-frame pose:

$$f_Q(\mathbf{Q}^*, \mathbf{T}_c^w) = \parallel \hat{\mathbf{C}}^* - \hat{\mathbf{C}}^*_{obs} \parallel^2_{\boldsymbol{\Sigma}_Q} = \text{trace}((\hat{\mathbf{C}}^* - \hat{\mathbf{C}}^*_{obs})^\top (\boldsymbol{\Sigma}_Q^{-\frac{1}{2}})^\top (\boldsymbol{\Sigma}_Q^{-\frac{1}{2}})(\hat{\mathbf{C}}^* - \hat{\mathbf{C}}^*_{obs})) \tag{4.16}$$

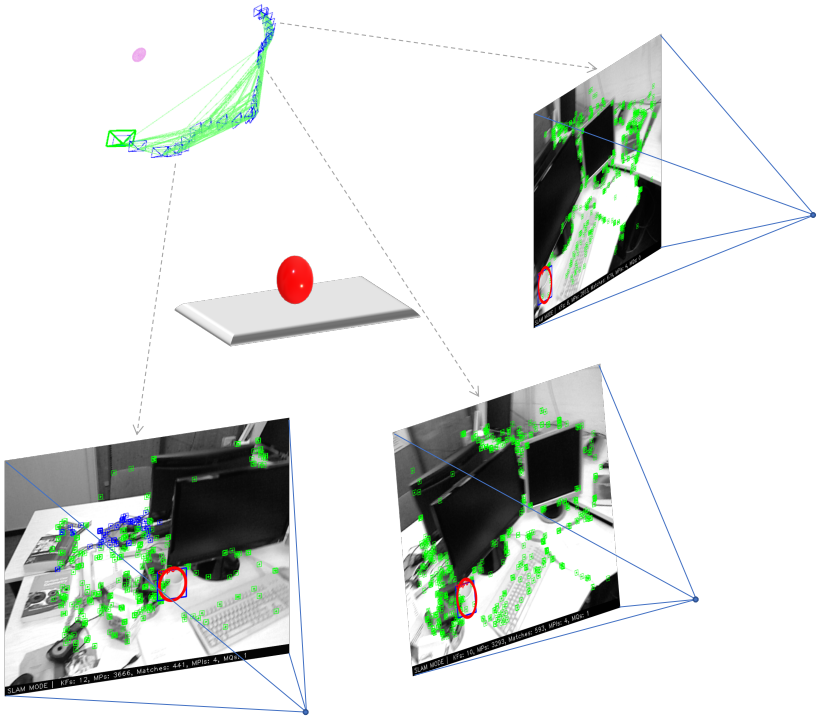


Fig. 4.2 The projective geometry of quadrics, conics, and perspective camera from multiple-views. The projection of a dual quadric (ellipsoid) in front of the camera is shown in different consecutive frames of a sequence in TUM [26] benchmarks, as a dual conic section rendered in red color in each frame.

where $\hat{\mathbf{C}}^*$ and $\hat{\mathbf{C}}^*_{obs}$ are normalized projected and observed dual conics respectively, and $\| \, . \, \|_{\mathbf{\Sigma}_Q}$ is the *weighted Frobenius norm* with the corresponding weight matrix of $\mathbf{\Sigma}_Q$ for the factor.

Note that to compute $\hat{\mathbf{C}}^*$ first we find the projection of $\mathbf{Q}^*$ by using the Equation 4.15 and then we normalize it based on its Frobenius norm. To calculate $\hat{\mathbf{C}}^*_{obs}$ first we find $\mathbf{C}^*_{obs}$ from detected 2D bounding box and then we normalize it by its Frobenius norm.

4.4.2 Observation Factor for Plane

If we denote the observation of the plane $\boldsymbol{\pi}$ from a camera pose $\mathbf{T}^w_c$ by $\boldsymbol{\pi}_{obs}$ (in camera coordinates frame), we can measure the observation error by the following observation factor

$$f_\pi(\boldsymbol{\pi}, \mathbf{T}^w_c) = \| \, d(\mathbf{T}^{w-\top}_c \boldsymbol{\pi}, \boldsymbol{\pi}_{obs}) \, \|^2_{\mathbf{\Sigma}_\pi} \tag{4.17}$$

where $\mathbf{T}^{w-\top}_c \boldsymbol{\pi}$ is the transformed plane to the camera coordinates frame, refer to 3.6, $\mathbf{\Sigma}_\pi$ is the covariance matrix of the Mahalanobis norm, and d is the difference of the

corresponding rotation matrices of the planes in the tangent space at the identity of $\mathbf{SO}(3)$. This observation factor is initially proposed in Chapter 3, for more details refer to Sections 3.3.3 and 3.4.2.

4.4.3 Point-Plane Constraint Factor

We introduce the following constraint factor node connected to point and plane variable nodes whenever the front-end of our SLAM system suggests that a point actually lies on that specific plane:

$$f_d(\mathbf{x}, \boldsymbol{\pi}) = \| \mathbf{n}^\top(\mathbf{x} - \mathbf{x}_o) \|^2_{\sigma_d} \tag{4.18}$$

which measures the Euclidean perpendicular distance of the point $\mathbf{x}$ from the plane $\boldsymbol{\pi}$ with the unit normal vector $\mathbf{n}$. Note that $\mathbf{x}_o$ can be any arbitrary point on the plane. The mentioned association between point and plane landmarks is established in the front-end of our system. For more details refer to Sections 3.4.3 and 3.5.2.

4.4.4 Plane-Plane Constraint Factor (Manhattan Assumption)

As introduced earlier in Chapter 3, enforcing Manhattan assumption on relative plane orientations is realized by introducing a constraint factor on the *inner product* of the plane surface normals. The Manhattan constraints between planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ with unit normal vectors $\mathbf{n}_1$ and $\mathbf{n}_2$, respectively, are implemented as

$$f_\parallel(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \| \, |\mathbf{n}_1^\top \mathbf{n}_2| - 1 \, \|^2_{\sigma_{par}} \qquad \textit{for parallel planes} \tag{4.19}$$

$$f_\perp(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \| \, \mathbf{n}_1^\top \mathbf{n}_2 \, \|^2_{\sigma_{per}} \qquad \textit{for perpendicular planes} \tag{4.20}$$

where σ_{par} and σ_{per} are the variances of the Mahalanobis norm of parallel and perpendicular constraints, respectively.

The efficacy of this assumption is demonstrated in Sections 3.6 and 4.6. For more details refer to Sections 3.4.4 and 3.5.2.

4.4.5 Supporting/Tangency Constraint Factor

Moving towards object-oriented semantic SLAM requires an ability to consider semantically meaningful observations or constraints for improving the accuracy of the localization and mapping and at the same time generating richer maps for higher

level tasks. One of these semantic constraints that can be interpreted to a geometric constraint is the supporting constraint.

The supporting/tangency constraint comes from this intuition that almost all stable objects in the scene are supported by structural entities of the scene like planes; e.g. commonly objects are found on the floor or on a desk. Such an affordance relationship can be imposed between a quadric object and a structural infinite plane by introducing a geometric tangency constraint between them. To the best of our knowledge, this is the first time that such a constraint has been included in an online SLAM problem.

Although imposing a tangency constraint in the space of point quadrics could be tricky because of indirect relationship between point quadrics and homogeneous planes, in the dual space representations such a constraint takes a particularly simple form. The elegance of our proposed supporting/tangency constraint is a direct consequence of our mathematical choice for landmark representation (see Section 4.3) and here is another testimony for the fact that *representation matters* in the context of graph-based SLAM.

As noted earlier in 4.3.1, the dual quadric $\mathbf{Q}^*$ is formed by the envelope of all the planes $\boldsymbol{\pi}$ tangent to the quadric that satisfies $\boldsymbol{\pi}^\top \mathbf{Q}^* \boldsymbol{\pi} = 0$. Inspired by this notion, we propose to use the mentioned equation as a measure of tangency between a quadric and a plane. If a plane $\boldsymbol{\pi}$ wants to support a dual quadric $\mathbf{Q}^*$, it has to minimize our proposed tangency measure introduced as factor such as following

$$f_t(\boldsymbol{\pi}, \mathbf{Q}^*) = \| \boldsymbol{\pi}^\top \hat{\mathbf{Q}}^* \boldsymbol{\pi} \|^2_{\sigma_t} \tag{4.21}$$

where σ_t is the covariance corresponding to this constraint factor node, $\boldsymbol{\pi}$ is the normalized homogeneous plane (our representation for plane landmarks that was explained in 3.3.3) and $\hat{\mathbf{Q}}^*$ is the normalized dual quadric by its matrix Frobenius norm such as following

$$\hat{\mathbf{Q}}^* = \frac{\mathbf{Q}^*}{\| \mathbf{Q}^* \|_{Fro}} = \frac{\mathbf{Q}^*}{\sqrt{\text{trace}(\mathbf{Q}^{*\top} \mathbf{Q}^*)}} \tag{4.22}$$

Note that since representations for planes and quadrics are homogeneous (equal up to scale) we normalize them before imposing the constraint factors that to be minimized.

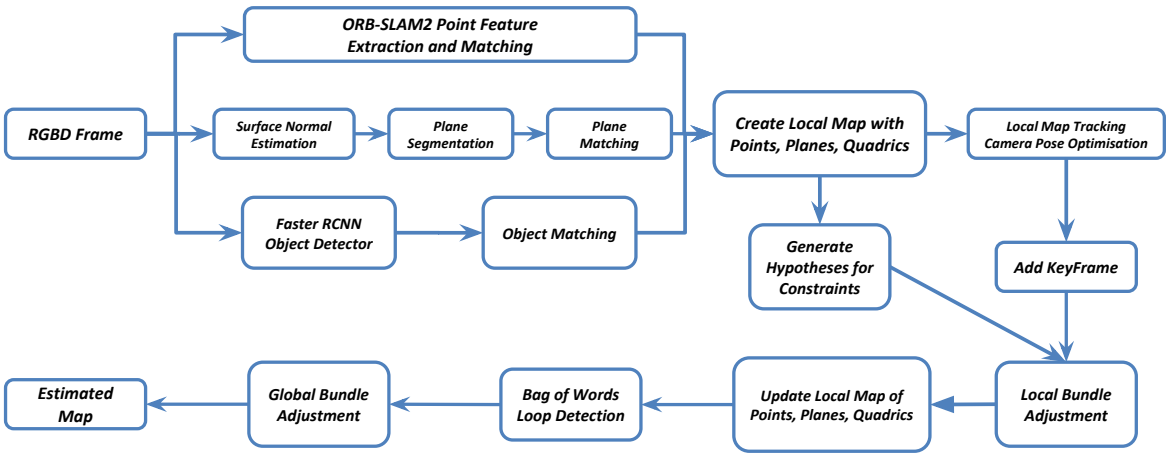


Fig. 4.3 The pipeline of our object-oriented SLAM system.

4.5 System Implementation

As mentioned earlier in chapters 2 and 3 modern SLAM system can be divided into two functional parts of front-end and back-end. We basically detailed the back-end of our SLAM system in previous Sections 4.3 and 4.4 by introducing the mathematical representations of variable nodes and observation/constraint factor nodes of the graph, respectively. Hence so far we have established the configuration of our factor graph, but now we want to mention the details of extracting and constructing this graph from the perceived RGB-Depth input stream of our visual SLAM.

In this section we provide an overview of the front-end of our SLAM. First we explain the landmarks detection mechanism, the initialization of the variable nodes associated with these landmarks, and also how we track them in the consecutive frames and map (data association). Then we discuss the details of hypotheses generation for observation/constraint factors.

Note that the back-end of our SLAM system optimizes this graph using a general least-squares framework [13]. It should be pointed out that all of the landmarks and factors participate in the back-end optimization after adding a new key-frame, as well as when a loop closure is detected. Our system is built on RGB-D variant of the publicly available ORB-SLAM2 [18]. Loops are detected using bag of words [5] based on ORB features. The pipeline of our system is demonstrated in Fig. 4.3.

4.5.1 Conic Observation

As discussed earlier in Section 4.4.1, for imposing an observation factor on a dual quadric variable node we need an observation of that dual quadric as a dual conic section on the image plane. Hence we need a sensor model to observe object landmarks as dual conics. With the recent successes of deep-learned object detectors, we can detect various objects in an RGB input image with bounding boxes and then extract the dual conic sections from those 2D rectangles.

To detect the bounding boxes for common objects of the scene, we use Faster-RCNN [22] with pre-trained model on COCO dataset [16]. To avoid outliers and having robust object detections we consider objects with 95% or more detection confidence.

From the axis-aligned detected bounding box, the inscribed ellipse is computed as conic projection of the observed quadric object. More precisely, we find the dual conic $\mathbf{C}^*$ representation of inscribed axis-aligned ellipse of the 2D bounding box $\mathcal{B}$ depicted in Fig 4.4, with the top-left corner $\mathbf{u}_{tl} = (u_{tl}, v_{tl})^\top$ and bottom-right corner $\mathbf{u}_{br} = (u_{br}, v_{br})^\top$ in the image plane coordinates system, such as following

$$\mathbf{C}^* = \mathbf{T}_c \mathbf{C}_c^* \mathbf{T}_c^\top = \left[\begin{array}{cc|c} 1 & 0 & u_c \\ 0 & 1 & v_c \\ \hline 0 & 0 & 1 \end{array}\right] \left[\begin{array}{cc|c} a^2 & 0 & 0 \\ 0 & b^2 & 0 \\ \hline 0 & 0 & -1 \end{array}\right] \left[\begin{array}{cc|c} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \hline u_c & v_c & 1 \end{array}\right] \tag{4.23}$$

where $\mathbf{C}_c^*$ is the axis-aligned dual ellipse centered on the origin that is translated by the $\mathbf{SE}(3)$ transformation $\mathbf{T}_c$ to the center of the bounding box $\mathbf{u}_c = (u_c, v_c)^\top = \frac{1}{2}(\mathbf{u}_{tl} + \mathbf{u}_{br})$, and a and b are the semi-axes of the ellipse that is calculated as following

$$\begin{aligned} a &= \frac{u_{br} - u_{tl}}{2} \\ b &= \frac{v_{br} - v_{tl}}{2} \end{aligned} \tag{4.24}$$

Note that the relationship that we use to represent the axis-aligned dual ellipse in Equation 4.23 is the 2D counterpart of the 3D dual quadric decomposition happened in Equation 4.6.

An implicit assumption in using dual ellipse as projective observation of dual ellipsoid is that we assume reasonably that the observed ellipsoid is in front of the camera, otherwise the projection of the ellipsoid can be a conic section other than ellipse. To show the effect of this "trivial" assumption on the projected dual conic, we consider a special scenario and project a canonical ellipsoid with the following dual

quadric representation

$$\mathbf{Q}_o^* = \begin{bmatrix} a^2 & 0 & 0 & 0 \\ 0 & b^2 & 0 & 0 \\ 0 & 0 & c^2 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \tag{4.25}$$

to the image plane of the camera centered on point $(d_x, 0, 0)^\top$ (on the $+x$ axis of the global world frame, $d_x > 0$) looking aligned towards the origin (the $+z$ direction of the local camera system is towards the origin). Hence the extrinsic pose and given intrinsic calibration matrix of the camera will be as follows:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad , \quad \mathbf{T}_{c_o}^w = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ -1 & 0 & 0 & d_x \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4.26}$$

Using 4.15 yields the projected dual conic as follows

$$\mathbf{C}_o^* = \mathbf{P}\mathbf{Q}_o^*\mathbf{P}^\top = \begin{bmatrix} f_x^2 b^2 & 0 & 0 \\ 0 & f_y^2 c^2 & 0 \\ 0 & 0 & a^2 - d_x^2 \end{bmatrix} \tag{4.27}$$

Note that a, b, and c are semi-axes of the $\mathbf{Q}_o^*$ along with the x, y, and z axes, respectively. To be an ellipse the sign of the $\mathbf{C}_o^*$ has to be $(+, +, -)$, so it is obvious from the result that $\mathbf{C}_o^*$ is a dual ellipse if and only if $|a| < |d_x|$ which means the camera has to be outside of the ellipsoids (looking towards from out), otherwise $\mathbf{C}_o^*$ will be some other conic section.

Quadric Landmark Initialization. One of the important roles to be performed by the front-end of a SLAM system is initialization of the variable nodes corresponding to the state of the landmarks in the map. Correct initialization of the quadrics is critical to find a good local minimum during factor-graph optimization. In previous works using quadrics as landmarks in the map, [27, 6, 23], quadric initialization is a challenging bottleneck due to the ill-conditioned problem of recovering a quadric from a single-view image.

However in our proposed SLAM we overcome this limitation by utilizing additional information present in other landmarks of the scene. For instance we use the robustly

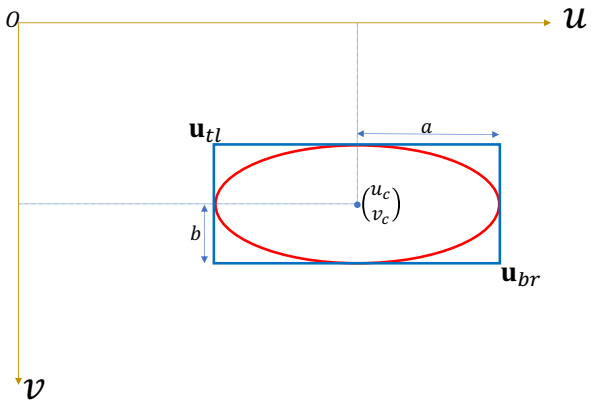


Fig. 4.4 Finding dual conic representation of inscribed ellipse of 2D bounding box detections in image plane, given by Faster-RCNN [22] object detector. Refer to section 4.5.1 for more details.

matched sparse 3D key-points in the map to help initialization of the quadric, as demonstrated in Fig 4.5. Our proposed parameterization allows independent initialization for the size and pose of the quadric. The 3D position of the center of the new quadric is initialized to be at the centroid of the 3D points that lie within the bounding box detection (i.e. the 3D key-points in the SLAM map). This initial estimate is more accurate for objects with uniform distribution of detected key-points. The size of the quadric is adjusted so that it fits the 2D bounding box and extent of the 3D point cloud in that box and it is then inserted into the factor-graph. More precisely, first by observing the 3D points projected inside the 2D bounding box detection we compute an enclosing sphere of those 3D points. Then we initialize another intermediate optimization problem with this sphere to minimize the observation factor 4.16 introduced in Section 4.4.1. The result of this intermediate optimization problem yields an effective initialization for the dual quadric variable node in the main back-end local map bundle-adjustment, as shown in Section 4.6. Note that this initialization scheme fails to initialize objects with no detected key-points – rarely happens for very small textureless objects, however missing objects and false negatives are better than false positive initializations.

