Nonparametric Directional Perception

by

Julian Straub

Diplom-Ingenieur Univ. in Electrical Engineering and Information Technology,
Technische Universität München, 2012
Master of Science in Electrical and Computer Engineering, Georgia Institute of
Technology, 2012

Submitted to the Department of Electrical Engineering and Computer Science in
partial fulfillment of the requirements for the degree of

Doctor of Philosophy
in
Electrical Engineering and Computer Science
at the Massachusetts Institute of Technology

June 2017

Signature of Author: _____
Department of Electrical Engineering and Computer Science
May 17, 2017

Certified by: _____
John W. Fisher III
Senior Research Scientist, Electrical Engineering and Computer Science
Thesis Supervisor

Certified by: _____
John J. Leonard
Samuel C. Collins Professor of Mechanical and Ocean Engineering
Thesis Co-Supervisor

Accepted by: _____
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair of the Committee on Graduate Students

Nonparametric Directional Perception

by Julian Straub

Submitted to the Department of Electrical Engineering
and Computer Science on May 17, 2017
in Partial Fulfillment of the Requirements for the Degree
of Doctor of Philosophy in Electrical Engineering and Computer Science

Abstract

Artificial perception systems, like autonomous cars and augmented reality headsets, rely on dense 3D sensing technology such as RGB-D cameras and LiDAR scanners. Due to the structural simplicity of man-made environments, understanding and leveraging not only the 3D data but also the local orientations of the constituent surfaces, has huge potential. From an indoor scene to large-scale urban environments, a large fraction of the surfaces can be described by just a few planes with even fewer different normal directions. This sparsity is evident in the surface normal distributions, which exhibit a small number of concentrated clusters. In this work, I draw a rigorous connection between surface normal distributions and 3D structure, and explore this connection in light of different environmental assumptions to further 3D perception. Specifically, I propose the concepts of the Manhattan Frame and the unconstrained directional segmentation. These capture, in the space of surface normals, scenes composed of multiple Manhattan Worlds and more general Stata Center Worlds, in which the orthogonality assumption of the Manhattan World is not applicable. This exploration is theoretically founded in Bayesian nonparametric models, which capture two key properties of the 3D sensing process of an artificial perception system: (1) the inherent sequential nature of data acquisition and (2) that the required model complexity grows with the amount of observed data. Herein, I derive inference algorithms for directional clustering and segmentation which inherently exploit and respect these properties. The fundamental insights gleaned from the connection between surface normal distributions and 3D structure lead to practical advances in scene segmentation, drift-free rotation estimation, global point cloud registration and real-time direction-aware 3D reconstruction to aid artificial perception systems.

Thesis Supervisor: John W. Fisher III
Title: Senior Research Scientist, Electrical Engineering and Computer Science
Thesis Co-Supervisor: John J. Leonard
Title: Samuel C. Collins Professor of Mechanical and Ocean Engineering

Acknowledgments

This research would not have been possible and definitely not as enjoyable without the support, camaraderie and friendships with various people in my life at MIT and beyond.

In relation to this amazing research journey, I want to thank my PhD committee John W. Fisher III, John Leonard, Leslie Kaelbling and Frank Dellaert. Their guidance and constructive feedback have shaped this research and enriched the breadth and potential impact of it. Specifically, I am deeply grateful to John W. Fisher III for his guidance, the freedom and support to pursue this research, and many fun and thought provoking white-board discussions. I want to thank John Leonard for his wise counsel, for all the encouragement and for serving as a reminder to to keep grounded in real-world applications. I am grateful for the honest and direct feedback from Leslie Kaelbling in our first meeting, when she asked why a roboticist should care about my work. This single question caused me to think hard about the potential impact of this research and spawned many ideas some of which are written up in Chapter 5. I am very thankful to Frank Dellaert, my long-term mentor of seven years, with whom I started this amazing research journey during my time at Georgia Tech. Thank you for believing in my potential when I had little to show and for your guidance and honest feedback along the way. I still remember your encouragement to see Michael Jordan's talk about Bayesian nonparametric models at Georgia Tech. This was one of the deciding factors in picking up research in that field.

Much of my research would not have happened in this form and so swiftly without my collaborators Trevor Campbell, Randi Cabezas, Jason Chang, Sue Zheng, Nishchal Bhandari, Oren Freifeld, Guy Rosman, and Jonathan How. Trevor von de Papiermolen, thank you for all the coffee- and green-tea-induced brain-storming, white boarding, function bounding, and paper smithing. It was and is a pure joy to work with you. Thank you also for introducing me to picking up and putting down heavy things and for always being ready for a joke and some good laughter. Randi, thanks for always being there to help. Be it with technical difficulties, research blues, having late night coffee, pointing out all the inconsistencies (literally all of them), or going through last-minute corrections after an all-nighter. I am inspired by your generosity in terms of spending your own time to help and just giving random gifts (I loved all the apples). Thank you Sue for your thoughtful comments, fixing thought-process and math bugs and always being there to listen. I throughly enjoyed our conversations and making you laugh every once in a while. Thanks Nishchal for exploring 3D perception on real robots with me. I enjoyed working with you and seeing our ideas and discussions turn into reality. Jason thank you for teaching me all about Dirichlet processes and inference methods for them. Similarly, thank you Oren for teaching me all about Riemannian

manifolds and Lie groups in the very beginning. Thank you Guy for helping with and teaching me about surface normal denoising. The mentoring by the three of you undoubtedly helped kickstart my PhD research. I am deeply thankful that all our paths crossed.

I am very grateful for having been part of the Sensing Learning and Inference and the Marine Robotics group. Thank you Christopher Dean, David Hayden, Randi Cabezas, Sue Zheng, Vadim Smolyakov, Jason Chang, Zoran Duric, Georgios Papachristoudis, Oren Freifeld, Guy Rosman, Jason Pacheco and Rujian Chen for giving feedback on various ideas, papers, posters and presentations. You helped shape, fix and improve my work, personality and this thesis. Beyond that I am grateful for all the camaraderie, support and laughter we shared. Thanks Chris for ordering all this vegan food, always being ready for a joke and for fixing my English like no other. Thanks David for encouraging reflection and sharing and discussing insights from your or my latest books. Thank you David Rosen, Sudeep Pillai, Michael Kaess, Liam Paull, Ross Finman, Dehan Fourie, Peter K.T. Yu for giving feedback from the robotics perceptive on various ideas and talks. You helped me broaden my horizon, and to keep grounded in practical application. I throughly enjoyed hearing about your various robotics perception problems in group meeting and over lunch or coffee.

Besides my PhD thesis research, I am very grateful for the opportunity to develop a 3D perception system to help breast-cancer patients by detecting lymphedema with Professor Regina Barzilay. Thanks to my undergraduate collaborators Eric Chen, Fernando Yordan, Hayley Song, Erik Nguyen and Gabriel Ginorio for joining me on this endeavor and helping to develop this 3D perception system. Thank you for trusting in my guidance and for bearing with me as I am figuring out how to be a good mentor. Thank you Regina for supporting this project, trusting my judgement, sharing your experience, and all the insightful discussions over tea about research and beyond.

And last but not least I am deeply thankful to my Mom and Dad for helping me find and then supporting my passion in electronics, robots and perception.

Contents

List of Figures

List of Algorithms

List of Tables

Notation

n	directional data, i.e. surface normal
x	Euclidean data
q	unit Quaternion rotation
${}^{B}R_{A}$	Rotation matrix $\in \mathbb{SO}(3)$ from coordinate system A to B
ω	Rotation expressed $\in \mathfrak{so}(3)$
w, θ	Rotation expressed in axis w and angle θ
${}^{B}T_{A}$	Transformation matrix $\in \mathbb{SE}(3)$ from coordinate system A to B
z	Labels
$\mathbb{1}_c$	Indicator function returning 1 of c is true and 0 otherwise
$\mathcal{I}_k$	Given a set of labels $\mathbf{z}$, $\mathcal{I}_k = \{i : z_i = k\}$ is the set of indices with label k
N_k	$N_k \triangleq \lvert\mathcal{I}_k\rvert = \sum_{i=1}^{N} \mathbb{1}_{z_i=k}$ counts the number of labels with value k
$\lceil x \rceil$	normalize $x \in \mathbb{R}^D$ to unit length, i.e. to lie on $\mathbb{S}^{D-1}$.

Chapter 1

Introduction and Problem Definition

At a fundamental level our environment consists of open space and textured surfaces with physical properties. Since this 3D structure sets the stage for the actions of biological and artificial agents, a capable perception system is key to any intelligent agent. Beyond textured surfaces, we humans associate meaning with scene parts that are based on geometrical and physical understanding of space, prior personal experiences, and otherwise acquired knowledge. Geometrical and physical understanding of space captures priors on how spaces are usually organized and shapes our expectations of which spatial configurations are even feasible. We associate meaning to surfaces such as "chair", "office" or "tea mug", probably by matching their shape, texture and context against our prior knowledge and experiences. Without our higher-level mental processes to segment and attach meaning to the 3D structure surrounding us, our mental processes would indeed be limited. Even the very basic task of avoiding injury in a cluttered environment necessitates the mental concept of obstacle versus traversable space.

In an effort to enable autonomous agents and artificial perception systems to interact and understand their surroundings, the main goal of decades of research effort in computer vision and robotics has been to obtain geometric computer representations of the environment. These representations have been mostly concerned with obtaining sparse [56, 67, 138, 172] and more recently dense 3D representations [110, 135, 177, 178, 248] of an environment.

Adding higher-level information about the relationships of scene parts and their semantic or categorical meaning into such structure representations is the natural next step in developing artificial perception systems that can perform at human levels and beyond. Scene-part relationships such as that chairs are usually close to tables, and meaning such as that a chair can be used to sit down are likely essential to higher-level scene understanding and truly intelligent autonomous agents. While human level capabilities are slowly being achieved in specialized and curated datasets (largely thanks to deep convolutional networks), general-purpose scene understanding and perception systems remains elusive. This is especially true in the realm of real-time 3D perception from streams of camera and depth sensor data as encountered in applications such as autonomous robotics and cars, and augmented reality.

An important question to ask in this context is: to what end do semantic segmentations actually facilitate operation of an artificial perception system? What is meaningful

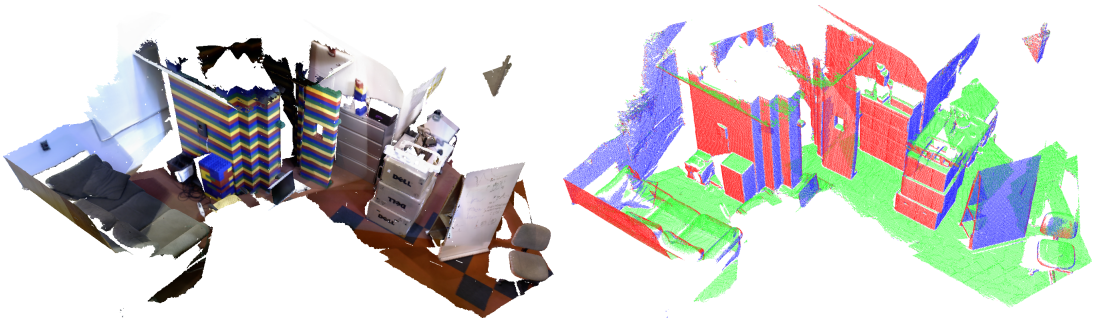


Figure 1.1: Scene segmentation of a large indoor reconstruction (reconstruction via Kintinuous [246]). For an autonomous agent or perception system, the segmentation is immediately useful: all green areas are either traversable or locations that could harbor objects the robot might be looking for. Blue and red areas should not be bumped into but might for example contain door signs.

to a human, such as good places to have vegan food, might be irrelevant to a machine (unless it is needed for interaction with the human). On the other hand, even simple (not necessarily semantic) scene-part relationships can help downstream inference and planning systems. Consider for example the segmentation of a 3D reconstruction shown in Fig. 1.1. The segmentation of the environment can be used by an autonomous agent to determine traversable areas, or where to look for door signs and other objects. In other words it can serve as a basis for higher-level, task-specific scene understanding.

More formally, the aforementioned problem of attaching meaning to the geometric perception of the surroundings is to infer not only a map m of the world that captures the textured geometric structure of the environment and the trajectory T of the perception system inside this map, but also a segmentation z of the inferred map given a batch or a stream of data x. The segmentation z assigns categorical information to each part of the world map. Because categorical information may be associated with semantic concepts, this problem is often termed semantic simultaneous localization and mapping (SLAM) [16, 72, 98, 143, 209, 211]. Semantic implies that z carries some meaning in relation to a human observer. The more general problem of categorical SLAM does not necessarily ask that the segmentation be meaningful to a human for reasons outlined before. Adopting, for the sake of argumentation, the Bayesian approach, the inference can be cast as reasoning about the posterior probability

$$p(m, T, z \mid x) \qquad \text{“categorical or semantic SLAM”} . \tag{1.1}$$

While the segmentation variable z has importance in itself it could also be thought of as an auxiliary variable which is explicitly instantiated for two purposes: (1) to enable more efficient and higher quality inference and (2) for the benefit of higher-level scene understanding and decision making systems as mentioned beforehand.

Most previous and some current 3D reconstruction systems purely aim to infer the 3D structure of the environment as well as the trajectory of the perception system.

More formally, they seek to characterize the joint belief of map m and perception system trajectory T given a set of observations x:

$$p(m, T \mid x) = \int_Z p(m, T, z \mid x)\, \mathrm{d}z \qquad \text{“SLAM”} . \tag{1.2}$$

As indicated in the equation above, this can be understood as aiming to directly infer the marginal distribution of the map and trajectory given observed data. Often for efficiency reasons only the most likely map $m^\star$ and trajectory estimate $T^\star$ is sought. This is usually performed in an alternating fashion by (1) finding the most likely map configuration $m^\star$ given data x and the current best estimate of the trajectory T and then (2) updating the most likely trajectory $T^\star$ given map and data:

$$m^\star = \arg\max_m p(m, \mid T^\star, x) \qquad \text{“mapping”} \tag{1.3}$$

$$T^\star = \arg\max_T p(T \mid m^\star, x) \qquad \text{“localization”} . \tag{1.4}$$

In isolation the first equation is called mapping, and the second equation is called localization. This alternating optimization hints at the chicken and egg problem faced in the simultaneous localization and mapping problem (SLAM) [148]: the trajectory of the perception system is needed to update the map but a map is also needed to update the belief of the perception system trajectory.

The problem of obtaining a meaningful segmentation of an environment has been studied extensively for 3D meshes and in computer vision for 2D projections of environments (i.e. images). In fact, while in some instances inference is performed over the underlying world map m before segmentation, most of the work on image and mesh segmentation and categorization including recent advances in deep learning [88, 91, 140, 218] may be understood as reasoning about the distribution of the segmentation marginalized over the map m and trajectory T:

$$p(z \mid x) = \int_m \int_T p(m, T, z \mid x)\, \mathrm{d}T\, \mathrm{d}m \qquad \text{“segmentation”} . \tag{1.5}$$

In this framework, deep learning systems could be understood as learning a function of the data, $f(x)$, that is proportional to the marginal segmentation posterior: $f(x) \propto p(z \mid x)$. They use a very flexible function approximator, namely convolutional neural networks, and large corpora of training data. In these systems the aforementioned marginalization is usually achieved by selecting a training dataset which comprises a range of viewpoints (T) and a range of geometrically differing instantiations (m) of the same category. It remains an open problem to see if this is a feasible approach or if instantiating map and trajectory explicitly, as auxiliary variables, leads to better scene understanding. There are indications that latter is indeed beneficial. An example is face recognition, where explicitly inferring face geometry and view-point improves recognition rates [230].

Returning to categorical SLAM of Eq. (1.1), we can extend the inference approach of SLAM to include the segmentation z. Since one might not only care about the most likely configurations of map, trajectory and segmentation, the alternating approach would iteratively reason[1] about the following posterior distributions:

$$p(m \mid T, z, x) \quad \text{“semantic mapping”} \tag{1.6}$$
$$p(T \mid m, z, x) \quad \text{“semantic localization”} \tag{1.7}$$
$$p(z \mid m, T, x) \quad \text{“scene understanding” .} \tag{1.8}$$

From these conditional distributions it can be gleaned that explicitly instantiating the scene segmentation z can be utilized to help mapping and localization because these tasks now have access to the output of a some form of scene understanding captured by z. Scene understanding in turn not only has access to all data but also to the consistent aggregation of 3D information into a world map and the perception system trajectory within the map. Note that in this setup any scene understanding may be used that can be expressed in terms of a segmentation of the scene.

Mapping as defined in Eq. (1.6) can take into account not only the trajectory T of the perception system but also the meaning or category of the surfaces as captured in the inferred scene segmentation z. This could be useful for defining segment-dependent operations such as category-dependent regularization of the reconstruction, scene part compression, or sub-mapping. We could call this categorical or semantic mapping. Indeed in Sec. 5.2 a geometric segmentation of the scene will be used to aid mapping and improve the accuracy of the system.

In a similar fashion localization can take into account the category of surfaces the system observed at certain timesteps to, for example, adapt the observation model accordingly. Scenarios such as taking into account the intent of the wearer of an augmented reality headset are captured under this model as well. We will see an instantiation of this in Sec. 5.2 where directional scene understanding can be leveraged to achieve efficient camera pose tracking. Reasoning about the trajectory posterior includes the essential problem of loop closure: the ability to determine switches between different modes of the trajectory posterior. Higher-level scene understanding and scene segmentation has the potential to aid and robustify current loop closure techniques. Current methods for loop closure involve extracting a high dimensional descriptor for key view-points of the scene [6, 89] to allow matching a current view to some known previous pose. This mechanism can be understood as a (non-semantic) segmentation or annotation of the map according to some feature extraction function.

Scene understanding, i.e. reasoning about the posterior of the segmentation z of an environment as defined in Eq. (1.8), can draw not only on the raw data x but also on the consistent integration of 3D information into a world map m and the perception system's trajectory T in it. Therefore the segmentation z could capture notions such as points of

[1] Many inference algorithms such as Gibbs sampling (see Sec. 2.2.2) and expectation maximization (EM) indeed subscribe to this alternating approach.

interest, inferred from the trajectory's viewing directions, spoken annotations obtained from sound data contained in data x, or geometric information inferred from the map such as which parts of the map are traversable, i.e. belong to the floor, and which parts are obstacles. In this context, I claim that unless an artificial perception system is able to utilize the segmentation of the environment it is not really "understanding" the environment but merely recalling and reproducing concepts without understanding their implications. Much like we can learn to say words in a new language without having to understand their meaning. Often segmentations are inferred but not at all or only partially used (see Sec. 1.1.2 for a full review). That is, while a scene segmentation is inferred, the conditional dependence of map and trajectory on the segmentation implied by Eq. (1.6) and (1.7) is ignored. There are only a handful of systems that fully integrate the higher-level concepts such as a scene segmentation into the SLAM process [29, 35, 41, 129, 158, 186, 209] such that mapping as well as trajectory estimation actually utilize the inferred segmentation. Falling into this category we introduce the first semi-dense nonparametric direction-aware SLAM system in Sec. 5.2.

It is essential for scene understanding, i.e. reasoning about Eq. (1.8), that the number of categories or semantic concepts expressed by z be flexible and adaptive to the observations of the perception system. There are many approaches to selecting a model of the right complexity. In this thesis we rely on Bayesian nonparametric models (see Sec. 2.4) since they offer an elegant way of making model selection part of the generative model and therefore the inference procedure. The Dirichlet process, used extensively in this thesis, explicitly models the sequential arrival of data from a potentially infinite number of different categories. The Bayesian approach also has the advantage of allowing explicit reasoning about uncertainty, a large body of theoretical guarantees for inference algorithms and has proved to be a reliable foundation for reasoning about the real-world quantities in applications ranging from autonomous robots to bio informatics.

Since the segmentation could attach a wide range of categorical or semantic information to the world, it is important to understand the aim of adding the auxiliary segmentation variable z. Here we focus on geometric segmentations of the world map m as a first step to higher-level and perhaps more human-centric scene segmentations. Similar to face recognition where best results are achieved after first solving for the geometry and aligning all faces into a canonical view for the convolutional neural network [230], I contest that solving for geometry and a better spatial awareness is crucial for higher-level inference and perception. For example, higher-level inference about scene part relations are routinely using spatial cues to orient 3D scenes into a common frame of reference to aid and improve scene understanding [96, 173].

Due to the structural simplicity of man-made environments, understanding and leveraging local orientations of the constituent surfaces, gives more direct access to some geometric properties of the environment. From an indoor scene to large-scale urban environments, a major fraction of surfaces can be described by just a few planes with even fewer different surface normal directions. This sparsity is evident in the surface normal distributions, which exhibit a small number of characteristic, concen-

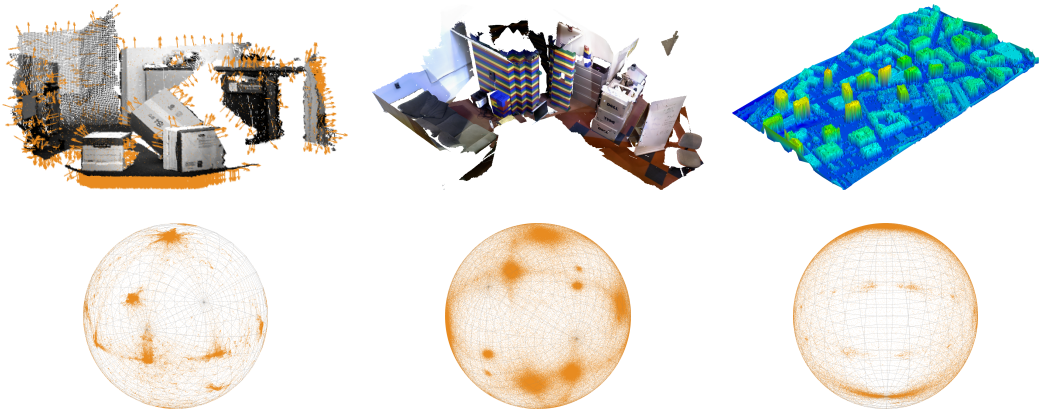


Figure 1.2: From a single view-point to large-scale urban scenes, the surface normal distributions are low entropy and exhibit similar characteristic patterns. This research seeks to characterize and utilize those patterns to further scene understanding and environment perception.

trated clusters as displayed in Fig. 1.2. For example, the segmentation in Fig. 1.1 was inferred solely based on the surface normal distribution of the scene. I term this a directional segmentation of the scene since surface areas in the same category share the same surface normal direction.

In this work I draw a rigorous connection between surface normal distributions and 3D structure, and explore this connection in light of different environmental assumptions. This leads us to propose several probabilistic scene models that facilitate directional scene segmentation and understanding. Hence, Chapters 3 and 4 essentially describe reasoning about the posterior of the scene segmentation (Eq. (1.8)) for different scene models expressed in terms of surface normal distributions. In Chapter 5 we first show how embedding a directional scene segmentation can be leveraged in the common problem of global point cloud alignment, thus reasoning about scene segmentation (Eq. (1.8)) and camera location (Eq. (1.7)). In the second part of the chapter we introduce the first semi-dense nonparametric direction-aware SLAM system that jointly reasons about the full categorical SLAM posterior (Eq. (1.1)) using the aforementioned alternating inference approach. These contributions are summarized in Sec. 1.2 after reviewing various related and prior work on geometric scene priors in the next section. We leave the detailed review of related work of the individual contributions to the respective chapters.

■ 1.1 Related Work

In the following we first explore the gamut of assumptions about scenes commonly employed in related work before focusing on the use of such assumptions in computer

vision and 3D reconstruction systems. Most approaches focus on the segmentation or scene understanding task given known 3D structure, camera poses and RGB images (or any subset thereof). These are reviewed in the next section. Some related work goes a step further and aims to jointly infer segmentation and 3D structure and/or the trajectory of the perception system. I will touch on such joint approaches in Sec. 1.1.1 with a focus on which assumptions they make and expand on how they perform joint reasoning in Sec. 1.1.2. Finally, in Sec. 1.1.3 I review related work that reason about semantic segmentations and other semantic concepts.

■ 1.1.1 Geometric Scene Segmentation Priors

The different assumptions made in the literature about the geometry of the environment can be categorized in terms of their expressiveness as depicted in Fig. 1.3. The assumptions range from mostly unrestricted representations such as point clouds, meshes and volumetric level-sets [52], which can in the limit represent any surface exactly, to the rather strict Manhattan World assumption as indicated by the inclusion diagram. The herein proposed Mixture of Manhattan Frames (MMF) assumption subsumes the Atlanta World (AW) which in turn subsumes the Manhattan World (MW) assumption. The MMF provides a directional segmentation under the orthogonality constraints imposed by the Manhattan World assumption. Relaxing the orthogonality constraints completely, we arrive at what we term the Stata Center World (SCW) (see Fig. 1.6 for a depiction of the Ray and Maria Stata Center). It captures only the directional composition of a scene. Planar scene representations differ from the Stata Center World in that different planes with the same orientation are separated in space.

These different assumptions about scenes can be observed directly in the 3D structure or indirectly in the projection of the 3D structure into a camera [102]. Models and inference algorithms based on the former utilize 3D representations such as meshes, point clouds and derived data such as surface normals. Intersections of planes in 3D are lines which can be observed as lines in the image space. A vanishing point (VP) is the intersection of multiple such image-space lines where the lines in 3D are all parallel to each other. Models built on VPs usually use image gradient orientations directly or indirectly via line segment extraction. Specifically, the Manhattan World is manifested as orthogonally-coupled VPs (OVPs) in the image space and a Manhattan Frame in the surface-normal space. Multiple Manhattan Worlds cause multiple orthogonal VPs and Manhattan Frames. The Stata Center World can be observed via independent VPs in the image or independent surface normal clusters.

Scene priors observed in the 2D image-space There is a vast literature on VP estimation from RGB images. The goals for VP estimation range from single-image scene parsing [18] and 3D reconstruction [57, 109, 146, 153], VP direction estimation for rotation estimation with respect to man-made environments [7, 20, 49, 139, 141] to VP direction tracking over time to estimate camera rotation and scene structure [29, 79, 169, 213].

While early VP extraction algorithms relied on image gradients [49, 214], most modern algorithms operate on line segments extracted from the image. This has been found

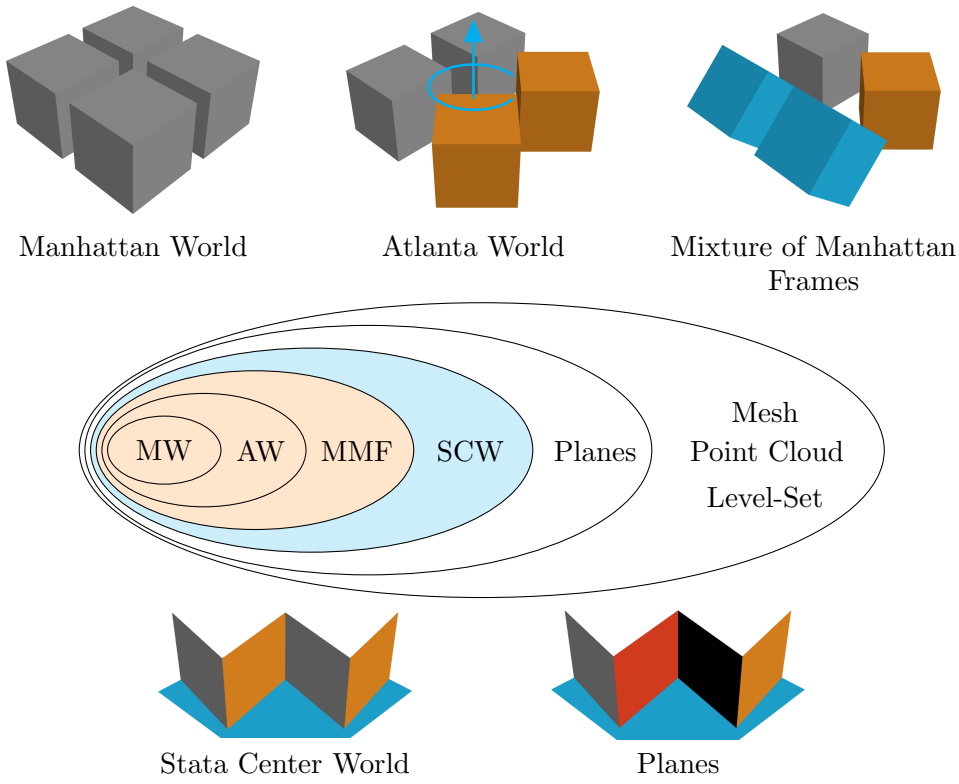


Figure 1.3: Depiction of the most common geometric scene priors and representations. The level of generality and expressiveness is indicated by the taxonomy. In Chapter 3 the Manhattan Frame and the MMF model are introduced (shaded orange). Chapter 4 contains two directional models to capture the Stata Center World in surface normal space (shaded blue). Chapter 5 introduces a plane-based 3D reconstruction algorithm that jointly infers and utilizes a Stata Center World scene segmentation.

to yield superior direction estimation results over dense image-gradient approaches [59]. Generally, VPs are extracted by intersecting lines in the image. These intersections are often found after mapping lines to the unit sphere [19, 48, 141], or into other accumulator spaces [149]. Introduced in [18], horizon estimation has emerged as a benchmark for VP estimation algorithms [252, 255].

Many VP extraction algorithms rely on the Manhattan World assumption [20, 29, 49, 79, 139, 157, 203, 252] which is manifested as three OVPs. Incorporating the Manhattan World assumption into the VP estimation algorithms not only increases estimation accuracy (if the Manhattan World assumption holds) [149] but also allows estimation of the focal length of the camera [40, 47, 139, 149, 203, 252], and rejection of spurious VP detections. Another avenue of research uses the Manhattan World assumption for single-image 3D reconstruction [57, 109, 146, 153]. The inferred Manhattan World and associations of lines to Manhattan World axes combined with geometric reasoning are used to reconstruct the 3D scene in [57, 146]. Hedau et al. [109] use a Manhattan World prior to iteratively infer the 3D room layout and segment out clutter in the room. Liu et al. [153] use a floor plan in conjunction with a set of monocular images to reconstruct whole apartments.

The AW model of Schindler et al. [214] assumes that the world is composed of multiple Manhattan Worlds sharing the same z-axis (which is assumed to be known). This facilitates inference from RGB images as only a single angle per MW has to be estimated as opposed to a full 3D rotation. The approach by Antunes et al. [9] infers the full MMF from RGB images. Relaxing the assumptions about the scene, VPs can be extracted independently [7, 18, 48, 59, 141, 149, 169, 231, 255] akin to the SCW assumption.

Scene priors in 3D representations There are many approaches that rely purely on 3D representations of surfaces and scenes. Assumptions such as the Manhattan World or SCW, are used to align scenes into a common frame of reference for scene segmentation and understanding [96, 173], and to regularize 3D reconstruction [170, 186] (more to that in the next section).

Similar to the image space, the Manhattan World assumption has been used most commonly [96, 173]. This is probably due to the fact that man-made environments tend to exhibit strong Manhattan World characteristics on a local scale, i.e. on the level of a single RGBD frame of a scene. In the application of Simultaneous Localization and Mapping (SLAM) [148], the Manhattan World assumption has been used to impose constraints on the inferred map [186]. Our original idea of the MF [225] has been adapted by Ghanem et al. [87] who propose a robust inference scheme for Manhattan Frame estimation and by Joo et al. [127] who use a branch-and-bound scheme to perform real-time globally optimal Manhattan Frame inference.

To the best of our knowledge the assumption of multiple Manhattan Worlds in the 3D data setting (as opposed to RGB 2D-images) has not been explored prior to our own work [225, 226] which is described in Chapter 3.

Similar to the Manhattan Frame and MMF model, the Stata Center World can be

inferred solely from surface-normal distributions. We explored this in two preliminary manuscripts [222, 224] and give a more detailed account in Chapter 4. Monszpart et al. [170] couple a local plane-based approach with global directional regularity constraints to regularize 3D reconstructions of man-made environments from point clouds. Gupta et al. [96] assume the only relevant direction for semantic scene segmentation is the direction of gravity to enable alignment of the ground plane across scenes. They propose a simple algorithm to segment the scene into the gravity and all other directions based on surface-normal observations. Triebel et al. [237] extract the main directions of planes in a scene using a hierarchical Expectation-Maximization (EM) approach. Using the Bayesian Information Criterion (BIC) they infer the number of main directions as well. Note, that the Manhattan Frame and MMF model could be inferred from the Stata Center World by grouping inferred directions into Manhattan Frames.

An alternative to the Manhattan World, MMF or Stata Center World model describes man-made structures by individual planes with no constraints on their relative normal directions. This assumption is widely used for scene segmentation [112] and understanding, to aid 3D reconstruction [129, 210, 219], and optical flow [200] computation. Triebel et al. [237] use hierarchical expectation maximization (EM) to jointly infer the main directions as well as a plane segmentation of a 3D scene. To overcome issues related to the sheer amount of data of dense meshes, Whelan et al. have proposed a planar simplification algorithm [249] which relies on plane segmentation. Surfel-based methods describe the environment as a set of localized planes each associated with a radius. Typically the radii are small and a large collection of surfels is used to describe an environment densely [133, 250]. Since surfels are assumed to be independent of each other, updates can be computed efficiently in parallel.

The orthogonality constraints in the Manhattan World or MMF models enable statistical pooling of measurements across different orientations. This means not only that fewer measurements (per plane) are needed to achieve the same amount of accuracy as without those constraints but also that reliable measurements from one or more directions help in handling cases where there are only few observations of other directions. Similarly, the Stata Center World encapsulates the notion of parallel planes which also provides a mechanism for statistical pooling across a scene as we will see in Sec. 5.2.

Joint 2D image and 3D structure based scene priors The connection between VPs in images and 3D Manhattan World structures has been used to infer dense 3D structure from sets of images by Furukawa et al. [82]. They employ a greedy algorithm for a single-MF extraction from normal estimates that works on a discretized sphere. Neverova et al. [176] integrate RGB images with associated depth data from a Kinect camera to obtain a 2.5D representation of indoor scenes under the Manhattan World assumption. Silberman et al. [173] infer the dominant Manhattan World using VPs extracted from the RGB image and surface normals computed from the depth image. They leverage the inferred Manhattan World rotation to align scenes into a common frame of reference for higher-level scene segmentation and reasoning.

Other shape-based segmentation algorithms Beyond global geometric scene priors that imply specific scene segmentations other approaches rely more on local shape features for segmentation.

One class of such approaches compute local feature vectors, train a feature-based classifier and then impose spatial label smoothness via a Markov Random field. Zhang et al. [257] follow this approach and show that using dense depth images leads to better segmentations than using sparse colored point clouds. Along the same lines, Rouhani et al. [204] partition an input mesh into superpixels, compute a feature vector for each vertex (geometric and texture) and then use a Markov random field to smooth the labeling produced by random forests.

Finman et al. [71] demonstrate geometry-feature-based online incremental scene segmentations of dense 3D scene. Their algorithm relies on the popular Felzenszwalb segmentation algorithm [69], which is based on a local neighborhood graph and relative geometric properties of neighbors. By local greedy segmentation choices the Felzenszwalb algorithm obtains controllable global segmentation properties.

■ 1.1.2 Geometric Scene Segmentation and Reconstruction

After reviewing the overarching literature on the different kinds of geometric scene priors in the previous section, we now focus on how scene priors are used in different 3D perception systems. On a high level there are three different classes of approaches in how scene priors or assumptions are used: (1) for trajectory estimation only, (2) for mapping or 3D reconstruction only, or (3) for trajectory and mapping jointly. The previous section includes all related work that purely reasons about scene priors without using it in either trajectory estimation or mapping.

Joint 3D mapping and segmentation systems We first turn to systems that jointly reason about the 3D structure and a segmentation of the scene. Triebel et al. [237] essentially use the Stata Center World assumption and extract the main directions of planes in a scene using a hierarchical Expectation-Maximization (EM) approach. Using the Bayesian Information Criterion (BIC) they infer the number of main directions as well. Their results show that the statistical pooling of plane orientation measurements for all planes with the same orientation improves the plane orientation estimate. Initiated by the seminal work of Delage et al. [57] there is a series of work [109, 146] on single-image Manhattan World segmentation and 3D reconstruction. Of these, Hedau et al. [109] add the capability to reason about clutter as an outlier to the Manhattan World model. This is extended to the multi-image case by Flint et al. [79] who utilize the Manhattan World assumption to perform 3D reconstruction and Manhattan World segmentation from a set of images from known poses. Häne et al. [98, 100] perform joint inference over scene segmentation and 3D structure and show how joint inference leads to better reconstructions and allows impressive, plausible hallucination of unobserved scene parts. Their approach relies on hand-crafted class-specific geometry priors that are trained on hand-labeled data. Observations are given in the form of geo-referenced depth and RGB images and the trajectory of the camera is assumed fixed and not part of the inference

process. Blaha et al. [26] building on the same approach, scale the inference to enable large city-scale semantic reconstructions. Monszpart et al. [170] couple a local plane-based approach with global directional regularity constraints to regularize plane-based 3D reconstructions of man-made environments from point clouds. In joint work lead by R. Cabezas [35] we define a joint distribution over geometry, camera poses and a scene segmentation. Elaborate sampling-based inference leads to a categorical scene segmentation that is informed by semantic observations from OpenStreetMap data. Results demonstrate that adding the scene segmentation into the inference procedure improves the quality of the map.

The ideas of Monszpart et al. [170] and Triebel et al. [237] are related to what we term the Stata Center World assumption and will be explored in Sec. 5.2 in the context of fully joint direction-aware SLAM.

Joint 3D pose or trajectory estimation and segmentation systems Some assumptions like the Manhattan World assumption lend themselves to estimation of the perception system trajectory. Specifically, there is a wealth of work on rotation estimation from vanishing points under the Manhattan World assumption[7, 20, 49, 139, 141]. The approach take in the Atlanta world model [214] also facilitates rotation estimation albeit in more complex environments following the Atlanta world. Chapter 3 introduces drift-free rotation estimation for Manhattan World environments based solely on the surface normal distribution. The Stata Center World models in Chapter 4 could also be used for rotation estimation by associating directional clusters over time using, for example, the proposed DDP-vMF-means algorithm. Using the Stata Center World model Zhou et al. [260] demonstrate drift-free rotation estimation with respect to a key-frame by associating and robustly aligning directional clusters obtained via mean-shift clustering. The clustering approach they employ includes keeping around unobserved clusters and is curiously similar to the DDP-vMF-means algorithm. Saurer et al. [213] improve the efficiency of a visual odometry system given a gravity direction estimate and assuming a Manhattan World. Liu et al. [153] jointly utilize a floor plan and a set of images of the environment to reason about the segmentation of the images into left, right, front, top and bottom walls and camera poses in the environment. Their algorithm yields a textured wall reconstruction given the room layout.

Section 5.1 demonstrates a global point cloud alignment system that relies on a directional and spatial segmentation of the environment for efficient operation.

Joint 3D SLAM and segmentation systems The following systems are among the few who jointly reason about 3D structure, geometric segmentation, and the trajectory of the perception system. Bosse et al. [29] utilize vanishing point detection and tracking of prominent lines in the environment in a SLAM system to jointly estimate a robot's trajectory and the 3D location of lines in the environment. Castle et al. [41] are among the visual SLAM systems to incorporate planar geometry into the camera tracking and reconstruction pipeline. They augment their visual SLAM system with the ability to detect known planar patches and use them to jointly improve mapping and localization.

Peasley et al. [186] use the Manhattan World assumption to impose constraints on the trajectory of a robot though an Manhattan World environment and show that this yields drift-free SLAM, eliminating the need for loop-closures, given that the assumption holds. They detect the dominant orientation using RANSAC on LiDAR scans and match it to one of the four possible Manhattan World directions (the system operates in 2D). Salas-Moreno et al. [209] integrate plane segmentation into the tracking and reconstruction pipeline of a dense surfel-based reconstruction system. Numeric results show that utilizing a plane segmentation of the environment leads to improved tracking accuracy. Kaess [129] explores a direct plane-based SLAM formulation wherein the map directly consists of infinite planes which are being jointly optimized with the camera pose. In a way this approach fuses map and segmentation into one entity. Ma et al. [158] demonstrate joint inference over a key-frame-based map and a plane segmentation of the environment. The joint formulation with soft plane-assignments reduces drift of the SLAM system.

Section 5.2 introduces the first directional-aware SLAM system that jointly reasons about the directional Stata Center World segmentation, camera trajectory and 3D structure of the environment.

■ 1.1.3 Beyond Geometric Scene Segmentation

Beyond geometric scene priors and segmentations several approaches have been proposed that aim at incorporating human-annotated semantic labels into the 3D reconstruction process. Before reviewing such joint reasoning systems I touch on systems that given a 3D reconstruction or in parallel to the reconstruction process infer a surface segmentation but do not use it further.

Semantics without further use Reasoning about semantic content of a scene is in itself an interesting problem since semantic scene understanding could be used for interactions with humans. Silberman et al. [173] are among the first to reason about the semantic segmentation and support relations of the environment using RGB and depth information captured via a Kinect camera. They orient all scenes into a canonical view using the Manhattan World assumption. Building on the NYU RGBD dataset by Silberman et al., Ren et al. [196] improve on the scene segmentation quality by improved feature extraction via the use of Kernel descriptors. Similarly, Gupta et al. [96] work on the NYU v2 depth data and improve on scene segmentation with up to 40 categories. Hermans et al. [111] run a dense SLAM system in parallel to a pixel-wise classifier in image space. They fuse the pixel-wise labels on the 3D map using a conditional random field. Dai et al. [54] use BundleFusion [55] to reconstruct 1513 scenes and crowd source dense semantic labels for each of them. ScanNet, their deep convolutional neuronal network, is trained on the annotated 3D reconstructions and can then be used to infer a semantic segmentation of voxelized 3D scenes.

Semantics inferred and used jointly Arguably the ability of a system to utilize the inferred (semantic) segmentation of an environment can be seen as some (rudimentary)

understanding of the meaning of the segmentation. Bao et al. [16, 17] jointly estimate object and region segmentation of a sparse point cloud in the structure from motion framework. Object detection is carried out in the image space. Fioraio et al. [72] jointly perform object detection, mapping and camera pose estimation in what they refer to as semantic bundle adjustment. In contrast to Bao et al. their algorithm works incrementally and not in batch, and the object detection is part of the SLAM system and not via an external object recognition algorithm. Xiao et al. [254] show how enforcing label consistency in a 3D reconstruction system leads to better 3D reconstruction by decreasing drift and correcting loop closures. Kundun et al. [143] jointly use dense image segmentation and the raw RGB image captured from a single camera to infer a semantic 3D reconstruction as well as the camera trajectory. Working in the realm of RGBD cameras as well, Kim et al. [137] use a voxel-based world representation and, for a given RGBD image, infer the 3D occupancy (i.e. the 3D structure) and the segmentation of the environment into semantic classes. Their model allows reasoning about occluded scene parts. Salas-Moreno et al. [211] are the first to demonstrate a SLAM system that utilizes dense 3D object models as beacons for camera tracking and map representation. At its core the system has an efficient way of detecting dense 3D mesh models of objects. The map is represented via a pose-graph of objects.

■ 1.2 Outline and Summary of Contributions

At a high level, Chapters 3 and 4 of this thesis introduce surface normal models for geometric scene segmentation. In Chapter 5 we leverage the nonparametric models developed in the previous chapters to improve the fundamental 3D perception problems of global point cloud alignment and 3D reconstruction.

In Chapter 3 we first investigate the properties of surface normal distributions of 3D environments that follow the Manhattan World assumption. That is the assumption that all planes in a scene are parallel to one of three major planes that are mutually orthogonal as displayed in Fig. 1.4. We propose and formalize the novel Manhattan Frame model which captures the Manhattan World assumption in the surface normal space. Based on this we introduce the first probabilistic Manhattan Frame model and derive efficient maximum a posteriori inference algorithms for two Manhattan Frame models with different surface normal noise assumptions. The resulting algorithm can be run in real-time, is robust to rapid camera motion and yields drift-free rotation estimation. This model and the results have been published in [221, 226].

The second contribution described in Chapter 3 is the generalization of the Manhattan World model to capture multiple Manhattan Worlds. This is necessary to describe man-made environments at a larger scale as shown in Fig. 1.5. While locally a scene or a part of a city may be well described by the Manhattan World assumption, the whole scene or the collection of neighborhoods in a city has to be modeled by multiple Manhattan Worlds. The contribution published in [225, 226] is to (1) outline this generalization, and to (2) draw the connection between 3D structure and surface normal

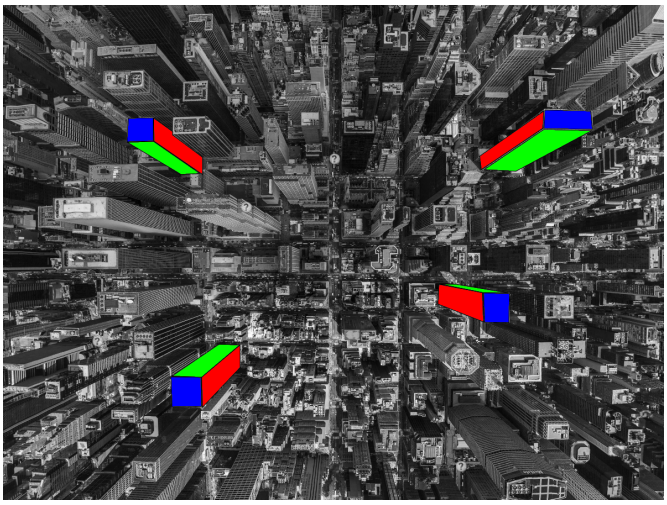


Figure 1.4: The Manhattan World model assumes that all planes in a scene are parallel to one of three mutually orthogonal planes. These planes are indicated in red, green, and blue.

space in the form of what we term the Mixture of Manhattan Frames (MMF). Again we define a probabilistic model describing the phenomenon of the MMF and show how to perform sampling-based inference such that the number of Manhattan Frames can be inferred as well as the Manhattan Frame rotations themselves. Additionally, we propose a fast alternative for MMF inference by maximum a posteriori inference. We show that the inferred MMF models lead to expressive directional scene segmentations that are consistent with a human annotator.

The contributions in Chapter 4 are two Bayesian nonparametric mixture models for surface normal modeling that can be applied to any-dimensional directional data. Both models, when applied to surface normal data, generalize the MMF model in that they relax the orthogonality constraints of the Manhattan World. These relaxed models capture environments we refer to as the Stata Center World (SCW) because of the "relaxed" building style of the Ray and Maria Stata Center of MIT as displayed in Fig. 1.6. While there are few orthogonal corners in the Stata Center, there are still parallel planes, which manifest as clusters in the surface normal space. First, in Sec. 4.3.2, we propose a Dirichlet process mixture model of Gaussians in dedicated tangent spaces to the sphere, the space of directional data. This model, published in [224], is able to adapt the number of clusters to fit the observed data, and can capture anisotropic directional clusters. We carefully exploit the geometry of the sphere to derive efficient sampling-based inference. Second, in Sec. 4.4.3, we derive a k-means-like directional clustering algorithm, called DP-vMF-means, as the low-variance limit of the posterior inference in the Dirichlet process von-Mises-Fisher (vMF) mixture model. Additionally, we propose the dependent Dirichlet process vMF mixture model to capture temporal

Figure 1.5: On a larger scale scenes can often be described by multiple Manhattan Frames as shown in this aerial view of Cambridge and Boston. Colored areas indicate areas following different Manhattan Worlds. Within each region planes are assumed to follow the Manhattan World assumption for some local orientation.

evolution of the mixture model. Again, by analyzing the low-variance limit, we derive a real-time capable k-means-like temporally consistent clustering algorithm we term DDP-vMF-means. Part of the contribution in this section is algorithmic development to exploit massively parallel processing on Graphics Processing Units to enable real-time operation on batches of 300k data-points at a frame-rate of 30 Hz. Related work on general directional clustering is reviewed in Sec. 4.1.

In Chapter 5 we leverage the Stata Center world surface normal model to improve global point cloud alignment and 3D reconstruction. The global point cloud alignment approach described in Sec. 5.1 can be seen as reasoning about the posterior over the camera pose given a segmentation and a map in Eq. (1.7). The contributions to global point cloud alignment include the use of Bayesian nonparametric mixture models to describe point and surface normal densities, the decomposition of the search into two 3D problems aided by surface normals, a novel discretization of the rotation space, and convergence and performance guarantees for the branch-and-bound search algorithm. In Sec. 5.2 we introduce the first nonparametric direction-aware SLAM system that reasons about the joint posterior over map, camera trajectory and directional geometric scene segmentation as posed in Eq. (1.1). Contributions comprise the first use of a Dirichlet-process-based scene prior in a real-time reconstruction system and the first use of a directional segmentation to improve camera tracking and map quality. The map formulation establishes a connection between the scene-wide directional segmentation and local surface properties for joint inference. The proposed inference architecture relies on sampling-based inference which allows quantification of uncertainty and opens

Figure 1.6: The Stata Center world relaxes the orthogonality constraints of the Manhattan World to describe environments that consist of sets of parallel planes with arbitrary orientation differences as illustrated in different colors in the figure via manual shading (not a result). The model is inspired by the very "relaxed" building style of the Ray and Maria Stata Center at MIT depicted above.

up the field of realtime 3D reconstruction to utilize complex Bayesian nonparametric models.

We conclude this thesis in Chapter 6 with a high level discussion of the results and propose future directions.

Code for all chapters can be found at `http://people.csail.mit.edu/jstraub/`.

■ 1.2.1 Acknowledgments

The research on the connection between the Manhattan World assumption and surface normal distributions described in Chapter 3 and published in [221, 225, 226] was conducted in collaboration with Oren Freifeld, Guy Rosman, Jason Chang, John J. Leonard, Nishchal Bhandari, and John W. Fisher III. The idea of the Manhattan Frame was formalized in a theoretically sound way with the help of Oren Freifeld. Guy Rosman helped with edge-preserving smoothing of surface normals, the robust extraction of surface normals from noisy LiDAR data and had the idea of depth camera focal length calibration using the MW assumption [225]. Jason Chang helped develop the Metropolis-Hastings-based inference for the MMF model. Nishchal Bhandari ran the Turtlebot experiment in Killian Court at MIT. Randi Cabezas kindly provided the LiDAR point cloud of Kendall Square.

The work described in Chapter 4 on more general directional segmentation under the Stata Center World was conducted in collaboration with Jason Chang, Trevor Campbell, Oren Freifeld, Jonathan P. How, John J. Leonard as well as John W. Fisher

III and published in [222, 224]. Specifically, Trevor Campbell's prior expertise with low-variance analysis helped make fast progress in deriving the DDP-vMF-means algorithm. He also formally proved the Laplace approximation for the sphere while I was busy working on the GPU-based DDP-vMF-means implementation. Jason Chang's input when deriving the DP-TGMM inference was crucial for the fast progress on the DP-TGMM project.

The final Chapter 5 describes research on global point cloud alignment that was developed jointly with Trevor Campbell, John W. Fisher III and John P. How. Trevor Campbell contributed most of the theoretical work (bounds, tessellation shrinkage and branch and bound convergence), as indicated by the equal contribution to the paper [223]. The nonparametric direction-aware SLAM system was developed jointly with John W. Fisher III.

Chapter 2

Background

This thesis draws on fundamentals in Bayesian statistics (Sec. 2.1), sampling-based inference (Sec. 2.2) and common distributions such as the categorical and the Gaussian distribution and their conjugate priors (Sec. 2.3). In addition, Bayesian nonparametric mixture models are reviewed in Sec. 2.4 as they form the backbone of flexible generative models that are used to describe aspects of the environment in Chapter 4. Because of its use throughout this work special attention is paid to the class of Dirichlet process mixture models. To derive efficient inference from such Bayesian nonparametric models one recent method is small variance asymptotic analysis which is reviewed in Sec. 2.5. Reasoning about surface normal distributions is a key contribution of this thesis and Sec. 2.6 therefore introduces different directional distributions used herein. In Sec. 2.7 we provide an introduction to rigid-body transformations and the three most prominent 3D rotation representations. A thorough understanding of these representations and their connections is important for the representation and inference of the Manhattan Frame as introduced in Chapter 3 and for the derivation of the branch and bound algorithm to optimally search over the space of rotations in Sec. 5.1. Finally, the connection between surface normals and the Gauss map as well as the computation of surface normals in practice is described in Sec. 2.8. Since all contributions in this thesis to some degree rely on surface normals extracted from point clouds, depth images, meshes or implicit shape representations, understanding the tradeoffs between the different techniques for extracting surface normals from various shape representations are essential and reviewed in Sec. 2.8 as well.

■ 2.1 Bayesian Inference

Bayes' rule is fundamental to modern probabilistic inference in fields ranging from robotics and computer vision to finance, politics, and biology applications. It states that the distribution over parameters θ given observed data x and hyper-parameters α, the so called *posterior* distribution, can be computed solely from the *likelihood* distribution $p(x \mid \theta)$ and the *prior* distribution over the parameters $p(\theta; \alpha)$:

$$p(\theta \mid x; \alpha) = \frac{p(x \mid \theta)p(\theta; \alpha)}{p(x; \alpha)} = \frac{p(x \mid \theta)p(\theta; \alpha)}{\int_\Theta p(x \mid \theta)p(\theta; \alpha)\,\mathrm{d}\theta}\,. \tag{2.1}$$

Bayes' rule is important since it characterizes how the belief about the parameters θ changes after observing some data x solely based on knowledge, via the likelihood, of how likely we are to observe the data and some prior distribution on the parameters. The utility of Bayes' formula is that is generally easier to define the likelihood and the prior distribution than it is to know the form of the posterior distribution $p(\theta \mid x; \alpha)$. Another way of looking at the prior and likelihood is that they captures how parameters θ are generated from a model and how data x are generated from the model parameterized by θ. In this thesis we often adopt a generative viewpoint when describing a probabilistic model. That is we describe how parameters θ and data x are generated under a proposed probabilistic model.

One important realization is, that since data x is given, the denominator of Eq. (2.1) is a constant. This constant is called the normalizer or partition function. This means the posterior is proportional to the likelihood times the prior:

$$p(\theta \mid x) \propto p(x \mid \theta)p(\theta; \alpha)\,. \tag{2.2}$$

For some applications such as maximum-a-posterior inference, it is sufficient to know the posterior up to proportionality, which eliminates the hard problem of computing or estimating the partition function.

Bayesian inference is the process of computing or estimating the posterior, that is to compute the distribution over parameters after observing some data given a prior distribution (i.e. a belief) about the parameter distribution. Broadly speaking there are the three main approaches to posterior inference: sampling-based inference, variational inference and expectation maximization. While we do utilize low variance analysis (a fourth, less common approach; see Sec. 2.5) in Chapter 4 we mostly focus on sampling-based inference for this thesis. See Sec. 2.2 for an introduction to sampling-based inference and for the motivation of its use. A broad overview over the different techniques may be found in [24].

In some applications it might be enough to get the most likely set of parameters on the support of the parameters Θ. This is called maximum a posteriori estimation and is formally defined as

$$\theta^\star = \arg\max_{\theta \in \Theta} p(\theta \mid x) = \arg\max_{\theta \in \Theta} p(x \mid \theta)p(\theta; \alpha)\,. \tag{2.3}$$

If the prior is uniform on the support of θ maximum a posteriori estimation reduces to maximum likelihood estimation

$$\theta^\star = \arg\max_{\theta \in \Theta} p(x \mid \theta)\,. \tag{2.4}$$

The difference to full posterior inference is that we only get a point estimate of the peak of the full posterior distribution. This for example means that we can get no information about the uncertainty of the parameter estimate $\theta^\star$, one major advantage of full posterior distribution inference.

In the following we briefly introduce and review key concepts of Bayesian modeling and inference.

Exchangeability Throughout this thesis we will often make the (silent) assumption that observations $x_1, x_2, \ldots, x_N$ in a batch of data are exchangeable. That is the order of observation does not matter, i.e. the probability of the set of data is invariant to reordering. Formally, for any permutation r_i of the indices of the data

$$p(x_1, x_2, \ldots, x_N) = p(x_{r_1}, x_{r_2}, \ldots, x_{r_N}) \,. \tag{2.5}$$

One of the simplest exchangeable distributions follows the form

$$p(\{x_i\}_{i=1}^N \mid \theta) = \prod_{i=1}^N p(x_i \mid \theta) p(\theta) \,. \tag{2.6}$$

Since the product of likelihood terms $p(x_i \mid \theta)$ can be arranged in any order this distribution is exchangeable. Indeed we will encounter variants of this model throughout the thesis.

Sufficient Statistics For some posterior distributions it is possible to "summarize" all information about a set of data $\{x_i\}_{i=1}^N$ in a statistic of the data $s_N = f(\{x_i\}_{i=1}^N)$. If the statistic captures all information such that

$$p(\theta \mid \{x_i\}_{i=1}^N) = p(\theta \mid s_N) \tag{2.7}$$

the statistic is called sufficient. Sufficient statistics are of great importance for deriving efficient inference algorithms: the amount of storage needed to capture all relevant information about a set of data can be compressed from $O(ND)$ to $O(\text{Poly}(D))$, where D is the dimension of the data and $\text{Poly}(D)$ is usually a low degree polynomial. Additionally, once the sufficient statistic is computed inference algorithms can save computation time since no further iterations over all data are necessary. Sufficient statistics are especially useful in CPU-GPU systems because of the limited memory bandwidth between CPU and GPU. By keeping the data to be processed on GPU memory and only transferring sufficient statistics to CPU, algorithms can be sped up significantly. This will be exploited in several places in this thesis.

Conjugate prior distributions Given a likelihood $p(x \mid \theta)$, which is usually known or defined because of the phenomenon to be modeled, different priors $p(\theta; \alpha)$ can be considered determining the form of the posterior $p(\theta \mid x; \alpha)$. It turns out that for a large class of likelihood distributions (e.g. the exponential family of distributions introduced in Sec. 2.3.1) there exists a conjugate prior. Under a conjugate prior distribution, the posterior has the same algebraic form as the prior with updated parameters α_N:

$$p(\theta \mid \mathbf{x}; \alpha) = p(\theta; \alpha_N) \,. \tag{2.8}$$

This means that inference is simplified because the form of the posterior is known exactly. Often the update to the hyper-parameters α can be written as a function of the sufficient statistics of the data x (e.g. in the case of exponential family distributions) allowing for efficient inference as discussed before.

Marginal data distribution The marginal data distribution is the distribution of the data x under the prior distribution marginalized over the parameters θ

$$p(x;\alpha) = \int_{\Theta} p(x \mid \theta)p(\theta;\alpha)\,\mathrm{d}\theta\,. \tag{2.9}$$

This captures the distribution of data x as predicted under all possible parameters θ distributed according to the prior $p(\theta;\alpha)$.

Posterior predictive distribution The posterior predictive likelihood is the likelihood of a new data $\tilde{x}$ given a set of observed data x marginalized over the parameters

$$p(\tilde{x} \mid x;\alpha) = \int_{\Theta} p(\tilde{x} \mid \theta)p(\theta \mid x;\alpha)\,\mathrm{d}\theta\,. \tag{2.10}$$

The posterior predictive distribution captures the predicted distribution of new data $\tilde{x}$ after observing data x. In practice, for example, we can use this distribution to evaluate how well an inferred model can explains the data in a held-out dataset.

■ 2.2 Sampling-based Inference

In contrast to parameter estimation which generally leads to optimization problems, Bayesian inference generally leads to the evaluation or estimation of expectations of functions f of the random variable

$$\mathbb{E}_p\left[f(x)\right] = \int_X f(x)p(x)\,\mathrm{d}x\,. \tag{2.11}$$

Sampling-based inference aims to draw samples from a given distribution $p(x)$ to allow the approximation of such expectations of functions of the random variable x via the strong law of large numbers which states that given N samples x_i from $p(x)$

$$\frac{1}{N}\sum_{i=1}^{N} f(x_i) \to \mathbb{E}_p\left[f(x)\right]\,. \tag{2.12}$$

This process is called Monte Carlo integration. Noting that expectations of random variables, probabilities and density normalizers can all be expressed as integrals over some function, it is clear that being able to obtain samples from arbitrary distributions is a very powerful tool.

For simple distributions like a uniform or a Gaussian distribution efficient methods exist to directly draw samples from the distribution. In general, for arbitrary and potentially quite complex $p(x)$, drawing samples is not straightforward especially if the normalization constant is unknown.

There are various algorithms for sampling arbitrary distributions. Importance sampling, rejection sampling, slice sampling and particle filtering are methods suitable for sampling low dimensional variables x, since they suffer from the curse of dimensionality

or are hard to extend to high dimensions. For distributions over a high dimensional state x such as the ones we are mostly investigating in this thesis, Markov chain Monte Carlo methods are better suited.

In [198] Robert and Casella give an excellent introduction and a comprehensive overview and discussion of sampling based inference methods. In the following we briefly review key algorithms used in this thesis.

2.2.1 The Metropolis-Hastings Algorithm

The Metropolis-Hastings algorithm [105] constructs an ergodic Markov chain over the state variables x such that the stationary distribution is $p(x)$. This means that in the limit of sampler iterations the samples x are guaranteed to be drawn from the desired distribution $p(x)$ (Sec. 7 [198]) Interestingly, such an ergodic Markov chain can be constructed from any proposal distribution $q(x' \mid x)$ known up to proportionality. In each iteration, the Metropolis-Hastings algorithm samples a proposal x' given the current state x from $q(x' \mid x)$ and accepts it with the Metropolis-Hastings acceptance probability:

$$P_a(x', x) = \min\left(1, \frac{p(x')q(x \mid x')}{p(x)q(x' \mid x)}\right) . \tag{2.13}$$

The Metropolis-Hastings algorithm is outlined in Alg. 1. From a practical standpoint one chooses the proposal distribution such that it is easy and efficient to sample x'.

1: Initialize $x = x_0$
2: **while** more samples desired **do**
3: Sample proposal $x' \sim q(x' \mid x)$
4: Compute $P_a(x', x) = \min\left\{1, \frac{p(x')q(x|x')}{p(x)q(x'|x)}\right\}$
5: Sample a uniformly at random between 0 and 1
6: **if** $a < P_a$ **then**
7: Accept proposal $x = x'$
8: **end if**
9: **end while**

Algorithm 1: Metropolis-Hastings algorithm.

To guarantee that $p(x)$ is indeed stationary distribution of the Markov chain established by the Metropolis-Hastings algorithm, the transition kernel $K(x', x)$ of the Markov Chain has to fulfill the detailed balance condition (Defn. 6.45. [198]):

$$K(x', x)p(x') = K(x, x')p(x) . \tag{2.14}$$

Detailed balance is established for any proposal distribution subject to the aforementioned constraints by construction of the Metropolis-Hastings algorithm as shown in Thrm. 7.2 [198]. To guarantee that the Markov Chain converges to the stationary

distribution, the Markov Chain has to be ergodic. A way of ensuring ergodicity is to construct the proposal distribution such that all parts of the support of $p(x)$ can be reached (see Lemma 7.6. [198]).

While in theory it takes infinite sample iterations to reach the stationary distribution, there are elaborate algorithms and techniques to asses approximate convergence to the stationary distribution (see Sec. 12 [198]). In practice it is often sufficient to just sample for some predetermined amount of time.

■ 2.2.2 Gibbs Sampling

Assume we can split the state variables into two sets x and y such that we can sample from the conditional distributions $p(y \mid x)$ and $p(x \mid y)$. As outlined in Alg. 2, the Gibbs sampling algorithm simply consists of alternating sampling from the two conditional distributions. Gibbs sampling can be understood as an instantiation of the Metropolis-

1: Initialize $x = x_0$ and $y = y_0$
2: **while** more samples desired **do**
3: Sample $x \sim p(x \mid y)$
4: Sample $y \sim p(y \mid x)$
5: **end while**

Algorithm 2: Gibbs-sampling algorithm.

Hastings algorithm where the proposal distributions are conditional distributions of $p(x, y)$ (see Sec. 10.2.2 [198]):

$$q(x', y' \mid x, y) = \delta_y(y')p(x' \mid y) \text{ (proposal for } x') \tag{2.15}$$
$$q(x', y' \mid x, y) = \delta_x(x')p(y' \mid x) \text{ (proposal for } y') \ . \tag{2.16}$$

This leads to an acceptance probability of 1 as can be verified for the proposal of x' (the same applies to the proposal of y'):

$$\frac{p(x', y')q(x, y \mid x', y')}{p(x, y)q(x', y' \mid x, y)} = \frac{p(x' \mid y)p(y)\delta_y(y')p(x \mid y)}{p(x \mid y)p(y)\delta_y(y')p(x' \mid y)} = 1 \, . \tag{2.17}$$

This is an important connection since it means that the guarantees of sampling from the target joint distribution $p(x, y)$ of the Metropolis-Hastings algorithm carry over to the Gibbs sampler.

Here we have examined the case of two sets of random variables, but the Gibbs algorithms extends to an arbitrary number of sets of random variables as long as one can sample from the full conditional distributions (see Sec. 10 [198]).

■ 2.2.3 Reversible Jump Markov Chain Monte Carlo

The Reversible Jump Markov chain Monte Carlo (RJMCMC) sampler by Green et al. [94] (see Sec. 11 [198]) is a slight generalization of the Metropolis-Hastings algorithm, that

utilizes auxiliary variables to propose deterministic moves to change between model orders. This is important in the context of Bayesian nonparametric models (introduced in Sec. 2.4) where the number of clusters, i.e. the model order, is part of the inference.

In general the RJMCMC algorithm executes the following three steps. First, draw auxiliary variables $\mathbf{v}$ given the current state $\mathbf{x}$ from some proposal distribution $q(\mathbf{v} \mid \mathbf{x})$:

$$\mathbf{v} \sim q(\mathbf{v} \mid \mathbf{x}) \,. \tag{2.18}$$

Second, apply a deterministic function $f([\mathbf{x}, \mathbf{v}])$ to generate the state after the move $\hat{\mathbf{x}}$ as well as auxiliary variables $\mathbf{u}$:

$$f([\mathbf{x}, \mathbf{v}]) = [\mathbf{u}, \hat{\mathbf{x}}] \,. \tag{2.19}$$

By $[\mathbf{x}, \mathbf{v}]$ we denote stacking of all parameters in $\mathbf{x}$ and $\mathbf{v}$. Third, accept the move from state $\mathbf{x}$ to state $\hat{\mathbf{x}}$ with acceptance probability P_a:

$$\mathrm{P}_a = \min \left\{ 1, \frac{p(\hat{\mathbf{x}})}{p(\mathbf{x})} \frac{q(\mathbf{x} \mid \hat{\mathbf{x}})}{q(\hat{\mathbf{x}} \mid \mathbf{x})} |\det(J_f)| \right\} \,, \tag{2.20}$$

where the Jacobian $J_f = \frac{\partial f([\mathbf{x},\mathbf{v}])}{\partial [\mathbf{x},\mathbf{v}]}$. As can be observed, the acceptance probability is the same as for the vanilla Metropolis-Hastings algorithm in Eq. (2.13) modulo the multiplication by the determinant of the Jacobian of the deterministic transformation. This factors in the scaling incurred by the transformation. It turns out that in all cases relevant to this thesis the determinant is 1, and that therefore the RJMCMC and the Metropolis-Hastings acceptance probabilities are equal.

■ 2.2.4 Slice Sampling

Slice sampling [175] allows sampling from any distribution $p(x)$ known up to proportionality. The idea is to sample uniformly from the area under $p(x)$. Introducing an auxiliary random variable u, the area is

$$A = \{(x, u) \;:\; 0 < u < p(x)\} \,. \tag{2.21}$$

The joint distribution of x and u is defined to be uniform over A. The probability density function value is Z^{-1} where $Z = \int_A \mathrm{d}x \, \mathrm{d}u = \int p(x) \, \mathrm{d}x$ is the normalizer of the target density $p(x)$. Sampling from this uniform joint distribution is achieved using a Gibbs sampler where the conditional distributions are

$$p(u \mid x) = \mathrm{Unif}(0, p(x)) \tag{2.22}$$

$$p(x \mid u) = \mathrm{Unif}(p(x) \geq u) \,. \tag{2.23}$$

While sampling u is straightforward and requires only a single evaluation of $p(x)$, sampling from x can be difficult if the distribution $p(x)$ is multimodal. Neal [175] proposes several algorithms for efficiently sampling from the conditional distribution of x. For additional discussion and practical examples refer to Sec. 8 [198].

2.3 Common Distributions

In the following we review the exponential distribution family before introducing two common family members in detail, the categorical and the Gaussian distribution, as well as their conjugate priors. These distributions are used throughout the thesis. More extensive discussion of these distributions and others can be found in [85].

2.3.1 Exponential Family

The exponential family of distributions is a class of distributions that follow the form

$$p(x \mid \theta) = h(x) \exp\left(\eta(\theta)^T T(x) - A(\theta)\right) , \tag{2.24}$$

where $\eta(\theta)$ are called the natural parameters, $T(x)$ are the sufficient statistics of the data and $A(\theta)$ is called the log-partition function. The log-partition function is determined such that the distribution is normalized to integrate to 1:

$$\exp(A(\theta)) = \int h(x) \exp\left(\eta(\theta)^T T(x)\right) \mathrm{d}\theta \,. \tag{2.25}$$

We will sometimes refer to the inverse partition function as $Z(\theta) = \exp(-A(\theta))$.

Various important distributions belong to the exponential family: the categorical, the Dirichlet, the Gaussian, the Wishart, the inverse Wishart, the beta, the Poisson, the exponential, the gamma, the Bernoulli, and the chi-squared distribution. Less common distributions like the von-Mises-Fisher distribution (see Sec. 2.6.3) and the Bingham distribution [23] are also exponential family members. The exponential family distributions have two important properties for efficient inference: they have a sufficient statistic and a conjugate prior associated with them. In the following we review a subset of the exponential family distributions that are relevant for this thesis.

2.3.2 Categorical and Dirichlet Distribution

The categorical distribution is a distribution over K independent events. The different event probabilities are captured by the parameter $\{\pi_k\}_{k=1}^K$ where $\sum_{k=1}^K \pi_k = 1$. The toss of a fair die, for example, can be modeled by a categorical distribution with all $\pi_k = \frac{1}{6}$. The distribution can be written as

$$p(z \mid \pi) = \prod_{k=1}^{K} \pi_k^{\mathbb{1}_{z=k}} , \tag{2.26}$$

where $z \in \{1, \ldots, K\}$. To designate the drawing or sampling of a random event $z \in \{1, \ldots, K\}$ from a categorical distribution we write:

$$z \sim \mathrm{Cat}(\pi) \,. \tag{2.27}$$

In practice one can sample from a categorical distribution using the inversion method (see the general treatment in Sec. 2.1.2 and Example 2.10 of [198]). This method involves

sampling a value v from the uniform distribution between 0 and 1 and comparing the value against the cumulative sums c_j over π_k, which represent the cumulative density function:

$$c_j = \sum_{k=1}^{j} \pi_k \,, \quad c_0 = 0 \,. \tag{2.28}$$

The random sample z is the value that satisfies $c_{z-1} < v \leq c_z$.

In many real-world Bayesian generative modeling problems the event probabilities π are not known and thus assumed to be random as well. The common distribution assumed for the event probabilities is the Dirichlet distribution. It is a conjugate prior distribution for the categorical (and the multinomial) distribution, which means that the posterior distribution is Dirichlet again. The distribution is

$$p(\pi; \alpha) = \frac{\Gamma\left(\sum_{k=1}^{K} \alpha_k\right)}{\prod_{k=1}^{K} \Gamma(\alpha_k)} \prod_{k=1}^{K} \pi_k^{\alpha_k - 1} \,, \tag{2.29}$$

where $\Gamma(\cdot)$ is the gamma function. Sampling event probabilities π from a Dirichlet distribution parameterized by the pseudo counts $\{\alpha_k\}_{k=1}^{K}$, $\alpha_k \geq 0$ is denoted as:

$$\pi \sim \mathrm{Dir}(\alpha) \,. \tag{2.30}$$

In practice sampling from a Dirichlet distribution can be accomplished via sampling from a gamma distribution (which is implemented in most scientific programming languages and libraries) and making the resulting samples sum to 1:

$$\pi_k = \frac{\tilde{\pi}_k}{\sum_{k=1}^{K} \tilde{\pi}_k} \,, \qquad\qquad \tilde{\pi}_k \sim \mathrm{Gamma}(\alpha_k, 1) \,. \tag{2.31}$$

In summary, the Dirichlet distribution is a distribution over a distributions. To go back to the example with rolling a die: The Dirichlet distribution is like having a big bag of infinitely many dice with different event probabilities. Before rolling a die we blindly draw a die from the bag. Now the die we are going to roll to obtain a number is distributed according to the distribution of dice that we put into the bag. The Dirichlet distribution is a distribution over the simplex defined by $\sum_{k=1}^{K} \pi_k = 1$. For $K = 3$ can visualize it for different parameter values α in Fig. 2.1.

Assume that we are given a set of labels $\mathbf{z} = \{z_i\}_{i=1}^{N}$ and are interested in knowing the posterior distribution on π given the observed labels. We will assume a Dirichlet prior with parameter α on the event probabilities π. Since the Dirichlet distribution is conjugate to the categorical distribution of the data the posterior is Dirichlet again:

$$p(\pi \mid \mathbf{z}; \alpha) = \mathrm{Dir}(\alpha_1 + N_1, \ldots, \alpha_K + N_k) \,, \; N_k = \sum_{i=1}^{N} \mathbb{1}_{z=k} \,, \tag{2.32}$$

where $\mathbb{1}_{(\cdot)}$ is the indicator function which is 1 if $(\cdot)$ is true and 0 otherwise. Hence N_k is the number of labels z having the value k.

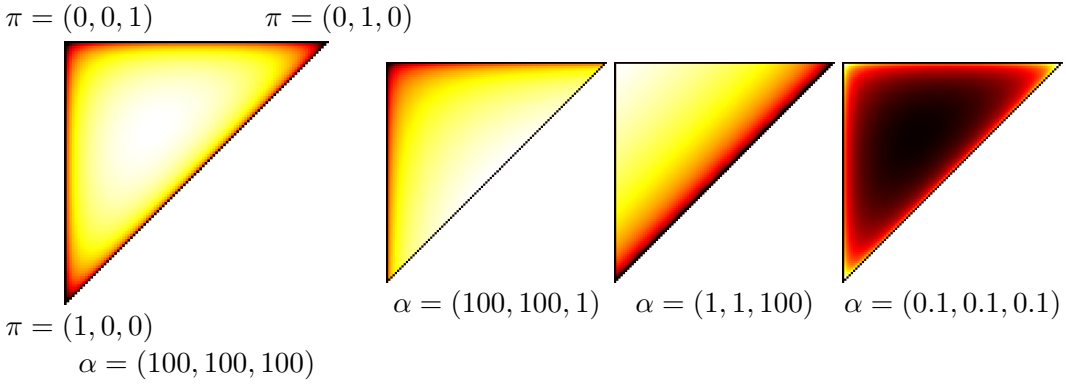


Figure 2.1: Visualization of the Dirichlet distribution density for $K = 3$ categories. Each location on the simplex corresponds to a Categorical distribution parameter π as indicated for the corners to the left. "Hotter" coloring indicates a higher density value. For $\alpha = (1, 1, 1)$ the distribution is uniform over the simplex.

■ 2.3.3 Gaussian and Normal Inverse Wishart Distribution

The Gaussian distribution is a well known distribution for a concentrated cluster of data in Euclidean space. Its density in D dimensions is given as

$$\mathcal{N}(x \mid \mu, \Sigma) = \left((2\pi)^D |\Sigma|\right)^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right) \tag{2.33}$$

$$= \left((2\pi)^D |\Sigma|\right)^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}\|x-\mu\|_\Sigma^2\right) . \tag{2.34}$$

There are various methods for sampling from the 1D Gaussian distribution (see Example 2.8 of [198]) and most scientific libraries and programming languages allow have one of the variants implemented. To sample from the multivariate Gaussian distribution we use the fact that we can sample from the D-dimensional zero-mean unit-variance Gaussian distribution by simply sampling D draws from $\mathcal{N}(0, 1)$:

$$x = (x_1, \ldots, x_D) \text{ where } x_i \sim \mathcal{N}(0, 1) \tag{2.35}$$

$$\Sigma^{\frac{1}{2}} x + \mu \sim \mathcal{N}(\mu, \Sigma) , \tag{2.36}$$

where $\Sigma^{\frac{1}{2}}$ is the matrix square root of the covariance matrix which can be obtained via the Cholesky decomposition.

In Bayesian generative modeling the parameters μ and Σ should often be part of the inference procedure and are thus assumed the have prior distribution. While one could put various prior distributions, it is convenient to choose the Normal inverse Wishart (NIW) distribution because it is a conjugate prior for the Normal distribution. This makes posterior inference easier since the posterior distribution of the Gaussian parameters is NIW again. The NIW distribution is given as the product of a Gaussian

distribution (as a prior on the Gaussian mean) and a inverse Wishart (IW) distribution:

$$\mathrm{NIW}(\mu, \Sigma; \kappa, \vartheta, \nu, \Delta) = \mathcal{N}\left(\mu; \vartheta, \tfrac{\Sigma}{\kappa}\right) \mathrm{IW}(\Sigma; \nu, \Delta)\,. \tag{2.37}$$

Sampling from the NIW distributions proceeds in two steps:

$$\Sigma \sim \mathrm{IW}(\nu, \Delta) \tag{2.38}$$

$$\mu \sim \mathcal{N}\left(\vartheta, \tfrac{\Sigma}{\kappa}\right)\,. \tag{2.39}$$

While sampling from the Gaussian is quite straightforward, sampling from the inverse Wishart amounts to sampling from a Wishart distribution as described in [181] and inverting the resulting matrix [85].

Given some Gaussian observations x the posterior distribution for the Gaussian mean and covariance matrix is NIW with the parameters [85]:

$$\kappa_N = \kappa_0 + N \tag{2.40}$$

$$\nu_N = \nu_0 + N \tag{2.41}$$

$$\vartheta_N = \frac{\kappa_0}{\kappa_N}\vartheta_0 + \frac{n}{\kappa_N}\bar{x} \tag{2.42}$$

$$\Delta_N = \Delta_0 + S + \tfrac{\kappa_0 N}{\kappa_N}(\vartheta_0 - \bar{x})(\vartheta_0 - \bar{x})^T\,, \tag{2.43}$$

where we have used the sufficient statistics N and

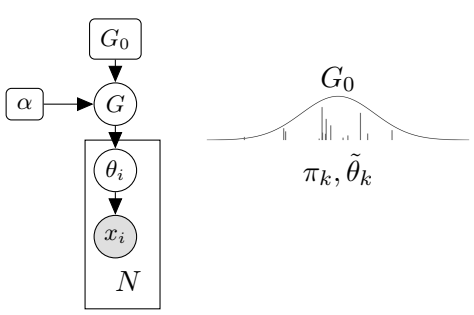
$$\bar{x} = \tfrac{1}{N}\sum_{i=1}^{N} x_i \tag{2.44}$$

$$S = \sum_{i=1}^{N} x_i x_i^T - N\bar{x}\bar{x}^T\,. \tag{2.45}$$

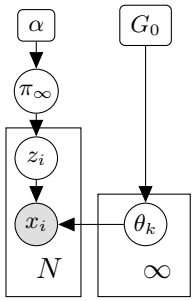
The importance of sufficient statistics in general is that once we have computed them, we can discard the underlying data points x_i because everything needed for inference is captured in the statistics. This is an important factor in deriving and implementing efficient inference algorithms. One can either incrementally add and remove data points to and from the statistics, or even run pyramid-schemed reduction operations that can be parallelized and run on a graphics processing unit (GPU).

■ 2.4 Bayesian Nonparametric Mixture Models

Bayesian nonparametrics (BNP) is a subset of Bayesian statistics that investigates models with an infinite parameter space. Like standard Bayesian models (see Sec. 2.1), Bayesian nonparametric models describe observations x as drawn randomly from the likelihood distribution $p(x \mid \theta)$ parameterized by θ. The parameter θ is assumed to be a random variable distributed according to the prior $p(\theta; \alpha)$ with hyper-parameters α. In contrast to standard Bayesian models, the dimension of θ is by construction



(a) Implicit graphical representation of the Dirichlet process mixture model.



(b) Explicit Dirichlet process mixture model with instantiated mixture weights π_∞ and indicators $\{z_i\}_{i=1}^N$.

Figure 2.2: Explicit and implicit Dirichlet process mixture model representations.

infinite in the Bayesian nonparametric case. Examples of BNP priors are the Dirichlet process (DP) prior [8, 70, 233], Indian buffet process [86], or the Gaussian process (GP) prior [194, 253]. For this work DP mixture models [8] (DP-MM) are of main interest and are thus introduced next.

The DP prior is used as a prior for probabilistic models with a countably-infinite number of mixture components. Formally, a DP is a stochastic process whose sample path defines a probability distribution over the event space Ω. It is defined via a base measure G_0 on Ω and a concentration $\alpha > 0$. A realization $G \sim \mathrm{DP}(G_0, \alpha)$ of the Dirichlet process satisfies the property that for any finite partition of the event space $\Omega = \bigcup_{k=1}^K A_k$,

$$(G(A_1), \ldots, G(A_K)) \sim \mathrm{Dir}(\alpha G_0(A_1), \ldots, \alpha G_0(A_K)) \,. \tag{2.46}$$

In other words, the probability mass of the partitions A_k under the realization G of the DP follows a Dirichlet distribution whose weights are proportional to the probability mass of the partitions under the base measure G_0. A draw $G \sim \mathrm{DP}(\alpha, G_0)$ can be represented as the sum of delta functions at the atom locations $\tilde{\theta}_i$:

$$G = \sum_{i=1}^{\infty} \pi_i \delta(\tilde{\theta}_i) \,. \tag{2.47}$$

Figure 2.2 depicts the base measure G_0 as well as the random measure G drawn from it by the atoms $\tilde{\theta}_k$ with associated weights π_k. From a generative point of view a data point x_i is generated by drawing a parameter θ_i from G followed by drawing $x_i \sim p(x_i \mid \theta_i)$. This model is depicted in Fig. 2.2a. The structure of G implies that samples $\theta_i \sim G$ will repeat with non-zero probability. This observation clarifies the clustering property of the DP prior and motivates its use in Bayesian mixture models.

An alternate construction due to Sethuraman [216] is called the *stick-breaking con-*

struction of a DP. The aforementioned atom weights π_k are defined recursively as:

$$\pi_k = c_k \prod_{j=1}^{k-1}(1 - c_j) \qquad c_k \sim \text{Beta}(1, \alpha) \quad \forall k \in [1, \ldots, \infty)\,. \tag{2.48}$$

Atoms θ_k are drawn from the base measure G_0. The stick-breaking formulation provides a way of sequentially sampling from the DP. Since parameters θ_i will repeat with certainty, it is not necessary to instantiate a dedicated parameter θ_i per data point. Instead a set of distinct parameters $\{\theta_k\}_{k=1}^{K}$ is maintained. To indicate the assignment of a data point x_i to one of the atoms θ_k indicator variables z_i are commonly instantiated. The resulting graphical model is depicted in the explicit model of a DP-MM in Fig. 2.2b. Using the stick-breaking construction the DP-MM is defined from a generative standpoint as

$$\pi_\infty \sim \text{GEM}(1, \alpha) \tag{2.49}$$
$$\theta \sim p(\theta; G_0) \tag{2.50}$$
$$z_i \sim \text{Cat}(\pi_\infty) \tag{2.51}$$
$$x_i \sim p(x_i \mid \theta, z_i)\,, \tag{2.52}$$

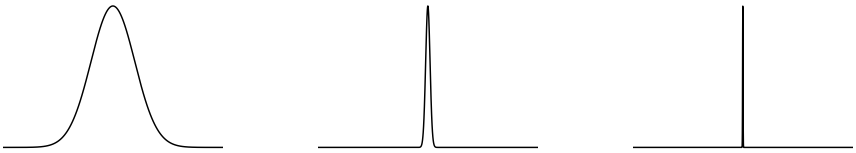
where the stick breaking process was denoted $\text{GEM}(1, \alpha)$ after its inventors Griffiths, Engen and McCloskey [190].

A second approach to constructing the draw G from a DP utilizes a Poisson process with a specific measure to define a Gamma process [151]. The DP is obtained as the normalization of the draw from a Gamma process. This approach can also be used to construct dependencies between DPs via dependencies between the underlying Poisson processes [151].

One way to intuitively understand the effects of a DP prior is via the so called Chinese Restaurant process (CRP). The CRP is obtained from the DP by marginalizing over the random measure G [25, 174]. It can be understood by visualizing the process of customers sitting down at tables in a restaurant. Customers are the equivalent of data-points and tables correspond to mixture components in the actual DP-MM. The probability of customer i being assigned to table k ($z_i = k$) can be derived to be:

$$p(z_i = k \mid z_{\setminus i}; \alpha) \propto \begin{cases} N_k & 1 \leq k \leq K \\ \alpha & k = K + 1 \end{cases}\,. \tag{2.53}$$

This highlights the "rich-get-richer" property of the CRP and thus the DP: the more customers sit at a certain table (i.e. are assigned to a mixture component) the more likely it is that new customers will join them. There are three main inference methods for DP-MMs: (1) sampling-based inference, (2) variational inference, and (3) low variance asymptotic analysis. MCMC-based inference algorithms [42, 174, 193] aiming to sample from the true posterior distribution using Gibbs or Metropolis-Hastings samplers as introduced in Sec. 2.2. Early work on inference for Dirichlet process mixture models (DP-MM) relied on the CRP in the sampling-based inference algorithm [174, 193].



(a) $\sigma = 1$ (b) $\sigma = 0.1$ (c) $\sigma = 0.01$

Figure 2.3: Zero-mean Gaussians with variance σ. In the limit as $\sigma \to 0$ the Gaussian distribution approaches a Dirac delta function.

This approach exhibits slow convergence, is prone to get stuck in local optima and does not lend itself to parallelization [42, 121]. Variational inference algorithms which aim to minimize the Kullback-Leibler divergence between the unknown true DP posterior and a mean-field approximation to it. The stick-breaking construction is usually used to derive variational inference methods. See [27, 128] for an introduction to these methods. Low variance analysis applied to the DP posterior distribution [33, 39, 123, 142, 222] yields k-means-like inference algorithms [101] (see Sec. 2.5 for more details).

A variety of more complex models has been proposed based on the DP such as the hierarchical DP [232] (HDP) or the dependent DP (DDP) [151, 160]. These models allow modeling of dependencies between DPs and hence sharing of data between them in specific ways.

■ 2.5 Low Variance Asymptotics

It is well known that the k-means clustering algorithm [155] can be obtained from the expectation-maximization (EM) inference in a finite Gaussian mixture model (GMM) by analyzing the limit as the variance σI of the Gaussians approaches zero [24]. In the limit as $\sigma \to 0$ the Gaussian distribution approaches a Dirac delta function (c.f. Fig. 2.3) and the label posterior distribution becomes an assignment to the closest cluster with probability 1.

Kulis et al. [142] were first to apply the same analysis to a BNP Dirichlet process GMM. The resulting DP-means algorithm resembles the k-means algorithm while also allowing the creation of additional clusters. As shown in Algorithm 3, a new cluster is created if a data-point's squared Euclidean distance from all currently existing clusters exceeds a predefined threshold λ on the cluster radius:

$$z_i = \underset{k \in \{1,\ldots,K+1\}}{\arg\min} \begin{cases} ||x_i - \mu_k||_2^2 & \text{for } k \leq K \text{ and } |\mathcal{I}_k \setminus i| > 0 \\ \infty & \text{for } k \leq K \text{ and } |\mathcal{I}_k \setminus i| = 0 \\ \lambda & \text{for } k = K+1 \end{cases} . \tag{2.54}$$

If data point i is the last in cluster k and hence $|\mathcal{I}_k \setminus i| = 0$, it cannot be re-assigned to the same cluster as expressed by the infinite cost on the assignment. If x_i gets assigned to a new cluster the new cluster is instantiated with mean $\mu_{K+1} = x_i$ and the cluster count K is incremented. After each assignment cycle the means μ_k are updated to the

sample means of the respective cluster:

$$\mu_k = \frac{1}{|\mathcal{I}_k|} \sum_{i \in \mathcal{I}_k} x_i \,. \tag{2.55}$$

Jiang et al. [123] generalize this derivation to DP mixtures of general exponential family distributions. Broderick et al. [33] show a different derivation for the DP-means algorithm by directly taking the low variance limit on the posterior distribution of the model. The result of this is an objective function that resembles the k-means objective with the addition of a term that penalizes the creation of new clusters:

$$J = \sum_{k=1}^{K} \sum_{i \in \mathcal{I}_k} ||x_i - \mu_k||_2^2 + \lambda(K-1) \,. \tag{2.56}$$

This objective function can be optimized in many different ways, one of which is the aforementioned DP-means algorithm.

$J \leftarrow \infty$
$\boldsymbol{\mu} \leftarrow \emptyset$
while J not converged **do**
  for $i \in \{1, \ldots, N\}$ **do**
    $z_i = \arg\min_{k \in \{1,\ldots,K+1\}} \begin{cases} ||x_i - \mu_k||_2^2 & \text{for } k \leq |\boldsymbol{\mu}| \text{ and } |\mathcal{I}_k \setminus i| > 0 \\ \infty & \text{for } k \leq |\boldsymbol{\mu}| \text{ and } |\mathcal{I}_k \setminus i| = 0 \\ \lambda & \text{for } k = |\boldsymbol{\mu}| + 1 \end{cases}$
    if $z_i = |\boldsymbol{\mu}| + 1$ **then**
      $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \cup \{x_i\}$ // add new cluster
    end if
  end for
  for $k \in \{1, \ldots, |\boldsymbol{\mu}|\}$ **do**
    if $|\mathcal{I}_k| > 0$ **then**
      $\mu_k \leftarrow \frac{1}{|\mathcal{I}_k|} \sum_{\mathcal{I}_k} n_i$
    else
      $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \setminus \mu_k$ // remove cluster k
    end if
  end for
  $J \leftarrow \sum_{k=1}^{|\boldsymbol{\mu}|} \sum_{i \in \mathcal{I}_k} \|x_i - \mu_k\|_2^2 + \lambda(|\boldsymbol{\mu}| - 1)$
end while

Algorithm 3: DP-means algorithm.

Campbell et al. [39] derive a k-means-like clustering algorithm for streaming data. The starting point for their derivation is a dependent DP-GMM (DDP-GMM) that couples clusters between batches data as constructed in [151]. The algorithm allows

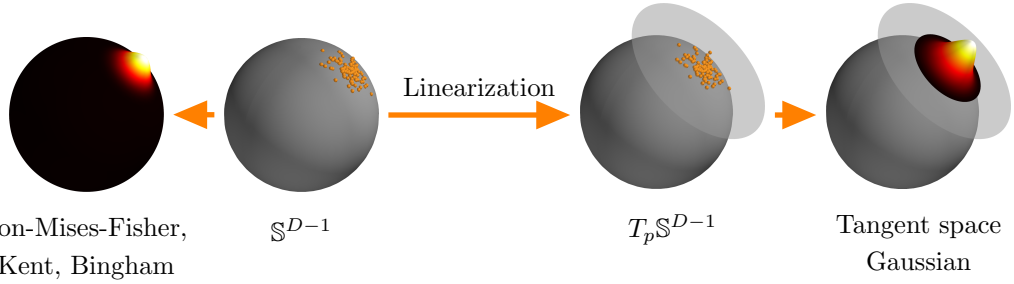


Figure 2.4: Two different approaches to modeling a cluster of directional data. A directional distribution can either be defined directly over the manifold of the sphere (left) or implicitly via a distribution in a linearization of the sphere (right).

for death, revival, movement, and birth of clusters. The technique has also been extended to develop asymptotic algorithms for mixtures with general exponential family likelihoods [123], and HMMs with an unknown number of states [206]. However, small-variance analysis has, to date, been limited to discrete and Euclidean spaces. In Sec. 4.4.3, we derive the first k-means-like algorithms for directional data.

■ 2.6 Distributions over the Unit Sphere

Classical statistics rely on the Euclidean structure of $\mathbb{R}^D$. If the data naturally resides on a manifold, special care needs to be taken since the notion of distance on a manifold is different than in Euclidean space [63]. In this section we focus on directional data n which naturally resides on the unit sphere in D dimensions $\mathbb{S}^{D-1}$. Due to the nonlinearity of the sphere the statistical analysis of spherical data requires special care [75, 164].

As depicted in Fig. 2.4, we explore two different approaches to modeling directional data. Since the sphere is a $(D-1)$-dimensional nonlinear Riemannian manifold [63] we can use Riemannian geometry in combination with the standard Euclidean distributions, such as the Gaussian distribution, to define distributions on the sphere such as the tangent space Gaussian depicted to the right in Fig. 2.4. The advantages of this approach are that there exists conjugate priors for the Gaussian distribution parameters, that the Gaussian distribution can represent anisotropic distributions, and that there are fast samplers for the Gaussian as well as its prior distributions. Additionally, we explore distributions from directional statistics [164] that are naturally defined over the unit sphere $\mathbb{S}^{D-1}$. Examples of directional distributions are the antipodal symmetric Bingham distribution [23], the anisotropic Kent or also called Fisher-Bingham distribution [134] and the isotropic von-Mises-Fisher (vMF) distribution [74]. Because of its comparative simplicity, the vMF distribution is most commonly used. Aside from being natively defined over the unit sphere, those distributions have some advantages in posterior inference as we will show. Unfortunately the normalizers tend to involve com-

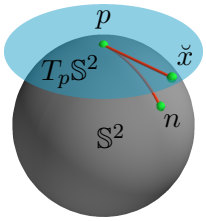


Figure 2.5: The blue plane illustrates $T_p\mathbb{S}^2$, the tangent space to the sphere $\mathbb{S}^2$ at $p \in \mathbb{S}^2$ (here p is taken to be the north pole). A tangent vector $\breve{x} \in T_p\mathbb{S}^2$ is mapped to $n \in \mathbb{S}^2$ via Exp_p.

plicated functions such as the Bessel function of the first kind [74]. Another drawback is that in some cases conjugate prior distributions are only known up to proportionality (see Sec. 2.6.3). We first introduce details on the manifold of the sphere to set the stage for distributions over data on the sphere.

