TUM School of Computation, Information and Technology
Technische Universität München

# Semantic and Geometric Priors for Monocular SLAM

**Nikolas Brasch**

Vollständiger Abdruck der von der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

**Doktors der Naturwissenschaften (Dr. rer. nat.)**

genehmigten Dissertation.

**Vorsitz:**

Prof. Dr. Cristina Piazza

**Prüfende der Dissertation:**

1. Priv.-Doz. Dr. Federico Tombari
2. Prof. Dr. Vasileios Belagiannis

Die Dissertation wurde am 15.05.2025 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 06.08.2025 angenommen.

# Abstract

Monocular SLAM methods aim at estimating the camera trajectory from an input video stream, while jointly building a map of the environment. They are used in many autonomous systems and augmented reality devices that move through and interact with the environment. Research and development over the last decades has produced strong real-time capable systems that are used in a wide range of products. However, these systems still have some important limitations. Most methods assume that the world is static, which can lead to reduced performance or even tracking failure, if there are too many dynamic objects present in the scene. Most real-time methods further depend on texture rich environments in order to extract a sufficient number of clues for the pose estimation. In this thesis we explore the use of different types of prior information to mitigate these limitations. First, we propose a probabilistic map point model that includes predicted semantic information about the classes of the objects in the scene as a prior into the estimation of the depth and inlier ratio of each map point in order to reduce the impact of dynamic objects on the pose estimation. Second, we show that using geometric priors in the form of normal maps predicted by a deep neural network can provide valuable information to improve the rotation estimation in man-made environments with regular but low-textured geometries such as walls, floors and ceilings.

# Zusammenfassung

Monokulare SLAM-Methoden zielen darauf ab, die Trajektorie einer Kamera aus einem Eingangsvideostroms zu schätzen, während gleichzeitig eine Karte der Umgebung erstellt wird. Sie sind ein wichtiger Baustein für autonome Systeme und Augmented-Reality-Geräte, die sich durch die Umgebung bewegen und mit ihr interagieren. Die Forschung und Entwicklung der letzten Jahrzehnte hat leistungsfähige und echtzeitfähige Systeme hervorgebracht, die in einer breiten Palette von Produkten eingesetzt werden. Diese Systeme haben jedoch noch einige wichtige Einschränkungen. Die meisten Methoden gehen davon aus, dass die Welt statisch ist, was zu Leistungseinbußen oder sogar zum Versagen der Positionsbestimmung führen kann, wenn zu viele dynamische Objekte in der Szene vorhanden sind. Die meisten Echtzeit-Methoden sind außerdem auf texturreiche Umgebungen angewiesen, um eine ausreichende Anzahl von Hinweisen für die Posenschätzung zu erhalten. In dieser Arbeit erforschen wir die Verwendung verschiedener Arten von a-priori Informationen, um diese Einschränkungen abzumildern. Erstens schlagen wir ein probabilistisches Modell für Kartenpunkte vor, das vorhergesagte semantische Informationen über die Klassen der Objekte in der Szene als Vorabinformation in die Schätzung der Tiefe und des Inlier-Verhältnisses jedes Kartenpunkts einbezieht, um den Einfluss dynamischer Objekte auf die Posenschätzung zu verringern. Zweitens zeigen wir, dass die Verwendung geometrischer Vorabinformationen in Form von Normalen, die von einem tiefen neuronalen Netzwerk prädiziert werden, wertvolle Informationen zur Verbesserung der Rotationsschätzung in vom Menschen geschaffenen Umgebungen mit regelmäßigen, aber wenig strukturierten Geometrien wie Wänden, Böden und Decken liefern kann.

# Contents

# 1 Introduction

## 1.1 Motivation

When we humans move around in the world we often rely on some form of map of the environment for navigation. It might be the abstract knowledge about the layout and the positions of the furniture in the room we are currently in, or it might be the map of a city or a building that we are looking at. In order to navigate to a certain goal within the map we also need to know our current position and orientation in the map. However, often we do not have a map of the environment, because we have never been at a place before and there is no public map available. In these cases we build our own abstract map of the new environment in order to find our way.

These two tasks of building a map of the environment and localizing oneself within the map are also essential capabilities of smart devices and autonomous systems such as augmented reality platforms and mobile robots. Augmented reality devices need to track the motion in physical space to align it to the virtual space with high accuracy and frame rate. Autonomous systems rely on localization to leverage map information that extends their field of view beyond the sensor range and any occlusions for planning and navigation. In addition, in their control loop autonomous systems often depend on accurate pose estimation in order to interact with the environment. Navigation and interaction often also require a high resolution reconstruction of the scene to enable safe obstacle avoidance and complex and delicate manipulations. To build such a high quality map from multiple observations over time accurate pose estimates are needed. In most cases the odometry, the measuring of the relative ego motion over time, either based on the accumulation of the executed motions or through other internal motion sensors is not precise enough for many applications and suffers from accumulated pose errors. Therefore, often additional remote sensors are installed to build an exact map of the environment and estimate an accurate relative pose to it.

**Simultaneous Localization and Mapping (SLAM)** systems aim to track and localize a platform moving through an unknown environment by simultaneously building a map of it.

Many robotic platforms rely on remote depth sensors to sense the environment around them, the direct measurements of the geometry facilitate the building of a map. In some systems LiDAR sensors are used that directly measure the distance to the environment. The sensor design and sequential scanning often lead to a limited spatial resolution and the high costs limit their use to a small number of devices. In comparison cameras are a cheaper sensor that provides a high spatial resolution. Many current smart devices and autonomous platforms are already equipped with one or more of them to take pictures and sense the environment.

This thesis is focused on visual SLAM, more specifically on monocular SLAM, which are systems that only use a single camera sensor, representing a frequently used minimal setup. However, most of the challenges and the proposed solutions can be directly extended to settings with multiple cameras as well as different sensors.

## Challenges

Visual SLAM and related tasks such as visual odometry, localization and mapping have seen tremendous research activity in the past years. This has lead to an increase in system performance enabling solutions in a wide range of tasks and domains. However, many of these domains are relatively controlled environments. There still exist several limitations that prevent the extension of the operational domain into more challenging environments and use cases. Some of the most common challenges addressed in the literature are the limited robustness to environments with lighting changes, high sensor noise or lack of texture, the operation in highly dynamic environments with many or large independently moving objects and the improvement of the processing speed on limited hardware.

Monocular systems have their own set of limitations compared to multi-camera, active depth sensor or visual-inertial systems. While active and multi-view systems can extract depth measurements at a specific point in time, monocular algorithms need to combine temporal measurements to estimate depth and are therefore more susceptible to dynamic changes. Furthermore, due to the ambiguity of size and distance in projected images, without a direct metric measurement such as depth or the acceleration or a known transformation between multiple sensors, monocular systems are not able to extract the true scale of the world.

In the following we will look at three particular challenges that are tackled within the scope of this thesis.

### Feature Quantity vs Quality

For localization we ideally want a map with just the right amount of well distributed features that can be computed efficiently and are unique so they can be match robustly in order to build correspondences between different images to estimate their relative poses with minimum resources.

However, in complex environments features are often cluttered and in dynamic scenes foreground objects can occlude large parts of the static environment only leaving a few features visible. Additionally, man-made indoor environments often consist of large homogeneous surfaces, likes walls, floors and ceilings, with little texture and unevenly distributed features, posing a challenge for visual SLAM systems, especially for monocular methods.

Traditionally, visual SLAM methods can be divided into descriptive and direct methods based on the type of features used. Descriptive methods leverage visual features consisting of keypoints and descriptors that enable feature matching to align camera poses and triangulate 3d landmarks by minimizing the re-projection error. Direct meth-

ods on the other hand avoid the computation of a descriptor and instead directly optimize the alignment of the measurements corresponding to the image pixels, starting from an initial pose estimate. While direct methods can leverage areas with soft gradients, point based descriptive methods require distinct corners and often struggle to find a sufficient number of landmarks. In addition to descriptive points also lines features, that are more common in these environments, can be used to provide additional clues. While descriptive features allow global matching of the descriptors to find correspondences even under large view-point changes, direct methods assume small motions and need a good pose initialization to find the optimal solution. However, the extraction of robust descriptive features requires well textured environments. Direct features on the other hand can also be extracted from areas with small local gradients and provide valuable information in low-textured scenes.

The recent success of deep learning has lead to methods for monocular depth and normal estimation that allow the prediction of dense geometric information from a single image. This enables monocular methods to leverage additional geometric features as it is done in RGB-D based methods.

In this thesis we will combine descriptive and direct features with a general formulation in a joint optimization framework and show synergies that provide advantages in challenging environments with a lack of visual features. Additionally we show how predicted normals can provide valuable additional features for pose estimation in structured environments.

### The static world assumption

In order to estimate the camera poses with respect to the world most algorithms assume that the majority of the observed environment is static. Smaller dynamic objects can then be filtered out or down-weighted during the optimization using outlier detection and robust norms. In complex highly dynamic environments this assumption may not hold anymore and can lead to incorrect solutions. A typical situation in the autonomous driving context is driving in dense traffic, where the other dynamic vehicles occlude large parts of the static environment leaving only a few static keypoints visible, while the vehicles themselves provide the majority of features.

Temporal information can help to mitigate the problem of majority voting by only using converged landmarks that have been confirmed as static and testing many potential cases preferring solutions close to a smooth trajectory. However, testing many hypothesis is expensive and sometimes not enough observations are available to confirm that a feature is static. Additionally, there are situations where parts of the environment initially are static, but then become dynamic. These stop-and-go motions are especially challenging. As monocular methods leverage temporal information to estimate depth, it is even more difficult for them to differentiate between static and dynamic parts of the scene. Apart from the geometric information in the image in the form of descriptive or direct features semantic context information about what objects are in the scene can help to better differentiate between static and moving parts. The success of deep learning has produced a wide range of networks that are able to extract semantic information from a

single image in the form of object detection and segmentation in real-time. The semantic detection of objects allows an independent estimation of their potential dynamics. This enables a faster identification of new features as likely static parts and a more careful treatment of landmarks on potentially dynamic objects.

In this thesis we will use learned semantic priors predicted from the input RGB images in a probabilistic key point model in order to improve the robustness in highly dynamic scenes.

### Relative pose drift

The sequential construction of the map and pose estimation suffers from propagation of small pose errors that leads to an accumulation of drift over distance. In addition to drift in position and orientation monocular systems also suffer from a drift in scale. If the trajectory contains loops where a certain location is re-visited, loop closure detection and pose graph optimization can be used to remove the drift around these locations and reduce it in between. In outdoor environments a global reference system, like GPS, can be used to keep the drift small, however the accuracy can be insufficient for certain applications. For indoor environments where these systems are not available, man-made regular global structure can be exploited. If there exists a common layout of the environment like the regular configuration between walls, floors and ceilings or a grid-like structure of streets and buildings, this knowledge can be leveraged to integrate constraints or align the local map with a global layout to avoid frame-to-frame error accumulation. The most commonly used global system is the Manhattan World (MW), consisting of three orthogonal orientations, like the axes in a 3d coordinate system. Previous works [1, 2] have shown that this assumptions often holds true for man-made indoor environments where the orientation of walls, the floor and ceiling are consistent in a building or even over multiple buildings.

There already exist monocular SLAM methods that leverage a Manhattan assumption [1, 2]. They rely on the estimation of the vanishing points of 2d lines in order to estimate the MW frame orientation, requiring at least 2 visible orthogonal lines in each frame. Methods leveraging RGB-D sensors [3, 4] enable the estimation of surface normals from the depth measurements directly and therefore provide additional strong geometric features to estimate the Manhattan frame orientation. However, existing systems based on the MW assumption will fail, if the visible structure does not strictly adhere to the MW assumption or not sufficiently many constraints are visible in the current field of view.

In this thesis we show that the use of normal maps, predicted from RGB images, can be used for the rotation estimation against the MW frame. We show that the non-strict combination with point and line features in a graph-based SLAM framework can reduce the drift generated by monocular SLAM systems in man-made environments.

## 1.2 Contributions

The following summarizes the combined contributions of the thesis and lists the related publications.

1. Comprehensive overview and theoretical and mathematical background of a full graph-based monocular SLAM pipeline and its components and principles with a focus on the relevant aspects related to the robustness to dynamic outliers and the use of structural constraints.

2. A SLAM system combining descriptive and direct features with a probabilistic key point model that uses semantic priors in combination with depth uncertainty and matching accuracy to better model dynamic outliers, expanding the operational domain to more challenging environments.

   N. Brasch, A. Bozic, J. Lallemand, and F. Tombari. Semantic monocular slam for highly dynamic environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 393–400, 2018

3. A monocular SLAM system that combines predicted normals and Manhattan World based rotation estimation with point and line features in a graph-based optimization framework to reduce the drift in man-made environments.

   Y. Li, N. Brasch, Y. Wang, N. Navab, and F. Tombari. Structure-slam: Low-drift monocular slam in indoor environments. *IEEE Robotics and Automation Letters*, 5(4):6583–6590, 2020

# 2 Background

The following chapter covers the background for this thesis. Starting from a general perspective on SLAM it introduces relevant fundamental aspects that are the foundation of the methods developed in this thesis. It then presents the historical evolution and state of the art of monocular SLAM methods.

## 2.1 The SLAM Problem Statement

Simultaneous Localization and Mapping (SLAM) aims at the joint estimation of sensor poses and map geometry from a sequence of measurements obtained while moving through an unknown environment. SLAM systems were designed to allow long-term localization in an unknown environment. In contrast to the localization task with a pre-existing map. Here we consider the online version, where we want to estimate the poses and map points given the measurements up to a certain point in time, ideally in real-time. This is in contrast to the full slam setting or Structure from Motion SfM problem [7] where all the measurements are usually processed offline after the fact, allowing the use of future measurements for the estimation of previous states. In contrast to Visual Odometry (VO) [8] we are not only interested in relative pose estimation in a small area, but aim to build a globally consistent map to enable re-localization and detect re-visits of previously seen places in order perform global optimization over all available observations to remove the drift caused by accumulated pose and map errors.

When formulated as a probabilistic model, the SLAM problem can be written in the form of a Bayesian posterior using Bayes' rule [9].

$$p(\mathbf{T}, \mathbf{M}|\mathbf{Z}) = \frac{\overbrace{p(\mathbf{Z}|\mathbf{T}, \mathbf{M})}^{likelihood}\overbrace{p(\mathbf{T}, \mathbf{M})}^{prior}}{\underbrace{p(\mathbf{Z})}_{marginal}} \tag{2.1}$$

| | |
|---|---|
| $p(x\|y)$ | probability of $x$ given $y$ |
| $\mathbf{T}$ | sensor poses $\{i, ..., N\}$ |
| $\mathbf{M}$ | map geometry $\{i, ..., L\}$ |
| $\mathbf{Z}$ | measurements $\{i, ..., K\}$ |

This formulation expresses the joint distribution of the poses and map conditioned on the measurements as equivalent to the ratio of the likelihood of the measurements given the poses and map times the prior probabilities of the poses and map divided by the marginal probability of the measurements for normalization to obtain a probability density function.

We aim to maximize the Maximum A Posteriori (MAP) of this joint distribution in order to find the best estimates for the poses and map.

$$\mathbf{T}^*, \mathbf{M}^* = \underset{\mathbf{T},\mathbf{M}}{\operatorname{argmax}}\, p(\mathbf{T}, \mathbf{M}|\mathbf{Z}) \tag{2.2}$$

$\mathbf{T}$ sensor poses $\{i, ..., N\}$
$\mathbf{M}$ map geometry $\{i, ..., L\}$
$\mathbf{Z}$ measurements $\{i, ..., K\}$

Assuming a non-informative prior on the poses and map we can drop the term. As the normalization by the marginal of the measurements is irrelevant for finding the maximum we can also drop it. This leaves us with the likelihood term, which means that we want to maximize the probability of the measurements given our estimates for the poses and map. In order to do so, we can define a measurement model and aim to maximize the alignment of the actual measurements with the virtual measurements estimated by the measurement model based on a pose estimate and estimate of the map.

The assumption of independent measurements allows the factorization into individual terms.

$$p(\mathbf{T}, \mathbf{M}|\mathbf{Z}) \propto P(\mathbf{Z}|\mathbf{T}, \mathbf{M}) = \prod_{k=1}^{K} p(z_i|\mathbf{T}, \mathbf{M}) \tag{2.3}$$

$\mathbf{T}$ sensor poses $\{i, ..., N\}$
$\mathbf{M}$ map geometry $\{i, ..., L\}$
$\mathbf{Z}$ measurements $\{i, ..., K\}$
$z_i$ measurement $i$

Assuming a measurement model with added Gaussian noise gives us a Normal distribution around the virtual measurement as a mean.

$$p(z_k|T_i, M_j) = \mathcal{N}(z_k|f(T_i, M_j), \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(z_k - f(T_i, M_j))^2\right) \tag{2.4}$$

$z_k$ measurement $k$
$T_i$ sensor pose $i$
$M_j$ map element $j$
$\sigma^2$ measurement noise variance
$f(T_i, M_j)$ measurement model

Given the exponential distribution it is more convenient to minimize the negative log likelihood instead of maximizing the probability. Applying the negative logarithm and dropping any constant factors leads us to a non-linear least squares problem.

$$\mathbf{T}^*, \mathbf{M}^* = \underset{\mathbf{T},\mathbf{M}}{\operatorname{argmin}} \sum_{k=1}^{K} \frac{(z_k - f(T_i, M_j))^2}{\sigma^2} \tag{2.5}$$

| | |
|---|---|
| $\mathbf{T}$ | sensor poses $\{i, ..., N\}$ |
| $\mathbf{M}$ | map geometry $\{i, ..., L\}$ |
| $K$ | number of measurements |
| $z_k$ | measurement $k$ |
| $T_i$ | sensor pose $i$ |
| $M_j$ | map element $j$ |
| $f(T_i, M_j)$ | measurement model |
| $\sigma^2$ | measurement noise variance |

## 2.2 SLAM Variants

Simultaneous Localization and Mapping has a long research history in robotics and computer vision [9, 10].

### 2.2.1 Filtering-based SLAM

The early SLAM systems were based on recursive Bayesian filtering techniques such as the Extended Kalman Filter (EKF) that models the current pose and map geometry as a joint probability distribution in the form of a multivariate Gaussian with mean state vector and covariance matrix. While the probabilistic approach and efficient updates provide good results in small scale setting, they have several limitations when it comes to scalability to larger areas. Keeping a joint distribution of only the current pose and the map geometry requires marginalization of the previous poses which leads to large and dense covariance matrices between poses and map geometry. The inversion of this matrix during the update, results in a cubic complexity with respect to the number of map elements, this limits the total number of landmarks in the map in real-time settings. The usually non-linear measurement functions are approximated by linearized versions around the current estimate at the time they are observed. The linearization around the initial estimate can introduce additional errors that are propagated through the system. Over the years many improved filter-based methods have been proposed that aim to improve the complexity and efficiency.

### 2.2.2 Graph-based SLAM

However, most modern SLAM systems are graph-based systems. These methods only keep a selected sparse graph of sensor poses, map geometry and measurements, while discarding others, and solve a non-linear optimization problem aligning the current estimate with the measurements [9, 10]. These methods have shown to efficiently leverage the sparsity between map, measurements and poses and scale linear in the number map elements. They can also re-linearize the measurement functions at each step during the optimization. This efficiency gain compensates for the discarded elements [11].

SLAM system consist of multiple modules. We will focus on the three that are the most relevant ones for this thesis: initialization, mapping and tracking.

**Initialization:** SLAM systems have to first overcome the chicken-egg problem of estimating poses relative to a map and building a map based on the poses. This is often done by bootstrapping the problem with an initialization step. The initialization is conducted at the beginning of a sequence and when tracking is lost in order to re-initialize the system.

**Tracking:** Tracking aims to estimate the pose of an incoming frame by aligning the image measurements with previous frames and the map. Tracking has to be fast to allow real-time performance also for high frame rates. It can include a method for re-localization and loop closures detection in order to detect previously visited places. Sometimes the loop closure detection also runs in a separate process in parallel.

**Mapping:** Mapping concerns the growing, culling and optimization of the map geometry and the poses of keyframes to ensure global consistency. The process usually runs at a lower frequency in the background using more computationally heavy methods that can improve the initial estimates coming from the tracking module.

### 2.2.3 Visual SLAM

Visual SLAM refers to SLAM systems using cameras. Modern visual SLAM systems use concepts from a long line of works in computer vision and robotics on SfM, BA and VO [12, 13]. In the following section we will first introduce the most important theoretical concepts in monocular SLAM related to the contributions of this thesis before discussion the details of previous works.

## 2.3 Theoretical Concepts

This section contains the basic theoretical concepts in monocular SLAM related to this thesis, like projective geometry, different types of visual features, depth measurement models, bundle adjustment, training deep neural networks and the metrics used in the evaluation.

### 2.3.1 Camera Projection

In this thesis we use the pinhole camera model for the cameras. We assume that all images have been undistorted beforehand, e.g. using a radial distortion model or a look-up-table.

The pinhole camera intrinsic matrix is defined by the focal length and optical center.

$$\mathbf{K} = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{2.6}$$

$f_x, f_y$ focal length in $x$ and $y$
$c_x, c_y$ center of optical axis

We can then define the pinhole camera projection of a point $\mathbf{X}$ onto the image at point $\mathbf{x}$.

$$\mathbf{x} = \begin{pmatrix} x \\ y \end{pmatrix} = \pi(\mathbf{K}\mathbf{X}) \tag{2.7}$$

| | |
|---|---|
| $\mathbf{x}$ | 2D point in image coordinates |
| $x, y$ | pixel coordinates of projected point $\mathbf{X}$ |
| $\pi$ | projection function |
| $\mathbf{K}$ | 3x3 camera intrinsic matrix (see eq. 2.6) |
| $\mathbf{X}$ | 3D point in camera coordinate frame that is projected |

We use the function $\pi(\mathbf{X})$ to represent the projection that converts from homogeneous coordinates in projective space to euclidean coordinates by dividing the first two coordinates by the homogeneous coordinate, in this case the depth of point $X$. In the notation the homogeneous coordinate is sometimes dropped.

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \pi(\begin{pmatrix} X \\ Y \\ Z \end{pmatrix}) = \begin{pmatrix} X/Z \\ Y/Z \\ Z/Z \end{pmatrix} \tag{2.8}$$

Its inverse operation converts back to homogeneous coordinates.

$$\begin{pmatrix} X \\ Y \\ 1 \end{pmatrix} = \pi^{-1}(\begin{pmatrix} x \\ y \end{pmatrix}) = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{2.9}$$

Combining the camera rotation and translation with the camera intrinsic matrix gives us the fundamental matrix that defines the epipolar geometry between two frames.

$$\mathbf{F}_{ij} = \mathbf{K}_i \underbrace{\mathbf{R}_{ij}[\mathbf{t}_{ij}]_\times}_{\text{essential matrix}} \mathbf{K}_j^{-1} \tag{2.10}$$

| | |
|---|---|
| $\mathbf{F}_{ij}$ | 3x3 fundamental matrix |
| $\mathbf{K}_i, \mathbf{K}_j$ | 3x3 intrinsic matrix of camera $i$, $j$ |
| $\mathbf{R}_{ij}$ | 3x3 rotation matrix of camera $i$, $j$ |
| $[\mathbf{t}_{ij}]_\times$ | 3x3 skew-symmetric translation matrix of camera $i$, $j$ |

Here we used the skew-symmetric matrix $[\mathbf{x}]_\times$ that maps a 3x1 vector $\mathbf{x}$ to a 3x3 matrix.

$$[\mathbf{x}]_\times = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{2.11}$$

| | |
|---|---|
| $\mathbf{x}$ | 3D vector |
| $x_i$ | vector component $i$ |

Given a pixel in image $j$ we can use the fundamental matrix to compute the corresponding epipolar line in frame $i$. All 3d points on the ray through that pixel lie on this line. So, if the relative poses between two frames is known, we can limit ourselves to a search along the epipolar line to find potential correspondences.

$$\mathbf{l}_i = \mathbf{F}_{ij}\mathbf{x}_j \tag{2.12}$$

| | |
|---|---|
| $\mathbf{l}_i$ | 3x1 epipolar line in image $i$ |
| $\mathbf{F}_{ij}$ | 3x3 fundamental matrix |
| $\mathbf{x}_j$ | 3x1 homogeneous 2d pixel coordinates in image $j$ |

### 2.3.2 Visual Features

Visual SLAM methods rely on image features in order to align multiple observations during tracking and to triangulate map elements. We can differentiate two types of approaches.

**Indirect methods** or descriptive methods extract keypoints and compute descriptors in order to establish correspondences between observations. The found matches are then used to align the observations based on their re-projection errors, the distance between their projections in image space. In addition to simple point features more complex geometric structures are also used such as lines and planes.

**Direct methods** do not use higher-level features, the pixels in the image themselves are the features and their values are used to align frames to each other without explicit correspondences.

#### 2.3.2.1 Descriptive Point Features

Most SLAM systems are landmark based systems [14, 15] that means they are tracking distinct landmarks between frames. Each new corresponding observation can then be used to update the current estimate of the landmark position. The use of descriptors allows the robust matching of features over larger distances and globally on an image level.

**Brief History** Over the years many different keypoint extraction methods and descriptors have been proposed to improve the invariance to scale, rotation, intensity changes, noise and other transformations [16, 17, 18, 19]. The early SLAM works used corner detectors like FAST [16] to extract keypoints and defined image patches around the keypoint as descriptors. Later methods added compact and robust descriptors in order to speedup and improve the matching process. In this thesis we use binary descriptors to enable robust and efficient matching also under larger motions, while keeping the size of the map as small as possible.

**Point Features** In this thesis ORB features [19] are used for descriptive point features as they provide a good trade-off between efficiency and robustness. We use the

implementation from ORB-SLAM [20] that is slightly different from the original implementation and uses a quadtree to evenly distribute the features over the whole image.
**Keypoint Extraction** First, a multi-scale image pyramid is build and ORB features are extracted on each level to create scale invariance. The keypoint detection starts with FAST corner [21] extraction by finding pixels with a continuous circular region of brighter or darker pixels. In order to ensure an equal distribution over the image it is divided into a grid with the goal of balancing the number of features in each cell. First, corners are extracted in each cell using a higher threshold and only if no corners could be found the threshold is lowered. This ensures that strong candidates are selected before weaker ones. It then computes the more expensive Harris corner response [22] on the candidate points and filters them with a minimum threshold. In order to reduce the number of features while maintaining a uniform distribution over the image a quadtree based selection step is performed. Therefore the image as the root node is recursively split into 4 equally sized cells until there is only a single point in each cell left or there are already sufficiently many nodes. Then from each node the keypoint with the highest response is kept.
**Descriptor Computation** First we estimate the feature orientation via the angle between the keypoint center and the intensity centroid in order to achieve rotation invariance.

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{2.13}$$

$$\theta = atan2(\frac{m_{01}}{m_{10}}) \tag{2.14}$$

| | |
|---|---|
| $m_{pq}$ | moment $pq$ |
| $x, y$ | relative image coordinates |
| $\theta$ | feature orientation angle |

Given the oriented feature BRIEF descriptors are computed for matching. The descriptor is constructed based on a pattern of binary differences between certain pixels in a smoothed patch around the keypoint. In total 256 tests are conducted to build a 256 bit descriptor.

$$f_{ORB} = \{b_i, ..., b_n\} \text{ where } b_i = I(a_i) - I(b_i) > \vartheta \tag{2.15}$$

| | |
|---|---|
| $b_i$ | bit $i$ |
| $I(a_i), I(b_i)$ | smoothed image intensity at pixels $a_i$ and $b_i$ |
| $\vartheta$ | intensity difference threshold |

### Parameterization

Descriptive features are represented as 2D points in the image

$$\mathbf{x} = [x, y] \in \mathbb{R}^2 \tag{2.16}$$

and 3D points in the map.

$$\mathbf{X} = [X, Y, Z] \in \mathbb{R}^3 \tag{2.17}$$

**Matching** With BRIEF being a binary descriptor the Hamming distance, that counts the number of different bits between two descriptors, can be used to measure the descriptor similarity. Assuming keypoints have been extracted in both source and target frames they can be matched by comparing with all potential matches in a neighborhood around an initial estimate or in the whole image.

$$d_{Hamming} = \|XOR(f_a, f_b)\| \tag{2.18}$$

| | |
|---|---|
| $\|x\|$ | count number of 1 bits |
| $XOR(a, b)$ | bit-wise exclusive or operation |
| $f_a, f_b$ | binary descriptors |

**Triangulation** Given two correspondences $x_i$ and $x_j$ of the same point we can triangulate the 3d point $X$. While we could triangulate the 3d point by computing the smallest distance between the camera rays of the two observations, this so called midpoint method does not properly account for the projective transform in the cameras and therefore is not optimal to produce minimal re-projection errors.

Instead we can use Direct Linear Transform (DLT) to solve a system of linear equations to triangulate the point.

Given the up-to-scale relationships

$$\mathbf{x_i} = \alpha_i \mathbf{P_i X} \tag{2.19}$$

$$\mathbf{x_j} = \alpha_j \mathbf{P_j X} \tag{2.20}$$

| | |
|---|---|
| $\mathbf{x_i}, \mathbf{x_j}$ | 2d observations in images $i$ and $j$ in inhomogeneous coordinates |
| $\alpha_i, \alpha_j$ | unknown scale factors |
| $\mathbf{P_i}, \mathbf{P_j}$ | 3x4 projection matrices ($\mathbf{P} = \mathbf{K}[\mathbf{R}\|\mathbf{t}]$) for frames $i$ and $j$ |
| $\mathbf{X}$ | 3d point in homogeneous coordinates |

and the constraint that the homogeneous vector $x$ and $PX$ are parallel, we can build a linear system independently of the scale factors based on the fact that the cross product between them should be zero. Using the skew-symmetric matrix (see eq. 2.11) to rewrite the cross product we get two independent linear equations (first two rows) per observation

$$\mathbf{x_i} \times \mathbf{P_i X} = \underbrace{[x_i]_\times \mathbf{P_i}}_{\mathrm{A}} \mathbf{X} = \begin{bmatrix} 0 & -1 & y_i \\ 1 & 0 & -x_i \\ -y_i & x_i & 0 \end{bmatrix} \begin{bmatrix} P_i(1) \\ P_i(2) \\ P_i(3) \end{bmatrix} X = 0 \tag{2.21}$$

| | |
|---|---|
| $\mathbf{x_i}$ | projection of $X$ in image $i$ using homogeneous coordinates |
| $\mathbf{P_i}$ | 3x4 projection matrix of frame $i$ |
| $\mathbf{X}$ | 3D point in homogeneous coordinates |
| $x_i, y_i$ | $x$ and $y$ component of $\mathbf{x_i}$ |
| $\mathbf{P_i(j)}$ | row $j$ of projection matrix of frame $i$ |

Using $N$, but at least two observations we can obtain the 3d point $\mathbf{X}$ via Singular Value Decomposition (SVD).

$$\mathbf{USV}^T = \mathbf{A} \tag{2.22}$$

| | |
|---|---|
| $U$ | 2Nx3 left ortho-normal matrix |
| $S$ | 3x4 diagonal matrix |
| $V$ | 4x4 right ortho-normal matrix |
| $A$ | 2Nx4 linear system $\mathbf{A} = [x_i]_\times \mathbf{P_i}, \mathbf{A}$ |

We can retrieve the triangulated 3d point from the last de-homogenized row of $V^T$ corresponding to the smallest singular value [12].

$$\mathbf{X} = \frac{\mathbf{V}^T(3)}{\mathbf{V}^T(3,4)} \tag{2.23}$$

| | |
|---|---|
| $\mathbf{X}$ | triangulated 3D point |
| $V(i)$ | row $i$ of right ortho-normal matrix |
| $V(i,j)$ | element $i$,$j$ of right ortho-normal matrix |

**Depth Variance Estimation** The uncertainty in the matched pixel position leads to uncertainty in the triangulated 3d position of the point. Assuming an error of $n$ pixels along the epipolar line one could use the fundamental matrix to compute the epipolar line, walk $n$ pixels in both directions from the 2d keypoint in the image, compute the respective triangulated points and then choose the maximum 3d distance between these points and the triangulated point as the uncertainty. However, as this is computationally expensive a more efficient approximation is often used that assumes a fixed angular error that is added to the current estimate in order to estimate the 3d point uncertainty [23].

$$\mathbf{a} = \mathbf{X} - \mathbf{t}\alpha = \arccos\left(\frac{\mathbf{Xt}}{|\mathbf{X}|\,|\mathbf{t}|}\right) \tag{2.24}$$

$$\beta = \arccos\left(-\frac{\mathbf{at}}{|\mathbf{a}|\,|\mathbf{t}|}\right) \tag{2.25}$$

$$\beta_+ = \arctan\left(\frac{1}{f}\right) \tag{2.26}$$

$$\gamma = \pi - \alpha - (\beta + \beta_+) \tag{2.27}$$

$$|X_+| = |t|\,\frac{\beta + \beta_+}{\gamma} \tag{2.28}$$

$$\sigma^2 = (|X_+| - |X|)^2 \tag{2.29}$$

| | |
|---|---|
| $\mathbf{a}$ | vector from $\mathbf{X}$ to camer center of frame $j$ |
| $\mathbf{X}$ | 3D Point |
| $\mathbf{t}$ | translation from frame $i$ to frame $j$ |
| $\alpha$ | angle between vectors $\mathbf{X}$ and $\mathbf{t}$ |
| $\beta$ | angle between vectors $\mathbf{a}$ and $\mathbf{t}$ |
| $\beta_+$ | angular error assuming 1 pixel error in the image |
| $\sigma^2$ | approximated depth variance |

The assumption of a fixed angle $\beta_+$ is an approximation that works well in practice as it considers the relative orientation and baseline between the two frames.