Data Association. For inter-frame data-association of the quadrics, we utilize the semantic/class label of the detections (given by Faster-RCNN) as well as matched key-points (ORB features in image space) inside the conic observations. Note that image key-points are already matched over multiple views before matching quadrics in frames. The quadric with the same class label that has the most common key-points

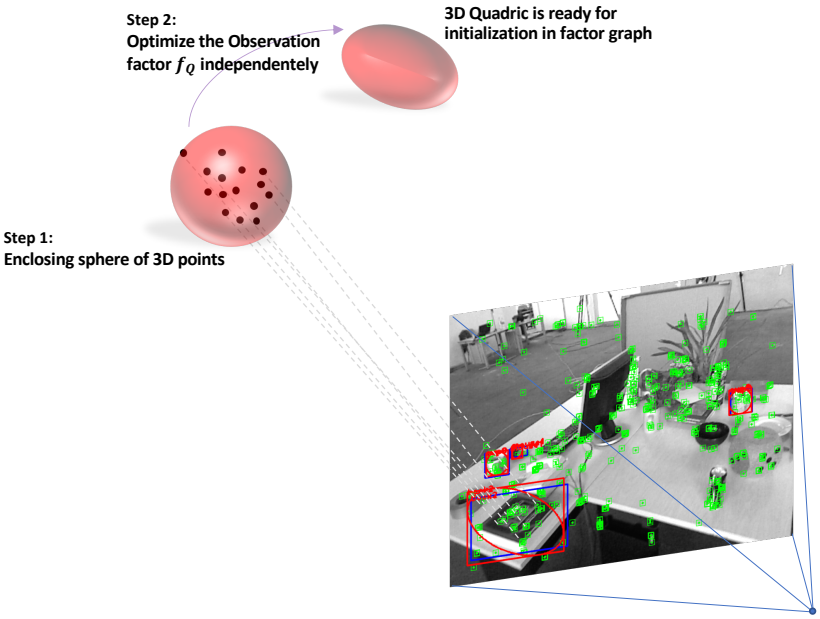


Fig. 4.5 The initialization process of the dual quadric landmark. First by observing the 3D points projected inside the 2D bounding box detection we compute an enclosing sphere of those 3D points. Then we initialize an optimization problem with this sphere to minimize the observation factor independently, introduced in section 4.4.1. The result of this intermediate optimization problem yields an effective initialization for the dual quadric variable node in the main back-end local map bundle-adjustment.

with the detected conic is deemed the correct match. This simple strategy is successful with high-confidence object detections, as shown in Section 4.6.

Note that the partial occlusions can result in inconsistent observations of same object from different viewpoints that can lead to inaccuracy in the trajectory and map. The following course of action is employed in our system to mitigate the negative impact of partial occlusions: **(a)** we use *robust kernels* (Huber) in our factor graph to robustify against large errors, and **(b)** as mentioned earlier we only consider objects with 95% or more detection confidence. With these two recourses we have seen almost consistent observations of COCO objects in our experiments shown in Section 4.6.

4.5.2 Plane Observation

For planar landmarks, we are interested not only in the parameters of the 3D planes, but also their extent visible in the current image, so that points can be associated to the planes on which they are observed. Most plane fitting models for RGB-D data use RANSAC which is extremely slow for the purpose of building a near real-time online

SLAM framework. Our plane segmentation module efficiently clusters and segments multiple planes from point-clouds generated from RGB-D data in near real-time. For more explanations refer to Section 3.5.2.1. For data-association across frames, we rely on the sparsity (few dominant planes in the scene) and inherent robustness (little variation frame-to-frame) of these landmarks. Using the difference between normals and the distance between planes, data-association is done in a nearest-neighbor fashion. Again for more details refer to Section 3.5.2.2.

Note that the plane segmentation and matching in our system uses RGB-D data, and is the only part of our system (other than ORB feature depth initialization) which relies on depth information. In the following chapter we aim to remove even this requirement and make the system truly monocular by hypothesizing planes using semantic segmentation and single-view normal estimation, as is now possible using deep networks, for instance [3].

4.5.3 Point Observation

In our object-oriented SLAM system, same as sparse point-plane SLAM described in Chapter 3, for key-point observation we utilize the RGB-D variant of the state-of-the-art sparse SLAM system, ORB-SLAM2; candidate features are extracted based on uniqueness and described using ORB features, with their depth initialized using the Depth channel of the RGB-D camera. For data-association across frames, ORB features are matched in a coarse-to-fine pyramid in a local window around the previous observation. For more details refer to the Section 3.5.1.

4.5.4 Point-Plane Hypothesis

Finding association between points and planes is established during plane detection and segmentation. After detecting each plane and its finite boundary in the image frame, its inlier points are determined to be those satisfying a threshold, th_{PP} distance, which we set as a function of the distance of the points from the camera, because further points have greater uncertainty. For more details refer to Section 3.5.2.

4.5.5 Plane-Plane (Manhattan) Hypothesis

The number of planes detected by our system is comparably smaller than the number of point landmarks, hence we can consider all possible pairs of planes, and introduce plane-

plane hypothesis without huge impact on overall speed of our system. As explained in more details in Section 3.5.2, we impose a parallel constraint if the smallest angle between the pair of planes is less than a threshold th_M^{par}, and if it is within th_M^{per} of 90 deg we introduce a perpendicular constraint factor. For our experiments we have used $th_M^{par} = 15\,\text{deg}$ and $th_M^{per} = 75\,\text{deg}$ in our system. For more details refer to Section 3.5.2.

4.5.6 Supporting/Tangency Hypothesis

A supporting/tangency constraint between a quadric and a plane is imposed based on the current estimated geometry and the Euclidean distance of the candidate quadric and plane. If this distance is less than th_S we enforce the tangency constraint. In our experiments this threshold depends on the size of the quadric $th_S = \max(20cm, a, b, c)$ where a, b, and c are half the length of the major axes of the ellipsoid.

4.6 Experiments

We perform a thorough evaluation of our proposed object-oriented SLAM system, both qualitatively and quantitatively, against the state-of-the-art SLAM systems in publicly available benchmarks, unlike other visual SLAM systems such as [11, 23, 25, 29, 23, 6] that lack an exhaustive comparison and ablation study against all common public benchmarks.

We evaluate the performance of our SLAM system using the common publicly available benchmarks RGB-D TUM [26] and NYU-v2 [19] datasets. These sequences have a wide range of conditions, from plane-rich scenes to scenes with little or no texture and also scenes with common objects such as those available in COCO dataset [16]. In addition to the mentioned benchmarks, we also gathered our own RGB-D sequences containing common objects with the aid of the UR5 robotic-arm in our lab[1] equipped with the Kinect RGB-D camera.

We show qualitative as well as quantitative results of our system using different combinations of the proposed landmarks and constraints and compare the accuracy in the estimated camera trajectory against the RGB-D variant of the state-of-the-art sparse localization and mapping system, ORB-SLAM2 [18]. The performance of our SLAM system is demonstrated in the video in this link: https://youtu.be/dR-rB9keF8M.

[1]The Robotics Vision lab affiliated with The Australian Centre for Robotic Vision and The University of Adelaide.

4.6.1 Qualitative Results

4.6.1.1 TUM and NYU Benchmarks

Some sequences in the TUM RGB-D benchmark contain little or no texture which makes it difficult for sparse point-based SLAM systems to extract and track key-points. However these sequences have rich planar structures which are exploited by our SLAM system. The results for using planes as additional landmarks with Manhattan constraints on different sequences such as `fr3/str_notex_far`, `fr3/str_notex_near`, `fr1/floor`, and `fr3/cabinet` are given in Fig 3.8, Fig 3.6, and Fig 3.7, respectively. The mentioned figures depict the image frame along with tracked ORB features, detected and segmented planes, and the reconstructed map consisting of points and planes from two different viewpoints.

For texture-less `fr3/str_notex_far` and `fr3/str_notex_near` sequences, ORB-SLAM2 [18] is unable to detect features in the environment with the normal settings and loses track. Lowering the feature detection threshold in ORB-SLAM2 yields a greater number of features, but also results in more outliers leading to more inaccurate trajectories. These sequences have no detectable objects (for Faster-RCNN [22]) and the effective landmarks are points and planes. However, our SLAM system significantly improves the trajectory estimates and results in richer maps for these sequences, particularly with Manhattan assumption, with a negligible amount of additional memory load.

To show the quality of the mapping and tracking with planes and objects along with the Manhattan and supporting constraints, we use the sequences `fr1/xyz`, `fr1/desk`, `fr2/desk` from TUM benchmark, and `nyu/basement_1a`, and `nyu/office_1` from NYU-v2 dataset. The reconstructions are shown in Fig 4.6. The reconstructed map of `fr1/xyz` is depicted in column (c) and (d) of the first row. The planar structure of the map is consistent with the ground truth scene which consists of two planar monitors orthogonal to the green desk. Quadrics corresponding to objects on the desk have been reconstructed tangent to the desk, their supporting plane. Column (a) shows tracked ORB features and possible detected COCO objects with confidence of at least 0.95 at the corresponding frame. The red ellipses in column (a) are the projection of the reconstructed quadric objects. They closely fit the detected blue bounding boxes and their corresponding green computed ellipses. Column (b) illustrates the detected and segmented planes, and the reconstructed map consisting of points, planes, and quadric objects from two different viewpoints are rendered in columns (c) and (d).

Tangency/Supporting Affordance. As seen in Chapter 3, we studied the importance and efficacy of Manhattan assumption (plane-plane constraints) in Section 3.6 and demonstrated in Fig 3.7. Now we want to explore the effectiveness of the other constraint in our system between two heterogeneous landmarks, namely quadrics and planes.

Figs 4.7(a,b) show the reconstructed quadric corresponding to the object on desk in the `fr1/xyz` before and after imposing the tangency constraint. Enforcing the tangency constraints makes sure that the quadric object is tangent to the supporting plane, for instance Fig 4.7 column (a) illustrates the reconstructed quadric enclosing the `cup` on the desk without supporting constraint and column (b) shows the quadric for the same object with supporting constraint, which is accurately reconstructed tangent to the supporting desk.

4.6.1.2 UR5-Kinect RGB-D Sequences

To evaluate the reconstruction of our SLAM system on a sequence with more common quadric objects, we captured an RGB-D sequence using the UR5 robotic-arm equipped with Kinect RGB-D camera and named it *UR5-Kinect*.

For capturing these RGB-D sequences we programmed the UR5 to move the grasped calibrated Kinect in a predefined smooth trajectory with controlled speed of the gripper (Robotiq 2 Finger Gripper 2F140) over a table containing multiple objects. The smooth motion of the robotic arm allows us to avoid image blur and rolling shutter effects to achieve robust object detection. The setup for our UR5 robotic-arm is demonstrated in Fig 4.8.

We captured sequences at full frame rate (30 Hz) containing registered RGB images (resolution of 640×480 8-bit `PNG` format) and depth maps (resolution of 640×480 16-bit monochrome images in `PNG` format). To be compatible with the publicly available RGB-D benchmarks, such as TUM, we scaled the depth maps by a factor of 5000, i.e., a pixel value of 5000 in the depth image corresponds to a distance of 1 meter from the camera and a pixel value of 0 means missing value or no data.

The detected objects in two different frames of our UR5-Kinect sequences, as well as the reconstructed map are shown in Fig 4.9. In this sequence, without adjusting the parameters of our plane detector for this benchmark, no planes are detected therefore the map consists of points and quadric objects as landmarks without any additional constraints. However, as shown in Fig 4.9, unlike other plane-only [11, 31] or quadric-

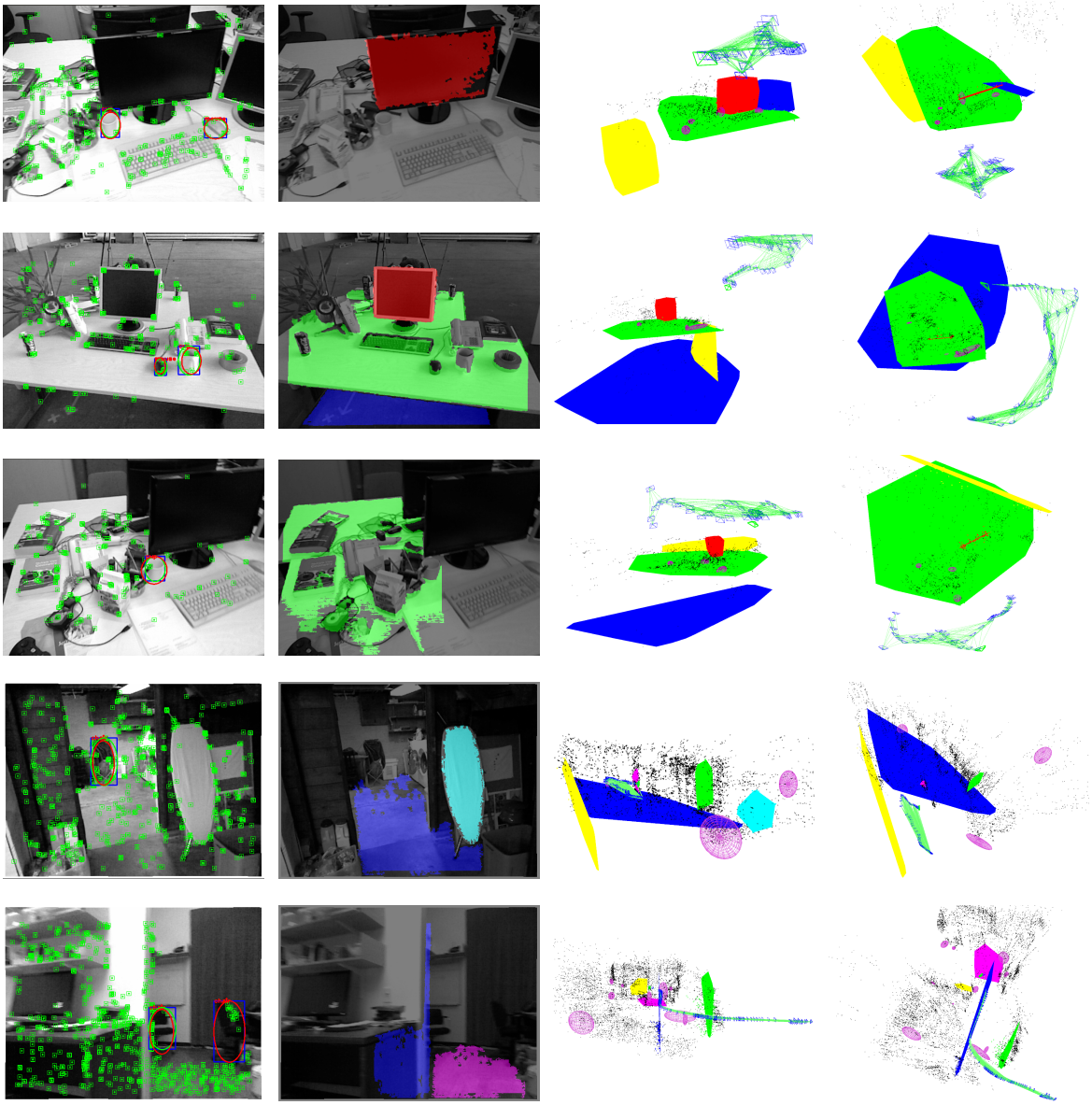
only [6, 23, 27, 20] frameworks, our SLAM operates successfully even with the absence of one kind of landmark in the scene.

4.6.2 Quantitative Comparison

We compare the performance of the proposed SLAM system against the RGB-D variant of the state-of-the-art system ORB-SLAM2 for TUM RGB-D dataset that the ground-truth trajectories are available. This baseline is a monocular sparse point-based system that uses the depth information in the D-channel to only initialize 3D points. We structure our results as an ablation study, considering the effects of introducing different landmarks and constraints. In each case, we report the Root Mean Square Error (RMSE) of Absolute Trajectory Error (ATE) and RMSE of relative errors, Relative Translational Error (RTE) and Relative Rotational Error (RRE), in Tables 4.1, 4.2, and 4.3, respectively.

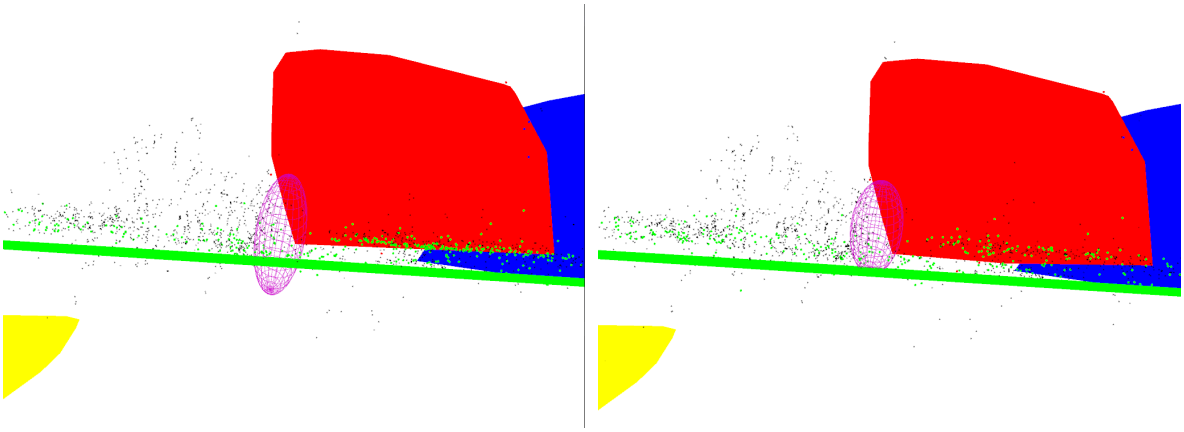
We first consider the case where points are augmented by the plane information (`PP`). This already improves the ATE in each case over the baseline, which improves even further by enforcing Manhattan constraints (`PP+M`). The Manhattan constraint significantly reduces the trajectory error when dominant planar structure is present in the scene.

Some sequences do not contain objects similar to the COCO dataset. For those that do, we investigate using the combination of points and quadrics (`PQ`) as landmarks. While this reduces the trajectory drift compared to baseline, the improvement is smaller compared with using `PP+M`. Finally, we report numbers for the full system (`PPQ+MS`) in which points, planes and quadrics are used as landmarks and Manhattan and supporting constraints are enforced. For `fr3/long_office` the improvement is significant (51.07%) because of the presence of a large loop in this sequence, where all of the points, planes and quadrics landmarks participate and are updated based on the loop closure.