■ 2.6.1 The Manifold of the Unit Sphere

Directional data n resides on the unit sphere in D dimensions defined as

$$\mathbb{S}^{D-1} = \{n : n^T n = 1\,;\, n \in \mathbb{R}^D\}\,. \tag{2.57}$$

We now give a brief introduction to the geometric concepts useful for statistical modeling on the manifold of the unit sphere $\mathbb{S}^{D-1}$. The interested reader can consult [63] for a more detailed discussion and other manifolds. In statistical modeling, distances are of paramount importance. On $\mathbb{S}^{D-1}$, rather than using the Euclidean distance of the ambient space, $\mathbb{R}^D$, an appropriate measure is the *geodesic distance* between points, which is simply the angle between them:

$$d_G(p, n) = \arccos(p^T n)\,, \tag{2.58}$$

where $p, n \in \mathbb{S}^{D-1}$. The probability measure we will define on $\mathbb{S}^{D-1}$ will exploit this distance measure implicitly through the concept of a tangent space. While $\mathbb{S}^{D-1}$ is nonlinear, every point $p \in \mathbb{S}^{D-1}$ is associated with a linear *tangent space*, denoted $T_p\mathbb{S}^{D-1}$, as illustrated in Fig. 2.5:

$$T_p\mathbb{S}^{D-1} \triangleq \{\breve{x} : p^T\breve{x} = 0\}\,. \tag{2.59}$$

Elements $\breve{x}$ of $T_p\mathbb{S}^{D-1}$ are called *tangent vectors* and may be viewed as "arrows" based at p and tangent to $\mathbb{S}^{D-1}$. Note that $\dim(T_p\mathbb{S}^{D-1}) = D-1$ and that the point of tangency, p, may be identified with the origin of $\mathbb{S}^{D-1}$ (i.e., a zero-length tangent vector). A point, $n \in \mathbb{S}^{D-1} \setminus \{-p\}$, is mapped to a point, $\breve{x} \in T_p\mathbb{S}^{D-1}$, via

$$\text{Log}_p(n) = (n - p\cos\theta)\tfrac{\theta}{\sin\theta} = \breve{x}\,, \tag{2.60}$$

where $\theta = d_G(p,n)$ and $\frac{0}{\sin 0} = 1$. At the antipodal point $-p$ the mapping is not defined. Conversely, $\breve{x} \in T_p\mathbb{S}^{D-1}$ is mapped to $x \in \mathbb{S}^{D-1}$ by the Riemannian exponential:

$$\mathrm{Exp}_p(\breve{x}) = p\cos(\|\breve{x}\|_2) + \frac{\breve{x}}{\|\breve{x}\|_2}\sin(\|\breve{x}\|_2) = n\,. \tag{2.61}$$

The ℓ_2 norm $\|\breve{x}\|_2$ in $T_p\mathbb{S}^{D-1}$ is equal to the distance between p and n: $\|\breve{x}\|_2 = \theta = d_G(p,n)$. However, this is true only since p is the *point of tangency*. This implies that $T_p\mathbb{S}^{D-1}$ is a bounded open set of radius π. In general, the distance between two *other* points in $T_p\mathbb{S}^{D-1}$ is not equal to the geodesic distance between their corresponding points in $\mathbb{S}^{D-1}$.

Due to their linearity, tangent spaces often provide a convenient way to model spherical data using distributions native to Euclidean spaces. In fact, this is also true for more general manifolds [78, 80, 187, 220]. This linearity, together with aforementioned mappings between $\mathbb{S}^{D-1}$ and $T_p\mathbb{S}^{D-1}$, enables modeling directional data points via a zero-mean Gaussian distribution in the tangents space as described in Sec. 2.6.2.

Note that in the formulas for the Riemannian exp and log map points $x \in T_p\mathbb{S}^{D-1}$ are vectors in the ambient Euclidean space $\mathbb{R}^D$ fulfilling Eq. (2.59). In the following we express $\breve{x}$ in local tangent space coordinates as $\breve{x}^m$ by parallel transporting all data into the tangent space around the north pole $m = (0,\ldots,0,1) \in \mathbb{S}^{D-1}$ using a rotation R. Vectors $\breve{x}^m \in T_n\mathbb{S}^{D-1}$ all have $\breve{x}^m_{D-1} = 1$ and hence $\breve{x}^m$ is obtained from $\breve{x}$ by simply discarding the last coordinate entry. In the following all this process is implicitly assumed whenever a point on the sphere is mapped into a tangent space via the log map. Likewise whenever the exp map is used the process is performed in the inverse direction: first augment the vector with another coordinate with value 1, then rotate the resulting point down to the tangent space via R^T.

While the rotation R can be computed in any dimension, it is straightforward in 3D via the axis-angle formulation with axis $w = \frac{n\times p}{\|n\times p\|_2}$ and angle $\theta = \arccos(n^T p)$ as described in Sec. 2.7.4.

Numeric and implementation considerations When implementing the log map numerical instabilities can be avoided by using Taylor expansions around the instable points:

$$\mathrm{Log}_p(n) = (n - pp^T n)\begin{cases} \frac{\arccos(p^T n)}{\sqrt{1-(p^T n)^2}} & \text{for } 1 - p^T n < \epsilon \\ 1 - \frac{1}{3}\left(p^T n - 1\right) + \frac{2}{15}\left(p^T n - 1\right)^2 & \text{otherwise} \end{cases}. \tag{2.62}$$

The Riemannian exponential map can be implemented as in Eq. (2.61) and using the Taylor expansion of the sinc $\|\breve{x}\|_2$ for small angles $\|\breve{x}\|_2$:

$$\mathrm{Exp}_p(\breve{x}) = p\cos(\|\breve{x}\|_2) + \breve{x}\begin{cases} \frac{\sin(\|\breve{x}\|_2)}{\|\breve{x}\|_2} & \text{for } \|\breve{x}\|_2 < \epsilon \\ 1 - \frac{\breve{x}^T\breve{x}}{6} + \frac{\left(\breve{x}^T\breve{x}\right)^2}{120} & \text{otherwise} \end{cases}. \tag{2.63}$$

Furthermore, due to numerical inaccuracies dot products like $p^T n$ might not be strictly inside $[-1,1]$. Hence, it is generally advisable to truncate the dot product to be inside

$[-1, 1]$ to ensure that, for example, the arccos can be computed:

$$p^T n = \min\left(1, \max\left(-1, p^T n\right)\right) . \tag{2.64}$$

Surface area and volume of unit $(D-1)$-sphere The area $A_{\mathbb{S}^{D-1}}$ and the volume $V_{\mathbb{S}^{D-1}}$ of the sphere in D dimensions, $\mathbb{S}^{D-1}$, are

$$A_{\mathbb{S}^{D-1}} = \frac{2\pi^{\frac{D}{2}}}{\Gamma\left(\frac{D}{2}\right)} \tag{2.65}$$

$$V_{\mathbb{S}^{D-1}} = \frac{\pi^{\frac{D}{2}}}{\Gamma\left(1+\frac{D}{2}\right)} . \tag{2.66}$$

Interestingly both volume and area approach 0 as the dimension increases.

The Karcher Mean

The Karcher mean $\tilde{n}$ is a generalization of the sample mean in Euclidean space [95, 130, 131] to manifold-valued data. For a set of points $\{n_i\}_{i=1}^N$ on manifold M with associated distance measure $d(\cdot,\cdot)$, the Karcher mean is defined as the (local) minimizer of the following weighted cost function:

$$\tilde{n} = \arg\min_{p \in M} \sum_{i=1}^{N} w_i d^2(p, n_i) , \quad \sum_{i=1}^{N} w_i = 1 . \tag{2.67}$$

For the purpose of this thesis we are focusing on the manifold of the sphere, $M = \mathbb{S}^{D-1}$, and usually have uniform weights $w_i = \frac{1}{N}$. For directional data the distance function is $d(\cdot,\cdot) = d_G(\cdot,\cdot) = \arccos(p^T n)$. In this case, excepting degenerate sets, the optimization problem has a single minimum. It may be obtained by iterating the computation of the data sample mean in the tangent space around the current estimate of the Karcher mean, and updating the Karcher mean to the projection of that sample mean back onto the sphere [187]. Outlined in Algorithm 4, the Karcher mean algorithm takes the geometry of the sphere into account and exhibits fast convergence.

1: Initialize $\tilde{n}_0$ to any data point n_i
2: **do**
3: $\quad \bar{x} = \sum_{i=1}^{N} w_i \mathrm{Log}_{\tilde{n}_t}(n_i)$
4: $\quad \tilde{n}_{t+1} = \mathrm{Exp}_{\tilde{n}_t}(\bar{x})$
5: **while** $\|\bar{x}\|_2 > \epsilon$

Algorithm 4: Karcher mean algorithm.

■ 2.6.2 The Tangent Space Gaussian (TG) Distribution

Under the tangent space Gaussian model the deviations of observed D-dimensional directional data from a mean direction μ are modeled by a $D-1$-dimensional zero-mean

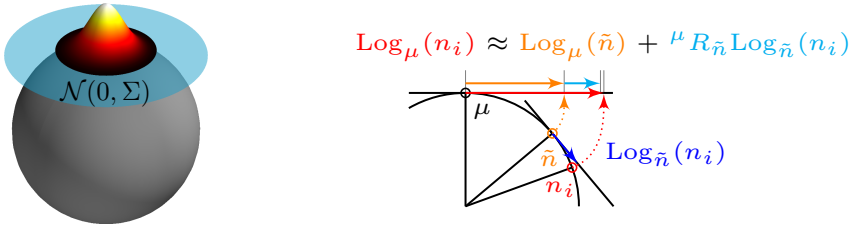


Figure 2.6: Left: Directional data can be described via zero-mean Gaussian distributions in cluster dependent tangent spaces $T_\mu\mathbb{S}^2$. Right: Illustration of the geometry underlying the approximation of the mapping of n_i into $T_\mu\mathbb{S}^2$ via $\mathrm{Log}_\mu(n_i)$.

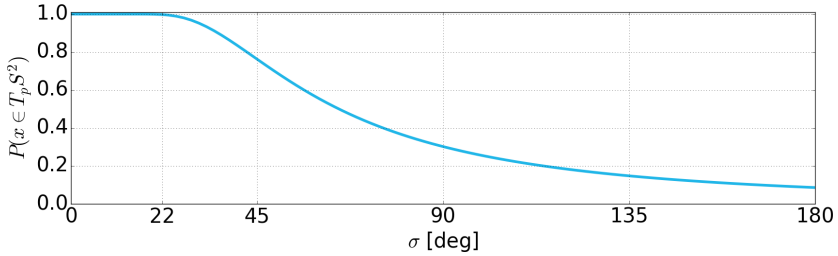


Figure 2.7: Probability that a data point x lies inside a disk with radius π under a zero-mean Gaussian distribution with covariance $\Sigma = \sigma^2\mathrm{I}$ as a function of σ. For $\sigma < 22°$ the probability mass of a Gaussian distribution is approximately contained in $T_p\mathbb{S}^2$. Beyond 22° the tangent-space Gaussian distribution is less and less well normalized.

Gaussian distribution with covariance $\Sigma \in \mathbb{R}^{D-1\times D-1}$ in the tangent space around μ. The conjugate prior distribution for covariance matrices Σ is the inverse Wishart (IW) distribution (see Sec. 2.3) parameterized by $\Delta \in \mathbb{R}^{D-1\times D-1}$ and $\nu \geq 0$:

$$p(n \mid \mu, \Sigma) = \mathcal{N}(\mathrm{Log}_\mu(n); 0, \Sigma) \tag{2.68}$$

$$p(\Sigma; \Delta, \nu) = \mathrm{IW}(\Sigma; \nu, \Delta)\,, \tag{2.69}$$

where $\mathrm{Log}_\mu(n) \in T_\mu\mathbb{S}^{D-1}$. In other words, we evaluate the probability density function of $n \in \mathbb{S}^{D-1}$ by first mapping it into $T_\mu\mathbb{S}^{D-1}$ and then evaluating it under the Gaussian distribution with covariance $\Sigma \in \mathbb{R}^{D-1\times D-1}$. The probability density function of the directional data over the nonlinear $\mathbb{S}^{D-1}$ is then induced by the Riemannian exponential map:

$$n \sim \mathrm{Exp}_\mu\left(\mathcal{N}(0, \Sigma)\right)\,. \tag{2.70}$$

This setup is depicted in Fig. 2.6. The range of Log_p is contained within a disk of finite radius (π) while the Gaussian distribution has infinite support. Hence, the tails of the Gaussian are "cut-off" outside that area. As a result the implied distribution on the sphere is only approximate since it does not integrate to one. As long as the variance is small, however, the approximation-error is small because most of the probability mass of

the Gaussian is inside the open set with radius π as depicted in Fig. 2.7. Consequently, for probabilistic inference, we use the inverse Wishart prior to favor small covariances resulting in a probability distribution that, except a negligible fraction, is within the range of Log_p and concentrated about the mean. Note that all proposed approaches utilizing Riemannian geometry and the notion of tangent spaces to define a probability distribution on a sphere can be extended to arbitrary simply connected Riemannian manifolds.

Log-map Approximation

The tangent space Gaussian model described previously necessitates the evaluation of $\mathrm{Log}_\mu(n_i)$ to evaluate the data likelihood. This can become computationally expensive especially in iterative inference algorithms such as MCMC or optimization-based methods. To improve computational efficiency in such cases we introduce an approximation to $\mathrm{Log}_\mu(n_i)$. The approximation necessitates the computation of the Karcher mean $\tilde{n}$ for the set of normals associated with the cluster on the sphere. After this preparation step, we can approximate $\mathrm{Log}_\mu(n_i)$ using the Karcher mean $\tilde{n}$ of its associated set of data $\{n_i\}_\mathcal{I}$:

$$\mathrm{Log}_\mu(n_i) \approx \mathrm{Log}_\mu(\tilde{n}) + {}^\mu R_{\tilde{n}} \mathrm{Log}_{\tilde{n}}(n_i)\,. \tag{2.71}$$

where ${}^\mu R_{\tilde{n}}$ rotates vectors in $T_{\tilde{n}}S^2$ to $T_\mu S^2$. Intuitively this approximates the mapping of n_i into μ by the mapping of the Karcher mean into μ plus a correction term that accounts for the deviation of n_i from the $\tilde{n}$. See Fig. 2.6 for an illustration of underlying geometry.

Clearly when $\mu = \tilde{n}$ the expression above is exact. For other cases, we analyze the nature of the approximation in more detail in the following.

μ, $\tilde{n}$ and n on geodesic If μ, $\tilde{n}$ and n lie on a geodesic, then the approximation is exact. This is because points on a straight line in $T_\mu \mathbb{S}^{D-1}$ correspond to points on a geodesic in the direction of $\breve{x} \in T_p \mathbb{S}^{D-1}$ by the definition of the Riemannian exponential map. The length of the vector $\breve{x}$ is equivalent to the angle from μ to $\mathrm{Exp}_\mu(\breve{x}) = n$. Hence we can write that

$$\mathrm{Log}_\mu(n) = \mathrm{Log}_\mu(\tilde{n}) + \delta\,, \tag{2.72}$$

where δ is the vector in the same direction of $\mathrm{Log}_\mu(\tilde{n})$ that leads to the equality. Because all points lie on a geodesic, δ has to have a length equal to the angle between $\tilde{n}$ and n and has to point in the same direction as $\mathrm{Log}_\mu(\tilde{n})$ in $T_\mu \mathbb{S}^{D-1}$. $\mathrm{Log}_{\tilde{n}}(n)$ has the appropriate length and by rotating it from being orthogonal to $\tilde{n}$ to being orthogonal to μ via the rotation ${}^\mu R_{\tilde{n}}$ we obtain the desired vector delta as:

$$\delta = {}^\mu R_{\tilde{n}} \mathrm{Log}_{\tilde{n}}(n)\,. \tag{2.73}$$

This geometric arguments shows that the approximation is exact for μ, $\tilde{n}$ and n on a geodesic. Numerical simulation corroborates this argument.

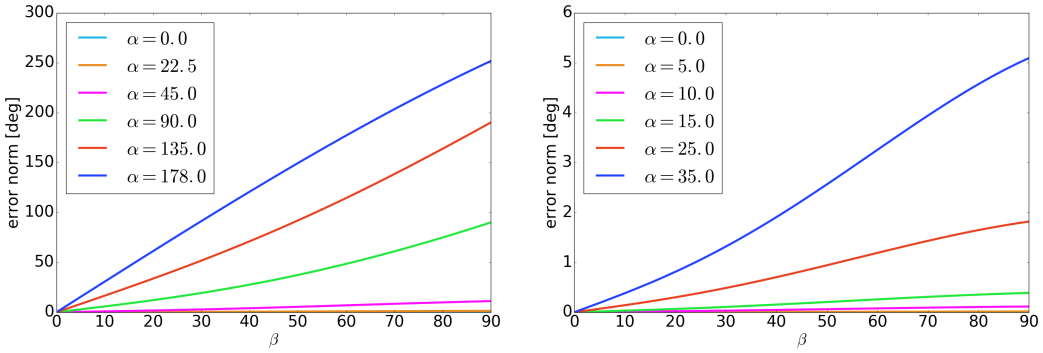


Figure 2.8: The norm of the log map approximation error as a function of angle α between μ and Karcher mean $\tilde{n}$ and β between $\tilde{n}$ and n. In the experiment μ was chosen to be the north pole, $\tilde{n}$ to be at elevation α and n to be at elevation α and azimuth β. This simulates the worst case situation where the two logarithm maps $\mathrm{Log}_\mu(\tilde{n})$ and $\mathrm{Log}_{\tilde{n}}(n)$ are orthogonal. The left plot shows that for small angles α the approximation is close to the true logarithm map. The plot on the right shows the error plots for smaller α values. Note that the inference implicitly aims to minimize α thus improving the approximation.

μ, $\tilde{n}$ and n in arbitrary locations With the angle α between μ and $\tilde{n}$ and the angle β between n and $\tilde{n}$:

$$\begin{aligned}
\mathrm{Log}_\mu(\tilde{n}) + R\mathrm{Log}_{\tilde{n}}(n) &= (\tilde{n} - \mu\cos\alpha)\tfrac{\alpha}{\sin\alpha} + R^\mu_{\tilde{n}}(n - \tilde{n}\cos\beta)\tfrac{\beta}{\sin\beta} \\
&= (\tilde{n} - \mu\cos\alpha)\tfrac{\alpha}{\sin\alpha} + (R^\mu_{\tilde{n}}n - \mu\cos\beta)\tfrac{\beta}{\sin\beta} \\
&= \tilde{n}\tfrac{\alpha}{\sin\alpha} + R^\mu_{\tilde{n}}n\tfrac{\beta}{\sin\beta} - \mu(\tfrac{\alpha}{\tan\alpha} + \tfrac{\beta}{\tan\beta}) \\
&= (n+\delta)\tfrac{\alpha}{\sin\alpha} + R^\mu_{\tilde{n}}(\tilde{n} - \delta)\tfrac{\beta}{\sin\beta} - \mu(\tfrac{\alpha}{\tan\alpha} + \tfrac{\beta}{\tan\beta}) \\
&= n\tfrac{\alpha}{\sin\alpha} + \mu\tfrac{\beta}{\sin\beta} - \mu(\tfrac{\alpha}{\tan\alpha} + \tfrac{\beta}{\tan\beta}) + \delta\tfrac{\alpha}{\sin\alpha} - R^\mu_{\tilde{n}}\delta\tfrac{\beta}{\sin\beta}\,,
\end{aligned} \tag{2.74}$$

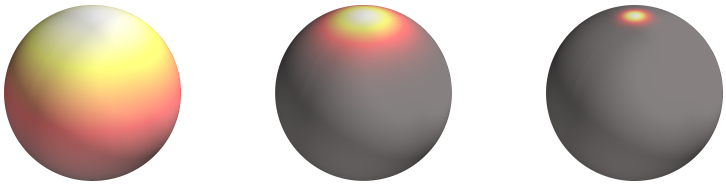
where we have used $n + \delta = \tilde{n}$. Note that $\|\delta\|_2 < \beta$. For small β, $\alpha \approx \theta$ and $\|\delta\|_2 \approx \beta$. Using $\frac{\beta}{\sin\beta} = 1 + \frac{x^2}{6} + O(x^4)$ and $\frac{\beta}{\tan\beta} = 1 - \frac{x^2}{3} - O(x^4)$, we can simplify for small β:

$$\begin{aligned}
\mathrm{Log}_\mu(\tilde{n}) + R\mathrm{Log}_{\tilde{n}}(n) &\approx n\tfrac{\theta}{\sin\theta} + \mu - \mu(\tfrac{\theta}{\tan\theta} + 1) + \delta\tfrac{\theta}{\sin\theta} - R^\mu_{\tilde{n}}\delta \\
&= \mathrm{Log}_\mu(n) + \delta\tfrac{\theta}{\sin\theta} - R^\mu_{\tilde{n}}\delta\,.
\end{aligned} \tag{2.75}$$

Inspecting the remaining approximation error $\delta\frac{\theta}{\sin\theta} - R^\mu_{\tilde{n}}\delta$, we see that the norm of the error is bounded as:

$$\|\delta\tfrac{\theta}{\sin\theta} - R^\mu_{\tilde{n}}\delta\|_2 < \|\delta\tfrac{\theta}{\sin\theta}\|_2 + \|R^\mu_{\tilde{n}}\delta\|_2 = \left(1 + \tfrac{\theta}{\sin\theta}\right)\|\delta\|_2 < \left(1 + \tfrac{\theta}{\sin\theta}\right)\beta\,. \tag{2.76}$$

Clearly, for μ and $\tilde{n}$ pointing in opposite directions ($\theta \to \pi$) the approximation is arbitrarily bad since $\lim_{\theta\to\pi}\frac{\theta}{\sin\theta} \to \infty$ (unless $\|\delta\|_2 = \beta = 0$). For μ less than orthogonal



(a) $\tau = 1$ (b) $\tau = 10$ (c) $\tau = 100$

Figure 2.9: The von-Mises-Fisher distributions on the unit sphere in 3D, $\mathbb{S}^2$, with mean at the north pole and concentrations τ. The color encodes the probability density function value of the vMF over the whole sphere. From the coloring it can be observed that the von-Mises-Fisher distribution is isotropic.

to $\tilde{n}$ ($\theta < \pi$), $\frac{\theta}{\sin\theta} < \theta$ and thus the error norm is bounded by $(1+\theta)\beta$. Hence, for small θ the approximation error vanishes.

Fig. 2.8 shows the norm of the approximation error as a function of α and β. In the experiment μ was chosen to be the north pole, $\tilde{n}$ to be at elevation α and n to be at elevation α and azimuth β. This simulates the worst case scenario where the two logarithm maps are at orthogonal angles. It is evident that the approximation is close to the true log-map value for $\tilde{n}$ in the proximity of μ (small α). We will see that in the inference procedure proposed in Sec. 4.3.2 this is automatically enforced because $\tilde{n}$ will be treated as the sufficient statistic for the mean μ of a cluster. So in general μ will be sampled close to $\tilde{n}$. Additionally, the data clusters from which $\tilde{n}$ will be computed in practice tends to be concentrated, making β small as well.

■ 2.6.3 The von-Mises-Fisher Distribution

The von-Mises-Fisher distribution is commonly used [14, 15, 92, 222] and can be regarded as akin to the isotropic Gaussian distribution of the sphere. It is parametrized by a mean direction $\mu \in \mathbb{R}^D$ and a concentration $\tau > 0$ (see Fig. 2.9). Its density is defined as [15]

$$\begin{aligned} \mathrm{vMF}(n;\mu,\tau) &= Z(\tau)\exp(\tau\mu^T n) \\ Z(\tau) &= (2\pi)^{-D/2}\frac{\tau^{D/2-1}}{\mathrm{I}_{D/2-1}(\tau)}\,, \end{aligned} \tag{2.77}$$

where I_ν is the modified Bessel function [1] of the first kind of order ν. Figure 2.9 illustrates the vMF distribution in 3D for different concentration parameters. In $D = 3$ dimensions, with $\frac{\tau^{1/2}}{\mathrm{I}_{1/2}(\tau)} = \sqrt{\frac{\pi}{2}}\frac{\tau}{\sinh(\tau)}$ and $\sinh\tau = \frac{\exp\tau - \exp(-\tau)}{2}$, the normalizer of the vMF distribution simplifies to

$$Z(\tau) = \frac{\tau}{4\pi\sinh(\tau)} = \frac{\tau}{2\pi(\exp\tau - \exp(-\tau))}\,. \tag{2.78}$$

A numerically more stable way of writing the vMF distribution in 3D is

$$\mathrm{vMF}(n;\mu,\tau) = \frac{\tau \exp\left(\tau\left(\mu^T n - 1\right)\right)}{2\pi\left(1-\exp(-2\tau)\right)} . \tag{2.79}$$

Sampling An efficient approach for sampling from a vMF distribution was describe by Ulrich [240]. While the approach works for any dimension we describe it for $D = 3$ dimensions for clarity reasons. To sample a random vector from a vMF distribution with mode $m = (0, 0, 1)$ first sample the two variables u and v:

$$v \sim \mathrm{Unif}(\mathbb{S}^{D-2}) \tag{2.80}$$

$$u \sim p(u;\tau) = \frac{\tau}{2\sinh\tau}\exp(\tau u) \tag{2.81}$$

and then compute the vMF distributed sample n as

$$n = \begin{pmatrix}\sqrt{1-u^2}v & u\end{pmatrix} . \tag{2.82}$$

In practice we obtain v by sampling from a zero-mean isotropic Gaussian with unit variance and normalizing the resulting sample to unit length. The inversion method is used to sample u (see general treatment in Sec. 2.1.2 of [198]): With the cumulative density of u, $F(u)$, derived from $p(u;\tau)$

$$\xi \sim \mathrm{Unif}(0,1) \tag{2.83}$$

$$u = F^{-1}(\xi) = 1 + \tau^{-1}\log\left(\xi + (1-\xi)\exp(-2\tau)\right) . \tag{2.84}$$

Finally, we rotate the sampled vector n from m to μ via the rotation ${}^{\mu}R_m$ which can be computed from axis $w = m \times \mu$ and angle $\theta = \arccos(\mu^T m)$ as outlined in Sec. 2.7.4. This makes the overall sampling of a vMF distributed data point very efficient for $D = 3$ dimensions and the computational complexity independent of the concentration τ (which rejection sampling is not for example).

Conjugate prior of μ given τ For Bayesian inference it is convenient to have conjugate priors for the parameters of a distribution because posterior distributions remain in the same class as the prior distribution. For the vMF distribution mean parameter μ the conjugate prior, given a fixed τ, is a vMF distribution $\mathrm{vMF}(\mu;\mu_0,\tau_0)$ [180]. Note that setting τ_0 to 0 amounts to assuming an uniform prior distribution for the mean μ. The corresponding posterior given data $\mathbf{n} = \{n_i\}_{i=1}^N$ is

$$\begin{aligned} p(\mu \mid \mathbf{n};\tau,\mu_0,\tau_0) &\propto \mathrm{vMF}(\mu;\mu_0,\tau_0)\prod_{i=1}^{N}\mathrm{vMF}(n_i;\mu,\tau) \\ &= Z(\tau_0)Z(\tau)^N \exp\left(\mu^T\left(\tau_0\mu_0 + \tau\sum_{i=1}^{N} n_i\right)\right) . \end{aligned} \tag{2.85}$$

The last expression has the form of a vMF distribution in μ and thus:

$$p(\mu \mid \mathbf{n}; \tau, \mu_0, \tau_0) = \text{vMF}\left(\mu; \tfrac{\vartheta_N}{\|\vartheta_N\|_2}, \|\vartheta_N\|_2\right) , \tag{2.86}$$

where $\vartheta_N = \tau_0\mu_0 + \tau\sum_{i=1}^N n_i$. Under the conjugate prior for μ we can compute the marginalization in closed form as:

$$\begin{aligned} p(n; \tau, \mu_0, \tau_0) &= \int_{\mathbb{S}^{D-1}} \text{vMF}(n; \mu, \tau)\, \text{vMF}(\mu; \mu_0, \tau_0)\, \mathrm{d}\mu \\ &= Z(\tau)Z(\tau_0) \int_{\mathbb{S}^{D-1}} \exp\left(\mu^T(\tau n + \tau_0\mu_0)\right) \mathrm{d}\mu \\ &= Z(\tau)Z(\tau_0) \int_{\mathbb{S}^{D-1}} \exp\left(\|\vartheta_1\|_2 \mu^T \tfrac{\vartheta_1}{\|\vartheta_1\|_2}\right) \mathrm{d}\mu \\ &= \frac{Z(\tau)Z(\tau_0)}{Z(\|\vartheta_1\|_2)} = \frac{Z(\tau)Z(\tau_0)}{Z(\|\tau n + \tau_0\mu_0\|_2)} , \end{aligned} \tag{2.87}$$

where $\vartheta_1 = \tau n + \tau_0\mu_0$ as introduced in Eq. (2.86).

Joint conjugate prior for μ and τ There also exists a conjugate prior distribution for the mean μ and the concentration parameter τ which unfortunately is only known up to proportionality [180]:

$$p(\mu, \tau \mid \mu_0, a, b) \propto \left(\frac{\tau^{D/2-1}}{\mathrm{I}_{D/2-1}(\tau)}\right)^a \exp\left(b\tau\mu^T\mu_0\right) , \tag{2.88}$$

where $0 < b < a$. The normalizing constant can only be computed analytically in special cases. Knowing this prior only up to proportionality still allows sampling from it for sampling-based inference. The posterior given observed data $\{n_i\}_{i=1}^N$ is

$$p(\mu, \tau \mid \{n_i\}_{i=1}^N, \mu_0, a, b) \propto \prod_{i=1}^N Z(\tau)\exp(n_i^T\mu\tau) \left(\frac{\tau^{D/2-1}}{\mathrm{I}_{D/2-1}(\tau)}\right)^a \exp\left(b\tau\mu^T\mu_0\right) \tag{2.89}$$

$$\propto \left(\frac{\tau^{D/2-1}}{\mathrm{I}_{D/2-1}(\tau)}\right)^{a+N} \exp\left(\tau\mu^T\left(\sum_{i=1}^N n_i + b\mu_0\right)\right) \tag{2.90}$$

$$= \left(\frac{\tau^{D/2-1}}{\mathrm{I}_{D/2-1}(\tau)}\right)^{a_N} \exp\left(\tau b_N\mu^T\mu_N\right) , \tag{2.91}$$

where the posterior parameters are

$$a_N = a + N\,, \qquad b_N = \|\vartheta\|_2\,, \qquad \mu_N = \lceil\vartheta\rceil\,, \qquad \vartheta = \sum_{i=1}^N n_i + b\mu_0\,. \tag{2.92}$$

Observe that $0 < b_N < a_N$ because of $0 < b < a$. This shows that a acts similar to the pseudo counts ν and κ of the normal inverse Wishart distribution introduced in Sec. 2.3.3.

Sampling from the joint prior One way to sample from this prior distribution is using Gibbs sampling (see Sec. 2.2.2):

$$\mu \sim p(\mu|\tau;\mu_0,a,b) \propto \text{vMF}(\mu;\mu_0,b\tau) \tag{2.93}$$
$$\tau \sim p(\tau|\mu;\mu_0,a,b)\,. \tag{2.94}$$

Sampling μ amounts to sampling from a vMF distribution as described earlier in this section, while sampling from the conditional distribution of τ needs special care. Inversion sampling is not applicable since the cumulative density could not be inverted. Instead, we choose to use a slice sampler [175], an efficient sampling strategy for low-dimensional distributions (see also Sec. 2.2.4). Since the distribution is unimodal (see Fig. 2.10 and Appendix A.2 for a full characterization), a slice sampler can be implemented more efficiently than standard more universal algorithms explored in [175]. In the following we use $f(\tau) \propto p(\tau|\mu;\mu_0,a,b)$.

Require: $f(\tau) \propto p(\tau|\mu;\mu_0,a,b)$ and τ_0
Find maximum $\tau^\star$ of $f(\tau)$
Initialize $\tau = \tau_0$
while more samples desired **do**
  Sample $u \sim \text{Unif}(0, f(\tau))$
  if $\tau^\star > 0$ **then**
    Find left slice border τ_L of $f(\tau)$ using Newton starting from $10^{-3}\tau^\star$
    Find right slice border τ_R of $f(\tau)$ using Newton starting from $1.5\tau^\star$
  else
    $\tau_L = 0$
    Find right slice border τ_R of $f(\tau)$ using Newton starting from 0.5
  end if
  Sample $\tau \sim \text{Unif}(\tau_L, \tau_R)$
end while

Algorithm 5: Slice sampler for the prior distribution on the von-Mises-Fisher concentration τ.

The slice sampler outlined in Alg. 5 alternates between sampling u uniformly from 0 to $f(\tau)$ and sampling τ uniformly from the set $\mathcal{T} = \{\tau : f(\tau) \geq u\}$. As discussed in depth in Appendix A.2, there are two cases for the set $\mathcal{T}$: either the maximum is attained at $\tau^\star = 0$ and the function decreases for $\tau > 0$ or the maximum is attained for some $\tau^\star > 0$ and the function increases for $\tau < \tau^\star$ and decreases for $\tau > \tau^\star$. In the first case $\mathcal{T}$ is the set from 0 to $f(\tau) = u$, which can be found efficiently using Newton's method starting from some arbitrary small $\tau_R^0 = 0.5$. In the second case $\mathcal{T}$ is set from τ_L to τ_R, where τ_L and τ_R are the locations of $f(\tau) = u$ left and right of the maximum. The two intersection points can be found by running Newton's algorithm from sufficiently far left/right of the maximum $\tau^\star$. In practice we start Newton's method from $\tau_L^0 = 10^{-3}\tau^\star$ to obtain the left intersection point and from $\tau_R^0 = 1.5\tau^\star$ to obtain

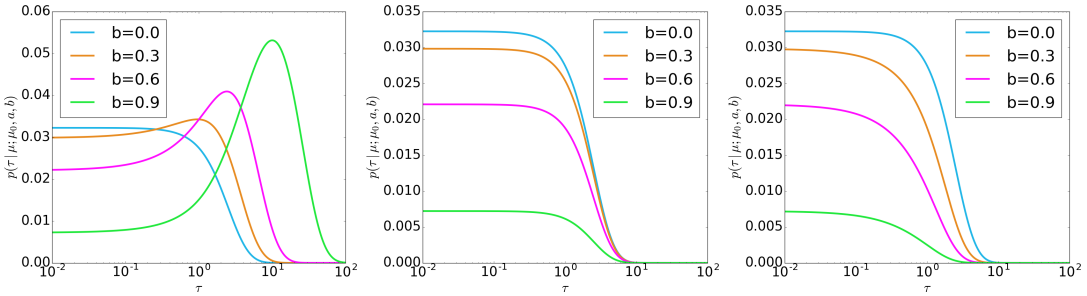


Figure 2.10: The conjugate prior for the parameters of a von-Mises-Fisher distribution is depicted for different values of b and dot-product $\mu^T\mu_0$. From left to right dot products of 1 (μ and μ_0 are equal), 0 (μ and μ_0 are orthogonal), and -1 (μ and μ_0 pointing in opposite directions) are shown. Hyper-parameters b are sampled in the allowed interval from 0 to 1.

the right intersection point. Starting closer to $\tau^\star$ leads to numerical problems. In all instances Newton's method converges in less than 10 iterations.

Normalization of joint prior for $a = 1$ and $D = 3$ While we can directly sample from the joint prior using the aforementioned method, we do need a parametric normalized form for the evaluation of the marginal data distribution for Bayesian nonparametric inference. The problem with finding a normalizer for the prior is largely due to the exponentiation with a. Setting $a = 1$ and working in $D = 3$ dimensions, we can derive a closed form normalizer as shown in Appendix A.3. Setting $a = 1$ amounts to assuming a weak prior since a can be thought of as pseudo counts as discussed in relation to the posterior parameter updates in Eq. (2.92). Using a weak prior is a common practice if there are is no strong prior information about the distribution of the parameters. With this the properly normalized prior for μ and τ is

$$p(\mu, \tau \mid \mu_0, 1, b) = \frac{b\tau}{2\pi^2} \frac{\exp\left(b\tau\mu^T\mu_0\right)}{\tan\left(\frac{b\pi}{2}\right)\sinh(\tau)} \,. \tag{2.95}$$

Slices of the prior density are plotted for aligned μ and μ_0 ($\mu^T\mu_0 = 1$), orthogonal μ and μ_0 ($\mu^T\mu_0 = 0$), and flipped μ and μ_0 ($\mu^T\mu_0 = -1$) and different b between 0 and 1 in Fig. 2.10. They show that the prior encourages a low concentration for unaligned μ and μ_0. Only once μ and μ_0 become aligned, the prior encourages higher concentrations. The magnitude of the most likely concentration then increases with b as can be seen from the left-most plot in Fig. 2.10.

Marginal data distribution For $D = 3$ dimensions and $a = 1$ we can derive a closed form normalized probability density function for the marginal distribution of the data under

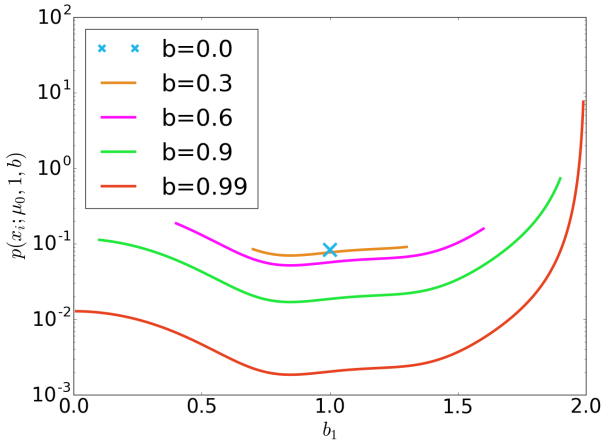


Figure 2.11: The marginal distribution of a von-Mises-Fisher distributed data point under the conjugate prior for $b_1 = \|x_i + b\mu_0\|_2$. Since a given value b restricts the range of $b_1 = \|x_i + b\mu_0\|_2$ the plots all have a different support.

the prior:

$$p(x_i;\mu_0,1,b) = \int_0^\infty \int_{\mathbb{S}^2} \mathrm{vMF}(x_i;\mu,\tau)p(\mu,\tau;\mu_0,1,b)\,\mathrm{d}\mu\,\mathrm{d}\tau \tag{2.96}$$

$$= \frac{b}{2^3 \tan\left(\frac{b\pi}{2}\right)} \frac{1-\mathrm{sinc}(b_1\pi)}{\sin^2(\frac{b_1\pi}{2})}\,, \tag{2.97}$$

where $0 < b_1 = \|x_i + b\mu_0\|_2 < 2$. For the full derivation refer to Appendix A.4. This marginal distribution is displayed as a function of b_1 for several b values in Fig. 2.11. Since a given value b restricts the range of $b_1 = \|x_i + b\mu_0\|_2$ the plots all have a different support. For $b = 0$ the prior is uniform over the sphere, $b_1 = 1$ for all x_i and we therefore expect $p(x_i;\mu_0,1,0)$ to be equal to one over the area of the sphere $\mathbb{S}^2$ which is indeed the case.

Cumulative density The cumulative density function (cdf) of the radially symmetric vMF distribution is the probability that the angle between the mode μ and a data point n is smaller than α. Working in spherical coordinates and arbitrarily fixing $\mu = (0,0,1)$

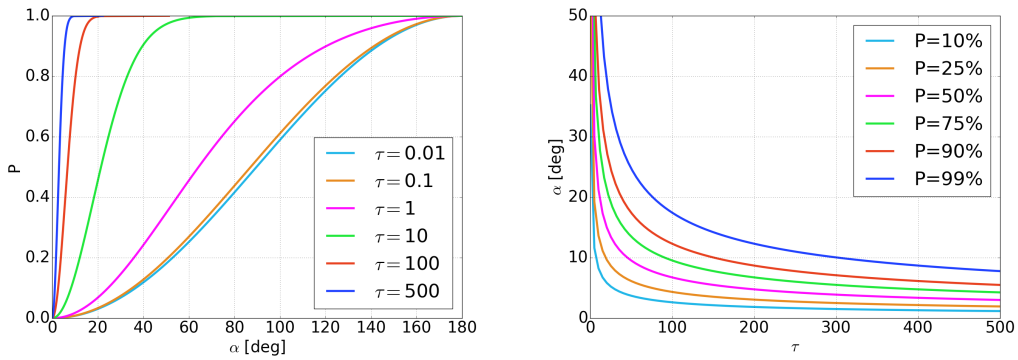


Figure 2.12: The left plot shows the cumulative density P of the vMF distribution as a function of the concentration τ depending on desired percentile P. On the right, the blue line of $P = 99\%$ indicates that a concentration of at least 300 leads to 99% of the probability mass to be concentrated within a solid angle of $\alpha = 10°$ around the mode. This is akin to the 3σ rule of the Gaussian distribution.

the cdf is:

$$\begin{aligned} P\left[\arccos(\mu^T n) < \alpha\right] &= \int_0^{2\pi}\int_0^{\alpha} Z(\tau)\exp(\tau\cos\phi)\sin\phi\,\mathrm{d}\phi\,\mathrm{d}\theta \\ &= 2\pi\int_0^{\alpha} Z(\tau)\exp(\tau\cos\phi)\sin\phi\,\mathrm{d}\phi \\ &= \frac{2\pi Z(\tau)}{\tau}\left(\exp\tau - \exp\left(\tau\cos\alpha\right)\right) \\ &= \frac{\exp\tau - \exp\left(\tau\cos\alpha\right)}{\exp\tau - \exp(-\tau)} \\ &= \frac{1-\exp\left(\tau(\cos\alpha - 1)\right)}{1-\exp(-2\tau)}\,. \end{aligned} \tag{2.98}$$

The cdf is shown in Fig. 2.12 to the left for different concentrations τ. The plot to the right shows the solid angle α as a function of concentration and probability P. The blue line for $P = 99\%$ shows the equivalent to the 3σ rule for the Gaussian distribution: for a concentration $\tau = 100$ 99% of the probability mass is within a solid angle of approximately $\alpha = 18°$. To get to a probability mass of 99% inside a solid angle of $10°$ a concentration of $\tau = 300$ is needed. Such intuitions are useful when judging an inferred τ or choosing a fixed concentration.

Maximum Likelihood Estimate Parameters μ and τ in 3D For ML estimation of the von-Mises-Fisher parameters it will be convenient to work in log scale. The log likelihood

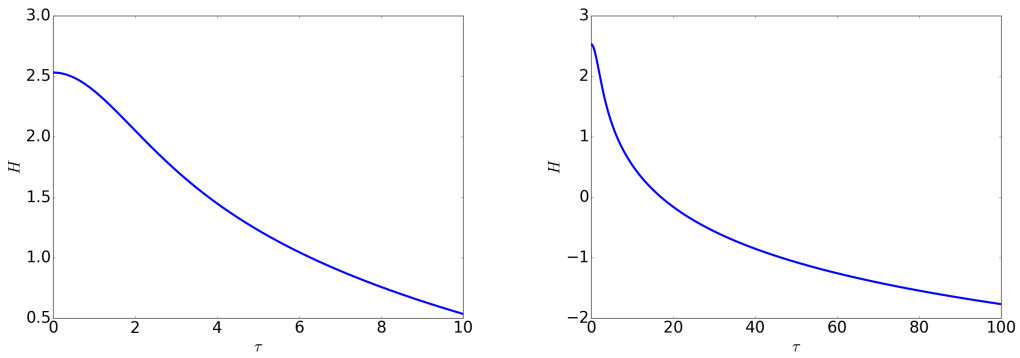


Figure 2.13: Entropy of the vMF distribution on $\mathbb{S}^2$ as a function of the concentration τ. The plot to the left is zoomed in to the range of $\tau \in [0, 10]$ whereas the plot to the right shows a larger range $\tau \in [0, 100]$.

of a set of data $\{x_i\}_{i=1}^N$ is:

$$\log p(\{n_i\} \mid \mu, \tau) = \sum_i \log \mathrm{vMF}(n_i \mid \mu, \tau) \tag{2.99}$$

$$= \tau \mu^T \sum_i n_i + N \log \tau - N \log \left(2\pi(\exp \tau - \exp(-\tau))\right) . \tag{2.100}$$

The ML estimate for μ can directly be read of: independent of the concentration τ the maximum with respect to the mode μ is attained if $\mu \in \mathbb{S}^2$ is directionally aligned with the sum over data vectors:

$$\mu^\star = \left\lceil \textstyle\sum_{i=1}^N n_i \right\rfloor . \tag{2.101}$$

To derive the maximum likelihood estimator for the concentration τ we set the derivative to 0:

$$\frac{\partial}{\partial \tau} \log p(\{n_i\} \mid \mu, \tau) = 0 \tag{2.102}$$

$$\tfrac{1}{N} \mu^T \textstyle\sum_i n_i + \frac{1}{\tau} - \frac{1+\exp(-2\tau))}{1-\exp(-2\tau)} = 0 . \tag{2.103}$$

Since no closed form solution can be found we resort to Newton's method to efficiently obtain the ML estimate for the concentration τ. The thus obtained extremum is indeed a maximum since the second derivative of the log likelihood is always negative:

$$\frac{\partial^2}{\partial \tau^2} \log p(\{n_i\} \mid \mu, \tau) = -\tfrac{1}{\tau^2} + \tfrac{4 \exp(2\tau))}{(\exp(2\tau)-1)^2} < 0 . \tag{2.104}$$

Entropy The entropy of a von-Mises-Fisher distribution in 3D is computed as:

$$
\begin{aligned}
H &= -\int_{x\in\mathbb{S}^2} \mathrm{vMF}(x;\mu,\tau)\log \mathrm{vMF}(x;\mu,\tau)\,\mathrm{d}x && (2.105)\\
&= -\int_{x\in\mathbb{S}^2} \mathrm{vMF}(x;\mu,\tau)(\log Z(\tau) + x^T\mu\tau)\,\mathrm{d}x && (2.106)\\
&= -\log(Z(\tau)) - \tau Z(\tau)\int_{x\in\mathbb{S}^2} \exp(x^T\mu\tau)x^T\mu\,\mathrm{d}x && (2.107)\\
&= -\log(Z(\tau)) - \tau Z(\tau)\int_0^{\pi}\int_0^{2\pi} \exp(\tau\cos\phi)\cos\phi\sin\phi\,\mathrm{d}\theta\,\mathrm{d}\phi && (2.108)\\
&= -\log(Z(\tau)) - 2\pi\tau Z(\tau)\int_0^{\pi} \exp(\tau\cos\phi)\cos\phi\sin\phi\,\mathrm{d}\phi && (2.109)\\
&= -\log(Z(\tau)) - 2\pi\tau Z(\tau)\frac{2\tau\cosh\tau - 2\sinh\tau}{\tau^2} && (2.110)\\
&= -\log\left(\frac{\tau}{4\pi\sinh\tau}\right) - 2\pi\frac{\tau^2}{4\pi\sinh\tau}\frac{2\tau\cosh\tau - 2\sinh\tau}{\tau^2} && (2.111)\\
&= -\log\left(\frac{\tau}{4\pi\sinh\tau}\right) - \frac{2\tau\cosh\tau - 2\sinh\tau}{2\sinh\tau} && (2.112)\\
&= -\log\left(\frac{\tau}{4\pi\sinh\tau}\right) - \frac{\tau}{\tanh\tau} + 1\,, && (2.113)
\end{aligned}
$$

where we have used $\mu = (0,0,1)$ without loss of generality (the integral and therefore the entropy is invariant to position of μ). At $\tau = 0$ the vMF distribution is uniform over the sphere. Hence its entropy is equivalent to the entropy of a uniform distribution over $\mathbb{S}^2$ which is $\log(4\pi) \approx= 2.53$, as can be verified in Fig. 2.13.

■ 2.7 Rigid-body Transformations

Rigid-body transformations are used to describe the pose of a perception system with respect to some world coordinate system. Rigid body transformation matrices lie on a manifold, the so called Special Euclidean group in 3D, $\mathbb{SE}(3)$.

A transformation matrix $T \in \mathbb{R}^{4\times4}$ is on the manifold if it has the structure:

$$
T = \begin{pmatrix} R & t \\ 0^T & 1 \end{pmatrix} \quad R \in \mathbb{SO}(3)\,, t \in \mathbb{R}^3\,, \tag{2.114}
$$

where R is a 3×3 rotation matrix in the Special Orthogonal group, $\mathbb{SO}(3)$, and t is a translation vector in $\mathbb{R}^3$. See Sec. 2.7.1 for details about the space of rotation matrices. A transformation takes points in one coordinate system to another. To be explicit about the starting and destination coordinate system, we use the following notation:

$$
{}^{\text{to}}T_{\text{from}}\,. \tag{2.115}
$$

Using homogeneous coordinates we can transform a point $p_A \in \mathbb{R}^3$ in coordinate system A to the coordinate system B as:

$$\begin{pmatrix} p_B \\ 1 \end{pmatrix} = {}^B T_A \begin{pmatrix} p_A \\ 1 \end{pmatrix} . \tag{2.116}$$

For notational brevity we sometimes implicitly assume homogeneous coordinates without explicitly denoting it. This is equivalent to directly writing out the transformation in terms of rotation ${}^B R_A$ and translation ${}^B t_A$:

$$p_B = {}^B R_A p_A + {}^B t_A . \tag{2.117}$$

This explicit notation suggests that the rotation matrix ${}^B R_A$ could be replaced by any other mechanism for rotating p_A appropriately. Indeed, another way of describing rotations that can be used to rotate points are unit Quaternions as described in Sec. 2.7.4.

We first introduce the Special Orthogonal group of rotation matrices, $\mathbb{SO}(3)$, before focusing on full rigid-body transformations, i.e. matrices on the Special Euclidean group, $\mathbb{SE}(3)$. After the introduction of the special Euclidean group we show two approaches for optimizing functions over $\mathbb{SO}(3)$ and $\mathbb{SE}(3)$ in Sec. 2.7.3. We complete the background survey of rotations by describing two alternate ways of describing rotations: unit Quaternions and axis and angle in Sec. 2.7.4. We conclude this background section by introducing two distributions over the space of rotations in Sec. 2.7.5.

■ 2.7.1 The Special Orthogonal Group $\mathbb{SO}(3)$

A general rotation matrix $R \in \mathbb{R}^{D \times D}$ is orthonormal and has unit determinant. The set of all such matrices is the Special Orthogonal Group in D dimensions and is denoted $\mathbb{SO}(D)$. It is formally defined as:

$$\mathbb{SO}(D) = \left\{ R \in \mathbb{R}^{D \times D} \ : \ R^T R = \mathrm{I} , \quad \det(R) = 1 \right\} . \tag{2.118}$$

Here we focus on 3-dimensional rotations as this thesis is concerned with transformations and rotations of 3D perception systems. As introduced in the beginning of this section, we will use the following notation to denote a rotation from coordinate frame A to B:

$${}^B R_A = {}^{\text{to}} R_{\text{from}} . \tag{2.119}$$

The inverse rotation is simply the transpose of the rotation matrix:

$${}^A R_B = {}^B R_A^T . \tag{2.120}$$

Before describing distributions or optimization strategies over $\mathbb{SO}(3)$, we introduce some tools for working with $\mathbb{SO}(3)$. The Lie algebra $\mathfrak{so}(3)$ can be understood as a linearization of the rotation manifold around the identity rotation. This linearization

is key for defining distributions over the rotation space and for deriving optimization algorithms that directly optimize on $\mathbb{SO}(3)$. Next we introduce mappings from $\mathbb{SO}(3)$ to $\mathfrak{so}(3)$ and vice versa.

The logarithm map of $R \in \mathbb{SO}(3)$ into the Lie algebra $\mathfrak{so}(3)$, $\mathrm{Log}(R) : \mathbb{SO}(3) \rightarrow \mathfrak{so}(3)$, is defined as

$$\mathrm{Log}(R) = \tfrac{\theta}{2\sin(\theta)} \left(R - R^T\right) \tag{2.121}$$

$$\theta = \arccos \tfrac{1}{2}(\mathrm{trace}(R) - 1)\,. \tag{2.122}$$

Elements of the Lie algebra are all skew-symmetric matrices W in $\mathbb{R}^{3\times 3}$. They can be represented as

$$\mathrm{Log}(R) = W = [\omega]_\times = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} = \omega_1 G_1 + \omega_2 G_2 + \omega_3 G_3 \tag{2.123}$$

$$= \omega_1 \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{bmatrix} + \omega_2 \begin{bmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{bmatrix} + \omega_3 \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}. \tag{2.124}$$

The matrices G_1, G_2 and G_3 are the so called bases or generators of $\mathfrak{so}(3)$. The *vee operator* $^\vee$ [45] extracts the three unique elements of $W = [\omega]_\times$ into a vector ω:

$$W^\vee = \omega = \begin{bmatrix} W_{32} \\ W_{13} \\ W_{21} \end{bmatrix} \in \mathbb{R}^3\,. \tag{2.125}$$

The exponential map $\mathrm{Exp} : \mathfrak{so}(3) \rightarrow \mathbb{SO}(3)$ maps an element of the Lie algebra back onto the manifold of rotations. It is indeed equivalent to the matrix exponential. Since $W \in \mathfrak{so}(3)$ are skew symmetric matrices the exponential map can be computed in closed form. The formula is also called Rodrigues' formula [5, 199]:

$$\mathrm{Exp}(W) = \mathrm{I} + \tfrac{\sin(\theta)}{\theta} W + \tfrac{1-\cos(\theta)}{\theta^2} W^2 \tag{2.126}$$

$$\theta = ||W^\vee||_2\,. \tag{2.127}$$

Sometimes we will, for the sake of notational simplicity, omit the explicit $[\cdot]_\times$ and directly write

$$\mathrm{Exp}(\omega) = \mathrm{Exp}([\omega]_\times)\,. \tag{2.128}$$

The exponential and logarithm map can be understood as mapping between the rotation manifold and its tangent space around the identity rotation. If we want to map a rotation into a different tangent space around R_0 we simply compute:

$$\mathrm{Exp}_{R_0}(W) = R_0 \mathrm{Exp}(W) \tag{2.129}$$

$$\mathrm{Log}_{R_0}(R) = R_0 \mathrm{Log}(R_0^T R) \tag{2.130}$$

$$\theta = \arccos \tfrac{1}{2}(\mathrm{trace}(R_0^T R) - 1)\,. \tag{2.131}$$

For further details on the manifold of rotations $\mathbb{SO}(3)$, the logarithm and exponential maps, and the relation to the Lie Algebra $\mathfrak{so}(3)$, refer to [45, 63].

We now focus on how to take derivatives over functions on $\mathbb{SO}(3)$. To take a derivative of a function $f(R) : \mathbb{SO}(3) \to \mathbb{R}$ with respect to the rotation matrix R we compose R with a small rotation $\mathrm{Exp}(\omega)$, take the derivative with respect to ω, and evaluate at $\omega = 0$ [2, 65, 83]. Note that we can compose R from the left or from the right with $\mathrm{Exp}(\omega)$. Let $R = {}^{B}R_{A}$ transform from coordinate system A to B. Then if we compose from the left, the derivative will be in coordinate system B, and vice versa.

The first derivative of the exponential map with respect to ω at $\omega = 0$ is

$$\frac{\partial}{\partial \omega_i}\mathrm{Exp}(\omega)\Big|_{\omega=0} = G_i\,. \tag{2.132}$$

See Appendix A.5 for the full derivation. As derived in Appendix A.6, the second derivative of the exponential map is

$$\frac{\partial^2}{\partial \omega_j \partial \omega_i}\mathrm{Exp}(\omega)\Big|_{\omega=0} = \tfrac{1}{2}(G_i G_j + G_j G_i)\,. \tag{2.133}$$

First derivative of functions over $\mathbb{SO}(3)$ Here we focus on scalar functions of the form $f(Rx)$, i.e. functions of rotated data points x, since these are commonly used in this thesis. For a more general derivation see Appendix A.7. Using the chain rule we obtain the right derivative with respect to R as

$$\frac{\partial f(Rx)}{\partial R} = \frac{\partial f(y)}{\partial y}\Big|_{y=Rx} \frac{\partial}{\partial \omega} R\mathrm{Exp}(\omega)x\Big|_{\omega=0} \tag{2.134}$$

$$= \frac{\partial f(y)}{\partial y}\Big|_{y=Rx} R\begin{pmatrix} G_1 x & G_2 x & G_3 x \end{pmatrix} \tag{2.135}$$

$$= -\frac{\partial f(y)}{\partial y}\Big|_{y=Rx} R[x]_\times\,. \tag{2.136}$$

Analogously the left derivative is

$$\frac{\partial f(Rx)}{\partial R} = \frac{\partial f(y)}{\partial y}\Big|_{y=Rx} \frac{\partial}{\partial \omega} \mathrm{Exp}(\omega)Rx\Big|_{\omega=0} \tag{2.137}$$

$$= \frac{\partial f(y)}{\partial y}\Big|_{y=Rx} \begin{pmatrix} G_1 Rx & G_2 Rx & G_3 Rx \end{pmatrix} \tag{2.138}$$

$$= -\frac{\partial f(y)}{\partial y}\Big|_{y=Rx} [Rx]_\times\,. \tag{2.139}$$

Second derivative of functions over $\mathbb{SO}(3)$ The second derivative, the so called Hessian, can be useful for second order optimization methods such as Newton's method and to get a covariance estimate. The approach is the same as for the gradient. The Hessian

is computed as the matrix of all combinations of second derivatives

$$H = \frac{\partial^2}{\partial R^2} f(R) = \begin{pmatrix} \frac{\partial^2 f(R)}{\partial \omega_1 \partial \omega_1} & \frac{\partial^2 f(R)}{\partial \omega_1 \partial \omega_2} & \frac{\partial^2 f(R)}{\partial \omega_1 \partial \omega_3} \\ \frac{\partial^2 f(R)}{\partial \omega_2 \partial \omega_1} & \frac{\partial^2 f(R)}{\partial \omega_2 \partial \omega_2} & \frac{\partial^2 f(R)}{\partial \omega_2 \partial \omega_3} \\ \frac{\partial^2 f(R)}{\partial \omega_3 \partial \omega_1} & \frac{\partial^2 f(R)}{\partial \omega_3 \partial \omega_2} & \frac{\partial^2 f(R)}{\partial \omega_3 \partial \omega_3} \end{pmatrix}, \tag{2.140}$$

where the individual second right derivatives are computed as

$$\frac{\partial^2 f(R)}{\partial \omega_j \partial \omega_i} = \operatorname{tr} \left\{ \left(\frac{\partial}{\partial \omega_j} \frac{\partial f(R)}{\partial R} \right)^T \Big|_{\omega=0} R G_i + \frac{\partial f(R)}{\partial R}^T \tfrac{1}{2} R (G_i G_j + G_j G_i) \right\}. \tag{2.141}$$

The left derivative is:

$$\frac{\partial^2 f(R)}{\partial \omega_j \partial \omega_i} = \operatorname{tr} \left\{ \left(\frac{\partial}{\partial \omega_j} \frac{\partial f(R)}{\partial R} \right)^T \Big|_{\omega=0} G_i R + \frac{\partial f(R)}{\partial R}^T \tfrac{1}{2} (G_i G_j + G_j G_i) R \right\}. \tag{2.142}$$

See Appendix A.8 for derivations as well as detailed examples.

■ 2.7.2 The Special Euclidean Group $\mathbb{SE}(3)$

As noted in the beginning of the section, a rigid body transformation T can be used to describe the pose of a perception system and to transform points from one coordinate system to another. Transformation matrices $T \in \mathbb{R}^{4\times4}$ are elements of the Special Euclidean group $\mathbb{SE}(3)$ which has the structure:

$$\mathbb{SE}(3) = \left\{ T \in \mathbb{R}^{4\times4} \; : \; \begin{pmatrix} R & t \\ 0^T & 1 \end{pmatrix} \quad R \in \mathbb{SO}(3) \, , t \in \mathbb{R}^3 \right\}, \tag{2.143}$$

where R is a 3×3 rotation matrix of the Special Orthogonal group, $\mathbb{SO}(3)$, and t is a translation vector in $\mathbb{R}^3$.

The inverse of a transformation can be computed in closed form as

$${}^A T_B = {}^B T_A{}^{-1} = \begin{pmatrix} {}^B R_A^T & -{}^B R_A^T {}^B t_A \\ 0^T & 1 \end{pmatrix} = \begin{pmatrix} {}^A R_B & {}^A t_B \\ 0^T & 1 \end{pmatrix}. \tag{2.144}$$

Similar to the rotation space $\mathbb{SO}(3)$, the Special Euclidean group is equipped with a Lie algebra, $\mathfrak{se}(3)$, which can again be understood as a linearization around the identity transformation. The exponential map for $\mathbb{SE}(3)$, mapping from $\mathfrak{se}(3)$ to $\mathbb{SE}(3)$, can be computed in closed form as:

$$\operatorname{Exp} \begin{pmatrix} \omega_R \\ \omega_t \end{pmatrix} = \begin{pmatrix} R(\omega_R) & V(\omega_R)\omega_t \\ 0 & 1 \end{pmatrix}, \tag{2.145}$$

where $R(\omega_R)$ is equivalent to the exponential map of the rotation group:

$$R(\omega_R) = \mathrm{I} + \tfrac{\sin(\theta)}{\theta} [\omega_R]_\times + \tfrac{1-\cos(\theta)}{\theta^2} [\omega_R]_\times [\omega_R]_\times \tag{2.146}$$

$$V(\omega_R) = \mathrm{I} + \tfrac{1-\cos\theta}{\theta^2} [\omega_R]_\times + \tfrac{\theta-\sin\theta}{\theta^3} [\omega_R]_\times [\omega_R]_\times \tag{2.147}$$

$$\theta = ||\omega_R||_2 \, . \tag{2.148}$$

The logarithm map associated with $\mathbb{SE}(3)$ maps a transformation into the Lie algebra:

$$\mathrm{Log}(T) = \begin{pmatrix} \mathrm{Log}(R) \\ V^{-1}(\mathrm{Log}(R))t \end{pmatrix}, \tag{2.149}$$

where the inverse $V^{-1}(\mathrm{Log}(R))$ can be computed in terms of ω_R computed via the logarithm map of $\mathbb{SO}(3)$ as

$$\omega_R = \mathrm{Log}(R)^\vee \tag{2.150}$$

$$\theta = \|\omega_R\|_2 \tag{2.151}$$

$$V^{-1} = \mathrm{I} - \tfrac{1}{2}[\omega]_\times + \tfrac{1}{\theta^2}\left(1 - \tfrac{\theta \sin\theta}{2(1-\cos\theta)}\right)[\omega_R]_\times[\omega_R]_\times . \tag{2.152}$$

For the purpose of optimization of functions over $\mathbb{SE}(3)$ we now focus on how to take derivatives over such functions properly. Analogous to taking derivatives of functions over $\mathbb{SO}(3)$, the strategy is to left or right multiply the transformation T with $\mathrm{Exp}(\omega)$, to take the derivative with respect to ω and to evaluate the resulting function at $\omega = 0$. To this end it is necessary to analyse the behavior of $V(\omega_R)$ around $\omega_R = 0$. As derived in Appendix A.9.1, the matrix has the following behavior:

$$\lim_{\omega_R \to 0} V(\omega) = \mathrm{I} \tag{2.153}$$

$$\tfrac{\partial}{\partial \omega_i} V(\omega)\Big|_{\omega=0} = \tfrac{1}{2} G_i . \tag{2.154}$$

First derivative of functions over $\mathbb{SE}(3)$ Since we exclusively encounter derivatives of functions of transformations of 3D points x, we focus on functions of the form $f(Tx)$. Via the chain rule we obtain the right derivative as

$$\frac{\partial f(Tx)}{\partial T} = \frac{\partial f(y)}{\partial y}\Big|_{y=Tx} \frac{\partial}{\partial \omega} T\mathrm{Exp}(\omega)x\Big|_{\omega=0} = \frac{\partial f(y)}{\partial y}\Big|_{y=Tx} \begin{pmatrix} -R[p]_\times & R \end{pmatrix}, \tag{2.155}$$

where we have used

$$\frac{\partial}{\partial \omega} T\mathrm{Exp}\begin{pmatrix}\omega_R \\ \omega_t\end{pmatrix} p\Big|_{\omega=0} = \frac{\partial}{\partial \omega}\begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix}\begin{pmatrix} R(\omega_R) & V(\omega_R)\omega_t \\ 0 & 1 \end{pmatrix} p\Big|_{\omega=0} \tag{2.156}$$

$$= \frac{\partial}{\partial \omega}\left(RR(\omega_R)p + RV(\omega_R)\omega_t + t\right)\Big|_{\omega=0} \tag{2.157}$$

$$= \begin{pmatrix} -R[p]_\times & R \end{pmatrix} . \tag{2.158}$$

Some more important derivatives are

$$\frac{\partial}{\partial \omega}\mathrm{Exp}\begin{pmatrix}\omega_R \\ \omega_t\end{pmatrix} Tp = \begin{pmatrix} -[Tp]_\times & \mathrm{I} \end{pmatrix} \tag{2.159}$$

$$\frac{\partial}{\partial \omega} T\mathrm{Exp}\begin{pmatrix}\omega_R \\ \omega_t\end{pmatrix} p = \begin{pmatrix} -R[p]_\times & R \end{pmatrix} \tag{2.160}$$

$$\frac{\partial}{\partial \omega}\left(T\mathrm{Exp}\begin{pmatrix}\omega_R \\ \omega_t\end{pmatrix}\right)^{-1} p = \begin{pmatrix} \left[R^T(p-t)\right]_\times & -\mathrm{I} \end{pmatrix} . \tag{2.161}$$

See detailed derivations in the Appendix A.9.

■ 2.7.3 Optimization of Functions over Transformations

Using the properties of $\mathbb{SO}(3)$ and $\mathbb{SE}(3)$ introduced in the previous sections, we now show how to extend the classical first and second order optimization methods to optimization over the special orthogonal and the special Euclidean group. It turns out that some optimizations over the rotation group, which we consider in this thesis, can be solved globally in closed form by utilizing the solution to the orthonormal Procrustes problem as introduced in the second subsection.

First and Second order Incremental Methods

In the following we describe the general optimization of a scalar valued function defined over the manifold of transformation matrices $\mathbb{SE}(3)$. By changing the exponential maps the same approach may be taken to optimize functions defined over $\mathbb{SO}(3)$.

Let $f(T)$ be a scalar function defined over the manifold of transformations i.e. $T \in \mathbb{SE}(3)$. Gradient descent on this objective function follows the usual form but has to be adapted to move along the manifold. Specifically, we compute the gradient of $f(T)$ at the current transformation estimate T_t, move in the negative direction of the gradient in the tangent space and then map the resulting vector back onto $\mathbb{SE}(3)$:

$$T_{t+1} = \mathrm{Exp}_{T_t}\left(-\delta_t J\right) , \tag{2.162}$$

where $J = \frac{\partial f(T)}{\partial T}$ is the so called Jacobian and δ_t is the step size which may vary with iteration t.

For second order optimization, i.e. Newton optimization, we need to additionally compute the Hessian as outlined in the previous section. Given the Hessian H and the Jacobian J, we again compute a update in the tangent space around the previous rotation and map it back onto $\mathbb{SE}(3)$ via the exponential map:

$$T_{t+1} = \mathrm{Exp}_{T_t}\left(-\delta_t H^{-1} J\right) . \tag{2.163}$$

where $H = \frac{\partial^2 f(T)}{\partial T^2}$. See the excellent book by Absil et al. [2] for a comprehensive guide to optimization over manifolds.

Closed-Form Rotation Optimization via Orthonormal Procrustes

In various inference problems considered in this thesis we aim to find the maximizing rotation $R^\star$ of a cost function of the form:

$$R^\star = \underset{R \in \mathbb{SO}(3)}{\arg\max}\, \mathrm{tr}\{RN\} = V \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{pmatrix} U^T \text{ where } N = USV^T . \tag{2.164}$$

The solution can be found in closed form using the SVD of N because the problem is an instantiation of the orthonormal Procrustes problem where $N = AB^T$.

The orthonormal Procrustes problem is the problem of finding a rotation matrix that brings two matrices A and B into alignment. Originally only the constraint of

orthogonality and not orthonormality was enforced [215]. It is straightforward to extend the original problem to orthonormality [93] as we show below.

We can derive the result of the orthonormal Procrustes problem as follows. With $\|X\|_F = \sqrt{\operatorname{tr} XX^T}$ denoting the Frobenius norm:

$$R^\star = \underset{R\in\mathbb{SO}(3)}{\arg\min} \|RA - B\|_F^2 = \underset{R\in\mathbb{SO}(3)}{\arg\min} \|A\|_F^2 + \|B\|_F^2 - 2\operatorname{tr}\{RAB^T\} \tag{2.165}$$

$$= \underset{R\in\mathbb{SO}(3)}{\arg\max} \operatorname{tr}\{RAB^T\} \text{ with } AB^T = USV^T \tag{2.166}$$

$$= \underset{R\in\mathbb{SO}(3)}{\arg\max} \operatorname{tr}\{RUSV^T\} = \underset{R\in\mathbb{SO}(3)}{\arg\max} \operatorname{tr}\{V^TRUS\} \tag{2.167}$$

$$= V \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{pmatrix} U^T, \tag{2.168}$$

where the last line follows because $\operatorname{tr}\{TS\}$ is maximized for $T = \mathrm{I}$ which is attained for $R = VU^T$. The diagonal matrix between V and U^T is needed to satisfy the constraint that $R^\star$ should have determinant 1.

■ 2.7.4 Additional Rotation Representations

Besides the representation of rotations via matrices in the special orthogonal group, two alternate common representations are unit Quaternions and axis and angle. In the following we highlight these representations and show their connections with rotation matrices.

Unit Quaternions $\mathbb{S}^3$

Quaternions are 4D extensions of the imaginary numbers $q = q_w + iq_x + jq_y + kq_z$. The unit length quaternions can be used to describe 3D rotations. For our purposes we think of unit quaternions $q = (q_w, q_{xyz})$ as points lying on the sphere in 4D, $\mathbb{S}^3$. $\mathbb{S}^3$ is a double cover of the rotation space. Hence it is sufficient to only consider the upper half sphere in 4D to completely cover the space of rotations.

Quaternion rotations can be composed and can directly be used to transform 3D points [114]. A rotation matrix R in $\mathbb{SO}(3)$ can be converted into a unit quaternion q via the intermediate axis angle representation:

$$\omega = \theta w = \operatorname{Log}(R)^\vee \tag{2.169}$$

$$q = (\cos\tfrac{\theta}{2}, \sin\tfrac{\theta}{2} w) = (\cos\tfrac{\|\omega\|_2}{2}, \sin\tfrac{\|\omega\|_2}{2} \tfrac{\omega}{\|\omega\|_2}). \tag{2.170}$$

From unit quaternion to rotation matrix we use the intermediate axis angle representation as well:

$$\theta = 2\arctan\tfrac{\|q_{xyz}\|_2}{q_w} \tag{2.171}$$

$$\omega = \tfrac{q_{xyz}}{\|q_{xyz}\|_2} \tag{2.172}$$

$$R = \operatorname{Exp}([\theta\omega]_\times). \tag{2.173}$$

The advantage of quaternions is that by re-normalizing to unit length accumulated numeric errors can be eliminated. Furthermore it only takes four values to store a rotation in this representation. On the downside it takes more operations to transform a point than using rotation matrices.

Axis Angle (AA)

A 3D rotation can be uniquely described via a rotation axis w and a rotation angle θ (employing the right hand rule). The axis angle notation is relevant because it connects rotation matrices and unit quaternions:

$$[\theta w]_\times = [\omega]_\times = \mathrm{Log}(R) \in \mathfrak{so}(3) \tag{2.174}$$

$$q = (\cos\tfrac{\theta}{2}, \sin\tfrac{\theta}{2} w) = (\cos\tfrac{\|\omega\|_2}{2}, \sin\tfrac{\|\omega\|_2}{2}\tfrac{\omega}{\|\omega\|_2}) \,. \tag{2.175}$$

Axis angle rotations can neither be composed directly nor can they directly transform 3D points. They have to first be converted to either rotation matrices or unit quaternions. Since it is sufficient to store the product $\theta\omega$ it only takes three values to store a rotation in this notation.

■ 2.7.5 Distributions over $\mathbb{SO}(3)$

Probability distributions over rotation matrices can be defined by exploiting the manifold structure of $\mathbb{SO}(3)$ (e.g. [45, 187]) as introduced in the first subsection. Similar to distributions over the sphere (Sec. 2.6), there exist distributions directly defined over the rotation space such as the matrix von-Mises-Fisher distribution introduced in the second subsection.

Gaussian Distribution in $\mathfrak{so}(3)$

In particular, a way to construct the analog of a Gaussian distribution utilizes the linearity of the tangent spaces. The logarithm map allows us to define a normal distribution with mean rotation R_μ and covariance $\Sigma_{\mathfrak{so}(3)} \in \mathbb{R}^{3\times3}$ in the tangent space $T_{R_\mu}\mathbb{SO}(3)$:

$$p(R; R_\mu, \Sigma_{\mathfrak{so}(3)}) = \mathcal{N}(\mathrm{Log}_{R_\mu}(R)^\vee; 0, \Sigma_{\mathfrak{so}(3)}) \,. \tag{2.176}$$

In order to sample from the distribution in Eq. (2.176), we sample a 3D vector and map it back from the tangent space $T_{R_\mu}\mathbb{SO}(3)$ to $\mathbb{SO}(3)$ using the Riemannian exponential map:

$$\omega \sim \mathcal{N}(0, \Sigma_{\mathfrak{so}(3)}) \tag{2.177}$$

$$R = \mathrm{Exp}_{R_\mu}([\omega]_\times) \,. \tag{2.178}$$

Note that $\mathfrak{so}(3)$ is a finite space (a ball with radius π). This means that the Gaussian in $\mathfrak{so}(3)$ is not properly normalized anymore since some of its probability mass lies outside

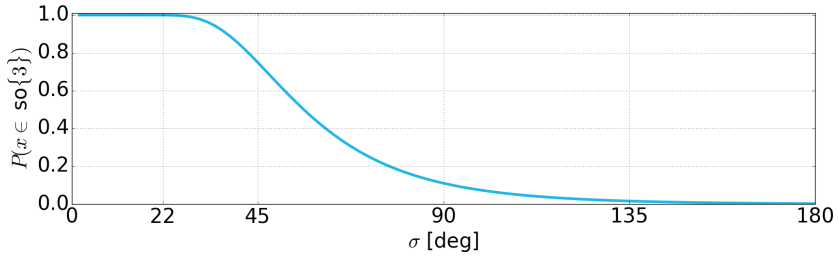


Figure 2.14: Probability that a data point lies inside $\mathfrak{so}(3)$, a 3D sphere with radius π, under a zero-mean Gaussian with covariance $\sigma^2\mathrm{I}$. For covariances $\sigma < 22°$ the distribution is approximately normalized.

$\mathfrak{so}(3)$. In practice, as long as the covariance $\Sigma_{\mathfrak{so}(3)}$ is small enough the fraction of the probability density function outside of $\mathfrak{so}(3)$ is negligible as can be observed in Fig. 2.14.

The Matrix von-Mises-Fisher Distribution

The von-Mises-Fisher distribution can also be defined for rotation matrices [32, 136] (Example 2). This is a special case for the general von Mises-distribution over elements of the Stiefel manifold [136] ($n = p$). Intuitively this is possible since every rotation can be described by a unit Quaternion as described in Sec. 2.7.4. After converting a rotation to its Quaternion representation, its distribution could hence be described by the vMF distribution over the sphere in 4D as introduced in Sec. 2.6.3. Alternatively one can directly define what is called the matrix vMF (MvMF) distribution directly over the space of rotation matrices $R \in \mathbb{SO}(3)$ as:

$$\mathrm{vMF}(R; R_\mu, S_R) = Z(S_R)^{-1} \exp(\mathrm{tr}(R^T R_\mu S_R)) \quad \text{where } S_R = \mathrm{diag}(s_0, s_1, s_2) \tag{2.179}$$

$$Z(S_R) = {}_0F_1(\tfrac{3}{2}, \tfrac{1}{4}S_R^2)\,, \tag{2.180}$$

where ${}_0F_1$ is the hypergeometric function of the matrix argument [136]. The diagonal entries in S_R, s_i, weigh the deviations along the respective axes and thus lead to a non-isotropic distribution.

Sometimes it is convenient to simply assume an isotropic distribution by letting all $s_i = \tau_R$. In this case the distribution can be simplified to [32]

$$\mathrm{vMF}(R; R_\mu, \tau_R) = Z(\tau_R)^{-1} \exp(\tau_R \,\mathrm{tr}(R_\mu^T R)) \tag{2.181}$$

$$Z(\tau_R) = \exp(\tau_R)(\mathrm{I}_0(2\tau_R) - \mathrm{I}_1(2\tau_R))\,, \tag{2.182}$$

where $\mathrm{I}_\nu(\cdot)$ is the modified Bessel function of the first kind of order ν, $R_\mu \in \mathbb{SO}(3)$ is the mode and $\tau_R \in \mathbb{R}$ is the concentration of the MvMF distribution. Rearranging the relationship between the angle θ and two rotation matrices R_μ and R defined in

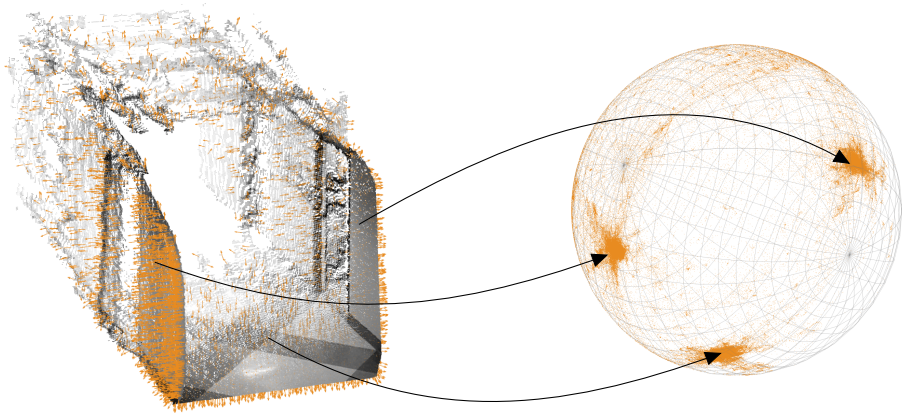


Figure 2.15: Correspondence between scene parts and surface normals.

Eq. (2.122) we can gain some intuition for this distribution:

$$\begin{aligned} \theta &= \arccos\left(\tfrac{1}{2}(\mathrm{tr}(R_\mu^T R) - 1)\right) \\ \Leftrightarrow \mathrm{tr}(R_\mu^T R) &= 2\cos(\theta) + 1\,. \end{aligned} \tag{2.183}$$

Plugging this result into Eq. (2.181) we can see the $+1$ gets absorbed into the normalizer and the factor of 2 can be absorbed into the concentration τ_R. That means that the MvMF distribution is essentially a distribution over the angle between a mean rotation and another rotation matrix.

■ 2.8 Surface Normals and Connection to the Gauss Map

In this work we will often utilize surface normals as observations of a scene as depicted in Fig. 2.15. Therefore it is important to understand the underlying theory of surfaces and normals. For a regular orientable surface S the map from the surface to the normals n to the surface is called the Gauss map [62]. Mathematically, for a parametrization $x(u, v)$ of the surface $x : U \subset \mathbb{R}^2 \to S$ the Gauss map $n : S \to \mathbb{S}^2$ at a point $x(u, v)$ on the surface is defined as

$$n(u, v) = \frac{x_u \times x_v}{\|x_u \times x_v\|_2}(u, v) = \lceil x_u \times x_v \rceil\,(u, v)\,,\ x(u, v) \in S\,, \tag{2.184}$$

where we have used the derivatives $x_u = \frac{\partial x(u,v)}{\partial u}$ and $x_v = \frac{\partial x(u,v)}{\partial v}$ and $\times$ denotes the cross-product operation and $\lceil x \rceil$ the normalization of x to unit length. In practice, when working with depth images the parameterization is given directly in terms of the location (u, v) of the depth value in the image. For point-clouds the surface normals are commonly extracted by robustly fitting a local tangent plane around the point p of interest. The normal to that tangent plane is $n(u, v)$.

Another issue in practice is that the resolution of surface observations decreases with increasing distance to the surface due to fundamental sensing methodology limitations. One way of taking this into account is the Extended Gaussian image (EGI) [116] proposed by Horn. The EGI corresponds to the image of a 3D object under the Gauss map where each surface normal is weighted by the surface area it represents. Equivalently, surface normals can be extracted at the same scale i.e. from the same constant surface area. Interestingly, for convex polyhedra the representation in terms of their surface normal distribution is unique [154] and it is possible to obtain the 3D structure solely from its surface normals.

One important property of the EGI and the surface normals of an object or a scene is that they are invariant to translations. This allows us to isolate the effects of rotations by analyzing only the surface normals. The advantages of being able to isolate rotation estimation will become clear in Chapter 5.

Surface normals are straightforward to extract from most 3D scene representations such as depth images [112], point clouds [168], triangle-meshes [31] and volumetric representations. In the following we review different surface normal extraction algorithms and their properties.

■ 2.8.1 Surface Normal Extraction Algorithms

There are various algorithms to extract surface normals from different 3D structure representations. By definition the surface normal is the vector orthogonal to the surface at a given location. In practice we are given an imperfect, noisy, not necessarily uniformly or densely sampled 3D representation of the true surface. Therefore surface normal extraction is an estimation process from noisy input data. Ideally we would rely only on the immediate neighborhood to estimate the surface normal direction but a larger neighborhood leads to more robust estimates in the face of noisy data. Therefore any practical algorithm has to either actively estimate a plane-inlier set in a larger neighborhood or choose a small neighborhood.

The two most common approaches to defining a neighborhood are via the k nearest neighbor search or via radius search in the point cloud. Both search operations are rather costly even if approximate nearest neighborhood algorithms are used [12, 171]. If we have an organized point cloud computed from a depth image, the neighborhood can be defined efficiently in terms of image coordinates. Surface normal extraction methods for point-clouds can therefore be utilized for depth images as well. The approximation of relying on the image space neighborhood eliminates the need for neighborhood searches but might lead to wrong associations at, for example, depth discontinuities. Extraction algorithms that are robust to such artifacts are important for high quality surface normal estimation.

For the different algorithms outlined in the following, Fig. 2.17 shows the performance for geometries captured by a depth camera. For each geometry, we show the zero-noise case and inverse depth noise of $\sigma = 0.01$ and $\sigma = 0.03$. Each shape, depicted in Fig. 2.16, is roughly $1m$ from the simulated depth camera.

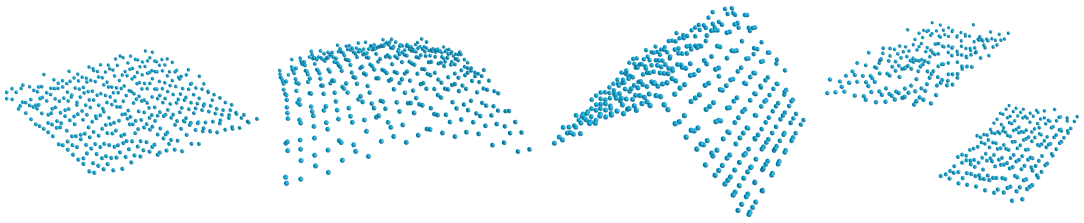


Figure 2.16: Point clouds of the different scenarios considered for surface normal extraction. From left to right: a plane, part of a sphere, a 90° corner, and a discontinuity. All point clouds contain additive Gaussian noise with a standard deviation of $\sigma = 0.01$.

There is a trade-off between speed and accuracy of the surface normal extraction algorithms. In general one should use the most accurate algorithm possible under the given time budget. For example, we use the fast extraction algorithm for the real-time Manhattan Frame algorithms described in Sec. 3.3. In Sec. 5.2, on the other hand, we utilize the slower more and accurate unconstrained least squares method to obtain high quality surface normals. In the following we first focus on accurate surface normal extraction from general point clouds, meshes and volumetric representations before deriving a fast and direct surface normal extraction method for depth images.

Least Squares Plane fitting The most commonly used approach to surface normal estimation is for a given neighborhood N_i of point x_i to seek the least squares fitting plane surface normal n_i:

$$n_i = \arg\min_{n \in \mathbb{S}^2} \sum_{j \in N_i} \|n^T p_j - a\|_2^2 \,, \tag{2.185}$$

where a is a parameter of the plane. Mitra et al. [167] show that the solution to this problem is the eigenvector of the smallest eigenvalue of the sample covariance matrix S:

$$S_i = \sum_{j \in N_i} x_j x_j^T - \tfrac{1}{N} \tilde{x} \tilde{x}^T \tag{2.186}$$

$$\tilde{x} = \sum_{j \in N_i} x_j \,, \tag{2.187}$$

where N is the number of points in the neighborhood N_i. This connection had been used previously [113] but not theoretically justified. The neighborhood can be defined in various ways as discussed previously.

Badino et al. [13] relax the constraint of $n_i \in \mathbb{S}^2$ and divide by a to arrive at

$$n_i = \arg\min_{n_i} \sum_{j \in N_i} \|n^T p_i - 1\|_2^2 \,, \tag{2.188}$$

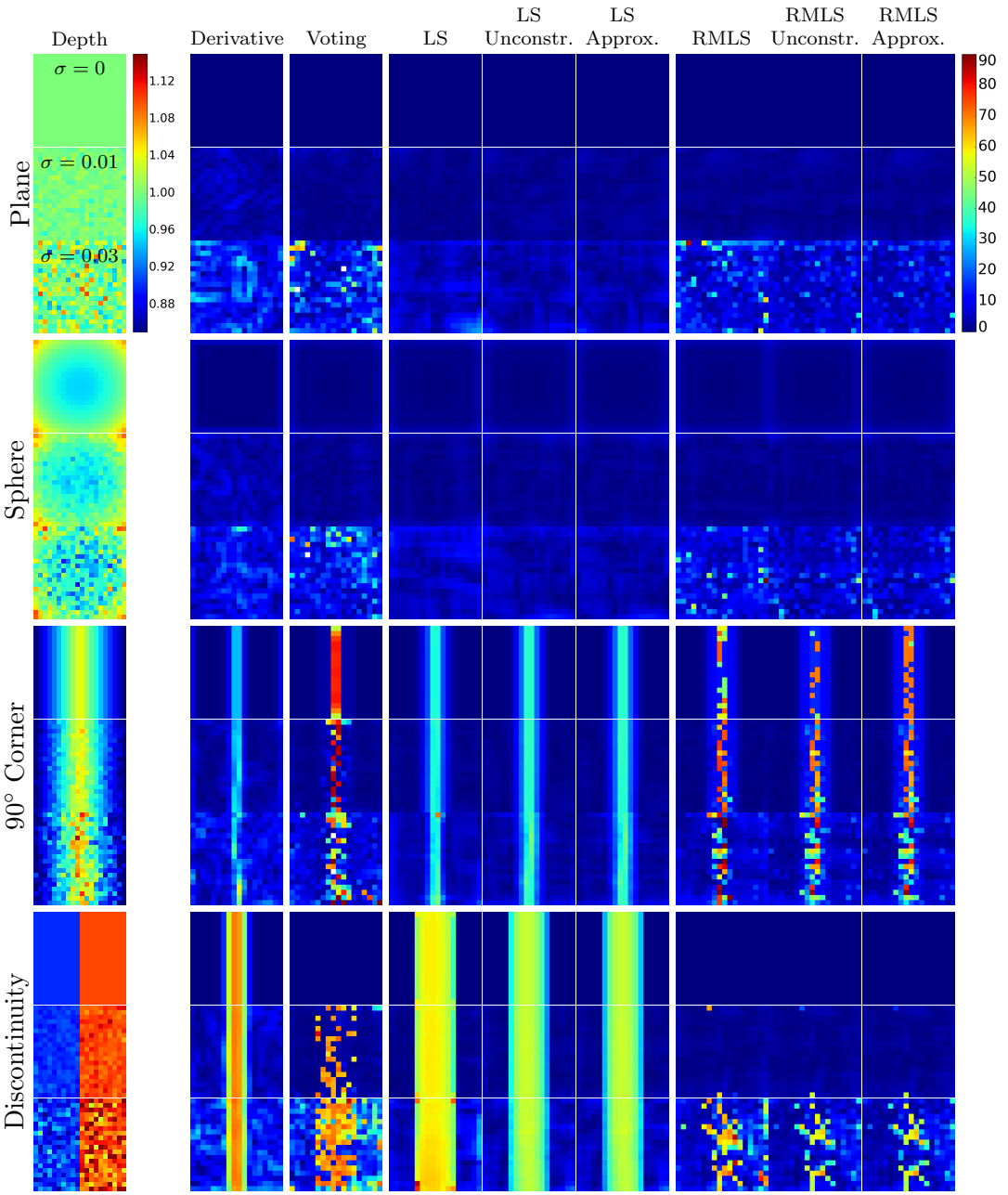


Figure 2.17: Comparison of surface normal extraction algorithms from depth images on four different types of surfaces. The left-most column shows the depth image, whereas the other columns show angular deviation between groundtruth and estimated surface normals in degree. The derivative and the least squares (LS) approaches exhibit consistent artifacts at depth discontinuities and 90° corners. Voting and robust moving least squares (RMLS) handle these challenging conditions better. Interestingly, the more efficient unconstrained and approximate RMLS perform very similarly to vanilla RMLS and are therefore the preferred method for accurate and robust surface normal extraction.

which can be solved more efficiently as:

$$n_i = \left\lceil S_i^{-1} \tilde{x} \right\rceil \,. \tag{2.189}$$

They also derive an approximate computation for organized point clouds obtained, for example, from a depth camera or a LiDAR scanner, by noting that the point x_i can be described by the product of a ray r_i with the depth d_i. The ray r_i is computed via the unprojection operation of the depth camera and can be precomputed for every ray of the sensor. With this the minimization can be approximated as

$$n_i = \arg\min_{n_i} \sum_{j \in N_i} d_j^2 \| n^T r_j - \tfrac{1}{d_j} \|_2^2 \approx \arg\min_{n_i} \sum_{j \in N_i} \| n^T r_j - \tfrac{1}{d_j} \|_2^2 \,. \tag{2.190}$$

The approximation lies in dropping the d_j^2 which is reasonable since we are assuming that the points on the plane are all roughly at the same depth. The solution to this approximate problem is:

$$n_i = \left\lceil \left(\textstyle\sum_{j \in N_i} r_j r_j^T \right)^{-1} \sum_{j \in N_i} \frac{r_j}{d_j} \right\rceil \,. \tag{2.191}$$

This approximation has been used previously by Hebert et al. [108]. Importantly the outer products and sums over rays r_i can be precomputed. Furthermore, an efficient implementation would use integral images to compute the sum over a rectangular neighborhood in constant time. Precomputing the inverse depths gives additional speedups.

We note that the constrained least square method is not robust to outliers to the plane model and tends to smooth over corners and depth discontinuities as can be seen in Fig. 2.17. Interestingly, the solution to the unconstrained and the approximate problems show slightly better performance. Badino et al. argue that this is due to bad conditioning of the solution to the constraint LS formulation.

Robust Moving Least Squares fitting (RMLS) The RMLS algorithm aims to robustify the previous method by growing the inlier set among the neighborhood [76] incrementally. For this purpose the algorithm keeps track of the error of all neighboring points that are not yet incorporated into the model and, one-by-one, adds the lowest error points into the model. Each time a point is added to the model a least squares fit to the inliers as well as the error of all outliers is updated. This approach for surface normal fitting is outlined in Algorithm 6. In practice we add more than one point per iteration to speed up the algorithm. Specifically, we add $\lceil 1.3^i \rceil$ points in iteration i. This ensures that initially points are added one-by-one and that the algorithm speeds up after a few initial model updates.

We can use any algorithm to update the surface normal given the inlier data points. In Fig. 2.17 we show angular fit for all three aforementioned least squares solutions. Again the unconstrained and the approximate least squares solutions yield (slightly) better results.

Select initial inlier set of 3 points (including query point p_0)
Construct outlier set O from all other points in neighborhood N_i
while error of p_i is smaller than threshold **do**
    Update errors of all outliers p_i via $\epsilon_i = n^T(p_0 - p_i)$
    Find point p_i with smallest error
    if error of p_i is greater than threshold **then**
        Return n
    end if
    Update n using the inlier data
    Remove p_i from O
end while

Algorithm 6: Algorithm for robust moving least squares (RMLS) fitting.