**Re-Projection Error** Based on the degrees of freedom (rotation and/or translation) and the type of correspondences (2D-3D or 3D-3D) different relationships and methods can be used to formulate error terms with point features.

**Rotation + Translation (3D-2D)** Given a 3d-2d-correspondence between a 3d keypoint from a keyframe or the map reference coordinate frame $i$ and a corresponding keypoint in frame $j$ we can formulate a re-projection error.

$$\mathbf{E_{rp}} = \mathbf{x} - \pi \left( \mathbf{K T_j T_i}^{-1} \mathbf{X} \right) \tag{2.30}$$

| | |
|---|---|
| $E_R$ | re-projection error |
| $\mathbf{x}$ | 2D pixel coordinates of observation in frame $j$ |
| $\pi$ | projection function (see eq. 2.8) |
| $\mathbf{K}$ | camera intrinsics matrix |
| $T_i, T_j$ | pose matrices of frames $i$ and $j$ |

The relationship between 3d points and their perspective projections represents the Perspective-n-Point (PnP) problem [24]. Several efficient methods [25, 26, 27] have been proposed to estimate the relative pose between a set of 3d points and their projections. These methods are employed in SLAM systems to obtain initial estimates of the pose of a new frame given some estimated 3d points. In the case of a loop closure this could be the alignment of a new frame to another part of a known map or in the case of tracking loss it can be used to localize the new frame with respect to the local map.

**Translation only (3D-2D)** In the special case that the rotation is already known, e.g. because it was estimated using MW rotation estimation, a simplified solution can be derived. By combining the relationship between the 3d points in frame $i$ and $j$ given by the transform $T = [R|t]$

$$\mathbf{X}_i = \mathbf{R}\mathbf{X}_j + \mathbf{t} \tag{2.31}$$

| | |
|---|---|
| $\mathbf{X}_i, \mathbf{X}_j$ | 3D point in frame $i$, $j$ |
| $\mathbf{R}$ | 3x3 rotation matrix |
| $\mathbf{t}$ | 3D translation vector |

with the relationship to the observed image coordinates given by the perspective projection.

$$\mathbf{X}_k = \mathbf{K} \left[ \frac{X_k}{Z_k}, \frac{Y_k}{Z_k}, 1 \right]^T Z_k = \begin{bmatrix} x_k \\ y_k \\ 1 \end{bmatrix} Z_k \tag{2.32}$$

| | |
|---|---|
| $\mathbf{X}_k$ | 3D point in frame $k$ |
| $\mathbf{K}$ | 3x3 camera intrinsic matrix |
| $X_k, Y_k, Z_k$ | X,Y,Z components of $\mathbf{X}_k$ |
| $x_k, y_k$ | 2D projected image coordinates |

we get the three rows

$$Z_i \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = Z_j \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \begin{bmatrix} x_j \\ y_j \\ 1 \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} \tag{2.33}$$

| | |
|---|---|
| $x_i, y_i, x_j, y_j$ | 2D image coordinates in frame $i$, $j$ |
| $Z_i, Z_j$ | z-component of the 3D point |
| $r_1, r_2, r_3$ | rows 1,2,3 of the 3x3 rotation matrix |
| $t_1, t_2, t_3$ | components 1,2,3 of the 3D translation vector |

by substituting $Z_i$ from the last row into the first two rows, rearranging for $Z_j$ and subtracting one row from the other we can eliminate both depth dependencies and obtain a single constraint for each point pair.

$$\begin{bmatrix} -y_i t_3 + t_2 \\ x_i t_3 - t_1 \\ -x_i t_2 + y_i t_1 \end{bmatrix}^T \underbrace{\begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} \begin{bmatrix} x_j \\ y_j \\ 1 \end{bmatrix}}_{\text{Constant}} = 0 \tag{2.34}$$

| | |
|---|---|
| $x_i, y_i, x_j, y_j$ | 2D image coordinates in frame $i$, $j$ |
| $t_1, t_2, t_3$ | components 1,2,3 of the 3D translation vector |
| $r_1, r_2, r_3$ | rows 1,2,3 of the 3x3 rotation matrix |

This gives us a system of linear equations that we can solve using SVD to estimate the translation.

**Rotation + Translation (2D-2D)** To solve the chicken-egg problem of the initialization many monocular SLAM methods rely on the estimation of the fundamental matrix (8-point [28, 29] and 5-point [30]) or homography matrix [31, 12] between two frames from a set of 2d correspondences. As the keypoints can be noisy and the correspondences erroneous a RANSAC scheme is often used. Here many random sets of correspondences are sampled to estimate hypotheses for the transformation. These are then scored based on the number of other correspondences that support the hypothesis. In practice these

methods can return multiple possible solutions and some additional checks are needed to ensure a valid and accurate solution [12, 15].

**Fundamental matrix estimation** The normalized 8-point [29] algorithm uses the relationship between the matched observations of the keypoints in two frames to estimate the relative rotation and translation between them up to an unknown scale (5 DoF). The two observations of the same 3d point are related via the fundamental matrix $F$ [28].

$$\mathbf{x_i}^T \mathbf{F}_{ij} \mathbf{x_j} = 0 \tag{2.35}$$

$x_i, x_j$ homogeneous coordinates for 2d observations of point $X$ in frame $i$, $j$
$F_{ij}$ 3x3 fundamental matrix from frame $j$ to frame $i$

Which can be rewritten as a system of homogeneous linear equations with respect to the 9 parameters of the fundamental matrix.

$$\mathbf{X}\mathbf{f}_{ij} = \mathbf{A}\mathbf{x} = 0 \tag{2.36}$$

$\mathbf{X}, \mathbf{A}$ Nx9 stacked constraints from multiple observations $x_i$, $x_j$
$f_{ij}, \mathbf{x}$ 9x1 vectorized fundamental matrix from frame $j$ to frame $i$

For improved stability of the method the 2d coordinates are standardized using mean subtraction and division by the standard deviation. Due to errors in the keypoint estimation and to be able to use more correspondences to ensure sufficiently many independent constraints the system is solved using a least-squares approximation.

$$|\mathbf{A}\mathbf{x}|_2 = 0 \text{ subject to } |\mathbf{x}|_2 = 1 \tag{2.37}$$

$\mathbf{A}$ Nx9 stacked constraints from multiple observations $x_i$, $x_j$
$x$ 9x1 vectorized fundmental matrix from frame $j$ to frame $i$
$|\mathbf{x}|_2$ L2-Norm of $\mathbf{x}$

To solve the least squares problem a first SVDs is used to find the smallest singular value.

$$\mathbf{A} = \mathbf{U}\mathbf{S}\mathbf{V}^T \tag{2.38}$$

$\mathbf{A}$ Nx9 stacked constraints from multiple observations $x_i$, $x_j$
$\mathbf{U}$ NxN left ortho-normal matrix
$\mathbf{S}$ Nx9 diagonal matrix
$\mathbf{V}$ 9x9 right ortho-normal matrix

The initial solution for the fundamental matrix is then the last row in the right ortho-normal matrix $\mathbf{V}^T$ corresponding to the smallest singular value.

$$\hat{\mathbf{F}}_{ij} = \mathbf{V}^T(8) \tag{2.39}$$

$F_{ij}$ 3x3 fundamental matrix from frame $j$ to frame $i$
$\mathbf{V}^T(8)$ 8th row of 9x9 right ortho-normal matrix

The fundamental matrix is composed of the essential matrix and the camera intrinsic transformations. We apply the inverse intrinsic matrix to recover the essential matrix from the fundamental matrix.

$$\mathbf{E}_{ij} = \mathbf{K}_i^{-1}\mathbf{F}_{ij}\mathbf{K}_j \tag{2.40}$$

| | |
|---|---|
| $E_{ij}$ | 3x3 essential matrix from frame $j$ to frame $i$ |
| $\mathbf{K}_i, \mathbf{K}_j$ | 3x3 intrinsic matrix of camera $i$, $j$ |
| $\mathbf{F}_{ij}$ | 3x3 fundamental matrix from frame $j$ to frame $i$ |

The essential matrix is the composition of relative rotation and translation expressed as a skew-symmetric matrix (reflecting the cross product with the transformed point).

$$\mathbf{E}_{ij} = \mathbf{R}_{ij}[\mathbf{t}_{ij}]_\times \tag{2.41}$$

| | |
|---|---|
| $\mathbf{E}_{ij}$ | 3x3 essential matrix |
| $\mathbf{R}_{ij}$ | 3x3 rotation matrix of camera $i$, $j$ |
| $[\mathbf{t}_{ij}]_\times$ | 3x3 skew-symmetric translation matrix of camera $i$, $j$ |

To make sure that the constraints of the rotated skew-symmetric essential matrix are fulfilled a second SVD is performed on the initial essential matrix estimate.

$$\mathbf{USV} = \hat{\mathbf{E}}_{ij} \tag{2.42}$$

| | |
|---|---|
| $\mathbf{U}$ | 3x3 left ortho-normal matrix |
| $\mathbf{S}$ | 3x3 diagonal matrix |
| $\mathbf{V}$ | 3x3 right ortho-normal matrix |
| $E_{ij}$ | 3x3 essential matrix from frame $j$ to frame $i$ |

The first constraint of the skew-symmetric essential matrix is that the smallest singular value is zero. Therefore, we set the last entry in the diagonal matrix to zero.

$$\mathbf{E}_{ij} = \mathbf{US'V} \tag{2.43}$$

| | |
|---|---|
| $E_{ij}$ | 3x3 essential matrix from frame $j$ to frame $i$ |
| $\mathbf{U}$ | 3x3 left ortho-normal matrix |
| $\mathbf{S'}$ | 3x3 modified diagonal matrix |
| $\mathbf{V}$ | 3x3 right ortho-normal matrix |

The essential matrix has an additional constraint that the first and second singular value need to be equal. Depending on the choice of which singular value is chosen and the choice of a positive or negative scale there are altogether 4 possible solutions. All of them are checked and scored based on the parallax and re-projected epipolar line distance error to find the most plausible one.

The rotation and translation can then be recovered from the essential matrix using the SVD decomposition of the essential matrix.

$$R_{ij} = \mathbf{U}\mathbf{W}\mathbf{V}^T \tag{2.44}$$

$$t_{ij} = \mathbf{U}\mathbf{W}\mathbf{S}''\mathbf{U}^T \tag{2.45}$$

| | |
|---|---|
| $R_{ij}$ | 3x3 essential matrix from frame $j$ to frame $i$ |
| $\mathbf{U}$ | 3x3 left ortho-normal matrix |
| $\mathbf{W}$ | 3x3 Rotation $\pm\frac{\pi}{2}$ around the z-axis |
| $\mathbf{V}$ | 3x3 right ortho-normal matrix |
| $\mathbf{S}''$ | 3x3 diagonal matrix $diag([1, 1, 0])$ |

Using a RANSAC scheme many different hypothesis are computed based on random subsets of the matches. Each hypothesis is scored based on the bi-directional reprojection error represented by the distance between the observation and the epipolar line.

$$d_E = \mathbf{l_j} x_i = \mathbf{F}_{ij} x_j x_i \tag{2.46}$$

| | |
|---|---|
| $d$ | distance between 2d point and epipolar line |
| $l_j$ | epipolar line in frame $i$ computed from keypoint in frame $j$ |
| $x_i, x_j$ | 2d keypoint positions in homogeneous coordinates $\in \mathcal{P}^2$ |
| $\mathbf{F}_{ij}$ | fundamental matrix of camera $i$, $j$ |

**Homography estimation** In the case of a planar scene we can make a more specific assumption based on a homography transformation between the matched keypoints in the two frames.

$$\mathbf{x_i}^T = \mathbf{H}_{ij}\mathbf{x_j} \tag{2.47}$$

| | |
|---|---|
| $x_i, x_j$ | 2d observations of point $X$ in frame $i$, $j$ |
| $H_ij$ | 3x3 homography from frame $j$ to frame $i$ |

Which also leads to a system of homogeneous linear equations with respect to the matrix parameters.

$$\mathbf{X}\mathbf{h}_{ij} = \mathbf{A}\mathbf{x} = 0 \tag{2.48}$$

| | |
|---|---|
| $\mathbf{X}, \mathbf{A}$ | 2Nx9 stacked constraints from multiple observations $\mathbf{x}_i$, $\mathbf{x}_j$ |
| $\mathbf{h}_{ij}, \mathbf{x}$ | 9x1 vectorized essential matrix from frame $j$ to frame $i$ |

Following the same approach as for the essential matrix by converting it into a least squares problem and using SVD we can solve the problem choosing the right singular vector corresponding to the smallest singular value.

To recover the rotation and translation from the homography we can use a second SVD decomposition, this time of the homography matrix [31].

$$\mathbf{U}\mathbf{S}\mathbf{V} = \hat{\mathbf{H}}_{ij} \tag{2.49}$$

| | |
|---|---|
| $\mathbf{U}$ | 3x3 left ortho-normal matrix |
| $\mathbf{S}$ | 3x3 diagonal matrix |
| $\mathbf{V}$ | 3x3 right ortho-normal matrix |
| $H_{ij}$ | 3x3 homography from frame $j$ to frame $i$ |

Given that the two outer matrices are ortho-normal the diagonal matrix can be interpreted as a homography between two rotated versions of the keypoints. Using the definition of the homography

$$\mathbf{x_i}^T = \mathbf{H}_{ij}\mathbf{x_j} = \mathbf{K_i}\left(\mathbf{R} + \mathbf{t}\mathbf{n}^T\right)\mathbf{K_j}^{-1}\mathbf{x_j} \tag{2.50}$$

| | |
|---|---|
| $x_i, x_j$ | 2d observations of point $X$ in frame $i$, $j$ |
| $H_ij$ | 3x3 homography from frame $j$ to frame $i$ |
| $\mathbf{K}_i, \mathbf{K}_j$ | 3x3 intrinsic matrix of camera $i$, $j$ |
| $\mathbf{R}$ | 3x3 rotation matrix |
| $\mathbf{t}$ | 3x1 translation vector |

we can decompose the diagonal homography into translation and rotation and re-apply the rotations $\mathbf{U}$ and $\mathbf{V}$ [31].

$$\mathbf{R} = det(\mathbf{U})det(\mathbf{V})\mathbf{U}(\pm_1 1)\begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix}\mathbf{V}^T \tag{2.51}$$

$$\mathbf{t} = \mathbf{U}(\sigma_1(\pm_1 1)\sigma_3)\begin{bmatrix} (\pm_2 1)\sqrt{\frac{\sigma_1*\sigma_1-\sigma_2*\sigma_2}{\sigma_1*\sigma_1-\sigma_3*\sigma_3}} \\ 0 \\ (\pm_1 1)(\pm_3 1)\sqrt{\frac{\sigma_2*\sigma_2-\sigma_3*\sigma_3}{\sigma_1*\sigma_1-\sigma_3*\sigma_3}} \end{bmatrix} \tag{2.52}$$

$$cos(\theta) = \frac{(\pm_1 1)\sigma_2*\sigma_2 + \sigma_1*\sigma_3}{(\sigma_1(\pm_1 1)\sigma_3)*\sigma_2} \tag{2.53}$$

$$sin(\theta) = \frac{\sqrt{(\sigma_1*\sigma_1 - \sigma_2*\sigma_2)*(\sigma_2*\sigma_2 - \sigma_3*\sigma_3)}}{(\sigma_1(\pm_1 1)\sigma_3)*\sigma_2} \tag{2.54}$$

| | |
|---|---|
| $\mathbf{U}$ | 3x3 left ortho-normal matrix |
| $\mathbf{V}$ | 3x3 right ortho-normal matrix |
| $H_{ij}$ | 3x3 homography from frame $j$ to frame $i$ |
| $\sigma_1, \sigma_2, \sigma_3$ | diagonal elements of matrix $\mathbf{S}$ |

As can be seen by the three $\pm_k$ signs in the equations the problem has in total 8 potential solutions [31]. In ORB-SLAM [20] all solutions are reconstructed and scored based on the plausibility of the reconstructed map ensuring low re-projection error, geometric correctness and sufficient parallax.

For the scoring of the RANSAC hypotheses the bi-directional re-projection error between the transformed and observed keypoints is used. To improve the quality of the

scoring ORB-SLAM uses the chi-square test assuming a standard deviation of 1 pixel and 95% inlier ratio threshold to filter out outlier matches.

$$d_H = |\mathbf{H_{ij}} x_j - x_i|_2 \tag{2.55}$$

| | |
|---|---|
| $d$ | distance between 2d point and epipolar line |
| $l_j$ | epipolar line in frame $i$ computed from keypoint in frame $j$ |
| $x_i, x_j$ | 2d keypoint positions in homogeneous coordinates $\in \mathcal{P}^2$ |
| $\mathbf{F}_{ij}$ | fundamental matrix of camera $i$, $j$ |

In ORB-SLAM [20] both homography and essential matrix variants are tested in parallel and the best scoring solution is chosen. The full algorithm is shown in alg. 1.

**Algorithm 1** Relative Pose Initialization

Input: Matches $X$, Intrinsics $\mathbf{K}_i$, $\mathbf{K}_j$
Output: Relative transform $\mathbf{R}$, $\mathbf{t}$
$\mathbf{X} \leftarrow \{\mathbf{X_1}, ..., \mathbf{X_i}, ..., \mathbf{X_N}\}$ ▷ Generate random sets
**procedure** FIND ESSENTIAL MATRIX
    $s_{E,best} \leftarrow \infty$
    **for** $i \in \{1, ..., N\}$ **do**
        $E_{ij} \leftarrow EstimateEssentialMatrix(X_i)$ ▷ Using eq. 2.43
        $s_E \leftarrow ComputeEpipolarDistance(E_{ij}, X)$ ▷ Compute score
        **if** $s_h > s_{E,best}$ **then**
            $s_{E,best} \leftarrow s$ ▷ Track best score
        **end if**
    **end for**
    Return $s_{E,best}$
**end procedure**
**procedure** FIND HOMOGRAPHY
    $s_{H,best} \leftarrow \infty$
    **for** $i \in \{1, ..., N\}$ **do**
        $H_{ij} \leftarrow EstimateHomographyMatrix(X_i)$ ▷ Using eq. 2.48
        $s_H \leftarrow ComputeReProjectionError(H_{ij}, X)$ ▷ Compute score
        **if** $s_H > s_{H,best}$ **then**
            $s_{H,best} \leftarrow s$ ▷ Track best score
        **end if**
    **end for**
    Return $s_{H,best}$
**end procedure**
**if** $s_{E,best} > s_{H,best}$ **then**
    $\mathbf{R}, \mathbf{t} \leftarrow DecomposeEssential(E_{ij})$ ▷ Decompose essential matrix
**else**
    $\mathbf{R}, \mathbf{t} \leftarrow DecomposeHomography(H_{ij})$ ▷ Decompose homography
**end if**

#### 2.3.2.2 Descriptive Line Features

In addition to the point features lines can also be used as landmarks. While points are concise local features and are either completely visible in an image or not, lines can span over large areas and different parts of it might be visible in the different images. While this theoretically allows the detection of the same line without any overlap in the field of view the lines are often described and limited by a start and end point that is different for each observation. In contrast to the alignment of two points only a co-linear constraint can be derived from line feature observations. However, they can still provide valuable additional information especially in structured texture-less scenes with a limited number of point features.

**Line Features** In this thesis we use the Line Segment Detector (LSD) [32] extractor to detect lines in the image and the Line Band Descriptor (LBD) [33] based descriptor for the matching of lines based on PL-SLAM [34].

**Line Extraction** Multi-scale LSD [32] features are extracted from the input images using a resolution pyramid. First the image gradients and their orientation angles are computed. A greedy region growing method is then used to find clusters of pixels that surpass a certain gradient threshold and whose gradient angles are close to the current accumulated average angle of the region. For each found connected component an oriented rectangle is fitted in order to estimate initial line parameters. A line is computed via the center of mass of all pixels in a cluster and using the first inertia moment using the pixels gradients as weights for the orientation estimation. The width and height are chosen such that the rectangle covers all contributing pixels. Each found line hypothesis is validated based on the number of supporting pixels inside the found rectangle. All lines that pass the validation are refined by testing small sub-pixel perturbations and keeping the one with the best score. All the steps are also shown in algorithm 2.

**Algorithm 2** Steps of the LSD extraction

Input: Image $\mathbf{I}$
Output: Set of lines $\mathbf{L}$
**procedure** EXTRACTLINES
    $\Delta I \leftarrow \frac{\partial I}{\partial x}$ ▷ Compute image gradient
    $\Theta \leftarrow LineAngle(\Delta I)$ ▷ Compute line angle
    $S \leftarrow RegionGrowing(\Delta I, \Theta, \Delta I_{min})$ ▷ Find connected components
    $R = FitRectangles(S)$ ▷ Compute Line
    $L = VaildateLines(R)$ ▷ Filtering
    $L = TestLinePermutations(L)$ ▷ Refinement
**end procedure**

**Descriptor Computation** We compute binary LBD [33] descriptors for each line. Therefore a canonical orientation is estimated for the line based on the main gradient direction. Next the oriented rectangle is divided into $M$ bands of width $W$ that are orthogonal to

the line direction. Using a global and local Gaussian smoothing kernel the positive and negative gradients along the direction of the line and orthogonal to it are averaged for each row. This leads to 4 (-x, -y, +x, +y) accumulated gradients per row. Next, for each gradient the mean and standard deviation over the rows in each band and its direct neighbors are computed to obtain a Band Descriptor (BD) of 8 scalars. In total the descriptor size over all bands has the size $N = M * 8$. In PL-SLAM, in order to create a more compact binary descriptor version, 32 fixed combinations of band pairs are used to compare the 4 mean and variance values resulting in a 256 bit descriptor [35].

$$f_{LBD} = \{b_i, ..., b_n\} \text{ where } b_i = BD(k) > BD(j) \tag{2.56}$$

$b_i$ 8 bits for pair $k$,$j$
$BD(k), BD(j)$ 8 scalar descriptor for band $k$

### Parameterization

For the 3d line features often the over-parameterized end-point representation is used.

$$\mathbf{L} = [\mathbf{X_s}^T, \mathbf{X_e}^T]^T \tag{2.57}$$

$\mathbf{X_s}$ start point of the line
$\mathbf{X_e}$ end point of the line

While being over-parameterized for a 3d line with 4 DoF, this representations still allows stable optimization without any gauge degrees when the two end points are updated independently [34]. Alternatively the ortho-normal minimal representation proposed by Bartoli *et al*. can be used [36].

**Matching** Using the binary LBD descriptors the matching of lines is done using the Hamming distance (see eq. 2.18), the same as for the ORB features.

**Triangulation** The problem of the triangulation of a 3d line from two 2d line observations can be seen as finding the intersection of the two planes spanned by the 2d end points and the camera centers. However, this method suffers from the same problem as the mid-point method for points, that it does not properly reflect how the distance to the line is reflected in the projection. Therefore, in practice often the start and end points of the line are treated as individual point observations and the same triangulation method as for the point features is used. In the case of partial occlusions of the line the visible end points in the two views might not correspond to the same point. However, the triangulated point should still lie somewhere on the line between the two visible endpoints. To properly account for the camera projection alternative triangulation methods have been proposed [36]. However, in this thesis we use the independent triangulation of the end points using the same DLT triangulation method as for the point features.

**Re-Projection Error** In order to estimate the re-projection error between two lines we first project the end points into the image using the pinhole camera model. Then we can compute the 2d line equations in image space using the normalized cross product.

$$\mathbf{l} = (a, b, c) = \frac{\mathbf{x_s} \times \mathbf{x_e}}{\|\mathbf{x_s}\|\|\mathbf{x_e}\|} \tag{2.58}$$

| | |
|---|---|
| $\mathbf{l}$ | 3D line vector representing projected 2d line |
| $a, b, c$ | parameters of projected line |
| $\mathbf{x_s}$ | 2D projected start point of the line |
| $\mathbf{x_e}$ | 2D projected end point of the line |

The re-projection error is then formulated as the sum of two error terms, one term for the start point and one for the end point, as for example in PL-SLAM [34].

$$\mathbf{E_{rl}} = \mathbf{l_i}\pi(K\mathbf{T_{ij}}\mathbf{X_{j,s}}) + \mathbf{l_i}\pi(K\mathbf{T_{ij}}\mathbf{X_{j,e}}) \tag{2.59}$$

| | |
|---|---|
| $\mathbf{l}$ | 3D line vector representing projected 2D line |
| $\pi(\mathbf{x})$ | projection function |
| $\mathbf{K}$ | 4x4 camera intrinsic matrix |
| $\mathbf{T}_{ij}$ | 4x4 transformation matrix from frame $j$ to frame $i$ |
| $\mathbf{X_{j,s}}, \mathbf{X_{j,e}}$ | 3D start and end points of the line in frame $j$ |

#### 2.3.2.3 Direct Point Features

Direct methods optimize the alignment of the raw measurements without explicit correspondences by using the direct comparisons of color or depth consistency [37, 38, 39, 40]. Directly using the image intensities for alignment save the computational effort for computing keypoints and descriptors. As direct methods do not require features they can better handle texture less scenes and are more agnostic towards some forms of noise like motion blur. However, the cost landscape of direct methods is usually difficult to optimize due to many local minima. Therefore, they often rely on a coarse-to-fine initialization strategy to increase the convergence basin. They are more sensitive to geometric noise from rolling shutter effects, distortions and intrinsic calibration errors. Due to the lack of descriptors for matching direct methods in general require additional features and logic for re-localization and loop-closure detection [38]. Direct methods are also sensitive to intensity changes due to illumination changes in the scene or changes in the camera setting. Therefore adaptive exposure models have been proposed that estimate an affine illumination model per image that accounts for global linear intensity changes and offsets. The pros and cons of direct methods are analyzed in detail by Engel *et al.* [40].

**Direct Features** DSO [40] proposes an efficient sparse feature selection strategy based on a local intensity threshold and a grid based iterative selection strategy. For each resolution pyramid level keypoints are extracted using a grid of increasing block

size. In each block the pixel with the maximum local intensity gradient is checked, if it is larger than the current threshold for the window it is added. The thresholds get smaller for increasing window sizes in order to also add larger, but more subtle gradient features. The whole extraction process is shown in algorithm 3.

**Algorithm 3** Direct feature extraction

Input: Image $I$
Output: Direct keypoints $\mathbf{P}$
**procedure** EXTRACTDIRECTKEYPOINTS
    $\mathbf{P} \leftarrow \{\}$
    **for** $l \in \{1, ..., L\}$ **do** ▷ For each pyramid level
        **for** $w \in \{W, 2W, 4W\}$ **do** ▷ For different block sizes
            **for** $X, Y \in \{1, ..., N\}, \{1, ..., M\}$ **do** ▷ For each block
                $p_{max} = null$
                $\Delta_{max} = null$
                $\overline{\Delta I} = \frac{1}{W^2} \sum_{X-W/2}^{X+W/2} \sum_{Y-W/2}^{Y+W/2} (\Delta I(x, y))$
                **for** $x, y \in \{X_b, ..., X_b + W\}, \{Y_b, ..., Y_b + W\}$ **do** ▷ For each pixel
                    **if** $\Delta I(x, y) > \overline{\Delta I} + \delta_w$ and $\Delta I(x, y) > \Delta_{max}$ **then**
                        $p_{max} \leftarrow (x, y)$
                        $\Delta_{max} \leftarrow \Delta I(x, y)$
                    **end if**
                **end for**
                **if** $p_{max} \neq null$ **then**
                    $P \mathrel{+}= p_{max}$ ▷ Add keypoint
                **end if**
            **end for**
        **end for**
    **end for**
    return $\mathbf{P}$
**end procedure**

**Parameterization** Direct features are parameterized by the 2d image coordinate in a reference frame

$$\mathbf{x} = \begin{bmatrix} x \\ y \end{bmatrix} \tag{2.60}$$

| | |
|---|---|
| $\mathbf{x}$ | 2D point in image coordinates |
| $x, y$ | x,y component of 2d point |

Their 3d position is usually modelled using the depth in a reference frame.

$$\mathbf{X} = \mathbf{K}^{-1}[x, y, 1]d \in \mathbb{R}^3 \tag{2.61}$$

| | |
|---|---|
| $\mathbf{X}$ | 3D point |
| $x, y$ | Pixel coordinates |
| $\mathbf{K}$ | camera intrinsics matrix |
| $d$ | depth value |

To model the image exposure we follow DSO [40] and use an affine transformation to model the global illumination of a frame.

$$\Phi = \frac{1}{t} e^{\alpha}(I - \beta) \tag{2.62}$$

| | |
|---|---|
| $\Phi$ | compensated image |
| $t$ | shutter time |
| $I$ | raw image |
| $\alpha, \beta$ | affine transformation parameters |

**Matching** Without an explicit descriptor direct point correspondences can only be evaluated by comparing the photometric error between image patches around them. In the case where the relative pose between frames is known epipolar line search can be used to find the best matching pixel [39, 40]. However, usually the match does not perfectly lie on the epipolar line due to errors in the pose estimates. Therefore, often the Lucas-Kanade [41] optical flow method is used to find the best match given an initial starting point. Starting from an initial guess, e.g. the pixel position in the previous frame, the method iterative computes a displacement vector based on the alignment of the pixels in a window around a center point. The displacement vector is computed based on the local and temporal gradients.

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \sum_p I_{i,x}(p)^2 & \sum_p I_{i,x}(p) I_{i,y}(p) \\ \sum_p I_{i,x}(p) I_{i,y}(p) & \sum_p I_{i,y}(p)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_p I_{i,x}(p)[I_i(p) - I_j(p)] \\ -\sum_p I_{i,y}(p)[I_i(p) - I_j(p)] \end{bmatrix} \tag{2.63}$$

| | |
|---|---|
| $I_{i,x}, I_{i,y}$ | gradient image of $I_i$ in $x$, $y$ direction |
| $p$ | pixel in window |
| $I_i, I_j$ | image $i$, $j$ |

The procedure is repeated until the photometric error between the aligned windows is smaller than a certain threshold or a maximum number of iterations have been reached.

**Triangulation** In case that the corresponding keypoint was found by epipolar search or projected to the epipolar line a simpler triangulation method computing the 3d line-line intersection can be used. However, due to errors in the relative poses the correspondences can lie off the epipolar line and therefore we use the same DLT based triangulation method as for the descriptive points.