(a) ORB Features and Detected Objects (b) Detected Planes (c) Generated Map (Side) (d) Generated Map (Top)

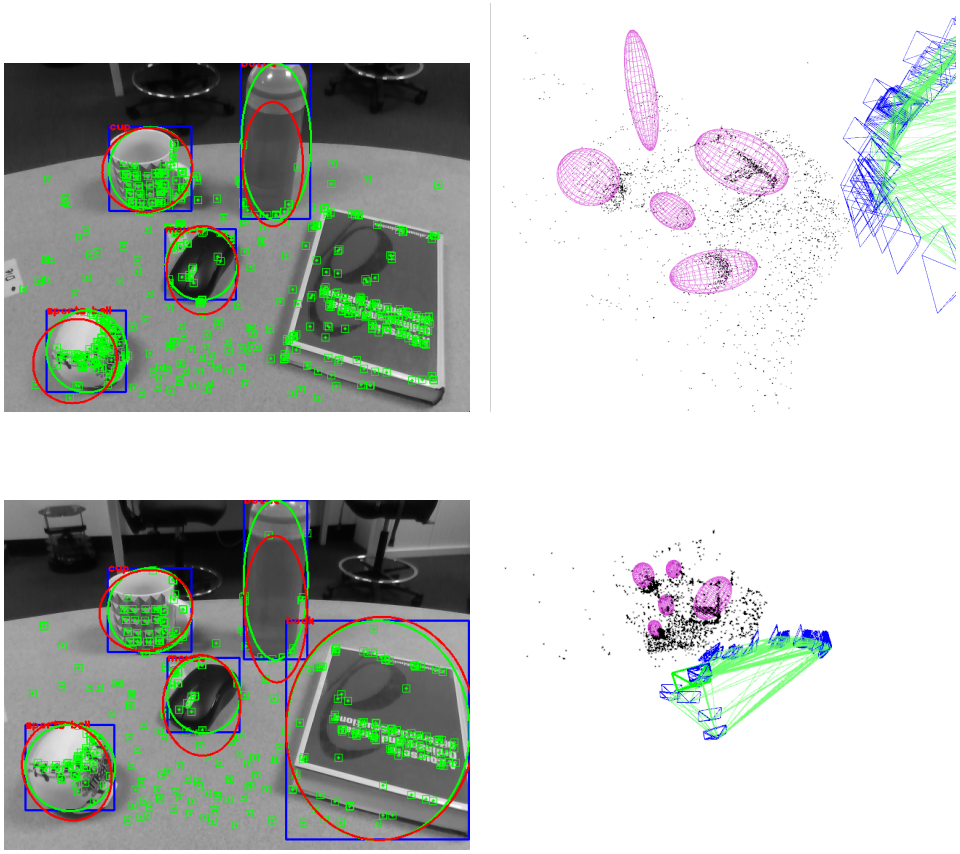
Fig. 4.6 Qualitative results for 3 different sequences from RGB-D TUM and 2 sequences from NYU-v2. These sequences contain rich planar structures and also common COCO [16] objects supported by planes.



(a) Generated Map (Before) (b) Generated Map (After)

Fig. 4.7 Qualitative comparison of the reconstructed quadric representing object `cup` before and after imposing supporting/tangency constraint between the quadric and the plane representing `desk` in the TUM benchmark `fr1/xyz` sequence.

Fig. 4.8 The setup of the UR5 robotic-arm equipped with Kinect RGB-D camera grassed with a Robotiq 2 Finger Gripper (2F140) in the Robotic Vision lab in the University of Adelaide, used to capture our sequences.



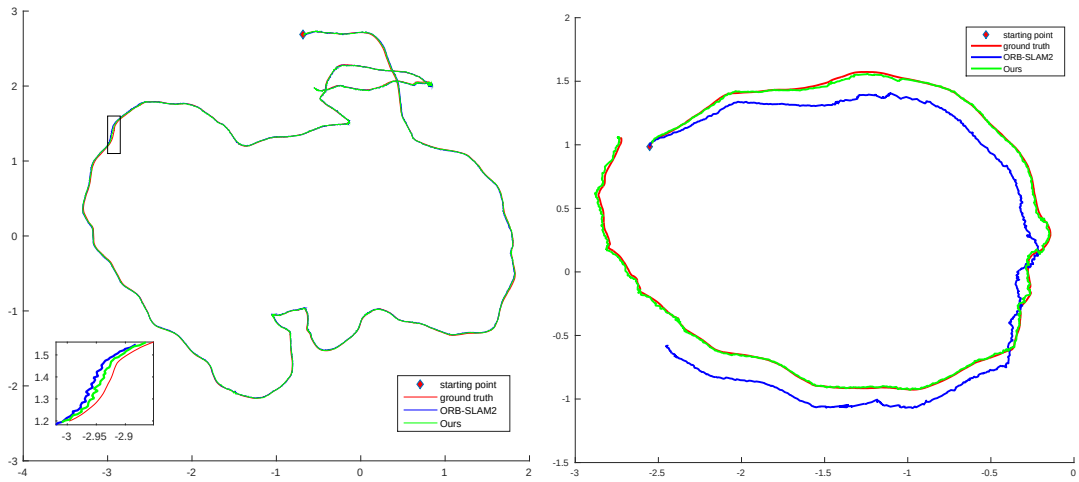
(a) ORB Features and Detected Objects

(b) Generated Map

Fig. 4.9 Qualitative results for our captured UR5-Kinect RGB-D sequences with the UR5 robotic-arm. Unlike other plane-only [11, 31] or quadric-only [6, 23, 27, 20] frameworks, our SLAM operates successfully even without detecting one kind of landmarks such as planar structures in this scene.

Table 4.1 **Comparison of Absolute Trajectory Error (ATE) against RGB-D ORB-SLAM2**. `PP`, `PP+M`, `PQ`, and `PPQ+MS` mean points-planes only, points-planes+Manhattan constraint, points-quadrics only, and all of the landmarks with Manhattan and supporting constraints, respectively. RMSE is reported for ATE in `cm` for 10 sequences in TUM RGBD datasets. Numbers in bold in each row represent the best performance for each sequence. Numbers in [ ] show the percentage of improvement over ORB-SLAM2.

Dataset	ORB-SLAM2	PP	PP+M	PQ	PPQ+MS
`fr1/floor`	1.4399	1.3798	**1.3246** [8.01%]	—	—
`fr3/cabinet`	7.9602	7.3724	**2.1675** [72.77%]	—	—
`fr3/str_notex_near`	1.6882	1.0883	**1.0648** [36.93%]	—	—
`fr3/str_notex_far`	2.0007	1.9092	**1.3722** [31.41%]	—	—
`fr1/xyz`	1.0457	0.9647	0.9231	0.9544	**0.9038** [13.57%]
`fr1/desk`	2.2668	1.5267	1.4831	1.9821	**1.4029** [38.11%]
`fr2/xyz`	0.3634	0.3301	0.3174	0.3453	**0.3097** [14.78%]
`fr2/rpy`	0.3207	0.3126	0.3011	0.3195	**0.2870** [10.51%]
`fr2/desk`	1.2962	1.2031	1.0186	1.1132	**0.8655** [33.23%]
`fr3/long_office`	1.5129	1.0601	0.9902	1.3644	**0.7403** [51.07%]



(a) `fr3/long_office` (b) `fr3/cabinet`

Fig. 4.10 Comparison of estimated trajectories of ORB-SLAM2, our system, and ground truth: **(a)** for TUM `fr3/long_office` that has a large loop closure and our trajectory is closer to the ground truth; **(b)** for TUM `fr3/cabinet` that ORB-SLAM2 drifts significantly in this feature-poor sequence and loses the track (more than 70% improvement in ATE).

Comparisons for RMSE of Relative Translational Error (RTE), and Relative Rotational Error (RRE) are reported in Table 4.2 and Table 4.3, respectively. The relative errors in the tables also confirm the mentioned comparisons for absolute trajectory errors in Table 4.1.

Note that column `PQ` in Tables 4.1, 4.2, and 4.3 presents the scenario in which both points and quadrics are used as landmarks. Constraints exist between point-camera and quadric-camera only (there is no explicit constraint imposed between points and quadrics). The column `PQ` shows that even if no other constraints are introduced, modeling quadrics as landmarks and tracking them gives a boost in performance over using points alone as landmarks. In addition, more information (planes, Manhattan assumption, supporting constraints) leads to further improvement as seen in other columns and scenarios.

In addition to the reported absolute and relative trajectory errors, we also compare the graph of estimated trajectories of our system against ground truth in Fig 4.10 for two example TUM sequences.

Table 4.2 **Comparison of Relative Translational Error (RTE) against RGB-D ORB-SLAM2**. `PP`, `PP+M`, `PQ`, and `PPQ+MS` signify points-planes only, points-planes+Manhattan constraint, points-quadrics only, and all of the landmarks with Manhattan and supporting constraints, respectively. RMSE is reported for RTE in `cm` for 10 sequences in TUM RGBD datasets. Numbers in bold in each row represent the best performance for each sequence. Numbers in [ ] show the percentage of improvement over ORB-SLAM2.

Dataset	ORB-SLAM2	PP	PP+M	PQ	PPQ+MS
`fr1/floor`	4.2161	3.8789	**3.6381** [13.71%]	—	—
`fr3/cabinet`	15.1002	14.4081	**5.1328** [66.01%]	—	—
`fr3/str_notex_near`	4.0383	2.4540	**2.3533** [41.73%]	—	—
`fr3/str_notex_far`	3.4869	3.3523	**3.0834** [11.57%]	—	—
`fr1/xyz`	1.5693	1.4675	1.2876	1.3795	**1.2464** [20.38%]
`fr1/desk`	4.0835	3.2994	3.1174	3.7453	**3.0237** [25.95%]
`fr2/xyz`	1.2107	1.0765	0.9659	1.1964	**0.9309** [23.11%]
`fr2/rpy`	0.5534	0.5322	0.5073	0.5484	**0.4883** [11.76%]
`fr2/desk`	4.7783	4.7110	3.6209	4.6309	**3.5545** [25.61%]
`fr3/long_office`	3.0555	2.6223	2.4750	2.5887	**1.8906** [38.12%]

Table 4.3 **Comparison of Relative Rotational Error (RRE) against RGB-D ORB-SLAM2**. `PP`, `PP+M`, `PQ`, and `PPQ+MS` signify points-planes only, points-planes+Manhattan constraint, points-quadrics only, and all of the landmarks with Manhattan and supporting constraints, respectively. RMSE is reported for RRE in `deg` for 10 sequences in TUM RGBD datasets. Numbers in bold in each row represent the best performance for each sequence. Numbers in [ ] show the percentage of improvement over ORB-SLAM2.

Dataset	ORB-SLAM2	PP	PP+M	PQ	PPQ+MS
`fr1/floor`	3.3229	2.8856	**2.7839** [16.22%]	—	—
`fr3/cabinet`	6.8639	6.5623	**2.9125** [57.57%]	—	—
`fr3/str_notex_near`	1.8476	1.1541	**1.1125** [39.79%]	—	—
`fr3/str_notex_far`	0.8479	0.7679	**0.6695** [21.04%]	—	—
`fr1/xyz`	0.9871	0.9534	0.9037	0.9433	**0.8822** [10.63%]
`fr1/desk`	1.8547	1.7817	1.6932	1.8044	**1.5214** [17.97%]
`fr2/xyz`	0.5036	0.4888	0.4456	0.4904	**0.4308** [14.46%]
`fr2/rpy`	0.9667	0.9586	0.9452	0.9611	**0.8770** [9.28%]
`fr2/desk`	1.6062	1.4232	1.1236	1.2559	**1.0034** [37.53%]
`fr3/long_office`	0.8927	0.8062	0.7109	0.8744	**0.6230** [30.21%]

Table 4.4 RMSE of Absolute Trajectory Error of some SLAM systems reported in `cm` for some common sequences in RGB-D TUM benchmark that are reported in their corresponding papers.

	Dense SLAM [12]	LSD-SLAM [4]	ORB-SLAM2 [18]	Ours
`fr1/xyz`	1.10	-	1.05	**0.90**
`fr2/xyz`	-	1.47	0.36	**0.31**
`fr2/desk`	1.70	4.52	1.30	**0.87**

Evaluation and Comparison against Direct SLAM Systems. We have chosen the RGB-D ORB-SLAM2 as a baseline for our ablation study (by gradually adding more landmarks and constraints in the RGB-D modality) since **(a)** ORB-SLAM2 is one of the state-of-the-art sparse SLAM systems outperforms most of the SLAM systems in the RGB-D case and **(b)** comparisons to monocular direct methods will not be fair due to the different sensing modalities (mono vs RGB-D). However, ignoring the sensing modalities, the camera trajectory errors for two direct method SLAM systems such as Dense SLAM [12] and LSD-SLAM [4] are reported in Table 4.4 for common sequences from their papers. As shown in Table 4.4, the mentioned direct methods are inferior to vanilla RGB-D ORB-SLAM2 in almost all of the evaluated sequences.

4.6.3 Runtime Analysis

All the experiments of our SLAM system have been carried out by a commodity machine with an Intel Core i7-4790 CPU at 3.6 GHz. All the source code is implemented in C++. The back-end optimization is also implemented in C++ with g2o [13], using Levenberg-Marquardt algorithm.

The chief enabler of near real time operation is the proposed representation of objects as dual quadrics, compared with the other works such as [1] that reasons about points, regions, and general objects but in a very slow framework with typical running time of $\sim$10 minutes for one image pair.

In terms of runtime, the bottle-neck of the system is the object detection component that is based on Faster-RCNN which operates at less than 10 frames-per-second (more than 100 msec per frame). Therefore, the object detections have been pre-evaluated offline for all of the sequences and the results of the per-frame object detection have been fed to the system during online operation. However this is not a fundamental restriction of our system and in next chapter will be alleviated by incorporating a real-time object detection method.

Table 4.5 Average runtime statistics of different components and threads of our SLAM system evaluated on the RGB-D TUM and NYU-v2 benchmarks with all the landmarks (points, planes, and quadric objects) with Manhattan assumption and supporting constraints.

Main Components and Threads	Runtime (msec)
`Plane Segmentation`	23.6
`Tracking & Matching Landmarks`	27.1
`Local Mapping Optimization`	348.4
`Global Bundle Adjustment`	2170.6
`Average Frame Time`	51.9

The runtime analysis and average statistics of different components and threads of our SLAM system evaluated on RGB-D TUM and NYU–v2 benchmarks are shown in Table 4.5. The system consists of three parallel modules: tracking, local map update, and global map update when a loop is closed. The tracking thread has to run at frame-rate while the other two can operate at a slower pace. Plane segmentation is done per-frame to do data-association against planes present in the map. The reported numbers are for the full system that utilizes all the landmarks (points, planes, quadric objects). The local map optimization is carried out in a parallel thread after creating and adding a new key-frame to the map.

4.7 Conclusions

In this chapter, we have explored the effects of incorporating quadrics as a representation for higher-level geometric entities and also planes in a sparse point-based SLAM framework. To do so we have introduced a new dual ellipsoid representation that is easily and effectively updated, and admits a simple method for imposing constraints between planes and objects. The improved performance due to using points and planes has been clearly shown by the experiments, most noticeably when there is dominant planar structure present. Of course in cases where enough planes are not present, the point based SLAM can still function as usual.

Note that our objective is not simply to boost the accuracy of the camera trajectory, but to develop a richer representation of the scene that models object locations and extents. We have shown in experiments that this leads to improved trajectory accuracy, but this is not the only advantage of our system.

Currently, the method works with RGB-D input. As in "vanilla" ORB-SLAM2, 3D map points are initialized with depth obtained from the Depth-channel of the RGB-D camera, we also use the D-channel to initialize planes, and this is both a bottleneck in terms of computation and presents a limitation on the sensor. In the next chapter, we will explore methods that can provide plane estimate from monocular input, which will enable us to transition to a purely monocular implementation. Another limitation of our method in this chapter is pre-computing objects in an offline fashion and using them later in the online operation of our system. We resolve this problem in the next chapter by using a real-time deep-learned object detector.

Bibliography

[1] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2012.

[2] G. Cross and A. Zisserman. Quadric reconstruction from dual-space geometry. In *Computer Vision, 1998. Sixth International Conference on*, pages 25–31. IEEE, 1998.

[3] D. Eigen and R. Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 2650–2658, 2015. doi: 10.1109/ICCV.2015.304. URL https://doi.org/10.1109/ICCV.2015.304.

[4] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[5] D. Gálvez-López and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[6] P. Gay, V. Bansal, C. Rubino, and A. D. Bue. Probabilistic structure from motion with objects (psfmo). In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3094–3103, Oct 2017. doi: 10.1109/ICCV.2017.334.

[7] A. P. Gee and W. Mayol-Cuevas. Real-time model-based slam using line segments. In *International Symposium on Visual Computing*, pages 354–363. Springer, 2006.

[8] R. Gomez-Ojeda, F.-A. Moreno, D. Scaramuzza, and J. Gonzalez-Jimenez. Pl-slam: a stereo slam system through the combination of points and line segments. *arXiv preprint arXiv:1705.09479*, 2017.

[9] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[10] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003. ISBN 0521540518.

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

[12] C. Kerl, J. Sturm, and D. Cremers. Dense visual slam for rgb-d cameras. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 2100–2106. Citeseer, 2013.

[13] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011.

[14] P. Lax. *Linear Algebra*. Pure and Applied Mathematics: A Wiley Series of Texts, Monographs and Tracts. Wiley, 1997. ISBN 9780471111115.

[15] T. Lemaire and S. Lacroix. Monocular-vision based slam using line segments. In *Robotics and Automation, 2007 IEEE International Conference on*, pages 2791–2796. IEEE, 2007.

[16] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[17] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4628–4635. IEEE, 2017.

[18] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[19] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[20] L. Nicholson, M. Milford, and N. Sünderhauf. Quadricslam: Dual quadrics from object detections as landmarks in object-oriented slam. *IEEE Robotics and Automation Letters*, 4(1):1–8, 2019.

[21] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer. Pl-slam: Real-time monocular visual slam with points and lines. In *Proc. International Conference on Robotics and Automation (ICRA), IEEE*, 2017.

[22] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[23] C. Rubino, M. Crocco, and A. D. Bue. 3d object localisation from multi-view image detections. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP (99):1–1, 2018. ISSN 0162-8828. doi: 10.1109/TPAMI.2017.2701373.

[24] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: simultaneous localisation and mapping at the level of objects. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 1352–1359, 2013. doi: 10.1109/CVPR.2013.178. URL https://doi.org/10.1109/CVPR.2013.178.

[25] R. F. Salas-Moreno, B. Glocken, P. H. J. Kelly, and A. J. Davison. Dense planar slam. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 157–164, Sept 2014. doi: 10.1109/ISMAR.2014.6948422.

[26] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[27] N. Sünderhauf and M. Milford. Dual Quadrics from Object Detection Bounding-Boxes as Landmark Representations in SLAM. *ArXiv preprints arXiv:1708.00965*, Aug. 2017.

[28] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid. Meaningful maps with object-oriented semantic mapping. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 5079–5085. IEEE, 2017.

[29] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. D. Reid. Meaningful maps with object-oriented semantic mapping. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5079–5085, 2017.

[30] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng. Point-plane slam for hand-held 3d sensors. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 5182–5189. IEEE, 2013.

