LiDAR-Based Object Tracking and Shape Estimation

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

von der KIT-Fakultät für Maschinenbau
des Karlsruher Instituts für Technologie (KIT)

angenommene

DISSERTATION

von

M.Sc. Stefan Krämer
geb. in Bensheim

Hauptreferent:	Prof. Dr.-Ing. Christoph Stiller
Korreferent:	Prof. Dr.-Ing. Klaus Diepold
Tag der mündlichen Prüfung:	18. Dezember 2020

Kurzfassung

Umfeldwahrnehmung stellt eine Grundvoraussetzung für den sicheren und komfortablen Betrieb automatisierter Fahrzeuge dar. Insbesondere bewegte Verkehrsteilnehmer in der unmittelbaren Fahrzeugumgebung haben dabei große Auswirkungen auf die Wahl einer angemessenen Fahrstrategie. Dies macht ein System zur Objektwahrnehmung notwendig, welches eine robuste und präzise Zustandsschätzung der Fremdfahrzeugbewegung und -geometrie zur Verfügung stellt.

Im Kontext des automatisierten Fahrens hat sich das Box-Geometriemodell über die Zeit als Quasistandard durchgesetzt. Allerdings stellt die Box aufgrund der ständig steigenden Anforderungen an Wahrnehmungssysteme inzwischen häufig eine unerwünscht grobe Approximation der tatsächlichen Geometrie anderer Verkehrsteilnehmer dar. Dies motiviert einen Übergang zu genaueren Formrepräsentationen.

In der vorliegenden Arbeit wird daher ein probabilistisches Verfahren zur gleichzeitigen Schätzung von starrer Objektform und -bewegung mittels Messdaten eines LiDAR-Sensors vorgestellt. Der Vergleich dreier Freiform-Geometriemodelle mit verschiedenen Detaillierungsgraden (Polygonzug, Dreiecksnetz und Surfel Map) gegenüber dem einfachen Boxmodell zeigt, dass die Reduktion von Modellierungsfehlern in der Objektgeometrie eine robustere und präzisere Parameterschätzung von Objektzuständen ermöglicht. Darüber hinaus können automatisierte Fahrfunktionen, wie beispielsweise ein Park- oder Ausweichassistent, von einem genaueren Wissen über die Fremdobjektform profitieren.

Es existieren zwei Einflussgrößen, welche die Auswahl einer angemessenen Formrepräsentation maßgeblich beeinflussen sollten: Beobachtbarkeit (Welchen Detaillierungsgrad lässt die Sensorspezifikation theoretisch zu?) und Modell-Adäquatheit (Wie gut bildet das gegebene Modell die tatsächlichen Beobachtungen ab?). Auf Basis dieser Einflussgrößen wird in der vorliegenden

Arbeit eine Strategie zur Modellauswahl vorgestellt, die zur Laufzeit adaptiv das am besten geeignete Formmodell bestimmt.

Während die Mehrzahl der Algorithmen zur LiDAR-basierten Objektverfolgung ausschließlich auf Punktmessungen zurückgreift, werden in der vorliegenden Arbeit zwei weitere Arten von Messungen vorgeschlagen: Information über den vermessenen Freiraum wird verwendet, um über Bereiche zu schlussfolgern, welche nicht von Objektgeometrie belegt sein können. Des Weiteren werden LiDAR-Intensitäten einbezogen, um markante Merkmale wie Nummernschilder und Retroreflektoren zu detektieren und über die Zeit zu verfolgen.

Eine ausführliche Auswertung auf über 1,5 Stunden von aufgezeichneten Fremdfahrzeugtrajektorien im urbanen Bereich und auf der Autobahn zeigen, dass eine präzise Modellierung der Objektoberfläche die Bewegungsschätzung um bis zu 30%-40% verbessern kann. Darüber hinaus wird gezeigt, dass die vorgestellten Methoden konsistente und hochpräzise Rekonstruktionen von Objektgeometrien generieren können, welche die häufig signifikante Überapproximation durch das einfache Boxmodell vermeiden.

Abstract

Environment perception is a primary prerequisite for the safe and comfortable operation of automated vehicles. In particular, moving traffic participants in the immediate proximity of an automated vehicle have significant influence on the inference of an appropriate driving policy. This motivates the demand for an object perception system that provides robust and accurate estimates of target object motion and geometry.

In the context of automated driving, the bounding box has traditionally been established as de-facto standard object model. Given the ever-increasing demands on perception systems, the box model often undesirably overapproximates the true geometry of other traffic participants, thus urging the transition to more detailed shape representations.

This thesis proposes a probabilistic framework for the simultaneous estimation of rigid object shape and motion from LiDAR measurements. Comparing three free-form models with varying level of detail (polyline, triangle mesh and surfel map) against the baseline box model, it is shown that a reduction of geometric modelling errors allows for more robust and accurate object state estimates. Moreover, applications such as parking and evasive steering can profit from an improved knowledge about the object shape.

There are two principal factors governing the choice of an appropriate shape representation: Observability ("What level of model detail is the sensor theoretically able to populate?") and model adequacy ("How effective is the model in representing the data?"). Therefore, an adaptive model switching strategy is proposed that selects an adequate representation at runtime, taking into account these factors.

While the majority of LiDAR-based tracking algorithms resorts to scan point observations only, two additional sources of measurement from LiDAR scans are introduced: Leveraging free-space information from LiDAR to reason about regions that must be free of object geometry as well as LiDAR intensities

that allow to identify salient object features such as license plates and retro-reflectors.

Extensive evaluation on more than 1.5h of object trajectories recorded in public urban and highway traffic shows that the precise modeling of object surface allows to improve the motion estimation by a margin of up to 30%-40%. Moreover, the method is able to produce consistent and high precision reconstructions of object shapes that avoid the often significant overapproximation of geometry by the simple box model.

Acknowledgements

This thesis results from a cooperation between the Institute of Measurement and Control Systems (MRT) at Karlsruhe Institute of Technology (KIT) and the Pre-Development for Automated Driving at AUDI AG.

First of all, I'd like to express my gratitude to Prof. Dr.-Ing. Christoph Stiller for rendering the research project possible in the first place, but also for his excellent supervision during the process. Moreover, I would like to thank Prof. Dr.-Ing. Klaus Diepold for his interest in my work and role as co-examiner. Dr.-Ing. Sayed Bouzouraa greatly contributed to the success of this thesis by serving as my industrial supervisor. Thanks, Sayed! Both the Audi AI Lab and MRT were pleasant environments for my research and I'd like to thank all my colleagues, especially Marvin Raaijmakers, Frank Dierkes, Moritz Sackmann and Matthias Steiner.

I wish to express my gratitude to all my family and friends who supported me along the way and provided me with the necessary distractions during my spare time. A special thanks to Amy, whose high spirits helped me through the tougher of days. Finally, I am endlessly grateful to Lari for being there in good times as in bad.

Ingolstadt, Juli 2020 Stefan Krämer

Table Of Contents

Notation and Symbols

General notation

a	scalar
$\mathbf{a}$	vector
$\mathbf{a}_{i:j}$	sequence of vectors $\mathbf{a}_i, \mathbf{a}_{i+1}, \ldots, \mathbf{a}_j, \quad i \leq j$
A	matrix

Distributions and Norms

$p(x)$	probability $p(X = x)$ of random variable X taking value x
$\mathcal{N}(\mu, \sigma^2)$	normal distribution with mean μ and variance σ^2
$\mathcal{U}(a, b)$	uniform distribution in $[a, b]$
$\|\mathbf{x} - \mathbf{y}\|^2_{\Sigma^{-1}}$	squared Mahalanobis distance between $\mathbf{x}$ and $\mathbf{y}$ under covariance Σ
$\det(\mathrm{A})$	determinant of matrix A
$\|A\|$	cardinality of set A

State Entities

$\mathbf{x}_t$	2D object pose at time instant t
(x_t, y_t)	2D object position at time instant t
θ_t	object orientation around z_{env}-axis at time instant t
$\mathbf{z}_t$	vector of observations obtained at time instant t
$\mathbf{z}_t^{(i)}$	i-th observation obtained at time instant t
$\mathbf{s}$	(rigid) surface

Acronyms

LiDAR	Light Detection and Ranging
SLAM	Simultaneous Localization and Mapping
ICP	Iterative Closest Point
NLS	Non-Linear Least Squares
AIC	Akaike Information Criterion
GPS	Global Positioning System
RMSE	Root-Mean-Square Error

1 Introduction

Automated driving has emerged as one of the megatrends in the automotive industry of the early 21st century. Besides the prospect of a significant increase in the safety and efficiency of transportation, the technology entails considerable social and economic implications by facilitating access to personal mobility and enabling a more productive use of travel time [12]. Nevertheless, apart from pending ethical controversies, academia and industry still face a wealth of open technical challenges on the way to full vehicle automation.

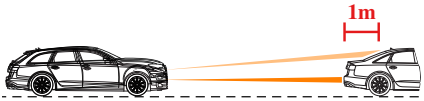
The software architecture [74] of an automated driving system is typically decomposed into a *perception and scene understanding* component, that derives an internal model of the environment using sensor data, from which a *behavior and motion planning* component infers vehicle actions taking into account the automated vehicle's objectives. An important requirement for comprehensive scene understanding and, eventually, for the derivation of safe and comfortable driving policies is the robust and accurate perception of other traffic participants in the presence of sensor noise and occlusion.

In object tracking, the extent of dynamic objects traditionally remained unmodeled (point targets) or was reduced to simple geometries, such as ellipses [11] or rectangles [41, 35]. More recently, the availability of precise measurements of vehicle contours, e.g., from Light Detection And Ranging (LiDAR), has sparked an interest in representations that capture the object surface in more detail. This thesis is concerned with the simultaneous recovery of precise object geometries alongside the estimation of their motion parameters from LiDAR sensor data.

There are a variety of motives driving the desire for detailed modelling of object surfaces: While the planar outline of an average passenger car is approximated by the box model reasonably well, the robust extraction of boxes from partial and noisy sensor observations proves challenging in practice. This *fitting dilemma* is illustrated in Fig. 1.1a, where the LiDAR scan points of an

(a) Individual observations of an oncoming passenger car annotated with the corresponding box fits.

(b) Truck wheel cases as salient features.



(c) Apparent motion due to ego pitch motion.

Figure 1.1: Illustration of various shortcomings of the box shape model.

oncoming passenger car are annotated with individual box fits from a standard box fitting algorithm [87]. Several of the extracted boxes feature significant deviations from the true vehicle orientation, which notably also imply considerable positional errors of the box corners. Interestingly, without additional context knowledge, a human observer would arguably consider the individual boxes a reasonable fit to the data as well.

Moreover, the precise association of observations to the tracked object surface is a premise for accurate motion estimation. In many situations, the oversimplification of object geometry by means of the box model result in a *loss of salient features*. An example are the wheel cases of trucks as indicated in Fig. 1.1b: These salient vehicle parts are abstracted away in the simple box representation, even though they provide valuable evidence on the longitudinal object motion, especially so in the presence of occlusion.

When dealing with more complex object surfaces, the oversimplification of geometry poses an additional problem: It introduces considerable tracking residuals that are interpreted as object motion, but in fact stem from violations of the shape model assumptions. This results in *apparent motion*. Fig. 1.1c displays how the pitch motion of an automated vehicle can cause LiDAR beams to intersect with varying horizontal cross-sections of a sedan. In this example, the residuals between the LiDAR returns from the rear bumper and C-pillar of the vehicle body amount to over one meter in range.

Besides the above perception-driven motives, the evolution to detailed surface models is also propelled by demands arising from the behavior generation for next-generation driving functions. In particular, when operating in confined spaces, such as parking lots or narrowings of the road, the oversimplification of object geometry often inappropriately restricts the available freespace, thus potentially hindering the inference of suitable driving policies. Furthermore, detailed contour information can assist the reasoning in critical situations, e.g., in the avoidance of collisions or the mitigation of the resulting damage if an impact is inevitable.

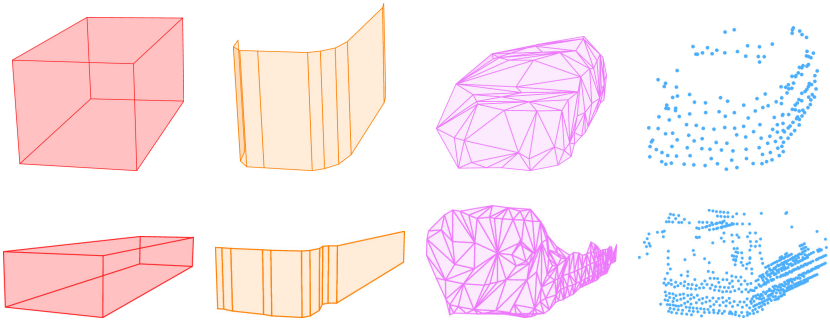
1.1 Contributions

This thesis presents a probabilistic framework for the estimation of rigid object shape and planar motion from automotive LiDAR sensors. The addressed problem essentially represents a chicken-and-egg problem: While the recovery of detailed object geometry necessitates precise motion estimates, an effective tracking process in turn relies on accurate knowledge about the object shape. This mutual dependence together with the demand for high quality state estimates motivates their simultaneous estimation using sliding window bundle adjustment.

Besides the use of conventional scan point observations, two novel measurement models are introduced: The first model leverages *freespace information* at object boundaries, which allows to considerably restrict the space of probable object state configurations. In addition, LiDAR *intensities* are incorporated into the tracking process, representing salient features on the otherwise often relatively homogeneous vehicle surfaces.

While the majority of related publications study the reconstruction of a single surface model at hand, the scalability of the proposed approach is demonstrated by the implementation of the *four surface models* depicted in Fig. 1.2, which capture varying levels of detail: the box, polyline, triangle mesh and surfel map[1] models. Moreover, the estimation framework is augmented by a novel

[1] A *surfel map* is a surface model that represents the object geometry by means of small, oriented disks, which are visualized as simple point clouds here.



(a) Box (b) Polyline (c) Triangle Mesh (d) Surfel Map

Figure 1.2: Sample reconstructions of an oncoming car (top row) and bus (bottom row) using the VLP-16HR data set. The box and polyline model height is populated with a default value of 1.5m.

model selection strategy, which allows for the choice of the most appropriate surface representation in situ.

Experiments based on more than 1.5 h of real-world object trajectories demonstrate the framework's capability to achieve precise trajectory estimates and its potential to generate accurate object surface reconstructions.

1.2 Outline

The remainder of this thesis is structured as follows: Chapter 2 provides a brief review of the literature on automotive object tracking and shape estimation. It motivates the particular research focus of this thesis and positions the contributions with respect to the state-of-the-art.

Chapter 3 describes a probabilistic framework for simultaneous object tracking and shape estimation. This includes a mathematical definition of the four surface models as well as methods for their reconstruction and refinement using three complementary measurement models. Together with a motion model for the temporal evolution of object states, details on the practical implementation of the framework are provided. In particular, the estimation

problem is approached using sliding window bundle adjustment rather than classical recursive state estimation.

The availability of multiple object surface models prompts the question for the most adequate representation from the candidates. Therefore, chapter 4 explores the possibility of an adaptive model selection based on various criteria and provides a prototypical implementation of the decision process.

The experimental results presented in chapter 5 prove the effectiveness of the proposed methods using real world data from automotive series and research laser scanners. Finally, chapter 6 offers concluding remarks and an outline of potential future research directions.

2 Background And Related Work

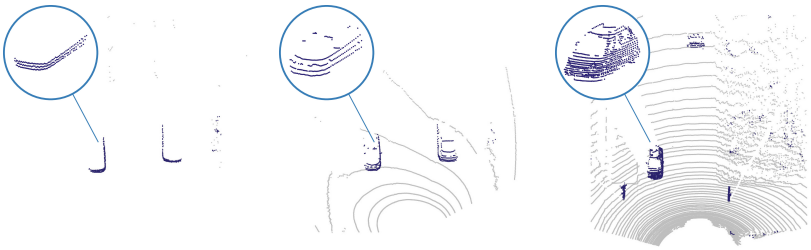
Object tracking is concerned with the estimation of object states (e.g., position, orientation and velocity) from a time sequence of observations and has a long-standing history in scientific and industrial applications. Similarly, the reconstruction of object surfaces from sensor observations is an extensive field, which caters a wide spectrum of applications such as medical imaging, augmented reality or terrain modelling in geographic information systems.

Rather than attempting to provide an exhaustive review of the literature on tracking and reconstruction with their numerous facets, this chapter focuses on the state-of-the-art where both disciplines intersect. A special emphasis is placed on LiDAR-based approaches for automotive applications, which are most relevant for the methods discussed in the subsequent chapters of this thesis.

The remainder of this chapter is structured as follows: Section 2.1 provides a brief overview of the measurement principle of LiDAR sensors, followed by a review of LiDAR-based tracking and shape estimation methods for dynamic objects in section 2.2. A summary of existing methods for the online selection of the most suitable surface model from a set of candidates is provided in section 2.3.

2.1 LiDAR Fundamentals

LiDAR is an optical measurement technology based on the principle of *time-of-flight* [80]: The distance to an obstacle is proportional to the time interval it takes for a laser pulse to travel from the sensor to the obstacle and back. The majority of automotive-grade LiDAR sensors on the market today sequentially scan the environment by a set of rotating laser diodes or a spinning deflection mirror. However, alternative technologies are rapidly



(a) Series sensor (b) Velodyne VLP-16HR (c) Velodyne HDL-64E

Figure 2.1: Exemplary LiDAR scan frames from a series laser scanner (three layers) and two research sensors (16 and 64 layers respectively). The distance to the enlarged car is approximately 10 m. The scan in Fig. 2.1c is extracted from the KITTI data set [28].

approaching deployment in series vehicles. Examples are scanners based on micro-electro-mechanical systems (MEMS) that substitute the mechanically spinning mirrors by micro-electro-mechanical parts, as well as flash LiDARs, which simultaneously illuminate a large area of the environment.

One major advantage of LiDARs over other sensing modalities is their capability for the explicit measurement of freespace, which renders them a key enabler for safety-critical applications such as automated driving. Moreover, LiDARs are able to directly measure depth at high angular resolutions, thus facilitating precise 3D models of the environment. Also, in contrast to triangulating sensors, such as stereo cameras, the range accuracy of LiDARs is independent of the obstacle distance.

On the downside, optical sensors tend to be susceptible to environmental influences such as dirt, fog or rain, which considerably deteriorate the sensor performance. Moreover, LiDARs still lack the capability for direct measurement of velocity that is available with radar sensors, even though the first prototypes of frequency-modulated continuous-wave LiDARs have been presented recently. Finally, the discriminative capabilities of LiDAR intensities as well as the angular resolution of automotive LiDAR sensors is not yet on par with that of state-of-the-art cameras, despite research sensors with considerable sampling densities being available on the market already.

Notably, a review of the state-of-the-art of LiDAR-based environment perception methods must always bear in mind the large spectrum of LiDAR sensors

employed in the literature. The considerable difference between automotive-grade series and research sensors is demonstrated in Fig. 2.1, which displays exemplary LiDAR scans captured by a series vehicle sensor (three layers) as well as two different research sensors (16 and 64 layers respectively). The apparent discrepancy in the quality of observations often renders the transferability of algorithms from expensive research to low-cost automotive sensors highly problematic.

2.2 LiDAR-Based Tracking and Shape Estimation

This section summarizes existing approaches for modelling and reconstructing the surface of dynamic objects (section 2.2.1), for the use of freespace information in tracking (section 2.2.2) as well as for utilizing LiDAR intensities in environment perception for automated driving (section 2.2.3).

2.2.1 Surface Representations For Dynamic Objects

Surface models provide a means for the mathematical description of object geometry. In particular, surface representations can be broadly divided into *pre-defined* and *freeform* models. While pre-defined models allow for the representation of relatively simple object geometries using a fixed number of parameters, freeform models provide more flexibility by building up surfaces from a collection of geometric primitives such as points, edges and triangles.

In automated driving, the parametric *box model* has traditionally emerged as de-facto standard surface representation for dynamic objects. There are two common abstraction levels for box model tracking: Numerous authors [1, 21, 35, 41, 67] propose a data reduction through the extraction of box fits from the raw scan data in a preprocessing step. The compact box measurements are subsequently used for data association and state innovation. This approach requires a careful and adaptive choice of the tracked reference point [41, 67] in order to account for varying visibility, e.g., caused by partial occlusion, which otherwise might result in apparent object motion. An alternative approach is the direct association of individual range readings to the outline of tracked box models, e.g., by ray casting [57] or nearest-neighbor association [19].

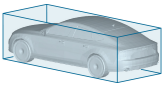
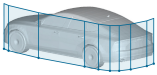
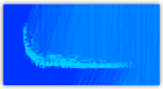
Box	Polyline	Gridmap	Point-based
			
[19, 21, 35, 41, 57, 67]	[77, 83]	[6, 7, 8, 18, 59, 69, 71]	[34, 52, 81, 85]

Table 2.1: An overview of the most common surface representations for dynamic objects in the context of automated driving.

This results in higher computational costs, but forgoes the error-prone process of finding accurate box fits in partial object views (cf. Fig. 1.1a). Various slightly augmented versions of the simple box model have been proposed in the literature, such as the twelve-parameter vehicle model described by Koller [44].

The availability of high-resolution measurements from state-of-the-art automotive sensors, such as LiDARs, has motivated a shift from the simple box to a multitude of freeform model approaches, which will be detailed in the remainder. Table 2.1 provides an overview of the most common surface representations discussed in the tracking literature.

The *polyline shape model* emerges as natural generalization of the box model. Vatavu et al. [77] propose to estimate object shape as polylines based on stereo vision and Rao-Blackwellized particle filters. They sample over the object's dynamic state, while estimating polyline control point positions in each particle separately using Extended Kalman Filters. The control point density is chosen equally-spaced along the object outline, thus disregarding the potential of adapting the model resolution to the local complexity of the object geometry. In contrast, Wyffles and Campbell [83] allow for non-uniform control point densities by representing object shape as the polyline connecting salient contour points from LiDAR scans. However, in practical applications the assumption of a robust detection of these salient contour points over time proves particularly problematic as consecutive scan patterns of the same object often differ considerably. This is due to sensor noise as well as natural ego pitch motion causing the sensor's scan lines to capture varying horizontal cross-sections of the target vehicle body (cf. Fig. 1.1c).

Baum and Hanebeck [10] propose a *polar representation* of object shape by means of the object centroid as vantage point and a function defining the radius $r = f(\theta)$ w.r.t. the azimuth angle θ. While Baum and Hanebeck parametrize the radial function by means of Fourier expansion, Wahlström and Özkan [79] propose an alternative formulation based on Gaussian processes, which offers the added possibility for incorporating symmetry assumptions. In both approaches, the parametrization using polar coordinates restricts the models to star-convex[1] object shapes. Moreover, determining the true vantage point from partial object scans with occlusion is often infeasible, which renders an estimation of the polar coordinate system origin necessary.

Originally, Badino et al. [9] introduced vertical rectangular sticks, called *stixels*, as compact representation for the static environment. Pfeiffer and Franke [58] extend the notion of stixel delimiters to dynamic objects. Noteworthy, the stixels on a single object's contour are tracked independently, thus disregarding the constraint that (rigid) object shape collectively undergoes rigid-body motion. Moreover, the stixels on object contours typically result in a non-continuous representation of the continuous object outline.

Various authors [6, 7, 8, 18, 59, 69] implement object shape estimation by embedding classical 2d *occupancy grid maps* [53] in a moving, object-local coordinate frame. This renders the shape accumulation particularly straightforward and naturally recovers the explicit distinction between free, occupied and unknown space that is common to grid maps. On the downside, the maps suffer from discretization effects and scale with the object extent rather than the surface complexity. Typically, this necessitates a post-processing step to extract a more compact representation at the interface to behavior generation. Moreover, a priori knowledge about the approximate object dimensions is required in order to prevent repeated reallocation of maps.

While some of the approaches described so far allow for the estimation of an additional height component (2.5d representations), their focus is on the precise modeling of planar object geometry. In contrast, Steinemann et al. [71] recover the "full" vehicle surface by populating 3d *voxel maps* of dynamic objects from LiDAR observations.

[1] An object shape is considered to be *star-convex*, if and only if, there exists a *vantage point* for which all line segments connecting this point with any other shape point are contained in the shape.

Various authors [34, 52, 85] propose the modeling of object shape by accumulating 3d *point clouds* over time: Moosmann and Stiller [52] jointly recover ego motion as well as 3d object motion and shape using the Iterative Closest Point (ICP) algorithm in conjunction with Kalman filtering. Similarly, Zeng [85] aligns shape state and observations using expectation maximization and estimates the object motion using Kalman filters. Finally, Held et al. [34] augment the raw 3D scan data with color information from a camera and use a grid-based estimation approach called annealed dynamic histograms to sample the state space. All three point cloud approaches employ high-resolution LiDAR scans from Velodyne HDL-64E (cf. Fig. 2.1c). Section 3.3.3 shows that the recursive alignment of raw point clouds of vehicle rears is highly problematic for LiDAR scans with low vertical opening angle, as the rounded object shape renders the objective function ill-posed. This hinders the transferability of the above approaches to the input from low-cost automotive series sensors.

Kumru and Özkan [46] extend the 2d approach proposed by Wahlström and Özkan [79] from (2d) polar to (3d) *spherical coordinates*. Similarly, Ebert and Wünsche [26] model star-convex object surfaces using spherical coordinates. They sample the azimuth and elevation angle of the sphere uniformly. Preliminary results are shown for input data from simulation only and disregard real-world effects such as self-occlusion. Moreover, in the presence of object rotation their tracker suffers from considerable drift in the orientation estimate.

More recently, Naujoks et al. [54] propose an extended target tracking approach based on *non-uniform rational B-splines* (NURBS). One variant of their approach restricts itself to estimating the scaling parameters of a NURBS surface, whereas the other approach incorporates the NURBS weights as well. On the downside, the method only tolerates small aberrations from cuboidal object structure.

Various authors have proposed methods to generate *triangle mesh* reconstructions of *parked* vehicles, including the works of Romanoni [61] as well as Kühner and Kümmerle [45]. Their approaches first accumulate a tetrahedral or cubic voxel representations of the environment and subsequently extract the meshes using Delaunay triangulation or marching cubes in a post-processing respectively. However, to the best of the author's knowledge there exists no prior publication, which is concerned with the simultaneous tracking and incremental triangle mesh reconstruction of dynamic vehicles.

2.2.2 Freespace Information

Evidence from range-based observations is commonly divided into *positive* and *negative* information. Positive information refers to actual observations, e.g., valid laser echos, and is leveraged in virtually all approaches to object tracking. In contrast, negative information indicates the absence of measurements and often remains disregarded in observation modeling. It is further classified into *missing* observations, i.e., the lack of any return echo, and *freespace* information, which denotes the unoccupied space between the emitting sensor and reflecting obstacle.

Petrovskaya and Thrun [57] harness missing information in order to detect dark surfaces from LiDAR scans, whereas Agate et al. [3] employ missing radar reflections for tracking objects whose radial velocity is lower than the minimal detectable velocity. Similarly, Miller et al. [51] as well as Wyffels and Campbell [82] preserve temporarily occluded tracks by reasoning about missing information.

Sampling-based methods [57, 75] use ray casting to model the space along every individual sensor beam and thus incorporate explicit freespace information into the tracking process. Moreover, correlation-based sensor models [75] leverage freespace evidence for the alignment of occupancy gridmaps. Both methods tend to be computationally expensive. In contrast, point-based approaches such as ICP only consider the beam endpoints, which renders them oblivious to negative sensor evidence. The novel freespace measurement model proposed in section 3.3.2 partially overcomes this drawback by allowing to augment scan point-based tracking and reconstruction processes by freespace information at object boundaries.

2.2.3 LiDAR Intensities

Most state-of-the-art LiDAR sensors report back a measure for the amount of radiant energy registered at the receiver diode, which is commonly referred to as LiDAR *intensity* or *reflectance*. Similar to intensities from camera pixel, their LiDAR counterparts offer great potential for informing the tracking process. Surprisingly, LiDAR intensities remain a mostly untapped measurement source in the tracking literature.

Various authors [33, 39, 68] propose to integrate color or LiDAR intensities into the popular ICP algorithm for motion estimation by augmenting the correspondence search space with an additional intensity component. In a similar fashion, Sasidharan and Lohani [64] describe a two-staged point cloud registration method, which first employs intensity-augmented ICP for finding a coarse alignment and then performs a refinement step using conventional ICP. One major challenge of search space augmentation are the disparate magnitudes of position and intensity values, which require a scaling of the quantities. Balancing between both position and intensity distance metrics is particularly challenging in object tracking, where the degree of motion is unknown a priori and varies over time.

In the context of object tracking, Kämpchen [41] filters LiDAR scans for points from reflective surface parts using a distance-dependent intensity threshold. Exploiting that reflective vehicle parts, such as license plates and retroreflectors, are usually arranged on the vehicle front or rear, the subset of filtered points is used to improve the orientation of box fits. Notably, the intensity information is not incorporated into the tracking process any further. Moreover, Kusenbach et al. [47] introduce a twelve-parameter geometric feature, which comprises the local surface structure as well as the horizontal and vertical intensity gradients. Rather than improving the online tracking process by incorporating the extracted features, the goal of their method is to condense the considerable amount of features in an offline postprocessing step. This generates an compact and generic object model, which is subsequently employed for object classification.

For the cross-calibration of cameras and LiDARs, Zhang [86] and Unnikrishnan [76] propose to exploit the fact that checkerboard patterns are typically visible in LiDAR intensities and determine the extrinsic calibration between the sensing modalities by minimization of the reprojection error of the scan points in the camera image. Similarly, Pandey et al. [56] propose a targetless calibration method that maximizes the correlation between LiDAR intensities and camera gray-scale values in the environment. Moreover, various authors employ LiDAR intensities for the detection and classification of road infrastructure, such as road markings and curbs [40, 49, 78] as well as road signs [27, 29, 60].

Beyond the ICP augmentation as well as the works of Kämpchen and Kusenbach et al., LiDAR intensities are mostly disregarded in the context of object

tracking. In particular, to the best of the author's knowledge, the novel measurement model introduced in section 3.3.3 is the first approach that explicitly extracts highly-reflective parts (e.g., license plates or retroreflectors) from LiDAR observations and tracks them over time.

2.3 Surface Model Selection

The vast majority of literature on extended object tracking focuses on representing the surface by means of a single, specific surface model at hand. This "one-model-fits-all" approach ineffectively reflects the diversity of street vehicle appearances encountered in public traffic. Arguably, generic surface models typically offer the possibility to represent simpler geometries by an adequate choice of parameters. However, employing the simpler models in the first place is preferable in such cases as added parameters and loss of (valid) inherent model assumptions typically degrade estimation efficiency and robustness. To the best of the author's knowledge, there are only two approaches discussing the possibility of an adaptive model selection for street vehicles, both of which will be discussed in the remainder.

Darms et al. [23, 22] propose an adaptive model switching approach between an extent-free point model and a 2d box representation based on *available sensor information*. Their model selection strategy employs a voting scheme that considers the number of sensor readings supporting any of the two given models. In their sensor setup, both point and box model are supported by laser scanners, whereas radar and fixed-beam laser sensors are restricted to the point model. If supported by observations, their voting scheme will always prefer the box model due to its higher level of detail.

Wyffels and Campbell [84] present a more sophisticated strategy for model selection. Based on probabilistic *object relevance* metrics, their framework trades off between required tracking precision and computational resources. They demonstrate a prototypical implementation of their approach for an anticipatory planner that switches between an ellipse shape model and the most recent LiDAR point cloud based on the object distance and collision probability.

While both approaches propose effective criteria for the choice of an appropriate surface model, they are oblivious to model adequacy in a geometric sense, i.e., the efficiency of a surface model in approximating the given observations. This motivates the model selection strategy described in chapter 4, which additionally allows to assess the approximative power of a given model based on an information-theoretic foundation.

3 Simultaneous Tracking and Shape Estimation

Extended objects are typically defined as objects that cause multiple spatially distributed sensor observations per time instant [31]. With LiDAR, these observations are precise and dense samples of the visible object boundary, i.e., of its *surface*. Modelling the extent of other traffic participants alongside their motion parameters is particularly desirable in automated driving, where precise knowledge of object shape can facilitate spatial reasoning and improve interpretability of new measurements. The goal of *shape estimation* is to reconstruct an approximation of the true object surface from a sequence of observations, whereas *tracking* is concerned with the estimation of object motion over time.