**Photometric Error** The photometric error function transforms and projects a point in frame $i$ onto frame $j$. Then bi-linear sampling is used to interpolate the image intensity at that point to compare it against the intensity value at the 2d observation point in frame $i$.

$$\mathbf{E_P} = I_i(\begin{pmatrix} x_i \\ y_i \end{pmatrix}) - I_j\left(\pi\left(\mathbf{K}\mathbf{T_j}\left(\mathbf{T_i}^{-1}\mathbf{X_i}\right)\right)\right) \tag{2.64}$$

| | |
|---|---|
| $E_R$ | photo-metric error |
| $I_i(\mathbf{x}), I_j(\mathbf{x})$ | bi-linear interpolation of image $i$, $j$ at $\mathbf{x}$ |
| $x_i, y_i$ | pixel coordinates of observation in frame $i$ |
| $\pi$ | projection function (see eq. 2.8) |
| $\mathbf{K}$ | 4x4 camera intrinsic matrix |
| $T_i, T_j$ | 4x4 transformation matrix of frame $i$, $j$ |

#### 2.3.2.4 Orientation Features

In addition to using lines as descriptive features their direction vector can be used for MW orientation estimation. Similarly can the normal orientation of planes and the orientation of surface normals from depth images be used as geometric features.

Zhou *et al*. [42] describe an efficient algorithm to track the rotation of the MW frame given these orientation measurements. Given a set of vectors $\mathbf{V}$ the algorithm processes each MW axis $a$ separately. First, it selects all vectors $\mathbf{V_a}$ that lie within a cone of angle $\alpha$ around the current axis estimate from all vectors $\mathbf{V}$. Then it rotates these vectors onto the corresponding axis using rotation $Q_a$

$$\mathbf{Q_a} = \begin{pmatrix} \mathbf{r_{mod(a+1,3)}} \\ \mathbf{r_{mod(a+2,3)}} \\ \mathbf{r_{mod(a+3,3)}} \end{pmatrix} \tag{2.65}$$

and projects them into the tangent space using the Riemannian $S^2$ logarithm map.

$$\mathbf{p} = \frac{sin^{-1}(sign(v_z)\sqrt{v_x^2 + v_y^2}}{\sqrt{v_x^2 + v_y^2}} \begin{pmatrix} v_x \\ v_y \end{pmatrix} \tag{2.66}$$

Then the mean $\mathbf{m_a}$ is computed using a Gaussian kernel.

$$\mathbf{m_a}' = \frac{\sum_N e^{-c\|\mathbf{p_a}\|^2}\mathbf{p_a}}{\sum_N e^{-c\|\mathbf{p_a}\|^2}} \tag{2.67}$$

It then projects the mean $m_a$ back onto the unit sphere using the Riemannian $S^2$ exponential map

$$\mathbf{V} = \left[\frac{tan(\|\mathbf{m}\|)}{\|\mathbf{m}\|}\mathbf{m}^T, 1\right]^T \tag{2.68}$$

The mean vector is then transformed back from the MW frame to the camera. All three mean vectors are then composed together to form a new rotation matrix that is ortho-normalized using SVD. The whole process is repeated multiple times until the change in rotation falls below a certain threshold. An iteration of the procedure is summarized in algorithm 4.

**Algorithm 4** Mean-shift iteration for MW based rotation tracking

$\mathbf{V}$: Measured orientation vectors
$\mathbf{R_{i-1}}$: Previous or initial MW rotation estimate
**for** $a \leftarrow 1$ to $3$ **do** ▷ For each axis
    **procedure** ESTIMATE ORIENTATION OF MW AXIS
        $\mathbf{r_a} \leftarrow$ row $a$ of $\mathbf{R_{i-1}}$
        $\mathbf{V_a} \leftarrow \{\mathbf{v_i}$ if $\|\mathbf{v_i} \times \mathbf{r_a}\| < sin(\alpha) \forall V\}$ ▷ Select all inside cone
        $\mathbf{Q_a} = [\mathbf{r_{mod(a+1,3)}}; \mathbf{r_{mod(a+2,3)}}; \mathbf{r_{mod(a+3,3)}}$ ▷ R to align $z$ with MW axis $a$
        $\mathbf{V_a} = \mathbf{Q_a}^{-1}\mathbf{V_a}$ ▷ Apply rotation to MW
        $\mathbf{P} \leftarrow \{Log(\mathbf{V_i}) \forall \mathbf{V_a}\}$ ▷ Project on tangent plane
        $\mathbf{m_a} = Mean(\mathbf{V_a})$ ▷ Compute mean
        $\mathbf{M_a} \leftarrow Exp(m_a)$ ▷ Project on sphere
        $\mathbf{M_a} = \mathbf{Q_a}\mathbf{M_a}$ ▷ Apply rotation to camera
    **end procedure**
**end for**
$\mathbf{R_i} = [\mathbf{M_1}; \mathbf{M_2}; \mathbf{M_3}]$ ▷ Compose found axes
$\mathbf{R_i} = \mathbf{U}\mathbf{V}^T$ where $\mathbf{U}, \mathbf{D}\mathbf{V} = SVD(\mathbf{R_i})$ ▷ Orthogonalize rotation

For the first frame there is no initial rotation, so a random set of rotations is sampled and optimized. The found rotations are then clustered and the dominant cluster is selected as the final rotation.

### 2.3.3 Depth Measurement Models

Over time a keypoint is being observed multiple times in different frames, each observation can be used to improve the estimation of the 3d landmark position and estimate a confidence measure about the estimate. Several models have been proposed in the literature, we will look at the most common ones and then explore one probabilistic model in depth as it is extended later in this thesis.

**Heuristic models** In order to do an informed update multiple statistics are tracked for each point in addition to its estimated position. Multiple heuristics and update methods have been proposed. Common strategies are based on the number and quality of the triangulations or based on the size of the re-projection residuals, discarding measurements based on a specified threshold [14, 15].

**Probabilistic models** A probabilistic formulation naturally allows a continuous update of the point depth estimate and uncertainty through new observations. In filtering based SLAM frameworks the landmark position is commonly represented using 2D or 3D positions and their co-variances. In EKF based SLAM a Gaussian measurement model is used to continuously update the mean and variance.

However, in visual SLAM the estimate of a 3d point is often represented using the 2d keypoint in a reference frame and the depth from the camera center to the point. Because the position in the image is often more accurately determined than the depth, usually a only 1-dimensional probabilistic model is used that reflects the depth measurement uncertainty.

The probabilistic models can be derived using Bayesian inference, where we have a prior belief of our model estimate, in our case the current estimate of the depth and a new measurement in the form of a likelihood, from both we can estimate a posterior that accounts for the new evidence.

$$\underbrace{p(Z|\theta_i)}_{\text{Posterior}} \approx \underbrace{p(d_i|Z)}_{\text{Likelihood}} \underbrace{p(Z|\theta_{\mathbf{i-1}})}_{\text{Prior}} \tag{2.69}$$

$Z$ true depth
$\theta$ model parameters
$d_i$ measured depth

We first derive efficient online updates for a simple Gaussian model using approximate inference to give a better intuition of the process. We then look at an extended model that explicitly models outlier measurements and highlight the differences between them.

#### 2.3.3.1 Simple Gaussian Model

We assume a simple Gaussian measurement model, where the probability of the measurement $d$ is given by a normal distribution with the real depth $Z$ as mean and the known measurement noise as variance $\tau^2$.

$$p(d|Z,\tau^2) = \mathcal{N}(d|Z,\tau^2) = \frac{1}{\sqrt{2\pi\tau^2}} \exp\left(-\frac{1}{2\tau^2}(d-Z)^2\right) \tag{2.70}$$

$d$ measured depth
$Z$ true depth
$\tau$ measurement depth variance

**Posterior derivation & approximation** We want to approximate the posterior as a Gaussian $q(Z|\mu,\sigma^2)$. Note that in this case, assuming the variance of the measurement is known the posterior of a a Gaussian likelihood and a Gaussian prior is actually also a Gaussian, which is usually not the case for approximate inference.

$$q(Z|\mu,\sigma^2) = N(Z|\mu,\sigma^2) \tag{2.71}$$

| | |
|---|---|
| $q(Z\|\mu, \sigma^2)$ | approximate posterior |
| $\mu$ | mean |
| $\sigma^2$ | variance |
| $N(x\|\mu, \sigma^2)$ | Gaussian distribution |

Using moment matching between the defined posterior and the actual posterior given by the product of the previous estimate and the new measurement we can derive the online updates.

$$q(Z|\mu_i, \sigma_i^2) := p(d|Z, \tau^2)q(Z|\mu_{i-1}, \sigma_{i-1}^2) \tag{2.72}$$

| | |
|---|---|
| $q(Z\|\mu, \sigma^2)$ | approximate posterior |
| $p(d\|Z, \tau^2)$ | likelihood of the new measurement |
| $q(Z\|\mu_{i-1}, \sigma_{i-1}^2)$ | previous estimate |
| $\mu_i, \mu_{i-1}$ | mean at time $i$, $i-1$ |
| $\sigma_i^2, \sigma_{i-1}$ | variance at time $i$, $i-1$ |
| $\tau^2$ | measurement variance |

We can see that the true posterior on the right-hand side is the product of two Gaussians.

**Online model updates** If we look at the derived updates for the simple Gaussian case we can see that the mean of the posterior Gaussian is the weighted mean of the current estimate $i-1$ and the new measurement $i$ using the variances as weights.

$$\hat{\mu}_i = \left( \frac{\tau_i^2 \mu_{i-1} + \sigma_{i-1}^2 d_i}{\sigma_{i-1}^2 + \tau_i^2} \right) \tag{2.73}$$

| | |
|---|---|
| $\hat{\mu}_i$ | updated mean |
| $\mu_{i-1}$ | previous mean |
| $\sigma_{i-1}$ | previous variance |
| $d_i$ | measured depth |
| $\tau^2$ | measurement variance |

The variance of the posterior Gaussian is the product of the variances divided by their sum.

$$\hat{\sigma}_i^2 = \frac{\sigma_{i-1}^2 \tau_i^2}{\sigma_{i-1}^2 + \tau_i^2} \tag{2.74}$$

| | |
|---|---|
| $\hat{\sigma}_i^2$ | updated variance |
| $\sigma_{i-1}$ | previous variance |
| $\tau^2$ | measurement variance |

#### 2.3.3.2 Gaussian + Uniform Mixture Model

Vogiatzis *et al*. [43] proposed a Bayesian model based on the idea of a mixture model of good and bad measurement models as described in the book of Jaynes [44, Chapter 21 (page 619)]. SVO [39] was the first work that used the model in a SLAM framework. In the following will introduce the details of their proposed model, as it is the basis for an extended model proposed later in the thesis.

In the proposed model the depth measurement is modelled as a mixture of a normal and uniform distribution, weighted by an inlier ratio.

$$p(d|\phi, Z) = \phi\mathcal{N}(d|Z, \tau^2) + (1-\phi)\mathcal{U}(d|Z_{min}, Z_{max}) \tag{2.75}$$

| | |
|---|---|
| $d$ | measured depth |
| $\phi$ | inlier ratio |
| $Z$ | true depth mean |
| $\tau$ | measurement depth variance |
| $Z_{min}, Z_{max}$ | min, max depth boundaries |

**Posterior derivation & approximation** In their paper the authors show that the approximation of the posterior with the smallest KL divergence is a mixture of a normal and beta distribution.

$$q(Z, \phi|\mu, \sigma^2, a, b) = \mathcal{N}(Z|\mu, \sigma^2)\mathcal{B}(\phi, a, b) \tag{2.76}$$

| | |
|---|---|
| $q(Z, \phi\|\mu, \sigma^2, a, b)$ | approximate posterior |
| $\mathcal{N}(Z\|\mu, \sigma^2)$ | Normal distribution |
| $\mathcal{B}(\phi, a, b)$ | Beta distribution |

The Beta distribution is defined as

$$\mathcal{B}(\phi, a, b) = \frac{1}{B(a,b)}\phi^{a-1}(1-\phi)^{b-1} \tag{2.77}$$

Where $B(a, b)$ is the beta function

$$B(a,b) = \int_{x=0}^{1} x^{a-1}(1-x)^{b-1}dx \tag{2.78}$$

By matching the first and second order moments between our posterior approximation and the real posterior resulting from the previous estimate and a new measurement

$$q(Z, \phi|\mu_i, \sigma_i^2, a_i, b_i) := p(d|Z, \phi, \tau^2)q(Z, \phi|\mu_{i-1}, \sigma_{i-1}^2, a_{i-1}, b_{i-1}) \tag{2.79}$$

| | |
|---|---|
| $q(Z, \phi \mid \mu_i, \sigma_i^2, a_i, b_i)$ | approximate posterior |
| $p(d \mid Z, \phi, \tau^2)$ | measurement likelihood |
| $q(Z, \phi \mid \mu_{i-1}, \sigma_{i-1}^2, a_{i-1}, b_{i-1})$ | previous estimate |
| $Z$ | true depth mean |
| $\phi$ | inlier ratio |
| $\mu_i, \mu_{i-1}$ | updated and previous mean |
| $\sigma_i^2, \sigma_{i-1}^2$ | updated and previous variance |
| $a_i, b_i, a_{i-1}, b_{i-1}$ | updated and previous beta parameters |
| $d$ | measured depth |
| $\tau$ | measurement depth variance |

we can derive efficient online updates for the parameters of the normal and beta distributions. We refer to the supplementary material of the original paper for the derivations [43].

### Online model updates

$$\phi = \frac{a}{a+b} \tag{2.80}$$

Looking at the derived updates for the outlier aware mixture model gives us an intuition of how the mixture model differs from the simple Gaussian model. The mean (the first matched moment) update can be understood as a weighted mean of the simple Gaussian update $\hat{\mu}_i$ (see eq. 2.73) (assuming the new measurement is an inlier) and the previous estimate $\mu_{i-1}$ (assuming the new measurement is an outlier).

$$\mu_i = \left[\frac{C_1}{C_1 + C_2}\right] \hat{\mu}_i + \left[\frac{C_2}{C_1 + C_2}\right] \mu_{i-1} \tag{2.81}$$

| | |
|---|---|
| $\mu_i$ | updated mean |
| $C_1, C_2$ | derived variables (see eq. 2.82) |
| $\hat{\mu}_i$ | update using simple Gaussian model |
| $\mu_{i-1}$ | previous mean |

Looking at the two weights $C_1$ and $C_2$ we can see that they are computed as a product of the current inlier or outlier ratio and the probability of the new measurement under the current inlier and outlier models.

$$C_1 = \underbrace{\frac{a_{i-1}}{a_{i-1} + b_{i-1}}}_{\text{Inlier ratio}} \underbrace{\mathcal{N}(d_i \mid \mu_{i-1}, \sigma_{i-1}^2 + \tau_i^2)}_{p(d \mid inlier)} \tag{2.82}$$

$$C_2 = \underbrace{\frac{b_{i-1}}{a_{i-1} + b_{i-1}}}_{\text{Outlier ratio}} \underbrace{\mathcal{U}(d_i \mid Z_{min}, Z_{max})}_{p(d \mid outlier)} \tag{2.83}$$

| | |
|---|---|
| $C_1, C_2$ | derived variables |
| $a_{i-1}, b_{i-1}$ | previous beta distribution parameters |
| $\mu_{i-1}$ | previous mean |
| $sigma_{i-1}^2$ | previous variance |
| $d_i$ | measured depth |
| $\tau$ | measurement depth variance |
| $Z_{min}, Z_{max}$ | min. and max. of uniform depth range |

Looking at the inlier distribution in $C_1$ the probability of the measurement is defined by the marginal probability $p(d|inlier)$, independent of the true depth $Z$.

$$p(d|inlier) = \int_Z p(d|Z)p(Z)dZ = \int_Z \mathcal{N}(d|Z, \tau^2)\mathcal{N}(Z|\mu, \sigma^2)dZ = \mathcal{N}(\mu, \sigma^2 + \tau^2) \tag{2.84}$$

| | |
|---|---|
| $p(d|inlier), \mathcal{N}(\mu, \sigma^2 + \tau^2)$ | probability of $d$ given inlier model and variance |
| $p(d|Z)$ | conditional distribution |
| $p(Z)$ | marginal distribution |
| $\mathcal{N}(Z|\mu, \sigma^2)$ | measurement likelihood |
| $\mathcal{N}(Z|\mu, \sigma^2)$ | probability of current inlier estimate |
| $d$ | measured depth |
| $Z$ | true depth |
| $\tau$ | measurement depth variance |
| $\mu$ | current mean |
| $sigma^2$ | current variance |

The variance $(\sigma_{i-1}^2 + \tau_i^2)$ can be interpreted as propagating the uncertainty of the current estimate $p(Z)$ into the conditional measurement model $p(d|Z)$, effectively summing up the variances.

For the outlier distribution in $C_2$ the probability depends on the considered depth range. The smaller the considered range the higher the probability under the outlier model.

$$p(d|outlier) = \frac{1}{(Z_{max} - Z_{min})} \tag{2.85}$$

| | |
|---|---|
| $p(d|outlier)$ | probability of $d$ under the uniform distribution |
| $Z_{min}, Z_{max}$ | min. and max. of uniform depth range |

The computation of the posterior variance can also be interpreted as a weighted mean of the second moments of the simple Gaussian posterior (assuming an inlier) and the previous posterior (assuming an outlier).

$$\sigma_i^2 = \frac{C_1}{C_1 + C_2} \underbrace{(\hat{\sigma}_i^2 + \hat{\mu}_i^2)}_{\text{Second Moment } p(d|\hat{\mu}_i, \hat{\sigma}_i^2)} + \frac{C_2}{C_1 + C_2} \underbrace{(\sigma_{i-1}^2 + \mu_{i-1}^2)}_{\text{Second Moment } p(d|\mu_{i-1}, \sigma_{i-1}^2)} - \mu_i^2 \tag{2.86}$$

| | |
|---|---|
| $\sigma_i^2$ | updated variance |
| $C_1, C_2$ | derived variables (see eq. 2.82) |
| $\hat{\sigma}_i^2$ | updated variance using simple Gaussian model |
| $\hat{\mu}_i$ | updated mean using simple Gaussian model |
| $\sigma_{i-1}^2$ | previous variance |
| $\mu_{i-1}$ | previous mean |

Compared to the simple Gaussian model in this model we have the additional parameters $a$ and $b$ of the beta distribution for each map point that we update after each measurement.

For an intuition on the update we can look at the first moments of the approximate and true beta distribution posteriors, that is the mean of the inlier ratio. We can see a similar weighted averaging as before. The left inlier term represents the parameters of a beta distribution with an inlier count $a$ increase by 1, whereas the right term represents the outlier case where $b$ has been increased by 1.

$$\phi = \frac{a_i}{a_i + b_i} := \frac{C_1}{C_1 + C_2} \underbrace{\left( \frac{a_{i-1} + 1}{a_{i-1} + b_{i-1} + 1} \right)}_{\text{Inlier increase a}} + \frac{C_2}{C_1 + C_2} \underbrace{\left( \frac{a_{i-1}}{a_{i-1} + b_{i-1} + 1} \right)}_{\text{Outlier increase b}} = f \tag{2.87}$$

| | |
|---|---|
| $\phi$ | inlier ratio |
| $a_i, b_i$ | current beta distribution parameters |
| $C_1, C_2$ | derived variables |
| $a_{i-1}, b_{i-1}$ | previous beta distribution parameters |

A look at the second moment of the beta distributions we can see the same weighted update as for the first moment.

$$\frac{a_i(a_i+1)}{(a_i+b_i)(a_i+b_i+1)} := \frac{C_1}{C_1 + C_2} \underbrace{\left( \frac{(a_{i-1}+1)(a_{i-1}+2)}{(a_{i-1}+b_{i-1}+1)(a_{i-1}+b_{i-1}+2)} \right)}_{\text{Inlier increase a}} + \frac{C_2}{C_1 + C_2} \underbrace{\left( \frac{a_{i-1}(a_{i-1}+1)}{(a_{i-1}+b_{i-1}+1)(a_{i-1}+b_{i-1}+2)} \right)}_{\text{Outlier increase b}} = e \tag{2.88}$$

| | |
|---|---|
| $\phi$ | inlier ratio |
| $a_i, b_i$ | current beta distribution parameters |
| $C_1, C_2$ | derived variables |
| $a_{i-1}, b_{i-1}$ | previous beta distribution parameters |

Solving the system composed by these two linear equations leads to the following update rules for $a$ and $b$. We use $e$ and $f$ to denote the first and second moments of the real posteriors on the right hand side of the equations above.

$$a_i = \frac{e - f}{f - \frac{e}{f}} \tag{2.89}$$

$$b_i = \frac{a_i}{f} - a_i \tag{2.90}$$

$a_i, b_i$ updated beta distribution parameters
$e, f$ first and second moments of true posterior

### 2.3.4 Bundle Adjustment

Bundle Adjustment (BA) [45, 46, 47] is widely used as an offline reconstruction method in the area of photogrammetry and SfM that formulates the pose and map geometry estimation as a joint optimization problem. Graph based SLAM system use it in different variations in the tracking and mapping process. BA is commonly formulated as a non-linear least squares optimization of the feature re-projection or direct photometric errors. In the following we present the main components of the BA optimization. We first introduce the minimal Lie algebra representation for the camera poses that is needed for the optimization. Then we look at the most common method to solve the least-squares problem and discuss robust norms that help to mitigate the effect of outliers in the optimization.

#### 2.3.4.1 Pose Parameterization

In general we use 4x4 transformation matrices to represent euclidean transformations in 3-dimensions.

$$\mathbf{T} = \begin{bmatrix} \mathbf{R}, \mathbf{t} \\ \mathbf{0}, 1 \end{bmatrix} \tag{2.91}$$

$\mathbf{T}$ 4x4 transformation matrix $\in SE(3)$
$\mathbf{R}$ 3x3 rotation matrix $\in SO(3)$
$\mathbf{t}$ 3x1 translation vector $\in \mathcal{R}^3$

For the optimization it is however important that the poses are parameterized in a minimal representation to avoid any gauge freedom. For this the poses are often parameterized using Lie algebra. In this thesis we mostly use the Special Euclidean group in 3 dimensions ($SE(3)$) Lie algebra parameterization for combined rotation ($SO(3)$) and translation ($\mathcal{R}^3$).

$$\mathbf{T} = \left[\omega^T, \mathbf{t}^T\right] \tag{2.92}$$

$\mathbf{T}$ 6d transformation vector
$\omega$ 3d rotation vector
$\mathbf{t}$ 3d translation vector

While the Lie algebra parameterization with 6 parameters for 6 DoF is a minimal one, it does have singularities. Therefore it is only used for incremental steps around the identity transform far away from the singularities. These increments are then applied to the current estimate that is expressed in an over-parameterized form without any singularities using the 4x4 transformation matrix.

We can use the associated logarithm (log) map to convert from matrix to Lie representation.

$$|\omega|_\times = \frac{\theta}{2sin(\theta)}(\mathbf{R} - \mathbf{R}^T) \tag{2.93}$$

$$\theta = \frac{trace(\mathbf{R}) - 1}{2} \tag{2.94}$$

$\omega$ 3d rotation vector
$|\mathbf{x}|_\times$ 3x3 skew-symmetric matrix from 3d vector $\mathbf{x}$ (see sec. 2.11)
$\mathbf{R}$ 3x3 rotation matrix

Using the exponential (exp) map we can convert the Lie representation back to a transformation matrix

$$\mathbf{R} = \mathbf{I} + \frac{sin\left(|\omega|\right)}{|\omega|}[\omega]_\times + \frac{[1 - cos\left(|\omega|\right)]}{|\omega|^2}[\omega]_\times^2 \tag{2.95}$$

$\mathbf{R}$ 3x3 rotation matrix
$\mathbf{I}$ 3x3 identity matrix
$\omega$ 3d rotation vector
$|\mathbf{x}|_\times$ 3x3 skew-symmetric matrix from 3d vector $\mathbf{x}$ (see sec. 2.11)

#### 2.3.4.2 Non-Linear Least-Squares Optimization

In Bundle Adjustment the optimization is usually formulated as a least-squares problem consisting of one term for each observation. In the SLAM setting each error term is often weighted in order to focus on the more reliable measurements e.g. using the inverse variance as a weight.

$$F(\mathbf{x}) = \sum_i^N R(\mathbf{x_i})^2 = \sum_i^N E(\mathbf{x_i})W(\mathbf{x_i})E(\mathbf{x_i}) \tag{2.96}$$

$N$ number of residuals
$R(\mathbf{x_i})$ residual term $i$
$E(\mathbf{x_i})$ error term $i$
$W(\mathbf{x_i})$ weight associated to term $i$

We can collect all residual terms and variables in the sum into a large system of equations. Following the Gauss-Newton approach we then linearize the non-linear error functions using first order Taylor approximation around their current estimate.

$$\mathbf{R}(\mathbf{x}) \approx \mathbf{R}(\mathbf{x} + \Delta\mathbf{x}) = \mathbf{R}(\mathbf{x}) + \mathbf{J}_x \Delta\mathbf{x} \tag{2.97}$$

$R(\mathbf{x})$ residual matrix
$\Delta\mathbf{x}$ linearized increment vector
$\mathbf{J}_x$ Jacobian matrix $\mathbf{J} = \frac{\partial R(\mathbf{x})}{\partial \mathbf{x}}\Big|_x$ evaluated at $x$

Afterwards, we set the derivative with respect the increment $\Delta\mathbf{x}$ to zero in order to find the minimum. We can then rearrange the system to compute an incremental update towards the minimum solution.

$$\Delta\mathbf{x_i} = -(\mathbf{J}^T\mathbf{J})^{-1}\mathbf{J}^T\mathbf{R}(\mathbf{x}) \tag{2.98}$$

$\Delta\mathbf{x}$ linearized increment
$\mathbf{J}_x$ Jacobian matrix $\mathbf{J} = \frac{\partial R(\mathbf{x})}{\partial \mathbf{x}}\Big|_x$ evaluated at $x$
$R(\mathbf{x})$ stacked residual vector

In practice we use the Levenberg-Marquardt method, that combines the second order Gauss-Newton method with simple gradient descent step in order to improve convergence. Here $\lambda$ is adjusted after each iteration of the optimization. It can be increased if the residual change is not very large in order to make larger steps using gradient descent and it can be decreased when there is a significant improvement in the residual to use the second order Gauss-Newton method for faster convergence.

$$\Delta\mathbf{x_i} = -(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})^{-1}\mathbf{J}^T\mathbf{R}(\mathbf{x}) \tag{2.99}$$

$\Delta\mathbf{x}$ linearized increment
$\mathbf{J}_x$ Jacobian matrix $\mathbf{J} = \frac{\partial R(\mathbf{x})}{\partial \mathbf{x}}\Big|_x$ evaluated at $x$
$\lambda$ weight for gradient descent step
$\mathbf{I}$ identity matrix
$R(\mathbf{x})$ stacked residual vector

Due to the dependency of the residual and the Jacobian on the current state of $x$ they need to be re-computed at every iteration. The structure of the resulting matrix $(\mathbf{J}^T\mathbf{J} + \lambda\mathbf{I})$, that needs to be inverted has a special pattern and is very sparse. Depending on the variables that we are optimizing for the matrix can be arranged in a way that allows the use of the Schur complement to make the inversion more efficient. Furthermore, a sparse matrix representation and Cholesky decomposition can be used to speed-up the matrix inversion. [20].

To account for outliers in the optimization often robust M-estimators from statistics like the Tukey bi-weight [14] and Huber norm [38, 15, 40] are used that put less emphasis on residuals above a certain threshold $\epsilon$.

$$|x|_M = \begin{cases} \text{polynomial \& slow,} & \text{if } |x| < \epsilon \\ \text{linear \& fast,} & \text{otherwise} \end{cases} \tag{2.100}$$

In this thesis we use the Huber norm, defined as.

$$|x|_H = \begin{cases} \frac{1}{2}x^2, & \text{if } |x| < \epsilon \\ \epsilon\left(|x| - \frac{\epsilon}{2}\right), & \text{otherwise} \end{cases} \tag{2.101}$$

$x$ residual
$\epsilon$ threshold

### 2.3.5 Deep Neural Networks

In this thesis we leverage Deep Neural Network (DNN) models to predict prior semantic and geometric information from a single RGB image. While we do not propose any new models or train them ourselves we still provide a high level overview of the basic network elements and training process to better understand their potential and limitations for later discussions.

#### 2.3.5.1 Network Architectures

In this thesis we use Convolutional Neural Network (CNN) architectures. These fully convolutional networks (FCN) learn to map the information in the input image pixels into output images with semantic class probabilities or depth and normal estimates. The networks are build up by many layers of linear convolutional layers stacked on-top of each other interlaced with non-linear activation functions to enable the learning of complex functions, normalization layers to keep the network stable and improve learning speed and residual connections to improve gradient flow. FCN networks are usually designed with a encoder-decoder architecture that learns hierarchical features by reducing the spatial dimension of the image and increases the latent channel dimension in the first half of the network and then increases the spatial dimension again in the second half to provide dense predictions. Using skip connections and multiple parallel branches of different scales Feature Pyramid Networks (FPN) [48] are often used to improve the global context reasoning. The convolutional architecture explicitly maintains positional information and shares learned convolution kernels between all the positions in the image making it position invariant and efficient for image processing.

#### 2.3.5.2 Training Process

The networks used in the thesis are trained by supervision using a labeled dataset of image and per-pixel semantic or depth annotations obtained via human labeling or captured by a RGB-D sensor. The quality of the annotations usually represents an upper bound of the achievable network performance. The network parameters are updated using gradient descent based on a loss function and additional regularizers. For semantic segmentation the loss measures the cross-entropy between the predicted and annotated classes. For the depth and normal prediction it evaluates the distance between the predicted and ground truth values. Often momentum based optimizers with a small learning rate are used to train the networks in order to keep the training stable. The iterative

training process is repeated over the training set until the loss has converged or the performance comparison with a validation dataset shows that the performance degrades, which indicates model over-fitting.

#### 2.3.5.3 Inference

After the training the learned network weights can be used on unseen images e.g. to predict priors. Due to limited available training data the generalization capability of the models is usually limited, so that the target domain should be similar to the images that have been seen during training.

### 2.3.6 Metrics

To evaluate and compare the SLAM system outputs we use two different metrics for the pose error as defined by Sturm *et al.* [49] and an additional metric to evaluate the estimated depths of keypoints.

**Pose Metrics** The Absolute Trajectory Error (ATE) measures the root mean squared error over the translation difference between the estimated and ground truth trajectory.

$$ATE = \sqrt{\sum_{i=0}^{N} (t_{est,i} - t_{gt,i})^2} \tag{2.102}$$

$t_{est,i}, t_{gt,i}$ estimated and groundtruth translation for frame $i$

The Relative Pose Error (RPE) computes the relative error between all possible pairs of poses in the trajectory.

$$RPE = \frac{1}{N} \sum_{i=0}^{N} \sqrt{\frac{1}{N-1} \sum_{j \forall j \neq i} \left| \underbrace{(T_{est,j}^{-1} T_{est,j+i})^{-1} (T_{gt,j}^{-1} T_{gt,j+i})}_{\text{Rel. pose error for pair } i \text{ and } j} \right|_t^2} \tag{2.103}$$

$N$ number of frames

$T_{est,j}, T_{est,j+1}, T_{gt,j}, T_{gt,j+1}$ estimated and groundtruth pose for frame $j$, $j+1$

$|x|_t$ extract translation vector

**Depth Metrics** For the evaluation of the estimated depth of the map points we use the percentage of correct depth or depth inlier ratio from Tateno *et al.* [50].

$$DR = \frac{1}{N} \sum_{i=0}^{N} \left| \frac{d_{est,i} - d_{gt,i}}{d_{gt,i}} \right| \text{ where } |x| = \begin{cases} 1, & \text{if } x < \tau \\ 0, & \text{otherwise} \end{cases} \tag{2.104}$$

$N$ number of points

$d_{est,i}, d_{gt,i}$ estimated and groundtruth depth for point $i$

## 2.4 Related Work

In the following we discuss the most relevant related works for this thesis. These can be works that we build on or that we use for comparison. To show the progress of the research we introduce the works in chronological order. Based on the thesis topic we focus on monocular SLAM methods, For more visual SLAM methods based on stereo an RGB-D sensors we refer the surveys of Cadena *et al*. [10]. First we look at indirect and direct monocular SLAM methods using different features and modules. Then we discuss methods designed to handle dynamic environments and methods using semantic information. Lastly, we look at methods that leverage the MW assumption.