Voting-based surface normal extraction Another approach to surface normal extraction, established in this research program, was motivated by the approach to gravity estimation of Gupta et al. [96]. The idea is to extract the local surface normal using inlier/outlier inference via simulated annealing from a starting surface normal estimate. Per iteration each point is classified as inlier if it is within $90° \pm \Delta\alpha$ with respect to the current surface normal estimate. Then a new surface normal estimate is computed from only the inlier points. $\Delta\alpha$ is annealed starting from 35° down to 15° in steps of 10°. The initial surface normal estimate is obtained as the normalized cross product of the local forward gradients of the organized point cloud $p(u, v)$ following Eq. (A.1):

$$n = \left\lceil \frac{\partial p(u,v)}{\partial u} \times \frac{\partial p(u,v)}{\partial v} \right\rceil = \lceil (p(u,v) - p(u+1,v)) \times (p(u,v) - p(u,v+1)) \rceil \ . \tag{2.192}$$

As Figure 2.17 shows, the voting-based approach yields comparable results to RMLS except for depth discontinuities, where it tends to use points from both planes and thus produce incorrect surface normal estimates at the discontinuity. The advantage of this method is that while producing estimates of similar quality it is generally faster then RMLS since no sorting is required.

Triangle meshes On first sight extracting surface normals from a triangle mesh is straightforward since each triangle describes a surface and we can directly compute the triangles surface normal as:

$$n = \lceil (p_1 - p_0) \times (p_2 - p_0) \rceil \ . \tag{2.193}$$

Since each point of the triangle mesh is potentially part of multiple triangles, it is unclear which triangles surface normal to choose as the surface normal for a given vertex. One approach would be to just compute a new point cloud from the triangle mesh as the set of triangle-center points. In general it is more desirable to use the point cloud of mesh

vertices for efficiency reasons. The surface normal of a mesh vertex is computed as the normalized sum over the normals of the triangles this vertex is part of:

$$n = \left\lceil \sum_{i \in N_i} \left\lceil (p_{i1} - p_{i0}) \times (p_{i2} - p_{i0}) \right\rceil \right\rceil \ . \tag{2.194}$$

To perform this operation efficiently an inverted index from vertex id to triangle id should be precomputed.

Implicit signed distance function representations Newcombe et al. [177] popularized the use of the truncated signed distance function (TSDF) for dense reconstruction systems. Originally proposed in [52], the TSDF represents a surface as the zero-crossing in this 3D volumetric function. In the TSDF volume the surface normals are computed as the normalized gradient of the TSDF at the zero-crossing:

$$n = \left\lceil \nabla_{x,y,z} \mathrm{TSDF}(x, y, z) \right\rceil \ . \tag{2.195}$$

We found that surface normals derived from TSDF-fused surfaces are of high quality.

Fast surface normal extraction from depth image While some tasks necessitate accurate surface normals and can afford the additional computation time, sacrificing some of the quality, we can obtain huge speedups. To efficiently extract all surface normals from the depth image, we can make use of the ordering imposed by the depth image itself and utilize GPUs for fast parallel processing. We derive the algorithm by going back to the definition of the Gauss map in Eq. (2.184). Accordingly, we can obtain the surface normals from the point cloud using the cross-product of the local gradients:

$$n = \left\lceil \frac{\partial p(u,v)}{\partial u} \times \frac{\partial p(u,v)}{\partial v} \right\rceil \ , \tag{2.196}$$

where the derivatives are along the u and v axis of the image coordinate system. Note that the derivatives and cross-products can be computed completely in parallel. We could simply compute the point cloud using any camera model, compute the local gradients using forward differences and then compute surface normals. But since the point cloud is fully determined by the unprojection function $\pi^{-1}(u, v, d)$ of the camera model and the depth image $d(u, v)$ we can compute the derivatives using the chain rule:

$$n = \left\lceil \frac{\partial \pi^{-1}(u, v, d(u,v))}{\partial u} \times \frac{\partial \pi^{-1}(u, v, d(u,v))}{\partial v} \right\rceil \tag{2.197}$$

$$= \left\lceil \left(\frac{\partial \pi^{-1}(u, v, d)}{\partial u} + \frac{\partial \pi^{-1}(u, v, d)}{\partial d} \frac{\partial d}{\partial u} \right) \times \left(\frac{\partial \pi^{-1}(u, v, d)}{\partial v} + \frac{\partial \pi^{-1}(u, v, d)}{\partial d} \frac{\partial d}{\partial v} \right) \right\rceil \ . \tag{2.198}$$

With this, surfaces normal computation formulas can be derived for any camera model for which the derivatives of the inverse projection operation can be computed in closed

form. Under the pinhole camera model [102], for example, the point cloud $p(u, v)$ can be recovered from a depth image $d(u, v)$ as:

$$p(u,v) = \pi^{-1}(u, v, d(u,v)) = \begin{pmatrix} \frac{d(u,v)}{f_u}(u - u_c) \\ \frac{d(u,v)}{f_v}(v - v_c) \\ d(u,v) \end{pmatrix} , \tag{2.199}$$

where f_u and f_v are the focal lengths of the depth camera (in u and v direction) and $[u_c, v_c]$ is the center of the depth-image. As shown in Appendix A.1 this leads to a direct formula for computing surface normals from a depth image:

$$n = \left\lceil \begin{pmatrix} -\frac{\partial d}{\partial u} f_d \\ -\frac{\partial d}{\partial v} f_d \\ \Delta u \frac{\partial d}{\partial u} + \Delta v \frac{\partial d}{\partial v} + d \end{pmatrix} \right\rceil . \tag{2.200}$$

The derivatives $\frac{\partial d}{\partial u}$ and $\frac{\partial d}{\partial v}$ can be approximated using first order finite differences. An efficient way of implementing this operation is by convolution with normalized Sobel or Schar kernels, which can be performed on a GPU as well. We use Schar kernels in practice since they were found to yield better gradients. Note that these kernels are separable which means we can compute the convolution more efficiently using two passes first with a column and then with a row filter or vice versa. The 3×3 Schar kernels are:

$$D_u = \tfrac{1}{36} \begin{pmatrix} -3 & 0 & 3 \\ -10 & 0 & 10 \\ -3 & 0 & 3 \end{pmatrix} = \tfrac{1}{32} \begin{pmatrix} 3 \\ 10 \\ 3 \end{pmatrix} \begin{pmatrix} -1 & 0 & 1 \end{pmatrix} , \tag{2.201}$$

$$D_v = \tfrac{1}{32} \begin{pmatrix} -3 & -10 & -3 \\ 0 & 0 & 0 \\ 3 & 10 & 3 \end{pmatrix} = \tfrac{1}{32} \begin{pmatrix} -1 \\ 0 \\ 1 \end{pmatrix} \begin{pmatrix} 3 & 10 & 3 \end{pmatrix} . \tag{2.202}$$

The vanilla algorithm of computing the depth image gradients using convolution and then computing surface normals according to Eq. (2.200) is sensitive to noise because of the reliance of only the smallest neighborhood. By preprocessing the depth image to remove noise this sensitivity can be alleviated. The common approach of convolution (smoothing) with a Gaussian kernel has problems at depth discontinuities where the filtering leads to interpolation of the depth values between two surfaces leading to erroneous depths. This can be improved by using anisotropic filtering methods such as bilateral filters [184, 235] which do not smooth across value discontinuities. Bilateral filters can be implemented efficiently using a guided filter [107]. Guided filters rely on box filters which can be sped up using integral images [242].

Figure 2.17 shows that this derivative-based approach works generally well for smooth surfaces but fails to accurately capture surface normals near depth discontinuities and around corners. The Gaussian pre-filtering smoothes across these surface features and leads to consistent (and predictable) inaccuracies. This motivates the use of the bilateral filter in practice.

■ 2.9 Summary

With an introduction to Bayesian sampling-based inference, Bayesian nonparametric models as well as common discrete, Euclidean and spherical distributions this chapter has set the stage for the models developed and employed in this thesis. Details on the manifolds of rigid body transformations, $\mathbb{SO}(3)$ and $\mathbb{SE}(3)$, and a discussion of different surface normal extraction algorithms complete the background material. In the next chapters we apply these tools and the introduced machinery to develop Manhattan constrained and unconstrained directional scene models before proceeding to derive a global localization algorithm and the first nonparametric direction-aware SLAM system.

Chapter 3

Manhattan World Constrained Scene Representation

Inference of geometry-based scene models and scene segmentations can be considered a stepping stone for higher-level scene understanding as argued in the introductory chapter. Such segmentations impose an organization of the scene implied by the specific geometry that can be used to simplify further reasoning tasks such as extracting traversable floor space and obstacles or inferring a canonical scene orientation to transfer

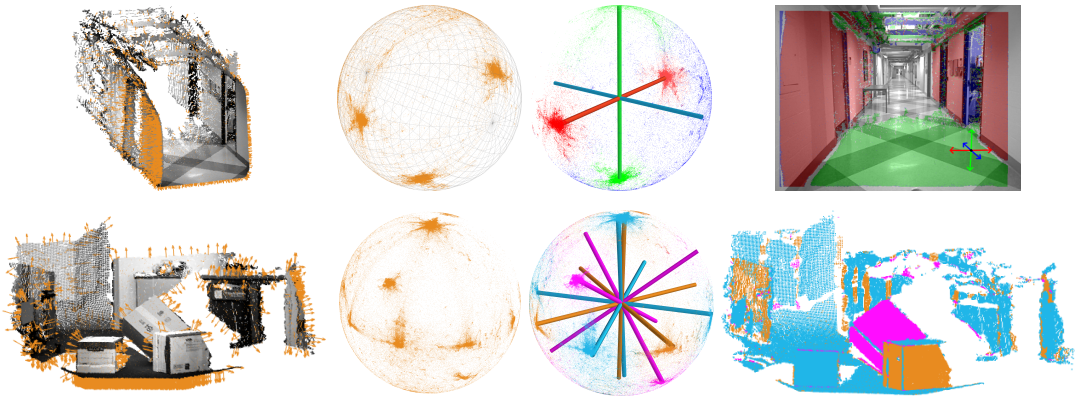


Figure 3.1: To infer the Manhattan Frame and MMF orientations and scene segmentation, we first extract surface normals from a given point cloud before performing inference to fit the probabilistic Manhattan Frame or MMF model, proposed in this chapter, to the surface normal distribution on the unit sphere in 3D. This is motivated by the observation that man-made environments exhibit characteristic low-entropy surface normals as can be seen in the middle left. The inferred Manhattan Frame or MMF clustering provides a directional scene segmentation as displayed to the right. Note that the orthogonality constraints imposed by the Manhattan Frame and MMF model inform about directional relationships between scene parts.

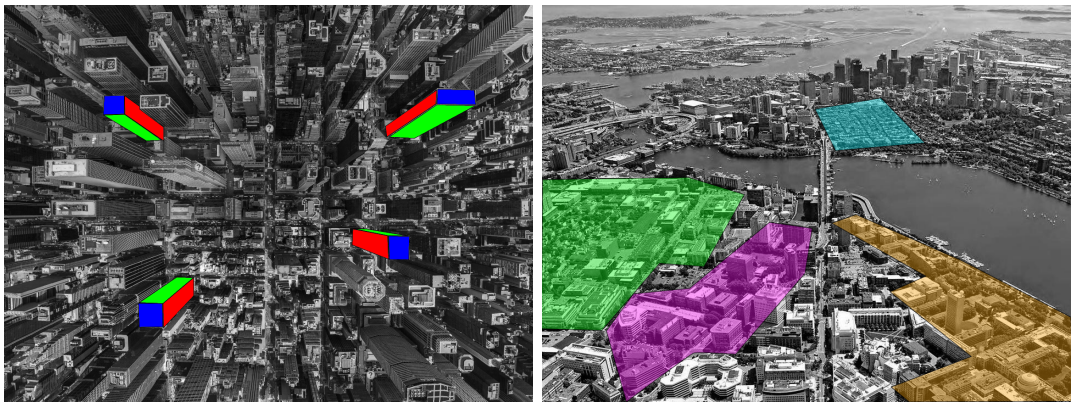


Figure 3.2: The Manhattan World model assumes that all planes in a scene are parallel to one of three mutually orthogonal planes. These planes are indicated in red, green, and blue to the left. While the Manhattan World assumption is often valid on a local scale in man-made environments, on a larger scale scenes are often better described by multiple Manhattan Worlds as shown to the right in aerial view of Cambridge and Boston. Colored areas indicate city parts following different Manhattan Worlds. Within each region planes are assumed to follow the Manhattan World assumption for some local orientation.

knowledge between different scenes [173]. Consider for example the scene segmentation displayed in the first row of Fig. 3.1 which was inferred by one of the methods proposed in this chapter. An autonomous agent with a gravity sensor would be able to associate the green areas with traversable floor-space while red and blue scene parts would represent obstacles at an orthogonal angle to the floor.

To facilitate reasoning about complex scenes we build on simplifying assumptions about the 3D structure of the environment. Observe that, on a wide range of scales, from the layout of a city to structures such as buildings, furniture and many other objects, man-made environments lend themselves to a description in terms of parallel and orthogonal planes as depicted in Fig. 3.2 to the left. This intuition is formalized as the Manhattan World (MW) assumption [49] which posits that most man-made structures may be approximated by planar surfaces that are parallel to one of the three principal planes of a common orthogonal coordinate system.

At a local level, this assumption holds for parts of city layouts, most buildings, hallways, offices and other man-made environments. However, the strict Manhattan World assumption is often a poor representation of real-world scenes on a global level: a rotated desk inside a room, more complex room and city layouts (as opposed to planned cities like Manhattan). While local parts of the scene can be modeled as a Manhattan World, the entire scene cannot. Figure 3.2 to the right depicts an aerial view of Cambridge and Boston to illustrate the concept of describing a scene via multiple Manhattan Worlds. This suggests a more flexible description of a scene is required—one

that is composed of multiple Manhattan Worlds of different orientations.

Motivated by the observation that across a wide range of scales, man-made environments exhibit structured, low-entropy surface-normal distributions as displayed in the middle of Fig. 3.1 we directly work in the surface normal space rather than via intermediate plane segmentations or via the point cloud. Surface-normal distributions are invariant to translation and scale [116] which makes the proposed approach largely independent of the measurement and 3D reconstruction process. Finally, surface normals are straightforward to extract from most 3D scene representations such as depth images, point clouds and meshes as discussed in Sec. 2.8. Careful implementations of surface normal extraction can mitigate but not eliminate the effects of noisy surface estimates and surface discontinuities. This motivates the use of latent variable models to model surface normal noise explicitly and properly.

The contributions in this chapter comprise the introduction of the Manhattan Frame (MF) which represents the Manhattan World structure in the space of surface normals, i.e., the unit sphere, as orthogonally-coupled clusters (see top row of Fig. 3.1). Modeling surface-normal noise with two different distributions on the sphere, we formulate two probabilistic generative Manhattan Frame models. Depending on the model, real-time MAP inference is carried out in closed form or via optimization on the manifold of rotation matrices $\mathbb{SO}(3)$. In the second part of this chapter the Manhattan Frame model is used to construct a mixture of Manhattan Frames (MMF) to represent complex scenes composed of multiple Manhattan Frames (see bottom row of Fig. 3.1). For the MMF model, we derive a simple MAP inference algorithm and a Gibbs-sampling-based algorithm that using Metropolis-Hastings [105] split/merge proposals [197], adapts the number of Manhattan Frames to best capture the surface-normal distribution of a scene.

Before introducing the Manhattan Frame and the MMF model, we review related scene representations in the next section. In Sec. 3.5, we examine the properties of the proposed models and inference algorithms in a variety of qualitative and quantitative experiments. These demonstrate the expressiveness and versatility of the novel Manhattan Frame and MMF model across scales: depth images of a single view of a scene, larger indoor reconstructions and large-scale aerial LiDAR data of an urban center.

■ 3.1 Related Work

The different assumptions made in the literature about the environment can be categorized in terms of their expressiveness as depicted in Fig. 3.3. The assumptions range from mostly unrestricted representations such as point clouds, mesh and level-set, which can in the limit represent any surface exactly, to the rather strict Manhattan World assumption as indicated by taxonomoy. The proposed MMF assumption subsumes the Atlanta World (AW) which in turn subsumes the Manhattan World assumption. The MMF provides a directional segmentation under the orthogonality constraints imposed by the Manhattan World assumption. Relaxing the orthogonality constraints completely, we arrive at what we term the Stata Center World (SCW). It captures only the

directional composition of a scene. Planar scene representations differ from the Stata Center World in that different planes with the same orientation are separated in space.

These different assumptions about scenes can be observed directly in the 3D structure or indirectly in the projection of the 3D structure into a camera [102]. Models and inference algorithms based on the former utilize 3D representations such as meshes, point clouds and derived data such as surface normals. Intersections of planes in 3D are lines which can be observed as lines in the image space. A vanishing point (VP) is the intersection of multiple such lines where the 3D lines are all parallel to each other. Models built on VPs usually use image gradient orientations directly or indirectly via line segment extraction. Specifically, the Manhattan World is manifested as orthogonally-coupled VPs (OVPs) in the image space and an Manhattan Frame in the surface-normal space. Multiple Manhattan Worlds cause multiple orthogonal VPs and Manhattan Frames. The Stata Center World can be observed via independent VPs in the image or independent surface normal clusters.

2D image-space There is a vast literature on VP estimation from RGB images. The goals for VP estimation range from single-image scene parsing [18] and 3D reconstruction [57, 109, 146, 153], VP direction estimation for rotation estimation with respect to man-made environments [7, 20, 49, 139] to VP direction tracking over time to estimate camera rotation and scene structure [29, 79, 141, 169, 213].

While early VP extraction algorithms relied on image gradients [49, 214], most modern algorithms operate on line segments extracted from the image. This has been found to yield superior direction estimation results over dense image-gradient approaches [59]. Generally, VPs are extracted by intersecting lines in the image. These intersections are often found after mapping lines to the unit sphere [19, 48, 141], or into other accumulator spaces [149]. Introduced in [18], horizon estimation has emerged as a benchmark for VP estimation algorithms [252, 255].

Many VP extraction algorithms rely on the Manhattan World assumption [20, 29, 49, 79, 139, 157, 203, 252] which is manifested as three orthogonal VPs. Incorporating the Manhattan World assumption into the VP estimation algorithms not only increases estimation accuracy (if the Manhattan World assumption holds) [149] but also allows estimation of the focal length of the camera [40, 47, 139, 149, 203, 252], and rejection of spurious VP detections. Another avenue of research uses the Manhattan World assumption for single-image 3D reconstruction [57, 109, 146, 153]. The inferred Manhattan World and associations of lines to Manhattan World axes combined with geometric reasoning are used to reconstruct the 3D scene in [57, 146]. Hedau et al. [109] use an Manhattan World prior to iteratively infer the 3D room layout and segment out clutter in the room. Liu et al. [153] use a floor plan in conjunction with a set of monocular images to reconstruct whole apartments.

The AW model of Schindler et al. [214] assumes that the world is composed of multiple Manhattan Worlds sharing the same z-axis (which is assumed to be known). This facilitates inference from RGB images as only a single angle per MW has to be estimated as opposed to a full 3D rotation. The approach by Antunes et al. [9] infers

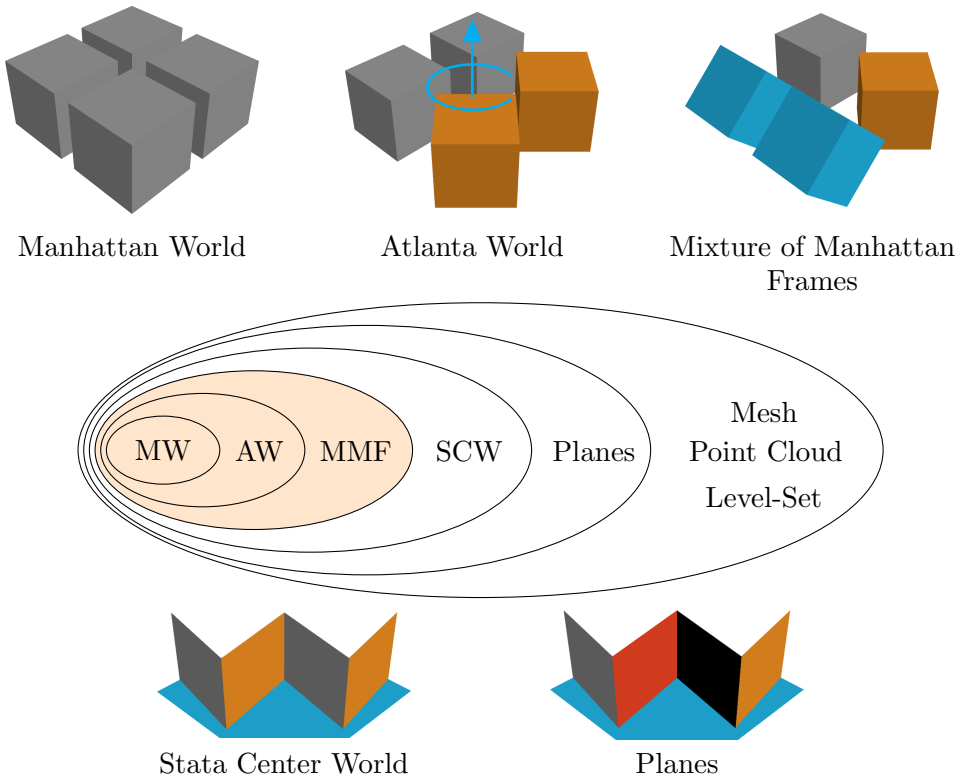


Figure 3.3: Depiction of different scene priors and their relationship to the proposed Manhattan Frame and MMF model. The MMF generalizes both the Manhattan World and the Atlanta world assumption. The Stata Center World relaxes the orthogonality constraints of the Manhattan World based models and is thus more general. In this chapter we focus on single and multiple Manhattan World-based models as indicated by the orange shading.

the full MMF from RGB images. Relaxing the assumptions about the scene, VPs can be extracted independently [7, 18, 48, 59, 141, 149, 169, 231, 255] akin to the Stata Center World assumption.

3D representations There are many approaches that rely purely on 3D representations of surfaces and scenes. Assumptions such as the Manhattan World or SCW, are used to align scenes into a common frame of reference for scene segmentation and understanding [96, 173], and to regularize 3D reconstruction [170, 186]. The AW and MMF model could be used similarly.

Similar to the image space, the Manhattan World assumption has been used most commonly [96, 173]. This is probably due to the fact that man-made environments exhibit strong Manhattan World characteristics on a local scale, i.e. on the level of a single RGBD frame of a scene. In the application of Simultaneous Localization and Mapping (SLAM) [148], the Manhattan World assumption has been used to impose constraints on the inferred map [186]. Our original idea of the Manhattan Frame [225] has been adapted by Ghanem et al. [87] who propose a robust inference scheme for MF estimation (RMF) and by Joo et al. [127] who use a branch-and-bound scheme to perform real-time globally optimal Manhattan Frame inference (MF BB).

To the best of our knowledge the assumption of multiple Manhattan Worlds in the 3D data setting (as opposed to RGB 2D-images) has not been explored prior to our own work described in this chapter.

Similar to the Manhattan Frame and MMF model, the Stata Center World can be inferred solely from surface-normal distributions as described in Chapter 4. Note, that the Manhattan Frame and MMF model could be inferred from the Stata Center World by grouping inferred directions into Manhattan Frames.

Somewhat outside all these categories, Gupta et al. [96] assume the only relevant direction for semantic scene segmentation is the direction of gravity to enable alignment of the ground plane across scenes. They propose a simple algorithm to segment the scene into the gravity and all other directions based on surface-normal observations.

An alternative to the Manhattan World, MMF or Stata Center World model describes man-made structures by individual planes with no constraints on their relative normal directions. The orthogonality constraints in the Manhattan World or MMW models enable statistical pooling of measurements across different orientations. This means not only that fewer measurements (per plane) are needed to achieve the same amount of accuracy as without those constraints but also that reliable measurements from one or more directions help in handling cases where there are only few reliable measurements from other directions.

2D & 3D The connection between VPs in images and 3D Manhattan World structures has been used to infer dense 3D structure from sets of images by Furukawa et al. [82]. They employ a greedy algorithm for a single-MF extraction from normal estimates that works on a discretized sphere. Neverova et al. [176] integrate RGB images with associated depth data from a Kinect camera to obtain a 2.5D representation of indoor

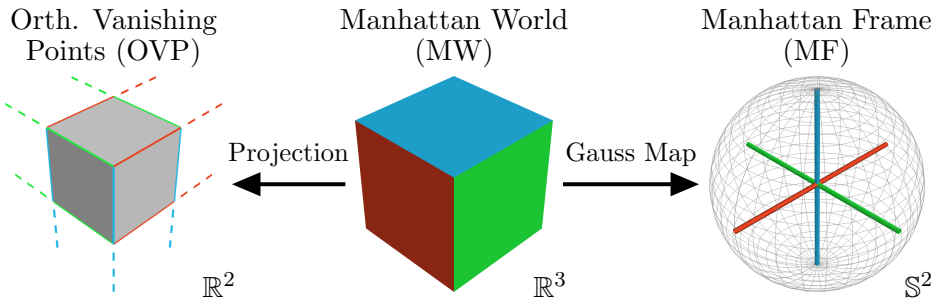


Figure 3.4: A 3D Manhattan World maps to a Manhattan Frame in the surface normal space under the Gauss map. Under camera projection it maps to three orthogonal vanishing points in the image plane.

scenes under the Manhattan World assumption. Silberman et al. [173] infer the dominant Manhattan World using VPs extracted from the RGB image and surface normals computed from the depth image.

■ 3.2 The Manhattan Frame (MF)

The Manhattan Frame (MF) is the image of a 3D Manhattan World structure under the Gauss Map (see Sec. 2.8) as depicted in Fig. 3.4. In other words, the Manhattan Frame describes the notion of the Manhattan World in the space of surface normals. In a noise-free, perfect Manhattan World the surface normals would align with the six directions in world coordinates collected as columns in:

$$E = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}, \; e_j \text{ denotes the } j\text{th col. of } E. \tag{3.1}$$

In the camera coordinate system these six directions will appear rotated by ${}^{c}R_{w}$:

$$M = {}^{c}R_{w}E\,, \; \mu_j \text{ denotes the } j\text{th col. of } M. \tag{3.2}$$

The rotation ${}^{c}R_{w}$ is an element of $\mathbb{SO}(3)$ the space of orthonormal matrices in 3D with determinant 1 as introduced in Sec. 2.7.1. This rotation of the camera, ${}^{w}R_{c} = {}^{c}R_{w}^{T}$, is unknown and hence a key parameter to be estimated by an inference algorithm. In the following we will use $R = {}^{c}R_{w}$ for the sake of notational simplicity. In other words, if a 3D scene consists of only planar surfaces such that the set of their surface normals is contained in the set $\{\mu_j\}_{j=1}^{6}$, then M captures all possible directions in the scene—the scene follows the Manhattan World assumption.

Specifically, let $n_i \in \mathbb{S}^2$ denote the ith observed surface normal. A latent label, $z_i \in \{1, \ldots, 6\}$, assigns n_i to one of the six directions of the Manhattan Frame. Hence μ_{z_i} is the direction associated with n_i. In the following we will denote the set of all labels as $\mathbf{z} = \{z_i\}_{i=1}^{N}$ and the set of all surface normals as $\mathbf{n} = \{n_i\}_{i=1}^{N}$. The unit normals are elements of the unit sphere in $\mathbb{R}^3$, denoted by $\mathbb{S}^2$, a 2D manifold

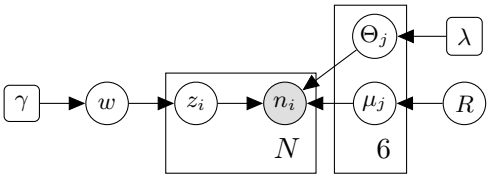


Figure 3.5: Graphical model capturing the assumed distribution of observed normals under the Manhattan Frame distribution.

whose geometry is outlined in Sec. 2.6.1. Commonly, in 3D processing pipelines (e.g. in surface fairing or reconstruction [124, 132]), the unit normals are estimated from noisy measurements of the 3D scene structure such as depth images [112, 221], point clouds [168] and meshes [35]. Surface normals $\mathbf{n}$ may deviate from their associated MF axis for several reasons. First, noisy input data such as depth images, point-clouds or meshes lead to noisy surface normal observations since they are computed from the 3D observations (see Sec. 2.8). Second, the underlying 3D structure being sensed may not follow the Manhattan World assumption in parts of the scene such as on curved, round or unstructured surfaces. For such scene parts it might still be useful to assign them to a Manhattan Frame direction as the Manhattan Frame provides an ordering and segmentation of the environment that can be exploited for downstream inferences. Even in environments dominated by curved shapes the Manhattan Frame can provide a quantization into orthogonal scene parts that can be utilized.

In order to fit the parameters of an Manhattan Frame, one would seek to penalize those deviations. While, in principle, this can be formulated directly as a deterministic optimization, we adopt a probabilistic modeling approach. This allows us to derive a real-time algorithm for single Manhattan Frame inference as well as MCMC inference (see Sec. 2.2.1) for the more complicated MMF model all from the same base Manhattan Frame model. Another key advantage over deterministic approaches is that probabilistic inference allows reasoning about uncertainty which is important for scene understanding. Furthermore, the probabilistic nature of the MMF model means that it can be integrated into larger and more complex environment models, as we showed in [35]. Another approach would be to utilize a non-parametric directional segmentation algorithm such as [222] or [224] and to fit MFs to the inferred modes of the surface normal distribution. The advantage of directly inferring an MF model is that data from the different (orthogonal and opposing directions) all jointly contributed to the estimation of the MF orientation. This is especially important in scenes like the urban scene in Fig. 1.2 where there is only few data points for some of the directions. To this end, we propose two different noise models to describe those random deviations: tangent space Gaussian (TG) noise as well as von-Mises-Fisher (vMF) noise.

■ 3.2.1 The Probabilistic Manhattan Frame Model

Let $R \in \mathbb{SO}(3)$ denote the rotation of the Manhattan Frame. Making no a-priori assumptions about which orientation of the Manhattan Frame is more likely than others, R is distributed uniformly:

$$R \sim \text{Unif}(\mathbb{SO}(3)) \,. \tag{3.3}$$

Since $\mathbb{SO}(3)$ is a manifold with finite support, we can compute its volume and obtain $8\pi^2$ [45] which implies that all rotations have equal likelihood of $1/8\pi^2$. In practice secondary sensors such as inertial measurement sensors (IMUs) can yield additional information about the rotation of the Manhattan Frame via the measurement of the gravity direction. Similarly, in the streaming setting the Manhattan Frame rotation is likely to be close to the previous frames Manhattan Frame rotation. In those situations one can capture the additional knowledge in the prior on R. Indeed in Sec. 3.3.4 we explore the streaming Manhattan Frame inference setting and show that it is straightforward to incorporate a zero-motion prior on the Manhattan Frame rotation.

As is standard in Bayesian mixture modeling, the Manhattan Frame axis assignments z_i of a surface normal n_i to an Manhattan Frame axis are assumed to be distributed according to a categorical distribution $\text{Cat}(w)$ with a Dirichlet distribution prior parameterized by γ:

$$w \sim \text{Dir}(\gamma) \tag{3.4}$$
$$z_i \sim \text{Cat}(w) \,. \tag{3.5}$$

The deviations of the observed normals from their assigned directions are modeled by a directional distribution parameterized by Θ. We only require this directional distribution to have the assigned Manhattan Frame direction μ_{z_i} as its mode. Following a Bayesian approach we assume a prior $p(\Theta; \lambda)$ for the parameters Θ of the Manhattan Frame axis distributions:

$$\Theta \sim p(\Theta; \lambda) \tag{3.6}$$
$$n_i \sim p(n_i \mid z_i, R, \Theta) \quad \text{s.t.} \quad \mu_{z_i} = \underset{n \in \mathbb{S}^2}{\arg\max}\; p(n \mid z_i, R, \Theta) \,, \tag{3.7}$$

where λ are the so-called hyperparameters of the prior on the Manhattan Frame axis distributions. As reviewed in Sec. 2.6, many directional distributions exist (e.g., the Bingham [23], and the Kent [134] distribution, and others [164]) and most are valid choices for the distribution of surface normals. We focus on the tangent-space Gaussian (see Sec. 3.2.2) and the von-Mises-Fisher distribution (see Sec. 3.2.3) as depicted in Fig. 3.6 and 3.8 respectively.

Finally, the graphical model for the MF is depicted in Fig. 3.5 and the joint distribution is:

$$p(\mathbf{z}, \mathbf{n}, w, R, \Theta; \gamma, \lambda) = \tfrac{1}{8\pi^2} p(w; \gamma) p(\Theta; \lambda) \prod_{i=1}^{N} w_{z_i} p(n_i \mid z_i, R, \Theta) \,. \tag{3.8}$$

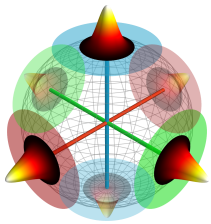


Figure 3.6: Graphical depiction of the tangent space Gaussian Manhattan Frame model.

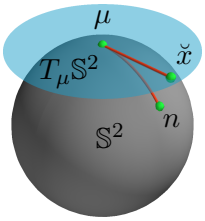


Figure 3.7: The blue plane illustrates $T_\mu\mathbb{S}^2$, the tangent space to the sphere $\mathbb{S}^2$ at Manhattan Frame axis $\mu \in \mathbb{S}^2$. A surface normal $n \in \mathbb{S}^2$ is mapped to tangent vector $\breve{x} \in T_p\mathbb{S}^2$ via Log_μ .

Both, the graphical model and the factoring of the joint distribution, suggest that the surfaces normals are assumed to be generated independently given assignments to Manhattan Frame axes, the Manhattan Frame rotation and other parameters of the Manhattan Frame axis distributions Θ. This is an approximate assumption, since nearby surface normals generally depend on each other. Most man-made environments consist of planes and smooth surfaces, making the directions of most surface normals dependent on its neighborhood. This assumption, however, enables more efficient inference because, for example, each normals assignment to a Manhattan Frame axis can be computed independently and in parallel. Furthermore, the results in Sec. 3.5.1 show smooth scene segmentations despite assuming conditional independence.

■ 3.2.2 Tangent Space Gaussian Manhattan Frame Model

The tangent-space Gaussian MF (TG-MF) model describes the deviations not on $\mathbb{S}^2$ directly but in a tangent plane to the sphere using the tanget-space Gaussian model as introduced in Sec. 2.6.2. The TG-MF distribution is visualized in Fig. 3.6. Under the TG-MF model observed normals are modeled by a tangent space Gaussian distribution with covariance $\Theta = \mathbf{\Sigma} \in \mathbb{R}^{2\times2}$ and mean μ_{z_i} centered on the respective assigned MF axis.

$$p(n_i \mid z_i, R, \Theta) = \mathcal{N}(\text{Log}_{\mu_{z_i}}(n_i); 0, \Sigma_{z_i}) \,, \tag{3.9}$$

where $\text{Log}_{\mu_{z_i}}(n_i) \in T_{\mu_{z_i}}\mathbb{S}^2$. In other words, we evaluate the probability density function of $n_i \in \mathbb{S}^2$ by first mapping it into $T_{\mu_{z_i}}\mathbb{S}^2$, as visualized in Fig. 3.7, and then evalu-

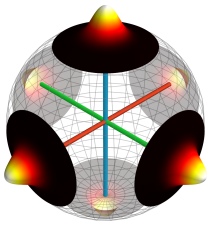
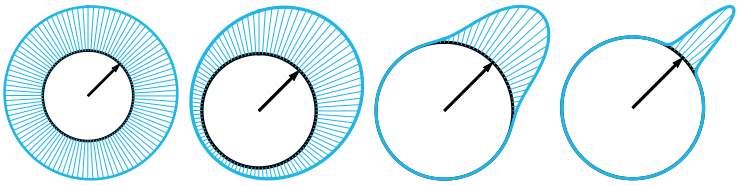


Figure 3.8: Graphical depiction of the von-Mises-Fisher based Manhattan Frame model.



(a) $\tau = 0$ (b) $\tau = 1$ (c) $\tau = 10$ (d) $\tau = 100$

Figure 3.9: Depiction of the von-Mises-Fisher distributions in 2D with different concentrations τ around the same mean μ. Note that the von-Mises-Fisher can capture a uniform distribution over the sphere for $\tau = 0$ as well as concentrated isotropic distributions for larger τ.

ating it under the Gaussian distribution with covariance $\Sigma_{z_i} \in \mathbb{R}^{2\times2}$. The conjugate prior distribution for covariance matrices $\mathbf{\Sigma}$ is the inverse Wishart distribution [85] parameterized by $\lambda = \{\Delta \in \mathbb{R}^{2\times2}, \nu \in \mathbb{R}\}$:

$$p(\Theta; \lambda) = \mathrm{IW}(\Sigma_{z_i}; \Delta, \nu) \tag{3.10}$$

As alluded to and examined in Sec. 2.6.2, the range of Log_p is contained within a disk of radius π while the Gaussian distribution has infinite support. Consequently, for probabilistic inference, we use the inverse Wishart prior to favor small covariances resulting in a probability distribution that, except a negligible fraction, is within the range of Log_p and concentrated about the respective axis. The advantage of choosing the tanget-space Gaussian distribution is that it allows modeling and capturing anisotropic surface normal distributions in contrast to the isotropic vMF distribution.

■ 3.2.3 von-Mises-Fisher Manhattan Frame Model

As introduced in Sec. 2.6.3, the vMF distribution is natively defined over the manifold of the sphere and commonly used to model directional data [14, 15, 92, 222]. We utilize the vMF distribution to model the deviation of a surface normal from its assigned Manhattan Frame axis. In 3D, the vMF defines an isotropic distribution for 3D directional data $n \in \mathbb{S}^2$, i.e. surface normals, around a mode $\mu \in \mathbb{S}^2$ with a concentration $\tau \in \mathbb{R}^0_+$

and has the following form:

$$\mathrm{vMF}(n;\mu,\tau) = Z(\tau)\exp(\tau n_i^T \mu) \tag{3.11}$$

$$Z(\tau) = \frac{\tau}{4\pi \sinh \tau}\,. \tag{3.12}$$

The concentration parameter behaves inversely to the variance of a Gaussian: increasing the concentration leads to a more peaked distribution whereas setting $\tau = 0$ leads to a uniform distribution over the sphere. This is illustrated in Fig. 3.9 for the 2D vMF.

In the vMF Manhattan Frame model, a surface normal $n_i \in \mathbb{S}^2$ is distributed according to a vMF distribution centered on the associated Manhattan Frame axis, μ_{z_i}, with concentration $\Theta = \tau$:

$$p(n_i \mid z_i, R, \Theta) = \mathrm{vMF}(n_i; \mu_{z_i}, \tau)\,, \tag{3.13}$$

$$p(\Theta;\lambda) \propto Z(\tau)^a \exp(b\tau)\,, \quad \lambda = \{a, b\}\,, \tag{3.14}$$

where the prior $p(\Theta;\lambda)$ on the concentration parameter of the vMF is only known up to proportionality [180]. The vMF Manhattan Frame distribution is conceptually depicted in Fig. 3.8.

■ 3.3 Real-time Manhattan Frame MAP Inference

Based on the probabilistic generative models for the Manhattan Frame setup in the previous sections, we now develop real-time Manhattan Frame (RTMF) maximum-a-posteriori (MAP) inference methods. These algorithms are used to infer the local Manhattan Frame structure of an environment efficiently. Starting from the TG-MF model we first derive the MAP inference algorithm directly before employing an approximation that yields more efficient inference. Lastly, the vMF Manhattan Frame MAP inference is derived. Those three Manhattan Frame algorithms are instantiations of the hard-assignment expectation maximization algorithm (EM): We iterate assigning surface normals to the most likely Manhattan Frame axis and updating the Manhattan Frame rotation estimate until convergence. It would be straightforward to convert the inference to a MCMC-based method such as Gibbs sampling. This is likely to yield more accurate inference of all parameters at the cost of increased computation time.

In this section, for efficiency reasons and in the absence of further knowledge about the scene, the surface normals are assumed to be generated with equal probability from any of the axes, i.e. all $w_j = \frac{1}{6}$. For the same reason we assume identical isotropic covariances $\Sigma_j = \sigma^2\mathrm{I}$ for all TG-MF axes and identical concentration parameters τ_j for all vMF Manhattan Frame axes. In Section 3.4, the TG-MF assumptions will be relaxed.

■ 3.3.1 Direct MAP Manhattan Frame Estimation for the TG-MF

Starting from the tangent-space Gaussian Manhattan Frame model set up in Sec. 3.2.2, we derive the direct MAP Manhattan Frame rotation estimation algorithm. The poste-

rior over assignments z_i of surface normals n_i to axis of the Manhattan Frame is given by

$$p(z_i = j \mid R, n_i; \pi, \Sigma) \propto w_j \, \mathcal{N}(\mathrm{Log}_{\mu_j}(n_i); 0, \Sigma) \,. \tag{3.15}$$

Therefore the MAP estimate for the label z_i becomes:

$$\begin{aligned} z_i &= \underset{j \in \{1 \ldots 6\}}{\arg\min} \, \mathrm{Log}_{\mu_j}(n_i)^T \Sigma^{-1} \mathrm{Log}_{\mu_j}(n_i) = \underset{j \in \{1 \ldots 6\}}{\arg\min} \, \arccos^2(n_i^T \mu_j) \\ &= \underset{j \in \{1 \ldots 6\}}{\arg\max} \, n_i^T \mu_j \,, \end{aligned} \tag{3.16}$$

where we have used $\arccos(n_i^T \mu_j) = \|\mathrm{Log}_{\mu_j}(n_i)\|_2$ and assumed that the covariance Σ is isotropic. This assumption is made for efficiency reasons: otherwise we would have to compute the logarithm map $\mathrm{Log}_{\mu_j}(n_i)$ and the product with Σ for all six different Manhattan Frame directions. While it would be possible to infer Σ as well in the MAP setting or using sampling-based inference, we keep it fixed for efficiency reasons. With $p(R) = \mathrm{Unif}(\mathbb{SO}(3))$, the posterior over the Manhattan Frame rotation R is

$$p(R \mid \mathbf{n}, \mathbf{z}; \Sigma) \propto p(\mathbf{n} \mid \mathbf{z}, R; \Sigma) p(R) \propto p(\mathbf{n} \mid \mathbf{z}, R; \Sigma) = \prod_{i=1}^{N} \mathcal{N}(\mathrm{Log}_{\mu_{z_i}}(n_i); 0, \Sigma) \,. \tag{3.17}$$

Working in the log-domain, the MAP estimate for R is

$$R^\star = \underset{R \in \mathbb{SO}(3)}{\arg\min} - \log p(R \mid \mathbf{n}, \mathbf{z}; \Sigma) \coloneqq \underset{R \in \mathbb{SO}(3)}{\arg\min} \, f(R) \,. \tag{3.18}$$

With the posterior in Eq. (3.17), the cost function $f(R)$ is

$$\begin{aligned} f(R) &= -\log \left[\prod_{i=1}^{N} \mathcal{N}(\mathrm{Log}_{\mu_{z_i}}(n_i); 0, \Sigma) \right] \propto \sum_{i=1}^{N} \mathrm{Log}_{\mu_{z_i}}(n_i)^T \Sigma^{-1} \mathrm{Log}_{\mu_{z_i}}(n_i) \\ &\propto \sum_{i=1}^{N} \arccos^2(n_i^T \mu_{z_i}) = \sum_{i=1}^{N} \arccos^2(n_i^T R e_{z_i}) \,, \end{aligned} \tag{3.19}$$

where we have used a derivation similar to Eq. (3.16). We call this method direct since the cost function directly penalizes a normal's deviation from its associated Manhattan Frame axis.

We enforce the constraints on R ($R^T R = \mathrm{I}$ & $\det(R) = 1$) by explicitly optimizing the cost function on the $\mathbb{SO}(3)$ manifold using gradient descent with backtracking linesearch. More details can be found in [2, 65, 83] and in Sec. 2.7.3. As introduced in Sec. 2.7.1, we use that perturbations of R from R_0 can be written as $R(W) = R_0 \mathrm{Exp}(W)$ where $W = [\omega]_\times \in \mathfrak{so}(3)$ and Exp is the exponential map from $\mathfrak{so}(3)$ to $\mathbb{SO}(3)$ defined

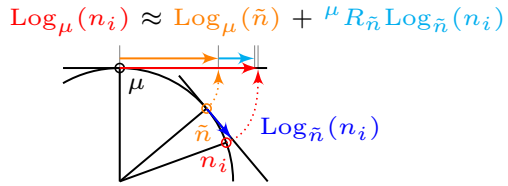


Figure 3.10: The geometry underlying the approximation of the mapping of n_i into $T_\mu\mathbb{S}^2$ via $\mathrm{Log}_\mu(n_i)$.

in Eq. (2.126). Then the Jacobian is $J = \frac{\partial f(R(W))}{\partial \omega}|_{\omega=0}$, the derivative of f with respect to the perturbation $\omega \in \mathbb{R}^3$ at $\omega = 0$:

$$J = \sum_{i=1}^{N} \frac{\partial \arccos^2(x)}{\partial x} \frac{\partial}{\partial \omega} n_i^T R\mathrm{Exp}(W) e_{z_i} \Big|_{\omega=0} = \sum_{i=1}^{N} \frac{2\arccos(n_i^T R e_{z_i})}{\sqrt{1-(n_i^T R e_{z_i})^2}} n_i^T [R e_{z_i}]_\times \tag{3.20}$$

Backtracking line search in the direction of the negative Jacobian at iteration t, J_t, until the Armijo conditions [2, 11] are met provides an appropriate step size δ which allows us to obtain a new rotation estimate using the exponential map:

$$R_t = R_{t-1}\mathrm{Exp}(-\delta J_t) \tag{3.21}$$

The complete algorithm is given in Algorithm 7.

■ 3.3.2 Approximate MAP Manhattan Frame Rotation Estimation

The direct approach derived in the previous section is inefficient since the cost function in Eq. (3.19) and the respective Jacobian involve a sum over all data-points. The Jacobian needs to be re-computed after each update to R and the cost function multiple times during the backtracking linesearch. To address this inefficiency, we derive an approximate estimation algorithm by exploiting the geometry of $\mathbb{S}^2$.

The approximation necessitates the computation of the Karcher means $\{\tilde{n}_j\}_{j=1}^6$ for each of the sets of normals, $\{n_i\}_{\mathcal{I}_j}$, associated with the respective Manhattan Frame axis. The Karcher mean is a generalization of the standard Euclidean sample mean to arbitrary manifolds as introduced in Sec. 2.6.1. After this preprocessing step, we approximate $\mathrm{Log}_{\mu_{z_i}}(n_i)$ using the Karcher mean $\tilde{n}_{z_i}$ as introduced and further analyzed in Sec. 2.6.2:

$$\mathrm{Log}_\mu(n_i) \approx \mathrm{Log}_\mu(\tilde{n}) + {}^{\tilde{n}}R_\mu \mathrm{Log}_{\tilde{n}}(n_i)\,. \tag{3.22}$$

where the subscript z_i was omitted for the sake of clarity and ${}^{\tilde{n}}R_\mu$ rotates vectors in $T_{\tilde{n}}\mathbb{S}^2$ to $T_\mu\mathbb{S}^2$. Intuitively this approximates the mapping of n_i into μ_{z_i} with the mapping of the Karcher mean into μ_{z_i} plus a correction term that accounts for the deviation of n_i

from the $\tilde{n}_{z_i}$. See Fig. 3.10 for an illustration of underlying geometry. Analysis outlined in Sec. 2.6.2 shows that even in the worst case the approximation holds well for small angles ($< 35°$) between μ and $\tilde{n}$. This will generally be fulfilled since aligning μ and $\tilde{n}$ is the part of the inference objective. In the best case, when μ, n_i and $\tilde{n}$ lie on a geodesic or if $\mu = \tilde{n}$, the approximation is exact. With this the cost function $f(R)$ from Eq. (3.19) can be approximated by $\tilde{f}(R)$ as

$$\begin{aligned} f(R) &\propto \sum_{i=1}^{N} \mathrm{Log}_{\mu_{z_i}}(n_i)^T \Sigma^{-1} \mathrm{Log}_{\mu_{z_i}}(n_i) \\ &\approx \sum_{j=1}^{6} \sum_{i \in \mathcal{I}_j} \mathrm{Log}_{\mu_j}(\tilde{n}_j)^T \mathrm{Log}_{\mu_j}(\tilde{n}_j) + 2\mathrm{Log}_{\tilde{n}_j}(n_i)^T (R_{\tilde{n}_j}^{\mu_j})^T \mathrm{Log}_{\mu_j}(\tilde{n}_j) \\ &= \sum_{j=1}^{6} |\mathcal{I}_j| \arccos^2(\tilde{n}_j^T \mu_j) = \sum_{j=1}^{6} |\mathcal{I}_j| \arccos^2(\tilde{n}_j^T R e_j) \coloneqq \tilde{f}(R)\,, \end{aligned} \tag{3.23}$$

where we have used that the sample mean in the tangent space of their Karcher mean $\sum_{i \in \mathcal{I}_j} \mathrm{Log}_{\tilde{n}_j}(n_i) = 0$ by definition. With $\mu_j = R e_j$ the Jacobian for $\tilde{f}(R)$ is

$$J = \frac{\partial \tilde{f}(R(\omega))}{\partial \omega} = \sum_{j=1}^{6} \frac{2|\mathcal{I}_j| \arccos(\tilde{n}_j^T R e_j)}{\sqrt{1 - (\tilde{n}_j^T R e_j)^2}} \tilde{n}_j^T [R e_j]_\times\,. \tag{3.24}$$

Thus the gradient descent optimization over R only utilizes the Karcher means $\{\tilde{n}_j\}_{j=1}^6$, which can be pre-computed since the labels are fixed for the rotation estimation. This eliminates the costly iteration through all data-points at each gradient descent iteration in Alg. 7.

■ 3.3.3 MAP Inference in the vMF Manhattan Frame Model

In the previous section we derived a direct and an approximate MAP inference algorithm for the Manhattan Frame which assumes zero-mean Gaussian noise for surface normals in the tangent space around an associated Manhattan Frame axis. In this section, we derive the MAP inference for the vMF Manhattan Frame model and show that the structure of the vMF distribution allows the Manhattan Frame rotation to be computed in closed form.

With the uniform distribution over labels, i.e. $\pi_j = \frac{1}{6}$, the posterior distribution over label z_i follows the proportionality:

$$p(z_i = j \mid n_i, R; \tau) \propto \mathrm{vMF}(n_i; \mu_j, \tau) \propto \exp(\tau n_i^T \mu_j)\,. \tag{3.25}$$

Since we assume equal concentration parameter τ for the six vMF distributions, the MAP assignment for z_i is:

$$z_i = \underset{j \in \{1,\dots,6\}}{\arg\max}\, n_i^T \mu_j\,. \tag{3.26}$$

Initialize R_0 (to identity or the previous timestep's MF rotation (streaming))
while $\log p(R \mid \mathbf{n}, \mathbf{z}; \Sigma)$ not converged **do**
    On GPU: obtain $z_i = \arg\max_{j \in \{1 \dots 6\}} n_i^T \mu_j \; \forall i \in \{1 \dots N\}$
    On GPU: compute statistics (for approx. method)
    Compute J_0 using Eq. (3.20) or (3.24) respectively
    for $t \in \{1 \dots T\}$ **do**
        $\delta \leftarrow$ backtracking line-search along the geodesic in direction J_t
        $R_{t+1} = \mathrm{Exp}_{R_t}(-\delta J_t)$
        Compute J_{t+1} using Eq. (3.20) or (3.24) respectively
    end for
end while
return R_T

Algorithm 7: Optimization over the Manhattan Frame rotation $R \in \mathbb{SO}(3)$. The difference between the proposed approaches (direct and approximate) is in how the labels $\{z_i\}_{i=1}^N$ and the Jacobians are computed and which statistics are used.

This amounts to assigning the data point n_i to the closest Manhattan Frame axis in terms of angle. Interestingly this is the same assignment rule as for the TG-MF in Eq. (3.16).

With $p(R) = \mathrm{Unif}(\mathbb{SO}(3)) = \frac{1}{8\pi^2}$, the posterior distribution over the Manhattan Frame rotation is:

$$p(R \mid \mathbf{n}, \mathbf{z}; \tau) \propto p(\mathbf{n} \mid \mathbf{z}, R; \tau) p(R) \propto p(\mathbf{n} \mid \mathbf{z}, R; \tau) = \prod_{i=1}^{N} \mathrm{vMF}(n_i \mid \mu_{z_i}; \tau) \,. \tag{3.27}$$

We find the optimal rotation as the maximizer of the log posterior according to:

$$\begin{aligned} R^\star &= \arg\max_{R \in \mathbb{SO}(3)} \log p(R \mid \mathbf{n}, \mathbf{z}; \tau) = \arg\max_{R \in \mathbb{SO}(3)} \sum_{i=1}^{N} \tau n_i^T \mu_{z_i} \\ &= \arg\max_{R \in \mathbb{SO}(3)} \sum_{j=1}^{6} \left(\sum_{i \in \mathcal{I}_j} n_i^T \right) R e_j \\ &= \arg\max_{R \in \mathbb{SO}(3)} \mathrm{tr}\left\{ RN \right\} \,, \; N = \sum_{j=1}^{6} e_j \sum_{i \in \mathcal{I}_j} n_i^T \,. \end{aligned} \tag{3.28}$$

This has the same form as the orthonormal Procrustes problem [215] outline as well in Sec. 2.7.3. Hence, the optimal rotation can be computed in closed form using the SVD $N = USV^T$ as

$$R^\star = V \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(VU^T) \end{bmatrix} U^T \,. \tag{3.29}$$

This has been used before to align point patterns by Umeyama [241] and applied to the Manhattan Frame rotation estimation in a slightly different way by [87]. The algorithm for vMF-MF inference is straight-forward as can be seen in Alg. 8.

1: Initialize R_0 (to identity or the previous timestep's MF rotation (streaming))
2: **while** $\log p(R \mid \mathbf{n}, \mathbf{z}; \tau)$ not converged **do**
3: On GPU: obtain $z_i = \arg\max_{j \in \{1 \dots 6\}} n_i^T \mu_j \; \forall i \in \{1 \dots N\}$
4: On GPU: compute statistics $\sum_{i \in \mathcal{I}_j} n_i \; \forall j \in \{1, \dots, 6\}$
5: Compute R using Eq. (3.29)
6: **end while**
7: **return** R

Algorithm 8: Computing the Manhattan Frame rotation $R \in \mathbb{SO}(3)$ under the vMF Manhattan Frame model.

■ 3.3.4 Real-time Manhattan Frame Inference on Streaming Data

In the case of a stream of batches of surface normals obtained, for example, from an RGB-D camera, we impose a matrix vMF [136] diffusion model with concentration τ_R. The conditional distribution of the current rotation R given the previous rotation R_- is:

$$p(R|R_-, \tau_R) \propto \exp(\tau_R \operatorname{tr}\{R_-^T R\}) \,. \tag{3.30}$$

This distribution is uniform over the rotation space for $\tau_R = 0$ and concentrates on R_- as τ_R increases. See [32] Ex. 2 and Sec. 2.7.5 for a modern treatment of the matrix vMF distribution. In practice we chose $\tau = 1$. This adds the negative log likelihood term

$$f_R = -\tau_R \operatorname{tr}\{R_-^T R\} \tag{3.31}$$

to the MAP cost functions derived in the previous sections. For the direct and the approximate RTMF algorithm this means an additional term in the Jacobian:

$$\begin{aligned} \tfrac{\partial f_R(R(\omega))}{\partial \omega} &= -\tau_R \tfrac{\partial}{\partial \omega} \operatorname{tr}\{R_-^T R(\omega)\} \\ &= -\tau_R \begin{bmatrix} \operatorname{tr}\{R_-^T G_1 R\} & \operatorname{tr}\{R_-^T G_2 R\} & \operatorname{tr}\{R_-^T G_3 R\} \end{bmatrix} . \end{aligned} \tag{3.32}$$

For the vMF-based algorithm we can still derive a closed from rotation MAP estimate:

$$\begin{aligned} R^\star &= \arg\max\nolimits_{R \in \mathbb{SO}(3)} \log p(R|\mathbf{q}, \mathbf{z}; \tau) + \tau_R \operatorname{tr}\{R_-^T R\} \\ &= \arg\max\nolimits_{R \in \mathbb{SO}(3)} \operatorname{tr}\{\tilde{N} R\} \,, \;\; \tilde{N} = N + \tau_R R_-^T \,. \end{aligned} \tag{3.33}$$

Note that the additional term stemming from the matrix vMF distribution acts as a regularizer if only one MF axis has associated observations.

3.4 The Mixture of Manhattan Frames

As alluded to in the introduction, the description of man-made environments on a global scale necessitates a more flexible model that can capture Manhattan Worlds with some relative rotation between them. This motivates the extension of the MF framework described in Sec. 3.2 to the MMF. In practice, scene representations may be composed of multiple intermediate representations, which may include MMFs, to facilitate higher-level reasoning (e.g. [35]). As such, adopting a probabilistic model allows one to describe and propagate uncertainty in the representation. Prior knowledge and model inherent measurement noise can be incorporated in a principled way. Conditional independence allows drawing samples in parallel and hence leads to tractable inference.

In the proposed MMF representation scenes consist of K MFs, $\{M_1, \ldots, M_K\}$ which jointly define $6K$ signed axes. For $K = 1$, the MMF coincides with the MF. Specifically, let $n_i \in \mathbb{S}^2$ denote the ith observed normal. In the MMF, each n_i has two levels of association. The first, $c_i \in \{1, \ldots, K\}$, assigns n_i to the c_ith MF. The second, $z_i \in \{1, \ldots, 6\}$, assigns n_i to a specific signed axis within the MF M_{c_i} as described in Sec. 3.2. In the following sections it will be convenient to collect all variables of the kth MF into $\Psi_k = \{\mathbf{c}_k, \mathbf{z}_k, w_k, R_k, \mathbf{\Sigma}_k\}$ where $\mathbf{\Sigma}_k = \{\Sigma_{kj}\}_{j=1}^6$, $\mathbf{c}_k = \{c_i\}_{i:c_i=k}$ and $\mathbf{z}_k = \{z_i\}_{i:c_i=k}$ denote all labels c_i or z_i which are associated to the kth MF via $\mathbf{c}$. The MF axes of the kth MF are a function of the rotation R_k according to Eq. (3.2) and will be denoted $\{\mu_{kj}\}_{j=1}^6$.

First we define the MMF's probabilistic model before we outline a sampling-based-inference scheme. We restrict the analysis and inference method to the TG-MF model because the vMF distributions in the vMF-MF model necessitate more involved inference methods since the prior on the concentration does not have a closed form as mentioned in Sec. 3.2.3. Hence an internal slice sampler would be required to sample posterior concentration parameters for the vMF-MF.

3.4.1 Probabilistic Model

Figure 3.11 depicts a graphical representation of the probabilistic MMF model. It is a Bayesian finite mixture model that takes into account the geometries of both $\mathbb{S}^2$ and $\mathbb{SO}(3)$. In this probabilistic model, the MMF parameters are regarded as random variables and we avoid assumptions from Sec. 3.3 about weights and covariances of the individual MFs.

A surface normal n_i is associated with an MF via the assignment variable c_i. These MF-level assignments are assumed to be distributed according to a categorical distribution with a Dirichlet distribution prior with parameters α:

$$\pi \sim \mathrm{Dir}(\alpha) \tag{3.34}$$

$$c_i \sim \mathrm{Cat}(\pi)\,. \tag{3.35}$$

Each MF follows the mixture distribution outlined in Sec. 3.2.2 and hence a surface

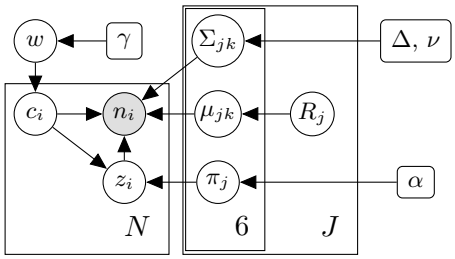


Figure 3.11: Graphical model for a mixture of K Manhattan Frames.

normal is distributed as

$$n_i \sim p\left(\Psi_{c_i}; \gamma, \Delta, \nu\right) . \tag{3.36}$$

We set $\alpha < 1$ to favor models with few MFs, which is typical for man-made scenes. Contemporary buildings, for example, customarily exhibit a small number of MFs. To encourage the association of equal numbers of normals to all MF axes, we place a strong prior $\gamma \gg 1$ on the distribution of axis assignments z_i. Intuitively, this encourages an MF to explain several normal directions and not just a single one.

■ 3.4.2 Metropolis-Hastings MCMC Inference

We perform inference over the probabilistic MMF model described in the previous section using Gibbs sampling with Metropolis-Hastings [105] split/merge proposals [197]. Specifically, the sampler iterates over the latent assignment variables $\mathbf{c}$ and $\mathbf{z}$, their categorical distribution parameters π and $\mathbf{w} = \{w_k\}_{k=1}^K$, as well as the covariances in the tangent spaces around the MF axes $\mathbf{\Sigma} = \{\mathbf{\Sigma}_k\}_{k=1}^K$ and the MF rotations $\mathbf{R} = \{R_k\}_{k=1}^K$. We first explain all posterior distributions needed for Gibbs sampling before we outline the algorithm.

Posterior Distributions for MCMC Sampling

The posterior distributions of both mixture weights are:

$$p(\pi \mid \mathbf{c}; \alpha) = \mathrm{Dir}(\alpha_1 + N_1, \ldots, \alpha_K + N_K) \tag{3.37}$$

$$p(w_k \mid \mathbf{c}, \mathbf{z}; \gamma) = \mathrm{Dir}(\gamma_1 + N_{k1}, \ldots, \gamma_{k6} + N_{k6}) , \tag{3.38}$$

where $N_k = \sum_{i=1}^N \mathbb{1}_{c_i=k}$ is the number of normals assigned to the kth MF and $N_{kj} = \sum_{i=1}^N \mathbb{1}_{c_i=k}\mathbb{1}_{z_i=j}$ is the number of normals assigned to the jth axis of the kth MF. The indicator function $\mathbb{1}_{a=b}$ is 1 if $a = b$ and 0 otherwise

Using the likelihood of n_i from Eq. (3.9), the conditional distributions for labels c_i

and z_i are given as:

$$p(c_i = k \mid \pi, n_i, \Theta) \propto \pi_k \sum_{j=1}^{6} w_{kj}\, p(n_i \mid \mu_{kj}, \Sigma_{kj}) \tag{3.39}$$

$$p(z_i = j \mid c_i, n_i, \Theta) \propto w_{c_i j}\, p(n_i \mid \mu_{c_i j}, \Sigma_{c_i j})\,, \tag{3.40}$$

where $\Theta = \{\mathbf{w}, \mathbf{\Sigma}, \mathbf{R}\}$. We compute $x_i = \mathrm{Log}_{\mu_{c_i z_i}}(n_i)$, the mapping of n_i into $T_{\mu_{c_i z_i}}\mathbb{S}^2$, to obtain the scatter matrix $S_{kj} = \sum_i^N \mathbb{1}_{c_i=k}\mathbb{1}_{z_i=j} x_i x_i^T$ in $T_{\mu_{kj}}\mathbb{S}^2$. Using S_{kj} the posterior distribution over covariances Σ_{kj} is:

$$p(\Sigma_{kj} \mid \mathbf{c}, \mathbf{z}, \mathbf{n}, \mathbf{R}; \Delta, \nu) = \mathrm{IW}\left(\Delta + S_{kj}, \nu + N_{kj}\right)\,. \tag{3.41}$$

Since there is no closed-form posterior distribution for an MF rotation given axis-associated normals, we approximate it as a narrow Gaussian distribution on $\mathbb{SO}(3)$ around the optimal rotation $R_k^\star$ under normal assignments $\mathbf{z}$ and $\mathbf{c}$:

$$p(R_k \mid \mathbf{z}, \mathbf{c}, \mathbf{n}) \approx \mathcal{N}(R_k; R_k^\star(R_k^0, \mathbf{z}, \mathbf{c}, \mathbf{n}), \Sigma_{\mathfrak{so}}(3))\,, \tag{3.42}$$

where $\Sigma_{\mathfrak{so}}(3) \in \mathbb{R}^{3\times 3}$ and R_k^0 is set to R_k from the previous Gibbs iteration. Refer to Sec. 2.7.1 for details on how to evaluate and sample from this distribution.

The (locally-) optimal rotation $R_k^\star \in \mathbb{SO}(3)$ of MF M_k given the assigned normals $\mathbf{n} = \{n_i\}_{i:c_i=k}$ and their associations z_i to one of the six axes μ_{kz_i} can be found using any of the MAP MF inference algorithms (i.e. Sec. 3.3.1 or 3.3.3).

Metropolis-Hastings MCMC Sampling

The Gibbs sampler with Metropolis-Hastings split/merge proposals is outlined in Algorithm 9. For K MFs and N normals the computational complexity per iteration is $O(K^2N)$. To let the order of the model adapt to the complexity of the distribution of normals on the sphere, we implement Metropolis-Hastings-based split/merge proposals. The details of the algorithm are described in the following sections.

■ 3.4.3 Split/Merge Proposals

Here we derive split and merge proposals for the MMF model as well as their acceptance probability in an approach similar to Richardson and Green [197]. Note that a merge involves moving all points from MF l and m into a new MF n and then removing MFs l and m. Similarly, a split creates two new MFs l and m from a single MF n. Hence, both a split and a merge change the number of parameters in the model. Specifically the parameters that change their dimension are the set of MF rotations, $\mathbf{R}$, and the set of covariances on the MF axes, $\mathbf{\Sigma}$. The labels $\mathbf{z}$ and $\mathbf{c}$ remain of the same dimensions; only the range for $\mathbf{c}$ changes from $[1, K]$ to $[1, K-1]$ (merge) or from $[1, K]$ to $[1, K+1]$ (split). Therefore, we employ the theory of Reversible Jump Markov Chain Monte Carlo (RJMCMC) [94] to derive a proper acceptance probability. RJMCMC is a

Draw $\pi \mid \mathbf{c}; \alpha$ using Eq. (3.37)
Draw $\mathbf{c} \mid \pi, \mathbf{n}, \mathbf{R}, \Sigma$ in parallel using Eq. (3.39)
for $k \in \{1, \ldots, K\}$ **do**
    Draw $w_k \mid \mathbf{c}, \mathbf{z}; \gamma$ using Eq. (3.38)
    Draw $\mathbf{z} \mid \mathbf{c}, \mathbf{w}, \mathbf{n}, \mathbf{R}, \boldsymbol{\Sigma}$ in parallel using Eq. (3.40)
    Draw $R_k \mid \mathbf{z}, \mathbf{c}, \mathbf{n}; \Sigma_{\mathfrak{so}}(3)$ using Eq. (3.42)
    Draw $\{\Sigma_{kj}\}_{j=1}^{6} \mid \mathbf{c}, \mathbf{z}, \mathbf{n}, \mathbf{R}; \Delta, \nu$ using Eq. (3.41)
end for
Propose splits for all MFs
Propose merges for all MF combinations

Algorithm 9: One iteration of the MMF inference algorithm.

generalization of Metropolis-Hastings MCMC [105] and provides a way of computing an acceptance probability when the number of parameters changes between moves. We will see that the split/merge proposals as well as the acceptance probabilities are similar to what one would expect from the Metropolis-Hastings algorithm. For this reason and because the MH algorithm is more well-known, we chose to refer to the inference algorithm as to Metropolis-Hastings MCMC.

RJMCMC Split/Merge Moves in an MMF

RJMCMC utilizes auxiliary variables to propose deterministic moves to change between model orders. In the following, we will give the RJMCMC algorithm for a merge proposal between two MFs. The inverse proposal of a split of an MF into two MFs follows the same but inverted process.

Let a MF A be parameterized by the random variables Ψ_A. An RJMCMC merge proposal between MFs m and l is executed in three steps. First, an auxiliary MF v is sampled from $q(\text{merge})$ to propose a merge of the current MFs l and m as will be described in Section 3.4.3. Second, the deterministic function $f([\Psi_l, \Psi_m, \Psi_v]) = [\mathbf{u}_1, \mathbf{u}_2, \hat{\Psi}_n]$ is used to obtain the merged MF n parameterized by $\hat{\Psi}_n$. The auxiliary MFs $\mathbf{u}_1$ and $\mathbf{u}_2$ absorb the MFs l and m from before the merge. The function $f([\Psi_l, \Psi_m, \Psi_v])$ is hence defined as

$$\mathbf{u}_1 = \Psi_l\,, \quad \mathbf{u}_2 = \Psi_m\,, \quad \hat{\Psi}_n = \Psi_v\,. \tag{3.43}$$

Therefore, the Jacobian J_f of the function $f([\Psi_l, \Psi_m, \Psi_v])$ is

$$J_f = \frac{\partial f([\Psi_l, \Psi_m, \Psi_v])}{\partial [\Psi_l, \Psi_m, \Psi_v]} = \mathrm{I}\,, \tag{3.44}$$

where I is the identity matrix with determinant 1.

Third, the proposed merge is accepted with probability

$$\min\left\{1, \frac{\prod_{k=1}^{K-1} p(\hat{\Psi}_k;\alpha,\gamma,\Delta,\nu)}{\prod_{k=1}^{K} p(\Psi_k;\alpha,\gamma,\Delta,\nu)} \frac{q(\text{split})}{q(\text{merge})} \det(J_f)\right\}, \tag{3.45}$$

where parameters after the merge are designated with a hat. The proposal distributions for a split of MFs l and m into MF n is denoted $q(\text{split})$.

The RJMCMC split proposal of an MF n into MFs l and m follows the same process except that a split is proposed according to Sec. 3.4.3 instead of a merge. The deterministic transformation is the inverse of $f(\cdot)$. This means that the determinant of the Jacobian is 1 and the acceptance probability for the split is Eq. 3.45 where the ratio has been inverted.

Note that the RJMCMC acceptance probability for split/merge moves in an MMF looks like the Metropolis-Hastings acceptance probability, because $|\det(J_f)| = 1$. However, since the model orders in the nominator and denominator of the fractions are different, it technically is not a Metropolis-Hastings acceptance probability.

Merge Proposal in an MMF

Let the two MFs l and m be parameterized by the random variables Ψ_l and Ψ_m. A merged MF n can be sampled from the current MFs l and m as follows. We first assign all normals of MF l and m to MF n: $\mathbf{c}_{\mathbf{c}\in\{l,m\}} = n$, which corresponds to the proposal distribution:

$$q(\mathbf{c}_l, \mathbf{c}_m \mid \mathbf{c}) = \delta(\{\mathbf{c}_l, \mathbf{c}_m\} - n)\,. \tag{3.46}$$

Second, we sample the axes assignments $\mathbf{z}_n$ according to

$$q(z_i = j \mid w_l, R_l, \mathbf{\Sigma}_l, \mathbf{n}) \quad \propto \quad w_{lj}\, p(n_i; \mu_{lj}, \Sigma_{lj})\,. \tag{3.47}$$

Next, given associations $\mathbf{c}_n$ and $\mathbf{z}_n$, we find the optimal rotation using the closed form solution of the vMF-based model derived in Sec. 3.3.3. This is justified because the direct and the vMF-based algorithms generally found the same optimum in our experiments. Then we sample R_n from a narrow Gaussian distribution over rotations with mean $R_n^\star$:

$$\begin{aligned} q(R_n \mid \mathbf{z}, \mathbf{c}, \mathbf{n}, R_l) &= \mathcal{N}(R_n; R_n^\star(\mathbf{z}_n, \mathbf{c}_n, \mathbf{n}), \Sigma_{\mathfrak{so}}(3)) \\ &= \mathcal{N}((R_n^{\star T} \text{Log}_{R_n^\star(\cdot)}(R_n))^\vee; 0, \Sigma_{\mathfrak{so}}(3))\,, \end{aligned} \tag{3.48}$$

where $\text{Log}_{R_n^\star}(R) : \mathbb{SO}(3) \to T_{R_n^\star}\mathbb{SO}(3)$ denotes the logarithm map of R into the tangent space $T_{R_n^\star}\mathbb{SO}(3)$ around $R_n^\star$. The vee operator $^\vee$ [45] extracts the unique elements of a skew-symmetric matrix $W \in \mathbb{R}^{3\times3}$ into a vector w: $W^\vee = w = [-W_{23}; W_{13}; -W_{12}] \in \mathbb{R}^3$. $\Sigma_{\mathfrak{so}}(3) \in \mathbb{R}^{3\times3}$ is the covariance of the Normal distribution in $T_{R_n^\star}\mathbb{SO}(3)$. Refer to Sec. 2.7.1 for an in depth discussion.

Finally, we obtain samples for the axis covariances $\mathbf{\Sigma}_n$ according to the proposal distribution

$$q(\mathbf{\Sigma}_n \mid \mathbf{c}, \mathbf{z}, R_n, \mathbf{n}) = \prod_{j=1}^{6} p(\Sigma_{nj} \mid \mathbf{z}_n, \mathbf{c}_n, \mathbf{n}, R_n)\,, \tag{3.49}$$

where $p(\Sigma_{nj} \mid \mathbf{z}_n, \mathbf{c}_n, \mathbf{n}, R_n; \Delta, \nu)$ is the posterior distribution over covariance Σ_{nj} under the IW prior given the assigned normals in the tangent space $T_{\mu_{nj}}\mathbb{S}^2$.

The proposal of merging MF l and m into MF n factors as

$$\begin{aligned} q(\text{merge}) =& q(\Psi_n \mid \Psi_l, \Psi_m, \mathbf{n}; \alpha, \gamma, \Delta, \nu) = q(\mathbf{c}_l, \mathbf{c}_m \mid \mathbf{c}) \\ & q(R_n \mid R_l, \mathbf{z}, \mathbf{c}, \mathbf{n}) q(\mathbf{\Sigma}_n \mid \mathbf{c}, \mathbf{z}, \mathbf{n}, R_n; \Delta, \nu) \prod_{i:c_i=n} q(z_i \mid w_l, R_l, \mathbf{\Sigma}_l, \mathbf{n})\,. \end{aligned} \tag{3.50}$$

Split Proposal in an MMF

First, we randomly assign normals in MF n to MF l or m by drawing MF labels according to the Dirichlet Multinomial (DirMult) distribution:

$$q(\mathbf{c}_n \mid \mathbf{c}; \alpha) = \text{DirMult}\,(\mathbf{c}_n; \alpha_l, \alpha_m) = \int_\pi \text{Cat}(\mathbf{c} \mid \pi)\,\text{Dir}(\pi; \alpha_l, \alpha_m)\,\mathrm{d}\pi\,, \tag{3.51}$$

$$\text{DirMult}(\mathbf{c}; \alpha) = \frac{\Gamma(\sum_{k=1}^K \alpha_k)}{\Gamma(\sum_{k=1}^K \alpha_k + N_k)} \prod_{k=1}^K \frac{\Gamma(\alpha_k + N_k)}{\Gamma(\alpha_k)}\,. \tag{3.52}$$

and the counts N_k of labels $c_i = k$ are $N_k = \sum_{i=1}^N \mathbb{1}_{c_i=k}$.

Within each of the MFs l and m we assign normals $\mathbf{n}$ to an axis by drawing the assignments $\mathbf{z}_n$ as

$$q(z_i = j \mid w_n, R_n, \mathbf{\Sigma}_n, \mathbf{n}) \propto w_{nj}\, p(n_i; \mu_{nj}, \Sigma_{nj})\,. \tag{3.53}$$

Using these assignments, we find optimal rotations $R_l^\star$ and $R_m^\star$ and draw R_l and R_m:

$$\begin{aligned} q(R_l, R_m \mid \mathbf{z}, \mathbf{c}, \mathbf{n}, R_n) =& \mathcal{N}(R_l; R_l^\star(R_n, \mathbf{z}, \mathbf{c}, \mathbf{n}), \Sigma_{\mathfrak{so}}(3)) \\ & \mathcal{N}(R_m; R_m^\star(R_n, \mathbf{z}, \mathbf{c}, \mathbf{n}), \Sigma_{\mathfrak{so}}(3))\,. \end{aligned} \tag{3.54}$$

Given rotations as well as labels, we can draw axis covariances $\mathbf{\Sigma}_{\{l,m\}}$ from the respective posterior:

$$q(\mathbf{\Sigma}_{\{l,m\}} \mid \mathbf{c}, \mathbf{z}, \mathbf{n}, R_{\{l,m\}}; \Delta, \nu) = \prod_{j=1}^{6} p(\Sigma_{lj} \mid \mathbf{z}, \mathbf{c}, \mathbf{n}, R_l) p(\Sigma_{mj} \mid \mathbf{z}, \mathbf{c}, \mathbf{n}, R_m) \tag{3.55}$$

The split proposal distribution factors as

$$\begin{aligned} q(\text{split}) =& q(\mathbf{x}_l, \mathbf{x}_m \mid \mathbf{x}_n, \mathbf{n}; \alpha, \gamma, \Delta, \nu) \\ =& q(\mathbf{c}_n \mid \mathbf{c}; \alpha) q(R_{\{l,m\}} \mid \mathbf{z}, \mathbf{c}, \mathbf{n}, R_n) q(\mathbf{\Sigma}_{\{l,m\}} \mid \mathbf{c}, \mathbf{z}, \mathbf{n}, R_{\{l,m\}}) \\ & \prod_{i:c_i=n} q(z_i \mid w_n, R_n, \mathbf{\Sigma}_n, \mathbf{n})\,. \end{aligned} \tag{3.56}$$

RJMCMC Acceptance Probability

After introducing the RJMCMC merge and the split proposals in the previous sections, we will now derive the acceptance probabilities for those two moves by detailing the distributions involved in the computation of Eq. (3.45).

The joint distribution for the MMF model is defined by the graphical model depicted in Fig. 3.11. For the evaluation of the acceptance probability, we marginalize over the categorical variables π and $\mathbf{w}$ as in the split proposal in Eq. (3.51):

$$p(\mathbf{c};\alpha) = \int_{\pi} p(\mathbf{c} \mid \pi) p(\pi;\alpha)\, \mathrm{d}\pi = \mathrm{DirMult}(\mathbf{c};\alpha) \tag{3.57}$$

$$p(\mathbf{z}_k \mid \mathbf{c};\gamma) = \int_{w_k} p(\mathbf{z}_k \mid \mathbf{c}, w_k) p(w_k;\gamma)\, \mathrm{d}w_k = \mathrm{DirMult}(\mathbf{z}_k;\gamma)\,., \tag{3.58}$$

After marginalization of π and $\mathbf{w}$, the joint distribution is:

$$\begin{aligned} p(\mathbf{n},\mathbf{c},\mathbf{z},\boldsymbol{\Sigma},\mathbf{R};\alpha,\gamma,\Delta,\nu) =& p(\mathbf{c};\alpha) \prod_{j}^{6} p(\Sigma_{kj};\Delta,\nu) \prod_{i=1}^{N} p(n_i \mid c_i, z_i, R_{c_i}, \Sigma_{c_i z_i}) \\ & \prod_{k=1}^{K} p(R_k) p(\mathbf{z}_k \mid \mathbf{c};\gamma)\,, \end{aligned} \tag{3.59}$$

where we have assumed that the prior over rotations factors according to $p(\mathbf{R}) = \prod_{k=1}^{K} p(R_k)$. Therefore, the ratio of joint probabilities in the merge move acceptance probability in Eq. (3.45) becomes

$$\begin{aligned} \frac{p(\mathbf{n},\hat{\mathbf{c}},\hat{\mathbf{z}},\hat{\boldsymbol{\Sigma}},\hat{\mathbf{R}};\alpha,\gamma,\Delta,\nu)}{p(\mathbf{n},\mathbf{c},\mathbf{z},\boldsymbol{\Sigma},\mathbf{R};\alpha,\gamma,\Delta,\nu)} &= \frac{p(\hat{\mathbf{c}};\alpha) p(\mathbf{n} \mid \hat{\mathbf{c}},\hat{\mathbf{z}},\hat{\mathbf{R}},\hat{\boldsymbol{\Sigma}}) p(\hat{\mathbf{z}} \mid \hat{\mathbf{c}};\gamma) p(\hat{\boldsymbol{\Sigma}};\Delta,\nu) p(\hat{\mathbf{R}})}{p(\mathbf{c};\alpha) p(\mathbf{n} \mid \mathbf{c},\mathbf{z},\mathbf{R},\boldsymbol{\Sigma}) p(\mathbf{z} \mid \mathbf{c};\gamma) p(\boldsymbol{\Sigma};\Delta,\nu) p(\mathbf{R})} \\ &= \frac{8\pi^2 p(\hat{\mathbf{c}};\alpha) \left(\prod_i^N p(n_i \mid \hat{c}_i, \hat{z}_i, \hat{\mathbf{R}}, \hat{\boldsymbol{\Sigma}})\right) \prod_{k=1}^{\hat{K}} p(\hat{\mathbf{z}}_k \mid \hat{\mathbf{c}};\gamma) \prod_{j=1}^{6} p(\hat{\Sigma}_{kj};\Delta,\nu)}{p(\mathbf{c};\alpha) \left(\prod_i^N p(n_i \mid c_i, z_i, \mathbf{R}, \boldsymbol{\Sigma})\right) \prod_{k=1}^{K} p(\mathbf{z}_k \mid \mathbf{c};\gamma) \prod_{j=1}^{6} p(\Sigma_{kj};\Delta,\nu)}\,, \end{aligned} \tag{3.60}$$

where $\hat{K} = K - 1$. For a split proposal this ratio is inverted.

The acceptance probability of splits and merges of MFs can be computed, by plugging Eq. (3.60) into Eq. (3.45).

■ 3.5 Evaluation and Results

We evaluate the properties and performance of the real-time MF (RTMF) before the MMF inference algorithms. All evaluations were run on an Intel Core i7-3940XM CPU at 3.00GHz with an NVIDIA Quadro K2000M GPU.

■ 3.5.1 Evaluation of Real-time MAP Inference

We show run-times and rotation estimation accuracy of all three derived real-time MF inference (RTMF) algorithms on two datasets with groundtruth (GT) camera rotations

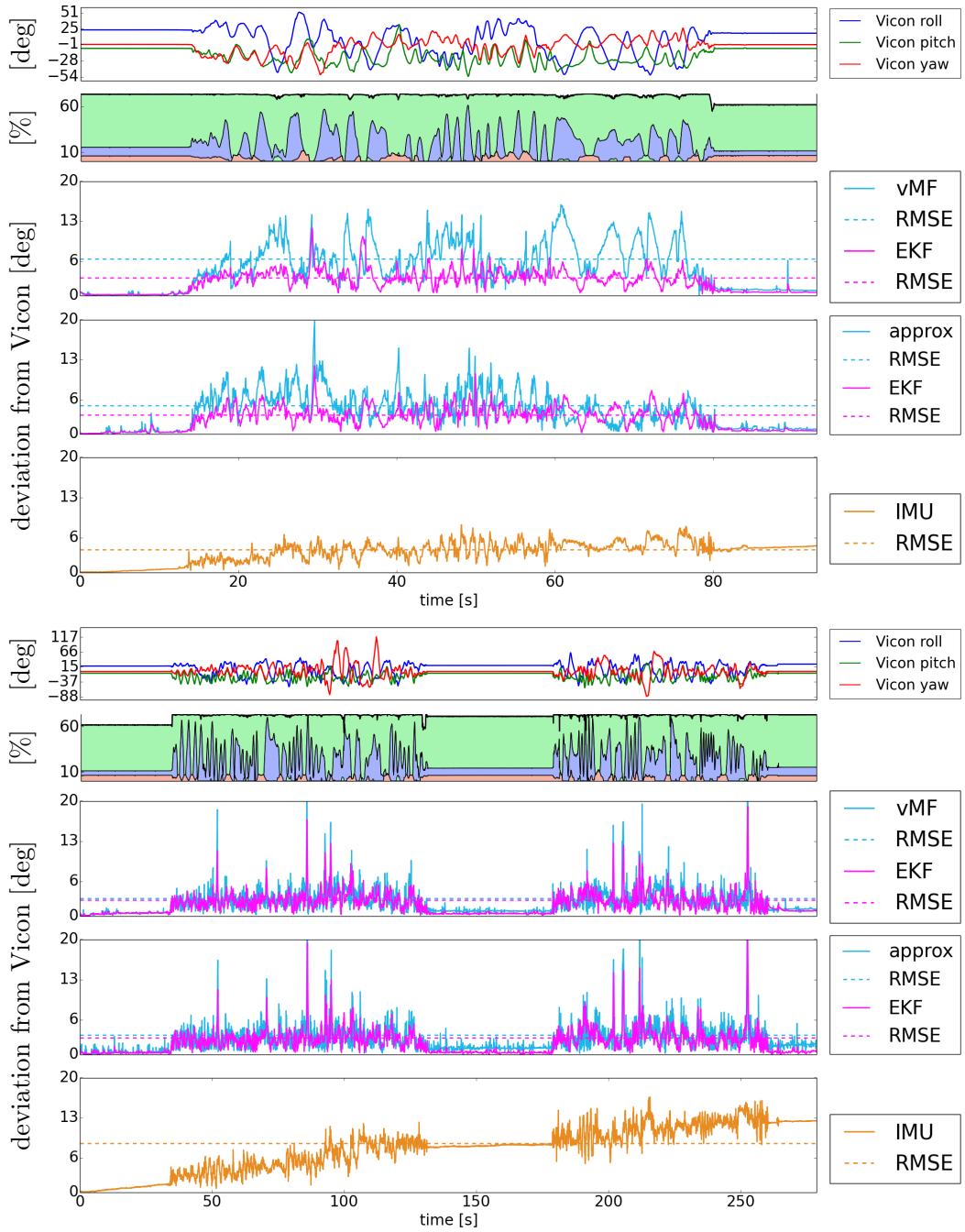


Figure 3.12: Two different datasets displayed below each other with groundtruth data (first rows). The percentages of points associated to a respective Manhattan Frame axis over time is color-coded in the second rows. Rows two to four show the angular deviation from the groundtruth of the approximate and the vMF-based RTMF algorithm with and without fusion with the IMU. The IMU orientation estimate is displayed in the last rows. Note that the RTMF algorithms' rotation estimates are drift free.

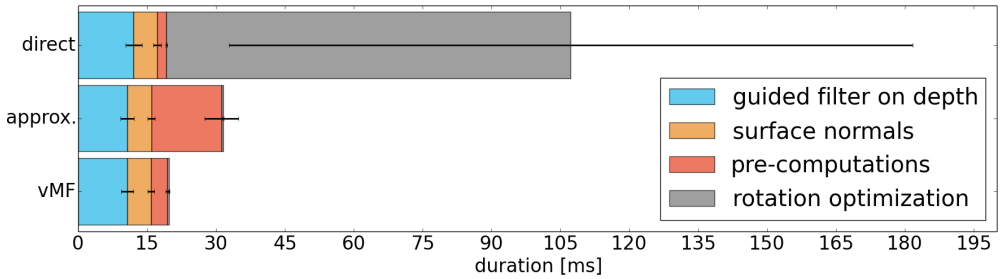


Figure 3.13: Timing breakdown for the three different real-time Manhattan Frame algorithms. The error bars show the one-σ range.

from a Vicon motion-capture system. The datasets were obtained by waving a Xtion Pro RGBD camera randomly in full 3D motion up-down as well as left-right in front of a simple MW scene for 90 s and 4:30 min respectively as can be seen in the GT yaw-pitch-roll angles in the first row of Fig. 3.12.

The approximate RTMF algorithm was run for 25 iterations at most while the direct RTMF algorithm was run for at most ten to keep computation time low. Any fewer iterations rendered the direct MF rotation estimation unusable.

Timings We split the computation times into the following stages: (1) applying a guided filter to the raw depth image, (2) computing surface normals from the smoothed depth image, (3) pre-computing of data statistics and (4) optimization for the MF rotation. The timings shown in Fig. 3.13 were computed over all frames of the dataset. At 111 ms per frame the direct method cannot be run in real-time. While the approximate method improves the runtime, it is 15 ms slower than the vMF-based approach which runs in 18 ms. The approximate algorithm is slower since the Karcher mean pre-computation is iterative whereas the vMF pre-computation is single pass. As intended by shifting all computations over the full data into a pre-processing step, the latter two RTMF algorithms can be run at a camera frame-rate of 30 Hz. In the following we omit the direct method from the evaluation due to its slow runtime.

Accuracy Besides the evaluation of the rotation estimates of the proposed RTMF algorithms, we show the rotation estimates obtained by integrating rotational velocities measured by a Microstrain 3DM-GX3 IMU using an EKF as proposed in [236]. The state of the EKF contains the estimated fused rotation represented as a Quaternion and the bias of the IMU. While the rotational velocities measurements of the IMU are used for the EKF prediction step, the RTMF rotation estimate is used in the update step. For the shorter groundtruth dataset of 90 s length, displayed on the top in Fig. 3.12, we obtain an angular RMSE from the Vicon groundtruth rotation of 6.36° for the vMF-based algorithm and 4.92° for the approximate method. The IMU rotation estimate drifts and exhibits an RMSE of 3.91°. Fusing the RTMF rotation estimates with the IMU using the EKF achieves even lower RMSEs of 3.05° for the vMF-based and 3.28° for the approximate method. Figure 3.12 on the bottom shows the angular deviation

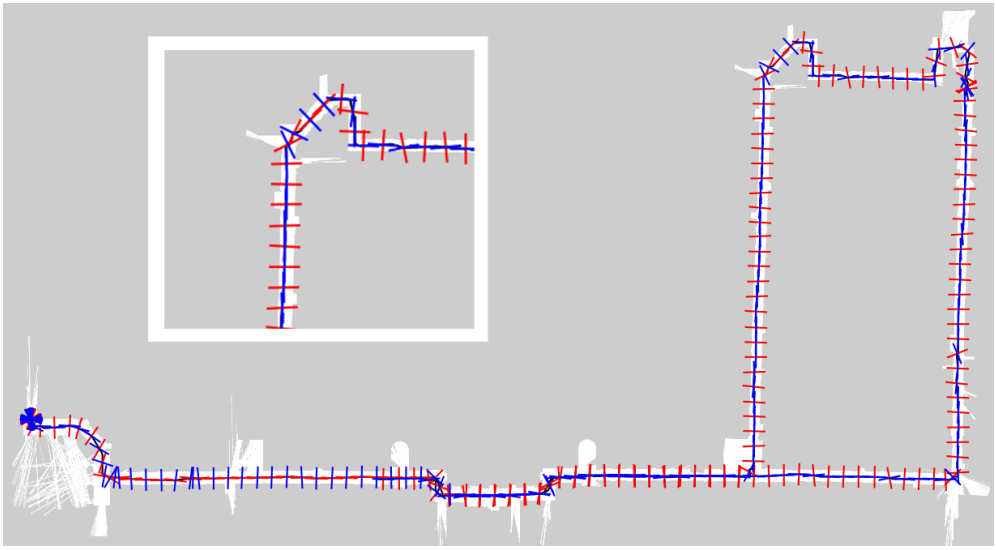


Figure 3.14: MF orientations extracted as a Turtlebot V2 traverses the hallways around Killian court in the main building of MIT. The zoomed in area displays the top left corner of the loop. Note that the orientations align with the local MW structure of the environment.

from Vicon groundtruth during a longer sequence of about 4:30 min taken at the same location. Similar to the shorter sequence, the two MF rotation estimation algorithms exhibit zero drift and an RMSE below 3.4°. The drift of the IMU is clearly observable and explains the high RMSE of 8.50°. Note that while the drift of the IMU can be reduced by utilizing acceleration and magnetic field data, it can not be fully eliminated. The fusion of the rotation estimates using the Quaternion EKF again improves the RMSEs to below 2.8°.

The percentages of surface normals associated with the MF axes displayed in the second rows of Fig. 3.12 support the intuition that a less uniform distribution of normals across the MF axes results in a worse rotation estimate: large angular deviations occur when there are surface normals on only one or two MF axes for several frames.

The angular accuracy results also highlight the complementary nature of the rotational estimates from IMU and RTMF algorithms: the IMU's rotation estimates over short timescales complements the MF rotation estimates if it is not well constrained. In turn the RTMF rotation estimates helps estimate and thus eliminate the bias from the gyroscope measurements. Note that while the drift of the IMU can be reduced by utilizing acceleration and magnetic field data, it can not be fully eliminated.

Manhattan Frame Inference on the Killian Court Dataset For this experiment a Turtlebot V2 robot equipped with a laser-scanner-based SLAM system was driven through the hallways surrounding Killian court in the main building of MIT. We ran the vMF-based RTMF algorithm on the depth stream from the Kinect camera of the robot.

Figure 3.15: In each row the MF segmentations of several scenes from the NYU depth dataset [173] are displayed for one of the different RTMF algorithms. For the direct method the third row displays results with the real-time configuration whereas the fourth row (direct HQ) shows segmentations obtained with slower but theoretically higher quality parameter settings. For each image, the segmentation is overlaid on top of the grayscale image of the respective scene. Note that unlabeled areas are due to lack of depth data. The inferred MF orientation is shown in the bottom right corner of each frame.

Figure 3.14 shows the inferred MF rotations for every 200th frame at the respective position obtained via the SLAM system. It can be seen that the algorithm correctly tracks the orientation of the *local MW*. A part of the hallway on the top right does not align with the overall MW orientation. The estimated orientations are thus aligned with this local MW, which is at an angle with respect to the rest of the map. This highlights that the MW assumption is best treated as a local property of the environment as argued in the introduction. In parts of the map without nearby structure the MW rotation estimate is off due to the lack of data.