[31] S. Yang, Y. Song, M. Kaess, and S. Scherer. Pop-up slam: Semantic monocular plane slam for low-texture environments. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 1222–1229. IEEE, 2016.

Chapter 5

Real-Time Monocular Object-Model Aware Sparse SLAM

After integrating depth-dependent plane and dual quadric landmarks in the factor graph framework of the sparse SLAM in Chapters 3 and 4, now in this chapter, we propose a monocular plane detector and tracker (Section 5.4) to move towards a full system operating real-time in monocular modality. Furthermore, we also propose new plane and object observation factors to avoid limitations of the previous chapters such as fixed global reference frame for landmarks, and axis-aligned conic observations (Sections 5.4.3 and 5.5.3). Coarse representation of objects as quadrics enables seamless integration in factor graph and also an elegant data-association which is exploited in Section 5.5.1 for a new proposed object matching scheme based on matched key-points. However we are not content with only coarse representation in this chapter and fine reconstructions of objects, learned by a CNN network, is also incorporated in the framework that provides a shape prior for the quadrics for further refinement (Section 5.5.2). In this chapter, we present our new contributions along with a brief overview of the components which are required from the previous chapters (Section 5.3), in order to present our monocular real-time semantic SLAM system in a standalone representation. Extensive experiments show that the introduced plane and object landmarks and the associated observation/constraint factors, using the proposed monocular plane detector and incorporated object detections, significantly improve the accuracy of camera trajectory and yield rich semantically meaningful maps consisting of planar layouts, and wherever available, high-fidelity object reconstructions.

5.1 Introduction

As a well-known and well-studied problem in computer vision and robotics, a sparse and efficient representation for Visual SLAM is to consider the map as collection of points in 3D, which carries information about geometry but not about the semantics of the scene. Denser representations such as [6, 5, 28, 30, 27], remain equivalent to a collection of points in this regard. Towards the aim of this work, we enrich this sparse map with more semantically meaningful landmarks and structures, while keeping the solution of SLAM problem computationally tractable in near real-time which is vital for interactive robotic applications.

In Chapter 3 we incorporated plane landmarks as structural entities providing information complimentary to points by representing significant portions of the environment (especially indoors) with few parameters. With carefully chosen representation for planes which is amenable to the factor graph formulation of the sparse SLAM frameworks, we continued that track to Chapter 4 by considering the fact that many objects in the scene can be used as landmarks in a SLAM map, encapsulating a higher level of abstraction than a set of points. Then we proposed a suitable coarse representation of the objects as bounded dual quadrics (ellipsoids) that encodes the rough extent and pose of the generic objects, unlike [33] that relies on a database of predefined objects. We also have inter-landmark relations formulated as constraint factors in Chapter 3 and Chapter 4 such as constraints among inlier points and their associated planes, inter-planar constraints (Manhattan world assumption), and newly introduced supporting affordance for planes as a tangency constraints among dual quadrics and their supporting surfaces. By considering proposed landmarks and constraints, in addition to gaining richer maps we boosted the estimate of the camera pose and its trajectory in near real-time, unlike [1] that performs in a very slow offline structure-from-motion context. However despite the near-time performance of the proposed structure and object aware SLAM in Chapter 4, we have some limitations in that system from two aspects:

★ From the front-end perspective such as:

- ✓ reliance on the *depth* channel for plane segmentation and parameter regression
- ✓ pre-computation of Faster R-CNN [32] object detections to permit real-time performance

- ✓ ad-hoc object and plane matching/tracking

★ From the back-end perspective:

- ✓ conic observations are assumed to be axis-aligned thus limiting the robustness of the quadric reconstruction
- ✓ all detected landmarks are maintained in a single fixed global reference frame, instead of a adaptable relative reference frame

This chapter in addition to addressing the mentioned limitations, proposes new observation factors amenable for real-time inclusion of plane and object detections while incorporating high-fidelity point-cloud reconstructions from a deep-learned CNN, wherever available, to the map and refines the quadric reconstruction according to this object model by the new proposed prior factor.

The main contributions of this chapter as follows: (1) proposing and integration a CNN-based plane detector to segment planes and regress the parameters (2) integrating a real-time deep-learned object detector in a monocular SLAM framework to detect general objects as landmarks along with a data-association strategy to track them, (3) proposing a new observation factor for objects to avoid axis-aligned conics inscribed in bounding boxes, (4) representing landmarks (planes and objects) relative to the camera key-frame where they are first observed instead of a global fixed reference frame, and (5) wherever available, integrating the reconstructed point-cloud model of the detected object from single image by a CNN to the map and imposing additional prior based on this point-cloud on the extent of the quadric to refine it even more.

The remainder of this chapter is organized as follows: In the next Section 5.2, we review some related work and then in Section 5.3 we overview the representations and factors adopted from previous chapters. Section 5.4 proposes a CNN-based monocular plane detector to segment and regress its parameters in our monocular SLAM framework along with the matching strategy to track planar regions. Section 5.5.1 describes the object detection module that has been integrated to our system and the proposed matching scheme to track objects. Section 5.5.2 presents the reconstruction of point-cloud model of objects from single-view RGB images integrated to our SLAM system, in addition to the prior constraint on the extent/scale of the quadric induced by the point-cloud. Sections 5.5.3 and 5.4.3 introduce our proposed multi-edge factors for relative plane and object observations, respectively. Experiments showing the efficacy and comparative performance of our system in extensive evaluation on standard SLAM

benchmarks are presented in Section 5.7. We conclude with a summary and discussion of future research directions.

5.2 Related Work

SLAM and semantic reasoning are two important and well-studied problems in robotics and computer vision communities that after recent success and attention to Deep Convolutional Neural Networks (Deep CNN) there has been a numerous effort to marry these two fields and build SLAM systems that are capable of building 3D map, reasoning and understanding 3D scene, and benefit from the semantic information to refine the map, such as [33], [12]. The most recent and popular approach to formulate Visual SLAM problem, particularly in sparse systems, like [24] and [6], is the graph-based nonlinear least squares methods [11].

To incorporate higher-level semantic landmarks, such as planes or objects, to the monocular setup of the SLAM problem, there has been multiple advances rooting from deep-learned approaches, such as: recent 2D object detection networks [32] and [31], 3D object detections such as hand-designed classic approaches [20] or deep-learned methods [37] that infers 6D object pose directly from RGB image. Also there has been efforts in 3D plane detection or inferring indoor planar layout from vanishing point based approaches like [14] to learning based ones such as [19], [22]. The predicted planes from these approaches are not consistent in consecutive frames and extracted from RGB-D modality with a considerable computation resource and time which makes them not suitable for a monocular near real-time SLAM graph-based framework. Utilizing planes as landmarks provides a large-scale consistent regularization for other landmarks like points, particularity for indoor scenes, [39], [16].

Despite methods such as [29] that uses point-based SLAM as backbone to build 3D map and then detect planes and objects, in our SLAM system we utilize 3D planes and objects as optimizable landmarks such as the well-known [1] which operates in a slow structure-from-motion fashion. There are methods depending on predefined object models such as [33] and [12], and also for generic objects such as [4] and [9] for structure-from-motion setup, and [35] for SLAM setup. [35] reconstructs quadrics based on detected 2D bounding boxes in frames, however it uses ground-truth data-association to build the graph-based SLAM and the objects are not explicitly modeled to remain bounded during bundle adjustment.

Unlike our semantic aware SLAM system that semantic objects refine camera localization, [36] presented a semantic mapping system by coupling RGB-D SLAM and object detections however object models do not inform localization, similar to [23] that presented a system to fuse semantic labels from multi-views within a dense map reconstruction. In this method SLAM is used as the backbone yet to discover multi-view associations for semantic label fusion but the semantics do not inform localization.

5.3 Overview of the Landmarks and Constraints

For the sake of completeness and convenience in following this chapter, this section presents a brief overview of the landmark representations and constraint factors that we utilize in this chapter, but proposed originally in previous Chapters 3 and 4. Then in the next sections, in addition to the proposed monocular pipeline to detect and track plane landmarks and objects in the map, we propose novel observation factors and sensor models to observe 3D planes and objects, and also object shape priors induced by the single-view point-cloud reconstructions.

We represent our SLAM problem as a bipartite graph $\mathcal{G}(\mathcal{V}, \mathcal{F}; \mathcal{E})$ where $\mathcal{V}$, the first part of the nodes, represents the set of *variable nodes* (landmarks) that need to be estimated and $\mathcal{F}$, the second part of the nodes, represents the set of *factor nodes* (observations/constraints) connected to the variable nodes via a set of edges $\mathcal{E}$. The solution of this problem is the optimum configuration of variable nodes, $\mathcal{V}^*$, that minimizes the overall error over the factors in the graph. The factor graph of our proposed SLAM system is demonstrated in Fig 5.1 with all of the incorporated landmarks and observation/constraint factors. In addition to the novel observation factors in this chapter, the newly imposed point-cloud induced shape prior factor is depicted in Fig 5.1, compared to the factor graph in previous chapter Fig 4.1.

5.3.1 Quadric Representation

As discussed earlier in Chapter 4, we represent generic objects in a scene using a bounded quadric or ellipsoid. A quadric surface in 3D space can be represented by a homogeneous quadratic form defined on the 3D projective space $\mathbb{P}^3$ that satisfies $\mathbf{x}^\top \mathbf{Q} \mathbf{x} = \mathbf{0}$, where $\mathbf{x} \in \mathbb{P}^3$ is a homogeneous 4-vector in $\mathbb{R}^4$ representing a 3D point and $\mathbf{Q} \in \mathbb{R}^{4\times4}$ is a symmetric homogeneous matrix representing the quadric surface.

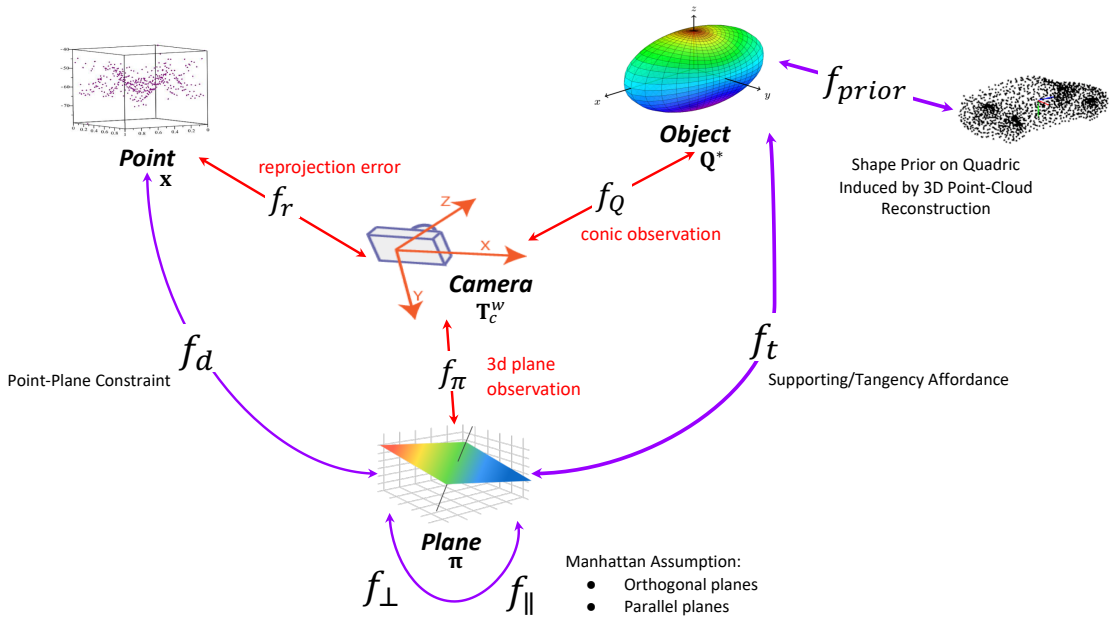


Fig. 5.1 The factor graph of our monocular object-model and structure aware sparse SLAM system demonstrating all types of our landmarks representations as variable nodes and observations/constraints as factors. Compared to the factor graph in chapter 4, in this factor graph we present novel observation factors f_π and f_Q, and also newly imposed point-cloud model induced shape prior factor on quadrics. For further details regarding these factors refer to sections 5.3, 5.4, and 5.5.

However, the relationship between a point-quadric $\mathbf{Q}$ and its projection into a camera (a *conic section*) is not straightforward [13].

A widely accepted alternative is to make use of the *dual space* ([4, 9, 35]) which represents a dual quadric $\mathbf{Q}^*$ by the envelope of planes $\boldsymbol{\pi}$ tangent to it, viz: $\boldsymbol{\pi}^\top \mathbf{Q}^* \boldsymbol{\pi} = 0$, which simplifies the relationship between the quadric and its projection to a conic. A dual quadric $\mathbf{Q}^*$ can be decomposed as $\mathbf{Q}^* = \mathbf{T}_Q \mathbf{Q}_c^* \mathbf{T}_Q^\top$ where $\mathbf{T}_Q \in \mathbf{SE}(3)$ transforms an axis-aligned quadric at the origin (canonical) $\mathbf{Q}_c^*$, to a desired $\mathbf{SE}(3)$ pose. Quadric landmarks need to remain bounded, i.e. ellipsoids, which requires $\mathbf{Q}_c^*$ to have 3 positive and 1 negative eigenvalues. In Chapter 4 we proposed a decomposition and incremental update rule for dual quadrics that guarantees this conditions and provides a good approximation for incremental update. More specifically, the dual ellipsoid $\mathbf{Q}^*$ is represented as a tuple $(\mathbf{T}, \mathbf{L})$ where $\mathbf{T} \in \mathbf{SE}(3)$ and $\mathbf{L}$ lives in $\mathbf{D(3)}$ the space of real diagonal 3×3 matrices, i.e. an axis-aligned ellipsoid accompanied by a rigid transformation.

The proposed approximate update rule for $\mathbf{Q}^* = (\mathbf{T}, \mathbf{L})$ from Chapter 4 is:

$$\mathbf{Q}^* \oplus \Delta\mathbf{Q}^* = (\mathbf{T}, \mathbf{L}) \oplus (\Delta\mathbf{T}, \Delta\mathbf{L}) = (\mathbf{T} \cdot \Delta\mathbf{T}, \mathbf{L} + \Delta\mathbf{L}) \tag{5.1}$$

where $\oplus : \mathbb{E} \times \mathbb{E} \longmapsto \mathbb{E}$ is the mapping for updating ellipsoids, $\Delta\mathbf{L}$ is the update for $\mathbf{L}$ and $\Delta\mathbf{T}$ is the update for $\mathbf{T}$ that are carried out in the corresponding lie-algebras of $\mathfrak{d}(3)$ (isomorphic to $\mathbb{R}^3$) and $\mathfrak{se}(3)$, respectively. For more specific details refer to the Section 4.3.1.

5.3.2 Plane Representation

We utilize the intuitive notion of 3D vectors to represent 3D sparse points and a plane $\boldsymbol{\pi}$ as a structural entity in the map is represented in 3D projective space by its normalized homogeneous coordinates $\boldsymbol{\pi} = (a, b, c, d)^\top \in \mathbb{P}^3$ where $\mathbf{n} = (a, b, c)^\top$ is the normal vector and its signed orthogonal distance from the origin is $d/\|\mathbf{n}\|_2$ where $\|.\|_2$ is the Euclidean norm defined in the corresponding vector space. For more precise details and explanations refer to Section 3.3.3.

5.3.3 Constraints between Landmarks

In addition to the classic point-camera constraint formed by the observation of a 3D point as 2D feature key-point in the camera, we model constraints between higher level landmarks and also their observations in the camera. These constraints also carry semantic information about the structure of the scene, such as Manhattan assumption and affordances. We present a brief overview of the previously proposed constraints here. In the next sections we present the newly introduced factors regarding plane and object observations and object shape priors induced by object's point-cloud model reconstruction.

5.3.3.1 Point-Plane Constraint

For a point $\mathbf{x}$ to lie on its associated plane $\boldsymbol{\pi}$ with the unit normal vector $\mathbf{n}$, we introduce the following factor between them:

$$f_d(\mathbf{x}, \boldsymbol{\pi}) = \| \mathbf{n}^\top(\mathbf{x} - \mathbf{x}_o) \|^2_{\sigma_d} \tag{5.2}$$

which measures the orthogonal distance of the point and the plane, for any arbitrary point $\mathbf{x}_o$ in the plane. Note that the notation of $\|\mathbf{e}\|_{\mathbf{\Sigma}}$ means the Mahalanobis norm of vector $\mathbf{e}$ and is defined as $\mathbf{e}^\top \mathbf{\Sigma}^{-1} \mathbf{e}$ where $\mathbf{\Sigma}$ is the associated covariance matrix of the norm. For more details refer to Section 3.4.3.

5.3.3.2 Plane-Plane Constraint (Manhattan assumption)

Manhattan world assumption where planes are mostly mutually parallel or perpendicular, is modeled as:

$$f_{\parallel}(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \parallel |\mathbf{n}_1^\top \mathbf{n}_2| - 1 \parallel^2_{\sigma_{par}} \qquad \textit{for parallel planes} \tag{5.3}$$

$$f_{\perp}(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = \parallel \mathbf{n}_1^\top \mathbf{n}_2 \parallel^2_{\sigma_{per}} \qquad \textit{for perpendicular planes} \tag{5.4}$$

where planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ have unit normal vectors $\mathbf{n}_1$ and $\mathbf{n}_2$, and σ_{par} and σ_{per} are the variances of the Mahalanobis norm of parallel and perpendicular constraints, respectively. For more details refer to Section 3.4.4.

5.3.3.3 Tangency Constraint (Supporting Affordance)

In normal situations planar structure of the scene affords stable support for common objects, for instance floors and tables support indoor objects and roads support outdoor objects like cars. To impose a supporting affordance relationship between planar entities of the scene and common objects, which models the tangency relationship between them, we introduce a factor between dual quadric object $\mathbf{Q}^*$ and plane $\boldsymbol{\pi}$ as:

$$f_t(\boldsymbol{\pi}, \mathbf{Q}^*) = \parallel \boldsymbol{\pi}^\top \hat{\mathbf{Q}}^* \boldsymbol{\pi} \parallel^2_{\sigma_t} \tag{5.5}$$

where σ_t is the covariance corresponding to this constraint factor and $\hat{\mathbf{Q}}^*$ is the normalized dual quadric by its matrix Frobenius norm. Please note that this tangency constraint is the direct consequence of choosing dual space for quadric representation, which would not have been straightforward in the space of point quadrics. For more details refer to Section 4.4.5.

5.4 Monocular Plane Detection

Man-made environments contain planar structures, such as table, floor, wall, road, etc. If modeled correctly, they can capture information about large feature-deprived

regions providing more map coverage. In addition, these landmarks act as a regularizer for other landmarks when constraints are introduced between them. The dominant approach for plane detection is to extract them from RGB-D input images as proposed in Chapter 3 which provides reliable detection and estimation of plane parameters.