### 2.4.1 Monocular SLAM

In the following we will look at the most related monocular SLAM methods to show the evolution of different approaches and put the proposed contributions into context. For a more thorough overview of monocular methods we refer to the survey of Herrera [13].

**PTAM [2007]** PTAM [14] is the first online descriptive graph based monocular slam with parallel tracking and mapping processes. The decoupling of the two processes allows to run efficient visual odometry based tracking at a higher frequency, while the mapping runs at a lower frequency to perform expensive optimizations of the map to improve the global consistency.

**Features:** PTAM uses fast-10 corners [21] as point features and the image patch around the corner as a descriptor for matching. Features are extracted on 4 levels of a resolution pyramid with scale 0.5. The Shi-Tomasi score [51] is used to conduct NMS and thresholding to avoid clutter and remove weak feature points. Each point has also a status, whether it is considered an inlier or outlier based on the last error estimate from the mapping process.

**Initialization:** The system is initialized with the help of the user that moves the camera while optical flow tracking is used to track the FAST corners extracted in the first image until there is sufficient baseline for the estimation of the camera pose via five-point essential matrix estimation in a RANSAC [52] scheme and triangulation of the initial map followed by bundle adjustment for refinement of the poses and map points. The first two frames then become the first keyframes of the system and it continues with the tracking and mapping processes.

**Mapping:** In order to work in real-time the system does not update the map based on every new frame. Instead it uses regularly spaced keyframes that provide sufficient information for the map. This reduces the computational overhead, while discarding frames with small baseline increments. The more distant spacing of keyframes also allows the system to use more computationally heavy methods to refine the pose and map such as bundle adjustment. New keyframes are initialized based on a minimum time to the previous keyframe and minimum distance to the map and number of inliers to ensure no redundant keyframes are added. The map stores a fixed number of keyframes. In addition it keeps a list of map points with a reference to the responsible keyframe and

pyramid level defining the image patch and normal direction used for warping the image patch descriptors. With a map of tens of keyframes the total number of map points can be several thousands. The map is refined using bundle adjustment, that uses the same re-projection error and optimization as used for tracking, just that in this case all poses and map points are optimized jointly. As the complexity of the system scales approximately cubic with the number of keyframes due to the inversion of the linear system matrix and linear with the number of map points local bundle adjustment is performed first in a window around the new keyframe and the last 4 keyframes and all the visible map points in these. Measurements that fall into the outlier region of the Tukey M-estimator are marked as outliers. When the mapping thread is idle it tries to find new observations of points in the keyframes or correct outlier measurements. Global bundle adjustment with all keyframes is only performed if the mapping process is idle as well.

**Tracking:** The frames in-between keyframes are not just thrown away, but tracked based on the current keyframe in order to provide a real-time pose estimate of the current camera pose and improve the tracking and matching of features points over larger baselines. The tracking is based on the re-projection error between projected map features and observations in the new frame. To find correspondences of the map points in the image a simple motion model is used to estimate the pose of the new frame and the map points are projected into the image. Correspondences are then found by affine template matching (using 8x8 pixel patch) via SSD scores around the projected points. Therefore all fast corners within a search radius are evaluated and the one with minimum distance is kept, if the difference falls below a certain threshold. For coarse resolutions the found position is refined via local gradient based optimization. With the found matches the pose of the frame can be refined using the re-projection error in a non-linear Levenberg-Marquardt optimization using the minimal SE3 parameterization of the poses. The use of the Tukey norm for the errors helps to filter out outliers. Each term is additionally weighted by its variance assuming a 1 pixel error.

**Relocalization:** If too many features in a frame are marked as outliers a re-localization is triggered that tries to estimate a relative pose to one of the keyframes in the map.

**DTAM [2011]** DTAM [37] is a dense direct SLAM system therefore it does not use any descriptive features. The system is implemented on a GPU to achieve real-time performance.

**Initialization:** Similar to PTAM the system is initialized using frame-to-frame feature tracking.

**Mapping:** The map consists of n keyframes, where each keyframe has a dense cost volume over inverse depth, that tracks the current estimate of dense per-pixel depths. To get a better estimate of the depth every new frame is integrated into the cost volume, collecting many small baseline stereo measurements. To improve the inverse depth estimate a regularization term based on the Huber norm over the depth gradient weighted by the image gradient is introduced and the final depth is estimated via optimization combining the cost volume data term and regularizer. A new keyframe is created when

the overlap between visible surface of the current keyframe and the new frame falls below a certain threshold.

**Tracking:** The tracking of new frames is done via direct photometric alignment to the current keyframe using the dense depth estimates. A coarse-to-fine scheme is used to improve convergence. The pose is optimized in a non-linear least squares scheme. Residuals with a large photometric error are filtered out as outliers. To find a good initialization for the optimization, first, a relative rotation to the previous frame is found via visual odometry.

**SDVO [2013]** Semi-Dense Visual Odometry [53] is a direct visual odometry method that estimated the poses between a sliding window of keyframes and therefore has no global map.

**Initialization:** A keypoint method is used to estimate the relative pose between the first two frames to bootstrap the system and estimate initial depth estimates.

**Mapping:** Instead of keeping a cost volume for each pixel as in DTAM the inverse depth estimate is modeled with a Gaussian filter. It integrates each new tracked frame with increasing baselines and updates the mean and variance of the Gaussian estimate. To find correspondences for each pixel it chooses the best reference frame and performs epipolar search to find the best correspondence in a 2-sigma range around the initial estimate. With the found observation it triangulates the estimated depth and computes the variance using covariance propagation using geometric and photometric input error models based on the pose and image noise and assumed 1-pixel error as in PTAM. It then updates the current Gaussian estimate by multiplication with the new estimate, as in the Gaussian update. After all pixels have been updated it performs local averaging for smoothness regularization. The local variance is also used to decide which pixels to integrate and which are considered outliers. It further estimates an outlier probability per pixel by keeping track of successful and unsuccessful observations based on stereo search, intensity change and gradient magnitude.

**Tracking:** The tracking is based on a coarse-to-fine (4 pyramid levels) scheme of iteratively re-weighted Gauss-Newton optimizations using the photometric error between the current keyframe and the new frame. The current depth estimate variances are used as weights for each pixel.

**SVO [2014]** SVO [39] is a hybrid method between direct and feature-based.

**Initialization:** It uses the same initialization based on homography estimation as PTAM,

**Features:** Each pyramid level of a new keyframe is split into cells (30x30px) to extract FAST corners with Shi-Tomasi scores for each.

**Mapping:** The mapping is similar to SDVO, also SVO keeps a fixed number of keyframes as a map. As in DTAM and SDVO it integrates each new frame as an observation. For each new frame it first projects the current points from the keyframe onto the new frame and searches for the best correspondence along the epipolar line within a window of 2 standard deviations. In contrast to SDVO it uses an outlier aware proba-

bilistic model that combines a Gaussian and a uniform distribution weighted by the inlier probability, as described by Vogiatzis *et al*. [43], but using the inverse depth instead of the depth due to a larger scene depth. Instead of just filtering out the estimate based on a variance threshold as in SDVO the probabilistic model jointly estimates the depth and outlier ratio. The estimated depth together with the assumed variance of 1-pixel error is then integrated into the probabilistic mixture depth model. The inlier ratio is used to estimate if a point has converged and can be used for tracking.

**Tracking:** As in SDVO, the hybrid tracking first estimates an initial pose and correspondences via direct image alignment using the current depth estimates from the previous frame using 4x4 pixel patches and Gauss-Newton optimization in a coarse-to-fine-scheme. Differently, it then projects all map points into the new frame and finds correspondences using Lukas-Kanade local optical flow optimization using 8x8 pixel patches. It then converts the refined correspondences into keypoint observations that it uses to refine the pose using re-projection errors, first using pose-only BA followed by points-only BA and full BA.

**LSD-SLAM [2014]** LSD-SLAM [38] is a semi dense direct SLAM method that is able to build a large scale map. The local mapping and tracking are based on SDVO.

**Initialization:** LSD-SLAM is able to initialize with a random depth and high variance initialization for the pixel depths of the first frame once there is sufficient translational motion.

**Mapping:** The map update is the same as in SDVO, accumulating small baseline stereo measurements in a Gaussian probabilistic model. Here the focus lies on building a larger consistent map.

**Loop closure:** Once a new keyframe is needed, the old one is added to a pose graph that makes up the map. Sim3 transforms are estimated to nearby keyframes to align the scale and check for potential loop closures (using OpenFABMAP [54]) and to perform global optimization, therefore the mean depth in each frame is scaled to 1. A pose graph optimization in Sim3 is proposed based on the photometric and depth residuals to constraint the scene scale. The depth estimates from the previous keyframe are propagated to the new keyframe. Loop closure detection and pose graph optimization are run in the background.

**Tracking:** The tracking is also the same as in SDVO using pose estimation against the current keyframe with the help of the photometric error.

**ORB-SLAM [2015]** The ORB-SLAM [15] pipeline combines tracking, mapping, relocalization and loop-closure detection modules. The tracking, mapping and loop closure detection are running in parallel threads.

**Features:** ORB-SLAM uses ORB features for tracking and loop closure detection. To allow the extraction of smaller and larger features and achieve some scale invariance a multi-scale image pyramid is build.

**Initialization:** ORB features are extracted on the first two frames and matches are found through descriptor matching. Based on these correspondences two different ini-

tialization strategies are tested in parallel. Each strategy tries to fit either a homography or an essential matrix using 4 or 8 point pairs in a RANSAC process. The found solutions are then checked and scored based on the triangulated points re-projection errors and parallax. The best solution is then refined via BA.

**Mapping:** In ORB-SLAM the map consists of keyframes, map points, a covisibility graph between keyframes, and an essential graph given by the minimum spanning tree over all keyframes. The graphs are cached structures that help finding neighboring frames and provide a sparse graph for global pose graph optimization. For each new keyframe existing points are transferred from the previous keyframe via projection during the tracking process and new map points are created by triangulation with all co-visible keyframes using descriptor matching and an epipolar check. Then a local BA optimization of the co-visible keyframes and visible map points is run keeping the other keyframes and map points fixed. A survival of the fittest selection of keyframes and map points is performed. Map points are culled based on a minimum number of co-visibility in other keyframes. Keyframes are culled, if most of the keypoints are also visible in one of the neighboring keyframes.

**Tracking:** First a relative pose is estimated with respect to the last or re-localized frame. For the normal case against the last frame a constant velocity motion model is assumed as an initial pose for the new frame and all 3d points from the keyframe are projected into the new frame. Correspondences are then searched for in a small area around the projected points. In case of re-localization the keyframe candidate 3d keypoints and their correspondences to the new frame are used in a RANSAC PnP scheme to find an initial pose. Pose-only BA is then used to estimated the relative pose. This initial pose estimate is then used to refine the pose by using the re-projected points from a local map built from the covisibility graph of the current keyframe in a pose-only BA. ORB-SLAM uses the g2o [55] library for BA optimization based on the minimal $SE(3)$ representation and re-projection error terms using the 1-pixel variances as weights and the robust Huber norm to mitigate the effect of outliers. During the optimization increasingly strict thresholds on the $\chi^2$ measure based on the re-projection error and covariance of each point are applied to remove observations that are considered as outliers.

**Loop closure:** ORB-SLAM relies on the DBoW2 [56] library for bag of words based matching of features for loop closure candidate selection. Once a candidate is found a $Sim(3)$ transformation between the 3d-3d correspondences of the frames is estimated using the method of Horn in a RANSAC process. Afterwards, the map is optimized using global BA on the essential minimum spanning tree graph using $Sim(3)$ transforms to account for the scale drift.

**PL-SLAM [2017]** Pumarola *et al*. [34] extend ORB-SLAM with line features to increase the robustness in low-textured environments.

**Features:** PL-SLAM use ORB features from ORB-SLAM and extends them with LSD line detections and associated LBD descriptors.

**Initialization:** The authors propose a new method for initialization that only relies on line features observed in three frames.

**Mapping & Tracking:** A re-projection error for lines based on the distance from the projected endpoints to the line observations is integrated into the BA framework of ORB-SLAM allowing the joint use of points and lines for tracking and mapping. See section 2.3.2.2 for the definition of the line re-projection error.

**DSO [2018]** DSO [40] is a visual odometry method as points and frames get marginalized out after a while. The method performs a thorough photometric camera calibration to increase the robustness of the direct photometric alignment. An affine global exposure model with two additional parameters per image is used to account for intensity variations.

**Features:** Feature extraction is based on a grid with local thresholds and multiple stages of increasing block size and decreasing thresholds to generate an equal distribution over the image. The grid size is adapted continuously to match the desired number of key points per image.

**Initialization:** A direct multi-scale photometric optimization is used for initialization, similar to LSD-SLAM.

**Mapping:** The map consists of a window of up to 7 keyframes. New keyframes are created when there is a large mean optical flow between the new frame and the current keyframe or a larger change in exposure time is observed. When a new keyframe is created all active points from the previous keyframe are propagated by projection. New points are extracted iteratively using the multi resolution grid. Before the points can be used for tracking they need to be initialized to get an initial depth. For this the minimum photometric error along the epipolar line is searched. Next, the depth is triangulated and the variance is estimated. New points are activated once old ones get marginalized. The points furthest away from the existing points get activated first until a target number of points (2000) is reached. Extract many candidates per image (2000), but sample active points from current key frame window to ensure there are sufficient points to sample from. Points are discarded if the found epipolar minimum is not distinct or the photometric residual is larger than a threshold. Keyframes are created frequently and then pruned if not needed as in ORB-SLAM.

**Tracking:** Tracking is based on a direct photometric BA of the new frame to the current keyframe window using a constant motion model for initialization. A coarse-to-fine pyramid approach is used based on the Gauss-Newton method using the Huber cost function for robustness.

### 2.4.2 Dynamic SLAM

In many applications of SLAM some kind of dynamic elements can be found. These can be independently moving agents or movement caused by external forces such as the wind. The dynamic motions can be rigid or deformable, stronger or subtle and fast or slow. Therefore, most monocular SLAM methods were designed under the assumption that there might be some dynamic objects in the scene, but that their effect is limited to a few outliers.

The challenge to differentiate between static points and dynamic outliers is especially difficult for monocular approaches as the depth estimates are based on temporal triangulation, while methods using direct instantaneous depth measurements from stereo, RGB-D or LiDAR know that their depth is not affected by the dynamics of the objects.

Most graph optimization based SLAM works employ robust norms to build M-estimators with the goal to filter out or down weight measurements with large residuals. Common choices are the Tukey [14] and Huber norms [38, 15, 40]. Some also apply thresholds on the error residuals to mark observations as outliers [37, 15] and discard them. In the following we will focus on monocular methods, for a broader overview we refer to the survey from Wang *et al*. [57].

**RTMSMB-SLAM [2010]** Realtime Motion Segmentation based Multibody Visual SLAM [58] extends a similar pipeline as PTAM to the estimation of the relative camera pose to multiple objects in addition to the static background. To disentangle the different motions a motion segmentation is performed first. Therefore dense features are tracked between the images and the tracks are clustered by their optical flow vectors. Then, for each cluster the pose is initialized using essential matrix estimation and they are tracked using 2d-3d correspondences in a RANSAC scheme and refined with bundle adjustment. The clustering of dynamic objects reduces the number of outliers for the static motion estimation. Additionally, the occlusion of static 3d points due to dynamic objects is modeled via a convex hull over their clustered points.

**Robust vSLAM [2011]** Shimamura *et al*. [59] add dynamic object removal to PTAM. For all the outliers estimated by the Tukey norm they estimate the optical flow between the re-projected map point and the found correspondences in the new frame. Outliers due to bad matching caused by weak or repetitive textures are filtered out via correlation of the keypoint patch with patches in the neighborhood, if the distance to the second best match is small it is likely an outlier due to bad matching and it is removed. From the flow vectors of the remaining points an angle histogram is build. Assuming similar flow directions for a dynamic object a GMM is fitted to the data in order to identify dynamic object clusters. The points from the clusters below a minimum mixture coefficient are then removed from the map.

**RTMBV-SLAM [2011]** This work [60] by Kundu *et al*. extends RTMSMB-SLAM to increase the robustness of the system. Feature matching in conducted in two stages, first with large search space to include moving objects, then a motion segmentation is performed. Once an initial camera motion is estimated a refined matching with more features is conducted. A variant of EPnP and bundle adjustment are used for tracking. Here each feature is modelled by a recursive Bayes filter based on Bearing only Tracking (BOT) using a particle filter to model the current depth estimate and solve the scale ambiguity between the different cluster poses. This is enabled with the help of ground plane detection in the static world and constraints for the dynamic objects as well as motion models constraining the movement of objects.

**DMB-SfM [2012]** Dense Multibody Motion Estimation and Reconstruction [61] is an offline SfM method that uses a hill climbing approach to optimize the joint problem of depth estimation, motion cluster assignment and rigid motion estimation by alternating the update of each set of variables. The depth map estimation is similar to DTAM combining the photometric error with spatial regularization. The rigid body pose estimation is also analogous to the tracking in DTAM. The rigid-body assignment is updated using a graph cut algorithm based on the current estimates for depth and motions. Using a GPU the runtime of the method is in the order of 10 seconds per image.

**Robust SLAM [2013]** Robust SLAM [62] uses a similar architecture to PTAM, but uses SIFT [17] features instead of FAST corners and image patches to enable efficient and robust matching. They propose a way to detect and update the 3D keypoints in the map due to appearance change or motion. Therefore, for each new keyframe they find the 5 closest keyframes by comparing the color histograms in order to check if there are any visual differences. If there is a significant visual difference all 3d points from that keyframe are projected into the new frame and a patch based search is conducted to find the best alignment in the area around the projected point. To differentiate between appearance changes and occlusions a heuristic based on the neighboring keypoints is proposed. If the number of outliers in a keyframe surpasses a certain threshold the whole keyframe will be removed. In addition to the detection and removal of keypoints and keyframes PARSAC a variant of RANSAC with a prior assumption on the point distribution is proposed. In scenes where the number of dynamic points is larger than the number of static points regular RANSAC will most likely select hypothesis on the majority dynamic cluster. Therefore, the proposed method divides the image into a grid (10x10 cells) and randomly samples points from the grid based on a cell probability defined by the inlier ratio from the previous frame to generate more uniformly distributed samples. For the evaluation of the hypothesis the covariance matrix of the 2d inlier points that fit the hypothesis is computed and used as a score instead of the inlier ratio. The proposed variant allows the method to prefer solutions with fewer but evenly distributed supporting samples over hypothesis with many supporting samples in a small region.

### 2.4.3 Semantic SLAM

The construction of a semantic map is a goal in many applications. Therefore, several methods have been proposed to integrate semantic information into SLAM frameworks in order to consolidate semantic measurements in a global map. The majority of these methods were proposed for modalities that provide dense depth in order to create complete maps. Also, most works do not use the semantic information to explicitly remove dynamic objects and only a few works leverage the semantic information in the map for pose estimation. In the following we will focus on the most related works to the monocular setting addressed in this thesis. For an overview of other works on semantic SLAM we refer to the survey from Pu *et al*. [63].

**JSS3DR [2014]** In Joint Semantic Segmentation and 3D Reconstruction from Monocular Video Kundu *et al*. [64] use independently estimated SfM camera poses and sparse point clouds together with the semantic segmentations of the images to build a dense semantic 3d reconstruction of the scene using dense CRFs over 3d voxels. In their process they estimate the variance between multiple semantic observations and reject points where the semantic labels do not agree. Due to the dense 3d CRF the method is not able to run in real-time.

**Dynamic Body VSLAM [2015]** Dynamic Body VSLAM [65] is a stereo method that allows them to combine stereo depth with optical flow to segment scene flow clusters using CRFs. The combine a boosting based semantic segmentation model with the difference between the observed optical flow and the estimated optical flow of a static world to estimate a separate transformation for each dynamic object. They also predict the compatibility between semantic class and motion hypothesis using a boosting framework. The static world and dynamic objects are first tracked using SIFT features and 2d-3d correspondences and then optimized using bundle adjustment. The stereo depth helps to align the independent trajectories of the dynamic objects with the scale of the static environment. The semantically informed motion segmentation removes many outliers for the camera pose estimation.

**IDSSF [2015]** In Incremental Dense Semantic Stereo Fusion for Large-Scale Semantic Scene Reconstruction [66] Vineet *et al*. aim at dense semantic 3d reconstruction from stereo videos. They use a TSDF surface representation combined with voxel hashing as a compact sparse representation and combine it with a 3d CRF to jointly reason about the geometry and semantics. They propose an efficient mean field update implemented on a GPU that allows real-time usage. By leveraging the semantic information they can adjust the weight of each measurement in the fusion process that helps with the removal of spurious trails from dynamic objects.

**CNN-SLAM [2017]** In CNN-SLAM Tateno *et al*. [50] use dense depth prediction as priors in a dense direct SLAM framework based on LSD-SLAM. They use a simple weighted online update to fuse new semantic measurements, predicted by a CNN from the RGB image, into a global surfel map.

### 2.4.4 Manhattan World SLAM

The work from Straub et al. [67] and Zhou et al. [42] has shown that the main cause of the drift in feature based SLAM systems stems from the estimation of the rotation. In many environments designed by humans, like the rooms inside buildings or the buildings along a road or even the roads inside a city, there exist a certain global structure. One of these assumptions is the Manhattan World (MW) that is defined by a set of three orthogonal orientations. Given the assumption that some of the observed planes and lines in a scene align to the global structure it is possible to estimate the per-frame rotation

with respect to a global coordinate system avoiding the accumulation of frame-to-frame rotation estimation errors. As there exist three indistinguishable orientations the orientation of the previous frame is usually used as a reference to resolve the ambiguity. This assumption limits the frame-to-frame rotation to less than 45 degrees. MW-based rotation estimation also works in pure rotation and small parallax scenarios where regular approaches struggle. With the rotation estimated fewer point features are needed to estimate the translation.

Over the past years there have been several works that aim to leverage the Manhattan world assumption in a SLAM pipeline. In the following we list and briefly discuss the most related works, that inspired our approach and to which we will compare ourselves later on. We include both monocular and RGB-D methods. For a more thorough overview we refer to the survey of Zhao *et al*. [68].

**LCVP SLAM [2012]** In Loop Closure Through Vanishing Points in a Line-based Monocular SLAM [69] Zhang *et al*. propose a monocular SLAM system that tracks the camera pose in 2 dimensions assuming a motion on a ground plane. They leverage dominant vanishing points in an EKF based slam. In each image the dominant vanishing point is estimated and used as an observation to compare against a vanishing line in the map. The authors derive that the error measurement model error limits the number of vanishing points to 8 equally spaced directions to be able to resolve the assignment problem robustly, which is more than in the MW, with 3 orthogonal directions.

**StructSLAM [2015]** StructSLAM [2] is a monocular full 3d SLAM system. Zhou *et al*. build line map features that are aligned to the MW frame orientation preventing the accumulation of rotational drift. To build the aligned lines they estimate the dominant directions using the vanishing points of clustered parallel lines in the image. They then track the camera pose using a EKF with point and line end-point re-projection errors.

**RTMF [2015]** In Real-time Manhattan World Rotation Estimation in 3D Straub [67] *et al*. propose different methods to estimate the rotation between a depth or normal image and the MW frame. They propose and compare three MAP approaches to estimate the Manhattan world orientation from a normal map that was build from RGB-D depth images. Two approaches are based on the tangent space of the Riemannian manifold unit sphere $S^2$ where the measurements are modelled by a Gaussian distribution in tangent space. While a direct approach needs to iterate multiple times over all measurements, an approximate solution is proposed that uses a pre-computed Karcher mean. The last method is based on a von Mises-Fisher distribution directly on the unit sphere instead of using the tangent space that also allows the pre-computation of a sum over all measurements that is then compared against the current axis estimate in each iteration. It is a theoretical paper focusing on the rotation estimation, that does not provide a full SLAM pipeline.

**DaC-MW [2016]** Divide and Conquer: Efficient Density-Based Tracking of 3D Sensors in Manhattan Worlds [42] presents a full RGB-D based VO system that leverages the MW assumption for rotation estimation. Zhou *et al*. propose a method to estimate the rotation between the camera and the Manhattan world based on mean shift clustering of plane orientations on the unit sphere. The algorithm is explained in detail in section 2.3.2.4. With available depth measurements from the RGB-D sensor for the translation estimation they propose to first align the pointclouds of both frames using the estimated rotation and then compute and align the density distributions in each of the 3 orthogonal directions using a sampling based method.

**OPVO [2017]** OPVO [70] is a full RGB-D SLAM system. As in DaC-MW they extract normal vectors from the depth image and use the mean shift algorithm to estimate the orientation to the Manhattan world. For the translation estimation they use Shi-Tomasi [51] features that are based on the Harris corner detector [22] and use the KLT-tracker [41] to find correspondences. The matched points are then aligned using the sensor depth and minimizing the re-projection error using Levenberg-Marquardt optimization.

**LPVO [2018]** With LPVO [3] Kim *et al*. present a full RGB-D SLAM system. They propose to use the vanishing points of lines in addition to the normal map estimated from the depth image in the mean-shift rotation estimation method from DaC-MW. To estimate the translation they combine points with and without depth. For points with depth they use the same error as OPVO and derive additional constraints for points without depth. These are also discussed in section 2.3.2.1. The resulting combined least squares problem is solved using Levenberg-Marquardt optimization.

**MonoSR-MW [2018]** In A Monocular SLAM System Leveraging Structural Regularity in Manhattan World [1] Li *et al*. propose a full monocular SLAM system that uses PL-SLAM [34] for initial tracking and mapping and extends it with an additional parallel optimization process that refines the pose of the keyframes using parallel, orthogonal and homography constraints between the features in the map and the MW. Parallel and orthogonal constraints to the MW frame are established via the vanishing points of the lines. The co-planar constraints are build from characteristic lines and plane homographies. For the rotation refinement they use parallel and orthogonal constraint between the lines and the global MW frame. The translation is also optimized using co-planar constraints between point and line features. Finally the 3d lines in the map are refined using the parallel and orthogonal constraints.

# 3 Semantic Priors for Dynamic Scenes

Dynamic objects are common occurrences in many robotic applications. Imagine an autonomous robot moving through a busy factory floor, a public space crowded with humans or driving in high traffic areas. The main limitation of existing SLAM methods using geometric variance and optical flow based filtering (see section 2.4.2) is the assumption that the majority of observations have to lie on the static background in order to remove inconsistent motions. This assumption might not hold true in situations where a significant amount of the sensor field of view is covered by dynamic objects.

In order to overcome this assumption, further understanding and decomposition of the scene is needed. Learning and especially neural network based methods have lately shown great potential in segmenting a scene into different semantic classes and instances [71, 72]. By knowing where in the sensor's field of view potentially dynamic objects are located it is possible to primarily concentrate on the parts of the scene that appear static with a high confidence. A straight forward way would be to mask out all areas that belong to potentially dynamic entities. However, in some scenes this might not leave enough static features to estimate the ego pose. While there might be some parts in the scene that are in general movable or able to move, they might be static at the moment and could therefore be used for the ego pose estimation. Imagine an empty parking lot that has a lot of repetitive patterns, here the parked cars can provide valuable information for localization. Therefore, instead of simply removing all potentially dynamic parts in the image we use the semantic information as a prior in a probabilistic measurement model. The model additionally uses on the estimated geometric variance to estimate an inlier ratio for each feature, allowing the system to leverage potentially dynamic objects as landmarks, if they appear static at the moment. The semantic prior allows us to rely more on points in the areas that were estimated to be static and enable them to converge faster to a stable point, while requiring more consistent observations for potentially dynamic points. The probabilistic model continuously fuses multiple measurements of the semantics over time, resulting in a more robust estimate of the points semantics.

To partially mitigate the smaller number of static observations, due to the probabilistic filtering, we combine descriptive and direct features to enable pose estimation even in highly dynamic scenarios where large parts of the image are occluded by dynamic objects. While the use of descriptive features enables global pose estimation even under large relative motions, additional direct features can extend the operational domain to low-textured scenes where not sufficient descriptive features are available and provide additional constraints in the pose refinement.

## 3.1 System Overview

The proposed graph-based SLAM system is based on the popular ORB-SLAM method [15], that was described in 2.4.1. We first give a high level overview of the SLAM system in figure 3.1 to highlight the proposed changes.

- For each new frame we predict the per-pixel semantic probabilities.
- On each new keyframe we extract direct features and create map points via triangulation using optical flow correspondences.
- We model both direct and descriptive map points with a probabilistic measurement model that jointly models depths and outlier ratio and is efficiently updated with each new measurement.
- In the tracking and mapping we then only use map points that are considered as inliers at that moment in time.

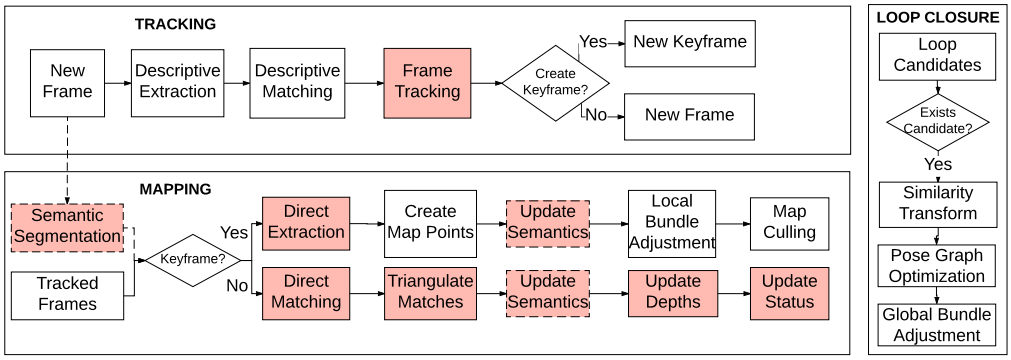



**Figure 3.1:** Overview of the different modules and steps of the SLAM system with differences highlighted in red. Taken from [5] (Copyright ©2018, IEEE)

Next we discuss the details of the proposed method and the ideas behind them.

## 3.2 Descriptive and Direct Features

In the following, we describe how we use descriptive and direct feature points in a unified way, from the feature extraction, the key point parameterization over the matching and triangulation to their use in the optimization.

### 3.2.1 Descriptive

We use ORB features as in ORB-SLAM [15] (see also section 2.3.2.1), but introduce an alternative parameterization and show how we can leverage the semantics in the matching process.

**Parameterization** In order to simplify the probabilistic model and unify the map point parameterizations we decided to replace the 3d map points representation in $[X, Y, Z]^T$ coordinates with an inverse depth parameterization based on the pixel coordinates and inverse depth in a reference keyframe, as in SVO [39].

$$\mathbf{P}_w = \mathbf{T}_{wc}\mathbf{P}_c = \mathbf{T}_{wc}\mathbf{K}^{-1}\begin{bmatrix} x/d \\ y/d \\ 1/d \\ 1 \end{bmatrix} \tag{3.1}$$

| | |
|---|---|
| $\mathbf{P}_w$ | 3D point in world coordinates |
| $\mathbf{P}_w$ | 3D point in reference keyframe coordinates |
| $x, y$ | 2D pixel coordinates in reference keyframe |
| $d$ | inverse depth along the z-axis |
| $\pi(\mathbf{X})$ | pinhole projection |
| $\mathbf{K}$ | 4x4 camera intrinsic matrix |
| $\mathbf{T}_{wc}$ | 4x4 pose matrix of reference keyframe |

This parameterization has the advantage of being able to represent points at infinity using a depth value of zero, that otherwise would be difficult to represent numerically. While these points are not useful for the estimation of the translation we can still use them to estimate the rotation. Additionally, Civera *et al*. [73] have shown that the inverse depth tends to be more Gaussian distributed than the regular depth.

**Extraction** Following ORB-SLAM [15] we use the same multi-level and quadtree based extraction procedure to obtain descriptive ORB features, see section 2.3.2.1 for the details.