Manhattan World Scene Segmentation As a by-product of the MF rotation estimate the algorithm also provides a segmentation of the frame into the six different orthogonal and opposite directions. This segmentation can be used as an additional source of information for further processing. For example, using the direction of gravity it would be easy to extract the ground plane for obstacle avoidance. We show several examples of segmented scenes taken from the NYU depth dataset [173] in Fig. 3.15. The RTMF algorithms used the same parameters as before. The segmentations show that the vMF-based and the approximate method perform well on a wide range of cluttered

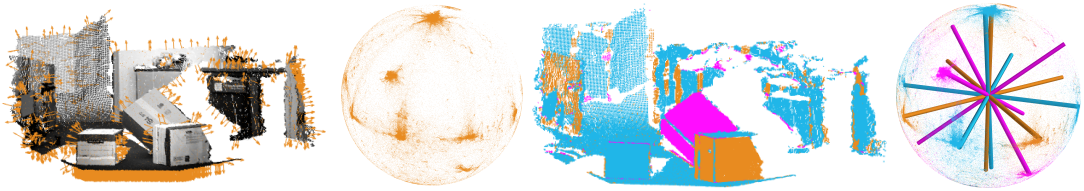


Figure 3.16: A complex indoor scene (left) and its surface normal distribution on the unit sphere (middle left). The MMF inference algorithm converges to three distinct Manhattan Frames, shown in different colors to the right. The inferred Manhattan Frames imply a directional scene segmentation as shown in the middle right.

scenes. The direct algorithm with a fine-grained line-search in the conjugate gradient optimization (denoted direct HQ) gives similar results to the two other approaches but is significantly slower.

■ 3.5.2 Evaluation of MMF Inference

We now evaluate MMF inference on various datasets across scales and compare against MF and VP estimation algorithms.

With the RJMCMC-based approach, we infer an MMF in a coarse-to-fine approach. First, we down-sample to 120k normals and run the algorithm for $T = 150$ iterations, proposing splits and merges throughout as described in Sec. 3.4.2. We use the following parameters: $\Sigma_{\mathfrak{so}}(3) = (2.5°)^2\ \mathrm{I}_{3\times3}$, $\alpha = 0.01$, $\gamma = 120$, $\nu = 12k$, and $\Delta = (11°)^2\nu\ \mathrm{I}_{2\times2}$. For the purpose of displaying results, we obtain MAP estimates from samples from the posterior distribution of the MMF. First, we find the most likely number of MFs $K^\star$ from all samples after a burn-in of 100 RJMCMC iterations. We then run MCMC starting form the latest sample that has $K^\star$ MFs using all data without proposing splits and merges. All MMF results displayed herein show the last MMF sample of that chain.

The vMF-MMF MAP inference algorithm is sensitive to the initial MF rotations. Hence, we run it 11 times each time starting from 6 randomly rotated MFs and choose one of the models with the moste likely number of MFs after discarding MFs with less than 10% of surface normals.

MMF Inference from Depth Images

We first highlight different aspects and properties of the inference using the 3-box scene depicted in Fig. 3.16. For this scene, we initialized the number of MFs to $K = 6$. The algorithm correctly infers $K = 3$ MFs corresponding to the three differently rotated boxes as displayed in Fig. 3.16 on the sphere and in the point cloud. While the pink Manhattan Frame consists only of the single box standing on one corner, the turquoise and orange MFs contain planes of the surrounding room in addition to their respective boxes. This highlights the ability of our model to pool normal measurements from the whole scene.

Figure 3.17: Segmentation and inferred (M)MF of various indoor scenes partly taken from the NYU V2 depth dataset [173]. The first and second row shows MAP inference results under the von-Mises-Fisher Manhattan Frame model and the vMF MMF model respectively. The third row shows the inferred MMF model using RJMCMC. For single-MF scenes we color-code the assignment to Manhattan Frame axes and for MMF scenes the assignments to MFs.

In Fig. 3.17 we show several typical indoor scenes of varying complexity and the inferred Manhattan Frame using the vMF-based RTMF algorithm, the MMF inferred by the MAP von-Mises-Fisher MMF algorithm (vMF-MMF) and the MMF inferred by the RJMCMC algorithm (MMF). The MMF inference algorithms were started with six MFs in all cases. For the single MW scenes, all these algorithms infer the same Manhattan Frame, for the multiple-MW scenes the MMF and the MAP-MMF algorithm infer the same reasonable MFs while the vMF RTMF algorithm seems to pick the most prominent Manhattan Frame.

Besides poor depth measurements due to reflections, strong ambient light, black surfaces, or range limitations of the sensor, the inference converged to the wrong number of MFs mainly because of violations of the MW assumption such as round objects or significant clutter in the scene. We observe that the algorithm fails gracefully, approximating round objects with several MFs or adding a "noise MF" to capture clutter as can be seen in Fig. 3.18. Hence, to eliminate "noise MFs", we consider only MFs with more than 10% of all normals for the following quantitative evaluation. A more principled approach that could be explored is to explicitly instantiate a noise model in the form of an uniform distribution over the sphere. For the RJMCMC-based inference approach the inference would not significantly change.

To evaluate the performance of the MMF inference algorithms, we ran them on the NYU V2 dataset [173] which contains 1449 RGB-D images of various indoor scenes.

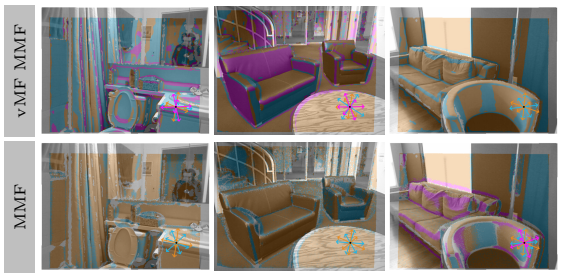


Figure 3.18: Failure cases for the MAP (first row) and RJMCMC MMF (second row) inference. MMF inference fails due to bad, noisy or erroneous surface normal estimates and if the multiple Manhattan World assumption is violated. In the case of round objects, a common violation of the Manhattan World assumption, the MMF inference fails gracefully by approximating the surfaces via multiple Manhattan Frames.

Inference of the number of Manhattan Frames For each scene, we compare the number of MFs the algorithm infers to the number of MFs a human annotator perceives. The confusion matrices for the two MMF algorithms are:

$$C_{\text{MMF}} = \begin{pmatrix} 557 & 467 & 108 & 3 & 0 \\ 130 & 152 & 28 & 1 & 1 \end{pmatrix} \tag{3.61}$$

$$C_{\text{vMF MMF}} = \begin{pmatrix} 528 & 283 & 186 & 138 \\ 37 & 118 & 83 & 74 \end{pmatrix} \tag{3.62}$$

The MMF algorithm infers the human perceived MMFs in 49.0% of the scenes while vMF-MMF is slightly worse with 44.6% and a tendency to overestimate the number of MFs.

Manhattan World orientation accuracy We use the groundtruth MW orientation of the most prominent MW provided by [87] to directly evaluate MW orientation estimation accuracy. We take into account that the same MF axes defined according to Eq. (3.2) can be described by 24 rotations $\{R_{\text{MF},i}\}_{i=1}^{24}$. These are constructed as: for all six permutations of choosing two columns from $R_{\text{MF},i}$, r and r', construct four rotation matrices:

$$[r, r', r_\times],\ [-r, r', -r_\times],\ [r, -r', -r_\times],\ [-r, -r', r_\times] \tag{3.63}$$

where $r_\times = r \times r'$. To compute the angular deviation θ of an estimated MF to the ground truth Manhattan Frame rotation we construct the set $\{R_{\text{MF},i}\}_{i=1}^{24}$ from the inferred MF rotation, compute all angular rotation deviations to the groundtruth R_{GT} and choose the smallest deviation:

$$\theta = \min_{i \in \{1,\ldots,24\}} \arccos\left(\tfrac{1}{2}\operatorname{tr}\left(R_{\text{GT}}^T R_{\text{MF},i}\right) - \tfrac{1}{2}\right) \tag{3.64}$$

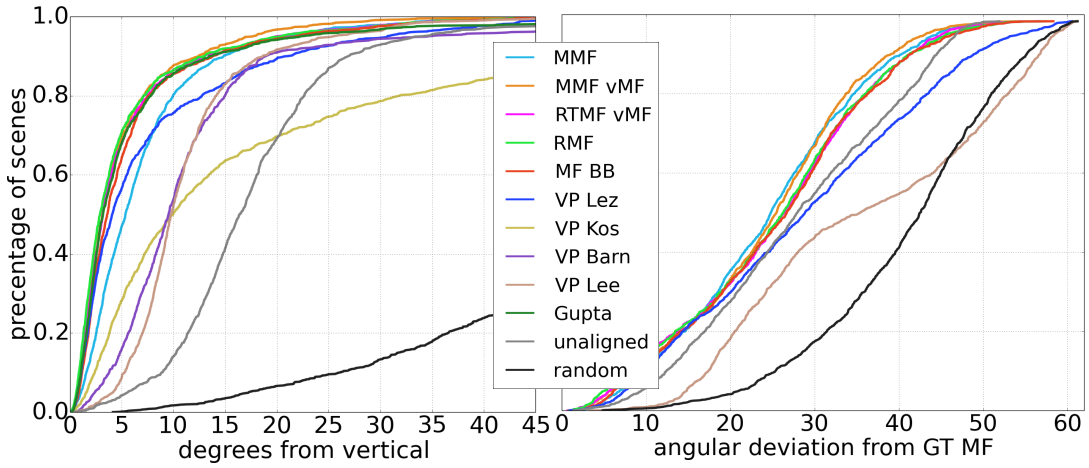


Figure 3.19: Cumulative density functions of deviations from true gravity and true Manhattan World orientation for different Manhattan Frame and vanishing point algorithms.

In case of MMFs we choose the smallest deviation across MFs.

Figure 3.19 (right) depicts cumulative distribution functions (CDF) for the angular deviation of the different MF and MMF algorithms, and two VP extraction algorithms, [149] (VP Lez) and [146] (VP Lee), which extract three OVPs. Evidently, MF algorithms estimate the MW rotation more accurately than the OVP algorithms. The MMF algorithms show higher accuracy on the whole dataset than single MF algorithms.

Gravity direction estimation Like the algorithms by Silberman et al. [173] and Gupta et al. [96] the proposed MF and MMF inference algorithms can be used to estimate the gravity direction to facilitate rotating scenes into a canonical frame for scene understanding. VPs are also indicative of the gravity direction and we show the performance of two additional VP algorithms [139] (VP Kos) and [19] (VP Barn). The mean direction of surface normals in the scene parts labeled as "floor" serves as a proxy for the true gravity direction in the evaluation.

The cumulative density functions of the angular deviation from the gravity direction in Fig. 3.19 (left) demonstrates that all Manhattan Frame inference algorithms match the performance of Gupta et al. and clearly outperform all VP-based estimates. The inferred MMF models outperform all other methods because of the higher flexibility of the model. The Manhattan Frame algorithms all show similar performance.

Timing Table 3.1 gives an overview of run-times for the different algorithms averaged over the 1449 scenes from the NYU V2 dataset. RTMF-vMF is the fastest algorithm while the sampling-based algorithm is, unsurprisingly, the slowest. It could, however, be sped up, e.g., by employing a sub-cluster approach for split-merge proposals [42, 224].

Method	RTMF	MMF vMF	MMF	RMF	MF BB	Lez	Lee	Kos	Barn
Time [s]	**0.037**	0.18	3312	23.6	0.061	3.99	5.76	0.21	0.015

Table 3.1: Comparison of Manhattan Frame and vanishing point algorithm timings over the whole NYU V2 dataset.

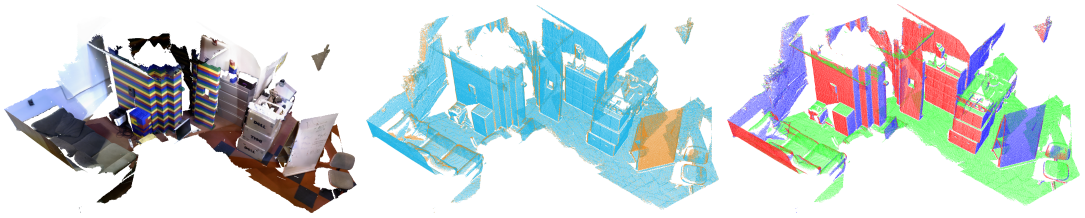


Figure 3.20: MMF extracted from a mesh obtained using Kintinuous [246]. Note that for an robot the MMF segmentation is immediately useful: All green areas are either traversable or locations that could harbor objects the robot might be looking for. Blue and red areas should not be bumped into but might for example contain door signs.

Additional Qualitative MMF Inference Results

Triangulated meshes Algorithms such as Kintinuous [246] and Elastic Fusion [248] allow dense larger scale indoor reconstructions from a stream of RGBD frames as depicted in to the right in Fig. 3.20 for a couch area with some boxes, shelves and a Lego house. Using the triangles' surface normals, the MMF can be inferred. The associations to one of the inferred Manhattan Frames is shown in the middle of Fig. 3.20 while the associations to the Manhattan Frame axes within each of the Manhattan Frames is shown to the right. Note that a robot could use the MMF segmentation directly: all green areas are either traversable or locations that could harbor objects the robot might be looking for. Blue and red areas should not be run into but might, for example, contain door signs.

LiDAR data The large-scale LiDAR scan of Cambridge (Fig. 3.21 left) has few measurements on the sides of buildings due to reflections off the glass facades and inhomogeneous point density because of overlapping scan-paths. To handle these properties, we implement a variant of robust moving-least-squares normal estimation [77]. The local plane is estimated using RANSAC, based on a preset width that defines outliers of the plane model. The normal votes are averaged for each point from neighboring estimates based on a Gaussian weight with respect to the Euclidean distance from the estimator. We count only votes whose estimation had sufficient support in the RANSAC computation in the nearby point set. Figure 3.21 to the left shows the point cloud colored according to Manhattan Frame assignment of the normals overlaid on a gray street-map. The inferred Manhattan Frames share the upward direction without imposing any constraints. Interestingly, the MMF captures large scale organizational structure in this man-made environment: blue and green are the directions of Boston and Harvard respectively,

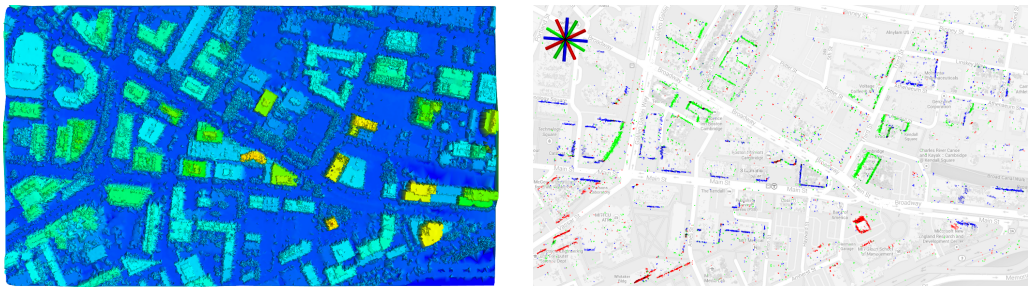


Figure 3.21: Left: Mesh of Kendall Square color-coded by height. Right: Inferred MMF overlayed on top of a gray street map. There is a clear separation into three MFs colored red, green and blue with the orientations indicated by the axes in the top-left corner. These MFs share the upward direction without imposing any constraints. Normals associated with upward axes are hidden for clarity. Note that the underlying point cloud has varying density due to the scan-paths of the airplane.

and red is aligned with the Charles river. The locations belonging to the Manhattan Frames are spatially separated supporting that the MW assumption is best treated as a local property as argued in the introduction.

■ 3.6 Discussion

Guided by the observation that regularities in man-made environments manifest in structured surface normal distributions, we have proposed the Manhattan Frame model which captures the Manhattan World assumption in the space of surface normals. We have formalized the notion of the Manhattan Frame and explored two different probabilistic models and resulting maximum a-posteriori inference algorithms. These real-time-capable inference algorithms are useful for extracting the local MW orientation and segmentation of a scene.

Motivated by the observation that on a larger scale the commonly-made MW assumption is often invalid, we have extended the Manhattan Frame model to a mixture of Manhattan Frames which can describe scenes consisting of multiple Manhattan Worlds. The proposed inference algorithm, a manifold-aware Gibbs sampler with Metropolis-Hastings split/merge proposals, allows adaptive and robust inference of MMFs. This enables the proposed model to describe both complex small-scale-indoor and large-scale-urban scenes. We have demonstrated the versatility of our model by extracting MMFs from 1.5k indoor scenes, from a larger dense indoor reconstruction and from an aerial LiDAR point cloud of Cambridge, MA. Code for sampling-based MMF inference and real-time Manhattan Frame rotation estimation can be found at `http://people.csail.mit.edu/jstraub/`.

The joint work with R. Cabezas [35] has demonstrated the use of Manhattan Frame scene priors to regularize 3D reconstructions at city-scale. It would be interesting to

see how Manhattan Frame priors can be incorporated into online 3D reconstruction of indoor scenes where there might be a higher degree of clutter. The experiments showing drift-free Manhattan World rotation tracking using the Manhattan Frame model hint at the potential to obtain drift-free 3D reconstructions if the system can successfully deal with clutter and detect transitions between Manhattan Worlds online. The latter task could be aided by an inertial measurement unit (IMU).

In the current instantiation of the MMF model all spatial information is ignored. As seen in the results, different scene parts belong to different Manhattan Frames and explicitly modeling this spatial smoothness of the Manhattan Frames might lead to more spatially scalable models as well as more precise Manhattan Frame rotation inference.

As discussed in the introduction, the Manhattan Frame and MMF segmentation of a scene should be useful for scene understanding. One way of utilizing inferred Manhattan Frame rotations is to reduce the search space for 3D bounding box proposal algorithms, a common initial step for convolutional neural networks for 3D object recognition in 3D reconstructions. Instead of having to marginalize over rotations in training or having to search over the full rotation space, only the four different directions implied by the Manhattan Frame rotation have to be searched. As demonstrated in the results section the Manhattan Frame inference algorithms proposed herein outperform related scene orientation estimators proposed in related work for scene understanding.

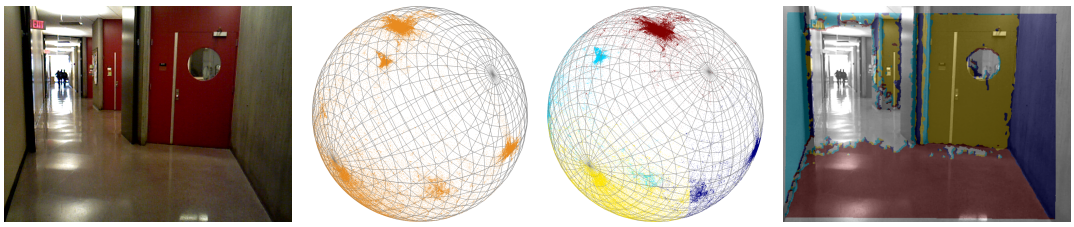
■ 3.7 Acknowledgments

This research published in [221, 225, 226] was conducted in collaboration with Oren Freifeld, Guy Rosman, Jason Chang, John J. Leonard, Nishchal Bhandari, and John W. Fisher III. The idea of the Manhattan Frame was formalized in a theoretically sound way with the help of Oren Freifeld. Guy Rosman helped with edge-preserving smoothing of surface normals, the robust extraction of surface normals from noisy LiDAR data and had the idea of depth camera focal length calibration using the MW assumption [225]. Jason Chang helped develop the Metropolis-Hastings-based inference for the MMF model. Nishchal Bhandari ran the Turtlebot experiment in Killian Court at MIT. Randi Cabezas kindly provided the LiDAR point cloud of Kendall Square.

Chapter 4

Unconstrained Directional Scene Representation

While directional models based on the Manhattan World assumption can capture a wide variety of man-made environments, it is natural to ask if the orthogonality constraints can be relaxed to capture more general and hence a wider class of environments. Clearly restricting a perception systems operation to Manhattan World-type environments would limit its capabilities. Therefore we explore a less expressive (because directions are assumed independent) but more general class of environments in this chapter. In the surface normal space such environments are characterized by a set of pronounced clusters that may have arbitrary angles with respect to eachother as depicted in Fig. 4.1. This assumption on the surface normal space manifests in the 3D structure of a scene via a number of principal planes to which all other planes are parallel. The Manhattan World is a special case with three principal planes that are orthogonal to each other. The Ray and Maria Stata Center (see Fig. 4.2) is an example of a man-made environment where, for the most part, the Manhattan World assumption neither applies locally nor globally: in many areas wall intersection angles are non-orthogonal. There



Scene | Surface Normals | Directional Clustering | Stata Center World Segmentation

Figure 4.1: An example of a Stata Center World scene: the left and the right wall are not parallel. This manifests in non-orthogonal surface normal clusters. The directional clustering of the surface normal distribution implies a directional segmentation of the scene which we term a Stata Center World segmentation.

Figure 4.2: The Stata Center World relaxes the orthogonality constraints of the Manhattan World to describe environments that consist of sets of parallel planes with arbitrary orientation differences as illustrated by different colors in the figure via manual shading (not a result). The model is inspired by the very "relaxed" building style of the Ray and Maria Stata Center at MIT depicted above.

are, however, still parallel walls (in hallways for example). For this reason we term the aforementioned scene prior the Stata Center World (SCW). By clustering the surface normals without orthogonality constraints we obtain a scene segmentation as depicted in Fig. 4.1. Observe that this unconstrained directional segmentations still captures regularities in the environment. The Stata Center World segmentation may therefore be used for further reasoning about the scene in a similar way as the Manhattan World and MMF segmentations from the previous chapter. Again, a perception system may use secondary information about the gravity direction to reason about traversable and obstructed scene parts. By aligning the directional clusters (as shown in Chapter 5) we may leverage unconstrained directional segmentations for scene alignment and thus knowledge transfer between scenes without making an Manhattan World assumption.

Inferring surface normal clusters to obtain the Stata Center World scene segmentation leads to the general problem of directional data modeling, i.e. modeling of data on the unit hyper-sphere in D dimensions, $\mathbb{S}^{D-1}$. In order to be able to model complex distributions on the sphere, this chapter extends Bayesian nonparametric Dirichlet process mixture models (DP-MM) to directional data. The resulting models describe data on the sphere with a potentially infinite number of clusters. This allows the posterior inference to adapt the number of mixture components to the complexity of the distribution of the data. In order to explicitly describe directional data in its natural space, the tangent space Gaussian and the von-Mises-Fisher distributions, introduced in Sec. 2.6, are used as mixture component distributions.

The two main contributions of this chapter are: (1) modeling the distribution of a batch of directional data with anisotropic mixture components and (2) efficient clustering of large batches or streams of batches of directional data with isotropic clusters.

While both contributions impact directional data modeling in general, the former yields more expressive inferred models that can capture correlations in directional data clusters. The latter is more applicable if the dataset size is large or if online operation on a stream of data is desired.

Specifically, the first contribution is a Dirichlet process tangent space Gaussian mixture model (DP-TGMM) that models anisotropic data distributions accurately and flexibly. Published in [224] and explained in Sec. 4.3.2, the DP-TGMM is a Bayesian nonparametric mixture model as introduced in Sec. 2.4 but for directional data. We utilize the tangent space Gaussian distributions (see Sec. 2.6.2) for similar reasons as in the MMF: it can describe anisotropic distributions and has a closed-form conjugate prior distribution. Part of the contribution lies in the adaption of the efficient subcluster split-based MCMC inference algorithm of [42] to exploit the manifold properties of the sphere.

The second contribution, described in Sec. 4.4.3, is to extend the capability of directional clustering algorithms to online time-constrained or large data scenarios. Inference algorithms in fields such as robotics or augmented reality, which would benefit from the use of surface normal statistics, are not generally provided a single batch of data a priori. Instead, they are often provided a stream of data batches from depth cameras. Thus, capturing the surface normal statistics of man-made structures often necessitates the temporal integration of observations from a vast data stream of varying cluster mixtures. Additionally, such applications pose hard constraints on the amount of computational power available, as well as tight timing constraints. We address these challenges by focusing on flexible Bayesian nonparametric (BNP) Dirichlet process mixture models (DP-MM) which describe the distribution of surface normals in their natural space, the unit sphere in 3D, $\mathbb{S}^2$. Taking the small-variance asymptotic limit of this DP-MM of von-Mises-Fisher (vMF) distributions, we obtain a fast k-means-like algorithm, which we call DP-vMF-means, to perform nonparametric clustering of data on the unit hypersphere. Furthermore, we propose a novel dependent DP mixture of vMF distributions to achieve integration of directional data into a temporally consistent streaming model. Small-variance asymptotic analysis yields the k-means-like DDP-vMF-means algorithm. Finally, we propose a method, inspired by optimistic concurrency control, for parallelizing the inherently sequential labeling process of BNP-derived algorithms. This allows real-time processing of batches of 300k data-points at 30 Hz. The use of vMF component distributions is partially motivated by the real-time Manhattan Frame algorithm evaluation in Sec. 3.5.1, which showed that the structure of the vMF posterior distribution lends itself to more efficient inference in comparison to approximate tangent space Gaussian Manhattan Frame algorithm.

After the literature review in the next section, we first formalize the Stata Center World assumption in Sec. 4.2 before we introduce the DP-TGMM model and manifold-aware MCMC inference for it in Sec. 4.3.2. In Sec. 4.4.3 we develop the (D)DP-vMF-means algorithms via low-variance asymptotic analysis.

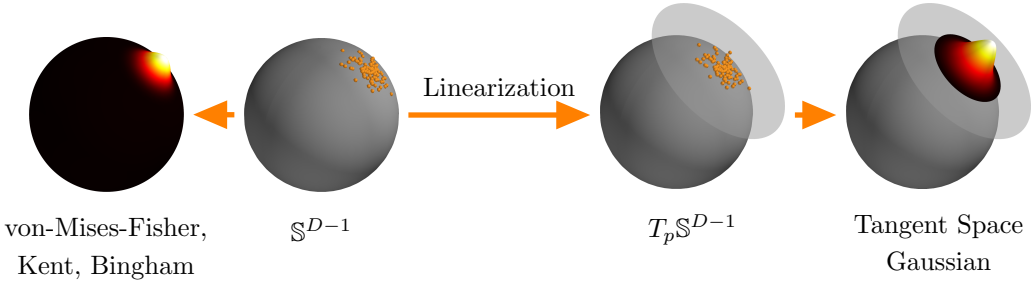


Figure 4.3: Two different approaches to modeling a cluster of directional data. A directional distribution can either be defined directly over the manifold of the sphere (left) or implicitly via a distribution in a linearization of the sphere (right). Here we specifically explore linearization via Riemannian geometry.

Impact beyond surface normal distributions The DP-TGMM as well as the (D)DP-vMF-MM are framed as models to capture the directional segmentation of an environment by describing its surface normal distribution. Surface normal distributions, however, are only one example of directional distributions that the proposed models can describe. There are numerous other naturally directional data sources such as protein backbone configurations in computational biology [192], or 3D rotations expressed as Quaternions (see Sec. 2.7.4). In other instances data is Euclidean, but the distinguishing factor between data-points is the angle between them and not their magnitude. For example, word-frequency vectors are often clustered using the cosine similarity [61], which measures the cosine of the angle formed by two vectors. This measure essentially treats the word-frequency vectors as directional data, and has been shown to be superior to Euclidean distance for document clustering [228]. Another example of directional data is semantic word vectors [165], which associate a high-dimensional vector with each word in a given corpus. The semantic word vectors capture the semantic context of the associated words, and should not to be confused with the word-frequency vectors of documents. Again, cosine similarity is used as the distance measure to find words with similar meaning. The use of angular distance metrics such as the cosine similarity reveals the use of directional information. This motivates normalizing the data to unit length and modeling their distribution on the unit sphere using the proposed models. An example of modeling the distribution of semantic word vectors can be found in our publication [224]. In short, the clustering of unit-length-normalized semantic word vectors captures semantically similar words.

■ 4.1 Related Work

As outlined in the previous introductory section, the models proposed herein fundamentally model directional data, i.e. surface normal distributions. Directional data can be modeled in two principally different ways as depicted in Fig. 4.3: (1) in a linearization

	[101]	[61, 258]	[14]	[68, 217]	[15]	[42]	[*]	[**]	[***]
Spherical geometry	·	✓	✓	✓	✓	·	✓	✓	✓
Bayesian inference	·	·	✓	✓	✓	✓	·	·	✓
Anisotropic cov.	·	·	·	✓	·	✓	·	·	✓
BNP	·	·	·	·	✓	✓	✓	✓	✓
Parallelizeable	✓	✓	✓	✓	·	✓	✓	✓	✓
Realtime capable	✓	✓	·	·	·	·	✓	✓	·
Streaming data	·	·	·	·	·	·	·	✓	·

Table 4.1: Properties of different typically-used clustering algorithms for directional data. Parametric variants are k-means [101], spkm [61, 258] vMF-MM [14], and TGMM [68, 217]. The Bayesian nonparametric (BNP) models and respective inference algorithms contain DP-vMF-MM[15], DP-GMM [42], and the proposed DP-vMF-means [*], DDP-vMF-means [**], and DP-TGMM [***] .

of the sphere and (2) directly on the manifold of the sphere. Sometimes the directional nature of the data is ignored as well (a poor choice as we will find in Sec. 4.3.4). Table 4.1 summarizes and highlights the differences between the proposed and previous approaches which are reviewed in the following.

Directional distributions A variety of distributions [164] has been proposed in the field of directional statistics to model data on the unit sphere. Examples are the antipodal symmetric Bingham distribution [23], the anisotropic Kent distribution [134], and the isotropic von-Mises-Fisher (vMF) distribution [74]. Because of its comparative simplicity, the vMF distribution is most commonly used (see Sec. 2.6.3). Other approaches to modeling data on the unit hypersphere use Riemanian geometry [63] to locally linearize the sphere. In this linearized space standard Euclidean distributions can then be used to model the data distribution. One example is the Tangent space Gaussian model introduced in Sec. 2.6.2.

von-Mises-Fisher mixture models vMF mixture models (vMF-MM) are especially popular for modeling and inference purposes. Banerjee et al. [14] perform Expectation Maximization (EM) for a finite vMF mixture model to cluster text and genomic data. This method is related to the spherical k-means (spkm) algorithm [61], which can be obtained from a finite vMF-MM by taking the infinite limit of the concentration parameter [14]. Zhong [258] extends the spherical k-means algorithm to an online clustering framework by performing stochastic gradient descent on the spkm objective. This approach requires the number of clusters to be known, does not allow the creation or deletion of clusters, and heuristically weights the contribution of old data. Gopal et al. [92] derive variational as well as collapsed Gibbs sampling inference for finite vMF-MMs, for a finite hierarchical vMF-MM and for a finite temporally evolving vMF-MM. The vMF distribution has also been used in BNP and hierarchical MMs. Bangert et al. [15] formulate an infinite vMF-MM using a Dirichlet process (DP) prior, but their sampling-based

inference is known to have convergence issues and to be inefficient [121]. Furthermore, scaling this method to large datasets is problematic because the inference procedure is based on the Chinese Restaurant Process [189], and cannot be parallelized. Reisinger et al. [195] formulate a finite, latent Dirichlet allocation (LDA) model [28] for directional data using the vMF distribution. Since there is no closed form prior for the concentration parameter in the vMF distribution they simply fix it and do not sample from its posterior distribution. This is akin to using a GMM with a fixed variance, which is known to perform poorly if the model variance does not match the noise characteristics. To the best of our knowledge there are no other related k-means-like algorithms for directional batch data besides the spkm algorithm.

Surface normal modeling In the context of object and 3D shape representation, extended Gaussian images (EGI) have been studied [116, 119]. The EGI of a surface is the distribution of surface normals where each surface normal is weighted by the area of the surface it represents. For computations the EGI was approximated via a tessellation of the sphere. A crucial property of EGI is that the representation is invariant to translation. Exploiting this property, Makadia et al. [163] extract maxima in the EGI and use the spherical FFT to compute an initial rotation estimate for point-cloud registration. Note that these maxima correspond to the cluster centers which the proposed (D)DP-vMF-means algorithm extracts. Furukawa et al. [81] use a tessellation of the unit sphere to extract the dominant directions in a scene as part of a depth regularization algorithm based on the Manhattan World (MW) assumption [49]. Tribel et al. [237] employ EM to perform plane segmentation using surface normals in combination with 3D locations. Assuming a finite vMF-MM and using a hierarchical clustering approach, Hasnat et al. [104] model surface normal distributions. The Manhattan Frame model presented in Sec. 3 and published as [221, 225, 226] describes orthogonally coupled clusters of surface normals on the sphere using vMF as well as the tangent space Gaussian model.

Other applications In other applications, the inherent geometry of the problem is often ignored and, without taking the spherical geometry into account, algorithms developed for Euclidean geometry are used; e.g., k-means [101], the finite Gaussian mixture model (GMM) [24], as well as the Dirichlet process GMM [70, 159]. The work in protein-configuration modeling from Ramachandran plots [192] exemplifies this well. First Dahl et al. [53] introduced modeling the angular data as a DP-GMM, ignoring the spherical manifold of the angular data. To solve this issue Lennox et al. [147] model the data on the 3D sphere as a DP-vMF mixture, but require an approximation to the vMF posterior. Work by Ting et al. [234] uses an HDP with normal-inverse-Wishart base measure to share data between proteins, but does not respect the manifold of the data. Approaches utilizing a single tangent space to define distributions over the hyper-sphere have been proposed for rotation estimation and tracking [46, 90]. Finite mixture models of Gaussians in separate tangent spaces have been explored to estimate rigid-body motion in robotics [68] and for human body-pose regression [217]. While

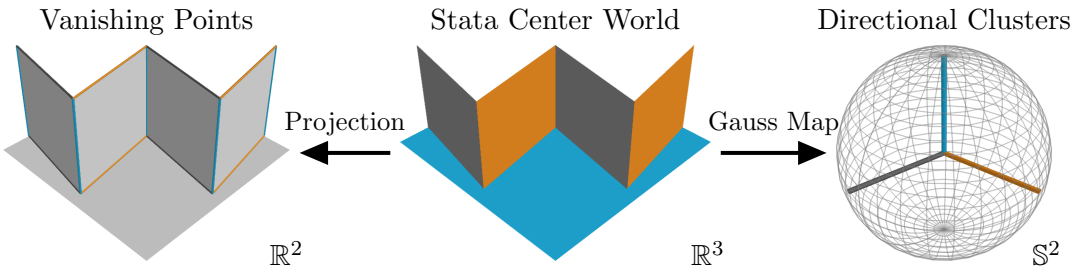


Figure 4.4: An environment following the Stata Center World assumption projects via the Gauss map to directional clusters in the surface normal space. Under the standard camera projection the Stata Center World leads to a collection of vanishing points. While the latter has received a lot of attention in previous work, the former has not.

Simo-Serra et al. [217] show a way to reduce (but not increase) the number of clusters within their EM inference framework, both models are finite mixture models in contrast to our DP-based infinite mixture model. Rotation data, which can be described as a 4D directional data in the form of Quaternions, has previously been described in the tangent space around a current estimate in rotation estimation [46, 90]. Feiten et al. [68] use a fixed and finite mixture of Gaussians in distinct tangent spaces to estimate rigid body motion in robotics. Simo-Serra et al. [217] do the same but allow for elimination of clusters in the EM inference. Archambeau et al. [10] propose a finit Gaussian mixture model for data-defined manifolds.

■ 4.2 The Stata Center World

As described in the introductory section of this chapter, the Stata Center World assumption relaxes the orthogonality constraints imposed by the common Manhattan World model. As such, it captures the idealized notion that an environment is composed of a set of planes such that all planes are parallel to some (smaller) set of principal planes as depicted in Fig. 4.2.

Under the Gauss map, in the ideal noise-free case, the Stata Center World maps to the set of direction vectors of the principal planes as illustrated in Fig. 4.4. Hence we define the Stata Center World model in surface normal space as the set of directional vectors μ_k:

$$\{\mu_k\}_{k=1}^{K} \quad \text{where } \mu_k \in \mathbb{S}^2 \,. \tag{4.1}$$

Generally the number of directional vectors is unknown a priori and should therefore be part of the inference process. This motivates the use of Dirichlet process mixture models, which assume an infinite number of clusters and that the number of clusters grows logarithmically with the amount of observed data. For a given set of data part of the posterior inference becomes determining the number of clusters K to best fit the data distribution (see Sec. 2.4).

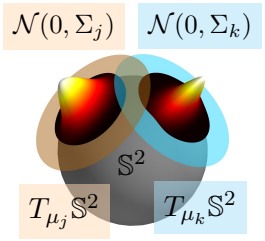


Figure 4.5: An illustration of the of the proposed Dirichlet process tangential Gaussian mixture model (DP-TGMM) for $K = 2$ clusters, k and j, in their respective tangent spaces to the sphere $\mathbb{S}^2$.

In reality, planes in an environment may not be exactly planar and the observation process is subject to noise. This is reflected in the surface normal distributions by directional clusters as can be seen in Fig. 4.1. Therefore, to model real surface normal distribution observations, a directional noise model should be employed. In Sec. 4.3.2 we explore the use of the tangent space Gaussian distribution (see Sec. 2.6.2) which allows modeling anisotropic distributions on the sphere. A simpler noise model based in the isotropic von-Mises-Fisher distribution (see Sec. 2.6.3) is introduced in Sec. 4.4.3.

As a side note, to draw the connection to image-based computer vision, a collection of vanishing points can be observed as the projection of the Stata Center World into a camera image. While one contribution of this chapter is the formalization of the Stata Center World and the exploration of its image under the Gauss map, the vanishing point estimation problem has been explored extensively in prior work as reviewed in Sec. 1.1.1.

■ 4.3 Dirichlet Process Tangential Gaussian Mixture Model

We present a flexible Bayesian nonparametric model for data residing on a hypersphere that respects the inherent geometry of the manifold. As shown in Fig. 4.5, our approach draws on the Dirichlet process Gaussian mixture model (DP-GMM), and models full covariance matrices on (linear) tangent spaces to the sphere, as opposed to the isotropic covariances associated with a von-Mises-Fisher distribution [14, 15, 195, 258]. Importantly, the covariances of the Gaussians, capturing intra-cluster correlations, have analytical conjugate priors that enable efficient inference. Additionally, the approach transparently scales to high-dimensional data. We extend the efficient inference method of Chang et al. [42], a parallelized restricted Gibbs sampler with sub-cluster split/merge moves, to account for the geometry of the sphere. Moreover, we show how to combine sufficient statistics from tangent spaces around *different points of tangency* to propose merges efficiently.

In contrast to previous approaches, the proposed DP-TGMM allows for anisotropic distributions on the sphere, lends itself to consistent Bayesian inference, and adapts the model complexity to the observations. We develop a corresponding inference algorithm

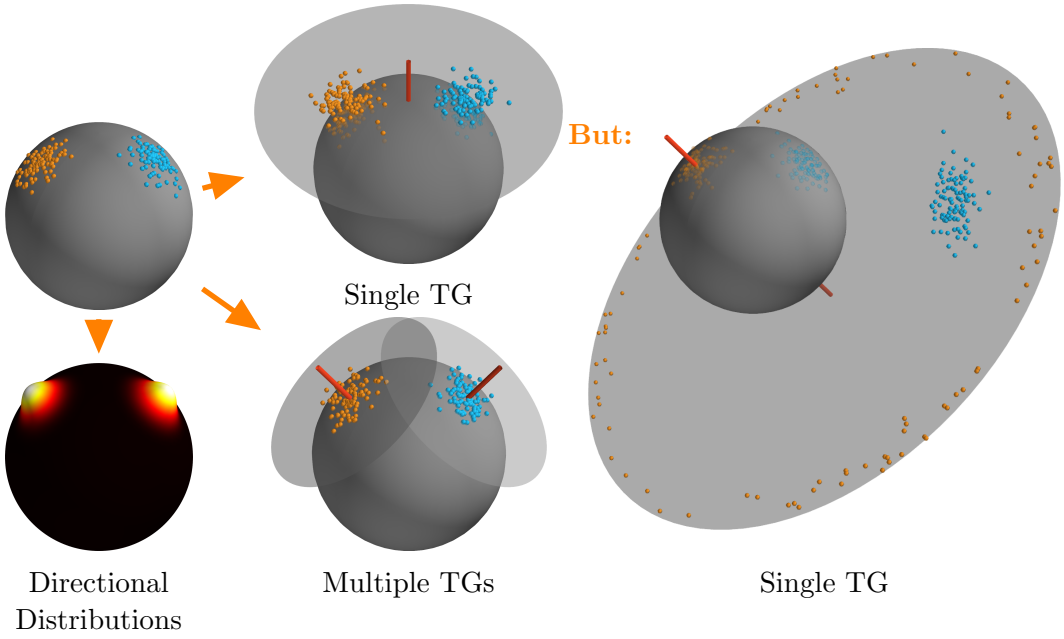


Figure 4.6: Different approaches to modeling clusters of directional data as Bayesian mixture models. One can either use directional distributions that are natively defined over the unit sphere or linearize the sphere around one or more tangent spaces. The problem of choosing just a single linearization point is that it might be poorly chosen (see to the right) and thus lead to very high distortion of the data. Instead we utilize multiple cluster-dependent linearization points.

that can be parallelized and respects the geometry of the unit sphere.

To highlight the differences of the proposed Dirichlet process tangential Gaussian mixture model (DP-TGMM), we quantitatively compare it with four other methods on synthetic directional data with ground-truth labels. Additionally, we demonstrate the scalability and efficiency of the inference algorithm and the applicability of the DP-TGMM to real-world directional data by modeling 3D surface normals under the Stata Center World assumption. Furthermore, we show its scalability to higher dimensions by clustering the 20-dimensional semantic word vectors of 41k words extracted from the English Wikipedia corpus [251].

■ 4.3.1 Bayesian Nonparametric Mixtures of Spherical Data

Classical Statistics rely on the Euclidean structure of $\mathbb{R}^D$. Thus, due to the nonlinearity of the sphere, the statistical analysis of spherical data requires special care [75, 164]. In this section we introduce the Dirichlet process tangential Gaussian mixture model (DP-TGMM), a Bayesian nonparametric mixture model for data lying on the unit sphere, $\mathbb{S}^{D-1}$. Importantly, the model as well as the inference algorithm respects that the sphere

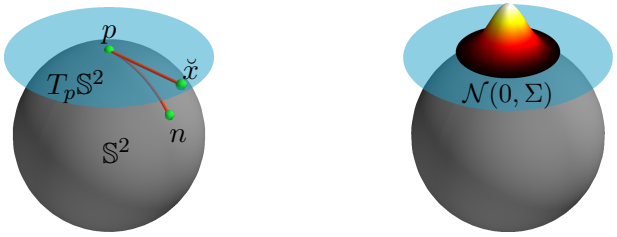


Figure 4.7: Left: The blue plane illustrates $T_p\mathbb{S}^2$, the tangent space to the sphere $\mathbb{S}^2$ at $p \in \mathbb{S}^2$. A tangent vector $\breve{x} \in T_p\mathbb{S}^2$ is mapped to $n \in \mathbb{S}^2$ via Exp_p. Right: We describe the data in each cluster as zero-mean Gaussian in $T_\mu\mathbb{S}^2$.

is a $(D-1)$-dimensional Riemannian manifold. Before introducing the probabilistic model, we now briefly restate the geometric concepts used in the DP-TGMM. See Sec. 2.6.1 for a more detailed discussion of the manifold of the sphere and Sec. 2.4 for a general introduction of Bayesian nonparametric mixture models.

While $\mathbb{S}^{D-1}$ is nonlinear, every point, $p \in \mathbb{S}^{D-1}$, is associated with a linear tangent space, denoted $T_p\mathbb{S}^{D-1}$. A point $\breve{x}$ in the tangent space around p satisfies $p^T\breve{x} = 0$. Elements of $T_p\mathbb{S}^{D-1}$ are called *tangent vectors* and may be viewed as "arrows" based at p and tangent to $\mathbb{S}^{D-1}$. Note that $\dim(T_p\mathbb{S}^{D-1}) = D-1$ and that the point of tangency, p, may be identified with the origin of $T_p\mathbb{S}^{D-1}$. Due to their linearity, tangent spaces often provide a convenient way to model spherical data. In fact, this is also true for more general manifolds [80, 106, 187, 220]. This linearity, together with mappings between $\mathbb{S}^{D-1}$ and $T_p\mathbb{S}^{D-1}$ (see Sec. 2.6.1), enables the modeling and clustering of data points via the tangent space Gaussian distribution, introduced in Sec. 2.6.2, a zero-mean Gaussian distribution in a cluster-dependent tangent space. Note that while a single zero-mean Gaussian in the tangent space around a point, p, provides an effective model for the within-cluster deviations from p provided it is the Karcher mean of this cluster. However, using a Gaussian mixture model whose (non-zero mean) components live on the *same* tangent space is a poor choice as depicted in Fig. 4.6. Thus, in the proposed model, each mixture component exists in its own tangent space, and each tangent space is unique with certainty due to the continuous base measure. We illustrate details for a single cluster in Fig. 4.7, where $\breve{x}$ denotes the point $n \in \mathbb{S}^{D-1}$ mapped to $T_p\mathbb{S}^{D-1}$, and the model for $K = 2$ clusters in Fig. 4.5.

■ 4.3.2 Probabilistic Dirichlet Process Mixture Model for Spherical Data

The Dirichlet process [70] has been extensively used to model data in Euclidean spaces. As introduced in Sec. 2.4, the DP mixture model (DP-MM) uses the DP as a prior to weight a countably-infinite set of clusters, where the distribution of weights is controlled by a concentration parameter, α. Here, we formulate the Dirichlet process tangential Gaussian mixture model (DP-TGMM) which extends DP-MMs to data on the unit sphere, $\mathbb{S}^{D-1}$, in a manner that explicitly respects the intrinsic geometry. The graphical

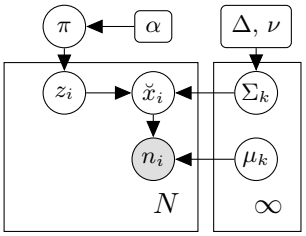


Figure 4.8: The graphical model of the proposed Dirichlet process tangential Gaussian mixture (DP-TGMM).

model is depicted in Fig. 4.8.

The generative DP-MM first samples the infinite-length cluster proportions, π, from a stick-breaking process [216]. Then cluster assignments, $\mathbf{z} = \{z_i\}_{i=1}^N$, are sampled from the categorical distribution defined by π:

$$\pi \sim \text{GEM}(1, \alpha) \tag{4.2}$$
$$z_i \sim \text{Cat}(\pi_1, \pi_2, \dots)\,. \tag{4.3}$$

Associated with each cluster, $k \in \{1, \dots, \infty\}$, is a mean location on the sphere, μ_k, and a covariance, Σ_k, in the corresponding tangent space, $T_{\mu_k}\mathbb{S}^{D-1}$. These parameters are drawn from the following priors:

$$\mu_k \sim \text{Unif}(\mathbb{S}^{D-1})\,, \tag{4.4}$$
$$\Sigma_k \sim \text{IW}(\Delta, \nu)\,, \tag{4.5}$$

where Unif and IW are the uniform and inverse-Wishart distributions, respectively. Note that $\mathbb{S}^{D-1}$ has a finite surface area of $2\pi^{\frac{D}{2}}/\Gamma(\frac{D}{2})$ which is used as the normalizing constant of Unif. This will later play a role in the inference procedure.

Observations, n_i, are drawn from their associated TG distributions with covariance Σ_{z_i} centered at μ_{z_i} followed by mapping them to $\mathbb{S}^{D-1}$ via the exponential map of Eq. (2.61). This can be understood as a two step process:

$$\breve{x}_i \sim \mathcal{N}(0, \Sigma_{z_i}) \tag{4.6}$$
$$n_i \sim \text{Exp}_{\mu_{z_i}}(\breve{x}_i) \qquad \forall i \in \{1, \dots, N\}\,. \tag{4.7}$$

We can also concatenate the two steps to reveal the tangent space Gaussian distribution:

$$n_i \sim \text{Exp}_{\mu_{z_i}}(\mathcal{N}(0, \Sigma_{z_i})) \qquad \forall i \in \{1, \dots, N\}\,. \tag{4.8}$$

The Gaussian on $T_{\mu_{z_i}}\mathbb{S}^{D-1}$ induces a probability measure on $\mathbb{S}^{D-1}$. Refer to Sec. 2.6.2 for an in depth characterization and discussion of the TG distribution. More detailed information about the aforementioned basic distributions (Cat and IW) as well as how to sample from them can be found in Sec. 2.3. We now describe efficient MCMC inference for aforementioned geometry-respecting model.

■ 4.3.3 Manifold-Aware MCMC Inference

Markov chain Monte Carlo (MCMC) techniques [198] provide a computational mechanism for sampling from complex Bayesian models. See Sec. 2.2.1 for a general introduction. Unfortunately, in DP mixture models, MCMC methods are often slow. When parameters are marginalized, inference scales poorly because algorithms cannot be parallelized. When parameters are instantiated, the algorithm is parallelizable, but typically requires approximations and exhibits slow convergence. The recent DP sub-cluster algorithm of [42] addresses these issues by combining Metropolis-Hastings (MH) split/merge moves with a restricted Gibbs sampler, which is not allowed to add or remove clusters. The resulting Markov chain is guaranteed to converge to the desired posterior distribution. Additionally, this approach allows parallelization and the support of non-conjugate priors.

The DP sub-cluster algorithm proposes splits effectively via the MH framework [105] by exploiting an inferred auxiliary two-component, "sub-cluster" model for each regular cluster. The sub-clusters are inferred within the restricted Gibbs sampler. Excluding the varying complexity of posterior parameter sampling ($O(KD^2)$ for a GMM), the computational complexity per MCMC iteration is $O(NK + K^2)$, K is the maximum number of non-empty clusters. While the algorithm from [42] was originally suggested for DP models in $\mathbb{R}^D$, we show here that it can be extended to the DP-TGMM. This extension requires: (1) respecting the geometry of $\mathbb{S}^{D-1}$ when computing posterior distributions; and (2) combining sufficient statistics from different tangent spaces to propose splits efficiently. Additionally, we propose a MH algorithm to sample from the true posterior of the mean location on $\mathbb{S}^{D-1}$.

Restricted Gibbs Sampling

We now discuss restricted Gibbs sampling of the labels $\mathbf{z} \triangleq \{z_i\}_{i=1}^N$, means $\boldsymbol{\mu} \triangleq \{\mu_k\}_{k=1}^K$ and covariances $\boldsymbol{\Sigma} \triangleq \{\Sigma_k\}_{k=1}^K$ for K clusters. We note that each Σ_k is defined over a separate tangent space, $T_{\mu_k}\mathbb{S}^{D-1}$.

The covariances for each cluster are first sampled. Conditioned on the mean, μ_k, the data, $\mathbf{n} \triangleq \{n_i\}_{i=1}^N$, are modeled via a zero-mean Gaussian distribution in the tangent plane, $T_{\mu_k}\mathbb{S}^{D-1}$, as defined in Eq. (4.8). Hence, the same analysis as in the Euclidean space applies, and we sample $\boldsymbol{\Sigma}$ from the IW posterior [85]:

$$\Sigma_k \sim p(\Sigma_k \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_k) = \mathrm{IW}(\Delta + S_k, \nu + N_k) \tag{4.9}$$

where $\mathcal{I}_k \triangleq \{i : z_i = k\}$ is the set of indices with label k, $N_k \triangleq |\mathcal{I}_k|$ counts the points assigned to cluster k, and S_{μ_k} is the scatter matrix at $T_{\mu_k}\mathbb{S}^{D-1}$, defined as:

$$S_{\mu_k} \triangleq \sum_{i \in \mathcal{I}_k} \mathrm{Log}_{\mu_k}(n_i)\mathrm{Log}_{\mu_k}(n_i)^T . \tag{4.10}$$

Since $T_{\mu_k}\mathbb{S}^{D-1}$ is a $(D-1)$ dimensional space, Σ_k and $S_{\mu_k} \in \mathbb{R}^{(D-1)\times(D-1)}$. Computationally, the Log_{μ_k} map returns D-dimensional vectors that are orthogonal to μ_k. To

obtain a $(D-1)$ dimensional representation, after applying the log map, we transport all data to the north pole $m = (0, \ldots, 0, 1)$ using a rotation ${}^{m}R_{\mu_k}$ and drop the last dimension of the resulting vector (which is going to be 1). This is discussed in more detail in Sec. 2.6.1 for arbitrary dimension D.

Note, however, that the geometry of $\mathbb{S}^{D-1}$ and the aforementioned process renders the frequently-required computation of S_{μ_k} inefficient. The bottleneck of the calculation is attributed to the computationally-intensive evaluation of $\{\text{Log}_{\mu_k}(n_i)\}_{i\in\mathcal{I}_k}$ that depends on the point of tangency, μ_k, which constantly changes during sampling-based inference.

To circumvent this issue, we exploit the logarithm map approximation outlined in Sec. 2.6.2 namely that

$$\text{Log}_{\mu_k}(n_i) \approx \text{Log}_{\mu_k}(\tilde{n}_k) + {}^{\mu_k}R_{\tilde{n}_k}\text{Log}_{\tilde{n}_k}(n_i)\,, \tag{4.11}$$

where $\tilde{n}_k$ is the Karcher mean of $\mathbf{n}_{\mathcal{I}_k}$. Under this approximation, starting from Eq. (4.10), and dropping indices k for the sake of clarity, the scatter matrix $S_{\mu_k} = S_\mu$ can be approximated as

$$\begin{aligned}
S_\mu &\approx \sum_{i\in\mathcal{I}} \left[\text{Log}_\mu(\tilde{n}) + {}^{\mu}R_{\tilde{n}}\text{Log}_{\tilde{n}}(n_i)\right]\left[\text{Log}_\mu(\tilde{n}) + {}^{\mu}R_{\tilde{n}}\text{Log}_{\tilde{n}}(n_i)\right]^T && (4.12)\\
&= \sum_{i\in\mathcal{I}} \text{Log}_\mu(\tilde{n})\text{Log}_\mu(\tilde{n})^T + 2\,{}^{\mu}R_{\tilde{n}}\text{Log}_{\tilde{n}}(n_i)\text{Log}_\mu(\tilde{n})^T \\
&\quad + {}^{\mu}R_{\tilde{n}}\text{Log}_{\tilde{n}}(n_i)\text{Log}_{\tilde{n}}(n_i)^T{}^{\mu}R_{\tilde{n}}^T && (4.13)\\
&= N\,\text{Log}_\mu(\tilde{n})\text{Log}_\mu(\tilde{n})^T + 2{}^{\mu}R_{\tilde{n}}\left[\sum_{i\in\mathcal{I}}\text{Log}_{\tilde{n}}(n_i)\right]\text{Log}_\mu(\tilde{n})^T \\
&\quad + {}^{\mu}R_{\tilde{n}}\left[\sum_{i\in\mathcal{I}}\text{Log}_{\tilde{n}}(n_i)\text{Log}_{\tilde{n}}(n_i)^T\right]{}^{\mu}R_{\tilde{n}}^T && (4.14)\\
&= N\,\text{Log}_\mu(\tilde{n})\text{Log}_\mu(\tilde{n})^T + {}^{\mu}R_{\tilde{n}}\left[\sum_{i\in\mathcal{I}}\text{Log}_{\tilde{n}}(n_i)\text{Log}_{\tilde{n}}(n_i)^T\right]{}^{\mu}R_{\tilde{n}}^T && (4.15)\\
&= N\,\text{Log}_\mu(\tilde{n})\text{Log}_\mu(\tilde{n})^T + {}^{\mu}R_{\tilde{n}}S_{\tilde{n}}{}^{\mu}R_{\tilde{n}}^T\,. && (4.16)
\end{aligned}$$

where $S_{\tilde{n}_k}$ is the scatter matrix computed in the tangent plane of $\tilde{n}_k$. From Eq. (4.14) to Eq. (4.15) we have used the definition of the Karcher mean that the sample mean in its tangent space is 0: $\sum_{i\in\mathcal{I}_k}\text{Log}_{\tilde{n}_k}(n_i) = 0$. This approximation has the desired advantage that unless the set, $\mathcal{I}_k$, of data points associated with cluster k changes, $\tilde{n}_k$ and $S_{\tilde{n}_k}$ do not have to be recomputed. If the mean μ_k changes the scatter matrix S_{μ_k} can be updated efficiently without having to iterate through all associated data points: the computation of $N_k\text{Log}_{\mu_k}(\tilde{n}_k)\text{Log}_{\mu_k}(\tilde{n}_k)^T$ involves just one outer product and neither N_k nor $\tilde{n}_k$ changes. The rotation of the unchanged S_{μ_k} by the modified ${}^{\mu_k}R_{\tilde{n}_k}$ is also at most $O(D^3)$, scaling with dimension rather than the number of data points as desired. We will also use this approximation for proposing merges.

Conditioned on the sampled covariance matrix, Σ_k, we then sample μ_k. Ideally, we would sample directly from the following posterior distribution of μ_k:

$$p(\mu_k \mid \mathbf{n}, \mathbf{z}, \Sigma_k) \propto p(\mu_k) p(\mathbf{n} \mid \mu_k, \mathbf{z}, \Sigma_k) = p(\mu_k) \prod_{i \in \mathcal{I}_k} \mathcal{N}\left(\mathrm{Log}_{\mu_k}(n_i); 0, \Sigma_k\right) . \tag{4.17}$$

Unfortunately, due to the nonlinearity of $\mathbb{S}^{D-1}$, this distribution cannot be expressed in a closed form. Instead, we utilize the Metropolis-Hastings framework to sample μ_k. It is well known in the literature that the closer the proposal distribution is to the target posterior distribution, the faster the convergence. We therefore use the following proposal as an approximation to Eq. (4.17):

$$q(\mu_k \mid \mathbf{n}, \mathbf{z}, \Sigma_k) = p(\mu_k) \mathcal{N}\left(\mathrm{Log}_{\tilde{n}_k}(\mu_k); 0, \tfrac{\Sigma_k}{N_k}\right) \tag{4.18}$$

The approximation lies in the assumption, that the data $\mathbf{n}_{\mathcal{I}_k}$ have a small spread. This implies that the distance $\theta_i = d_G(n_i, \mu_k)$ from a datapoint n_i to the mean μ_k is approximately the same as the distance $\bar{\theta} = d_G(\tilde{n}_k, \mu_k)$ of the Karcher mean to the mean. This can be seen when looking more closely at the product of Gaussians when expanding the log map (Eq. (2.60)):

$$\begin{aligned}
&\prod_{i \in \mathcal{I}_k} \mathcal{N}\left(\mathrm{Log}_{\mu_k}(n_i); 0, \Sigma_k\right) \\
&\propto \exp\left\{-\tfrac{1}{2} \textstyle\sum_{i=1}^{N} \left[\frac{\theta_i^2}{\sin^2\theta_i} n_i^T \Sigma^{-1} n_i - 2\frac{\theta_i^2 \cos\theta_i}{\sin^2\theta_i} n_i^T \Sigma^{-1} \mu + \frac{\theta_i^2 \cos^2\theta_i}{\sin^2\theta_i} \mu^T \Sigma^{-1} \mu\right]\right\} \\
&\approx \exp\left\{-\tfrac{1}{2} N \left[\frac{\bar{\theta}^2}{\sin^2\bar{\theta}} \tilde{n}^T \Sigma^{-1} \tilde{n} - 2\frac{\bar{\theta}^2 \cos\bar{\theta}}{\sin^2\bar{\theta}} \tilde{n}^T \Sigma^{-1} \mu + \frac{\bar{\theta}^2 \cos^2\bar{\theta}}{\sin^2\bar{\theta}} \mu^T \Sigma^{-1} \mu\right]\right\} \\
&\propto \mathcal{N}\left(\mathrm{Log}_{\mu_k}(\tilde{n}); 0, \tfrac{\Sigma_k}{N_k}\right) \approx \mathcal{N}\left(\mathrm{Log}_{\tilde{n}}(\mu_k); 0, \tfrac{\Sigma_k}{N_k}\right)
\end{aligned} \tag{4.19}$$

The approximation in the last line stems from the fact, that both Gaussian densities have the same covariances that shrink with N_k. That means the distribution $\mathcal{N}\left(\mathrm{Log}_{\mu_k}(\tilde{n}); 0, \frac{\Sigma_k}{N_k}\right)$ is (very) concentrated about $\mathrm{Log}_{\mu_k}(\tilde{n})$ allowing us to flip μ_k and $\tilde{n}$ while approximately describing the same distribution for μ_k close to $\tilde{n}$.

The proposed mean $\hat{\mu}_k$ is accepted according to the Metropolis-Hastings algorithm, with probability $\Pr(\text{accept}) = \min(1, r)$, where the Hastings ratio, r, is

$$r = \frac{p(\mathbf{n}|\mathbf{z}, \hat{\mu}_k, \Sigma_k) p(\hat{\mu}_k) q(\mu_k | \mathbf{n}, \mathbf{z}, \Sigma_k)}{p(\mathbf{n}|\mathbf{z}, \mu_k, \Sigma_k) p(\mu_k) q(\hat{\mu}_k | \mathbf{n}, \mathbf{z}, \Sigma_k)} = \frac{\mathcal{N}(\mathrm{Log}_{\tilde{n}_k}(\mu_k); 0, \Sigma_k/N_k)}{\mathcal{N}(\mathrm{Log}_{\tilde{n}_k}(\hat{\mu}_k); 0, \Sigma_k/N_k)} \prod_{i \in \mathcal{I}_k} \frac{\mathcal{N}(\mathrm{Log}_{\hat{\mu}_k}(n_i); 0, \Sigma_k)}{\mathcal{N}(\mathrm{Log}_{\mu_k}(n_i); 0, \Sigma_k)} . \tag{4.20}$$

We have used the modeling assumption that the distribution of means, $p(\mu_k)$, is uniform over the sphere. Without prior information this is generally the most reasonable assumption. In some problems one might want to have a more informative prior based on other additional information. Since we are using the Metropolis-Hastings sampling algorithm, it is straightforward to extend the herein described method to other prior distributions (like for example a tangent space Gaussian distribution).

Finally, given means, $\boldsymbol{\mu}$, and covariances, $\boldsymbol{\Sigma}$, we sample new labels, $\mathbf{z}$, for all data, $\mathbf{n}$, as

$$z_i \overset{\propto}{\sim} \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathrm{Log}_{\mu_k}(n_i); 0, \Sigma_k) \mathbb{1}_{z_i=k} \,, \tag{4.21}$$

where $\overset{\propto}{\sim}$ denotes sampling from the distribution proportional to the right side, and the indicator function $\mathbb{1}_{z_i=k}$ is 1 if $z_i = k$ and 0 otherwise. In practice, one computes the K different probability density function values, normalizes these values to sum to 1, and samples from the resulting Categorical distribution as described in Sec. 2.3.2.

Sub-Cluster Split/Merge Proposals

We now describe the Metropolis-Hastings split-and-merge proposals that are specialized to the geometry of $\mathbb{S}^{D-1}$. For a general introduction to Metropolis-Hastings see Sec. 2.2.1. The previously-defined posterior distributions for directional data uniquely define posterior inference in the sub-clusters [42]. When constructing split-and-merge moves, joint proposals over the entire latent space, $\{\mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma}\}$, must be constructed. The proposed labels, $\hat{\mathbf{z}}$, will be constructed from the inferred sub-clusters. Ideally, the parameters, $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, will be proposed from the true posteriors. However, as discussed previously, no conjugate prior exists for $\boldsymbol{\mu}$. Hence we propose the parameters from

$$\hat{\mu}_a \sim q(\mu_a \mid \mathbf{n}, \mathbf{z}) = \mathcal{N}(\mathrm{Log}_{\tilde{n}_a}(\mu_a); 0, \Sigma_a^\star) \,, \tag{4.22}$$

$$\Sigma_a \sim p(\Sigma_a \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_a) \,, \tag{4.23}$$

where $\Sigma_a^\star \triangleq \arg\max_\Sigma \mathrm{IW}(\Delta + S_{\tilde{n}_a}, \nu + N_a)$. The scatter matrix, $S_{\tilde{n}_a}$, in $T_{\tilde{n}_a}\mathbb{S}^{D-1}$ is computed according to Eq. (4.47), and $p(\Sigma_a \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_a)$ is the true posterior denoted in Eq. (4.9). We note that Eq. (4.22) uses the covariance $\Sigma_a^\star$ instead of Σ_a because Σ_a depends on $\hat{\mu}_a$ through the point of tangency.

The Dirichlet process Sub-Cluster algorithm then deterministically constructs moves from the sub-clusters. We extend the algorithm to the DP-TGMM by splitting cluster a into clusters b and c with:

$$\begin{aligned} \hat{\mathbf{z}}_{\mathcal{I}_a} &= \text{split-}a(\mathbf{z}) \,, \\ (\hat{\mu}_b, \hat{\mu}_c) &= (\hat{\mu}_{a\ell}, \hat{\mu}_{ar}) \,, \\ (\hat{\Sigma}_b, \hat{\Sigma}_c) &\sim p(\hat{\Sigma}_b \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_b) p(\hat{\Sigma}_c \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_c) \,, \end{aligned} \tag{4.24}$$

and by merging clusters b and c into cluster a with:

$$\begin{aligned} \hat{\mathbf{z}}_{\mathcal{I}_b \cup \mathcal{I}_c} &= \text{merge-}bc(\mathbf{z}) \,, \\ \hat{\mu}_a &\sim q(\mu_a \mid \mathbf{n}, \mathbf{z}) \,, \\ \hat{\Sigma}_a &\sim p(\hat{\Sigma}_a \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_a) \,, \end{aligned} \tag{4.25}$$

where split-$a(\mathbf{z})$ splits cluster a into clusters b and c deterministically based on the sub-cluster labels, and merge-$bc(\mathbf{z})$ merges clusters b and c into cluster a.

Next we derive the Hastings ratio for a deterministic split proposal based on the sub-clusters. In general the Hastings ratio for the DP-TGMM model is:

$$r = \frac{p(\mathbf{x}, \hat{\mathbf{z}}, \hat{\boldsymbol{\mu}}, \hat{\boldsymbol{\Sigma}})}{p(\mathbf{x}, \mathbf{z}, \boldsymbol{\mu}, \boldsymbol{\Sigma})} \frac{q(\mathbf{z}, \boldsymbol{\Sigma}, \boldsymbol{\mu})}{q(\hat{\mathbf{z}}, \hat{\boldsymbol{\Sigma}}, \hat{\boldsymbol{\mu}})} \,. \tag{4.26}$$

Since we can propose covariances $\hat{\boldsymbol{\Sigma}}$ from the posterior given the means $\hat{\boldsymbol{\mu}}$, we factor the Hastings ratio as follows:

$$r_{\text{split}} = \frac{p(\hat{\mathbf{z}})p(\hat{\boldsymbol{\mu}})p(\mathbf{x} \mid \hat{\mathbf{z}}, \hat{\boldsymbol{\mu}})p(\hat{\boldsymbol{\Sigma}} \mid \mathbf{x}, \hat{\mathbf{z}}, \hat{\boldsymbol{\mu}})}{p(\mathbf{z})p(\boldsymbol{\mu})p(\mathbf{x} \mid \mathbf{z}, \boldsymbol{\mu})p(\boldsymbol{\Sigma} \mid \mathbf{x}, \mathbf{z}, \boldsymbol{\mu})} \frac{q(\mathbf{z})q(\boldsymbol{\mu} \mid \mathbf{x}, \mathbf{z})p(\boldsymbol{\Sigma} \mid \mathbf{x}, \mathbf{z}, \boldsymbol{\mu})}{q(\hat{\mathbf{z}})q(\hat{\boldsymbol{\mu}} \mid \mathbf{x}, \hat{\mathbf{z}})p(\hat{\boldsymbol{\Sigma}} \mid \mathbf{x}, \hat{\mathbf{z}}, \hat{\boldsymbol{\mu}})} \,. \tag{4.27}$$

We can further expand and simplify this to

$$r_{\text{split}} = \frac{p(\hat{\mathbf{z}})p(\hat{\boldsymbol{\mu}})p(\mathbf{x} \mid \hat{\mathbf{z}}, \hat{\boldsymbol{\mu}})}{p(\mathbf{z})p(\boldsymbol{\mu})p(\mathbf{x} \mid \mathbf{z}, \boldsymbol{\mu})} \frac{q(\mathbf{z})q(\boldsymbol{\mu} \mid \mathbf{x}, \mathbf{z})}{q(\hat{\mathbf{z}})q(\hat{\boldsymbol{\mu}} \mid \mathbf{x}, \hat{\mathbf{z}})} \tag{4.28}$$

$$= \frac{p(\hat{\mathbf{z}})q(\mathbf{z})}{p(\mathbf{z})q(\hat{\mathbf{z}})} \frac{p(\hat{\mu}_b)p(\hat{\mu}_c)p(\mathbf{x} \mid \hat{\mathbf{z}}, \hat{\mu}_b)p(\mathbf{x} \mid \hat{\mathbf{z}}, \hat{\mu}_c)q(\mu_a \mid \mathbf{x}, \mathbf{z})}{p(\mu_a)p(\mathbf{x} \mid \mathbf{z}, \mu_a)q(\hat{\mu}_b \mid \mathbf{x}, \hat{\mathbf{z}})q(\hat{\mu}_c \mid \mathbf{x}, \hat{\mathbf{z}})} \,. \tag{4.29}$$

Since the labels $\hat{\mathbf{z}}$ and the parameters $\hat{\boldsymbol{\mu}}_{b,c}$ are proposed analogous to [42] and because the prior distribution on the means $\boldsymbol{\mu}$ are uniform we have

$$r_{\text{split}} = \frac{\alpha\Gamma(\hat{N}_b)\Gamma(\hat{N}_c)}{\Gamma(N_a)} \frac{p(\hat{\mu})p(\mathbf{x} \mid \hat{\mathbf{z}}, \hat{\mu}_b)p(\mathbf{x} \mid \hat{\mathbf{z}}, \hat{\mu}_c)}{p(\mathbf{x} \mid \mathbf{z}, \mu_a)} q(\mu_a \mid \mathbf{x}, \mathbf{z}) \,. \tag{4.30}$$

As shown in [42], any proposed deterministic merge move will be rejected with very high probability. As such, the DP Sub-Cluster incorporates a set of randomized split/merge proposals that are generated from a data-independent, two-dimensional Dirichlet distribution. We use this formulation as well, and introduce the following randomized split/merge proposals. The random splits are proposed as:

$$\begin{aligned} \hat{\mathbf{z}}_{\mathcal{I}_a} &\sim \text{DirMult}(\alpha/2, \alpha/2) \,, \\ (\hat{\mu}_b, \hat{\mu}_c) &\sim q(\hat{\mu}_b \mid \mathbf{n}, \hat{\mathbf{z}})q(\hat{\mu}_c \mid \mathbf{n}, \hat{\mathbf{z}}) \,, \\ (\hat{\Sigma}_b, \hat{\Sigma}_c) &\sim p(\hat{\Sigma}_b \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_b)p(\hat{\Sigma}_c \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_c) \,, \end{aligned} \tag{4.31}$$

and the random merges according to:

$$\begin{aligned} \hat{\mathbf{z}}_{\mathcal{I}_b \cup \mathcal{I}_c} &= \text{merge-}bc(\mathbf{z}) \,, \\ \mu_a &\sim q(\hat{\mu}_a \mid \mathbf{n}, \mathbf{z}) \,, \\ \Sigma_a &\sim p(\hat{\Sigma}_a \mid \mathbf{n}, \mathbf{z}, \hat{\mu}_a) \,. \end{aligned} \tag{4.32}$$

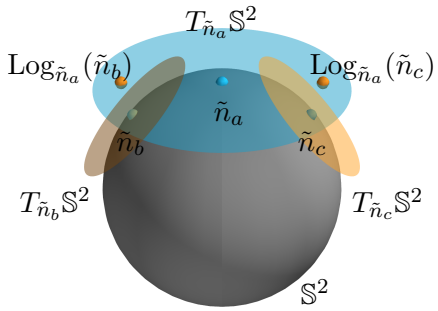


Figure 4.9: Illustration of the problem of computing sufficient statistics for cluster a from clusters b and c. Depicted are the tangent plane around the Karcher mean $\tilde{n}_a$ of cluster a in blue and the two tangent planes by clusters b and c to the left and right in orange.

The Metropolis-Hastings ratio for the random split/merge proposals can be derived starting from Eq. (4.30). Using derivations from [42] we arrive at

$$r^{\text{rand}}_{\text{split}} = \frac{\alpha\Gamma(\alpha/2)^2\Gamma(\alpha+N_a)\Gamma(\hat{N}_b)\Gamma(\hat{N}_c)}{\Gamma(\alpha)\Gamma(N_a)\Gamma(\alpha/2+\hat{N}_b)\Gamma(\alpha/2+\hat{N}_c)} \frac{p(\mathbf{x}|\hat{\mathbf{z}},\hat{\mu})p(\hat{\mu}_b)}{p(\mathbf{x}|\mathbf{z},\mu)} \frac{q(\mu_a|\mathbf{n},\mathbf{z})}{q(\hat{\mu}_b|\mathbf{n},\hat{\mathbf{z}})q(\hat{\mu}_c|\mathbf{n},\hat{\mathbf{z}})} \tag{4.33}$$

$$r^{\text{rand}}_{\text{merge}} = \frac{\Gamma(\alpha)\Gamma(\hat{N}_a)\Gamma(\alpha/2+N_b)\Gamma(\alpha/2+N_c)}{\alpha\Gamma(\alpha/2)^2\Gamma(\alpha+\hat{N}_a)\Gamma(N_b)\Gamma(N_c)} \frac{p(\mathbf{x}|\mathbf{z},\hat{\mu})}{p(\mathbf{x}|\hat{\mathbf{z}},\mu)p(\mu_b)} \frac{q(\mu_b|\mathbf{n},\mathbf{z})q(\mu_c|\mathbf{n},\mathbf{z})}{q(\hat{\mu}_a|\mathbf{n},\hat{\mathbf{z}})} . \tag{4.34}$$

Note that while $\tilde{n}_b$, $\tilde{n}_c$, S_{μ_b}, and S_{μ_c} must be recomputed for the random split proposal, we efficiently approximate these quantities for the deterministic split and the random merge from the statistics of clusters b and c as described in the next section.

Merging Sufficient Statistics between Tangent Spaces

When we propose merges, and to compute the sufficient statistics of the "upper" cluster consisting of the left and right sub-clusters, we use the following approach to efficiently compute the needed sufficient statistics solely from the already computed sufficient statistics. While we describe the process in the context of merging two clusters b and c into cluster a, the approach is the same for computing the sufficient statistics of the "upper" cluster from left and right sub-cluster.

Assume we want to merge cluster b with cluster c to obtain the merged cluster a as depicted in Fig. 4.9. We need to obtain its Karcher mean $\tilde{n}_a \in \mathbb{S}^{D-1}$ as well as the sufficient statistics

$$N_a = N_b + N_c \tag{4.35}$$

$$\bar{\breve{x}}_a = \frac{1}{N_a} \sum_{i:z_i=a} \mathrm{Log}_{\mu_a}(n_i) \tag{4.36}$$

$$S_a = \sum_{i:z_i=a} \mathrm{Log}_{\mu_a}(n_i)\mathrm{Log}_{\mu_a}(n_i)^T \tag{4.37}$$

Clearly, we could just compute the Karcher mean and the sufficient statistics from scratch each time we propose a merge. Instead, in order to save computations and make the inference more efficient we reuse the already computed statistics and Karcher means for clusters b and c.

Merged Karcher mean The Karcher mean $\tilde{n}_a$ can be computed form the Karcher means $\tilde{n}_b$ and $\tilde{n}_c$ without having to run the Karcher mean algorithm on the joint set of data points as follows. Let cluster b contain N_b and cluster c N_c data points. We approximate the Karcher mean of the merged cluster $\tilde{n}_a$ as the weighted Karcher mean, of $\tilde{n}_b$ and $\tilde{n}_c$ with weights N_b and N_c respectively. Using Eq. (2.67) the optimization problem that yielding $\tilde{n}_a$ becomes:

$$\tilde{n}_a = \arg\min_{p \in \mathbb{S}^{D-1}} N_b \arccos(p^T \tilde{n}_b)^2 + N_c \arccos(p^T \tilde{n}_c)^2. \tag{4.38}$$

Since the geodesic between any two points is the shortest path on the manifold between the two of them, we know that p has to lie on the geodesic. On the unit sphere we can describe the location on the geodesic as a rotation about the axis defined by the cross product of the two vectors $\tilde{n}_b$ and $\tilde{n}_c$ by an angle θ_b, which we define such that the location of $\tilde{n}_b$ on the geodesic has $\theta_b = 0$. This implies that the location of $\tilde{n}_c$ on the geodesic has angle $\theta_c = \arccos(\tilde{n}_b^T \tilde{n}_c)$. With this intuition we can reformulate the optimization problem in terms of angles on the geodesic as:

$$\theta^\star = \arg\min_{\theta} N_b \theta^2 + N_c(\theta - \theta_c)^2 \,. \tag{4.39}$$

The minimizer of this function is $\theta^\star = \frac{N_b}{N_c + N_b}\theta_c$. Hence we can compute $\tilde{n}_a$ by rotating $\tilde{n}_b$ by an angle of $\theta^\star$ about the aforementioned axis.

Merged sufficient statistics The sufficient statistics for clusters b and c are computed in the tangent spaces around their respective Karcher means. Therefore their sample means $\bar{\tilde{x}}_{b,c}$ in the tangent space will be very close to zero. However, the sample mean in $T_{\tilde{n}_a}\mathbb{S}^{D-1}$ is generally non-zero and we compute it as the weighted mean between the sample means of cluster b and c mapped into $T_{\tilde{n}_a}\mathbb{S}^{D-1}$:

$$\bar{\tilde{x}}_a = \frac{1}{N_a}\left(N_b \mathrm{Log}_{\tilde{n}_a}(\mathrm{Exp}_{\tilde{n}_b}(\bar{\tilde{x}}_b)) + N_c \mathrm{Log}_{\tilde{n}_a}(\mathrm{Exp}_{\tilde{n}_c}(\bar{\tilde{x}}_c))\right) \tag{4.40}$$

Similarly, we can map the scatter matrices $S_{b,c}$ into $T_{\tilde{n}_a}\mathbb{S}^{D-1}$ to obtain $\tilde{S}_{b,c}$ with the

following approximation:

$$\tilde{S}_b = \sum_{i \in \mathcal{I}_b} \mathrm{Log}_{\tilde{n}_a}(n_i) \mathrm{Log}_{\tilde{n}_a}(n_i)^T \tag{4.41}$$

$$\approx \sum_{i \in \mathcal{I}_b} \left(\mathrm{Log}_{\tilde{n}_a}(\mu_b) + {}^{\tilde{n}_a}R_{\tilde{n}_b} \mathrm{Log}_{\tilde{n}_b}(n_i) \right) \left(\mathrm{Log}_{\tilde{n}_a}(\mu_b) + {}^{\tilde{n}_a}R_{\tilde{n}_b} \mathrm{Log}_{\tilde{n}_b}(n_i) \right)^T \tag{4.42}$$

$$\begin{aligned} =& {}^{\tilde{n}_a}R_{\tilde{n}_b} \sum_{i \in \mathcal{I}_b} \mathrm{Log}_{\tilde{n}_b}(n_i) \mathrm{Log}_{\tilde{n}_b}(n_i)^T {}^{\tilde{n}_a}R_{\tilde{n}_b}^T + 2\, {}^{\tilde{n}_a}R_{\tilde{n}_b} \left(\sum_{i \in \mathcal{I}_b} \mathrm{Log}_{\tilde{n}_b}(n_i) \right) \mathrm{Log}_{\tilde{n}_a}(\mu_b)^T \\ &+ N_b \mathrm{Log}_{\tilde{n}_a}(\mu_b) \mathrm{Log}_{\tilde{n}_a}(\mu_b)^T \end{aligned} \tag{4.43}$$

$$= {}^{\tilde{n}_a}R_{\tilde{n}_b} S_b {}^{\tilde{n}_a}R_{\tilde{n}_b}^T + N_b \mathrm{Log}_{\tilde{n}_a}(\mu_b) \mathrm{Log}_{\tilde{n}_a}(\mu_b)^T \,, \tag{4.44}$$

where we have used the fact, that the Karcher mean algorithm gives us $\tilde{n}_b$ such that $\sum_{i \in \mathcal{I}_b} \mathrm{Log}_{\tilde{n}_b}(n_i) = 0$. Equation (4.44) gives us an approximate way of computing the statistics $\tilde{S}_b$ in $T_{\tilde{n}_a}\mathbb{S}^{D-1}$ using only the already computed statistics S_b in $T_{\tilde{n}_b}\mathbb{S}^{D-1}$ and the mean $\tilde{n}_b$ of cluster b. Note that we can do exactly the same computation for cluster c to obtain $\tilde{S}_c$.

The approximation we made lies in the fact that $\{n_i\}_{\mathcal{I}_b}$ were linearized around $\tilde{n}_b$ and hence the deviations from $\tilde{n}_b$ which they describe are only valid in $T_{\tilde{n}_b}\mathbb{S}^{D-1}$. By approximating

$$\mathrm{Log}_{\tilde{n}_a}(n_i) \approx \mathrm{Log}_{\tilde{n}_a}(\mu_b) + {}^{\tilde{n}_a}R_{\tilde{n}_b} \mathrm{Log}_{\tilde{n}_b}(n_i) \tag{4.45}$$

we make a small error that stems from the different linearizations. However, if the spread of cluster b (or c) is small, the approximation error is small. Note that after burn-in of the MCMC algorithm $\tilde{n}_b \approx \mu_b$ and Eq. 4.45 thus converges to the log map approximation discussed in Sec. 2.6.2.

Using $\tilde{S}_b$ and $\tilde{S}_c$ we compute the scatter matrix S_a of the merged cluster in $T_{\tilde{n}_a}\mathbb{S}^{D-1}$ as

$$S_a = \tilde{S}_b + \tilde{S}_a - N_a \bar{\breve{x}}_a \bar{\breve{x}}_a^T \tag{4.46}$$

$$\begin{aligned} =& {}^{\tilde{n}_a}R_{\tilde{n}_b} (S_b + N_b \bar{\breve{x}}_b \bar{\breve{x}}_b^T) {}^{\tilde{n}_a}R_{\tilde{n}_b}^T + N_b \mathrm{Log}_{\tilde{n}_a}(\tilde{n}_b) \mathrm{Log}_{\tilde{n}_a}(\tilde{n}_b)^T \\ &+ {}^{\tilde{n}_a}R_{\tilde{n}_c} (S_c + N_c \bar{\breve{x}}_c \bar{\breve{x}}_c^T) {}^{\tilde{n}_a}R_{\tilde{n}_c}^T + N_c \mathrm{Log}_{\tilde{n}_a}(\tilde{n}_c) \mathrm{Log}_{\tilde{n}_a}(\tilde{n}_c)^T - N_a \bar{\breve{x}}_a \bar{\breve{x}}_a^T \end{aligned} \tag{4.47}$$

■ 4.3.4 Evaluation and Results

In the following we compare the DP-TGMM inference algorithm on synthetic data with ground-truth labels against four related algorithms. Subsequently, we evaluate the clustering on real data, namely, surface normals extracted from Kinect depth images, and 20-dimensional semantic word vectors [165]. All MCMC inference algorithms are evaluated based on one sample from the Markov chain after burn-in.

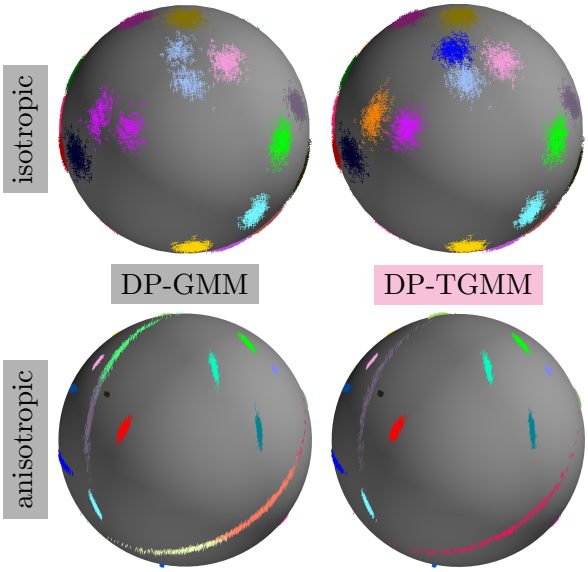


Figure 4.10: Visualization of synthetic datasets of 30 isotropic (top) and anisotropic (bottom) clusters on $\mathbb{S}^2$. In the sphere-plots to the right it can be observed that, in contrast to the DP-TGMM, the DP-GMM fails to separate (top) or incorrectly splits (bottom) clusters.

Comparisons on Synthetic Data We generate ground-truth data on the 3D unit sphere by sampling from a 30-component mixture model with equi-probable classes. The cluster means are drawn from a uniform distribution on the unit sphere and covariances from an IW prior. As depicted in Fig. 4.10 the datasets for evaluation encompass isotropic as well as anisotropic data.

We compare the DP-TGMM with two commonly-used optimization-based clustering algorithms, k-means [101] and spherical k-means (spkm) [61], as well as with the finite symmetric Dirichlet approximation (FSD-TGMM) [120] to the DP-TGMM. Additionally, we show the performance of the DP-GMM, a BNP infinite GMM, that does not exploit the geometry of the sphere. DP-GMM inference uses the sub-cluster-split algorithm [42]. All algorithms are initialized with a random labeling of the data. We use normalized mutual information (NMI) [227] between the groundtruth and the inferred labels as a measure for clustering quality which penalizes the use of superfluous clusters. Additionally, we show the number of clusters per iteration, which changes only for the BNP models.

From the right plots in Fig. 4.11 it can be seen that the inference for the DP-TGMM finds the true number of clusters in both cases, while the DP-GMM does not. The FSD-TGMM method gives an incorrect estimate of the number of clusters, which is consistent with what was observed in [43]. This motivates the need for our proposed sub-cluster inference algorithm. The depiction of the clustering results on the sphere

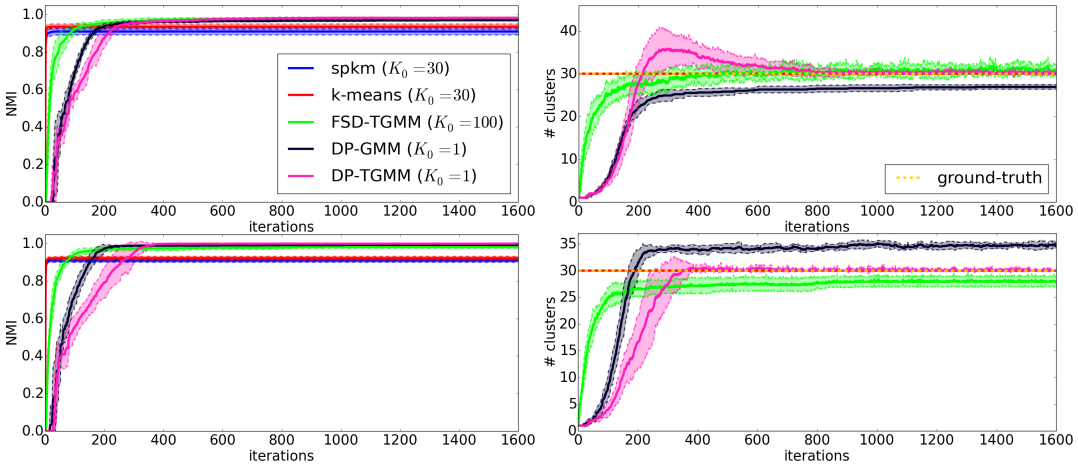


Figure 4.11: Mean and standard deviation over ten sampler runs of normalized mutual information (NMI) and cluster-count for synthetic datasets of 30 isotropic (top) and anisotropic (bottom) clusters on $\mathbb{S}^2$. The colors for the different algorithms are consistent across all plots.

in Fig. 4.10, shows that the DP-GMM fails to separate isotropic clusters and splits anisotropic clusters incorrectly. The parametric algorithms, k-means and spkm, were set to the true number of clusters, which is unknown in many problems of interest.

The evolution of the NMI with iterations, depicted in the left plots of Fig. 4.11, shows that the optimization-based methods quickly converge to a (sub-optimal) solution. The sampling based algorithms generally achieve better solutions. The DP-TGMM finds the best fit to the data as it explicitly allows for anisotropic distributions and respects the geometry of the sphere.

These findings are corroborated by additional results on different synthetic datasets displayed in Fig. 4.12 The two rows show the NMI and cluster-counts for two different mixed isotropic and anisotropic datasets. The dataset in the second row is more difficult since it contains more spread-out clusters with overlaps.

All the algorithms converge to close to the true number of clusters for the dataset in the first row. For the more difficult dataset in the second row, the DP-GMM and the FSD-TGMM both overestimate the number of clusters while the DP-TGMM converges to the true number of clusters on average. The larger standard deviation of the DP-TGMM and the overestimation of the number of clusters for the DP-GMM and the FSD-TGMM are likely due to the spread-out and overlapping clusters in the dataset.

Surface Normals in Point-cloud Data Surface normals extracted from point-clouds exhibit clusters on the unit sphere since planes in a scene create sets of normals pointing into the same direction. Hence, clustering these normals amounts to segmenting the scene into planes with similar orientation. We extract surface normals from raw Kinect depth images of the NYU V2 dataset [173] using the algorithm described in [112] and

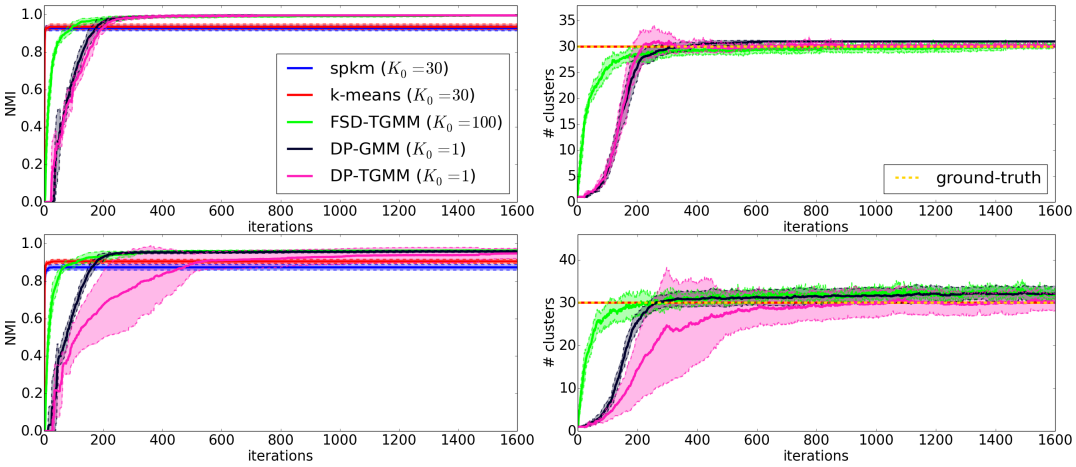


Figure 4.12: Mean and standard deviation over ten sampler runs of normalized mutual information (NMI) and cluster-count for synthetic datasets of 30 more challenging mixed isotropic and anisotropic clusters on $\mathbb{S}^2$.

apply total variation regularization [201] to smooth the initial normal estimate.

We show a set of example scenes in Fig. 4.13 and the segmentation into planes with equal orientation as implied by the clustering of normals on the unit sphere obtained using three different algorithms. In the second and third row we display results from clustering with spkm where $k = 4$ and 5, in the fourth row clusterings obtained using the DP-TGMM inference algorithm, and in the last row the segmentation obtained with the DP-GMM sub-cluster algorithm. Note that the scene images in the first row are only for reference – only surface normals were used as input to the algorithms. The DP-GMM as well as the DP-TGMM inference was initialized to two clusters with the hyper-parameters of the IW prior set to $\nu = 10k$ and $\Delta = (12^\circ)^2\nu\mathrm{I}_{3\times3}$. Each scene contains around $300k$ data points on $\mathbb{S}^2$.

The differences in segmentation illuminate the shortcomings of spkm and DP-GMM. The spkm algorithm finds a decent segmentation, but we get spurious clusters since the number of clusters is generally unknown. The inferred DP-GMM tends to under-segment the data because it ignores the manifold of the data and hence does not properly split clusters of normals in the presence of significant noise in the real data. For example in column two and five of Fig. 4.13 the floor and the wall are not separated into distinct clusters. By respecting the manifold as well as adopting a BNP model the DP-TGMM infers the intuitively correct segmentation as can be seen in the last row of Fig. 4.13.