In a monocular setting, 3D planes need to be detected using a single RGB image and their parameters estimated, which is an ill-posed problem. However, recent breakthroughs in deep neural networks suggest the potential to detect and estimate planes from single RGB image. Recently, PlaneNet [22] presented a deeply learned network to directly predict plane parameters and corresponding segmentation masks. As discussed in more details in Section 5.7, while planar segmentation masks are highly reliable, the regressed parameters are not accurate enough for small planar regions in indoor scenes. To address this shortcoming, instead of segmenting and estimating plane parameters directly with training a Convolutional Neural Network (CNN), we propose to first train a network to predict depth, surface normals, and semantic segmentations and then utilize the redundant information present in these predictions to detect and estimate plane parameters reliable from single monocular images. Depth and surface normal contain complementary information about the orientation and distance of the planes, while semantic segmentation allows reasoning about identity of the regions such as wall, floor, etc. We show this to be more accurate than PlaneNet [22].

We propose our monocular plane detector in next Section 5.4.1 to detect and segment plane masks and regress parameters by using separated outcomes of the learned models for semantic segmentation, depth, and surface normal estimations. In Section 5.4.2 we describe our data association scheme for plane landmarks detected in input image frames and finally we propose our novel observation factor for plane landmarks in Section 5.4.3 that lets plane landmarks live in relative key-frames instead of global world coordinates system.

5.4.1 Planes from predicted depth, surface normals, and semantic segmentation

To jointly predict depth, surface normal, and semantic segmentation and to exploit the redundant information of those predictions, we make use of the state-of-the-art real-time joint network architecture proposed in [26] to estimate depth, normals, and semantic segmentations for each RGB frame in real-time without crippling computational cost. This network is trained on the benchmarks used in the experiments, Section 5.7.

We denote the depth, surface normal, and semantic segmentation output predictions of the network with $\mathcal{D}$, $\mathcal{N}$, and $\mathcal{S}$, respectively, and exploit the redundancy in these three separate predictions to boost the robustness of the plane detection by generating plane hypotheses in two steps as illustrated in Fig 5.2:

1) In the first step, we create the point-cloud associated with the RGB image frame by using predicted depth of the network $\mathcal{D}$ (we call it *virtual depth* as context permits) and then we segment the resulting point-cloud by clustering surface normals and depths one after another as proposed in Chapter 3. Note that we estimated surface normals from depth in Chapter 3, however here we make use of the predicted surface normals $\mathcal{N}$ from the joint network. Hence in this step we generate plane segmentations and parameter estimations based on the point-cloud segmentation resulting from virtual depth and surface normals.

2) In the second step, we use the intersection of segmented planes from step 1 and planar regions in the semantic segmentation $\mathcal{S}$ (regions such as floor, wall, monitor, etc.) to instantiate the semantic segmentations and find instances out of planar semantic masks, in addition to assigning semantics to the segmented planes from step 1. Then in order to reduce the false positive detections even more and make the parameter estimation more robust, we generate the second plane hypothesis by considering these semantic segmentations. As depicted in Fig 5.2, in this step we are only reconstructing the point-cloud for each instance of plane segmentation (the intersection of step 1 masks and $\mathcal{S}$) by using the virtual depth $\mathcal{D}$ and surface normals $\mathcal{N}$ of that region. Then we fit a plane to the resulting point-cloud and consider that as the second hypothesis for the plane parameters.

Finally, a positive plane detection is determined based on the consensus of two hypotheses generated in the above mentioned steps. More precisely, plane detection $\boldsymbol{\pi} = (a, b, c, d)^\top$ is considered to be valid if the cosine distance of normal vectors $\mathbf{n} = (a, b, c)^\top$ and also the distance between the d value of the two planes from two hypotheses are within a certain threshold. The corresponding plane segmentation mask is taken to be the intersection of the mentioned plane masks from step 1 and semantic segmentation $\mathcal{S}$ prediction of the network. The resulting plane detections for some frames of the TUM [34] and NYU-v2 [25] are illustrated in the column (b) of Fig 5.8.

We extensively compare the performance of the SLAM system with our proposed monocular plane detection with the state-of-the-art monocular plane detection network, PlaneNet [22], and also a *baseline* plane detector (which uses ground truth depth to segment planes as benchmarks permit) in experiments Section 5.7.

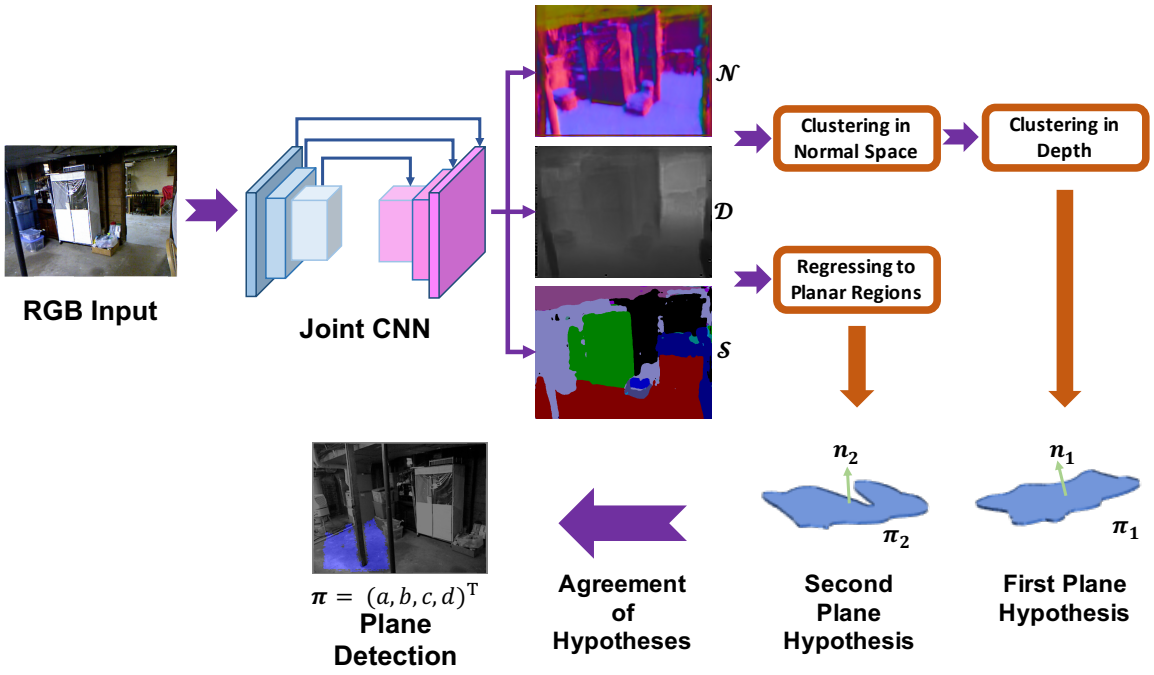


Fig. 5.2 The pipeline of the proposed monocular plane detection. For more details refer to Section 5.4

Point-Plane Associations. Note that the association between inlier 3D point landmarks and 3D planes, useful for the constraint factors described in Section 5.3.3, is extracted from the resulting segmentation masks after the above mentioned step 2. The 3D point is considered as an inlier (hence there is a point-plane constraint factor connected to both landmarks in the factor graph) if the corresponding 2D key-point inside the mask also satisfies the certain geometric perpendicular distance threshold.

5.4.2 Plane Data Association

Once detected, initialized, and added to the map as new landmarks, the planes need to be associated with the prospective detected planes in the incoming frames – the data association problem.

Points of the map are matched and tracked based on ORB descriptors in each frame in a coarse-to-fine pyramid, however matching planes is more robust than feature point matching due to the inherent geometrical nature of infinite planes that was demonstrated in Chapter 3.

In Chapter 3 we only used the geometrical distances of candidate planes and observed plane to match it against the existing planes in map (the angle between

surface normals and the difference between their perpendicular distances from origin), however in this chapter to make data association for planes even more robust in cluttered scenes, when available, we additionally use the detected key-points that lie inside the segmented plane in the image and satisfy the geometric threshold distance (or the set of *inlier points* $\mathcal{P}_{in}$ of a plane) to match the observations. An existing plane in the map and a detected plane in the current frame are deemed to be a match if the number of common inlier key-points is higher than a threshold th_H and also the unit normal vector and distance of them are within certain thresholds. If the number of common key-points is less than another threshold th_L (or zero for feature-deprived regions) meaning that there is no corresponding map plane for the detected plane, the observed plane is initialized and added to the map as a new landmark. By this scheme, unlike Chapter 3, the map can now contain two or more planar regions that might belong to the same infinite plane such as two tables with same height in the office. However, additional constraints on parallel planes are also introduced in these situations according to evidence. For more details about the constraint factors refer to Section 5.3.3.

5.4.3 Multi-Edge Factor for Plane Observation

After successful data association, we can connect observation/constraint factors to the associated variable nodes. Constraint factors, which originally proposed in previous chapters, are reviewed in Section 5.3.3. In this section we propose a novel observation factor, based on a new relative representation of landmarks, connecting plane landmarks and camera (key-frame).

In previous chapters, variable nodes (that need to be estimated) live in a global fixed world coordinates frame and all of the involved observation/constraint factors impose constraints based on the representations in that fixed frame. For instance, using the previous approach, a plane landmark is initialized with respect to the global world frame using the pose ($\mathbf{SE}(3)$ transformation) of the first key-frame which observes that plane. Any further changes or updates that happen in the local map bundle adjustment (factor graph optimization in the back-end) to that first key-frame (observer of the plane landmark) will not have significant direct effect on the representation of the plane node. To overcome this problem and its potential subsequent accumulated errors, we use a relative key-frame formulation (instead of the global frame) to represent each landmark and for instance a plane landmark $\boldsymbol{\pi}_r$ is expressed relative to the first key-frame that observes it (its reference key-frame) denoted by $\mathbf{T}_r^w$, and in this case

the representation of the plane in the global world coordinates system, denoted by $\boldsymbol{\pi}$, is found by the help of plane transformation Equation 3.6 such as the following

$$\boldsymbol{\pi} = \mathbf{T}_r^{w\top} \boldsymbol{\pi}_r \tag{5.6}$$

where $\mathbf{T}_r^w$, the pose of the reference key-frame, is an $\mathbf{SE}(3)$ transformation which takes a point from the world coordinates frame $\mathbf{x}_w$ to the reference key-frame coordinates system $\mathbf{x}_r$ as $\mathbf{x}_r = \mathbf{T}_r^w \mathbf{x}_w$.

Hence for a plane observation $\boldsymbol{\pi}_{obs}$ from a camera pose $\mathbf{T}_c^w$, we propose the following *multi-edge* factor node for measuring the plane observation:

$$f_\pi(\boldsymbol{\pi}_r, \mathbf{T}_r^w, \mathbf{T}_c^w) = \parallel d(\mathbf{T}_c^{r-\top} \boldsymbol{\pi}_r, \boldsymbol{\pi}_{obs}) \parallel^2_{\boldsymbol{\Sigma}_\pi} \tag{5.7}$$

where $\mathbf{T}_c^{r-\top} \boldsymbol{\pi}_r$ is the transformed plane from its reference key-frame to the camera coordinates frame while $\mathbf{T}_c^r$ can be found by composing the $\mathbf{SE}(3)$ transformations such as the following

$$\mathbf{T}_c^r = \mathbf{T}_c^w . (\mathbf{T}_r^w)^{-1} \tag{5.8}$$

Note that $\boldsymbol{\Sigma}_\pi$ is the covariance matrix of the Mahalanobis norm, and d is the difference of the corresponding rotation matrices of the planes in the tangent space at the identity of $\mathbf{SO}(3)$ (see Section 3.3.3).

The graphical interpretation of the proposed multi-edge observation factor is depicted in Fig 5.3 that is showing f_π as a node connected to more than two variable nodes (multi-edge) since the error function associated with the observation factor f_π depends on three variable nodes. All of the previously proposed factor nodes in Chapters 3 and 4 are *binary-edge* factors, since each is connected to exactly two variable nodes of the graph.

5.5 Incorporating Objects with Point-Cloud Reconstruction

As noted previously in Chapter 4, incorporating general objects in the map as bounded quadrics leads to a compact representation of the rough 3D extent and pose (location and orientation) of the object while facilitating elegant data association. With the recent success of deep-learned object detectors, the state-of-the-art object detector such as YOLOv3 [31] can provide object labels and bounding boxes in real-time for general

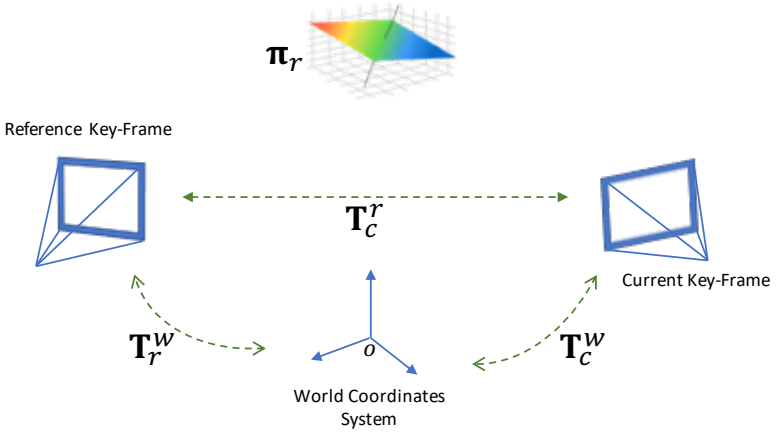


Fig. 5.3 Demonstration of multi-edge factors for planes connected to multi variable nodes (camera key-frames). For more details refer to Section 5.4.3.

objects. The goal of introducing objects in our SLAM is both to increase the accuracy of the localization and to yield a richer semantic map of the scene. While our SLAM proposes a sparse and coarse realization of the objects, wherever the fine 3D point-cloud model reconstruction of each object is available it can be seamlessly incorporated on top of the corresponding quadric and even refines the quadric reconstruction as will be discussed later in 5.5.2.

5.5.1 Object Detection and Matching

In Chapter 4 we used offline precomputed object detections from Faster-RCNN [32], however towards a real-time online SLAM system, in this chapter for real-time detection of objects, we use YOLOv3 [31] trained on COCO dataset [21] that provides detections as axis aligned bounding boxes for common objects. For reliability we consider only detections with 85% or more confidence for initialization and matching process. Note that we obtained similar empirical robust detections with Faster-RCNN [32] in Chapter 4 with higher confidence score (95%).

5.5.1.1 Object Matching Based on Tracked Points

Tracking object detections against the map is not robust enough by relying only on the geometry of the reconstructed quadrics – comparing re-projection errors – and matched key-points in one bounding box, independent of the other detections (the matching scheme in Chapter 4). In particular, object matching is less reliable in cluttered scenes with high-number of overlapping or partially-occluded detections,

which is more evident most of all in KITTI [10] benchmark experiments in Section 5.7. Therefore to find optimum matches for all the detected objects in current frame with the existing objects in the map, we solve the classic optimal assignment problem with Hungarian/Munkres [17] algorithm which is a combinatorial optimization algorithm that solves the assignment problem in polynomial time $O(n^3)$.

The challenge of using this classic assignment algorithm is how to define the appropriate cost matrix. We establish the cost matrix of this algorithm based on the idea of maximizing the number of common robustly matched key-points (2D ORB features) inside the detected bounding boxes. Since we want to solve the minimization problem, we define the non-negative cost matrix such as the following

$$\mathbf{C} = [c_{ij}]_{N \times M} \tag{5.9}$$

$$c_{ij} = K - proj(b_i, q_j) \tag{5.10}$$

where $proj(b_i, q_j)$ gives the number of projected key-points associated with candidate quadric q_j inside the bounding box b_i, and $K = \max_{i,j} proj(b_i, q_j)$ is the maximum number of all of these projected key-points. N and M are the total number of bounding box detections in the current frame and the candidate quadrics of the map for matching, respectively. Candidate quadrics for matching are considered to be the quadrics of the map that are currently in a certain frustum in front of the camera. Note that c_{ij} is the cost of assigning detected bounding box b_i to the candidate quadric q_j in the map.

To reduce the number of mismatches even more, after solving the optimal assignment problem with the proposed cost matrix, the optimum assignment of b_i^* to q_j^* is considered successful if the number of common key-points satisfies a certain high threshold $p(b_i^*, q_j^*) \geq th_{high}$ and the new quadric is initialized in the map if $p(b_i^*, q_j^*) \leq th_{low}$. Assignments with $p(b_i^*, q_j^*)$ values between these thresholds are ignored. These low and high thresholds are determined based on the minimum and average number of matched key-points inside detected bounding boxes. However note that this matching strategy might fail when there are too few key-points extracted at detected objects.

Quadric objects are initialized based on the method described in Chapter 4 after solving an intermediate optimization problem to refine a sphere encloses the inlier 3D points to an ellipsoid that is projected inside the detected bounding box. For more details refer to Section 4.5.1 and Fig 4.5. After successful initialization of the quadric, 2D key-points inside the bounding box are considered as associated key-points of the

quadric and are used in solving the above mentioned matching problem in the next consecutive frames.

5.5.2 Point-Cloud Reconstruction and Shape Prior Factor

So far we efficiently represent generic objects in our SLAM framework by the proposed method for dual quadrics in Chapter 4 that provides a coarse representation for pose (location and orientation) and extent (size and scale) of objects in the scene.

As discussed in Chapter 2, incorporating fine reconstruction of objects directly in the bundle adjustment is expensive in terms of computation and memory. In this section, we integrate high-fidelity reconstruction of objects on top of coarse quadrics to enrich the map, while underlying quadric objects participate in back-end graph optimization. Then we introduce a shape *prior factor* for the quadrics induced by the reconstructed 3D point-cloud of objects, in order to refine the shape of the coarse quadrics, furthermore.

5.5.2.1 Point-Cloud Object Model Reconstruction

It is difficult to estimate the full 3D model of objects from sparse views using purely classic geometric methods. To bypass the ill-posed single-view object reconstruction problem, in this section we train a CNN adopted from Point Set Generation Net [7] to predict (or hallucinate) the accurate 3D shape of objects as point-clouds from single view RGB images. By adopting this CNN for our SLAM system that hallucinates the complete shape of the objects by making several plausible predictions, we reconstruct 3D point-clouds straight from the input RGB image. Using 3D point-cloud models in our SLAM instead of other 3D shape representations such as volumetric grids, preserves us from using different artifacts to keep the model invariant under rigid transformations. We train the CNN on a CAD model repository ShapeNet [2]. ShapeNet is a public large-scale, richly-annotated repository of about 3 million CAD models from several semantic categories. In training, we render 2D images of CAD models from random viewpoints and, to simulate the background in real images, we overlay random scene backgrounds from the SUN dataset [38] on the rendered images.