**Matching** For the matching we follow ORB-SLAM [15] using the Hamming distance between the binary BRIEF descriptors, see also section 2.3.2.1. We further use the Hamming distance as a measure of the matching accuracy.

$$\alpha_{desc} := 1 - \min\left(1, \frac{d(f_i, f_j)}{d_{max}}\right) \tag{3.2}$$

| | |
|---|---|
| $\alpha_{desc}$ | matching accuracy |
| $f_i$, $f_j$ | BRIEF descriptors of the two key points |
| $d(f_1, f_2)$ | Hamming distance |
| $d_{max}$ | maximum distance for normalization |

**Triangulation** For the triangulation we also follow ORB-SLAM and then convert the obtained 3d point into the inverse depth parameterization.

$$\begin{bmatrix} x \\ y \\ 1/d \end{bmatrix} = \pi(\mathbf{K}\mathbf{T}_{cw}\mathbf{X}_w) \tag{3.3}$$

| | |
|---|---|
| $x, y$ | 2D pixel coordinates in reference key frame |
| $d$ | inverse depth along the z-axis |
| $\pi(\mathbf{X})$ | pinhole projection |
| $\mathbf{K}$ | 4x4 camera intrinsic matrix |
| $\mathbf{T}_{cw}$ | 4x4 pose matrix of reference keyframe |
| $\mathbf{X}_w$ | triangulated 3D point in world coordinates |

### 3.2.2 Direct Features

The direct features are based on the sparse features proposed in DSO [40].

**Parameterization** We follow the inverse depth parameterization of DSO [40] and SVO [39] as for the descriptive keypoints.

To compensate for global illumination changes due to different exposure times, sensor gains or changes in the overall scene illumination we employ an affine model estimating the parameters $\alpha$ and $\beta$ for each frame as done in DSO [40] jointly with the poses and map points during optimization, see section 2.3.2.3.

**Extraction** We follow Engel *et al*. [40] and extract direct features on the key frames using a multi-resolution image pyramid and a grid based selection strategy with a local threshold to obtain well distributed features over the whole image, see section 2.3.2.3. Direct features are only extracted on the key frames reducing the computational overhead compared to the descriptive features.

**Matching** In DSO [40] new features are first tracked over multiple frames to estimate an initial depth. Therefore, in each frame the best match is searched along the epipolar line. In contrast, we use a more relaxed procedure by first searching for matches using Lukas-Kanade optical flow estimation [41] (see also 2.3.2.3) around the expected position of the key point. We then apply an epipolar check that rejects matches that deviate too much from the epipolar line. This allows us to compensate for pose errors that can shift the actual match away from the epipolar line.

For the direct features we determine the matching accuracy based on the photometric difference between the two normalized image patches.

$$\alpha_{dir} = 1 - \min\left(1, \frac{\Delta\Phi}{\Delta\Phi_{max}}\right) \tag{3.4}$$

$$\Delta\Phi = \sum_{x=x_i-W}^{x_i+W} \sum_{y=y_i-H}^{y_i+H} (e^{\alpha_i}\Phi(x_i, y_i) + \beta_i) - (e^{\alpha_j}\Phi(x_j, y_j) + \beta_j) \tag{3.5}$$

| | |
|---|---|
| $\alpha_{dir}$ | matching accuracy |
| $\Delta\Phi$ | SSD between the patches |
| $\Delta\Phi_{max}$ | normalization constant |
| $e^{\alpha_i}, e^{\alpha_j}$ | linear affine illumination model parameter for frame $i$, $j$ |
| $\beta_i, \beta_j$ | constant affine illumination model parameter for frame $i$, $j$ |

**Triangulation** We follow the same triangulation procedure as for the descriptive keypoints.

### 3.2.3 Joint Optimization

We combine both descriptive and direct errors via the respective weights $\eta_R$ and $\eta_P$ to account for the different units between re-projection error in pixels and photo-metric error in image intensity.

$$E = \eta_R \sum_M E_R \Sigma_{R,i}^{-1} E_R + \eta_P \sum_N E_P \Sigma_{P,i}^{-1} E_P \tag{3.6}$$

| | |
|---|---|
| $\eta_R$ | weight for re-projection error |
| $M$ | number of descriptive points |
| $E_R$ | re-projection error |
| $\Sigma_{R,i}$ | re-projection variance |
| $\eta_P$ | weight for photometric error |
| $N$ | number of photometric points |
| $E_P$ | photometric errror |
| $\Sigma_{P,i}$ | photometric variance |

Following ORB-SLAM [20] we use covariance scaling, weighting each error term by its corresponding inverse covariance matrix reflecting the individual measurement uncertainty.

## 3.3 Probabilistic Keypoint Model

Next, we describe the proposed probabilistic keypoint model combining semantic priors with geometric measurements and matching accuracy to estimate an inlier ratio, that decides whether a point is used for tracking or not. The proposed model is an extension of the one from Vogiatzis *et al*. [43, 39] discussed in section 2.3.3.2. We define an extended measurement model while keeping it compatible with their approximate posterior mixture of a Normal and Beta distribution, leading to similar derivations and efficient approximate updates. In the following we will highlight the changes in the proposed measurement model and corresponding updates.

**Depth Measurement Model** As in SVO [39] we use the inverse depth instead of the regular depth in our probabilistic model, because it better fits the Gaussian noise assumption especially in larger distances.

In order to incorporate the matching accuracy into the model we extend it with an additional outlier term, similar to the one for depth measurement outliers.

$$p(d|\phi, Z) = \alpha \left( \underbrace{\phi \mathcal{N}(d|Z, \tau^2) + (1-\phi)\mathcal{U}(d|Z_{min}, Z_{max})}_{\text{Base model}} \right) + (1-\alpha)\mathcal{U}(d|Z_{min}, Z_{max}) \tag{3.7}$$

| | |
|---|---|
| $d$ | measured depth |
| $\phi$ | inlier ratio |
| $Z$ | true depth |
| $\alpha$ | matching accuracy |
| $\tau$ | measurement depth variance |
| $Z_{min}, Z_{max}$ | min, max depth boundaries |

Intuitively the model from Vogiatzis *et al*. counts the number of inlier and outlier observations using the $a$ and $b$ parameters of the Beta distribution. In their case a non-informative prior for the inlier ratio is assumed in the posterior, as no further information was available to make an estimate. We propose to use the semantic information as a prior. When we define a Beta distribution as the prior, it can be interpreted as additional observations counts $a_{sem}$ and $b_{sem}$ in the inlier model with different ratios based on the predicted static and dynamic classes probabilities.

$$q(Z, \phi|D, S) = \underbrace{\mathcal{N}(Z|\mu, \sigma^2) Beta(\phi, a_{obs}, b_{obs})}_{\text{Base model}} \underbrace{Beta(\phi, a_{sem}, b_{sem})}_{\text{Semantic Prior}} \tag{3.8}$$

| | |
|---|---|
| $Z$ | estimated real depth |
| $\phi$ | estimated inlier ratio |
| $D = \{d_1, ..., d_N\}$ | set of depth measurements |
| $S = \{s_1, ..., s_N\}$ | set of observations with semantic information |
| $\mu$ | estimated inverse depth mean |
| $\sigma^2$ | estimated inverse depth variance |
| $a_{obs}, b_{obs}$ | $a$ and $b$ parameters of the base model |
| $a_{sem}, b_{sem}$ | $a$ and $b$ parameters of the semantic model |

**Semantic Measurement Model** Given the RGB camera image $I_i$ we use a neural network $CNN(c_k|I_i)$ to predict a semantic segmentation map that assigns a probability to each pixel indicating how likely it belongs to semantic class $c$.

To account for incorrect matches we propose to use an outlier model combining the output of the CNN with a uniform outlier distribution based on the matching accuracy.

$$p(c_k|I_i) := \alpha_i CNN(c_k|I_i) + (1-\alpha_i)\mathcal{U}(0, 1) \tag{3.9}$$

| | |
|---|---|
| $c_k$ | binary variable for class $k$ |
| $I_k$ | Image $k$ |
| $p(c_k\|I_i)$ | probability of pixel belonging to class $k$ |
| $\alpha_i$ | matching accuracy of the measurement in image $i$ |
| $CNN(c_k\|I_i)$ | predicted probability of pixel belonging to class $k$ |
| $\mathcal{U}(0,1)$ | uniform probability distribution |

**Online Updates** Similar to Vogiatzis *et al*. we can derive efficient online parameter updates using first and second order moment matching against the full posterior, see section 2.3.3.2. In the following we will present and discuss the derived update equations. We refer to the appendix for the full derivations in section A.1.

**Semantics** For the map point semantics we obtain an approximate posterior distribution where the semantic measurements from the network are weighted by the matching accuracy to obtain the current estimate of the map points semantics.

$$q(c_k|S) = \underbrace{\frac{1}{\sum_k \prod_{i=1}^{N} CNN(c_k|I_i)^{\alpha_i}}}_{\text{Normalization over all classes}} \underbrace{\prod_{i=1}^{N} CNN(c_k|I_i)^{\alpha_i}}_{\text{Accumulated predictions}} \tag{3.10}$$

| | |
|---|---|
| $c_k$ | binary variable for class $k$ |
| $S = \{s_1, ..., s_N\}$ | set of observations with semantic information |
| $CNN(c_k\|I_i)$ | network probability of pixel belonging to class $k$ |
| $\alpha_i$ | matching accuracy |

In practice, for improved numerical stability, we accumulate for each map point the negative log likelihood of each class instead of the product of probabilities.

**Depth Mean** In the update of the Gaussian mean we can see an additional third term compared to the base formulation (see section 2.3.3.2) that reflects the matching accuracy based outlier handling.

$$\mu_i = \left[\frac{C_1}{C_1 + C_2 + C_3}\right] \hat{\mu}_i + \left[\frac{C_2}{C_1 + C_2 + C_3}\right] \mu_{i-1} + \left[\frac{C_3}{C_1 + C_2 + C_3}\right] \mu_{i-1} \tag{3.11}$$

| | |
|---|---|
| $\mu_i$ | updated mean after measurement $i$ |
| $C_1, C_2, C_3$ | derived weighting terms |
| $\hat{\mu}_i$ | updated mean using the simple Gaussian update (see sec. 2.3.3.1) |
| $\mu_{i-1}$ | previous mean estimate |

Looking at the constants we can see that $C_1$ and $C_2$ are weighted by the matching accuracy $\alpha$ and $C_3$ by it's inverse probability. So for a small matching accuracy $\alpha$ the

new mean will be mostly based on the previous estimate.

$$C_1 = \alpha_i \underbrace{\frac{a_{i-1}}{a_{i-1}+b_{i-1}}}_{\text{Inlier ratio}} \underbrace{\mathcal{N}(d_i|\mu_{i-1}, \sigma_{i-1}^2 + \tau_i^2)}_{p(d|inlier)} \tag{3.12}$$

$$C_2 = \alpha_i \underbrace{\frac{b_{i-1}}{a_{i-1}+b_{i-1}}}_{\text{Outlier ratio}} \underbrace{\mathcal{U}(d_i|Z_{min}, Z_{max})}_{p(d|outlier)} \tag{3.13}$$

$$C_3 = (1-\alpha_i) \underbrace{\mathcal{U}(d_i|Z_{min}, Z_{max})}_{p(d|outlier)} \tag{3.14}$$

| | |
|---|---|
| $\alpha_i$ | matching accuracy |
| $a_{i-1}, b_{i-1}$ | previous estimate of Beta parameters |
| $p(d\|inlier)$ | probability of new measurement $d_i$ under previous model |
| $p(d\|outlier)$ | probability of new measurement under uniform model |

**Depth Variance** We can see the same third term in the variance update, based on the second moments.

$$\begin{aligned}\sigma_i^2 = &\frac{C_1}{C_1+C_2+C_3} \underbrace{(\hat{\sigma}_i^2 + \hat{\mu}_i^2)}_{\text{Second Moment } p(d|\hat{\mu}_i, \hat{\sigma}_i^2)} + \\ &\frac{C_2}{C_1+C_2+C_3} \underbrace{(\sigma_{i-1}^2 + \mu_{i-1}^2)}_{\text{Second Moment } p(d|\mu_{i-1}, \sigma_{i-1}^2)} + \\ &\frac{C_3}{C_1+C_2+C_3} \underbrace{(\sigma_{i-1}^2 + \mu_{i-1}^2)}_{\text{Second Moment } p(d|\mu_{i-1}, \sigma_{i-1}^2)} - \mu_i^2\end{aligned} \tag{3.15}$$

| | |
|---|---|
| $\sigma_i^2$ | updated variance after measurement $i$ |
| $C_1, C_2, C_3$ | derived weighting terms |
| $\hat{\sigma}_i^2$ | updated variance using the simple Gaussian update (see sec. 2.3.3.1) |
| $\hat{\mu}_i$ | updated mean using the simple Gaussian update (see sec. 2.3.3.1) |
| $\mu_{i-1}$ | previous mean estimate |
| $\sigma_{i-1}^2$ | Previous variance estimate |
| $\mu_i$ | updated mean after measurement $i$ |

**Inlier Ratio** Looking at the update of the inlier ratio we can see that the additional third term models the case of a low matching accuracy. It keeps the previous estimate of the inlier ratio, as the new measured value probably does not belong to the same point and therefore does not provide us any further information.

$$\begin{aligned}\phi = \frac{a_i}{a_i + b_i} :=& \frac{C_1}{C_1 + C_2 + C_3} \underbrace{\left( \frac{a_{i-1} + 1}{a_{i-1} + b_{i-1} + 1} \right)}_{\text{Inlier increase a}} + \\ & \frac{C_2}{C_1 + C_2 + C_3} \underbrace{\left( \frac{a_{i-1}}{a_{i-1} + b_{i-1} + 1} \right)}_{\text{Outlier increase b}} + \\ & \frac{C_3}{C_1 + C_2 + C_3} \underbrace{\left( \frac{a_{i-1}}{a_{i-1} + b_{i-1}} \right)}_{\text{No match keep values}} = f \end{aligned} \tag{3.16}$$

| | |
|---|---|
| $a_i, b_i$ | updated beta distribution parameters |
| $C_1, C_2, C_3$ | derived weighting terms |
| $a_{i-1}, b_{i-1}$ | previous beta distribution parameters |

The same holds for the second moment of the Beta distribution.

$$\begin{aligned}\frac{a_i(a_i + 1)}{(a_i + b_i)(a_i + b_i + 1)} :=& \frac{C_1}{C_1 + C_2 + C_3} \underbrace{\left( \frac{(a_{i-1} + 1)(a_{i-1} + 2)}{(a_{i-1} + b_{i-1} + 1)(a_{i-1} + b_{i-1} + 2)} \right)}_{\text{Inlier increase a}} + \\ & \frac{C_2}{C_1 + C_2 + C_3} \underbrace{\left( \frac{a_{i-1}(a_{i-1} + 1)}{(a_{i-1} + b_{i-1} + 1)(a_{i-1} + b_{i-1} + 2)} \right)}_{\text{Outlier increase b}} \\ & \frac{C_3}{C_1 + C_2 + C_3} \underbrace{\left( \frac{a_{i-1}(a_{i-1} + 1)}{(a_{i-1} + b_{i-1})(a_{i-1} + b_{i-1} + 1)} \right)}_{\text{No match keep values}} = e \end{aligned} \tag{3.17}$$

| | |
|---|---|
| $a_i, b_i$ | updated beta distribution parameters |
| $C_1, C_2, C_3$ | derived weighting terms |
| $a_{i-1}, b_{i-1}$ | previous beta distribution parameters |

From the two matched moment equations we can derive closed form updates for the inlier distribution parameters in the same way as for the base model, see the appendix A.1 for the full derivation. We use $a_{obs}$ and $b_{obs}$ to name the parameters of the Beta distribution for the geometric outlier model based on the depth observations computed by the equations above.

$$a_{obs} = \frac{e - f}{f - \frac{e}{f}} \tag{3.18}$$

$$b_{obs} = \frac{a_{obs}}{f} - a_{obs} \tag{3.19}$$

$a_{obs}, b_{obs}$ updated beta distribution parameters
$e, f$ moments of beta distribution (see eq. 3.16 and 3.16)

We now define the initial count parameters for our semantic prior. For this we specify $A_k$ and $B_k$, where $A_k, B_k > 0$, as a set of constants for each semantic class defining the likelihood of the class to be dynamic and therefore an outlier or static and therefore an inlier.

$$a_{sem} = \sum_{k=1}^{K} A_k p(c_k|S) \tag{3.20}$$

$$b_{sem} = \sum_{k=1}^{K} B_k p(c_k|S) \tag{3.21}$$

$a_{sem}, b_{sem}$ Beta distribution parameters
$A_k, B_k$ prior constants for class $k$
$p(c_k|S)$ class posterior probability

With each new image we update the map point semantic class probabilities $p(c_k|S)$ and can then compute updated prior counts. In addition to giving higher prior counts to dynamic classes theses weights also weigh the trade-off between semantic and depth measurements as higher values for all semantic classes will put more importance to the semantic prior compared to the motion prior. As can be seen in the computation of the inlier ratio these semantic priors are simply added to the original counts of the base Beta distribution.

$$\phi = \frac{a_{obs} + a_{sem}}{a_{obs} + a_{sem} + b_{obs} + b_{sem}} \tag{3.22}$$

$\phi$ inlier ratio
$a_{obs}, b_{obs}$ Beta distribution parameters of base model
$a_{sem}, b_{sem}$ Beta distribution parameters of semantic model

For a map point belonging to a dynamic class we want a low prior $a_sem$ and a high $b_sem$ to be added to the $a_{obs}$ and $b_{obs}$ in order to reduce the risk of using potentially dynamic points before they can be confirmed as currently static. We use the inlier ratio to estimate whether a point is active or not. The continuous update of the two beta distributions allows a continuous transition between dynamic and static states.

## 3.4 System Integration

In the following we will detail the modifications we made to each part of the ORB-SLAM pipeline, as highlighted in figure 3.1.

**Semantic Segmentation** For each new frame we predict the per-pixel semantic probabilities using a CNN. We use the publicly available IC-Net model [72] pre-trained on the CityScapes dataset [74] on all datasets. The original model predicts 19 classes, for our use case we map these to one of 3 classes, see table 3.1.

| Class $k$ | Label | CityScapes classes | $A_k$ | $B_k$ |
|---|---|---|---|---|
| 0 | static | road, sidewalk, building, wall, fence, pole, traffic light, traffic sign, vegetation, terrain | 5 | 0 |
| 1 | potentially dynamic | person, rider, car, truck, bus, train, motorcycle, bicycle | 0 | 5 |
| 2 | invalid | sky | 0 | $\infty$ |

**Table 3.1:** Semantic class mapping from the CityScapes classes to the static, dynamic and invalid classes used in the probabilistic model. The last two columns specify the prior counts for the semantic beta model for each class.

### 3.4.1 Initialization

We use the parallel fundamental matrix and homography based initialization strategy from ORB-SLAM, see section 2.3.2.1. We initialize using only descriptive features, as without an initial pose the optical flow based matching of the direct features often fails. We further leverage the available semantic segmentation to filter out potentially moving objects by requiring a minimum static probability. Once initialized we extract multi-resolution direct features in the first frame and use optical flow and epipolar line checks to find matches in the second frame and triangulate their initial 3d positions to initialize direct map points.

### 3.4.2 Tracking

In the front-end tracking process each new frame is processed in real-time. We use the same procedure as in ORB-SLAM to extract descriptive features. We also follow the multi-stage process of using a constant velocity model as initial pose estimate to efficiently find correspondences with the current keyframe, followed by a refinement against the local map. In addition to the descriptive features we also leverage the direct features in the current keyframe for the pose estimation. We only use map points that are considered as inliers at this moment in time by computing their inlier ratio based on our probabilistic model parameters and defining a minimum inlier ratio threshold to activate them. We follow a multi-resolution iteratively re-weighted optimization scheme as in ORB-SLAM and DSO using our joint error function for both direct and descriptive features.

We use each measurement to update our probabilistic measurement model. With each new frame we update the semantic class probabilities, the depth mean, depth variance and the inlier model parameters.

### 3.4.3 Mapping

As in ORB-SLAM, if not sufficient matches between the new frame and the current keyframe can be found, or a certain distance threshold is crossed we create a new keyframe. We only consider active points based on the current inlier estimate in the decision. For each new keyframe we extract new multi-resolution direct features. The

estimates from the previous keyframe are propagated to the new one and new descriptive and direct features are triangulated.

The pose of the new keyframe together with the poses of the co-visible keyframes and their map points are refined with local bundle adjustment using our joint direct and descriptive error formulation. We only use map points that are currently considered inliers by the probabilistic model at this moment in time. After the bundle adjustment we update our probabilistic model with the refined depth estimates.

## 3.5 Experiments

In the following we discuss our experimental setup describing the used datasets and metrics and provide the implementation details and evaluation protocol, before we present and discuss the results of our experiments.

### 3.5.1 Evaluation protocol

The use of different methods for initialization, loop-closure, re-localization and re-initialization, different optimization strategies, different runtime budgets and differences in the parameterizations (e.g. number of used key points, number of keyframes used in local bundle adjustment) makes it difficult to compare and identify the differences between SLAM systems with respect to the choice of features and dynamic outlier handling. Therefore in our evaluation we focus on a comparison of the feature and outlier model variants in the same framework built on top of ORB-SLAM, that we use as a reference.

### 3.5.2 Implementation details

Similar to Zia *et al*. [75] we added an option to run the originally multi-threaded asynchronous ORB-SLAM baseline in a deterministic frame-by-frame manner where tracking and mapping alternate as needed. The sequential mode makes the evaluation independent of the dataset frame rate, the available hardware performance and reflects the results using an ideal hardware setup. We further adjusted parts of the pipeline that have non-deterministic behaviour. The ORB-SLAM baselines often uses a greedy strategy to increase efficiency that, in combination with non-deterministic data structures, can lead to random behavior. Therefore we provide comparators for datatypes such as keyframes and map points, so that these can be used in a deterministic way in tree based lookup maps. Now we can repeat a certain run exactly the same way using a fixed initial random seed for pseudo random processes inside the pipeline. A deterministic performance allows us to better evaluate variations with different hyper parameters. However, under real conditions the use of different seeds can still lead to different results for every run. For the evaluation we also turn off the loop-closure detection to better evaluate the pose error over long distances. In table 3.2 we provide the parameter values used for the experiments. Building on top of ORB-SLAM, we focus on the most relevant parameters, the ones that differ from the original implementation and the ones we added.

| Parameter | Value | Description |
|---|---|---|
| $N_F^{Descr}$ | 5000 | number of extracted ORB features |
| $N_F^{Direct}$ | 5000 | number of extracted direct features |
| $N_P^{Direct}$ | 9 | number of pixels in direct feature pattern |
| $\tau_{min}^{Direct}$ | 0.2 | minimum gradient std for direct features in range [0, 1] |
| $N_L^{Descr}$ | 8 | number of resolution pyramid levels for ORB features |
| $s_L^{Desc}$ | 1.2 | scale factor for resolution pyramid |
| $\tau_{init}^{Descr}$ | 20 | initial FAST corner threshold for ORB features |
| $\tau_{min}^{Descr}$ | 7 | minimum FAST corner threshold for ORB features |
| $\eta_R$ | 0.5 | weight for descriptive feature re-projection loss |
| $\eta_P$ | $0.5/N_P^{Direct}$ | weight for direct feature photometric loss |
| $\delta_R$ | 2.448 | Huber threshold for descriptive feature re-projection loss |
| $\delta_P$ | $0.4\sqrt{\eta_P}$ | Huber threshold for direct feature photometric loss |
| $\chi_R$ | 5.991 | outlier threshold for descriptive feature re-projection loss |
| $\chi_P^2$ | $0.05N_P^{Direct}$ | outlier threshold for direct photometric loss |
| $N_lBA$ | 20 | number of key frames used in local bundle adjustment |
| $p_{min}^{static}$ | 0.8 | minimum probability of static class for initialization |
| $\psi_{min}$ | 0.6 | min. inlier ratio threshold for activation |

**Table 3.2:** Overview of the pipeline parameters

### 3.5.3 Metrics

For the evaluation of the poses we use the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) metrics. To evaluate the depth estimates of the estimated map points we use the DR (Depth inlier Ratio). The definition of the metrics is given in section 2.3.6. For the depth inlier ratio we set the threshold to $0.1$, counting a depth measurement as correct when the error is less than $10\%$ of the ground truth depth.

### 3.5.4 Datasets

We evaluate our method in an outdoor setting, where a vehicle drives over a larger distance through different environments, where dynamic objects occur regularly. For many of these datasets relatively good ground truth poses, based on additional on-board or external sensors, are available. However, ground truth for the depth measurements is often not available or too sparse. Therefore we additionally use synthetic data to evaluate the impact of our probabilistic model on the depth estimation.

#### Static scenes

We first evaluate the proposed method using a popular benchmark that contains mostly static scenes to allow for comparisons with other methods and ensure that there are no negative effects from the changes we made towards more robustness to dynamic objects.

**KITTI** For this we use the established KITTI odometry dataset and benchmark [76]. Here the ground truth pose was obtained using a fusion of GPS, RTK corrections and IMU measurements. While there are two stereo cameras available in the dataset, we only use the left color camera for our experiments. The images were recorded with a frame rate of 10 fps leading to a relatively long average distance of 0.5 meters between frames.

#### Dynamic scenes

Due to a lack of a dataset with longer dynamic sequences and to focus the comparison on challenging situations we collect some shorter sequences from different datasets that include highly dynamic scenarios to better evaluate the impact of our contributions. We collect scenes from three different datasets.

**CityScapes** The CityScapes dataset [74] uses a similar sensor setup as the KITTI dataset but records with a higher frame rate of 17 Hz. We select a subset from the Frankfurt sequence 0 including frames 21,850-22,349 that show a challenging situation approaching a red traffic light.

**Virtual KITTI** The Virtual KITTI dataset [77] built digital twins for some sequences of the KITTI dataset in order to generate a new synthetic dataset with accurate ground truth annotations. We evaluate our method on sequence 20 showing stop-and-go traffic on a highway entrance.

**Synthia** Synthia [78] is also a synthetic dataset that aims to provide semantic segmentation ground truth. Therefore the frame rate of the camera images of 5 Hz is relatively low. The selected scene 01 includes some faster vehicles that perform lane change and take-over maneuvers in front of or next to the ego vehicle.

### 3.5.5 Results

In the following we provide the results of our experiments. First we look at the evaluation in the static scenes and then move to the results for the dynamic scenarios.

#### 3.5.5.1 Static Scenes

We use the static scenes to compare the impact of the proposed probabilistic model with the baseline and a naive masking approach that masks out all potential dynamic areas.

**Probabilistic vs heuristic map point model** At first we want to compare our proposed probabilistic map point model to the heuristic model in ORB-SLAM. For a better comparison we decided to only use descriptive features and use a slightly modified version of ORB-SLAM that uses the same inverse depth parameterized map point representation as our approach. Table 3.3 shows the results of the individual sequences

of the KITTI odometry dataset as well as the accumulated results over all sequences. We can see that the probabilistic model does not lead to any negative effects on static scenes, in some sequences it performs a bit better in others a bit worse than the heuristic baseline. Overall, the performance is a bit better than the baseline.

| Seq. | Length (m) | Configuration | ATE (m) | RPE (m) |
|---|---|---|---|---|
| 00 | 713.17m | **Ours** | **37.63** | **49.38** |
| | | ORB-SLAM* | 45.28 | 58.18 |
| | | Masked | 57.29 | 80.78 |
| 01 | 512.70m | Ours | 30.05 | 39.98 |
| | | **ORB-SLAM*** | **20.01** | **23.19** |
| | | Masked | 43.75 | 60.57 |
| 02 | 5,065.70m | Ours | 105.30 | 155.80 |
| | | **ORB-SLAM*** | **101.06** | **151.32** |
| | | Masked | 99.29 | 151.52 |
| 03 | 629.09m | Ours | 33.00 | 39.85 |
| | | **ORB-SLAM*** | **26.59** | **30.89** |
| | | Masked | 28.29 | 35.98 |
| 04 | 552.43m | Ours | 14.79 | 15.95 |
| | | ORB-SLAM* | 14.08 | 14.59 |
| | | **Masked** | **7.51** | **9.12** |
| 05 | 2,205.01m | Ours | 31.88 | 41.12 |
| | | ORB-SLAM* | 43.70 | 53.29 |
| | | **Masked** | 20.49 | 28.67 |
| 06 | 2,119.37m | Ours | 21.06 | 24.24 |
| | | **ORB-SLAM*** | **17.16** | **20.79** |
| | | Masked | 17.58 | 21.52 |
| 07 | 1,386.62m | **Ours** | **9.81** | **13.59** |
| | | ORB-SLAM* | 12.17 | 17.69 |
| | | Masked | 13.32 | 16.37 |
| 08 | 3,221.97m | **Ours** | **50.92** | **76.06** |
| | | ORB-SLAM* | 87.06 | 114.77 |
| | | Masked | 126.31 | 164.63 |
| 09 | 1,703.55m | **Ours** | **49.85** | **67.14** |
| | | ORB-SLAM* | 90.50 | 125.12 |
| | | Masked | 113.18 | 157.87 |
| 10 | 919.08m | Ours | 26.85 | **33.19** |
| | | ORB-SLAM* | 32.14 | 40.57 |
| | | Masked | **26.30** | 33.69 |
| sum | 19,028.69m | **Ours** | **411.14** | **556.30** |
| | | ORB-SLAM* | 489.75 | 650.40 |
| | | Masked | 559.15 | 767.59 |

**Table 3.3:** Evaluation results of the 11 KITTI odometry dataset sequences, comparing the proposed probabilistic map point model (Ours) with the heuristic ORB-SLAM model (ORB-SLAM*) and a simple dynamic object masking approach (Masked). Taken from [5]

The relatively low frame rate in the KITTI dataset and therefore larger baseline between frames limits the number of observations of each map point that are available to improve the depth and inlier ratio estimate. In some sequences this can lead to the case

where not sufficient depth measurements are available to update the inlier ratio of a map point with a dynamic prior to activate it, so that it can be used in the tracking. However, overall there does not appear to be a significant difference between the proposed probabilistic and the heuristic baseline models on static scenes. This can also be seen in 3.2 that shows the estimated trajectories for sequence 07 from the KITTI odometry dataset, showing similar differences for both approaches.

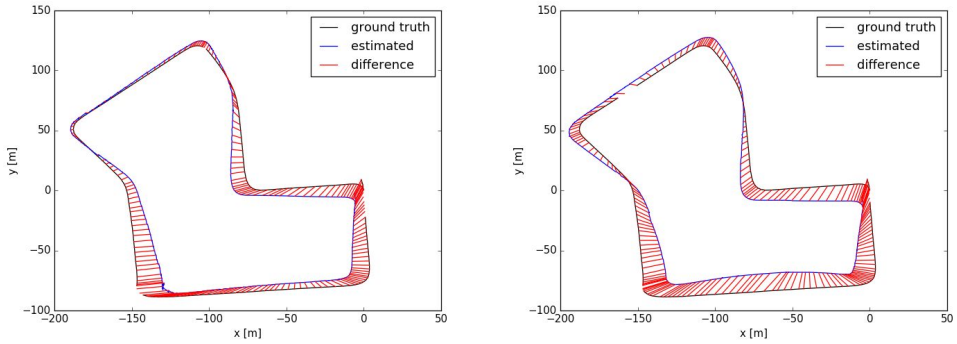



**Figure 3.2:** Absolute trajectory error of `Ours` (left) and `ORB-SLAM*` (right) on the KITTI 07 sequence. Taken from [5] (Copyright ©2018, IEEE)

To get a better understanding of the inner workings of the probabilistic model we look at two examples. Figure 3.3 shows some of the features on the parked van and car change from dynamic red to static green status when more observations get integrated. The points on the parked cars stay inactive longer than the points on the static areas in the image as the semantic prior requires more consistent depth measurements to activate them compared to the points with a static prior.

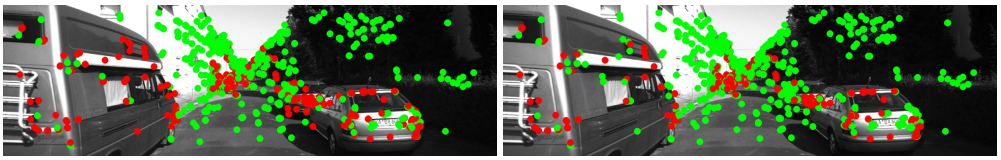


**Figure 3.3:** Example for map point status updates in frame 342 at initialization (left) and after the integration of additional observations until frame 345 (right) from sequence 00 of the KITTI dataset.