Motion and shape estimation are closely interlinked: While motion estimation requires good knowledge of the object shape for the proper association of new observations, shape estimation in turn necessitates precise information about object poses. This mutual dependence motivates a framework for the *simultaneous* estimation of motion and shape. The problem is concerned with mapping under relative motion and bears large similarity with the well-studied problem of simultaneous localization and mapping (SLAM), where a robot builds up a map of its environment while concurrently localizing itself within that map [75]. However, in moving object tracking the maps are considerably smaller (vehicle surface vs. static world) and exhibit only very few salient features given that vehicle surfaces are often relatively homogeneous and smooth. Moreover, SLAM frameworks typically rely on a sequence of odometry measurements of the relative motion (e.g., wheel ticks), whereas the tracking problem is restricted to ascribing a presumed model (e.g., constant

velocity) to the target object motion[1] in order to inform the observation-based estimation.

In the context of automated driving, the most recent object pose and its derivatives (typically velocity and acceleration) as well as an up-to-date shape estimate are of great interest. This motivates an *online* estimation together with *incremental* surface reconstruction methods rather than batch approaches, which first collect the entire sequence of measurements and subsequently reconstruct the geometry in an offline post-processing step. Moreover, the presence of measurement noise urges for a *probabilistic* approach to the problem. Under the assumption of negligible interdependency between the motion of separate objects in the environment, the tracking process is decomposable into multiple independent single object state estimators. Following this notion, the remainder of this chapter is concerned with the estimation of the single object pose and shape posterior probability

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) \tag{3.1}$$

at discrete time instants t,

where $\mathbf{x}_{0:T}$: sequence of time-dependent object poses $\mathbf{x}_t$ up to time T
$\mathbf{s}$: time-invariant (i.e., rigid) object surface
$\mathbf{z}_{1:T}$: sequence of sensor observations $\mathbf{z}_t$ up to time T.

The tracking problem is a *time-discrete dynamic random system* that is approached using a Bayes estimator and the factorization of the object pose and surface posterior depicted by the graphical model in Fig. 3.1. In particular, the posterior probability in Eqn. (3.1) can be further decomposed by applying Bayes rule (see Eqn. (A.1)), resulting in

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \frac{1}{p(\mathbf{z}_{1:T})} \cdot p(\mathbf{z}_{1:T} \mid \mathbf{x}_{0:T}, \mathbf{s}) \cdot p(\mathbf{x}_{0:T}, \mathbf{s}).$$

[1] In the remainder, ego motion is assumed to be available from an external module, which allows to decouple target from ego motion.

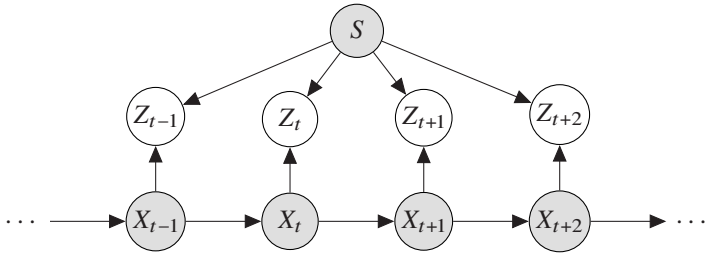


Figure 3.1: Graphical model representation of the tracking problem.

For improved readability, the constant normalization factor $1/p(\mathbf{z}_{1:T})$ is abbreviated as η in the remainder. Under the assumption of independence (see Eqn. (A.2)) between object poses and the surface, the problem simplifies to

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \eta \cdot p(\mathbf{z}_{1:T} \mid \mathbf{x}_{0:T}, \mathbf{s}) \cdot p(\mathbf{x}_{0:T}) \cdot p(\mathbf{s}).$$

Moreover, assuming conditional independence of individual sensor measurements over time allows for factorization of the conditional probability as

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \eta \cdot \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_{0:T}, \mathbf{s}) \cdot p(\mathbf{x}_{0:T}) \cdot p(\mathbf{s}).$$

For the measurement formation process, in particular with LiDAR sensors, it is reasonable to assume that an observation $\mathbf{z}_t$ at time instant t is exclusively governed by the *instantaneous* configuration of the object pose and shape, i.e.,

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \eta \cdot \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) \cdot p(\mathbf{x}_{0:T}) \cdot p(\mathbf{s}).$$

Applying the chain rule of probability (see Eqn. (A.4)) to the joint probability of object poses results in the final decomposition of the posterior probability as

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \eta \cdot \prod_{t=1}^{T} \underbrace{p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s})}_{\text{measurement model}} \cdot \prod_{t=0}^{T} \underbrace{p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})}_{\text{motion model}} \cdot \underbrace{p(\mathbf{s})}_{\text{surface prior}} . \tag{3.2}$$

The individual components of Eqn. (3.2) are discussed in the remainder of this chapter as follows:

- The *state space* consists of the time-dependent object poses $\mathbf{x}_{0:T}$ and the rigid surface $\mathbf{s}$. Section 3.1 provides an overview of the coordinate systems involved in the tracking process. In particular, the object poses $\mathbf{x}_t$ specify an *object-local coordinate system* in which the object shape is aggregated. A range of different approximate representations for the object surface $\mathbf{s}$ as well as potential surface prior probabilities $p(\mathbf{s})$ are given in section 3.2.
- The *measurement model* $p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s})$ describes the formation of sensor observations given a particular configuration of the object pose and shape. Section 3.3 discusses several measurement models suited for LiDAR sensors.
- The *motion model* $p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})$ imposes assumptions and constraints on the evolution of object poses over time. Section 3.4 describes a well-established motion model for street vehicle kinematics and its implementation in the context of this thesis.
- Finally, section 3.5 places the tracking and shape estimation problem into the overall data processing pipeline and provides insights into details of the *practical implementation*. In particular, it describes a sliding window maximum likelihood estimation approach to the posterior in Eqn. (3.2) based on non-linear least squares optimization, which renders the estimation tractable in online applications.

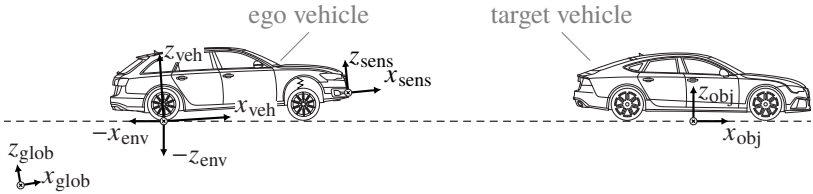


Figure 3.2: Side view illustration of the five coordinate systems: sensor (abbr. *sens*), vehicle (abbr. *veh*), environment (abbr. *env*), object-local (abbr. *obj*) and global (abbr. *glob*). Note that environment coordinate system axes x_{env} and z_{env} are flipped for improved visualization.

3.1 Coordinate Systems

Most perception systems decompose the space into distinct logical units using multiple coordinate systems. These separate geometric environments are typically defined by their relative poses and, if applicable, their relative motion. The five coordinate systems relevant in this thesis are illustrated in Fig. 3.2 and described in more detail in this section. If not stated differently, all coordinates systems are assumed to be right-handed and Cartesian.

In a multi-sensor setup, each individual sensor captures measurements independently and determines their location in a local *sensor coordinate system* (abbr. *sens*). Depending on the specific sensing modality, the sensor observations are often specified in spherical coordinates (e.g., LiDAR) or pixel coordinates (e.g., cameras). For optical sensors, the coordinate system origin is most commonly placed at their optical center.

An extrinsic sensor calibration specifies the relative poses of the individual sensor coordinate systems with respect to a common *vehicle coordinate system* (abbr. *veh*). This chassis-fixed systems undergoes all ego motion and allows to relate concurrent observations from different sensors. Its axes are defined in ISO 8855 [38] as follows: Pointing forward, the x_{veh}-axis is horizontal and parallel to the chassis' longitudinal plane of symmetry. The y_{veh}-axis is perpendicular to the chassis' longitudinal plane of symmetry and points to the left. The z_{veh}-axis points upwards (right-hand convention). The coordinate

system origin is typically located at the vertical projection of the rear axle center onto the ground plane (with the vehicle at rest).

Under the flat world assumption, the road surface in a local environment around the ego vehicle is approximated by a ground plane. This plane is determined using the four contact points of the tyres with the road surface. The *environment coordinate system* (abbr. *env*) is then defined with its origin at the instantaneous vertical projection of the rear axle center onto the ground plane. The z_{env}-axis is chosen perpendicular to the ground plane, whereas the x_{env}- and y_{env}-axes are parallel to the ground plane, with the x_{env}-axis being aligned with the vertical projection of the corresponding vehicle coordinate system axis x_{veh} [38]. This intermediate environment coordinate system factors out ego vehicle pitch and roll motion as well as vertical suspension. It is typically employed as embedding for 2D environment representations, but also provides a convenient space for fusing measurements of other traffic participants over time: Under the assumption that these operate on the same planar ground surface as the ego vehicle, it is sufficient to estimate their 2D pose in environment coordinates.

Given that tracked objects move relatively to the ego vehicle, it is practical to introduce an *object-local coordinate system* (abbr. *obj*), which is subject to all target motion and thus used as an embedding for the object shape $\mathbf{s}$. In particular, the coordinate system is assumed to move on the xy-plane of the environment coordinate system. It is defined by the 2D object pose $\mathbf{x}_t = (x_t, y_t, \theta_t)$ at time instant t, i.e., is located at position (x_t, y_t) with rotation θ_t around the z_{env}-axis as depicted in Fig. 3.3. Note that, in principle, the initial placement of the coordinate system with respect to the actual object geometry is arbitrary. However, placing it at the (estimated) target object center proves beneficial for motion modeling (see section 3.4) and thus the offset between the coordinate system origin and the estimated object centroid should regularly be eliminated by shifting all object poses and the geometry accordingly. Moreover, it is worth noting that - with exception of the box surface model - the orientation θ_t of the object-local coordinate system is not necessarily incident with the true object yaw angle ψ_t. In particular, freeform shape models lack the concept of pre-defined principal axes and thus there can be an offset between coordinate system orientation and the longitudinal heading (i.e., yaw angle) of the object.

The earth-fixed *global coordinate system* (abbr. *glob*) allows to describe the trajectory of the vehicle coordinate system, i.e., the ego motion, with respect

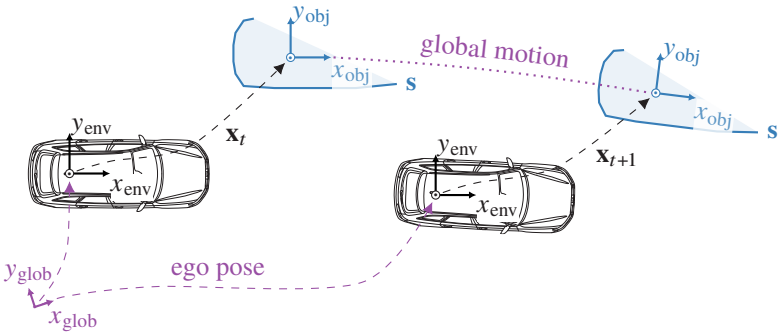


Figure 3.3: Illustration of the object-local coordinate system.

to the static world. Moreover, transforming target object poses into the global coordinate system enables a decoupling of ego and target object motion, which is a prerequisite for leveraging meaningful motion models in the tracking process. Describing target object motion in global coordinates also tends to be a more convenient representation for the subsequent behavior generation. In particular, the magnitude of target velocities and accelerations is significantly more tangible when provided with respect to the static road surface rather than relative to the environment coordinate system, which is potentially in motion itself. Finally, the global coordinate system allows to place the ego vehicle and other participating actors in relation to external data sources such as digital maps or information from vehicle-to-vehicle and vehicle-to-infrastructure communication.

If not stated differently, the object surface $\mathbf{s}$ is defined with respect to the object-local coordinate system in the remainder, whereas all other parameters are specified in environment coordinates.

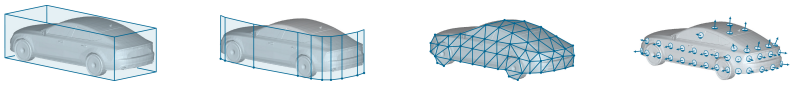
3.2 Surface Models and Reconstruction Methods

Surface reconstruction is concerned with recovering the boundary of a solid object from a sequence of noisy measurements. Botsch et al. [16] define a *surface* $\mathbf{s}$ as "an orientable, continuous 2D manifold embedded in $\mathbb{R}^3$". Orientability refers to a consistent definition of surface normal vectors, whereas manifoldness implies that locally the surface topology is equivalent to a disk. This section describes surface representations that allow to approximate the true object boundary from a sequence of sensor observations and provides methods for their reconstruction and refinement over time.

In the context of automated driving, virtually all sensing modalities are susceptible to object (self-)occlusion, which results in partial observations of the object surface. Notably, in most real-world driving scenarios, the complete boundary of other traffic participants is rarely ever entirely observed even over a longer period of time. Without additional model assumptions, such as symmetry, the reconstructions methods thus must be able to handle *open* surfaces. These are surfaces that can be converted into proper manifolds by hole filling and that locally are topologically equivalent to a half-disk [16]. Note that in the application at hand, reconstructions with only a single surface boundary are desirable and thus hole closing is applied whenever possible, e.g., at vehicle windows.

Approximate representations of surfaces can be classified into *pre-defined* and *freeform* models. Pre-defined models reduce the geometry to a fixed and typically small number of parameters and usually entail a variety of strong model assumptions. In contrast, freeform representations compose the surface from a variable set of simple geometric primitives (e.g., points or edges) with few model assumptions and thus offer larger flexibility over the geometry that can be represented.

Furthermore, surface representations can be classified as either *parametric* or *implicit*. Parametric models describe the surface using a vector-valued function $f : \Omega \rightarrow \mathbf{s}$ which maps from some parameter domain $\Omega \subseteq \mathbb{R}^2$ (e.g. piecewise linear triangle patches) to surface points $\mathbf{s}_i \in \mathbb{R}^3$. In contrast, implicit models represent the surface as zero-set of a scalar-valued function $F : \mathbb{R}^3 \rightarrow \mathbb{R}$, where the domain is often discretized into voxel grids.



Box	**Polyline**	**Triangle Mesh**	**Surfel Map**
pre-defined parametric	freeform parametric	freeform parametric	freeform implicit

Figure 3.4: An overview and classification of the surface representations covered in this thesis.

The four surface representations covered in this thesis are the box model, the polyline model, the triangle mesh model as well as the surfel map model, all of which are illustrated in Fig. 3.4. The true object surface is assumed to be *rigid* in the remainder, i.e., the object geometry is invariant over time. This assumption holds for the majority of street vehicles, but is violated for, e.g., articulated vehicles and pedestrians, which are beyond the scope of this thesis.

The mutual dependence of surface parameters and object poses necessitates a simultaneous estimation of both states using the measurement and motion models described in sections 3.3 and 3.4. However, the remainder of this section assumes that object poses (or equivalently: the sensor viewpoints) are already known and focuses on the surface reconstruction part of the problem. In particular, it covers

- a formal definition of the surface models, which together with the object poses constitute the state space
- methods for the initialization as well as incremental extension and refinement of the surface representation given already aligned measurements
- prior probabilities over surfaces, where applicable

3.2.1 Box Surface Model

The box surface model has been established as de-facto standard object shape model in the field of automated driving [35, 57, 41, 19, 21, 67]. In its most general 3D variant, the object surface is approximated by the six faces of a cuboid placed at an object pose with six degrees of freedom. However, in the remainder objects are assumed to undergo planar motion on the ground

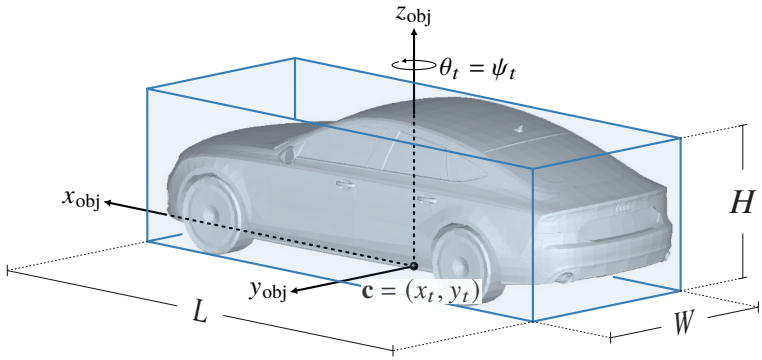


Figure 3.5: An illustration of the box surface model.

surface in order to relax the requirements on the perception system[2]. Moreover, observing the full object height is subject to a sufficiently fine vertical resolution and field-of-view. It therefore is reasonable to omit the height component of the model for low-resolution sensors. The overall box model is defined as follows:

Definition 1. A box surface $\mathbf{s}_{\text{box}} = (\mathbf{c}, \psi, L, W, H)$ is represented by

- a center reference point position $\mathbf{c} \in \mathbb{R}^2$ on the ground surface
- a yaw angle ψ around the $\mathbf{z}_{\text{env}}$-axis
- a box width W, length L, and optional height H

For the predefined box surface model, the object coordinate system origin (x_t, y_t) coincides with the box center position $\mathbf{c}$, whereas the system's orientation θ is equivalent to the yaw angle ψ as depicted in Fig. 3.5.

[2] Note that negligible target object roll, pitch and vertical suspension with respect to the true ground surface are indeed a reasonable assumption. However, the flat world assumption makes this simplification problematic in special cases such as inclined ramps. There, a more detailed road surface estimation would be required from externally or estimated implicitly by adding further object state parameters such as the height and tilt w.r.t. the ground plane.

The (inherently closed) object surface is parameterized by the piecewise linear function that interpolates between the box corner points. Moreover, the normal at each surface point is defined as the outwards-facing normal of the respective box face.

In practical applications, the bounding box model proves to be a versatile tool for tracking: It provides a good approximation for the horizontal cross-section of a wide variety of object classes. Moreover, its compactness and the implied symmetry assumption render it particularly robust. One of its major strength is the ability to recover from erroneous orientation by enforcing zero slip angle, i.e., aligning the box orientation with the direction of the estimated velocity (see section 3.4), which is not directly applicable for freeform models that lack the notion of a pre-defined model orientation.

On the downside, the cuboidal shape assumption is often significantly violated when considering the full 3D object surface rather than only the horizontal cross-section, e.g., for notchbacks and more complex object classes. This introduces tracking residuals that potentially are misinterpreted as object motion rather than geometric modelling errors. In addition, the model abstracts away salient geometric features such as wheel cases that could otherwise assist the association and tracking process. Finally, determining an accurate box fit in an individual LiDAR scan is often a challenging task even for a human observer due to partial observation and sensor noise.

Initial Box Fit

In order to determine an initial guess for the box model parameters as the first observation becomes available, the box fitting algorithm described by Zhang et al. [87] is employed. They propose to uniformly sample the space of possible box orientations ψ and, for every individual orientation sample, determine the bounding box that encloses the complete sequence of scan point observations. From these box fits, they select the box achieving the highest fitting quality score.

Considering the three quality scores proposed in [87], the *closeness score C* achieves the best trade-off between runtime efficiency and fitting quality. It considers, for every observation $\mathbf{z}_t^{(i)}$, the distance d_i to the closest box edge as

$$C = \sum_{i=1}^{M} \frac{1}{\max(d_i, d_{\min})}$$

where M denotes the number of observations and $d_{\min}$ represents a lower cut-off distance threshold that limits the influence of scan points that are very close to their respective edge. Noteworthy, it is sufficient to sample the box orientation in the range $0 \leq \psi < \frac{\pi}{2}$ due to the periodicity of the box model symmetry.

The bounding box fitting method proposed by Zhang et al. arguably is susceptible to extreme outliers as all associated points are used for determining the box extent. However, the preceding scan segmentation step (see section 3.5.1) results in the filtering of far-off scan points. Moreover, discrete sampling of box orientations might appear as an undesirable property of the algorithm as the quantization limits the resolution of the resulting box orientations. However, it is important to note that the algorithm is employed only to determine an approximate initial guess for the object pose and shape model parameters, which are later refined using the probabilistic measurement and motion models described in sections 3.3 and 3.4 respectively.

3.2.2 Polyline Surface Model

The polyline surface model [83, 77, 90] generalizes the box surface model via description of the horizontal object cross-section by an arbitrary, non-intersecting chain of line segments, i.e., a polyline. The optional object height is constant along the 2D curve as illustrated in Fig. 3.6. In contrast to the box model, there is no inherent symmetry assumption and the model allows to represent open surfaces. Moreover, polylines enable an adaptive level-of-detail by matching the control point density to the local surface curvature.

The inherent assumption of locally planar surface patches renders the polyline an efficient and flexible representation for the 2D outline of object geometries. In particular, the polyline model presents itself as well-suited shape represen-

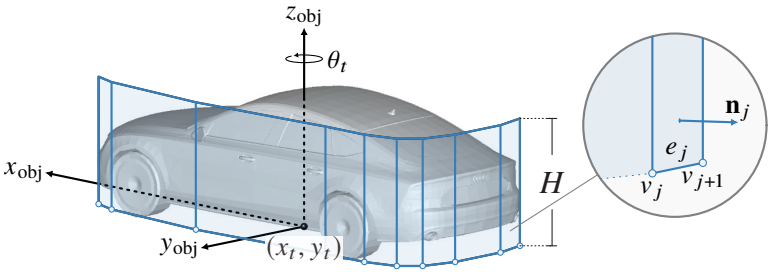


Figure 3.6: An illustration of the polyline surface model.

tation at the interface to behavior generation: While state-of-the-art planning algorithms still operate in the 2D space and thus do not require complex 3D surface descriptions, the classical box model often undesirably oversimplifies the object geometry, e.g., in narrow-spaced bottleneck or parking scenarios as well as evasive steering maneuvers.

On the downside, the polyline surface model provides a crude approximation of more complex 3D surfaces whose geometry varies over their height. This introduces tracking residuals caused by geometric modelling errors rather than actual object motion and might result in *apparent motion*. Some of the concepts described in this section were previously published by the author [90].

A polyline surface consists of a *topological* and a *geometric* component: The topological component specifies the polyline vertices and their connectivity, whereas the geometric component establishes the exact vertex positions in object-local coordinates. In summary, the polyline surface model is defined as follows:

Definition 2. A polyline surface $\mathbf{s}_{\text{polyline}}$ is represented by

- a sequence of *vertices* $v = (v_1, ..., v_{|v|})$, its *control points*, each associated with a position $\mathbf{p}_j \in \mathbb{R}^2$
- an optional height H

The set of polyline *edges* $E = \{e_1, ..., e_{|v|-1}\}$, $e_j \in v \times v$ connects consecutive vertices of the surface model and is implicitly defined by the vertex order.

Polylines are piecewise linear surface representations, where every surface point $\mathbf{s}_i \in \mathbf{s}$ at height h can be determined by linearly interpolating between the respective vertex positions $\mathbf{p}_j$ and $\mathbf{p}_{j+1}$ as

$$\mathbf{s}_i = \lambda \cdot (\mathbf{p}_j^{(x)}, \mathbf{p}_j^{(y)}, h)^T + (1-\lambda) \cdot (\mathbf{p}_{j+1}^{(x)}, \mathbf{p}_{j+1}^{(y)}, h)^T \quad \text{with} \quad 0 \leq \lambda \leq 1\,, \quad 0 \leq h \leq H.$$

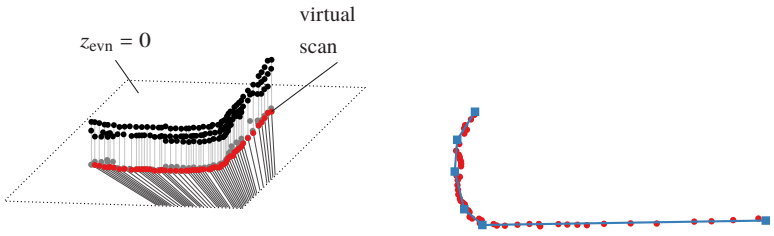
The planar surface normal at that point is equivalent to the outward-facing normal of the corresponding polyline segment. In addition to its actual position, every polyline vertex v_j is associated with a covariance $\Sigma_{v_j} \in \mathbb{R}^{2\times 2}$, which encodes the uncertainty about its position.

Due to the combinatorics of possible vertex counts and positions, an approximate polyline surface model is established in a separate heuristic-based meshing step rather than directly in the probabilistic estimation framework. In the subsequent estimation, the control point count and connectivity then remains fixed and only the vertex positions are refined. There are two separate meshing algorithms: One for the polyline surface model initialization and one for its incremental expansion over time, both of which are detailed in the remainder.

Polyline Model Initialization

Starting from the initial scan point observations of an object, an approximate representation of the surface needs to be established. Given that the polyline surface representation models the object outline as invariant over the height allows for a reduction of the input scan frame to a *virtual 2D scan* [57] as illustrated in Fig. 3.7a. This virtual scan comprises only the closest range readings per azimuth angle after projection onto the ground plane. In addition to decreasing the data load, virtual scans also naturally filter out observations from surface parts other than the object cross-section, e.g., reflections corresponding to the vehicle roof, interior or underbody.

The result is a reduced 2D point set, whose order is inherently known from the sequential scan pattern of the sensor and thus already constitutes a polyline representation of the object surface. However, due to the fine horizontal sampling resolution of LiDARs, the control point density will typically be inappropriately high, in particular at planar surface parts, and thus the model requires



(a) Virtual 2d scan

(b) Polyline simplification ($\epsilon = 0.1\,\mathrm{m}$)

Figure 3.7: Initialization of the polyline model.

additional simplification. The popular Douglas-Peucker algorithm [25] allows to determine a subset of control points that approximate the true object surface without significant loss of information. It is a divide-and-conquer algorithm which, starting from the edge between the first and last control points, recursively subdivides the point set at the point with the largest distance from the current edge. Once the distance to the point furthest away from the current edge is below a threshold ϵ, all points between the current start and end point are discarded.

An exemplary polyline simplification is displayed in Fig. 3.7b. Note that the Douglas-Peucker algorithm results in an interpolating representation, i.e., the control points are a subset of the input. This interpolating surface proxy is sufficiently accurate for initialization as the exact vertex positions are later refined with the *full set* of observations using the probabilistic estimation framework.

Incremental Polyline Expansion and Remeshing

The visibility of previously unobserved object parts requires an expansion of the most recent surface representation. A possible approach would be to generate an approximate polyline model for every individual scan frame using the Douglas-Peucker algorithm and then stitch the individual polylines together. However, the overlapping parts of consecutive polylines will typically be considerably larger than the non-overlapping parts as the viewpoint of the

object only changes gradually. Therefore, any set of new measurements is categorized into *support* and *expansion* points. Support points are scan point observations associated to an existing polyline segment and are used to refine the corresponding vertex positions. In contrast, expansion points represent the scan point observations without correspondence and are added at the polyline ends in order to incrementally complete the surface representation.

Intermediate vertices are inserted to and removed from the polyline model in order to retain an appropriate control point density using a simple heuristics: Firstly, a user-specified upper and lower threshold on the segment lengths is enforced throughout the complete polyline. Secondly, intermediate vertices are removed where a particularly small angle between consecutive line segments occurs. Finally, polyline segments at which large residuals emerge are subdivided temporarily: If a local polyline optimization using the point measurement model (see section 3.3.1) significantly reduces the sum of residuals at the subdivided segment, the intermediate vertex is permanently added to topology. Otherwise, the temporary control point is discarded again.

Surface Prior

The prior surface probability $p(\mathbf{s})$ allows to integrate background knowledge about the structure of surfaces in the world. In particular, all surface geometries are not equally likely, considering that man-made objects tend to feature smooth and homogeneous surfaces. This smoothness assumption is particularly true for traffic vehicles and can act as strong regularization in the geometry reconstruction process. Szeliski and Tonnesen [73] as well as Diebel and Thrun [24] describe the favourable effect of *quadratic co-normality potentials*, which facilitate reconstructions whose surface patch normals are locally aligned.

Noteworthy, these co-normality potentials arise from Gaussian distributions and can be integrated into the probabilistic shape and motion estimation framework via the smoothness prior $p(\mathbf{s})$. As proposed by Diebel and Thrun [24],

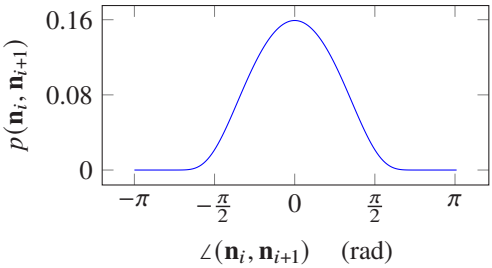


Figure 3.8: Illustration of the smoothness prior with respect to the angle between two adjacent line segment normals $\mathbf{n}_i$ and $\mathbf{n}_{i+1}$. The covariance $\Sigma_{i,i+1}$ is chosen as identity in this example.

given the adjacent edge pairs (e_i, e_{i+1}) with normals $\mathbf{n}_i$ and $\mathbf{n}_{i+1}$ and covariance $\Sigma_{i,i+1} \in \mathbb{R}^{2\times 2}$ the smoothness prior is expressible as

$$p(\mathbf{s}) = \prod_{i=1}^{|E|-1} \left((2\pi)^2 \det(\Sigma_{\mathrm{i,i+1}})\right)^{-1/2} \exp\left(-\frac{1}{2}(\mathbf{n}_i - \mathbf{n}_{i+1})^{\mathrm{T}} {\Sigma_{\mathrm{i,i+1}}}^{-1} (\mathbf{n}_i - \mathbf{n}_{i+1})\right).$$

The resulting probability density function of reconstructions w.r.t. the angle between two adjacent line segment normals is illustrated in Fig. 3.8. It becomes apparent that the smoothness prior favors surface reconstructions with smooth normal transitions and thus locally flat surfaces.

3.2.3 Triangle Mesh Model

Triangle meshes are a popular freeform representation for 3D surfaces in various applications such as computer games, 3D graphics rendering as well as geometric reconstruction. They allow for the approximation of complex surfaces by a set of connected triangle patches. Collectively, these triangles form a piecewise linear continuous surface representation that allows for adaptive resolution by varying the local triangle density. The mesh representation assumes that the object surface is locally planar, but in contrast to the polyline surface model, allows to model varying object outline over height, which is especially beneficial for typical street vehicle fronts (i.e., the hood contour)

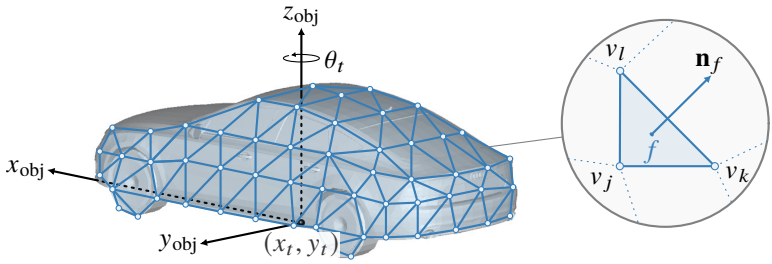


Figure 3.9: An illustration of the triangle mesh surface model.

as well as the rear of object classes lacking a vertically flat rear surface (e.g., notchbacks).

Following the definition provided by Botsch et al. [16], a triangle mesh is defined as follows:

Definition 3. A triangle mesh $\mathbf{s}_{\text{mesh}}$ consists of

- a *topological component* (V, F, E) representing a graph structure with
  - a set of *vertices* $V = \{v_1, ..., v_{|V|}\}$
  - a set of *faces* $F = \{f_1, ..., f_{|F|}\}$, $f_i \in V \times V \times V$ connecting three vertices to form triangle patches
- a *geometric component* P embedding the mesh into $\mathbb{R}^3$ by associating a position $\mathbf{p}_i \in \mathbb{R}^3$ to each vertex v_i

While the mesh topology is already fully described by means of vertices and faces, the additional set of *edges* $E \subset V \times V$ connecting vertex pairs is typically explicitly stored for faster mesh processing.