Clustering of Semantic Word-Vectors We extract 20-dimensional semantic word vectors [165] from the English Wikipedia corpus and filter out all words with less than 100 counts to arrive at a set of 41k semantic word vectors for English words. Note that we normalize the word vectors to unit length before clustering. This is motivated by the fact that [165] utilizes the cosine similarity to find the semantically closest word to a

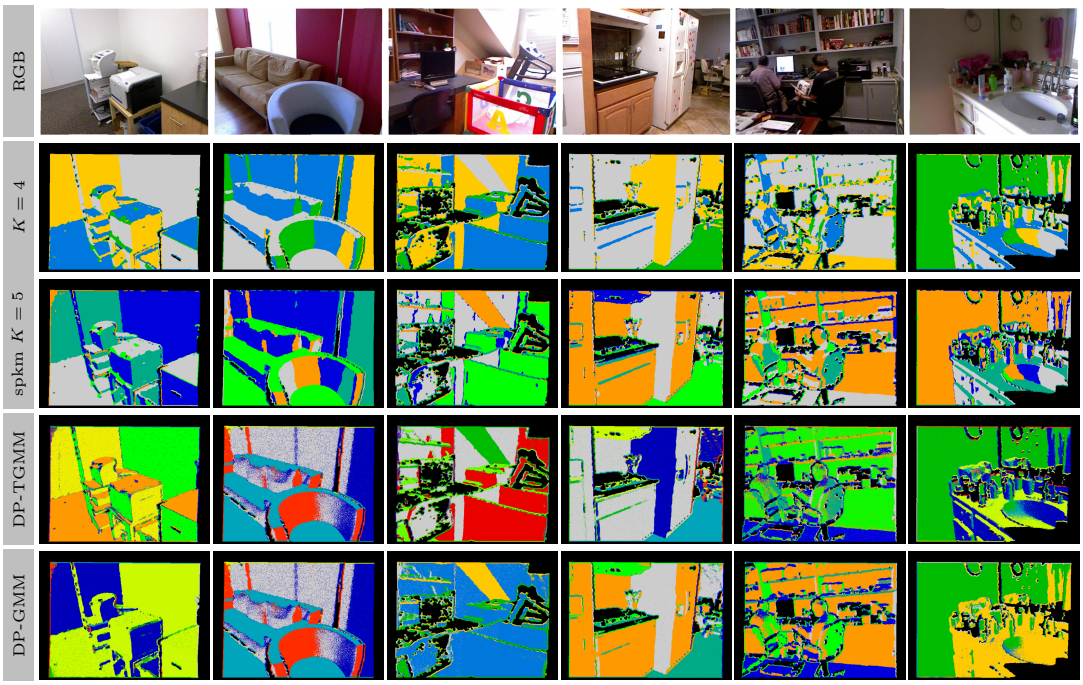


Figure 4.13: Directional segmentation of scenes from the NYU v2 RGB-D dataset [173] as implied by surface normal clusters. The complexity of the scenes increases from left to right as can be observed from the RGB images in the top row. The second and third show directional segmentations obtained via spherical k-means for comparison. The fourth and fifth rows depict samples from the DP-TGMM and DP-GMM respectively after 200 MCMC iterations. Note that DP-TGMM adapts the number of clusters to the complexity of the scene while DP-GMM fails to split clusters. Black denotes missing data due to sensor limitations.

"finance"	"music"	"religion "	"leisure"	"government"	"f
funding	symphonic	orthodoxy	malls	parliamentary	to
prospective	operatic	orthodox	hotels	enacted	ed
loans	soloists	evangelical	dining	parliament	fr
financing	orchestral	christians	nightlife	delegation	m
funds	music	primacy	outdoor	unanimously	m
contracts	waltz	preaching	upscale	granting	ve
compensation	trios	doctrines	shopping	mandate	ju
regulations	lute	rabbis	restaurants	constitutional	ba
assets	soloist	clergy	taverns	citizenship	co
investors	flute	catholicism	shops	committee	ta

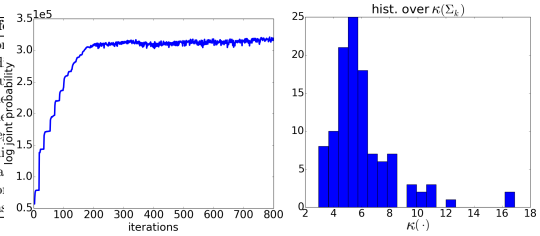


Figure 4.14: Evaluation of DP-TGMM inference on 20D semantic word vectors trained on the Wikipedia corpus. Left: most likely words for 6 clusters. Right: log probability and histogram over condition numbers for clusters.

given location in the vector space. The use of the cosine similarity is equivalent to the assumption that all the information about semantic proximity resides in the angular difference.

Therefore, by clustering the semantic word-vectors by their directions we obtain clusters of semantically similar words as can be observed in the table of Fig. 4.14. The table lists the ten most likely words of a subset of the clusters obtained when running the DP-TGMM inference algorithm. We start the algorithm from 20 centroids and run it for 800 iterations. After about 250 iterations the algorithm converges to 96 clusters. Note that this clustering is different from conventional topic modeling, which relies on document-level word counts. Semantic word-vectors depend on nearby words, and our clustering disregards document groupings.

To validate our hypothesis that real directional data exhibits anisotropic distributions on the sphere, we compute the condition number of the inferred cluster covariance matrices $\kappa(\Sigma_k) = \frac{\max[\sigma(\Sigma_K)]}{\min[\sigma(\Sigma_K)]}$ where $\sigma(\Sigma_K)$ is the set of all eigenvalues of Σ_k. The condition number thus serves as a measure for how elliptical the clusters are: $\kappa(\Sigma_k) \approx 1$ means the cluster is isotropic whereas $\kappa(\Sigma_k) \gg 1$ indicates an elliptical or anisotropic distribution. The spread-out histogram over condition numbers shown in Fig. 4.14 indicates that the inferred covariances are indeed anisotropic.

■ 4.4 Fast Nonparametric Directional Clustering Algorithms for Batch and Streaming Data

After introducing the DP-TGMM that can demonstrably capture anisotropic directional distributions, we now turn to the problem of clustering large amounts of directional data more efficiently by relaxing the expressiveness of the mixture component distribution to model only isotropic von-Mises-Fisher distributions.

Based on the low-variance limit of Bayesian nonparametric von-Mises-Fisher (vMF) mixture distributions, we propose two new flexible and efficient k-means-like clustering algorithms for directional data such as surface normals. The first, DP-vMF-means, is a batch clustering algorithm derived from the Dirichlet process (DP) vMF mixture. Recognizing the sequential nature of data collection in many applications, we extend

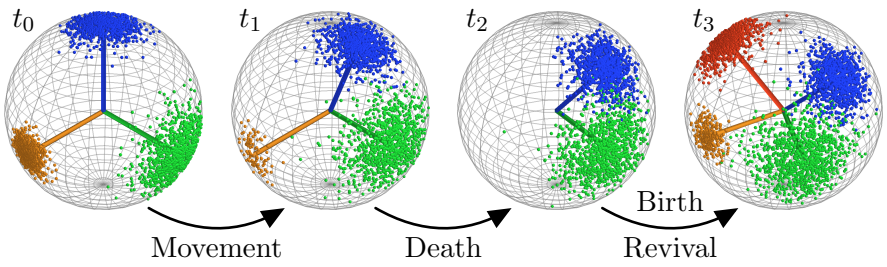


Figure 4.15: Evolution of the data distribution on the sphere which we model using a Dependent Dirichlet Process von-Mises-Fisher mixture model (DDP-vMF-MM).

this algorithm to DDP-vMF-means, which infers temporally evolving cluster structure from streaming data as depicted in Fig. 4.15. Both algorithms naturally respect the geometry of directional data, which lies on the unit sphere. We demonstrate their performance on synthetic directional data and real 3D surface normals from RGB-D sensors. While our experiments focus on 3D data, both algorithms generalize to high dimensional directional data such as protein backbone configurations and semantic word vectors.

In the following we first introduce the low-variance asymptotic analysis of vMF-based mixture models by re-deriving the spherical k-means algorithm from a Dirichlet Distribution vMF-MM. Second we introduce the DP-vMF-MM and derive the DP-vMF-means algorithm from it before extending the analysis to the dependent DP-vMF-MM and the DDP-vMF-means algorithm.

■ 4.4.1 Dirichlet Distribution vMF-MM

A finite mixture of K vMF distributions (introduced in Sec. 2.6.3) with known concentration τ may be obtained by placing a Dirichlet distribution prior $\mathrm{Dir}(\alpha)$ on the mixture weights π, and a vMF prior on the mean directions μ_k. Data points $\mathbf{n} \triangleq \{n_i\}_{i=1}^N$ are assigned to clusters via latent indicator variables $\mathbf{z} = \{z_i\}_{i=1}^N$. From a generative modeling perspective one first samples mixture weights π, labels $\mathbf{z}$ and mean vMF parameters:

$$\begin{aligned} \pi &\sim \mathrm{Dir}(\alpha) \\ z_i &\sim \mathrm{Cat}(\pi) \qquad \forall i \in \{1,\dots,N\} \\ \mu_k &\sim \mathrm{vMF}(\mu_0, \tau_0) \quad \forall k \in \{1,\dots,K\}\,. \end{aligned} \tag{4.48}$$

Given labels and vMF parameters the data $\mathbf{n}$ are sampled as:

$$n_i \sim \mathrm{vMF}(\mu_{z_i}, \tau) \quad \forall i \in \{1,\dots,N\}\,. \tag{4.49}$$

Let $\mathbf{z} = \{z_i\}_{i=1}^N$, and $\boldsymbol{\mu} = \{\mu_k\}_{k=1}^K$. Further, let negative subscript j, $\mathbf{u}_{-j} = \mathbf{u} \setminus u_j$ denote removal of item j from a set, and $\mathcal{I}_k$ denote the set $\{i : z_i = k\}$. The Gibbs

sampling inference algorithm for the finite vMF mixture iterates between sampling the label z_i given $\{\mathbf{z}_{-i}, \boldsymbol{\mu}\}$ and the mean direction μ_k given $\{\mathbf{z}\}$. The posterior to sample labels from is

$$p(z_i = k \mid \mathbf{z}_{-i}, \boldsymbol{\mu}, \mathbf{n}) = \frac{p(z_i = k \mid \mathbf{z}_{-i})\,\mathrm{vMF}(n_i; \mu_k, \tau)}{\sum_{\kappa=1}^{K} p(z_i = \kappa \mid \mathbf{z}_{-i})\,\mathrm{vMF}(n_i; \mu_\kappa, \tau)}\,, \tag{4.50}$$

and, as introduced in Sec. 2.6.3, the posterior for sampling the means μ_k from is

$$p(\mu_k \mid \mathbf{z}, \mathbf{n}) = \mathrm{vMF}\left(\mu_k; \tfrac{\vartheta_k}{\|\vartheta_k\|_2}, \|\vartheta_k\|_2\right)\,, \tag{4.51}$$

where $\vartheta_k = \tau_0\mu_0 + \tau\sum_{i\in\mathcal{I}_k} n_i$.

Parallel to the connection between k-means and the Gaussian mixture model in the small-variance asymptotic limit (see Sec. 2.5), we can derive a deterministic clustering algorithm, called spherical k-means [14]. Taking the small-variance limit in the Gaussian case amounts to analyzing the limit as the Gaussian becomes a delta function. The parallel for the vMF distribution is the limit as the concentration goes to infinity as depicted in Fig. 4.16. Taking $\tau \to \infty$ yields the following deterministic updates:

$$\begin{aligned}\lim_{\tau\to\infty} p(z_i = k \mid \mathbf{z}_{-i}, \boldsymbol{\mu}, \mathbf{n}) &= \lim_{\tau\to\infty} \frac{p(z_i = k \mid \mathbf{z}_{-i})\,\mathrm{vMF}(n_i; \mu_k, \tau)}{\sum_{\kappa=1}^{K} p(z_i = \kappa \mid \mathbf{z}_{-i})\,\mathrm{vMF}(n_i; \mu_\kappa, \tau)} \\ = \lim_{\tau\to\infty} \frac{p(z_i = k \mid \mathbf{z}_{-i})\exp(\tau\mu_k^T n_i)}{\sum_{k=1}^{K} p(z_i = \kappa \mid \mathbf{z}_{-i})\exp(\tau\mu_k^T n_i)} &= \begin{cases} 1 & n_i^T\mu_k \geq n_i^T\mu_\kappa\ \forall\kappa \in \{1,\dots,K\} \\ 0 & \text{otherwise} \end{cases}\end{aligned} \tag{4.52}$$

and

$$\begin{aligned}\lim_{\tau\to\infty} p(\mu_k \mid \mathbf{z}, \mathbf{n}) &= \lim_{\tau\to\infty} Z(\|\vartheta_k\|_2)\exp(\mu_k^T\vartheta_k) \\ &= \lim_{\tau\to\infty} Z\left(\|\tau_0\mu_0 + \tau\textstyle\sum_{i\in\mathcal{I}_k} n_i\|_2\right)\exp\left(\tau_0\mu_k^T\mu_0 + \tau\mu_k^T\textstyle\sum_{i\in\mathcal{I}_k} n_i\right) \\ &= \lim_{\tau\to\infty} Z\left(\|\tau\textstyle\sum_{i\in\mathcal{I}_k} n_i\|_2\right)\exp\left(\tau\mu_k^T\textstyle\sum_{i\in\mathcal{I}_k} n_i\right) \\ &= \delta\left(\mu_k = \left\lceil\textstyle\sum_{i\in\mathcal{I}_k} n_i\right\rfloor\right)\,.\end{aligned} \tag{4.53}$$

These two updates together form the spherical k-means algorithm [14]. The algorithm iterates between assigning data points n_i to the closest cluster via label z_i and recomputing the cluster means μ_k:

$$\begin{aligned} z_i &\leftarrow \operatorname*{arg\,max}_{k\in\{1,\dots,K\}} \mu_k^T n_i \qquad \forall i \in \{1,\dots,N\} \\ \mu_k &\leftarrow \left\lceil\textstyle\sum_{i\in\mathcal{I}_k} n_i\right\rfloor \qquad \forall k \in \{1,\dots,K\}\,. \end{aligned} \tag{4.54}$$

If the number of desired clusters is known the spherical k-means algorithm provides a data-adaptive clustering of $\{n\}_{i=1}^N$. Often the number of clusters is not known a priori and should be part of the inference procedure. This motivates the small-variance analysis of the Dirichlet process vMF mixture model in the next section.

■ 4.4.2 Dirichlet Process vMF-MM

As reviewed in Sec. 2.4, the Dirichlet process (DP) [70, 233] has been widely used as a prior for mixture models with a countably-infinite set of clusters [8, 15, 42, 174]. Assuming a base distribution $\text{vMF}(\mu; \mu_0, \tau_0)$, the DP is an appropriate prior for a vMF mixture with an unknown number of components and known vMF concentration τ. See Sec. 2.6.3 for an in depth characterization of the vMF distribution. Gibbs sampling inference only differs from the finite Dirichlet vMF-MM in the label sampling step; the mixing weights π are integrated out, resulting in the Chinese Restaurant Process (CRP) [25, 174]

$$p(z_i = k \mid \mathbf{z}_{-i}, \boldsymbol{\mu}, \mathbf{n}; \tau, \mu_0, \tau_0) \propto \begin{cases} |\mathcal{I}_k| \, \text{vMF}(n_i \mid \mu_k; \tau) & k \leq K \\ \alpha \, p(n_i; \tau, \mu_0, \tau_0) & k = K + 1 \,, \end{cases} \tag{4.55}$$

where $\mathcal{I}_k$ is the set of data indices assigned to cluster k not counting the previous label of z_i. Note that the DP concentration parameter, $\alpha > 0$, influences the likelihood of adding a new clusters. The conjugate prior for μ_k yields $p(n_i; \tau, \mu_0, \tau_0)$ via marginalization:

$$p(n_i; \tau, \mu_0, \tau_0) = \int_{\mathbb{S}^{D-1}} \text{vMF}(n_i \mid \mu_k; \tau) \, \text{vMF}(\mu_k; \mu_0, \tau_0) \, \mathrm{d}\mu_k = \frac{Z(\tau) Z(\tau_0)}{Z(\|\tau n_i + \tau_0 \mu_0\|_2)} \,. \tag{4.56}$$

Refer to Sec. 2.6.3 for a detailed derivation.

■ 4.4.3 DP-vMF-means

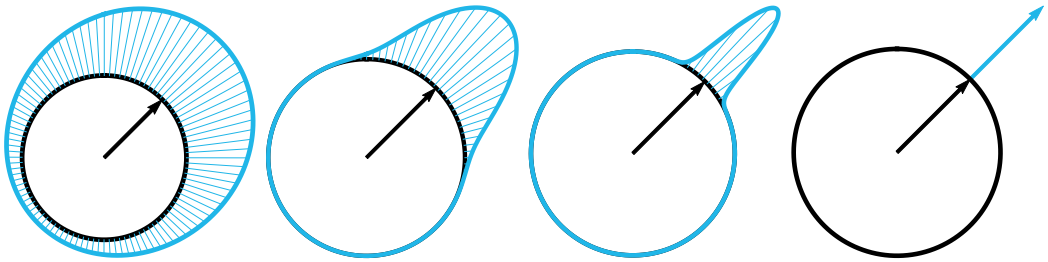
In this section, we provide a small-variance asymptotic analysis of the label and parameter update steps of the Gibbs sampling algorithm for the DP-vMF mixture, yielding deterministic updates. Figure 4.16 shows that, similar to the Gaussian case, the vMF distribution approaches a delta function as the concentration increases. As with k-means, the label assignments are computed sequentially for all datapoints before the means are updated, and the process is iterated until convergence as outlined in Algorithm 10.

Label Update To derive a hyperspherical analog to DP-means [142], consider the limit of the label sampling step (4.55) as $\tau \to \infty$. The normalizer $Z(\|\tau n_i + \tau_0 \mu_0\|_2)$ approaches

$$Z(\|\tau n_i + \tau_0 \mu_0\|_2) \overset{\tau^{(*)}}{\to} \exp(-\tau) \,, \tag{4.57}$$

where overscript $\tau^{(*)}$ denotes proportionality up to a finite power of τ, and where we have used the fact that as $\tau \to \infty$, the modified Bessel function of the first kind satisfies [1]

$$\mathrm{I}_{D/2-1}(\tau) = \frac{\exp(\tau)}{\sqrt{2\pi\tau}} \left(1 - O\left(\frac{1}{\tau}\right)\right) \overset{\tau^{(*)}}{\to} \exp(\tau) \,. \tag{4.58}$$



(a) $\tau = 1$ (b) $\tau = 10$ (c) $\tau = 100$ (d) $\tau \to \infty$

Figure 4.16: Depiction of 2D von-Mises-Fisher distributions with increasing concentration τ. As $\tau \to \infty$ the von-Mises-Fisher distribution approaches a delta function on the sphere at its mode μ.

To achieve a nontrivial result, the asymptotic behavior of $Z(\tau)$ must be matched by α, so let $\alpha = \exp(\lambda\tau)$ to obtain

$$\frac{\alpha Z(\tau) Z(\tau_0)}{Z(\|\tau n_i + \tau_0 \mu_0\|_2)} \overset{\tau(*)}{=} Z(\tau) \exp(\tau(\lambda + 1)) \,. \tag{4.59}$$

Therefore, as $\tau \to \infty$, the label sampling step becomes

$$\lim_{\tau\to\infty} p(z_i = k \mid \mathbf{z}_{-i}, \boldsymbol{\mu}, \mathbf{n}; \tau, \mu_0, \tau_0) = \lim_{\tau\to\infty} \begin{cases} \frac{|\mathcal{I}_k| e^{\tau(n_i^T \mu_k - \lambda - 1)}}{\sum_{j=1}^K |\mathcal{I}_j| e^{\tau(n_i^T \mu_j - \lambda - 1)} + c(\tau)} & k \leq K \\ \frac{c(\tau)}{\sum_{j=1}^K |\mathcal{I}_j| e^{\tau(n_i^T \mu_j - \lambda - 1)} + c(\tau)} & k = K + 1 \,, \end{cases} \tag{4.60}$$

where we have used that the normalizers $Z(\tau)$ of $\text{vMF}(n_i \mid \mu_k; \tau)$ and Eq. (4.59) cancel and $c(\tau) \overset{\tau(*)}{=} 1$. Thus, as $\tau \to \infty$, sampling from $p(z_i \mid \mathbf{z}_{-i}, \boldsymbol{\mu}, \mathbf{n}; \tau, \mu_0, \tau_0)$ is equivalent to the following assignment rule:

$$z_i = \underset{k \in \{1,\ldots,K+1\}}{\arg\max} \begin{cases} n_i^T \mu_k & k \leq K \\ \lambda + 1 & k = K + 1 \,. \end{cases} \tag{4.61}$$

Since $-1 \leq n_i^T \mu_k \leq 1$, the parameter λ can be restricted to the set $\lambda \in [-2, 0]$ without loss of generality. Intuitively λ defines the maximum angular spread ϕ_λ of clusters about their mean direction, via $\lambda = \cos(\phi_\lambda) - 1$. Note, that upon assigning a datapoint to a new cluster, i.e. $z_i = K + 1$, the mean of that cluster is initialized to $\mu_{K+1} = n_i$. Finally, if an observation n_i is the last one in its cluster, the cluster is removed prior to finding the new label for n_i using Eq. (4.61).

Parameter Update Taking $\tau \to \infty$ in the parameter posterior for cluster k from Eq. (2.86) causes τ_0 and μ_0 to become negligible. Hence the parameter update be-

comes:

$$\mu_k = \left\lceil \sum_{i \in \mathcal{I}_k} n_i \right\rceil = \frac{\sum_{i \in \mathcal{I}_k} n_i}{\| \sum_{i \in \mathcal{I}_k} n_i \|_2} \quad \forall k \in \{1, \ldots, K\} \,. \tag{4.62}$$

This amounts to summing all data points in cluster k. In practice one can keep track of these sums by adding and removing data points n_i as their cluster assignments change. That way the parameter update step just involves normalizing k vectors to unit length. The increase in efficiency is especially noticeable once the algorithm gets close to convergence, when only very few data points are reassigned each iteration.

Objective Function From Eq. (4.61) we can see that assigning a datapoint n_i to cluster k provides a score of $n_i^T \mu_k$, whereas adding a new cluster provides a score of $\lambda + 1 - n_i^T \mu_{K+1} = \lambda$, since new mean directions are initialized directly to $\mu_{K+1} = n_i$. Hence, the objective function that DP-vMF-means maximizes is

$$J_{\text{DP-vMF}} = \sum_{k=1}^{K} \sum_{i \in \mathcal{I}_k} n_i^T \mu_k + \lambda K \,. \tag{4.63}$$

In practice the objective is used to determine convergence of the DP-vMF-means algorithm.

DP-vMF-means algorithm The DP-vMF-means algorithm is given in full detail in Algorithm 10. For the label assignment step both Algorithms 11 and 14 can be used. The former, sequential label assignment Algorithm 11 is directly derived from the Gibbs sampling posterior. The latter label assignment algorithm is parallelized using the concept of optimistic iterated restarts as described in Sec. 4.4.6.

$J_{\text{DP-vMF}} \leftarrow \infty$
$\boldsymbol{\mu} \leftarrow \emptyset$
while $J_{\text{DP-vMF}}$ not converged **do**
  $\{z_i\}_{i=1}^N, \boldsymbol{\mu} \leftarrow$DP-vMF-meansLabelAssignments$(\{n_i\}_{i=1}^N, \boldsymbol{\mu}, \lambda)$
  for $k \in \{1, \ldots, |\boldsymbol{\mu}|\}$ **do**
    if $n_k > 0$ **then**
      $\mu_k \leftarrow \left\lceil \sum_{\mathcal{I}_k} n_i \right\rceil$
    else
      $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \setminus \mu_k$ // remove cluster k
    end if
  end for
  $J_{\text{DP-vMF}} \leftarrow \sum_{k=1}^{|\boldsymbol{\mu}|} \sum_{i \in \mathcal{I}_k} n_i^T \mu_k + \lambda |\boldsymbol{\mu}|$
end while

Algorithm 10: DP-vMF-means algorithm.

function DP-vMF-meansSequentialLabelAssignment($\{n_i\}_{i=1}^N, \boldsymbol{\mu}, \lambda$)
  for $i \in \{1, \ldots, N\}$ **do**
    // only consider clusters that contain more than n_i
    $z_i \leftarrow \underset{k \in \{1,\ldots,|\boldsymbol{\mu}|+1\}}{\arg\max} \begin{cases} n_i^T \mu_k & k \leq |\boldsymbol{\mu}| \text{ and } |\mathcal{I}_k \setminus z_i| > 0 \\ \lambda + 1 & k = |\boldsymbol{\mu}| + 1 \end{cases}$
    if $z_i = |\boldsymbol{\mu}| + 1$ **then**
      $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \cup \{\mu_{|\boldsymbol{\mu}|+1} \leftarrow n_i\}$ // add cluster $K+1$ and initialize to n_i
    end if
  end for
  return $\{z_i\}_{i=1}^N, \boldsymbol{\mu}$
end function

Algorithm 11: DP-vMF-means sequential label assignments algorithm.

■ 4.4.4 Dependent Dirichlet Process vMF-MM

Suppose now that, in addition to an unknown number K of components, the vMF mixture undergoes temporal evolution in discrete timesteps $t \in \mathbb{N}$ as depicted in Fig. 4.15: mixture components can move, be destroyed, and new ones can be created at each timestep. For such a scenario, the dependent Dirichlet process (DDP) [39, 151, 160] is an appropriate prior over the mixture components and weights. Using intermediate auxiliary DPs F_0 and F_1, the DDP constructs a Markov chain of DPs G_t, where G_{t+1} is sampled from G_t as follows:

1. **(Death)** For each atom θ in G_t, sample from Bernoulli(q). If the result is 1, add θ to F_0.
2. **(Motion)** Replace each θ in F_0 with $\theta' \sim T(\theta' \mid \theta)$.
3. **(Birth)** Sample a DP $F_1 \sim \mathrm{DP}(\alpha, H)$. Let G_{t+1} be a random convex combination of F_0 and F_1.

There are four parameters in this model: $\alpha > 0$ and $H(\cdot)$, the concentration parameter and base measure of the innovation process; $q \in (0, 1)$, the Bernoulli cluster survival probability; and finally $T(\cdot \mid \cdot)$, the random walk transition distribution. In the present work, both the base and random transition distributions are von-Mises-Fisher: $H(\mu) = \mathrm{vMF}(\mu; \mu_0, \tau_0)$, and $T(\mu \mid \nu) = \mathrm{vMF}(\mu; \nu, \xi)$.

Suppose at timestep t, a new batch of data $\mathbf{n}$ is observed. Then Gibbs sampling posterior inference for the DDP mixture, as in the previous sections, iteratively samples labels and parameters. Let the set of tracked mean directions from previous timesteps be $\{\mu_{k0}\}_{k=1}^K$, where $\mu_{k0} \sim \mathrm{vMF}(m_k, \tau_k)$ and Δt_k denotes the number of timesteps since cluster k was last instantiated (i.e. when it last had data assigned to it).[1] Then the

[1] Note that all quantities ($\mathbf{n}$, μ_{k0}, m_k, τ_k, c_k, n_k, etc.) are now time-varying. This dependence is not shown in the notation for brevity, and all quantities are assumed to be shown for the current timestep t.

label sampling distribution is

$$p(z_i = k \mid \boldsymbol{\mu}, \mathbf{z}_{-i}, \mathbf{n}) \propto \begin{cases} \alpha \frac{1-q^t}{1-q} p(n_i; \mu_0, \tau_0) & k = K+1 \\ (c_k + |\mathcal{I}_k|)\, \mathrm{vMF}(n_i \mid \mu_k; \tau) & k \leq K, |\mathcal{I}_k| > 0 \\ q^{\Delta t_k} c_k p(n_i; m_k, \tau_k) & k \leq K, |\mathcal{I}_k| = 0\,, \end{cases} \tag{4.64}$$

where c_k is the number of observations assigned to cluster k in past timesteps. The parameter sampling distribution is

$$p(\mu_k \mid \boldsymbol{\mu}_{-k}, \mathbf{z}, \mathbf{n}) \propto \begin{cases} \mathrm{vMF}(\mu_k; \frac{\vartheta_k}{\|\vartheta_k\|_2}, \|\vartheta_k\|_2) & c_k = 0 \\ p(\mu_k \mid \mathbf{n}, \mathbf{z}; m_k, \tau_k) & c_k > 0\,, \end{cases} \tag{4.65}$$

where $\vartheta_k = \tau_0 \mu_0 + \tau \sum_{i \in \mathcal{I}_k} n_i$, and $p(\mu_k \mid \mathbf{n}, \mathbf{z}; m_k, \tau_k)$ is the distribution over the current cluster k mean direction μ_k given the assigned data and the old mean direction μ_{k0}.

■ 4.4.5 DDP-vMF-means

In the following, we analyze the small-variance asymptotics of the DDP-vMF mixture model. We first derive the label assignment rules, followed by the parameter updates.

Label Update First, let $\alpha = \exp(\lambda\tau)$, $q = \exp(Q\tau)$, $\xi = \exp(\beta\tau)$, and $\tau_k = \tau w_k$, with $\lambda \in [-2, 0]$ as before, $Q \leq 0$, and $\beta, w_k \geq 0$. Note that $\lim_{\tau\to\infty} \frac{1-q^t}{1-q} = 1$, and thus the asymptotics of the label assignment probability for current and new clusters is the same as in Section 4.4.3.

Hence, we focus on the assignment of a datapoint to a previously observed, but currently not instantiated, cluster k. During the Δt_k timesteps since cluster k was last observed, the mean direction μ_k underwent a random vMF walk $\mu_{k0} \to \mu_{k1} \to \cdots \to \mu_{k\Delta t_k} = \mu_k$ with initial distribution $\mu_{k0} \sim \mathrm{vMF}(\mu_{k0}; m_k, \tau_k)$. Therefore, the intermediate mean directions $\{\mu_{kn}\}_{n=1}^{\Delta t_k}$ must be marginalized out when computing $p(n_i; m_k, \tau_k)$:

$$\begin{aligned} p(n_i; m_k, \tau_k) &= \int \cdots \int_{\mu_{k0}, \ldots, \mu_{k\Delta t_k}} p(n_i \mid \mu_{k\Delta t_k}; \tau) \cdot p(\mu_0; m_k, \tau_k) \cdot \prod_{n=1}^{\Delta t_k} p(\mu_{kn} \mid \mu_{k(n-1)}; \xi) \\ &= Z(\tau) Z(\beta\tau)^{\Delta t_k} Z(w_k \tau) \int \cdots \int_{\mu_{k0}, \ldots, \mu_{k\Delta t_k}} \exp(\tau f) \\ f &= n_i^T \mu_{k\Delta t_k} + \beta \sum_{n=1}^{\Delta t_k} \mu_{kn}^T \mu_{k(n-1)} + w_k \mu_{k0}^T m_k\,. \end{aligned} \tag{4.66}$$

The integration in Eq. (4.66) cannot be computed in closed form; however, the value of the integral is only of interest in the limit as $\tau \to \infty$. Therefore, Theorem 4.4.1, an extension of Laplace's approximation to general differentiable manifolds, may be used to obtain an exact formula.

Theorem 4.4.1 (Manifold Laplace Approximation)**.** *Suppose $M \subset \mathbb{R}^n$ is a bounded m-dimensional differentiable manifold and $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth function on M. Further, suppose f has a unique global maximum on M, $x^\star = \arg\max_{x \in M} f(x)$. Then*

$$\lim_{\tau \to \infty} \frac{\int_M e^{\tau f(x)}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla^2 f(x^\star) U\right|^{\frac{1}{2}} e^{\tau f(x^\star)}} = 1\,, \tag{4.67}$$

where $U \in \mathbb{R}^{n \times m}$ is a matrix whose columns are an orthonormal basis for the tangent space of M at $x^\star$.

Proof. See Appendix B.1. The general technique of this proof is to transform coordinates between the manifold and its tangent plane using the exponential map [63], and then apply the multidimensional Laplace approximation in the transformed Euclidean space. ■

Corollary 4.4.1. *Given a smooth function $f : \left(\mathbb{R}^D\right)^N \to \mathbb{R}$, with a unique global maximum over the N-product of $(D-1)$-spheres $x^\star \in \left(\mathbb{S}^{D-1}\right)^N$,*

$$\int_{\left(\mathbb{S}^{D-1}\right)^N} e^{\tau f(x)} \overset{\tau(*)}{\to} e^{\tau f(x^\star)}\,. \tag{4.68}$$

Proof. This is Theorem 4.4.1 applied to $\left(\mathbb{S}^{D-1}\right)^N$. ■

Using Corollary 4.4.1 and the limiting approximation of the modified Bessel function in Eq. (4.58), Eq. (4.66)) yields the following asymptotic behavior as $\tau \to \infty$:

$$p(n_i; m_k, \tau_k) \overset{\tau(*)}{\to} \exp\left(\tau(f^\star - 1 - \beta \Delta t_k - w_k)\right)\,. \tag{4.69}$$

The only remaining unknown in the asymptotic expression, $f^\star$, can be found via constrained optimization

$$\begin{aligned} \max_{\{\mu_{kn}\}_{n=1}^{\Delta t_k}} \; & n_i^T \mu_{k\Delta t_k} + \beta \sum_{n=1}^{\Delta t_k} \mu_{kn}^T \mu_{k(n-1)} + w_k \mu_{k0}^T m_k \\ \text{s.t. } & \mu_{kn}^T \mu_{kn} = 1 \; \forall n \in \{0, \ldots, \Delta t_k\}\,. \end{aligned} \tag{4.70}$$

The optimization (4.70) has a closed-form solution:

$$\begin{aligned} \mu_{k0} &= \frac{w_k m_k + \beta \mu_{k1}}{||w_k m_k + \beta \mu_{k1}||_2} \\ \mu_{kn} &= \frac{\mu_{k(n+1)} + \mu_{k(n-1)}}{||\mu_{k(n+1)} + \mu_{k(n-1)}||_2} \; \forall n \in \{1, \ldots, \Delta t_k - 1\} \\ \mu_{k\Delta t_k} &= \frac{n_i + \beta \mu_{k(\Delta t_k - 1)}}{||n_i + \beta \mu_{k(\Delta t_k - 1)}||_2}\,. \end{aligned} \tag{4.71}$$

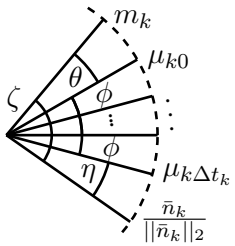


Figure 4.17: Geometry of the maximum likelihood setting of $\mu_{k0}, \mu_{k1}, \ldots, \mu_{k\Delta t_k}$ for the transition distribution.

These ternary relationships enforce that the optimal von-Mises-Fisher mean directions along the random walk lie on the geodesic between m_k and n_i. Therefore, this walk can be described geometrically by three angles, as shown in Fig. 4.17 (with $\bar{n}_k = n_i$): the angle ϕ between consecutive μ_{kn}, the angle η between n_i and $\mu_{k\Delta t_k}$, and the angle θ between m_k and μ_{k0}. Given these definitions, standard trigonometry yields a set of three equations in ϕ, η, and θ:

$$\begin{aligned} w_k \sin(\theta^\star) = \beta \sin(\phi^\star) = \sin(\eta^\star) \\ \zeta = \theta^\star + \Delta t_k \phi^\star + \eta^\star = \arccos(m_k^T n_i)\,, \end{aligned} \tag{4.72}$$

where ζ is the full angle between n_i and m_k. Since (4.72) cannot be solved in closed-form, Newton's method is used to compute $\phi^\star$, $\theta^\star$, and $\eta^\star$, which in turn determines $f^\star$:

$$f^\star = w_k \cos(\theta^\star) + \beta \Delta t_k \cos(\phi^\star) + \cos(\eta^\star)\,. \tag{4.73}$$

Returning to (4.69), the transition asymptotics are

$$p(n_i; m_k; \tau_k) \overset{\tau^{(*)}}{\to} \exp\left(\tau\left(w_k(\cos\theta^\star - 1) + \beta\Delta t_k(\cos\phi^\star - 1) + \cos\eta^\star - 1\right)\right)\,. \tag{4.74}$$

Substituting this into Eq. (4.64) with the earlier definition $q = \exp(\tau Q)$, and taking the limit $\tau \to \infty$ yields the assignment rule

$$z_i = \arg\max_k \begin{cases} \lambda + 1 & k = K+1 \\ \mu_k^T n_i & k \le K, |\mathcal{I}_k| > 0 \\ \begin{pmatrix} \Delta t_k \beta(\cos\phi^\star - 1) + \cos\eta^\star \\ +w_k(\cos\theta^\star - 1) + \Delta t_k Q \end{pmatrix} & k \le K, |\mathcal{I}_k| = 0\,. \end{cases} \tag{4.75}$$

Note that if $Q\Delta t_k < \lambda$, cluster k can be removed permanently as it will never be revived again. Furthermore, for any $Q \le \lambda$ all clusters are removed after each timestep, and the algorithm reduces to DP-vMF-means.

Parameter Update The parameter update rule for DDP-vMF-means comes from the asymptotic behavior of (4.65) as $\tau \to \infty$. The analysis for any new cluster is the same as that in Section 4.4.3, so our focus is again on the transitioned mean direction posterior $p(\mu_{k\Delta t_k} \mid \mathbf{n}, \mathbf{z}; m_k, \tau_k)$ (recall that $\mu_k = \mu_{k\Delta t_k}$ in the definition of the random vMF walk). This distribution can be expanded, similarly to (4.66), as:

$$\begin{aligned} p(\mu_{k\Delta t_k} \mid \mathbf{n}, \mathbf{z}; m_k, \tau_k) &= \int \cdots \int_{\mu_{k0},\ldots,\mu_{k(\Delta t_k - 1)}} p(\mathbf{n} \mid \mu_{k\Delta t_k}; \tau) p(\mu_0; m_k, \tau_k) \prod_{n=1}^{\Delta t_k} p(\mu_{kn} \mid \mu_{k(n-1)}; \xi) \\ &= Z(\tau)^{|\mathcal{I}_k|} Z(\beta\tau)^{\Delta t_k} Z(w_k\tau) \int \cdots \int_{\mu_{k0},\ldots,\mu_{k(\Delta t_k - 1)}} \exp(\tau f) \\ f &= \sum_{i \in \mathcal{I}_k} n_i^T \mu_{k\Delta t_k} + \beta \sum_{n=1}^{\Delta t_k} \mu_{kn}^T \mu_{k(n-1)} + w_k \mu_{k0}^T m_k \,. \end{aligned} \tag{4.76}$$

Define $\bar{n}_k = \sum_{i \in \mathcal{I}_k} n_i$. Once again, applying Corollary 4.4.1, the limit $\tau \to \infty$ removes the integrals over the marginalized mean directions. However, in contrast to the label assignment update, $\mu_{k\Delta t_k}$ is not marginalized out. Therefore, an additional maximization with respect to $\mu_{k\Delta t_k}$ to find the concentration point of the posterior yields

$$\begin{aligned} p(\mu_{k\Delta t_k} | \mathbf{n}, \mathbf{z}; m_k, \tau_k) &\overset{\tau(*)}{\to} \exp(\tau(f^\star - |\mathcal{I}_k| - \Delta t_k \beta - w_k)) \\ f^\star &= w_k \cos(\theta^\star) + \beta \Delta t_k \cos(\phi^\star) + ||\bar{n}_k||_2 \cos(\eta^\star) \,. \end{aligned} \tag{4.77}$$

Analyzing the geometry of the geodesic between $\bar{n}_k / \|\bar{n}_k\|_2$ and m_k (Fig. 4.17) there exist $\phi^\star$, $\theta^\star$ and $\eta^\star$ such that

$$\begin{aligned} w_k \sin(\theta^\star) = \beta \sin(\phi^\star) &= ||\bar{n}_k||_2 \sin(\eta^\star) \\ \zeta = \theta^\star + \Delta t_k \phi^\star + \eta^\star &= \arccos(m_k^T \tfrac{\bar{n}_k}{\|\bar{n}_k\|_2}) \,, \end{aligned} \tag{4.78}$$

which can be solved via Newton's method. Given the solution, μ_k can be obtained by rotating $\frac{\bar{n}_k}{\|\bar{n}_k\|_2}$ by angle $\eta^\star$ on the geodesic shown in Fig. 4.17 towards m_k,

$$\mu_k = R(\eta^\star) \tfrac{\bar{n}_k}{||\bar{n}_k||_2} \,. \tag{4.79}$$

Weight Update After the iteration of label and parameter updates has converged, the weight w_k must be updated for all clusters to reflect the new uncertainty in the mean direction of cluster k. This can be done by examining (4.77): Since at the maximum of a vMF$(\mu; m_k, w_k\tau)$ density, $\exp(\tau w_k m_k^T \mu) = \exp(\tau w_k)$, w_k is updated to $f^\star$.

Solving for state transition geometry for label and parameter update As depicted in Fig. 4.17 and as alluded to previously the setting of $\mu_{k0}, \mu_{k1}, \ldots, \mu_{k\Delta t_k}$ for the transition distribution can be described in terms of the angles ϕ, η, and θ. The maximum likelihood

parameters $\phi^\star$, $\eta^\star$ and $\theta^\star$ follow Eq. 4.78 for the parameter update with $\bar{n}_k = \sum_{i \in \mathcal{I}_k} n_i$ and Eq. 4.72 for the label assignment with $\bar{n}_k = n_i$.

This set of equations can be solved exactly and efficiently using Newton's method which in practice converges very quickly for this problem. Starting from $\phi = 0$, Newton's method iterates over the following steps until convergence of ϕ:

- compute $f(\phi) = \arcsin\left(\frac{\beta}{w_k}\sin(\phi)\right) + \Delta t_k \phi + \arcsin\left(\frac{\beta}{||\bar{n}_k||_2}\sin(\phi)\right) - \zeta$
- compute $\frac{\partial f(\phi)}{\partial \phi} = \Delta t_k + \frac{\beta\cos(\phi)}{\sqrt{||\bar{n}_k||_2^2 - \beta^2\sin^2(\phi)}} + \frac{\beta\cos(\phi)}{\sqrt{w_k^2 - \beta^2\sin^2(\phi)}}$
- update $\phi \leftarrow \phi - \frac{f(\phi)}{df(\phi)}$

A second approach, which is faster but approximate, is to assume that all angles are small. For small angles $\sin(\alpha) \approx \alpha$ is a good approximation. With this we obtain the following closed form solutions:

$$\phi^\star \approx \zeta\left[\beta\left(1 + \frac{1}{w_k}\right) + \Delta t_k\right]^{-1} \tag{4.80}$$

$$\theta^\star \approx \zeta\left[1 + w_k\left(1 + \frac{\Delta t_k}{\beta}\right)\right]^{-1} \tag{4.81}$$

$$\eta^\star \approx \zeta\left[1 + \frac{1}{w_k} + \frac{\Delta t_k}{\beta}\right]^{-1} \tag{4.82}$$

DDP-vMF-means algorithm Algorithm 12 outlines the necessary operations of the DDP-vMF-means algorithm per timestep. The sequential label assignment algorithm for DDP-vMF-means is shown in Algorithm 13.

■ 4.4.6 Optimistic Iterated Restarts (OIR)

In our implementation of the algorithm we pay special attention to speed and parallel execution to enable real-time performance for streaming RGB-D data. Observe that the main bottleneck of DP-based hard clustering algorithms, such as the proposed (D)DP-vMF-means, DP-means [142] or Dynamic means [39], is the inherently sequential assignment of labels: due to the creation of new clusters, the label assignments depend on all previous assignments. While this is a key feature of the streaming clustering algorithms, it poses a computational hindrance. We address this issue with an optimistic parallel label assignment procedure inspired by techniques for database concurrency control [183].

First, we compute assignments in parallel (e.g. on a GPU). If all datapoints were assigned only to instantiated clusters, we output the labeling. Otherwise, we find the lowest observation id i that modified the number of clusters, apply the modification, and recompute the assignments for all observations $i' > i$ in parallel. Thus, per data-batch, DP-vMF-means restarts once per new cluster, while DDP-vMF-means restarts once for each new or revived cluster.

$\boldsymbol{\mu} \leftarrow \emptyset$
$\{z_i\}_{i=1}^N \leftarrow$ unassigned
while not converged **do**
  $\{z_i\}_{i=1}^N, \boldsymbol{\mu} \leftarrow$DDP-vMF-meansLabelAssignments$(\{n_i\}_{i=1}^N, \boldsymbol{\mu}, \lambda)$
  for $k \in \{1, \ldots, |\boldsymbol{\mu}|\}$ **do**
    if $|\mathcal{I}_k| > 0$ **then** // if the cluster is currently instantiated
      if $k < |\boldsymbol{\mu}_{t-1}|$ **then** // cluster is not a novel cluster
        $\mu_k \leftarrow R(\eta^\star)\frac{\bar{n}_k}{\|\bar{n}_k\|_2}$ // reinstantiate the cluster
      else
        $\mu_k \leftarrow \frac{\sum_{\mathcal{I}_k} n_i}{\|\sum_{\mathcal{I}_k} n_i\|_2}$ // update the center of the novel cluster
      end if
    end if
  end for
end while
for $k \in \{1, \ldots, |\boldsymbol{\mu}|\}$ **do**
  if $|\mathcal{I}_{z_i}| > 0$ **then** // if the cluster is currently instantiated
    Solve for $\phi^\star$, $\theta^\star$, $\eta^\star$ in Eq. (4.72) with $\bar{n}_k = \sum_{i \in \mathcal{I}_k} n_i$ as described in Sec. 4.4.5
    $w_k = w_k \cos(\theta^\star) + \beta \Delta t_k \cos(\phi^\star) + \|\bar{n}_k\|_2 \cos(\eta^\star)$
  end if
  if $Q\Delta t_k < \lambda$ **then** // cluster cannot be revived again
    $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \setminus \mu_k$ // remove the cluster
  end if
end for

Algorithm 12: DDP-vMF-means algorithm for a single time-step.

Algorithm 14 details the optimistic iterated restarts algorithm which can be massively parallelized both in CPU and GPU for the DP-vMF-means algorithm. The OIR label assignment algorithm for DDP-vMF-means follows the same pattern as Alg. 14 with the additional possibility of reviving a cluster. Reviving a cluster changes the number of active clusters and thus requires a restart in the same way as creating a new cluster does.

■ 4.4.7 Evaluation and Results

In the following we first quantitatively evaluate the clustering quality achieved by DP-vMF-means in comparison to spkm on synthetic and then real-wold data derived form RGBD images of the NYU depth dataset. Second we qualitatively show the behavior and clustering quality of DDP-vMF-means.

Synthetic Data First, we evaluate the behavior of the DP-vMF-means algorithm in comparison to its parametric cousin, the spkm algorithm, on synthetic 3D spherical data sampled from $K_\mathrm{T} = 30$ true vMF distributions. All evaluation results are shown

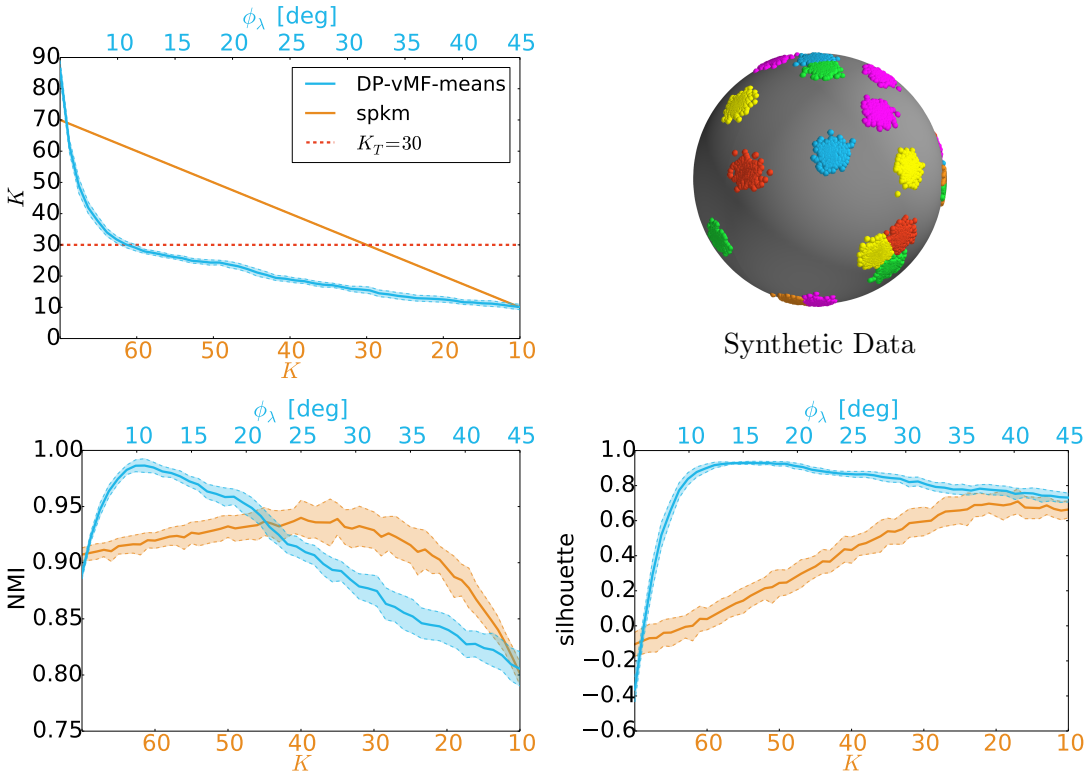


Figure 4.18: Comparison of the spkm and the DP-vMF-means clustering algorithms on synthetic spherical data with $K_{\mathrm{T}} = 30$ clusters. Note DP-vMF-means' better labeling quality indicated by higher maximum normalized mutual information (NMI) as well as silhouette score. Red doted lines indicate the parameters achieving the true number of clusters.

function DDP-vMF-meansSequentialLabelAssignment($\{n_i\}_{i=1}^N, \boldsymbol{\mu}, \lambda$)
  for $i \in \{1, \ldots, N\}$ **do**
    Solve for $\phi^\star$, $\theta^\star$, $\eta^\star$ in Eq. (4.72) with $\bar{n}_k = n_i$ as described in Sec. 4.4.5
    $$z_i \leftarrow \underset{k \in \{1,\ldots,|\boldsymbol{\mu}|+1\}}{\arg\max} \begin{cases} \lambda + 1 & k = |\boldsymbol{\mu}| + 1 \\ \mu_k^T n_i & k \leq |\boldsymbol{\mu}|, |\mathcal{I}_k \setminus z_i| > 0 \\ \Delta t_k \beta(\cos(\phi^\star) - 1) + \Delta t_k Q & \\ \quad + w_k(\cos(\theta^\star) - 1) + \cos(\eta^\star) & k \leq |\boldsymbol{\mu}|, |\mathcal{I}_k| = 0\,. \end{cases}$$
    if $z_i = |\boldsymbol{\mu}| + 1$ **then**
      $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \cup \{\mu_{|\boldsymbol{\mu}|+1} \leftarrow n_i\}$ // add cluster $K + 1$ and initialize to n_i
    else
      // check that cluster z_i is not yet instantiated
      if $|\mathcal{I}_{z_i}| = 0$ **then**
        $\mu_k \leftarrow R(\eta^\star) n_i$ // reinstantiate the cluster
      end if
    end if
  end for
  return $\{z_i\}_{i=1}^N, \boldsymbol{\mu}$
end function

Algorithm 13: DDP-vMF-means sequential label assignments.

as the mean and standard deviation over 50 runs. The top left plot of Fig. 4.18 depicts the inferred number of clusters K on the horizontal axis as a function of the respective parameters of the two algorithms: the number of clusters K for spkm and the parameter ϕ_λ for DP-vMF-means (recall that $\phi_\lambda = \cos^{-1}(\lambda + 1)$ as defined in Sec. 4.4.3). This figure demonstrates the ability of the DP-vMF-means algorithm to discover the correct number of clusters K_T, and the relative insensitivity of the discovered number of clusters with respect to its parameter ϕ_λ. The number of inferred clusters of the DP-vMF-means algorithm bends towards the true number of clusters, K_T, for a wide range of ϕ_λ parameters. This indicates the ability of the algorithm to adapt the number of clusters to the data and its relative insensitivity to ϕ_λ in comparison to K for spkm.

The bottom row of plots show two measures for clustering quality. The normalized Mutual Information (NMI) [227], depicted in the middle, is computed using the true labels. DP-vMF-means achieves an almost perfect NMI of 0.99, while spkm only reaches 0.94 NMI even with $K = K_\mathrm{T}$. The slightly superior performance of DP-vMF-means stems from its enhanced ability to avoid local optima due to the way labels are initialized: while spkm is forced to initialize K cluster parameters, DP-vMF-means starts with an empty set and adds clusters on the fly as more data are labelled. The NMI results are corroborated by the silhouette score [205], shown to the bottom right in Fig. 4.18. The silhouette score is an internal measure for clustering quality that can be computed without knowledge of the true clustering, and is used to tune parametric clustering algorithms. With a maximum of 0.92 DP-vMF-means reaches a close to perfect

function DP-vMF-meansOIRLabelAssignment($\{n_i\}_{i=1}^N, \boldsymbol{\mu}, \lambda$)
  $I \leftarrow N$
  repeat
    for $i \in \{I, \ldots, N\}$ **in parallel do**
      // only consider clusters that contain more than n_i
      $z_i \leftarrow \underset{k \in \{1,\ldots,|\boldsymbol{\mu}|+1\}}{\arg\max} \begin{cases} n_i^T \mu_k & k \leq |\boldsymbol{\mu}| \text{ and } |\mathcal{I}_k \setminus z_i| > 0 \\ \lambda + 1 & k = |\boldsymbol{\mu}| + 1 \end{cases}$
      if $z_i = |\boldsymbol{\mu}| + 1$ or **any**$(|\mathcal{I}_k| = 0)$ **then**
        // obtain the first index of cluster number change
        atomic: $I = \min(I, i)$
      end if
    end for
    if $I < N$ **then**
      if $z_I = |\boldsymbol{\mu}| + 1$ **then**
        $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \cup \{\mu_{|\boldsymbol{\mu}|+1} \leftarrow n_I\}$ // add cluster $K+1$ and initialize to n_I
      else
        $\boldsymbol{\mu} \leftarrow \boldsymbol{\mu} \setminus \boldsymbol{\mu}_{k:|\mathcal{I}_k|=0}$ // remove empty cluster
      end if
    end if
  until $I = N$
  return $\{z_i\}_{i=1}^N, \boldsymbol{\mu}$
end function

Algorithm 14: DP-vMF-means optimistic iterated restarts (OIR) label assignments algorithm. The min-reduction which is key to this algorithm can be implemented efficiently on GPUs using built in atomic functions. The creation and removal of clusters happens on CPU.

silhouette score, indicating well-separated, concentrated clusters. Again, spkm does not reach the same clustering performance even for $K = K_{\mathrm{T}}$ for the same aforementioned reasons.

NYU v2 depth dataset In this experiment, the DP-vMF-means and spkm algorithms were compared on the NYU v2 RGB-D dataset [173]. Surface normals were extracted from the depth images [112] and preprocessed with total variation smoothing [202]. We quantify the clustering quality in terms of the average silhouette score over the clusterings of the 1449 scenes of the NYU v2 depth dataset. Since we do not possess the true scene labeling, we use the silhouette quality metric as a proxy for the NMI metric; this was motivated by the agreement between the maxima of the NMI and silhouette scores in the previous synthetic experiment.

Across the whole NYU v2 dataset, the DP-vMF-means algorithm achieves the highest average silhouette score of 0.75 for $\phi_\lambda^\star = 100°$ as depicted in Fig. 4.19. The histogram over the number of inferred clusters by DP-vMF-means for $\phi_\lambda^\star$ indicates the varying com-

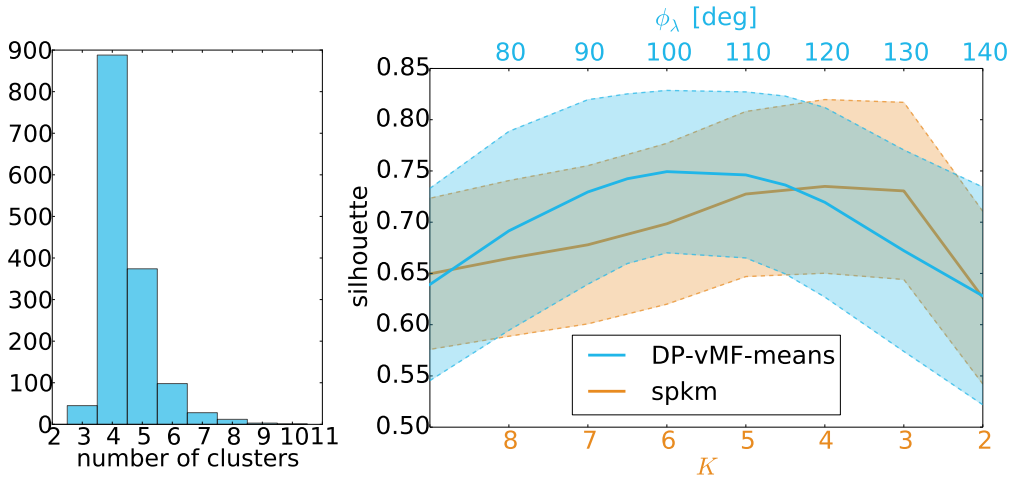


Figure 4.19: Histogram over the number of clusters for optimal ϕ_λ found by DP-vMF-means (left) and silhouette values for DP-vMF-means and spkm (right) across the whole NYU dataset [173]. DP-vMF-means achieves higher overall silhouette scores (right) because it can adapt the number of clusters to the observed data as the cluster number statistics show (left).

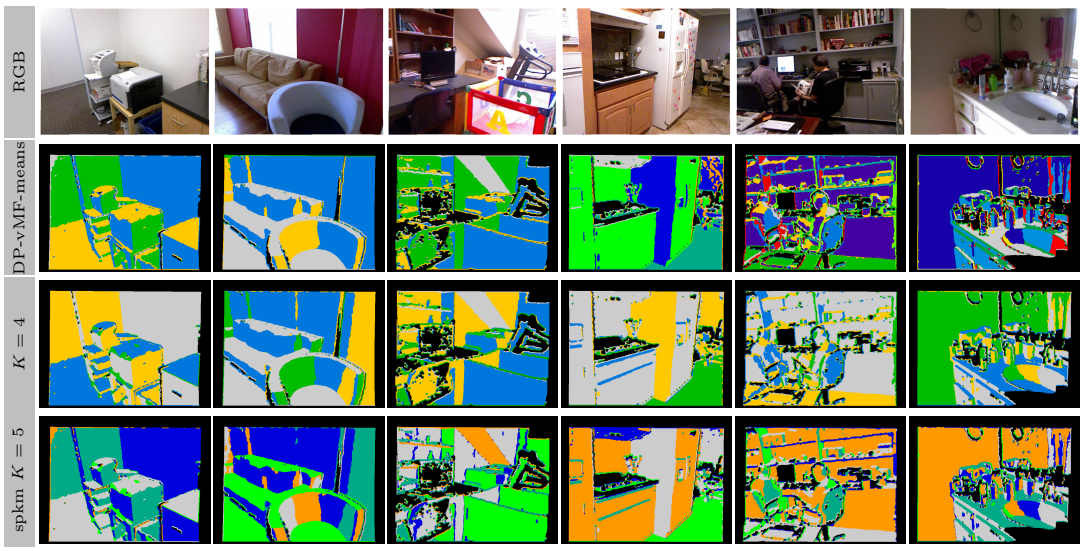


Figure 4.20: Directional segmentation of scenes from the NYU v2 RGB-D dataset [173] as implied by surface normal clusters. The complexity of the scenes increases from left to right as can be observed from the RGB images in the top row. The second row shows the clustering inferred using DP-vMF-means with $\phi_\lambda = 100°$ while the third and fourth show the spherical k-means results for comparison. Black denotes missing data due to sensor limitations. Note that DP-vMF-means adapts the number of clusters to the complexity of the scene.

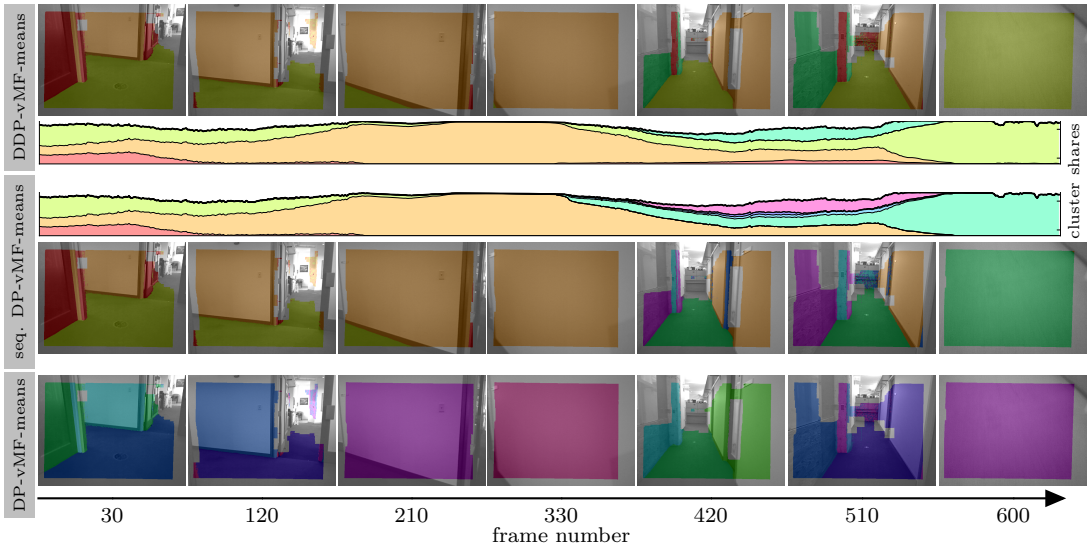


Figure 4.21: Clustering of a surface normal stream recorded when walking a 90° turn in an office environment. We depict key-frames color-coded with the implied surface-normal clustering for three different clustering algorithms. The plots in the second and third row depict the percentage of normals associated to the respective cluster for the DDP-vMF-means as well as sequential DP-vMF-means. Note that the clustering obtained via the DDP-vMF-means algorithm is consistent across the whole run as opposed to the other algorithms.

plexity of the scenes ranging from three to eleven. The clear peak at $K = 4$ coincides with the highest silhouette score for spkm (0.73) and explains the only slightly lower silhouette score of spkm: most scenes in the dataset exhibit four primary directions.

Figure 4.20 shows a qualitative comparison of the scene segmentation implied by the clustering of surface normals. In comparison to spkm, the DP-vMF-means clustering results show the ability of the algorithm to adapt the number of clusters to the scene at hand. If the right number of clusters is selected for the spkm clustering, the results have similar quality; however, the number of clusters is generally not known a priori and varies across scenes. This demonstrates two major advantages of DP-vMF-means over spkm: (1) DP-vMF-means is less sensitive to the parameter setting (see Fig, 4.18, top left) and (2) it is easier to choose ϕ_λ than K since it intuitively corresponds to the maximum angular radius of a cluster, which can be gauged from the type of data and its noise characteristics. For this experiment $\phi_\lambda = 100°$ is justified by the typical Manhattan structure [49] of the indoor environment plus 10° to account for sensor noise.

Real-time Sequential Directional Segmentation In fields such as mobile robotics or augmented reality, it is uncommon to observe just a single RGB-D frame of a scene; more typically, the sensor will observe a temporal sequence of frames. The following exper-

iment demonstrates the temporally consistent clustering capability of the DDP-vMF-means algorithm on surface normals extracted from a sequence of depth images recorded in an indoor environment. Each frame is preprocessed in 11ms using edge-preserving smoothing with a hybrid CPU-GPU guided filter [107]. Choosing the guidance image equivalent to the input image results in an $O(N)$ filter that provides similar smoothing results to a bilateral filter [235], which is $O(Nr^2)$, where N is the number of pixels and r the filter radius. The independence of the guided filters runtime from the filter radius allows us to use a filter window of 20×20 pixels.

We compare against the ad-hoc approaches of clustering on a frame-by-frame basis using DP-vMF-means, both with and without initializing the algorithm from the previous frame's clusters. The former is referred to as sequential DP-vMF-means (sDP-vMF-means). sDP-vMF-means achieves a greedy frame-to-frame label consistency, but, unlike DDP-vMF-means, it cannot reinstantiate previous clusters after multiframe lapses. Motivated by the DP-vMF-means evaluation, all algorithms were run with $\phi_\lambda = 100°$. For DDP-vMF-means $\beta = 10^5$ and $Q = \frac{\lambda}{400}$.

The differences in labeling consistency can be observed in rows two and three of Fig. 4.21, which shows the percentage of normals associated with a specific cluster. While DDP-vMF-means is temporally consistent and reinstantiates the lime-green and red clusters, observed in the first half of the run, DP-vMF-means erroneously creates new clusters. We do not depict the percentages of surface normals associated with the clusters for the batch DP-vMF-means algorithm, since the there is no label consistency between time-steps as can be observed in the last row of Fig. 4.21.

The average run-time per frame was 28.4 ms for batch DP-vMF-means, 12.8 ms for sDP-vMF-means, 20.4 ms for DDP-vMF-means, and 13.6 ms for spkm with $K = 5$. The increased running time of batch DP-vMF-means is a result of clustering each batch of surface normals in isolation; optimistic iterated restarted label assignment needs several restarts to assign labels to all surface normals. By initializing the clusters from a previous frame, sDP-vMF-means only incurs labeling restarts if a new cluster is observed, and hence has significantly lower run time. DDP-vMF-means is slightly slower than sDP-vMF-means since it is keeping track of both observed and unobserved clusters.

■ 4.5 Discussion

In the first part of this chapter we introduce the DP-TGMM, a Dirichlet process mixture model over Gaussian distributions in multiple tangent spaces to the unit sphere in $\mathbb{R}^D$. Aimed at modeling directional data, this Bayesian nonparametric model not only adapts to the complexity of the data but also describes anisotropic distributions on the sphere. Experiments on synthetic data demonstrate that the proposed DP-TGMM is more expressive in describing anisotropic directional data than other commonly-used approaches. Moreover, we have shown the scalability and effectiveness of the inference algorithm as well as the applicability and versatility of the model on batches of 300k

real-world 3D surface normals and on 20-dimensional semantic word-vectors for 41k English words. Code for the proposed sub-cluster-based inference algorithm can be found at http://people.csail.mit.edu/jstraub/.

In the second half of this chapter, we have derived two novel k-means-like algorithms for efficient batch and streaming clustering of data on the unit hypersphere by taking the small-variance asymptotic limit of the Bayesian nonparametric DP-vMF and DDP-vMF mixture models. The performance and flexibility of DP-vMF-means was demonstrated on both synthetic data and the NYU v2 RGB-D dataset. For DDP-vMF-means, optimistic iterated restarts parallelized label assignments, enable real-time temporally consistent clustering of batches of 300k surface normals collected at 30 Hz from a RGB-D camera. Implementations of the DP-vMF-means and the DDP-vMF-means algorithms can be found at http://people.csail.mit.edu/jstraub/.

Future work could investigate the extension of the DP-TGMM model to other Riemannian manifolds. The approach of modeling data in tangent spaces to a manifold is quite universal as long as a manifold is equipped with an exp and log map as well as a mechanism for parallel transport to a common location on the manifold (for $\mathbb{S}^{D-1}$ this amounts to rotations). Potential manifolds for such extension would be the Stiefel manifold, the manifold of rotations $\mathbb{SO}(3)$ and rigid-body transformations $\mathbb{SE}(3)$.

Another promising research direction is embedding DP-TGMM and/or DP-vMF-MM in a hierarchical structure (akin to the Hierarchical DP-GMM for $\mathbb{R}^D$-valued data) to allow information sharing between batches of data in applications such as protein backbone configuration modeling or hierarchical scene segmentation across a corpus of indoor scenes.

A third avenue of further exploration would be how to incorporate the notion that Stata Center World segmentations are mostly spatially smooth (whole planes should belong to the same Stata Center World segment). There is some interesting work on imposing spatial smoothness on the segmentation via a Markov Random Field (MRF) defined via local neighborhood information [182] that we will utilize to some degree in Chapter 5. However, it is unclear how the addition of a MRF changes the low-variance analysis of Sec. 4.4.3.

■ 4.6 Acknowledgments

This part of the overall research was conducted in collaboration with Jason Chang, Trevor Campbell, Oren Freifeld, Jonathan P. How, John J. Leonard as well as John W. Fisher III and published in [222, 224]. Specifically, Trevor Campbell's prior expertise with low-variance analysis helped make fast progress in deriving the DDP-vMF-means algorithm. He also formally proved the Laplace approximation for the sphere while I was busy working on the GPU-based DDP-vMF-means implementation. Jason Chang's input when deriving the DP-TGMM inference was crucial for the fast progress on the DP-TGMM project.

Chapter 5

Nonparametric Directional Perception Systems

In this final chapter we focus on equipping perception systems with direction-awareness, i.e. the ability to infer and utilize a directional segmentation of their environment. We explore the idea that directional models such as introduced in Chapters 3 and 4, provide useful information for general 3D perception systems because they capture structural regularities of the environment. Recognizing and utilizing such regularities should help localization, mapping and higher-level inference about an environment. In this chapter we focus on showing the impact of direction-awareness onto mapping and localization, two of the most fundamental perception tasks.

Recall the categorical SLAM problem from the introductory chapter. Under a directional scene segmentation we term this problem direction-aware SLAM. Specifically, direction-aware SLAM amounts to reasoning about the joint distribution of a world map m, the trajectory of a perception system T, and a directional segmentation z of the environment given a set of observations x:

$$p(m, T, z \mid x) \qquad \text{“direction-aware SLAM”} . \tag{5.1}$$

To approach this joint inference problem we follow the approach of iteratively reasoning about the conditional distributions, a common strategy in inference algorithms such as Gibbs sampling and expectation maximization. The conditional distributions are

$$p(m \mid T, z, x) \qquad \text{“direction-aware mapping”} , \tag{5.2}$$
$$p(T \mid m, z, x) \qquad \text{“direction-aware localization”} , \tag{5.3}$$
$$p(z \mid m, T, x) \qquad \text{“directional segmentation”} . \tag{5.4}$$

That the full joint direction-aware SLAM is a hard problem can be gleaned from the fact that the problem of SLAM itself has spawned decades of research [36, 148]. Therefore we approach incorporating direction-awareness into the SLAM problem in two steps: we first focus on direction-aware localization in the form of global point cloud alignment before introducing the first approach to solving the full direction-aware SLAM problem.

In the first half of this chapter we focus on the subproblem of direction-aware point cloud alignment. In the aforementioned framework this amounts to inference on the

directional segmentation of the two point clouds, i.e. reasoning about Eq. (5.4), and inference about the relative pose between the point clouds, i.e. Eq. (5.3). Specifically, we seek to find the optimal pose to align two point clouds that have been segmented in terms of their directional and spatial distributions via Dirichlet process mixture models of von-Mises-Fisher and Gaussian distributions. Operating over these nonparametric mixture models instead of the point densities directly makes the approach tractable. By exploiting the fact that surface normal distributions are invariant to translation, we decompose the alignment problem into first finding the rotation R by aligning the surface normal distributions and then, given R, finding the translation by aligning the point distributions. Both optimization problems are solved optimally using branch-and-bound search. Contributions include the use of Bayesian nonparametric density estimates for the alignment task, a novel, approximately uniform tessellation of the rotation space and branch-and-bound convergence guarantees for this new tessellation. We demonstrate that the novel tessellation improves branch-and-bound exploration efficiency and thus runtime of the algorithm. Comparisons to related algorithms show the advantages of the decomposed approach which achieves better alignment with significant speedups over competing algorithms.

In the second half of the chapter we turn to the full direction-aware SLAM problem and introduce the first realtime-capable system that performs iterated inference on all three posteriors in Equations (5.2), (5.3), and (5.4). Based on a sparse collection of surfels to represent the map, the system performs joint inference on map, camera trajectory, and nonparametric Stata Center World segmentation. Specifically, we establish an explicit connection between the scene-wide directional segmentation and local surface properties which yields direction-aware mapping as defined in Eq. (5.2). Furthermore, the directional segmentation is used to guide observation selection for camera pose estimation as suggested by Eq. (5.3). This leads to improved 3D reconstruction and improved and more efficient camera tracking. The proposed system architecture is the first to demonstrate real-time performance while running sampling-based inference over the map and segmentation distributions. This allows the computation of expectations over arbitrary functions of the random variables such as expectations, variances and uncertainty of surfel locations, orientations, and color. It opens up further avenues of investigation with more complex models (such as the Bayesian nonparametric models utilized in this work) for which inference is not directly possible using mode-seeking methods such as maximum a posterior estimation, the standard tool of 3D reconstruction systems.

■ 5.1 Global Point Cloud Alignment using Bayesian Nonparametric Mixtures

Point cloud alignment is a fundamental problem for many applications in robotics [110, 161] and computer vision [177, 212, 247]. Finding the global transformation is generally hard: point-to-point correspondences typically do not exist, the point clouds might only

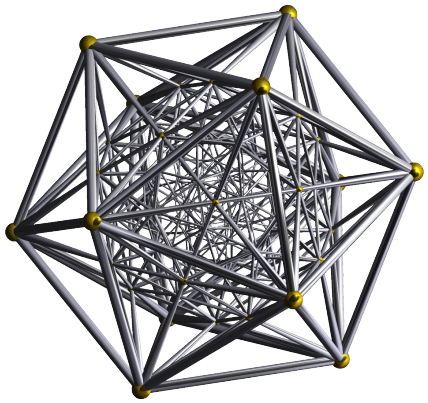


Figure 5.1: A 3D projection of the 600-cell [243]—a 4D object tessellating the space of rotations for the proposed branch and bound approach to point cloud alignment.

have partial overlap, and the underlying objects themselves are often nonconvex, leading to a potentially large number of alignment local minima. As such, popular local optimization techniques suffice only in circumstances with small true relative transformations and large overlap, such as in dense 3D incremental mapping [110, 177, 247]. Solving the alignment problem for large unknown relative transformations and small point cloud overlap calls for a global approach. Example applications are the loop-closure problem in SLAM [30] and the model-based detection of objects in 3D scenes [125].

Motivated by the observation that surface normal distributions are translation invariant [116] and straightforward to compute [168, 221], we develop a two-stage branch and bound (BB) [144, 145] optimization algorithm for point cloud alignment. We model the surface normal distribution of each point cloud as a Dirichlet process (DP) [70, 233] von-Mises-Fisher (vMF) [74] mixture [222] (DP-vMF-MM). To find the optimal rotation, we minimize the L^2 distance between the distributions over the space of 3D rotations. We develop a novel refinable tessellation consisting of 4D tetrahedra (see Fig. 5.1) which more uniformly approximates rotation space and is more efficient than the common axis-angle tessellation [103, 150] during BB optimization. Given the optimal rotation and modeling the two point distributions as DP Gaussian mixtures [8, 42] (DP-GMM), we obtain the optimal translation similarly via BB over the space of 3D translations. The use of mixture models circumvents discretization artifacts, while still permitting efficient optimization. In addition to algorithmic developments, we provide corresponding theoretical bounds on the convergence of both BB stages, linking the quality of the derived rotation and translation estimates to the depth of the search tree and thus the computation time of the algorithm. Experiments on real data corroborate the theory, and demonstrate the accuracy and efficiency of BB as well as its robustness to real-world conditions, such as partial overlap, high noise, and large relative transformations.

■ 5.1.1 Related Work

Local Methods There exists a variety of approaches for local point cloud alignment [38, 212]. Iterative closest point (ICP) [21], the most common of these, alternates between associating the points in both clouds and updating the relative transformation estimate under those associations. There are many variants of ICP [207] differing in their choice of cost function, how correspondences are established, and how the objective is optimized at each iteration. An alternative developed by Magnusson et al. [161] relies on the normal distribution transform (NDT) [22], which represents the density of the scans as a structured GMM. This approach has been shown to be more robust than ICP in certain cases [162]. Approaches that use correlation of kernel density estimates (KDE) for alignment [238] or GMMs [122] use a similar representation as the proposed approach. KDE-based methods scale poorly with the number of points. In contrast, we use mixture models inferred by nonparametric clustering algorithms (DP-means [142] and DP-vMF-means [222]). This allows adaptive compression of the data, enabling the processing of large noisy point clouds (see Sec. 5.1.5 for experiments with more than 300k points). Straub et al. propose two local rotational alignment algorithms [221, 222] that, similarly to the proposed approach, utilize surface normal distributions modeled as vMF mixtures. Common to all local methods is the assumption of an initialization close to the true transformation and significant overlap between the two point clouds. If either of these assumptions are violated, local methods become unreliable as they tend to get stuck in suboptimal local minima [162, 207, 212].

Global Methods Global point cloud alignment algorithms make no prior assumptions about the relative transformation or amount of overlap. For those reasons global algorithms, such as the proposed one, are often used to initialize local methods. 3D-surface-feature-based algorithms [3, 84, 125, 208] involve extracting local features, obtaining matches between features in the two point clouds, and finally estimating the relative pose using RANSAC [73] or other robust estimators [117]. Recently, Albarelli et al. [4] have proposed an alignment algorithm that is resilient to large outlier ratios without resorting to robust estimators. Another recent featured-based approach by Zhou et al. [259] uses correspondences established once before optimization and is robust to large fractions of wrong correspondences. Though popular, feature-based algorithms are generally vulnerable to large fractions of incorrect feature matches, as well as repetitive scene elements and textures. A second class of approaches, including the proposed approach, rely on statistical properties of the two point clouds. Makadia et al. [163] separate rotational and translational alignment. Rotation is obtained by maximizing the convolution of the peaks of the extended Gaussian images (EGI) [116] of the two surface normal sets. This search is performed using the spherical Fourier Transform [64]. After rotational alignment, the translation is found similarly via the fast Fourier Transform. The use of histogram-based density estimates for the surface normal and point distributions introduces discretization artifacts. Additionally, the sole use of the peaks of the EGI makes the method vulnerable to noise in the data.

For the alignment of 2D scans, Weiss et al. [245] and Bosse et al. [30] follow a similar convolution-based approach. Early work by Li, Hartley and Kahl [103, 150] on BB for point cloud alignment used the axis-angle (AA) representation of rotations. A drawback of this approach is that a uniform AA tessellation does not lead to a uniform tessellation in rotation space (see Sec. 5.1.3). As we show in Sec. 5.1.5, this leads to less efficient BB search. Parra Bustos et al. [185] propose improved bounds for rotational alignment by reasoning carefully about the geometry of the AA tessellation. GoICP [256] nests BB over translations inside BB over rotations and utilizes ICP internally to improve the BB bounds. GOGMA [37] uses a similar approach, but replaces the objective with a convolution of GMMs. Both GoICP and GOGMA involve BB over the joint 6-dimensional rotation and translation space; since the complexity of BB is exponential in the dimension, these methods are relatively computationally expensive (see results Fig. 5.14).

■ 5.1.2 The Point Cloud Alignment Problem

Our approach to point cloud alignment relies on the fact that surface normal distributions are invariant to translation [116] and easily computed [168, 221], allowing us to isolate the effects of rotation. Thus we decompose the task of finding the relative transformation into first finding the rotation using only the surface normal distribution, and then obtaining the translation given the optimal rotation.

Let a noisy sampling of a surface S be described by the joint point and surface normal density $p(x, n)$, where $x \in \mathbb{R}^3$ and $n \in \mathbb{S}^2$. A sensor observes two independent samples from this model: one from $p_1(x, n) = p(x, n)$, and one from $p_2(x, n) = p(R^{\star T}(x - t^\star), R^{\star T} n)$ differing in an unknown rotation $R^\star \in \mathbb{SO}(3)$ (see Sec 2.7.1) and translation $t^\star \in \mathbb{R}^3$. Given these samples, we model the marginal point densities $\hat{p}_1(x)$, $\hat{p}_2(x)$ using the posterior of a Dirichlet process Gaussian mixture (DP-GMM) [8], and model the marginal surface normal densities $\hat{p}_1(n)$, $\hat{p}_2(n)$ using the posterior of a Dirichlet process von Mises-Fisher mixture (DP-vMF-MM) [15, 222]. Note that the formulation using Dirichlet process mixture models admits arbitrarily accurate estimates of a large class of noisy surface densities (Theorem 2.2 in [60]). Given the density estimates, we formulate the problem of finding the relative transformation as

$$\begin{aligned} \hat{q} &= \arg\max_{q \in \mathbb{S}^3} \int_{\mathbb{S}^2} \hat{p}_1(n) \hat{p}_2(q \circ n) \mathrm{d}n \\ \hat{t} &= \arg\max_{t \in \mathbb{R}^3} \int_{\mathbb{R}^3} \hat{p}_1(x) \hat{p}_2(\hat{q} \circ x + t) \mathrm{d}x, \end{aligned} \tag{5.5}$$

where we represent rotations using unit quaternions in $\mathbb{S}^3$, the 4D sphere [115], and where $q \circ n$ denotes the rotation of a surface normal n by a unit quaternion q (see Sec. 2.7.4). Eq. (5.5) minimizes the L_2 metric via maximization of the convolution, which has been shown to be robust in practice [122]. The equivalence between L_2

metric minimization and maximization of the convolution follows for $\hat{q}$ from:

$$\begin{aligned}\hat{q} &= \arg\min_{q\in\mathbb{S}^3} \|\hat{p}_1(n) - \hat{p}_2(q\circ n)\|_2^2 \\ &= \arg\min_{q\in\mathbb{S}^3} \int_{\mathbb{S}^2} \hat{p}_1^2(n) - 2\hat{p}_1(n)\hat{p}_2(q\circ n) + \hat{p}_2^2(q\circ n)\,\mathrm{d}n \\ &= \arg\max_{q\in\mathbb{S}^3} \int \hat{p}_1(n)\hat{p}_2(q\circ n)\,\mathrm{d}n\,.\end{aligned} \tag{5.6}$$

The same is true for the optimization of $\hat{t}$. The L_2 norm minimization is a common approach for Gaussian MMs [37, 122, 238] but to our knowledge has not been explored for vMF-MMs, nor for Bayesian nonparametric DP mixtures. In fact, the use of DP mixtures is critical, as it allows the automatic selection of a parsimonious, but accurate, representation of the point cloud data. This improves upon both kernel density estimates [238], which are highly flexible but make optimizing Eq. (5.5) intractable for large RGB-D datasets, and fixed-sized GMMs [37, 122], which require heuristic model selection and may not be rich enough to capture complex scene geometry. While exact posterior predictive DP-MM densities cannot be computed tractably, excellent estimation algorithms are available, which we use in this work [142, 222].

Both optimization problems in Eq. (5.5) are nonconcave maximizations. Considering the geometry of the problem, we expect many local maxima, rendering typical gradient-based methods ineffective. This motivates the use of a global approach. We develop a two-step BB procedure [144, 145] that first searches over $\mathbb{S}^3$ for the optimal rotation $\hat{q}$, and then over $\mathbb{R}^3$ for the optimal translation $\hat{t}$. As BB may return multiple optimal rotations (e.g. if the scene has rotational symmetry) we estimate the optimal translation under each of those rotations, and return the joint transformation with the highest translational cost lower bound. Note that while $\hat{q}, \hat{t}$ is not necessarily the optimal transformation under rotation and translation *jointly*, the decoupling of rotation and translation we propose reduces the computational complexity of BB significantly. This is because the complexity scales exponentially in the search space dimension; optimizing over two 3D spaces ($\mathbb{R}^3$ and $\mathbb{S}^3$) separately is significantly less costly than over the joint 6D space.

BB requires three major components: (1) a tessellation method for covering the optimization domain with subsets (see Sec. 5.1.3 and 5.1.4); (2) a branch/refinement procedure for subdividing any subset into smaller subsets (see Sec. 5.1.3 and 5.1.4); and (3) upper and lower bounds of the maximum objective on each subset to be used for pruning (see Sec. 5.1.3 and 5.1.4). As shown in Algorithm 15, BB proceeds by bounding the optimal objective in each subset, pruning those which cannot contain the maximum, subdividing the best subset to refine the bounds, and iterating. Note that in this work we select the node with the highest upper bound for subdivision. More nuanced strategies have been developed and could also be utilized [118, 145].

$\mathcal{T} \leftarrow \textsc{Tessellation}(\Omega)$
repeat
  $\mathcal{Q}^\star \leftarrow \arg\max_{\mathcal{Q}\in\mathcal{T}} \textsc{UpperBound}(\mathcal{Q})$
  $\mathcal{T} \leftarrow (\mathcal{T} \setminus \mathcal{Q}^\star) \cup \textsc{Branch}(\mathcal{Q}^\star)$
  $l = \max_{\mathcal{Q}\in\mathcal{T}} \textsc{LowerBound}(\mathcal{Q})$
  Prune any $\mathcal{Q}$ with $\textsc{UpperBound}(\mathcal{Q}) < l$ from $\mathcal{T}$
  $u = \max_{\mathcal{Q}\in\mathcal{T}} \textsc{UpperBound}(\mathcal{Q})$
until $u - l < \epsilon$ **return** $\mathcal{T}$

Algorithm 15: Branch and Bound on a space Ω. We first run BB over the space of rotations ($\Omega = \mathbb{S}^3$) and then over the space of translations ($\Omega = \mathbb{R}^3$).

■ 5.1.3 von-Mises-Fisher Mixture Rotational Alignment

We model the distributions of surface normals n as von-Mises-Fisher [74] mixture models (vMF-MM) with means $\{\mu_{ik}\}_{k=1}^{K_i}$, concentrations $\{\tau_{ik}\}_{k=1}^{K_i}$, and positive weights $\{\pi_{ik}\}_{k=1}^{K_i}$, $\sum_{k=1}^{K_i} \pi_{ik} = 1$, for $i \in \{1, 2\}$, with density

$$\hat{p}_i(n) = \sum_{k=1}^{K_i} \pi_{ik} C_{ik} \exp\left(\tau_{ik} \mu_{ik}^T n\right) \quad C_{ik} \triangleq \frac{\tau_{ik}}{4\pi \sinh(\tau_{ik})}. \tag{5.7}$$

See Sec. 2.6.3 for more details about the vMF distribution. While there are many techniques for inferring vMF-MMs [14, 61, 222], we use the nonparametric method described in [222] and Sec. 4.4.3 that infers an appropriate K_i automatically. This model simultaneously circumvents discretizing the sphere, thus avoiding the artifacts associated with discretization, and is more expressive than a collection of delta functions on $\mathbb{S}^2$ as used by [163]. The rotational alignment problem from Eq. (5.5) with this model becomes

$$\begin{aligned} \hat{q} &= \arg\max_{q\in\mathbb{S}^3} \sum_{k,k'} \frac{D_{kk'}}{2\pi} \int_{\mathbb{S}^2} \exp\left((\tau_{1k}\mu_{1k} + \tau_{2k'} q \circ \mu_{2k'})^T n\right) \mathrm{d}n \\ D_{kk'} &\triangleq (2\pi)\pi_{1k}\pi_{2k'} C_{1k} C_{2k'} \,. \end{aligned} \tag{5.8}$$

We obtain the following objective function by noting that the integral is the normalization constant of a vMF density with concentration $z_{kk'}(q) \triangleq \|\tau_{1k}\mu_{1k} + \tau_{2k'} q \circ \mu_{2k'}\|$:

$$\begin{aligned} \hat{q} = \arg\max_{q\in\mathbb{S}^3} & \sum_{k,k'} D_{kk'} f(z_{kk'}(q)) \\ & \text{where } f(z) \triangleq 2\sinh(z) z^{-1} = \left(e^z - e^{-z}\right) z^{-1} \,. \end{aligned} \tag{5.9}$$

We now provide a novel refinable tessellation of $\mathbb{S}^3$ in Sec. 5.1.3 and upper/lower objective function bounds in each of its subsets in Sec. 5.1.3 for use with BB in optimizing Eq. (5.9). Finally, given the proposed tessellation and objective bounds, we provide convergence guarantees for the rotational alignment optimization algorithm in Sec. 5.1.3.

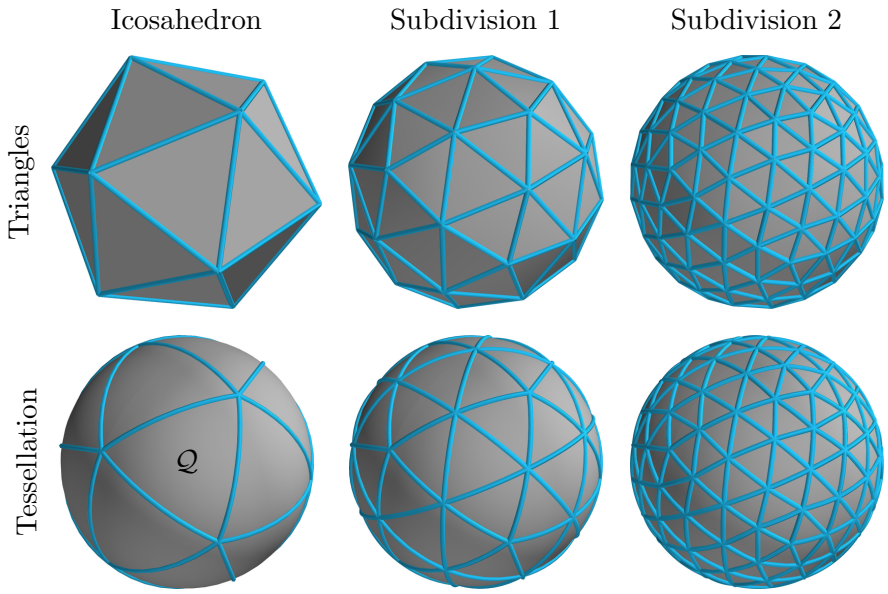


Figure 5.2: Tessellation of $\mathbb{S}^2$ via iterated triangle subdivision. The tessellation of $\mathbb{S}^3$ follows the same principles, but with 4D tetrahedra instead of 3D triangles. Note the uniformity of the tessellation.

Cover and Refinement of the Rotation Space $\mathbb{S}^3$

In this section, we develop a novel tessellation scheme for the space of rotations, and show how to refine it in a way that guarantees convergence of BB for rotational alignment. We follow a similar approach to the geodesic grid tessellation of a sphere in 3D (i.e. $\mathbb{S}^2$): as depicted in Fig. 5.2, starting from an icosahedron, each of the 20 triangular faces is subdivided into four triangles of equal size. Then the newly created triangle corners are normalized to unit length, projecting them onto the unit sphere. Note that in this last step, the formerly equally-sized triangles become slightly distorted, as the middle triangle becomes larger than the other three triangles surrounding it. For this reason, the resulting tiling of the sphere is only approximately uniform.

In four dimensions we instead start with the analogue of the icosahedron, the *600-cell* [50] (shown in Fig. 5.1), an object composed of 600 4D tetrahedra. We first generate its 120 vertices with the following algorithm [50, pp. 402–403]. Let $\phi = \frac{1}{2}\left(1+\sqrt{5}\right)$. Then the (unnormalized) 120 vertices of the 600-cell in 4D are

- even permutations of $\left[\pm\phi, \pm 1, \pm\phi^{-1}, 0\right]^T$ (96 vertices),
- all permutations of $\left[\pm 2, 0, 0, 0\right]^T$ (8 vertices), and
- all permutations of $\left[\pm 1, \pm 1, \pm 1, \pm 1\right]^T$ (16 vertices).

We then scale the 120 vertices to each have unit norm, representing a 3D quaternion

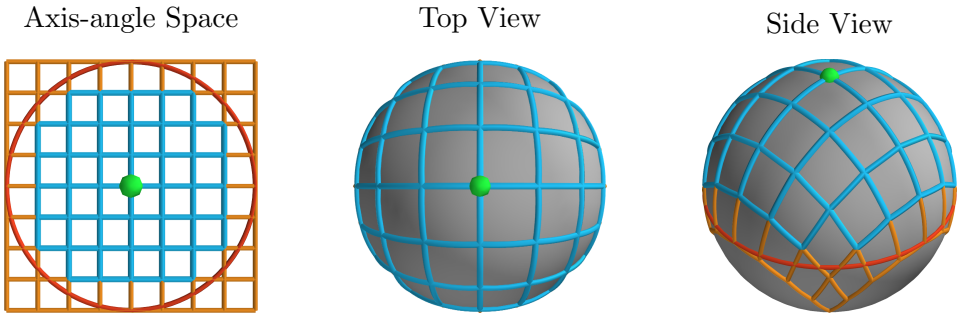


Figure 5.3: Tessellation of $\mathbb{S}^2$ via uniform tessellation in the axis-angle (AA) space. The axis-angle tessellation of $\mathbb{S}^3$ follows the same principle and incurs similar distortion. Note that orange tiles contain surface area on the lower half-sphere, so parts of the rotation space are covered twice, making BB inefficient.

rotation. Next, noting that the angle between any two connected tetrahedra vertices is 36°, we iterate over all $\binom{120}{4}$ possible choices of 4 vertices, and only select those 600 tetrahedra for which all pairwise angles are 36°. This collection of tetrahedra, which are "flat" in 4D analogous to triangles in 3D, comprises a 4D object which approximates the 4D sphere, $\mathbb{S}^3$. Then, since the set of all quaternion rotations may be represented by any hemisphere of $\mathbb{S}^3$ (q and $-q$ describe the same rotation), we define the "north" vector to be $[0,\, 0,\, 0,\, 1]^T \in \mathbb{S}^3$, and only keep those tetrahedra for which at least one vertex has angle $< 90°$ to the north vector. This results in 330 tetrahedra that approximate the 4D upper hemisphere in $\mathbb{S}^3$, i.e. the space of quaternion rotations. Note that this construction procedure is the same for *any* optimization on $\mathbb{S}^3$, so it can be performed once and the result may be stored for efficiency.