We demonstrate the efficacy of this approach for outdoor scenes, particularly for general car objects (which the 3D point-cloud models are faithful reconstructions) in KITTI [10] benchmark in Section 5.7.2. The results of some sample 3D point-cloud reconstructions of our trained CNN is illustrated in Fig 5.4. Running alongside with

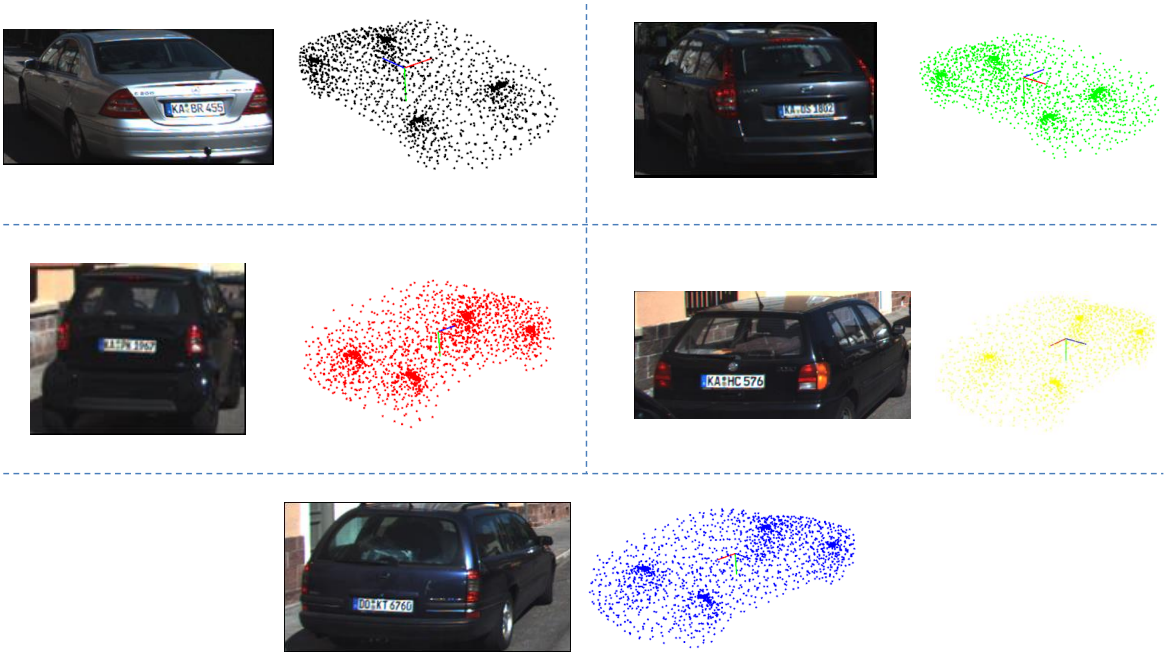


Fig. 5.4 The result of the trained CNN for reconstructing 3D point-cloud models for some example detected cars in KITTI [10] benchmark.

our SLAM system, as demonstrated in Fig 5.5, the CNN takes a detected bounding box of an object as input and generates a point-cloud to represent the 3D shape of the object. However, to ease the training of the CNN, the reconstructed point-cloud is in a normalized scale and canonical pose. To incorporate the point-cloud model into the SLAM system, we need to estimate seven parameters to scale, rotate and translate this point cloud. First we compute the minimum enclosing ellipsoid of the normalized reconstructed point-cloud, and then estimate the parameters by aligning it to the object quadric (ellipsoid) from the SLAM system.

As an expedient approach, we currently pick a single high-quality detected bounding box as the input to the CNN, however, it is not complicated to extend to multiple bounding boxes in future by using a Recurrent Neural Net to fuse information from different bounding boxes, as done in 3D-R2N2 [3].

5.5.2.2 Induced Shape Prior Factor on Quadrics

After registering the hallucinated point-cloud model on the reconstructed quadric from SLAM, as depicted in Fig 5.5, we propose a further prior constraint factor only on the shape (extent) of the quadric in order to further refine the coarse representation within the local map bundle adjustment of our SLAM. This is amenable to the factor graph

framework, thanks to the proposed decomposed quadric representation in Chapter 4. The prior factor (*unary-edge factor*) is a node of the graph that is connected to exactly one variable node and acts as a prior knowledge regarding the estimation of that variable.

We introduce this factor inspired by the idea of *intersection over union* (IoU) loss, however we extend it to 3D case by computing the IoU of the registered enclosing normalized cuboid of the point-cloud $\mathcal{M}$ and enclosing normalized cuboid of the quadric such as following

$$f_{prior}(\mathbf{Q}^*) = \| \ 1 - IoU_{cu}(cuboid(\mathbf{Q}^*), cuboid(\mathcal{M})) \ \|^2_{\sigma_p} \tag{5.11}$$

where *cuboid* is a function that gives the normalized enclosing aligned cuboid of an ellipsoid and σ_p is the associated covariance of the factor in the graph. Note that IoU_{cu} is a real-valued function between 0 and 1 for disjoint and congruent 3D cubes, respectively.

The point-cloud induced shape prior on quadrics refines the quadric reconstructed from multiple-views detected bounding boxes and imposes more straight prior knowledge about the real shape of the objects out of the fine reconstruction through the CNN. For a multitude of reasons, such as partial occlusions or fewer observations from certain view points, the SLAM reconstructed quadric may lack a precise scale in certain directions, while the enclosing quadric of the CNN reconstructed point-cloud contains more information about the real extent of the object, encoded in the size and scale of the ellipsoid.

5.5.3 Multi-Edge Factor for Non-Aligned Object Observation

In Chapter 4 we proposed an object observation factor that encourages dual quadrics to be reconstructed in the map such that their observations from camera key-frames are axis-aligned inscribed ellipses (bounded conic sections) in the detected bounding boxes. To overcome this limitation, in this chapter we propose a novel observation factor for the dual quadrics without enforcing them to be observed as axis-aligned dual conics. Unlike Chapter 4 that imposes a direct constraint on the "difference" of the projected and detected conics (Mahalanobis norm in the space of 5-vectors) and also unlike [35] that uses the Mahalanobis distance of detected and projected bounding boxes (Mahalanobis norm in the space of 4-vectors), which is not robust and penalizes more for large errors and outliers, we propose an observation factor based on *Intersection-over-Union* (IoU)

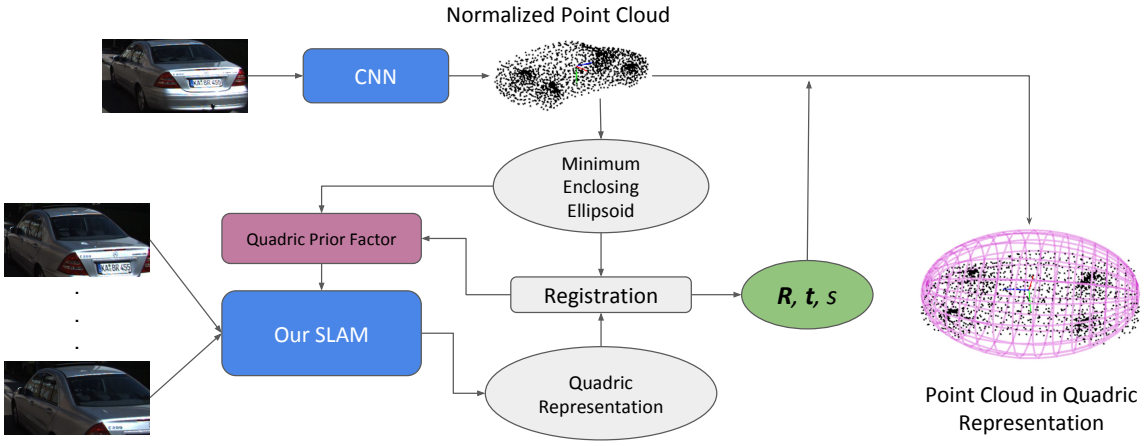


Fig. 5.5 Single-view point-cloud reconstruction imposes a shape prior constraint on a multi-view reconstructed quadric in our system (see section 5.5.2).

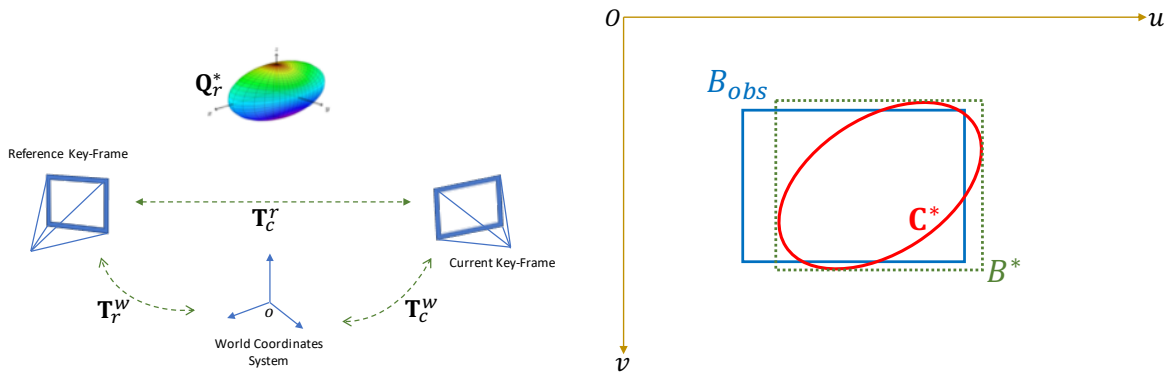
of these bounding boxes that is also weighted according to the *confidence score* of the object detection. This factor provides an inherent capped error which mitigates the effect of outliers, however it implicitly emphasizes the significance of the good initialization of quadrics to have a successful optimization.

In addition to the novel cost function for object observation, similar to plane landmarks in Section 5.4.3, we aim to use the relative reference key-frame $\mathbf{T}_r^w$ to represent the dual quadric associated with the object $\mathbf{Q}_r^*$, instead of the fixed global world coordinates system. In order to do so, we introduce the following multi-edge factor for object observation error between dual quadric $\mathbf{Q}_r^*$ and camera pose $\mathbf{T}_c^w$ as:

$$f_Q(\mathbf{Q}_r^*, \mathbf{T}_r^w, \mathbf{T}_c^w) = \parallel 1 - IoU_{bb}(B^*, B_{obs}) \parallel_{s^{-1}}^2 \tag{5.12}$$

where B_{obs} is the detected bounding box and B^* is the circumscribed bounding box of the projected dual conic $\mathbf{C}^*$. Recall that the projected bounded dual conic (ellipse) is computed with the projection matrix $\mathbf{P} = \mathbf{K}\begin{bmatrix}\mathbf{I}_{3\times3} & \mathbf{0}_{3\times1}\end{bmatrix}\mathbf{T}_c^r$ of the camera with calibration matrix $\mathbf{K}$ and relative pose of $\mathbf{T}_c^r$ calculated by 5.8. For more details regarding projection of dual quadrics refer to Section 4.5.1.

Note that in the above introduced factor 5.12, IoU_{bb} is a real-valued function ranges from 0 to 1 for disjoint and congruent 2D bounding boxes, respectively. We give weight to the significance of this factor in the total cost function of the factor graph by considering the inverse of *confidence score* (obtained from the object detector) of the detected bounding box as the associated covariance of this factor. In another words,



(a) Multi-Edge Observation Factor (b) Observation Error based on IoU

Fig. 5.6 Demonstration of multi-edge factors for quadrics connected to multi variable nodes (camera key-frames) in (a) and the observation error based on the IoU of circumscribed and detected bounding boxes. For more details refer to Section 5.5.3.

we penalize more for the observation error of more confident detections and less for uncertain bounding boxes.

The graphical interpretation of the introduced observation factor is illustrated in Fig 5.6.

5.6 The Pipeline of our SLAM System

The pipeline of our proposed SLAM system is demonstrated in Fig 5.7. We built our SLAM on top of the state-of-the-art sparse ORB-SLAM2 [24] and utilize its front-end for tracking ORB features. All the component of our system are implemented in C++ and we develop the back-end optimization of the system using general framework for graph optimization, g2o [18], in C++ as well.

The majority of the components are already elaborated in previous sections, however we briefly describe the remaining components of the pipeline that are also commonly found in the state-of-the-art sparse SLAM systems.

Our monocular SLAM system is a *key-frame* based SLAM system which is a widely used strategy in visual SLAM systems to avoid expensive local or global map updates for every frames. After adding a new key-frame to the map along with its associated observations of points, planes, and quadrics, and updating the local map of landmarks and key-frames, and also generating the hypotheses for constraint factors, then we optimize the local map (*bundle adjustment*) to update the estimate variables. The updated local map after bundle adjustment is considered as a basis for tracking and

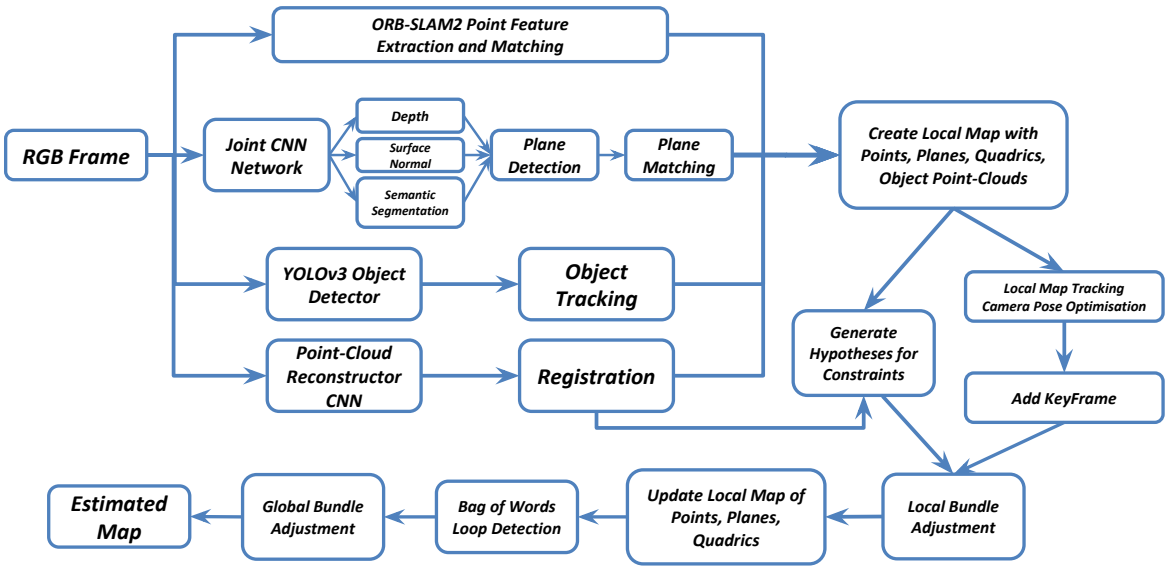


Fig. 5.7 The pipeline of our monocular semantic object-model aware sparse SLAM system.

matching landmarks in the consecutive frames while keeping the local map unchanged. We also perform the global bundle adjustment to update the whole map after detecting loop closures by the front-end of ORB-SLAM2 which uses bag of words [8] based on ORB features.

To accomplish the real-time operation for our system, **(1)** the ORB feature extraction and tracking is performed in one thread, as ORB-SLAM2, and **(2)** plane detection and tracking, **(3)** YOLOv3 [31] object detector, **(4)** local bundle adjust, and **(5)** global bundle adjustment are all performed in separate parallel threads.

5.7 Experiments

We evaluate the performance of our proposed monocular SLAM system extensively on publicly available TUM [34], NYU-v2 [25], and KITTI [10] benchmarks that contain a wide variety of scenes from rich planar low-texture to multi-object common offices and outdoor scenes.

Qualitative and quantitative evaluations are carried out in this section using different mixtures of landmarks and comparisons are presented against point-based monocular ORB-SLAM2 [24] as an ablation study. To study the accuracy of our proposed plane detection scheme in Section 5.4 and the effect of the quality of plane detection in the SLAM framework, we compare the performance of SLAM system with the proposed

plane detector against the performance of the same system with two different detector, one is the state-of-the-art CNN network to segment and regress plane parameters, and the other one is the baseline plane detector that we explain in detail in Section 5.7.1.1.

As shown in this section, including additional proposed factors along with the monocular plane and object detection and tracking schemes improves the accuracy of the camera trajectory (which implies more accurate map reconstruction indirectly), while yielding more semantically meaningful maps containing structural planar entities and semantic objects. The online performance of our SLAM system is demonstrated in the following video links:

- *All the benchmarks:* https://youtu.be/UMWXd4sHONw
- *KITTI Exclusive:* https://youtu.be/QPQqVrvP0dE

5.7.1 TUM and NYU-v2 Benchmarks

Qualitative evaluations for sequences `fr1/xyz` and `fr2/desk` from TUM and for sequences `nyu/office_1b` and `nyu/nyu_office_1` from NYU-v2 are illustrated in Fig. 5.8 that contain diverse landmarks and scenes. Columns (a)-(d) show the image frame with tracked features and possible detected objects, detected and segmented planes, and the reconstructed map from two different viewpoints, respectively. For some low or no texture sequences in TUM and NYU-v2 datasets point-based SLAM system fails to track the camera, however the existent rich planar structure is exploited by our system along with the Manhattan constraints to yield more accurate trajectories and semantically meaningful maps.