For a better insight into the probabilistic update process figure 3.4 shows the evolution of the inlier ratio distribution over multiple measurements of a point lying on a parked vehicle. We initialize each map point with the Beta distribution parameters $a = b$ assuming an initial inlier ratio of $0.5$. For points with a predicted high dynamic class probability the semantic prior provides a $b_{sem}$ that is larger than $a_{sem}$ resulting in an initially low inlier ratio. It then requires multiple observations to update the $a_{obs}$ and $b_{obs}$ to compensate for the prior and increase the inlier ratio. We can see that the mean inlier ratio increases after each successful triangulation and depth measurement eventually

reaching the required min inlier ratio needed to activate it so that it can be used for tracking.

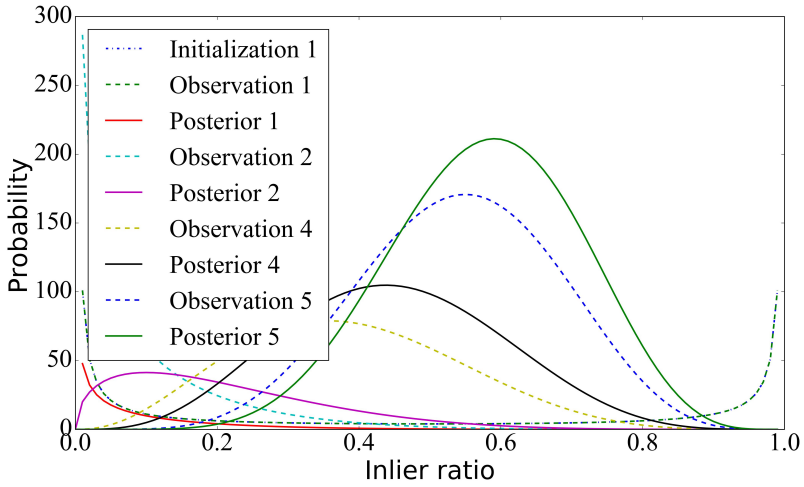


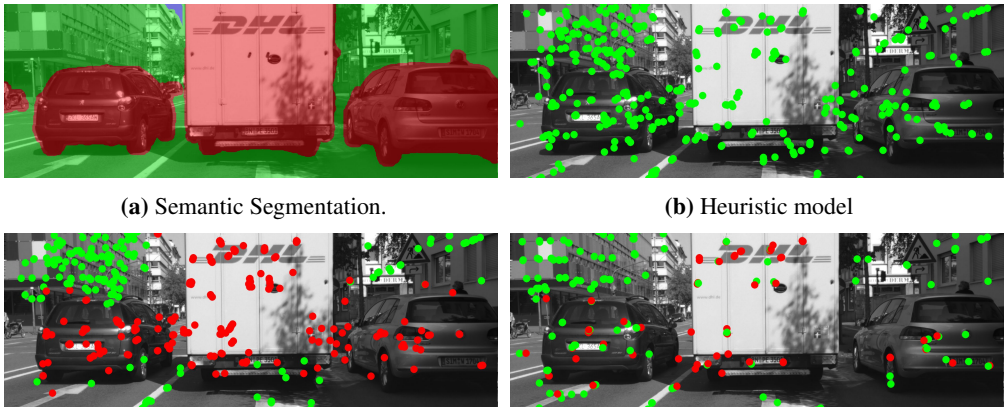

**Figure 3.4:** Example probability distributions of observations and posteriors of the inlier ratio of a map point on a parked car in the KITTI 00 sequence showing the convergence of the inlier ratio from an outlier to an inlier. Taken from [5] (Copyright ©2018, IEEE)

In the case of a static semantic measurement the prior assigned to the static class might lead to a larger $a_{sem}$ than $b_{sem}$ providing a larger inlier ratio from the start, leading to a quicker activation. In general at least 3 observations of a point are needed to evaluate the uncertainty of the triangulation, however in practice it can often take more measurements to get a more reliable estimate. With the semantic priors it is possible to reduce the number of required successful triangulations for points belonging to a static class and increase the number for potentially dynamic points.

**Probabilistic model vs masking** To motivate the advantages of a joint probabilistic model for depth and inlier ratio we compare our approach against a simple masking approach that deactivates points based on their dynamic class probability estimated from the semantic predictions. In this approach a point gets deactivated if the dynamic class probability is larger than a threshold of $0.5$. In 3.3 we also list the results for the simple masking approach on the KITTI odometry dataset. The results show that while the approach performs best in a few sequences the overall errors are larger than the ORB-SLAM baseline without using any semantics at all, suggesting a negative effect in static scenes.

To better understand the difference between the masking approach and a probabilistic model we can look at figure 3.5 that shows an example of a difficult scene. The ego vehicle is approaching a red traffic light, while a preceding car and truck have stopped in front of it. A considerable part of the image belongs to different potentially dynamic

objects, while some of them belong to actually dynamic cars, others belong to parked cars along the road. The status of each map point is visualized by its color, where green represents active points and red corresponds to inactive points. The direct comparison shows that the naive heuristic approach is not able to detect any of the points as outliers. The simple masking approach also removes features from the static parked cars leading to significantly fewer and unevenly distributed active points. The proposed probabilistic map point model shows several but fewer tracked points on the cars in general with some of them inactive.



**(a)** Semantic Segmentation. **(b)** Heuristic model

**(c)** Simple masking **(d)** Our probabilistic model

**Figure 3.5:** Example scene from the CityScapes dataset [74] Franfurt sequence 0 frame 19059 showing the comparison of the map point status (green - active, red - inactive) for different approaches. Taken from [5] (Copyright ©2018, IEEE)

Many of the KITTI sequences are located in urban environments with parked vehicles along the road. As in many scenes there exist sufficient static context in the form of buildings on both sides of the road the parked cars seem like features that are more difficult to track as they mostly consist of reflective paint and glass that change their appearance under different viewing angles. In some scene however, where not sufficient static structures are available, parked cars can provide the few additional points needed to prevent tracking loss. The probabilistic model is also able to correct erroneous semantic measurements by accumulating sufficient geometric evidence to active the map points, while this would not be possible in the simple masking approach. The differences to the masking based approach also show that usually a number of additional potentially dynamic points get activated to help the pose estimation even in the case of relatively low frame rate updates as in KITTI.

#### 3.5.5.2 Dynamic Scenes

**Traffic jam** We start with sequence 20 from the Virtual KITTI dataset that represents a slowly moving stop-and-go traffic jam on a highway, as can be seen in figure 3.6.

**Figure 3.6:** Traffic jam scene from the Virtual KITTI [77] dataset. Vehicles are moving with at a low speed

Due to a larger domain gap between the real training data used to train the semantic segmentation network and the synthetic images from the Virtual KITTI dataset we are using the ground truth semantic maps provided by the dataset instead.

As in the static case we compare our proposed approach against the ORB-SLAM* baseline. The first two rows in table 3.4 shows the trajectory errors for both methods using only ORB features. We can see that the probabilistic outlier model significantly reduces the accumulated drift.

**Table 3.4:** Evaluation results for sequence 20 of the Virtual KITTI dataset. The total length of the sequence is 989.96m. Taken from [5]

| Configuration | Approach | ATE [m] | RPE [m] | DR [%] |
|---|---|---|---|---|
| ORB | **Ours** | **49.34** | **60.98** | **38.32** |
| | ORB-SLAM* | 69.80 | 87.52 | 23.58 |
| ORB + Direct | **Ours** | **27.78** | **34.44** | **41.71** |
| | ORB-SLAM* | 66.05 | 78.84 | 18.60 |

Figure 3.7 shows the comparison of the two estimated trajectories, it can be seen that the differences between the predicted and ground truth trajectories are smaller for our proposed method.

In this scene the road is surrounded on both sides by large homogeneous vegetation that makes it difficult to find good static features to track. The lane markings and guardrails also do not provide unique features and often get occluded by the other vehicles on the road. This leads to the case where the dynamic vehicles provide some of the best matched features resulting in errors in the pose estimation.

In the last two rows we show the results using both descriptive ORB and direct features. While the baseline results improve slightly, the impact of using additional features is larger for our proposed method. In this case, due to the lack of unique descriptive features the direct features can help the pose estimation, if there is a good initial pose estimate. It seems that the baseline that only uses geometric cues for the outlier filtering is not able to separated the descriptive and direct features on the dynamic cars from the ones on the static background. The proposed semantic aware method can preferably

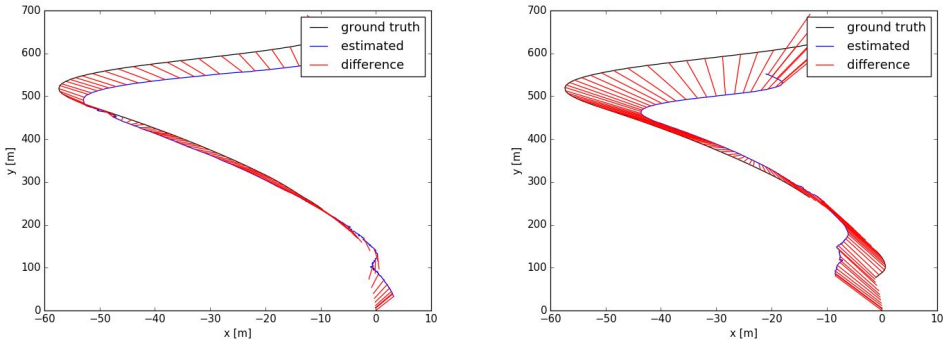
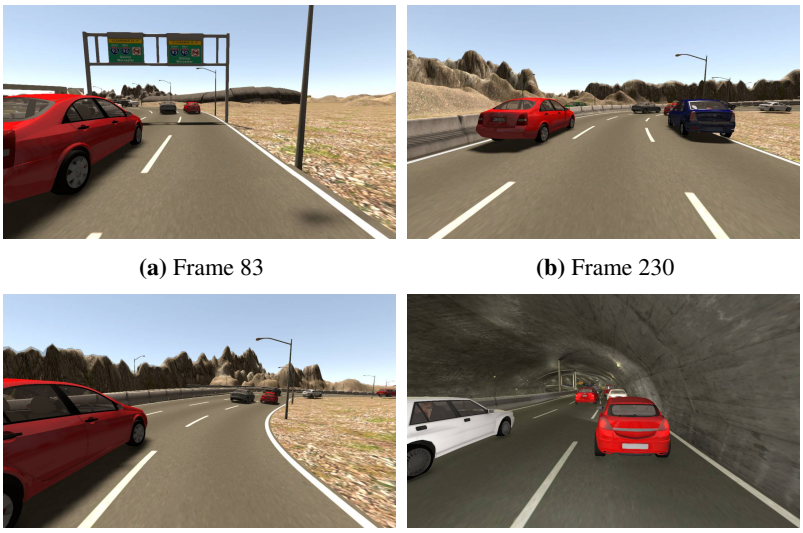



**Figure 3.7:** Absolute trajectory error of `Ours` (left) and `ORB-SLAM*` (right) on the Virtual KITTI traffic jam sequence 20. Taken from [5] (Copyright ©2018, IEEE)

select stable features on static scene elements, ignoring the features on the dynamic cars making the pose estimation more robust.

**Motorway driving** Next we look at sequence 01 of the Synthia dataset [78] showing the ego vehicle driving on a motorway together with other dynamic vehicles. Images from the sequence can be seen in figure 3.8. For the Synthia dataset we also use the ground truth semantic maps provided by the dataset due to a large domain gap to the synthetic data.



**(a)** Frame 83 **(b)** Frame 230

**(c)** Frame 260 **(d)** Frame 435

**Figure 3.8:** Examples from the Synthia dataset [78] scene 01 sequence.

In table 3.5 we show the quantitative results comparing the proposed method with the ORB-SLAM* baseline. It can be seen that the difference is not that large. While

overtaking and preceding vehicles can temporarily cover large parts of the image, in this scene sufficient features can be found on the static environment to enable stable pose estimation.

**Table 3.5:** Trajectory evaluation on the sequence Synthia 01 (411.68 m). Taken from [5]

| Configuration | ATE (m) | RPE (m) | DR (%) |
|---|---|---|---|
| **Ours** | **2.99** | **6.19** | **73.95** |
| ORB-SLAM* | 4.43 | 7.11 | 70.60 |

**Red traffic light** In figure 3.9 on the right we can see a scene where the ego car approaches a red traffic light with several cars waiting in front of it. Once the traffic light turns green the cars start moving again. The estimated trajectories from our proposed method and the ORB-SLAM* baseline are shown in the same figure on the left side. One can see that the baseline trajectory shows larger distances to the ground truth than our method. In this case the ground truth trajectory is measured via GPS and therefore very coarse and might not reflect the actual driving path. Looking at the trajectory of the baseline it can be seen that after starting at the bottom right the car first goes straight, then it seems to be moving backwards before it starts moving forward again and turns left. This can be explained by the use of features on the other temporarily stopped vehicles for pose estimation. When these cars start moving forward again the feature motion makes the ego vehicle appear to move backwards by assuming a static world. With the help of the semantic priors the proposed method seems to trust the points on the vehicles less and therefore is able to estimate a better trajectory using the static features. Here it seems that there have not been sufficient consistent observations to active the features on the stopped cars. However under a different setting this might still happen with semantic priors leading to the same problem that the baseline faces.

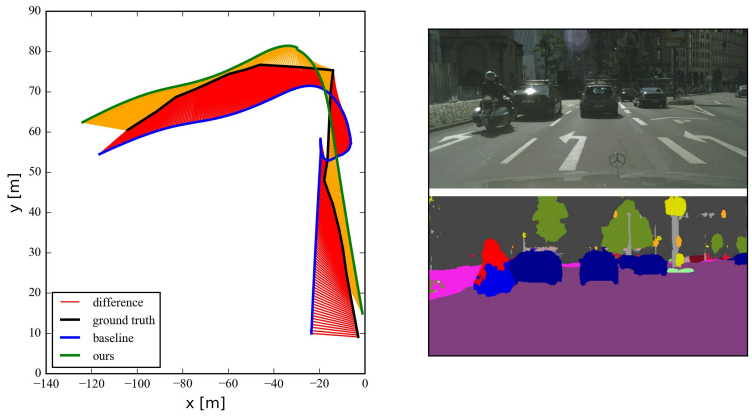



**Figure 3.9:** Example from the Frankfurt sequence of the CityScapes dataset [74]. Taken from [5] (Copyright ©2018, IEEE)

# 4 Geometric Priors for Structured Scenes

Existing monocular SLAM systems often struggle in human-made indoor environments due to low-textured scenes (e.g. containing large areas of texture-less floor, wall and ceilings) and frequent sharp turns in the camera trajectory. Without loop-closures monocular systems suffer not only from pose drift in rotation and translation but also from scale drift. To mitigate the accumulation of errors Manhattan World based SLAM methods have been proposed. However, most of these assume that the global MW frame is visible in all frames, this assumption fails especially in cluttered environments and if the camera moves too close to surface loosing the global context. While MW approaches based on RGB-D sensors have direct access to the scene structure via the measured depth, monocular approaches are limited to estimating points, lines and planes via triangulation, vanishing point estimation and homography estimation that is error prone and requires scenes with sufficient texture.

Therefore, we propose to leverage a neural network that estimates per-pixel depth and normals from the RGB image. The network is able to predict the normals of planes in the image from the global and local image context and therefore can also predict geometry in areas without any informative textures. We integrate line and plane features together with an Manhattan World based rotation estimation step into the ORB-SLAM framework [15]. This allows us to find a global Manhattan pose if it is visible, but also to fall back to a regular point and line feature based SLAM pipeline in case the found solution is not plausible.

## 4.1 System Overview

We build our system around the open-source point-based ORB-SLAM framework [15] that we extend with line and predicted normal features.

For each new RGB frame we extract point and line features and use a neural network to predict dense surface normals. We use the predicted normals to estimate the rotation against the world Manhattan frame. If successful, we use the rotation to estimate the translation from the point and line correspondences. In the case that the solution is not supported by sufficient inlier point and line features, we fall back to a regular feature based full pose estimation using points and lines. Either way, the initial solution is then refined using the local map and bundle adjustment.

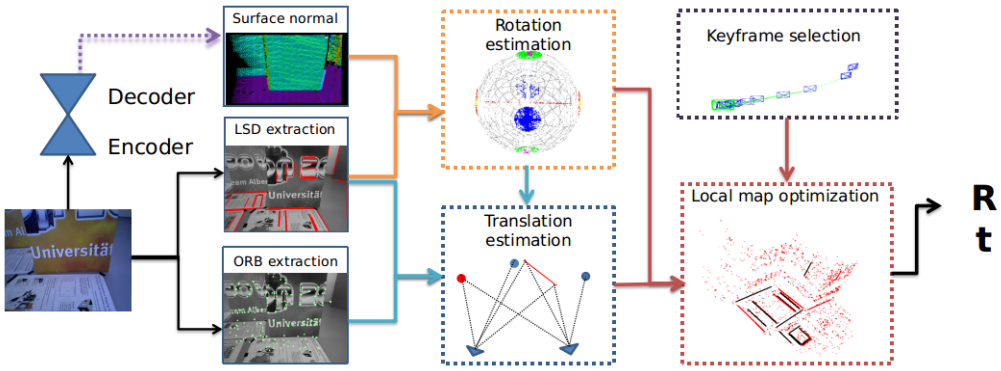



**Figure 4.1:** Overview of the proposed pipeline. Taken from [6] (Copyright © 2020, IEEE)

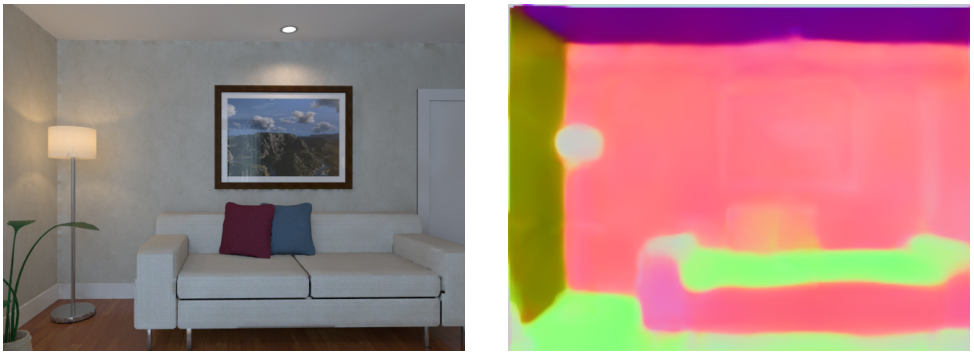
## 4.2 Normal Estimation & Segmentation

There exist many networks to predict depth from RGB images for a wide range of applications. For the Manhattan based rotation estimation we are only interested in the surface normals. Given a depth map we would need to compute the depth gradients to estimate the normals. However, most of the predicted depth maps show high frequency noise in their predictions that gets amplified by the gradient operation in the normal estimation. Therefore, we decided to use a network that was trained to directly predict smooth normal maps. We use the model from Yu *et al.* [48] that is based on a ResNet101-FPN encoder architecture and has a three-branch decoder, one for planar mask segmentation, one for plane parameter estimation and one for instance embedding prediction. The planar segmentation branch predicts an image mask that contains the probability that a pixel lies on a planar region in the image. The plane segmentation branch predicts the plane normal and depth parameters $[n, d]$ per pixel. And the instance segmentation branch predicts an instance embedding vector per pixel that can be used to cluster the planes into instances.

While the network was not originally designed for it, we only use the predicted normals $n$, as the predicted depth values have a relatively high relative error around $\frac{\delta d}{d} = 13.4\%$. We only use the normals inside the predicted planar pixel mask to focus on the planar structures for the rotation estimation. An example is shown in 4.2 and more examples of predicted normal maps can be found in figure 4.7.

## 4.3 Point & Line Features

We use both point and lines features to increase the robustness in low-textured environments. We show an example of the extracted point and line features in figure 4.3.



**(a)** RGB Input **(b)** Normal prediction

**Figure 4.2:** Example of the normal map prediction from a single RGB image from the ICL-NUIM dataset [79]. Taken from [6] (Copyright © 2020, IEEE)

**Points** As points we use ORB features using the three dimensional euclidean coordinate representation as in ORB-SLAM [15]. We follow the same pipeline for the ORB feature extraction, matching, triangulation and error functions, see also section 2.3.2.1.

**Lines** We parameterize the 3D lines based on their 3D start and end points. We use the LSD [32] extractor to detect lines in the image and compute binary LBD [33] based descriptors for each line as also done in PL-SLAM [34], see also section 2.3.2.2.


**(a)** Structured scene **(b)** 2D features

**Figure 4.3:** Example of point and line feature detections. Taken from [6] (Copyright © 2020, IEEE)

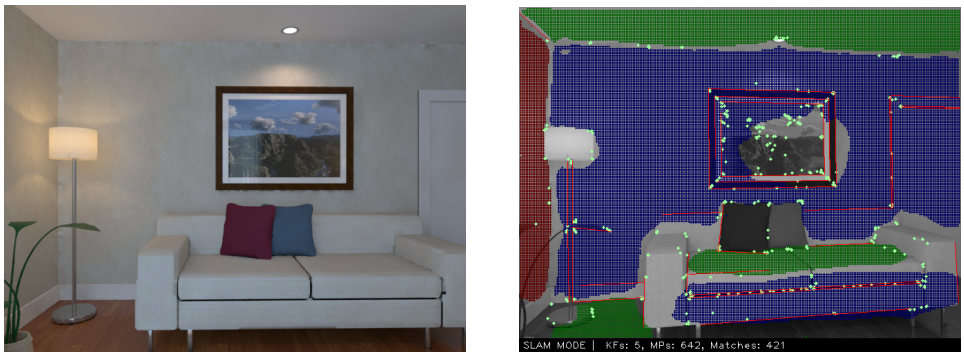
## 4.4 System Integration

### 4.4.1 Initialization

For the initialization we perform independent estimation of the rotation and translation. We use a MW based rotation estimation that also works in pure rotation and small par-

allax scenarios where regular approaches struggle. With the rotation estimated fewer point features are needed to estimate the translation, enabling initialization in scenes with limited texture.

**Rotation** We follow the mean shift based approach from Zhou *et al.* [42, 3] (see section 2.4.4) to estimate the dominant Manhattan frame from the predicted normal map. In the case that only two orthogonal planes can be found we compute the third one from the cross product of the other two. We then use the MW orientation found in the first image as our global MW frame. For the second frame and all sequential frames we use the same approach to find the MW frame, this time starting from the previous estimate of the first frame to ensure consistency, see also figure 4.4 for an illustration.




**(a)** Structured scene **(b)** plane segmentation from MW

**Figure 4.4:** Example Manhattan world frame detection. Taken from [6] (Copyright © 2020, IEEE)

**Translation** Based on the constraint derived by Kim *et al.* [3] given in equation 2.34 we build a linear system where each point pair provides an additional row.

$$[X_{k-1}(R_3 y_k - R_2), X_{k-1}(R_1 - R_3 x_k), X_{k-1}(R_2 x_k - R_1 y_k)] \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix} = 0 \quad (4.1)$$

We are only using the ORB point features and not the lines during the initialization, this allows a closed form solution of the linear system using SVD.

**Verification & Refinement** Once the relative pose between the first two frames is found, we compute the number of inliers based on the available point features in order to assess the validity of the found solution. If the number of inliers is lower than a certain threshold we discard the second frame and keep searching for a better estimate. If too many frames are skipped we reset the first frame and start over again, as also done in ORB-SLAM.

The advantages of the decoupled Manhattan rotation estimation and 3 DoF translation estimation are that we require fewer feature correspondences to estimate the translation and that we can find a solution even under pure rotational motion.

If the estimate has sufficient inliers we convert the first two frames into keyframes, triangulate initial point and line estimates and run bundle adjustment using both point and line features to refine the initial pose and map.

### 4.4.2 Tracking

In the regular tracking process we follow a similar decoupled pose estimation approach as during the initialization with some differences. Here we combine point and line features from the beginning to increase robustness in low-textured environments and leverage the previously estimated map points. We also allow a fallback to a frame-to-frame and frame-to-map based pose estimation approach in the case of a failed MW based rotation estimation.

**Rotation** For the rotation estimation against the MW frame we follow the same steps as for the second frame in the initialization to find the orientation of the MW frame in the predicted normal map, see figure 4.5.

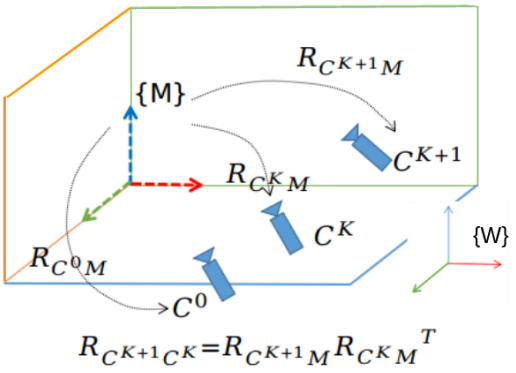



**Figure 4.5:** Rotation estimation between multiple frames via the Manhattan world. Taken from [6] (Copyright © 2020, IEEE)

**Translation** For the translation estimation we again assume the rotation has already been estimated and is fixed as in the initialization. However, as we now have a local map available we use both point and line feature optimizing their alignment to the current 2d observations via re-projection errors. For the points we use the standard re-projection error between 3d point and 2d observation given by equation 2.30 in section 2.3.2.1. For the lines we use the point-line re-projection error between the two 3d line end points and the 2d line observation given by equation 2.59 in section section 2.3.2.2, as in PL-SLAM [34].

Combing the point and line error terms we get the following total error term.

$$E = \sum_{M} E_P \Sigma_{P,i}^{-1} E_P + \sum_{N} E_L \Sigma_{L,i}^{-1} E_L \tag{4.2}$$

| | |
|---|---|
| $E_P$ | Point re-projection error |
| $\Sigma_{R,i}$ | Associated point variance |
| $E_L$ | Line re-projection error |
| $\Sigma_{L,i}$ | Associated line variance |

We use the Levenberg-Marquardt method to optimize the non-linear least squares BA problem only optimizing the translation of the new frame. As the rotation has already been estimated in theory we only require at least two 2d-3d correspondences to estimate the translation, assuming uncorrelated points.

**Pose Validation** In cases were a Manhattan frame does not exist or is currently not visible the rotation estimation and following translation estimation will fail to estimate the correct pose. Therefore we apply a validation step to check the quality of the estimate. Following ORB-SLAM we apply a threshold to the re-projection error of all points from the last $n$ keyframes onto the newly estimated frame and count the number of inliers that have a lower error than a certain threshold. In the case of a bad estimate we fallback to a unconstrained frame-to-frame and frame-to-map based tracking method as used in ORB-SLAM. However, we are using both points and lines as in PL-SLAM [34]. See section 2.4.1 for more details on these methods.

**Pose Refinement** Based on our experiments it is beneficial to run bundle adjustment on the local map of the last $n$ keyframes, as done in the mapping of ORB-SLAM, even though it does not consider the MW assumption. It seems that using the MW based initial pose estimate helps to find a better initial solution, by keeping the rotations aligned. As the refinement is not limited to MW conform solutions, the method can also be used in environments without a regular structure.

### 4.4.3 Mapping

The mapping process is the same as in ORB-SLAM for the points and PL-SLAM [34] for the lines, see section 2.4.1. We choose new keyframes based on the total number of available point and line features. Once a new keyframe is created we add new point and line features by finding correspondences with the co-visible keyframes and triangulate them. For each new keyframe we perform local Bundle Adjustment together with all the keyframes that are connected via the co-visibility graph to optimize their poses and the associated map points and lines to refine the map.

## 4.5 Experiments

In the following we detail our experimental setup, describe the used datasets and metrics and discuss our experimental results.

### 4.5.1 Implementation details

In table 4.1 we provide the most relevant parameters of our method used during the experiments, the remaining parameters are used as specified in the original ORB-SLAM [20] framework.

| Parameter | Value | Description |
|---|---|---|
| $N_F^{Point}$ | 1000 | Number of extracted ORB features |
| $N_L^{Point}$ | 8 | Number of resolution pyramid levels for ORB features |
| $\tau_{init}^{Point}$ | 20 | Initial FAST corner threshold for ORB features |
| $\tau_{min}^{Point}$ | 7 | Minimum FAST corner threshold for ORB features |
| $s_L^{Desc}$ | 1.2 | Scale factor for resolution pyramid |
| $N_F^{Line}$ | 40 | Number of extracted LSD features |
| $N_L^{Line}$ | 1 | Number of resolution pyramid levels for LSD features |
| $c$ | 2 | Width of the mean shift kernel |
| $\rho_{th}$ | 10° | Angle threshold for conic section |

**Table 4.1:** Overview of the pipeline parameters

For all experiments in this section we used a workstation with an Intel Core i7-8700 CPU (3.20 GHz), 64 GB of memory and a NVIDIA 2080 Ti GPU. As we are running the method under real-time settings and compute the median over 5 runs for each experiment.

### 4.5.2 Datasets

For our experiments we use three different indoor datasets each containing their own challenging scenes.

**ICL-NUIM** The ICL-NUIM dataset [79] is a synthetic indoor dataset that includes two scenes. The living room (lr) scene is a small rectangular room with a door, a TV and some other furniture (see first row in figure 4.7). The office (of) scene is also a rectangular room with some desks and other furniture (see second row in figure 4.7). For each scene the RGB video, depth maps and corresponding ground truth poses from 4 different trajectories are provided at a frame rate of 30 Hz and a resolution of $640 \times 480$ pixels.

**TUM RGB-D** The TUM RGB-D dataset [80] is a real dataset, captured with a RGB-D sensor at 30 Hz and a resolution of $640 \times 480$ pixels, that includes several different scenes. Some regular scenes like a desk setup, but also others that include e.g. humans or

**(a)** Frame 500 **(b)** Frame 2000

**(c)** Frame 5500 **(d)** Frame 7000

**Figure 4.6:** Examples from the HRBB4 dataset [81] sequence

artificial scenes with challenging conditions. Here we focus our evaluation on the scenes that include an artificial structure with and without texture. Specifically we look at the settings structure-texture-near (s-t-near), structure-texture-far (s-t-far) and structure-notexture-near (s-not-near), structure-notexture-far (s-not-far). Each setting provides a trajectory around an artificial structure consisting of some walls standing on the floor. In the texture setting textured sheets were applied to the structure (see third row in figure 4.7), whereas in the notexture setting a homogeneous white cover is applied to the structure (see fourth row in figure 4.7). The provided ground truth camera pose was obtained with an external tracking system.

**HRBB4** The HRBB4 dataset [81] includes one 70m long circular trajectory along the corridor of an office building. The dataset includes 12,000 RGB frames with a resolution of $640 \times 360$ pixels. The ground truth pose is obtained using artificial landmarks with a specified error of 1cm. Figure 4.6 shows some example images throughout the trajectory.

### 4.5.3 Metrics

In our experiments we evaluate the pose estimation performance of the proposed method as well as the performance of the normal estimation from RGB images.

**Normal Estimation** To evaluate the quality of the normal estimation results we use the mean, median and RMSE errors. We use the cosine similarity to compare the esti-

mated normal vectors and the ground truth normals. In addition we use three different thresholds to evaluate the accuracy of the normal estimation.

The per normal error is defined by the cosine similarity. In addition we convert from radians to degrees.

$$\epsilon_i = \frac{180}{\pi} \arccos\left(n_{est,i} n_{gt,i}\right) \tag{4.3}$$

$\epsilon_i$ Error term $i$
$n_{est,i}$ Estimated normal $i$
$n_{gt,i}$ Groundtruth normal $i$

Based on the per pixel error we compute the mean, median and RMSE.

$$mean = \frac{1}{N} \sum_{i=0}^{N} \epsilon_i \tag{4.4}$$

$$median = Median(\{\epsilon_0, ..., \epsilon_{N-1}\}) \tag{4.5}$$

$$rmse = \sqrt{\frac{1}{N} \sum_{i=0}^{N} \epsilon_i^2} \tag{4.6}$$

$\epsilon_i$ Error term $i$

The accuracy is defined by the ratio of normals with an error below a certain threshold to the total number of normals.

$$accuracy = \frac{1}{N} \sum_{i=0}^{N} |\epsilon_i| \text{ where } |x| = \begin{cases} 1, & \text{if } x < \tau \\ 0, & \text{otherwise} \end{cases} \tag{4.7}$$

$\epsilon_i$ Error term $i$
$\tau$ Threshold

**Pose Estimation** We use the Absolute Trajectory Error (ATE) and Relative Pose Error (RPE) metrics defined in section 2.3.6 to evaluate the estimated trajectories [80].