All components of the mesh are illustrated in Fig. 3.9. Every surface point $\mathbf{s}_i$ on a triangle can be determined using a linear combination of its vertex positions $\mathbf{p}_j$, $\mathbf{p}_k$ and $\mathbf{p}_l$ via the *barycentric coordinates* [16] $(\lambda_j, \lambda_k, \lambda_l)$ as

$$\mathbf{s}_i = \lambda_j \cdot \mathbf{p}_j + \lambda_k \cdot \mathbf{p}_k + \lambda_l \cdot \mathbf{p}_l \quad \text{with} \quad \lambda_j, \lambda_k, \lambda_l \geq 0\,, \quad \lambda_j + \lambda_k + \lambda_l = 1.$$

The resulting piecewise linear vector-valued function of the surface is of class C^0, i.e., continuous in position. The *per-triangle* surface normals $\mathbf{n}_f$ are implicitly defined by the vertex positions as unit-length vector

$$\mathbf{n}_f = \frac{(\mathbf{p}_k - \mathbf{p}_j) \times (\mathbf{p}_l - \mathbf{p}_j)}{\|(\mathbf{p}_k - \mathbf{p}_j) \times (\mathbf{p}_l - \mathbf{p}_j)\|}$$

where the vertex ordering is chosen such that $\mathbf{n}_f$ is outward-facing. Through the topological component of the mesh, the direct neighborhood of any surface vertex, edge or triangle is available without the need for additional search structures. Due to the probabilistic nature of the reconstruction, every vertex v_j in the mesh is associated with a covariance $\Sigma_{v_j} \in \mathbb{R}^{3\times3}$, which encodes the uncertainty about its position. Neglecting the correlation between covariances of different vertices, the covariance $\Sigma_{\mathbf{s}_i}$ at any intermediate surface point $\mathbf{s}_i$ with barycentric coordinates $(\lambda_j, \lambda_k, \lambda_l)$ can be determined using

$$\Sigma_{\mathbf{s}_i} = \lambda_j^2 \cdot \Sigma_{v_j} + \lambda_k^2 \cdot \Sigma_{v_k} + \lambda_l^2 \cdot \Sigma_{v_l}.$$

The uncertainty of a surface point is additionally increased with its distance d from the nearest triangle vertex using a distance-dependent *trust factor* $\xi(d)$ as proposed by Rutishauser et al. [63], i.e.,

$$\overline{\Sigma}_{\mathbf{s}_i} = \xi(d) \cdot \Sigma_{\mathbf{s}_i}.$$

Mesh Generation

The mesh consists of a discrete topological and a continuous geometric component. Due to the combinatorics of possible mesh topologies (i.e., count and connectivity of vertices), the reconstruction process is decomposed into two separate parts:

- The goal of *meshing* is the generation and maintenance of a *surface proxy*, which sets up the mesh topology as well as approximate vertex positions. This step is described in the remainder of this section.
- Subsequently, the mesh geometry, i.e., exact vertex positions, is refined alongside the estimation of object motion using the measurement and motion models. This process utilizes the full set of scan point observa-

tions. It leaves the mesh topology unaltered and is described in more detail in section 3.5.

The input to the meshing process is an organized point cloud sampled from the true object surface. Given the initial observation of an object, *mesh initialization* is performed using a simple and efficient algorithm that exploits the known neighborhood relations from the sequential scan pattern of the LiDAR sensor. As additional measurements are captured over time, the *mesh growing* algorithm expands the triangulation using the aligned observations via a greedy method based on local projections. Finally, the mesh structure is refined in a postprocessing *remeshing* step.

Both mesh initialization and growing generate *interpolating* meshes, i.e., employ a subset of the input points as mesh vertices. In order to reduce the amount of data to be processed and to gain some control over the resulting mesh resolution, prior to triangulation the input point cloud is sparsified using a Euclidean minimum distance threshold $d_{\min}$ between any two neighboring scan points. The final mesh refinement, however, is performed using the full set of scan point observations and thus the vertex positions will typically not coincide with the original observation's positions anymore, i.e., the mesh is transformed from an *interpolating* to an *approximating* representation.

Mesh Model Initialization

The triangulation method employed at mesh initialization exploits the scan line structure of the organized LiDAR point cloud which contains implicit neighborhood information. It is adapted from the greedy algorithm proposed by Carlberg et al. [20], which, starting from an initial seed edge, incrementally advances the mesh along the measurement grid. The algorithm iterates over neighboring scan line pairs and propagates the mesh by stitching together the scan lines using triangles. To grant some control over the mesh structure, a maximum edge length $d_{\max}$ is specified. Moreover, a lower bound $d_{\min}$ on the edge length naturally results from the measurement sparsification in the preprocessing step.

The algorithm is illustrated in Fig. 3.10: The initial step encompasses determining the observation point pair $(\mathbf{z}_t^{(i)}, \mathbf{z}_t^{(j)})$ between scan line l and $l+1$ that

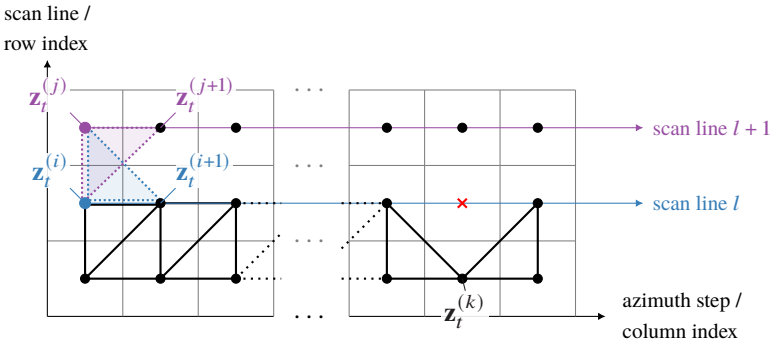


Figure 3.10: Mesh initialization algorithm that exploits neighborhood relations from the scan line structure. It generates the mesh between any two consecutive scan lines by advancing the triangulation along the measurement grid. Figure adapted from [20].

has the smallest column indices in the measurement grid and whose distance is still within the threshold $d_{\max}$. The edge between the points $\mathbf{z}_t^{(i)}$ and $\mathbf{z}_t^{(j)}$ is then taken as the seed edge. Advancing from this edge, there are two potential candidate triangles for growing the mesh, namely

- advance on scan line l, i.e., triangulate the observations $\mathbf{z}_t^{(i)}$, $\mathbf{z}_t^{(i+1)}$ and $\mathbf{z}_t^{(j)}$ *or*
- advance on scan line $l+1$, i.e., triangulate the observations $\mathbf{z}_t^{(i)}$, $\mathbf{z}_t^{(j)}$ and $\mathbf{z}_t^{(j+1)}$

If both triangles are valid, i.e., the maximum edge length criterion is satisfied for all their triangle edges, the candidate with the smaller diagonal grid length is chosen. The corresponding index i or j is incremented to the next valid point in the scan line accordingly. If none of the triangles is valid, the smaller grid index is incremented to keep the indices on par. This process is repeated until the triangulation between the scan lines is completed and the algorithm proceeds to stitching together the next scan line pair $l+1$ and $l+2$.

Note that the resulting triangle mesh can contain holes and non-manifolds (e.g., at observation $\mathbf{z}_t^{(k)}$ in Fig. 3.10) due to the maximum edge length threshold and the fact that the triangulation only considers adjacent scan lines. These

defects are removed in the remeshing step. Moreover, in the presence of large depth discontinuities or missing measurements the algorithm might produce non-connected mesh parts, which are stitched together in the mesh growing step once evidence for their connectivity becomes available.

Mesh Growing

As new measurements are collected over time, the existing surface boundary is incrementally expanded using an adapted variant of the greedy, projection-based surface reconstruction algorithm proposed by Gopi and Krishnan [30]. Their algorithm is founded on three major assumptions

- **Locally uniform sampling**: Under the assumption of locally negligible object curvature, the typically regular scan pattern of state-of-the-art LiDAR sensors results in a fairly regular distribution of scan points on the object surface.
- **Separability of surface layers**: The extent of most traffic participants is comparably large with respect to the sampling density in the near- and mid-range, therefore opposite object surface parts are typically sufficiently far apart to distinctly associate observations to the correct object side. However, thin surface elements such as side-view mirrors might mildly violate this assumption.
- **Surface smoothness**: Due to the choice of the projection plane as average of all neighboring face normals, in [30] the difference of normal vectors at any surface part must not exceed ninety degrees. To avoid this limitation, a slightly altered and more robust choice of projection plane is described below.

The algorithm iteratively considers every vertex on the mesh boundary as reference vertex v_R and tries to propagate the mesh boundary by appending triangles at this location of the mesh. For every reference vertex v_R, a *pruning* step determines an ordered list of candidate points for the mesh extension around v_R, which are subsequently connected in the *triangulation* step. The individual algorithm steps performed at every reference vertex v_R are detailed below and illustrated in Fig. 3.11.

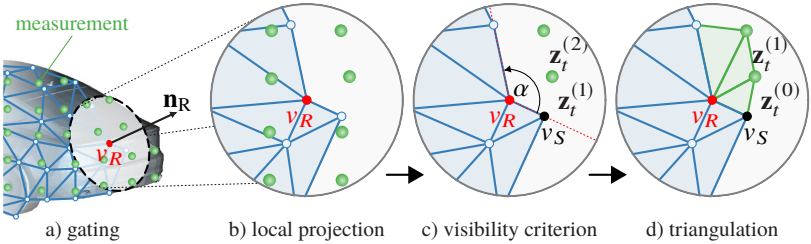


Figure 3.11: Illustration of the four algorithm steps of the greedy, projection-based triangulation method adapted from [30].

- **Gating**: Applying a *distance criterion* around v_R determines all new measurement points and existing mesh parts (vertices, edges and triangles) that are in v_R's *sphere of influence*, i.e., potentially relevant for the triangulation.

- **Projection**: The normal $\mathbf{n}_R$ of the surface at v_R is calculated from all elements in the reference vertex's sphere of influence, i.e., from existing mesh parts as well as new observations [50]. Considering the new measurements ensures a more robust tangential plane estimation at sharp surface parts in contrast to only averaging over the normals of v_R's incident triangles as described in [30]. All measurement points and mesh parts are then projected onto the tangential plane.

- **Visibility & Ordering** Projected measurement points that are not visible from v_R, i.e., occluded by any projected mesh edge, are discarded. The remaining projected measurement points $\{\mathbf{z}_t^{(i)}\}$ are ordered around the reference vertex by increasing angle α with respect to the boundary edge connecting v_R and its adjacent boundary vertex v_S as illustrated in Fig. 3.11c.

- **Triangulation** New triangles are appended to the existing mesh using the remaining candidate points. Starting from the mesh boundary edge between v_R and v_S as seed edge, the observation points $\mathbf{z}_t^{(i)}$ are triangulated in increasing order. After inserting a triangle, the edge between v_R and the respective candidate point $\mathbf{z}_t^{(i)}$ becomes the new seed edge. To grant some control over the quality of the triangulation, only triangles with inner angles of less than $\Delta\alpha_{\min}$ are added to the mesh. More-

over, triangles with an inner angle at v_R above a threshold $\Delta\alpha_{\max}$ (e.g., $\Delta\alpha_{\max} = 120°$ as proposed in [30]) are discarded, since such large angles typically occur at true surface boundaries, e.g., at holes in the object surface.

Note that the pool of candidate points is not confined to new observations $\mathbf{z}_t^{(i)}$, but extended to all mesh boundary vertices except v_R and those vertices sharing an edge with v_R. This results in the algorithm automatically closing holes and stitching together non-connected mesh parts if they are closer than the distance criterion.

Due to the typically high frequency of automotive LiDARs and the smoothness of most scanned object surfaces, consecutive scans of an object tend to be highly overlapping. However, it might happen that individual scan points are far off the boundary of the existing mesh, e.g., through (self-)occlusion or small glancing angles, and thus not considered as candidate points. These points are preserved and added to the measurement set of the mesh growing stage of the successive time steps.

Remeshing

In contrast to the polyline model, the vertex density of the triangle mesh is not adapted to the local curvature of the true surface as this would considerably add to the runtime complexity of the algorithm. However, a heuristic remeshing is performed in order to enable some control over the mesh quality: A minimum and maximum triangle size is enforced via triangle simplification and subdivision respectively. Moreover, faces with very small inner angles at one of their vertices are removed from the topology. Finally, the surface is continuously analyzed for defects such as non-manifolds and self-intersection, which are removed using a set of standard mesh healing operations [16].

Surface Prior

The smoothness prior $p(\mathbf{s})$ described for the polyline surface model in section 3.2.2 is readily applicable to the triangle mesh surface representation [24]: Rather than reasoning about adjacent line segment normals, it aligns the nor-

mal $\mathbf{n}_f$ of any face with the normals from the set F_f of adjacent triangle patches. Given the covariance matrix $\Sigma_{fg} \in \mathbb{R}^{3\times3}$, the surface prior becomes

$$p(\mathbf{s}) = \prod_{f\in F} \prod_{g\in F_f} \left((2\pi)^3 \det(\Sigma_{fg}) \right)^{-1/2} \exp\left(-\frac{1}{2} (\mathbf{n}_f - \mathbf{n}_g)^{\mathrm{T}} \Sigma_{fg}{}^{-1} (\mathbf{n}_f - \mathbf{n}_g) \right)$$

3.2.4 Surfel Map Model

The surfel map model represents object surfaces by a set of oriented discs, the *surfels*, where the surfel orientation allows to distinguish between object interior and exterior [42]. Given the point cloud character of LiDAR scans, surfel maps emerge as a natural choice of representation: While the surfel positions are directly available from the input data, surfel normals can be conveniently inferred from a plane fit into a local neighborhood of any scan point, e.g., via the *principal component analysis*. The disc size can either be a user-specified parameter or be automatically adapted to the curvature of the object. An exemplary surfel map is depicted in Fig. 3.12. Some of the concepts in this section were previously published by the author in [88].

In contrast to parametric surface representations, surfel maps lack any explicit connectivity information. This considerably decreases complexity and fosters robustness as meshing, i.e., establishing the correct topology, remains one of the most critical steps in the reconstruction of polylines and triangle meshes. On the downside, the lack of connectivity information necessitates auxiliary search structures, such as k-d trees [13], for efficient neighborhood queries.

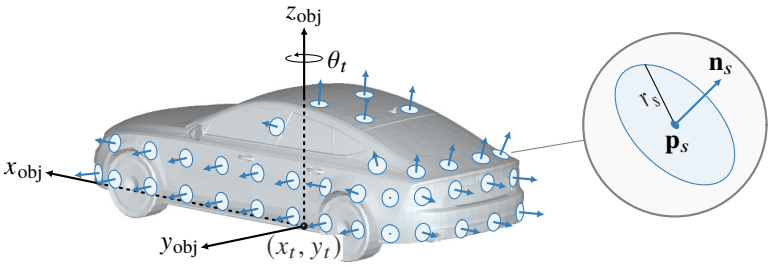


Figure 3.12: Illustration of the surfel map model.

Moreover, additional post-processing is required to extract a compact representation that is suitable as interface to behavior generation. Finally, if the surfel size is chosen uniformly the surfel maps might require a considerable number of surfels to model planar surface parts, which could be represented significantly more compact by polyline segments or triangle patches. Overall, a surfel map is defined as follows [42]:

Definition 4. A surfel map $\mathbf{s}_{\text{surfel}}$ is represented by a

- set of individual surfels $\mathbf{s}_i = (\mathbf{p}_i, \mathbf{n}_i, r_i, c_i)$, each of which comprises
  - a position $\mathbf{p}_i \in \mathbb{R}^3$
  - a normal $\mathbf{n}_i \in \mathbb{R}^3$
  - a radius $r_i \in \mathbb{R}$.
  - a confidence score $c_i \in \mathbb{R}$

The confidence score c_i is a counter variable for the amount of observations that supported the surfel parameters over time and thus reflects the trust in the surfel state.

From the set $\mathbf{s}_{\text{surfel}}$ of surfels, the actual object surface is implicitly defined by the zero-set

$$\left\{\mathbf{y} \in \mathbb{R}^3 \mid d(\mathbf{y}, \mathbf{s}_{\text{surfel}}) = 0\right\}$$

where $d(\mathbf{y}, \mathbf{s}_{\text{surfel}})$ denotes the distance of the point $\mathbf{y}$ to the closest surfel in the set $\mathbf{s}_{\text{surfel}}$, i.e.,

$$d(\mathbf{y}, \mathbf{s}_{\text{surfel}}) = \min(\left\{d'(\mathbf{y}, \mathbf{s}_i) \mid \mathbf{s}_i \in \mathbf{s}_{\text{surfel}}\right\})$$

with $d'(\mathbf{y}, \mathbf{s}_i)$ denoting the distance of the point $\mathbf{y}$ to the surface of the individual surfel $\mathbf{s}_i$ with center position $\mathbf{p}_i$, unit normal $\mathbf{n}_i$ and radius r_i, i.e.,

$$d'(\mathbf{y}, \mathbf{s}_i) = \begin{cases} \mathbf{n}_i \cdot (\mathbf{y} - \mathbf{p}_i), & \text{for } ||\mathbf{y} - \mathbf{p}_i - (\mathbf{n}_i \cdot (\mathbf{y} - \mathbf{p}_i))\mathbf{n}_i|| < r_i \\ & \text{(orthogonal projection of } \mathbf{y} \text{ lies on disc)} \\ \sqrt{\begin{array}{l}(\mathbf{n}_i \cdot (\mathbf{y} - \mathbf{p}_i))^2 \\ + (||\mathbf{y} - \mathbf{p}_i - (\mathbf{n}_i \cdot (\mathbf{y} - \mathbf{p}_i))\mathbf{n}_i|| - r_i)^2\end{array}} & \text{otherwise.} \end{cases}$$

Being part of the surface, every individual surfel theoretically represents a random variable in the estimation process. However, this quickly leads to an intractable dimension of the state space even for comparably large surfel sizes. In this thesis, the surfel map $\mathbf{s}_{\text{surfel}}$ is therefore directly represented by the superposition map $\mathbf{s}_{\text{surfel},0:T}$ of all individual per-observation surfel maps $\mathbf{s}_{\text{surfel},t}$ at any time instant t (i.e., input point clouds) after motion compensation. The overall state space in the estimation process is thus reduced to the object poses $\mathbf{x}_{0:T}$.

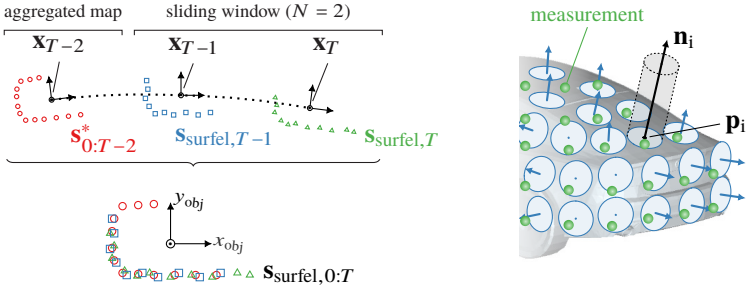
Surfel Map Aggregation

Section 3.5.3 details a sliding window estimation method, which decomposes the sequence of object poses $\mathbf{x}_{0:T}$ into

- a sliding window of the N most recent object poses $\mathbf{x}_{T-N+1:T}$, which are *variable*
- the sequence of all preceding object poses $\mathbf{x}_{0:T-N}$, which remain *fixed*, i.e., are omitted from the estimation

Since the relative location of the poses in the fixed sequence is static, the superimposed surfel map $\mathbf{s}_{0:T-N}$ remains constant as well and thus can be recursively condensed into a single aggregated surfel map $\mathbf{s}^*_{0:T-N}$. The overall surfel map is then superimposed from this aggregated surfel map $\mathbf{s}^*_{0:T-N}$ and the individual surfel maps $\mathbf{s}_{T-N+1:T}$ from the sliding window as illustrated in Fig. 3.13a. The aggregation mechanism fuses overlapping surfels and prevents the map from growing arbitrarily large. Moreover, the fusion of surfels allows to reduce the surface noise and compensate for artifacts from minor tracking errors.

As poses transition from the sliding window into the constant pose sequence, the aggregated surfel map must be updated recursively. For every surfel $\mathbf{s}_i \in \mathbf{s}^*_{0:T-N}$ in the aggregated map, a cylindrical gate with support point $\mathbf{p}_i$ and radius r_i is defined around the surfel normal $\mathbf{n}_i$ as illustrated in Fig. 3.13b. Any new surfel $\mathbf{s}_j$ overlapping with the gate is assumed to originate from the



(a) Surfel map superposition

(b) Recursive integration

Figure 3.13: An illustration of the surfel map formation.

same true surface point and hence is used to update $\mathbf{s}_i$ using the weighted average as proposed by Keller et al. [42]:

$$\mathbf{p}_i \leftarrow \frac{c_i \cdot \mathbf{p}_i + c_j \cdot \mathbf{p}_j}{c_i + c_j} \quad , \quad \mathbf{n}_i \leftarrow \frac{c_i \cdot \mathbf{n}_i + c_j \cdot \mathbf{n}_j}{c_i + c_j} \quad , \quad c_i \leftarrow c_i + c_j .$$

New surfels that are not associated to any existing surfel are added to the aggregated map in order to extend the surface reconstruction. Note that the choice of the gate size r_i allows to control the resolution of the surfel map. Adaption of the gate size to the local surface curvature would be possible. However, for simplicity the gate is set constant, but sufficiently small to capture all relevant surface features, over the entire map in the remainder.

3.3 Measurement Models

Measurement models describe how sensor observations emerge from object states. In particular, LiDAR sensors measure the distance to an obstacle surface by emitting laser pulses and measuring the time taken for the return signal to be received back at the sensor. Besides the raw range readings, LiDARs typically supply an additional measure for the amount of received laser power that allows for reasoning about the reflectance of the sampled surface.

Most state-of-the-art LiDARs scan the environment by rotating a set of sender and/or receiver diodes, which results in varying capture timestamps for individual range readings of a single scan. The result is a potentially distorted scan which, in theory, requires compensation for ego [65] and object motion during the capture. However, due to the high scanning frequency and the usually very restricted azimuth angle range occupied by other traffic participants, in the remainder the local distortion on the object contour is assumed to be insignificant and all range readings of an object are considered to be captured at the mean time instant t. Since only a single instant of time is considered in this section, the time index t is dropped for ease of notation.

In a probabilistic framework, the measurement formation process is modeled as conditional probability density $p(\mathbf{z} \mid \mathbf{x}, \mathbf{s})$, i.e., the probability of observing the vector of measurements $\mathbf{z} = (\mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(M)})$ given the object pose $\mathbf{x}$ and surface $\mathbf{s}$. Assuming independence of the M individual observations $\mathbf{z}^{(i)}$ in the measurement vector, the probability density can be factorized as

$$p(\mathbf{z} \mid \mathbf{x}, \mathbf{s}) = \prod_{i=1}^{M} p(\mathbf{z}^{(i)} \mid \mathbf{x}, \mathbf{s}).$$

Depending on the specific application at hand, it might be favorable to either represent a LiDAR scan as a collection of individual range readings $r^{(i)}$ or as a *point cloud* of individual scan points $\mathbf{p}^{(i)}$. Moreover, the surface normal $\mathbf{n}^{(i)}$ at a measurement sample $\mathbf{z}^{(i)}$ will be required in some parts of this chapter. It can be inferred by a plane fit through a local neighborhood of the scan point, e.g., via the *principal component analysis*.

In the literature, a wide variety of possible measurement model formulations has been proposed, where the following three models have emerged as particularly popular variants for LiDAR sensors [75]:

- **Beam-based models** imitate the physical principle of range sensors by describing the probability $p(r^{(i)} \mid \mathbf{x}, \mathbf{s})$ of observing a particular range reading along the sensor beam. The expected range is determined by ray casting operations, resulting in support for occlusion and freespace reasoning. Moreover, ray casting eliminates the need for any explicit data association as the establishing of correspondences naturally emerges from the ray intersection. On the downside, the ray casting operation comes with high computational costs and results in distributions that typically lack smoothness in the object state: For intersection points close to object corners, for example, a small change in the object position or orientation potentially lead to a significant change in the expected range, thus impeding the suitability of beam-based models for gradient-based estimation methods.

- **Likelihood field models** consider only the range reading endpoints, i.e., provide the likelihood $p(\mathbf{p}^{(i)} \mid \mathbf{x}, \mathbf{s})$ of observing a scan point $\mathbf{p}^{(i)}$ given the object pose $\mathbf{x}$ and surface $\mathbf{s}$. Rather than modelling the likelihood along the beam pathway, endpoints are associated with their closest surface points, thus creating a *likelihood field* around the surface. The required nearest neighbor search is typically much faster than ray casting and the resulting distributions are smooth in the object state, hence facilitating gradient methods. Taking into account these advantages, section 3.3.1 describes a likelihood field-based measurement model which is employed in this thesis. As a drawback, the model requires explicit data association and, not being founded on the physical sensor principle, is oblivious to occlusion or freespace information. As knowledge about freespace is a powerful tool for the inference of probable object states, section 3.3.2 proposes a novel measurement model that explicitly integrates this *negative* sensor evidence into the tracking process and, in contrast to beam-based models, is suitable for gradient-based estimation.

- **Feature-based models** detect salient patterns from the overall measurements, attempting to reduce the data to its most informative part. The models require explicit data association, called *feature matching*, which

can typically be implemented very efficient. More importantly, given the salient signature of features, the association tends to be much more robust than the plain nearest neighbor search employed by likelihood field models. Section 3.3.3 introduces a novel feature extraction method utilizing LiDAR intensities and describes how these features are integrated into the tracking process.

Note that the three measurement models described in the remainder of this section are complementary, i.e., the estimation framework is not restricted to selecting an exclusive model. Also, the object surface parameters are defined in object-local coordinates, as noted in section 3.1. In order to relate the surface to the sensor observations, which are defined in environment coordinates, in the remainder the notation $\bar{\mathbf{s}}$ refers to the surface after transformation into the environment coordinate system by means of the object pose $\mathbf{x}$.

3.3.1 Point Measurements

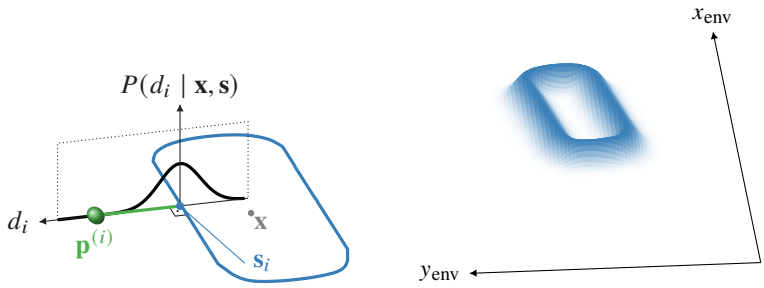
The outline of street vehicles tends to be relatively homogeneous and smooth. This shortage of salient geometric features renders the association of observed scan point samples to their corresponding surface points challenging. Under the assumption that a reasonably accurate initial guess for the object state is available, selecting the closest surface point to any observation represents a simple yet effective association strategy. This approach is commonly referred to as *nearest neighbor association*.

Following this notion, likelihood field models [57, 75] presume any observed scan point $\mathbf{p}^{(i)}$ to be sampled from its nearest neighbor $\bar{\mathbf{s}}_i$ on the surface $\mathbf{s}$ as illustrated in Fig. 3.14a. The residual distance $d_i = ||\mathbf{p}^{(i)} - \bar{\mathbf{s}}_i||$ is assumed to be normally distributed with zero mean and variance σ_d^2, i.e.,

$$d_i \sim \mathcal{N}(0; \sigma_d^2).$$

The likelihood field of this model then reflects the probability of making an observation at any possible scan point position in the measurement space as depicted in Fig. 3.14b.

For the suppression of erroneous associations, nearest neighbor association is typically complemented with *gating*, which refers to the rejection of correspon-



(a) Data association (b) Likelihood field

Figure 3.14: An illustration of the point measurement model.

dences beyond a maximum distance threshold. Further discarding associations with *normal incompatibility*, i.e., large differences between the normals of a scan point and its closest surface point, allows to effectively eliminate the majority of incorrect correspondences.

Residual Formulation For Gradient Methods

Monte Carlo methods, such as particle filters, evaluate the residual functions d_i for a discrete set of state space samples and assign them importance weights based on each samples' measurement likelihood [75]. The association of any scan point $\mathbf{p}^{(i)}$ to its closest surface point $\bar{\mathbf{s}}_i$ is determined for each state space sample individually. Therefore, the underlying functional form of the residual functions (especially their derivative) is not of particular interest. The Euclidean distance, also referred to as *point-to-point* error metric [62], offers itself as natural choice, i.e.,

$$d_i = ||\mathbf{p}^{(i)} - \bar{\mathbf{s}}_i|| \tag{3.3}$$

In contrast, gradient methods start from an initial guess of the state and progress into the direction of the most likely state as indicated by the gradient of the measurement function. Noteworthy, the associated surface point $\bar{\mathbf{s}}_i$ remains fixed during one (or several) iteration steps towards the optimum. This means the residual function is required to capture that the association (i.e., the closest surface point $\bar{\mathbf{s}}_i$) usually changes as the state $(\mathbf{x}, \mathbf{s})$ varies. In fact, the naive

Euclidean residual definition from Eqn. (3.3) fails to factor in the changing association as the state $(\mathbf{x}, \mathbf{s})$ changes. As a consequence, estimation based on these residuals tends to be very sensitive to the quality of the initial guess and risks getting trapped in a suboptimal state.

This problem is mitigated by the *point-to-plane* error metric [62]: Under the assumption of local planarity of the surface $\mathbf{s}$, the immediate neighborhood around $\bar{\mathbf{s}}_i$ can be approximated by the plane with support point $\bar{\mathbf{s}}_i$ and surface normal $\bar{\mathbf{n}}_i$. The point-to-plane error metric then only considers the distance along the surface normal $\bar{\mathbf{n}}_i$, which results in an alternative formulation of the residual d_i as

$$d_i^* = ||\bar{\mathbf{n}}_i \cdot (\mathbf{p}^{(i)} - \bar{\mathbf{s}}_i)||. \tag{3.4}$$

Geometrically, d_i^* is equivalent to the distance between the scan point $\mathbf{p}^{(i)}$ and its orthogonal projection onto the planar surface patch. For planar surfaces, this projection indeed always represents the surface point closest to $\mathbf{p}^{(i)}$ as the state $(\mathbf{x}, \mathbf{s})$ varies and therefore the point-to-plane distances (approximatively) factors in the changing association. The difference between both residual formulations is illustrated in Fig. 3.15: At the initial object pose guess, both distance metrics result in the same residual. As the object pose varies by $\Delta\mathbf{x}$ and the associated surface point $\mathbf{s}_i$ remains fixed, the point-to-point distance d_i fails to capture that the closest surface point changes, whereas (at planar surface parts) the point-to-plane formulation d_i^* correctly determines the distance to the altered closest surface point $\mathbf{s}_i^*$.

Using the real-world example from Fig. 3.16a, the influence of the residual definitions in Eqn. (3.3) and (3.4) is illustrated in more detail. The initial guess of the object position is displaced from the actual observations as depicted in Fig. 3.16a. For the purpose of illustration, the surface state $\mathbf{s}$ remains fixed in this example and potential object rotation is disregarded. The goal is then to determine the translation (t_x, t_y) of the surface, which renders the observation of the depicted point cloud most probable.

Maximizing the measurement likelihood is equivalent to minimizing its negative logarithm, which in turn is proportional to the sum of squared residuals E, i.e.,

$$-\log p(\mathbf{z} \mid \mathbf{x}, \mathbf{s}) = -\sum_{i=1}^{M} \log p(\mathbf{z}^{(i)} \mid \mathbf{x}, \mathbf{s}) \quad \propto \quad E := \sum_i d_i^2. \tag{3.5}$$

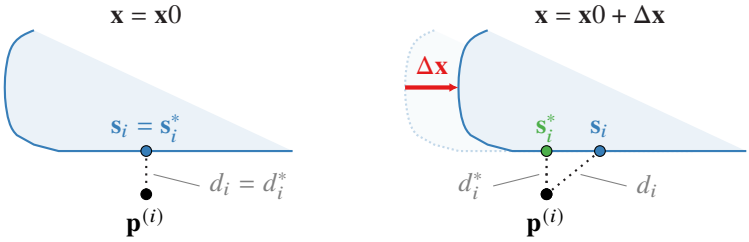
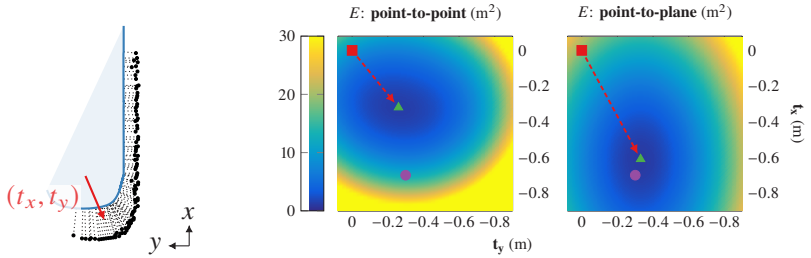


Figure 3.15: Illustration of the difference between the point-to-point residual d_i and point-to-plane residual d_i^* for a fixed association between the observation $\mathbf{p}^{(i)}$ and the closest surface point $\mathbf{s}_i$ as determined at the initial object pose guess $\mathbf{x}0$. As the object pose is varied by $\Delta\mathbf{x}$, the definition of d_i fails to capture that the closest surface point actually changes, whereas (on a planar surface part) the projected residual d_i^* gives the desired distance to the altered closest surface point $\mathbf{s}_i^*$.