One major advantage of the proposed $\mathbb{S}^3$ tessellation is that it is exactly uniform at the 0th level and approximately uniform for deeper subdivision levels (Fig. 5.2 shows the analogous near-uniformity for $\mathbb{S}^2$). This generally tightens bounds employed by BB, leading to more efficient optimization. Another advantage is that this tessellation is a near-exact covering of the upper hemisphere of $\mathbb{S}^3$. Only 7% of rotation space is covered twice, meaning that BB wastes little time with duplicate searching. The widely employed AA-tessellation scheme [103, 150, 185, 256], in contrast, uniformly tessellates a cube enclosing the axis-angle space, a 3D sphere with radius π, and maps that tessellation onto the rotation space. There are two major issues with the AA approach. First, it covers 46% of rotation space twice [103, 150] (see Fig. 5.3 for a depiction of the analogous scenario for $\mathbb{S}^2$). Second, it does not lead to uniform tessellation in rotation space. The reason for this is that the Euclidean metric in AA space is a poor approximation of the distance on the rotation manifold [150]. Fig. 5.3 shows the AA tessellation analog for $\mathbb{S}^2$, highlighting its significant non-uniformity. We empirically find that the $\mathbb{S}^3$ tessellation leads to more efficient BB optimization than the AA tessellation (see results in Figs. 5.9 and 5.10).

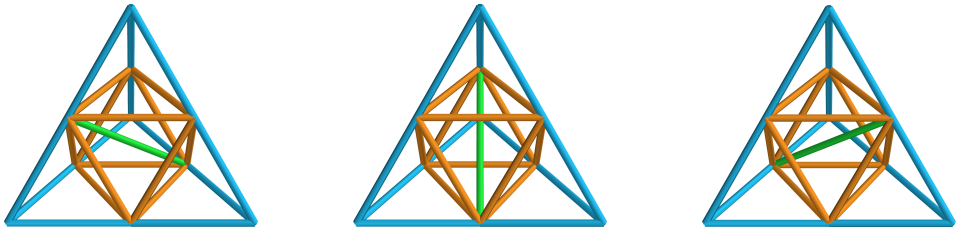


Figure 5.4: The three subdivision patterns—due to the choice of the green edge—of a tetrahedron displayed in 3D. Colors designate different edge types: corner edges (blue) from an edge midpoint to a vertex; tie edges (orange) between two edge midpoints, running along a tetrahedron face; and skew edges (green) between two edge midpoints, running through the inside of the tetrahedron. The internal skew edge (green) is chosen to minimize distortion.

We now discuss two properties of the proposed tessellation required by BB: (1) that it is a *cover* for the upper hemisphere of $\mathbb{S}^3$, guaranteeing that BB will search the whole space of rotations; and (2) that it is *refinable*, so BB can search promising subsets in increasingly more detail.

Cover Let the four vertices of a single tetrahedron from our approximation of $\mathbb{S}^3$ be denoted $q_j \in \mathbb{S}^3$, $j \in \{1, \dots, 4\}$. Then, stacking them horizontally into a matrix $Q \in \mathbb{R}^{4\times 4}$, the projection $\mathcal{Q}$ of the tetrahedron onto $\mathbb{S}^3$ is:

$$\mathcal{Q} = \left\{ q \in \mathbb{R}^4 : \|q\| = 1,\ q = Q\alpha,\ \alpha \in \mathbb{R}^4_+ \right\}. \tag{5.10}$$

In other words, $\mathcal{Q}$ is the set of unit quaternions found by extending the (flat in 4D) tetrahedron to the unit sphere using rays from the origin. For $\mathbb{S}^2$, this is displayed in the second row of Fig. 5.2. The proposed set of 330 projected tetrahedra $\mathcal{Q}$ forms a cover of the upper hemisphere of $\mathbb{S}^3$.

Refinement Next, we require a method of subdividing any $\mathcal{Q}$ in the cover. Similar to the triangle subdivision method for refining the tessellation of $\mathbb{S}^2$, each 4D tetrahedron can be subdivided into eight smaller tetrahedra [152] as depicted in Fig. C.1. The resulting six new vertices for the subdivided tetrahedra are scaled to unit length. As we have the freedom to choose one of three internal edges for subdivision, we choose the internal edge with the minimum angle between its unit-norm vertices. In other words, denoting ξ_k for $k \in \{1, 2, 3\}$ to be the three internal dot products,

$$k^\star = \underset{k\in\{1,2,3\}}{\arg\max}\ \xi_k. \tag{5.11}$$

This process forms the eight new subdivided cover elements $\mathcal{Q}$. For example, if q_i, $i \in \{1, \dots, 4\}$ are the vertices of $\mathcal{Q}$, then one of the subdivisions (corresponding to one

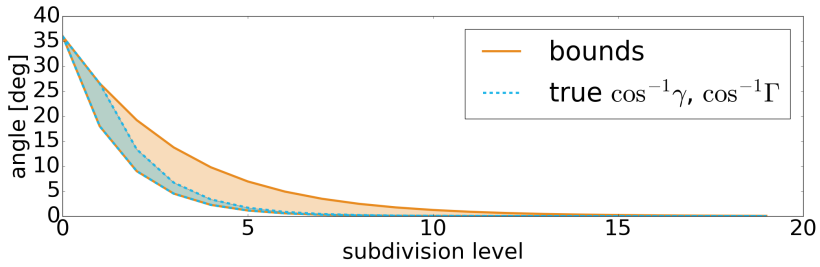


Figure 5.5: The bounds in Eq. (5.13) compared to the true min & max angles between tetrahedron vertices for increasing refinement level.

of the "corner" subtetrahedra in Fig. C.1) of $\mathcal{Q}$ would have vertices

$$q_1, \quad \frac{q_1+q_2}{\|q_1+q_2\|}, \quad \frac{q_1+q_3}{\|q_1+q_3\|}, \quad \text{and} \quad \frac{q_1+q_4}{\|q_1+q_4\|}. \tag{5.12}$$

Selecting the internal edge via Eq. (5.11) is critical to our BB convergence guarantee in Sec. 5.1.3. If Eq. (5.11) is not used, the individual subsets $\mathcal{Q}$ can become highly skewed due to repeated distortion from the unit-norm projection of the vertices, and refining $\mathcal{Q}$ does not necessarily correspond to shrinking the angular range of rotations it captures. Since we use Eq. (5.11), however, Lemma 5.1.1 guarantees that subdividing $\mathcal{Q}$ shrinks its set of rotations appropriately:

Lemma 5.1.1. *Let γ_N be the min dot product between vertices of any one $\mathcal{Q}$ at refinement level N. Then*

$$\tfrac{2\gamma_{N-1}}{1+\gamma_{N-1}} \le \gamma_N, \qquad \textit{where} \qquad \gamma_0 \triangleq \cos 36^\circ. \tag{5.13}$$

This result (proof in Appendix C.1.3) shows that the tetrahedra shrink and allow BB to improve its bounds during subdivision. Figure 5.5 demonstrates the tightness of this bound, showing that $\cos^{-1}\gamma_N$ converges to 0 as $N \to \infty$. We conjecture that the *max* dot product Γ_N satisfies a similar recursion, $\Gamma_N \le \sqrt{(1+\Gamma_{N-1})/2}$, although this is not required for our convergence analysis. Fig. 5.5 shows empirically that this matches the true max dot product, but we leave the proof as an open problem.

von-Mises-Fisher Mixture Model Bounds

BB requires both upper and lower bounds on the maximum of the objective function within each projected tetrahedron $\mathcal{Q}$, i.e. we need L and U such that

$$L \le \max_{q\in\mathcal{Q}} \textstyle\sum_{k,k'} D_{kk'} f(z_{kk'}(q)) \le U\,. \tag{5.14}$$

For the lower bound L, one can evaluate the objective at *any* point in $\mathcal{Q}$ (e.g. its center).

For the upper bound U, we use a quadratic upper bound on $f(z)$ (see Fig. 5.6 and

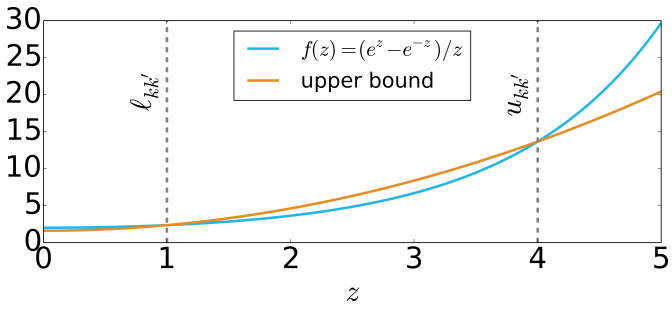


Figure 5.6: The function $f(z)$ quantifying the rotational alignment between two DP-vMF mixture models and its quadratic upper bound, valid for $z \in [\ell_{kk'}, u_{kk'}]$ (here, $\ell_{kk'} = 1$ and $u_{kk'} = 4$).

Appendix C.1.2 for details), noting that $\ell_{kk'} \leq z_{kk'}(q) \leq u_{kk'}$ for all $q \in \mathcal{Q}$, where

$$\ell_{kk'} \triangleq \min_{q \in \mathcal{Q}} z_{kk'}(q) \quad \text{and} \quad u_{kk'} \triangleq \max_{q \in \mathcal{Q}} z_{kk'}(q), \tag{5.15}$$

whose computation is discussed in Sec. 5.1.3. This results in the upper bound U where

$$\begin{aligned} U &= \max_{q \in \mathcal{Q}} \; q^T A q + B \\ A &\triangleq \textstyle\sum_{k,k'} 2 D_{kk'} \tau_{1k} \tau_{2k'} g_{kk'} \Xi_{kk'} \\ B &\triangleq \textstyle\sum_{k,k'} D_{kk'} \left((\tau_{1k}^2 + \tau_{2k'}^2) g_{kk'} + h_{kk'} \right) \\ g_{kk'} &\triangleq \tfrac{f(u_{kk'}) - f(\ell_{kk'})}{u_{kk'}^2 - \ell_{kk'}^2} \\ h_{kk'} &\triangleq \tfrac{u_{kk'}^2 f(\ell_{kk'}) - \ell_{kk'}^2 f(u_{kk'})}{u_{kk'}^2 - \ell_{kk'}^2}, \end{aligned} \tag{5.16}$$

and $\Xi_{kk'} \in \mathbb{R}^{4 \times 4}$ is defined as the matrix for which $\mu_{1k}^T (q \circ \mu_{2k'}) = q^T \Xi_{kk'} q$ for any quaternion q. For clarity reasons let $u = \mu_{1k}$ and $v = \mu_{2k'}$ then

$$\Xi(u, v) = \begin{bmatrix} u_i v_i - u_j v_j - u_k v_k & u_j v_i + u_i v_j & u_i v_k + u_k v_i & u_k v_j - u_j v_k \\ u_j v_i + u_i v_j & u_j v_j - u_i v_i - u_k v_k & u_j v_k + u_k v_j & u_i v_k - u_k v_i \\ u_i v_k + u_k v_i & u_j v_k + u_k v_j & u_k v_k - u_i v_i - u_j v_j & u_j v_i - u_i v_j \\ u_k v_j - u_j v_k & u_i v_k - u_k v_i & u_j v_i - u_i v_j & u^T v \end{bmatrix}. \tag{5.17}$$

See Appendix C.1.1 for derivation and details. Writing $q = Q\alpha$ as a linear combination of vertices of $\mathcal{Q}$ as in Eq. (5.10), the upper bound can be found as

$$\begin{aligned} U = \max_{\alpha \in \mathbb{R}^4} \quad & \alpha^T Q^T A Q \alpha + B \\ \text{s.t.} \quad & \alpha^T Q^T Q \alpha = 1 \,, \; \alpha \geq 0 \,. \end{aligned} \tag{5.18}$$

Since $\alpha \in \mathbb{R}^4$, and we have the constraint $\alpha \geq 0$, we can search over all $\sum_{i=1}^{4} \binom{4}{i} = 15$ possible combinations of components of α being zero or nonzero. Thus we solve the

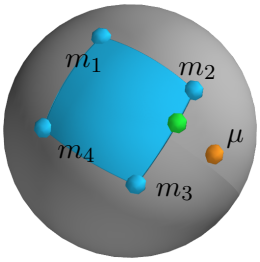


Figure 5.7: Closest point (green) inside the area implied by a tetrahedra cell of the tessellation from a query point μ (orange, Eq. 5.22).

optimization for $U_{\mathcal{I}}$ given each possible subset $\mathcal{I} \subseteq \{1,2,3,4\}$ of nonzero components of α, and set

$$U = B + \max_{\mathcal{I} \subseteq \{1,2,3,4\}} U_{\mathcal{I}}. \tag{5.19}$$

For $U_{\mathcal{I}}$, we use a Lagrange multiplier for the equality constraint in Eq. (5.18) and set the derivative to 0, yielding a small generalized eigenvalue problem of dimension $|\mathcal{I}| \leq 4$,

$$U_{\mathcal{I}} = \max\left\{\lambda : \exists v \geq 0,\ \left(Q^T A Q\right)_{\mathcal{I}} v = \lambda \left(Q^T Q\right)_{\mathcal{I}} v\right\}, \tag{5.20}$$

where v is a $|\mathcal{I}|$-dimensional vector, and subscript $\mathcal{I}$ denotes the submatrix with rows and columns selected from $\mathcal{I}$. The condition that all elements of v are nonnegative in Eq. (5.20) enforces that $\alpha \geq 0$ and thus α corresponds to a solution q that lies in $\mathcal{Q}$. Note that if v is an eigenvector, so is $-v$. If no v satisfies $v \geq 0$, then we define $U_{\mathcal{I}} = -\infty$.

Computing $\ell_{kk'}$ and $u_{kk'}$ To find the upper bound U in Eq. (5.18), we require the constants $\ell_{kk'}$ and $u_{kk'}$ for each pair of mixture components k, k'. Given their definitions in Eq. (5.15), we have

$$\begin{aligned} u_{kk'} &= \sqrt{\tau_{1k}^2 + \tau_{2k'}^2 + 2\tau_{1k}\tau_{2k'} \max_{q \in \mathcal{Q}} \mu_{1k}^T (q \circ \mu_{2k'})}\,, \\ \ell_{kk'} &= \sqrt{\tau_{1k}^2 + \tau_{2k'}^2 - 2\tau_{1k}\tau_{2k'} \max_{q \in \mathcal{Q}} (-\mu_{1k})^T (q \circ \mu_{2k'})}\,. \end{aligned} \tag{5.21}$$

Since the inner optimization objective only depends on the rotation of $\mu_{2k'}$ by q, we can reformulate the optimization as being over the set of 3D vectors $v \in \mathbb{S}^2$ such that $v = q \circ \mu_{2k'}$ for some $q \in \mathcal{Q}$. Thus, finding $u_{kk'}$ and $\ell_{kk'}$ is equivalent to finding the closest and furthest unit vectors in 3D to μ_{1k} over the set of such vectors v, shown in Fig. 5.7. To solve this problem, let the vertices of $\mathcal{Q}$ be q_i, $i \in \{1, \ldots, 4\}$, and define the matrix $M \triangleq [m_1, \ldots, m_4] \in \mathbb{R}^{3\times 4}$ where $m_i \triangleq q_i \circ \mu_{2k'}$. The inner optimization in

Eq. (5.21) can be written as (for $u_{kk'}$ set $\mu = \mu_{1k}$; for $\ell_{kk'}$ set $\mu = -\mu_{1k}$)

$$\begin{aligned} J = \max_{\alpha \in \mathbb{R}^4} \quad & \mu^T M \alpha \\ \text{s.t.} \quad & \alpha^T M^T M \alpha = 1 \quad \alpha \geq 0. \end{aligned} \tag{5.22}$$

Showing that Eq. (5.22) is equivalent to solving the inner optimizations of Eq. (5.21) is quite technical and is deferred to Appendix C.1.5. Again we search over all $\sum_{i=1}^{3} \binom{4}{i} = 14$ possible combinations of components of α being zero or nonzero (we do not check the $i = 4$ case since in this case the matrix $M_{\mathcal{I}}$ below is rank-deficient). We thus solve the optimization for $J_{\mathcal{I}}$ given each subset $\mathcal{I} \subseteq \{1, \ldots, 4\}$, $|\mathcal{I}| \leq 3$ of nonzero components, and set

$$J = \max_{\mathcal{I} \subseteq \{1,2,3,4\} \text{ s.t. } |\mathcal{I}| \leq 3} J_{\mathcal{I}}. \tag{5.23}$$

To solve for $J_{\mathcal{I}}$, we use a Lagrange multiplier for the equality constraint, and set derivatives to 0 to find that

$$J_{\mathcal{I}} = \sigma \sqrt{\mu^T M_{\mathcal{I}} \left(M_{\mathcal{I}}^T M_{\mathcal{I}} \right)^{-1} M_{\mathcal{I}}^T \mu} \tag{5.24}$$

where

$$\sigma = \begin{cases} 1 & \left(M_{\mathcal{I}}^T M_{\mathcal{I}} \right)^{-1} M_{\mathcal{I}}^T \mu \geq 0 \\ -1 & \left(M_{\mathcal{I}}^T M_{\mathcal{I}} \right)^{-1} M_{\mathcal{I}}^T \mu \leq 0 \\ -\infty & \text{else}, \end{cases} \tag{5.25}$$

and $M_{\mathcal{I}}$ is the matrix constructed from the set of columns in M corresponding to $\mathcal{I}$. Note that σ is also defined to be $\sigma = -\infty$ if $M_{\mathcal{I}}^T M_{\mathcal{I}}$ is not invertible. After solving for the value of J via Eq. (5.23), we substitute it back into Eq. (5.21) to obtain $u_{kk'}$ or $\ell_{kk'}$ as desired.

Convergence Properties

We have now developed all the components necessary to optimize Eq. (5.9) via BB on $\mathbb{S}^3$. Theorem 5.1.1 (proof in Appendix C.1.4) provides a bound on the worst-case search tree depth N to guarantee BB terminates with rotational precision of ϵ degrees, along with the overall computational complexity. Note that the complexity of BB is exponential in N, but since N is logarithmic in ϵ^{-2} (by Theorem 5.1.1, Eq. (5.26) and $\cos x \simeq 1 - x^2$ for $x \ll 1$), the complexity of BB is polynomial in ϵ^{-1}. Recall from Sec. 5.1.3 that γ_0 for the 600-cell is $\gamma_0 \triangleq \cos 36°$.

Theorem 5.1.1. *Suppose γ_0 is the initial maximum angle between vertices in the tetrahedra tessellation of $\mathbb{S}^3$, and let*

$$N \triangleq \max \left\{ 0, \left\lceil \log_2 \frac{\gamma_0^{-1} - 1}{\cos(\epsilon/2)^{-1} - 1} \right\rceil \right\}. \tag{5.26}$$

Then at most N refinements are required to achieve an angular tolerance of ϵ on $\mathbb{S}^2$, and BB has complexity $O(\epsilon^{-6})$.

■ 5.1.4 Gaussian Mixture Translational Alignment

In this section, we reuse notation for simplicity and to highlight parallels between the translational and rotational alignment problems. We model the density of points in the two point clouds as Gaussian mixture models (GMMs) with means $\{\mu_{ik}\}_{k=1}^{K_i}$, covariances $\{\Sigma_{ik}\}_{k=1}^{K_i}$, and weights $\{\pi_{ik}\}_{k=1}^{K_i}$, $\sum_{k=1}^{K_i} \pi_{ik} = 1$, for $i \in \{1, 2\}$, with density

$$\hat{p}_i(x) = \sum_{k=1}^{K_i} \pi_{ik} \, \mathcal{N}\left(x; \mu_{ik}, \Sigma_{ik}\right) . \tag{5.27}$$

If BB over the rotation space returns multiple optimal rotations $q^\star$, we solve the translational alignment problem for each of them. For the remainder of this section, we fix $q^\star$ and show how to solve the translational alignment problem for that particular $q^\star$ without explicit representation in notation. GMMs can be inferred in a variety of ways [42, 142]. Let $R^\star \in \mathbb{SO}(3)$ be the optimal rotation corresponding to a $q^\star$ recovered using BB over $\mathbb{S}^3$. Then defining

$$\begin{aligned} m_{kk'} &\triangleq R^\star \mu_{2k'} - \mu_{1k} \, , \\ S_{kk'} &\triangleq \Sigma_{1k} + R^\star \Sigma_{2k'} R^{\star T} \, , \\ z_{kk'}(t) &\triangleq -\tfrac{1}{2} \left(t - m_{kk'}\right)^T S_{kk'}^{-1} \left(t - m_{kk'}\right) \, , \end{aligned} \tag{5.28}$$

the translational optimization in Eq. (5.5) becomes:

$$\begin{aligned} \hat{t} = &\arg\max_{t \in \mathbb{R}^3} \sum_{k,k'} D_{kk'} f(z_{kk'}(t)) \\ &\text{where } f(z) \triangleq \exp(z) \, , \; D_{kk'} \triangleq \frac{\pi_{1k} \pi_{2k'}}{\sqrt{(2\pi)^3 \left|S_{kk'}\right|}} \, . \end{aligned} \tag{5.29}$$

This is again a non-concave maximization, motivating the use of a global approach. Thus, we develop a second BB procedure on $\mathbb{R}^3$ to find the optimal translation.

Note that this section has strong parallels with the rotational alignment section to make it easier to follow but also to point out parallels in the approaches that can be exploited in an actual implementation of the algorithms. An implementation of the previous branch and bound procedure can be modified to solve this problem by replacing f, $D_{kk'}$, the covering/subdivision procedure, and the upper/lower bounds appropriately.

Cover and Refinement of $\mathbb{R}^3$

We tessellate the space of translations, $\mathbb{R}^3$ with rectangular cells. The initial tessellation is obtained by enclosing both point clouds with a single rectangular bounding box with diagonal length γ_0. For the refinement step, we choose to subdivide the cell into eight equal-sized rectangular cells. Thus, the minimum γ_N diagonal of the rectangular cells at

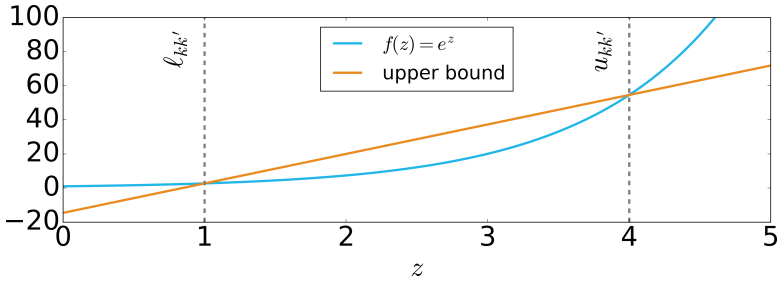


Figure 5.8: The translational alignment cost function $f(z)$ and its linear upper bound, valid for $z \in [\ell_{kk'}, u_{kk'}]$ (here, $\ell_{kk'} = 1$ and $u_{kk'} = 4$).

refinement level N possesses a straightforward shrinkage property similar to Eq. (5.13),

$$\frac{\gamma_{N-1}}{2} = \gamma_N. \tag{5.30}$$

Gaussian Mixture Model Bounds

As in the rotational problem, the translational BB algorithm requires lower and upper bounds on the objective function in Eq. (5.29):

$$L \leq \max_{t \in \mathcal{Q}} \textstyle\sum_{k,k'} D_{kk'} f(z_{kk'}(t)) \leq U\,. \tag{5.31}$$

For the lower bound L, one can evaluate the objective at *any* $t \in \mathcal{Q}$ (e.g. its center).

For the upper bound U, we use a linear upper bound on $f(z)$ (see Fig. 5.8 and Appendix C.2.1 for details), noting that $\ell_{kk'} \leq z_{kk'}(t) \leq u_{kk'}$ for all $q \in \mathcal{Q}$, where

$$\ell_{kk'} \triangleq \min_{t \in \mathcal{Q}} z_{kk'}(t) \quad \text{and} \quad u_{kk'} \triangleq \max_{t \in \mathcal{Q}} z_{kk'}(t)\,, \tag{5.32}$$

whose computation is discussed in Section 5.1.4. This results in the upper bound U, where

$$\begin{aligned}
U &\triangleq \max_{t \in \mathcal{Q}} t^T A t + B^T t + C \\
A &\triangleq -\tfrac{1}{2} \textstyle\sum_{k,k'} D_{kk'} g_{kk'} S_{kk'}^{-1} \\
B &\triangleq \textstyle\sum_{k,k'} D_{kk'} g_{kk'} S_{kk'}^{-1} m_{kk'} \\
C &\triangleq \textstyle\sum_{kk'} D_{kk'} \left(h_{kk'} - \tfrac{1}{2} g_{kk'} m_{kk'}^T S_{kk'}^{-1} m_{kk'} \right) \\
g_{kk'} &\triangleq \tfrac{f(u_{kk'}) - f(\ell_{kk'})}{u_{kk'} - \ell_{kk'}} \\
h_{kk'} &\triangleq \tfrac{u_{kk'} f(\ell_{kk'}) - \ell_{kk'} f(u_{kk'})}{u_{kk'} - \ell_{kk'}}\,.
\end{aligned} \tag{5.33}$$

This is a concave quadratic maximization over a rectangular cell $\mathcal{Q}$. Thus, we obtain U as the maximum over all local optima in the interior, faces, edges, and vertices of $\mathcal{Q}$.

Computing $\ell_{kk'}$ and $u_{kk'}$ Using the form of $z_{kk'}(t)$ in Eq. (5.28), we have that

$$\begin{aligned} \ell_{kk'} &= \min_{t \in \mathcal{Q}} t^T A t + B^T t + C \\ u_{kk'} &= \max_{t \in \mathcal{Q}} t^T A t + B^T t + C \\ A &\triangleq -\tfrac{1}{2} S_{kk'}^{-1} \\ B &\triangleq S_{kk'}^{-1} m_{kk'} \\ C &\triangleq -\tfrac{1}{2} m_{kk'}^T S_{kk'}^{-1} m_{kk'}. \end{aligned} \tag{5.34}$$

Because of the concavity of the objective, $u_{kk'}$ can be obtained with the exact same algorithm as used to solve Eq. (5.33). Namely, $\ell_{kk'}$ can be obtained by checking the 8 vertices of $\mathcal{Q}$, as the minimum of a concave function over a rectangular cell must occur at one of its vertices.

Convergence Properties

We now have all the components necessary to optimize Eq. (5.29) via BB on $\mathbb{R}^3$. As in the rotational alignment case, we provide a characterization (Theorem 5.1.2, proof in Appendix C.2.2) of the maximum refinement depth N required for a desired translational precision ϵ, along with the complexity of the algorithm. Note that while the complexity of BB is exponential in N, N is logarithmic in ϵ^{-1} (Theorem 5.1.2), so BB has polynomial complexity in ϵ^{-1}.

Theorem 5.1.2. *Suppose γ_0 is the initial diagonal length of the translation cell in $\mathbb{R}^3$, and let*

$$N \triangleq \max\left\{0, \left\lceil \log_2 \tfrac{\gamma_0}{\epsilon} \right\rceil\right\}. \tag{5.35}$$

Then at most N refinements are required to achieve a translational tolerance of ϵ, and BB has complexity $O(\epsilon^{-3})$.

■ 5.1.5 Evaluation and Results

We evaluate BB (both with and without final local refinement [44]) on four datasets [52, 191, 239] compared to three global methods: an FT-based method [163], GoICP [256] (20% trimming), and GOGMA [37]. To generate the vMF-MMs and GMMs for BB, we cluster the data with DP-vMF-means [222] and DP-means [142], and fit maximum likelihood MMs to the clustered data. To account for nonuniform point densities due to the sensing process, we weight each point's contribution to the MMs by its surface area, estimated by the disc of radius equal to the fifth nearest neighbor distance. We use kNN+PCA [167, 261, 262] to extract surface normals. See Sec. 2.8.1 for more details. To improve the robustness of BB, it is run three times on each problem with scale values $\lambda_n \in \{45°, 65°, 80°\}$ in DP-vMF-means (included in the timing results). The scale λ_x for DP-means is manually selected to yield around 50 mixture components.

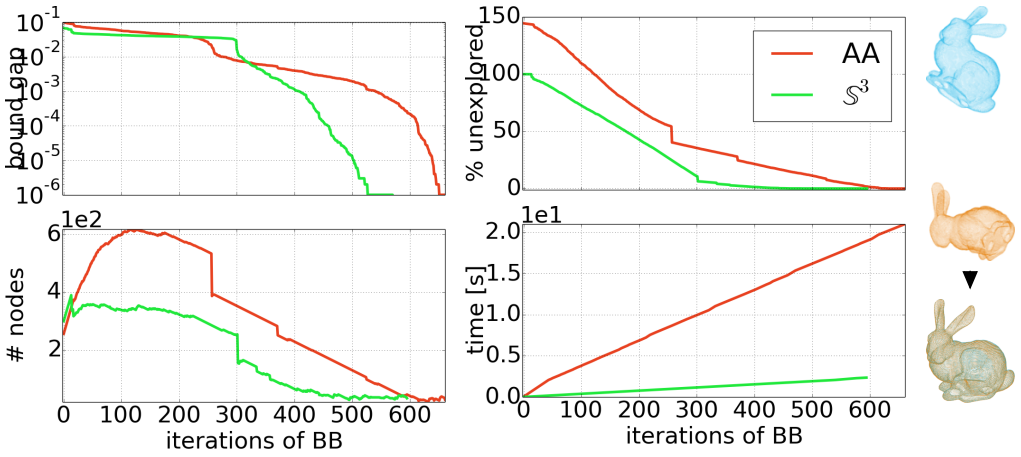


Figure 5.9: BB alignment of the full Stanford Bunny. The proposed tessellation approach ($\mathbb{S}^3$) is compared against axis-angle space tessellation (AA). The proposed approach's more uniform tessellation leads to faster exploration, faster reduction of the bound gap, more pruned nodes and roughly 4× faster runtime.

Using Theorems 5.1.1 and 5.1.2, we terminate rotational BB at $N = 11$ and translational BB at $N = 10$ for a rotational accuracy of 1° and a translational accuracy of $\frac{\gamma_0}{1024}$, where γ_0 is defined in Eq. (5.30). All timing results include algorithm-specific preprocessing of the data. We used a 3GHz core i7 CPU and a GeForce GTX 780 GPU. While clustering via DP-means and DP-vMF-means uses the GPU, we only use parallel CPU threads for the eight BB bound evaluations after each branch step.

Stanford Bunny [239] Independent of the tessellation strategy, BB perfectly aligns the Stanford Bunny with a randomly transformed version of itself, as shown in Fig. 5.9. This perfect alignment is expected for completely overlapping scans with low noise. The results of aligning two partial scans of the Stanford Bunny with relative viewpoint difference 45° are shown in Fig. 5.10. For partial scans the algorithm will generally not produce a perfect alignment by itself. Hence we run ICP, a local alignment method starting from the pose obtained via BB. As can be seen, BB's initial alignment is close enough to allow ICP to converge to a perfect alignment.

We compare the proposed $\mathbb{S}^3$ tessellation to the commonly used axis-angle-based tessellation (AA). The AA-based BB algorithms upper bounds are obtained as the maximum upper bound of the five Tetrahedra that tessellate a quadratic AA-cell. In comparison to axis-angle-based BB, the proposed approach leads to a faster reduction in the bound gap, faster exploration, and a smaller number of active nodes. These factors combine to reduce the overall necessary number of iterations by 20% as well as the computation time per iteration by an order of magnitude vs the AA tessellation. This shows conclusively that the proposed tessellation leads to more efficient BB optimization. Note that the AA tessellation starts at 146% unexplored space because it

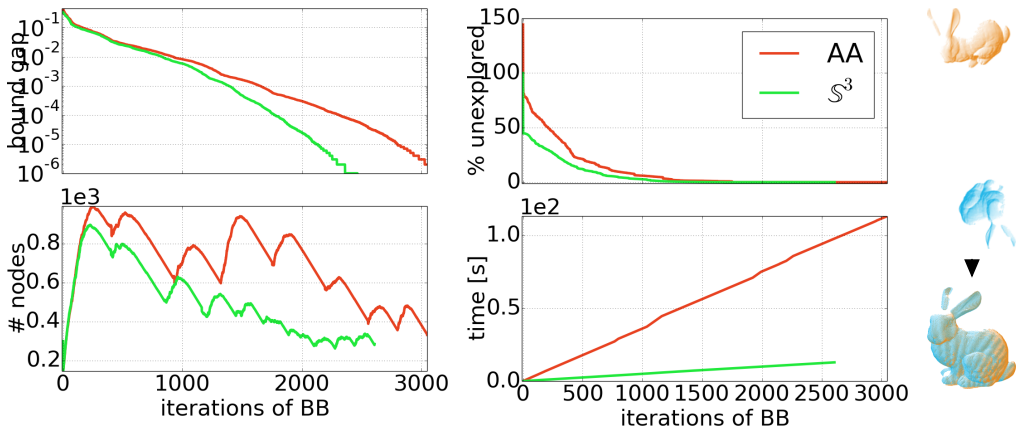


Figure 5.10: Alignment of partial scans of the Stanford Bunny with 45° viewpoint difference. The proposed tessellation approach ($\mathbb{S}^3$) is compared against axis-angle space tessellation (AA). The proposed approach's more uniform tessellation leads to faster exploration, faster reduction of the bound gap, more pruned nodes and roughly 4× faster runtime.

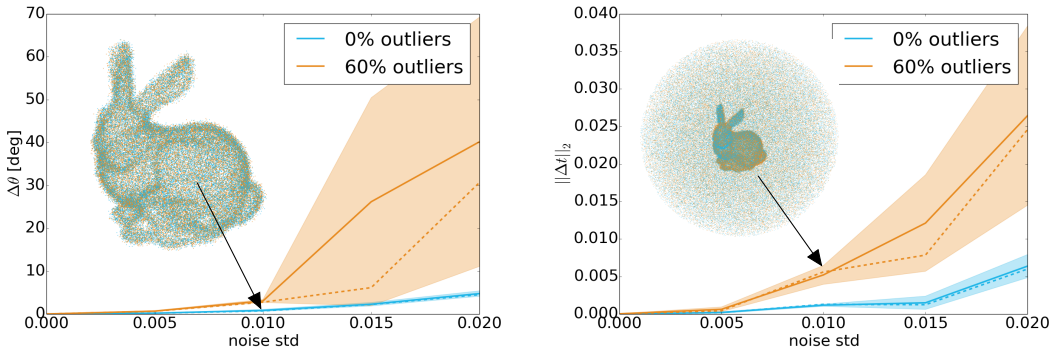


Figure 5.11: Evaluation of translational and rotational error under additive isotropic Gaussian noise and outliers. Shaded areas show one standard deviation around the mean (solid line). The median errors are indicated with dashed lines.

covers the rotation space more than once as discussed in Sec. 5.1.3. In both cases BB finds the optimal translation within 200 iterations.

Noise and Outliers The robustness to noise and outliers is important for any alignment method. In Fig. 5.11 we show the angular and rotational BB+ICP alignment error as a function of noise standard deviation and outlier ratio for the alignment of the full Stanford Bunny. The synthetic data is created by first adding isotropic Gaussian noise (with the designated standard deviation) and then sampling random outlier points

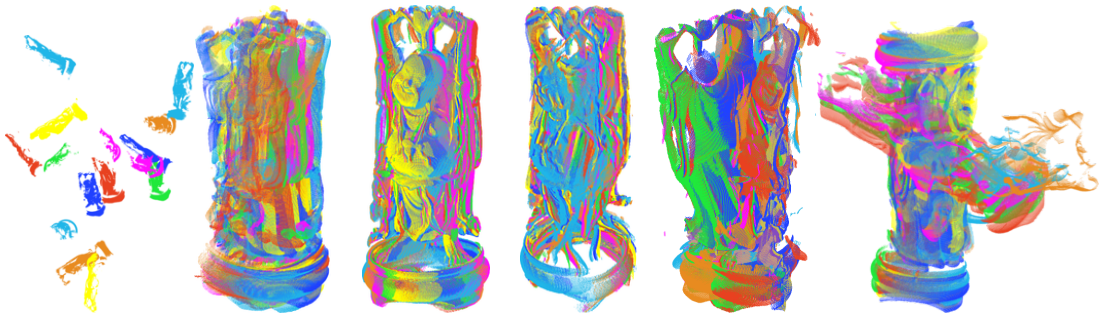


Figure 5.12: Alignment of partial scans of Happy Buddha taken at 24° increments. The only successful alignment is obtained by BB+ICP.

uniformly inside a sphere with twice the radius of the size of the Stanford Bunny. Standard deviations and translational errors are reported as a fraction of the diameter of the original Stanford Bunny point cloud. The evaluation of error statistics over 336 instantiations of the alignment problem, shows the robustness of our method to unrealistic amounts of corruption (high noise, 60% outliers). Above a noise threshold, surface normal computation fails leading to high alignment error.

Happy Buddha [52] This dataset consists of 15 scans taken at 24° rotational increments about the vertical axis of a statue. This dataset is challenging, as the scans contain few overlapping points, and the surface normal distributions are anisotropic. We perform pairwise alignment of consecutive scans, and render the aligned scans together in one coordinate system (Fig. 5.12). The only successful alignment is produced by BB+ICP. This shows the advantage of using surface normals for rotational alignment. Other methods using points (GoICP) or GMMs (GOGMA) have difficulty dealing with ambiguities due to the "flatness" of the scans.

Office Scan Figure 5.13 demonstrates that BB+ICP finds accurate registrations on noisy, incomplete, cluttered and irregular point clouds as long as good surface normal estimates are available. This demonstrates the potential use of BB+ICP for loop closure detection.

Apartment Dataset [191] This dataset consists of 44 LiDAR scans with an average overlap of 84%. We pick the Apartment dataset from [191] for evaluation since the scans are sufficiently dense across the whole dataset to compute surface normals of sufficient quality.

Figure 5.14 shows the BB+ICP aligned scans of the dataset. Table 5.1 compares the accuracy and inlier percentages defined by (C)oarse (2m; 10°), (M)edium (1m; 5°) and (F)ine (0.5m; 2.5°) thresholds for all algorithms. For GoICP, we used 100 scan points and an accuracy threshold of 0.01. Increasing N or decreasing ϵ by an order of magnitude lead to unreasonably long run times as also noted previously in [37]. We

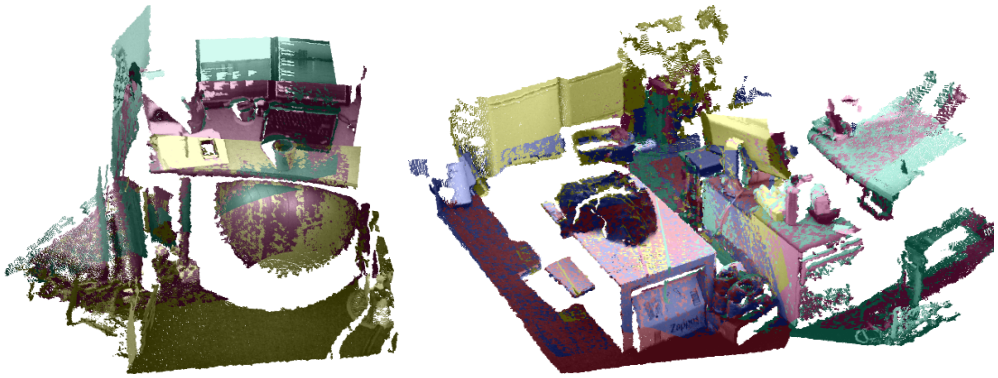


Figure 5.13: Correct alignment of noisy, incomplete, and partially overlapping RGB-D point clouds of cluttered indoor scenes using BB+ICP. Left shows a desk with monitors and a gymnastic ball for sitting and right shows a larger pretty cluttered indoor area with a coffee table, parts of a table and a fridge and parts of the floor. Colors indicate different scans.

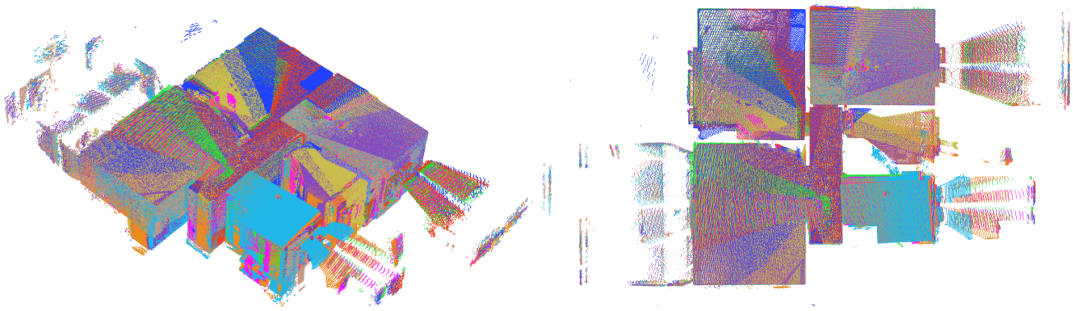


Figure 5.14: Alignment of the apartment dataset [191] using BB+ICP. The different colors indicate distinct LiDAR scans that were automatically aligned.

used the scale parameter of $\lambda_x = 1.3$m for GMM computations in both GOGMA and BB. This yields roughly 50 mixture components on average across the dataset. Picking the scale to small leads to GOGMA being faster but also falling for the Manhattan World ambiguity.

Man-made environments such as this dataset exhibit Manhattan World symmetry in their surface normal distributions as explored in Chapter 3. We thus transform the rotation obtained via rotational BB by all 24 Manhattan World rotations, and search over all using translational BB. Note that doing this is straightforward in the proposed decoupled BB approach, as opposed to a joint approach, e.g. GoICP and GOGMA.

Table 5.1 and Fig. 5.15 show that BB with searching over both scale and MW rotations leads to the best accuracy among all algorithms, with a 3× speedup over the 2nd best method, GOGMA (which uses a GPU). From the inlier percentages it is clear that FT and GoICP do not perform well. The CDFs in Fig. 5.15 show that

Method	$[*]_{\lambda}$	$[*]_{\lambda+}$	$[*]^{M}$	$[*]^{M}_{+}$	$[*]^{M}_{\lambda}$	$[*]^{M}_{\lambda+}$	[37]	[37]$_{+}$	[256]	[163]
Rotation [°]	28.6	26.9	5.52	1.61	3.77	**1.36**	7.14	5.14	24.2	30.0
Translation [m]	0.48	0.43	0.12	0.04	0.08	**0.03**	0.22	0.09	0.46	0.65
Inlier % C	79.6	81.8	90.9	95.5	93.2	**97.7**	97.5	97.5	47.7	29.5
Inlier % M	75.0	81.8	79.6	95.5	86.4	**97.7**	85.0	97.5	34.1	18.2
Inlier % F	54.6	81.8	36.4	95.5	61.4	**97.7**	47.5	97.5	13.6	2.27
Time [s]	**32.6**	50.0	38.4	57.3	140	156	405	675	62.0	470

Table 5.1: Apartment [191] results using BB [∗], GOGMA [37], GoICP [256], and FT [163]. We denote search over rotational scale via $_{\lambda}$, search over MW ambiguities with ^{M} and local refinement with +. We report rotational (Rot), translational (Tran), timing, and inlier (Inl) percentages for (C)oarse (2m; 10°), (M)edium (1m; 5°) and (F)ine (0.5m; 2.5°) alignment.

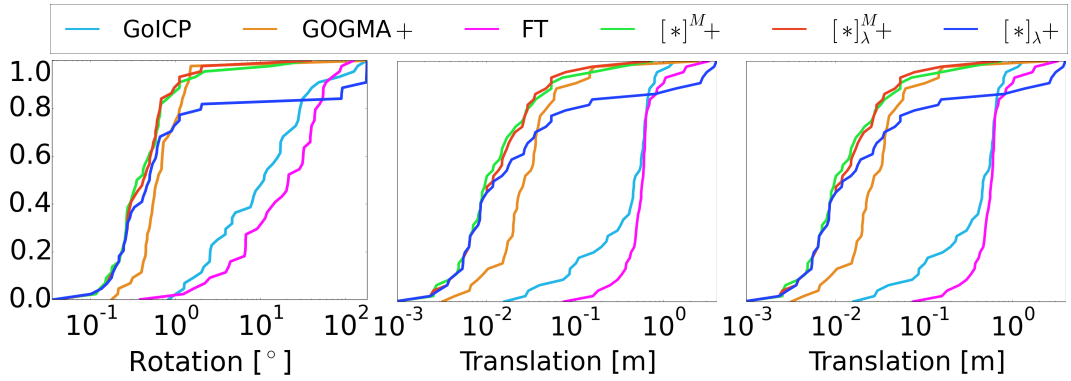


Figure 5.15: Cumulative density functions of rotational error, translational error, and runtime for the alignment of the Apartment dataset. The variants of the proposed BB algorithm show higher rotational and translational alignment quality and are an order of magnitude faster than the competing GOGMA algorithm.

Method	BB$_{\lambda}$	BB$_{\lambda+}$	GOGMA	GOGMA$_{+}$	GoICP	FT
Rotation [°]	3.92	**2.01**	19.3	16.5	58.3	5.58
Translation [m]	0.25	**0.11**	1.45	0.71	0.66	0.68
Inlier % C	**96.8**	**96.8**	87.1	90.3	41.9	87.1
Inlier % M	77.4	**87.1**	54.8	**87.1**	38.7	80.7
Inlier % F	19.4	67.7	16.1	**83.9**	6.45	51.6
Time [s]	**23.70**	28.3	105	164	138	242

Table 5.2: Gazebo Summer [191] results for BB, GOGMA, GoICP, FT. We report rotational (Rot), translational (Tran), timing, and inlier (Inl) percentages for (C)oarse (2m; 10°), (M)edium (1m; 5°) and (F)ine (0.5m; 2.5°) alignment.

Figure 5.16: Depiction of the BB+ICP alignment of the first 6 LiDAR scans of the Gazebo Summer dataset [191]. While details of the alignment could be improved, the overall large scale alignment is inferred correctly. Different colors indicate distinct LiDAR scans.

accounting for Manhattan World symmetry (red, green) is important. Ignoring it (blue) causes scans to be flipped by 90°/180°, affecting the mean error strongly. Our method's runtime is spent 80% on preprocessing (scale estimation, DP-vMF-means and DP-means clustering, and DP-vMF-MM and DP-GMM parameter fitting) and 20% on the actual BB search.

Gazebo Summer Dataset [191] The dataset contains of 33 scans taken in a mostly unstructured outdoor setting with trees, bushes and a gazebo. We evaluate the alignment in the same way as the Apartment dataset. Figure 5.16 shows some alignments of the point cloud obtained via BB+ICP. The high degree of clutter, noise and outliers is clearly visible. Despite those difficult conditions, the coarse alignment is correct. Table 5.2 lists results for the alignment of all scans in the dataset. It is clear that all algorithms have a harder time aligning the scans. Still BB performs well both in speed and quality. Specifically BB gets the coarse alignment right in almost all cases whereas GOGMA fails in 10% more cases. Looking at the alignments this is due to GOGMA flipping scans on the head (this is also indicated by the high mean rotational error), whereas BB has no trouble finding the right upward direction due to the strong upward surface normal cluster from the ground.

■ 5.2 Nonparametric Direction-aware 3D Reconstruction

After exploring and demonstrating the advantages of inferring and using a directional segmentation of the environment for global point cloud alignment in the previous section, we now turn to the full nonparametric direction-aware SLAM problem of joint inference over map, camera trajectory and directional scene segmentation.

While the SLAM problem has received decades of attention [36] it is by definition

a subproblem to semantic or categorical SLAM which adds a component of scene understanding to the purely geometric reasoning about a world map and the pose of the perception system. In a first step towards semantic SLAM and perception we explore the utility of incorporating direction-awareness into the SLAM problem. As explored and argued in the previous Chapters 3 and 4, directional segmentations capture regularities of the environment that can be leveraged for further reasoning about the scene. Because of its flexibility we use the Stata Center World assumption as a directional scene prior. The utility of this assumption is twofold: (1) it directly provides a prior on the surface normal distribution as shown in Chapter 4 and (2) by the definition of surface normals it implies that locally surface areas that are in the same segment are planar to the extend of the concentration of the directional cluster. In other words: planar surface areas of the same segment lead to concentrated surface normal clusters whereas curved surface areas lead to less peaked and more uniform surface normal distributions. This connection between the scene-wide surface normal distribution and the local surface properties suggests the direction-aware mapping formulation we explore in this section. Furthermore, we show that the directional clustering of a scene's surface normals is useful to guide the selection of a diverse set of observations to improve camera pose estimation. Since both mapping and localization use the inferred directional segmentation, we ascribe direction-awareness to the SLAM system.

In contrast to plane-based categorical SLAM methods [41, 129, 158, 210] the proposed directional-segmentation-approach has the advantage that no planes have to be extracted explicitly and that non-planar areas are captured by the model implicitly without having to introduce a special "non-planar" class. Furthermore sampling-based inference allows soft associations to directions whereas all related work but the EM-based algorithm of CPA-SLAM [158] make hard assignments to specific planes that are not revisited [41, 129, 210]. Being able to revisit assignments allows refining and correcting the model under additional observations.

The system described in this section is the first semi-dense nonparametric direction-aware SLAM system and also the first system to perform joint inference over a Bayesian nonparametric scene segmentation and the world map using Gibbs-sampling without precluding real-time operation (see literature review in Sec. 1.1). We demonstrate experimentally that using the directional Stata Center World segmentation improves the efficiency of camera pose estimation and leads to higher joint accuracy in mapping and camera tracking.

■ 5.2.1 Joint Directional Segmentation, Localization and Mapping

As alluded to in the introduction, we define direction-aware SLAM as reasoning about the joint distribution of a world map m, the trajectory of the perception system T and the directional segmentation z given observations x. Concretely, we choose to represent the map as a collection of surfels [97, 133, 244]. Surfels are localized planes with position p_i, orientation n_i, color I_i (RGB for visualization and gray-scale for camera tracking) and some radius r_i. For notational clarity let $s_i = \{p_i, n_i, I_i, r_i\}$ collect all properties of

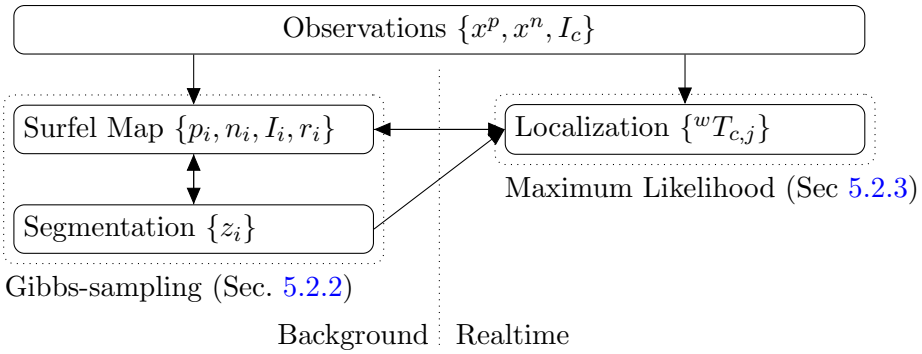


Figure 5.17: An high-level overview over the direction-aware SLAM model and inference. Observation acquisition and maximum likelihood camera localization runs in realtime whereas joint Gibbs-sampling-based inference over map and directional segmentation runs in the background.

surfel i. The world is observed via a RGB-D camera at poses $\{^mT_{c,j}\}$ where j indexes the pose at the reception of the jth camera frame. From the depth image, we compute point observations x^p and surface normal observations x^n. The RGB image gives direct access to surface color information I_c. We collect all observations of the jth frame in the variable $x_j = \{x^n, x^p, I_c\}$. The directional Stata Center World segmentation is associated to surfels via labels $\{z_i\}$. In this setup, the directional SLAM problem becomes inference over the posterior distribution:

$$p\left(\{s_i\}, \{z_i\}, \{^wT_{c,j}\} \mid \{x_j\}\right) \qquad \text{“direction-aware SLAM”} . \tag{5.36}$$

where the sets $\{\cdot\}$ are over all surfels for s_i and z_i and over all frames for $^wT_{c,j}$ and x_j. We perform inference on this nonparametric direction-aware SLAM posterior by iterating inference about the three subproblems of mapping, localization and directional segmentation.

The direction-aware mapping posterior has the form:

$$p\left(\{s_i\} \mid \{z_i\}, \{^wT_{c,j}\}, \{x\}\right) \qquad \text{“direction-aware mapping”} . \tag{5.37}$$

It incorporates the directional segmentation into mapping by encapsulating the assumption that neighboring surfels in the same segment should be part of the same plane. As described in Sec. 5.2.1, the distribution capturing this assumption couples surface normals and point locations and thus the global directional segmentation to the local 3D reconstruction.

For direction-aware localization we extend the iterative closest point (ICP) algorithm to use the directional segmentation of the surfel map as indicated in the posterior

$$p\left(\{^wT_{c,j}\} \mid \{z_i\}, \{s_i\}, \{x\}\right) \qquad \text{“direction-aware localization”} . \tag{5.38}$$

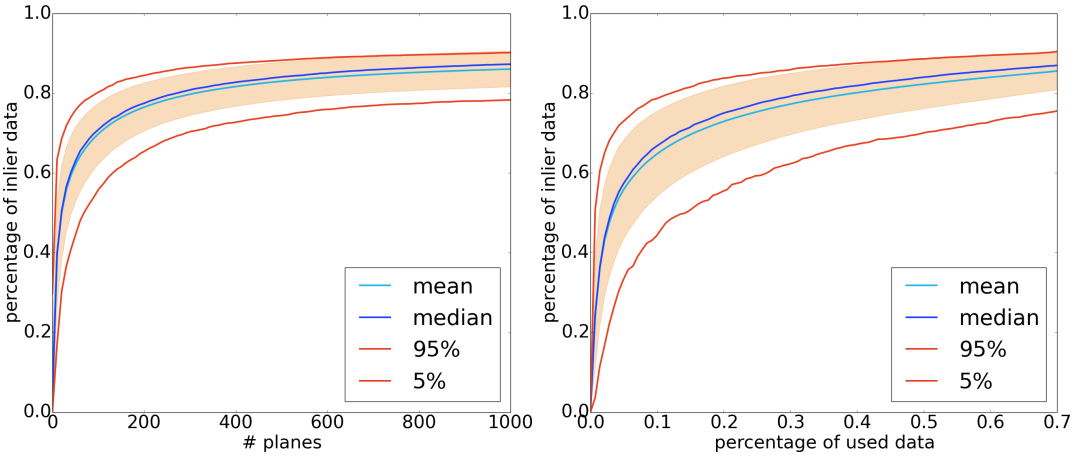


Figure 5.18: Percentage of inlier scene-points as described by a randomly sampled set of planes as a function of the number of planes and as a function of the percentage of scene points used to compute those planes. The plots summarize statistics over all of the scenes in the NYU v2 dataset [173].

In Sec. 5.2.3 we argue that the selection of a directionally-diverse set of plane observations via the directional segmentation is expected to improve pose estimation before demonstrating it in practice in Sec. 5.2.5.

The nonparametric directional segmentation captured in the following posterior distribution follows the Stata Center World assumption:

$$p\left(\{z_i\} \mid \{s_i\}, \{{}^{w}T_{c,j}\}, \{x\}\right) \qquad \text{“directional segmentation”} \,. \tag{5.39}$$

As described in detail in Sec. 5.2.1, we cast reasoning about the directional segmentation as inference about a Dirichlet process von-Mises-Fisher mixture model given surface normal observations similar to Sec. 4.4.3.

While it would be possible to perform fully joint inference using MCMC sampling-based methods (see Sec. 2.2), this is not practical when realtime operation is desired. To accommodate operation at camera frame-rate the inference is split into two main parts: (1) sampling-based joined inference on map and segmentation which runs in the background and (2) realtime camera pose estimation. A high-level overview over the parts involved in the nonparametric direction-aware SLAM system is depicted in Fig. 5.17.

With this introduction of the approach and the three dominant reasoning tasks, we now turn to the details of the world representation and the models in the following subsections before we describe the inference for segmentation and map in Sec. 5.2.2 and camera tracking in Sec. 5.2.3.

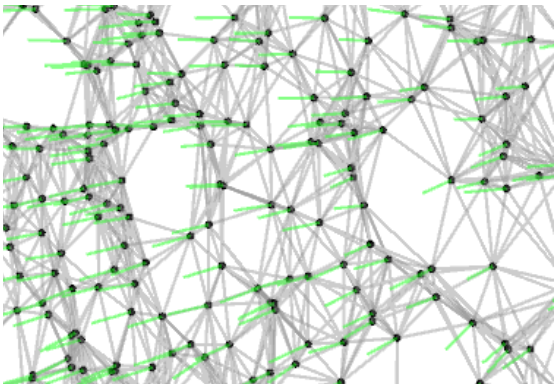


Figure 5.19: We maintain a nearest neighbor graph (grey lines) over surfel locations (black dots). Surfel orientations are depicted in green. The graph is used to the define a Markov random field over labels of the directional segmentation to yield spatially smooth labeling. A conditional random field over the same graph encourages local planarity between neighboring surfels in the same directional segment.

World Representation

As already mentioned previously, we describe the environment as a collection of surfels with locations p_i, surface orientations n_i, grey-scale intensity I_i and radius r_i. However, instead of aiming to represent all surfaces in the environment densely, as in related work [97, 133, 244], we opt to sample the surfaces of the environment sparsely with a bias towards high intensity gradient areas for three reasons: (1) a sparse sampling of environment surface captures the majority of surfaces and scene structure, (2) a bias towards high intensity gradient areas captures visually salient regions for camera tracking [67] and (3) sparse sampling facilitates reasoning about the proposed joint map and directional segmentation on current hardware. To substantiate the first point we show the percentage of inlier scene-points to a randomly sampled set of planes as a function of the number of planes in Fig. 5.18 across all 1449 scenes of the NYU v2 dataset [173]. It can be observed that as little as 50 planes are enough to describe an average of 60% of the scene. To compute the plane parameters around 6% of the scene points had to be used in the current approach. If we had a sensor that allowed to directly measure distance and surface normal only 0.002% of image pixels would have had to be examined. This motivates the use of a sparse set of surfels to describe the surfaces of an environment and hints at the potential for efficient algorithm operation.

The directional segmentation as well as mapping are defined via probabilistic models that take into account the local neighborhood of each surfel. Therefore, as depicted in Fig. 5.19, we maintain a nearest neighborhood graph $\mathcal{G}$ for the surfel map for efficient algorithm operation. The initial observed positions of surfels is used for the construction of $\mathcal{G}$ so as to not mix posterior inference over surfel locations and local nearest neighbor structure. This can be understood as conditioning all inference on the nearest

neighborhood graph of surfels. Further implementation details of online graph construction and maintenance are not important for the model and inference description and are left to Sec. 5.2.4.

Stata Center World Directional Segmentation

Under the Stata Center World model we make the assumption that the surface normal distribution of surfels has characteristic, low-entropy patterns (see Chapter 4). Similar to Sec. 4.4.3, we capture the notion of the Stata Center World model, that the surfel surface normal distribution consists of some variable, unknown number of clusters by a Dirichlet process von-Mises-Fisher mixture model. Following the proposal of [182], we impose spatial smoothness of the Stata Center World segmentation by assuming a Markov random field (MRF) over the segmentation z that encourages uniform labeling inside a neighborhood $\mathcal{N}_i$ defined by the nearest neighbor graph $\mathcal{G}$.

From a generative standpoint, this model first samples a countably infinite set of cluster weights w_k, von-Mises-Fisher means μ_k, and concentrations τ_k from a Dirichlet process with concentration parameter α and base measure G_0:

$$\{w_k, \mu_k, \tau_k\}_{k=1}^{\infty} \sim \mathrm{DP}(\alpha, G_0) \tag{5.40}$$

We utilize the conjugate prior for the von-Mises-Fisher distribution (see Sec. 2.6.3) to define the base measure. Second, given the cluster weights w_k and the local neighborhood $\mathcal{N}_i$, a label $z_i \in \{1, \ldots, \infty\}$ is sampled to assign each surfel to a von-Mises-Fisher distribution z_i:

$$z_i \sim \mathrm{MRF}_z\left(z_i, \{z_j\}_{j \in \mathcal{N}_i}; \lambda\right) \mathrm{Cat}\left(\{w_k\}_{k=1}^{\infty}\right) \tag{5.41}$$

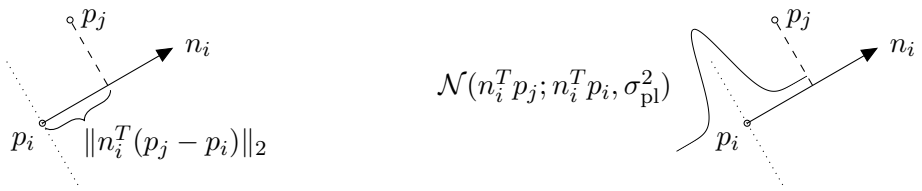
The MRF smoothness component in practice helps speed up inference and leads to more uniform segmentations in the face of noise. It takes the form:

$$\mathrm{MRF}_z\left(z_i, \{z_j\}_{j \in \mathcal{N}_i}; \lambda\right) = \exp\left(\lambda \sum_{j \in \mathcal{N}_i} \mathbb{1}_{z_i = z_j} - \lambda |\mathcal{N}_i|\right), \tag{5.42}$$

where λ is the weight of the MRF contribution. As described in Sec. 5.2.2, we use the Chinese restaurant process (CRP) representation (see Sec. 2.4) to facilitate posterior inference.

Direction-aware Mapping

Following the Stata Center World assumption map surface normals are von-Mises-Fisher distributed according to the parameters of the assigned (via z_i) directional cluster. Conditioned on the surface direction and the directional segmentation we utilize a Markov random field (MRF) over neighboring surfels to express a local planarity assumption over points in the same directional segment. In terms of probability distributions we



(a) Point-to-Plane Cost Function (b) Out-of-Plane Gaussian Distribution

Figure 5.20: Illustration of the point-to-plane cost function $\|n_i^T(p_j - p_i)\|_2$ for a surfel at location p_i with orientation n_i and a point p_j (left). A probabilistic interpretation of this cost function is a Gaussian over the out-of-plane deviation of a point p_j (right).

therefore assume the following factorization of the joint distribution over the surfel-based map:

$$\prod_i p(p_i, n_i \mid p_{\mathcal{N}_i}, n_{\mathcal{N}_i}, z) \propto \prod_i p(n_i \mid \mu_{z_i}, \tau_{z_i}, z_i) \Psi_{ij}^{\text{pl}}(p_i, p_{\mathcal{N}_i}, n_i, n_{\mathcal{N}_i}, z_i, z_{\mathcal{N}_i}) \tag{5.43}$$

$$= \prod_i \text{vMF}(n_i; \mu_{z_i}, \tau_{z_i}) \prod_{j \in \mathcal{N}_i} \left(\mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\text{pl}}^2) \mathcal{N}(n_j^T p_j; n_j^T p_i, \sigma_{\text{pl}}^2) \right)^{\mathbb{1}_{z_i = z_j}} \tag{5.44}$$

where $\mathbb{1}_{z_i=z_j}$ is 1 only if $z_i = z_j$ and 0 otherwise. The set $\mathcal{N}_i$ contains the IDs of the neighboring surfels to surfel i as defined by the nearest neighbor graph $\mathcal{G}$. The MRF term that encapsulates local planarity stems from the well known point-to-plane distance function used in implementations of ICP [207]. This can be seen by analyzing the negative log likelihood of the MRF term for surfel i:

$$-\log \Psi_{ij}^{\text{pl}} = C + \tfrac{1}{2\sigma_{\text{pl}}^2} \left(\|n_i^T(p_j - p_i) + \|_2^2 + \|n_j^T(p_i - p_j)\|_2^2 \right) . \tag{5.45}$$

While the point-to-plane cost function penalizes the out-of-plane deviation of point p_j, the MRF potential employed herein is a symmetriced out-of-plane Gaussian which makes the assumption that the out-of-plane deviation of the point p_j is Gaussian with variance σ_{pl}^2. The geometry of both is illustrated in Fig. 5.20. The variance σ_{pl}^2 can be small since the MRF assumes the symmetriced out-of-plane Gaussian distribution only if two surfels are in the same segment and thus likely to be in the same plane.

Surfel locations and orientations are observed via the camera located at pose ${}^wT_c$. This pose is the maximum likelihood estimate under the observed RGB-D frame as described in Sec. 5.2.3. Associations between observations located at pixels (u, v) in the camera frame and surfels i comprising the map are established by transforming surfels into the camera frame and then projecting them under a given camera model $\pi(\cdot)$ as is commonly done [177]:

$$(u, v) = \pi({}^cT_w p_i) . \tag{5.46}$$

In a second step of occlusion reasoning, we prune associations if the probability of the associated surfel is too low under the observation distribution marginalized over the camera pose uncertainty as described in Sec. 5.2.3. Capturing the camera-frames at which observations of surfel i were taken in the set O_i, we assume an iid Gaussian observation model for locations $\{x_j^p\}_{j \in O_i}$:

$$p\left(\{x_i^p\} \mid p_i\right) = \prod_{j \in O_i} \mathcal{N}(x_j^p; {}^{c,j}T_w p_i, \Sigma_{p,j}) \,. \tag{5.47}$$

where we have used the inferred camera poses $\{{}^{c,j}T_w\}_{j \in O_i}$ at the camera frames the observations were taken. The observation covariances $\Sigma_{p,j}$ are computed according to a realistic depth camera noise model [179] and marginalizing over the camera pose distribution as outlined in Sec. 5.2.3.

The surfel orientation observations $\{x_j^p\}_{j \in O_i}$ are assumed to be iid von-Mises-Fisher distributed:

$$p(\{x_j^n\}_{O_i} \mid n_i; \tau_O) = \prod_{j \in O_i} \text{vMF}(x_j^n; {}^{c,j}R_w n_i, \tau_O) \,, \tag{5.48}$$

where we have used the inferred camera rotations $\{{}^{c,j}R_w\}_{j \in O_i}$. We use the fast yet robust unconstrained scatter-matrix approach to surface normal extraction as outlined in Sec. 2.8. The effect of depth sensor noise on the surface normal estimation uncertainty is hard to analyze analytically. While we simply use a conservative observation concentration of $\tau_O = 100$, a more detailed model could be obtained with a controlled experiment similar to [179]. Additionally, because it is unclear how to marginalize over the rotation uncertainty, we keep τ_O fixed independent of the camera pose distribution. A concentration of $\tau_O = 100$ makes the realistic assumption that 99% of the observed surface normals lie within a solid angle of about 18° around the true surface normal (see Sec. 2.6.3).

■ 5.2.2 Sampling-based Inference over Map and Directional Segmentation

Having outlined the joint mapping and directional segmentation model in the previous section, we now turn to describing how to perform posterior inference on the model given observations $\{x^p, x^n, I_c\}$ from camera inferred camera poses ${}^xT_{c,j}$. As described in Sec. 5.2.1, inference of the map and of the directional segmentation conditions on the camera pose estimation which is explained in the next section.

Because the directional segmentation involves a Bayesian nonparametric Dirichlet process prior, inference can be performed using either MCMC-based techniques or variational inference (see Sec. 2.4). Here we choose to rely on Gibbs sampling, which in the limit of sampling guarantees samples from the true posterior distribution and allows the approximate computation of any expectation over the data such as the mean, uncertainty, or probabilities (see Sec. 2.2.2).

As outlined in Algorithm 16 the Gibbs sampler iterates sampling from the different conditional distributions of each random variable belonging to directional segmentation and surfel map. In the following we provide details on each of the conditional distributions and give guidance as to how to sample from them before detailing which expectations are computed from the samples to inform camera tracking.

Gibbs-sampling Conditionals

while more samples desired **do**
  Update set of surfels, neighborhoods, and surfel observations
  Sample surface normals $n_i \sim p(n_i \mid p_i, p_{N_i}, \mu_{z_i}, \tau_{z_i}, \{x_j^n\}_{j \in O_i})$ (Eq. (5.58))
  Sample directional segmentation $z_i \sim p(z_i \mid n_i, \mu, \tau; \alpha)$ (Eq. (5.59))
  Sample directional clusters $\mu_k, \tau_k \sim p(\mu_k, \tau_k \mid n, z, \mu_0, a, b)$ (Eq. (5.62))
  Sample surfel locations $p_i \sim p(p_i \mid n_i, p_{N_i}, z_i, z_{N_i})$ (Eq. (5.69))
  Update maximum likelihood estimates $\hat{p}_i$ and $\hat{n}_i$ for camera tracking
  Update other statistics such as covariance estimates and entropy
end while

Algorithm 16: Gibbs-sampling algorithm for joint inference over surfel-based map and directional segmentation.

Sampling Normals n_i Via Bayes' law, the conditional distribution of surfel direction n_i is proportional to the joint distribution over the position and surface normal of surfel i (Eq. (5.43)) times the observation model (Eq. (5.48)):

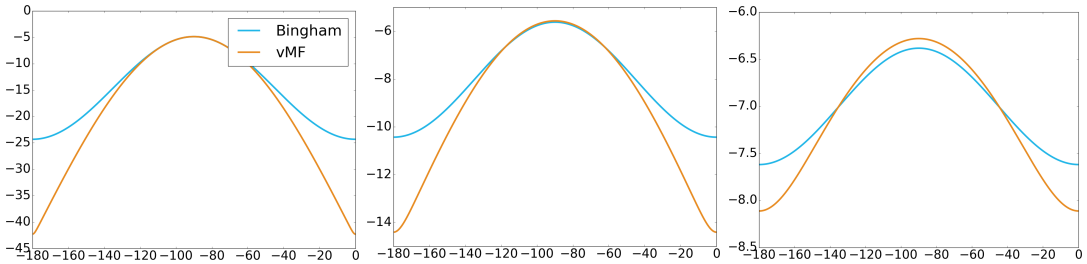
$$\begin{aligned} p(n_i \mid \{x_j^n\}_{O_i}, z, \mu, \tau) &\propto p(n_i, p_i \mid \mu_{z_i}, \tau_{z_i}, p_{\mathcal{N}_i}, z) p\left(\{x_j^n\}_{O_i} \mid n_i, \tau_O\right) \\ &\propto \mathrm{vMF}(n_i; \mu_{z_i}, \tau_{z_i}) \prod_{j \in \mathcal{N}_i} \mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\mathrm{pl}}^2)^{\mathbb{1}_{z_i = z_j}} \prod_{j \in O_i} \mathrm{vMF}(x_j^n, {}^{c,j}R_w n_i, \tau_O) \,. \end{aligned} \tag{5.49}$$

The first factor stems from the directional Stata Center World mixture model, the second from the MRF capturing planarity of neighboring points, and the third from the surface normal observation model. At first glance it is not obvious how to sample from this distribution efficiently because of the out-of-plane Gaussian distributions:

$$\textstyle\prod_{j \in \mathcal{N}_i} \mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\mathrm{pl}}^2)^{\mathbb{1}_{z_i = z_j}} \,. \tag{5.50}$$

By analyzing this distribution we derive a close approximation that has the form of a vMF distribution which in turn makes the posterior over surface normals von-Mises-Fisher distributed and straightforward and efficient to sample.

The Gaussian distribution on out-of-plane deviations of neighboring points can be



(a) Stds of $[0.2, 0.2, 0.001]$ (b) Stds of $[0.1, 0.1, 0.001]$ (c) Stds of $[0.05, 0.05, 0.001]$

Figure 5.21: Comparison of the true Bingham distribution around one of its two modes and the approximation with a von-Mises-Fisher distributions for different standard deviations of the Bingham distribution (in log scale). The distributions shown are evaluated on a great circle around the sphere leading through both antipodal modes. Note that for small standard deviations, both distributions are essentially uniform over the sphere as can be seen from the scale of the plot. For larger standard deviations, the approximation is close to the true distribution around the mode.

re-arranged as

$$\prod_{j\in\mathcal{N}_i} \mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\text{pl}}^2)^{\mathbb{1}_{z_i=z_j}} \propto \exp\left(\tfrac{1}{2}\sum_{j\in\mathcal{N}_i} \frac{-\mathbb{1}_{z_i=z_j}}{\sigma_{\text{pl}}^2} \|n_i^T(p_j - p_i)\|_2^2\right) \tag{5.51}$$

$$\propto \exp\left(-\tfrac{1}{2} n_i^T \sum_{j\in\mathcal{N}_i} \frac{\mathbb{1}_{z_i=z_j}}{\sigma_{\text{pl}}^2}(p_i - p_j)(p_i - p_j)^T n_i\right) \tag{5.52}$$

$$= \exp\left(-\tfrac{1}{2} n_i^T S_{ij} n_i\right) . \tag{5.53}$$

This distribution has the form of a Bingham distribution [23]. To keep in the realm of the von-Mises-Fisher distribution, we approximate this Bingham with a vMF distribution using the eigen decomposition of S_{ij} with eigenvalues $e_1 < e_2 < e_3$ and associated eigenvectors q_1, q_2, q_3:

$$\exp(-\tfrac{1}{2} n_i^T S_{ij} n_i) \approx \exp\left(\frac{2e_2e_2}{e_2+e_3} q_1^T n_i\right) \propto \text{vMF}(n_i; q_1, \tfrac{2e_2e_2}{e_2+e_3}) . \tag{5.54}$$

The Bingham can model more complex distributions than the vMF and it would be interesting to extend this model and the inference to use Bingham instead of vMF distributions. However, the vMF is sufficiently expressive to capture flat and uniformly round surface geometries. The approximation is very close to the Bingham for planar surfaces with a peaked surface normal distribution. At corners the Bingham can capture the resulting anisotropy in the surface normal distribution while the isotropic vMF distribution has to approximate the true density with a smaller concentration. In practice, since S_{ij} incorporates only neighbors in the same directional segment (which are therefore likely to lie roughly in the same plane), we find the approximation to work well. Figure 5.21 shows the approximation for several realistic standard deviations of the Bingham distribution. The derivation can be found in Appendix D.0.3.

Under this approximation the posterior over surface normal n_i is indeed von-Mises-Fisher:

$$p(n_i \,|\{x_j^n\}_{O_i}, z, \mu, \tau, \tau_O) \propto p\left(\{x_j^n\}_{O_i} \mid n_i, \tau_O\right) p(n_i \mid p_i, p_{\mathcal{N}_i}, \mu_{z_i}, \tau_{z_i}) \tag{5.55}$$

$$\propto \exp\left(\textstyle\sum_{j\in O_i} n_i^T {}^w R_{c,j} x_j^n \tau_{O,j} + n_i^T \mu_{z_i}\tau_{z_i} - \frac{1}{2} n_i^T S_{ij} n_i\right) \tag{5.56}$$

$$= \exp\left(n_i^T \left(\textstyle\sum_{j\in O_i} {}^w R_{c,j} x_j^n \tau_{O,j} + \mu_{z_i}\tau_{z_i} + \frac{2e_2e_2}{e_2+e_3} q_1\right)\right) \tag{5.57}$$

$$\propto \text{vMF}(n_i; \lceil\vartheta\rceil, \|\vartheta\|_2) \text{ where } \vartheta = \textstyle\sum_{j\in O_i} {}^w R_{c,j} x_j^n \tau_{O,j} + \mu_{z_i}\tau_{z_i} + \frac{2e_2e_2}{e_2+e_3} q_1\,. \tag{5.58}$$

An efficient method for sampling from a von-Mises-Fisher distribution is outlined in Sec. 2.6.3.

Sampling Directional Segmentation Labels z_i We use the Chinese restaurant process (CRP) representation of the Dirichlet process since it lends itself to straightforward sampling-based inference. While other approaches have been proposed such as the sub-cluster split/merge algorithm [42, 224] we find that the CRP inference converges sufficiently fast when combined with the MRF for label smoothness. The posterior for the directional segmentation label of surfel i is:

$$p(z_i \mid \pi, \{z_j\}_{j\in\mathcal{N}_i}, \mu, \tau) \propto \text{MRF}_z(\mathcal{N}_i, \lambda) \begin{cases} N_k \,\text{vMF}(n_i; \mu_k, \tau_K) & \text{for } k \le K \\ \alpha p(n_i; G_0) & \text{for } k = K+1 \end{cases} \tag{5.59}$$

$$\text{MRF}_z(\mathcal{N}_i, \lambda) \propto \exp\left(\lambda \textstyle\sum_{j\in\mathcal{N}_i} \mathbb{1}_{z_i=z_j} - \lambda|\mathcal{N}_i|\right)\,, \tag{5.60}$$

where N_k is the number of surfels associated to cluster k excepting the ith surfel, α is the Dirichlet process concentration and λ is the weight of the MRF contribution. The marginal distribution of surface normal n_i under the prior on the vMF component distribution, $p(n_i; G_0)$, can be derived in closed form for the vMF prior parameters $a = 1$ and $0 < b < 1$ in $D = 3$ dimensions (see Sec. 2.6.3):

$$p(n_i; \mu_0, a = 1, b) = \frac{b\tau}{2\pi^2} \frac{\exp\left(b\tau\mu^T\mu_0\right)}{\tan\left(\frac{b\pi}{2}\right)\sinh(\tau)}\,. \tag{5.61}$$

The parameters of the prior are the directional mode μ_0 and a and b where a can be understood as pseudo-counts and b as the concentration mode. In practice z_i is in $\{1, \ldots, K\}$ (and not infinite as the DP prior formulation might suggest), thus making it possible to compute the unnormalized probability density value of Eq. 5.59 for each $k \in \{1, \ldots, K+1\}$. After normalization such that the values sum to 1, the inversion method is used to sample from this categorical distribution.

Sampling von-Mises-Fisher Mixture Component Parameters μ_k and τ_k Given sampled normals n_i assigned to von-Mises-Fisher clusters via labels z_i the posterior over the kth vMF mixture component mode μ_k and concentration τ_k is:

$$p(\mu_k, \tau_k \mid n, z; \mu_0, b, a = 1) \propto p(\mu_k, \tau_k; \mu_0, b, a = 1) \prod_{i\in\mathcal{I}_k} p(n_i \mid \mu_k, \tau_k) \tag{5.62}$$

$$\propto p(\mu_k, \tau_k \mid \tilde{\mu}_0^k, \tilde{a}_k, \tilde{b}_k)\,, \tag{5.63}$$

where $\mathcal{I}_k$ collects all surfels associated to cluster k. The updated parameters $\tilde{a}_k$ and $\tilde{b}_k$ are computed as

$$\tilde{a}_k = a + |\mathcal{I}_k|\,, \qquad \tilde{b}_k = \|\vartheta\|_2\,, \qquad \tilde{\mu}_0^k = \lceil \vartheta \rceil\,, \qquad \vartheta = \sum_{i \in \mathcal{I}_k} n_i + b\mu_0\,. \tag{5.64}$$

See Sec. 2.6.3 for more details.

Sampling Locations p_i Conditioned on point observations $\{x_j^p\}_{j \in O_i}$, and a surfel's neighborhood $\mathcal{N}_i$, its position is distributed as:

$$p(p_i \mid \{x_i^p\}, n_i, p_{\mathcal{N}_i}, z) \propto \prod_{j \in \mathcal{N}_i} \Psi_{ij}^{\mathrm{pl}} \prod_{j \in O_i} p(x_j^p \mid p_i) \tag{5.65}$$

$$\propto \prod_{j \in \mathcal{N}_i} (\mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\mathrm{pl}}^2)\, \mathcal{N}(n_j^T p_j; n_j^T p_i, \sigma_{\mathrm{pl}}^2))^{\mathbb{1}_{z_i = z_j}} \prod_{j \in O_i} \mathcal{N}(x_j^p; {}^{c,j}T_w p_i, \Sigma_{p,j})\,, \tag{5.66}$$

where the observation model is Gaussian with a observation dependent covariance $\Sigma_{p,j}$, as described in Sec. 5.2.3. The symmetriced out-of-plane Gaussian distribution in the MRF potential may be rearranged as

$$\mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\mathrm{pl}}^2)\, \mathcal{N}(n_j^T p_j; n_j^T p_i, \sigma_{\mathrm{pl}}^2) \propto \exp\left(\tfrac{-1}{2\sigma_{\mathrm{pl}}^2} p_i^T (n_i n_i^T + n_j n_j^T) p_i \right. \tag{5.67}$$

$$\left. + \tfrac{1}{\sigma_{\mathrm{pl}}^2} p_i^T (n_i n_i^T + n_j n_j^T) p_j \right) \tag{5.68}$$

to reveal a degenerate Gaussian. This can be handled in information form with information matrix $I_{ij} = \frac{1}{\sigma_{\mathrm{pl}}^2}(n_i n_i^T + n_j n_j^T)$ and the scaled mean $I_{ij} p_j$. Since the individual distributions are all Gaussian the posterior over surfel location p_i is also Gaussian [34] with the following mean and variance:

$$p(p_i \mid \{x_i^p\}, n_i, p_{\setminus i}, z) \propto \mathcal{N}(p_i; \tilde{\mu}_i, \tilde{\Sigma}_i) \tag{5.69}$$

$$\tilde{\Sigma}_i = \left(\textstyle\sum_{j \in O_i} {}^{w}R_{c,j} \Sigma_{p,j}^{-1 c,j} R_w + \sum_{j \in \mathcal{N}_i} \mathbb{1}_{z_i = z_j} I_{ij} \right)^{-1} \tag{5.70}$$

$$\tilde{\mu}_i = \tilde{\Sigma}_i \left(\textstyle\sum_{j \in O_i} {}^{w}R_{c,j} \Sigma_{p,j}^{-1 c,j} R_w {}^{w}T_{c,j} x_j^p + \sum_{j \in \mathcal{N}_i} \mathbb{1}_{z_i = z_j} I_{ij} p_j \right)\,. \tag{5.71}$$

Note that there is always at least one observation associated with a point ($(|O_i| > 0)$ and therefore the inversion to compute the variance is always determined. For efficiency reasons we keep track of the sums over observations and do not keep around all individual observations. The downside to this approach is that the pose from which observations were taken are baked into the sums and cannot be updated given later pose corrections. Since we do not currently re-estimate older camera poses, this is not a problem.

Expectations and Estimates Computed from Samples

In the limit of sampling from the Markov chain constructed via the Gibbs-sampler described in the first part of this section, samples are by construction guaranteed to be drawn from the true joint posterior distribution over surfel map and directional segmentation. In the Bayesian approach these samples are used to perform Monte Carlo integration of arbitrary expectations of the random variables (see Sec. 2.2). In the limit Monte Carlo integration converges to the true expectation via the law of large numbers:

$$\mathbb{E}_x\left[f(x)\right] = \int_{\mathbb{X}} f(x)p(x)\,\mathrm{d}x = \lim_{N\to\infty}\frac{1}{N}\sum_{i=1}^{N} f(\xi_i) \text{ where } \xi_i \sim p(x)\,. \tag{5.72}$$

While in practice we only have a finit amount of samples that are only approximately drawn from the true distribution since we sample only for a finite amount of time, we can still compute useful approximate expectations from them. Intuitively, since in this application most marginal distributions $p(p_i)$ or $p(n_i)$ are strongly concentrated about a single mode, the estimation converges quite quickly. In our experiments in the order of ten samples were sufficient to get usable estimates for real-time camera tracking as described in Sec. 5.2.3.

Surfel Location p_i Given a set of samples $\{\xi_j^p\}_{\mathcal{S}_i}$ from the distribution of surfel locations p_i, we keep track of the sufficient statistics of a Gaussian:

$$|\mathcal{S}_i|\,, \qquad \tilde{p}_i = \textstyle\sum_{j\in\mathcal{S}_i}\xi_j^p\,, \qquad O_{p_i} = \textstyle\sum_{j\in\mathcal{S}_i}\xi_j^p\xi_j^{pT}\,. \tag{5.73}$$

This allows the estimation of mean and variance of the surfel location:

$$\bar{p}_i = \mathbb{E}_{p_i}\left[p_i\right] \approx \tfrac{\tilde{p}_i}{|\mathcal{S}_i|}\,, \qquad \bar{\Sigma}_{p_i} = \mathrm{Var}\left(p_i\right) \approx \tfrac{1}{|\mathcal{S}_i|}O_{p_i} - \tilde{p}_i\tilde{p}_i^T\,. \tag{5.74}$$

Given the first two moments of the underlying true distribution of p_i, the maximum entropy distribution is Gaussian with those moments. Therefore, the entropy computed from these estimates is an upper bound on the true entropy of the point distribution:

$$H(p_i) \leq \tfrac{3}{2}\log(2\pi e) + \tfrac{1}{2}\log|\bar{\Sigma}_{p_i}|\,. \tag{5.75}$$

This computable bound on entropy serves as a scalar indicator of uncertainty in the estimate of the surfel location. It is crucial to realize that samples from the distribution of p_i are not samples from a Gaussian distribution, even though the conditional distribution in the Gibbs sampler is.

For camera pose estimation as outlined in Sec. 5.2.3, the estimated means and covariances are used once the entropy of p_i has converged. Before that we use the initial point location and uncertainty. Since these estimates converge within a few frames, the procedure is similar to the delayed addition of points to the map commonly employed in related 3D reconstruction systems.

Surfel Orientation n_i From the surfel orientation samples $\{\xi_j^n\}_{j\in O_i}$ we accumulate the following statistics:

$$|\mathcal{S}_i|\,, \qquad \tilde{n}_i = \textstyle\sum_{j\in\mathcal{S}_i} \xi_j^n\,. \tag{5.76}$$

This allows us to compute the mode of a von-Mises-Fisher distribution as the surface normal estimate to be used for camera tracking purposes:

$$\bar{n}_i = \lceil \tilde{n}_i \rceil\ . \tag{5.77}$$

We do not currently use the fact that these statistics are sufficient to estimate the vMF concentration (see Sec. 2.6.3) and the entropy of the surfel orientation.

Surfel Segment Label z_i To compute the most likely directional segment of surfel i we would ideally keep a count of the number of times the surfel is assigned to each directional cluster via label z_i. Since the number of clusters keeps growing and we aim for this estimation to be efficient for large numbers of surfels, we only keep track of the $\tilde{K} = 3$ most likely cluster assignments incrementally. The most likely surfel segment is used by the camera tracker to guide measurement selection as described in Sec. 5.2.3.

Surfel Intensity Value I_i Another property that we estimate using surfel position samples is the gray-scale and RGB intensity of the surfel. Given camera poses the expected surfel intensity is

$$\bar{I}_i = \mathbb{E}_{p_i}\left[I_c(\pi({}^{c,j}T_w p_i))\right] \approx \tfrac{1}{|\mathcal{S}_i|}\textstyle\sum_{j\in\mathcal{S}_i} I_c(\pi({}^{c,j}T_w \xi_j^p)) \tag{5.78}$$

In practice we collect the sum over observed intensity values for the current sample ξ_i^p across all frames that observed p_i to estimate the intensity. Note that the common approach of computing the average over the intensity of the most likely surfel location (as opposed to the sample locations) does not capture the inherent uncertainty of the point location. In fact using the surfels, we could compute the variance of the surfel intensity as implied by uncertainty in its location.

■ 5.2.3 Direction-aware Camera Pose Estimation

After having explained Gibbs-sampling inference for the joint distribution over map and directional segmentation, and how to compute estimates for surfel locations, orientations and segment labels, we now turn to realtime localization. Note that while we use the familiar notation for surfel properties, in practice sample-based estimates of the quantities are used. These are computed as described in the previous section.

For a given association $\mathcal{A}$ between observations $\{x^p, I_c\}$ and surfels the posterior over the camera pose ${}^wT_c$ under the proposed model is proportional to the likelihood of point observations x^p and photometric intensities in the form of the current gray-scale image I_c as:

$$\begin{aligned} p({}^wT_c \mid \{s_i\}, \{x^p, I_c\}) &\propto p(\{x^p, I_c\} \mid {}^wT_c, \{s_i\}) && (5.79)\\ &= p(\{x^p\} \mid {}^wT_c, \{p_i, n_i\})p(\{I_c\} \mid {}^wT_c, \{I_i, p_i\}) && (5.80)\\ &= \textstyle\prod_{i\in\mathcal{A}} \mathcal{N}(n_i^{T\,w}T_c x_i^p; n_i^T p_i, \sigma_{\text{pl}}^2)\,\mathcal{N}(I_c(\pi({}^wT_c{}^{-1}p_i)); I_i, \sigma_I)^{\lambda_I}\,, && (5.81)\end{aligned}$$

where $\pi(\cdot)$ is the camera model projection function from 3D point in camera coordinates to the image space. The form of this camera pose posterior encapsulates the assumption that the point-to-plane deviations and the photometric errors are Gaussian distributed conditioned on the camera transformation ${}^{w}T_c$. The parameter λ_I weighs the contribution of the photometric with respect to the geometric term.

In general the posterior in Eq. (5.79) might be multi-modal and hard to characterize fully. Any global localization scheme would essentially seek to explore all modes in order to detect what is commonly called loop closure. We will use the iterative closest point (ICP) algorithm [207] to locally find the most likely camera pose starting from a previous pose. ICP alternates between updating the associations $\mathcal{A}$ between map and observations and updating the camera pose. The associations are obtained by rendering the surfel map from the current camera pose estimate [177]. ICP can be understood as performing alternating joint optimization over data association and camera pose. The success of this local camera tracking method shows that the posterior distribution generally does have a clear mode close to the mode characterized by the previous frame.

For each observed RGBD frame we run the iterative closest point algorithm to find a local optimum of the camera pose as well as data association between the observations and the global map surfels. The optimization of the camera pose given data association amounts to maximizing the negative log likelihood of the camera pose:

$$ {}^{w}T_c^{\star} = \underset{{}^{w}T_c \in \mathbb{SE}(3)}{\arg\min}\; f_{\text{p2pl}}({}^{w}T_c) + \lambda_I f_{\text{photo}}({}^{w}T_c) \tag{5.82} $$

This cost function is the commonly used combined point-to-plane (p2pl) and photometric (photo) cost function [250]. Both cost functions $f_{(\cdot)}$ can be formulated as sums over squared error terms.

One strategy to derive the maximum likelihood camera pose estimate is to Taylor-expand the error terms around the current transformation estimate:

$$ f_{(\cdot)} = \sum_{i\in\mathcal{A}} \|e_i(T)\|_2^2 = \sum_{i\in\mathcal{A}} \|e_i(T) + \tfrac{\partial}{\partial\omega} e_i(T\text{Exp}(\omega))\omega\|_2^2 = \|J\omega - b\|_2^2 \,, \tag{5.83} $$

where we have collected the individual derivatives and error terms into the rows of J and b respectively. The variable $\omega \in \mathfrak{se}(3)$ is a small perturbation of the transformation in the tangent space to $\mathbb{SE}(3)$ at the current estimate of the transformation T. Specifically, row i in J and b have the values:

$$ J_i = \tfrac{\partial}{\partial\omega} e_i(T\text{Exp}(\omega)) \,, \qquad\qquad b_i = -e_i(T) \,. \tag{5.84} $$

Then the least-squares solution for the (small) motion ω along the manifold $\mathbb{SE}(3)$ is

$$ \omega = (J^T J)^{-1} J^T b \,. \tag{5.85} $$

An efficient implementation will exploit that $J^T J \in \mathbb{R}^{6\times 6}$ and $J^T b \in \mathbb{R}^6$ can be computed by accumulating the contributions of the individual error terms as

$$ J^T J = \sum_{i\in\mathcal{A}} J_i^T J_i \,, \qquad\qquad J^T b = \sum_{i\in\mathcal{A}} J_i^T b_i \,. \tag{5.86} $$

The contributions of the different cost functions are weighted according to different weights (here λ_I) and accumulated into a single J^TJ and J^Tb to solve for the overall most likely pose change ω via Eq. (5.85). The pose is updated by taking a ω step in the tangent space of $\mathbb{SE}(3)$ and mapping back onto $\mathbb{SE}(3)$ via the exponential map (see Sec. 2.7.2):

$$T_{t+1} = \mathrm{Exp}_{T_t}(\omega) = T_t \mathrm{Exp}(\omega)\,. \tag{5.87}$$

In the following we give the error functions and derivatives for the different cost function terms.

Point-to-Plane Alignment Contribution The point-to-plane cost function penalizes the out of plane deviation of an observed point ${}^wT_c x_i^p$. The surfel is located at p_i and has orientation n_i:

$$f_{\mathrm{p2pl}} = \sum_{i\in\mathcal{A}} \frac{1}{\sigma^2_{\mathrm{p2pl},i}} \| n_i^T ({}^wT_c x_i^p - p_i) \|_2^2 \tag{5.88}$$

$$= \sum_{i\in\mathcal{A}} \frac{1}{\sigma^2_{\mathrm{p2pl},i}} \| n_i^T ({}^wT_c x_i^p - p_i) + n_i^T \frac{\partial}{\partial\omega} ({}^wT_c \mathrm{Exp}(\omega) x_i^p)\,\omega \|_2^2\,. \tag{5.89}$$

Following the aforementioned strategy we can derive the rows of J and b as:

$$J_i = \frac{1}{\sigma_{\mathrm{p2pl},i}} \begin{pmatrix} -[x_i^p]_\times {}^Tc R_w n_i & {}^cR_w n_i \end{pmatrix} = \frac{1}{\sigma_{\mathrm{p2pl},i}} \begin{pmatrix} x_i^p \times ({}^cR_w n_i) & {}^cR_w n_i \end{pmatrix} \tag{5.90}$$

$$b_i = -\frac{1}{\sigma_{\mathrm{p2pl},i}} n_i^T ({}^wT_c x_i^p - p_i)\,. \tag{5.91}$$

See Appendix D.0.4 for the full derivation.