In addition we use the rotation error defined as

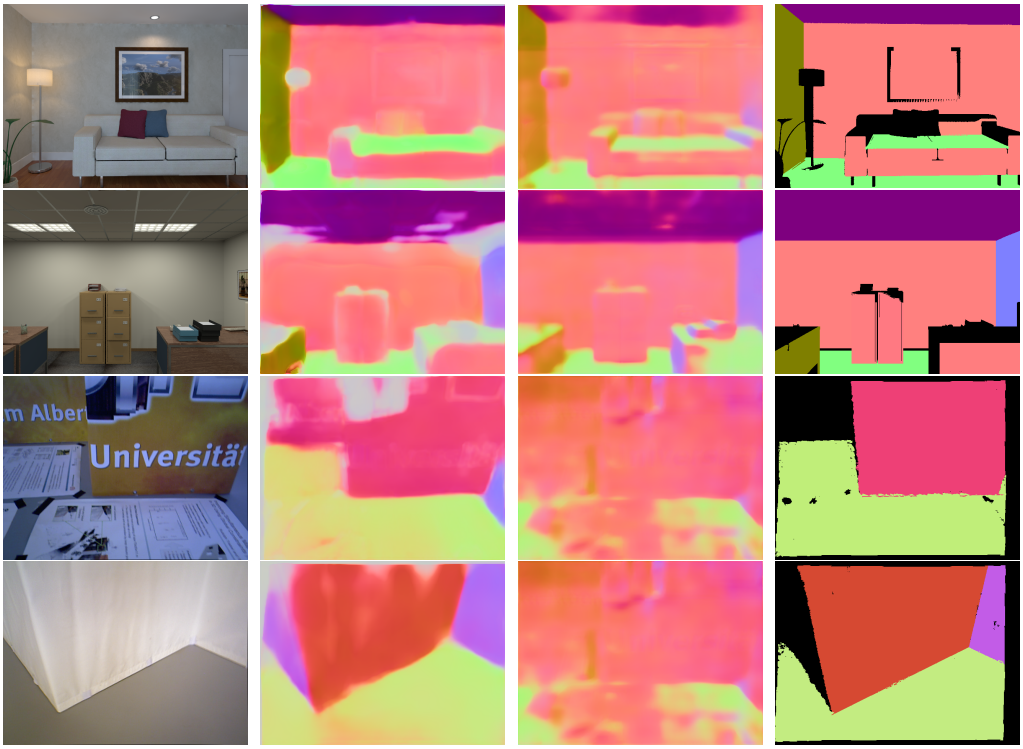
$$RE = \frac{1}{N} \sum_{i=0}^{N} \left[ \frac{180}{\pi} \arccos\left(R_{z,est,i} R^z_{z,gt,i}\right) \right] \tag{4.8}$$

$R_{z,est,i}$ Z-axis of estimated rotation matrix $i$
$R_{z,gt,i}$ Z-axis of groundtruth rotation matrix $i$

To account for the ambiguous scale in the monocular results we use a similarity transformation to align the monocular trajectories with the metric ground truth trajectory as in ORB-SLAM [15].

### 4.5.4 Normal Prediction

Our proposed method relies on accurate normal predictions for the MW based rotation estimation. Therefore we first evaluate the normal estimation performance of the network. In figure 4.7 we compare the results from the plane segmentation network [48] and another approach GeoNet [82] against the result obtained via Agglomerative Hierarchical Clustering (AHC) [83] from the depth maps provided by the datasets that are used here as ground truth. We use the publicly available pre-trained networks for both methods. The plane segmentation network [48] was trained on the ScanNet [84] dataset, while GeoNet [82] was trained on the NYU-Depth V2 dataset [85]. We show a qualitative comparison in figure 4.7 on images from the ICL-NUIM living room and office scenes as well as the structure-texture and structure-notexture sequences from the TUM RGB-D dataset. While the results from GeoNet look decent on the ICL-NUIM images, the ones from the artificial TUM sequences show degraded performance. The outputs from the plane segmentation method look much better, comparing the color coded normal maps one can see that the normal direction is similar to the ground truth one provided by AHC.



**(a)** Input **(b)** Segmentation [48] **(c)** GeoNet [82] **(d)** AHC [83]

**Figure 4.7:** Comparison of predicted normal maps from different methods on example images from the ICL-NUIM and TUM RGB-D datasets. Taken from [6] (Copyright © 2020, IEEE)

The bad performance of GeoNet might be due to the different training dataset. The NYU-Depth V2 includes less images than the ScanNet dataset. It is also possible that the plane segmentation network better generalizes to unseen settings. Based on these results and the high computational costs of GeoNet that prevent real-time performance we decided to use the plane segmentation network for our method.

In order to get a better understanding of the expected error in the normal estimation we perform a quantitative evaluation of the chosen network on the ScanNet test dataset [84]. The results are provided in table 4.2 .

**Table 4.2:** Evaluation of the surface normal prediction on the ScanNet [84] test dataset. Taken from [6]

| **Error** | | | **Accuracy** | | |
|---|---|---|---|---|---|
| mean | median | rmse | $<11.25^\circ$ | $<22.5^\circ$ | $<30^\circ$ |
| $15.2^\circ$ | $7.8^\circ$ | $24.4^\circ$ | 0.672 | 0.797 | 0.841 |

While the mean error of $15$ degrees evaluated over the whole image seems large, we expect an error closer to the median of around $8$ degrees, as with the use of the planar segmentation mask and the filtering by the conic regions around the current MW axes will remove some of the outliers.

### 4.5.5 Pose Estimation

**Baseline Comparison** We compare the proposed method against three other monocular methods. As we base our method on ORB-SLAM [15] we evaluate against the original implementation. In addition we evaluate against the dense direct method LSD-SLAM [38] and CNN-SLAM [50] that uses predicted depth maps as priors in a direct SLAM framework based on LSD-SLAM [38].

Tables 4.3 and 4.4 show the ATE for the individual sequences of the ICL-NUIM and TUM RGB-D datasets. In the sequences marked with a cross ($\times$) the method lost tracking and did not provide a trajectory that can be successfully registered to the ground truth trajectory.

**Table 4.3:** Quantitative results of the ATE metric in meters for the sequences from the ICL-NUIM [79]. Taken from [6]

| Methods | Input | lr-kt1 | lr-kt2 | lr-kt3 | of-kt1 | of-kt2 | of-kt3 | mean |
|---|---|---|---|---|---|---|---|---|
| LSD-SLAM [38] | RGB | 0.059 | 0.323 | - | **0.157** | 0.213 | - | 0.188 |
| CNN-SLAM [50] | RGB | 0.540 | 0.211 | - | 0.790 | 0.172 | - | 0.428 |
| ORB-SLAM [15] | RGB | 0.024 | 0.061 | 0.035 | $\times$ | **0.031** | 0.326 | 0.119 |
| -$w$ Segmentation | RGB | **0.016** | **0.045** | 0.046 | $\times$ | **0.031** | 0.065 | 0.041 |
| -$w$ GeoNet [82] | RGB | $\times$ | 0.047 | **0.026** | $\times$ | 0.048 | **0.043** | **0.040** |
| -$w$ AHC [83] | RGB-D | *0.016* | *0.028* | *0.020* | *0.763* | *0.021* | *0.020* | 0.145 |
| LPVO [3] | RGB-D | *0.04* | *0.03* | *0.10* | *0.05* | *0.04* | *0.03* | 0.048 |

**Table 4.4:** Quantitative results of the ATE metric in meters for the sequences from the TUM RGB-D [80] datasets. Taken from [6]

| Methods | Input | s-t-near | s-t-far | s-not-near | s-not-far | mean |
|---|---|---|---|---|---|---|
| LSD-SLAM [38] | RGB | - | 0.214 | - | - | 0.214 |
| CNN-SLAM [50] | RGB | - | 0.037 | - | - | 0.037 |
| ORB-SLAM [15] | RGB | 0.016 | 0.015 | × | × | **0.016** |
| -*w* Segmentation | RGB | **0.014** | **0.014** | **0.065** | **0.281** | 0.094 |
| -*w* GeoNet [82] | RGB | 0.107 | **0.014** | 0.068 | × | 0.063 |
| -*w* AHC [83] | RGB-D | *0.015* | *0.013* | *0.015* | *0.220* | *0.066* |
| LPVO [3] | RGB-D | *0.11* | *0.17* | *0.08* | *0.07* | *0.108* |

Comparing the lines for ORB-SLAM and Ours with the normals from the plane segmentation (-w Segmentation) network shows similar performance on the living room scenes. For trajectory 3 of the office scene (of-kt3) the tables show that ORB-SLAM leads to a significantly larger ATE, while the proposed method shows an error comparable to the other scenes. This is also emphasized in the RPE plot in figure 4.8 and the trajectory comparison in figure 4.9. The error on the textured scenes from the TUM RGB-D is comparable for both methods. However, ORB-SLAM looses tracking on the notexture sequences. Our method shows a low error for the near case, but also struggles with the far trajectory. It is however able to keep tracking through out the whole trajectory.

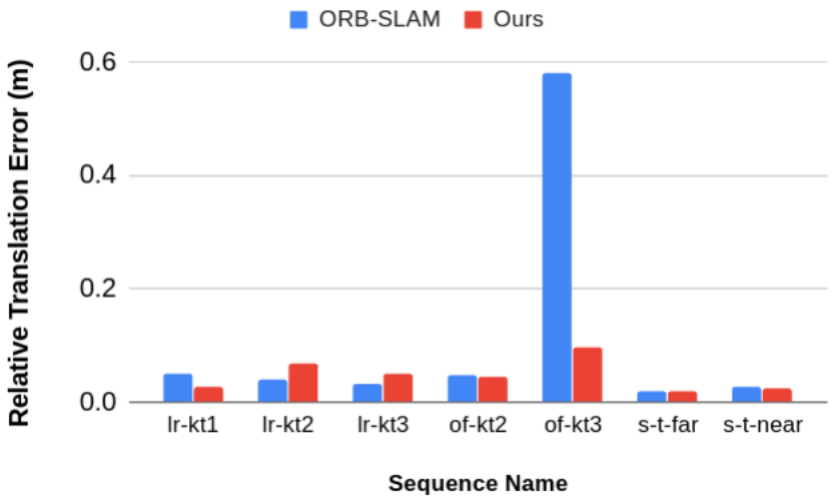



**Figure 4.8:** Comparison of the RPE error on the different sequences from ICL-NUIM and the TUM RGB-D dataset. Taken from [6] (Copyright © 2020, IEEE)

It seems that in the case of the ICL-NUIM of-kt3 trajectory ORB-SLAM struggles especially with the initialization. As the camera barely moves in the beginning of the sequence it can not find a sufficient number of point features to estimate a homography or fundamental matrix. Using the decoupled MW based rotation estimation and point-line based translation estimation our method is able to initialize also in these difficult scenarios with only a few available features.

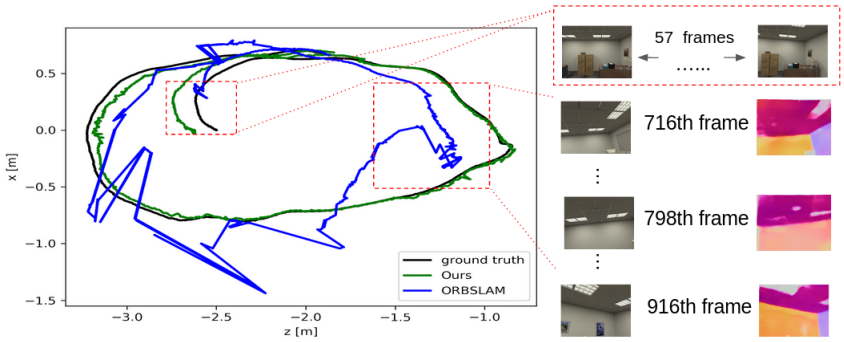



**Figure 4.9:** Comparison of the estimated trajectories from ORB-SLAM and the proposed method on the of-k3 sequence from the ICL NUIM dataset. Taken from [6] (Copyright © 2020, IEEE)

Figure 4.10 shows the rotation error of both ORB-SLAM and our method over the sequence. It can be seen that the rotation error estimated by the proposed method using the MW initialization shows a lower error overall.

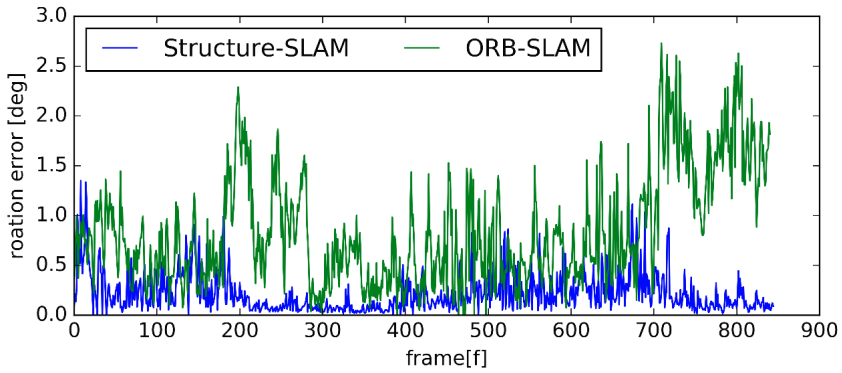



**Figure 4.10:** Relative rotation error comparison between ORB-SLAM and our method on sequence lr-k2. Taken from [6] (Copyright © 2020, IEEE)

The evaluation on the longer sequence of the HRBB4 dataset in figure 4.11 shows that our method is able to start tracking a bit earlier than ORB-SLAM, as the robot starts moving very slowly in the beginning. While ORB-SLAM initializes around frame 628, we are able to initialized around frame 110. Looking at the upper right corner we can

also see a smoother, more stable trajectory during the turn. This difference is probably due to the relatively fast rotation of the robot that reduces the number of features shared between frames.

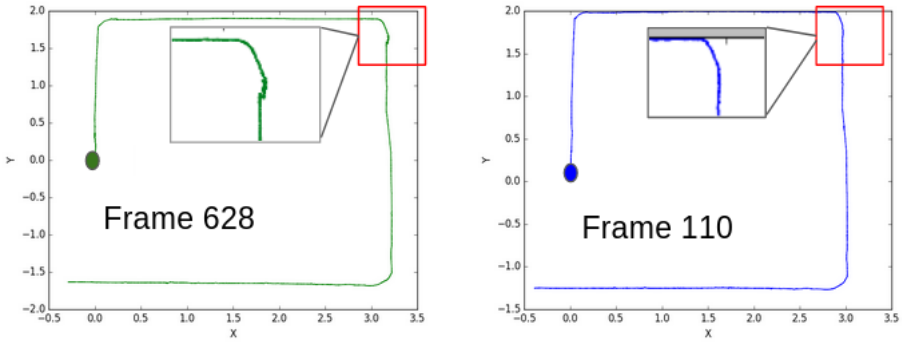



**Figure 4.11:** The estimated trajectories of the camera on the HRBB4 [81] dataset. Left: ORB-SLAM, Right: Structure-SLAM. Taken from [6] (Copyright © 2020, IEEE)

**Impact of Normals** The bottom three lines in tables 4.3 and 4.4 show the results of our method using different sources for the normals. Using the normal maps extracted from the sensor depth using AHC shows the best results and is able to track all of the sequences successfully, highlighting the potential of using highly accurate normal maps for rotation estimation. The comparison between the plane segmentation network and GeoNet further highlights the importance of good normals showing more outlier results including loss of tracking altogether. However, the quality of the predicted normals from the segmentation network seem to be sufficiently good to act as priors to reduce the rotational drift in structured environments.

# 5 Conclusion & Outlook

## 5.1 Summary

In this thesis we have explored two different types of learned priors to improve the performance of monocular SLAM systems.

We have proposed a probabilistic map point model that leverages semantics, predicted by a deep neural network, as a prior for the estimation of the points inlier ratio. This allows the system to focus on the static parts of the environment and delay the activation of potentially dynamic points until sufficient consistent geometric measurements have been obtained. To account for the reduced number of active points and to improve the robustness to fast motions and texture-less environments we complement descriptive features with direct ones and optimize them jointly. In our experiments we show that the proposed method is more robust than the baseline, especially in highly dynamic environments.

We also introduced a monocular SLAM system that uses predicted surface normals to pre-align incoming frames to a global Manhattan frame reducing drift in structured environments. To further improve performance in texture-less regions we combines point and lines features for the translation estimation and following refinement. The use of the Manhattan world alignment for initialization and initial pose estimation enables the tracking in scenes where only few features are available. An unconstrained refinement and fallback to incremental pose estimation enables the tracking also in cases where the Manhattan frame is not visible or in environments that do not strictly adhere to the MW assumption. We show in our experiments that the proposed pipeline can improve the pose estimation performance in challenging environments with a lack of texture information.

While this thesis focuses on the monocular setting, the proposed use of semantic and geometric priors in form of a probabilistic map point model and the predicted dense normals for rotation estimation can also be used in other settings using RGB-D and LiDAR sensors to improve the performance in dynamic environments and in the case of sparse or noisy measurements.

## 5.2 Limitations

Each of the proposed methods come with their own set of limitations.

In their current form the semantic priors only capture coarse information on the dynamic state of the objects and all potentially dynamic objects are treated in the same way. This leads to the same prior for parked cars, driving cars and pedestrians. While

some semantic classes could be considered as almost always static, like walls and buildings, some objects are mostly static, but can be moved like windows and doors can be opened, objects can be carried or moved by people and trees and traffic signs can be moved by the wind. This can make it difficult to define a general probability just based on the object class. A more fine grained differentiation based on the image context, e.g. by differentiating between cars parked on a parking lot and those stopped in front of a traffic light based on the scene context, could help to speed up the convergence of map points. The special case of objects changing between static and dynamic states over time is especially challenging, reducing the prior probability of the inlier ratio for dynamic classes could prevent the inclusion of potentially dynamic, but temporarily static objects. However, this would also increase the average time until a potentially dynamic point can be used for tracking. In general, due to the continuous accumulation of measurements, the probabilistic model can cause a certain latency in detecting state transitions.

The assumption of an underlying global Manhattan structure is only helpful in areas that show this underlying structure. The need for sequential tracking of the Manhattan frame due to the ambiguity of its orientation limits the possible relative rotation between frames for which the correct axis assignment can be safely detected. The detection of the MW frame after tracking loss or the detection of multiple independent frames in different parts of the map is currently not supported.

The success of using learned models to estimate priors is strongly affected by the generalization capability of the trained networks, which in turn depends on the training data. In the case of the prediction of a normal map from a RGB image the network can overfit to a certain camera model. When the model is then later applied to images recorded by a different camera the orientation of the normals might not be correct anymore. In the case of semantic segmentation the model can show bad performance, if the domain gap between the training data and the environment the system is applied in is too large. A lack of larger datasets for the evaluation in dynamic and texture-less environments limits the generalization of the conducted experiments to the real world. The use of different pipelines, implementations, parameters and hardware makes the direct comparison of SLAM frameworks difficult, limiting the comparability of the results. Open-source re-implementations and realistic benchmark settings as proposed in SLAMBench [86] can help to improve the comparability of different methods.

## 5.3 Recent Developments

Since the main work of this thesis was conducted several works have been proposed tackling the same challenges. While there are several works targeted at RGB-D settings, where the measured depth can provide crucial information about the dynamics and structure of the scenes, only a few new approaches focused on monocular SLAM methods [57, 68].

The use of compact point and line features makes the pipeline efficient, but limits their sensitivity and descriptiveness decreasing the matching accuracy. Recent visual SLAM frameworks [57] more and more use learned features [87, 88]. These may initially

require more computation, but the improved matching accuracy allows the reduction of the overall number of points that need to be tracked.

The availability of real-time depth prediction networks [89] brings the monocular setting closer to the RGB-D methods, enabling the direct alignment of local structures, reducing the dependency on sparse features.

However, recent developments in implicit 3d reconstruction and novel view synthesis have shown that they face similar challenges with respect to transient object occlusions and obtaining highly accurate poses [90, 91, 92, 93]. Also recent learning based methods that aim to solve the pose estimation and map reconstruction problem jointly with learned priors on the scene geometry and camera poses show similar limitations [89, 94].

Therefore, methods have been proposed that try to jointly reason about the dynamic and static elements in the scene. Some use multi-body tracking to reason about motion on an instance level [95]. Others aim at decoupling dynamic elements from the static parts of the scene using additional clues such as optical flow [96].

## 5.4 Future Challenges

In addition to the limitations and recent developments listed above, here we discuss additional future directions in monocular SLAM.

The scale ambiguity of the estimated poses and maps from monocular systems limits their use for downstream applications and re-use by other devices. Recently it has been shown that deep neural networks can learn to predict metric depth from images [97, 98], using this depth would allow to recover the absolute scale of the scene as well as provide additional priors for tracking and map reconstruction.

Many modern mobile platforms have multiple cameras and other sensor modalities that can be used to mitigate the challenges tackled in this thesis. Combining multiple cameras with other modalities such as inertial and magnetic measurements from an attached IMU and the mapping of WiFi and mobile network stations can provide scale information, disambiguate and validate poses and allow global localization improving the overall robustness.

Further high-level reasoning about dynamic objects in the environment and the structure and layout of the scene can further improve the robustness and accuracy. Tracking objects over time in combination with the analysis of the context allows to better judge, if an object should be used for ego pose estimation or not. The interpretation of the map as room and floor plans or as a road network and the use of learned knowledge or publicly available maps can enable the reasoning about the global structure of the map in order to correct the alignment over larger distances. A good estimation of the current pose drift and map uncertainty can help to find the most plausible scene structure.

# Bibliography


[1] H. Li, J. Yao, J.-C. Bazin, X. Lu, Y. Xing, and K. Liu. A monocular slam system leveraging structural regularity in manhattan world. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2518–2525. IEEE, 2018.

[2] H. Zhou, D. Zou, L. Pei, R. Ying, P. Liu, and W. Yu. Structslam: Visual slam with building structure lines. *IEEE Transactions on Vehicular Technology*, 64(4):1364–1375, 2015.

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

[4] P. Kim, B. Coltin, and H. J. Kim. Linear rgb-d slam for planar environments. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 333–348, 2018.

[5] N. Brasch, A. Bozic, J. Lallemand, and F. Tombari. Semantic monocular slam for highly dynamic environments. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 393–400, 2018.

[6] Y. Li, N. Brasch, Y. Wang, N. Navab, and F. Tombari. Structure-slam: Low-drift monocular slam in indoor environments. *IEEE Robotics and Automation Letters*, 5(4):6583–6590, 2020.

[7] O. Özyeşil, V. Voroninski, R. Basri, and A. Singer. A survey of structure from motion*. *Acta Numerica*, 26:305–364, 2017.

[8] D. Scaramuzza and F. Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.

[9] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. MIT Press, 2005.

[10] 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.

[11] H. Strasdat, J. M. Montiel, and A. J. Davison. Visual slam: why filter? *Image and Vision Computing*, 30(2):65–77, 2012.

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

[13] E. P. Herrera-Granda, J. C. Torres-Cantero, and D. H. Peluffo-Ordóñez. Monocular visual slam, visual odometry, and structure from motion methods applied to 3d reconstruction: A comprehensive survey. *Heliyon*, 2024.

[14] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2007.

[15] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *Transactions on Robotics*, 31(5), 2015.

[16] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006: 9th European Conference on Computer Vision, Graz, Austria, May 7-13, 2006. Proceedings, Part I 9*, pages 430–443. Springer, 2006.

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60:91–110, 2004.

[18] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[19] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011.

[20] 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.

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

[22] C. G. Harris and M. J. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[23] M. Pizzoli, C. Forster, and D. Scaramuzza. REMODE: Probabilistic, monocular dense reconstruction in real time. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2609–2616. IEEE, may 2014.

[24] L. Quan and Z. Lan. Linear n-point camera pose determination. *IEEE Transactions on pattern analysis and machine intelligence*, 21(8):774–780, 1999.

[25] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003.

[26] V. Lepetit, F. Moreno-Noguer, and P. Fua. Ep n p: An accurate o (n) solution to the p n p problem. *International journal of computer vision*, 81:155–166, 2009.

[27] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the pnp problem: A fast, general and optimal solution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2344–2351, 2013.

[28] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293(5828):133–135, 1981.

[29] R. I. Hartley. In defense of the eight-point algorithm. *IEEE Transactions on pattern analysis and machine intelligence*, 19(6):580–593, 1997.

[30] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004.

[31] O. D. Faugeras and F. Lustman. Motion and structure from motion in a piecewise planar environment. *International Journal of Pattern Recognition and Artificial Intelligence*, 2(03):485–508, 1988.

[32] R. G. Von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd: A fast line segment detector with a false detection control. *IEEE transactions on pattern analysis and machine intelligence*, 32(4):722–732, 2008.

[33] L. Zhang and R. Koch. An efficient and robust line segment matching approach based on lbd descriptor and pairwise geometric consistency. *Journal of visual communication and image representation*, 24(7):794–805, 2013.

[34] 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.

[35] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez. Pl-slam: A stereo slam system through the combination of points and line segments. *IEEE Transactions on Robotics*, 35(3):734–746, 2019.

[36] A. Bartoli and P. Sturm. Structure-from-motion using lines: Representation, triangulation, and bundle adjustment. *Computer vision and image understanding*, 100(3):416–441, 2005.

[37] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *2011 International Conference on Computer Vision*. IEEE, nov 2011.

[38] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *European Conference on Computer Vision (ECCV)*, 2014.

[39] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *International Conference on Robotics and Automation (ICRA)*. IEEE, 2014.

[40] J. Engel, V. Koltun, and D. Cremers. Direct sparse odometry. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(3):611–625, 2018.

[41] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'81, San Francisco, CA, USA, 1981. Morgan Kaufmann Publishers Inc.

[42] Y. Zhou, L. Kneip, C. Rodriguez, and H. Li. Divide and conquer: Efficient density-based tracking of 3d sensors in manhattan worlds. In *Asian Conference on Computer Vision*, pages 3–19. Springer, 2016.

[43] G. Vogiatzis and C. Hernández. Video-based, real-time multi-view stereo. *Image and Vision Computing*, 29(7), 2011.

[44] E. T. Jaynes. *Probability theory: The logic of science*. Cambridge university press, 2003.

[45] D. Brown. The bundle adjustment-progress and prospect. In *XIII Congress of the ISPRS, Helsinki, 1976*, 1976.

[46] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon. Bundle adjustment—a modern synthesis. In *Vision Algorithms: Theory and Practice: International Workshop on Vision Algorithms Corfu, Greece, September 21–22, 1999 Proceedings*, pages 298–372. Springer, 2000.

[47] C. Engels, H. Stewénius, and D. Nistér. Bundle adjustment rules. *Photogrammetric computer vision*, 2(32), 2006.

[48] Z. Yu, J. Zheng, D. Lian, Z. Zhou, and S. Gao. Single-image piece-wise planar 3d reconstruction via associative embedding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1029–1037, 2019.

[49] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. *IEEE International Conference on Intelligent Robots and Systems*, 2012.

[50] K. Tateno, F. Tombari, I. Laina, and N. Navab. CNN-SLAM: Real-time dense monocular SLAM with learned depth prediction. In *Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.

[51] J. Shi et al. Good features to track. In *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pages 593–600. IEEE, 1994.

[52] 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.

[53] J. Engel, J. Sturm, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *Proceedings of the IEEE international conference on computer vision*, pages 1449–1456, 2013.

[54] A. Glover, W. Maddern, M. Warren, S. Reid, M. Milford, and G. Wyeth. Openfabmap: An open source toolbox for appearance-based loop closure detection. In *2012 IEEE international conference on robotics and automation*, pages 4730–4735. IEEE, 2012.

[55] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g 2 o: A general framework for graph optimization. In *2011 IEEE international conference on robotics and automation*, pages 3607–3613. IEEE, 2011.

[56] 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.

[57] Y. Wang, Y. Tian, J. Chen, K. Xu, and X. Ding. A survey of visual slam in dynamic environment: The evolution from geometric to semantic approaches. *IEEE Transactions on Instrumentation and Measurement*, 2024.

[58] A. Kundu, K. M. Krishna, and C. Jawahar. Realtime motion segmentation based multibody visual slam. In *Proceedings of the Seventh Indian Conference on Computer Vision, Graphics and Image Processing*, pages 251–258, 2010.

[59] J. Shimamura, M. Morimoto, and H. Koike. Robust vSLAM for dynamic scenes. *Proceedings of the 12th IAPR Conference on Machine Vision Applications*, 2011.

[60] A. Kundu, K. M. Krishna, and C. Jawahar. Realtime multibody visual slam with a smoothly moving monocular camera. In *2011 International Conference on Computer Vision*, pages 2080–2087. IEEE, 2011.

[61] A. Roussos, C. Russell, R. Garg, and L. Agapito. Dense multibody motion estimation and reconstruction from a handheld camera. In *2012 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 31–40. IEEE, 2012.

[62] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao. Robust monocular slam in dynamic environments. In *International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2013.

[63] H. Pu, J. Luo, G. Wang, T. Huang, and H. Liu. Visual slam integration with semantic segmentation and deep learning: A review. *IEEE Sensors Journal*, 23(19):22119–22138, 2023.

[64] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg. Joint semantic segmentation and 3d reconstruction from monocular video. In *Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI*. Springer International Publishing, 2014.

[65] N. D. Reddy, P. Singhal, V. Chari, and K. M. Krishna. Dynamic body vslam with semantic constraints. In *International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015.

[66] 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 *International Conference on Robotics and Automation (ICRA)*. IEEE, 2015.

[67] J. Straub, N. Bhandari, J. J. Leonard, and J. W. Fisher. Real-time manhattan world rotation estimation in 3d. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1913–1920. IEEE, 2015.

[68] W. Zhao, H. Sun, X. Zhang, and Y. Xiong. Visual slam combining lines and structural regularities: Towards robust localization. *IEEE Transactions on Intelligent Vehicles*, 9(6):5047–5064, 2023.

[69] G. Zhang, D. H. Kang, and I. H. Suh. Loop closure through vanishing points in a line-based monocular slam. In *2012 IEEE International Conference on Robotics and Automation*, pages 4565–4570. IEEE, 2012.

[70] P. Kim, B. Coltin, and H. J. Kim. Visual odometry with drift-free rotation estimation using indoor scene regularities. In *BMVC*, volume 2, page 7, 2017.

[71] 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.

[72] H. Zhao, X. Qi, X. Shen, J. Shi, and J. Jia. Icnet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 405–420, 2018.

[73] J. Civera, A. J. Davison, and J. M. M. Montiel. Inverse Depth Parameterization for Monocular {SLAM}. *IEEE Transactions on Robotics*, 24(5), 2008.

[74] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[75] M. Z. Zia, L. Nardi, A. Jack, E. Vespa, B. Bodin, P. H. Kelly, and A. J. Davison. Comparative design space exploration of dense and semi-dense SLAM. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, may 2016. arXiv:1509.04648.

[76] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[77] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, 2016.

[78] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. 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)*, 2016.

[79] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *2014 IEEE international conference on Robotics and automation (ICRA)*, pages 1524–1531. IEEE, 2014.

[80] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012.

[81] Y. Lu, D. Song, and J. Yi. High level landmark-based visual navigation using unsupervised geometric constraints in local bundle adjustment. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1540–1545. IEEE, 2014.

[82] X. Qi, R. Liao, Z. Liu, R. Urtasun, and J. Jia. Geonet: Geometric neural network for joint depth and surface normal estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 283–291, 2018.

[83] C. Feng, Y. Taguchi, and V. R. Kamat. Fast plane extraction in organized point clouds using agglomerative hierarchical clustering. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6218–6225. IEEE, 2014.