(a) Initial configuration of object and point cloud.

(b) Objective function with initial guess (■), fixed-association optimum (▲) and ground truth (●).

Figure 3.16: Illustration of the influence of different residual functions on the alignment of L shapes.

The objective function E for both error metrics is plotted in Fig. 3.16b. The correspondences between observations and surface points is determined once at the initial guess and remains fixed for all evaluated state configurations. Observing the objective function plots, it becomes apparent that the optimal translation for the point-to-plane residuals is considerably closer to the ground truth than for the point-to-point error metric. In particular, the longitudinal motion component is resolved with significantly higher accuracy: Due to the point-to-plane metric only considering residuals along the surface normals, the L shapes are able to slide along each other. In contrast, the point-to-point correspondences on the lateral object surface inflict high costs for longitudinal shifts of the surface and thus prevent a correct alignment. It is worth noting that even iterative reassociation of observations to the surface would not enable the point-to-point error metrics' convergence to the ground truth. Instead, it would approach a translation for which the residuals on the object rear are in balance with the residuals on the object side.

3.3.2 Freespace Measurements

For every emitted laser pulse, the receiver diode of a LiDAR sensor either detects a return signal (*hit*) or the lack thereof (*miss*). Misses emerge from the absence of any obstacle along the beam within the maximum sensor range $r_{\max}$, but also occur at small glancing angles and dark surfaces. This ambiguity renders their interpretation difficult. In contrast, hits are a strong indicator for the presence of an obstacle at the measured range. In particular, a range reading $r^{(i)}$ allows to infer the following about the space along the sensor beam:

A. There is an obstacle at range $r \approx r^{(i)}$.

B. There is no obstacle[3], i.e., freespace, in the range $0 \leq r < r^{(i)}$.

C. There is occlusion for the range $r > r^{(i)}$.

The scan point-based measurement model described in section 3.3.1 leverages beam end points (A), but is oblivious to the extra information about freespace

[3] Transparent surfaces and multi-echo capability are not considered here.

(B). This is undesirable as knowledge about freespace has the potential to considerably narrow down the space of probable object pose and geometry configurations.

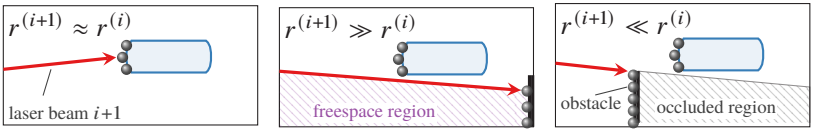
The remainder of this section presents a novel and compact representation for freespace information adjacent to LiDAR scan segments, called *segment delimiters*, which can be extracted alongside scan segmentation at virtually no additional computational costs. A novel measurement model allows to leverage the freespace information provided by segment delimiters. In contrast to existing methods built upon ray casting, it is suitable for gradient methods and deployable in combination with the point measurement model from section 3.3.1. Some of the concepts described in this section have previously been published by the author in [90].

Extraction of segment delimiters

Regular sampling of surfaces with continuous and reasonably smooth structure results in a sequence of measurements with steady range values along the surface. The *Adaptive Breakpoint Detector* [15, 43] exploits this condition for the segmentation of LiDAR scans into groups of scan points presumably arising from the same physical object: It detects depth discontinuities larger than a threshold, referred to as *breakpoints*, between adjacent range readings of a LiDAR scan. These breakpoints mark the division between two separate scan segments. The implicit knowledge of neighborhood relationships from the sequential scan pattern of state-of-the-art LiDARs renders the runtime complexity of the algorithm linear in the number of observations.

Observing that LiDAR beams diverge from a common optical center, the range difference between two neighboring rays impacting on a surface under the same angle of incidence increases with range. To factor in the diverging beam geometry, the threshold $\Delta r_{\max}^{(i,i+1)}$ for detecting a depth discontinuity between adjacent range readings $r^{(i)}$ and $r^{(i+1)}$ is therefore chosen adaptive to the observed range $r^{(i)}$ [15] using the geometric relationship

$$\Delta r_{\max}^{(i,i+1)}(r^{(i)}) = \frac{\sin(\Delta\theta_{i,i+1})}{\sin(\lambda - \Delta\theta_{i,i+1})} \cdot r^{(i)} + 3\sigma_r$$



(a) No breakpoint (b) Freespace (c) Occlusion

Figure 3.17: Possible types of transition between range readings $r^{(i)}$ and $r^{(i+1)}$.

where $\Delta\theta_{i,i+1}$: azimuth angle between laser beams i and $i+1$
λ : user-specified minimal glancing angle
σ_r : sensor noise associated with the range.

Previously published versions of the algorithm [15, 43] only consider if the absolute range difference $|\Delta r^{(i,i+1)}|$ exceeds the given threshold. However, an analysis of the signed difference offers potential for extra inference about the scene. In particular, it is proposed to augment both boundaries of a LiDAR segment with the *breakpoint type* as well as the geometry of the beam causing the breakpoint. There are five possible conditions for the signed range difference $\Delta r^{(i,i+1)} = r^{(i+1)} - r^{(i)}$ between observation $r^{(i)}$ and $r^{(i+1)}$:

No breakpoint	$\left\|\Delta r^{(i,i+1)}\right\| \leq \Delta r_{\max}^{(i,i+1)}$ (Fig. 3.17a)	The range difference is below the detection threshold, thus both observations $r^{(i)}$ and $r^{(i+1)}$ are assigned to the same segment.
Freespace breakpoint	$\Delta r^{(i,i+1)} > \Delta r_{\max}^{(i,i+1)}$ (Fig. 3.17b)	The beam $i+1$ traverses the segment associated to beam i sideways and is reflected by another obstacle in the background[4]. The (local) space beyond beam i is considered to be freespace, i.e., not occupied by any geometry of the object associated with beam i as indicated in Fig. 3.17b.

[4] An additional distance tolerance corresponding to the maximal object size is added to $\Delta r_{\max}^{(i,i+1)}$ for the detection of freespace and occlusion breakpoints. This ensures that the depth dis-

Occlusion breakpoint	$\Delta r^{(i,i+1)} < -\Delta r_{\max}^{(i,i+1)}$ (Fig. 3.17c)	The beam $i+1$ is reflected by an occluding obstacle in the foreground[4], which prevents the observation of potential surface geometry adjacent to range reading $r^{(i)}$ as indicated in Fig. 3.17c.
Field-of-view breakpoint		There is no adjacent observation to range reading $r^{(i)}$ due to the limited field-of-view of the sensor.
Missing echo breakpoint		No adjacent return echo $r^{(i+1)}$ is registered. While individual misses within a segment are tolerated, a large sequence of absent echos results in a breakpoint.

Knowledge of areas with occlusion or at the field-of-view allows for *occlusion reasoning*, e.g., to temporarily preserve object tracks despite the lack of observations [51, 82] or to refine their state estimate [82]. In contrast, breakpoints from missing echos are inherently ambiguous, which renders conclusive reasoning difficult. Finally, freespace breakpoints indicate the absence of any object geometry adjacent to the segment. The remainder of this section describes a novel measurement model for leveraging the negative sensor evidence from these freespace breakpoints in the tracking process.

Note that above conditions can be reversed when determining the breakpoints type on the opposite segment boundary.

Leveraging Freespace Information

The ray casting utilized in beam models implicitly integrates freespace information. However, the resulting measurement models lack smoothness in the object state thus rendering them unfit for estimation methods based on the

continuity is indeed caused by a separate object rather than a jump in the original object's surface.

gradient of the measurement function: Small shifts in the object position, for example, often alter the surface patch intersected by a beam and thus might inflict a step discontinuity in the expected range reading. This occurs, e.g., when the beam's impact point jumps from the vehicle rear to the vehicle side. The effect is even more significant at the object boundary, where transitions from object surface points to the maximum range reading emerge. Inspired by the likelihood field model described in section 3.3.1, a novel measurement model is proposed, which exploits freespace breakpoints and is suitable for gradient methods. Since the model reasons from evidence (i.e., observed freespace) to causes (i.e., object state), it in fact represents an *inverse sensor model* $p(\mathbf{x}, \mathbf{s} \mid \mathbf{z})$. In the remainder, a *segment delimiter* R is defined as

$$R : \mathbf{o} + \lambda \cdot \mathbf{r}^{(i)}$$

where $\mathbf{o}$ denotes the optical center of the sensor, $\mathbf{r}^{(i)}$ specifies the direction of beam i and λ is a scalar.

Fig. 3.18 illustrates an exemplary segment delimiter R generated by a freespace breakpoint at the counter-clockwise segment boundary. Considering an opaque object associated with the corresponding segment, no object geometry should protrude into the freespace, i.e., occupy the left-hand side of the delimiter. In contrast, self-occlusion by the left vehicle corner hinders reasoning about the area on the right-hand side of the delimiter. The freespace information is modeled as follows: Without the pretence of a grounding in the physical principle of the sensor, every surface point $\mathbf{s}_i$ is associated to the closest point on the delimiter R. The signed distance

$$d_i = \mathbf{r}_{\perp}^{(i)} \cdot (\bar{\mathbf{s}}_i - \mathbf{o}) \tag{3.6}$$

denotes the shortest distance from the delimiter to the surface point $\mathbf{s}_i$, where the normal $\mathbf{r}_{\perp}^{(i)}$ on the delimiter is chosen such that d_i is negative if $\mathbf{s}_i$ lies on the freespace-facing side of the delimiter. The distance value then allows to distinguish the location of the surface point with respect to the delimiter. Following the rationale of decreasing surface point probability with increasing protrusion into the freespace and the presumption that the lack of knowledge

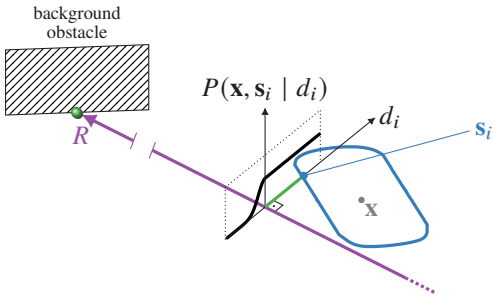


Figure 3.18: Freespace measurement model.

about the occluded area should render all surface point positions in there equally likely, the probability density is constructed as

$$p(d_i) = \begin{cases} \eta \cdot \mathcal{N}(d_i \mid 0, \sigma_d^2), & \text{for } d_i < 0 \\ & \text{(in freespace)} \\ \eta \cdot \frac{d_{\max}}{\sqrt{2\pi\sigma_d^2}} \cdot \mathcal{U}(d_i \mid 0, d_{\max}), & \text{for } 0 \leq d_i \leq d_{\max} \\ & \text{(in occlusion)} \end{cases}$$

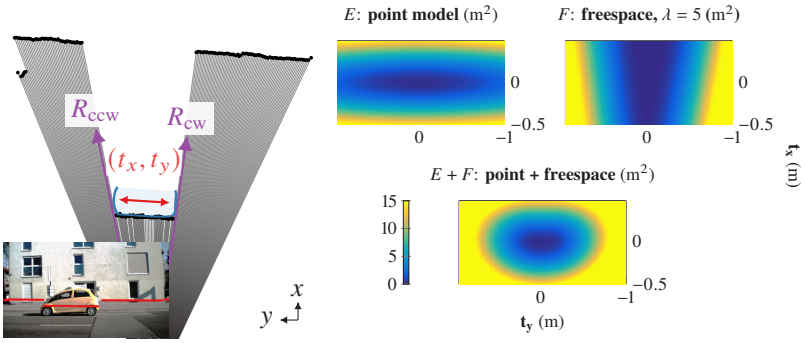
where $\mathcal{N}(x \mid \mu, \sigma^2)$: probability density of the normal distribution with mean μ and variance σ^2

$\mathcal{U}(x \mid a, b)$: probability density of the uniform distribution in $[a, b]$

$\eta = \left(0.5 + d_{\max}/\sqrt{2\pi\sigma_d^2}\right)^{-1}$: normalization factor

$d_{\max}$: upper threshold for the right-hand distance.

The resulting distribution is illustrated in Fig. 3.18. Note that, due to the linear parametrization of the delimiter, the definition of the residual d_i using the point-to-plane error metric in Eqn. (3.6) here accurately factors in the changing distance of $\bar{\mathbf{s}}_i$ to the delimiter as the object state $(\mathbf{x}, \mathbf{s})$ varies.



(a) Initial configuration

(b) Objective function of the point and freespace models

Figure 3.19: Illustration of the freespace measurement model. Note that the freespace model here displays the superimposed costs of the clockwise and counter-clockwise segment boundary delimiters.

The real-world scenario in Fig. 3.19a demonstrates the effect of integrating freespace information into the estimation: The sensor observes the lateral surface of a crossing vehicle, for which the associated delimiters R_{ccw} and R_{cw} on both segment boundaries originate from freespace breakpoints. The goal is to determine the most likely object translation (t_x, t_y) from the observations, where the object orientation and surface state variability are once more disregarded in order to simplify the state space for illustration. Fig. 3.19b plots the corresponding objective function E for the point measurement model (see Eqn. (3.5) in section 3.3.1) and the objective function F derived from the negative log-likelihood of the freespace measurement model

$$F := \lambda \cdot \sum_i d_i^{*2} \quad \propto \quad p(\mathbf{x}, \mathbf{s} \mid \mathbf{z}) \tag{3.7}$$

with distance terms $d_i^* := \max(0, d_i)$ that are truncated on the occluded side of the delimiters and the prefactor λ, which accounts for different weights in the point and freespace measurement models. Note that Fig. 3.19b in fact displays the superimposed costs for the delimiters R_{ccw} and R_{cw} on both segment boundaries, rather than for a single delimiter alone.

From Fig. 3.19b, it can be seen that the point measurement model purposefully restricts the space of probable t_x-shifts, but fails to prevent t_y-shifts that are

extremely implausible when considering the beams reflected by the building in the background, i.e., the observed freespace. In contrast, the freespace measurement model inflicts high costs for surface protrusion into the freespace, but is ignorant to the actual object position within the occluded area. A combination of both measurement models results in a sharper distribution of costs that captures the actual likelihood of object positions much more accurately as shown in Fig. 3.19b (bottom).

3.3.3 Intensity Measurements

Supplementary to the actual range readings, most state-of-the-art LiDAR sensors report back an additional measure for the amount of incoming photons that triggered the measurement at the receiver diode. The physical meaning of this measure differs between sensor types: Some models provide the raw amplitude or width of the received pulse signal, whereas other models supply calibrated quantities that are already mostly independent of sensor properties such as emitter power. To abstract away from the specific sensor model at hand, the quantity is loosely referred to as *intensity* in the remainder of this section. Some of the presented concepts were previously published by the author in [89].

LiDAR intensity values are highly correlated with the sampled surface's *reflectance*, i.e., the proportion of incoming radiant energy that was reflected back by the surface. Therefore, they allow for a detection of highly-reflective vehicle parts such as license plates or reflectors. In the context of automated driving, these distinct features represent particularly versatile means for tracking as traffic regulations render license plates and reflectors omnipresent on public roads.

However, there is a large variety of factors influencing LiDAR intensity readings that, if not accounted for, might hinder their interpretation. Despite (known) physical sensor properties, these include

- distance to target surface
- ratio between beam and object cross sections
- the beam's angle of incidence at the surface

(a) Automotive series sensor (b) Velodyne VLP-16HR

Figure 3.20: Exemplary intensity values provided by LiDAR sensors. The yellow color indicates high intensity as determined for the sensors by (a) the method described in this section and (b) the pre-calibrated values already provided by the manufacturer.

- environmental effects, e.g., weather and ambient light

Some of these factors are hard to quantify and model in practice. The remainder of this section proposes a normalization scheme for the intensity values (here: echo pulse width) provided by an automotive series laserscanner that accounts for the distance to the target surface. Experiments show that this alone already suffices to achieve a considerable detection performance for license plates and reflectors. These are extracted from the normalized intensity values using a compact feature representation and subsequently harnessed in the tracking process.

For illustration, Fig. 3.20 depicts exemplary LiDAR scans from a automotive series and a research sensor respectively. While the intensities of the series sensor have been normalized using the scheme proposed in the remainder of this section, the research sensor already supplies calibrated reflectance out-of-the-box. Even though the discriminative power of LiDAR intensities is not yet on par with that of camera pixels, highly-reflective vehicle parts can clearly be distinguished as illustrated by the intensity-based coloring scheme in Fig. 3.20.

Intensity normalization

The more recent versions of the popular Velodyne research sensors provide calibrated intensity values that are already independent of laser emitter power and distance to the target surface [37]. In contrast, the automotive series

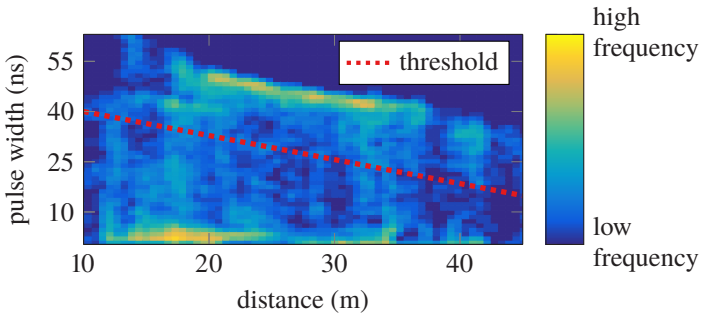


Figure 3.21: Heatmap displaying the frequency of scan points with the corresponding pulse width and range for a vehicle rear.

sensor used in this thesis reports back the *pulse width*[5] of the return signal as registered at the receiver diode. This requires additional normalization for the distance to the obstacle, which is inherently known from the range reading. Noteworthy, a reasonably accurate approximation of the sensor beam's angle of incidence with the target surface is possible from the known beam geometry and the estimated obstacle surface normal. However, an explicit modeling of the incident angle shows no significant gain in detection performance for highly-reflective surface parts and is therefore omitted in the remainder.

The heatmap in Fig. 3.21 displays the frequency of pulse width readings at various ranges collected from a sequence of 7407 individual LiDAR scans of a vehicle rear at different distances and orientations. The cluster at the upper end of the spectrum corresponds to highly-reflective vehicle parts, i.e., the license plate and retro-reflectors, and shows a clear correlation between range and pulse width. Even though the linear distance-dependent threshold indicated in Fig. 3.21 captures the exact relationship only approximately, it suffices to robustly separate reflectors and license plates from the remaining surface in practice. Similar findings were previously reported by Kämpchen [41]. The threshold is chosen tolerant intentionally in order to make allowance for minor unmodeled effects such as mild debris on the vehicle surface. For the particular

[5] The pulse width indicates the time interval for which the return signal is above a manufacturer-specified threshold.

sensor model at hand, the detection range for reflectors and license plates is confined to approximately 10 to 45 meters: While the pulse width is governed by near-field effects for closer obstacles, its expressiveness rapidly declines beyond the range of 45 meters.

Intensity Feature Representation

The fact that LiDAR intensity values are not yet as discriminative as their camera counterparts prevents the extraction of salient feature descriptors, e.g., the signatures of patterns in the plate registration number, from the actual intensity values. Instead, the linear threshold is employed to filter scans for high-intensity points that are then clustered into spatially disparate groups representing distinct vehicle parts, e.g., left and right retro-reflectors. Each individual point cluster is condensed into a rectangular feature representation as depicted in Fig. 3.22. Under the assumption of negligible local surface curvature and mounting of the corresponding vehicle parts orthogonally to the ground plane, the planar feature orientation is determined using linear regression. The overall intensity feature representation consists of

- the feature center position $\mathbf{v} \in \mathbb{R}^3$
- the feature normal $\mathbf{n} \in \mathbb{R}^2$ in the xy_{env}-plane
- the feature width w and height h.

Note that this compact representation renders the feature descriptor size independent of the local sampling density of the input point cloud, i.e., the number of scan points on the feature.

Intensity feature tracking

Intensity features are associated over time based on the distance between their respective center positions (*gating*) as well as their extent and normal compatibility. Any true feature center $\mathbf{u}$ is assumed to be the source for normally-distributed center point observations $\mathbf{v}$, i.e.,

$$\mathbf{v} \sim \mathcal{N}(\mathbf{u}; \Sigma_v)$$

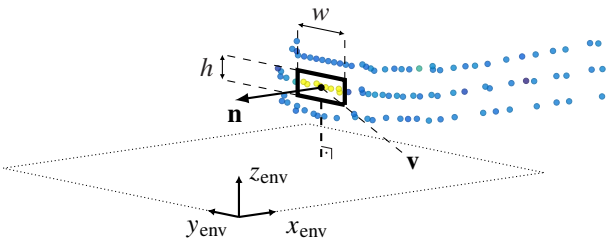


Figure 3.22: Schematic intensity feature on the license plate of an oncoming vehicle.

where Σ_v denotes the covariance associated with the measurement position. Notably, the Euclidean sampling resolution of LiDAR sensors decreases with the distance to the object. Therefore, the measurement uncertainty orthogonal to the feature normal, i.e., along the object surface, is increased with the range. Moreover, observations on highly-reflective surface parts tend to be particularly noisy in the range, which results in a relatively large constant measurement uncertainty in direction of the feature normal.

Finally, the noise in high-intensity scan points also negatively affects the accuracy of extracted feature normals. While the feature orientation theoretically comprises additional evidence on the object orientation, due to the high noise it is intentionally disregarded in the measurement model.

Comparison with other measurement models

The remainder of this section compares the point, freespace and intensity measurement models using the example of a rounded vehicle rear. The experimental setup is shown in Fig. 3.23: Given the measured point cloud, the extracted freespace delimiters as well as the intensity feature, the goal is to evaluate the effectiveness of the individual measurement models in determining the correct pose of the test vehicle (Audi A6 Avant).

For the purpose of illustration, the surface model is fixed to the known ground truth polyline of the vehicle's rear contour depicted in Fig. 3.23. Moreover, the license plate center position is assumed to be located at the rear center of the polyline, which also serves as origin of the object-local coordinate system.

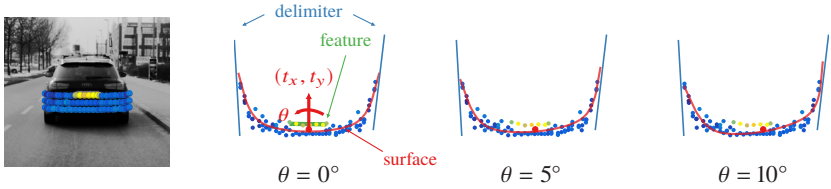


Figure 3.23: Exemplary configurations between a fixed polyline surface model and the corresponding scan point cloud of the vehicle rear. Despite their considerable rotational offsets, for a human observer without additional background knowledge, all configurations represent a reasonable fit to the data.

This leaves the estimation of the model translation (t_x, t_y) and the rotation θ, which are initialized by the ground truth of zero rotation and translation. The analysis in the remainder will show that the intensity measurement model is effective in restricting the space of probable object states in situations where the other models exhibit considerable deficiencies.

A) The objective function E of the *point measurement model* (see section 3.3.1) in Fig. 3.24 displays, for any given model translation (t_x, t_y), the costs under the model rotation θ that minimizes the point measurement objective function from Eqn. (3.5), i.e.,

$$E(t_x, t_y) = \min_{\theta} \sum_i ||\bar{\mathbf{n}}_i \cdot (\mathbf{p}^{(i)} - \bar{\mathbf{s}}_i)||^2 \quad \text{with} \quad \bar{\mathbf{s}}_i = \mathrm{R}_{z_{\text{obj}}}(\theta) \cdot \mathbf{s}_i + (t_x, t_y)^T$$

where $\mathrm{R}_{z_{\text{obj}}}(\theta)$ denotes the rotation by θ around the z_{obj}-axis. In this setting, the problem corresponds to the alignment of the point cloud and polyline using the ICP algorithm with point-to-plane error metric.

The objective function plot in Fig. 3.24 (top left) illustrates the point measurement model's tolerance for considerable sliding of the polyline along the point cloud without significant influence on the resulting costs. The consequence are substantial rotations and shifts of the surface model. For illustration of this effect, Fig. 3.23 visualizes the polyline model under the optimal translations for the three given model rotations $\theta \in \{0°, 5°, 10°\}$, all of which are rendered highly probable by the point measurement model. Interestingly, the alignment task is a non-trivial even for a human observer:

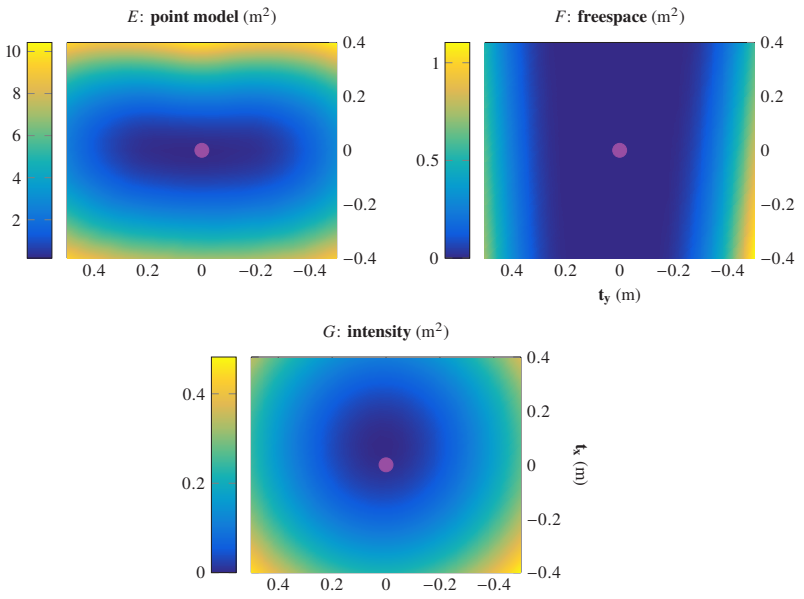


Figure 3.24: Objective function for the measurement models annotated with ground truth (●).

Without additional context knowledge, all three alignments depicted in the figure arguably represent a plausible fit of the polyline to the point cloud.

B) The *freespace measurement model* (see section 3.3.2) renders alignments implausible for which the polyline geometry protrudes into the freespace defined by the delimiters. The corresponding objective function F in Fig. 3.24 (top right) displays, for any given model translation (t_x, t_y), the freespace measurement model costs from Eqn. (3.7) under the rotation θ that minimizes the *point* measurement model[6]. It becomes apparent that, for the example at hand, the freespace measurement model is incapable

[6] For a more realistic analysis, the model rotations θ are determined by minimization of the *point* rather than the freespace measurement model itself. In fact, the freespace measurement models allows for extremely implausible rotations as its objective is solely to keep the object geometry between the delimiters.

of further restricting the space of probable configurations since the most likely alignments determined by the point measurement model already comply with the freespace evidence.

C) The objective function G is derived from the negative log-likelihood of the *intensity measurement model*. It is depicted in Fig. 3.24 (bottom) and represented by the squared distance between the surface model's license plate center $\mathbf{u}$ and the measured feature center $\mathbf{v}$, i.e.,

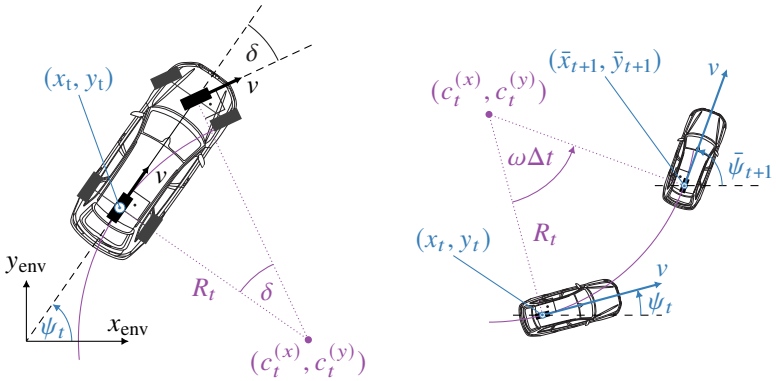
$$G(\theta, t_x, t_y) := \sum_i \| \left[\mathrm{R}_{z_{\text{obj}}}(\theta) \cdot \mathbf{u} + (t_x, t_y)^T \right] - \mathbf{v} \|^2 \quad \propto \quad \log p(\mathbf{z} \mid \mathbf{x}, \mathbf{s})$$
$$\overset{\mathbf{u}=0}{=} \sum_i \| \, (t_x, t_y)^T - \mathbf{v} \, \|^2 = E(t_x, t_y).$$

Noteworthy, the model rotation θ has no influence on the costs inflicted by the intensity measurement model for the special choice of the license plate center as object coordinate system origin and thus center of rotation. The plot in Fig. 3.24 (bottom) therefore displays the costs with respect to the model translation (t_x, t_y) independent of θ. Due to the quadratic residuals in the Euclidean distance of the centers, the measurement model penalizes deviations in longitudinal and lateral directions equally strong. However, there is a considerable longitudinal shift of the optimum from the ground truth model translation. This is in part caused by the large noise in the high-intensity scan points, which is compensated for by the selection of a large variance in the feature normal direction, but also by the fact that in reality the license plate is mounted with a slight inward-shift to the vehicle contour.

In summary, the analysis shows that in the given scene, the point and intensity measurement model compensate for the deficiencies of the respective other model. In particular, intensity measurements tend to correctly determine the surface position along the vehicle contour, i.e., prevent the model from arbitrarily sliding along the observed point cloud. In contrast, employing the full sequence of observations, the point measurement model is effective in determining accurate the model translation in direction of the contour normal as well as the corresponding model rotation. In this particular scenario, the freespace measurements are not able to additionally restrict the space of probable surface

model configurations. However, the results from section 3.3.2 still motivate a complementary application of all three measurement models.

Note that, in order to avoid using point observations multiple times, the high-intensity points are discarded from the point measurement model when using it in conjunction with the intensity measurement model.



(a) Kinematic single-track model.

(b) Stationary vehicle motion on a circle.

Figure 3.25: Illustration of the geometric relationships in the kinematic single-track motion model.

3.4 Motion Model

Motion models provide a mathematical description of the constraints governing the target object's motion. They allow to stabilize object tracks by enforcing a physically plausible evolution of object poses between consecutive time steps. Moreover, motion models enable the prediction of future object poses, which provide an initial guess for consecutive object states and facilitate the association of new measurements to existing tracks. Besides tracking, predicting trajectories of other traffic participants with the help of motion models also enhances the overall scene understanding and plays a significant role when deriving an appropriate driving policy.

Given the high complexity of vehicle kinematics, it is desirable to employ an approximate model that describes the principal geometric relationships of object motion with sufficient precision. In particular, two-axled street vehicles are subject to nonholonomic motion constraints, which pose significant restrictions on the possible object motion. Fig. 3.25a illustrates the kinematic *single-track model* [4], which provides an established approximation to the planar motion of street vehicles with front-wheel steering.

In this model, the front and rear wheel pairs are coalesced into a single wheel on the center of each axle. If the vehicle is not driving close to its' physical limits,

the assumption of negligible slip angle depicted in Fig. 3.25a is justified [4], i.e., the velocity vector of the wheel is always aligned with the wheel orientation. Moreover, given a constant steering angle δ and velocity v, the rear axle center point is undertaking circular motion on the circle with radius R_t and instantaneous center of rotation $(c_t^{(x)}, c_t^{(y)})$. Note that in this thesis the object coordinate system origin (x_t, y_t) is actually placed at the estimated object center (see section 3.1) rather than the true rear axle center point. As the axle center location is difficult to determine reliably, the object coordinate system origin is assumed to be a sufficiently good approximation to the true rear axle center point in the remainder.