Photometric Alignment Contribution The photometric error is the difference between the surfel intensity I_i at the surfel location p_i in world coordinates and its current observed intensity I_c. The current estimate of the camera pose ${}^wT_c$ is used to transform the point into the camera frame before projecting it into the image plane using the camera projection function $\pi(\cdot)$.

$$f_{\mathrm{photo}} = \sum_{i\in\mathcal{A}} \frac{1}{\sigma_I^2} \| I_c(\pi({}^cT_w p_i)) - I_i \|_2^2 \tag{5.92}$$

$$= \sum_{i\in\mathcal{A}} \frac{1}{\sigma_I^2} \| I_c + \nabla I_c \frac{\partial \pi(x)}{\partial x} \frac{\partial ({}^wT_c \mathrm{Exp}(\omega))^{-1} p_i}{\partial \omega} \omega - I_i \|_2^2\,. \tag{5.93}$$

Again, the rows of b can be read of directly whereas the derivation of the rows of J is more involved and left to Appendix D.0.5:

$$J_i = \frac{1}{\sigma_I} \nabla I_c \frac{\partial \pi(p)}{\partial p} \begin{pmatrix} \left[{}^wR_c^T (p_i - {}^wt_c) \right]_\times & -\mathrm{I} \end{pmatrix} \tag{5.94}$$

$$b_i = \frac{1}{\sigma_I} (I_i - I_c)\,. \tag{5.95}$$

The image gradient ∇I_c is computed numerically using convolution. The derivative of the camera projection under the pinhole camera model is:

$$\frac{\partial \pi(p)}{\partial p} = \begin{pmatrix} \frac{f_u}{p_z} & 0 & -\frac{p_x}{p_z^2} f_u \\ 0 & \frac{f_v}{p_z} & -\frac{p_y}{p_z^2} f_v \end{pmatrix} . \tag{5.96}$$

Note that more complicated camera models with distortion can also be utilized by using the appropriate projection function.

Uncertainty of the ICP Transformation Estimate As introduced in this section, ICP seeks to maximize the negative log likelihood of the data under the pose of the camera. As shown in Eq. (5.85) the maximum likelihood estimator is equivalent to the least squares estimate $\omega = (J^T J)^{-1} J^T b$. The Fisher information matrix of the estimator is $J^T J$, as previously noted by Kerl et al. [135]. The Cramer-Rao bound [51] then states that the actual variance of the estimate is lower-bound by the inverse of the Fisher information matrix:

$$\text{Var}\,(\omega) \geq (J^T J)^{-1} \triangleq \tilde{\Sigma}_{\text{ICP}} . \tag{5.97}$$

This means that independent of the pose estimate, purely due to the observe data, the variance of the estimate ω can never decrease below $(J^T J)^{-1}$. Ignoring higher-order moments, we may understand ω as distributed according to a Gaussian with mean $(J^T J)^{-1} J^T b$ and variance of at least $(J^T J)^{-1}$. This means the entropy of the estimate can be lower-bound via

$$H(\omega) \geq H(\mathcal{N}((J^T J)^{-1} J^T b, (J^T J)^{-1}) \tag{5.98}$$

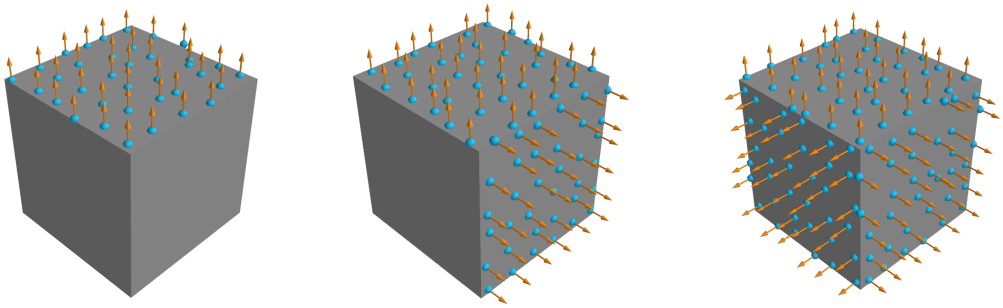
$$\geq 3 \log(2\pi e) - \tfrac{1}{2} \log\left(\left|J^T J\right|\right) . \tag{5.99}$$

The task of the perception system is then to improve the lower-bound on the true variance, and entropy to enable more certain estimates (even if the estimate might not actually be more certain). To save computational resources, one would like to select the smallest set of observations to minimize the entropy lower-bound. Since this is a hard problem to solve exactly, most approaches rely on greedy strategies of sequentially picking the next observation J_i that minimizes the conditional entropy given the set of already chosen observations J_N. The conditional entropy of a set of data comprised in J_N and a new J_i is:

$$H(J_i \mid J_N) = H(J_{N+i}) - H(J_N) = -\tfrac{1}{2} \log\left(\left|J_{N+i}^T J_{N+i}\right|\right) + \tfrac{1}{2} \log\left(\left|J_N^T J_N\right|\right) \tag{5.100}$$

$$= \tfrac{1}{2} \log\left(\frac{\left|J_N^T J_N\right|}{\left|J_{N+i}^T J_{N+i}\right|}\right) = \tfrac{1}{2} \log\left(\frac{\left|\sum_j^N J_j^T J_j\right|}{\left|\sum_j^N J_j^T J_j + J_i^T J_i\right|}\right) \tag{5.101}$$

$$= -\tfrac{1}{2} \log\left(1 + J_i \left(\sum_j^N J_j^T J_j\right)^{-1} J_i^T\right) . \tag{5.102}$$



2 translation
1 rotation unconstrained

1 translation
unconstrained

fully constrained

Figure 5.22: Intuition about constraints on the ICP cost function depending on the geometry of the map.

Note that J_i is a column vector. An efficient incremental technique for computing the inverse of $\sum_j^N J_j^T J_j = C$ after adding $J_i^T J_i = D$ was described by Miller [166]:

$$(C + D)^{-1} = C^{-1} - \frac{1}{1+\text{tr}(DC^{-1})} C^{-1} D C^{-1} \,. \tag{5.103}$$

Even the greedy approach is still computationally expensive since it requires the evaluation of all conditional entropies for each next data point choice. Instead we choose to randomly select a set of planes with diverse directional distribution and strong texture gradients as outlined next.

Direction-aware ICP Here we analyze the point-to-plane cost function for ICP given in Eq. (5.88) to motivate the use of direction-aware ICP. As depicted in Fig. 5.22, if the map consists of just a single plane, there are three unconstrained degrees of freedom in the point-to-plane cost function: rotation about the surface normal direction, and in-plane translation. Adding a second plane with non-negligible orientation difference constrains the solution up to one degree of freedom in translation along the direction of the plane intersection. Once the map consists of at least three planes with different orientation the point-to-plane cost function is fully constrained. Intuitively, this suggest that choosing a diverse set of orientations among the planes used for ICP leads to a well constrained local optimum and hence camera tracking.

We utilize the directional segmentation of the map under the Stata Center World model to guide the plane measurement selection for ICP. Instead of picking plane observations uniformly from the depth image (either all or at random), we pick plane observations uniformly across the different observable directional segments. Additionally, we sort the surfels in each directional segment by their image gradient strength to bias towards visually distinct surfels which are expected to help photometric alignment.

From an uncertainty standpoint the intuition is that to reduce uncertainty in all six pose parameters, we would like to add measurements such that the eigenvalues

of the information matrix $J^T J$ are uniformly increased. One simple (yet suboptimal) strategy to achieve this is to add a diverse set of $J_i^T J_i$ with large magnitude to the information matrix. Due to the strong dependence of the J_i of the point-to-plane cost function on the surface normal orientation (see Eq. (5.90)) choosing a directionally diverse set of surfel observations contributes more to decrease the uncertainty uniformly than choosing surfels with less diversity. For example, if only surfels with one or two distinct directions are used the information matrix of the point-to-plane cost function is singular. This is because one or two eigenvalues of the translation component remain 0: $(J^T J)_{\text{trans}} = Nnn^T$ is rank one and hence has two zero eigenvalues and $(J^T J)_{\text{trans}} = N_a n_a n_a^T + N_b n_b n_b^T$ for two different directions n_a and n_b is rank two and thus has one zero eigenvalue.

For the photometric contribution to the Fisher information matrix, while there certainly is a dependence on the orientation of the image gradient, a simpler dependency is on the norm of the image gradient. Since it is computationally expensive to perform directional analysis of the image gradient statistics each frame, we instead simply utilize the image gradient norm by sorting all putative surfels in a directional segment by the image gradient norm. High gradients contribute more information to the Fisher information matrix as may be verified in Eq. (5.94). We observe that this combined information driven surfel observation selection strategy yields accurate and efficient camera tracking as described in Sec. 5.2.5.

Our variant of ICP, outlined in Algorithm 17, incrementally adds planes to the cost function until low enough uncertainty is reached. Planes are chosen in a round-robin style from each of the Stata Center World segments in order of decreasing surfel texture gradient strength. We only update the entropy every K (the number of currently instantiated DP-vMF clusters) new data points to determine convergence of ICP. The second criteria used is a threshold on the smallest eigenvalue of the Fisher information matrix. Since the inverse of the Fisher information matrix lower-bounds the true variance of the estimator, this criterion enforces that the largest best-case variance is below some threshold. We find that this second criterion is effective in ensuring efficient yet high quality camera tracking. Similar to the approach by Dellaert et al. [58], the proposed ICP variant selectively integrates informative observations. As in [58] this decreases the number of necessary observations in practice and thus speeds up camera tracking.

Uncertainty Propagation into Observation Uncertainty The inverse of the Fisher information matrix is only a lower bound on the actual variance on the camera pose. We nevertheless propagate this uncertainty into the measurement uncertainty. Incorporating only the best-case uncertainty will lead to overconfident inference, but it is better than not propagating uncertainty and thus assuming full certainty.

The observation of surfel location p_i in the camera coordinate system via the point x_j^p, is affected by two noise sources: (1) the noise of the sensing process captured by Gaussian noise with variance $\Sigma_{O,j}$ computed according to a realistic depth camera model and (2) uncertainty $\Sigma_{{}^w T_c}$ in the camera pose estimate ${}^w T_c$. Due to the formulation of

```
Obtain list of observable surfels by project all surfels into current camera view
while ICP not converged do
    k = 0
    while uncertainty too large do
        Pick surfel with the next lower gradient norm from segment k
        if plane passes occlusion reasoning then
            Add to ICP J^T J and J^T b
            Update entropy
        end if
        k = (k + 1)%K
    end while
    Compute transformation update
end while
```

Algorithm 17: Direction-aware incremental ICP. Surfels are added to the ICP estimator incrementally until a threshold on the uncertainty is reached. The selection of surfels is performed uniformly across the different directional segments and biased towards high image gradient surfels.

ICP the uncertainty in the pose estimate is expressed as a Gaussian in the tangent space to $\mathbb{SE}(3)$ at ${}^{w}T_c$. Therefore the noise model is:

$$\omega \sim \mathcal{N}(0, \Sigma_{{}^{w}T_c}) \tag{5.104}$$

$$\epsilon \sim \mathcal{N}(0, \Sigma_{O,j}) \tag{5.105}$$

$$x_j^p = (\mathrm{Exp}(\omega)\,{}^{w}T_c)^{-1} p_i + \epsilon\,. \tag{5.106}$$

By linearising around the current pose estimate we obtain:

$$x_j^p = {}^{c}T_w p_i + \tfrac{\partial}{\partial \omega}\left(({}^{w}T_c \mathrm{Exp}(\omega))^{-1} p_i\right)\big|_{\omega=0} \omega + \epsilon \tag{5.107}$$

$$= {}^{c}T_w p_i + \begin{pmatrix} \left[{}^{w}R_c^T (p_i - {}^{w}t_c)\right]_\times & -\mathrm{I} \end{pmatrix} \omega + \epsilon \tag{5.108}$$

$$= {}^{c}T_w p_i + J_{{}^{w}T_c}\omega + \epsilon\,. \tag{5.109}$$

From this we read off the distribution of x_j^p:

$$x_j^p \sim \mathcal{N}({}^{c}T_w p_i, \Sigma_{O,j} + J_{{}^{w}T_c} \Sigma_{{}^{w}T_c} J_{{}^{w}T_c}^T)\,. \tag{5.110}$$

Therefore we add to the noise model of the RGB-D camera the projected covariance matrix of the camera pose estimate.

A surface normal observation x_j^n is similarly exposed to noise from the sensing process as well as to noise of the camera pose rotation estimate. There is no straightforward derivation for how the camera rotation noise influences the observation concentration τ_O. This derivation is hard because of the different distributions involved: Gaussian sensing noise, Gaussian camera pose noise and von-Mises-Fisher noise on the surface normals. We therefore simply choose τ_O conservatively low.

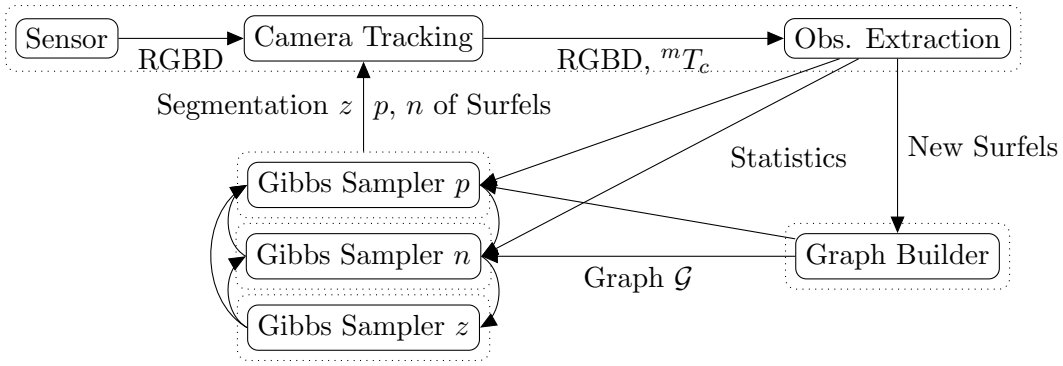


Figure 5.23: Architecture of the direction-aware 3D reconstruction system. Boxes denote algorithm components, and dotted boxes designate the three different threads that are running in parallel. Arrows are marked with the output produced by an algorithm block and consumed by another. Note that the graph builder and Gibbs sampler threads run at their own slower speeds in the background and jointly produce a maximum a-posteriori surfel-map and segmentation estimate that is consumed in real-time by the camera tracking thread. Splitting up the sampler into three threads is straightforward due to the structure of the Gibbs-sampling algorithm.

■ 5.2.4 Implementation

In practice, to use the proposed approach we architect a multi-threaded system as depicted in Fig. 5.23. The main five threads are (1) a real-time data acquisition, camera tracking and observation extraction thread, (2) a nearest neighborhood graph builder thread and (3-5) three Gibbs sampler threads. As shown in the diagram, camera tracking utilizes RGBD frames and the current most likely estimate of the segmentation and surfel map to infer the current camera pose ${}^{w}T_c$. To be able to deal with fast motions we perform photometric rotational pre-alignment of the current RGBD frame against the previous frame as described below. Given the pose, a set of new surfels and observations of existing map-surfels are extracted from the RGBD frame. The new surfels are appended to the map and incorporated into the nearest neighborhood graph by the graph builder thread. The observations of existing surfels are incorporated into the statistics associated with the respective surfel as described in Sec. 5.2.2.

The Gibbs sampler threads produce samples from the joint map and segmentation posterior and provide maximum a-posteriori estimates of the surfel map and segmentation to the camera tracker. Due to the structure of the Gibbs sampling algorithm it is straightforward to split the sampler into three threads each sampling (at its own speed) from the respective posterior given samples from the other threads. Since each thread produces one set of variables that is consumed by the others, no explicit thread synchronization is implemented. The parallel execution of a Gibbs sampler and implications on theoretical guarantees have been investigated in literature under the term Hogwild Gibbs sampling [126].

Observation Extraction The observation extraction algorithm seeks to add new surfels by uniformly sampling so-far unobserved surfaces. To improve camera tracking quality we add a bias towards sampling high gradient surface areas similar to [67]. To perform this sampling efficiently, the algorithm first projects all surfel locations into the current camera view. This rendering directly allows associating new observations to existing map surfels.

While our model makes the assumption that observations of the same surfel are independent conditioned on the camera pose and the surfel location, this is not true in practice. Since this assumption is violated, incorporating every single observation of a surfel may lead the system to become overconfident in the surfels location. To mitigate this effect, we randomly sample a subset of observations to take at each new frame. This has the effect of taking fewer observations of each surfel with some temporal distance between samples thus violating the assumption of conditional independence less.

Data Association and Occlusion Reasoning From a given camera pose estimate, data association between observed RGB and depth image pixels and the surfel-map is obtained by projecting surfel locations into the camera using the OpenGL rendering pipeline. Each surfel is rendered as a colored point, where the 24 bits of RGB color value are used to encode the ID of a surfel. This rendering yields putative associations which might still be wrong due to occlusions and the observation of the back side of a surfel. The angle between viewing direction and surfel normal is used for back-face culling which takes care of the latter incorrect associations.

Occlusion reasoning explicitly takes into account the uncertainty of the camera pose estimate as well as measurement uncertainty. Along a viewing ray r_j the depth uncertainty of an observed depth value x_j^d is obtained by projecting the 3D point uncertainty (see Sec. 5.2.3) onto the viewing ray:

$$\sigma_{d_j}^2 = r_j^T (\Sigma_{O,j} + J_{w T_c} \Sigma_T J_{w T_c}^T) r_j \,. \tag{5.111}$$

We reject an association if the difference between observed depth x_j^d and the predicted depth of the associated surfel exceeds the predicted standard deviation σ_{d_j} by some factor. Theoretically 99.7% of observations should be within $\pm 3\sigma_{d_j}$. In practice, the estimated variance is overly confident most likely due to the over-confident pose variance estimate (see Sec. 5.2.3) and we therefore use a factor of 20 to yield robust occlusion reasoning. The robustness of the occlusion reasoning is important since eliminating correct associations leads the system to add unnecessary surfels.

k-Nearest-Neighbor Graph Building and Maintenance The graph building thread uses the initial observed location of all surfels to maintain a k-nearest-neighbor graph over surfels based on the negative log MRF potential in Eq. (5.45). This is an approximation to the directed graph that could be obtained by connecting all surfels within some distance. Retaining only the top k closest (under the potential) surfels improves algorithm efficiency without notable differences in the reconstruction results. The thread has two states: (1) incorporating a new surfels and (2) revisiting and potentially updating the nearest neighbors of already incorporated surfels. Revisiting existing surfels

is important to maintain the proper nearest neighbors when adding new surfels as well as pruning bad surfels (as described below). In practice if no new surfels are to be processed the thread randomly picks surfels to revisit. In both cases the graph builder thread incrementally computes distances to all other surfels in the map while maintaining a k-sorted list of the closest surfels.

Pruning of noisy surfels facilitates efficient operation of the overall algorithm as well as a clean map. The map builder thread takes on this objective as well when it is revisiting existing surfels. For all surfels that have been in the map for at least Δt_{map} frames, before recomputing the nearest neighbors, it checks if the entropy of the surfel location exceeds a threshold or if the number of observations of the surfel is below some threshold. A surfel is also pruned if it still does not have a full set of nearest neighbors after Δt_{map}.

In practice we construct the $k = 12$ nearest neighbor graph and consider another surfel only a neighbor if it is within 0.2 m. The initial grace period before considering pruning a surfel is $\Delta t_{\text{map}} = 100$ frames, the entropy threshold is $\log \Sigma = -7.5$ and the observation count threshold is 10.

Photometric Rotational Pre-alignment As in most state-of-the-art visual SLAM systems [178, 250] we pre-align a new frame I_c by solving for the rotation that minimizes the photometric error to the previous frame I_p as introduced by Lovegrove [156]. Under pure rotational motion 3D structure is not needed to compute the location of a pixel from the previous frame in the current frame. It is sufficient to compute rotate the ray $\pi^{-1}(u, v)$ into the coordinate system of the current camera via ${}^cR_p$ and to project the resulting ray back onto the image plane:

$$(u_c, v_c) = \pi({}^cR_p \pi^{-1}(u, v)) \,. \tag{5.112}$$

This allows us to formulate the photometric error between the current and the previous frame as a function of the rotation between them as:

$$f_{\text{photo}} = \sum_{u,v} \| I_c(\pi({}^cR_p \pi^{-1}(u, v))) - I_p(u, v) \|_2^2 \,. \tag{5.113}$$

By Taylor-expanding the error term to the first derivative, writing the linearized error term as a linear least squares problem we can solve for the rotation change ω_R via the pseudo inverse in the same manner as in ICP. The rows of A and b of the linear system are:

$$A_i = \nabla I_c \frac{\partial \pi(x)}{\partial x} \left(-{}^cR_p \left[\pi^{-1}(u, v) \right]_\times \right) \tag{5.114}$$

$$b_i = I_p - I_c \,. \tag{5.115}$$

To speed up the process and to yield a wider basin of attraction, we perform this optimization on an image pyramid, starting at the highest level down to the first level. In practice we utilize pyramid level 3 down to 1 and ignore the lowest level, the full-resolution image. In building the image pyramid each higher level is half the resolution

of the lower image. During pyramid construction a 9×9 Gaussian blur kernel is applied. The rotational pre-alignment takes $5ms$ using GPU acceleration. The resulting rotation ${}^{c}R_{p}$ is applied to ${}^{w}T_{c}$ before running ICP:

$$ {}^{w}R_{c} = {}^{w}R_{c}\,{}^{c}R_{p}^{T} \,. \tag{5.116} $$

Multi-scale ICP As described for the rotational pre-alignment, we also run direction-aware ICP on incrementally lower pyramid levels to improve the basin of attraction for camera tracking against the surfel-map. In practice we rely on the 1st and 0th pyramid level.

Gibbs Posterior-sample Balancing The usual setup for Gibbs sampling is batch processing where all data is available a priori. In the 3D reconstruction setting, data is obtained incrementally and the underlying state expands as well as new surfels are added to the joint distribution. Simple iterative or uniformly random posterior sampling of the individual conditional distributions means that new parts of the map are sample-starved. This means estimates for new surfel locations, orientations and segmentation are slow to converge. To mitigate this, we implement a random schedule that biases towards sampling parameters with low sample counts with a higher probability. The sampler iterates over all indices and for each samples the posterior with probability

$$ \Pr\left(\text{sample from posterior } i\right) = 1 - \frac{\alpha + |\mathcal{S}_i|}{\alpha + \sum_i |\mathcal{S}_i|} \,, \tag{5.117} $$

where $|\mathcal{S}_i|$ is the number of samples drawn from posterior i and α is a scalar count that can be increased to decrease the bias towards sampling low sample-count indices. Implementing this sample balancing mechanism is straightforward and creates little overhead since we only need to incrementally keep track of $|\mathcal{S}_i|$ and $\sum_i |\mathcal{S}_i|$.

■ 5.2.5 Evaluation and Results

In the following we evaluate the proposed direction-aware 3D reconstruction system on various challenging datasets quantitatively as well as qualitatively.

All experiments are performed on a machine with a Intel Xeon CPU with 16 cores at 2.4 GHz and a Nvidia GTX-1080 graphics card. As described in Sec. 5.2.4, the algorithm utilizes a total of 5 CPU cores for Gibbs sampling, camera tracking and nearest neighbor graph maintenance. The GPU is only utilized for the full-frame operations of $\mathbb{SO}(3)$ pre-alignment and data preprocessing like smoothing input images, image pyramid construction, point cloud computation, and image gradient computation. Surface normals are computed only sparsely wherever needed.

Qualitative Reconstructions and Directional Segmentation In Figures 5.24 to 5.28 we show different layers of 3D reconstructions of different scenes obtained from the proposed system. Besides the 3D structure captured via the collection of surfels, the algorithm infers the RGB and grey-scale color of each surfel. The former is used for visualization purposes, the latter is used by ICP for the photometric contribution to

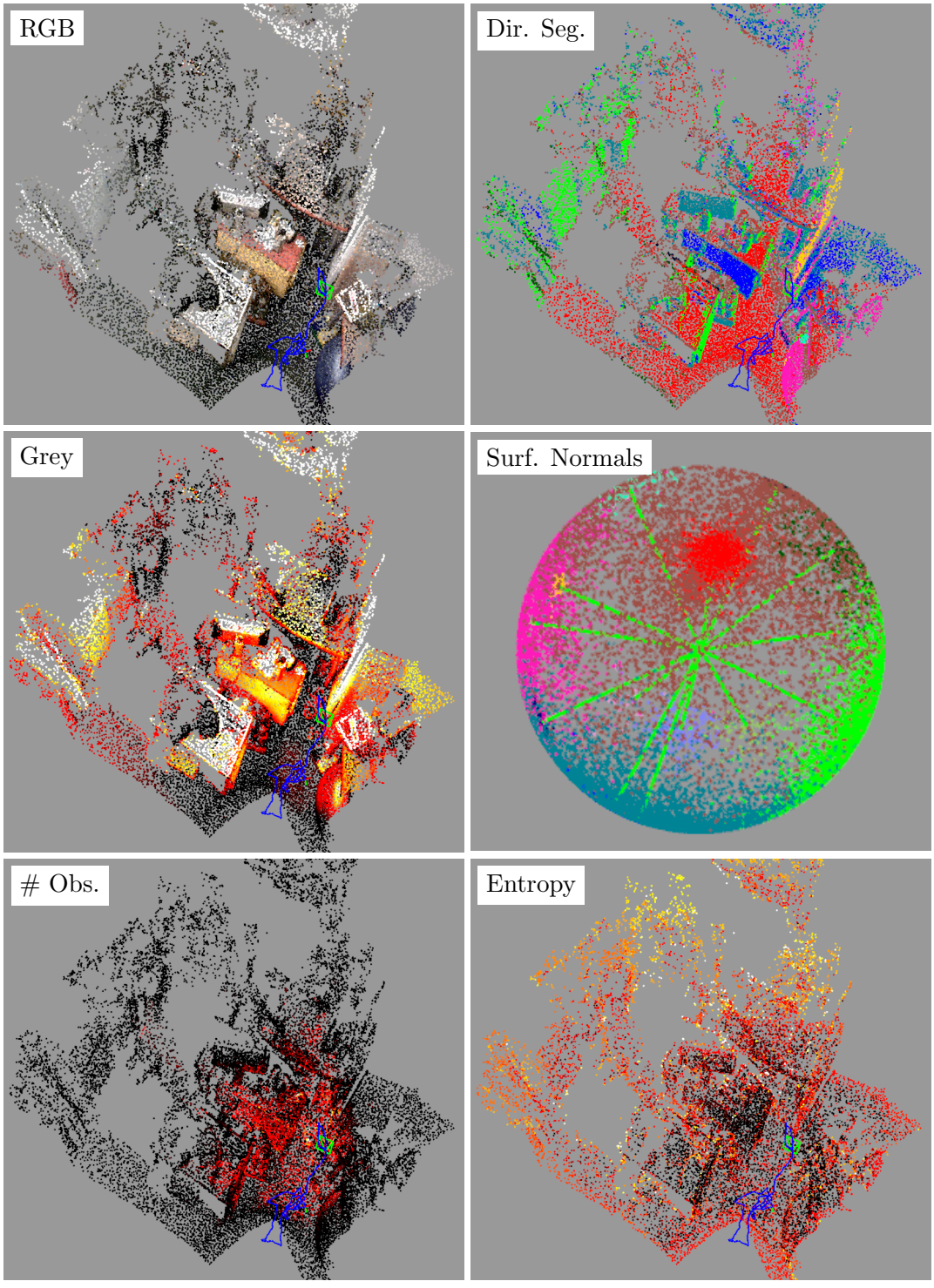


Figure 5.24: Different aspects of the inferred 3D reconstruction of an office area are visualized. Besides RGB color, grey-scale value, and directional segmentation, the entropy of each surfel's location using the hot colorscheme ("hotter" means higher). Notice how areas in the scene that have been observed from close-by or for longer have lower entropy, indicating better knowledge of the surfels location.

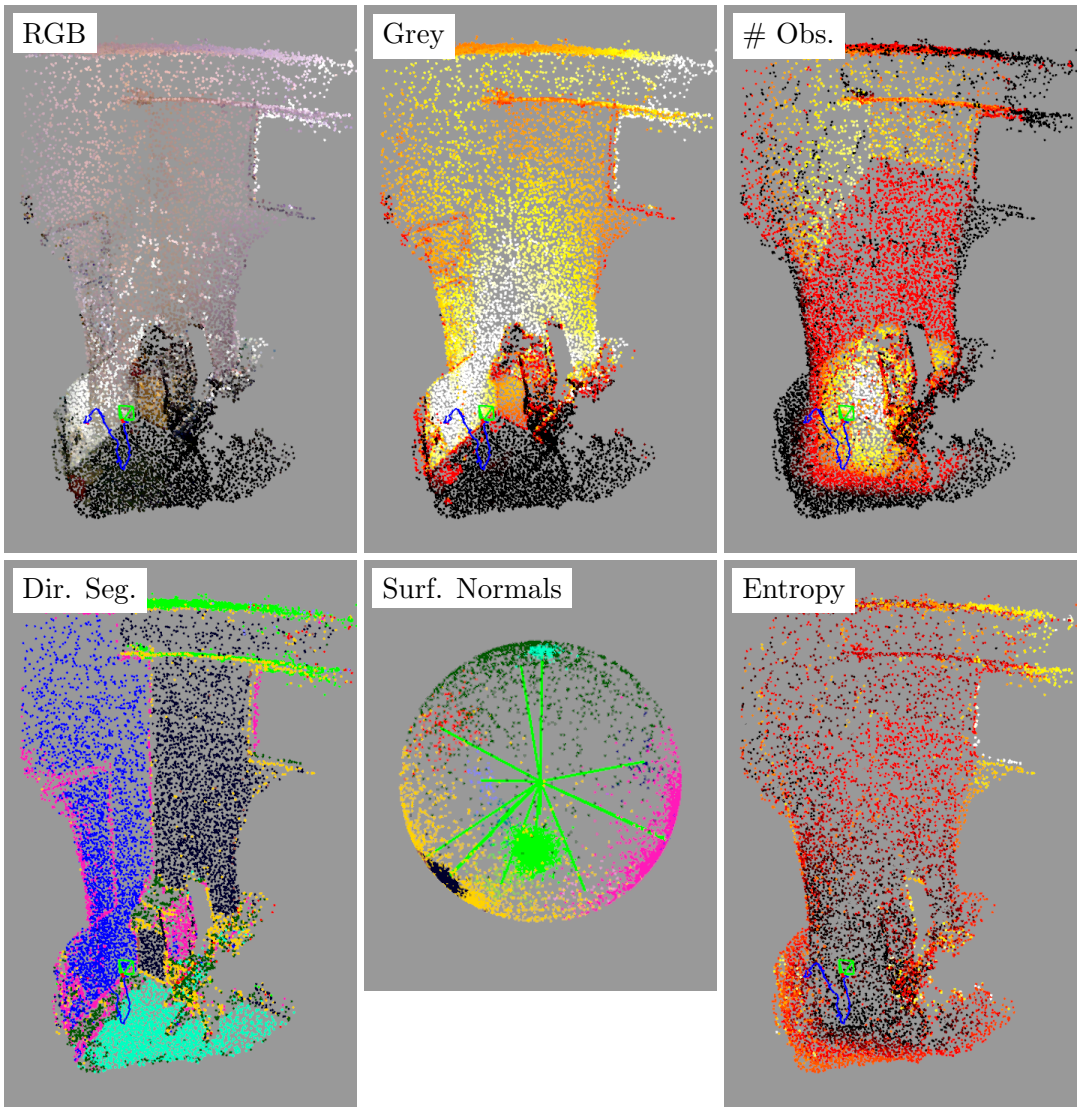


Figure 5.25: Different aspects of the inferred 3D reconstruction of a room corner are visualized. Besides RGB color, grey-scale value, and directional segmentation, the entropy of each surfel's location using the hot colorscheme ("hotter" means higher). Notice how areas in the scene that have been observed from close-by or for longer have lower entropy, indicating better knowledge of the surfels location. The plot of the number of observations taken for each surfel corroborates this. The large dark green, yellow and purple directional clusters captures noisy parts of the reconstruction at corners and discontinuities.

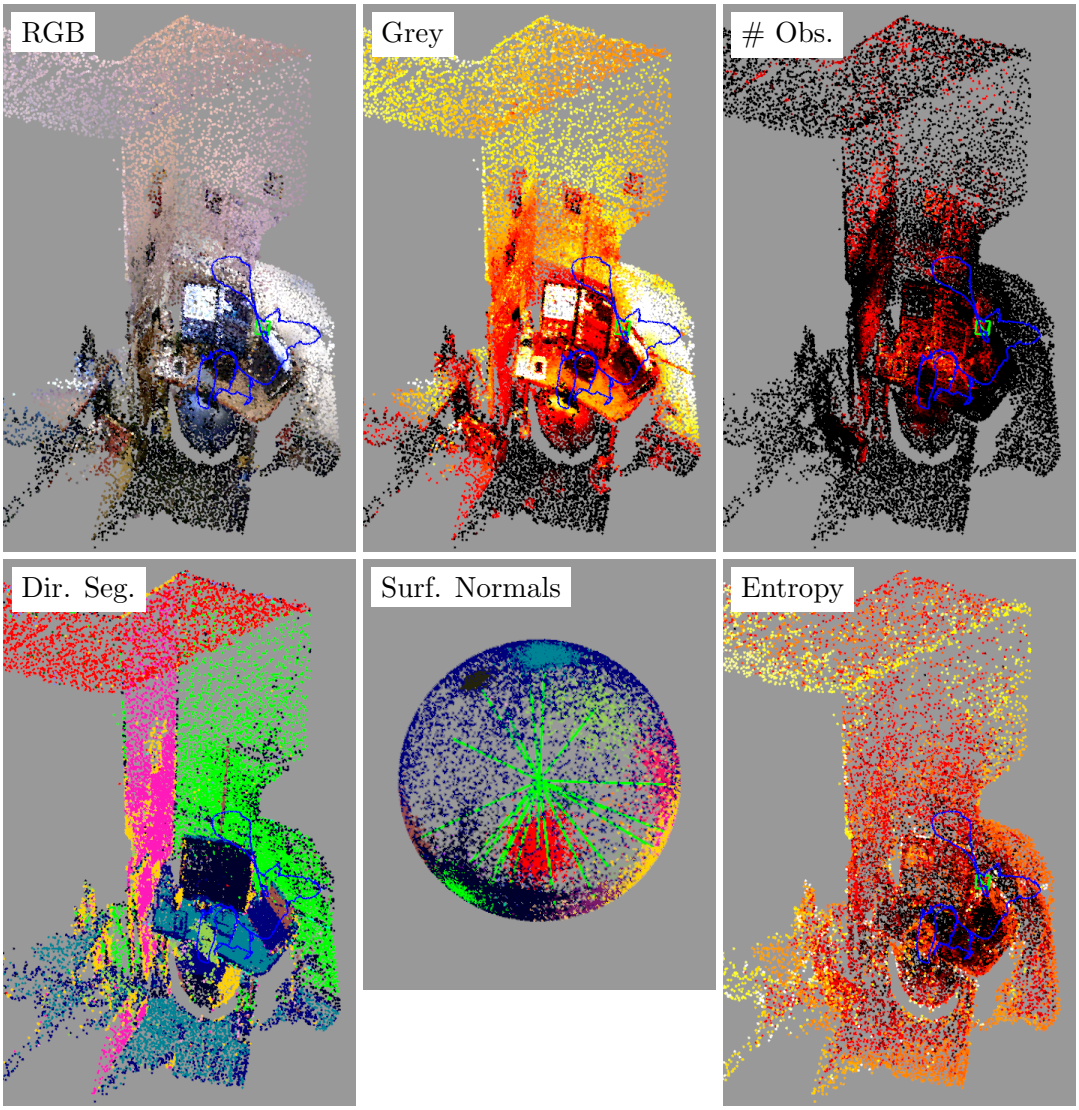


Figure 5.26: Different aspects of the inferred 3D reconstruction of a desk area are visualized. Besides RGB color, grey-scale value, and directional segmentation, the entropy of each surfel's location using the hot colorscheme ("hotter" means higher). Notice how areas in the scene that have been observed from close-by or for longer have lower entropy, indicating better knowledge of the surfels location. The plot of the number of observations taken for each surfel corroborates this. Furthermore, notice the detail in the directional segmentation: the laptop to the right is segmented into its own two directional clusters for example. The large dark blue directional cluster captures noisy parts of the reconstruction at corners and discontinuities.

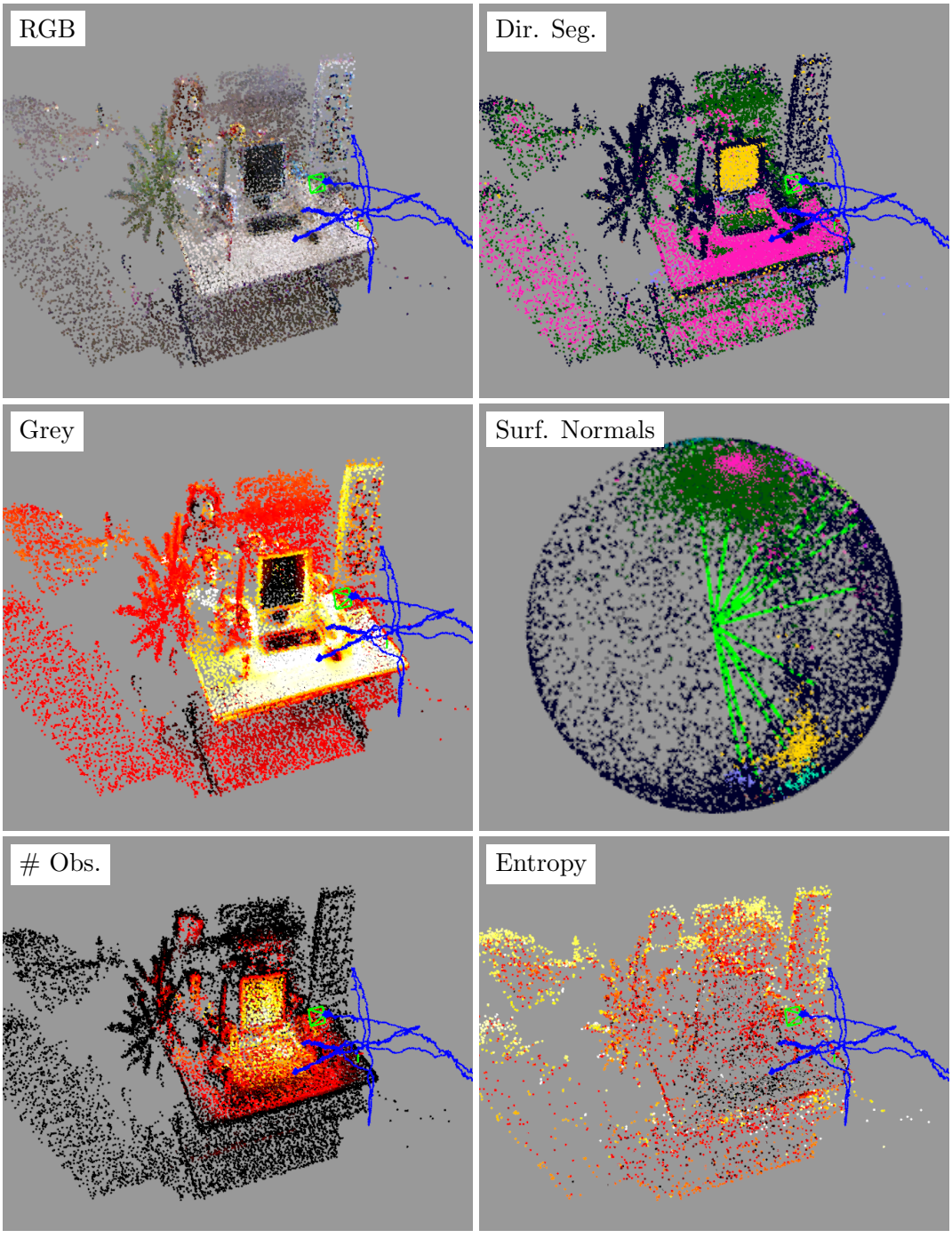


Figure 5.27: Different aspects of the inferred 3D reconstruction of the `fr2_xyz` dataset are visualized. See previous figures for description. Of note in this dataset is the large black cluster that captures all the non-planar parts of the scene. The besides the main "up" direction the sampling algorithm nevertheless captures also the finer details of the direction of the monitor as well as the keyboard.

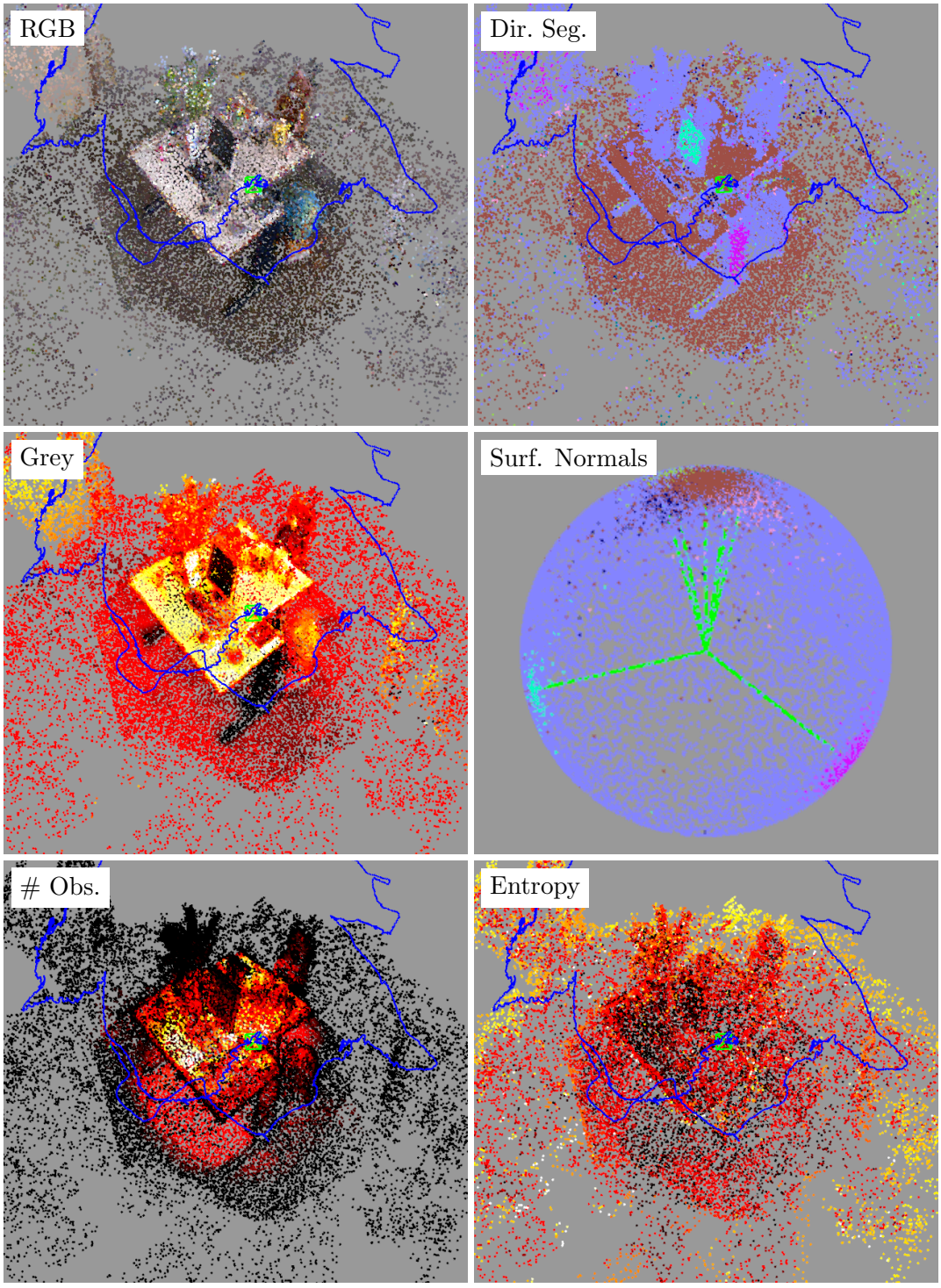
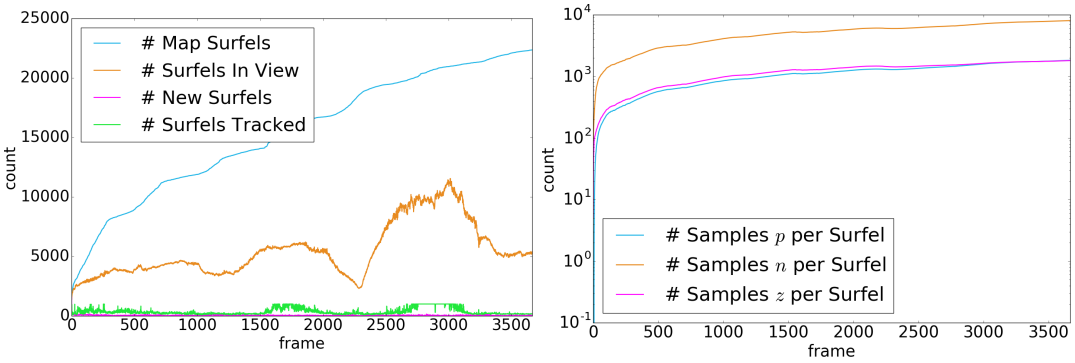


Figure 5.28: Different aspects of the inferred 3D reconstruction of the `fr2_desk` dataset are visualized. See previous figures for description. Of note in this dataset is the large cluster that captures all the non-planar parts of the scene. The inferred model nevertheless also captures the main "up" direction, and finer details such as the direction of the monitor and the side of the desk.



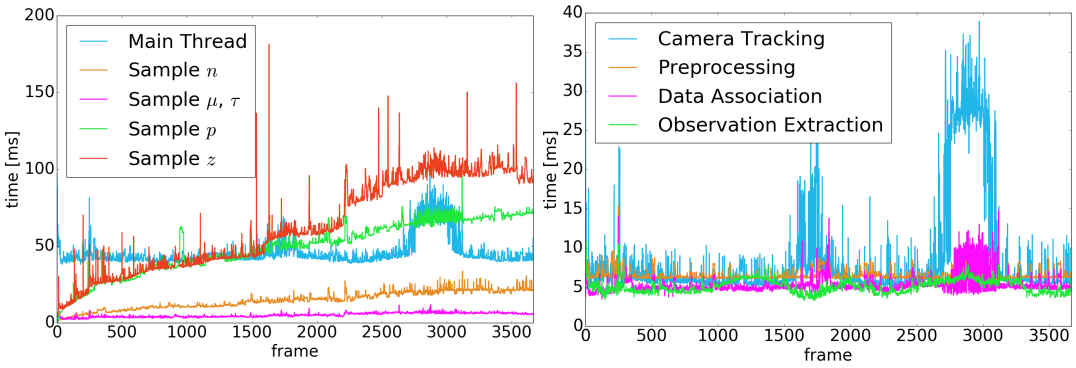
(a) Statistics over surfels.

(b) Number of samples per surfel.

Figure 5.29: Surfel and sample count statistics for the direction-aware SLAM system for the `fr2_xyz` dataset [229]. While the number of map-surfel keeps growing (slowly) as the scene is explored, the number of surfels that are currently observed depends more on the viewpoint. Of the surfels in view, the direction-aware SLAM system utilizes only a fraction for camera tracking. As displayed to the right, the sampling threads are able to keep up with the growth of the map, yielding ample samples per surfel to allow the computation of confident estimates of the surfel and segmentation distribution.

camera tracking. The maximum a-posteriori estimate of the directional segmentation of the environment sensibly partitions the environment according to the surfel directions. A general pattern between all inferred directional segmentations is that the inference extracts the main peaks of the distribution which correspond to planar regions in the scene. The inference additionally automatically infers low concentration clusters that capture noisy, non-planar regions. The availability of samples from the joint surfel-based map posterior also allows the quantification of uncertainty via the computation of entropy for a surfel's location as displayed in the figures. Observe how parts of the map that have been observed from closer range or that have received more observations generally have lower entropy indicating more certainty about the surfel location.

Algorithm Operation and Properties To give intuition for the operation of the algorithm, we discuss timings, surfel and sampling statistics collected during the reconstruction of the `fr2_xyz` dataset [229] displayed in Fig. 5.27. Figure 5.30 shows the runtime of different parts of the 3D reconstruction system. The plot to the left shows that the main camera tracking thread runs in less than 50ms per frame. The runtime of the sampling threads scales with the number of surfels in the map which grows as new scene parts are observed (see Fig 5.29). Since the number of DP-vMF clusters scales with the complexity of the surface normal distribution, the samplers runtime is virtually constant for this scene. The bump in the timing of the main thread happens when the camera moves far away from the scene and ICP processes (see Fig. 5.30b) a lot more points to bring down the largest standard deviation as described in Sec. 5.2.3. The other algorithm components of the main thread are virtually constant time. The statistics in



(a) Main and sampling threads.

(b) Components in the main thread.

Figure 5.30: Timings of the direction-aware SLAM system during the reconstruction of the `fr2_xyz` dataset [229]. To the left timings of the main threads running in parallel are shown (sampling μ, τ, and z happens in one thread). While time for sampling all surfel parameters for the given map increases with the map size, both sampling the parameters μ and τ of the DP-vMF mixture model and camera tracking and observation extraction in the main thread have constant runtimes. As can be seen to the right, where the main thread's components are timed, the spike in the runtime of the main thread is due to camera tracking. In this part of the dataset the camera is far from any structure and thus the algorithms collects more observations leading to slower camera tracking.

Fig. 5.29 show that while the number of surfels in the map keep growing, the sampling threads yield sufficient samples per surfel to compute credible statistics and estimates for reasoning about the underlying joint distribution of surels and segmentation. As can also be seen, the number of surfels utilized for camera tracking is usually less than 1000 surfels even if a magnitude more surfels are in view. This efficient ICP operation is enabled by the direction and gradient-aware selection of surfel observations.

On the same dataset, Fig. 5.31 compares direction-aware ICP against selecting observation points for ICP at random. Direction-guided observation selection leads to a lower uncertainty lower-bound. The maximum standard deviation extracted from the Fisher information matrix of the camera pose estimator is lower in hard-to-track phases of the dataset. This superior tracking is achieved with significantly fewer surfel observations. These two facts show the power of utilizing direction-guided measurement selection for ICP.

Ablation Study on `fr2_xyz` To quantify the effect of utilizing the directional segmentation we perform an ablation study on the `fr2_xyz` dataset [229] with groundtruth trajectory. Specifically we explore all possible combinations of the following three components of the system: (1) sampled map vs. standard surfel fusion, (2) direction-aware surfel observation selection or not, and (3) image gradient norm sorting of surfel observations or not. In the experiments with standard surfel fusion, the directional seg-

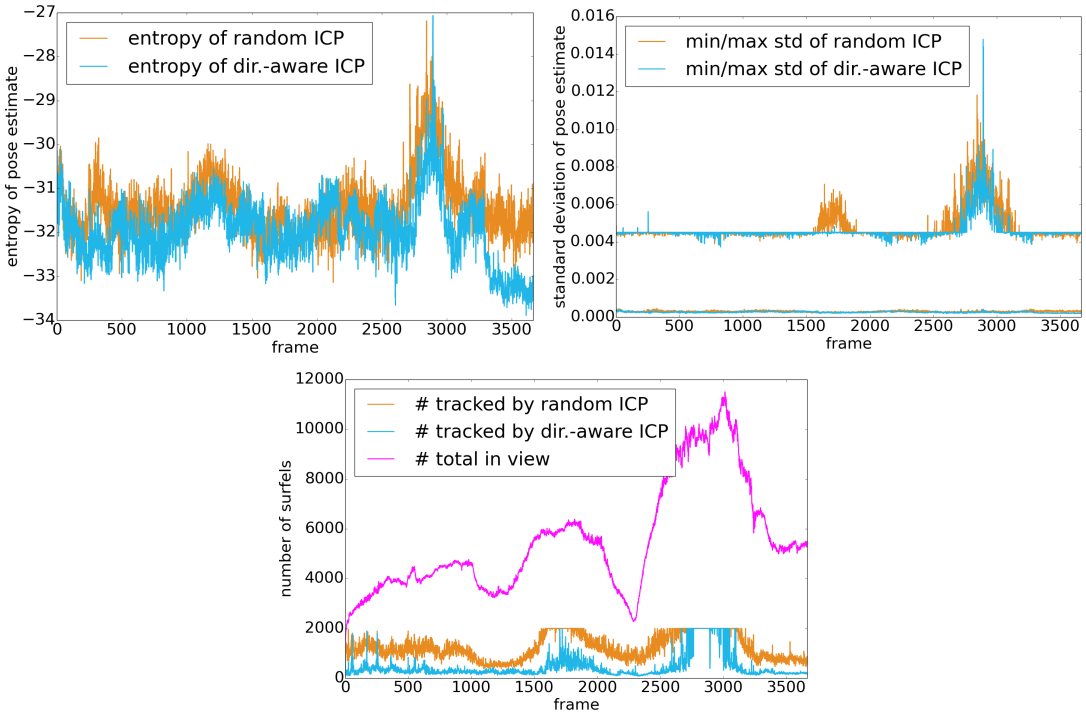


Figure 5.31: Comparison of direction-aware to random observation selection for ICP in terms of the lower-bound on the entropy of the pose estimate as well as the min and max standard deviations extracted from the Fisher information matrix. The direction-aware observation selection strategy leads to a better lower-bound on the true uncertainty of the pose estimate. The plot in the second row shows that the improved tracking uncertainty is obtained despite tracking fewer surfels. The statistics were obtained on the `fr2_xyz` dataset.

	Surfel Fusion	Direction-aware Mapping
Direction & Gradient	0.018m, 44.6ms	**0.014m**, 45.7ms
Direction-aware	0.017m, 48.7ms	0.015m, 48.7ms
Gradient	0.018m, **42.8ms**	**0.014m**, 43.5ms
Random	0.018m, 47.2ms	0.016m, 46.0ms

Table 5.3: The ablation study presented in this table was conducted over the `fr2_xyz` dataset. The tracking accuracy is improved by using direction-aware mapping. The use of the directional segmentation and gradient norm for surfel observation selection further improves camera pose estimation.

mentation is inferred solely based on the surface normal observations. Table 5.3 shows camera tracking accuracy and algorithm timing on the `fr2_xyz` dataset for the eight different configurations. The use of direction-aware mapping vs. standard surfel fusion clearly improves tracking accuracy independent of the configuration of the observation selection of ICP. This leads to the conclusion that the underlying cause for the improved camera tracking accuracy is a higher quality map. Among the camera tracking configurations both direction-awareness and gradient sorting improve the tracking performance over just random selection. Interestingly as we will see next the accuracy improvement is achieved by utilizing fewer surfel observations.

Camera Tracking Accuracy Comparison We use the TUM indoor dataset [229] and the synthetic dataset by Handa et al. [99] to evaluate the camera tracking accuracy against groundtruth tracking data and compare our system to related 3D SLAM systems. Unfortunately the noise model used to simulate the synthetic dataset has very different artifacts than the one assumed herein. While we assume Kinect-like depth cameras wich use triangulation for depth estimation, the dataset simulates time-of-flight (ToF) sensor noise such as found in the Kinect v2 camera. Specifically at depth discontinuities the simulated ToF noise manifests as large depth outliers anywhere between the actual depths and the minimum depth. Spurious surfel observations are added due to the aforementioned artifacts. This precludes any meaningful evaluation of the surface reconstruction accuracy. The camera tracking system is robust to such noise and we therefore report pose estimation accuracy in Table 5.4. As a measure for alignment accuracy we use the absolute trajectory error (ATE) which is the error between time-associated poses after aligning the estimated and the ground truth trajectory [229]. Table 5.4 demonstrates that the proposed nonparametric direction-aware SLAM system is on par or better than related algorithms in terms of camera trajectory estimation for datasets without the need for loop closures. The `kt3` dataset is a hard dataset necessitating loop closure and the proposed system suffers from the lack of such capability. The related algorithms perform only slightly better on this challenging dataset.

■ 5.3 Discussion

In the first section of this chapter, we introduced a branch-and-bound approach to global point cloud alignment with convergence guarantees, based on a Bayesian nonparametric point cloud and surface normal distribution representation and a novel tessellation of the rotation space. The method decouples translation and rotation via the use of surface normals, making it more efficient than previous joint approaches. Experiments demonstrate the robustness of the method to noisy real world data, partial overlap, and angular viewpoint differences. We expect that the proposed novel tessellation of $\mathbb{S}^3$ will be useful in optimizing other functions over the space of rotations with theoretical guarantees using branch and bound. An efficient implementation of the proposed global alignment algorithm can be found at `http://people.csail.mit.edu/jstraub/`.

In the future it would be interesting to explore ways of guaranteeing global optimal-

System	kt0	kt1	kt2	kt3	fr2_xyz	fr2_desk
DVO-SLAM [135]	0.032	0.061	0.119	0.053	0.021	0.017
RGB-D SLAM [66]	0.044	0.032	0.031	0.167	0.023	0.095
ElasticFusion [250]	0.009	0.009	0.014	0.106	0.011	-
Kintinuous [246]	0.072	0.005	0.010	0.355	0.029	0.034
Dense Planar SLAM [210]	0.246	0.0169	-	-	-	-
CPA-SLAM [158]	-	-	-	-	0.014	0.046
Directional SLAM	0.005	0.007	0.017	0.417	0.015	0.066

Table 5.4: Comparison of the absolute trajectory error (ATE) in meters as defined in [229] of different SLAM systems for different synthetic datasets from the benchmark dataset by Handa et al. [99] (kt0-3) and the TUM indoor dataset [229]. Note that all other systems utilize loop-closure techniques to correct for drift and tracking errors. This explains the poor performance of the proposed algorithm on kt3 which necessitates loop closure capabilities.

ity for the proposed point cloud alignment approach. Currently there is no reasoning about which parts of the scene are actually co-observed leading to poor performance in the case of small overlap. Incorporating such reasoning in effect would amount to inference about the map and make the approach an instance of joint categorical SLAM. If higher accuracy and better performance on smooth shapes were required, using a sampling based inference method for fitting the Dirichlet process von-Mises-Fisher and Gaussian mixture models would likely improve the alignment at the detriment of speed.

Another avenue of research for global alignment would be to explore rapid alignment approaches based on matching directional clustering modes between frames for rotational alignment that could be obtained in closed form via the orthonormal Procrustes problem (see Sec. 2.7.3). Given a putative rotation the algorithm could then seek to match clusters of points projected onto their associated directional cluster (via their surface normal). Such matching could deliver translational alignment.

In the second part of this chapter, we have introduced the first nonparametric direction-aware SLAM system that jointly reasons about directional segmentation, a surfel-based world map, and the trajectory of a RGBD camera. We ascribe direction-awareness to the system since it utilizes the directional segmentation for its other tasks as opposed to inferring the segmentation without further purpose. A key contribution is the proposed map formulation which establishes a connection between scene-wide directional segmentation and local surface properties. The proposed system architecture demonstrates that the Gibbs-sampling-based inference algorithm for the Bayesian nonparametric directional segmentation and surfel locations and orientations can be run in the background to facilitate realtime operation. Experiments show that incorporating directional segmentation into the mapping and camera tracking problem yields improved camera tracking accuracy. Furthermore, in comparison to uninformed measurement selection, direction-aware camera tracking improves camera pose estima-

tion certainty and accuracy while reducing the number of surfels needed for accurate tracking.

The use of Gibbs-sampling based inference on a complex Bayesian nonparametric model in a realtime reconstruction system has not been demonstrated before and due to the flexibility of Gibbs-sampling opens up exciting possibilities for inference on more complex and detailed environment models. Having access to samples from the true posterior also allows reasoning about uncertainty which is not possible with mode-seeking inference methods such as maximum-a-posteriori estimation. An implementation of the proposed nonparametric direction-aware SLAM system can be found at http://people.csail.mit.edu/jstraub/.

Along the line of nonparametric direction-aware SLAM it would be interesting to explore fully joint inference approach based on variational inference under, for example, the mean-field assumption. This optimization-based, more efficient approach would still provide not only the modes but also uncertainty estimates for the map as well as the directional segmentation in a holistic fashion. As described in the background section on Bayesian nonparametric models and inference Sec. 2.4, variational inference is a common inference technique for such models, but it has not been applied to 3D reconstruction techniques.

It would also be interesting to explore other scene priors for categorical SLAM, such as the Manhattan Frame or the more flexible MMF model of Chapter 3. The Manhattan World-based models could add regularity in terms of orthogonal angles to the proposed Stata Center World-based directional segmentation and allow for more pronounced pooling of measurements across the scene to yield more accurate 3D reconstructions. Another approach would be to investigate a hybrid approach of Stata Center World and Manhattan World. Since the directional segmentation is inferred using Gibbs-sampling it is quite straightforward to incorporate Metropolis-Hastings proposals to join different directional clusters into a Manhattan Frame.

■ 5.4 Acknowledgments

The global point cloud alignment work of Sec. 5.1, published in [223], was developed jointly with Trevor Campbell, John W. Fisher III and John P. How. While the initial idea of the rotation space tessellation and the use of Dirichlet process mixture models stemmed from me, Trevor Campbell contributed most of the theoretical work (bounds, tessellation shrinkage and branch and bound convergence), as indicated by the equal contribution to the paper [223].

The work on direction-aware SLAM outlined in Sec. 5.2 was developed jointly with John W. Fisher III.

Chapter 6

Conclusion

On a high level, the first main contribution of this thesis is the introduction of theoretical and algorithmic concepts that are instrumental in providing artificial perception systems with a directional understanding of their environment. Specifically, I have thoroughly investigated the connection between the 3D structure of different environment types and their surface normal distributions. Chapter 3 investigates Manhattan-constrained directional scene models while Chapter 4 relaxes the orthogonality assumption and focuses on general directional scene models. The models and algorithms herein for Manhattan World and Stata Center World scene segmentation from surface normal data provide an initial step towards directional scene understanding of man-made environments. Since Mixture of Manhattan Frames and Stata Center World environments cannot be described with a fixed number of parameters, a key contribution is showing how to utilize Bayesian nonparametric modeling concepts to make model selection part of the probabilistic inference process.

The second main contribution of this thesis is showing how to embed these nonparametric directional scene models into 3D perception systems to improve 3D point cloud alignment and realtime 3D reconstruction. The global point cloud alignment algorithm described in Sec. 5.1 leverages a probabilistic directional Stata Center World scene model for rotation alignment. This decomposes the global alignment problem into rotational and translational alignment which makes the branch-and-bound search an order of magnitude faster than related algorithms. Finally, the 3D reconstruction system introduced in Sec. 5.2 is the first semi-dense nonparametric direction-aware SLAM system, that jointly infers the nonparametric Stata Center World directional segmentation, the 3D reconstruction, and the camera pose in realtime. The fact that inference over map and Bayesian nonparametric directional segmentation is performed using Gibbs-sampling opens up exciting possibilities to model more complex phenomena using the proposed architecture. I believe that adding any categorical or semantic segmentation should first and foremost serve the purpose of improving the operation of the perception system. The underlying notion is that one way of gauging if an artificial perception system "understands" a concept is by determining if it is able to use it in ways that further its own purpose. In this context the systems presented in the last chapter can be seen as "understanding" the directional composition of the environment.

Taken together these contributions yield the first nonparametric directional per-

ception systems which in some sense are aware of the directional composition of the scene and demonstrate this by improving on key 3D perception tasks. Since the proposed directional scene models can capture the majority of surfaces in most man-made environments, this ability presents a significant first step towards general scene understanding.

To make progress towards truly intelligent artificial perception systems, it will be necessary to push beyond hand-engineered scene priors and assumptions. While the directional scene models presented in this thesis capture a large fraction of surfaces in man-made environments, there are long tails to the class of non-planar surfaces that need to be addressed with more flexible data-driven approaches. For these cases, future perception systems should be structured such that they can learn to extract geometric or other concepts by themselves and such that they can discover how to use such concepts to further their goal of operation.

Appendix A

Derivations Pertaining to the Background

A.1 Direct Surface Normal Extraction from Depth Images

As defined by the Gauss map in Eq. (2.184), we can obtain the surface normals from the point cloud using the cross-product of the local gradients:

$$n = \left\lceil \frac{\partial p(u,v)}{\partial u} \times \frac{\partial p(u,v)}{\partial v} \right\rceil \tag{A.1}$$

where the derivatives are along the u and v axis of the image coordinate system. Note that the derivatives and cross-products can be computed completely in parallel. We could simply compute the point cloud using any camera model, compute the local gradients using forward differences and then compute surface normals. But since the point cloud is fully determined by the unprojection function $\pi^{-1}(u,v,d)$ of the camera model and the depth image $d(u,v)$ we can compute the derivatives using the chain rule:

$$n = \left\lceil \frac{\partial \pi^{-1}(u,v,d(u,v))}{\partial u} \times \frac{\partial \pi^{-1}(u,v,d(u,v))}{\partial v} \right\rceil \tag{A.2}$$

$$= \left\lceil \left(\frac{\partial \pi^{-1}(u,v,d)}{\partial u} + \frac{\partial \pi^{-1}(u,v,d)}{\partial d}\frac{\partial d}{\partial u} \right) \times \left(\frac{\partial \pi^{-1}(u,v,d)}{\partial v} + \frac{\partial \pi^{-1}(u,v,d)}{\partial d}\frac{\partial d}{\partial v} \right) \right\rceil \tag{A.3}$$

Under the pinhole camera model [102], the point cloud $p(u,v)$ can be recovered from a depth image $d(u,v)$ as:

$$p(u,v) = \pi^{-1}(u,v,d(u,v)) = \begin{pmatrix} \frac{d(u,v)}{f_u}(u-u_c) \\ \frac{d(u,v)}{f_v}(v-v_c) \\ d(u,v) \end{pmatrix}, \tag{A.4}$$

where f_u and f_v are the focal lengths of the depth camera (in u and v direction) and $[u_c, v_c]$ is the center of the depth-image. Now the derivatives can be computed in terms

of the depth image derivatives as:

$$\frac{\partial p(u,v)}{\partial u} = \begin{pmatrix} \frac{\partial d(u,v)}{\partial u}\frac{\Delta u}{f_d} + \frac{d(u,v)}{f_d} \\ \frac{\partial d(u,v)}{\partial u}\frac{\Delta v}{f_d} \\ \frac{\partial d(u,v)}{\partial u} \end{pmatrix} \qquad \frac{\partial p(u,v)}{\partial v} = \begin{pmatrix} \frac{\partial d(u,v)}{\partial v}\frac{\Delta u}{f_d} \\ \frac{\partial d(u,v)}{\partial v}\frac{\Delta v}{f_d} + \frac{d(u,v)}{f_d} \\ \frac{\partial d(u,v)}{\partial v} \end{pmatrix} \tag{A.5}$$

where we used $\Delta u = u - u_c$ and $\Delta v = v - v_c$. For the purpose of computing the surface normal we can simplify further:

$$n = \left| \begin{pmatrix} \frac{\partial d}{\partial u}\frac{\Delta u}{f_d} + \frac{d}{f_d} \\ \frac{\partial d}{\partial u}\frac{\Delta v}{f_d} \\ \frac{\partial d}{\partial u} \end{pmatrix} \times \begin{pmatrix} \frac{\partial d}{\partial v}\frac{\Delta u}{f_d} \\ \frac{\partial d}{\partial v}\frac{\Delta v}{f_d} + \frac{d}{f_d} \\ \frac{\partial d}{\partial v} \end{pmatrix} \right| = \left| \begin{pmatrix} \frac{\partial d}{\partial u}\Delta u + d \\ \frac{\partial d}{\partial u}\Delta v \\ \frac{\partial d}{\partial u} f_d \end{pmatrix} \times \begin{pmatrix} \frac{\partial d}{\partial v}\Delta u \\ \frac{\partial d}{\partial v}\Delta v + d \\ \frac{\partial d}{\partial v} f_d \end{pmatrix} \right| \tag{A.6}$$

$$= \left| \frac{\partial d}{\partial u}\frac{\partial d}{\partial v} \begin{pmatrix} \Delta v f_d - [\Delta v + d/\frac{\partial d}{\partial v}] f_d \\ f_d \Delta u - f_d[\Delta u + d/\frac{\partial d}{\partial u}] \\ [\Delta u + d/\frac{\partial d}{\partial u}][\Delta v + d/\frac{\partial d}{\partial v}] - \Delta u \Delta v \end{pmatrix} \right| \tag{A.7}$$

$$= \frac{1}{f_d}\frac{\partial d}{\partial u}\frac{\partial d}{\partial v} \begin{pmatrix} \Delta v - [\Delta v + d/\frac{\partial d}{\partial v}] \\ \Delta u - [\Delta u + d/\frac{\partial d}{\partial u}] \\ \frac{1}{f_d}\{[\Delta u + d/\frac{\partial d}{\partial u}][\Delta v + d/\frac{\partial d}{\partial v}] - \Delta u \Delta v\} \end{pmatrix} \tag{A.8}$$

$$= \frac{1}{f_d}\frac{\partial d}{\partial u}\frac{\partial d}{\partial v} \begin{pmatrix} -d/\frac{\partial d}{\partial v} \\ -d/\frac{\partial d}{\partial u} \\ \frac{1}{f_d}\{\Delta u d/\frac{\partial d}{\partial v} + \Delta v d/\frac{\partial d}{\partial u} + d/\frac{\partial d}{\partial v} d/\frac{\partial d}{\partial u}\} \end{pmatrix} \tag{A.9}$$

$$= \frac{d}{f_d}\frac{\partial d}{\partial u}\frac{\partial d}{\partial v} \begin{pmatrix} -1/\frac{\partial d}{\partial v} \\ -1/\frac{\partial d}{\partial u} \\ \frac{1}{f_d}\{\Delta u/\frac{\partial d}{\partial v} + \Delta v/\frac{\partial d}{\partial u} + \frac{d}{\frac{\partial d}{\partial v}\frac{\partial d}{\partial u}}\} \end{pmatrix} \tag{A.10}$$

$$= \left| \begin{pmatrix} -\frac{\partial d}{\partial u} f_d \\ -\frac{\partial d}{\partial v} f_d \\ \Delta u \frac{\partial d}{\partial u} + \Delta v \frac{\partial d}{\partial v} + d \end{pmatrix} \right| . \tag{A.11}$$

This derivation can be executed for any camera model for which the derivatives of the inverse projection operation can be computed in closed form.

■ A.2 Analysis of the Joint Prior for the von-Mises-Fisher Distribution

As introduced in Sec. 2.6.3, the joint prior of the vMF distribution is known up to proportionality as

$$p(\mu, \tau; \mu_0, a, b) \propto f(\tau, \mu; a, b, \mu_0) = \left(\frac{\tau}{\sinh \tau}\right)^a \exp\left(\tau b \mu^T \mu_0\right) \tag{A.12}$$

We will now characterize this distribution with a focus on the variation in τ to facilitate the implementation and theoretical justification of a slice sampler to sample from $p(\mu \mid \tau; \mu_0, a, b)$. Note that all analysis applies for the posterior distribution as well by using

the posterior parameter a_N, b_N, and μ_N instead of a, b and μ_0. It will be convenient to work in log space:

$$\log f(\tau, \mu; a, b, \mu_0) = a \log \tau - a \log \sinh \tau + \tau b \mu^T \mu_0 \tag{A.13}$$

$$= a \log \tau + a \log 2 - a \log(1 - \exp(-2\tau)) + \tau \left(b \mu^T \mu_0 - a \right) \tag{A.14}$$

Keeping in mind that $0 < b < a$, the limit of the function as $\tau \to 0$ is 0 and as $\tau \to \infty$ is $-\infty$.

The derivative of $\log f(\tau)$ is

$$\frac{\partial}{\partial \tau} \log f(\tau) = \frac{a}{\tau} - \frac{2a \exp(-2\tau)}{1 - \exp(-2\tau)} + b \mu^T \mu_0 - a \tag{A.15}$$

Unfortunately once cannot solve for the maximum in closed form by setting the derivative to 0 (I have tried). The limits of this first derivative as $\tau \to 0$ is $b \mu^T \mu_0$ and as $\tau \to \infty$ is $b \mu^T \mu_0 - a$.

The second derivative of $\log f(\tau)$ is

$$\frac{\partial^2}{\partial \tau^2} \log f(\tau) = -\frac{a}{\tau^2} - a \frac{-4(1 - \exp(-2\tau)) \exp(-2\tau) - 4 \exp(-2\tau) \exp(-2\tau)}{(1 - \exp(-2\tau))^2} \tag{A.16}$$

$$= -\frac{a}{\tau^2} - a \frac{-4 \exp(-2\tau) + 4 \exp(-4\tau) - 4 \exp(-4\tau)}{(1 - \exp(-2\tau))^2} \tag{A.17}$$

$$= -\frac{a}{\tau^2} + \frac{4a \exp(-2\tau)}{(1 - \exp(-2\tau))^2} \tag{A.18}$$

$$= -\frac{a}{\tau^2} + \frac{4a \exp(-2\tau)}{1 - 2 \exp(-2\tau) + \exp(-4\tau)} \tag{A.19}$$

The limit of this second derivative as $\tau \to 0$ is $-\frac{a}{3}$ and as $\tau \to \infty$ is 0. With $a > 0$ we

can show that the second derivative is always negative:

$$-\frac{a}{\tau^2}+\frac{4a\exp(-2\tau)}{1-2\exp(-2\tau)+\exp(-4\tau)}<0 \tag{A.20}$$

$$\frac{4\exp(-2\tau)}{1-2\exp(-2\tau)+\exp(-4\tau)}<\frac{1}{\tau^2} \tag{A.21}$$

$$4\tau^2\exp(-2\tau)<1-2\exp(-2\tau)+\exp(-4\tau) \tag{A.22}$$

$$(4\tau^2+2)\exp(-2\tau)<1+\exp(-4\tau) \tag{A.23}$$

$$2\tau^2+1<\frac{\exp(2\tau)+\exp(-2\tau)}{2} \tag{A.24}$$

$$2\tau^2+1<\cosh(2\tau) \tag{A.25}$$

$$2\tau^2+1<1+\frac{4\tau^2}{2}+\frac{16\tau^4}{24}+\cdots=\sum_{n=0}^{\infty}\frac{(2\tau)^{2n}}{(2n)!} \tag{A.26}$$

$$2\tau^2+1<1+2\tau^2+\frac{4\tau^4}{6}+\cdots=\sum_{n=0}^{\infty}\frac{(2\tau)^{2n}}{(2n)!} \tag{A.27}$$

$$0<\frac{4\tau^4}{6}+\cdots=\sum_{n=2}^{\infty}\frac{(2\tau)^{2n}}{(2n)!}\,. \tag{A.28}$$

The last statement is true since the infinite series is over strictly positive numbers because of the powers of even numbers $2n$. Therefore the second derivative is strictly negative with a limit of 0 for $\tau\to\infty$. This means that the first derivative is monotonically decreasing with a starting point (in the limit for $\tau\to 0$) of $b\mu^T\mu_0$ and an ending point at $b\mu^T\mu_0-a$ for $\tau\to\infty$. That in turn means that the first derivative has exactly one zero crossing (and hence the function one maximum) if $b\mu^T\mu_0\geq 0$ and none if $b\mu^T\mu_0<0$ (the largest function value is at 0). Therefore $\log f(\tau)$ is monotonically decreasing in the latter case and has a single maximum in the former.

The location of the maximum $\tau^\star$ cannot be computed in closed form, but we can use the Newton algorithm to obtain its location less than 10 iterations on average.

The locations of the zero-crossings of $g(\tau)=\log f(\tau)-\log(u)$ needed for the slice sampler are then obtained using Newtons method. Note that $g(\tau)$ has the same derivative as $\log f(\tau)$ derived in Eq. (A.15). The starting locations are set to $\tau_0^L=0.001\tau^\star$ for the left zero-crossing and to $\tau_0^R=1.5\tau^\star$ for the right zero-crossing. This ensures that Newton's method reliably converges to the desired zero crossing within a few iterations.

■ A.3 Normalizer of the Joint von-Mises-Fisher Prior for $D = 3$ and $a = 1$

We can derive a closed form normalizer for $a = 1$, $0 < b < a = 1$ and $D = 3$ dimensions:

$$Z(\mu_0, 1, b)^{-1} = \int_0^\infty \int_{\mu \in \mathbb{S}^2} (2\pi)^{1/2} \frac{\tau \exp(\tau b \mu^T \mu_0)}{2 \sinh \tau} \, \mathrm{d}\mu \, \mathrm{d}\tau \tag{A.29}$$

$$= \int_0^\infty 2^{-1/2} \pi^{1/2} \frac{\tau}{\sinh \tau} \int_{\mathbb{S}^2} \exp(\tau b \mu^T \mu_0) \, \mathrm{d}\mu \, \mathrm{d}\tau \tag{A.30}$$

$$= \int_0^\infty 2^{-1/2} \pi^{1/2} \frac{\tau}{\sinh \tau} Z^{-1}(\tau b) \, \mathrm{d}\tau \tag{A.31}$$

$$= \int_0^\infty 2^{-1/2} \pi^{1/2} \frac{\tau}{\sinh \tau} \frac{4\pi \sinh(\tau b)}{\tau b} \, \mathrm{d}\tau \tag{A.32}$$

$$= \frac{(2\pi)^{3/2}}{b} \int_0^\infty \frac{\sinh(\tau b)}{\sinh \tau} \, \mathrm{d}\tau \tag{A.33}$$

$$= \frac{2^{1/2} \pi^{5/2}}{b} \tan\left(\frac{b\pi}{2}\right) , \forall -1 < b < 1 \tag{A.34}$$

where we have used that the integral over the vMF exponential term yields the normalizer of the vMF distribution.

■ A.4 Marginal Data Distribution of the von-Mises-Fisher Distribution

For $D = 3$ dimensions and $a = 1$ we can derive a closed form normalized probability density function for the marginal distribution of the data under the prior:

$$p(x_i;\mu_0,1,b) = \int_0^\infty \int_{\mathbb{S}^2} \text{vMF}(x_i;\mu,\tau)p(\mu,\tau;\mu_0,1,b)\,\mathrm{d}\mu\,\mathrm{d}\tau \tag{A.35}$$

$$= \int_0^\infty \int_{\mathbb{S}^2} \frac{\tau}{4\pi\sinh(\tau)} \frac{b\tau}{2\pi^2} \frac{\exp\left(\tau\mu^T(x_i+b\mu_0)\right)}{\tan\left(\frac{b\pi}{2}\right)\sinh(\tau)}\,\mathrm{d}\mu\,\mathrm{d}\tau \tag{A.36}$$

$$= \frac{b}{2^3\pi^3\tan\left(\frac{b\pi}{2}\right)} \int_0^\infty \frac{\tau^2}{\sinh^2(\tau)} \int_{\mathbb{S}^2} \exp\left(\tau\mu^T(x_i+b\mu_0)\right)\mathrm{d}\mu\,\mathrm{d}\tau \tag{A.37}$$

$$= \frac{b}{2^3\pi^3\tan\left(\frac{b\pi}{2}\right)} \int_0^\infty \frac{\tau^2}{\sinh^2(\tau)} Z^{-1}\left(\tau\|x_i+b\mu_0\|_2\right)\mathrm{d}\tau \tag{A.38}$$

$$= \frac{4\pi b}{2^3\pi^3\tan\left(\frac{b\pi}{2}\right)} \int_0^\infty \frac{\tau^2\sinh(\tau\tilde{b})}{\tau\tilde{b}\sinh^2(\tau)}\,\mathrm{d}\tau \tag{A.39}$$

$$= \frac{b}{2\pi^2\tilde{b}\tan\left(\frac{b\pi}{2}\right)} \int_0^\infty \frac{\tau\sinh(\tau\tilde{b})}{\sinh^2(\tau)}\,\mathrm{d}\tau \tag{A.40}$$

$$= \frac{b}{2\pi^2\tilde{b}\tan\left(\frac{b\pi}{2}\right)} \frac{\pi(\tilde{b}\pi-\sin(\tilde{b}\pi))}{4\sin^2(\frac{\tilde{b}\pi}{2})}, 0<\tilde{b}<2 \tag{A.41}$$

$$= \frac{b}{2^3\tan\left(\frac{b\pi}{2}\right)} \frac{1-\text{sinc}(\tilde{b}\pi)}{\sin^2(\frac{\tilde{b}\pi}{2})} \tag{A.42}$$

where we have used that the integrating over the vMF exponential term yields the inverse of the normalizer of the vMF distribution and that $0 < \tilde{b} < 2$ because $0 < b < a = 1$ which is imposed by the prior distributions properties.

■ A.5 First Derivative of the $\mathbb{SO}(3)$ Exponential Map

The first derivative of the exponential map associated with $\mathbb{SO}(3)$ can be derived using the closed form expression also called the Rodrigues' formula in Eq. (2.126):

$$\begin{aligned}
\tfrac{\partial}{\partial\omega_i}\text{Exp}(\omega)\Big|_{\omega=0} &= \tfrac{\partial}{\partial\omega_i}\left(\mathrm{I} + \tfrac{\sin(||\omega||)}{||\omega||}W(\omega) + \tfrac{1-\cos||\omega||}{||\omega||^2}W^2(\omega)\right)\Big|_{\omega=0} \\
&= \tfrac{\partial}{\partial\omega_i}\tfrac{\sin(||\omega||)}{||\omega||}W(\omega)\Big|_{\omega=0} + \tfrac{\sin(||\omega||)}{||\omega||}\tfrac{\partial}{\partial\omega_i}W(\omega)\Big|_{\omega=0} \\
&+ \tfrac{\partial}{\partial\omega_i}\tfrac{1-\cos||\omega||}{||\omega||^2}W^2(\omega)\Big|_{\omega=0} + \tfrac{1-\cos||\omega||}{||\omega||^2}\tfrac{\partial}{\partial\omega_i}W^2(\omega)\Big|_{\omega=0} \\
&= G_i
\end{aligned} \tag{A.43}$$

Additionally the following limits and relationships have been used:

$$\lim_{\omega\to 0}\frac{\partial}{\partial\omega_i}\frac{\sin||\omega||}{||\omega||} = \lim_{\omega\to 0}\omega_i\frac{||\omega||\cos||\omega||-\sin||\omega||}{||\omega||^3} = -\frac{1}{3} \tag{A.44}$$

$$\lim_{x\to 0}\frac{\sin x}{x} = 1 \tag{A.45}$$

$$\lim_{\omega\to 0}\frac{\partial}{\partial\omega_i}\frac{1-\cos||\omega||}{||\omega||^2} = \lim_{\omega\to 0}\omega_i\frac{||\omega||\sin||\omega||+2\cos||\omega||-2}{||\omega||^4} = -\frac{1}{12} \tag{A.46}$$

$$\lim_{x\to 0}\frac{1-\cos x}{x^2} = \lim_{x\to 0}\frac{\sin x}{2x} = \frac{1}{2} \tag{A.47}$$

$$\lim_{\omega\to 0}\frac{\partial}{\partial\omega_i}W^2(\omega) = 0 \tag{A.48}$$

$$\frac{\partial}{\partial\omega_i}W(\omega) = G_i \tag{A.49}$$

$$W(0) = W^2(0) = 0 \tag{A.50}$$

The derivative of the transpose of the exponential map is readily derived by noting $W^T = -W$ because of skew-symmetry:

$$\begin{aligned}\frac{\partial}{\partial\omega_i}\mathrm{Exp}(\omega)^T\Big|_{\omega=0} &= \frac{\partial}{\partial\omega_i}\left(\mathrm{I} - \frac{\sin(||\omega||)}{||\omega||}W(\omega) + \frac{1-\cos||\omega||}{||\omega||^2}W^2(\omega)\right)\Big|_{\omega=0} \\ &= -G_i\end{aligned} \tag{A.51}$$

■ A.6 Second Derivative of the $\mathbb{SO}(3)$ Exponential Map

The second derivative of the exponential map associated with $\mathbb{SO}(3)$ is derived using the closed form expression for the map in Eq. (2.126):

$$\begin{aligned}&\frac{\partial^2}{\partial\omega_j\partial\omega_i}\exp(W(\omega))\Big|_{\omega=0} = \frac{\partial^2}{\partial\omega_j\partial\omega_i}\left(\mathrm{I} + \frac{\sin(||\omega||)}{||\omega||}W + \frac{1-\cos||\omega||}{||\omega||^2}W^2\right)\Big|_{\omega=0} \\ &= \frac{\partial}{\partial\omega_j}\left(\frac{\partial}{\partial\omega_i}\frac{\sin(||\omega||)}{||\omega||}W + \frac{\sin(||\omega||)}{||\omega||}\frac{\partial}{\partial\omega_i}W + \frac{\partial}{\partial\omega_i}\frac{1-\cos||\omega||}{||\omega||^2}W^2 + \frac{1-\cos||\omega||}{||\omega||^2}\frac{\partial}{\partial\omega_i}W^2\right)\Big|_{\omega=0} \\ &= \frac{\partial^2}{\partial\omega_j\partial\omega_i}\frac{\sin(||\omega||)}{||\omega||}W + \frac{\partial}{\partial\omega_i}\frac{\sin(||\omega||)}{||\omega||}\frac{\partial W}{\partial\omega_j} + \frac{\partial}{\partial\omega_j}\frac{\sin(||\omega||)}{||\omega||}\frac{\partial W}{\partial\omega_i} + \frac{\sin(||\omega||)}{||\omega||}\frac{\partial^2 W}{\partial\omega_j\partial\omega_i} \\ &+ \frac{\partial^2}{\partial\omega_j\partial\omega_i}\frac{1-\cos||\omega||}{||\omega||^2}W^2 + \frac{\partial}{\partial\omega_i}\frac{1-\cos||\omega||}{||\omega||^2}\frac{\partial W^2}{\partial\omega_j} + \frac{\partial}{\partial\omega_j}\frac{1-\cos||\omega||}{||\omega||^2}\frac{\partial W^2}{\partial\omega_i} + \frac{1-\cos||\omega||}{||\omega||^2}\frac{\partial^2 W^2}{\partial\omega_j\partial\omega_i}\Big|_{\omega=0} \\ &= \tfrac{1}{2}(G_iG_j + G_jG_i)\end{aligned} \tag{A.52}$$

where we have used relations from Sec. A.5 in addition to

$$\frac{\partial^2}{\partial\omega_i\partial\omega_j}W(\omega) = 0 \tag{A.53}$$

$$\frac{\partial^2}{\partial\omega_i\partial\omega_j}W^2(\omega) = G_iG_j + G_jG_i \tag{A.54}$$

■ A.7 First Derivative of Functions over $\mathbb{SO}(3)$

For a scalar function $f(R) : \mathbb{SO}(3) \to \mathbb{R}$, we can take the left gradient on the manifold with respect to R as

$$\begin{aligned}\frac{\partial f(R)}{\partial R} &= \frac{\partial}{\partial \omega} f(\mathrm{Exp}(\omega)R)\big|_{\omega=0} = \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T \frac{\partial}{\partial \omega}\mathrm{Exp}(\omega)R\big|_{\omega=0} \right\} \\ &= \begin{pmatrix} \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T G_1 R \right\} & \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T G_2 R \right\} & \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T G_3 R \right\} \end{pmatrix}\end{aligned} \tag{A.55}$$

In the special case of $f(Rx)$ Eq. (A.55) which is commonly encountered the derivative simplifies to

$$\begin{aligned}\frac{\partial f(Rx)}{\partial R} &= \mathrm{tr}\left\{ \frac{\partial f(p)}{\partial p}^T\Big|_{p=Rx} \frac{\partial}{\partial \omega}\mathrm{Exp}(\omega)Rx\big|_{\omega=0} \right\} = \frac{\partial f(p)}{\partial p}^T\Big|_{p=Rx} \frac{\partial}{\partial \omega}\mathrm{Exp}(\omega)Rx\big|_{\omega=0} \\ &= \frac{\partial f(p)}{\partial p}^T\Big|_{p=Rx} \begin{pmatrix} G_1 Rx & G_2 Rx & G_3 Rx \end{pmatrix} = -\frac{\partial f(p)}{\partial p}^T\Big|_{p=Rx} [Rx]_\times\end{aligned} \tag{A.56}$$

The right gradients are obtained analogously as:

$$\frac{\partial}{\partial \omega} f(R\mathrm{Exp}(\omega))\big|_{\omega=0} = \begin{pmatrix} \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T R G_1 \right\} & \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T R G_2 \right\} & \mathrm{tr}\left\{ \frac{\partial f(R)}{\partial R}^T R G_3 \right\} \end{pmatrix} \tag{A.57}$$

$$\frac{\partial}{\partial \omega} f(R\mathrm{Exp}(\omega)x)\big|_{\omega=0} = -\frac{\partial f(p)}{\partial p}^T\Big|_{p=Rx} R[x]_\times \tag{A.58}$$

As an example consider $f(R) = a^T Rx$:

$$\frac{\partial a^T Rx}{\partial R} = -\frac{\partial a^T p}{\partial p}^T [Rx]_\times = -a^T [Rx]_\times \tag{A.59}$$

Another common function appearing in a rotated Gaussian distribution is the quadratic $f(R) = x^T R^T A R x$. First note that [188] (Eq. (82))

$$\frac{\partial f(R)}{\partial R} = \frac{\partial x^T R^T A R x}{\partial R} = (A^T + A) R x x^T := \tilde{A} R x x^T . \tag{A.60}$$

Then with Eq. (A.55) we can derive the derivative as

$$\begin{aligned}\frac{\partial f(R)}{\partial R} &= \begin{pmatrix} \mathrm{tr}\left\{ (\tilde{A} R x x^T)^T G_1 R \right\} & \mathrm{tr}\left\{ (\tilde{A} R x x^T)^T G_2 R \right\} & \mathrm{tr}\left\{ (\tilde{A} R x x^T)^T G_3 R \right\} \end{pmatrix} \\ &= \begin{pmatrix} \mathrm{tr}\left\{ x x^T R^T \tilde{A}^T G_1 R \right\} & \mathrm{tr}\left\{ x x^T R^T \tilde{A}^T G_2 R \right\} & \mathrm{tr}\left\{ x x^T R^T \tilde{A}^T G_3 R \right\} \end{pmatrix} \\ &= \begin{pmatrix} x^T R^T \tilde{A}^T G_1 R x & x^T R^T \tilde{A}^T G_2 R x & x^T R^T \tilde{A}^T G_3 R x \end{pmatrix} .\end{aligned} \tag{A.61}$$

■ A.8 Second Derivative of Functions over $\mathbb{SO}(3)$

The second derivative, the so called Hessian, can be useful for second order optimization methods such as Newton's method and to get a covariance estimate. The approach is the same as for the gradient. The Hessian is computed as the matrix of all combinations of second derivatives

$$H = \frac{\partial^2}{\partial R^2} f(R) = \begin{pmatrix} \frac{\partial^2 f(R)}{\partial \omega_1 \partial \omega_1} & \frac{\partial^2 f(R)}{\partial \omega_1 \partial \omega_2} & \frac{\partial^2 f(R)}{\partial \omega_1 \partial \omega_3} \\ \frac{\partial^2 f(R)}{\partial \omega_2 \partial \omega_1} & \frac{\partial^2 f(R)}{\partial \omega_2 \partial \omega_2} & \frac{\partial^2 f(R)}{\partial \omega_2 \partial \omega_3} \\ \frac{\partial^2 f(R)}{\partial \omega_3 \partial \omega_1} & \frac{\partial^2 f(R)}{\partial \omega_3 \partial \omega_2} & \frac{\partial^2 f(R)}{\partial \omega_3 \partial \omega_3} \end{pmatrix} \tag{A.62}$$

where the individual second right derivatives are computed as

$$\begin{aligned} \frac{\partial^2 f(R)}{\partial \omega_j \partial \omega_i} &= \frac{\partial}{\partial \omega_j} \operatorname{tr} \left\{ \frac{\partial f(R)}{\partial R}^T R \frac{\partial}{\partial \omega_i} \operatorname{Exp}(\omega) \right\} \Big|_{\omega=0} \\ &= \operatorname{tr} \left\{ \frac{\partial}{\partial \omega_j} \left(\frac{\partial f(R)}{\partial R}^T \right) R \frac{\partial}{\partial \omega_i} \operatorname{Exp}(\omega) + \frac{\partial f(R)}{\partial R}^T R \frac{\partial^2}{\partial \omega_j \partial \omega_i} \operatorname{Exp}(\omega) \right\} \Big|_{\omega=0} \\ &= \operatorname{tr} \left\{ \left(\frac{\partial}{\partial \omega_j} \frac{\partial f(R)}{\partial R} \right)^T \Big|_{\omega=0} RG_i + \frac{\partial f(R)}{\partial R}^T \tfrac{1}{2} R(G_i G_j + G_j G_i) \right\} \end{aligned} \tag{A.63}$$

The left derivative is:

$$\frac{\partial^2 f(R)}{\partial \omega_j \partial \omega_i} = \operatorname{tr} \left\{ \left(\frac{\partial}{\partial \omega_j} \frac{\partial f(R)}{\partial R} \right)^T \Big|_{\omega=0} G_i R + \frac{\partial f(R)}{\partial R}^T \tfrac{1}{2} (G_i G_j + G_j G_i) R \right\} \tag{A.64}$$

For example consider again $f(R) = a^T R b$. Since $\frac{\partial}{\partial \omega_j} \frac{\partial f(R)}{\partial R} = \frac{\partial}{\partial \omega_j} a b^T = 0$:

$$\frac{\partial^2 f(R(\omega))}{\partial \omega_j \partial \omega_i} = \tfrac{1}{2} a^T R (G_i G_j + G_j G_i) b \tag{A.65}$$

And the second derivative of $f(R) = b^T R^T A R b$ is:

$$\begin{aligned} \frac{\partial^2 f(R(\omega))}{\partial \omega_j \partial \omega_i} &= \operatorname{tr} \left\{ \left(\frac{\partial}{\partial \omega_j} \tilde{A} R b b^T \right)^T \Big|_{\omega=0} RG_i + \tfrac{1}{2} (\tilde{A} R b b^T)^T R (G_i G_j + G_j G_i) \right\} \\ &= \operatorname{tr} \left\{ \left(\tilde{A} R G_j b b^T \right)^T RG_i + \tfrac{1}{2} b b^T R^T \tilde{A}^T R (G_i G_j + G_j G_i) \right\} \\ &= b^T \left(G_j^T R^T \tilde{A}^T R G_i + \tfrac{1}{2} R^T \tilde{A}^T R (G_i G_j + G_j G_i) \right) b \end{aligned} \tag{A.66}$$

■ A.9 Derivatives Involving the $\mathbb{SE}(3)$ Exponential Map

As stated in Sec. 2.7.2, the exponential map for $\mathbb{SE}(3)$ can be computed in closed form as:

$$\operatorname{Exp} \begin{pmatrix} \omega_R \\ \omega_t \end{pmatrix} = \begin{pmatrix} R(\omega_R) & V(\omega_R)\omega_t \\ 0 & 1 \end{pmatrix} \tag{A.67}$$

where

$$R(\omega_R) = \mathrm{Exp}(\omega_R) \tag{A.68}$$

$$V(\omega_R) = \mathrm{I} + \frac{1-\cos\theta}{\theta^2}[\omega_R]_\times + \frac{\theta-\sin\theta}{\theta^3}[\omega_R]_\times[\omega_R]_\times \tag{A.69}$$

Before deriving several important derivatives involving the exponential map we analyze function $V(\omega_R)$ at $\omega_R = 0$ to ease further derivations.