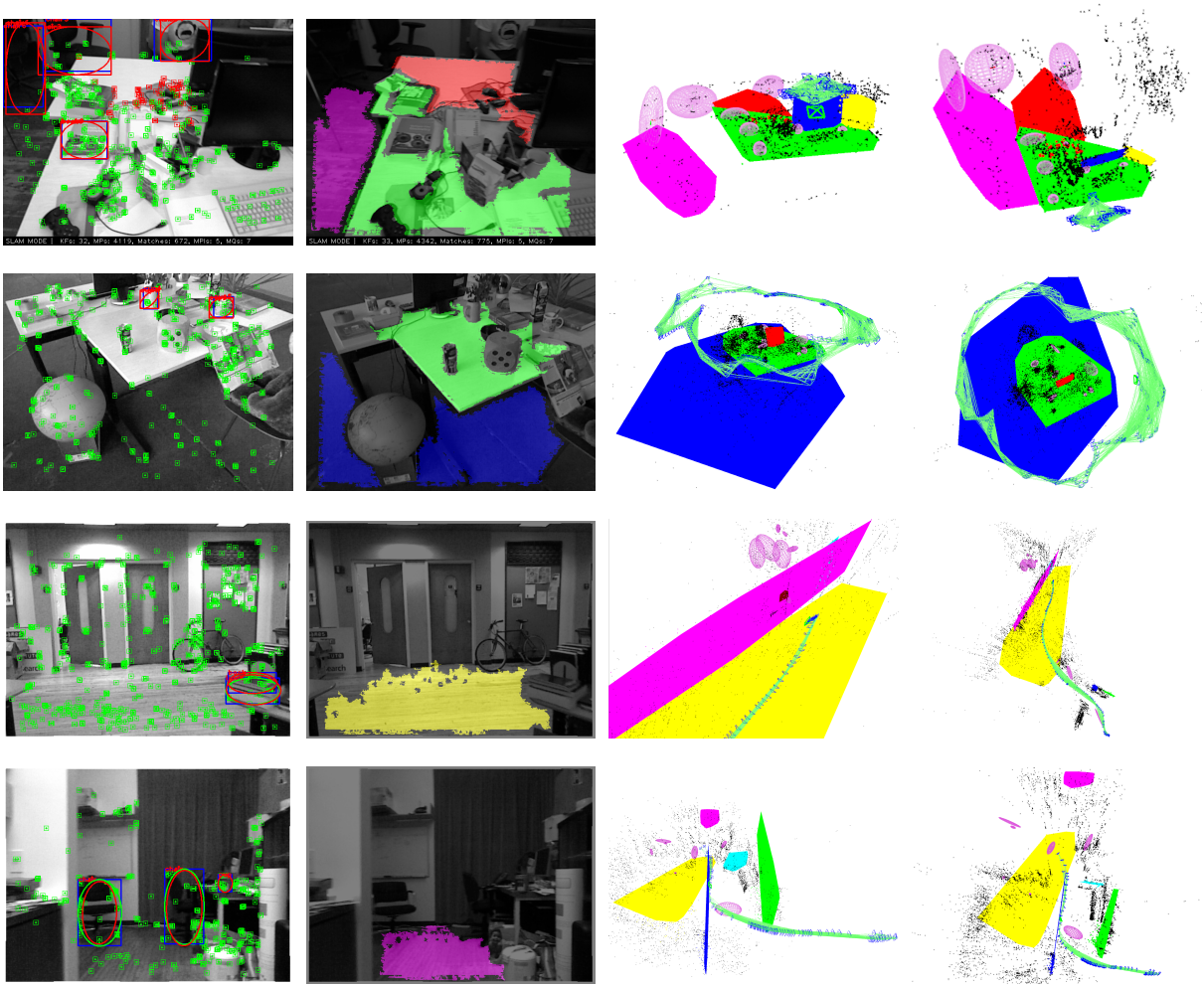
The reconstructed maps are semantically rich and consistent with the ground truth 3D scene, for instance in `fr2/desk`, with the presence of all landmarks and constraints, the map consists of planar monitor orthogonal to the desk, and quadrics corresponding to objects are tangent to the supporting desk, congruous with the real scene. Red ellipses in Fig. 5.8 column (a) are the projection of their corresponding quadric objects in the map.

One of the main reasons for the improved accuracy of camera trajectory and consistency of the global map is the addressing of subtle but extremely important problem of scale drift. In a monocular setting, the estimated scale of the map can change gradually over time. In our system, the consistent metric scale of the plane detections (out of the CNN) and the presence of point-plane constraints allow observations of the absolute scale, which can further be improved by adding priors about the extent of the objects represented as quadrics.

We perform an ablation study to demonstrate the efficacy of introducing various combinations of the proposed semantic landmarks and constraints. The RMSE of Absolute Trajectory Error (ATE) is reported in Table 5.1. Estimated and ground-truth trajectories are aligned using a similarity transformation [15]. In the first case, points are augmented with planes (`PP`) and constraints for inlier points and corresponding planes are included. This already improves the accuracy over baseline and imposing additional Manhattan (plane-plane) constraint in the second case (`PP+M`) improves ATE even further. In these two cases the error is significantly reduced by first exploiting the planar structure of the scene and second by reducing the scale-drift problem, as discussed earlier, using metric information about the planes, induced from trained CNNs.

For the sequences containing common COCO [21] objects, the presence of objects represented by dual quadric landmarks along with the points is explored in the third case (`PO`). This case demonstrates the effectiveness of integrating objects in the SLAM map. Finally, the performance of our full monocular system (`PPO+MS`) is detailed in the last column of Table 5.1 with the presence of all landmarks points, planes, and objects and also Manhattan and supporting/tangency constraints.

Similar to the RGB-D counterpart in Chapter 4, the `PPO+MS` scenario shows an improvement against the baseline in all of the evaluated sequences, in particular for `fr3/long_office` we have seen a significant decline in ATE (18.47%) as a result of a large loop detection in this sequence, where our proposed multiple-edges for observations of planes and quadric objects in key-frames have shown their effectiveness in the global loop closure.



(a) ORB Features and Detected Objects (b) Detected and Segmented Planes (c) Generated Map (Side) (d) Generated Map (Top)

Fig. 5.8 Qualitative results for different sequences from TUM [34] and NYU-v2 [25] benchmarks. The sequences vary from rich planar structures to multi-object cluttered office scenes.

Table 5.1 **Comparison of RMSE of ATE for our monocular SLAM against monocular ORB-SLAM2** [24]. Different `PP`, `PP+M`, `PO`, and `PPO+MS` scenarios mean points-planes only, points-planes + Manhattan constraint, points-objects only, and all of the landmarks with Manhattan and supporting/tangency constraints, respectively. RMSE is reported for ATE in `cm` for 7 sequences in TUM benchmark. Number of included key-frames in the map for each sequence is reported in column #KF. Numbers in bold in each row represent the best performance for each sequence and numbers in [ ] represent the percentage of improvement over ORB-SLAM2. See Section 5.7.1 for more detail.

Dataset	# KF	ORB-SLAM2	PP	PP+M	PO	PPO+MS
`fr1/floor`	125	1.7971	1.6923	**1.6704** [7.05%]	—	—
`fr1/xyz`	30	1.0929	1.0291	0.9802	1.0081	**0.9680** [11.43%]
`fr1/desk`	71	1.3940	1.2961	1.2181	1.2612	**1.2126** [13.01%]
`fr2/xyz`	28	0.2414	0.2213	0.2189	0.2243	**0.2179** [9.72%]
`fr2/rpy`	12	0.3728	0.3356	0.3354	0.3473	**0.3288** [11.79%]
`fr2/desk`	111	0.8019	0.7317	0.7021	0.7098	**0.6677** [16.74%]
`fr3/long_office`	193	1.0697	0.9605	0.9276	0.9234	**0.8721** [18.47%]

5.7.1.1 Plane Detectors Comparison

One of the important elements that affects the performance of the proposed structure and semantic object aware SLAM system is the quality of plane detection and parameter estimation. To study this effect, we compare the reconstructed map along with the estimation error of the camera trajectory of our system operating with different plane detectors shown in Fig 5.9 and Table 5.2.

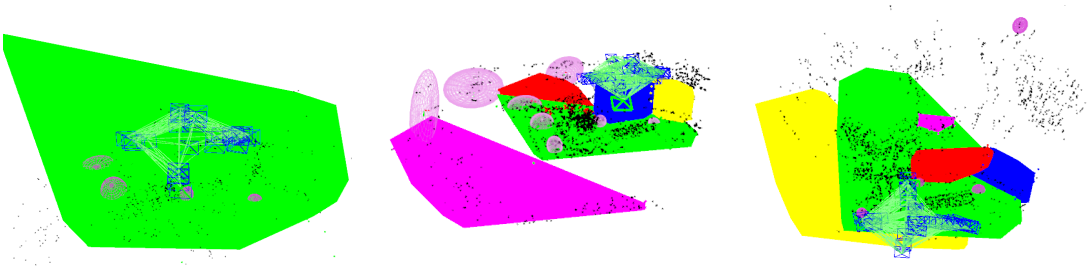
We select two plane detectors to compare against our proposed monocular plane detector in Section 5.4:

Baseline Detector: Since the quality of depth information has a critical impact on the performance of the plane detector and particularly its parameter regression, in this so called *baseline detector*, we substitute the virtual depth (the depth estimation of the trained CNN) in the proposed detector with the ground-truth depth channel to construct the point-cloud. Refer to Section 5.4 for details. Note that the remainder of our SLAM system operating with this baseline detector is unaware of the ground-truth depth and operates in a monocular modality to conduct a fair comparison.

PlaneNet Detector: To compare against a dedicated CNN for detecting planes and regressing its parameters, we use the state-of-the-art CNN-based plane detector, PlaneNet [22], to segment planes and estimate their 3D parameters. End-to-end learned PlaneNet infers a set of planes (segmentation masks and parameters) directly from an RGB image, by the notion of reconstructing piecewise planar depth-maps.

Reconstructed map for sequence `fr1/xyz` from TUM benchmark is shown in Fig. 5.9 for the above mentioned plane detectors incorporated in our system and compared against our proposed one. As seen in Fig 5.9(a) PlaneNet only captures the planar table region successfully and fails for the other regions. The proposed detector captures the monitors on the table shown in column (b), however it misses the third monitor behind and also reconstructs the two same height tables with a slight vertical distance. As shown in Fig. 5.9(c) the baseline plane detector captures the smaller planar regions more accurately and same height tables as one plane, as expected because of using *ground-truth* depth information.

Table 5.2 reports the comparison of these three approaches for plane detection in different sequences of TUM dataset. It can be seen that the baseline detector is the most informative, however the proposed method is better than PlaneNet in most cases.



(a) PlaneNet detector (b) Proposed detector (c) Baseline detector

Fig. 5.9 Qualitative comparison of using different plane detectors in our monocular SLAM system for `fr1/xyz` sequence from TUM benchmark [34]. By using PlaneNet [22] in (a) the only consistent detected plane is the table rendered in green color. The proposed detector in (b) assists our SLAM to reconstruct two same height tables by a slight difference in the height from the floor rendered in magenta color. The baseline detector in (c) is the most informative enables the SLAM to capture more planar regions such as three monitors in the scene. For more details refer to Section 5.7.1.1.

Table 5.2 **RMSE of ATE for our monocular SLAM system using different plane detection methods**. The RMSE is reported in `cm` for four sequences from TUM [34] benchmark. The comparison is carried out with three plane detectors: PlaneNet [22], proposed one in Section 5.4, and depth-aware baseline detector (see Section 5.7.1.1).

Dataset	PlaneNet [22]	Proposed Detector	Baseline
`fr1/xyz`	0.9701	**0.9680**	0.8601
`fr1/desk`	1.2191	**1.2126**	1.0397
`fr2/xyz`	0.2186	**0.2179**	0.2061
`fr1/floor`	**1.6562**	1.6704	1.4074

5.7.2 KITTI Benchmark

To demonstrate the efficacy of our proposed multi-edge object observation factor, object matching, and also shape prior factor induced from incorporated point-cloud (reconstructed by CNN from single-view) in our SLAM system, we evaluate our system on KITTI benchmark. The high-fidelity reconstructed cars in KITTI, shown in Fig 5.4, and overlapping multiple detections in each frame, makes this benchmark challenging for evaluating our system with mentioned factors and matching schemes. For more reliable frame-to-frame tracking, we use the stereo variant of ORB-SLAM2 to match 3D points, however object detection and plane detection and parameters estimation are still carried out in a monocular modality.

The reconstructed map with quadric objects and incorporated point-clouds (see Section 5.5.2) is illustrated for `KITTI-7` from different view-points in Fig 5.10. The instances of different cars are rendered in different colors. The reconstructed car models for two varieties, `sedan` and `hatchback`, beside the road are demonstrated in Fig 5.11 and also the estimated trajectory after loop closure from top and side views is rendered in Fig 5.12. Note that in these experiments all the object related factors such as multi-edge observations and point-cloud induced shape priors, are effective during local and global map bundle adjustment.

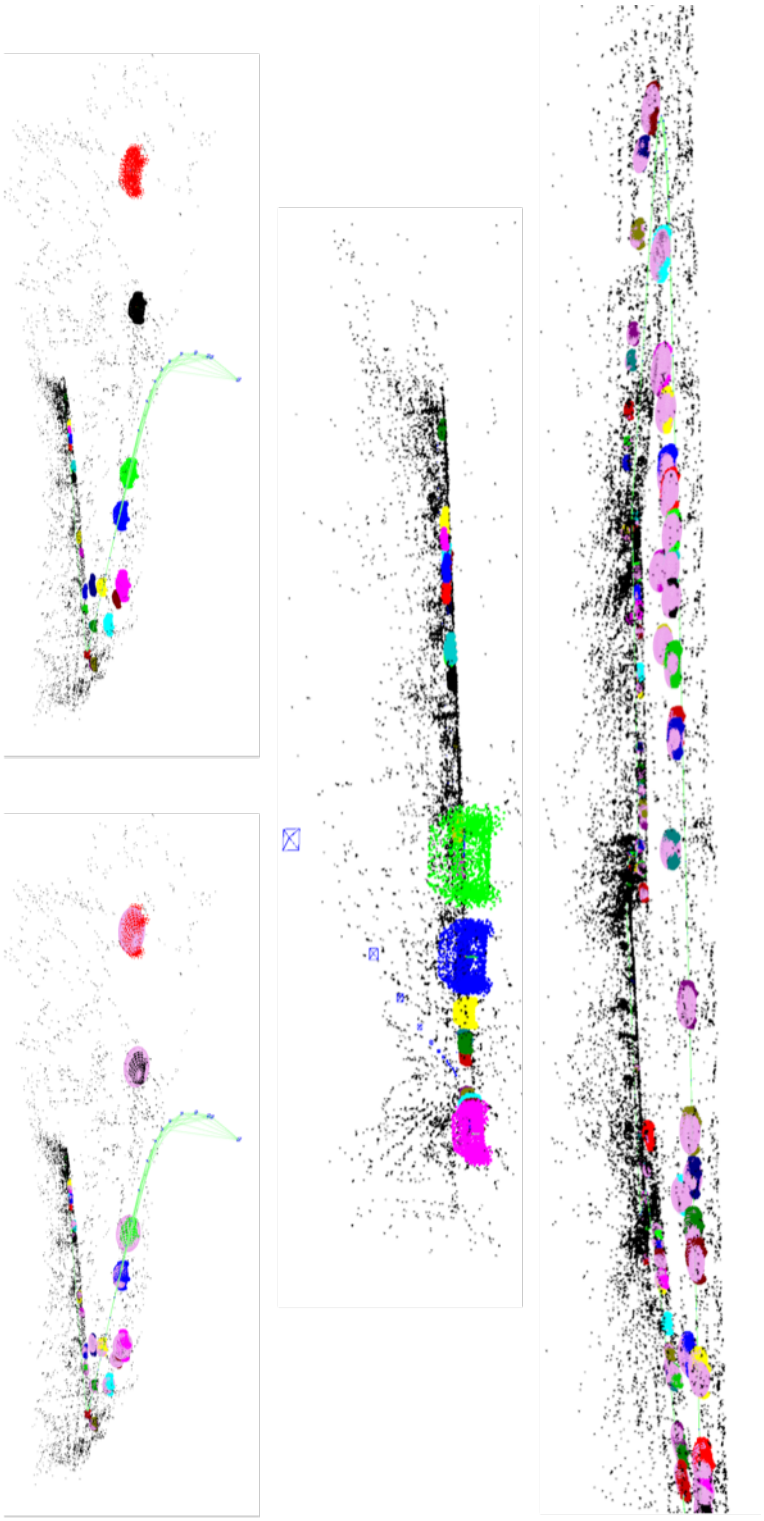


Fig. 5.10 Reconstructed map for KITTI-7 sequence with our SLAM system from different side viewpoints with/without rendering quadrics. Proposed object observation and point-cloud-induced prior factors are effective in this reconstruction.

Fig. 5.11 The reconstructed models for a `sedan` and `hatchback` car parked beside the road in sequence `KITTI-7` are rendered along with the quadrics. Note that again all object related factors are effective during the operation of the SLAM system.

Fig. 5.12 Reconstructed map and estimated trajectory for KITTI-7 sequence with our SLAM system from top and side viewpoints after loop closure. Proposed object observation and point-cloud-induced prior factors are effective in this reconstruction.

5.8 Conclusions

This chapter introduced a monocular SLAM system that can incorporate learned priors in terms of plane and object models in an online real-time capable system. We show that introducing these quantities in a SLAM framework allows for more accurate camera tracking and a richer map representation without huge computational cost. This chapter also makes a case for using deep-learning to improve the performance of traditional SLAM techniques by introducing higher level learned structural entities and priors in terms of planes and objects.

Bibliography

[1] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*, 2012.

[2] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[3] C. B. Choy, D. Xu, J. Gwak, K. Chen, and S. Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *European conference on computer vision*, pages 628–644. Springer, 2016.

[4] G. Cross and A. Zisserman. Quadric reconstruction from dual-space geometry. In *Computer Vision, 1998. Sixth International Conference on*, pages 25–31. IEEE, 1998.

[5] J. Engel, T. Schöps, and D. Cremers. Lsd-slam: Large-scale direct monocular slam. In *European Conference on Computer Vision*, pages 834–849. Springer, 2014.

[6] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[7] H. Fan, H. Su, and L. J. Guibas. A point set generation network for 3d object reconstruction from a single image. In *CVPR*, volume 2, page 6, 2017.

[8] D. Gálvez-López and J. D. Tardos. Bags of binary words for fast place recognition in image sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, 2012.

[9] P. Gay, V. Bansal, C. Rubino, and A. D. Bue. Probabilistic structure from motion with objects (psfmo). In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 3094–3103, Oct 2017. doi: 10.1109/ICCV.2017.334.

[10] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[11] G. Grisetti, R. Kummerle, C. Stachniss, and W. Burgard. A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010.

[12] D. Gálvez-López, M. Salas, J. D. Tardós, and J. Montiel. Real-time monocular object slam. *Robotics and Autonomous Systems*, 75:435 – 449, 2016. ISSN 0921-8890.

[13] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003. ISBN 0521540518.

[14] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *Computer vision, 2009 IEEE 12th international conference on*, pages 1849–1856. IEEE, 2009.

[15] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642, Apr 1987.

[16] M. Hsiao, E. Westman, G. Zhang, and M. Kaess. Keyframe-based dense planar slam. In *IEEE International Conference on Robotics and Automation, ICRA, Singapore*, 2017.

[17] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. doi: 10.1002/nav.3800020109.

[18] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 3607–3613. IEEE, 2011.

[19] C.-Y. Lee, V. Badrinarayanan, T. Malisiewicz, and A. Rabinovich. Roomnet: End-to-end room layout estimation. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 4875–4884. IEEE, 2017.

[20] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing ikea objects: Fine pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2992–2999, 2013.

[21] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[22] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa. PlaneNet: Piecewise Planar Reconstruction from a Single RGB Image. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[23] J. McCormac, A. Handa, A. Davison, and S. Leutenegger. Semanticfusion: Dense 3d semantic mapping with convolutional neural networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4628–4635. IEEE, 2017.