From the instantaneous motion circle at object pose $\mathbf{x}_t$, the evolution to the successor pose $\mathbf{x}_{t+1}$ is modeled by predicting the object along the circle as depicted in Fig. 3.25b. The well-known *Constant Turn Rate and Velocity (CTRV)* model [66] allows to predict the delta vehicle motion $\Delta\mathbf{x}_{t,t+1}$ over a small time interval Δt by means of the object velocity v and yaw rate ω as

$$\Delta\mathbf{x}_{t,t+1} = \begin{pmatrix} \Delta x_{t,t+1} \\ \Delta y_{t,t+1} \\ \Delta\psi_{t,t+1} \end{pmatrix} = \begin{pmatrix} R_t \cdot \sin(\psi_t + \omega\Delta t) - R_t \cdot \sin(\psi_t) \\ R_t \cdot \cos(\psi_t) - R_t \cdot \cos(\psi_t + \omega\Delta t) \\ \omega\Delta t \end{pmatrix}. \tag{3.8}$$

The corresponding travelled distance is equal to the length $l = R_t\omega\Delta t$ of the circle sector.

It is worth noting, that the CTRV motion model describes the vehicle motion with respect to the (earth-fixed) environment coordinate system at time instant t. In contrast, the successor pose $\mathbf{x}_{t+1}$ is defined in the (potentially moved) environment coordinate system at time instant $t+1$. Therefore, the auxiliary successor pose $\bar{\mathbf{x}}_{t+1} = (\bar{x}_{t+1}, \bar{y}_{t+1}, \bar{\psi}_{t+1})$ is introduced, which represents $\mathbf{x}_{t+1}$ prior to ego motion compensation (i.e., in the environment coordinate system at time instant t) and thus allows to directly relate the poses $\mathbf{x}_t$ and $\mathbf{x}_{t+1}$. Taking the geometric relationships from Eqn. (3.8) into consideration, the state transition is assumed to be normally distributed around the predicted vehicle pose $\mathbf{x}_t + \Delta\mathbf{x}_{t,t+1}$ with covariance matrix $\Sigma_{\Delta\mathbf{x}}$, i.e.,

$$\bar{\mathbf{x}}_{t+1} \sim \mathcal{N}(\mathbf{x}_t + \Delta\mathbf{x}_{t,t+1}; \Sigma_{\Delta\mathbf{x}}).$$

Object Orientation

Noteworthy, all surface models except for the box representation lack the concept of an explicit longitudinal object axis, i.e., a yaw angle ψ_t. Instead, the object pose defines the object coordinate system orientation θ_t, which may have a constant – but unknown – offset to the actual object yaw angle. Since the above motion model provides the *relative* motion over a time interval, the constant offset cancels out and the predicted change in orientation is also applicable to the object coordinate system orientation θ_t.

The box surface model, with its explicit definition of a yaw angle, however allows for an additional *absolute* orientation constraint at any time instant t. In particular, under the *no slip assumption* the object heading is equivalent to the object's movement direction. Therefore, the box orientation ψ_t is assumed to be normally distributed with variance σ_β^2 around the angle β of the motion circle tangent at pose $\mathbf{x}_t$, i.e.,

$$\psi_t \sim \mathcal{N}(\beta; \sigma_\beta^2).$$

This additional constraint considerably stabilizes box tracks in practical applications by aligning the box orientation with the velocity vector.

Construction of Instantaneous Motion Circles

The motion circle from Fig. 3.25a can be constructed geometrically by intersecting the lines perpendicular to the wheel orientation at each axle. However, this would require the accurate detection of the target vehicle's axle positions as well as its steering angle, which is not practicable in real world applications. Following the rationale that the motion circle represents the *osculating circle* of the object trajectory at the object pose $\mathbf{x}_t$, as depicted in Fig 3.26, the circle is instead determined using a weighted circle fit through a local $2k$-neighborhood of poses around $\mathbf{x}_t$. The circle parameters result from solving the optimization problem [70]

$$\underset{c_t^{(x)}, c_t^{(y)}, R_t}{\operatorname{argmin}} \sum_{i=t-k}^{t+k} w_i \cdot (r_i - R_t)^2 \; ; \quad r_i = \sqrt{(c_t^{(x)} - \bar{x}_i)^2 + (c_t^{(y)} - \bar{y}_i)^2}.$$

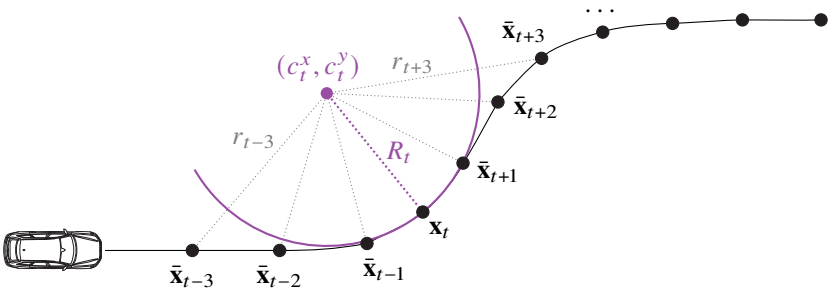


Figure 3.26: The motion circle at any given time t represents the osculating circle of the object trajectory at that time. It is determined by a local circle fit of $\mathbf{x}_t$'s neighboring poses.

The weights w_i are decreased with the (absolute) time difference between poses $\mathbf{x}_t$ and $\mathbf{x}_i$, such that the influence of neighboring poses on the resulting circle fit decreases with their temporal distance. Note that the neighboring poses $\bar{\mathbf{x}}_i = (\bar{x}_i, \bar{y}_i)$ are again transformed into the environment coordinate system at time instant t to allow for directly relating them to the motion circle in that system.

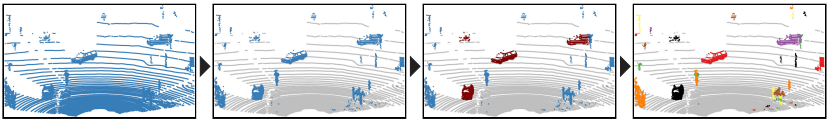


Figure 3.27: Preprocessing pipeline using an exemplary scan from the KITTI dataset [28].

3.5 Practical Implementation

Following up on the theoretic descriptions of the previous sections, this section provides details on the practical implementation of the tracking framework. This includes the preprocessing of raw range images (section 3.5.1), the data flow within the tracking pipeline (section 3.5.2) and a description of bundle adjustment (section 3.5.3) as a method for the simultaneous estimation of object poses and surface.

3.5.1 Preprocessing

Each LiDAR scan is subjected to a sensor-specific preprocessing step, which exploits known sensor characteristics for transforming the raw range image into a segmented point cloud, where every point is augmented with a dynamic classification label either as static, dynamic or of unknown dynamic state. Most state-of-the-art LiDARs sample the environment in a sequential scan-line pattern and typically report back an ordered grid of range readings. This implicit knowledge of neighborhood relations facilitates particularly efficient preprocessing algorithms.

An overview of the preprocessing steps is provided in Fig. 3.27: The *ground classification* algorithm separates ground surface observations from obstacle scan points by considering the distance difference of range readings captured at the same azimuth angle. The rationale behind the analysis is that similar depth measurements at different elevation angles indicate the presence of a vertical obstacle.

For all obstacle points, the dynamic state is inferred by detecting cell conflicts in a gridmap of the static environment [17]. This *dynamic classification* results in per-scan point labels as either static, dynamic or of unknown state.

The Adaptive Breakpoint Detector [43] is implemented for the *segmentation* of the obstacle point cloud into distinct clusters of observations presumably sampled from the same physical object by detection of depth discontinuities. The point groups are augmented with segment delimiters (see section 3.3.2) in order to retain information about the type of space adjacent to the object.

Moreover, an optional *sparsification* of the obstacle point cloud takes into account the varying sampling density at different ranges due to the laser beams sharing a common optical center. Installing a threshold for the minimal permissible Euclidean distance between neighboring point observations allows to considerably reduce the data load, especially in the near-field, without significant loss of information.

Finally, shape representations modelling the object outline as invariant over the height, i.e., the box and polyline model, allow for further reduction of the point cloud data into a *virtual 2D scan* (see section 3.2.2). This decreases the data load and naturally filters out observations from surface parts other than the horizontal object cross-section, e.g., LiDAR echos corresponding to the vehicle roof or interior.

3.5.2 Tracking Pipeline

At the core of the tracking pipeline is a database of active object tracks as depicted in Fig. 3.28. On the availability of a new segmented LiDAR scan, the existing tracks are predicted onto the measurement time. Considering the observed scan point clusters and predicted tracks, the *segment-level association* establishes correspondences between them using velocity-adaptive gating, which will be detailed at the end of this section.

Noteworthy, the association is not one-to-one, but allows for the assignment of multiple segments to an individual track. This renders the matching process tolerant against *oversegmentation*, i.e., the fragmentation of a single object into multiple distinct segments, which might for example occur due to partial occlusion or gaps in sequences of observations under small glancing angles.

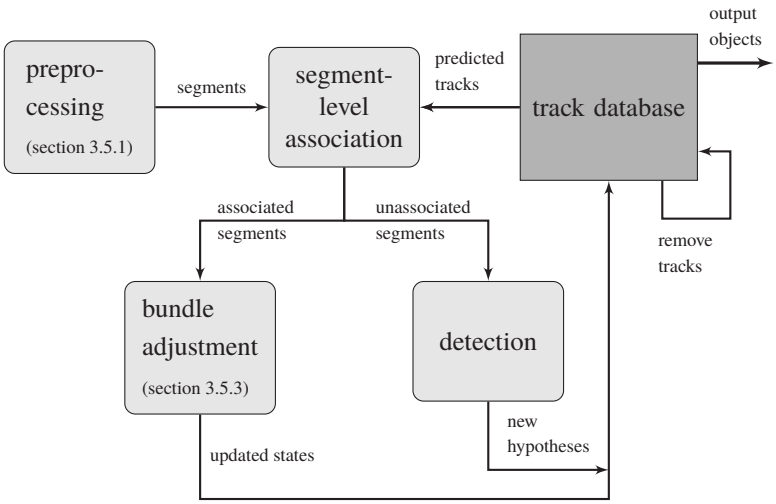


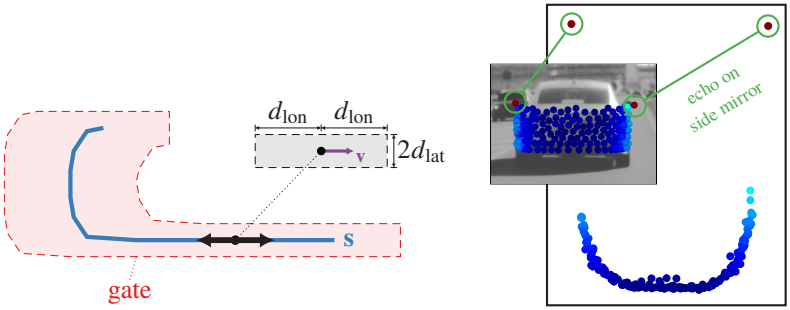
Figure 3.28: Tracking pipeline.

Conversely, the association of several tracks to an individual segment results in their merging.

The state estimates of tracks that have associated segments are updated using *bundle adjustment*, which section 3.5.3 describes in more detail. From the set of unassociated segments, a voting-based *track detection* initiates new object hypothesis for clusters in which the majority of points are labeled as dynamic by the gridmap. In order to suppress false detections, tracks are only forwarded to behavior generation after they have been confirmed for a minimum number of cycles and removed if the elapsed time since the last observation exceeds a given threshold.

Velocity-Adaptive Gating

Scan segments overlapping with the immediate area around the predicted surface of a track, called the *association gate*, are associated to the corresponding track. In the tracking literature, the size of the association gate typically is



(a) Gate construction (b) Side-view mirror association

Figure 3.29: Velocity-adaptive gating.

determined by the uncertainty in the object position estimate. As an object estimate is confirmed over observation time, the uncertainty in the position (and thus the gate size) decreases. For extended objects, however, the gate size must not only take into consideration the uncertainty in the estimation process, but should also account for effects such as partial object (self-)occlusion.

Therefore, a velocity-dependent association gate size is proposed in this thesis, which follows the rationale that faster objects will keep greater safety margin to their environment – in particular in longitudinal direction. This allows for large association gate sizes even for objects with position estimates of high certainty.

Fig. 3.29a displays the velocity-adaptive gate construction: A rectangular gate is aligned with the estimated velocity vector $\mathbf{v}$, comprising a more tolerant longitudinal component $d_{\text{lon}}(\mathbf{v})$ and a more conservative lateral component $d_{\text{lat}}(\mathbf{v})$. A segment is associated to the track, if it contains any scan point $\mathbf{p}$ overlapping with the gate, i.e., that satisfies the condition

$$\frac{\mathbf{v}}{\|\mathbf{v}\|} \cdot (\mathbf{p} - \bar{\mathbf{s}}) \leq d_{\text{lon}} \quad \wedge \quad \frac{\mathbf{v}_{\perp}}{\|\mathbf{v}\|} \cdot (\mathbf{p} - \bar{\mathbf{s}}) \leq d_{\text{lat}}$$

where $\bar{\mathbf{s}}$ denotes the point on the predicted object surface that is closest to $\mathbf{p}$.

At relatively high velocities, e.g., on country roads or highways, this method allows to even associate far-off LiDAR echos on the side-view mirrors of

leading vehicles as depicted in Fig. 3.29b. These observations not only substantially improve orientation estimation, but also facilitate longitudinal extent estimation despite the occlusion of the lateral object surface.

3.5.3 Bundle Adjustment

The tracking literature is prevailed by recursive *filtering* methods, such as Kalman or particle filters, which incrementally update the most recent state estimate over time [75]. In contrast, *smoothing* methods extend the estimation horizon to multiple timesteps and attempt to determine the state configuration maximally consistent with all observations from the time interval under consideration. Notably, the latter approach has evolved under the term *bundle adjustment* [72] in multi-view reconstruction, whereas the SLAM literature refers to the same principle as *GraphSLAM* [32].

In the light of the strong mutual dependence of object poses and shape, temporally consistent estimates are particularly vital. In particular, the use of the shape state for the interpretation of new observations renders the process especially sensitive to error propagation. Therefore, sliding window bundle adjustment is proposed as estimation method for the simultaneous recovery of target object trajectory and shape. This implies the formulation of a non-linear least squares problems that maximizes the likelihood of all observations together with additional motion model constraints. The remainder of this section details the full optimization problem as well as a sliding window approach that renders its solution tractable in online applications.

Full pose and surface optimization

The maximum a posteriori (MAP) estimate of the pose and surface posterior probability in Eqn. (3.2)

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \eta \cdot \prod_{t=1}^{T} \underbrace{p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s})}_{\text{measurement model}} \cdot \prod_{t=0}^{T} \underbrace{p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})}_{\text{motion model}} \cdot \underbrace{p(\mathbf{s})}_{\text{surface prior}} \tag{3.2 revisited}$$

is equivalent to minimizing its negative log-likelihood (see appendix B for details) and thus can be determined by solving the optimization problem

$$\min_{\mathbf{x}_{0:T},\mathbf{s}} \underbrace{\sum_{t=1}^{T}\sum_{i=1}^{M_t} \|\mathbf{z}_t^{(i)} - h_i(\mathbf{x}_t,\mathbf{s})\|^2_{\Sigma^{-1}_{\mathrm{h_i}}}}_{\text{measurement constraints}} + \underbrace{\sum_{t=1}^{T} \|\mathbf{x}_t - f(\mathbf{x}_{0:t-1})\|^2_{\Sigma^{-1}_{\mathrm{f}}}}_{\text{motion constraints}} + \underbrace{\sum_{(\mathbf{n}_i,\mathbf{n}_j)} \|\mathbf{n}_i - \mathbf{n}_j\|^2_{\Sigma^{-1}_{\mathbf{n}}}}_{\text{surface prior constraints}} \tag{3.9}$$

where $h_i(\mathbf{x}_t,\mathbf{s})$: describes the measurement formation of the i-th observation $\mathbf{z}_t^{(i)}$ at time instant t with covariance Σ_{h_i}, see section 3.3

M_t : denotes the number of conditionally independent observations $\mathbf{z}_t^{(i)}$ at time instant t

$f(\mathbf{x}_{0:t-1})$: models the evolution of the object trajectory over time with covariance Σ_f, see section 3.4.

Note that all terms in Eqn. (3.9) are *soft constraints*, i.e., violations are possible, but will be penalized with quadratic costs weighted by the inverse covariance matrices. This weighted non-linear least squares problem is tackled using the well-known Levenberg-Marquardt [55, 2] algorithm, which iteratively approaches the (local) optimum using linearizations of the objective function.

Sliding window optimization

The amount of parameters in the full object trajectory $\mathbf{x}_{0:T}$ as well as the number of constraints in the optimization problem scale with the observation time. This rapidly eventuates in estimation problems that are intractable under the requirement of online processing. The time-dependence of object poses allows to decompose the full object trajectory $\mathbf{x}_{0:T}$ into

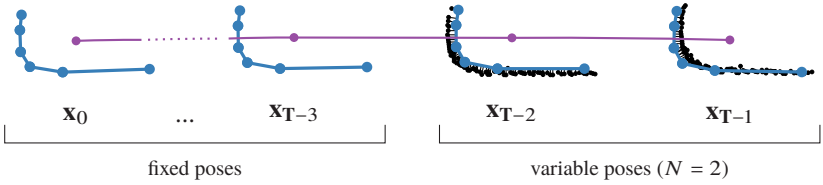


Figure 3.30: Illustration of sliding window optimization with exemplary window size $N = 2$.

- the sliding window $\mathbf{x}_{T-N+1:T}$ of the most recent N object poses, which are treated as *variable* parameters in the optimization
- the sequence $\mathbf{x}_{0:T-N}$ of precedent object poses which remain *fixed*, i.e., are not part of the optimization.

This effectively reduces the number of optimization parameters. The specific choice of the sliding window size is always a trade-off between computational efficiency and the accuracy of the estimate.

Note that, despite fixing a subsequence of the object poses, the estimation of the time-invariant shape state $\mathbf{s}$ theoretically still requires all measurement constraints from the full sequence of observations $\mathbf{z}_{1:T}$. Therefore, the surface state is estimated semi-recursively: The measurement constraints for the observations $\mathbf{z}_{T-N+1:T}$ in the sliding window are integrated as indicated in the original Eqn. (3.9), whereas the contribution of the precedent observations $\mathbf{z}_{1:T-N}$ on the surface state $\mathbf{s}$ is modeled as Gaussian distribution of their position. In particular, given the prior belief of the surface primitive v_i's position $\mathbf{p}_i$ with mean $\bar{\mathbf{p}}_i$ and covariance Σ_i, the optimization is augmented with the quadratic costs

$$\sum_{v_i} \|\bar{\mathbf{p}}_i - \mathbf{p}_i\|^2_{\Sigma_i}.$$

This reduces the sequence of relevant observations to $\mathbf{z}_{T-N+1:T}$ as illustrated in Fig. 3.30.

Robustification

One advantage of bundle adjustment over recursive estimators is that it allows for re-evaluation – and in particular relinearization – of the measurement and motion model functions h and f at the permanently updated estimates of the state history.

More importantly still, replacing the squared loss in the optimization problem with a less heavy-tailed distribution renders the estimation more robust against *outliers*, which might hinder the convergence of other estimators. In particular, the quadratic residuals from Eqn. (3.9) are augmented with a loss function $\rho(\cdot)$, i.e.,

$$E(r_i) = \rho(r_i)$$

The *trivial loss* $\rho(r_i) = r_i^2$ leaves the optimization unaltered, whereas the use of the piece-wise defined *Huber loss* [36] results in the objective function

$$E_H(r_i) = \begin{cases} r_i^2, & \text{for } |r_i| \leq \delta \\ 2\delta|r_i| - \delta, & \text{otherwise} \end{cases}$$

which renders the estimation less susceptible to residuals larger than the threshold δ and thus drastically mitigates the influence of outliers.

3.6 Conclusion

This chapter provided a comprehensive summary of the probabilistic framework for the simultaneous tracking and shape estimation of rigid traffic participants. This includes the definition of the coordinate systems involved (section 3.1) and methods for the reconstruction of four different surface models (section 3.2). Three complementary measurement models (section 3.3) leverage evidence from range reading endpoints, freespace at the segment boundaries as well as LiDAR intensities to refine the object motion and surface estimates. The evolution of object states over time is constrained by a motion model (section 3.4) based on the well-established single-track motion model for two-axled street vehicles. Finally, implementation details (section 3.5) are provided for data preprocessing, association and the estimation via sliding window bundle adjustment.

4 Adaptive Surface Model Selection

The availability of a set of different surface models for representing the object shape prompts the question for the *most appropriate* model from the given set. The model selection should desirably be governed by a diverse set of criteria, including the geometric properties of the observed object, its present relevance for the driving policy as well as the capabilities of the automated vehicle's perception system itself.

Noting that some of the decision criteria are object-specific and variant over time, motivates the development of a surface model selection strategy that continually (re)elects the best model *in situ* based on object properties and the evolution of the traffic scenario at hand. This results in a mixed surface model tracking framework which employs the most suitable surface representation for every individual object as depicted in Fig. 4.1.

The remainder of this chapter is structured as follows: Section 4.1 describes a set of potential decision criteria for a surface model selection scheme. Based on a subset of these criteria, section 4.2 proposes a prototypical selection strategy that adaptively switches between the box and triangle mesh models, followed by a brief conclusion in section 4.3.

4.1 Selection criteria

Despite mutual interdependencies often rendering a clear distinction of criteria challenging, in the remainder they are categorized into the four broad classes:

- **Observability** (section 4.1.1) comprises selection criteria derived from the system's perceptual capabilities.
- **Model adequacy** (section 4.1.2) reflects the efficiency of a surface model in approximating the given observations.

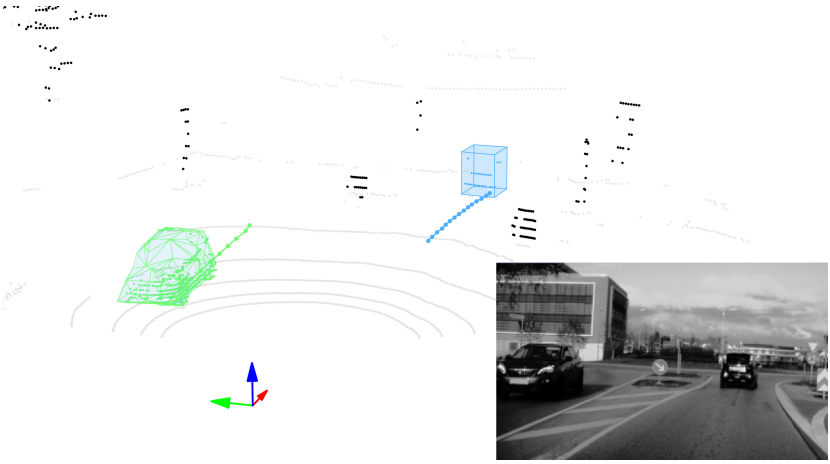


Figure 4.1: Exemplary mixed surface model scenario, where the vehicle in the near-field (left) is represented using a triangle mesh surface model and the vehicle farther away (right) is modeled using a simple box.

- **Contextual criteria** (section 4.1.3) take into account the object's present role in the traffic scenario at hand.
- **Functional requirements** (section 4.1.4) describe demands on the surface model imposed by extrinsic entities such as motion planning.

In the context of this chapter, it is important to bear in mind the differentiation between the *internal* representation and the actual *output* object description: Often, it might be desirable to employ a sophisticated internal model for precise and stable dynamics estimates, whereas a more abstract output representation potentially suffices for the driving function at hand.

4.1.1 Observability

Observability criteria comprise factors derived from the capabilities of the automated vehicle's environment perception system. The key question is: To what extent is the system able to adequately populate the surface model at hand? In large part, this is governed by the sensor characteristics: Some sensor

modalities are capable of perceiving multiple spatially distributed samples of the target surface at any instant of time (e.g., LiDAR or modern radar sensors), whereas other sensor types might only return a single point observation per object (e.g., classical radars). In a multi-sensor setup, this allows for model selection based on the type of sensor modalities supporting a model at any given instant of time as proposed by Darms et al. [22].

The *field-of-view* and *sampling resolution* of a sensor also play a crucial role for the choice of a reasonable surface representation. The observation of a single scan line on an object, for instance, clearly renders a full 3D representation an inappropriate model choice. The radial measurement principle of LiDARs renders their spatial sampling density highly dependent on the distance to the target object, which motivates range-dependent observability criteria such as

- Within what range is the sensor's sampling resolution sufficiently dense in order to perceive a relevant surface curvature?
- Within what range is the sensor's sampling resolution sufficiently dense in order to approximate the surface between two neighboring scan points by planar patches (i.e., connect them by an edge)?

Despite raw sensor characteristics, (partial) *occlusion* of the target object as well as *environmental influences* such as rain, fog or snow potentially degrade the performance of vision systems and thus might hinder the use of sophisticated surface models.

4.1.2 Model Adequacy

Model adequacy criteria assess the efficiency of a surface model in approximating the given observations. Noting that the best approximation of any set of observations are the observations themselves, renders the *principle of parsimony* a key policy in model selection theory [5]: It states that simple models (in terms of structure and number of parameters) are generally preferable over complex models.

Information criteria follow this notion and provide an estimate of the model adequacy by trading off model complexity against the discrepancy between the actual observations and their predictions from the model fit [5]. In contrast

to other decision criteria described in this chapter, the model selection via an information criterion is *evidence-driven*, i.e., assesses the adequacy of a model as indicated by the actual observations, which renders it a highly appropriate decision policy.

In particular, model selection based on model adequacy allows to eliminate tracking instabilities potentially originating from severe violations of surface model assumptions, e.g., when the simple box provides an inappropriate over-approximation of the object surface. Section 4.2.2 describes a prototypical implementation of a model selection strategy via the Akaike information criterion in full detail.

4.1.3 Contextual Criteria

Contextual selection criteria take into account situative aspects of a traffic scenario as well as the target object's present or predicted future role within the scene. An example for contextual knowledge is the *traffic domain*: Highway driving is typically reduced to relatively homogeneous traffic flow which might be approximated sufficiently precise by simple box models. In contrast, urban driving might require more sophisticated shape models for accurately capturing the more complex object maneuvers and diversity of object classes involved. The *object class* itself also provides a useful selection criterion, particularly in conjunction with a set of pre-defined surface models for common object classes such as sedans, vans, trucks and two-wheeled vehicles.

Wyffels and Campbell [84] propose an attention control of the perception system that selects geometric representations based on an object's *relative position to the ego vehicle*, which is not motivated by the limited sensor capabilities (see section 4.1.1), but rather considers an object's relevance for the automated vehicle. Additionally taking into account the object's *velocity and heading* would be an useful extension to their concept in order to predict the future relevance of an object.

Moreover, surface model selection could be governed by *maneuver detection*, such as merging, U-turns or roundabout motion patterns, as well as by the detection of *driving intentions*, for instance derived by the vehicle's turn indicator or the orientation of the wheels. In addition, transitions to a more sophisticated surface model are conceivable on the detection of *departures from the*

normal behaviour, e.g., caused by an unexpected evasive steering maneuver of a leading vehicle.

Finally, given the limited computational resources available, the model selection should be influenced by the total number of objects in the scene in order to meet the time constraints posed by the online nature of the application [84].

4.1.4 Functional Requirements

Functional requirements comprise demands on the perception system imposed by extrinsic entities such as motion planning. This involves the level of geometric detail required by a particular driving function as well as demands on the quality of dynamic estimates, which might be implicitly influenced by the choice of a surface model.

Noteworthy, the relevance of object properties is heavily conditioned on the specific driving function at hand, which renders the derivation of a generic assessment strategy difficult. For instance, an implementation of adaptive cruise control primarily relies on the location and dynamic estimates of the leading vehicle as well as potential candidates for a cut-in maneuver, but is oblivious to the exact contour of other road users. In contrast, a collision avoidance system might evaluate object relevance based on the likelihood of collision and the severity of the resulting damage. Exact shape information of high risk objects is vital for this function in order to infer trajectories that avoid a collision or at least mitigate the resulting damage.

4.2 Prototypical Implementation

This section presents a prototypical implementation of an adaptive surface model selection based on observability and model adequacy. In particular, the model detail in any selection strategy should be bounded by a “goodness of fit” test as there is hardly any justification for the choice of an advanced model, if the actual observations are best represented by a simple one. This motivates the selection of the most adequate surface model based on the Akaike information criterion, which provides a trade-off between model complexity and approximation error. For simplicity, the selection is restricted to the binary

choice between the box and triangle mesh models in the remainder. Still, the selection process allows for natural extension to a wider variety of candidate models as the information criterion inherently provides a quantitative ranking between any number of models.

The model adequacy test is augmented with a fast and simple preselection mechanism, which assesses object observability using a constant range threshold derived from the sensor specification. Contextual and functional decision criteria are deliberately omitted from the prototype due to their strong dependency on external modules and context knowledge.

An important prerequisite for adaptive model switching is the ability to transition between the surface models. Downgrading a model instance to a more abstract representation typically is a viable process, e.g., by determining a box fit of the mesh vertices. In contrast, the upgrade from an abstract to a more detailed model often is far less practicable: While it would theoretically be feasible to triangulate the six face of the box model into a triangle mesh, the symmetry assumption of the box model results in an automatic completion of the surface model even for previously unobserved surface parts. An improper estimate of the box dimensions (e.g., due to occlusion) thus results in an inappropriate triangulation from which robust recovery is particularly difficult. Therefore, the upgrade from box to triangle mesh is realized by the initialization of an entirely new triangle mesh from the observations available in the sliding window. While the prior belief about the object poses is retained as initial guess, unfortunately the model switch might entail a loss of shape information.

4.2.1 Observability-Based Preselection

From the angular resolution α of a LiDAR sensor, its Euclidean sampling density d_s at range r under perpendicular angle of incidence can be approximated by the projection

$$d_\mathrm{s} \approx 2 \cdot \sin\left(\frac{\alpha}{2}\right) \cdot r.$$

Considering a surface model which requires a sampling density of at least $d_s^{\max}$, the maximum range $r_{\max}$ at which the model is still admissible due to its observability with the given sensor is approximately

$$r_{\max} \approx \frac{d_s^{\max}}{2 \cdot \sin(\alpha/2)}.$$

For illustration, consider the research LiDAR sensor Velodyne VLP-16HR [37] with a vertical resolution of $\alpha_{\text{vert}} = 1.33°$ and (spinning at 12.5 Hz) a horizontal resolution of $\alpha_{\text{hor}} = 0.25°$. In conjunction with a minimal Euclidean sampling density of $d_s^{\max} = 0.4\,\text{m}$ assumed for the triangle mesh model, the limited vertical resolution results in a range threshold $r_{\max} \approx 17\,\text{m}$. For objects farther away from the sensor, the preselection algorithm will always choose the box model. In contrast, the model selection in the near-field is performed using the observation-driven Akaike information criterion described in the remainder of this chapter.

4.2.2 The Akaike Information Criterion

The *Akaike information criterion (AIC)* is an estimator for the information lost when approximating the true data generation by a given model of its formation process [5]. Therefore, the AIC allows to rank a set of candidate models with respect to their adequacy in predicting the observed data. This ranking directly enables the identification of the best model based on an information-theoretic foundation.

In general, given the likelihood $\mathcal{L}(\hat{\boldsymbol{\theta}} \mid \mathbf{z})$ of the most likely realization $\hat{\boldsymbol{\theta}}$ of a model under observations $\mathbf{z}$, the AIC [5] for that model is defined as

$$\text{AIC} = -2 \ln \left[\mathcal{L}(\hat{\boldsymbol{\theta}} \mid \mathbf{z}) \right] + 2K$$

where K denotes the number of model parameters and the maximum likelihood estimate $\hat{\boldsymbol{\theta}}$ of the model is given by

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} \; \mathcal{L}(\boldsymbol{\theta} \mid \mathbf{z}).$$

Note that a lower AIC implies a higher model adequacy. Moreover, trading off the model size K against the negative log-likelihood of the observations, the AIC criterion is in accordance with the *principle of parsimony*.

For AIC-based model ranking, it is proposed to consider the adequacy of the model realization $\hat{\boldsymbol{\theta}} = (\hat{\mathbf{x}}_t, \hat{\mathbf{s}})$ at any time instant t in isolation and thus the time index t is omitted in the remainder. An assessment of the model fit determined from the most recent N or all scan frames *collectively* would also be feasible. However, this would imply the use of pose estimates from the tracking, whose quality in turn depends on the adequacy of the employed surface model.

In order to prevent frequent model switches from frame to frame, the selection process is based on the majority of model votes from the sequence of the most recent N frames in the sliding window.