■ A.9.1 Analysis of $V(\omega)$ around $\omega = 0$

Around $\omega = 0$ the function $V(\omega)$ behaves as:

$$\lim_{\omega_R\to 0} V(\omega) = \lim_{\omega_R\to 0} \mathrm{I} + \frac{1-\cos\theta}{\theta^2}[\omega_R]_\times + \frac{\theta-\sin\theta}{\theta^3}[\omega_R]_\times[\omega_R]_\times \tag{A.70}$$

$$= \mathrm{I} + \frac{1}{2}[0]_\times + \frac{1}{6}[0]_\times[0]_\times = \mathrm{I}\,, \tag{A.71}$$

where we have used limits from Sec. A.5.

The derivative of $V(\omega_R)$ at $\omega_R = 0$ is:

$$\frac{\partial}{\partial\omega_i}V(\omega)\Big|_{\omega=0} = \frac{\partial}{\partial\omega_i}\left(\mathrm{I} + \tfrac{1-\cos\|\omega\|}{\|\omega\|^2}[\omega_R]_\times + \tfrac{\|\omega\|-\sin\|\omega\|}{\|\omega\|^3}[\omega_R]_\times[\omega_R]_\times\right)\Big|_{\omega=0} \tag{A.72}$$

$$= \frac{\partial}{\partial\omega_i}\tfrac{1-\cos\|\omega\|}{\|\omega\|^2}[\omega_R]_\times + \frac{\partial}{\partial\omega_i}\tfrac{\|\omega\|-\sin\|\omega\|}{\|\omega\|^3}[\omega_R]_\times[\omega_R]_\times\Big|_{\omega=0} \tag{A.73}$$

$$= \frac{\partial}{\partial\omega_i}\tfrac{1-\cos\|\omega\|}{\|\omega\|^2}[\omega_R]_\times + \tfrac{1-\cos\|\omega\|}{\|\omega\|^2}\frac{\partial}{\partial\omega_i}[\omega_R]_\times\Big|_{\omega=0} \tag{A.74}$$

$$+\frac{\partial}{\partial\omega_i}\tfrac{\|\omega\|-\sin\|\omega\|}{\|\omega\|^3}[\omega_R]_\times[\omega_R]_\times + \tfrac{\|\omega\|-\sin\|\omega\|}{\|\omega\|^3}\frac{\partial}{\partial\omega_i}[\omega_R]_\times[\omega_R]_\times\Big|_{\omega=0} \tag{A.75}$$

$$= \tfrac{1}{2}G_i\,, \tag{A.76}$$

where we have used

$$\frac{\partial}{\partial\omega_i}\|\omega\| = \frac{\omega_i}{\|\omega\|} \tag{A.77}$$

$$\frac{\partial}{\partial\omega_i}\|\omega\|^3 = 3\|\omega\|\omega_i \tag{A.78}$$

$$\lim_{\omega\to 0}\frac{\|\omega\|-\sin\|\omega\|}{\|\omega\|^3} = \frac{1}{6} \tag{A.79}$$

$$\lim_{\omega\to 0}\frac{\partial}{\partial\omega_i}\frac{\|\omega\|-\sin\|\omega\|}{\|\omega\|^3} = \lim_{\omega\to 0}\frac{\|\omega\|^3\frac{\partial}{\partial\omega_i}(\|\omega\|-\sin\|\omega\|)-(\|\omega\|-\sin\|\omega\|)\frac{\partial}{\partial\omega_i}\|\omega\|^3}{\|\omega\|^6} \tag{A.80}$$

$$= \lim_{\omega\to 0}\frac{\|\omega\|^3\frac{\partial}{\partial\omega_i}(\|\omega\|-\sin\|\omega\|)-(\|\omega\|-\sin\|\omega\|)\frac{\partial}{\partial\omega_i}\|\omega\|^3}{\|\omega\|^6} \tag{A.81}$$

$$= \lim_{\omega\to 0}\frac{\|\omega\|^3\frac{\omega_i}{\|\omega\|}(1-\cos\|\omega\|)-(\|\omega\|-\sin\|\omega\|)3\|\omega\|\omega_i}{\|\omega\|^6} \tag{A.82}$$

$$= \lim_{\omega\to 0}\omega_i\frac{\|\omega\|(1-\cos\|\omega\|)-3(\|\omega\|-\sin\|\omega\|)}{\|\omega\|^5} \tag{A.83}$$

$$= \lim_{\omega\to 0}\omega_i\frac{-\|\omega\|\cos\|\omega\|-2\|\omega\|+3\sin\|\omega\|}{\|\omega\|^5} = 0 \tag{A.84}$$

$$\lim_{\omega\to 0}\frac{\partial}{\partial\omega_i}[\omega_R]_\times[\omega_R]_\times = \lim_{\omega\to 0}\frac{\partial}{\partial\omega_i}\begin{pmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{pmatrix}\begin{pmatrix}0 & -\omega_3 & \omega_2\\ \omega_3 & 0 & -\omega_1\\ -\omega_2 & \omega_1 & 0\end{pmatrix} \tag{A.85}$$

$$= 0 \tag{A.86}$$

The last limit follows from the fact that the product of the skew matrices results in a matrix with only quadratic terms in ω.

Similar to the derivation in Sec. A.5 using the properties of the generator matrices G_i of $\mathbb{SO}(3)$ we have

$$\frac{\partial}{\partial\omega}V(\omega)p\big|_{\omega=0} = -\tfrac{1}{2}[p]_\times \tag{A.87}$$

■ A.9.2 Derivative of $\mathrm{Exp}(\omega)p$

The derivative of the exponential map directly transforming a point p is:

$$\frac{\partial}{\partial\omega}\mathrm{Exp}\begin{pmatrix}\omega_R\\ \omega_t\end{pmatrix}p = \frac{\partial}{\partial\omega}\begin{pmatrix}R(\omega_R) & V(\omega_R)\omega_t\\ 0 & 1\end{pmatrix}p \tag{A.88}$$

$$= \frac{\partial}{\partial\omega}\left(R(\omega_R)p + V(\omega_R)\omega_t\right) \tag{A.89}$$

$$= \begin{pmatrix}-[p]_\times & \mathrm{I}\end{pmatrix}, \tag{A.90}$$

where we have used the partial derivatives with respect to ω_R and ω_t:

$$\frac{\partial}{\partial\omega_R}\left(R(\omega_R)p + V(\omega_R)\omega_t\right) = \frac{\partial}{\partial\omega_R}R(\omega_R)p + \frac{\partial}{\partial\omega_R}V(\omega_R)\omega_t\Big|_{\omega=0} \tag{A.91}$$

$$= -[p]_\times - \tfrac{1}{2}[\omega_t]_\times\big|_{\omega=0} \tag{A.92}$$

$$= -[p]_\times \tag{A.93}$$

and

$$\frac{\partial}{\partial \omega_t}\left(R(\omega_R)p + V(\omega_R)\omega_t\right) = \frac{\partial}{\partial \omega_t} V(\omega_R)\omega_t \Big|_{\omega=0} = V(\omega_R)|_{\omega=0} = \mathrm{I} \tag{A.94}$$

■ A.9.3 Derivative of $T\mathrm{Exp}(\omega)p$

The derivative of a change $\mathrm{Exp}(\omega)$ (right multiplied) in the translation T transforming a point p is:

$$\frac{\partial}{\partial \omega} T\mathrm{Exp}\begin{pmatrix}\omega_R \\ \omega_t\end{pmatrix} p = \frac{\partial}{\partial \omega} T \begin{pmatrix} R(\omega_R) & V(\omega_R)\omega_t \\ 0 & 1 \end{pmatrix} p \tag{A.95}$$

$$= \frac{\partial}{\partial \omega} \begin{pmatrix} R & t \\ 0 & 1 \end{pmatrix} \begin{pmatrix} R(\omega_R) & V(\omega_R)\omega_t \\ 0 & 1 \end{pmatrix} p \tag{A.96}$$

$$= \frac{\partial}{\partial \omega} \begin{pmatrix} RR(\omega_R) & RV(\omega_R)\omega_t + t \\ 0 & 1 \end{pmatrix} p \tag{A.97}$$

$$= \frac{\partial}{\partial \omega} \left(RR(\omega_R)p + RV(\omega_R)\omega_t + t\right) \tag{A.98}$$

$$= \begin{pmatrix} -R[p]_\times & R \end{pmatrix} \tag{A.99}$$

with

$$\frac{\partial}{\partial \omega_R}\left(RR(\omega_R)p + RV(\omega_R)\omega_t + t\right) = \frac{\partial}{\partial \omega_R}\left(RR(\omega_R)p + RV(\omega_R)\omega_t\right) \tag{A.100}$$

$$= R\frac{\partial}{\partial \omega_R} R(\omega_R)p \Big|_{\omega=0} \tag{A.101}$$

$$= -R[p]_\times \tag{A.102}$$

and

$$\frac{\partial}{\partial \omega_t}\left(RR(\omega_R)p + RV(\omega_R)\omega_t + t\right) = \frac{\partial}{\partial \omega_t} RV(\omega_R)\omega_t = RV(\omega_R)|_{\omega=0} = R\,. \tag{A.103}$$

■ A.9.4 Derivative of $(T\mathrm{Exp}(\omega))^{-1}p$

The derivative of a change $\mathrm{Exp}(\omega)$ (right multiplied) in the translation T inverse-transforming a point p is:

$$\frac{\partial}{\partial\omega}\left(T\mathrm{Exp}\begin{pmatrix}\omega_R\\ \omega_t\end{pmatrix}\right)^{-1}p = \frac{\partial}{\partial\omega}\left(T\begin{pmatrix}R(\omega_R) & V(\omega_R)\omega_t\\ 0 & 1\end{pmatrix}\right)^{-1}p \tag{A.104}$$

$$= \frac{\partial}{\partial\omega}\left(\begin{pmatrix}R & t\\ 0 & 1\end{pmatrix}\begin{pmatrix}R(\omega_R) & V(\omega_R)\omega_t\\ 0 & 1\end{pmatrix}\right)^{-1}p \tag{A.105}$$

$$= \frac{\partial}{\partial\omega}\begin{pmatrix}RR(\omega_R) & RV(\omega_R)\omega_t + t\\ 0 & 1\end{pmatrix}^{-1}p \tag{A.106}$$

$$= \frac{\partial}{\partial\omega}\begin{pmatrix}R(\omega_R)^TR^T & -R(\omega_R)^TR^T\left(RV(\omega_R)\omega_t + t\right)\\ 0 & 1\end{pmatrix}p \tag{A.107}$$

$$= \frac{\partial}{\partial\omega}\begin{pmatrix}R(\omega_R)^TR^T & -R(\omega_R)^TV(\omega_R)\omega_t - R(\omega_R)^TR^Tt\\ 0 & 1\end{pmatrix}p \tag{A.108}$$

$$= \frac{\partial}{\partial\omega}\left(R(\omega_R)^TR^Tp - R(\omega_R)^TV(\omega_R)\omega_t - R(\omega_R)^TR^Tt\right)\Big|_{\omega=0} \tag{A.109}$$

$$= \begin{pmatrix}\left[R^T(p-t)\right]_\times & -\mathrm{I}\end{pmatrix} \tag{A.110}$$

where we have used the partial derivatives

$$\frac{\partial}{\partial\omega_R}\left(R(\omega_R)^TR^T(p-t) - R(\omega_R)^TV(\omega_R)\omega_t\right)\Big|_{\omega=0} = \left[R^T(p-t)\right]_\times \tag{A.111}$$

and

$$\frac{\partial}{\partial\omega_t}\left(R(\omega_R)^TR^T(p-t) - R(\omega_R)^TV(\omega_R)\omega_t\right)\Big|_{\omega=0} = -\frac{\partial}{\partial\omega_t}R(\omega_R)^TV(\omega_R)\omega_t\Big|_{\omega=0} \tag{A.112}$$

$$= -R(\omega_R)^TV(\omega_R)\big|_{\omega=0} \tag{A.113}$$

$$= -\mathrm{I} \tag{A.114}$$

Appendix B

Derivations Pertaining to Directional Clustering

■ B.1 Proof of Laplace Approximation on General Differentiable Manifolds

Lemma B.1.1 (Integration on a Manifold)**.** *Suppose $M \subset \mathbb{R}^n$ is an m-dimensional differentiable manifold given by the parametric form $g : \mathbb{R}^m \to \mathbb{R}^n$, where $g \in C^1$, and $g(A) = M$ for some measurable $A \subset \mathbb{R}^m$, and let $f : \mathbb{R}^n \to \mathbb{R}$ be an integrable function on M. Then*

$$\int_M f(x) = \int_A f(g(a)) \sqrt{\det Dg^T Dg}$$
$$Dg = \begin{pmatrix} \frac{\partial g_1}{\partial a_1} & \cdots & \frac{\partial g_1}{\partial a_m} \\ \ddots & & \ddots \\ \frac{\partial g_n}{\partial a_1} & \cdots & \frac{\partial g_n}{\partial a_m} \end{pmatrix}. \tag{B.1}$$

Theorem B.1.1 (Manifold Laplace Approximation)**.** *Suppose $M \subset \mathbb{R}^n$ is a bounded m-dimensional differentiable manifold and $f : \mathbb{R}^n \to \mathbb{R}$ is a smooth function on M. Further, suppose f has a unique global maximum on M, $x^\star = \arg\max_{x \in M} f(x)$. Then*

$$\lim_{\tau \to \infty} \frac{\int_M e^{\tau f(x)}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}} e^{\tau f(x^\star)}} = 1 \tag{B.2}$$

where $U \in \mathbb{R}^{n \times m}$ is a matrix whose columns are an orthonormal basis for $T_{x^\star} M$.

Proof. Suppose $\{u_i\}_{i=1}^m$, $u_i \in \mathbb{R}^n$ is an orthonormal basis for $T_{x^\star} M$, and add to it any orthonormal completion $\{u_i\}_{i=m+1}^n$ to $\mathbb{R}^n$. Then $U = \begin{bmatrix} u_1 & \dots & u_m \end{bmatrix} \in \mathbb{R}^{n \times m}$ is a matrix that maps $\mathbb{R}^m \to T_{x^\star} M$. Finally, define $g : V \to M$ as the transformed exponential map $g(v) = \exp_{x^\star}(Uv)$, where $UV \subset T_{x^\star} M$ is the local domain of validity of the exponential map. Then by Lemma B.1.1,

$$\int_M e^{\tau f(x)} = \int_{g(V)} e^{\tau f(x)} + \int_{M \setminus g(V)} e^{\tau f(x)} = \int_V e^{\tau h(v)} \sqrt{\det Dg^T Dg} + \int_{M \setminus g(V)} e^{\tau f(x)}. \tag{B.3}$$

where $h(v) \equiv f(g(v))$. First, note that

$$[Dg]_{ij}(0) = \frac{\partial g_i}{\partial v_j}(0) = \lim_{h\to 0} \frac{\exp_{x^\star}(U1_j h) - \exp_{x^\star}(0)}{h} \tag{B.4}$$

$$= \lim_{h\to 0} \frac{\exp_{x^\star}(u_j h) - \exp_{x^\star}(0)}{h} \tag{B.5}$$

$$= U_{ij}, \tag{B.6}$$

and thus by the fact that U is unitary and hence $U^T U = I$, $\sqrt{\det Dg(0)^T Dg(0)} = 1$. Next, using a second-order Taylor expansion of h,

$$h(v) \simeq h(0) + \nabla_v h(0)^T v + \frac{1}{2} v^T \nabla_v^2 h(0) v \tag{B.7}$$

Note that $h(0) = f(x^\star)$,

$$\left.\frac{\partial h}{\partial v_j}\right|_0 = \sum_{k=1}^n \left.\frac{\partial f}{\partial x_k}\frac{\partial g_k}{\partial v_j}\right|_0 = \sum_{k=1}^n U_{kj} \left.\frac{\partial f}{\partial x_k}\right|_{x^\star} = u_j^T \nabla_x f(x^\star) \tag{B.8}$$

and since f reaches a maximum at $x^\star$ on M, and u_j span $T_{x^\star}M$,

$$\nabla_v h(0) = 0. \tag{B.9}$$

Further,

$$\frac{\partial^2 h}{\partial v_i v_j} = \sum_{k=1}^n \frac{\partial f}{\partial x_k}\frac{\partial^2 g_k}{\partial v_i v_j} + \sum_{l=1}^n \frac{\partial^2 f}{\partial x_k x_l}\frac{\partial g_k}{\partial v_j}\frac{\partial g_l}{\partial v_i} \tag{B.10}$$

once again, since f reaches a maximum at $x^\star$ on M, and $\frac{\partial^2 g_k}{\partial v_i v_j}$ has columns contained in $T_{x^\star}M$, the first term is zero and thus

$$\left.\frac{\partial^2 h}{\partial v_i v_j}\right|_0 = \left.\sum_{k=1}^n \sum_{l=1}^n \frac{\partial^2 f}{\partial x_k x_l} U_{kj} U_{li}\right|_{x^\star} \tag{B.11}$$

so

$$\nabla_v^2 h(0) = U^T \nabla_x^2 f(x^\star) U. \tag{B.12}$$

Returning to the integral from earlier,

$$\lim_{\tau\to\infty} \frac{\int_M e^{\tau f(x)}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}} e^{\tau f(x^\star)}} \tag{B.13}$$

$$= \lim_{\tau\to\infty} \frac{\int_V e^{\tau h(v)} \sqrt{\det Dg^T Dg} + \int_{M\setminus g(V)} e^{\tau f(x)}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}} e^{\tau f(x^\star)}} \tag{B.14}$$

$$= \lim_{\tau\to\infty} \frac{\int_V e^{\tau(h(v)-f(x^\star))} \sqrt{\det Dg^T Dg} + \int_{M\setminus g(V)} e^{\tau(f(x)-f(x^\star))}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}}}. \tag{B.15}$$

By assumption, $\max_{x \in M \setminus g(V)} f(x) \leq f(x^\star) - \epsilon$ for some $\epsilon > 0$, so using the Laplace approximation for multivariate Euclidean spaces,

$$\lim_{\tau \to \infty} \frac{\int_M e^{\tau f(x)}}{\dots} \leq \lim_{\tau \to \infty} \frac{\int_V e^{\tau(h(v) - f(x^\star))} \sqrt{\det Dg^T Dg} + \operatorname{vol}(M \setminus g(V))\, e^{-\tau\epsilon}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}}} \tag{B.16}$$

$$= \lim_{\tau \to \infty} \frac{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}} \sqrt{\det Dg(0)^T Dg(0)}}{\left(\frac{2\pi}{\tau}\right)^m \left|\det U^T \nabla_x^2 f(x^\star) U\right|^{\frac{1}{2}}} \tag{B.17}$$

$$= 1 \tag{B.18}$$

and since $\int_{M \setminus g(V)} e^{\tau f(x)} \geq 0$, the lower bound is also 1, and the result follows. ■

Appendix C

Derivations Pertaining to Global Point Cloud Alignment

C.1 Rotational Alignment Details

C.1.1 The Matrix $\Xi_{kk'}$

In the main text, we are given two unit vectors μ_{1k} and $\mu_{2k'}$ in $\mathbb{R}^3$. We define $\Xi_{kk'} = \Xi(\mu_{1k}, \mu_{2k'})$, where $\Xi(u, v) \in \mathbb{R}^{4\times 4}$ is defined by $u^T(q \circ v) = q^T \Xi(u, v) q$, where $u = (u_i, u_j, u_k)$, $v = (v_i, v_j, v_k)$, and $q = (q_i, q_j, q_k, q_r)$. By standard quaternion rotation formula, we have

$$\begin{aligned} u^T(q \circ v) &= \begin{bmatrix} u_i \\ u_j \\ u_k \end{bmatrix}^T \begin{bmatrix} 1-2q_j^2-2q_k^2 & 2(q_iq_j - q_kq_r) & 2(q_iq_k + q_jq_r) \\ 2(q_iq_j + q_kq_r) & 1-2q_i^2-2q_k^2 & 2(q_jq_k - q_iq_r) \\ 2(q_iq_k - q_jq_r) & 2(q_jq_k + q_iq_r) & 1-2q_i^2-2q_j^2 \end{bmatrix} \begin{bmatrix} v_i \\ v_j \\ v_k \end{bmatrix} \\ &= q_i^2(-2u_jv_j - 2u_kv_k) + q_j^2(-2u_iv_i - 2u_kv_k) + q_k^2(-2u_iv_i - 2u_jv_j) \\ &+ q_iq_j(2u_jv_i + 2u_iv_j) + q_jq_k(2u_kv_j + 2u_jv_k) + q_iq_k(2u_iv_k + 2u_kv_i) \\ &+ q_iq_r(2u_kv_j - 2u_jv_k) + q_jq_r(2u_iv_k - 2u_kv_i) + q_kq_r(2u_jv_i - 2u_iv_j) + u^Tv \end{aligned}$$

Rearranging the quadratic expression in q into the form q^TMq, we find the formula for $\Xi(u, v)$:

$$\Xi(u,v) = \begin{bmatrix} u_iv_i - u_jv_j - u_kv_k & u_jv_i + u_iv_j & u_iv_k + u_kv_i & u_kv_j - u_jv_k \\ u_jv_i + u_iv_j & u_jv_j - u_iv_i - u_kv_k & u_jv_k + u_kv_j & u_iv_k - u_kv_i \\ u_iv_k + u_kv_i & u_jv_k + u_kv_j & u_kv_k - u_iv_i - u_jv_j & u_jv_i - u_iv_j \\ u_kv_j - u_jv_k & u_iv_k - u_kv_i & u_jv_i - u_iv_j & u^Tv \end{bmatrix}$$

C.1.2 Quadratic Upper Bound on f

First, for any $z \in [a, b]$ where $0 \le a \le b$, we can express z^2 as a convex combination of a^2 and b^2, i.e.

$$z^2 = \lambda a^2 + (1-\lambda)b^2 \implies \lambda = \frac{z^2 - a^2}{b^2 - a^2} \tag{C.1}$$

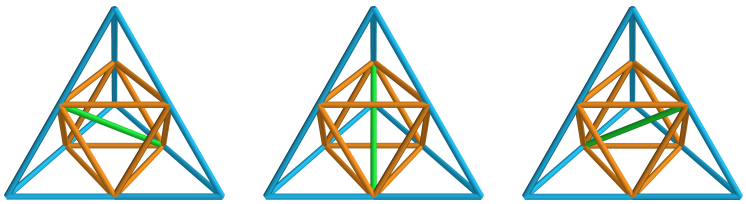


Figure C.1: The three subdivision patterns—due to the choice of the green edge—of a tetrahedron displayed in 3D. Colors designate different edge types: corner edges (blue) from an edge midpoint to a vertex; tie edges (orange) between two edge midpoints, running along a tetrahedron face; and skew edges (green) between two edge midpoints, running through the inside of the tetrahedron.

Since $f(\sqrt{z}) = \frac{e^{\sqrt{z}} - e^{-\sqrt{z}}}{\sqrt{z}}$ for $z \geq 0$ is convex (this can be shown by taking the second derivative and showing it is nonnegative), we have

$$f(z) = f\left(\sqrt{z^2}\right) = f\left(\sqrt{\lambda a^2 + (1-\lambda)b^2}\right) \tag{C.2}$$

$$\leq \lambda f(a) + (1-\lambda) f(b) \tag{C.3}$$

$$= z^2 \left(\frac{f(b) - f(a)}{b^2 - a^2}\right) + \left(\frac{b^2 f(a) - a^2 f(b)}{b^2 - a^2}\right). \tag{C.4}$$

In the main text, since we know $\ell_{kk'} \leq z_{kk'}(q) \leq u_{kk'}$ for any $q \in \mathcal{Q}$, we can use the above upper bound formula with $a = \ell_{kk'}$ and $b = u_{kk'}$.

■ C.1.3 Derivation of the γ_N Bound

Lemma C.1.1. *Let γ_N be the minimum dot product between any two tetrahedral vertices at refinement level N. Then*

$$\frac{2\gamma_{N-1}}{1 + \gamma_{N-1}} \leq \gamma_N. \tag{C.5}$$

Proof. Let the vertices of the projected tetrahedron be q_i, $i \in \{1, 2, 3, 4\}$. Let $\gamma = \min_{j \neq k} q_j^T q_k$, $\Gamma = \max_{j \neq k} q_j^T q_k$ and define the vertex between q_i and q_j as $q_{ij} = \frac{q_i + q_j}{\|q_i + q_j\|}$. Upon subdividing the tetrahedron, there are three different types of edge in the new smaller tetrahedra. Refer to Fig. C.1 for a depiction of these three types.

The first type of edge (blue in Fig. C.1) is a *corner edge* from a vertex to an edge midpoint. The cosine angle between the vertices created by a corner edge is

$$q_i^T q_{ij} = \sqrt{\frac{1 + q_i^T q_j}{2}} \geq \sqrt{\frac{1 + \gamma}{2}}. \tag{C.6}$$

The second type of edge (orange in Fig. C.1) is a *tie edge* from an edge midpoint to an edge midpoint along a face. The cosine angle between the vertices created by a tie

edge is

$$q_{ij}^T q_{ik} = \frac{1 + q_i^T q_k + q_i^T q_j + q_j^T q_k}{2\sqrt{1 + q_i^T q_j}\sqrt{1 + q_i^T q_k}} \geq \frac{1 + q_i^T q_k + q_i^T q_j + \gamma}{2\sqrt{1 + q_i^T q_j}\sqrt{1 + q_i^T q_k}} > \frac{1 + 3\gamma}{2(1 + \gamma)}\,. \tag{C.7}$$

To see the rightmost inequality, consider the minimization

$$\min_{x,y} \frac{1 + \gamma + x + y}{2\sqrt{1 + x}\sqrt{1 + y}} \quad \text{s.t. } \gamma \leq x, y \leq \Gamma\,. \tag{C.8}$$

The optimum solution is at $x = y = \gamma$, since the function is symmetric and monotonic in x, y:

$$\begin{aligned} \frac{d}{dx}\left(\frac{1 + \gamma + x + y}{2\sqrt{1 + x}\sqrt{1 + y}}\right) &= \frac{1}{4\sqrt{1 + x}\sqrt{1 + y}}\left(1 - \frac{\gamma + y}{(1 + x)}\right) \\ &> \frac{1}{4\sqrt{1 + x}\sqrt{1 + y}}\left(1 - \frac{\gamma + \Gamma}{1 + \gamma}\right) > 0. \end{aligned} \tag{C.9}$$

The final type of edge (green in Fig. C.1) is a *skew edge* from an edge midpoint to an edge midpoint through the interior of the tetrahedron. The cosine angle between vertices created by a skew edge is

$$q_{ij}^T q_{kl} = \frac{q_i^T q_k + q_i^T q_l + q_j^T q_k + q_j^T q_l}{2\sqrt{1 + q_i^T q_j}\sqrt{1 + q_k^T q_l}}\,. \tag{C.10}$$

Note that we can choose any of three skew edges in our refinement. Therefore, we can formulate bounding the skew edge dot product as a process where "nature" creates three skew edges, and we select the best one (i.e. the one of maximum dot product). Thus, in the worst case, nature solves the following problem: given a selection of a skew edge, minimize its dot product such that the other two dot products are lower (and thus nature forces us to pick that edge). Let

$$\begin{aligned} s_1 &= q_1^T q_3 + q_2^T q_4 \quad p_1 = (q_1^T q_3)(q_2^T q_4) \\ s_2 &= q_1^T q_4 + q_2^T q_3 \quad p_2 = (q_1^T q_4)(q_2^T q_3) \\ s_3 &= q_1^T q_2 + q_3^T q_4 \quad p_3 = (q_1^T q_2)(q_3^T q_4)\,. \end{aligned} \tag{C.11}$$

Then without loss of generality, we assume the ordering

$$\frac{s_1 + s_2}{2\sqrt{1 + s_3 + p_3}} \geq \frac{s_1 + s_3}{2\sqrt{1 + s_2 + p_2}} \geq \frac{s_2 + s_3}{2\sqrt{1 + s_1 + p_1}}\,. \tag{C.12}$$

Now since the function $f(x, y) = (1 + x)(1 + y)$ constrained by $x + y = c$, $x, y \geq 0$, reaches its maximum at $x = y = \frac{c}{2}$, we can reduce all of the fractions above until

$1 + s_i + p_i = (1 + s_i/2)^2$, and therefore redefining $x_i = s_i/2$, this problem is reduced to minimizing the maximum fraction of

$$\frac{x_1 + x_2}{1 + x_3} \geq \frac{x_1 + x_3}{1 + x_2} \geq \frac{x_2 + x_3}{1 + x_1}. \tag{C.13}$$

Note that while the ordering of the inequalities may switch, we can assume without loss of generality that the above holds (since we can simply redefine labels 1, 2, and 3 accordingly). Next, note that the first inequality above implies that $x_2 \geq x_3$, and the second inequality likewise implies that $x_1 \geq x_2$. Therefore, minimizing over x_1 and x_2 while keeping x_3 fixed yields

$$\frac{2x_3}{1 + x_3}. \tag{C.14}$$

And finally, minimizing over $x_3 \in [\gamma, \Gamma]$ yields

$$\max_{\text{skew edges}} q_{ij}^T q_{kl} \geq \frac{2\gamma}{1 + \gamma}. \tag{C.15}$$

For the final result of the proof, note that

$$\sqrt{\frac{1 + \gamma}{2}} \geq \frac{1 + 3\gamma}{2(1 + \gamma)} \geq \frac{2\gamma}{1 + \gamma} \quad \forall \gamma \in [0, 1]. \tag{C.16}$$

■

■ C.1.4 Proof of Theorem 1 (Rotational Convergence)

Theorem C.1.1. *Suppose $\gamma_0 = 36°$ is the initial maximum angle between vertices in the tetrahedra tessellation of $\mathbb{S}^3$, and let*

$$N \triangleq \max\left\{0, \left\lceil \log_2 \frac{\gamma_0^{-1} - 1}{\cos(\epsilon/2)^{-1} - 1} \right\rceil\right\}. \tag{C.17}$$

Then at most N refinements are required to achieve a rotational tolerance of ϵ degrees, and BB has complexity $O(\epsilon^{-6})$.

Proof. Using Lemma C.1.1, we know that the minimum dot product between any two vertices in a single cover element $\mathcal{Q}$ at refinement level N satisfies

$$\gamma_N \geq \frac{2\gamma_{N-1}}{1 + \gamma_{N-1}}. \tag{C.18}$$

This function is monotonically increasing (by taking the derivative and showing it is positive). So we recursively apply the bound:

$$\gamma_N \geq \frac{2\frac{2\gamma_{N-2}}{1+\gamma_{N-2}}}{1 + \frac{2\gamma_{N-2}}{1+\gamma_{N-2}}} = \frac{4\gamma_{N-2}}{1 + 3\gamma_{N-2}} \geq \cdots \geq \frac{2^N \gamma_0}{1 + (2^N - 1)\gamma_0}. \tag{C.19}$$

If we require a rotational tolerance of ϵ degrees, we need that $2\cos^{-1}\gamma_N \leq \epsilon$ (noting that the rotation angle between two quaternions is 2 times the angle between their vectors in $\mathbb{S}^3$). Therefore, we need

$$\gamma_N \geq \cos(\epsilon/2). \tag{C.20}$$

Using our lower bound, this is satisfied if

$$\frac{2^N\gamma_0}{1+(2^N-1)\gamma_0} \geq \cos(\epsilon/2) \implies N \geq \log_2 \frac{\gamma_0^{-1}-1}{\cos(\epsilon/2)^{-1}-1}. \tag{C.21}$$

Since N must be a nonnegative integer, the formula in Eq. (C.17) follows. At search depth M, the BB algorithm will have examined at most M tetrahedra, where

$$M = 600(1+8+8^2+\cdots+8^N) = 600\frac{8^{N+1}-1}{7} \tag{C.22}$$

Using the formula for N in Eq. (C.17) (and noting $8=2^3$), we have

$$M = O\left(\left(\frac{\gamma_0^{-1}-1}{\cos(\epsilon/2)^{-1}-1}\right)^3\right) = O\left(\left(\frac{\cos(\epsilon/2)}{1-\cos(\epsilon/2)}\right)^3\right). \tag{C.23}$$

Finally, using the Taylor expansion of cosine,

$$M = O\left(\left(\frac{1-\epsilon^2}{\epsilon^2}\right)^3\right) = O\left(\epsilon^{-6}\right). \tag{C.24}$$

■

■ C.1.5 Derivation for the $\ell_{kk'}$ and $u_{kk'}$ Optimization

We need to show that maximizing $\mu^T(q\circ\nu)$ for $q\in\mathcal{Q}$ is equivalent to maximizing $\mu^T v$ for $v=M\alpha$, $\alpha\geq 0$, $\alpha\in\mathbb{R}^4$, for some $M\in\mathbb{R}^{3\times 4}$. The following lemma establishes this fact.

Lemma C.1.2. *Let $\mathcal{Q}$ be a projected tetrahedron cover element on $\mathbb{S}^3$ with vertices q_i, $i=1,\ldots,4$, define $m\in\mathbb{R}^3$ satisfying $\|m\|=1$ (i.e. $m\in\mathbb{S}^2$), and let $\mathcal{M}$ be the set of vectors reached by rotating m by $q\in\mathcal{Q}$,*

$$\mathcal{M} \triangleq \left\{x\in\mathbb{R}^3 : x = q\circ m, q\in\mathcal{Q}\right\}. \tag{C.25}$$

Then $\mathcal{M}$ can be described as a combination of vectors in $\mathbb{R}^3$ via

$$\mathcal{M} = \left\{x\in\mathbb{R}^3 : \|x\|=1,\ x=M\alpha,\ \alpha\in\mathbb{R}^4_+\right\}. \tag{C.26}$$

where $m_i \triangleq q_i\circ m\in\mathbb{R}^3$, and $M\triangleq[m_1\cdots m_4]\in\mathbb{R}^{3\times 4}$.

Proof. In this proof, we make use of quaternion notation. If $q = xi + yj + zk + w$ is a quaternion, then its pure component is $\overrightarrow{q} = xi + yj + zk$, its scalar component is $\widetilde{q} = w$, and conjugation is denoted q^*.

To begin the proof, note that $q \in \mathcal{Q}$ implies that $q = Q\alpha$ for some $\alpha \in \mathbb{R}^4_+$, by definition. Since $q \circ m$ is a rotation of a vector, it returns a pure quaternion; thus,

$$\begin{aligned} q \circ m = \overrightarrow{q \circ m} &= \overrightarrow{\sum_{i,j} \alpha_i \alpha_j q_i m q_j^*} = \sum_{i,j} \alpha_i \alpha_j \overrightarrow{q_i m q_j^*} \\ &= \sum_{i,j} \alpha_i \alpha_j \overrightarrow{q_i m q_i^* q_i q_j^*} = \sum_{i,j} \alpha_i \alpha_j \overrightarrow{m_i q_i q_j^*} \end{aligned} \tag{C.27}$$

where α_i is the i^{th} component of α. Now note that $q_i q_j^*$ is the quaternion that rotates m_j to m_i:

$$(q_i q_j^*) \circ m_j = (q_i q_j^*) m_j (q_i q_j^*)^* = q_i q_j^* q_j m q_j^* q_j q_i^* = q_i m q_i^* = m_i. \tag{C.28}$$

Therefore, the axis of rotation of $q_i q_j^*$ is the unit vector directed along $m_j \times m_i$, and the angle is θ_{ij}. Since $m_j \times m_i = \sin(\theta_{ij})\, \widehat{m_j \times m_i}$, we have that

$$q_i q_j^* = \left(m_j \times m_i \frac{\sin(\theta_{ij}/2)}{\sin \theta_{ij}} \right)^T \begin{bmatrix} i \\ j \\ k \end{bmatrix} + \cos \frac{\theta_{ij}}{2} w\,. \tag{C.29}$$

Using this expansion along with the identity $\overrightarrow{rs} = \tilde{r}\overrightarrow{s} + \tilde{s}\overrightarrow{r} + \overrightarrow{r} \times \overrightarrow{s}$, we have that

$$\begin{aligned} q \circ m &= \sum_{i,j} \alpha_i \alpha_j \overrightarrow{m_i q_i q_j^*} \\ &= \sum_{i,j} \alpha_i \alpha_j \left(m_i \widetilde{q_i q_j^*} + m_i \times \overrightarrow{q_i q_j^*} \right) \\ &= \sum_i \alpha_i^2 m_i + \sum_{i \neq j} \alpha_i \alpha_j \left(m_i \widetilde{q_i q_j^*} + m_i \times \overrightarrow{q_i q_j^*} \right) \\ &= \textstyle\sum_i \alpha_i^2 m_i + \sum_{i<j} \alpha_i \alpha_j \left((m_i + m_j) \cos\left(\frac{\theta_{ij}}{2}\right) \right. \\ &\quad \left. + \frac{\sin(\theta_{ij}/2)}{\sin \theta_{ij}} \left(m_i \times (m_j \times m_i) + m_j \times (m_i \times m_j) \right) \right) \end{aligned} \tag{C.30}$$

Now noting that for any unit vectors $a, b \in \mathbb{R}^3$ with angle θ between them, we have

$$a \times (b \times a) = b - (\cos \theta) a \tag{C.31}$$

which can be derived from the triple product expansion identity $a \times (b \times c) = b(a \cdot c) - c(a \cdot b)$. So applying this to $m_i \times (m_j \times m_i)$ and $m_j \times (m_i \times m_j)$

$$\begin{aligned} q \circ m = \sum_i \alpha_i^2 m_i + \sum_{i<j} \alpha_i \alpha_j &\left((m_i + m_j) \cos\left(\frac{\theta_{ij}}{2}\right) \right. \\ &\left. + \frac{\sin(\theta_{ij}/2)}{\sin \theta_{ij}} \left(m_j - \cos\theta_{ij} m_i + m_i - \cos\theta_{ij} m_j \right) \right) \end{aligned} \tag{C.32}$$

and finally using the double angle formulas,

$$q \circ m = \sum_i \alpha_i^2 m_i + \sum_{i<j} \alpha_i \alpha_j \left((m_i + m_j) \sec\left(\frac{\theta_{ij}}{2}\right)\right) \tag{C.33}$$

combining, thus

$$q \circ m = \sum_{i,j} \alpha_i \alpha_j m_i \sec\left(\frac{\theta_{ij}}{2}\right) \tag{C.34}$$

Since $\sec(\theta) \geq 0 \; \forall \theta \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$, the coefficients are $\geq 0 \; \forall \theta_{ij} \in (-\pi, \pi)$. Therefore, $q \circ m$ is a linear combination of the vectors m_i with nonnegative coefficients. ■

■ C.2 Translational Alignment Derivations and Proofs

Recall that we reuse notation in this section from the rotational section to simplify the discourse and draw parallels to the rotational problem.

■ C.2.1 Linear Upper Bound on f

For any $z \in [a, b]$ where $0 \leq a \leq b$, we can express z as a convex combination of a and b, i.e.

$$z = \lambda a + (1 - \lambda) b \implies \lambda = \frac{z - a}{b - a}. \tag{C.35}$$

And, since $f(z) = e^z$ is convex,

$$f(z) = f(\lambda a + (1 - \lambda) b) \leq \lambda f(a) + (1 - \lambda) f(b) \tag{C.36}$$

$$= z \left(\frac{f(b) - f(a)}{b - a} \right) + \left(\frac{b f(a) - a f(b)}{b - a} \right). \tag{C.37}$$

In the main text, since we know $\ell_{kk'} \leq z_{kk'}(q) \leq u_{kk'}$ for any $q \in \mathcal{Q}$, we can use the above upper bound formula with $a = \ell_{kk'}$ and $b = u_{kk'}$.

■ C.2.2 Proof of Theorem 2 (Translational Convergence)

For translation, we have a similar result to Lemma C.1.1, but it is much simpler to show; the diagonal of each rectangular cell is simply $1/2$ that of the previous refinement level, i.e.

$$\frac{\gamma_{N-1}}{2} = \gamma_N = \Gamma_N = \frac{\Gamma_{N-1}}{2}. \tag{C.38}$$

Theorem C.2.1. *Suppose γ_0 is the initial diagonal of the translation cell in $\mathbb{R}^3$, and let*

$$N \triangleq \max\left\{0, \left\lceil \log_2 \frac{\gamma_0}{\epsilon} \right\rceil \right\}. \tag{C.39}$$

Then at most N refinements are required to achieve a translational tolerance of ϵ, and BB has complexity $O(\epsilon^{-3})$.

Proof. If γ_0 is the initial diagonal length, then $\gamma_N = 2^{-N}\gamma_0$. So to achieve a translational tolerance of ϵ, we need that $\gamma_N \leq \epsilon$, meaning

$$2^{-N}\gamma_0 \leq \epsilon \implies N \geq \log_2 \frac{\gamma_0}{\epsilon}. \tag{C.40}$$

Since N must be at least 0 and must be an integer, the formula in the theorem follows. As the branching factor at each refinement is 8, the BB algorithm at level N will have examined at most M cells, where

$$M = 1 + 8 + 8^2 + \cdots + 8^N = \frac{8^{N+1} - 1}{7}. \tag{C.41}$$

Substituting the result in Eq. (C.39) (and noting $8 = 2^3$), we have

$$M = O\left(\left(\frac{\gamma_0}{\epsilon}\right)^3\right) = O\left(\epsilon^{-3}\right). \tag{C.42}$$

■

Appendix D

Derivations Pertaining to Direction-aware SLAM

D.0.3 Bingham Distribution Approximation via a von-Mises-Fisher Distribution

Here we show how to approximate the Bingham distribution which is obtained as the posterior of the surfel orientation given surfel locations, with a von-Mises-Fisher distribution. Recapitulate from Sec. 5.2.2 that

$$p(n_i \mid p_i, p_{N_i}, z_{N_i}, z_i) = \prod_{j \in N_i} \mathcal{N}(n_i^T p_i; n_i^T p_j, \sigma_{\text{pl}}^2)^{\mathbb{1}_{z_i = z_j}} \tag{D.1}$$

$$\propto \exp\left(\tfrac{1}{2} \sum_{j \in N_i} \frac{-\mathbb{1}_{z_i = z_j}}{\sigma_{\text{pl}}^2} \| n_i^T (p_j - p_i) \|_2^2 \right) \tag{D.2}$$

$$\propto \exp\left(-\tfrac{1}{2} n_i^T \sum_{j \in N_i} \frac{\mathbb{1}_{z_i = z_j}}{\sigma_{\text{pl}}^2} (p_i - p_j)(p_i - p_j)^T n_i \right) \tag{D.3}$$

$$= \exp\left(-\tfrac{1}{2} n_i^T S_{ij} n_i \right) . \tag{D.4}$$

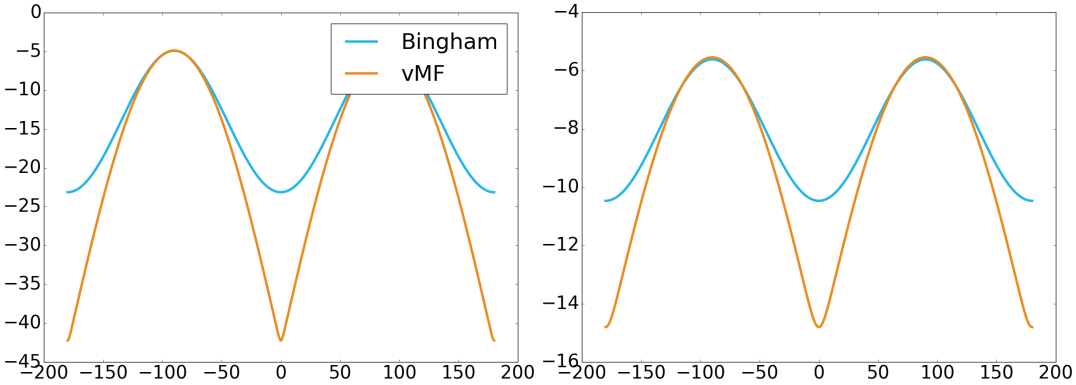
This distribution has the form of a Bingham distribution [23]. To keep in the realm of the von-Mises-Fisher distribution, we seek to approximate this Bingham with a vMF distribution. We make the assumption that the Bingham distribution is peaked about a single mode (and not fir example uniform on a great circle). This assumption is well founded since only surfel locations belonging to the same directional cluster contribute to it. This assumption manifests in the eigen decomposition of S_{ij} which then has one eigenvalue that is significantly smaller than the other two: $e_1 \ll e_2 < e_3$. The eigenvector q_1 corresponding to this eigenvalue then is the direction of the mode of the Bingham distribution. We show this via Lagrangian multipliers:

$$\arg\min_{n \in \mathbb{S}^2} -\tfrac{1}{2} n^T S n \quad \Leftrightarrow \quad \arg\min_{n, \lambda} -\tfrac{1}{2} n^T S n + \lambda (n^T n - 1) \tag{D.5}$$

The derivatives with respect to n and λ are:

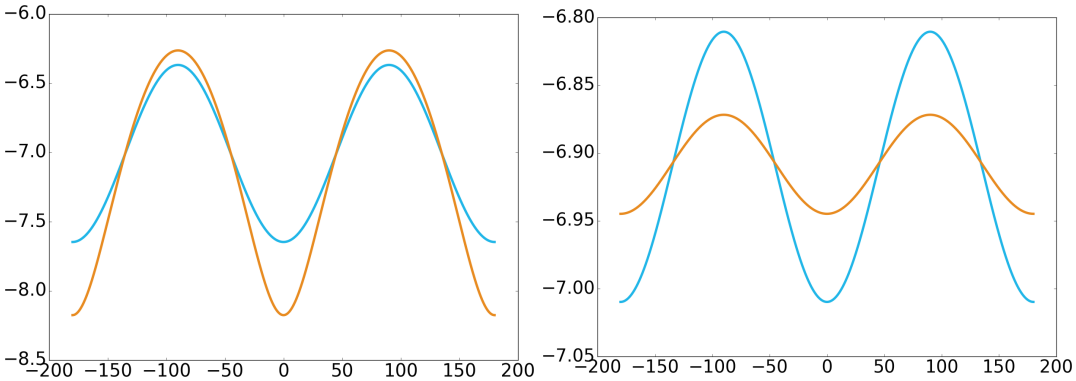
$$\frac{\partial}{\partial n} \left(-\tfrac{1}{2} n^T S n + \lambda (n^T n - 1) \right) = -Sn + \lambda n \tag{D.6}$$

$$\frac{\partial}{\partial \lambda} \left(-\tfrac{1}{2} n^T S n + \lambda (n^T n - 1) \right) = n^T n - 1 . \tag{D.7}$$



(a) Stds of 0.2m, 0.2m, 0.001m.

(b) Stds of 0.1m, 0.1m, 0.001m.



(c) Stds of 0.05m, 0.05m, 0.001m.

(d) Stds of 0.02m, 0.02m, 0.001m.

Figure D.1: Comparison of the true Bingham distribution and the approximation with two von-Mises-Fisher distributions for different standard deviations of the Bingham distribution (in log scale). The distributions shown are evaluate on a great circle around the sphere leading through both antipodal modes. Note that for small standard deviations, the both distributions are essentially uniform over the sphere as can be seen from the scale of the plot. For larger standard deviations, the approximation is close to the true distribution around the modes.

Setting the derivatives to 0:

$$Sn = \lambda n \qquad\qquad n^T n = 1\,. \tag{D.8}$$

By inspection the solution to these two equations are n equal to the eigenvectors and λ to equal to the eigenvectors of matrix S. In particular the maximum is attained at the eigenvector corresponding to the smallest eigenvalue: $n = q_1$. This motivates setting the mode of the approximating von-Mises-Fisher distribution to the eigenvector corresponding to the smallest eigenvalue: $\mu = q_1$.

The remaining unknown is the concentration τ of the approximating von-Mises-Fisher distribution. While we conjecture that there is a derivation for the value of τ as a function of the eigenvalues of S via for example KL-Divergence minimization, the involved math is hard because of the integrals over quantities on the sphere.

Instead, by simulating various matrices S from sampled Gaussian distributed vectors $p_i - p_j$ we find the relationship

$$\tau = \frac{2e_2e_2}{e_2 + e_3} \tag{D.9}$$

to yield von-Mises-Fisher distributions that closely match the Bingham distributions. In Fig. D.1 we plot the true Bingham log density in comparison to an approximation with two von-Mises-Fisher distributions (one in the opposite direction) for different realistic noise levels of S. The approximation seems valid especially around the modes of the distribution and for concentrated Bingham distributions with standard deviations in the range of 0.1m to 0.2m which is the range of standard deviations the direction-aware SLAMsystem implicitly encourages by sampling sparse surfel locations and constraining the nearest neighborhood graph to neighbors within 0.2m.

■ D.0.4 ICP Point-to-Plane Alignment Contribution

The point-to-plane cost function is

$$f_{\text{p2pl}} = \sum_{i\in\mathcal{A}} \|n_i^T({}^wT_c x_i^p - p_i)\|_2^2 \tag{D.10}$$

$$= \sum_{i\in\mathcal{A}} \|n_i^T({}^wT_c x_i^p - p_i) + n_i^T \tfrac{\partial}{\partial\omega}({}^wT_c \text{Exp}(\omega) x_i^p)\,\omega\|_2^2 \tag{D.11}$$

where, dropping the indices, the derivative is

$$n^T \tfrac{\partial}{\partial\omega} T\text{Exp}(\omega)p = \tfrac{\partial}{\partial\omega} n^T\left(RR(\omega_R)p + RV(\omega_R)\omega_t + t\right) \tag{D.12}$$

$$= \tfrac{\partial}{\partial\omega}\left(n^T RR(\omega_R)p + n^T RV(\omega_R)\omega_t + n^T t\right) \tag{D.13}$$

$$= \begin{pmatrix} -n^T R[p]_\times & n^T R \end{pmatrix}, \tag{D.14}$$

where we have used properties of the exponential map for $\mathbb{SE}(3)$ and its derivative outlined in Sec. 2.7.2. Following the strategy from Sec. 5.2.3, the rows of J and b are:

$$J_i = \begin{pmatrix} -[x_i^p]_\times {}^T {}^c R_w n_i & {}^c R_w n_i \end{pmatrix} = \begin{pmatrix} x_i^p \times ({}^c R_w n_i) & {}^c R_w n_i \end{pmatrix} \tag{D.15}$$

$$b_i = -n_i^T ({}^w T_c x_i^p - p_i) \tag{D.16}$$

■ D.0.5 ICP Photometric Term

As introduced in Sec. 5.2.3 the photometric error is the difference between the model intensity I_i at a given point p_i in world coordinates and its current observed intensity I_c:

$$f_{\mathrm{I}} = \sum_{i \in \mathcal{A}} \| I_c(\pi({}^c T_w p_i)) - I_i \|_2^2 \tag{D.17}$$

$$= \sum_{i \in \mathcal{A}} \| I_c + \nabla I_c \frac{\partial \pi(x)}{\partial x} \frac{\partial ({}^w T_c \mathrm{Exp}(\omega))^{-1} p_i}{\partial \omega} \omega - I_i \|_2^2 . \tag{D.18}$$

Dropping the indices, the derivative with respect to a small motion ω is

$$\frac{\partial (T \mathrm{Exp}(\omega))^{-1} p}{\partial \omega} = \begin{pmatrix} \left[R^T (p - t) \right]_\times & -\mathrm{I} \end{pmatrix} \tag{D.19}$$

as derived in Appendix A.9.4.

Adopting the pinhole camera model the projection function $\pi(p)$ is:

$$\pi(p) = \begin{pmatrix} \frac{p_x}{p_z} f_u + u_c \\ \frac{p_y}{p_z} f_v + u_v \end{pmatrix} , \tag{D.20}$$

where f_u and f_v are the focal lengths and u_c and v_c are the centers of the camera in u (column) and v (row) direction.

The derivative of this projection operator is

$$\frac{\partial \pi(p)}{\partial p} = \begin{pmatrix} \frac{f_u}{p_z} & 0 & -\frac{p_x}{p_z^2} f_u \\ 0 & \frac{f_v}{p_z} & -\frac{p_y}{p_z^2} f_v \end{pmatrix} . \tag{D.21}$$

Putting everything together according to the strategy outlined in Sec. 5.2.3, the rows of J and b are

$$J_i = \nabla I_c \frac{\partial \pi(p)}{\partial p} \begin{pmatrix} \left[{}^w R_c^T \left(p_i - {}^w t_c \right) \right]_\times & -\mathrm{I} \end{pmatrix} \tag{D.22}$$

$$b_i = I_i - I_c . \tag{D.23}$$

Bibliography

[1] M. Abramowitz and I. Stegun, editors. *Handbook of Mathematical Functions*. Dover Books on Mathematics. Dover Publications, 1965.

[2] P. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton University Press, 2009.

[3] D. Aiger, N. J. Mitra, and D. Cohen-Or. 4-points congruent sets for robust pairwise surface registration. In *ACM Transactions on Graphics (TOG)*, volume 27, page 85. ACM, 2008.

[4] A. Albarelli, E. Rodolà, and A. Torsello. Fast and accurate surface alignment through an isometry-enforcing game. *Pattern Recognition*, 48(7):2209–2226, 2015.

[5] S. L. Altmann. *Rotations, Quaternions, and double groups*. Courier Corporation, 2005.

[6] A. Angeli, S. Doncieux, J.-A. Meyer, and D. Filliat. Real-time visual loop-closure detection. In *ICRA*, pages 1842–1847. IEEE, 2008.

[7] M. E. Antone and S. Teller. Automatic recovery of relative camera rotations for urban scenes. In *CVPR*, volume 2, pages 282–289. IEEE, 2000.

[8] C. Antoniak. Mixtures of Dirichlet processes with applications to Bayesian nonparametric problems. *The annals of statistics*, pages 1152–1174, 1974.

[9] M. Antunes and J. P. Barreto. A global approach for the detection of vanishing points and mutually orthogonal vanishing directions. In *CVPR*, pages 1336–1343. IEEE, 2013.

[10] C. Archambeau and M. Verleysen. Manifold constrained finite Gaussian mixtures. In *Computational Intelligence and Bioinspired Systems*, pages 820–828. Springer, 2005.

[11] L. Armijo. Minimization of functions having Lipschitz continuous first partial derivatives. *Pacific Journal of mathematics*, 16(1):1–3, 1966.

[12] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, 45(6):891–923, 1998.

[13] H. Badino, D. Huber, Y. Park, and T. Kanade. Fast and accurate computation of surface normals from range images. In *ICRA*, pages 3084–3091. IEEE, 2011.

[14] A. Banerjee, I. S. Dhillon, J. Ghosh, S. Sra, and G. Ridgeway. Clustering on the unit hypersphere using von Mises-Fisher distributions. *JMLR*, 6(9), 2005.

[15] M. Bangert, P. Hennig, and U. Oelfke. Using an infinite von Mises-Fisher mixture model to cluster treatment beam directions in external radiation therapy. In *ICMLA*, pages 746–751. IEEE, 2010.

[16] S. Y. Bao, M. Bagra, Y.-W. Chao, and S. Savarese. Semantic structure from motion with points, regions, and objects. In *CVPR*, pages 2703–2710. IEEE, 2012.

[17] S. Y. Bao and S. Savarese. Semantic structure from motion. In *CVPR*, pages 2025–2032. IEEE, 2011.

[18] O. Barinova, V. Lempitsky, E. Tretiak, and P. Kohli. Geometric image parsing in man-made environments. In *ECCV*, pages 57–70. Springer, 2010.

[19] S. T. Barnard. Interpreting perspective images. *Artif. Intell.*, 21(4):435–462, Nov. 1983.

[20] J.-C. Bazin and M. Pollefeys. 3-line RANSAC for orthogonal vanishing point detection. In *IROS*, pages 4282–4287. IEEE, 2012.

[21] P. J. Besl and N. D. McKay. Method for registration of 3-D shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

[22] P. Biber and W. Straßer. The normal distributions transform: A new approach to laser scan matching. In *IROS*, volume 3, pages 2743–2748. IEEE, 2003.

[23] C. Bingham. An antipodally symmetric distribution on the sphere. *The Annals of Statistics*, 2(6):1201–1225, 1974.

[24] C. M. Bishop. *Pattern recognition and machine learning*, volume 1. Springer New York, 2006.

[25] D. Blackwell and J. B. MacQueen. Ferguson distributions via pólya urn schemes. *The annals of statistics*, pages 353–355, 1973.

[26] M. Bláha, C. Vogel, A. Richard, J. D. Wegner, T. Pock, and K. Schindler. Large-scale semantic 3D reconstruction: an adaptive multi-resolution model for multi-class volumetric labeling. In *CVPR*, 2016.

[27] D. M. Blei, M. I. Jordan, et al. Variational inference for Dirichlet process mixtures. *Bayesian analysis*, 1(1):121–143, 2006.

[28] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent Dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[29] M. Bosse, R. Rikoski, J. Leonard, and S. Teller. Vanishing points and three-dimensional lines from omni-directional video. *The Visual Computer*, 19(6):417–430, 2003.

[30] M. Bosse and R. Zlot. Map matching and data association for large-scale two-dimensional laser scan-based SLAM. *IJRR*, 27(6):667–691, 2008.

[31] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy. *Polygon mesh processing*. CRC press, 2010.

[32] N. Boumal, A. Singer, P.-A. Absil, and V. D. Blondel. Cramér–rao bounds for synchronization of rotations. *Information and Inference*, 3(1):1–39, 2014.

[33] T. Broderick, B. Kulis, and M. Jordan. MAD-Bayes: MAP-based asymptotic derivations from Bayes. In *ICML*, 2013.

[34] P. Bromiley. Products and convolutions of Gaussian probability density functions. Technical Report Tina Memo No. 2003-003, University of Manchester.

[35] R. Cabezas, J. Straub, and J. W. Fisher III. Semantically-Aware Aerial Reconstruction from Multi-Modal Data. In *ICCV*, December 2015.

[36] 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. *TRO*, 32(6):1309–1332, 2016.

[37] D. Campbell and L. Petersson. GOGMA: Globally-optimal Gaussian mixture alignment. In *CVPR*, June 2016.

[38] R. J. Campbell and P. J. Flynn. A survey of free-form object representation and recognition techniques. *Computer Vision and Image Understanding*, 81(2):166–210, 2001.

[39] T. Campbell, M. Liu, B. Kulis, J. P. How, and L. Carin. Dynamic clustering via asymptotics of the dependent Dirichlet process mixture. In *NIPS*, pages 449–457, 2013.

[40] B. Caprile and V. Torre. Using vanishing points for camera calibration. *IJCV*, 4(2):127–139, 1990.

[41] R. O. Castle, D. Gawley, G. Klein, and D. W. Murray. Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras. In *ICRA*, pages 4102–4107. IEEE, 2007.

[42] J. Chang and J. W. Fisher III. Parallel sampling of DP mixture models using sub-clusters splits. In *NIPS*, Dec 2013.

[43] J. Chang and J. W. Fisher III. MCMC sampling in HDPs using sub-clusters. In *NIPS*, Dec 2014.

[44] Y. Chen and G. Medioni. Object modeling by registration of multiple range images. In *ICRA*, pages 2724–2729. IEEE, 1991.

[45] G. S. Chirikjian. *Stochastic Models, Information Theory, and Lie Groups, Volume 2: Analytic Methods and Modern Applications*, volume 2. Springer, 2011.

[46] S. B. Choe. *Statistical analysis of orientation trajectories via Quaternions with applications to human motion*. PhD thesis, The University of Michigan, 2006.

[47] R. Cipolla, T. Drummond, and D. P. Robertson. Camera calibration from vanishing points in image of architectural scenes. In *BMVC*, volume 99, pages 382–391, 1999.

[48] R. T. Collins and R. S. Weiss. Vanishing point calculation as a statistical inference on the unit sphere. In *ICCV*, pages 400–403, 1990.

[49] J. M. Coughlan and A. L. Yuille. Manhattan world: Compass direction from a single image by Bayesian inference. In *ICCV*, volume 2, pages 941–947. IEEE, 1999.

[50] H. S. M. Coxeter. *Regular polytopes*. Courier Corporation, 1973.

[51] H. Cramér. *Mathematical Methods of Statistics*, volume 9. Princeton university press, 2016.

[52] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996.

[53] D. B. Dahl, Z. Bohannan, Q. Mo, M. Vannucci, and J. Tsai. Assessing side-chain perturbations of the protein backbone: A knowledge-based classification of residue Ramachandran space. *Journal of Molecular Biology*, 378(3):749 – 758, 2008.

[54] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. *arXiv preprint arXiv:1702.04405*, 2017.

[55] A. Dai, M. Nießner, M. Zollhöfer, S. Izadi, and C. Theobalt. BundleFusion: Real-time globally consistent 3D reconstruction using online surface re-integration. *arXiv preprint arXiv:1604.01093*, 2016.

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

[57] E. Delage, H. Lee, and A. Y. Ng. Automatic single-image 3D reconstructions of indoor Manhattan world scenes. In *Robotics Research*, pages 305–321. Springer, 2007.

[58] F. Dellaert and R. Collins. Fast image-based tracking by selective pixel integration. In *ICCV Workshop on Frame-Rate Vision*, pages 1–22, 1999.

[59] P. Denis, J. H. Elder, and F. J. Estrada. Efficient edge-based methods for estimating Manhattan frames in urban imagery. In *ECCV*, pages 197–210. Springer-Verlag, 2008.

[60] L. Devroye. *A Course in Density Estimation*. Birkhauser Boston Inc., Cambridge, MA, USA, 1987.

[61] I. S. Dhillon and D. S. Modha. Concept decompositions for large sparse text data using clustering. *Machine learning*, 42(1-2):143–175, 2001.

[62] M. P. do Carmo. *Differential geometry of curves and surfaces*, volume 2. Prentice-hall Englewood Cliffs, 1976.

[63] M. P. do Carmo. *Riemannian Geometry*. Birkhäuser Verlag, Boston, MA, 1992.

[64] J. R. Driscoll and D. M. Healy. Computing Fourier transforms and convolutions on the 2-sphere. *Advances in applied mathematics*, 15(2):202–250, 1994.

[65] E. Eade. *Monocular simultaneous localisation and mapping*. PhD thesis, Cambridge University, 2009.

[66] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard. An evaluation of the RGB-D SLAM system. In *ICRA*, pages 1691–1696. IEEE, 2012.

[67] J. Engel, T. Schöps, and D. Cremers. LSD-SLAM: Large-scale direct monocular SLAM. In *ECCV*, pages 834–849. Springer, 2014.

[68] W. Feiten, M. Lang, and S. Hirche. Rigid motion estimation using mixtures of projected Gaussians. In *FUSION*, pages 1465–1472, July 2013.

[69] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004.

[70] T. Ferguson. A Bayesian analysis of some nonparametric problems. *The annals of statistics*, pages 209–230, 1973.

[71] R. Finman, T. Whelan, M. Kaess, and J. J. Leonard. Efficient incremental map segmentation in dense RGB-D maps. In *ICRA*, pages 5488–5494. IEEE, 2014.

[72] N. Fioraio and L. Di Stefano. Joint detection, tracking and mapping by semantic bundle adjustment. In *CVPR*, pages 1538–1545, 2013.

[73] M. Fischler and R. 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.

[74] N. I. Fisher. *Statistical Analysis of Circular Data*. Cambridge University Press, 1995.

[75] R. Fisher. Dispersion on a sphere. *Proc. R. Soc. A: Math. & Phy. Sci.*, 217(1130):295–305, 1953.

[76] S. Fleishman, D. Cohen-Or, and C. T. Silva. Robust moving least-squares fitting with sharp features. In *ACM transactions on graphics (TOG)*, volume 24, pages 544–552. ACM, 2005.

[77] S. Fleishman, D. Cohen-Or, and C. T. Silva. Robust moving least-squares fitting with sharp features. In *SIGGRAPH*, pages 544–552, New York, NY, USA, 2005. ACM.

[78] P. Fletcher, C. Lu, S. Pizer, and S. Joshi. Principal geodesic analysis for the study of nonlinear statistics of shape. *IEEE TMI*, 23(8):995–1005, 2004.

[79] A. Flint, D. Murray, and I. Reid. Manhattan scene understanding using monocular, stereo, and 3D features. In *ICCV*, pages 2228–2235. IEEE, 2011.

[80] O. Freifeld, S. Hauberg, and M. J. Black. Model transport: Towards scalable transfer learning on manifolds. In *CVPR*, 2014.

[81] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Manhattan-world stereo. In *CVPR*, pages 1422–1429. IEEE, 2009.

[82] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Reconstructing building interiors from images. In *ICCV*, pages 80–87. IEEE, 2009.

[83] J. Gallier. Basics of classical Lie groups: The exponential map, Lie groups, and Lie algebras. In *Geometric Methods and Applications*, pages 367–414. Springer, 2001.

[84] N. Gelfand, N. J. Mitra, L. J. Guibas, and H. Pottmann. Robust global registration. In *Symposium on geometry processing*, volume 2, page 5, 2005.

[85] A. Gelman, J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. *Bayesian data analysis*. CRC press, 2013.

[86] Z. Ghahramani and T. L. Griffiths. Infinite latent feature models and the Indian buffet process. In *NIPS*, pages 475–482. MIT Press, 2006.

[87] B. Ghanem, A. Thabet, J. C. Niebles, and F. Caba Heilbron. Robust Manhattan frame estimation from a single RGB-D image. In *CVPR*, pages 3772–3780. IEEE, 2015.

[88] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014.

[89] B. Glocker, J. Shotton, A. Criminisi, and S. Izadi. Real-time RGB-D camera relocalization via randomized ferns for keyframe encoding. *IEEE transactions on visualization and computer graphics*, 21(5):571–583, 2015.

[90] J. S. Goddard and M. A. Abidi. Pose and motion estimation using dual Quaternion-based extended Kalman filtering. In *Photonics West'98 Electronic Imaging*, pages 189–200. International Society for Optics and Photonics, 1998.

[91] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014.

[92] S. Gopal and Y. Yang. von Mises-Fisher clustering models. In *ICML*, pages 154–162, 2014.

[93] J. C. Gower. Generalized Procrustes analysis. *Psychometrika*, 40(1):33–51, 1975.

[94] P. J. Green. Reversible jump Markov chain Monte Carlo computation and Bayesian model determination. *Biometrika*, 82(4):711–732, 1995.

[95] K. Grove and H. Karcher. How to conjugate 1-close group actions. *Mathematische Zeitschrift*, 132(1):11–20, 1973.

[96] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In *CVPR*, June 2013.

[97] M. Habbecke and L. Kobbelt. A surface-growing approach to multi-view stereo reconstruction. In *CVPR*, pages 1–8. IEEE, 2007.

[98] C. Haene, C. Zach, A. Cohen, and M. Pollefeys. Dense semantic 3D reconstruction. *TPAMI*, 2016.

[99] A. Handa, T. Whelan, J. McDonald, and A. J. Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *ICRA*, pages 1524–1531. IEEE, 2014.

[100] C. Hane, C. Zach, A. Cohen, R. Angst, and M. Pollefeys. Joint 3D scene reconstruction and class segmentation. In *CVPR*, pages 97–104, 2013.

[101] J. A. Hartigan and M. A. Wong. Algorithm 136: A k-means clustering algorithm. *Applied statistics*, pages 100–108, 1979.

[102] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, Apr. 2004.

[103] R. I. Hartley and F. Kahl. Global optimization through rotation space search. *IJCV*, 82(1):64–79, 2009.

[104] M. A. Hasnat, O. Alata, and A. Trémeau. Hierarchical 3-D von Mises-Fisher mixture model. In *Workshop on Divergences and Divergence Learning, ICML*, 2013.

[105] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[106] S. Hauberg, F. Lauze, and K. Pedersen. Unscented Kalman filtering on Riemannian manifolds. *JMIV*, pages 1–18, 2012.

[107] K. He, J. Sun, and X. Tang. Guided image filtering. In *ECCV*, pages 1–14. Springer, 2010.

[108] M. Hebert and T. Kanade. Outdoor scene analysis using range data. In *ICRA*, volume 3, pages 1426–1432. IEEE, 1986.

[109] V. Hedau, D. Hoiem, and D. Forsyth. Recovering the spatial layout of cluttered rooms. In *ICCV*, pages 1849–1856. IEEE, 2009.

[110] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments. *IJRR*, 31(5):647–663, 2012.

[111] A. Hermans, G. Floros, and B. Leibe. Dense 3D semantic mapping of indoor scenes from RGB-D images. In *ICRA*, pages 2631–2638. IEEE, 2014.

[112] D. Holz, S. Holzer, and R. B. Rusu. Real-Time Plane Segmentation using RGB-D Cameras. In *Proceedings of the RoboCup Symposium*, 2011.

[113] H. Hoppe, T. DeRose, T. Duchamp, J. McDonald, and W. Stuetzle. *Surface reconstruction from unorganized points*, volume 26. ACM, 1992.

[114] B. K. Horn. Closed-form solution of absolute orientation using unit Quaternions. *JOSA A*, 4(4):629–642, 1987.

[115] B. K. Horn. Some notes on unit Quaternions and rotation. 2001.

[116] B. K. P. Horn. Extended Gaussian images. *Proceedings of the IEEE*, 72(12):1671–1686, 1984.

[117] P. J. Huber. *Robust statistics*. Springer, 1981.

[118] T. Ibaraki. Theoretical comparisons of search strategies in branch-and-bound algorithms. *International Journal of Computer & Information Sciences*, 5(4):315–344, 1976.

[119] K. Ikeuchi. Recognition of 3-D objects using the extended Gaussian image. In *IJCAI*, pages 595–600, 1981.

[120] H. Ishwaran and M. Zarepour. Exact and approximate sum representations for the Dirichlet process. *Canadian Journal of Statistics*, 30(2):269–283, 2002.

[121] S. Jain and R. Neal. A split-merge Markov chain Monte Carlo procedure for the Dirichlet process mixture model. *Journal of Computational and Graphical Statistics*, 13:158–182, 2000.

[122] B. Jian and B. C. Vemuri. Robust point set registration using Gaussian mixture models. *TPAMI*, 33(8):1633–1645, 2011.

[123] K. Jiang, B. Kulis, and M. Jordan. Small-variance asymptotics for exponential family Dirichlet process mixture models. In *NIPS*, 2012.

[124] A. E. Johnson and M. Hebert. Surface registration by matching oriented points. In *3DIM*, pages 121–128. IEEE, 1997.

[125] A. E. Johnson and M. Hebert. Surface matching for object recognition in complex three-dimensional scenes. *Image and Vision Computing*, 16(9):635–651, 1998.

[126] M. Johnson, J. Saunderson, and A. Willsky. Analyzing Hogwild parallel Gaussian Gibbs sampling. In *NIPS*, pages 2715–2723, 2013.

[127] K. Joo, T.-H. Oh, J. Kim, and I. S. Kweon. Globally optimal Manhattan frame estimation in real-time. In *CVPR*. IEEE, 2016.

[128] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul. An introduction to variational methods for graphical models. *Machine learning*, 37(2):183–233, 1999.

[129] M. Kaess. Simultaneous localization and mapping with infinite planes. In *ICRA*, pages 4605–4611. IEEE, 2015.

[130] H. Karcher. Riemannian center of mass and mollifier smoothing. *Communications on pure and applied mathematics*, 30(5):509–541, 1977.

[131] H. Karcher. Riemannian center of mass and so called Karcher mean. *arXiv preprint arXiv:1407.2087*, 2014.

[132] M. Kazhdan. Reconstruction of solid models from oriented point sets. In *SGP*, page 73. Eurographics Association, 2005.

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

[134] J. T. Kent. The Fisher-Bingham distribution on the sphere. *Journal of the Royal Statistical Society*, pages 71–80, 1982.

[135] C. Kerl, J. Sturm, and D. Cremers. Dense visual SLAM for RGB-D cameras. In *IROS*, pages 2100–2106. IEEE, 2013.

[136] C. Khatri and K. Mardia. The von Mises-Fisher matrix distribution in orientation statistics. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 95–106, 1977.

[137] B.-s. Kim, P. Kohli, and S. Savarese. 3D scene understanding by voxel-CRF. In *ICCV*, pages 1425–1432, 2013.

[138] G. Klein and D. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR*, pages 1–10, Nara, Japan, 2007. IEEE Computer Society.

[139] J. Košecká and W. Zhang. Video compass. In *ECCV*, pages 476–490. Springer, 2002.

[140] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[141] T. Kroeger, D. Dai, and L. Van Gool. Joint vanishing point extraction and tracking. In *CVPR*, pages 2449–2457, 2015.

[142] B. Kulis and M. I. Jordan. Revisiting k-means: New algorithms via Bayesian nonparametrics. In *ICML*, 2012.

[143] A. Kundu, Y. Li, F. Dellaert, F. Li, and J. M. Rehg. Joint semantic segmentation and 3D reconstruction from monocular video. In *ECCV*, pages 703–718. Springer, 2014.

[144] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica: Journal of the Econometric Society*, pages 497–520, 1960.

[145] E. L. Lawler and D. E. Wood. Branch-and-bound methods: A survey. *Operations research*, 14(4):699–719, 1966.

[146] D. C. Lee, M. Hebert, and T. Kanade. Geometric reasoning for single image structure recovery. In *CVPR*, pages 2136–2143. IEEE, 2009.

[147] K. P. Lennox, D. B. Dahl, M. Vannucci, and J. W. Tsai. Density estimation for protein conformation angles using a bivariate von mises distribution and Bayesian nonparametrics. *JASA*, 104(486):586–596, 2009.

[148] J. J. Leonard and H. F. Durrant-Whyte. Simultaneous map building and localization for an autonomous mobile robot. In *IROS*, pages 1442–1447. Ieee, 1991.

[149] J. Lezama, R. Gioi, G. Randall, and J.-M. Morel. Finding vanishing points via point alignments in image primal and dual domains. In *CVPR*, pages 509–515, 2014.

[150] H. Li and R. Hartley. The 3D-3D registration problem revisited. In *ICCV*, pages 1–8. IEEE, 2007.

[151] D. Lin, E. Grimson, and J. Fisher. Construction of dependent Dirichlet processes based on Poisson processes. *NIPS*, 2010.

[152] A. Liu and B. Joe. Quality local refinement of tetrahedral meshes based on 8-subtetrahedron subdivision. *Mathematics of Computation of the American Mathematical Society*, 65(215):1183–1200, 1996.

[153] C. Liu, A. G. Schwing, K. Kundu, R. Urtasun, and S. Fidler. Rent3D: Floor-plan priors for monocular layout estimation. In *CVPR*, pages 3413–3421. IEEE, 2015.

[154] L. A. Liusternik and L. A. Liusternik. *Convex figures and polyhedra*. Dover Publications, 1963.

[155] S. Lloyd. Least squares quantization in pcm. *Transactions on Information Theory*, 28(2):129–137, 1982.

[156] S. Lovegrove and A. J. Davison. Real-time spherical mosaicing using whole image alignment. In *ECCV*, pages 73–86. Springer, 2010.

[157] E. Lutton, H. Maitre, and J. Lopez-Krahe. Contribution to the determination of vanishing points using hough transform. *TPAMI*, 16(4):430–438, 1994.

[158] L. Ma, C. Kerl, J. Stueckler, and D. Cremers. CPA-SLAM: Consistent plane-model alignment for direct RGB-D SLAM. In *ICRA*, May 2016. to appear.

[159] S. N. MacEachern. Estimating normal means with a conjugate style Dirichlet process prior. In *Communications in Statistics: Simulation and Computation*, 1994.

[160] S. N. MacEachern. Dependent nonparametric processes. In *ASA proceedings of the section on Bayesian statistical science*, pages 50–55, 1999.

[161] M. Magnusson, A. Lilienthal, and T. Duckett. Scan registration for autonomous mining vehicles using 3D-NDT. *Journal of Field Robotics*, 24(10):803–827, 2007.

[162] M. Magnusson, A. Nüchter, C. Lörken, A. J. Lilienthal, and J. Hertzberg. Evaluation of 3D registration reliability and speed-a comparison of ICP and NDT. In *ICRA*, pages 3907–3912. IEEE, 2009.

[163] A. Makadia, A. Patterson, and K. Daniilidis. Fully automatic registration of 3D point clouds. In *CVPR*, volume 1, pages 1297–1304. IEEE, 2006.

[164] K. V. Mardia and P. E. Jupp. *Directional statistics*, volume 494. John Wiley & Sons, 2009.

[165] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[166] K. S. Miller. On the inverse of the sum of matrices. *Mathematics Magazine*, 54(2):67–72, 1981.

[167] N. J. Mitra and A. Nguyen. Estimating surface normals in noisy point cloud data. In *Proceedings of the nineteenth annual symposium on Computational geometry*, pages 322–328. ACM, 2003.

[168] N. J. Mitra, A. Nguyen, and L. Guibas. Estimating surface normals in noisy point cloud data. *International Journal of Computational Geometry & Applications*, 14(04n05):261–276, 2004.

[169] P. Moghadam and J. F. Dong. Road direction detection based on vanishing-point tracking. In *IROS*, pages 1553–1560. IEEE, 2012.

[170] A. Monszpart, N. Mellado, G. J. Brostow, and N. J. Mitra. Rapter: Rebuilding man-made scenes with regular arrangements of planes. *ACM Trans. Graph.*, 34(4):103:1–103:12, July 2015.

[171] M. Muja and D. G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09)*, pages 331–340. INSTICC Press, 2009.

[172] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-SLAM: a versatile and accurate monocular SLAM system. *TRO*, 31(5):1147–1163, 2015.

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

[174] R. Neal. Markov chain sampling methods for Dirichlet process mixture models. *Journal of computational and graphical statistics*, 9(2):249–265, 2000.

[175] R. M. Neal. Slice sampling. *Annals of statistics*, pages 705–741, 2003.

[176] N. Neverova, D. Muselet, and A. Trémeau. 2 1/2D scene reconstruction of indoor scenes from single RGB-D images. In *Computational Color Imaging*, pages 281–295. Springer, 2013.

[177] R. A. Newcombe, A. J. Davison, S. Izadi, P. Kohli, O. Hilliges, J. Shotton, D. Molyneaux, S. Hodges, D. Kim, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. In *ISMAR*, pages 127–136. IEEE, 2011.

[178] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. DTAM: Dense tracking and mapping in real-time. In *ICCV*, pages 2320–2327. IEEE, 2011.

[179] C. V. Nguyen, S. Izadi, and D. Lovell. Modeling Kinect sensor noise for improved 3D reconstruction and tracking. In *3DIMPVT*, pages 524–530. IEEE, 2012.

[180] G. Nunez-Antonio and E. Gutiérrez-Pena. A Bayesian analysis of directional data using the von Mises-Fisher distribution. *Communications in StatisticsSimulation and Computation®*, 34(4):989–999, 2005.

[181] P. Odell and A. Feiveson. A numerical procedure to generate a sample covariance matrix. *Journal of the American Statistical Association*, 61(313):199–203, 1966.

[182] P. Orbanz and J. Buhmann. Smooth image segmentation by nonparametric Bayesian inference. *ECCV*, pages 444–457, 2006.

[183] X. Pan, J. E. Gonzalez, S. Jegelka, T. Broderick, and M. Jordan. Optimistic concurrency control for distributed unsupervised learning. In *NIPS*, pages 1403–1411, 2013.

[184] S. Paris and F. Durand. A fast approximation of the bilateral filter using a signal processing approach. In *ECCV*, pages 568–580. Springer, 2006.

[185] A. J. Parra Bustos, T.-J. Chin, and D. Suter. Fast rotation search with stereographic projections for 3D registration. In *CVPR*, pages 3930–3937. IEEE, 2014.

[186] B. Peasley, S. Birchfield, A. Cunningham, and F. Dellaert. Accurate on-line 3D occupancy grids using Manhattan world constraints. In *IROS*, pages 5283–5290. IEEE, 2012.

[187] X. Pennec. Probabilities and statistics on Riemannian manifolds: Basic tools for geometric measurements. In *NSIP*, pages 194–198, 1999.

[188] K. B. Petersen et al. The matrix cookbook.

[189] J. Pitman. Combinatorial stochastic processes. Technical Report 621, U.C. Berkeley Department of Statistics, August 2002.

[190] J. Pitman et al. Combinatorial stochastic processes. Technical report, Technical Report 621, Dept. Statistics, UC Berkeley, 2002. Lecture notes for St. Flour course, 2002.

[191] F. Pomerleau, M. Liu, F. Colas, and R. Siegwart. Challenging data sets for point cloud registration algorithms. *IJRR*, 31(14):1705–1711, Dec. 2012.

[192] C. Ramakrishnan and G. Ramachandran. Stereochemical criteria for polypeptide and protein chain conformations: Ii. allowed conformations for a pair of peptide units. *Biophysical Journal*, 5(6):909–933, 1965.

[193] C. E. Rasmussen. The infinite Gaussian mixture model. In *NIPS*, volume 12, pages 554–560, 1999.

[194] C. E. Rasmussen. Gaussian processes for machine learning. 2006.

[195] J. Reisinger, A. Waters, B. Silverthorn, and R. J. Mooney. Spherical topic models. In *ICML*, pages 903–910, 2010.

[196] X. Ren, L. Bo, and D. Fox. RGB-(D) scene labeling: Features and algorithms. In *CVPR*, pages 2759–2766. IEEE, 2012.

[197] S. Richardson and P. J. Green. On Bayesian analysis of mixtures with an unknown number of components (with discussion). *Journal of the Royal Statistical Society: series B (statistical methodology)*, 59(4):731–792, 1997.

[198] C. P. Robert and G. Casella. *Monte Carlo statistical methods*. Springer, 1999.

[199] O. Rodrigues. *Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace: et de la variation des cordonnées provenant de ces déplacements considérés indépendamment des causes qui peuvent les produire*. unkown, 1840.

[200] G. Rosman, S. Shemtov, D. Bitton, T. Nir, G. Adiv, R. Kimmel, A. Feuer, and A. M. Bruckstein. Over-parameterized optical flow using a stereoscopic constraint. In *SSVN*, volume 6667 of *LNCS*, pages 761–772, 2011.

[201] G. Rosman, Y. Wang, X.-C. Tai, R. Kimmel, and A. M. Bruckstein. Fast regularization of matrix-valued images. In *ECCV*, pages 173–186, Berlin, Heidelberg, 2012. Springer-Verlag.

[202] G. Rosman, Y. Wang, X.-C. Tai, R. Kimmel, and A. M. Bruckstein. Fast regularization of matrix-valued images. In *ECCV*, volume 7574 of *LNCS*, pages 173–186. Springer, 2012.

[203] C. Rother. A new approach to vanishing point detection in architectural environments. *Image and Vision Computing*, 20(9):647–655, 2002.

[204] M. Rouhani, F. Lafarge, and P. Alliez. Semantic segmentation of 3D textured meshes for urban scene analysis. *ISPRS Journal of Photogrammetry and Remote Sensing*, 123:124–139, 2017.

[205] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

[206] A. Roychowdhury, K. Jiang, and B. Kulis. Small-variance asymptotics for hidden Markov models. In *NIPS*, 2013.

[207] S. Rusinkiewicz and M. Levoy. Efficient variants of the ICP algorithm. In *3-D Digital Imaging and Modeling, 2001. Proceedings. Third International Conference on*, pages 145–152. IEEE, 2001.

[208] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. In *ICRA*, pages 3212–3217. IEEE, 2009.

[209] R. F. Salas-Moreno, B. Glocken, P. H. Kelly, and A. J. Davison. Dense planar SLAM. In *ISMAR*, pages 157–164. IEEE, 2014.

[210] R. F. Salas-Moreno, B. Glocker, P. H. J. Kelly, and A. J. Davison. Dense planar SLAM. In *ISMAR*, September 2014.

[211] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. SLAM++: Simultaneous localisation and mapping at the level of objects. In *CVPR*, pages 1352–1359, 2013.

[212] J. Salvi, C. Matabosch, D. Fofi, and J. Forest. A review of recent range image registration methods with accuracy evaluation. *Image and Vision computing*, 25(5):578–596, 2007.

[213] O. Saurer, F. Fraundorfer, and M. Pollefeys. Homography based visual odometry with known vertical direction and weak Manhattan world assumption. *ViCoMoR*, page 25, 2012.

[214] G. Schindler and F. Dellaert. Atlanta world: An expectation maximization framework for simultaneous low-level edge grouping and camera calibration in complex man-made environments. In *CVPR*, volume 1, pages I–203. IEEE, 2004.

[215] P. H. Schönemann. A generalized solution of the orthogonal Procrustes problem. *Psychometrika*, 31(1):1–10, 1966.

[216] J. Sethuraman. A constructive definition of Dirichlet priors. Technical report, DTIC Document, 1991.

[217] E. Simo-Serra, C. Torras, and F. Moreno-Noguer. Geodesic finite mixture models. *BMVC*, 2014.

[218] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[219] S. N. Sinha, D. Steedly, and R. Szeliski. Piecewise planar stereo for image-based rendering. In *ICCV*, 2009.

[220] S. Sommer, F. Lauze, S. Hauberg, and M. Nielsen. Manifold valued statistics, exact principal geodesic analysis and the effect of linear approximation. *ECCV*, pages 43–56, 2010.

[221] J. Straub, N. Bhandari, J. J. Leonard, and J. W. Fisher III. Real-time Manhattan world rotation estimation in 3D. In *IROS*, 2015.

[222] J. Straub, T. Campbell, J. P. How, and J. W. Fisher III. Small-variance nonparametric clustering on the hypersphere. In *CVPR*, 2015.

[223] J. Straub, T. Campbell, J. P. How, and J. W. Fisher III. Efficient global point cloud alignment using Bayesian nonparametric mixtures. In *CVPR*, 2017.

[224] J. Straub, J. Chang, O. Freifeld, and J. W. Fisher III. A Dirichlet process mixture model for spherical data. In *AISTATS*, 2015.

[225] J. Straub, G. Rosman, O. Freifeld, J. J. Leonard, and J. W. Fisher III. A mixture of Manhattan frames: Beyond the Manhattan world. In *CVPR*, 2014.

[226] J. Straub, G. Rosman, O. Freifeld, J. J. Leonard, and J. W. Fisher III. The Manhattan frame model – Manhattan world inference in the space of surface normals. In *TPAMI*, 2017.

[227] A. Strehl and J. Ghosh. Cluster ensembles—a knowledge reuse framework for combining multiple partitions. *JMLR*, 3:583–617, 2003.

[228] A. Strehl, J. Ghosh, and R. Mooney. Impact of similarity measures on web-page clustering. pages 58–64, 2000.

[229] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IROS*, pages 573–580. IEEE, 2012.

[230] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. DeepFace: Closing the gap to human-level performance in face verification. In *CVPR*, pages 1701–1708, 2014.

[231] J.-P. Tardif. Non-iterative approach for fast and accurate vanishing point detection. In *ICCV*, pages 1250–1257. IEEE, 2009.

[232] Y. Teh, M. Jordan, M. Beal, and D. Blei. Hierarchical Dirichlet processes. *JASA*, 101(476):1566–1581, 2006.

[233] Y. W. Teh. Dirichlet processes. In *Encyclopedia of Machine Learning*. Springer, New York, 2010.

[234] D. Ting, G. Wang, M. Shapovalov, R. Mitra, M. I. Jordan, and R. L. Dunbrack Jr. Neighbor-dependent Ramachandran probability distributions of amino acids developed from a hierarchical Dirichlet process model. *PLoS computational biology*, 6(4):e1000763, 2010.

[235] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *ICCV*, pages 839–846. IEEE, 1998.

[236] N. Trawny and S. I. Roumeliotis. Indirect Kalman filter for 3D attitude estimation. Technical Report 2005-002, University of Minnesota, Dept. of Comp. Sci. & Eng., 2005.

[237] R. Triebel, W. Burgard, and F. Dellaert. Using hierarchical em to extract planes from 3D range scans. In *ICRA*, pages 4437–4442. IEEE, 2005.

[238] Y. Tsin and T. Kanade. A correlation-based approach to robust point set registration. In *Computer Vision-ECCV 2004*, pages 558–569. Springer, 2004.

[239] G. Turk and M. Levoy. Zippered polygon meshes from range images. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 311–318. ACM, 1994.

[240] G. Ulrich. Computer generation of distributions on the m-sphere. *Applied Statistics*, pages 158–163, 1984.

[241] S. Umeyama. Least-squares estimation of transformation parameters between two point patterns. *TPAMI*, (4):376–380, 1991.

[242] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, volume 1, pages I–I. IEEE, 2001.

[243] R. Webb. Stella software. http://www.software3d.com/Stella.php and https://en.wikipedia.org/wiki/600-cell.

[244] T. Weise, T. Wismer, B. Leibe, and L. Van Gool. In-hand scanning with online loop closure. In *ICCV Workshops*, pages 1630–1637. IEEE, 2009.

[245] G. Weiss, C. Wetzler, and E. Von Puttkamer. Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. In *IROS*, volume 1, pages 595–601. IEEE, 1994.

[246] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald. Kintinuous: Spatially extended KinectFusion. In *RSS Workshop on RGB-D: Advanced Reasoning with Depth Cameras*, Sydney, Australia, Jul 2012.

[247] T. Whelan, M. Kaess, H. Johannsson, M. Fallon, J. Leonard, and J. McDonald. Real-time large scale dense RGB-D SLAM with volumetric fusion. *IJRR*, 2014.

[248] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison. ElasticFusion: dense SLAM without a pose graph. In *RSS*. RSS, 2015.

[249] T. Whelan, L. Ma, E. Bondarev, P. de With, and J. McDonald. Incremental and batch planar simplification of dense point cloud maps. *Robotics and Autonomous Systems*, (0):–, 2014.

[250] T. Whelan, R. F. Salas-Moreno, B. Glocker, A. J. Davison, and S. Leutenegger. ElasticFusion: Real-time dense SLAM and light source estimation. *IJRR*, pages 1697–1716, 2016.

[251] Wikipedia. Database download, June 2014.

[252] H. Wildenauer and A. Hanbury. Robust camera self-calibration from monocular images of Manhattan worlds. In *CVPR*, pages 2831–2838. IEEE, 2012.

[253] C. K. I. Williams and C. E. Rasmussen. Gaussian processes for regression. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *NIPS*, pages 514–520. MIT Press, 1996.

[254] J. Xiao, A. Owens, and A. Torralba. Sun3D: A database of big spaces reconstructed using SFM and object labels. In *ICCV*, pages 1625–1632, 2013.

[255] Y. Xu, S. Oh, and A. Hoogs. A minimum error vanishing point detection approach for uncalibrated monocular images of man-made environments. In *CVPR*, pages 1376–1383, 2013.

[256] J. Yang, H. Li, and Y. Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In *ICCV*, pages 1457–1464. IEEE, 2013.

[257] C. Zhang, L. Wang, and R. Yang. Semantic segmentation of urban scenes using dense depth maps. In *ECCV*, pages 708–721. Springer, 2010.

[258] S. Zhong. Efficient online spherical k-means clustering. In *IJCNN*, volume 5, pages 3180–3185. IEEE, 2005.

[259] Q.-Y. Zhou, J. Park, and V. Koltun. Fast global registration. In *ECCV*, pages 766–782. Springer, 2016.

[260] Y. Zhou, L. Kneip, and H. Li. Real-time rotation estimation for dense depth sensors in piece-wise planar environments. In *IROS*, pages 2271–2278. IEEE, 2016.

[261] Meshlab. http://meshlab.sourceforge.net/. Accessed: 2016-11-15.

[262] Point cloud library. http://pointclouds.org/. Accessed: 2016-11-15.