[24] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[25] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[26] V. Nekrasov, T. Dharmasiri, A. Spek, T. Drummond, C. Shen, and I. Reid. Real-time joint semantic segmentation and depth estimation using asymmetric annotations. *arXiv preprint arXiv:1809.04766*, 2018.

[27] 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 *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*, pages 127–136. IEEE, 2011.

[28] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2320–2327. IEEE, 2011.

[29] S. Pillai and J. Leonard. Monocular slam supported object recognition. *Robotics: Science and Systems*, 2015.

[30] V. A. Prisacariu, O. Kähler, S. Golodetz, M. Sapienza, T. Cavallari, P. H. Torr, and D. W. Murray. Infinitam v3: A framework for large-scale 3d reconstruction with loop closure. *arXiv preprint arXiv:1708.00783*, 2017.

[31] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *arXiv*, 2018.

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[33] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. J. Kelly, and A. J. Davison. SLAM++: simultaneous localisation and mapping at the level of objects. In *2013 IEEE Conference on Computer Vision and Pattern Recognition, Portland, OR, USA, June 23-28, 2013*, pages 1352–1359, 2013. doi: 10.1109/CVPR.2013.178. URL https://doi.org/10.1109/CVPR.2013.178.

[34] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.

[35] N. Sünderhauf and M. Milford. Dual Quadrics from Object Detection Bounding-Boxes as Landmark Representations in SLAM. *arXiv preprints arXiv:1708.00965*, Aug. 2017.

[36] N. Sünderhauf, T. T. Pham, Y. Latif, M. Milford, and I. Reid. Meaningful maps with object-oriented semantic mapping. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 5079–5085. IEEE, 2017.

[37] B. Tekin, S. N. Sinha, and P. Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 292–301, 2018.

[38] J. Xiao, J. Hays, K. A. Ehinger, A. Oliva, and A. Torralba. Sun database: Large-scale scene recognition from abbey to zoo. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3485–3492, June 2010. doi: 10.1109/CVPR.2010.5539970.

[39] S. Yang, Y. Song, M. Kaess, and S. Scherer. Pop-up slam: Semantic monocular plane slam for low-texture environments. In *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pages 1222–1229. IEEE, 2016.

Chapter 6

Conclusion and Future Directions

This chapter provides a summary of the research in this thesis and a brief discussion of some potential future directions.

6.1 Conclusion

Simultaneous Localization and Mapping (SLAM) is one of the fundamental and well-studied problems in robotics and computer vision. There has been decades of research in the establishment of general frameworks and formulation of this problem as a purely geometric method to reconstruct the map of the scene as geometric entities. After the recent success of semantic scene understanding, particularly with impressive modern tools such as deep-learning and Convolutional Neural Networks (CNNs), building pure geometric maps is not sufficient for high-level tasks in robotics and computer vision, such as intelligent interactions between robots and humans, autonomous vehicles, and augmented reality (AR) applications. *Semantic SLAM* aims to combine geometric reconstruction and visual scene understanding in one framework and answer more complicated queries about the semantics and objects reconstructed in the scene.

This thesis aimed to integrate semantics, structures, and objects on top of the traditional point-based SLAM in one single framework in order to yield more accurate camera trajectories and mappings by directly involving these high-level entities as independent landmarks in the process of bundle-adjustment optimization, while producing semantically meaningful maps in near real-time with negligible computational and memory overload. Towards this goal, first we integrated dominant structure of the scene as planes to the SLAM framework. To do so, we proposed factor graph compatible mathematical representations for these landmarks and consequently, we

proposed regularizer for imposing global structural priors on points by using point-plane constraints, and also imposing structured environment hypothesis by introducing Manhattan world assumption as new constraint factors in the factor graph formulation of the sparse point-based SLAM. The result of this integration was a sparse point-plane SLAM system that captured large feature-deprived planar regions along with points in real-time from the RGB-D input modality.

Following by the introduction of structural entities to our system, we introduced objects directly to the semantic SLAM framework by proposing coarse representations amenable to factor graph context. We proposed a novel representation and approximate incremental update rules for dual quadrics as coarse representation of objects that captures the rough pose and extent of the generic objects in the scene. By this representation, we merged the semantic and geometric clues out of the deep-learned object detections into the geometric pipeline of our SLAM system. That representation permitted the integration of another high-level semantically meaningful constraint into our system. In the structured man-made environments, in addition to Manhattan assumption, commonly we observe supporting relationships for planar regions, such as objects supported by tables or floors and roads. We imposed supporting affordance by considering tangency constraint between dual quadrics and their supporting planes. The result of these integrations was a real-time RGB-D structure aware SLAM system using dual quadrics and planes.

In the last leap towards the aim of this thesis to represent a general-purpose monocular semantic SLAM, we embedded high-fidelity CNN-based point-cloud reconstructions of objects on top of their coarse representations in the map. Furthermore, we proposed new observation factors to facilitate the final goal of this thesis. In order to extricate our system from depth-dependent plane detections, we proposed a new pipeline in the front-end of our system to detect, segment, and estimate the parameters of the planes from the redundant information of CNN-based depth, surface normal, and semantic segmentation predictions. Moreover, we involved the prior information regarding the shape of the objects, induced from the point-cloud fine reconstructions, in our SLAM framework as prior factors to further refine the coarse reconstructions. The result of all these new landmarks and observations/constraints was the real-time monocular structure and object model aware sparse SLAM system.

We extensively evaluated our proposed systems in almost all publicly available SLAM benchmarks. In addition to more semantically meaningful maps, illustrated in qualitative results, we demonstrated that direct inclusion of the proposed high-level

structures and objects as independent landmarks enriches the map semantically while yielding more accurate localizations and camera trajectories - implicitly meaning more reliable maps.

6.2 Future Directions

In the following, we discuss some potential future directions of our research in this thesis.

6.2.1 Local Dense Priors on Planar Regions

A common approach in enforcing local geometric priors in a dense SLAM is adding deep-learned depth or surface normal predictions as regularizer to the energy function minimization problem. While adding more global priors is more challenging in dense map representation, we incorporated them in our sparse SLAM system. To do so in the monocular modality, we detected planar structures by exploiting the redundant information in separate depth, surface normal, and semantic segmentation outputs of the trained CNN and outperformed the state-of-the-art dedicated RGB-only plane detector ([9]) in estimating plane parameters. However future work can explore this direction to train a dedicated CNN to discover *dense per-pixel plane detections*. The advantage of this per-pixel dense plane prediction is combining the information of depth and surface normal to provide the *tangent planes* to each point directly from raw imagery. These tangent surfaces can be used in to impose local geometric priors on sparse points without in-painting the sparse map. Furthermore these dense plane predictions can be utilized to come up with more accurate plane parameters estimation and plane mask segmentations.

The preliminary results of a trained CNN for dense per-pixel plane prediction is illustrated in Fig 6.1. We adopted this CNN from ([17]) and trained the modified architecture for dense predictions on the synthetic SYNTHIA benchmark ([13]).

6.2.2 3D Object Pose Priors

The proposed semantic SLAM framework in this thesis based on factor graphs, facilitates the intuitive extension of the system with more complicated constraints, geometric or semantic, and also prior knowledge about landmarks or general scene. One of these priors which can be exploited immediately by the future work is the introduction of 3D

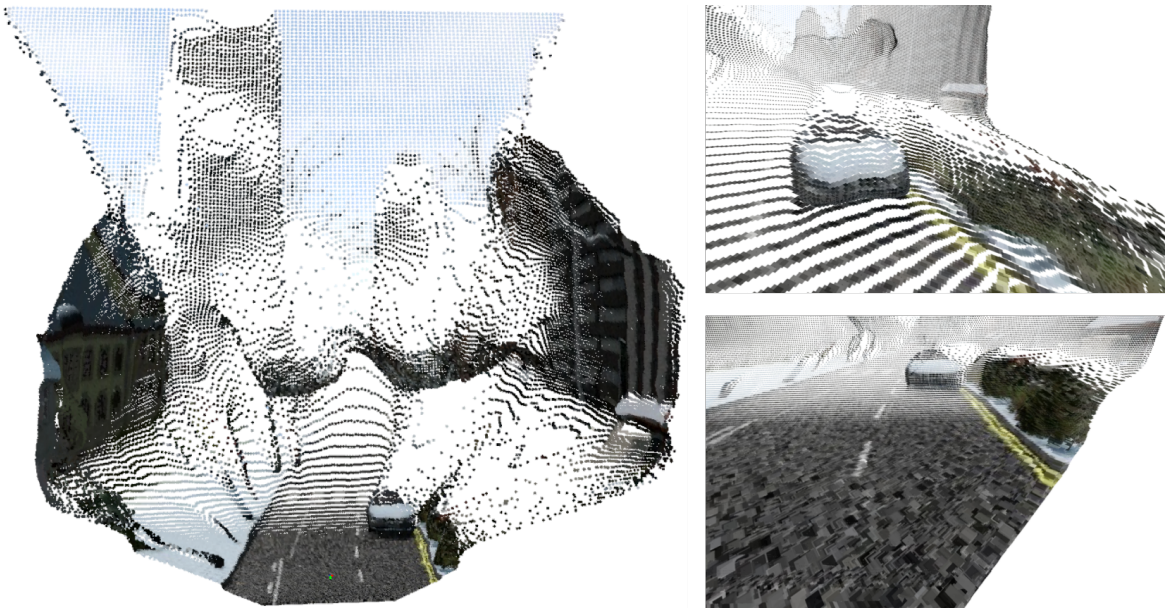


Fig. 6.1 The preliminary predictions of a trained CNN for dense per-pixel plane prediction is illustrated for a sample scene from the synthetic SYNTHIA benchmark [13]. We adopted the architecture of the CNN from [17] and modified it in order to model dense per-pixel estimations. The predicted plane patches for each 3D point is rendered which shows the tangent planar surfaces to the scene, for instance the road and the car.

orientation or pose of the observed objects in our SLAM. For instance, we registered the fine point cloud reconstruction of the object on top of its associated quadric in the map based on aligning the enclosing quadric of the point-cloud and the reconstructed quadric from the SLAM pipeline. However due to a multitude of reasons, such as insufficient observations of objects or partial occlusions, sometimes the embedded fine reconstruction is not congruent with the ground-truth map or physics of the scene. For example the incorporated fine model of the car which is illustrated in Fig 6.2 is not feasible in the real-world, and the reason for this infeasible pose of the ellipsoid in the map is insufficient observation of the detected car in successive frames.

Recent accomplishments in 3D object pose discovery from single RGB images ([11, 12, 5, 10, 16]) allows the future work to exploit the graph-based structure of our SLAM and impose prior information on top of the coarse object representation by attaching prior factors to object variable nodes in the graph, as illustrated in Fig 6.3.

Fig. 6.2 An example of the infeasible car pose in the reconstructed map of the sequence from KITTI [3]. In our SLAM we register the fine point-cloud model of the object on top of the reconstructed coarse dual quadric based on the alignment of it with the enclosing quadric of the point-cloud. Due to a multitude of reasons such as insufficient observations in successive frames, the pose of the reconstructed dual quadric sometimes is not congruent with the ground-truth scene.

6.2.3 Object Latent Space in Factor Graph

We proposed our SLAM systems in this thesis in the category of semantic SLAM frameworks which we called *direct semantic SLAMs*. It means the objects and structures are integrated directly in the bundle-adjustment of SLAM, in addition to the tracking phase in the front-end, that yields more globally consistent large scale maps. One of the key enablers of this achievement is the deploying of coarse representations (dual quadrics) for objects. We integrated fine reconstructions later in a parallel pipeline on top of the quadrics, however those fine point-cloud models are not involved in the back-end bundle-adjustment.

Recent achievements in 3D model reconstruction or hallucination of objects from single image using CNNs ([7, 6]) provide proper tools to represent objects in high-dimensional latent spaces. Using these latent representations directly in the factor graph formulation of a SLAM framework can be a potential future direction of our work. However real-time incremental optimization of the factor graph with the presence of

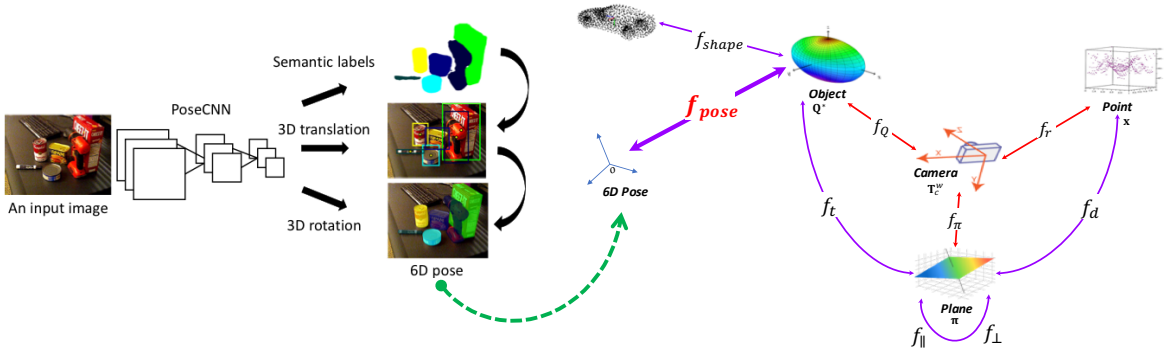


Fig. 6.3 The recent achievements in 6D pose prediction of objects from single RGB images, such as [16] which we adopted the left image from, encourage the future work to impose additional prior factor for the 6D pose of the object, separate from the shape as shown in the figure, thanks to our proposed decoupled representation for coarse dual quadric objects. This unary factor encodes the priori knowledge about the object pose which comes from a trained CNN.

variables living in high-dimensional manifolds is interesting yet not a widely addressed problem.

6.2.4 Dynamic Objects in the Scene

Due to the same mentioned reason for using coarse representations of objects, future work can explore the possibility of introducing *dynamics* on factor graphs and its nodes representing objects. Recent works such as ([14, 1, 15]) try to address the modeling of dynamic objects in dense SLAM problem, however they are limited to small objects in room-scale environments. Imposing dynamics on quadrics as a whole entity encapsulating the pose and extent of the object can be exploited to estimate the state of the objects in the factor graph along with their dynamic behavior. An immediate benchmark for evaluating this scenario can be modeling cars with their associated dynamics in outdoor scenes.

6.2.5 From Scene Graph to Factor Graph

In our thesis, we proposed new landmarks and constraints as a graph which is optimized in the back-end of SLAM and accordingly we proposed appropriate components in the front-end to extract the topology of the graph with reasonable initializations from the raw input imagery. The hypothesis generation for introducing constraints, particularly between landmarks (such as Manhattan assumption, supporting affordances), was

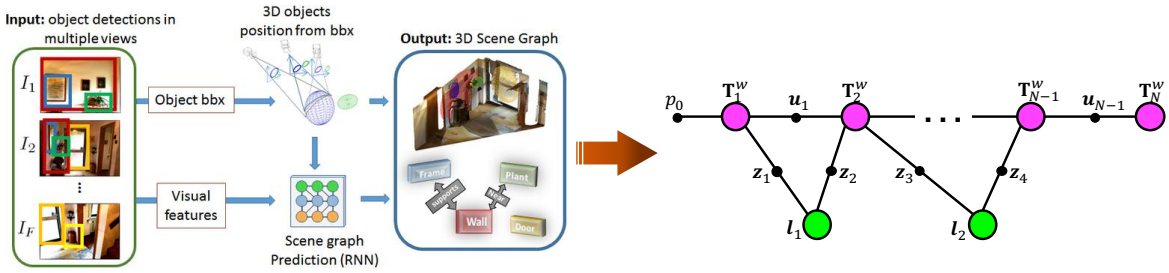


Fig. 6.4 The future work can utilize the inferred high-level scene graph from a trained deep net, such as [2] which we adopted the left image from, in order to identify the topology of the factor graph in the back-end of SLAM problem and introduce geometric or semantic constraints based on this scene graph, instead of using heuristics or ad-hoc mechanisms to generate hypotheses. The scene graph encodes high-level geometric and semantic relationships.

based on geometric or semantic state of the map and some ad-hoc heuristics. Recently some works such as ([2, 8, 4]) attempt to encode the spatial and semantic relationships of the objects or structural entities of the scene in a so called *scene graph*, as shown in Fig 6.4. One of the intriguing future direction of our work is to investigate the possibilities of translating those scene graphs directly to factor graphs in the back-end of the SLAM problem. In that case, instead of using heuristics to estimate the topology of the factor graph, connections between variables and factor nodes will be established more robustly based on the trained CNNs for encoding relational scene graphs.

Bibliography

[1] I. A. Bârsan, P. Liu, M. Pollefeys, and A. Geiger. Robust dense mapping for large-scale dynamic environments. In *International Conference on Robotics and Automation (ICRA)*, 2018.

[2] P. Gay, S. James, and A. D. Bue. Visual graphs from motion (vgfm): Scene understanding with object geometry reasoning. *ACCV*, 2018.

[3] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[4] R. Herzig, M. Raboh, G. Chechik, J. Berant, and A. Globerson. Mapping images to scene graphs with permutation-invariant prediction. In *Advances in Neural Information Processing Systems (NIPS)*, 2018.

[5] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 1530–1538, 2017.

[6] S. Lan, R. Yu, G. Yu, and L. S. Davis. Modeling local geometric structure of 3d point clouds using geo-cnn. *arXiv preprint arXiv:1811.07782*, 2018.

[7] K. Li, T. Pham, H. Zhan, and I. Reid. Efficient dense point cloud object reconstruction using deformation vector fields. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 497–513, 2018.

[8] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang. Scene graph generation from objects, phrases and region captions. In *Proceedings of the IEEE International Conference on Computer Vision*, 2017.

[9] C. Liu, J. Yang, D. Ceylan, E. Yumer, and Y. Furukawa. Planenet: Piecewise planar reconstruction from a single rgb image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2579–2588, 2018.

[10] S. Mahendran, H. Ali, and R. Vidal. A mixed classification-regression framework for 3d pose estimation from 2d images. In *BMVC*, 2018.

[11] F. Manhardt, W. Kehl, and A. Gaidon. Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape. *arXiv preprint arXiv:1812.02781*, 2018.

[12] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka. 3d bounding box estimation using deep learning and geometry. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5632–5640, 2017.

[13] G. Ros, L. Sellart, J. Materzynska, D. Vázquez, and A. M. López. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016.

[14] M. Rünz and L. Agapito. Maskfusion: Real-time recognition, tracking and reconstruction of multiple moving objects. *ISMAR 2018*, 2018.

[15] R. Scona, M. Jaimez, Y. R. Petillot, M. Fallon, and D. Cremers. Staticfusion: Background reconstruction for dense rgb-d slam in dynamic environments. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9, 2018.

[16] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *RSS*, 2018.

[17] F. Yang and Z. Zhou. Recovering 3d planes from a single image via convolutional neural networks. In *The European Conference on Computer Vision (ECCV)*, September 2018.