For the selection problem at hand, the data generation process is modeled using the point measurement model (see section 3.3.1) as follows: Given an initial fit of the box or triangle mesh model from the M observations at the current instant of time, the maximum likelihood estimate $(\hat{\mathbf{x}}, \hat{\mathbf{s}})$ of the respective surface model is determined by minimizing its negative log-likelihood, i.e., the sum of squared residuals between observations and their closest surface points. Thus

$$(\hat{\mathbf{x}}, \hat{\mathbf{s}}) = \underset{(\mathbf{x},\mathbf{s})}{\arg\min} \sum_{i=1}^{M} d_i^{*2} \tag{4.1}$$

where

$$d_i^* = ||\bar{\mathbf{n}}_i \cdot (\mathbf{p}^{(i)} - \bar{\mathbf{s}}_i)|| \tag{3.4 revisited}$$

denotes the point-to-plane distance of an observation points $\mathbf{p}^{(i)}$ to its closest surface point $\bar{\mathbf{s}}_i$ with normal $\bar{\mathbf{n}}_i$. Remember that the overline on $\bar{\mathbf{s}}_i$ and $\bar{\mathbf{n}}_i$ indicates a transformation of the point and normal from object-local to environment coordinates by means of the object pose $\hat{\mathbf{x}}$ in order to transform observations and surface in a common system. The optimization determines the most likely box and mesh model fits under the point measurement model.

It is worth noting that the AIC evaluates the adequacy of the given surface model *realization* $(\hat{\mathbf{x}}, \hat{\mathbf{s}})$ rather than the surface model in general. In particular, the most likely model parametrization as determined by the point measurement model disregards the variability of surface topology (i.e., vertex count

and connectivity) in free-form models. Therefore, there always remains the possibility of a more adequate realization of the surface model (as measured by the AIC), e.g., with a different parameter count K or connectivity, which is not examined in the selection process. Still, the surface reconstructions produced by this method are considered a sufficiently accurate approximation of the true most adequate realization of the given surface model.

The stated assumption of normally distributed residuals d_i (representing the distance from observation to closest surface point) allows to significantly simplify the calculation of the criterion. In particular, the model's maximum log-likelihood value (see Eqn. (A.7) for a detailed derivation) becomes

$$\ln \mathcal{L}(\hat{\mathbf{x}}, \hat{\mathbf{s}} \mid \mathbf{z}) = -\frac{M}{2} \ln \left[\sum_{i=1}^{M} d_i^{*2} \right] + C$$

where the offset C only depends on the (constant) observation count M, thus not influencing the ranking. The overall AIC-based score for any candidate model realization can then be determined as

$$\mathrm{AIC}^* = \underbrace{M \ln \left[\sum_{i=1}^{M} d_i^{*2} \right]}_{\text{goodness of fit}} + \underbrace{2K}_{\text{model complexity}} \tag{4.2}$$

where the model with the lowest score provides the best trade-off between goodness of fit and model complexity, i.e., represents the most adequate model choice under the AIC. In order to mitigate the influence of outliers on the selection process, the squared residuals in Eqn. (4.2) can be additionally substituted by the less heavy-tailed cost terms, e.g., Huber loss (see *robustification*, section 3.5.3).

4.2.3 Example

The model selection strategy is illustrated using the traffic scenario with two different object classes depicted in Fig. 4.2: The car on the left-hand side is a station wagon (Audi A6), whereas the vehicle on the right-hand side is a van (Volkswagen Caddy). Since both objects are closer than 17 meters to

the sensor origin, they both pass the range-based preselection stage and are separately analyzed in detail using the AIC-based model selection mechanism.

For each of the two objects, an initial model fit for the box and triangle mesh surface models (see sections 3.2.1 and 3.2.3 respectively) is generated. Starting from the initial fit, each model's maximum likelihood estimate is determined by minimization of the measurement residuals as indicated in Eqn. (4.1). The resulting parameters define the model realizations that are subsequently ranked using the AIC-based score from Eqn. (4.2).

The four maximum likelihood model fits in Fig. 4.2 are annotated with their respective residuals d_i. For the station wagon (left), the varying horizontal cross-section of the trunk causes considerable residuals when employing the box model. This results in a small model likelihood $\mathcal{L}(\hat{\mathbf{x}}, \hat{\mathbf{s}} \mid \mathbf{z})$ and thus the selection of the mesh model. In contrast, the almost constant horizontal cross-section of the van, together with the relatively high vertex count of the triangle mesh, results in the selection of the box model.

4.3 Conclusion

This chapter described a set of potential decision criteria for the online selection of a surface model in the context of object tracking (section 4.1). Based on two exemplary criteria, a prototypical selection strategy was presented (section 4.2): Following model preselection based on sensor resolution, a goodness-of-fit test based on the Akaike Information Criterion determines the model providing the best trade-off between model error and complexity.

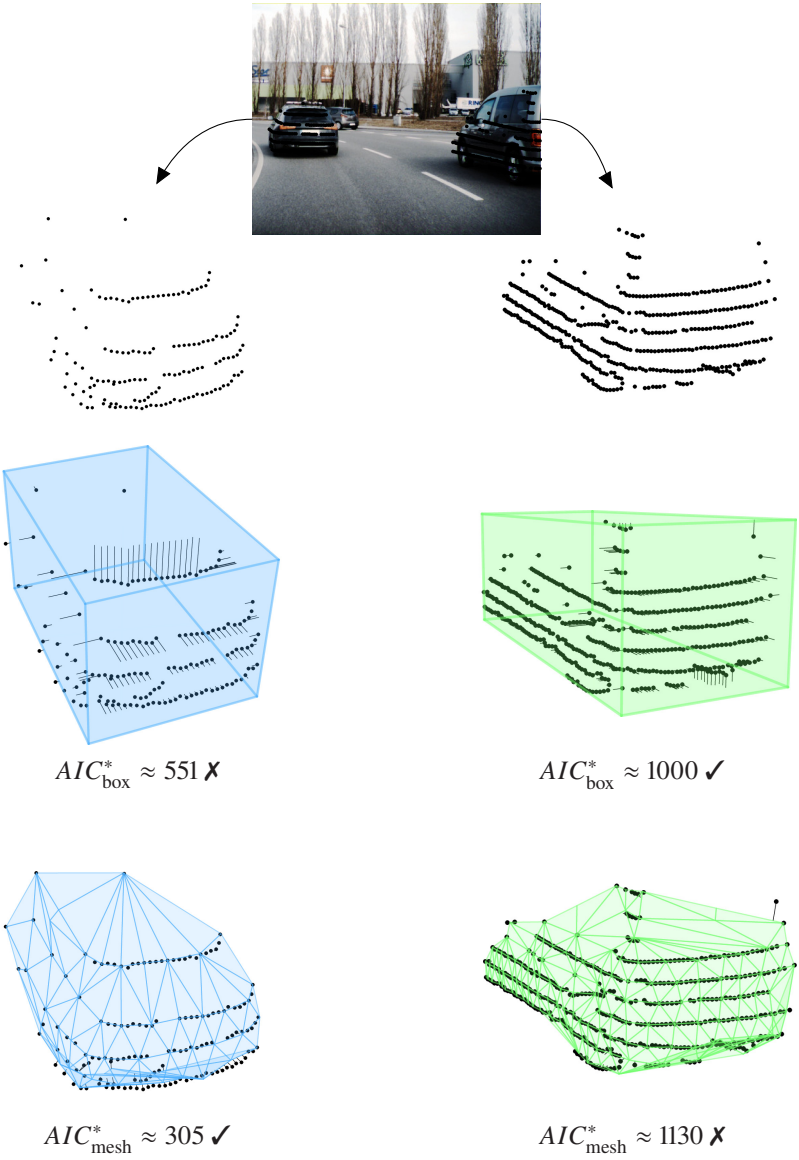


Figure 4.2: AIC-based model selection example. The mesh surface model is selected for the station wagon (left), whereas the box surface model is favored for the van (right).

5 Experimental Results

This chapter presents experimental results of the proposed estimation framework based on three real-world data sets totaling over 1.5 h of object trajectories in public traffic. The data sets are briefly introduced in section 5.1, followed by an analysis of the tracking and surface estimation process in two parts: Section 5.2 evaluates the quality of object pose and motion estimates against ground truth trajectories without special emphasis on the object shape. In contrast, section 5.3 presents a qualitative and quantitative assessment of the surface reconstruction accuracy. Finally, a short conclusion is provided in section 5.4.

In the remainder of this chapter, the sliding window size is fixed to the 10 most recent sensor frames and the surfel map resolution is set to 0.1 m. The estimation runtime per object is approximately 10 ms, 21 ms, 86 ms and 82 ms for the box, polyline, mesh and surfel map surface models respectively[1].

5.1 Data Sets

The three data sets used for evaluation most prominently differ in the sensor model, ranging from a low-resolution automotive series laser scanner to high-end research LiDARs. While the majority of test sequences cover urban scenarios, a fraction of the data was also collected during highway driving.

The *Scala* and *VLP-16HR* data sets comprise sensor measurements with 3 and 16 vertical scan lines respectively, both captured with front-facing LiDARs mounted in the radiator grill of a research vehicle. The data sets focus on single-object tracking with high-accuracy ground truth for a station wagon (Audi A6 Avant) and a liftback with sloping roofline (Audi A7). The ego and

[1] The runtime evaluation is based on the VLP-16HR data set.

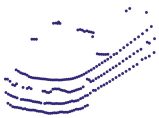
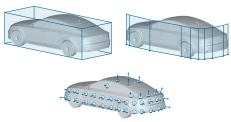
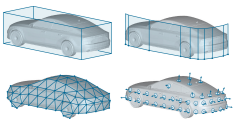
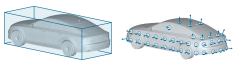
Scala	VLP-16HR	KITTI [28]
		
single object motion & 3d shape	single object motion & 3d shape	multi-object manual box labels
		
57 min 52 sec 2 object types 113 trajectories	15 min 39 sec 2 object types 29 trajectories	23 min 14 sec 252 object types 252 trajectories

Table 5.1: An overview of the data sets, including the available type of ground truth, the evaluated surface models, the total length of ground truth trajectories and the number of captured object types.

target vehicles are equipped with high-class inertial measurement units and differential GPS, providing precise labels for vehicle poses and their motion. Additionally, accurate ground truth for the object shape is available via the original CAD models supplied by the manufacturer. Due to the very limited vertical resolution of the sensor, the triangle mesh surface model is omitted from the evaluation of the Scala data set.

The public *KITTI* [28] data set supplies 64 layers of scan data from a roof-mounted research LiDAR. A rich variety of different object types (252 individual trajectories) enables an evaluation of the multi-object tracking performance. The data set is annotated with 3D bounding boxes generated by human labelers. Since the labeling is limited to the front-facing sensor field-of-view, the scans are cropped accordingly. Noteworthy, these box labels are also used for simulating dynamic classification of individual LiDAR points in the evaluation, i.e., bypassing the detection task performed by a gridmap for the other two data

sets. Due to the lack of precise ground truth for the object shape in the KITTI data set, the evaluation is limited to the most basic and most advanced surface representations, i.e., the box and surfel map models respectively.

A brief overview of the data sets is shown in Table 5.1. For more detailed information (e.g., data set statistics and sensor specifications) the reader is referred to appendix C.

5.2 Pose and Motion Estimation

This section is subdivided into an evaluation of the estimation accuracy of object positions (section 5.2.1) and object motion (section 5.2.2) respectively.

5.2.1 Object Position

The *multiple object tracking precision* (MOTP) [14] is an established evaluation metric for object tracking algorithms. It is defined as the average distance between estimated and ground truth object positions over time, i.e.,

$$\text{MOTP} = \frac{\sum_t \sum_{i=1}^{c_t} d_t^{(i)}}{\sum_t c_t}$$

where c_t denotes the number of ground truth trajectories with a corresponding track at time instant t and $d_t^{(i)}$ represents a measure of the distance between their positions, which will be detailed in the remainder.

In the literature, the distance between objects is commonly defined as difference of object center positions. This metric suffers from a major drawback: Due to the virtually omnipresent self-occlusion of target vehicles in LiDAR-based perception, a center point distance of zero between estimate and ground truth is not feasible until the object has been observed from a wide range of different viewpoints – a condition that rarely arises in real-world scenarios.

For illustration, consider the three-point turn maneuver in Fig. 5.1: The plot on the left-hand side depicts the raw scan points as observed by the stopped ego vehicle over time, whereas the right-hand plot displays the object pose and

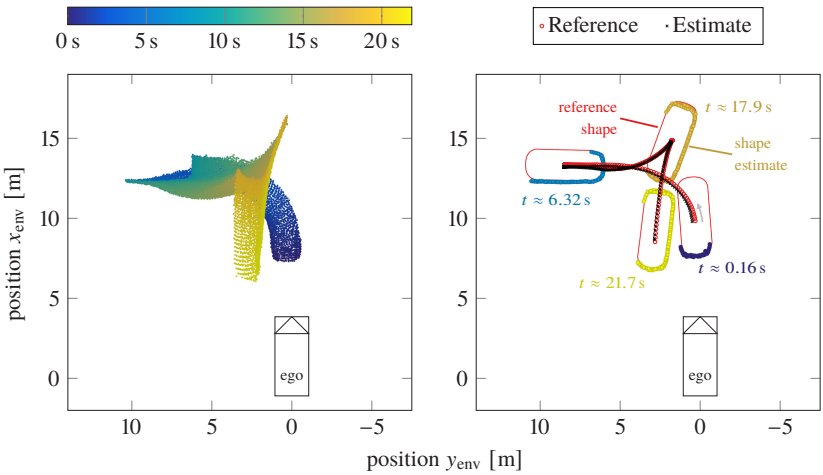
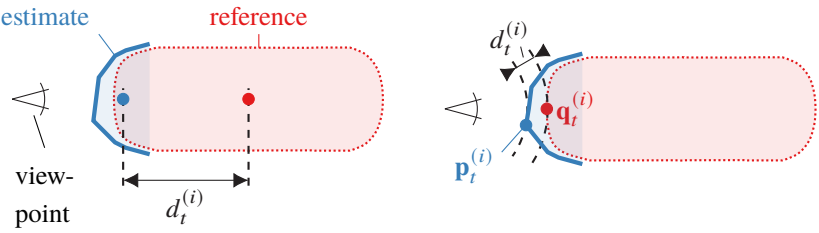


Figure 5.1: Three point turn maneuver of a target vehicle. The left-hand plots depicts the raw scan points as observed by the stopped ego vehicle over time. The right-hand plot displays the state estimates at various stages of the maneuver. The indicated trajectory estimate corresponds to the motion of the object centroid as determined at $t \approx 21.7$ s.

surfel map estimates at various stages of the maneuver. In the initial phase of the maneuver ($t \approx 0.16$ s), the self-occlusion of the tracked vehicle would inflict a positional error larger than 2 m between the surfel map centroid and the ground truth object center. An accurate estimation of the object center becomes feasible only once the vehicle side is observed ($t \approx 6.32$ s). Note that Fig. 5.1 (right) is additionally annotated with the trajectory of the ground truth center as well as the estimated trajectory of the object centroid as determined in the final phase of the maneuver ($t \approx 21.7$ s). Comparing both trajectories, it becomes apparent that the maneuver is tracked with high precision even in the initial phase, where online computation of the center point distance would result in an undesirably large error due to the self-occlusion.

An alternative popular definition of the distance $d_t^{(i)}$ between correspondences is represented by the 2d area overlap or intersection over union of the estimated shape with the ground truth. However, this metric similarly suffers from limited visibility: The track in the initial phase of the three-point turn maneuver ($t \approx$



(a) Center point distance

(b) Range difference of closest visible points

Figure 5.2: Illustration of the difference between an definition of the distance $d_t^{(i)}$ between an object estimate and the corresponding ground truth. Fig. 5.2a displays the large center point error caused by self-occlusion, whereas Fig. 5.2b depicts the range distance between the closest visible estimated and ground truth surface points $\mathbf{p}_t^{(i)}$ and $\mathbf{q}_t^{(i)}$ respectively.

0.16 s) in Fig. 5.1 would again achieve a low overlap ratio due to self-occlusion. Moreover, the overlap metric poses the question of appropriately dealing with freeform models producing *open* surfaces for which the notion of area is not well-defined.

In order to overcome the viewpoint dependency, in the remainder the distance from the ego vehicle to the closest visible surface point is evaluated. In particular, given the closest visible points $\mathbf{p}_t^{(i)}$ and $\mathbf{q}_t^{(i)}$ on the surface estimate and the ground truth CAD model respectively, the distance of the correspondence is defined by their absolute range difference w.r.t. the environment coordinate system origin, i.e.,

$$d_t^{(i)} = \left| \, ||\mathbf{p}_t^{(i)}|| - ||\mathbf{q}_t^{(i)}|| \, \right| .$$

The sum of distances over all correspondences is then denoted as MOTP-r, where the "r" stands for the difference in *range* to the closest surface point. Fig. 5.2 illustrates the difference between the center point distance and MOTP-r. Arguably, a deficiency of the MOTP-r metric is that it remains unaltered when the closest surface point moves on a circle around the ego vehicle. However, practical evaluation shows this limitation to be of rather theoretical nature.

	Scala			VLP-16HR				KITTI	
	Box	Polyline	Surfel	Box	Polyline	Mesh	Surfel	Box	Surfel
MOTP-*r* [m]	0.08	0.05	0.06	0.09	0.04	0.05	0.05	0.28	0.32
IDS	19	10	11	11	8	7	5	111	67
SPLITS	2	4	2	1	4	5	2	56	55
COV [%]	99.1	99.2	99.2	98.7	98.6	98.7	98.8	94.3	95.3
# labels	43,402			11,732				13,940	
# trajectories	113			29				252	

Table 5.2: Tracking performance metrics: The tracking precision (MOTP), number of identity switches (IDS), number of object splits (SPLITS) and coverage (COV) of the ground truth trajectory for the different data sets and surface models. The last two rows provide the number of ground truth labels and trajectories used for the evaluation.

The MOTP-*r* values for the three data sets and the corresponding surface models are provided in Table 5.2. For the Scala and VLP-16HR data sets, the average range error remains below 0.1 m for all surface models. Notably, the modeling of object shape by a freeform model results in an improvement in MOTP-*r* of 0.02 m to 0.05 m over the box model. Since the MOTP-*r* is primarily governed by the vertical cross-section of the object, there is no added benefit in modeling the full 3d surface with a mesh or surfel map over using the simpler polyline model. While the impact of freeform modeling on the MOTP-*r* metric might appear insignificant at first glance, it must be taken into account that the target vehicle's vertical cross-sections violate the box assumption only at the rounded vehicle corners. In the majority of test sequences, however, the closest surface point is located on the vehicle rear, thus averaging out the benefits of precisely modeling the rounded vehicle bumper.

For the KITTI data set, the MOTP-*r* values are almost an order of magnitude larger. While this in part is caused by the more complex traffic scenarios

at hand as well as the more challenging sensor position[2], it is worth noting that the limited ground truth precision feasible by manual annotations also presents an important limiting factor. Interesting, for KITTI the difference in MOTP-r values is almost exactly reversed over the other data sets, with the box model outperforming the surfel map by a margin of 0.04 m. Given the similarity in error differences, this suggests the assumption that the ground truth box annotations are overapproximating the true object geometry, whereas the surfel map estimates indeed reflect the true object shape more precisely than the original ground truth box labels.

In addition to MOTP-r, Table 5.2 lists the total number of identifier switches (IDS [48]) of the tracks associated to the ground truth trajectories. Most notably, the IDS score is approximately twice as high for the box model than for all other surface representations. Analysis shows that this is primarily caused by poor initial box model fits (refer to Fig. 1.1a for an illustration) resulting in object losses during the initialization phase of the tracks. Since freeform models avoid the fitting ambiguity, identifier switches occur considerably less frequently for these surface representations.

Finally, Table 5.2 provides the amount of instances in which more than one track is associated to a single ground truth trajectory (SPLITS) as well as the percentage of ground truth trajectories covered by associated tracks (COV). While the COV metric primarily benchmarks the object detection performance, which is beyond the focus of this thesis, the SPLITS metric shows no noticeable difference across the different surface models. This is not surprising as the association of segments to existing tracks as well as the creation of new object hypotheses is implemented very similarly for all surface models (see section 3.5.2).

[2] The roof mounting enables a richer field-of-view of the sensor, since small and medium-sized objects do not block the line-of-sight. However, the "second row" objects in the background tend to suffer from temporary occlusion by static infrastructure (e.g., trees) more often than objects in the immediate environment, thus rendering their tracking more challenging.

5.2.2 Object Motion

The object motion is evaluated in terms of the root-mean-square error (RMSE) of the velocity and yaw rate estimates extracted from the circular motion model described in section 3.4. Note that this requires an object to be observed for at least three sensor frames. From the high-precision IMU installed in the target vehicle, accurate motion ground truth is available for the Scala and VLP-16HR data sets. Since the KITTI data set only provides manual box annotations, the reference motion is determined by locally fitting constant velocity and yaw rate motion profiles into the box positions and orientations. As a consequence, the accuracy of these signals is limited and the results must be treated with appropriate caution.

Table 5.3 shows the RMSE of the velocity estimates for the different data sets and surface models. The modeling of the object shape with the polyline surface representation reduces the error by approximately 20% over the baseline box model. Capturing the varying vertical cross-section of objects with a triangle mesh further increases the improvement to 25%. Finally, tracking a surfel map outperforms the box model tracker's velocity estimate by a total of approximately 30%.

The RMSE of the yaw rate estimate is displayed in Table 5.4. While modeling the rounded vehicle corners using a polyline results in an improvement of approximately 25% over the box model, the mesh representation surprisingly achieves a slightly lower gain of 20%. For the Scala and VLP-16HR data sets, the surfel map model lowers the RMSE by approximately 30% over the baseline, while the KITTI data set even achieves an improvement of 40%. This is assumed to be due to the elevated sensor mounting, which allows to observe a larger fraction of the vehicle geometry.

For detailed histograms of the velocity and yaw rate errors, the reader is referred to Figs. D.1, D.2 and D.3 in appendix D.

	Box	Polyline	Mesh	Surfel
Scala (> 42k samples)	0.59	0.46 (−22 %)	–	0.43 (−27 %)
VLP-16HR (> 11k samples)	0.80	0.66 (−18 %)	0.60 (−25 %)	0.59 (−27 %)
KITTI (> 13k samples)	5.07	–	–	3.57 (−30 %)

Table 5.3: Root-mean square errors of the *velocity* estimates in km/h, annotated with the relative improvement over the baseline box surface model.

	Box	Polyline	Mesh	Surfel
Scala (> 42k samples)	3.55	2.71 (−24 %)	– –	2.50 (−30 %)
VLP-16HR (> 11k samples)	3.16	2.37 (−26 %)	2.54 (−20 %)	2.28 (−28 %)
KITTI (> 13k samples)	13.70	–	–	8.20 (−40 %)

Table 5.4: Root-mean square errors of the *yaw rate* estimates in °/s, annotated with the relative improvement over the baseline box surface model.

5.3 Surface Estimation

The surface estimation quality is evaluated in three separate parts: Section 5.3.1 provides *qualitative* results of the reconstruction process for the three data sets and varying object classes. Using precise ground truth available from the manufacturer's CAD models, section 5.3.2 *quantitatively* assesses the surface estimates of two specific vehicle types. Finally, section 5.3.3 discusses limitations of the reconstruction process that emerge in practical application.

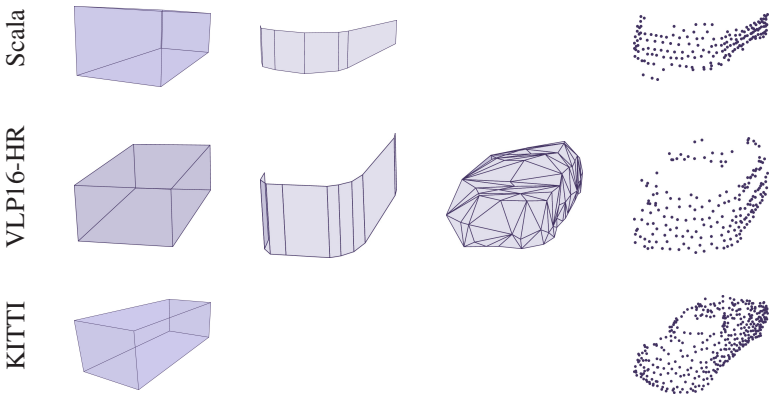


Figure 5.3: Sample reconstructions of oncoming cars using the different surface models. The data is taken from the Scala (top), VLP-16HR (middle) and KITTI (bottom) data sets respectively. Note that the specific car models differ between the data sets.

5.3.1 Qualitative Evaluation

Fig. 5.3 displays the surface estimates of an oncoming car using observations from the three different data sets. While there are no distinctive variations in the reconstructed box and polyline surface models across the data sets, the generated surfel maps vividly illustrate the difference in sensor data. In particular, the surfel representation from the Scala data set is restricted to the lower vehicle outline, whereas the research sensors allow for an accurate recovery of the car's hood geometry. Moreover, the elevated roof-mounting of the KITTI sensor enables a sampling of the target vehicle's roof surface, which largely remains occluded from the viewpoint of the radiator grill, in which the other two sensors are integrated.

For an oncoming car, the evolution of the four surface models over time is depicted in Fig. 5.4. The four models are reconstructed incrementally from the VLP16-HR data set while simultaneously estimating the object motion. The bottom row of the figure additionally provides the accumulated input observations after compensation for ground truth ego and target motion. Notably, the box and polyline surface models are already close to convergence from the three vertical cross-sections of the vehicle surface (first column). In contrast, the mesh and surfel models evolve from a mere representation of the lower ve-

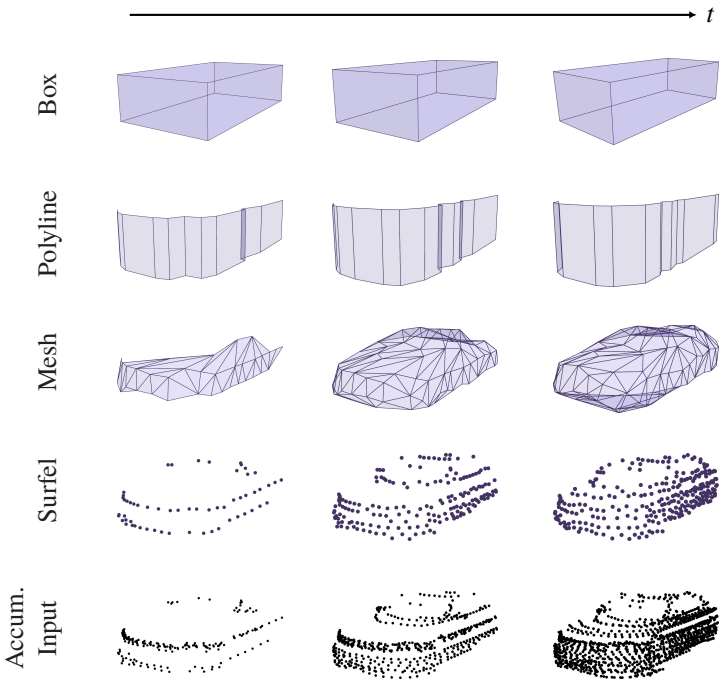


Figure 5.4: Evolution of the incrementally reconstructed object shape of an oncoming car in the VLP-16HR data set. The bottom rows shows the accumulated input observations after compensation for ground truth ego and target motion.

hicle outline and A-pillar (first column) to a rich and detailed 3d representation (last column) as more observations become available over time. A comparison of the final surfel map estimate (last column) with the accumulated input also illustrates the sparsifying effect of surfel fusion. Finally, note the polyline model's sharp kinks on level with the A-pillar that originate from samples of the wheel cases.

The freeform reconstructions for a variety of different object classes from the VLP-16HR data set are provided in Fig. 5.5. The first column displays an oncoming truck with drivers cab and cargo box, whereas the second column displays a minicar (Toyota Aygo). Notably, the bus in the third column features a large array of side windows that do not produce LiDAR echos and are

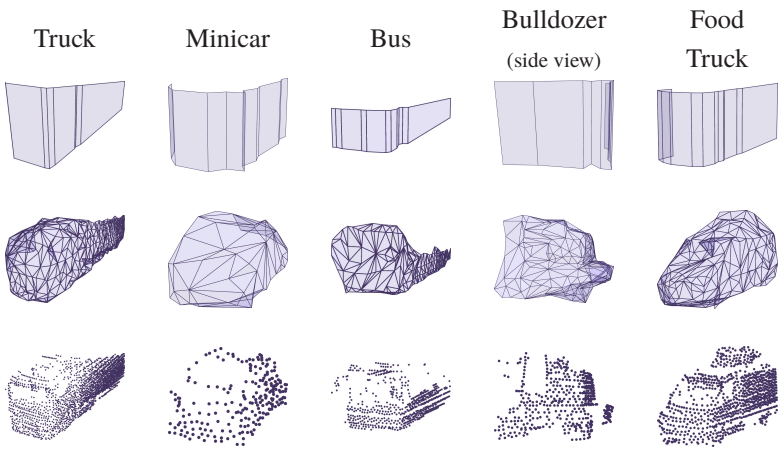


Figure 5.5: A collection of various exemplary freeform reconstructions from the VLP-16HR data set.

therefore absent from the mesh and surfel reconstructions. Finally, the last two columns display the side view of a bulldozer with overhanging, rear-mounted bucket as well as a food truck with half a chicken mounted on its roof.

5.3.2 Quantitative Evaluation

The quantitative evaluation of the surface estimation compares the ground truth CAD models of a station wagon (Audi A6 Avant) and a liftback with sloping roofline (Audi A7) against the respective reconstructions of their surface. In this context, the error ϵ_i of any reconstructed surface point $\mathbf{p}_i$ is defined as its absolute distance to the closest point $\mathbf{q}_i$ on the corresponding ground truth geometry, i.e.,

$$\epsilon_i = \|\mathbf{p}_i - \mathbf{q}_i\|.$$

Besides providing a heatmap of the reconstruction error for the four different surface models, the average (AVG) and maximum (MAX) error over the entire surface estimate are examined in the following.

In order to eliminate the influence of temporary inaccuracies in the instantaneous object pose estimate at the time of evaluation, the reconstructed surfaces are registered against the ground truth models prior to error computation[3]. This additionally allows to compensate for potential errors introduced by the lack of a vertical object position estimation. For the surfel maps, the reconstruction error is computed at every surfel center position, whereas the error of the parametric surface representations (i.e., box, polyline and triangle mesh) is densely sampled at a resolution of 0.01 m.

Fig. 5.6 depicts the error heatmap for the surface estimate of an oncoming Audi A7. Note that the range of the colormaps differs for the top and bottom row of the figure. It becomes apparent that surface models with constant vertical cross-section considerably overapproximate the vehicle's hood geometry. While the maximum reconstruction error for the upper front corners of the box amounts to almost 0.7 m, modeling the rounded vehicle bumper using the polyline representation allows to reduce the maximum error to approximately 0.5 m already. Reconstructing the full 3d geometry of the vehicle using a triangle mesh or surfel map further reduces the maximum error to approximately 0.2 m. Interestingly, the maximum error for these models originates from scan points of the object interior, namely the front passenger seat's headrest, which will be discussed in more detail in section 5.3.3. Cropping the interior points from the reconstruction allows to diminish the maximum error below 0.1 m.

The average reconstruction error is an approximation for the average absolute distance between estimate and ground truth over the complete reconstructed surface. While this error does not exceed the value of 0.03 m for the triangle mesh and surfel map, it is nearly an order of magnitude larger for the box and polyline models. Similar results are found when evaluating the reconstruction error of the rear views of a Audi A7 and A6 Avant for which the detailed results are supplied in Figs. D.4 and D.5 in appendix D. The primary difference for the rear views is that the reconstructions of the vehicle's lateral surface do not entirely stretch up to the vehicle's front corner. Since the upper front corner represents the error hotspot for the box and polyline models, their maximum surface estimation error is reduced to approximately 0.4 m for the rear perspective.

[3] Note that this is a fine-registration for which the planar xy-translation and the vertical z-translation remain below 0.1 m and 0.18 m respectively for all evaluated models.