[84] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[85] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *Computer Vision–ECCV 2012: 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V 12*, pages 746–760. Springer, 2012.

[86] M. Bujanca, P. Gafton, S. Saeedi, A. Nisbet, B. Bodin, M. F. O'Boyle, A. J. Davison, P. H. Kelly, G. Riley, B. Lennox, et al. Slambench 3.0: Systematic automated reproducible evaluation of slam systems for robot vision challenges and scene understanding. In *2019 international conference on robotics and automation (ICRA)*, pages 6351–6358. IEEE, 2019.

[87] D. DeTone, T. Malisiewicz, and A. Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018.

[88] P. Gleize, W. Wang, and M. Feiszli. Silk: Simple learned keypoints. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22499–22508, 2023.

[89] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20697–20709, 2024.

[90] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7210–7219, 2021.

[91] W. Ren, Z. Zhu, B. Sun, J. Chen, M. Pollefeys, and S. Peng. Nerf on-the-go: Exploiting uncertainty for distractor-free nerfs in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8931–8940, 2024.

[92] J. Kulhanek, S. Peng, Z. Kukelova, M. Pollefeys, and T. Sattler. Wildgaussians: 3d gaussian splatting in the wild. In *The Thirty-Eighth Annual Conference on Neural Information Processing Systems (NeurIPS 2024)*, 2024.

[93] D. Zhang, C. Wang, W. Wang, P. Li, M. Qin, and H. Wang. Gaussian in the wild: 3d gaussian splatting for unconstrained image collections. In *European Conference on Computer Vision*, pages 341–359. Springer, 2024.

[94] J. Wang, N. Karaev, C. Rupprecht, and D. Novotny. Vggsfm: Visual geometry grounded deep structure from motion. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 21686–21697, 2024.

[95] L. Yang, Y. Zhang, R. Tian, S. Liang, Y. Shen, S. Coleman, and D. Kerr. Fast, robust, accurate, multi-body motion aware slam. *IEEE Transactions on Intelligent Transportation Systems*, 25(5):4381–4397, 2023.

[96] J. Yang, B. Ivanovic, O. Litany, X. Weng, S. W. Kim, B. Li, T. Che, D. Xu, S. Fidler, M. Pavone, et al. Emernerf: Emergent spatial-temporal scene decomposition via self-supervision. In *The Twelfth International Conference on Learning Representations*, 2024.

[97] M. Hu, W. Yin, C. Zhang, Z. Cai, X. Long, H. Chen, K. Wang, G. Yu, C. Shen, and S. Shen. Metric3d v2: A versatile monocular geometric foundation model for zero-shot metric depth and surface normal estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[98] L. Piccinelli, Y.-H. Yang, C. Sakaridis, M. Segu, S. Li, L. Van Gool, and F. Yu. Unidepth: Universal monocular metric depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10106–10116, 2024.

[99] C. Bishop. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer, 2006.

[100] S. S. Dragomir, R. P. Agarwal, and N. S. Barnett. Inequalities for beta and gamma functions via some classical and new integral inequalities. *Journal of Inequalities and Applications*, 2000(2):504054, 2000.

# Acronyms

| | |
|---|---|
| ATE | Absolute Trajectory Error. |
| BA | Bundle Adjustment. |
| BRIEF | Binary Robust Independent Elementary Features. |
| CNN | Convolutional Neural Network. |
| CRF | Conditional Random Field. |
| DLT | Direct Linear Transform. |
| DoF | Degree(s) of Freedom. |
| DR | Depth inlier Ratio. |
| EKF | Extended Kalman Filter. |
| EM | Expectation Maximization. |
| FAST | Features from Accelerated Segment Test. |
| GMM | Gaussian Mixture Model. |
| GPS | Global Positioning System. |
| IMU | Inertial Measurement Unit. |
| LBD | Line Band Descriptor. |
| LiDAR | Light Detection and Ranging. |
| LSD | Line Segment Detector. |
| MAP | Maximum A Posteriori. |
| MW | Manhattan World. |
| NCC | Normalized Cross Correlation. |
| NMS | Non-Maximum Suppression. |
| ORB | Oriented FAST and rotated BRIEF. |
| PnP | Perspective-n-Point. |

| | |
|---|---|
| RANSAC | RANdom SAmple Consensus. |
| RMSE | Root Mean Squared Error. |
| RPE | Relative Pose Error. |
| RTK | Real Time Kinematic. |
| SAM | Smoothing and Mapping. |
| SfM | Structure from Motion. |
| SIFT | Scale Invariant Feature Transform. |
| SLAM | Simultaneous Localization and Mapping. |
| SSD | Sum of Squared Differences. |
| std | STandard Deviation. |
| SVD | Singular Value Decomposition. |
| VO | Visual Odometry. |

# A Appendix

## A.1 Semantic Outlier Model Derivations

In the following we provide the derivations of the approximate posterior and online updates for the proposed probabilistic map point model. First, we will derive the optimal approximate posteriors for the semantic and depth models. We then compute the true posteriors and apply moment matching to compute the efficient online updates.

### A.1.1 Approximate Posteriors

We want to derive the approximate posterior distributions for the semantic class probability

$$q(c_k|S) \propto \prod_{i=1}^{N} CNN(c_k|I_i)^{\alpha_i}$$

and the joint depth and inlier ratio distribution.

$$q(Z, \phi|D, S) = N(Z|\mu, \sigma^2)Beta(\phi, a_{obs}, b_{obs})Beta(\phi, a_{sem}, b_{sem})$$

Therefore, we follow Vogiatzis *et al.* [43] using the theorem of approximate inference for factorized posteriors [99, Chapter 10] that states: If the posterior can be factorized into independent parts for each variable

$$q(\boldsymbol{x}) = \prod_i q_i(x_i)$$

the best approximation must fulfill the following constraint.

$$\ln q_i^*(\boldsymbol{z_i}) = \mathbf{E}_{j \neq i}\left[\ln p(\boldsymbol{x}, \boldsymbol{z})\right] + const.$$

This means that to estimate the best approximate posterior for factor $i$ we can compute the expectation of the joint distribution with respect to all other factors. Therefore we first have to assemble the joint distributions and can then compute the expectation.

The expected value is defined as:

$$\mathbf{E}_{j \neq i}\left[x\right] = \int x p(x) dx = \sum_{i}^{N} x_i p(x_i)$$

However, first we introduce two binary latent variables $x_n \in \{0, 1\}$ for the matching accuracy and $y_n \in \{0, 1\}$ for the inlier ratio of each measurement $n$.

$$x_n := \begin{cases} 1; & \text{matching is accurate at measurement } n \\ 0; & \text{otherwise} \end{cases}$$

$$y_n := \begin{cases} 1; & \text{map point is static at measurement } n \\ 0; & \text{otherwise} \end{cases}$$

We model $x_n$ and $y_n$ as Bernoulli random variables using $\alpha_n$ and $\phi$ respectively.

$$p(x_n) := Bernoulli(\alpha_n, 1-\alpha_n) = \alpha_n^{x_n}(1-\alpha_n)^{1-x_n}$$
$$p(y_n|\phi) := Bernoulli(\phi, 1-\phi) = \phi^{y_n}(1-\phi)^{1-y_n}$$

We can then represent a vector of random variables for $n$ measurements as

$$\boldsymbol{X} = \{x_1, \ldots, x_n\}$$
$$\boldsymbol{Y} = \{y_1, \ldots, y_n\}$$

With this re-parameterization we can re-write the probability density function of our depth measurement model to be conditioned on $x_n$ and $y_n$:

$$p(d_n|Z, x_n, y_n) = \left[N(d_n|Z, \tau_n^2)^{y_n} U(d_n)^{1-y_n}\right]^{x_n} U(d_n)^{1-x_n} = \\ = N(d_n|Z, \tau_n^2)^{x_n y_n} U(d_n)^{1-x_n y_n}$$

We can do the same for the semantic class probability.

$$p(c_k|I_n, x_n) = CNN(c_k|I_n)^{x_n} U(c_k)^{1-x_n}$$

Here we use $CNN(c_i|I_n)$ to denote the predicted probability distribution from the CNN.

#### A.1.1.1 Semantic Measurement Model

First, we write down the joint posterior for the semantic classes $\boldsymbol{c} = \{c_1, \ldots, c_K\}$, the matching accuracies and the images $\boldsymbol{I}$.

$$p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) = p(\boldsymbol{I})p(\boldsymbol{X})p(\boldsymbol{c}|\boldsymbol{X}, \boldsymbol{I})$$

First, we assume that the class labels $c_i$ are independent.

$$p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) = p(\boldsymbol{I})p(\boldsymbol{X}) \prod_{i=1}^{K} p(c_i|\boldsymbol{X}, \boldsymbol{I})$$

Then we apply Bayes theorem

$$p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) = p(\boldsymbol{I})p(\boldsymbol{X}) \prod_{i=1}^{K} \frac{p(\boldsymbol{X}, \boldsymbol{I}|c_i)p(c_i)}{p(\boldsymbol{X}, \boldsymbol{I})}$$

We further assume that the image measurements $I_n$ and the latent variables $x_n$ are independent

$$p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) = p(\boldsymbol{I})p(\boldsymbol{X}) \prod_{i=1}^{K} \frac{\left(\prod_{n=1}^{N} p(x_n, I_n|c_i)\right) p(c_i)}{\prod_{n=1}^{N} p(x_n, I_n)}$$

Finally, we apply Bayes rule again

$$p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) = p(\boldsymbol{I})p(\boldsymbol{X}) \prod_{i=1}^{K} \left( \left( \prod_{n=1}^{N} \frac{p(c_i|x_n, I_n)p(x_n, I_n)}{p(c_i)} \right) \cdot \frac{p(c_i)}{\prod_{n=1}^{N} p(x_n, I_n)} \right)$$

to obtain the joint distribution

$$p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) = p(\boldsymbol{I})p(\boldsymbol{X}) \prod_{i=1}^{K} \left( \frac{1}{p(c_i)^{N-1}} \prod_{n=1}^{N} p(c_i|x_n, I_n) \right)$$

If we now further assume, that we can factorize the posterior into independent parts for semantic class $\boldsymbol{c}$ and matching accuracy $\boldsymbol{X}$.

$$p(\boldsymbol{c}, \boldsymbol{X}|\boldsymbol{I}) := q(\boldsymbol{c}, \boldsymbol{X}) = q_{\boldsymbol{c}}(\boldsymbol{c}) q_{\boldsymbol{X}}(\boldsymbol{X})$$

we can use the theorem of approximate inference to compute the optimal posterior.

$$\ln q_{\boldsymbol{c}}(\boldsymbol{c}) = \mathbf{E}_{\boldsymbol{X}} \left[ \ln p(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{I}) \right] + const. =$$
$$\ln p(\boldsymbol{I}) + \mathbf{E}_{\boldsymbol{X}} \left[ \ln p(\boldsymbol{X}) \right] - (N-1) \sum_{i=1}^{K} \ln p(c_i) +$$
$$\sum_{i=1}^{K} \sum_{n=1}^{N} \mathbf{E}_{\boldsymbol{X}} \left[ \ln p(c_i|x_n, I_n) \right] + const.$$

Several terms in the above equation are constant with respect to $\boldsymbol{c}$ and can therefore be merged into a single constant value. Furthermore, we assume that the prior for $\boldsymbol{c}$ is non-informative, i.e. $p(c_i) = \mathcal{U}(c_i)$ and also integrate it into the constant value.

$$\ln q_{\boldsymbol{c}}(\boldsymbol{c}) = \sum_{i=1}^{K} \sum_{n=1}^{N} \left( \mathbf{E}_{\boldsymbol{X}} \left[ x_n \right] \ln CNN(c_i|I_n) + (1 - \mathbf{E}_{\boldsymbol{X}} \left[ x_n \right]) U(c_i) \right) + const. =$$
$$= \sum_{i=1}^{K} \sum_{n=1}^{N} \mathbf{E}_{\boldsymbol{X}} \left[ x_n \right] \ln CNN(c_i|I_n) + const.$$

We can compute the expected value as

$$\mathbf{E}_{\boldsymbol{X}}[x_n] = 0 \cdot p(x_n = 0) + 1 \cdot p(x_n = 1) = \alpha_n$$

We then exponentiate both sides of the equation:

$$q_{\boldsymbol{c}}(\boldsymbol{c}) = K_1 \prod_{i=1}^{K} \prod_{n=1}^{N} CNN(c_i|I_n)^{\alpha_n}$$

Assuming independence of the different class probabilities this gives us the following term for each class, where $K_i'$ is the normalization constant.

$$q_{c_i}(c_i) = K_i' \prod_{n=1}^{N} CNN(c_i|I_n)^{\alpha_n}$$

### A.1.1.2 Depth Measurement Model

For the depth we write down the joint probability distribution of $Z$, $\phi$ and $\boldsymbol{c}$ as:

$$\begin{aligned}
&p(Z, \phi, \boldsymbol{c}, \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{D}, \boldsymbol{I}) = \\
&= p(\boldsymbol{I})p(Z)p(\boldsymbol{X})p(\boldsymbol{c}|\boldsymbol{I}, \boldsymbol{X})p(\phi|\boldsymbol{c})p(\boldsymbol{Y}|\phi)p(\boldsymbol{D}|Z, \boldsymbol{X}, \boldsymbol{Y}) = \\
&= p(\boldsymbol{I})p(Z)p(\boldsymbol{X})p(\boldsymbol{c}|\boldsymbol{I}, \boldsymbol{X})p(\phi|\boldsymbol{c}) \prod_{n=1}^{N} (p(y_n|\phi)p(d_n|Z, x_n, y_n))
\end{aligned}$$

Here we again assume that the $N$ measurements are independent and denote the joint posterior distribution with

$$r(Z, \phi, \boldsymbol{c}, \boldsymbol{X}, \boldsymbol{Y}) := p(Z, \phi, \boldsymbol{c}, \boldsymbol{X}, \boldsymbol{Y}|\boldsymbol{D}, \boldsymbol{I})$$

We further assume that the discrete variables and the continuous variables can split into independent factors:

$$r(Z, \phi, \boldsymbol{c}, \boldsymbol{X}, \boldsymbol{Y}) = q_{Z,\phi}(Z, \phi) q_{\boldsymbol{c},\boldsymbol{X},\boldsymbol{Y}}(\boldsymbol{c}, \boldsymbol{X}, \boldsymbol{Y})$$

We can once again use the property of the theorem of approximate inference

$$\ln q_{Z,\phi}(Z, \phi) = \mathbf{E}_{\boldsymbol{c},\boldsymbol{X},\boldsymbol{Y}}[\ln p(Z, \phi, \boldsymbol{c}, \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{D}, \boldsymbol{I})] + const.$$

to compute the approximate posterior.

$$\ln q_{Z,\phi}(Z,\phi) = \ln p(\boldsymbol{I}) + \ln p(Z) + \mathbf{E}_{\boldsymbol{X}}\left[\ln p(\boldsymbol{X})\right] + \\ + \mathbf{E}_{\boldsymbol{c},\boldsymbol{X}}\left[\ln p(\boldsymbol{c}|\boldsymbol{I},\boldsymbol{X})\right] + \mathbf{E}_{\boldsymbol{c}}\left[\ln p(\phi|\boldsymbol{c})\right] + \\ + \sum_{n=1}^{N}\Big(\mathbf{E}_{\boldsymbol{Y}}\left[\ln p(y_n|\phi)\right] + \mathbf{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[\ln p(d_n|Z,x_n,y_n)\right]\Big) + const.$$

Here we can again summarize many parts that do not depend on the parameters $Z$ and $\phi$ in a single constant value. We can also integrate the terms $\mathbf{E}_{\boldsymbol{X}}\left[x_n\right]$, $\mathbf{E}_{\boldsymbol{Y}}\left[y_n\right]$ and $\mathbf{E}_{\boldsymbol{c}}\left[\boldsymbol{c}\right]$ that are constant with respect to $Z$ and $\phi$.

$$\ln q_{Z,\phi}(Z,\phi) = \ln p(Z) + \mathbf{E}_{\boldsymbol{c}}\left[\ln p(\phi|\boldsymbol{c})\right] + \sum_{n=1}^{N}\Big(\mathbf{E}_{\boldsymbol{Y}}\left[y_n\right]\ln\phi + (1-\mathbf{E}_{\boldsymbol{Y}}\left[y_n\right])(1-\phi) + \\ + \mathbf{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[x_n y_n\right]\ln N(d_n|Z,\tau_n^2) + (1-\mathbf{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[x_n y_n\right])\ln U(d_n)\Big) + const.$$

We already know that $\mathbf{E}_{\boldsymbol{X}}\left[x_n\right] = \alpha_n$. We can further simplify the notation by defining:

$$\beta_n := \mathbf{E}_{\boldsymbol{Y}}\left[y_n\right] \text{ for } n = 1,\ldots,N$$
$$\gamma_i := \mathbf{E}_{\boldsymbol{c}}\left[c_i\right] \text{ for } i = 1,\ldots,K$$

Assuming independence between the matching accuracy $x_n$ and the static state $y_n$ of a point we can write

$$\mathbf{E}_{\boldsymbol{X},\boldsymbol{Y}}\left[x_n y_n\right] = \mathbf{E}_{\boldsymbol{X}}\left[x_n\right]\mathbf{E}_{\boldsymbol{Y}}\left[y_n\right] = \alpha_n\beta_n$$

We can now again apply the exponential function to both sides $\ln q_{Z,\phi}(Z,\phi)$ to obtain the posterior probability.

$$q_{Z,\phi}(Z,\phi) = K_2\left[\prod_{n=1}^{N} N(d_n|Z,\tau_n^2)^{\alpha_n\beta_n}\right]\phi^S(1-\phi)^{N-S}p(Z)\exp(\mathbf{E}_{\boldsymbol{c}}\left[\ln p(\phi|\boldsymbol{c})\right])$$

Here we use $S := \sum_{n=1}^{N}\beta_n$ and $K_2$ as the normalization constant.

Looking at the posterior probability, we can see that the inlier ratio $\phi$ is distributed by a Beta distribution. The inverse depth $Z$ is present as a product of Gaussian distributions, with each term raised to the power of $\alpha_n\beta_n$ which results in a Gaussian posterior term.

$$\prod_{n=1}^{N} N(d_n|Z,\tau_n^2)^{\alpha_n\beta_n} \propto \prod_{n=1}^{N} \mathrm{e}^{-\alpha_n\beta_n\frac{(Z-d_n)^2}{2\tau_n^2}} = \\ \mathrm{e}^{\sum_{n=1}^{N}-\alpha_n\beta_n\frac{(Z-d_n)^2}{2\tau_n^2}} \propto \mathrm{e}^{-\frac{(Z-\mu)^2}{2\sigma^2}} = N(Z|\mu,\sigma^2)$$

Therefore the approximate joint posterior $q_{Z,\phi}(Z, \phi)$ can be written as a product of a Gaussian and a Beta distribution, if we define the prior of $Z$ and $\phi$ favorably. Therefore, we set the prior of $Z$ to a Gaussian distribution:

$$p(Z) := N(Z|\mu_0, \sigma_0^2)$$

And we choose a Beta distribution $\mathcal{B}(a, b)$ as the prior for $\phi$ and for each class $c_i$ define the constants $A_i$ as a static prior and $B_i$ as a dynamic prior. So we can write:

$$p(\phi|\boldsymbol{c}) := \prod_{i=1}^{K} \left( \frac{1}{B(A_i, B_i)} \phi^{A_i - 1} (1 - \phi)^{B_i - 1} \right)^{c_i}$$

With this we can compute the missing term $\exp(\mathbf{E}_{\boldsymbol{c}} [\ln p(\phi|\boldsymbol{c})])$ in the approximate posterior:

$$\begin{aligned}
&\mathbf{E}_{\boldsymbol{c}} [\ln p(\phi|\boldsymbol{c})] \\
&= \sum_{i=1}^{K} \mathbf{E}_{\boldsymbol{c}} [c_i] \ln \left( \frac{1}{B(A_i, B_i)} \phi^{A_i - 1} (1 - \phi)^{B_i - 1} \right) \\
&= \ln \left( \phi^{\sum_{i=1}^{K} \mathbf{E}_{\boldsymbol{c}}[c_i] A_i - \sum_{i=1}^{K} \mathbf{E}_{\boldsymbol{c}}[c_i]} (1 - \phi)^{\sum_{i=1}^{K} \mathbf{E}_{\boldsymbol{c}}[c_i] B_i - \sum_{i=1}^{K} \mathbf{E}_{\boldsymbol{c}}[c_i]} \right) + const. \\
&= \ln \left( \phi^{\sum_{i=1}^{K} \gamma_i A_i - 1} (1 - \phi)^{\sum_{i=1}^{K} \gamma_i B_i - 1} \right) + const.
\end{aligned}$$

With $\boldsymbol{c}$ being one-hot encoded we get

$$\sum_{i=1}^{K} \mathbf{E}_{\boldsymbol{c}} [c_i] = \mathbf{E}_{\boldsymbol{c}} \left[ \sum_{i=1}^{K} c_i \right] = 1$$

If we apply the exponential function we get

$$\exp(\mathbf{E}_{\boldsymbol{c}} [\ln p(\phi|\boldsymbol{c})]) \propto \phi^{\sum_{i=1}^{K} \gamma_i A_i - 1} (1 - \phi)^{\sum_{i=1}^{K} \gamma_i B_i - 1}$$

which is also a Beta function, parameterized by the semantic priors.

Combining both Beta distributions we can see that the final approximate posterior has the form

$$\begin{aligned}
q_{Z,\phi}(Z, \phi) =& N(Z|\mu, \sigma^2) Beta(\phi, a_{obs}, b_{obs}) Beta(\phi, a_{sem}, b_{sem}) \\
=& N(Z|\mu, \sigma^2) Beta(\phi|a, b)
\end{aligned}$$

So far the values of $\beta_n$ and $\gamma_i$ are still unknown. However, in the derivation of the online updates we will see that we can compute $\beta_n$ implicitly and that we can find an approximation for $\gamma_i$.

### A.1.2 Online Updates

Next, we are going to derive the online updates using first and second order moment matching between the derived approximate posteriors and the real posteriors.

#### A.1.2.1 True Posterior Derivation

Therefore we are first writing down the true posterior distribution. Given our current distribution is $q_{Z,\phi}(Z, \phi|a, b, \mu, \sigma^2)$, we want to compute the posterior distribution $r_{Z,\phi}(Z, \phi|a, b, \mu, \sigma^2, d_n)$ after measurement $p(d_n|Z, \phi)$. By applying Bayes theorem we get:

$$r_{Z,\phi}(Z, \phi|a, b, \mu, \sigma^2, d_n) = K' p(d_n|Z, \phi) q_{Z,\phi}(Z, \phi|a, b, \mu, \sigma^2)$$

where $K'$ is the normalizing constant from the marginal distribution. Inserting the two distributions gives us

$$\begin{aligned} r_{Z,\phi}(Z, \phi|a, b, \mu, \sigma^2, d_n) = K' \big[\alpha_n \left[\phi N(d_n|Z, \tau_n^2) + (1 - \phi)U(d_n)\right] + (1-\alpha_n)U(d_n)\big] \\ N(Z|\mu, \sigma^2) Beta(\phi|a, b) \end{aligned}$$

Multiplying out the terms results in

$$\begin{aligned} r_{Z,\phi}(Z, \phi|a, b, \mu, \sigma^2, d_n) = K' \big[&\alpha_n \phi N(d_n|Z, \tau_n^2) N(Z|\mu, \sigma^2) Beta(\phi|a, b) + \\ &+ \alpha_n (1 - \phi) U(d_n) N(Z|\mu, \sigma^2) Beta(\phi|a, b) + \\ &+ (1 - \alpha_n) U(d_n) N(Z|\mu, \sigma^2) Beta(\phi|a, b)\big] \end{aligned}$$

Using the property about the Gaussian posterior probability from [99, Chapter 2.3] we can rewrite the product $N(d_n|Z, \tau_n^2)N(Z|\mu, \sigma^2)$:

$$N(d_n|Z, \tau_n^2)N(Z|\mu, \sigma^2) = N(d_n|\mu, \sigma^2 + \tau_n^2)N(Z|m, s^2)$$

With:

$$\begin{aligned} m &= s^2 \left(\frac{\mu}{\sigma^2} + \frac{d_n}{\tau_n^2}\right) \\ s^2 &= \left(\frac{1}{\sigma^2} + \frac{1}{\tau_n^2}\right)^{-1} \end{aligned}$$

Using the following properties of the Beta function [100]

$$B(a+1,b) = \frac{a}{a+b}B(a,b) = \phi B(a,b)$$
$$B(a,b+1) = \frac{b}{a+b}B(a,b) = (1-\phi)B(a,b)$$

we can include the factors $\phi$ and $(1-\phi)$ into the Beta distribution

$$\begin{aligned} & r_{Z,\phi}(Z,\phi|a,b,\mu,\sigma^2,d_n) = \\ & = K'\big[\alpha_n \frac{a}{a+b} N(d_n|\mu,\sigma^2+\tau_n^2)N(Z|m,s^2)Beta(\phi|a+1,b)+ \\ & \quad + \alpha_n \frac{b}{a+b} U(d_n)N(Z|\mu,\sigma^2)Beta(\phi|a,b+1)+ \\ & \quad + (1-\alpha_n)U(d_n)N(Z|\mu,\sigma^2)Beta(\phi|a,b)\big] \end{aligned}$$

To simplify the notation we define the constants $C_1$, $C_2$ and $C_3$:

$$\begin{aligned} C_1 &:= \alpha_n \frac{a}{a+b} N(d_n|\mu,\sigma^2+\tau_n^2) \\ C_2 &:= \alpha_n \frac{b}{a+b} U(d_n) \\ C_3 &:= (1-\alpha_n)U(d_n) \end{aligned}$$

We can then re-write the posterior as follows:

$$\begin{aligned} r_{Z,\phi}(Z,\phi|a,b,\mu,\sigma^2,d_n) = K'\big[ & C_1 N(Z|m,s^2)Beta(\phi|a+1,b)+ \\ & + C_2 N(Z|\mu,\sigma^2)Beta(\phi|a,b+1)+ \\ & + C_3 N(Z|\mu,\sigma^2)Beta(\phi|a,b)\big] \end{aligned}$$

Finally, we compute the normalizing factor $K'$ via integration:

$$\frac{1}{K'} = \int_Z \int_\phi r_{Z,\phi}(Z,\phi|a,b,\mu,\sigma^2,d_n)d(Z)d(\phi) = C_1 + C_2 + C_3$$
$$K' = \frac{1}{C_1 + C_2 + C_3}$$

#### A.1.2.2 Moment Matching

In order to perform an online probability update we would like to know how to compute the posterior coefficients $a'$, $b'$, $\mu'$ and $\sigma'^2$ from the prior coefficients $a$, $b$, $\mu$ and $\sigma^2$.

We computed the true posterior probability using the Bayes theorem and we also have derived the optimal approximation of the posterior, which we want to be as close as possible.

$$r_{Z,\phi}(Z,\phi|a,b,\mu,\sigma^2,d_n) \approx q_{Z,\phi}(Z,\phi|a',b',\mu',\sigma'^2) = N(Z|\mu',\sigma'^2)Beta(\phi|a',b')$$

Therefore we want the moments of both representations to match. By comparing the first and second moment we get equations that involve both prior and posterior parameters, and we can compute the posterior parameters from the prior ones:

We can compute the $n$-th moment using

$$\mathbf{E}_{j\neq i}\left[x^n\right] = \int x^n p(x)dx$$

The first and second moments of the approximate Gaussian-Beta mixture posterior are

$$\begin{aligned}
\mathbf{E}_{r'}\left[Z\right] &= \mu' \\
\mathbf{E}_{r'}\left[\phi\right] &= \frac{a'}{a'+b'} \\
\mathbf{E}_{r'}\left[Z^2\right] &= \sigma'^2 \\
\mathbf{E}_{r'}\left[\phi^2\right] &= \frac{a'(a'+1)}{(a'+b')(a'+b'+1)}
\end{aligned}$$

The moments of the posterior $r_{Z,\phi}(Z,\phi|a,b,\mu,\sigma^2,d_n)$ can be computed using the same properties about the moments of the Gaussian and Beta distributions. We can derive them in the following way:

$$\begin{aligned}
\mathbf{E}_r\left[Z\right] =& K' \int_Z d\left(C_1 N(Z|m,s^2) + C_2 N(Z|\mu,\sigma^2) + C_3 N(Z|\mu,\sigma^2)\right) d(Z) = \\
=& \frac{1}{C_1+C_2+C_3}\left(C_1 m + C_2\mu + C_3\mu\right) \\
\mathbf{E}_r\left[\phi\right] =& K' \int_\phi \phi(C_1 Beta(\phi|a+1,b) + C_2 Beta(\phi|a,b+1) + \\
& C_3 Beta(\phi|a,b))d(\phi) = \\
=& \frac{1}{C_1+C_2+C_3}\left(C_1\frac{a+1}{a+b+1} + C_2\frac{a}{a+b+1} + C_3\frac{a}{a+b}\right)
\end{aligned}$$

$$\begin{aligned}
\mathbf{E}_r\left[Z^2\right] =& K' \int_Z d^2\left(C_1 N(Z|m,s^2) + C_2 N(Z|\mu,\sigma^2) + C_3 N(Z|\mu,\sigma^2)\right) d(Z) = \\
=& \frac{1}{C_1+C_2+C_3}\left(C_1(m^2+s^2) + C_2(\mu^2+\sigma^2) + C_3(\mu^2+\sigma^2)\right)
\end{aligned}$$

$$\mathbf{E}_r\left[\phi^2\right] = K' \int_\phi \phi^2 (C_1 Beta(\phi|a+1,b) + C_2 Beta(\phi|a,b+1) + C_3 Beta(\phi|a,b)) d(\phi) =$$
$$= \frac{1}{C_1+C_2+C_3}\left(C_1 \frac{(a+1)(a+2)}{(a+b+1)(a+b+2)} + C_2 \frac{a(a+1)}{(a+b+1)(a+b+2)} + C_3 \frac{a(a+1)}{(a+b)(a+b+1)}\right)$$

### A.1.2.3 Update Equations

The matched moments give us four equations for the parameters $a'$, $b'$, $\mu'$ and $\sigma'^2$. With a couple of simplifications we compute the final update equations for a depth measurement.

We can further notice that, when we matched the moments of the real posterior probability and the Gaussian-Beta mixture approximation, we implicitly calculated parameter $\beta_n$. At the same time we assumed that the parameters $\gamma_i$ are constant, but they are actually not. They should be computed from the true posterior as:

$$\gamma_i = \mathbf{E}_{\boldsymbol{c}}\left[c_i\right] = 0 \cdot r_{c_i}(c_i = 0) + 1 \cdot r_{c_i}(c_i = 1) = r_{c_i}(c_i)$$

As these are unknown, we use the posterior probability $q_{c_i}(c_i)$ as an approximation for our parameter $\gamma_i$:

$$\gamma_i \approx q_{c_i}(c_i)$$

This gives us the following updates for a depth measurement

$$\begin{aligned}
\mu' &= \mathbf{E}_r\left[Z\right] \\
\sigma'^2 &= \mathbf{E}_r\left[Z^2\right] - \mathbf{E}_r\left[Z\right]^2 \\
a'_{obs} &= \frac{\mathbf{E}_r\left[\phi\right]\left(\mathbf{E}_r\left[\phi\right] - \mathbf{E}_r\left[\phi^2\right]\right)}{\mathbf{E}_r\left[\phi^2\right] - \mathbf{E}_r\left[\phi\right]^2} \\
b'_{obs} &= \frac{\left(\mathbf{E}_r\left[\phi\right] - \mathbf{E}_r\left[\phi^2\right]\right)\left(1 - \mathbf{E}_r\left[\phi\right]\right)}{\mathbf{E}_r\left[\phi^2\right] - \mathbf{E}_r\left[\phi\right]^2}
\end{aligned}$$

For the semantic measurement we derived the parameters of the prior Beta distribution as

$$a'_{sem} = \sum_{i=1}^{K} q_{c_i}(c_i) A_i$$
$$b'_{sem} = \sum_{i=1}^{K} q_{c_i}(c_i) B_i$$

that we continuously update after each new measurement, based on the current posterior estimate $q_{c_i}$.