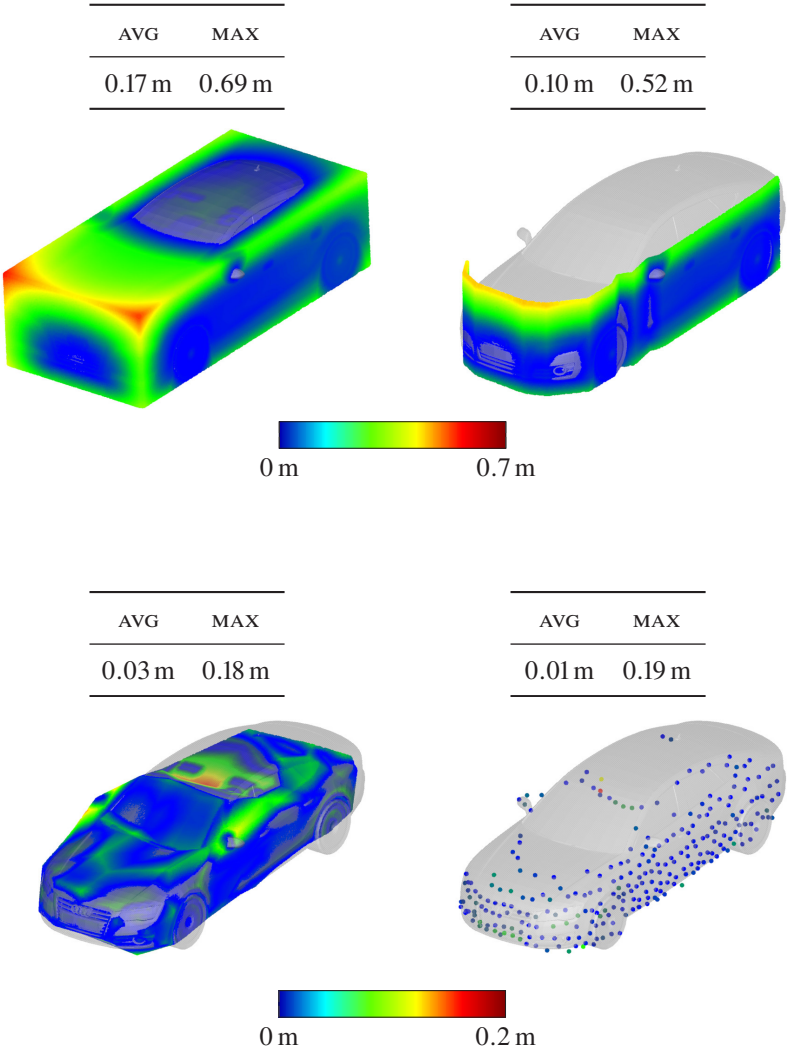


Figure 5.6: Heatmaps of the reconstruction error for an oncoming Audi A7 from the VLP-16HR data set. The estimates correspond to the snapshots in the right column of Fig. 5.4. Note that, for the mesh and surfel surfaces, the maximum values are caused by outlier points sampled from the object interior. Cropping these points from the reconstruction reduces the maximum error below 0.1 m.

5.3.3 Limitations

There are a couple of limitations of the reconstruction process emerging in practical applications. While the box model generally is extremely robust in recovering from erroneous surface state estimates, the continuous freeform models (i.e., polyline and mesh model) occasionally get trapped in local minima of the surface state. This includes the handling of object interior points, accurately capturing thin structures as well as coping with irregular surfaces. The main challenge in these situations is establishing the surface topology and the approximate vertex positions. These deficiencies can most likely be resolved by implementing more advanced reconstruction methods, which are beyond the scope of this thesis.

Interior points

Interior object points arise when LiDAR beams are able to traverse transparent elements of the object surface, e.g., windshields or car windows. These observations cause two different types of problems: The presence of interior points in the *mesh initialization* phase might result in triangles that intrude far into the object interior, since the mesh topology (i.e., vertex connectivity) is directly gathered from the grid structure of the scan. Fig. 5.7 illustrates the surface estimate of an oncoming van, where interior points captured through the windshield lead to a triangle mesh with considerable intrusion. Arguably, the mesh with intrusions accurately represents the object surface as sampled by the sensor, however their removal might be desirable in order to reduce the complexity of the mesh.

In contrast, interior points in the incremental *mesh growing* phase of the reconstruction offer the potential for true misinterpretation due to the lack of a priori knowledge about vertex connectivity. As the growing of the mesh topology is established by pure spatial reasoning of scan point positions, interior scan points (e.g., from the headrests or roof liner) might erroneously be interpreted as observations of the exterior (e.g., vehicle roof) and are then used to grow the mesh. This effect is displayed in Fig. 5.8, where samples from the headrests and roof lining of an oncoming car are used to grow the reconstruction of the vehicle roof. The headrests are situated several centimeters below the true vehicle roof, whereas the roof liner is almost on level with the roof geome-

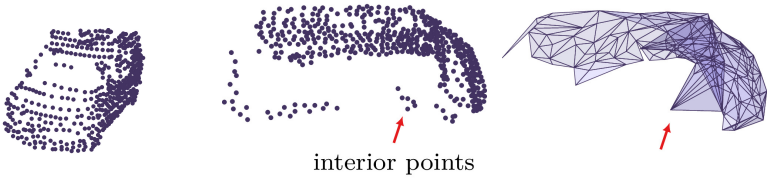


Figure 5.7: Surfel map (left, middle) and triangle mesh (right) reconstruction of an oncoming van. Scan points from the object interior are observed through the windshield and lead to intruding triangles *during* the mesh initialization process.

Figure 5.8: Surfel map (left) and triangle mesh (right) reconstruction of an oncoming car. Scan points from the object interior are observed through the windshield *after* the initialization phase and are used to *grow* the mesh, leading to the indicated dent on the roof.

try. As a consequence, the reconstruction suffers from a considerable dent. Note that the triangle mesh in Fig. 5.8 corresponds to the error heatmap in Fig. 5.6, where indeed the erroneous roof surface reconstruction results in an error hotspot.

Thin structures

Thin structures, such as pillars or overhanging freight, prompt a considerable challenge for the construction of triangle mesh representations. Fig. 5.9 depicts the surface estimates of a leading trailer that is open to the top and carries a carousel. While the surfel map accurately captures the pillars holding the carousel's roof, the triangulation algorithm closes the holes between the pillars and generates a closed surface estimate that encircles the full carousel diameter.

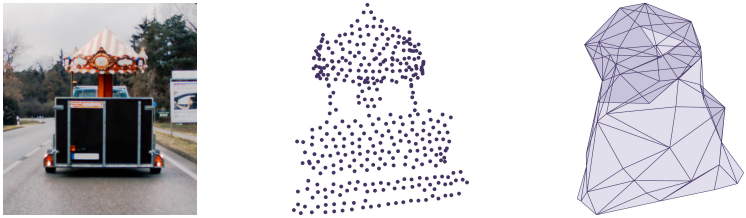


Figure 5.9: Surfel map (middle) and triangle mesh (right) reconstruction of leading trailer carrying a carousel. While the surfel map accurately represents the thin pillars holding the roof, the triangulation produces closed mesh parts that encircles the full carousel diameter.

Figure 5.10: Reconstructions of an oncoming bulldozer with highly non-planar surface. The surfel map (middle) is able to represent the object surface reasonably well, whereas the triangulation produces a ragged mesh (right).

Irregular surfaces

While the majority of street vehicles feature reasonable regular and smooth surfaces, this presumption is violated by a variety of more exceptional vehicle types encountered in public traffic. For example, Fig. 5.10 displays the surfel and triangle mesh reconstructions of an oncoming bulldozer which is composed from several separate building blocks (e.g., elevated driver's cab and arms for holding the shovel), which result in a highly non-planar vehicle surface. While the surfel representation captures the object geometry reasonably well, establishing the mesh topology for this complex surface is extremely challenging and the triangulation process leads to a ragged mesh.

5.4 Conclusion

This chapter presented experimental results of the simultaneous tracking and shape estimation framework. It was shown that the proposed methods are able to produce high-precision object trajectories as well as accurate surface reconstructions. A comparison of four different surfel models showed that freeform representations of the object surface enable a reduction in motion estimation error of up to 30-40% over the traditional box model. Moreover, freeform surface estimation allows to diminish the maximum reconstruction error by approximately up to 0.40 m even for common car geometries. Establishing surface topology remains one of the greatest challenges in the reconstruction process, particularly in the presence of interior points, thin object structures as well as irregular surfaces, thus leaving scope for future research.

6 Conclusion and Future Work

6.1 Conclusion

This thesis presented a probabilistic framework for the estimation of rigid object surface and motion from LiDAR measurements. Given the strong mutual dependency of object position and shape, both states are estimated simultaneously using sliding window bundle adjustment.

A special emphasis is placed on the modelling of object shape by investigation of four different surface models with varying level of detail: In addition to the traditional bounding box, which serves as baseline model, the object surface is approximated using polylines, triangle meshes as well as surfel maps. Besides diminishing tracking residuals caused by an overapproximation of object geometry, precise surface estimates also offer potential benefits for driving functions, e.g., when operating in confined spaces or in collision avoidance applications.

Sensor evidence from LiDAR is incorporated into the estimation process using three complementary observation models. Firstly, a common point measurement model based on likelihood fields leverages scan point endpoints in the estimation process. Secondly, evidence of freespace adjacent to objects is integrated into the reasoning about probable state configurations using a novel freespace observation model. Thirdly, a novel measurement model is proposed, which utilizes LiDAR intensity in order to identify salient, highly-reflective features on the vehicle contour and track them over time.

Considering the diversity of object geometries present in public traffic, a one-surface-model-fits-them-all approach seems inappropriate. Therefore, the possibility of an adaptive surface model switching was discussed. In particular, a prototypical selection scheme based on observability and model adequacy

criteria factors in sensor capabilities and balances model detail against approximative power based on the Akaike information criterion.

An extensive evaluation of the proposed framework on more than 1.5 hours of object trajectories recorded in public traffic shows that the methods are able to produce precise object pose and motion estimates as well as accurate surface reconstructions. Moreover, the scalability of the approach to various sensor models and target vehicle types is demonstrated by the use of three different data sets, ranging from a low-resolution automotive-grade laserscanner to high-end research LiDARs.

The evaluation illustrates the benefits of detailed surface modelling by a direct comparison of the four surface models. In particular, freeform modeling of the object shape allows to reduce the error in motion estimation by a margin of up to 30%-40% over the baseline box model. In terms of reconstruction quality, the triangle mesh and surfel map are able to diminish the average reconstruction error of the box model by a factor of 4 to 14 even for standard cars, depending on the specific vehicle model and point-of-view.

6.2 Future Work

Despite these promising results, there remain a couple of open topics for future research. First of all, while the box model tends to be extremely robust in recovering from erroneous surface estimates, the *topology generation of freeform models* proves to be challenging, especially so in the presence of interior points, thin surface elements as well as irregular surface structure. Therefore, more advanced meshing methods are required for a robust application of the estimation framework in automated driving.

In addition, the *explicit modeling of holes* in the object surface, e.g., at the windshield or other vehicle openings, represents an interesting subject for future investigation. While the reconstruction methods discussed in the previous chapters attempt to produce hole-free representations, the transitions between surface and holes, e.g., window frames, present salient features that might facilitate the tracking process.

In order to accurately capture the object motion in a more diverse set of situations, an estimation of the *vertical object motion*, i.e., the z-component

of the object pose, is desirable. However, the limited vertical resolution of in-the-market LiDARs renders the tracking of vertical motion possible only in the near-field, where in the majority of applications the local road surface is reasonably flat anyway. Important exceptions are ramps, e.g., in parking lots, where the use of LiDAR intensity patterns on the vehicle contour might additionally promote the capture of vertical object motion.

Moreover, the adaptive selection of an appropriate surface model presented in this thesis requires further investigation. Besides additionally factoring in contextual and functional requirements into the selection process, the topology generation of freeform surfaces should preferably be governed by the information criterion in order to make the comparison between the models more fair. Ideally, an efficient selection framework provides a small set of pre-defined surface models for the most common vehicle types, e.g., car, bus and truck, and constructs the more involved freeform models only if none of the pre-defined models appropriately fits the object at hand.

Finally, bundle adjustment provides an elegant solution for handling asynchronous observations in a multi-sensor setup with varying latencies. While fully recursive methods, like Kalman filters, typically require special care for the processing of out-of-order observations, these can be inserted into the appropriate position of the pose graph in bundle adjustment, with a reasonable initial guess of the state being directly available from the neighboring poses.

A Basic Laws Of Probability

The following basic laws of probability are used throughout this thesis. They can be found in any standard textbook such as Thrun et al. [75].

- *Bayes rule* allows to express the conditional probability of the random variable X given Y as

$$p(x \mid y) = \frac{p(y \mid x) \cdot p(x)}{p(y)}. \tag{A.1}$$

- The joint probability distribution of two *independent* random variables X and Y can be expressed as

$$p(x, y) = p(x) \cdot p(y). \tag{A.2}$$

- Using the *chain rule*, the joint probability distribution of two random variables X and Y can be expressed as

$$p(x, y) = p(x \mid y) \cdot p(y). \tag{A.3}$$

  The recursive application of the chain rule to a set of N random variables $\{X_1, \ldots, X_N\}$ results in

$$p(x_{1:N}) = \prod_{i=1}^{N} p(x_i \mid x_{1:i-1}) \cdot p(x_{1:i-1}). \tag{A.4}$$

- Under the *Markov assumption*, the probability distribution of a random variable X_{t+1} conditioned on a time sequence of t previous random variables $X_{1:t}$ only depends on the most recent state x_t, i.e.,

$$p(x_{t+1} \mid x_{1:t}) = p(x_{t+1} \mid x_t). \tag{A.5}$$

- The log-likelihood of a Gaussian probability density $p(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma), \mathbf{x} \in \mathbb{R}^k$ can be expressed as

$$\begin{aligned}
\ln p(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) &= \ln \left[\left((2\pi)^k \det(\Sigma) \right)^{-1/2} \exp\left(-\frac{1}{2} \|\mathbf{x} - \boldsymbol{\mu}\|^2_{\Sigma^{-1}} \right) \right] \\
&= \ln \left[\left((2\pi)^k \det(\Sigma) \right)^{-1/2} \right] + \ln \left[\exp\left(-\frac{1}{2} \|\mathbf{x} - \boldsymbol{\mu}\|^2_{\Sigma^{-1}} \right) \right] \\
&= \text{const.} - \frac{1}{2} \|\mathbf{x} - \boldsymbol{\mu}\|^2_{\Sigma^{-1}}.
\end{aligned} \tag{A.6}$$

- Given the maximum likelihood estimate $\hat{\boldsymbol{\theta}}$ of a least squares model fit with M observations $\mathbf{z} = (\mathbf{z}_1, \ldots, \mathbf{z}_M)$ and normally distributed residuals with zero mean

$$d_i(\hat{\boldsymbol{\theta}}, \mathbf{z}) \sim \mathcal{N}(0; \hat{\sigma}^2) \quad \text{for } i = 1, \ldots, M$$

the *maximum* log-*likelihood value* is expressible as [5]

$$\ln \mathcal{L}(\hat{\boldsymbol{\theta}} \mid \mathbf{z}) = \ln \left[\prod_{i=1}^{M} \frac{1}{\sqrt{2\pi\hat{\sigma}^2}} \exp\left(-\frac{d_i^2}{2\hat{\sigma}^2} \right) \right].$$

Applying basic laws of logarithms allows to separate the product into the sum

$$\begin{aligned}
\ln \mathcal{L}(\hat{\boldsymbol{\theta}} \mid \mathbf{z}) &= M \cdot \ln\left[\frac{1}{\sqrt{2\pi\hat{\sigma}^2}}\right] + \ln\left[\prod_{i=1}^{M} \exp\left(-\frac{d_i^2}{2\hat{\sigma}^2}\right)\right] \\
&= -\frac{M}{2} \cdot \ln\left[2\pi\hat{\sigma}^2\right] - \sum_{i=1}^{M} \frac{d_i^2}{2\hat{\sigma}^2} \\
&= -\frac{M}{2} \cdot \ln\left[2\pi\right] - \frac{M}{2} \ln\left[\hat{\sigma}^2\right] - \frac{1}{2\hat{\sigma}^2} \sum_{i=1}^{M} d_i^2 .
\end{aligned}$$

Substituting the variance $\hat{\sigma}^2 = \frac{1}{M}\sum_{i=1}^{M} d_i^2$ of the residuals into the equation results in

$$\begin{aligned}
\ln \mathcal{L}(\hat{\boldsymbol{\theta}} \mid \mathbf{z}) &= -\frac{M}{2} \cdot \ln\left[2\pi\right] - \frac{M}{2} \ln\left[\frac{1}{M}\sum_{i=1}^{M} d_i^2\right] - \frac{M}{2} \\
&= -\frac{M}{2} \cdot \ln\left[2\pi\right] - \frac{M}{2} \ln\left[\sum_{i=1}^{M} d_i^2\right] + \frac{M}{2} \ln\left[M\right] - \frac{M}{2} \\
&= -\frac{M}{2} \ln\left[\sum_{i=1}^{M} d_i^2\right] + C(M) \qquad \text{(A.7)}
\end{aligned}$$

where the constant offset $C(M)$ is only dependent on the number of observations M.

B Tracking and Shape Estimation As Bundle Adjustment

Maximum a posteriori (MAP) estimation is concerned with recovering the mode of a posterior probability. In the simultaneous tracking and shape estimation problem, the posterior probability is expressible as (see chapter 3)

$$p(\mathbf{x}_{0:T}, \mathbf{s} \mid \mathbf{z}_{1:T}) = \eta \prod_{t=1}^{T} \underbrace{p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s})}_{\text{measurement model}} \prod_{t=0}^{T} \underbrace{p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1})}_{\text{motion model}} \underbrace{p(\mathbf{s})}_{\text{surface prior}} \tag{3.2 revisited}$$

where $\mathbf{x}_{0:T}$: sequence of time-dependent object poses up to time T

$\mathbf{s}$: time-invariant (i.e., rigid) object surface

$\mathbf{z}_{1:T}$: sequence of sensor observations up to time T.

The monotonicity of the log-function renders the maximum of Eqn. (3.2) equivalent to the minimum of its negative log-likelihood, which can be determined by solving the optimization problem

$$\min_{\mathbf{x}_{0:T}, \mathbf{s}} \underbrace{-\ln \left[\eta \cdot \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) \cdot \prod_{t=0}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) \cdot p(\mathbf{s}) \right]}_{:= L \quad \text{(loss function)}} . \tag{B.1}$$

Considering basic logarithmic identities, the loss function L can be further simplified as

$$\begin{aligned} L &= -\ln\left[\eta \cdot \prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) \cdot \prod_{t=0}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) \cdot p(\mathbf{s})\right] \\ &= -\ln\eta - \ln\prod_{t=1}^{T} p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) - \ln\prod_{t=0}^{T} p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) - \ln p(\mathbf{s}) \\ &= \text{const.} - \sum_{t=1}^{T} \ln p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) - \sum_{t=0}^{T} \ln p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) - \ln p(\mathbf{s}). \end{aligned}$$

Moreover, under the assumption that

- all M_t observations $\mathbf{z}_t^{(i)}$ are conditionally independent, i.e.,

$$p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) = \prod_{i=1}^{M_t} p(\mathbf{z}_t^{(i)} \mid \mathbf{x}_t, \mathbf{s})$$

  and that the measurement function $h_i(\mathbf{x}_t, \mathbf{s})$ models the expected value of the i-th observation $\mathbf{z}_t^{(i)} \in \mathbb{R}^l$. Under the assumption of a Gaussian measurement formation with covariance $\Sigma_{h_i}^{-1}$, the probability density function is expressible as

$$p(\mathbf{z}_t \mid \mathbf{x}_t, \mathbf{s}) = \prod_{i=1}^{M_t} \left((2\pi)^l \det(\Sigma_{h_i})\right)^{-1/2} \exp\left(-\frac{1}{2}\|\mathbf{z}_t^{(i)} - h_i(\mathbf{x}_t, \mathbf{s})\|^2_{\Sigma_{\mathrm{h_i}}^{-1}}\right).$$

- the motion model function $f(\mathbf{x}_{0:t-1})$ describes the expected object pose at time instant t. Under the assumption of Gaussian state evolution with covariance Σ_f, the motion model becomes

$$p(\mathbf{x}_t \mid \mathbf{x}_{0:t-1}) = \left((2\pi)^3 \det(\Sigma_f)\right)^{-1/2} \exp\left(-\frac{1}{2}\|\mathbf{x}_t - f(\mathbf{x}_{0:t-1})\|^2_{\Sigma_{\mathrm{f}}^{-1}}\right).$$

- that the Gaussian potentials from sections 3.2.2 and 3.2.3 represent the surface priors between neighboring surface normals $\mathbf{n}_i \in \mathbb{R}^k$ and $\mathbf{n_j} \in \mathbb{R}^k$ with covariance $\Sigma_\mathbf{n}$, i.e.,

$$p(\mathbf{s}) = \prod_{(\mathbf{n}_i, \mathbf{n}_j)} \left((2\pi)^k \det(\Sigma_\mathbf{n}) \right)^{-1/2} \exp\left(-\frac{1}{2} \|\mathbf{n}_i - \mathbf{n}_j\|^2_{\Sigma_\mathbf{n}^{-1}} \right).$$

Together with the identity (A.6) for the log-likelihood of Gaussians, these definitions allow to rewrite the loss function L as

$$\begin{aligned} L = \text{const.} &+ \frac{1}{2} \sum_{t=1}^{T} \sum_{i=1}^{M_t} \|\mathbf{z}_t^{(i)} - h_i(\mathbf{x}_t, \mathbf{s})\|^2_{\Sigma_{h_i}^{-1}} \\ &+ \frac{1}{2} \sum_{t=1}^{T} \|\mathbf{x}_t - f(\mathbf{x}_{0:t-1})\|^2_{\Sigma_f^{-1}} + \frac{1}{2} \sum_{(\mathbf{n}_i, \mathbf{n}_j)} \|\mathbf{n}_i - \mathbf{n}_j\|^2_{\Sigma_\mathbf{n}^{-1}}. \end{aligned}$$

Noting that the minimum of L neither is affected by the constant offset, nor by the common prefactor $1/2$, renders the optimization problem in Eqn. (B.1) equivalent to the weighted non-linear least squares problem

$$\begin{aligned} \min_{\mathbf{x}_{0:T}, \mathbf{s}} \quad & \underbrace{\sum_{t=1}^{T} \sum_{i=1}^{M_t} \|\mathbf{z}_t^{(i)} - h_i(\mathbf{x}_t, \mathbf{s})\|^2_{\Sigma_{\mathrm{h_i}}^{-1}}}_{\text{measurement constraints}} + \underbrace{\sum_{t=1}^{T} \|\mathbf{x}_t - f(\mathbf{x}_{0:t-1})\|^2_{\Sigma_{\mathrm{f}}^{-1}}}_{\text{motion constraints}} \\ & + \underbrace{\sum_{(\mathbf{n}_i, \mathbf{n}_j)} \|\mathbf{n}_i - \mathbf{n}_j\|^2_{\Sigma_\mathbf{n}^{-1}}}_{\text{surface prior constraints}}. \end{aligned}$$

The solution of this optimization problem simultaneously maximizes the likelihood of measurements, motion model and the surface prior.

C Data Sets

Hardware Setup

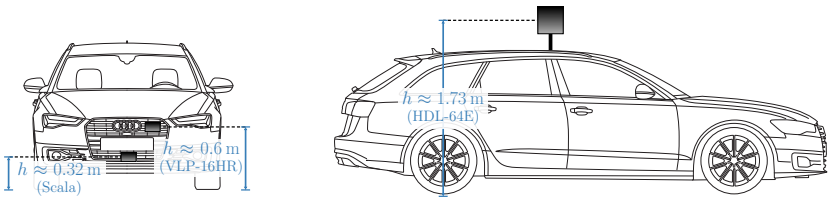
The three data sets used for the experimental evaluation are captured with three different LiDAR sensors, whose most relevant specifications are given in Table C.1. While the KITTI data set [28] provides observations from an elevated position on the vehicle roof, the two other sensors are mounted in the radiator grill of a research vehicle.

Data distribution

For the data sets collected with the front-facing Scala and VLP-16HR sensors, both ego and target vehicles are equipped with differential GPS and high-class inertial measurement units that enable high-precision estimates of object poses and motion. The estimated standard deviation of the setup's global

	Valeo Scala	Velodyne VLP-16HR	Velodyne HDL-64E (KITTI [28])
Layers	3	16	64
Frequency	12.5 Hz	12.5 Hz	10 Hz
Field-of-view (H × V)	145° × 2.4°	180° × 20°	180° × 26.9°
Resolution (H × V)	0.25° × 0.8°	0.25° × 1.33°	0.16° × 0.4°

Table C.1: An excerpt of the most relevant data set specifications. Note that the original sensor specifications might differ as, e.g., for the Velodyne sensors only the subset of front-facing sensor readings is processed.



(a) Research vehicle (b) KITTI data set [28]

Figure C.1: An illustration of the mounting positions of the different sensors. Most notably, the sensor in the KITTI data set is roof-mounted, whereas the other sensors are integrated into the radiator grill of a research vehicle.

position estimate is below 5 cm. In addition, for the two vehicle types under consideration (Audi A7 and Audi A6 Avant) the original CAD models from the manufacturer are available as highly accurate ground truth for the object geometry.

In contrast, the KITTI Raw data set[1] provides the raw sensor observations annotated with 3d cuboid object annotations that were manually created by human labelers. Since labels are only available for the environment in front of the vehicle, the LiDAR data is cropped to the front-facing half scan. Moreover, only the object class VEHICLE with a total trajectory length of at least 5 m are evaluated.

Given the limited horizontal field-of-views of the sensors, the ground truth trajectories are stripped from leading and trailing labels that do not contain any LiDAR observations. In total the data sets comprise 394 individual ground truth trajectories covering a total of 86.3 km or 1 h 36 min 45 s of public driving with 69,074 ground truth labels. There is an overall of 254 different vehicles models covered in the data sets[2].

[1] The City sequence 2011_09_26_DRIVE_0093_SYNC is excluded from the evaluation due to apparent errors in the provided ego motion data.

[2] For the KITTI data set, each trajectory is counted as an individual vehicle model, even though occasionally trajectories from different sequences might correspond to exactly the same vehicle model.

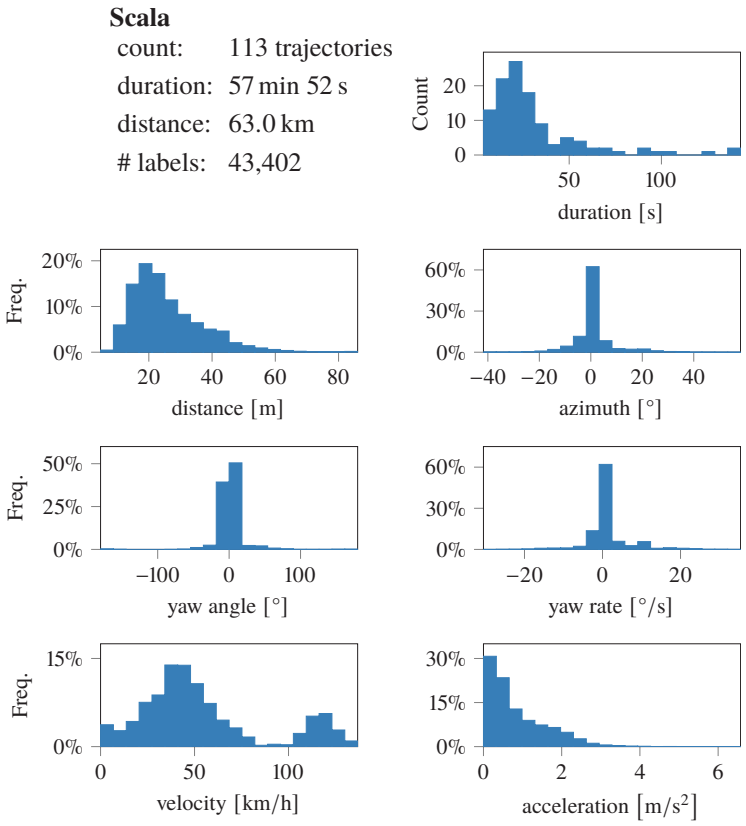


Figure C.2: Statistics for the Scala data set. The object yaw angle is defined w.r.t. the ego vehicle's environment coordinate system, whereas the velocity and acceleration are absolute over ground.

More details on the distributions of the individual trajectory durations, relative target positions (distance and azimuth), yaw angle and rate as well as velocity are provided for the different data sets in Figs. C.2, C.3 and C.4.

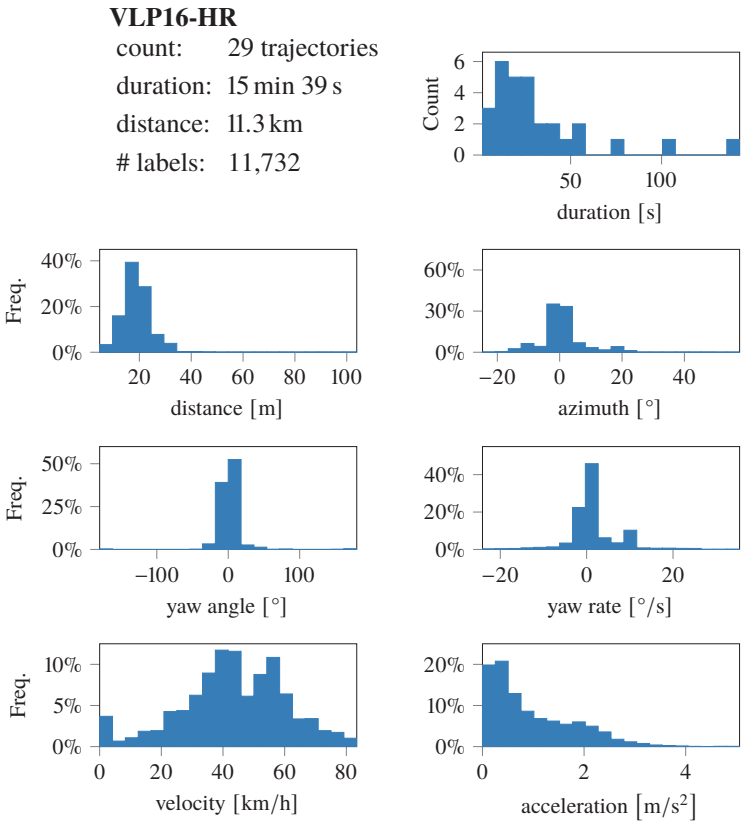


Figure C.3: Statistics for the VLP-16HR data set. The object yaw angle is defined w.r.t. the ego vehicle's environment coordinate system, whereas the velocity and acceleration are absolute over ground.

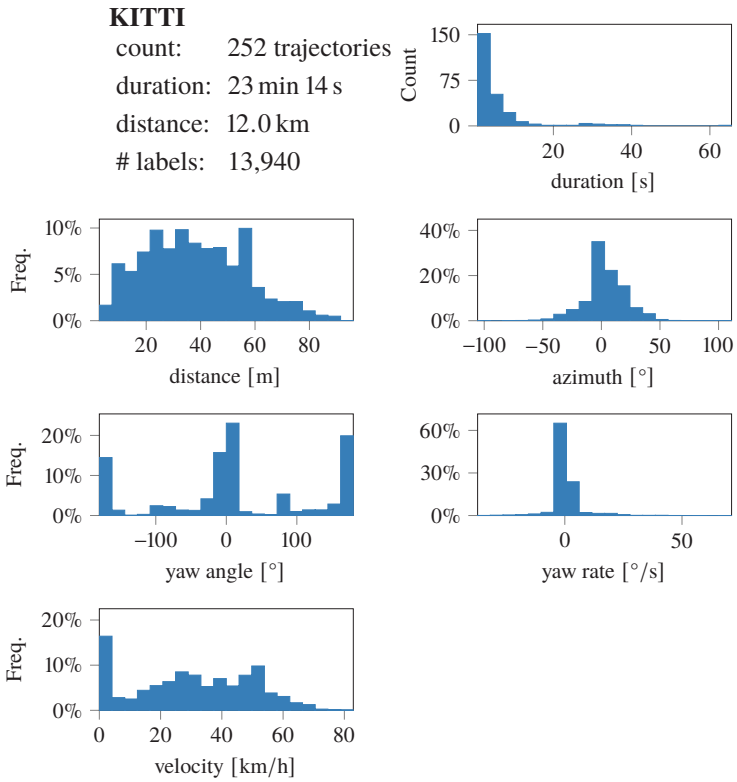


Figure C.4: Statistics for the KITTI data set. The object yaw angle is defined w.r.t. the ego vehicle's environment coordinate system, whereas the velocity and acceleration are absolute over ground. Note that the object velocities were artificially computed from the manual box labels of the data set.

D Supplementary Results

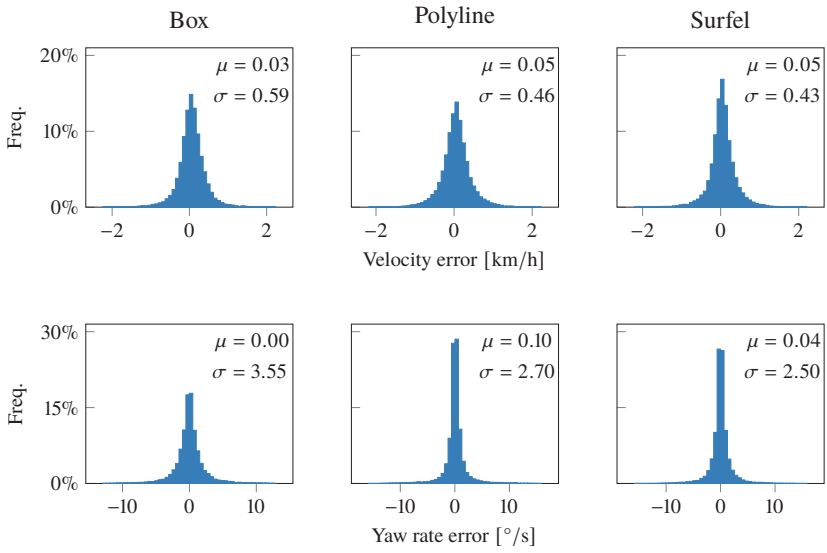


Figure D.1: Histograms of the velocity and yaw rate error for the Scala data set. Note that the mean (μ) and standard deviation (σ) of the errors correspond to the original error distributions, whereas the largest 1% of errors are removed from the histograms for better illustration.

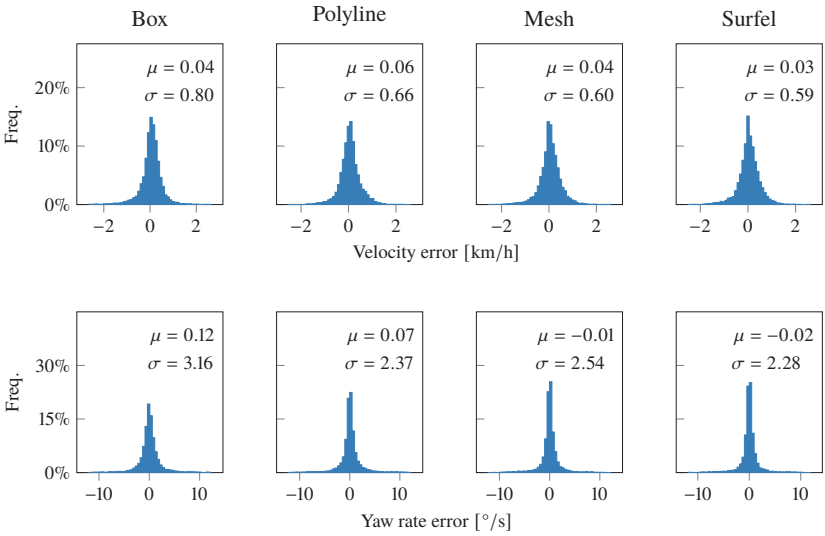


Figure D.2: Histograms of the velocity and yaw rate error for the VLP-16HR data set. Note that the mean (μ) and standard deviation (σ) of the errors correspond to the original error distributions, whereas the largest 1% of errors are removed from the histograms for better illustration.

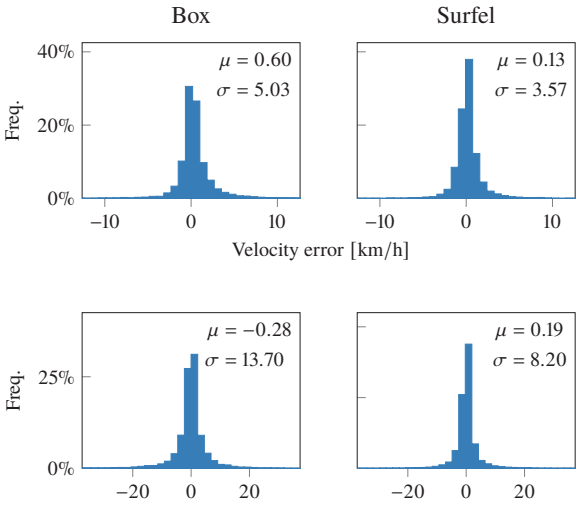


Figure D.3: Histograms of the velocity and yaw rate error for the KITTI data set. Note that the mean (μ) and standard deviation (σ) of the errors correspond to the original error distributions, whereas the largest 1% of errors are removed from the histograms for better illustration. Note that the data set lacks ground truth for the object motion, thus the values used for evaluation were artificially computed from the manual box labels.

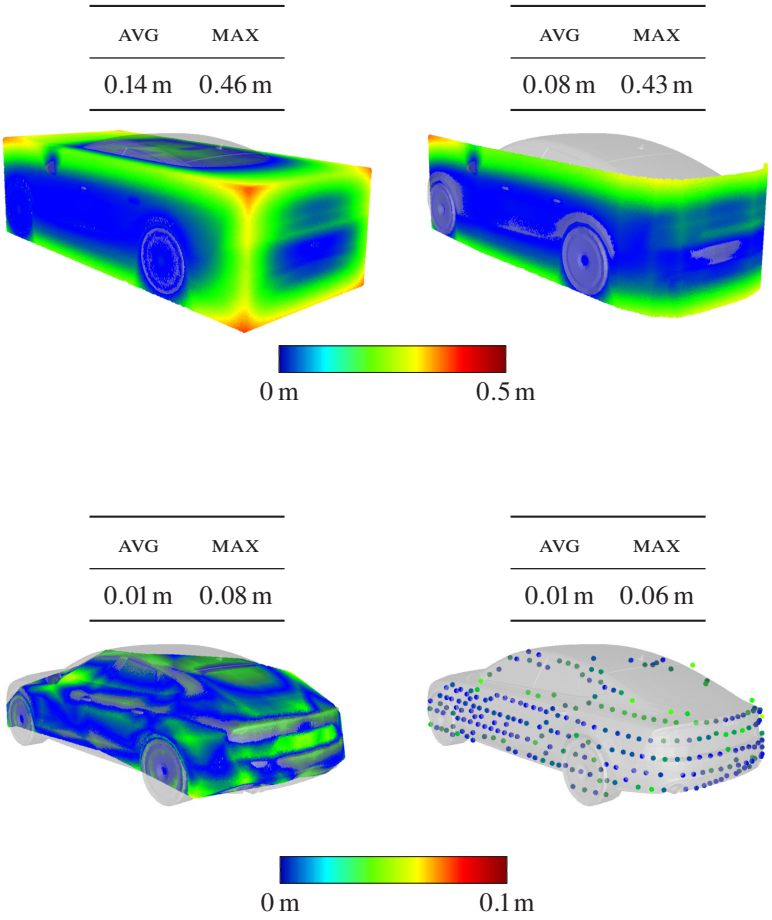


Figure D.4: Heatmaps of the reconstruction error for a leading Audi A7 from the VLP-16HR data set.

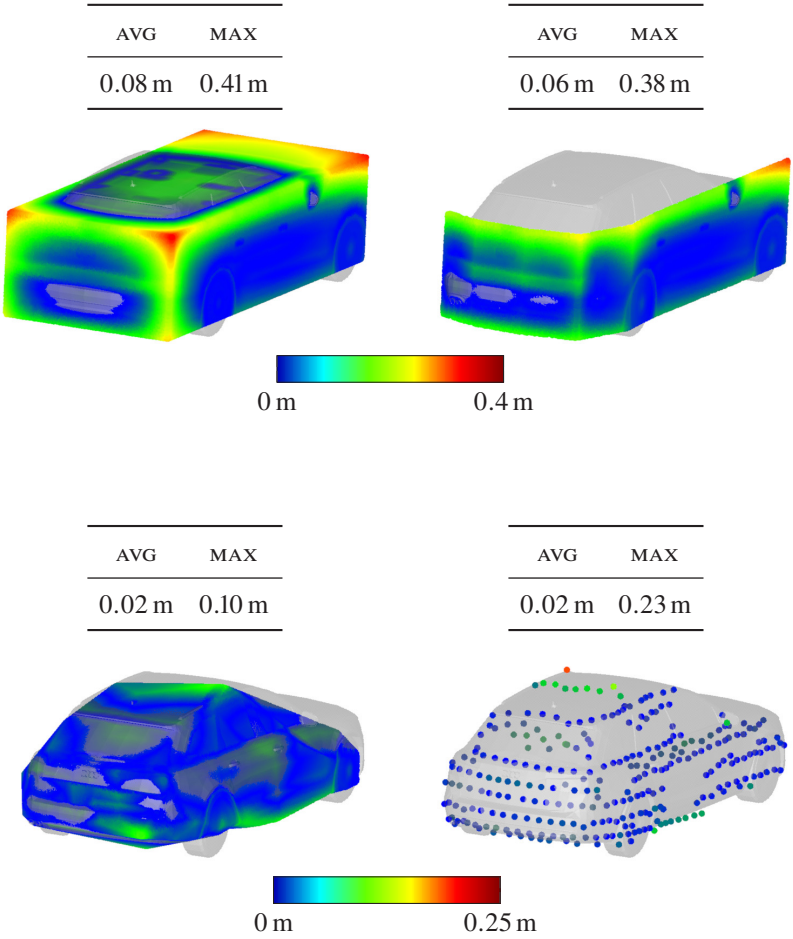


Figure D.5: Heatmaps of the reconstruction error for a leading Audi A6 Avant from the VLP-16HR data set. Note that the target vehicle is equipped with a roof-mounted antenna rack, which causes the largest errors in the surfel map. Cropping the corresponding surfels reduces the MAX error to 0.07 m.

References

[1] M. Aeberhard. “Object-Level Fusion for Surround Environment Perception in Automated Driving Applications”. Doctoral dissertation. Sammlung: Lehrstuhl für Regelungssystemtechnik, Technische Universität Dortmund, Germany, 2017. DOI: 10.17877/DE290R-18029.

[2] S. Agarwal, K. Mierle, et al. *Ceres Solver*. `http://ceres-solver.org`. Last retrieved 2019-09-19.

[3] C. S. Agate, R. M. Wilkerson, and K. J. Sullivan. “Utilizing negative information to track ground vehicles through move-stop-move cycles”. In: *Signal Processing, Sensor Fusion, and Target Recognition XIII*. Ed. by Ivan Kadar. Vol. 5429. International Society for Optics and Photonics. Orlando, Florida, USA: SPIE, 2004, pp. 273–283.

[4] M. Althoff, M. Koschi, and S. Manzinger. “CommonRoad: Composable Benchmarks for Motion Planning on Roads”. In: *2017 IEEE Intelligent Vehicles Symposium (IV 2017)*. Redondo Beach, CA, USA, June 2017, pp. 719–726.

[5] D. Anderson and K. Burnham. *Model selection and multi-model inference*. 2nd ed. New York, USA: Springer-Verlag, 2002. ISBN: 978-0-387-95364-9.

[6] A. Asvadi, P. Peixoto, and U. Nunes. “Detection and Tracking of Moving Objects Using 2.5D Motion Grids”. In: *18th IEEE International Conference on Intelligent Transportation Systems (ITSC 2015)*. Las Palmas de Gran Canaria, Spain, Sept. 2015, pp. 788–793.

[7] J. Aue, M. R. Schmid, T. Graf, and J. Effertz. “Improved object tracking from detailed shape estimation using object local grid maps with stereo”. In: *16th IEEE International Conference on Intelligent Transportation Systems (ITSC 2013)*. The Hague, Netherlands, Oct. 2013, pp. 330–335.

[8] A. Azim and O. Aycard. “Layer-based supervised classification of moving objects in outdoor dynamic environment using 3D laser scanner”. In: *2014 IEEE Intelligent Vehicles Symposium (IV 2014)*. Michigan, USA, June 2014, pp. 1408–1414.

[9] H. Badino, U. Franke, and D. Pfeiffer. “The Stixel World - A Compact Medium Level Representation of the 3D-World”. In: *Pattern Recognition. DAGM 2009. Lecture Notes in Computer Science*. Ed. by Süße H. Denzler J. Notni G. Berlin, Heidelberg: Springer, 2009, pp. 51–60.

[10] M. Baum and U. D. Hanebeck. “Shape tracking of extended objects and group targets with star-convex RHMs”. In: *14th International Conference on Information Fusion (FUSION 2011)*. Chicago, Illinois, USA, July 2011, pp. 1–8.

[11] M. Baum, B. Noack, and U. D. Hanebeck. “Extended object and group tracking with elliptic random hypersurface models”. In: *13th International Conference on Information Fusion (FUSION 2010)*. IEEE. Edinburgh, UK, 2010, pp. 1–8.

[12] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller, and H. Winner. “Three decades of driver assistance systems: Review and future perspectives”. In: *IEEE Intelligent Transportation Systems Magazine* 6.4 (2014), pp. 6–22.

[13] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. 3rd ed. Santa Clara, CA, USA: Springer-Verlag TELOS, 2008. ISBN: 9783540779735.

[14] K. Bernardin and R. Stiefelhagen. “Evaluating multiple object tracking performance: the CLEAR MOT metrics”. In: *EURASIP Journal on Image and Video Processing* 246309 (2008), pp. 1–10.

[15] G.A. Borges and M.-J. Aldon. “Line Extraction in 2D Range Images for Mobile Robotics”. In: *Journal of Intelligent and Robotic Systems* 40.3 (July 2004), pp. 267–297.

[16] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Lévy. *Polygon mesh processing*. Natick, MA, USA: AK Peters/CRC Press, 2010. ISBN: 9781568814261.

[17] M. E. Bouzouraa and U. Hofmann. “Fusion of occupancy grid mapping and model based object tracking for driver assistance systems using laser and radar sensors”. In: *2010 IEEE Intelligent Vehicles Symposium (IV 2010)*. San Diego, CA, USA, June 2010, pp. 294–300.

[18] A. Broggi, S. Cattani, M. Patander, M. Sabbatelli, and P. Zani. “A full-3D voxel-based dynamic obstacle detection for urban scenario using stereo vision”. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. The Hague, Netherlands, Oct. 2013, pp. 71–76.

[19] P. Broßeit, M. Rapp, N. Appenrodt, and J. Dickmann. “Probabilistic rectangular-shape estimation for extended object tracking”. In: *2016 IEEE Intelligent Vehicles Symposium (IV 2016)*. Gothenburg, Sweden, June 2016, pp. 279–285.

[20] M. Carlberg, J. Andrews, P. Gao, and A. Zakhor. *Fast surface reconstruction and segmentation with ground-based and airborne lidar range data*. Tech. rep. Berkeley, USA: UC Berkeley, 2008.

[21] J. Choi, S. Ulbrich, B. Lichte, and M. Maurer. “Multi-Target Tracking using a 3D-Lidar sensor for autonomous vehicles”. In: *16th International IEEE Conference on Intelligent Transportation Systems (ITSC 2013)*. The Hague, Netherlands, Oct. 2013, pp. 881–886.

[22] M. Darms, P. Rybski, and C. Urmson. “An Adaptive Model Switching Approach for a Multisensor Tracking System used for Autonomous Driving in an Urban Environment. The Approach used by Tartan Racing in the Urban Grand Challenge”. In: *Steuerung und Regelung von Fahrzeugen und Motoren: AUTOREG 2008*. Baden-Baden, Germany, Feb. 2008, pp. 521–530.

[23] M. Darms, P. Rybski, and C. Urmson. “Classification and tracking of dynamic objects with multiple sensors for autonomous driving in urban environments”. In: *2008 IEEE Intelligent Vehicles Symposium (IV 2008)*. Eindhoven, The Netherlands, June 2008, pp. 1197–1202.

[24] J. R. Diebel, S. Thrun, and M. Brünig. “A Bayesian Method for Probable Surface Reconstruction and Decimation”. In: *ACM Transactions on Graphics* 25.1 (2006), pp. 39–59.

[25] D.H. Douglas and T.K. Peucker. “Algorithms for the reduction of the number of points”. In: *The Canadian Cartographer*. Ed. by M. Dodge. Vol. 10. Toronto, Canada: Canadian Cartographic Association, 1973, pp. 112–122.

[26] F. Ebert and H. Wuensche. “Dynamic Object Tracking and 3D Surface Estimation using Gaussian Processes and Extended Kalman Filter”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC 2019)*. Auckland, New Zealand, Oct. 2019, pp. 1122–1127.

[27] S. Gargoum, K. El-Basyouny, J. Sabbagh, and K. Froese. “Automated Highway Sign Extraction Using Lidar Data”. In: *Transportation Research Record* 2643.1 (2017), pp. 1–8.

[28] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. “Vision meets robotics: The KITTI dataset”. In: *The International Journal of Robotics Research* 32.11 (2013), pp. 1231–1237.

[29] H. González-Jorge, J.M. Sánchez, L. Díaz-Vilariño, I. Puente, and P. Arias. “Automatic Registration of Mobile LiDAR Data Using High-Reflectivity Traffic Signs”. In: *Journal of Construction Engineering and Management* 142.8 (2016), pp. 04016022-1–5.

[30] M. Gopi and S. Krishnan. “A Fast and Efficient Projection-Based Approach for Surface Reconstruction”. In: *XV Brazilian Symposium on Computer Graphics and Image Processing*. Fortaleza, Brazil, 2002, pp. 179–186.

[31] K. Granström, M. Baum, and S. Reuter. “Extended Object Tracking: Introduction, Overview and Applications”. In: *Journal of Advances in Information Fusion (JAIF)* 12.2 (Dec. 2017), pp. 139–174.

[32] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. “A Tutorial on Graph-Based SLAM”. In: *IEEE Intelligent Transportation Systems Magazine* 2.4 (2010), pp. 31–43.

[33] S. Hefford, C. Samson, J.W. Harrison, F.P. Ferrie, K. Kusevic, P. Mrstik, and P. Iles. “Augmenting the iterative closest point (ICP) alignment algorithm with intensity”. In: *Geomatica* 63 (Jan. 2009), p. 407.

[34] D. Held, J. Levinson, S. Thrun, and S. Savarese. “Robust real-time tracking combining 3D shape, color, and motion”. In: *The International Journal of Robotics Research* 35.1–3 (2016), pp. 30–49.

[35] M. Himmelsbach, A. Mueller, T. Luettel, and H.-J. Wuensche. “LIDAR-based 3D Object Perception”. In: *Proceedings of 1st International Workshop on Cognition for Technical Systems*. Munich, Germany, Oct. 2008, pp. 1–7.

[36] P. J. Huber. “Robust Estimation of a Location Parameter”. In: *The Annals of Mathematical Statistics* 35.1 (Mar. 1964), pp. 73–101.

[37] Velodyne Lidar Inc. *VLP-32C User Manual*. Version Rev. B. 2018.

[38] ISO Central Secretary. *Road vehicles – Vehicle dynamics and road-holding ability – Vocabulary*. Standard ISO 8855:2011(E). Geneva, CH: International Organization for Standardization, 2011.

[39] A. E. Johnson and S. B. Kang. “Registration and integration of textured 3D data”. In: *Image and Vision Computing* 17.2 (1999), pp. 135–147.

[40] S. Kammel and B. Pitzer. “Lidar-based lane marker detection and mapping”. In: *2008 IEEE Intelligent Vehicles Symposium (IV 2008)*. Eindhoven, The Netherlands, 2008, pp. 1137–1142.

[41] N. Kämpchen. “Feature-level fusion of laser scanner and video data for advanced driver assistance systems”. Doctoral dissertation. Ulm, Germany: Universität Ulm, 2007. DOI: 10.18725/OPARU-382.

[42] M. Keller, D. Lefloch, M. Lambers, S. Izadi, T. Weyrich, and A. Kolb. “Real-time 3d reconstruction in dynamic scenes using point-based fusion”. In: *2013 International Conference on 3D Vision (3DV 2013)*. Tokyo, Japan, June 2013, pp. 1–8.

[43] B. Kim, B. Choi, M. Yoo, H. Kim, and E. Kim. “Robust Object Segmentation Using a Multi-layer Laser Scanner”. In: *Sensors* 14.11 (2014), pp. 20400–20418.

[44] D. Koller. “Moving Object Recognition and Classification based on Recursive Shape Parameter Estimation”. In: *12th Israeli Conference on Artificial Intelligence, Computer Vision, and Neural Networks*. Tel-Aviv, Israel, 1993, pp. 359–368.

[45] T. Kühner and J. Kümmerle. “Large-Scale Volumetric Scene Reconstruction using LiDAR”. In: *IEEE International Conference on Robotics and Automation (ICRA 2020)*. Paris, France, 2020, pp. 6261–6267.

[46] M. Kumru and E. Özkan. “3D Extended Object Tracking Using Recursive Gaussian Processes”. In: *2018 21st International Conference on Information Fusion (FUSION 2018)*. Cambridge, UK, July 2018, pp. 1–8.

[47] M. Kusenbach, M. Himmelsbach, and H.-J. Wünsche. “A new geometric 3D LiDAR feature for model creation and classification of moving objects”. In: *2016 IEEE Intelligent Vehicles Symposium (IV 2016)*. Gothenburg, Sweden, 2016, pp. 272–278.

[48] Y. Li, C. Huang, and R. Nevatia. “Learning to associate: Hybridboosted multi-target tracker for crowded scene”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2009)*. Miami, Florida, 2009, pp. 2953–2960.

[49] P. Lindner, E. Richter, G. Wanielik, K. Takagi, and A. Isogai. “Multi-channel lidar processing for lane detection and estimation”. In: *12th IEEE International Conference on Intelligent Transportation Systems (ITSC 2009)*. Las Vegas, NV, USA, Oct. 2009, pp. 1–6.

[50] Z. C. Marton, R. B. Rusu, and M. Beetz. “On Fast Surface Reconstruction Methods for Large and Noisy Point Clouds”. In: *2009 IEEE International Conference on Robotics and Automation (ICRA 2009)*. Kobe, Japan, 2009, pp. 2829–2834.

[51] I. Miller, M. Campbell, D. Huttenlocher, A. Nathan, F.-R. Kline, P. Moran, N. Zych, B. Schimpf, S. Lupashin, E. Garcia, J. Catlin, M. Kurdziel, and H Fujishima. “Team Cornell’s Skynet: Robust Perception and Planning in an Urban Environment”. In: *The DARPA Urban Challenge: Autonomous Vehicles in City Traffic*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 257–304.

[52] F. Moosmann and C. Stiller. “Joint self-localization and tracking of generic objects in 3D range data”. In: *2013 IEEE International Conference on Robotics and Automation (ICRA 2013)*. Karlsruhe, Germany, May 2013, pp. 1146–1152.

[53] H. Moravec and A. Elfes. “High resolution maps from wide angle sonar”. In: *1985 IEEE International Conference on Robotics and Automation (ICRA 1985)*. Vol. 2. St. Louis, MO, USA, 1985, pp. 116–121.

[54] B. Naujoks, P. Burger, and H.-J. Wuensche. “Fast 3D Extended Target Tracking using NURBS Surfaces”. In: *2019 IEEE Intelligent Transportation Systems Conference (ITSC 2019)*. Auckland, New Zealand, 2019, pp. 1104–1109.

[55] J. Nocedal and S. Wright. *Numerical Optimization*. Springer Series in Operations Research and Financial Engineering. New York, USA: Springer New York, 2006. ISBN: 9780387303031.

[56] G. Pandey, J. R. McBride, S. Savarese, and R. M. Eustice. “Automatic Extrinsic Calibration of Vision and Lidar by Maximizing Mutual Information”. In: *Journal of Field Robotics* 32.5 (2015), pp. 696–722.

[57] A. Petrovskaya and S. Thrun. “Model based vehicle detection and tracking for autonomous urban driving”. In: *Autonomous Robots* 26.2 (Apr. 2009), pp. 123–139.

[58] D. Pfeiffer and U. Franke. “Efficient representation of traffic scenes by means of dynamic stixels”. In: *2010 IEEE Intelligent Vehicles Symposium (IV 2010)*. San Diego, CA, USA, June 2010, pp. 217–224.

[59] J. Quehl, S. Yan, S. Wirges, J.-H. Pauls, and M. Lauer. “Estimating Object Shape and Movement Using Local Occupancy Grid Maps”. In: *IFAC-PapersOnLine* 52.8 (2019), pp. 87–92.

[60] B. Riveiro, L. Díaz-Vilariño, B. Conde-Carnero, M. Soilán, and P. Arias. “Automatic Segmentation and Shape-Based Classification of Retro-Reflective Traffic Signs from Mobile LiDAR Data”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 9.1 (Jan. 2016), pp. 295–303.

[61] A. Romanoni. “Incremental Large-Scale Visual 3D Mesh Reconstruction”. PhD thesis. Milan, Italy: Polytechnic University of Milan, 2016.

[62] S. Rusinkiewicz and M. Levoy. “Efficient variants of the ICP algorithm”. In: *3rd International Conference on 3-D Digital Imaging and Modeling*. Quebec, Canada, May 2001, pp. 145–152.

[63] M. Rutishauser, M. Stricker, and M. Trobina. “Merging Range Images of Arbitrarily Shaped Objects”. In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 1994)*. Seattle, WA, USA, 1994, pp. 573–580.

[64] S. Sasidharan and B. Lohani. “Intensity Augmented ICP for Registration of Laser Scanner Point Clouds”. In: *XXXII INCA International Congress on Cartography for Sustainable Earth Resource Management*. Sept. 2013, pp. 30–34.

[65] S. Schneider, M. Himmelsbach, T. Luettel, and H.-J. Wuensche. “Fusing vision and LIDAR - Synchronization, correction and occlusion reasoning”. In: *2010 IEEE Intelligent Vehicles Symposium (IV 2010)*. San Diego, CA, USA, June 2010, pp. 388–393.

[66] R. Schubert, E. Richter, and G. Wanielik. “Comparison and evaluation of advanced motion models for vehicle tracking”. In: *11th International Conference on Information Fusion (FUSION 2008)*. Cologne, Germany, June 2008, pp. 1–6.

[67] K. Schueler, T. Weiherer, E. Bouzouraa, and U. Hofmann. “360 Degree Multi Sensor Fusion for Static and Dynamic Obstacles”. In: *2012 IEEE Intelligent Vehicles Symposium (IV 2012)*. Alcala de Henares, Spain, June 2012, pp. 692–697.

[68] C. Schutz, T. Jost, and H. Hugli. “Multi-feature matching algorithm for free-form 3D surface registration”. In: *14th International Conference on Pattern Recognition (ICPR 1998)*. Brisbane, Australia, Aug. 1998, pp. 982–984.

[69] M. Schütz, N. Appenrodt, J. Dickmann, and K. Dietmayer. “Occupancy grid map-based extended object tracking”. In: *2014 IEEE Intelligent Vehicles Symposium (IV 2014)*. Michigan, USA, June 2014, pp. 1205–1210.

[70] A. Al-Sharadqah and N. Chernov. “Error analysis for circle fitting algorithms”. In: *Electronic Journal of Statistics* 3 (2009), pp. 886–911.

[71] P. Steinemann, J. Klappstein, J. Dickmann, F. von Hundelshausen, and H.-J. Wuensche. “Geometric-Model-Free Tracking of Extended Targets Using 3D-LIDAR-Measurements”. In: *Proceedings of SPIE Defense, Security + Sensing*. Baltimore, MD, USA, Apr. 2012, pp. 104–115.

[72] R. Szeliski. *Computer Vision: Algorithms and Applications*. 1st ed. Berlin, Heidelberg, Germany: Springer-Verlag, 2011. ISBN: 9781848829343.

[73] R. Szeliski and D. Tonnesen. “Surface modeling with oriented particle systems”. In: *ACM SIGGRAPH* 26.2 (1992), pp. 185–194.

[74] Ö. Ş. Taş, F. Kuhnt, J. M. Zöllner, and C. Stiller. “Functional system architectures towards fully automated driving”. In: *2016 IEEE Intelligent Vehicles Symposium (IV 2016)*. Gothenburg, Sweden, 2016, pp. 304–309.

[75] S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Cambridge, MA, USA: MIT Press, 2015. ISBN: 9780262201629.

[76] R. Unnikrishnan and M. Hebert. *Fast Extrinsic Calibration of a Laser Rangefinder to a Camera*. Tech. rep. CMU-RI-TR-05-09. Pittsburgh, PA, USA: Carnegie Mellon University, July 2005.

[77] A. Vatavu, R. Danescu, and S. Nedevschi. “Stereovision-Based Multiple Object Tracking in Traffic Scenarios Using Free-Form Obstacle Delimiters and Particle Filters”. In: *IEEE Transactions on Intelligent Transportation Systems* 16.1 (Feb. 2015), pp. 498–511.

[78] A. von Reyher, A. Joos, and H. Winner. “A lidar-based approach for near range lane detection”. In: *2005 IEEE Intelligent Vehicles Symposium (IV 2005)*. Las Vegas, NV, USA, June 2005, pp. 147–152.

[79] N. Wahlström and E. Özkan. “Extended Target Tracking Using Gaussian Processes”. In: *IEEE Transactions on Signal Processing* 63.16 (Aug. 2015), pp. 4165–4178.

[80] H. Winner, S. Hakuli, F. Lotz, and C. Singer. *Handbook of Driver Assistance Systems: Basic Information, Components and Systems for Active Safety and Comfort*. 1st ed. Basel, Switzerland: Springer International Publishing, 2015. ISBN: 9783319123516.

[81] K. Wyffels and M. Campbell. “Joint tracking and non-parametric shape estimation of arbitrary extended objects”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA 2015)*. Seattle, WA, USA, May 2015, pp. 3360–3367.

[82] K. Wyffels and M. Campbell. “Negative Information for Occlusion Reasoning in Dynamic Extended Multiobject Tracking”. In: *IEEE Transactions on Robotics* 31.2 (Apr. 2015), pp. 425–442.

[83] K. Wyffels and M. Campbell. “Precision Tracking via Joint Detailed Shape Estimation of Arbitrary Extended Objects”. In: *IEEE Transactions on Robotics* 33.99 (2016), pp. 1–20.

[84] K. Wyffels and M. Campbell. “Priority-Based Tracking of Extended Objects”. In: *Journal of Advances in Information Fusion (JAIF)* 12.2 (Dec. 2017), pp. 211–227.

[85] S. Zeng. “An Object-Tracking Algorithm for 3-D Range Data Using Motion and Surface Estimation”. In: *IEEE Transactions on Intelligent Transportation Systems* 14.3 (Sept. 2013), pp. 1109–1118.

[86] Q. Zhang and R. Pless. “Extrinsic calibration of a camera and laser range finder”. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*. Sendai, Japan, Jan. 2004, pp. 2301–2306.

[87] X. Zhang, W. Xu, C. Dong, and J.M. Dolan. “Efficient L-Shape Fitting for Vehicle Detection Using Laser Scanners”. In: *2017 IEEE Intelligent Vehicles Symposium (IV 2017)*. Redondo Beach, CA, USA, June 2017, pp. 54–59.

Publications

[88] S. Kraemer, M. E. Bouzouraa, and C. Stiller. “Simultaneous Tracking and Shape Estimation Using a Multi-Layer Laserscanner”. In: *20th IEEE International Conference on Intelligent Transportation Systems (ITSC 2017)*. Yokohama, Japan, Oct. 2017, pp. 1–7.

[89] S. Kraemer, M. E. Bouzouraa, and C. Stiller. “Utilizing LiDAR Intensity in Object Tracking”. In: *2019 IEEE Intelligent Vehicles Symposium (IV 2019)*. Paris, France, June 2019, pp. 1543–1548.

[90] S. Kraemer, C. Stiller, and M. E. Bouzouraa. “LiDAR-Based Object Tracking and Shape Estimation Using Polylines and Free-Space Information”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2018)*. Madrid, Spain, Oct. 2018, pp. 4515–4522.

Supervised Theses

[91] Tobias Drung. “Nutzung von LiDAR-Intensitätsinformationen in der Umfeldwahrnehmung”. MA thesis. Hochschule Kempten, Feb. 2019.

[92] Stefan Lischer. “Objekttracking mittels Kombination von Laserscanner- und (Mono-)Kameradaten”. MA thesis. Universität Duisburg-Essen, May 2018.

[93] Fang Yuan. “Laser-based Detection and Tracking of Multiple Vehicles with a 2D Polyline Shape Model”. MA thesis. Technische Universität München, Aug. 2017.